IBM DB2 10.1
for Linux, UNIX, and Windows

数据库监视指南和参考

S151-1759-00

IBM DB2 10.1
for Linux, UNIX, and Windows

数据库监视指南和参考

S151-1759-00

注意

使用此信息及其支持的产品前，请先阅读第 1463 页的附录 B，『声明』下的常规信息。

修订版声明

此文档包含 IBM 的所有权信息。它在许可协议中提供，且受版权法的保护。本出版物中包含的信息不包括对任何产品的保证，且提供的任何语句都不需要如此解释。

您可在线或通过当地的 IBM 代表处订购 IBM 出版物。

- 要在线订购出版物，请转至 IBM 出版物中心，网址为: http://www.ibm.com/shop/publications/order
- 要查找当地的 IBM 代表处，请转至 IBM 全球联系人目录，网址为: http://www.ibm.com/planetwide/

要从美国或加拿大的 DB2 市场和销售部订购 DB2 出版物，请致电 1-800-IBM-4YOU（426-4968）。

您发送信息给 IBM 后，即授予 IBM 非独占权限，IBM 可以按它认为适当的任何方式使用或分发您所提供的任何信息而无须对您承担任何责任。

目录

关于本书

《*系统监视器指南和参考*》描述了如何收集关于数据库和数据库管理器的不同类型的信息。

它还解释了如何使用收集的信息来了解数据库活动、改善性能和确定问题的原因。

第 1 部分 数据库监视器的接口

有两种方法可监视数据库中的操作。可查看显示数据库在特定时间点的各方面的状态的信息。或者，可设置事件监视器以捕获特定类型的数据库事件发生时的历史信息。

可使用监视表功能来实时监视数据库操作。例如，可使用监视表函数来检查表空间中使用的总空间量。这些表函数允许您使用 SQL 检查*监视元素*和度量，它们几乎报告了数据库操作的所有方面。监控表函数使用V9.7中引入的更新轻量级高速监视基础结构。除表函数之外，快照监视例程也可用。DB2® 中的快照监视设施使用 V9.7 之前就已存在的监视基础结构。一般来讲，快照监视设施在产品中不再增强；如果可能，请使用监视表功能来检索要查看的数据。

事件监视器在特定类型的事件发生时捕获有关随时间变化的数据库操作的信息。例如，可创建事件监视器来捕获系统中发生锁定和死锁时有关它们的信息。或者，您可创建事件监视器来记录超过您指定的阈值（例如，应用程序或工作负载使用的总处理器时间）的时间。事件监视器以不同格式生成输出；所有事件监视器都会将事件数据写至常规表；某些事件监视器有其他输出选项。

IBM® InfoSphere® Optim™ Performance Manager 提供了一个 Web 界面，可用来隔离和分析典型数据库性能问题。还可查看数据库运行状况摘要并深入了解。有关更多详细信息，请参阅使用 Optim Performance Manager 进行监视（网址为 http://publib.boulder.ibm.com/infocenter/idm/docv3/topic/com.ibm.datatools.perfmgmt.monitor.doc/p_monitor.html）。

第 1 章 数据库监视

词汇*数据库监视*指的是与检查数据库操作状态关联的任务。

对于数据库管理系统的性能和运行状况的维护而言，数据库监视是一个非常重要的活动。为便于进行监视，DB2 从数据库管理器、数据库及所有已连接的应用程序收集信息。可借助此信息执行下列类型及其他类型的任务:

- 根据数据库使用模式预计硬件要求。
- 分析各个应用程序或 SQL 查询的性能。
- 跟踪索引和表的使用情况。
- 查明系统性能下降的原因。
- 评估优化活动（如更改数据库管理器配置参数、添加索引或修改 SQL 查询）的效果。

第 2 章 用于监视的表函数

从 DB2 V9.7 开始，可以通过传统系统监视器的轻量级替代项来访问监视器数据。请使用监视器表函数来收集和查看系统、活动或数据对象的数据。

受监视元素的数据将在内存中持续累积并可供查询。您可以选择接收单一对象（例如服务类 A 或表 TABLE1）的数据或者所有对象的数据。

在数据库分区环境中使用这些表函数时，您可以选择接收单一分区的数据或所有分区的数据。如果您选择接收所有分区的数据，那么这些表函数将对每个分区返回一行。通过使用 SQL，可以对各个分区的值进行求和，以获取跨分区的监视元素值。

使用表函数来监视系统信息

系统监视透视图涵盖数据服务器为了处理应用程序请求而执行的全部工作。在此透视图中，您可以确定数据服务器的整体工作以及为特定应用程序请求子集完成的工作。

此透视图的监视元素被称为“请求监视元素”，它们涵盖所有与处理请求相关联的数据服务器操作。

请求监视元素将在内存中持续地进行累积和聚集，因此立即可供查询。请求监视元素将对各个级别的工作负载管理（WLM）对象层次结构的请求进行聚集：按工作单元、按工作负载或者按服务类。它们还按连接进行聚集。

请使用下列表函数来访问当前系统监视信息:

- MON_GET_SERVICE_SUBCLASS 和 MON_GET_SERVICE_SUBCLASS_DETAILS
- MON_GET_WORKLOAD 和 MON_GET_WORKLOAD_DETAILS
- MON_GET_CONNECTION 和 MON_GET_CONNECTION_DETAILS
- MON_GET_UNIT_OF_WORK 和 MON_GET_UNIT_OF_WORK_DETAILS

这组表函数使您能够下寻到或侧重于特定聚集级别的请求监视元素。表函数成对提供：一个表函数用于对常用数据进行关系访问，另一个表函数用于对全部可用的监视元素进行 XML 访问。

缺省情况下，对于新数据库，这些表函数将收集系统监视信息。您可以使用下列设置中的一个或全部来更改缺省设置:

- 数据库配置参数 **mon_req_metrics** 指定所有服务类中的最低收集级别。
- CREATE/ALTER SERVICE CLASS 语句的 COLLECT REQUEST METRICS 子句指定服务超类的收集级别。使用此设置来将给定服务类的收集级别增大为高于对所有服务类设置的最低收集级别。

每个设置的可能值如下所示:

NONE 不收集请求监视元素

BASE 收集所有请求监视元素

例如，要只收集部分服务类的系统监视信息，请执行以下操作:

1. 将数据库配置参数 **mon_req_metrics** 设置为 NONE。
2. 对于每个需要的服务类，将 CREATE/ALTER SERVICE CLASS 语句的 COLLECT REQUEST METRICS 子句设置为 BASE。

使用表函数来监视活动

活动监视透视图侧重于与执行活动相关的数据服务器处理子集。在 SQL 语句的上下文中，术语“活动”是指 SQL 语句节的执行。

此透视图的监视元素被称为“活动监视元素”，它们是请求监视元素的子集。活动监视元素用于度量为了执行语句节而完成的工作的各个方面。活动监视包括其他信息，例如活动的 SQL 语句文本。

对于进行中的活动而言，活动度量值将在内存中累积。对于作为 SQL 语句的活动而言，活动度量值还将在程序包高速缓存中累积。在程序包高速缓存中，活动度量值将针对每个 SQL 语句节的所有执行进行累积。

请使用下列表函数来访问活动的当前数据:

MON_GET_ACTIVITY_DETAILS
返回关于调用此表函数时进行中的各个活动的数据。数据以关系格式返回，但是，详细度量值在 XML 文档的结果表的 DETAILES 列中返回。

MON_GET_PKG_CACHE_STMT
返回数据库程序包高速缓存中的静态 SQL 语句和动态 SQL 语句的时间点视图。数据以关系格式返回。

MON_GET_PKG_CACHE_STMT_DETAILS
返回一个或多个程序包高速缓存条目的详细度量值。数据以关系格式返回，但是，详细度量值在 XML 文档的结果表的 DETAILES 列中返回。

缺省情况下，对于新数据库，将收集活动监视信息。您可以使用下列设置中的一个或全部来更改缺省设置:

- **mon_act_metrics** 数据库配置参数指定所有工作负载中的最低收集级别。
- CREATE/ALTER WORKLOAD 语句的 COLLECT ACTIVITY METRICS 子句用于指定给定工作负载的收集级别（高于对所有工作负载设置的最低收集级别）。

每个设置的可能值如下所示:

NONE 不收集活动监视元素

BASE 收集所有活动监视元素

例如，要仅收集所选工作负载的活动监视元素，请执行以下操作:

1. 将 **mon_act_metrics** 数据库配置参数设置为 NONE。
2. 将 CREATE/ALTER WORKLOAD 语句的 COLLECT ACTIVITY METRICS 子句设置为 BASE。缺省情况下，其他工作负载的值为 NONE。

使用表函数来监视数据对象

数据对象监视透视图提供关于对数据对象（即，表、索引、缓冲池、表空间和容器）执行的操作的信息。

对于每种对象类型，都有一组不同的监视元素。每当一个请求涉及处理数据对象时，该对象的监视元素都将递增。例如，在处理涉及从特定表中读取行的请求时，该表的“读取行数”度量值将递增。

请使用下列表函数来访问数据对象的当前详细信息:

- MON_GET_BUFFERPOOL
- MON_GET_TABLESPACE
- MON_GET_CONTAINER
- MON_GET_TABLE
- MON_GET_INDEX

这些表函数以关系格式返回数据。

您无法访问数据对象的历史数据。

缺省情况下，将对新数据库收集数据对象监视元素。您可以使用 **mon_obj_metrics** 数据库配置参数来减少表函数所收集的数据量。

此配置参数的可能值如下所示:

NONE 不收集数据对象监视元素

BASE 收集某些数据对象监视元素

扩展 收集所有数据对象监视元素

要停止收集下列表函数所报告的数据对象监视元素，请将 **mon_obj_metrics** 配置参数设置为 NONE。

- MON_GET_BUFFERPOOL
- MON_GET_TABLESPACE
- MON_GET_CONTAINER

对象用法

执行 SQL 语句时，这些语句会使用各种数据库对象（例如，表和索引）。知道语句访问的数据库对象以及此语句对它们的影响可帮助您标识目标以进行监视或性能调整。

下表显示可用于探查数据库对象与语句之间的关系的实体。

表 1. 用于标识对象用法的方法

机制	定义	用法
用法列表	用法列表是一个数据库对象，用于记录以下每个 DML 语句片段，该片段引用特定表或索引并在该片段执行时捕获其相关统计信息。	标识影响表或索引的语句。如果注意到监视数据库对象时某个度量值不寻常，请使用一个用法列表来确定特定语句是否影响了该度量值。还可查看影响该对象的每个语句的统计信息。

表 1. 用于标识对象用法的方法 (续)

机制	定义	用法
带有实际值的代码段说明	段说明是有关优化器针对 SQL 语句选择的存取方案的一组信息。可捕获部分实际值来作为说明的一部分。部分实际值是部分执行时收集的运行时统计信息。	标识语句影响的表或索引。可查看每个表或索引的统计信息并使用这些统计信息来确定此语句对每个对象的影响以及可能需要调整的位置。

可使用某个用法列表或带有实际值的段说明中的信息作为性能调整的基线数据。在调整语句或数据库配置参数之前收集有关对象用法的信息。进行调整后，再次收集此信息以验证调整是否改进了性能。

标识影响表的语句

使用一些用法列表以在影响特定表的 DML 语句段执行时标识这些语句段。可查看每个语句的统计信息并使用这些统计信息来确定可能需要其他监视或调整的位置。

开始之前

执行以下任务:

- 标识要查看其对象用法统计信息的表。可使用 MON_GET_TABLE 表函数来查看一个或多个表的监视度量值。
- 要发出必需语句，请确保每个语句的授权标识所持有的特权包括 DBADM 权限或 SQLADM 权限。
- 确保您对 MON_GET_TABLE_USAGE_LIST 和 MON_GET_USAGE_LIST_STATUS 表函数具有 EXECUTE 特权。

关于此任务

查看 MON_GET_TABLE 表函数的输出时，可能会见到不寻常的监视元素值。可使用一些用法列表来确定是否有任何 DML 语句影响了此值。

用法列表包含有关特定时间段内影响表的每个语句的锁定和缓冲池使用情况之类的因子的统计信息。如果确定某个语句对表有负面影响，请使用这些统计信息来确定是否需要进一步监视或如何调整此语句。

过程

要标识影响表的语句，请执行以下操作:

1. 通过发出以下命令，将 **mon_obj_metrics** 配置参数设置为 EXTENDED:

   ```
   DB2 UPDATE DATABASE CONFIGURATION USING MON_OBJ_METRICS EXTENDED
   ```

   将此配置参数设置为 EXTENDED 可确保针对用法列表中的每个条目收集统计信息。
2. 通过使用 CREATE USAGE LIST 语句来为表创建用法列表。 例如，要为 SALES.INVENTORY 表创建 INVENTORYUL 用法列表，请发出以下命令:

   ```
   CREATE USAGE LIST INVENTORYUL FOR TABLE SALES.INVENTORY
   ```
3. 通过使用 SET USAGE LIST STATE 语句来激活对象用法统计信息的收集。 例如，要激活针对 INVENTORYUL 用法列表的收集，请发出以下命令:

```
SET USAGE LIST INVENTORYUL STATE = ACTIVE
```

4. 在收集对象统计信息期间，使用 MON_GET_USAGE_LIST_STATUS 表函数来确保用法列表处于活动状态并且已对此用法列表分配足够内存。 例如，要检查 INVENTORYUL 用法列表的状态，请发出以下命令:

```
SELECT MEMBER,
       STATE,
       LIST_SIZE,
       USED_ENTRIES,
       WRAPPED
FROM TABLE(MON_GET_USAGE_LIST_STATUS('SALES', 'INVENTORYUL', -2))
```

5. 经历要收集对象用法统计信息的时间段后，应使用 SET USAGE LIST STATE 语句来取消激活用法列表数据的收集。 例如，要取消激活对 INVENTORYUL 用法列表的收集，请发出以下命令:

```
SET USAGE LIST SALES.INVENTORYUL STATE = INACTIVE
```

6. 使用 MON_GET_TABLE_USAGE_LIST 函数来查看收集的信息。 可查看收集统计信息的时间段内影响表的一部分或全部语句的统计信息。 例如，如果只想查看读取最多表行的 10 个语句，请发出以下命令:

```
SELECT MEMBER,
       EXECUTABLE_ID,
       NUM_REFERENCES,
       NUM_REF_WITH_METRICS,
       ROWS_READ,
       ROWS_INSERTED,
       ROWS_UPDATED,
       ROWS_DELETED
FROM TABLE(MON_GET_TABLE_USAGE_LIST('SALES', 'INVENTORYUL', -2))
ORDER BY ROWS_READ DESC
FETCH FIRST 10 ROWS ONLY
```

7. 如果要查看影响此表的语句的文本，请将 MON_GET_TABLE_USAGE_LIST 输出中的 **executable_id** 元素的值用作 MON_GET_PKG_CACHE_STMT 表函数的输入。例如，发出以下命令以查看特定语句的文本:

```
SELECT STMT_TEXT
FROM TABLE
(MON_GET_PKG_CACHE_STMT(NULL,
x'0100000000000000 7C000000000000000000000000020020081126171720728997', NULL,
-2))
```

8. 使用语句列表和为语句提供的统计信息来确定需要额外监视或调整的位置。 例如，**pool_writes** 监视元素值很低（相对于 **direct_writes** 监视元素值）的语句可能存在需要注意的缓冲池问题。

下一步做什么

不需要用法列表中的信息时，请使用 SET USAGE LIST STATE 语句来释放与用法列表相关联的内存。例如，要释放与 INVENTORYUL 用法列表相关联的内存，请发出以下命令:

```
SET USAGE LIST SALES.INVENTORYUL STATE = RELEASED
```

标识语句对数据库对象的影响

使用包括部分实际值信息的部分说明来标识语句对数据库对象的影响。可使用有关语句段对每个表或索引的影响的统计信息来确定是否需要其他监视或调整。

开始之前

执行以下任务:

- 标识要查看其对象用法统计信息的语句。
- 确保您已将说明表迁移至 DB2 V10.1。
- 确保未启用自动统计信息概要文件生成。
- 确保您具有调用 EXPLAIN_FROM_ACTIVITY 过程所需的特权。

关于此任务

标识要查看其对象用法统计信息的语句后，可获取包括段实际值信息的段说明。段实际值信息指示此语句对执行它时它使用的每个表或索引的影响。

实际值信息包括每个表或索引的锁定和缓冲池使用情况之类的因素的运行时统计信息。可将这些统计信息与基线数据进行比较并使用它们来确定可能需要其他监视或调整的位置。

过程

要确定语句对数据库对象的影响，请执行以下操作:

1. 通过发出以下命令在数据库级别启用段实际值收集:

   ```
   DB2 UPDATE DATABASE CONFIGURATION USING SECTION_ACTUALS BASE
   ```

2. 创建工作负载以收集以下活动的段实际值信息: 这些活动由发出此语句的应用程序提交。 例如，要为 TEST 应用程序提交的活动创建 ACTWORKLOAD 工作负载并启用对这些活动的收集，请发出以下命令:

   ```
   CREATE WORKLOAD ACTWORKLOAD APPLNAME ('TEST')
   COLLECT ACTIVITY DATA ON ALL WITH DETAILS,SECTION INCLUDE ACTUALS BASE
   ```

   还可通过以下方式来启用对部分实际值的收集:

   - CREATE SERVICE CLASS 或 ALTER SERVICE CLASS 语句
   - CREATE WORK ACTION SET 或 ALTER WORK ACTION SET 语句
   - WLM_SET_CONN_ENV 过程
   - **section_actuals** 配置参数

3. 通过使用 CREATE EVENT MONITOR 语句来创建活动事件监视器。 例如，要创建 ACTEVMON 活动事件监视器，请发出以下命令:

   ```
   CREATE EVENT MONITOR ACTEVMON
          FOR ACTIVITIES
          WRITE TO TABLE
          CONTROL (TABLE CONTROL_ACTEVMON ),
          ACTIVITY (TABLE ACTIVITY_ACTEVMON ),
          ACTIVITYSTMT (TABLE ACTIVITYSTMT_ACTEVMON ),
          ACTIVITYVALS (TABLE ACTIVITYVALS_ACTEVMON ),
          ACTIVITYMETRICS (TABLE ACTIVITYMETRICS_ACTEVMON )
   ```

4. 通过使用 SET EVENT MONITOR STATE 语句来激活您所创建的活动事件监视器。例如，要激活 ACTEVMON 活动事件监视器，请发出以下命令:

   ```
   SET EVENT MONITOR ACTEVMON STATE 1
   ```

5. 运行发出以下语句的应用程序: 您想要查看此语句的对象统计信息。
6. 通过使用以下命令来查找语句段的标识信息，以查询活动事件监视器表:

```
SELECT APPL_ID,
       UOW_ID,
       ACTIVITY_ID,
       STMT_TEXT
FROM ACTIVITYSTMT_ACTEVMON
```

7. 通过使用活动标识信息作为 EXPLAIN_FROM_ACTIVITY 过程的输入来获取带有实际值的段说明。 例如，要获取应用程序标识为 *N2.DB2INST1.0B5A12222841 工作单元标识为 16 并且活动标识为 4 的段的段说明，请发出以下命令:

```
CALL EXPLAIN_FROM_ACTIVITY( '*N2.DB2INST1.0B5A12222841', 16, 4, 'ACTEVMON',
 'MYSCHEMA', ?, ?, ?, ?, ? )
```

   您将获得类似如下样本输出的输出:

```
Value of output parameters
--------------------------
Parameter Name : EXPLAIN_SCHEMA
Parameter Value : MYSCHEMA

Parameter Name : EXPLAIN_REQUESTER
Parameter Value : GSDBUSER3

Parameter Name : EXPLAIN_TIME
Parameter Value : 2010-11-23-10.51.09.631945

Parameter Name : SOURCE_NAME
Parameter Value : SQLC2J21

Parameter Name : SOURCE_SCHEMA
Parameter Value : NULLID

Parameter Name : SOURCE_VERSION
Parameter Value :

Return Status = 0
```

8. 使用 **db2exfmt** 命令来格式化说明数据。 使用 EXPLAIN_FROM_ACTIVITY 过程输出中的 **explain_requester**、**explain_time**、**source_name**、**source_schema** 和 **source_version** 参数的值作为此命令的输入。
9. 查看说明输出以确定此段对执行此段时此段使用的数据库对象的影响。 输出中的统计信息可能指示需要其他监视或调整。 例如，如果此段使用的表的 **lock_wait** 监视元素值很高，那么可能需要锁定管理。
10. 如果调整此语句，请重复步骤 第 10 页的 5 到 9 以验证性能是否得到改善。

下一步做什么

通过使用 SET EVENT MONITOR STATE 语句来取消激活活动事件监视器。例如，要取消激活 ACTEVMON 活动事件监视器，请发出以下命令:

```
SET EVENT MONITOR ACTEVMON STATE 0
```

使用表函数来监视锁定

可使用表函数来检索有关锁定的信息。与请求、活动或数据对象监视元素不同，有关锁定的信息始终可从数据库管理器获取。不必启用此信息的收集。

使用以下监视器表函数来访问系统中的锁定的当前信息:

- MON_GET_LOCKS
- MON_GET_APPL_LOCKWAIT

这两个表函数都以关系格式返回数据。

使用表函数来监视系统内存

可以使用表函数来检索有关系统内存使用量的信息。

可以在内存集合（这些集合是操作系统中的内存分配）级别检查内存使用量。还可以通过给定内存集合中的特定内存池来检查内存使用量。使用以下监视函数来访问有关内存使用量的当前信息:

- MON_GET_MEMORY_SET
- MON_GET_MEMORY_POOL

其他监视表函数

除了返回有关系统、活动、锁定或数据对象的信息的表函数以外，还有一些表函数返回各种类型的其他信息。其中某些函数返回与快速通信管理器 (FCM) 及表空间扩展数据块移动状态相关的信息。

可随时使用下面的每个表函数。与返回请求度量值（系统监视透视图）、系统度量值（活动监视视图）或与数据对象相关的度量值（数据对象监视透视图）的表函数不同，不必先启用这些函数返回的监视元素的集合。

- MON_GET_FCM
- MON_GET_FCM_CONNECTION_LIST
- MON_GET_EXTENT_MOVEMENT_STATUS

在 XML 文档中返回监视数据的接口

从 DB2 V9.7 开始，将通过 XML 文档中的元素来报告某些监视数据。

通过使用 XML 来报告监视信息，提高了可扩展性和灵活性。可以对产品添加新的监视元素，而不必向输出表中添加新列。此外，根据您的需要，可以采用多种方法来处理 XML 文档。例如:

- 可以使用 XQuery 对 XML 文档运行查询。
- 可以使用 XSLTRANSFORM 标量函数将文档变换为其他格式。
- 可以通过使用内置 MON_FORMAT_XML_* 格式化函数或者 XMLTABLE 表函数将 XML 文档的内容作为格式化文本来查看。

包含监视元素的 XML 文档由多个监视界面生成。下列各节描述了如何将结果作为 XML 文档返回。

- 『其名称以“_DETAILS”结尾的监视表函数』
- 第 15 页的『由事件监视器返回的 XML 数据』.

其名称以“_DETAILS”结尾的监视表函数

这些表函数的示例包括:

- MON_GET_PKG_CACHE_STMT_DETAILS
- MON_GET_WORKLOAD_DETAILS

- MON_GET_CONNECTION_DETAILS
- MON_GET_SERVICE_SUBCLASS_DETAILS
- MON_GET_ACTIVITY_DETAILS
- MON_GET_UNIT_OF_WORK_DETAILS

这些表函数将从系统和活动监视透视图中返回监视元素。这些函数所返回的大多数监视元素都包含在 XML 文档中。例如，MON_GET_CONNECTION_DETAILS 表函数将返回以下各列:

- APPLICATION_HANDLE
- MEMBER
- DETAILS

每一行的 DETAILS 列中都包含一个 XML 文档，此 XML 文档中包含监视元素数据。此 XML 文档由若干个与监视元素相对应的文档元素组成。第 14 页的图 1 说明包含 XML 文档的 DETAILS 列。此外，它显示了在 DETAILS 列的 XML 文档中返回的监视元素。

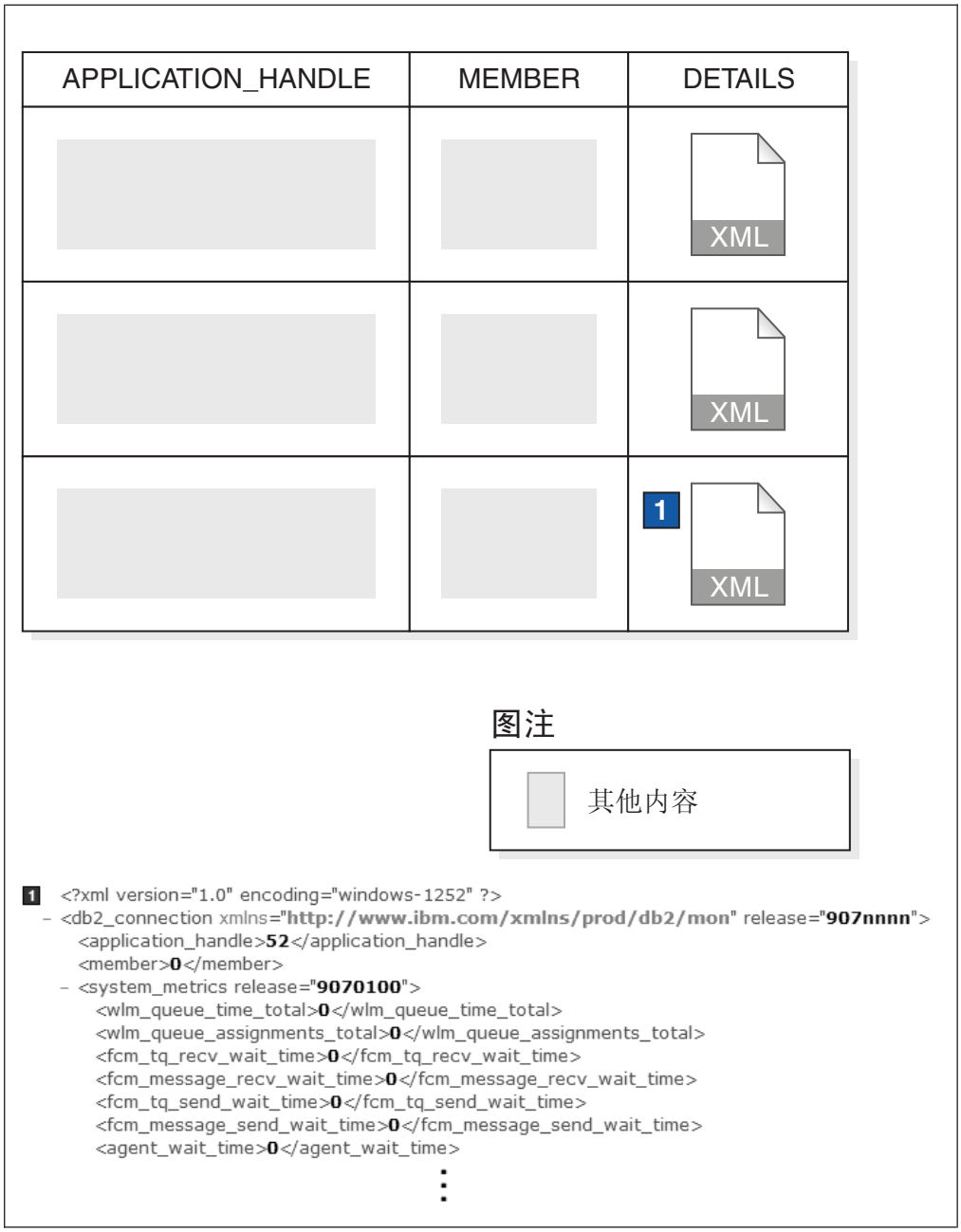


图 1. MON_GET_CONNECTION_DETAILS 返回的表，它显示了包含 XML 文档的 DETAILS 列. 在表的下方显示了第三行中的 XML 文档的内容（ **1** ）。

在前一个示例中，XML 文档元素 <agent_wait_time> 对应于 **agent_wait_time** 监视元素。

在 sqllib/misc/DB2MonRoutines.xsd 文件中提供了 DETAILS 列中所返回的 XML 文档的模式。可以在 sqllib/misc/DB2MonCommon.xsd 文件中找到更多详细信息。

DETAILS 列中的文档所包含的某些监视元素可以组成更高级别的文档元素。例如，用于报告与活动相关的度量值的监视元素是 **activity_metrics** 元素的一部分。同样，系统级别的度量值是 **system_metrics** 元素的一部分。

由事件监视器返回的 XML 数据

一些事件监视器以 XML 格式返回数据。这些格式在表 2 中进行了汇总。后面的部分中描述了由各种事件监视器返回的 XML 文档的有关详细信息。

表 2. 由各种事件监视器返回的 XML 文档

事件监视器	事件监视器输出格式	返回的 XML 文档
『统计信息事件监视器』	• Relational 表 • 文件 • 命名管道	DETAILS_XML
第 16 页的『活动事件监视器』	• Relational 表 • 文件 • 命名管道	DETAILS_XML
第 17 页的『程序包高速缓存事件监视器』	无格式事件 (UE) 表	METRICS 只有在 UE 表已变换为 XML 或关系表后，才能查看此文档。
第 17 页的『工作单元事件监视器』	无格式事件 (UE) 表	METRICS 只有在 UE 表已变换为 XML 或关系表后，才能查看此文档。

统计信息事件监视器

当您创建统计信息事件监视器以报告 event_scstats 和 event_wlstats 逻辑数据组（请参阅 第 57 页的『event_scstats 逻辑数据组』和 第 63 页的『event_wlstats 逻辑数据组』）中的监视元素时，生成的其中一列是 DETAILS_XML。如果将事件监视器写入表中，那么 DETAILS_XML 是其中一列。如果将它写入某个文件或命名管道，那么 DETAILS_XML 是自描述数据流的一部分。文档中包含 **system_metrics** 监视元素，而此监视元素中又包含许多用于报告与系统相关的度量值的监视元素。第 16 页的图 2 显示由统计信息事件监视器所生成表的 DETAILS_XML 列中的 XML 文档:

图 2. 统计信息事件监视器的输出（在写入表时），显示了 DETAILS_XML 列. 在表的下方显示了第三行中的 XML 文档的内容（ **1** ）。

请参阅第 252 页的『为 system_metrics 和 activity_metrics 监视元素写入 XML 的信息』以了解统计信息事件监视器的 XML 输出的模式。

注: 由统计信息事件监视器生成的 DETAILS_XML 列中的 XML 文档所报告的 system_metrics，也是由 MON_GET_SERVICE_SUBCLASS_DETAILS 和 MON_GET_WORKLOAD_DETAILS 表函数返回的 DETAILS 列中包含的 XML 文档的一部分。

活动事件监视器

当您创建活动事件监视器以报告 event_activity 逻辑数据组（请参阅 第 40 页的『event_activity 逻辑数据组』）中的监视元素时，生成的其中一列是 DETAILS_XML。如果将事件监视器写入表中，那么 DETAILS_XML 是其中一列。如果将它写入某个文件或命名管道，那么 DETAILS_XML 是自描述数据流的一部分。无论是哪种方式，文档中都包含 **activity_metrics** 监视元素，而此监视元素中又包含许多用于报告与活动相关的度量值的监视元素。请参阅第 252 页的『为 system_metrics 和 activity_metrics 监视元素写入 XML 的信息』以了解活动事件监视器的 XML 输出的模式。

注: 由活动事件监视器生成的 DETAILS_XML 列中的 XML 文档所报告的 activity_metrics，也是由 MON_GET_ACTIVITY_DETAILS 表函数返回的 DETAILS 列中包含的 XML 文档的一部分。

程序包高速缓存事件监视器

程序包高速缓存事件监视器将它的输出写入无格式的事件 (UE) 表。如果您使用 EVMON_FORMAT_UE_TO_TABLES 表函数来转换此表中的数据，那么生成的其中一个表是 PKGCACHE_EVENT。此表中包含 METRICS 列。在每一行中，此列都包含一个 XML 文档，此 XML 文档具有与程序包高速缓存事件监视元素相关联的元素。

注: 从 DB2 V9.7 修订包 1 开始，EVMON_FORMAT_UE_TO_TABLES 还会为此事件监视器所收集的度量值创建一个单独的表（称为 PKGCACHE_METRICS）。此表中包含的信息与 PKGCACHE_EVENT 表的 METRICS 列中报告的信息相同。因此，您可以从 PKGCACHE_METRICS 表的各列中检索度量值，也可以使用 PKGCACHE_EVENT 表的 METRICS 列中所包含的 XML 文档。请参阅第 205 页的『EVMON_FORMAT_UE_TO_TABLES 为程序包高速缓存事件监视器写至关系表的信息』以了解详细信息。

EVMON_FORMAT_UE_TO_XML 函数也会生成一个 XML 文档，此 XML 文档具有与程序包高速缓存事件监视元素相关联的元素。例如，XML 文档元素 <num_executions> 对应于 **num_executions** 监视元素。请参阅第 213 页的『为程序包高速缓存事件监视器写入 XML 的信息』以了解程序包高速缓存事件监视器的 XML 输出的模式。

工作单元事件监视器

工作单元事件监视器将它的输出写入无格式的事件 (UE) 表。如果您使用 EVMON_FORMAT_UE_TO_TABLES 表函数来转换此表中的数据，那么生成的其中一个表是 UOW_EVENT。此表包含一个 METRICS 列，此列中包含一个 XML 文档，此 XML 文档具有与工作单元事件监视元素相关联的元素。

注: 从 DB2 V9.7 修订包 1 开始，EVMON_FORMAT_UE_TO_TABLES 还会为此事件监视器所收集的度量值创建一个单独的表（称为 UOW_METRICS）。此表中包含的信息与 UOW_EVENT 表的 METRICS 列中报告的信息相同。因此，您可以从 UOW_METRICS 表的各列中检索度量值，也可以使用 UOW_EVENT 表的 METRICS 列中所包含的 XML 文档。请参阅第 160 页的『EVMON_FORMAT_UE_TO_TABLES 为工作单元事件监视器写至关系表的信息』以了解详细信息。

EVMON_FORMAT_UE_TO_XML 函数也会生成一个 XML 文档，此 XML 文档具有与工作单元事件监视元素相关联的元素。例如，XML 文档元素 <workload_name> 对应于 **workload_name** 监视元素。请参阅第 170 页的『EVMON_FORMAT_UE_TO_XML 为工作单元事件监视器写至 XML 的信息』以了解工作单元事件监视器的 XML 输出的模式。

用于将 XML 监视器信息作为格式化文本来查看的界面

根据您想如何查看或使用由监视器界面所生成的 XML 文档中包含的数据，可以采用多种方法来查看这些数据。可以使用 XQuery 来查询和处理由监视界面返回的 XML 文档。还可以使用表函数来格式化 XML 文档，以便更容易阅读。

XQuery 提供了一个功能强大并且灵活的界面来查询和处理 XML 数据。但是，有时候您可能想采用基于文本的格式来查看元素数据。根据您的需要，可以采用面向列的格式或者面向行的格式来查看 XML 文档中所包含的监视元素。如果您知道要查看哪些监视元素，那么采用面向列的格式很有用。如果您事先不知道要检查哪些监视元素（例

如，当您想查看最靠前的五种类型的等待时间时），那么采用面向行的格式很有用。接下来的章节描述了您可以采用两种方式将 XML 文档中所包含的监视数据作为格式化文本来查看。

- 『查看采用面向列的格式的监视元素』
- 第 19 页的『查看采用面向行的格式的监视元素』

查看采用面向列的格式的监视元素

XMLTABLE 表函数将 XML 文档作为输入并将它转换为关系表，以便所选择的每个 XML 文档元素都显示为一列。如果您知道要显示哪些监视元素，那么此方法很有用。例如，假定您已经创建了一个称为 DBSTATS 的统计信息事件监视器，用来收集 event_scstats 逻辑数据组中的信息。（请参阅 第 57 页的『event_scstats 逻辑数据组』，以了解有关与此逻辑数据组相关联的监视元素的更多信息。）此逻辑组中的监视元素包括 **details_xml**，[1]它实际上是一个 XML 文档，它本身包含组成 **system_metrics** 监视元素的度量值。（请参阅 第 252 页的『system_metrics』，以了解有关与 system_metrics 监视元素相关联的监视元素的更多信息。）要查看 details_xml 中包含的特定 **system_metrics** 监视元素（例如，rows_returned、total_section_time 或 total_cpu_time），可以使用 XMLTABLE 表函数来格式化从统计信息事件监视器所返回的 details_xml 文档中选择的监视元素。以下示例对此进行了说明。（为了便于表示，SQL 将仅返回特定服务类的结果。）

```
SELECT partition_number,
       service_class_id,
       statistics_timestamp,
       event.rows_returned,
       event.total_section_time,
       event.total_cpu_time
 FROM  SCSTATS_DBSTATS as DBSTATS,
 XMLTABLE( XMLNAMESPACES( DEFAULT 'http://www.ibm.com/xmlns/prod/db2/mon' ),
           '$metrics/system_metrics' PASSING XMLPARSE( DOCUMENT DBSTATS.details.xml ) as "metrics"
           COLUMNS
           rows_returned         BIGINT      PATH 'rows_returned',
           total_section_time    BIGINT      PATH 'total_section_time',
           total_cpu_time        BIGINT      PATH 'total_cpu_time'
          ) AS EVENT
 WHERE service_class_id = 12;
```

以下输出显示了此查询的结果:

```
PARTITION_NUMBER SERVICE_CLASS_ID STATISTICS_TIMESTAMP       ROWS_RETURNED        TOTAL_SECTION_TIME   TOTAL_CPU_TIME
---------------- --------------- -------------------------- -------------------- -------------------- --------------------
               0              12 2010-01-05-12.14.37.001717                  402                  990              1531250
               0              12 2010-01-05-12.15.00.035409                  402                  990              1531250
               0              12 2010-01-05-12.20.00.021884                  412                 1064              1609375
               0              12 2010-01-05-12.25.00.039175                  422                 1075              1687500
               0              12 2010-01-05-12.29.59.950137                  432                 1104              1765625
               0              12 2010-01-05-12.34.59.948979                  442                 1130              1796875
               0              12 2010-01-05-12.39.59.903928                  452                 1149              1890625
               0              12 2010-01-05-12.44.59.953596                  462                 1178              1953125
               0              12 2010-01-05-12.49.59.970059                  473                 1207              2062500
               0              12 2010-01-05-12.54.59.971990                  483                 1230              2109375

  10 record(s) selected.
```

在此实例中，前三列将直接显示在由统计信息事件监视器生成的 SCSTATS_DBSTATS 表中。最后三列是从该表的 DETAILS_XML 列中的 XML 文档抽取的度量值监视元素。

1. 注意：在这些主题中，采用小写字母形式的 details_xml 指的是 XML 文档 details_xml。而采用大写字母形式的 DETAILS_XML 指的是称为 DETAILS_XML 的关系表中的一列，此关系表中包含 details_xml 文档。

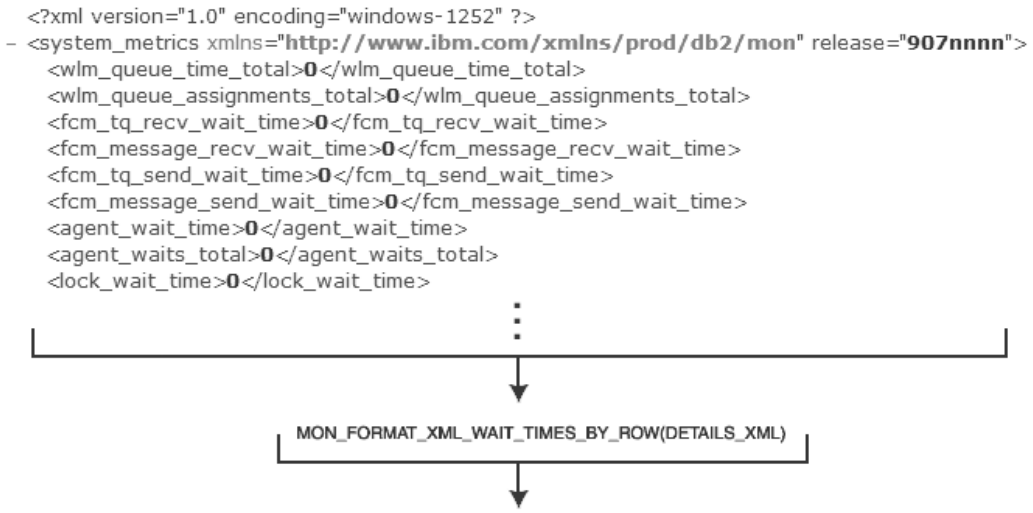
有关使用 XMLTABLE 的更多信息，请参阅有关该函数的文档。还可以在有关各种 MON_GET_*_DETAILS 函数的文档中查看有关使用 XMLTABLE 来查看监视元素的示例。

查看采用面向行的格式的监视元素

在 DB2 V9.7 修订包 1 中，引入了其名称格式为 MON_FORMAT_XML_* _BY_ROW 的表函数，通过这些表函数能够快速显示 XML 文档中所包含的度量值监视元素。它们采用基于行的格式报告度量值，每个监视元素都单独占用一行。此组中包括下列函数:

- MON_FORMAT_XML_COMPONENT_TIMES_BY_ROW
- MON_FORMAT_XML_TIMES_BY_ROW
- MON_FORMAT_XML_WAIT_TIMES_BY_ROW
- MON_FORMAT_XML_METRICS_BY_ROW

例如，统计信息事件监视器 DETAILS_XML 所返回的 XML 文档看起来可能与 图 3 的第一部分中所显示的内容相似。如果您使用 MON_FORMAT_XML_WAIT_TIMES_BY_ROW 函数来调整 DETAILS_XML 内容的格式，那么输出看起来将与该图底部的表相似。




```
METRIC_NAME                       TOTAL_TIME_VALUE     COUNT                PARENT_METRIC_NAME
--------------------------------- -------------------- -------------------- ---------------------------
WLM_QUEUE_TIME_TOTAL                                 0                    0 TOTAL_WAIT_TIME
FCM_TQ_RECV_WAIT_TIME                                0                    0 FCM_RECV_WAIT_TIME
FCM_MESSAGE_RECV_WAIT_TIME                           0                    0 FCM_RECV_WAIT_TIME
FCM_TQ_SEND_WAIT_TIME                                0                    0 FCM_SEND_WAIT_TIME
FCM_MESSAGE_SEND_WAIT_TIME                           0                    0 FCM_SEND_WAIT_TIME
AGENT_WAIT_TIME                                      0                    0 TOTAL_WAIT_TIME
LOCK_WAIT_TIME                                       0                    0 TOTAL_WAIT_TIME
DIRECT_READ_TIME                                     0                    0 TOTAL_WAIT_TIME
DIRECT_WRITE_TIME                                    0                    0 TOTAL_WAIT_TIME
LOG_BUFFER_WAIT_TIME                                 0                    0 TOTAL_WAIT_TIME
LOG_DISK_WAIT_TIME                                   0                    0 TOTAL_WAIT_TIME
⋮
```

图 3. 包含由其中一个 MON_FORMAT_XML_* 函数处理的监视数据的 XML 文件. 此示例说明了如何使用 MON_FORMAT_XML_WAIT_TIMES_BY_ROW 函数。将只返回等待时间；此特定函数将排除 XML 文件中所包含的其他度量值（例如，wlm_queue_assignments_total）。

根据您使用的特定函数不同，返回的列数也将不同。例如，MON_FORMAT_XML_METRICS_BY_ROW 将返回两列，一列用于显示度量值名称，一列用于显示其相应值:

```
METRIC_NAME               VALUE
------------------------- --------------------
WLM_QUEUE_TIME_TOTAL                         0
WLM_QUEUE_ASSIGNMENTS_TOT                    0
FCM_TQ_RECV_WAIT_TIME                        0
FCM_MESSAGE_RECV_WAIT_TIM                    0
FCM_TQ_SEND_WAIT_TIME                        0
⋮
```

相比之下，MON_FORMAT_XML_TIMES_BY_ROW 将返回四列:

```
METRIC_NAME                 TOTAL_TIME_VALUE     COUNT      PARENT_METRIC_NAME
--------------------------- -------------------- ---------- ----------------------
WLM_QUEUE_TIME_TOTAL                           0          0 TOTAL_WAIT_TIME
FCM_TQ_RECV_WAIT_TIME                          0          0 FCM_RECV_WAIT_TIME
FCM_MESSAGE_RECV_WAIT_TIME                     0          0 FCM_RECV_WAIT_TIME
FCM_TQ_SEND_WAIT_TIME                          0          0 FCM_SEND_WAIT_TIME
FCM_MESSAGE_SEND_WAIT_TIME                     0          0 FCM_SEND_WAIT_TIME
⋮
```

当您不知道要查看哪些元素时，MON_FORMAT_XML_* _BY_ROW 函数就很有用。例如，您可能想查看名为 CLPWORKLOAD 的工作负载的前 10 个等待时间监视元素。要收集此信息，可以创建一个称为 DBSTATS 的统计信息事件监视器（event_wlstats 逻辑数据组）。假定您设置此事件监视器以写入某个表，那么它会将度量值记录在 DETAILS_XML 列中。一旦使用监视数据填充了事件监视器的输出表，就可以构造一个使用 MON_FORMAT_XML_WAIT_TIMES_BY_ROW 函数来抽取您要查看的监视元素的查询。

```
SELECT SUBSTR(STATS.WORKLOAD_NAME,1,15) AS WORKLOAD_NAME,
       SUBSTR(METRICS.METRIC_NAME,1,30) AS METRIC_NAME,
       SUM(METRICS.TOTAL_TIME_VALUE) AS TOTAL_TIME_VALUE
FROM   WLSTATS_DBSTATS AS STATS,
       TABLE(MON_FORMAT_XML_WAIT_TIMES_BY_ROW(STATS.DETAILS_XML)) AS METRICS
WHERE  WORKLOAD_NAME='CLPWORKLOAD' AND (PARENT_METRIC_NAME='TOTAL_WAIT_TIME')
GROUP  BY WORKLOAD_NAME,METRIC_NAME
ORDER  BY TOTAL_TIME_VALUE DESC
FETCH FIRST 10 ROWS ONLY
```

切记: “耗用时间”监视元素按层次结构进行组织。在此示例中，为了避免存在重复计算的等待时间，将只包括会累积到 **total_wait_time** 中的监视元素（请参阅前一个 SQL 语句中的 WHERE 子句）。否则，**total_wait_time** 本身也会包括在结果中，total_wait_time 包括若干个单独的等待时间。
下面的输出显示上述查询的结果可能为如下所示:

```
WORKLOAD_NAME   METRIC_NAME                    TOTAL_TIME_VALUE
--------------- ------------------------------ --------------------
CLPWORKLOAD     LOCK_WAIT_TIME                        15138541
CLPWORKLOAD     DIRECT_READ_TIME                       6116231
CLPWORKLOAD     POOL_READ_TIME                         6079458
CLPWORKLOAD     DIRECT_WRITE_TIME                       452627
CLPWORKLOAD     POOL_WRITE_TIME                         386208
CLPWORKLOAD     IPC_SEND_WAIT_TIME                      283172
CLPWORKLOAD     LOG_DISK_WAIT_TIME                      103888
CLPWORKLOAD     DIAGLOG_WRITE_WAIT_TIME                  78198
CLPWORKLOAD     IPC_RECV_WAIT_TIME                       15612
CLPWORKLOAD     TCPIP_SEND_WAIT_TIME                      3291

  10 record(s) selected.
```

注: MON_FORMAT_XML_*_BY_ROW 函数仅返回用于跟踪测量值或*度量值*的监视元素。这些监视元素包括用于跟踪等待时间、组件时间以及计数器的监视元素。它们不会返回 XML 文档中包含的非度量值监视元素（例如，uow_id 或 activity_id）。

可以使用 XMLTABLE 函数来查看 XML 文档中包含的*任何*元素（包括非度量值元素在内）。但是，最常用的非度量值监视元素是由以 MON_GET_* 开头的监视函数（例如，MON_GET_UNIT_OF_WORK 或 MON_GET_CONNECTION）以列的形式返回的。如果不熟悉 XML，那么您可能会发现使用这些函数来创建查询，比使用 XMLTABLE 函数从 XML 文档中抽取监视元素更快并且更容易。

总结：如果您希望查看非度量值监视元素，那么可能可以将表函数的 MON_GET_* 系列作为 XMLTABLE 函数的良好替代者。如果您希望查看度量值监视元素，那么 MON_FORMAT_XML_*_BY_ROW 表函数可能会满足您的需要。

将 XML 文档中的度量值监视元素作为表行来查看

要查看从事件监视器返回的 XML 文档中包含的与度量值相关的信息，其中一种方法是将它转换为以下格式：让每个监视元素单独占用一行。如果您希望采用基于文本的格式来查看信息，但是具体又不知道要检查哪些监视元素，那么采用此格式就很有用。

关于此任务

要查看由各种监视界面返回的 XML 文档中采用基于行的格式的度量值信息，请使用 MON_FORMAT_XML_* _BY ROW 表函数。在 DB2 V9.7 修订包 1 中引入了这些函数。

过程

对于程序包高速缓存事件监视器所跟踪的语句，此任务中所显示的示例使用 MON_FORMAT_XML_TIMES_BY_ROW 表函数来查看该语句的组件时间。它假定已经创建并激活了一个称为 PKGCACHEEVENTS 的程序包高速缓存事件监视器。程序包高速缓存事件监视器将它的输出写入无格式的事件 (UE) 表。在可以使用 UE 表之前，必须使用 EVMON_FORMAT_UE_TO_TABLES 存储过程将此 UE 表中的数据转换为关系表，或者使用 EVMON_FORMAT_UE_TO_XML 表函数将它转换为 XML。此任务说明了这两种方法中的第一种方法。

1. 首先，对于程序包高速缓存事件监视器写入的无格式事件 (UE) 表，使用 EVMON_FORMAT_UE_TO_TABLES 过程将此表转换为关系表

   ```
   call EVMON_FORMAT_UE_TO_TABLES ('PkgCache',NULL,NULL,NULL,NULL,NULL,
        NULL,0,'SELECT * FROM PKGCACHEEVENTS')
   ```

   此过程将创建两个表:

   - 一个表称为 PKGCACHE_EVENT，它包含称为 METRICS 的一列。此列中又包含具有度量值监视元素的 XML 文档。
   - 另一个表称为 PKGCACHE_METRICS。

     注：您可以直接查看 PKGCACHE_METRICS 表的各列中的度量值，而不必从 PKGCACHE_EVENT 表的 METRICS 列中抽取度量值。但是，当您检查 PKGCACHE_METRICS 表时，度量值将按列而不是按行显示；并不是很容易获得如具有最大值的度量值的排名。

2. 查询上一步所生成的两个表，以根据执行时间来确定哪个语句的成本最高:

   ```
   SELECT EVENTS.EXECUTABLE_ID,
          SUM(METRICS.STMT_EXEC_TIME) AS TOTAL_STMT_EXEC_TIME
   FROM   PKGCACHE_EVENT AS EVENTS,
          PKGCACHE_METRICS AS METRICS
   ```

```
WHERE  EVENTS.XMLID = METRICS.XMLID
GROUP  BY EVENTS.EXECUTABLE_ID
ORDER  BY TOTAL_STMT_EXEC_TIME DESC
FETCH  FIRST 5 ROWS ONLY
```

在上述查询中，连接了在步骤 第 21 页的1 中所生成的两个表，以便 PKGCACHE_EVENT 表中的语句标识可以与它们在 PKGCACHE_METRICS 表中的执行时间相关联。

```
EXECUTABLE_ID                                                                    TOTAL_STMT_EXEC_TIME
-------------------------------------------------------------------------------- --------------------
x'0100000000000001A0300000000000000000000020020091215115933859000'                  250
x'0100000000000001503000000000000000000002002009121511585032800 0'                  191
x'0100000000000002102000000000000000000002002009121511581834300 1'                  129
x'010000000000000C402000000000000000000002002009121511583857800 0'                   41
x'010000000000000B002000000000000000000002002009121511583820300 0'                   38

  5 record(s) selected.
```

结果中的第一项表示总体执行时间最长的语句。

3. 可选： 如果您愿意，可以使用以下 SQL 来显示该语句的文本:

```
SELECT SUBSTR(STMT_TEXT,1,60) AS STMT_TEXT
FROM PKGCACHE_EVENT
WHERE EXECUTABLE_ID = x'0100000000000001A0300000000000000000000020020091215115933859000'
```

结果:

```
STMT_TEXT
------------------------------------------------------------
DROP XSROBJECT MYSCHEMA.EVMON_PKGCACHE_SCHEMA_SQL09070

  1 record(s) selected.
```

4. 对于您在步骤 第 21 页的 2 中所标识的语句，使用 MON_FORMAT_XML_TIMES_BY_ROW 表函数来查看该语句的“耗用时间”监视元素的列表:

```
SELECT SUBSTR(XMLMETRICS.METRIC_NAME,1,30) AS METRIC_NAME,
       XMLMETRICS.TOTAL_TIME_VALUE,
       SUBSTR(XMLMETRICS.PARENT_METRIC_NAME,1,30) AS PARENT_METRIC_NAME
FROM PKGCACHE_EVENT AS EVENTS,
     TABLE(MON_FORMAT_XML_TIMES_BY_ROW(EVENTS.METRICS)) AS XMLMETRICS
WHERE EVENTS.EXECUTABLE_ID=
x'0100000000000001A0300000000000000000000020020091215115933859000'
 AND PARENT_METRIC_NAME='STMT_EXEC_TIME'
ORDER BY XMLMETRICS.TOTAL_TIME_VALUE DESC
```

注意:

- 请记住，“耗用时间”监视元素按层次结构进行组织。为了避免重复计算，结果中将只包括那些累积到 **stmt_exec_time** 的度量值。否则，stmt_exec_time 本身也会包括在结果中，stmt_exec_time 包括若干个单独的组件时间。
- 为了便于说明，包括了由 MON_FORMAT_XML_TIMES_BY_ROW 返回的其中一列 PARENT_METRIC_NAME。

运行此查询时，它将返回下列结果:

```
METRIC_NAME                    TOTAL_TIME_VALUE     PARENT_METRIC_NAME
------------------------------ -------------------- ------------------------------
TOTAL_ACT_WAIT_TIME                             234 STMT_EXEC_TIME
TOTAL_SECTION_PROC_TIME                          15 STMT_EXEC_TIME
```

在此处，您可以看到总的处理时间加起来为 249 ms。将此时间与步骤 第 21 页的 2 中所显示的 250 ms 的总时间进行比较；多余的毫秒数是未包括在 **stmt_exec_time** 中的其他时间（例如，等待时间）造成的。

结果

在前一个示例的结果中，您可以看到度量值的布置：它们按面向行的格式进行显示，每行只显示一个度量值。使用此方法的优点在于，您不需要提前知道要查看哪些度量值或者监视元素。如果您要查看其值排名前五位的耗时度量或位于特定值范围内的度量，那么可轻松创建查询以返回您所关心的结果。相比之下，如果您使用 XMLTABLE 函数将监视元素按列显示，那么需要指定要显示哪些监视元素（否则将显示所有监视元素）。

示例

查看由 MON_GET_*_DETAILS 表函数生成的 DETAILS 列的内容

还可以使用 MON_FORMAT_XML_*_BY_ROW 函数来查看由任何 MON_GET_*_DETAILS 函数返回的 DETAILS 列的内容。例如，MON_GET_CONNECTION_DETAILS 将返回一个 DETAILS 列，此列中包含一个 XML 文档，而此 XML 文档中具有与数据库连接有关的度量值。

例如，要查看所有成员中每个连接的非零组件时间，可以使用以下查询:

```
SELECT CONDETAILS.APPLICATION_HANDLE, SUBSTR(XMLMETRICS.METRIC_NAME,1,30) AS METRIC_NAME,
       SUM(XMLMETRICS.TOTAL_TIME_VALUE) AS TOTAL_TIME_VALUE,
       SUBSTR(XMLMETRICS.PARENT_METRIC_NAME,1,30) AS PARENT_METRIC_NAME
FROM TABLE(MON_GET_CONNECTION_DETAILS(NULL,-1)) AS CONDETAILS,
     TABLE(MON_FORMAT_XML_COMPONENT_TIMES_BY_ROW(CONDETAILS.DETAILS))AS XMLMETRICS
WHERE TOTAL_TIME_VALUE > 0 AND XMLMETRICS.PARENT_METRIC_NAME='TOTAL_RQST_TIME'
GROUP BY CONDETAILS.APPLICATION_HANDLE,
         XMLMETRICS.PARENT_METRIC_NAME,
         XMLMETRICS.METRIC_NAME
ORDER BY CONDETAILS.APPLICATION_HANDLE ASC, TOTAL_TIME_VALUE DESC
```

注意:

- 为了避免重复计算，结果中将只包括那些累积到 **total_rqst_time** 的度量值 (WHERE .... XMLMETRICS.PARENT_METRIC_NAME='TOTAL_RQST_TIME')。否则，total_rqst_time 本身也会包括在结果中，total_rqst_time 包括若干个单独的组件时间。
- 为了便于说明，包括了由 MON_FORMAT_XML_COMPONENT_TIMES_BY_ROW 返回的其中一列 PARENT_METRIC_NAME。

上述查询将返回下列结果:

```
APPLICATION_HANDLE   METRIC_NAME                    TOTAL_TIME_VALUE     PARENT_METRIC_NAME
-------------------- ------------------------------ -------------------- ------------------------------
                  52 TOTAL_SECTION_TIME                             3936 TOTAL_RQST_TIME
                  52 TOTAL_COMPILE_TIME                              482 TOTAL_RQST_TIME
                  52 TOTAL_COMMIT_TIME                                15 TOTAL_RQST_TIME
                  52 TOTAL_ROLLBACK_TIME                               1 TOTAL_RQST_TIME
                 496 TOTAL_COMPILE_TIME                              251 TOTAL_RQST_TIME
                 496 TOTAL_SECTION_TIME                               46 TOTAL_RQST_TIME
                 496 TOTAL_IMPLICIT_COMPILE_TIME                      5 TOTAL_RQST_TIME

  7 record(s) selected.
```

如此示例所示，将只包括组成 **total_rqst_time** 的度量值。如果查询中不包含 WHERE .... XMLMETRICS.PARENT_METRIC_NAME='TOTAL_RQST_TIME' 子句，那么结果将类似如下显示的这些结果:

```
APPLICATION_HANDLE   METRIC_NAME                    TOTAL_TIME_VALUE     PARENT_METRIC_NAME
-------------------- ------------------------------ -------------------- ------------------------------
                  52 TOTAL_RQST_TIME                                4603 -
                  52 TOTAL_SECTION_TIME                             3942 TOTAL_RQST_TIME
                  52 TOTAL_COMPILE_TIME                              537 TOTAL_RQST_TIME
                  52 TOTAL_SECTION_SORT_TIME                         299 TOTAL_SECTION_TIME
                  52 TOTAL_COMMIT_TIME                                15 TOTAL_RQST_TIME
                  52 TOTAL_ROLLBACK_TIME                               1 TOTAL_RQST_TIME
                 496 TOTAL_RQST_TIME                                 341 -
                 496 TOTAL_COMPILE_TIME                              251 TOTAL_RQST_TIME
                 496 TOTAL_SECTION_TIME                               46 TOTAL_RQST_TIME
                 496 TOTAL_IMPLICIT_COMPILE_TIME                      5 TOTAL_RQST_TIME
                 496 TOTAL_SECTION_SORT_TIME                          2 TOTAL_SECTION_TIME

  11 record(s) selected.
```

在此示例中，结果中包括每个连接的 **total_rqst_time** 值，其中包括作为其子代的所有其他元素的值。同样，用*斜体字*显示的各项的值都累积到 **total_section_time** 中。如果未从 WHERE 子句中将它们排除，由于 **total_section_time** 本身也累积到 **total_rqst_time** 中，所以结果中已经将它们重复计算了三次。

第 3 章 事件监视器

监视表函数和快照例程会返回运行该例程的特定时间点的监视元素值，这在您想要检查系统的当前状态时很有用。但是，许多时候您需要正好在特定事件发生时捕获有关系统状态的信息。事件监视器用于此用途。

可创建事件监视器以捕获与系统中发生的不同种类*事件*有关的时间点信息。例如，可创建事件监视器以在超过您定义的特定阈值时捕获信息。捕获的信息包括超过该阈值时运行的应用程序的标识之类的信息。或者，可创建事件监视器以确定发生锁定事件时在运行什么语句。

事件监视器对其捕获数据的事件类型

可使用事件监视器来捕获与系统上发生的许多不同事件种类相关的信息。

下表列示系统中发生的以下事件的类型，您可使用事件监视器监视这些事件。它还描述了为不同事件收集的数据的类型以及收集监视数据的时间。第二列中显示的事件监视器的名称与用于使用 CREATE EVENT MONITOR 语句创建该类型的事件监视器的关键字相对应。

表 3. 事件类型

要监视的事件的类型	事件监视器名称	事件监视器属性	详细信息
锁定和死锁	LOCKING	*此事件监视器的用途*	用于确定锁定或死锁的发生时间以及涉及的应用程序。使用 LOCKING 事件监视器而不是建议不要使用的 DEADLOCKS 事件监视器的优点包括整合报告锁定和死锁事件及包含有关锁定等待和锁定超时的信息。
		收集的数据	有关涉及的应用程序的综合信息，包括参与语句（和语句文本）的标识和要挂起的锁定的列表。
		生成事件数据的时间[1]	检测下列任何事件类型时，根据您配置事件监视器的方式: • 锁定超时 • 死锁 • 超过指定持续时间的锁定等待

表 3. 事件类型 (续)

要监视的事件的类型	事件监视器名称	事件监视器属性	详细信息
执行 SQL 语句或其他衍生数据库活动的操作。	ACTIVITIES	*此事件监视器的用途*	用于跟踪个别语句和其他活动的执行以了解哪些活动正在系统中运行。而且还可用于因为诊断而捕获活动和研究 SQL 的资源消耗。
		收集的数据	活动级别数据，通常对应涉及工作负载管理对象的活动。 • 如果指定 WITH DETAILS 作为针对工作负载管理对象的 CREATE 或 ALTER 语句的 COLLECT ACTIVITY DATA 子句的一部分，那么所收集信息包括具有该对象的活动的语句和编译环境信息。如果还指定了 WITH SECTION，那么还会捕获语句、编译环境、片段环境数据和片段实际情况。 • 如果还在针对工作负载管理对象的 CREATE 或 ALTER 语句上指定了 AND VALUES，那么所收集的信息还将包括具有该对象的活动的输入数据值。
		生成事件数据的时间[1]	• 在已启用 COLLECT ACTIVITY DATA 选项的服务类、工作负载或工作类中执行的活动完成时。 • 已启用 COLLECT ACTIVITY DATA 选项的活动违反阈值时。 • 执行 WLM_CAPTURE_ACTIVITY_IN_PROGRESS 存储过程时。 • 使用 WLM_SET_CONN_ENV 存储过程对其启用活动收集的连接执行活动时。
SQL 语句的执行	STATEMENTS	*此事件监视器的用途*	用于查看因为执行 SQL 语句对数据库发出了什么请求。
		收集的数据	语句启动或停止时间、使用的 CPU、动态 SQL 的文本、SQLCA（SQL 语句的返回码）及其他度量值，如访存计数。对于分区数据库: 使用的 CPU、执行时间、表和表队列信息。 **注意:** • 使用语句事件监视器、数据操作语言 (DML) 语句（例如，INSERT、SELECT、DELETE 和 UPDATE）监视 SQL 过程的执行时，会生成事件。过程语句，例如，变量赋值和控制结构（例如，WHILE 或 IF），不会以确定性方式生成事件。 • 当时间戳记开关设置为 OFF 时，语句启动或停止时间不可用。
		生成事件数据的时间	SQL 语句的结束[2]；对于分区数据库，子节的结束[2]
工作单元（事务）完成	UNIT OF WORK	*此事件监视器的用途*	用于收集在系统上运行的工作单元的资源使用信息和性能指标。此信息可用于的范围包括: 从为应用程序使用的系统资源的付款或退款用途生成报告到诊断运行速度缓慢的例程导致的性能问题的用途。 对 TRANSACTIONS 事件监视器的建议。
		收集的数据	有关工作单元（事务）的信息，例如，开始时间和停止时间以及运行这些工作单元的工作负载和服务类。用于包括有关在工作单元中运行的语句的包或可执行标识的信息以及请求度量值的选项。
		生成事件数据的时间[1]	在工作单元完成之后

表 3. 事件类型 (续)

要监视的事件的类型	事件监视器名称	事件监视器属性	详细信息
从程序包高速缓存中去除片段	PACKAGE CACHE	*此事件监视器的用途*	用于捕获不再在程序包高速缓存中的语句（和相关度量值）的历史记录。如果需要检查内存中不再可用的语句的性能指标，那么可使用此信息。
		收集的数据	包括针对该片段的所有执行聚集的语句文本和度量值。
		生成事件数据的时间[1]	因为从程序包高速缓存中去除了条目。
应用程序建立的与数据库的连接	CONNECTIONS	*此事件监视器的用途*	用于捕获应用程序建立的与数据库的每个连接的度量值和其他监视元素。
		收集的数据	所有应用程序级别计数器。例如，应用程序连接至数据库或与数据库断开连接的时间，或者涉及该应用程序的锁定升级的数目。
		生成事件数据的时间	连接结束[2]
取消激活数据库	DATABASE	*此事件监视器的用途*	用于捕获以下度量值和其他监视元素，它们反映激活后有关整个数据库的信息。
		收集的数据	所有数据库级别计数器。例如，自激活后与数据库建立的连接数、等待锁定所花的时间或插入的数据行数。
		生成事件数据的时间	数据库取消激活[2]
	BUFFERPOOLS TABLESPACES	*此事件监视器的用途*	用于捕获与缓冲池和表空间相关的度量值
		收集的数据	缓冲池、预取程序、页清除程序和每个缓冲池的直接 I/O 的计数器。
		生成事件数据的时间	数据库取消激活[2]
	TABLES	*此事件监视器的用途*	用于捕获与数据库激活后更改的表有关的度量值。
		收集的数据	表级别计数器，例如，读取或写入的行数，或数据、LOB 或索引对象使用的磁盘页数。
		生成事件数据的时间	数据库取消激活[2]
有关工作负载管理对象的统计信息和度量值	STATISTICS	*此事件监视器的用途*	用于捕获与数据库中的工作负载管理对象（例如，服务超类或工作负载）相关的处理度量值。例如，可使用统计信息事件监视器来检查给定工作负载随时间变化的 CPU 利用率。
		收集的数据	从在系统中每个服务类、工作负载或工作类内执行的活动计算而来的统计信息。
		生成事件数据的时间	可按固定时间间隔自动收集统计信息。此时间间隔是使用 **wlm_collect_int** 数据库配置参数定义的。 还可使用 WLM_COLLECT_STATS 存储过程手动收集数据。 **注**: 通过任一收集机制，统计信息监视元素的值在发生收集后重置为 0。
超过工作负载管理器阈值	THRESHOLD VIOLATIONS	*此事件监视器的用途*	用于确定数据库操作期间何时超过您设置的特定阈值。可对各种对象（范围从 CPU 时间、数据库连接数到特定语句的执行）设置阈值。所收集数据可用于各种用途，包括监视潜在问题（例如，达到对临时表空间的限制）。
		收集的数据	阈值违例信息。
		生成事件数据的时间	检测到阈值违例时。阈值是使用 CREATE THRESHOLD 语句定义的。

表 3. 事件类型 (续)

要监视的事件的类型	事件监视器名称	事件监视器属性	详细信息
对数据库或数据库管理器配置的更改	CHANGE HISTORY	*此事件监视器的用途*	捕获对数据库和数据库管理器配置的更改、对注册表设置的更改、DDL 语句的执行和实用程序的执行
		收集的数据	数据库配置参数和数据库管理器配置参数更改、注册表变量更改、DDL 语句的执行、某些 DB2 实用程序和命令的执行以及变更历史记录事件监视器启动。 **注**: 通常，不会捕获在变更历史记录事件监视器处于不活动状态或数据库脱机时发生的事件的相关信息。但是，会记录对注册表变量和配置参数的更改。
		生成事件数据的时间[1]	监视器启动期间，参数或变量发生更改的时间或命令、DDL 或实用程序的完成时间。

注意:

1. 如果在活动事件监视器处于活动状态时数据库被取消激活，那么会废弃队列中积压的活动记录。为确保您获取所有活动事件监视器记录并且没有任何废弃记录，请在取消激活数据库之前取消激活该活动事件监视器。显式取消激活活动事件监视器后，事件监视器取消激活前会处理队列中所有积压的活动记录。
2. 除了自动发生数据收集的已定义时间外，还可使用 FLUSH EVENT MONITOR SQL 语句来生成事件。此方法生成的事件将使用所有监视器类型（DEADLOCKS 和 DEADLOCKS WITH DETAILS 除外）的当前数据库监视器值写入，这些监视器类型与清空的事件监视器相关联。

表 4. 不推荐使用的事件监视器的事件类型

要监视的事件的类型	事件监视器名称	事件监视器属性	详细信息
死锁	DEADLOCKS[2]	*此事件监视器的用途*	用于确定死锁的发生时间以及涉及的应用程序。
		收集的数据	涉及的应用程序及处于争用状态的锁定。
		生成事件数据的时间	死锁检测
	DEADLOCKS WITH DETAILS[2]	*此事件监视器的用途*	用于确定死锁的发生时间以及涉及的应用程序。
		收集的数据	有关涉及的应用程序的综合信息，包括参与语句（和语句文本）的标识和要挂起的锁定的列表。如果使用 DEADLOCKS WITH DETAILS 事件监视器而不是 DEADLOCKS 事件监视器，那么会导致发生死锁时性能下降，原因是收集了其他的信息。
		生成事件数据的时间	死锁检测
	DEADLOCKS WITH DETAILS HISTORY[2]	*此事件监视器的用途*	用于确定死锁的发生时间以及涉及的应用程序。
		收集的数据	DEADLOCKS WITH DETAILS 事件监视器中报告的所有信息以及每个应用程序的当前工作单元的语句历史记录，这些应用程序拥有的锁定参与了挂起该锁定的数据库分区的死锁方案。如果使用 DEADLOCKS WITH DETAILS HISTORY 事件监视器，那么会导致激活时性能轻微下降，原因是进行了语句历史记录跟踪。
		生成事件数据的时间	死锁检测
	DEADLOCKS WITH DETAILS HISTORY VALUES[2]	*此事件监视器的用途*	
		收集的数据	带有详细信息的死锁历史记录中报告的所有信息，以及在执行语句时对所有参数标记提供的值。如果使用 DEADLOCKS WITH DETAILS HISTORY VALUES 事件监视器，那么会导致激活时性能较为严重的下降，原因是额外复制数据值。
		生成事件数据的时间	死锁检测

表 4. 不推荐使用的事件监视器的事件类型 (续)

要监视的事件的类型	事件监视器名称	事件监视器属性	详细信息
工作单元（事务）完成	TRANSACTIONS[3]	*此事件监视器的用途*	
		收集的数据	UOW 工作启动或停止时间、先前的 UOW 时间、耗用的 CPU 以及锁定和记录度量值。如果使用 XA 运行，那么不会生成事务记录。
		生成事件数据的时间	工作单元完成时[1]

注意:

1. 除了自动发生数据收集的已定义时间外，还可使用 FLUSH EVENT MONITOR SQL 语句来生成事件。此方法生成的事件将使用所有监视器类型（DEADLOCKS 和 DEADLOCKS WITH DETAILS 除外）的当前数据库监视器值写入，这些监视器类型与清空的事件监视器相关联。
2. 建议不要使用此事件监视器。建议不要再使用此选项，将来的发行版可能会将其除去。请使用 CREATE EVENT MONITOR FOR LOCKING 语句来监视与锁定相关的事件，例如锁定超时、锁定等待和死锁。
3. 建议不要使用此事件监视器。建议不要再使用此选项，将来的发行版可能会将其除去。请使用 CREATE EVENT MONITOR FOR UNIT OF WORK 语句来监视事务事件。

注：将为每个新创建的数据库创建详细的死锁事件监视器。此事件监视器称为 DB2DETAILDEADLOCK，将在激活数据库时启动，并且写至数据库目录中的文件。可通过删除此事件监视器来避免它需要的额外处理器时间。建议不要使用 DB2DETAILDEADLOCK 事件监视器。建议不要再使用此选项，将来的发行版可能会将其除去。请使用 CREATE EVENT MONITOR FOR LOCKING 语句来监视与锁定相关的事件，例如锁定超时、锁定等待和死锁。

使用事件监视器

通常，对于所有事件监视器类型，创建和使用事件监视器以在某些事件发生时捕获有关系统的信息的过程都很相似。首先，创建事件监视器，然后启用数据收集，最后访问收集的数据。

关于此任务

本主题概述使用事件监视器时要遵循的常规步骤。

过程

要使用事件监视器来捕获事件信息，请执行以下操作:

1. 创建事件监视器。 要创建事件监视器，请使用适当版本的 CREATE EVENT MONITOR 语句。创建事件监视器时，必须选择如何记录该事件监视器收集的数据。所有事件监视器都可将其输出写至关系表；但是，根据您的特定用途，有一些不同选项可能更为适合。
2. 激活事件监视器。 要激活事件监视器，请使用 SET EVENT MONITOR STATE 语句。 例如，对于名为 **capturestats** 的事件监视器，请使用以下命令:

   ```
   SET EVENT MONITOR capturestats STATE 1
   ```

   要关闭事件监视器进行的数据收集，请使用以下语句:

   ```
   SET EVENT MONITOR capturestats STATE 0
   ```

缺省情况下，一些事件监视器在数据库激活时自动激活；其他事件监视器需要您手动激活。但是，使用 AUTOSTART 选项创建的事件监视器直到下次数据库激活时才会自动激活。使用 SET EVENT MONITOR STATE 语句可强制最新创建的事件监视器进入活动状态。要确定事件监视器是否自动启动，请参阅相关 CREATE EVENT MONITOR 语句的参考信息。

3. 启用数据收集。（仅适用于 LOCKING、ACTIVITIES、STATISTICS，UNIT OF WORK 和 PACKAGE CACHE 事件监视器） 启用数据收集包括配置数据库管理器以收集事件监视器记录的特定类型的数据。

   并非所有事件监视器都需要启用数据收集；对于不需要启用数据收集的事件监视器（例如，TABLE 事件监视器），创建并激活它们足以收集数据。阈值违例事件监视器也会自动启动数据收集；但是，在此情况下，还必须使用 CREATE THRESHOLD 语句来定义要针对其捕获数据的阈值。

   对于需要启用数据收集的那些事件监视器，系统为您提供了不同选项。根据要使用的事件监视器的类型，可设置数据库配置参数以在整个数据库中启用数据收集。或者，可选择对特定工作负载对象类型启用特定数据种类的收集。 例如，要配置工作单元事件监视器在系统中的任何工作单元完成时收集基本信息，可将 **mon_uow_data** 参数设置为 BASE。或者，要仅对特定工作负载捕获工作单元信息，可在 CREATE WORKLOAD 或 ALTER WORKLOAD 语句中指定 COLLECT UNIT OF WORK DATA BASE 子句。
4. 运行应用程序或查询。 事件监视器已创建并激活，并且您已启用数据收集后，运行您要为其收集数据的应用程序或查询。
5. 可选: 取消激活事件监视器。 运行您要为其收集数据的应用程序或查询后，可使用 SET EVENT MONITOR STATE 语句来取消激活事件监视器。（请参阅步骤 第 29 页的 2）。不必在执行下一步骤前取消激活事件监视器，但让该事件监视器保留活动状态将导致磁盘空间被您可能没兴趣查看的数据占用。
6. 检查事件监视器收集的数据。 根据事件监视器创建的输出的类型，有不同选项可用于访问所收集数据。如果该数据直接写至关系表，那么可使用 SQL 来访问表列中包含的数据。另一方面，如果该事件监视器写至无格式事件 (UE) 表，那么必须使用 **db2evmonfmt** 之类的命令或 EVMON_FORMAT_UE_TO_TABLES 之类的过程对 UE 表执行后处理，才能查看事件数据。
7. 可选: 从事件监视器表中剪除不再需要的数据。 对于您日常使用的事件监视器，您可能想要从表中剪除不需要的数据。例如，如果使用工作单元事件监视器来生成有关不同应用程序使用的系统资源的日常财务报告，那么您可能想要在报告生成后立即从事件监视器表中删除当前日期的数据。

   提示: 如果需要定期剪除事件监视器输出，请考虑使用无格式事件 (UE) 表来记录事件监视器输出。从 DB2 V10.1 开始，可在数据传输至常规表后自动剪除 UE 表。

创建事件监视器

通过在 CREATE EVENT MONITOR 语句上使用变体来创建不同类型的事件监视器。可对该语句使用选项来指定事件监视器收集的数据类型以及这些事件监视器生成其输出的方式。下面的几节描述不同输出选项以及如何创建生成这些输出类型的事件监视器。

开始之前

在创建事件监视器之前，了解对事件监视器可生成的输出的不同选项很重要。大部分事件监视器可生成至少两种格式的输出；某些事件监视器允许您选择最多四种格式。

过程

要创建事件监视器，请执行以下操作:

1. 确定所需的事件监视器种类。
2. 决定要从事件监视器捕获哪种类型的输出？您想要将数据写至常规表、无格式事件表、文件还是管道？
3. 发出 CREATE EVENT MONITOR 语句。
4. 可选: 如果您创建的事件监视器类型需要激活，请通过发出 SET EVENT MONITOR STATE 语句将其激活。

事件监视器的输出选项

事件监视器可报告它们通过多种方式收集的数据。所有事件监视器都可将它们收集的数据写至表；某些会写至无格式事件 (UE) 表，这可帮助改进性能。其他数据还可直接写至文件或命名管道。

根据您想如何使用事件监视器收集的信息以及事件监视器的类型，可选择以不同方式生成事件监视器收集的输出。可用的输出类型包括:

常规表 从 DB2 V10.1 开始，所有事件监视器都可写至可直接使用 SQL 查询的常规表。对于给定事件，为该事件收集的每个监视元素或度量值都会写至表中它自己的列。这使得能够使用 SELECT 语句来查询输出以检查特定监视元素的值。

要创建写至表的事件监视器，请在 CREATE EVENT MONITOR 语句中指定 WRITE TO TABLE 子句。根据事件监视器，系统会创建一个或多个表来包含输出，每个表包含属于单个逻辑组的监视元素。有关为每个逻辑组生成的特定表的详细信息，请参阅第 81 页的『管理目标表、控制表和事件监视器表』。

表可存储在您选择的表空间中；但是，CREATE EVENT MONITOR 语句的目标表必须为非分区表。

注: 有两种类型的事件监视器写至表。第一个类型包括在 V9.7 及更高发行版中创建的事件监视器。它们包括工作单元事件监视器、程序包高速缓存事件监视器、锁定事件监视器和变更历史记录事件监视器。从 DB2 V10.1 开始，这些事件监视器中的前三个可将其输出写至常规表（作为 UE 表的替代）。变更历史记录事件监视器仅写至常规表。

第二种类型是 DB2 V9.7 之前实现的事件监视器。它们包括所有其他事件监视器。

通常，在创建任一类型的事件监视器后，它们的工作方式大致一样。即，可使用 SQL 直接访问它们生成的表中的数据。但是，第二个类别中的较旧事件监视器有其他选项，可在创建事件监视器时指定这些选项。此外，只有第二个类别中的事件监视器能够同时写至文件和命名管道。

无格式事件 (UE) 表

DB2 V9.7 中为该发行版中添加的新事件监视器引入了 UE 表。UE 表是关系表，但是，它们只有有限数目的列。与每个事件相关联的大部分数据会写至包

含直接插入二进制 (BLOB) 对象的列。以二进制格式写入事件数据可缩短将每个记录写至表时所花的时间。由于此原因，事件监视器性能很重要（高 I/O 或 CPU 受限系统上可能会出现此情况）时 UE 表特别有用。

但是，因为事件数据以二进制格式写入，所以不能使用 SQL 来抽取合格数据。必须对 UE 表执行后处理来抽取以二进制格式存储的数据。使用 UE 表的另一个优点是，您可在后处理期间自动剪除 UE 表数据。EVMON_FORMAT_UE_TO_TABLES 过程有一个选项用于在成功抽取数据后从 UE 表中删除该数据。

要创建写至无格式事件表的事件监视器，请在 CREATE EVENT MONITOR 语句中指定 WRITE TO UNFORMATTED EVENT TABLE 子句。仅对每个事件监视器创建一个 UE 表。

文件 某些事件监视器支持将其输出直接发送至文件系统维护的文件。如果不希望事件监视器输出成为数据库内要管理时导致的额外处理时间，或者您想要在数据库脱机时查看数据，那么此类型的输出很有用。要创建写至文件的事件监视器，请在 CREATE EVENT MONITOR 语句中指定 WRITE TO FILE 子句。

命名管道

如果想要生成应用程序进程事件数据时立即获得该数据，请使用命名管道事件监视器。这些类型的事件监视器直接将其输出发送至命名管道，所以另一应用程序可立即使用该数据。如果需要实时处理事件数据，那么这可能很有用。

要创建写至命名管道的事件监视器，请在 CREATE EVENT MONITOR 语句中指定 WRITE TO PIPE 子句。

根据您的需要，一种类型的事件监视器输出可能比另一种更适用。表 5提供何时特定输出类型特别有用的摘要。

表 5. 不同事件监视器输出类型的摘要

输出类型	此输出类型很有用的场景
常规表	• 要在稍后时间点检查监视数据时 • 在未达到 CPU、日志文件或磁盘存储器的最大容量的系统中 • 期望使用 SQL 即时访问数据时
无格式事件 (UE) 表	• 要在稍后时间点检查监视数据时 • 在优先考虑事件监视性能的系统中或 CPU、日志文件或磁盘使用情况存在限制时 • 对数据执行后处理这一添加步骤并非问题时
文件	• 在您不想或不必管理数据库中的监视数据的系统中。（除去了日志记录、插入和维护一致性的额外处理时间） • 想要在要监视的数据库外部存储数据时 • 要在稍后时间点脱机检查数据时
管道	• 将事件数据传送至立即处理该数据的应用程序。 • 不必在稍后时间点访问事件数据时。

并非所有事件监视器支持所有输出类型。例如，只有工作单元事件监视器、程序包高速缓存事件监视器和锁定事件监视器才能生成 UE 表。第 33 页的表 6显示哪些输出选项可供不同类型的事件监视器使用:

表 6. 事件监视器的输出选项

事件监视器类型	常规表	无格式事件表	文件	命名管道
活动	是		是	是
缓冲池	是		是	是
变更历史记录	是			
连接	是		是	是
数据库	是		是	是
死锁 *（所有变体）	是		是	是
锁定	是	是		
程序包高速缓存	是	是		
语句	是		是	是
统计信息	是		是	是
表空间	是		是	是
表	是		是	是
阈值违例	是		是	是
事务*	是		是	是
工作单元	是	是		
* 建议不要使用的事件监视器。				

写至表的事件监视器

从 DB2 V10.1 开始，所有事件监视器都可将输出写至可直接使用 SQL 查询的常规表。

此外，从 DB2 V10.1 开始，可使用过程 EVMON_UPGRADE_TABLES 来升级由之前发行版中的事件监视器生成的表。此功能使您能更轻松地在升级 DB2 产品时保留事件监视器数据。

创建写至表的事件监视器:

要创建事件监视器，请使用 CREATE EVENT MONITOR STATEMENT。根据您计划监视的事件类型，此语句有不同形式可供您使用。

开始之前

- 您需要 SQLADM 或 DBADM 权限才能创建表事件监视器。
- CREATE EVENT MONITOR 语句的目标表 - 即，事件监视器要将其输出写至的表 - 必须是非分区表。

关于此任务

表事件监视器的各种选项将在 CREATE EVENT MONITOR 语句中设置。要进一步获取对写至表事件监视器生成 CREATE EVENT MONITOR SQL 语句的帮助，可使用 **db2evtbl** 命令。只需要提供事件监视器的名称和需要的事件类型，就会生成 CREATE EVENT MONITOR 语句，完成所有目标表列表。然后，您可复制所生成语句，进行修改，然后通过命令行处理器执行该语句。

过程

要创建将其输出写至常规表的事件监视器，请执行以下步骤:

1. 构造 CREATE EVENT MONITOR 语句并使用 WRITE TO TABLE 子句来指示要在一个表（或一组表）中收集事件监视器数据。

   ```
   CREATE EVENT MONITOR evmon-name FOR eventtype
      WRITE TO TABLE
   ```

   其中 evmon-name 是事件监视器的名称，eventtype 是下列其中一个值:
   - ACTIVITIES
   - BUFFERPOOLS
   - CHANGE HISTORY
   - CONNECTIONS
   - DATABASE
   - DEADLOCKS
   - LOCKING
   - PACKAGE CACHE
   - STATEMENTS
   - STATISTICS
   - TABLE
   - TABLESPACE
   - THRESHOLD VIOLATIONS
   - TRANSACTIONS
   - UNIT OF WORK

   例如，要创建名为 myevmon 的工作单元事件监视器，请使用类似如下的语句:

   ```
   CREATE EVENT MONITOR myevmon FOR UNIT OF WORK
      WRITE TO TABLE
   ```

   以上语句创建工作单元事件监视器，该事件监视器对所收集的监视元素的逻辑组、对应输出表名和这些表的目标表空间使用缺省值。有关这些缺省值的更多信息，请参阅适当 CREATE EVENT MONITOR 语句的文档。
2. 可选: 指定要对其收集数据的逻辑组。 缺省情况下，会针对该事件监视器类型的所有逻辑数据组收集事件数据。（有关详细信息，请参阅第 81 页的『管理目标表、控制表和事件监视器表』。）如果仅需要收集所选逻辑组的数据，那么可在 CREATE EVENT MONITOR 语句中指定要包括的逻辑组的名称。例如，对于锁定事件监视器，您可能想要仅收集与 LOCK 和 PARTICIPANT 逻辑组相关联的信息。要仅包括这些逻辑组，可使用类似如下的语句:

   ```
   CREATE EVENT MONITOR mylocks FOR LOCKING
      WRITE TO TABLE
      LOCK, PARTICIPANTS
   ```
3. 可选: 指定要用于输出表的表名。 除非您另行指定，否则会对监视元素的每个逻辑组的表使用缺省名称。使用的缺省名称是通过将逻辑组名与事件监视器名并置派生的。例如，对于上一步中的语句创建的锁定事件监视器，所生成表的非限定名称为 LOCK_MYLOCKS 和 PARTICIPANTS_MYLOCKS。要覆盖缺省名称，请在指定逻辑组时包括要使用的表名:

   ```
   CREATE EVENT MONITOR mylocks FOR LOCKING
      WRITE TO TABLE
      LOCK(TABLE LOCKDATA), PARTICIPANTS(TABLE PARTICIP)
   ```

在上一示例中，用于 LOCK 和 PARTICIPANTS 逻辑组的表的名称为 LOCKDATA_MYLOCKS 和 PARTICIP_MYLOCKS。

还可通过包括要使用的表空间的名称来覆盖要用于每个表的表空间:

```
CREATE EVENT MONITOR mylocks FOR LOCKING
   WRITE TO TABLE
   LOCK(TABLE LOCKDATA IN EVMONSPACE), PARTICIPANTS(TABLE PARTICIP IN EVMONSPACE)
```

在上一示例中，同时将 EVMONSPACE 表空间用于这两个输出表。

其他选项

不同事件监视器提供不同配置选项。有关特定类型的事件监视器可用的选项的详细信息，请参阅对应要使用的事件监视器类型的 CREATE EVENT MONITOR 语句的文档。下面的示例显示您可针对不同事件监视器选择的一些配置选项:

使用单个事件监视器来捕获多个事件类型

对于某些类型[2]的事件监视器，可使用单个事件监视器来捕获不同类型的事件。如果要使用此事件监视器捕获多个类型的事件，请对 eventtype 指定用逗号分隔的附加值。例如，您可能想要在单个事件监视器中组合缓冲池监视和表空间监视:

```
CREATE EVENT MONITOR myevmon FOR BUFFERPOOLS, TABLESPACES
                     WRITE TO TABLE
```

此事件监视器将监视 BUFFERPOOL 和 TABLESPACE 事件类型。假定用户 dbadmin 发出了以上列出的语句，那么目标表的派生名称和表空间如下所示:

- DBADMIN.BUFFERPOOL_MYEVMON
- DBADMIN.TABLESPACE_MYEVMON
- DBADMIN.CONTROL_,MYEVMON

调整事件监视器输出缓冲区的大小

可通过调整 BUFFERSIZE 值来改变某些类型[2]的事件监视器的表事件监视器缓冲区的大小（以 4K 页计）。例如，在以下语句中:

```
CREATE EVENT MONITOR myevmon FOR BUFFERPOOLS, TABLESPACES
                     WRITE TO TABLE BUFFERSIZE 8
```

8 是两个事件表缓冲区的组合容量（以 4K 页计）。这加起来高达 32K 缓冲区空间，每个缓冲区 16K。

每个缓冲区的缺省大小为 4 页（分配两个 16K 缓冲区）。最小大小为 1 页。由于缓冲区是根据监视器堆进行分配，所以缓冲区的最大大小受该监视器堆大小限制。由于性能原因，高可用事件监视器的缓冲区应该比不活动事件监视器的缓冲区大。

控制事件监视器输出是已分块还是未分块

某些事件监视器[2]允许您控制事件监视器输出缓冲区变满时如何处理。对于分块事件监视器，如果事件缓冲区已满，那么生成事件的每个代理程序将等待事件缓冲区写至表。这可能会导致数据库性能降低，原因是暂挂的代理程序和所有从属代理程序在缓冲区清空之前不能运行。使用 BLOCKED 子句来确保事件数据不会丢失:

---

2. BUFFERPOOLS、CONNECTIONS、DATABASE、DEADLOCKS、STATEMENTS、TABLES 和 TABLESPACES 支持此选项。

```
CREATE EVENT MONITOR myevmon FOR BUFFERPOOLS, TABLESPACES
                     WRITE TO TABLE BUFFERSIZE 8 BLOCKED
```

如果数据库性能比收集每个事件记录更重要，那么使用非分块事件监视器。在此情况下，如果事件缓冲区已满，那么生成事件的每个代理程序将不会等待事件缓冲区写至表。因此，非分块事件监视器可能会导致活动频繁的系统上的数据丢失。使用 NONBLOCKED 子句来使事件监视导致的额外处理时间降至最少:

```
CREATE EVENT MONITOR myevmon FOR BUFFERPOOLS, TABLESPACES
                     WRITE TO TABLE BUFFERSIZE 8 NONBLOCKED
```

注: 要了解如何将有关已废弃事件的信息写入事件监视器的控制表的更多信息，请参阅第 81 页的『管理目标表、控制表和事件监视器表』和第 97 页的『"写至表"和文件事件监视器缓存』。

控制对其收集数据的监视元素

要对其收集数据的监视元素。如果您仅关心几个监视元素，那么可通过在 CREATE EVENT MONITOR 语句中指定元素名称来对某些事件监视器[2]指定要收集的监视元素:

```
CREATE EVENT MONITOR myevmon FOR DATABASE, BUFFERPOOLS, TABLESPACES
              WRITE TO TABLE DB, DBMEMUSE,
              BUFFERPOOL (EXCLUDES(db_path, files_closed)),
              TABLESPACE (INCLUDES
                (tablespace_name, direct_reads, direct_writes))
              BUFFERSIZE 8 NONBLOCKED
```

将捕获 DB 和 DBMEMUSE 逻辑数据组的所有监视元素（这是缺省行为）。对于 BUFFERPOOL，会捕获除 **db_path** 和 **files_closed** 以外的所有监视元素。 最后，对于 TABLESPACE，仅捕获 **tablespace_name**、**direct_reads** 和 **direct_writes** 监视元素。

设置根据所使用表空间取消激活事件监视器的阈值

所有事件监视器都提供了一个选项，用于指定事件监视器自动取消激活前表空间可达到的充满程度:

```
CREATE EVENT MONITOR myevmon FOR BUFFERPOOLS, TABLESPACES
                     PCTDEACTIVATE 90
```

如果表空间已使用 90% 的容量，那么 myevmon 事件监视器会自动关闭。PCTDEACTIVATE 子句只能用于 DMS 表空间。如果目标表空间已启用自动调整大小，那么将 PCTDEACTIVATE 子句设置为 100。

下一步做什么

缺省情况下，V9.7 或更高版本中引入的事件监视器会创建为 AUTOSTART 事件监视器。数据库下次激活时以及此后后续数据库激活时，它们会自动激活。如果要立即激活该事件监视器，请在下一次数据库激活之前使用 SET EVENT MONITOR STATE 语句来手动启动该事件监视器。此外，对于锁定事件监视器、工作单元事件监视器和程序包高速缓存事件监视器中的每一个，还必须启用数据收集。

事件监视器逻辑数据组和监视元素:

一起检查时通常很有用的监视元素被组合到*逻辑数据组*中。

所有事件监视器按一种方式或另一种方式使用逻辑数据组。对于某些事件监视器类型，可通过指定要记录其信息的逻辑数据组来指定要收集什么信息。逻辑数据组还用于将数据组合到事件监视器生成的输出中；例如，写至表的事件监视器通常为每个监视元素逻辑数据组创建一个表。

下表列示事件监视可能返回的逻辑数据分组和监视元素。

- 第 65 页的『lock 逻辑数据组』
- 第 66 页的『lock_participants 逻辑数据组』
- 第 65 页的『lock_participant_activities 逻辑数据组』
- 第 65 页的『lock_activity_values 逻辑数据组』
- 第 68 页的『pkgcache 逻辑数据组』
- 第 69 页的『pkgcache_metrics 逻辑数据组』
- 第 72 页的『pkgcache_stmt_args 逻辑数据组』
- 第 72 页的『regvar 逻辑数据组』
- 第 72 页的『sqlca 逻辑数据组』
- 第 73 页的『txncompletion 逻辑数据组』
- 第 73 页的『uow 逻辑数据组』
- 第 75 页的『uow_metrics 逻辑数据组』
- 第 79 页的『uow_package_list 逻辑数据组』
- 第 74 页的『uow_executable_list 逻辑数据组』
- 第 80 页的『utillocation 逻辑数据组』
- 第 80 页的『utilphase 逻辑数据组』
- 第 80 页的『utilstart 逻辑数据组』
- 第 81 页的『utilstop 逻辑数据组』

changesummary 逻辑数据组

第 754 页的『event_id -“事件标识”监视元素』
第 756 页的『event_type -“事件类型”监视元素』
第 755 页的『event_timestamp -“事件时间戳记”监视元素』
第 896 页的『member -“数据库成员”监视元素』
第 695 页的『coord_member -“协调程序成员”监视元素』
第 1328 页的『utility_invocation_id -“实用程序调用标识”』
第 1333 页的『utility_type -“实用程序类型”』
第 616 页的『appl_id -“应用程序标识”监视元素』
第 620 页的『appl_name -“应用程序名称”监视元素』
第 625 页的『application_handle -“应用程序句柄”监视元素』
第 1188 页的『system_auth_id -“系统授权标识”监视元素』
第 1137 页的『session_auth_id -“会话授权标识”监视元素』
第 662 页的『client_platform -“客户机操作平台”监视元素』
第 664 页的『client_protocol -“客户机通信协议”监视元素』
第 662 页的『client_port_number -“客户机端口号”监视元素』
第 661 页的『client_pid -“客户机进程标识”监视元素』
第 659 页的『client_hostname -“客户机主机名”监视元素』
第 665 页的『client_wrkstnname -“客户机工作站名称”监视元素』
第 657 页的『client_acctng -“客户机记帐字符串”监视元素』
第 664 页的『client_userid -“客户机用户标识”监视元素』

第 658 页的『client_applname -“客户机应用程序名称”监视元素』

第 638 页的『backup_timestamp -“备份时间戳记”』

dbdbmcfg 逻辑数据组

第 754 页的『event_id -“事件标识”监视元素』

第 755 页的『event_timestamp -“事件时间戳记”监视元素』

第 896 页的『member -“数据库成员”监视元素』

第 756 页的『event_type -“事件类型”监视元素』

第 653 页的『cfg_name -“配置名称”』

第 654 页的『cfg_value -“配置值”』

第 654 页的『cfg_value_flags -“配置值标志”』

第 653 页的『cfg_old_value -“配置旧值”』

第 654 页的『cfg_old_value_flags -“配置旧值标志”』

第 652 页的『cfg_collection_type -“配置收集类型”』

第 730 页的『deferred -“延迟”』

ddlstmtexec 逻辑数据组

第 754 页的『event_id -“事件标识”监视元素』

第 755 页的『event_timestamp -“事件时间戳记”监视元素』

第 896 页的『member -“数据库成员”监视元素』

第 756 页的『event_type -“事件类型”监视元素』

第 794 页的『global_transaction_id -“全局事务标识”监视元素』

第 841 页的『local_transaction_id -“本地事务标识”监视元素』

第 1125 页的『savepoint_id -“保存点标识”』

第 1316 页的『uow_id -“工作单元标识”监视元素』

第 725 页的『ddl_classification -“DDL 分类”』

第 1178 页的『stmt_text -“SQL 语句文本”监视元素』

dllock 逻辑数据组

第 713 页的『data_partition_id -“数据分区标识”监视元素』

第 842 页的『lock_attributes -“锁定属性”监视元素』

第 843 页的『lock_count -“锁定计数”监视元素』

第 844 页的『lock_current_mode -“转换前的原始锁定方式”监视元素』

第 845 页的『lock_escalation -“锁定升级”监视元素』

第 851 页的『lock_hold_count -“锁定挂起计数”监视元素』

第 852 页的『lock_mode -“锁定方式”监视元素』

第 854 页的『lock_name -“锁定名称”监视元素』

第 855 页的『lock_object_name -“锁定对象名称”』

第 856 页的『lock_object_type -“等待的锁定对象类型”监视元素』

第 858 页的『lock_release_flags -“锁定释放标志”监视元素』

第 858 页的『lock_status -“锁定状态”监视元素』

第 908 页的『node_number -“节点号”』

第 1190 页的『table_file_id -“表文件标识”监视元素』

第 1191 页的『table_name -“表名”监视元素』

第 1193 页的『table_schema -“表模式名”监视元素』

第 1200 页的『tablespace_name -“表空间名称”监视元素』

注: 此逻辑数据组的底层实现是快照监视器 LOCK 逻辑数据组。如果在用于文件和管道输出选项的自描述流中检查此逻辑组的输出，那么可见到 LOCK 组用于生成该输出。

event_activity 逻辑数据组

第 591 页的『act_exec_time -“活动执行时间”监视元素』

第 595 页的『activate_timestamp -“激活时间戳记”监视元素』

第 597 页的『activity_id -“活动标识”监视元素』

第 597 页的『activity_secondary_id -“活动辅助标识”监视元素』

第 598 页的『activity_type -“活动类型”监视元素』

第 600 页的『address - 从中发起连接的 IP 地址』

第 601 页的『agent_id -“应用程序句柄（代理程序标识）”监视元素』

第 616 页的『appl_id -“应用程序标识”监视元素』

第 620 页的『appl_name -“应用程序名称”监视元素』

第 626 页的『arm_correlator -“应用程序响应测量相关因子”监视元素』

第 696 页的『coord_partition_num -“协调程序分区号”监视元素』

第 722 页的『db_work_action_set_id -“数据库工作操作集标识”监视元素』

第 723 页的『db_work_class_id -“数据库工作类标识”监视元素』

details_xml（此 XML 文档是一个 activity_metrics 类型的度量值文档，如 XML 模式文档 `sqllib/misc/DB2MonCommon.xsd` 中所述。）还可通过 event_activitymetrics 逻辑数据组访问此文档中报告的度量值。

第 827 页的『intra_parallel_state -“分区内并行性的当前状态”监视元素』

第 919 页的『num_remaps -“重新映射次数”监视元素』

第 948 页的『parent_activity_id -“父活动标识”监视元素』

第 948 页的『parent_uow_id -“父工作单元标识”监视元素』

第 949 页的『partial_record -“部分记录”监视元素』

第 982 页的『pool_data_l_reads -“缓冲池数据逻辑读取数”监视元素』

第 984 页的『pool_data_p_reads -“缓冲池数据物理读取数”监视元素』

第 1091 页的『query_actual_degree -“实际运行时分区内并行度”监视元素』

第 1125 页的『sc_work_action_set_id -“服务类工作操作集标识”监视元素』

第 1125 页的『sc_work_class_id -“服务类工作类标识”监视元素』

第 1127 页的『section_actuals -“部分实际值”监视元素』

第 1135 页的『service_subclass_name -“服务子类名”监视元素』

第 1136 页的『service_superclass_name -“服务超类名”监视元素』

第 1137 页的『session_auth_id -“会话授权标识”监视元素』

第 1149 页的『sort_overflows -“排序溢出数”监视元素』

第 1154 页的『sqlca -“SQL 通信区（SQLCA）”』
第 1228 页的『time_completed -“完成时间”监视元素』
第 1228 页的『time_created -“创建时间”监视元素』
第 1229 页的『time_started -“开始时间”监视元素』
第 1291 页的『total_sort_time -“排序时间总计”监视元素』
第 1292 页的『total_sorts -“排序总数”监视元素』
第 1303 页的『tpmon_acc_str -“TP 监视器客户机记帐字符串”监视元素』
第 1304 页的『tpmon_client_app -“TP 监视器客户机应用程序名称”监视元素』
第 1304 页的『tpmon_client_userid -“TP 监视器客户机用户标识”监视元素』
第 1305 页的『tpmon_client_wkstn -“TP 监视器客户机工作站名称”监视元素』
第 1316 页的『uow_id -“工作单元标识”监视元素』
第 1341 页的『workload_id -“工作负载标识”监视元素』
第 1343 页的『workload_occurrence_id -“工作负载项标识”监视元素』
第 1012 页的『pool_index_l_reads -“缓冲池索引逻辑读取数”监视元素』
第 1014 页的『pool_index_p_reads -“缓冲池索引物理读取数”监视元素』
第 1046 页的『pool_temp_data_l_reads -“缓冲池临时数据逻辑读取数”监视元素』
第 1048 页的『pool_temp_data_p_reads -“缓冲池临时数据物理读取数”监视元素』
第 1050 页的『pool_temp_index_l_reads -“缓冲池临时索引逻辑读取数”监视元素』
第 1052 页的『pool_temp_index_p_reads -“缓冲池临时索引物理读取数”监视元素』
第 1053 页的『pool_temp_xda_l_reads -“缓冲池临时 XDA 数据逻辑读取数”监视元素』
第 1055 页的『pool_temp_xda_p_reads -“缓冲池临时 XDA 数据物理读取数”监视元素』
第 1065 页的『pool_xda_l_reads -“缓冲池 XDA 数据逻辑读取数”监视元素』
第 1069 页的『pool_xda_p_reads -“缓冲池 XDA 数据物理读取数”监视元素』
第 1083 页的『prep_time -“编译时间”监视元素』
第 1091 页的『query_card_estimate -“行查询数估计”』
第 1092 页的『query_cost_estimate -“查询估算成本”监视元素』
第 1116 页的『rows_fetched -“访存的行数”监视元素』
第 1117 页的『rows_modified -“修改的行数”监视元素』
第 1120 页的『rows_returned -“返回的行数”监视元素』
第 1189 页的『system_cpu_time -“系统 CPU 时间”监视元素』
第 1326 页的『user_cpu_time -“用户 CPU 时间”监视元素』
第 1335 页的『wl_work_action_set_id -“工作负载工作操作集标识”监视元素』
第 1336 页的『wl_work_class_id -“工作负载工作类标识”监视元素』
第 905 页的『mon_interval_id -“监视时间间隔标识”监视元素』
第 896 页的『member -“数据库成员”监视元素』
第 1093 页的『query_data_tag_list -“估算的查询数据标记列表”监视元素』
第 1294 页的『total_stats_fabrication_time -“统计信息生成时间总计”监视元素』
第 1295 页的『total_stats_fabrications -“统计信息生成总计”监视元素』
第 1299 页的『total_sync_runstats -“同步 RUNSTATS 活动总数”监视元素』

第 1296 页的『total_sync_runstats_time -“同步 RUNSTATS 时间总计”监视元素』

event_activitymetrics 逻辑数据组

第 628 页的『audit_events_total -“审计事件总数”监视元素』
第 631 页的『audit_file_writes_total -“写审计文件总次数”监视元素』
第 632 页的『audit_subsystem_wait_time -“审计子系统等待时间”监视元素』
第 634 页的『audit_subsystem_waits_total -“审计子系统等待总次数”监视元素』
第 696 页的『coord_stmt_exec_time -“协调代理程序执行语句的时间”监视元素』
第 728 页的『deadlocks -“检测到的死锁数”监视元素』
第 732 页的『diaglog_write_wait_time -“诊断日志文件写等待时间”监视元素』
第 734 页的『diaglog_writes_total -“写诊断日志文件总次数”监视元素』
第 735 页的『direct_read_reqs -“直接读请求数”监视元素』
第 737 页的『direct_read_time -“直接读时间”监视元素』
第 738 页的『direct_reads -“直接读数据库数目”监视元素』
第 740 页的『direct_write_reqs -“直接写请求数”监视元素』
第 742 页的『direct_write_time -“直接写时间”监视元素』
第 744 页的『direct_writes -“直接写数据库数目”监视元素』
第 766 页的『fcm_message_recv_volume -“接收 FCM 消息量”监视元素』
第 768 页的『fcm_message_recv_wait_time -“接收 FCM 消息等待时间”监视元素』
第 769 页的『fcm_message_recvs_total -“接收 FCM 消息总数”监视元素』
第 770 页的『fcm_message_send_volume -“发送 FCM 消息量”监视元素』
第 772 页的『fcm_message_send_wait_time -“发送 FCM 消息等待时间”监视元素』
第 773 页的『fcm_message_sends_total -“发送 FCM 消息总数”监视元素』
第 774 页的『fcm_recv_volume -“FCM 接收量”监视元素』
第 775 页的『fcm_recv_wait_time -“FCM 接收等待时间”监视元素』
第 777 页的『fcm_recvs_total -“FCM 接收总计”监视元素』
第 778 页的『fcm_send_volume -“FCM 发送量”监视元素』
第 779 页的『fcm_send_wait_time -“FCM 发送等待时间”监视元素』
第 780 页的『fcm_sends_total -“FCM 发送总计”监视元素』
第 782 页的『fcm_tq_recv_volume -“FCM 表队列接收量”监视元素』
第 783 页的『fcm_tq_recv_wait_time -“FCM 表队列接收等待时间”监视元素』
第 784 页的『fcm_tq_recvs_total -“FCM 表队列接收总量”监视元素』
第 786 页的『fcm_tq_send_volume -“FCM 表队列发送量”监视元素』
第 787 页的『fcm_tq_send_wait_time -“FCM 表队列发送等待时间”监视元素』
第 788 页的『fcm_tq_sends_total -“FCM 表队列发送总次数”监视元素』
第 845 页的『lock_escals -“锁定升级次数”监视元素』
第 860 页的『lock_timeouts -“锁定超时次数”监视元素』
第 863 页的『lock_wait_time -“等待锁定时间”监视元素』
第 868 页的『lock_waits -“等待锁定次数”监视元素』
第 872 页的『log_buffer_wait_time -“日志缓冲区等待时间”监视元素』

第 874 页的『log_disk_wait_time -“日志磁盘等待时间”监视元素』

第 875 页的『log_disk_waits_total -“日志磁盘等待总次数”监视元素』

第 913 页的『num_log_buffer_full -“日志缓冲区变满而导致代理程序等待的次数”监视元素』

第 917 页的『num_lw_thresh_exceeded -“超过锁定等待阈值的次数”监视元素』

第 982 页的『pool_data_l_reads -“缓冲池数据逻辑读取数”监视元素』

第 984 页的『pool_data_p_reads -“缓冲池数据物理读取数”监视元素』

第 986 页的『pool_data_writes -“缓冲池数据写次数”监视元素』

第 1012 页的『pool_index_l_reads -“缓冲池索引逻辑读取数”监视元素』

第 1014 页的『pool_index_p_reads -“缓冲池索引物理读取数”监视元素』

第 1016 页的『pool_index_writes -“缓冲池索引写次数”监视元素』

第 1043 页的『pool_read_time -“缓冲池物理读时间总计”监视元素』

第 1046 页的『pool_temp_data_l_reads -“缓冲池临时数据逻辑读取数”监视元素』

第 1048 页的『pool_temp_data_p_reads -“缓冲池临时数据物理读取数”监视元素』

第 1050 页的『pool_temp_index_l_reads -“缓冲池临时索引逻辑读取数”监视元素』

第 1052 页的『pool_temp_index_p_reads -“缓冲池临时索引物理读取数”监视元素』

第 1053 页的『pool_temp_xda_l_reads -“缓冲池临时 XDA 数据逻辑读取数”监视元素』

第 1055 页的『pool_temp_xda_p_reads -“缓冲池临时 XDA 数据物理读取数”监视元素』

第 1058 页的『pool_write_time -“缓冲池物理写时间总计”监视元素』

第 1065 页的『pool_xda_l_reads -“缓冲池 XDA 数据逻辑读取数”监视元素』

第 1069 页的『pool_xda_p_reads -“缓冲池 XDA 数据物理读取数”监视元素』

第 1071 页的『pool_xda_writes -“缓冲池 XDA 数据写次数”监视元素』

第 1073 页的『post_shrthreshold_sorts -“共享阈值后排序数”监视元素』

第 1079 页的『post_threshold_sorts -“超出阈值后的排序次数”监视元素』

第 1117 页的『rows_modified -“修改的行数”监视元素』

第 1118 页的『rows_read -“读取行数”监视元素』

第 1120 页的『rows_returned -“返回的行数”监视元素』

第 1149 页的『sort_overflows -“排序溢出数”监视元素』

第 1168 页的『stmt_exec_time -“语句执行时间”监视元素』

第 1223 页的『thresh_violations -“阈值违例次数”监视元素』

第 1231 页的『total_act_time -“活动时间总计”监视元素』

第 1232 页的『total_act_wait_time -“活动等待时间总计”监视元素』

第 1235 页的『total_app_section_executions -“应用程序执行部分执行的总次数”监视元素』

第 1249 页的『total_cpu_time -“CPU 时间总计”监视元素』

第 1272 页的『total_routine_invocations -“例程调用总计”监视元素』

第 1274 页的『total_routine_non_sect_proc_time -“非部分处理时间” 监视元素』

第 1274 页的『total_routine_non_sect_time -“非部分例程执行时间” 监视元素』

第 1275 页的『total_routine_time -“例程时间总计”监视元素』

第 1276 页的『total_routine_user_code_proc_time -“例程用户代码处理时间总计”监视元素』

第 1278 页的『total_routine_user_code_time -“例程用户代码时间总计”监视元素』

第 1284 页的『total_section_proc_time -“部分处理时间总计”监视元素』

第 1285 页的『total_section_sort_proc_time -“节排序处理时间总计”监视元素』

第 1287 页的『total_section_sort_time -“节排序时间总计”监视元素』

第 1288 页的『total_section_sorts -“节排序总次数”监视元素』

第 1289 页的『total_section_time -“部分时间总计”监视元素』

第 1292 页的『total_sorts -“排序总数”监视元素』

第 1311 页的『tq_tot_send_spills -“溢出表队列缓冲区总数”监视元素』

第 1336 页的『wlm_queue_assignments_total -“工作负载管理器队列分配总次数”监视元素』

第 1337 页的『wlm_queue_time_total -“工作负载管理器队列时间总计”监视元素』

event_activitystmt 逻辑数据组

第 595 页的『activate_timestamp -“激活时间戳记”监视元素』

第 597 页的『activity_id -“活动标识”监视元素』

第 597 页的『activity_secondary_id -“活动辅助标识”监视元素』

第 616 页的『appl_id -“应用程序标识”监视元素』

第 669 页的『comp_env_desc -“编译环境”监视元素』

第 709 页的『creator -“应用程序创建者”』

第 749 页的『eff_stmt_text -“有效语句文本”监视元素』

第 762 页的『executable_id -“可执行文件标识”监视元素』

第 827 页的『intra_parallel_state -“分区内并行性的当前状态”监视元素』

第 941 页的『package_name -“程序包名”监视元素』

第 943 页的『package_version_id -“程序包版本”监视元素』

第 1091 页的『query_actual_degree -“实际运行时分区内并行度”监视元素』

第 1115 页的『routine_id -“例程标识”监视元素』

第 1128 页的『section_env -“节环境”监视元素』

第 1128 页的『section_number -“节号”监视元素』

第 1169 页的『stmt_first_use_time -“第一次使用语句时的时间戳记”监视元素』

第 1170 页的『stmt_invocation_id -“语句调用标识”监视元素』

第 1171 页的『stmt_isolation -“语句隔离”』

第 1171 页的『stmt_last_use_time -“上一次使用语句时的时间戳记”监视元素』

第 1172 页的『stmt_lock_timeout -“语句锁定超时”监视元素』

第 1172 页的『stmt_nest_level -“语句嵌套级别”监视元素』

第 1174 页的『stmt_pkgcache_id -“语句程序包高速缓存标识”监视元素』

第 1175 页的『stmt_query_id -“语句查询标识”监视元素』

第 1176 页的『stmt_source_id -“语句源标识”』

第 1178 页的『stmt_text -“SQL 语句文本”监视元素』

第 1179 页的『stmt_type -“语句类型”监视元素』

第 1316 页的『uow_id -“工作单元标识”监视元素』

第 896 页的『member -“数据库成员”监视元素』

event_activityvals 逻辑数据组

第 595 页的『activate_timestamp -“激活时间戳记”监视元素』

第 597 页的『activity_id -“活动标识”监视元素』

第 597 页的『activity_secondary_id -“活动辅助标识”监视元素』

第 616 页的『appl_id -“应用程序标识”监视元素』

第 827 页的『intra_parallel_state -“分区内并行性的当前状态”监视元素』

第 1091 页的『query_actual_degree -“实际运行时分区内并行度”监视元素』

第 1181 页的『stmt_value_data -“值数据”』

第 1182 页的『stmt_value_index -“值索引”』

第 1182 页的『stmt_value_isnull -“包含空值”监视元素』

第 1183 页的『stmt_value_isreopt -“用于语句重新优化的变量”监视元素』

第 1184 页的『stmt_value_type -“值类型”监视元素』

第 1316 页的『uow_id -“工作单元标识”监视元素』

第 896 页的『member -“数据库成员”监视元素』

event_bufferpool 逻辑数据组

第 642 页的『bp_id -“缓冲池标识”监视元素』

第 642 页的『bp_name -“缓冲池名称”监视元素』

第 719 页的『db_name - 数据库名称监视元素』

第 720 页的『db_path -“数据库路径”』

第 735 页的『direct_read_reqs -“直接读请求数”监视元素』

第 737 页的『direct_read_time -“直接读时间”监视元素』

第 738 页的『direct_reads -“直接读数据库数目”监视元素』

第 740 页的『direct_write_reqs -“直接写请求数”监视元素』

第 742 页的『direct_write_time -“直接写时间”监视元素』

第 744 页的『direct_writes -“直接写数据库数目”监视元素』

第 755 页的『event_time -“事件时间”』

第 757 页的『evmon_activates -“事件监视器激活数”』

第 763 页的『evmon_flushes -“事件监视器清空数”』

第 790 页的『files_closed -“关闭数据库文件数”监视元素』

第 949 页的『partial_record -“部分记录”监视元素』

第 961 页的『pool_async_data_read_reqs -“缓冲池异步读请求数”监视元素』

第 962 页的『pool_async_data_reads -“缓冲池异步数据读次数”监视元素』

第 963 页的『pool_async_data_writes -“缓冲池异步数据写次数”监视元素』

第 966 页的『pool_async_index_reads -“缓冲池异步索引读取数”监视元素』

第 967 页的『pool_async_index_writes -“缓冲池异步索引写次数”监视元素』

第 968 页的『pool_async_read_time -“缓冲池异步读取时间”』

第 969 页的『pool_async_write_time -“缓冲池异步写入时间”监视元素』

第 982 页的『pool_data_l_reads -“缓冲池数据逻辑读取数”监视元素』
第 984 页的『pool_data_p_reads -“缓冲池数据物理读取数”监视元素』
第 986 页的『pool_data_writes -“缓冲池数据写次数”监视元素』
第 1012 页的『pool_index_l_reads -“缓冲池索引逻辑读取数”监视元素』
第 1014 页的『pool_index_p_reads -“缓冲池索引物理读取数”监视元素』
第 1016 页的『pool_index_writes -“缓冲池索引写次数”监视元素』
第 1043 页的『pool_read_time -“缓冲池物理读时间总计”监视元素』
第 1058 页的『pool_write_time -“缓冲池物理写时间总计”监视元素』
第 639 页的『block_ios -“块 I/O 请求数”监视元素』
第 946 页的『pages_from_block_ios -“块 I/O 读取总页数”监视元素』
第 947 页的『pages_from_vectored_ios -“向量 I/O 读取总页数”监视元素』
第 966 页的『pool_async_index_read_reqs -“缓冲池异步索引读请求数”监视元素』
第 972 页的『pool_async_xda_read_reqs -“缓冲池异步 XDA 读请求数”监视元素』
第 973 页的『pool_async_xda_reads -“缓冲池异步 XDA 数据读取数”监视元素』
第 974 页的『pool_async_xda_writes -“缓冲池异步 XDA 数据写次数”监视元素』
第 1018 页的『pool_no_victim_buffer -“缓冲池无牺牲缓冲区次数”监视元素』
第 1065 页的『pool_xda_l_reads -“缓冲池 XDA 数据逻辑读取数”监视元素』
第 1069 页的『pool_xda_p_reads -“缓冲池 XDA 数据物理读取数”监视元素』
第 1071 页的『pool_xda_writes -“缓冲池 XDA 数据写次数”监视元素』
第 1314 页的『unread_prefetch_pages -“未读取的预取页数”监视元素』
第 1334 页的『vectored_ios -“向量 I/O 请求数”监视元素』

event_conn 逻辑数据组

第 588 页的『acc_curs_blk -“接受的块游标请求数”』
第 601 页的『agent_id -“应用程序句柄（代理程序标识）”监视元素』
第 616 页的『appl_id -“应用程序标识”监视元素』
第 621 页的『appl_priority -“应用程序代理程序优先级”』
第 621 页的『appl_priority_type -“应用程序优先级类型”』
第 622 页的『appl_section_inserts -“节插入数”监视元素』
第 622 页的『appl_section_lookups -“节查询数”』
第 636 页的『authority_bitmap -“用户权限级别”监视元素』
第 636 页的『authority_lvl -“用户权限级别”监视元素』
第 639 页的『binds_precompiles -“尝试的绑定次数/预编译次数”』
第 646 页的『cat_cache_inserts -“目录高速缓存插入数”监视元素』
第 647 页的『cat_cache_lookups -“目录高速缓存查询数”监视元素』
第 649 页的『cat_cache_overflows -“目录高速缓存溢出数”』
第 668 页的『commit_sql_stmts -“尝试的落实语句数”』
第 726 页的『ddl_sql_stmts -“数据定义语言（DDL）SQL 语句数”』
第 728 页的『deadlocks -“检测到的死锁数”监视元素』
第 735 页的『direct_read_reqs -“直接读请求数”监视元素』

第 737 页的『direct_read_time -“直接读时间”监视元素』
第 738 页的『direct_reads -“直接读数据库数目”监视元素』
第 740 页的『direct_write_reqs -“直接写请求数”监视元素』
第 742 页的『direct_write_time -“直接写时间”监视元素』
第 744 页的『direct_writes -“直接写数据库数目”监视元素』
第 747 页的『disconn_time -“数据库释放时间戳记”』
第 748 页的『dynamic_sql_stmts -“尝试的动态 SQL 语句数”』
第 764 页的『failed_sql_stmts -“失败的语句操作”』
第 809 页的『hash_join_overflows -“散列连接溢出数”』
第 809 页的『hash_join_small_overflows -“散列连接小溢出数”』
第 820 页的『int_auto_rebinds -“内部自动重新绑定次数”』
第 821 页的『int_commits -“内部落实数”监视元素』
第 823 页的『int_deadlock_rollbacks -“死锁导致的内部回滚数”』
第 823 页的『int_rollbacks -“内部回滚数”监视元素』
第 825 页的『int_rows_deleted -“删除的内部行数”』
第 826 页的『int_rows_inserted -“插入的内部行数”』
第 826 页的『int_rows_updated -“更新的内部行数”』
第 845 页的『lock_escalation -“锁定升级”监视元素』
第 860 页的『lock_timeouts -“锁定超时次数”监视元素』
第 863 页的『lock_wait_time -“等待锁定时间”监视元素』
第 868 页的『lock_waits -“等待锁定次数”监视元素』
第 933 页的『olap_func_overflows -“OLAP 函数溢出次数”监视元素』
第 949 页的『partial_record -“部分记录”监视元素』
第 826 页的『int_rows_updated -“更新的内部行数”』
第 955 页的『pkg_cache_inserts -“程序包高速缓存插入数”监视元素』
第 956 页的『pkg_cache_lookups -“程序包高速缓存查询数”监视元素』
第 982 页的『pool_data_l_reads -“缓冲池数据逻辑读取数”监视元素』
第 984 页的『pool_data_p_reads -“缓冲池数据物理读取数”监视元素』
第 986 页的『pool_data_writes -“缓冲池数据写次数”监视元素』
第 1012 页的『pool_index_l_reads -“缓冲池索引逻辑读取数”监视元素』
第 1014 页的『pool_index_p_reads -“缓冲池索引物理读取数”监视元素』
第 1016 页的『pool_index_writes -“缓冲池索引写次数”监视元素』
第 1043 页的『pool_read_time -“缓冲池物理读时间总计”监视元素』
第 1046 页的『pool_temp_data_l_reads -“缓冲池临时数据逻辑读取数”监视元素』
第 1048 页的『pool_temp_data_p_reads -“缓冲池临时数据物理读取数”监视元素』
第 1050 页的『pool_temp_index_l_reads -“缓冲池临时索引逻辑读取数”监视元素』
第 1052 页的『pool_temp_index_p_reads -“缓冲池临时索引物理读取数”监视元素』
第 1058 页的『pool_write_time -“缓冲池物理写时间总计”监视元素』
第 1080 页的『prefetch_wait_time -“等待预取的时间”监视元素』
第 1085 页的『priv_workspace_num_overflows -“专用工作空间溢出数”』

第 1086 页的『priv_workspace_section_inserts -“专用工作空间节插入数”』
第 1086 页的『priv_workspace_section_lookups -“专用工作空间节查询数”』
第 1087 页的『priv_workspace_size_top -“最大专用工作空间大小”』
第 1102 页的『rej_curs_blk -“拒绝的块游标请求数”』
第 1112 页的『rollback_sql_stmts -“尝试的回滚语句数”』
第 1118 页的『rows_read -“读取行数”监视元素』
第 1122 页的『rows_selected -“选择的行数”』
第 1123 页的『rows_written -“写入的行数”』
第 1129 页的『select_sql_stmts -“执行的 Select SQL 语句数”』
第 1130 页的『sequence_no -“序号”监视元素』
第 1138 页的『shr_workspace_num_overflows -“共享工作空间溢出数”』
第 1138 页的『shr_workspace_section_inserts -“共享工作空间节插入数”』
第 1139 页的『shr_workspace_section_lookups -“共享工作空间节查询数”』
第 1140 页的『shr_workspace_size_top -“最大共享工作空间大小”』
第 1149 页的『sort_overflows -“排序溢出数”监视元素』
第 1165 页的『static_sql_stmts -“尝试的静态 SQL 语句数”』
第 1189 页的『system_cpu_time -“系统 CPU 时间”监视元素』
第 1256 页的『total_hash_joins -“散列连接总数”』
第 1256 页的『total_hash_loops -“总散列循环数”』
第 1264 页的『total_olap_funcs -“OLAP 函数总数”监视元素』
第 1283 页的『total_sec_cons -“辅助连接数”』
第 1291 页的『total_sort_time -“排序时间总计”监视元素』
第 1292 页的『total_sorts -“排序总数”监视元素』
第 1313 页的『uid_sql_stmts -“执行的 Update/Insert/Delete SQL 语句数”』
第 1314 页的『unread_prefetch_pages -“未读取的预取页数”监视元素』
第 1326 页的『user_cpu_time -“用户 CPU 时间”监视元素』
第 1344 页的『x_lock_escals -“互斥锁定升级数”监视元素』
第 1346 页的『xquery_stmts -“尝试的 XQuery 语句数”』
第 623 页的『appl_status - 应用程序状态监视元素』
第 650 页的『cat_cache_size_top -“目录高速缓存高水位标记”监视元素』
第 695 页的『coord_node -“协调节点”』
第 751 页的『elapsed_exec_time -“语句执行耗用时间”』
第 763 页的『evmon_flushes -“事件监视器清空数”』
第 845 页的『lock_escals -“锁定升级次数”监视元素』
第 1053 页的『pool_temp_xda_l_reads -“缓冲池临时 XDA 数据逻辑读取数”监视元素』
第 1055 页的『pool_temp_xda_p_reads -“缓冲池临时 XDA 数据物理读取数”监视元素』
第 1065 页的『pool_xda_l_reads -“缓冲池 XDA 数据逻辑读取数”监视元素』
第 1069 页的『pool_xda_p_reads -“缓冲池 XDA 数据物理读取数”监视元素』
第 1071 页的『pool_xda_writes -“缓冲池 XDA 数据写次数”监视元素』

第 1115 页的『rows_deleted -“删除行数”监视元素』

第 1116 页的『rows_inserted -“插入行数”监视元素』

第 1123 页的『rows_updated -“更新行数”监视元素』

CAT_CACHE_HEAP_FULL

event_connheader 逻辑数据组

第 601 页的『agent_id -“应用程序句柄（代理程序标识）”监视元素』

第 616 页的『appl_id -“应用程序标识”监视元素』

第 620 页的『appl_name -“应用程序名称”监视元素』

第 635 页的『auth_id -“授权标识”』

第 659 页的『client_db_alias -“应用程序使用的数据库别名”』

第 661 页的『client_pid -“客户机进程标识”监视元素』

第 662 页的『client_platform -“客户机操作平台”监视元素』

第 663 页的『client_prdid -“客户机产品和版本标识”监视元素』

第 664 页的『client_protocol -“客户机通信协议”监视元素』

第 666 页的『codepage_id -“应用程序使用的代码页标识”』

第 682 页的『conn_time -“数据库连接时间”监视元素』

第 697 页的『corr_token -“DRDA 关联标记”』

第 764 页的『execution_id -“用户登录标识”』

第 908 页的『node_number -“节点号”』

第 1130 页的『sequence_no -“序号”监视元素』

第 1223 页的『territory_code -“数据库地域代码”』

第 661 页的『client_nname -“客户机名称”监视元素』

event_connmemuse 逻辑数据组

第 908 页的『node_number -“节点号”』

第 974 页的『pool_config_size -“内存池的已配置大小”』

第 975 页的『pool_cur_size -“内存池的当前大小”』

第 1004 页的『pool_id -“内存池标识”』

第 1045 页的『pool_secondary_id -“内存池辅助标记”』

第 1057 页的『pool_watermark -“内存池水位标记”』

第 616 页的『appl_id -“应用程序标识”监视元素』

第 763 页的『evmon_flushes -“事件监视器清空数”』

POOL_LIST_ID

POOL_MAX_SIZE

event_data_value 逻辑数据组

第 727 页的『deadlock_id -“死锁事件标识”』

第 727 页的『deadlock_node -“发生死锁的分区号”』

第 757 页的『evmon_activates -“事件监视器激活数”』

第 950 页的『participant_no -“死锁参与者”』

event_db 逻辑数据组

第 1291 页的『total_sort_time -“排序时间总计”监视元素』

第 1292 页的『total_sorts -“排序总数”监视元素』

第 1313 页的『uid_sql_stmts -“执行的 Update/Insert/Delete SQL 语句数”』

第 1314 页的『unread_prefetch_pages -“未读取的预取页数”监视元素』

第 1344 页的『x_lock_escals -“互斥锁定升级数”监视元素』

第 1346 页的『xquery_stmts -“尝试的 XQuery 语句数”』

第 751 页的『elapsed_exec_time -“语句执行耗用时间”』

第 915 页的『num_log_part_page_io -“部分日志页写入数”』

第 972 页的『pool_async_xda_read_reqs -“缓冲池异步 XDA 读请求数”监视元素』

第 973 页的『pool_async_xda_reads -“缓冲池异步 XDA 数据读取数”监视元素』

第 974 页的『pool_async_xda_writes -“缓冲池异步 XDA 数据写次数”监视元素』

第 1053 页的『pool_temp_xda_l_reads -“缓冲池临时 XDA 数据逻辑读取数”监视元素』

第 1055 页的『pool_temp_xda_p_reads -“缓冲池临时 XDA 数据物理读取数”监视元素』

第 1065 页的『pool_xda_l_reads -“缓冲池 XDA 数据逻辑读取数”监视元素』

第 1069 页的『pool_xda_p_reads -“缓冲池 XDA 数据物理读取数”监视元素』

第 1071 页的『pool_xda_writes -“缓冲池 XDA 数据写次数”监视元素』

第 1151 页的『sort_shrheap_top -“排序共享堆高水位标记”』

CAT_CACHE_HEAP_FULL

LOG_FILE_ARCHIVE

LOG_FILE_NUM_CURR

LOG_FILE_NUM_FIRST

LOG_FILE_NUM_LAST

NUM_LOG_BUFF_FULL

NUM_LOG_DATA_IN_BUFF

event_dbheader 逻辑数据组

第 682 页的『conn_time -“数据库连接时间”监视元素』

第 719 页的『db_name - 数据库名称监视元素』

第 720 页的『db_path -“数据库路径”』

event_dbmemuse 逻辑数据组

第 908 页的『node_number -“节点号”』

第 974 页的『pool_config_size -“内存池的已配置大小”』

第 975 页的『pool_cur_size -“内存池的当前大小”』

第 1004 页的『pool_id -“内存池标识”』

第 1057 页的『pool_watermark -“内存池水位标记”』

第 757 页的『evmon_activates -“事件监视器激活数”』

第 763 页的『evmon_flushes -“事件监视器清空数”』

第 1045 页的『pool_secondary_id -“内存池辅助标记”』

POOL_MAX_SIZE

event_deadlock 逻辑数据组

第 727 页的『deadlock_id -“死锁事件标识”』

第 727 页的『deadlock_node -“发生死锁的分区号”』

第 748 页的『dl_conns -“死锁中涉及的连接数”监视元素』

第 757 页的『evmon_activates -“事件监视器激活数”』

第 1113 页的『rolled_back_agent_id -“回滚的代理程序”』

第 1114 页的『rolled_back_appl_id -“回滚的应用程序”』

第 1114 页的『rolled_back_participant_no -“回滚的应用程序参与者”监视元素』

第 1114 页的『rolled_back_sequence_no -“回滚的序号”』

第 1164 页的『start_time -“事件启动时间”』

event_detailed_dlconn 逻辑数据组

第 601 页的『agent_id -“应用程序句柄（代理程序标识）”监视元素』

第 616 页的『appl_id -“应用程序标识”监视元素』

第 618 页的『appl_id_holding_lk -“挂起锁定的应用程序标识”』

第 640 页的『blocking_cursor -“分块游标”』

第 684 页的『consistency_token -“程序包一致性标记”监视元素』

第 709 页的『creator -“应用程序创建者”』

第 711 页的『cursor_name -“游标名称”』

第 713 页的『data_partition_id -“数据分区标识”监视元素』

第 727 页的『deadlock_id -“死锁事件标识”』

第 727 页的『deadlock_node -“发生死锁的分区号”』

第 757 页的『evmon_activates -“事件监视器激活数”』

第 845 页的『lock_escalation -“锁定升级”监视元素』

第 852 页的『lock_mode -“锁定方式”监视元素』

第 853 页的『lock_mode_requested -“请求的锁定方式”监视元素』

第 855 页的『lock_node -“锁定节点”』

第 855 页的『lock_object_name -“锁定对象名称”』

第 856 页的『lock_object_type -“等待的锁定对象类型”监视元素』

第 863 页的『lock_wait_start_time -“锁定等待开始时间戳记”监视元素』

第 871 页的『locks_held -“挂起的锁定数”监视元素』

第 872 页的『locks_in_list -“报告的锁定数”』

第 941 页的『package_name -“程序包名”监视元素』

第 943 页的『package_version_id -“程序包版本”监视元素』

第 950 页的『participant_no -“死锁参与者”』

第 950 页的『participant_no_holding_lk -“对应用程序所需对象挂起锁定的参与者”』

第 1128 页的『section_number -“节号”监视元素』

第 1130 页的『sequence_no -“序号”监视元素』

第 1131 页的『sequence_no_holding_lk -“挂起锁定的序号”』

第 1164 页的『start_time -“事件启动时间”』

第 1173 页的『stmt_operation/operation -“语句操作”监视元素』

第 1178 页的『stmt_text -“SQL 语句文本”监视元素』

第 1179 页的『stmt_type -“语句类型”监视元素』

第 1191 页的『table_name -“表名”监视元素』

第 1193 页的『table_schema -“表模式名”监视元素』

第 1200 页的『tablespace_name -“表空间名称”监视元素』

第 842 页的『lock_attributes -“锁定属性”监视元素』

第 843 页的『lock_count -“锁定计数”监视元素』

第 844 页的『lock_current_mode -“转换前的原始锁定方式”监视元素』

第 851 页的『lock_hold_count -“锁定挂起计数”监视元素』

第 854 页的『lock_name -“锁定名称”监视元素』

第 858 页的『lock_release_flags -“锁定释放标志”监视元素』

第 1303 页的『tpmon_acc_str -“TP 监视器客户机记帐字符串”监视元素』

第 1304 页的『tpmon_client_app -“TP 监视器客户机应用程序名称”监视元素』

第 1304 页的『tpmon_client_userid -“TP 监视器客户机用户标识”监视元素』

第 1305 页的『tpmon_client_wkstn -“TP 监视器客户机工作站名称”监视元素』

event_dlconn 逻辑数据组

第 601 页的『agent_id -“应用程序句柄（代理程序标识）”监视元素』

第 616 页的『appl_id -“应用程序标识”监视元素』

第 618 页的『appl_id_holding_lk -“挂起锁定的应用程序标识”』

第 713 页的『data_partition_id -“数据分区标识”监视元素』

第 727 页的『deadlock_id -“死锁事件标识”』

第 727 页的『deadlock_node -“发生死锁的分区号”』

第 757 页的『evmon_activates -“事件监视器激活数”』

第 842 页的『lock_attributes -“锁定属性”监视元素』

第 843 页的『lock_count -“锁定计数”监视元素』

第 844 页的『lock_current_mode -“转换前的原始锁定方式”监视元素』

第 845 页的『lock_escalation -“锁定升级”监视元素』

第 851 页的『lock_hold_count -“锁定挂起计数”监视元素』

第 852 页的『lock_mode -“锁定方式”监视元素』

第 853 页的『lock_mode_requested -“请求的锁定方式”监视元素』

第 854 页的『lock_name -“锁定名称”监视元素』

第 855 页的『lock_node -“锁定节点”』

第 855 页的『lock_object_name -“锁定对象名称”』

第 856 页的『lock_object_type -“等待的锁定对象类型”监视元素』

第 858 页的『lock_release_flags -“锁定释放标志”监视元素』

第 863 页的『lock_wait_start_time -“锁定等待开始时间戳记”监视元素』

第 950 页的『participant_no -“死锁参与者”』

第 950 页的『participant_no_holding_lk -“对应用程序所需对象挂起锁定的参与者”』

第 1130 页的『sequence_no -“序号”监视元素』

第 1131 页的『sequence_no_holding_lk -“挂起锁定的序号”』

第 1164 页的『start_time -“事件启动时间”』

第 1191 页的『table_name -“表名”监视元素』

第 1193 页的『table_schema -“表模式名”监视元素』

第 1200 页的『tablespace_name -“表空间名称”监视元素』

第 1303 页的『tpmon_acc_str -“TP 监视器客户机记帐字符串”监视元素』

第 1304 页的『tpmon_client_app -“TP 监视器客户机应用程序名称”监视元素』

第 1304 页的『tpmon_client_userid -“TP 监视器客户机用户标识”监视元素』

第 1305 页的『tpmon_client_wkstn -“TP 监视器客户机工作站名称”监视元素』

event_histogrambin 逻辑数据组

第 638 页的『bin_id -“直方图条形标识”监视元素』

第 641 页的『bottom -“直方图类别底部”监视元素』

第 810 页的『histogram_type -“直方图类型”监视元素』

第 921 页的『number_in_bin -“条形中的数目”监视元素』

第 1134 页的『service_class_id -“服务类标识”监视元素』

第 1165 页的『statistics_timestamp -“统计信息时间戳记”监视元素』

第 1230 页的『top -“最大直方图类别数”监视元素』

第 1339 页的『work_action_set_id -“工作操作集标识”监视元素』

第 1340 页的『work_class_id -“工作类标识”监视元素』

第 1341 页的『workload_id -“工作负载标识”监视元素』

第 905 页的『mon_interval_id -“监视时间间隔标识”监视元素』

第 896 页的『member -“数据库成员”监视元素』

event_log_header 逻辑数据组

第 646 页的『byte_order -“事件数据的字节顺序”』

第 666 页的『codepage_id -“应用程序使用的代码页标识”』

第 754 页的『event_monitor_name -“事件监视器名称”』

第 918 页的『num_nodes_in_db2_instance -“分区中的节点数”』

第 1132 页的『server_instance_name -“服务器实例名称”』

第 1133 页的『server_prdid -“服务器产品/版本标识”』

第 1223 页的『territory_code -“数据库地域代码”』

第 1334 页的『version -“监视器数据版本”』

event_overflow 逻辑数据组

第 697 页的『count -“事件监视器溢出数”』

第 791 页的『first_overflow_time -“第一次事件溢出时间”』

第 836 页的『last_overflow_time -“最后一次事件溢出时间”』

第 908 页的『node_number -“节点号”』

event_qstats 逻辑数据组

第 838 页的『last_wlm_reset -“最后一次重置时间”监视元素』

第 1093 页的『queue_assignments_total -“队列分配总次数”监视元素』

第 1094 页的『queue_size_top -“最大队列大小”监视元素』

第 1094 页的『queue_time_total -“总队列时间”监视元素』

第 1135 页的『service_subclass_name -“服务子类名”监视元素』

第 1136 页的『service_superclass_name -“服务超类名”监视元素』

第 1165 页的『statistics_timestamp -“统计信息时间戳记”监视元素』

第 1225 页的『threshold_domain -“阈值域”监视元素』

第 1226 页的『threshold_name -“阈值名称”监视元素』

第 1226 页的『threshold_predicate -“阈值谓词”监视元素』

第 1228 页的『thresholdid -“阈值标识”监视元素』

第 1339 页的『work_action_set_name -“工作操作集名称”监视元素』

第 1340 页的『work_class_name -“工作类名”监视元素』

第 905 页的『mon_interval_id -“监视时间间隔标识”监视元素』

第 896 页的『member -“数据库成员”监视元素』

event_scstats 逻辑数据组

第 590 页的『act_cpu_time_top -“最长活动 CPU 时间”监视元素』

第 592 页的『act_remapped_in -“重新映入的活动数”监视元素』

第 593 页的『act_remapped_out -“重新映出的活动数”监视元素』

第 593 页的『act_rows_read_top -“最大活动读取行数”监视元素』

第 594 页的『act_throughput -“活动吞吐量”监视元素』

第 610 页的『agg_temp_tablespace_top -“最大聚集临时表空间”监视元素』

第 672 页的『concurrent_act_top -“最大并行活动数”监视元素』

第 674 页的『concurrent_wlo_top -“最大并行工作负载项数”监视元素』

第 673 页的『concurrent_connection_top -“最大并行连接数”监视元素』

第 687 页的『coord_act_aborted_total -“异常终止的协调程序活动总数”监视元素』

第 688 页的『coord_act_completed_total -“完成的协调程序活动总数”监视元素』

第 689 页的『coord_act_est_cost_avg -“平均协调程序活动估计成本”监视元素』

第 689 页的『coord_act_exec_time_avg -“平均协调程序活动执行时间”监视元素』

第 690 页的『coord_act_interarrival_time_avg -“平均协调程序活动到达时间”监视元素』

第 691 页的『coord_act_lifetime_avg -“平均协调程序活动生存期”监视元素』

第 692 页的『coord_act_lifetime_top -“协调程序活动生存期项部”监视元素』

第 692 页的『coord_act_queue_time_avg -“平均协调程序活动队列时间”监视元素』

第 693 页的『coord_act_rejected_total -“被拒绝的协调程序活动总数”监视元素』

第 697 页的『cost_estimate_top -“最高估计成本”监视元素』

第 704 页的『cpu_utilization -“CPU 利用率”监视元素』

details_xml（此 XML 文档是一个 system_metrics 类型的文档，如 XML 模式文档 sqllib/misc/DB2MonCommon.xsd 中所述。 为超类 SYSDEFAULTSYSTEMCLASS 下的缺省子类 SYSDEFAULTSUBCLASS 报告的度量的值为 0。）

第 838 页的『last_wlm_reset -“最后一次重置时间”监视元素』

第 1111 页的『request_exec_time_avg -“平均请求执行时间”监视元素』

第 1122 页的『rows_returned_top -“最高实际返回行数”监视元素』

第 1134 页的『service_class_id -“服务类标识”监视元素』

第 1135 页的『service_subclass_name -“服务子类名”监视元素』

第 1136 页的『service_superclass_name -“服务超类名”监视元素』

第 1165 页的『statistics_timestamp -“统计信息时间戳记”监视元素』

第 1222 页的『temp_tablespace_top -“最大临时表空间”监视元素』

第 1249 页的『total_cpu_time -“CPU 时间总计”监视元素』

第 1251 页的『total_disp_run_queue_time -“分派器运行队列时间总计”监视元素』

第 1315 页的『uow_completed_total -“完成的工作单元总数”监视元素』

第 1317 页的『uow_lifetime_avg -“工作单元平均生存期”监视元素』

第 1321 页的『uow_throughput -“工作单元吞吐量”监视元素』

第 1321 页的『uow_total_time_top -“最长 UOW 时间总计”监视元素』

第 612 页的『app_act_aborted_total -“失败的外部协调程序活动总数”监视元素』

第 613 页的『app_act_completed_total -“成功的外部协调程序活动总数”监视元素』

第 614 页的『app_act_rejected_total -“拒绝的外部协调程序活动总数”监视元素』

第 905 页的『mon_interval_id -“监视时间间隔标识”监视元素』

第 896 页的『member -“数据库成员”监视元素』

event_start 逻辑数据组

第 1164 页的『start_time -“事件启动时间”』

event_stmt 逻辑数据组

第 601 页的『agent_id -“应用程序句柄（代理程序标识）”监视元素』

第 609 页的『agents_top -“创建的代理程序数”』

第 616 页的『appl_id -“应用程序标识”监视元素』

第 640 页的『blocking_cursor -“分块游标”』

第 684 页的『consistency_token -“程序包一致性标记”监视元素』

第 709 页的『creator -“应用程序创建者”』

第 711 页的『cursor_name -“游标名称”』

第 789 页的『fetch_count -“成功的访存数”』

第 825 页的『int_rows_deleted -“删除的内部行数”』

第 826 页的『int_rows_inserted -“插入的内部行数”』

第 826 页的『int_rows_updated -“更新的内部行数”』

第 941 页的『package_name -“程序包名”监视元素』

第 943 页的『package_version_id -“程序包版本”监视元素』

第 949 页的『partial_record -“部分记录”监视元素』

第 982 页的『pool_data_l_reads -“缓冲池数据逻辑读取数”监视元素』

第 984 页的『pool_data_p_reads -“缓冲池数据物理读取数”监视元素』

第 1012 页的『pool_index_l_reads -“缓冲池索引逻辑读取数”监视元素』

第 1014 页的『pool_index_p_reads -“缓冲池索引物理读取数”监视元素』

第 1046 页的『pool_temp_data_l_reads -“缓冲池临时数据逻辑读取数”监视元素』

第 1048 页的『pool_temp_data_p_reads -“缓冲池临时数据物理读取数”监视元素』

第 1050 页的『pool_temp_index_l_reads -“缓冲池临时索引逻辑读取数”监视元素』

第 1052 页的『pool_temp_index_p_reads -“缓冲池临时索引物理读取数”监视元素』

第 1118 页的『rows_read -“读取行数”监视元素』

第 1123 页的『rows_written -“写入的行数”』

第 1128 页的『section_number -“节号”监视元素』

第 1130 页的『sequence_no -“序号”监视元素』

第 1149 页的『sort_overflows -“排序溢出数”监视元素』

第 1153 页的『sql_req_id -“SQL 语句的请求标识”』

第 1154 页的『sqlca -“SQL 通信区（SQLCA）”』

第 1164 页的『start_time -“事件启动时间”』

第 1166 页的『stats_fabricate_time -“生成统计信息的活动所花的总时间”监视元素』

第 1173 页的『stmt_operation/operation -“语句操作”监视元素』

第 1178 页的『stmt_text -“SQL 语句文本”监视元素』

第 1179 页的『stmt_type -“语句类型”监视元素』

第 1185 页的『stop_time -“事件停止时间”』

第 1188 页的『sync_runstats_time -“同步 RUNSTATS 活动所花的总时间”监视元素』

第 1189 页的『system_cpu_time -“系统 CPU 时间”监视元素』

第 1291 页的『total_sort_time -“排序时间总计”监视元素』

第 1292 页的『total_sorts -“排序总数”监视元素』

第 1326 页的『user_cpu_time -“用户 CPU 时间”监视元素』

第 763 页的『evmon_flushes -“事件监视器清空数”』

第 1053 页的『pool_temp_xda_l_reads -“缓冲池临时 XDA 数据逻辑读取数”监视元素』

第 1055 页的『pool_temp_xda_p_reads -“缓冲池临时 XDA 数据物理读取数”监视元素』

第 1065 页的『pool_xda_l_reads -“缓冲池 XDA 数据逻辑读取数”监视元素』

第 1069 页的『pool_xda_p_reads -“缓冲池 XDA 数据物理读取数”监视元素』

event_stmt_history 逻辑数据组

第 669 页的『comp_env_desc -“编译环境”监视元素』

第 709 页的『creator -“应用程序创建者”』

第 727 页的『deadlock_id -“死锁事件标识”』

第 727 页的『deadlock_node -“发生死锁的分区号”』

第 757 页的『evmon_activates -“事件监视器激活数”』

第 941 页的『package_name -“程序包名”监视元素』

第 943 页的『package_version_id -“程序包版本”监视元素』

第 950 页的『participant_no -“死锁参与者”』

第 1128 页的『section_number -“节号”监视元素』

第 1130 页的『sequence_no -“序号”监视元素』

第 1169 页的『stmt_first_use_time -“第一次使用语句时的时间戳记”监视元素』

第 1169 页的『stmt_history_id -“语句历史记录标识”』

第 1170 页的『stmt_invocation_id -“语句调用标识”监视元素』

第 1171 页的『stmt_isolation -“语句隔离”』

第 1171 页的『stmt_last_use_time -“上一次使用语句时的时间戳记”监视元素』

第 1172 页的『stmt_lock_timeout -“语句锁定超时”监视元素』

第 1172 页的『stmt_nest_level -“语句嵌套级别”监视元素』

第 1174 页的『stmt_pkgcache_id -“语句程序包高速缓存标识”监视元素』

第 1175 页的『stmt_query_id -“语句查询标识”监视元素』

第 1176 页的『stmt_source_id -“语句源标识”』

第 1178 页的『stmt_text -“SQL 语句文本”监视元素』

第 1179 页的『stmt_type -“语句类型”监视元素』

第 616 页的『appl_id -“应用程序标识”监视元素』

event_subsection 逻辑数据组

第 601 页的『agent_id -“应用程序句柄（代理程序标识）”监视元素』

第 909 页的『num_agents -“正在处理语句的代理程序数”』

第 949 页的『partial_record -“部分记录”监视元素』

第 1161 页的『ss_exec_time -“子节执行耗用时间”』

第 1161 页的『ss_node_number -“子节节点号”』

第 1162 页的『ss_number -“子节号”监视元素』

第 1162 页的『ss_sys_cpu_time -“子节使用的系统 CPU 时间”』

第 1163 页的『ss_usr_cpu_time -“子节使用的用户 CPU 时间”』

第 1306 页的『tq_max_send_spills -“最大表队列缓冲区溢出数”』

第 1307 页的『tq_rows_read -“从表队列读取的行数”』

第 1307 页的『tq_rows_written -“写至表队列的行数”』

第 1311 页的『tq_tot_send_spills -“溢出表队列缓冲区总数”监视元素』

第 616 页的『appl_id -“应用程序标识”监视元素』

第 763 页的『evmon_flushes -“事件监视器清空数”』

第 1153 页的『sql_req_id -“SQL 语句的请求标识”』

event_table 逻辑数据组

第 712 页的『data_object_pages -“数据对象页数”』

第 713 页的『data_partition_id -“数据分区标识”监视元素』

第 755 页的『event_time -“事件时间”』

第 757 页的『evmon_activates -“事件监视器激活数”』

第 763 页的『evmon_flushes -“事件监视器清空数”』

第 817 页的『index_object_pages -“索引对象页数”』

第 839 页的『lob_object_pages -“LOB 对象页数”』

第 880 页的『long_object_pages -“长对象页数”』

第 940 页的『overflow_accesses -“访问溢出记录次数”监视元素』

第 944 页的『page_reorgs -“页重组”监视元素』

第 949 页的『partial_record -“部分记录”监视元素』

第 1118 页的『rows_read -“读取行数”监视元素』

第 1123 页的『rows_written -“写入的行数”』

第 1191 页的『table_name -“表名”监视元素』

第 1193 页的『table_schema -“表模式名”监视元素』

第 1194 页的『table_type -“表类型”监视元素』

第 1197 页的『tablespace_id -“表空间标识”监视元素』

第 1345 页的『xda_object_pages -“XDA 对象页数”』

event_tablespace 逻辑数据组

第 735 页的『direct_read_reqs -“直接读请求数”监视元素』

第 737 页的『direct_read_time -“直接读时间”监视元素』

第 738 页的『direct_reads -“直接读数据库数目”监视元素』

第 740 页的『direct_write_reqs -“直接写请求数”监视元素』

第 742 页的『direct_write_time -“直接写时间”监视元素』

第 744 页的『direct_writes -“直接写数据库数目”监视元素』

第 755 页的『event_time -“事件时间”』

第 757 页的『evmon_activates -“事件监视器激活数”』

第 763 页的『evmon_flushes -“事件监视器清空数”』

第 790 页的『files_closed -“关闭数据库文件数”监视元素』

第 949 页的『partial_record -“部分记录”监视元素』

第 961 页的『pool_async_data_read_reqs -“缓冲池异步读请求数”监视元素』

第 962 页的『pool_async_data_reads -“缓冲池异步数据读次数”监视元素』

第 963 页的『pool_async_data_writes -“缓冲池异步数据写次数”监视元素』

第 966 页的『pool_async_index_read_reqs -“缓冲池异步索引读请求数”监视元素』

第 966 页的『pool_async_index_reads -“缓冲池异步索引读取数”监视元素』

第 967 页的『pool_async_index_writes -“缓冲池异步索引写次数”监视元素』

第 968 页的『pool_async_read_time -“缓冲池异步读取时间”』

第 969 页的『pool_async_write_time -“缓冲池异步写入时间”监视元素』

第 982 页的『pool_data_l_reads -“缓冲池数据逻辑读取数”监视元素』

第 984 页的『pool_data_p_reads -“缓冲池数据物理读取数”监视元素』

第 986 页的『pool_data_writes -“缓冲池数据写次数”监视元素』

第 1012 页的『pool_index_l_reads -“缓冲池索引逻辑读取数”监视元素』

第 1014 页的『pool_index_p_reads -“缓冲池索引物理读取数”监视元素』

第 1016 页的『pool_index_writes -“缓冲池索引写次数”监视元素』

第 1018 页的『pool_no_victim_buffer -“缓冲池无牺牲缓冲区次数”监视元素』

第 1043 页的『pool_read_time -“缓冲池物理读时间总计”监视元素』

第 1046 页的『pool_temp_data_l_reads -“缓冲池临时数据逻辑读取数”监视元素』

第 1048 页的『pool_temp_data_p_reads -“缓冲池临时数据物理读取数”监视元素』

第 1050 页的『pool_temp_index_l_reads -“缓冲池临时索引逻辑读取数”监视元素』

第 1052 页的『pool_temp_index_p_reads -“缓冲池临时索引物理读取数”监视元素』

第 1058 页的『pool_write_time -“缓冲池物理写时间总计”监视元素』

第 1200 页的『tablespace_name -“表空间名称”监视元素』

第 972 页的『pool_async_xda_read_reqs -“缓冲池异步 XDA 读请求数”监视元素』

第 973 页的『pool_async_xda_reads -“缓冲池异步 XDA 数据读取数”监视元素』

第 974 页的『pool_async_xda_writes -“缓冲池异步 XDA 数据写次数”监视元素』

第 1065 页的『pool_xda_l_reads -“缓冲池 XDA 数据逻辑读取数”监视元素』

第 1069 页的『pool_xda_p_reads -“缓冲池 XDA 数据物理读取数”监视元素』

第 1071 页的『pool_xda_writes -“缓冲池 XDA 数据写次数”监视元素』

TABLESPACE_FS_CACHING

第 1314 页的『unread_prefetch_pages -“未读取的预取页数”监视元素』

event_thresholdviolations 逻辑数据组

第 595 页的『activate_timestamp -“激活时间戳记”监视元素』

第 596 页的『activity_collected -“收集的活动”监视元素』

第 597 页的『activity_id -“活动标识”监视元素』

第 601 页的『agent_id -“应用程序句柄（代理程序标识）”监视元素』

第 616 页的『appl_id -“应用程序标识”监视元素』

第 696 页的『coord_partition_num -“协调程序分区号”监视元素』

第 731 页的『destination_service_class_id -“目标服务类标识”监视元素』

第 1152 页的『source_service_class_id -“源服务类标识”监视元素』

第 1225 页的『threshold_action -“阈值操作”监视元素』

第 1226 页的『threshold_maxvalue -“阈值最大值”监视元素』

第 1226 页的『threshold_predicate -“阈值谓词”监视元素』

第 1227 页的『threshold_queuesize -“阈值队列大小”监视元素』

第 1228 页的『thresholdid -“阈值标识”监视元素』

第 1229 页的『time_of_violation -“违例时间”监视元素』

第 1316 页的『uow_id -“工作单元标识”监视元素』

第 896 页的『member -“数据库成员”监视元素』

event_wcstats 逻辑数据组

第 590 页的『act_cpu_time_top -“最长活动 CPU 时间”监视元素』

第 593 页的『act_rows_read_top -“最大活动读取行数”监视元素』

第 595 页的『act_total -“活动总数”监视元素』

第 689 页的『coord_act_est_cost_avg -“平均协调程序活动估计成本”监视元素』

第 689 页的『coord_act_exec_time_avg -“平均协调程序活动执行时间”监视元素』

第 690 页的『coord_act_interarrival_time_avg -“平均协调程序活动到达时间”监视元素』

第 691 页的『coord_act_lifetime_avg -“平均协调程序活动生存期”监视元素』

第 692 页的『coord_act_lifetime_top -“协调程序活动生存期项部”监视元素』

第 692 页的『coord_act_queue_time_avg -“平均协调程序活动队列时间”监视元素』

第 697 页的『cost_estimate_top -“最高估计成本”监视元素』

第 838 页的『last_wlm_reset -“最后一次重置时间”监视元素』

第 1122 页的『rows_returned_top -“最高实际返回行数”监视元素』

第 1165 页的『statistics_timestamp -“统计信息时间戳记”监视元素』

第 1222 页的『temp_tablespace_top -“最大临时表空间”监视元素』

第 1339 页的『work_action_set_id -“工作操作集标识”监视元素』

第 1339 页的『work_action_set_name -“工作操作集名称”监视元素』

第 1340 页的『work_class_id -“工作类标识”监视元素』

第 1340 页的『work_class_name -“工作类名”监视元素』

第 905 页的『mon_interval_id -“监视时间间隔标识”监视元素』

第 896 页的『member -“数据库成员”监视元素』

event_wlstats 逻辑数据组

第 590 页的『act_cpu_time_top -“最长活动 CPU 时间”监视元素』

第 593 页的『act_rows_read_top -“最大活动读取行数”监视元素』

第 594 页的『act_throughput -“活动吞吐量”监视元素』

第 673 页的『concurrent_wlo_act_top -“最大并行 WLO 活动数”监视元素』

第 674 页的『concurrent_wlo_top -“最大并行工作负载项数”监视元素』

第 687 页的『coord_act_aborted_total -“异常终止的协调程序活动总数”监视元素』

第 688 页的『coord_act_completed_total -“完成的协调程序活动总数”监视元素』

第 689 页的『coord_act_est_cost_avg -“平均协调程序活动估计成本”监视元素』

第 689 页的『coord_act_exec_time_avg -“平均协调程序活动执行时间”监视元素』

第 690 页的『coord_act_interarrival_time_avg -“平均协调程序活动到达时间”监视元素』

第 691 页的『coord_act_lifetime_avg -“平均协调程序活动生存期”监视元素』

第 692 页的『coord_act_lifetime_top -“协调程序活动生存期项部”监视元素』

第 692 页的『coord_act_queue_time_avg -“平均协调程序活动队列时间”监视元素』

第 693 页的『coord_act_rejected_total -“被拒绝的协调程序活动总数”监视元素』

第 697 页的『cost_estimate_top -“最高估计成本”监视元素』

第 704 页的『cpu_utilization -“CPU 利用率”监视元素』

details_xml（此 XML 文档是一个 system_metrics 类型的度量值文档，如 XML 模式文档 `sqllib/misc/DB2MonCommon.xsd` 中所述。）

第 838 页的『last_wlm_reset -“最后一次重置时间”监视元素』

第 867 页的『lock_wait_time_top -“最长锁定等待时间”监视元素』

第 1122 页的『rows_returned_top -“最高实际返回行数”监视元素』

第 1165 页的『statistics_timestamp -“统计信息时间戳记”监视元素』

第 1222 页的『temp_tablespace_top -“最大临时表空间”监视元素』

第 1249 页的『total_cpu_time -“CPU 时间总计”监视元素』

第 1251 页的『total_disp_run_queue_time -“分派器运行队列时间总计”监视元素』

第 1315 页的『uow_completed_total -“完成的工作单元总数”监视元素』

第 1317 页的『uow_lifetime_avg -“工作单元平均生存期”监视元素』

第 1321 页的『uow_throughput -“工作单元吞吐量”监视元素』

第 1321 页的『uow_total_time_top -“最长 UOW 时间总计”监视元素』

第 1338 页的『wlo_completed_total -“完成的工作负载项总数”监视元素』

第 1341 页的『workload_id -“工作负载标识”监视元素』

第 1342 页的『workload_name -“工作负载名称”监视元素』

第 612 页的『app_act_aborted_total -“失败的外部协调程序活动总数”监视元素』

第 613 页的『app_act_completed_total -“成功的外部协调程序活动总数”监视元素』

第 614 页的『app_act_rejected_total -“拒绝的外部协调程序活动总数”监视元素』

第 867 页的『lock_wait_time_global_top -“最长全局锁定等待时间”监视元素』

第 905 页的『mon_interval_id -“监视时间间隔标识”监视元素』

第 896 页的『member -“数据库成员”监视元素』

event_xact 逻辑数据组

第 601 页的『agent_id -“应用程序句柄（代理程序标识）”监视元素』

第 616 页的『appl_id -“应用程序标识”监视元素』

第 845 页的『lock_escals -“锁定升级次数”监视元素』

第 863 页的『lock_wait_time -“等待锁定时间”监视元素』

第 871 页的『locks_held_top -“挂起的最大锁定数”监视元素』

第 949 页的『partial_record -“部分记录”监视元素』

第 1084 页的『prev_uow_stop_time -“上一个工作单元完成时间戳记”』

第 1118 页的『rows_read -“读取行数”监视元素』

第 1123 页的『rows_written -“写入的行数”』

第 1130 页的『sequence_no -“序号”监视元素』

第 1189 页的『system_cpu_time -“系统 CPU 时间”监视元素』

第 1303 页的『tpmon_acc_str -“TP 监视器客户机记帐字符串”监视元素』

第 1304 页的『tpmon_client_app -“TP 监视器客户机应用程序名称”监视元素』

第 1304 页的『tpmon_client_userid -“TP 监视器客户机用户标识”监视元素』

第 1305 页的『tpmon_client_wkstn -“TP 监视器客户机工作站名称”监视元素』

第 1318 页的『uow_log_space_used -“使用的工作单元日志空间”监视元素』

第 1319 页的『uow_start_time -“工作单元开始时间戳记”监视元素』

第 1320 页的『uow_status -“工作单元状态”』

第 1185 页的『stop_time -“事件停止时间”』

第 1326 页的『user_cpu_time -“用户 CPU 时间”监视元素』

第 1344 页的『x_lock_escals -“互斥锁定升级数”监视元素』

第 763 页的『evmon_flushes -“事件监视器清空数”』

evmonstart 逻辑数据组

第 754 页的『event_id -“事件标识”监视元素』

第 755 页的『event_timestamp -“事件时间戳记”监视元素』

第 896 页的『member -“数据库成员”监视元素』

第 756 页的『event_type -“事件类型”监视元素』

第 717 页的『db2start_time -“启动数据库管理器时间戳记”』

第 717 页的『db_conn_time -“数据库激活时间戳记”监视元素』

lock 逻辑数据组

第 951 页的『partition_key -“分区键”监视元素』

第 748 页的『dl_conns -“死锁中涉及的连接数”监视元素』

第 754 页的『event_id -“事件标识”监视元素』

第 755 页的『event_timestamp -“事件时间戳记”监视元素』

第 756 页的『event_type -“事件类型”监视元素』

第 951 页的『partition_number -“分区号”』

第 1114 页的『rolled_back_participant_no -“回滚的应用程序参与者”监视元素』

第 728 页的『deadlock_type -“死锁类型”监视元素』

lock_activity_values 逻辑数据组

第 951 页的『partition_key -“分区键”监视元素』

第 597 页的『activity_id -“活动标识”监视元素』

第 754 页的『event_id -“事件标识”监视元素』

第 755 页的『event_timestamp -“事件时间戳记”监视元素』

第 756 页的『event_type -“事件类型”监视元素』

第 950 页的『participant_no -“死锁参与者”』

第 951 页的『partition_number -“分区号”』

第 1182 页的『stmt_value_index -“值索引”』

第 1182 页的『stmt_value_isnull -“包含空值”监视元素』

第 1183 页的『stmt_value_isreopt -“用于语句重新优化的变量”监视元素』

第 1184 页的『stmt_value_type -“值类型”监视元素』

第 1316 页的『uow_id -“工作单元标识”监视元素』

第 1181 页的『stmt_value_data -“值数据”』

第 754 页的『event_id -“事件标识”监视元素』

lock_participant_activities 逻辑数据组

第 951 页的『partition_key -“分区键”监视元素』

第 597 页的『activity_id -“活动标识”监视元素』

第 598 页的『activity_type -“活动类型”监视元素』

第 684 页的『consistency_token -“程序包一致性标记”监视元素』

第 750 页的『effective_isolation -“有效隔离级别”监视元素』

第 750 页的『effective_query_degree -“有效查询并行度”监视元素』

第 754 页的『event_id -“事件标识”监视元素』

第 755 页的『event_timestamp -“事件时间戳记”监视元素』

第 756 页的『event_type -“事件类型”监视元素』

第 816 页的『incremental_bind -“增量绑定”监视元素』
第 941 页的『package_name -“程序包名”监视元素』
第 942 页的『package_schema -“程序包模式”监视元素』
第 943 页的『package_version_id -“程序包版本”监视元素』
第 950 页的『participant_no -“死锁参与者”』
第 951 页的『partition_number -“分区号”』
第 1091 页的『query_actual_degree -“实际运行时分区内并行度”监视元素』
第 1104 页的『reopt -“REOPT 绑定选项”监视元素』
第 1128 页的『section_number -“节号”监视元素』
第 1169 页的『stmt_first_use_time -“第一次使用语句时的时间戳记”监视元素』
第 1170 页的『stmt_invocation_id -“语句调用标识”监视元素』
第 1171 页的『stmt_last_use_time -“上一次使用语句时的时间戳记”监视元素』
第 1172 页的『stmt_lock_timeout -“语句锁定超时”监视元素』
第 1172 页的『stmt_nest_level -“语句嵌套级别”监视元素』
第 1174 页的『stmt_pkgcache_id -“语句程序包高速缓存标识”监视元素』
第 1175 页的『stmt_query_id -“语句查询标识”监视元素』
第 1176 页的『stmt_source_id -“语句源标识”』
第 1179 页的『stmt_type -“语句类型”监视元素』
第 1316 页的『uow_id -“工作单元标识”监视元素』
第 1178 页的『stmt_text -“SQL 语句文本”监视元素』
第 1181 页的『stmt_unicode -“语句 Unicode 标志”监视元素』
第 1173 页的『stmt_operation/operation -“语句操作”监视元素』

lock_participants 逻辑数据组

第 951 页的『partition_key -“分区键”监视元素』
第 603 页的『agent_status -“DCS 应用程序代理程序数”』
第 601 页的『agent_id -“应用程序句柄（代理程序标识）”监视元素』
第 616 页的『appl_id -“应用程序标识”监视元素』
第 620 页的『appl_name -“应用程序名称”监视元素』
第 625 页的『application_handle -“应用程序句柄”监视元素』
第 635 页的『auth_id -“授权标识”』
第 657 页的『client_acctng -“客户机记帐字符串”监视元素』
第 658 页的『client_applname -“客户机应用程序名称”监视元素』
第 664 页的『client_userid -“客户机用户标识”监视元素』
第 665 页的『client_wrkstnname -“客户机工作站名称”监视元素』
第 692 页的『coord_agent_tid -“协调代理程序引擎可分派单元标识”监视元素』
第 711 页的『current_request -“当前操作请求”监视元素』
第 754 页的『event_id -“事件标识”监视元素』
第 755 页的『event_timestamp -“事件时间戳记”监视元素』
第 756 页的『event_type -“事件类型”监视元素』

第 842 页的『lock_attributes -“锁定属性”监视元素』

第 843 页的『lock_count -“锁定计数”监视元素』

第 844 页的『lock_current_mode -“转换前的原始锁定方式”监视元素』

第 845 页的『lock_escalation -“锁定升级”监视元素』

第 851 页的『lock_hold_count -“锁定挂起计数”监视元素』

第 852 页的『lock_mode -“锁定方式”监视元素』

第 853 页的『lock_mode_requested -“请求的锁定方式”监视元素』

第 854 页的『lock_name -“锁定名称”监视元素』

第 856 页的『lock_object_type -“等待的锁定对象类型”监视元素』

LOCK_OBJECT_TYPE_ID

第 858 页的『lock_release_flags -“锁定释放标志”监视元素』

LOCK_RRIID

第 858 页的『lock_status -“锁定状态”监视元素』

第 859 页的『lock_timeout_val -“锁定超时值”监视元素』

第 863 页的『lock_wait_end_time -“锁定等待结束时间戳记”监视元素』

第 863 页的『lock_wait_start_time -“锁定等待开始时间戳记”监视元素』

第 867 页的『lock_wait_val -“锁定等待值”监视元素』

第 896 页的『member -“数据库成员”监视元素』

第 928 页的『object_requested -“所请求对象”监视元素』

第 950 页的『participant_no -“死锁参与者”』

第 950 页的『participant_no_holding_lk -“对应用程序所需对象挂起锁定的参与者”』

第 950 页的『participant_type -“参与者类型”监视元素』

第 953 页的『past_activities_wrapped -“合并过去活动列表”监视元素』

第 1134 页的『service_class_id -“服务类标识”监视元素』

第 1135 页的『service_subclass_name -“服务子类名”监视元素』

第 1190 页的『table_file_id -“表文件标识”监视元素』

第 1191 页的『table_name -“表名”监视元素』

第 1193 页的『table_schema -“表模式名”监视元素』

第 1228 页的『thresholdid -“阈值标识”监视元素』

第 1226 页的『threshold_name -“阈值名称”监视元素』

第 1341 页的『workload_id -“工作负载标识”监视元素』

第 1342 页的『workload_name -“工作负载名称”监视元素』

第 604 页的『agent_tid -“代理程序线程标识”监视元素』

第 615 页的『appl_action -“应用程序操作”监视元素』

第 727 页的『deadlock_member -“死锁成员”监视元素』

第 1094 页的『queue_start_time -“队列开始时间戳记”监视元素』

第 1095 页的『queued_agents -“已排队阈值代理程序数”监视元素』

第 1136 页的『service_superclass_name -“服务超类名”监视元素』

第 1200 页的『tablespace_name -“表空间名称”监视元素』

第 1346 页的『xid -“事务标识”』

第 1328 页的『utility_invocation_id -“实用程序调用标识”』

pkgcache 逻辑数据组

第 951 页的『partition_key -“分区键”监视元素』

第 669 页的『comp_env_desc -“编译环境”监视元素』

第 750 页的『effective_isolation -“有效隔离级别”监视元素』

第 754 页的『event_id -“事件标识”监视元素』

第 755 页的『event_timestamp -“事件时间戳记”监视元素』

第 762 页的『executable_id -“可执行文件标识”监视元素』

第 820 页的『insert_timestamp -“插入时间戳记”监视元素』

第 835 页的『last_metrics_update -“最近一次更新度量的时间戳记”监视元素』

第 896 页的『member -“数据库成员”监视元素』

第 910 页的『num_coord_exec -“协调代理程序执行的次数”监视元素』

第 910 页的『num_coord_exec_with_metrics -“协调代理程序执行的次数以及度量”监视元素』

第 912 页的『num_exec_with_metrics -“在收集度量值情况下的执行次数”监视元素』

第 911 页的『num_executions -“语句执行次数”监视元素』

第 941 页的『package_name -“程序包名”监视元素』

第 942 页的『package_schema -“程序包模式”监视元素』

第 943 页的『package_version_id -“程序包版本”监视元素』

第 951 页的『partition_number -“分区号”』

第 1083 页的『prep_time -“编译时间”监视元素』

第 1092 页的『query_cost_estimate -“查询估算成本”监视元素』

第 1093 页的『query_data_tag_list -“估算的查询数据标记列表”监视元素』

第 1115 页的『routine_id -“例程标识”监视元素』

第 1128 页的『section_env -“节环境”监视元素』

第 1128 页的『section_number -“节号”监视元素』

第 1129 页的『section_type -“节类型指示器”监视元素』

第 1174 页的『stmt_pkgcache_id -“语句程序包高速缓存标识”监视元素』

第 1180 页的『stmt_type_id -“语句类型标识”监视元素』

第 1294 页的『total_stats_fabrication_time -“统计信息生成时间总计”监视元素』

第 1295 页的『total_stats_fabrications -“统计信息生成总计”监视元素』

第 1299 页的『total_sync_runstats -“同步 RUNSTATS 活动总数”监视元素』

第 1296 页的『total_sync_runstats_time -“同步 RUNSTATS 时间总计”监视元素』

第 1178 页的『stmt_text -“SQL 语句文本”监视元素』

第 881 页的『max_coord_stmt_exec_time -“最长协调程序语句执行时间”监视元素』

第 884 页的『max_coord_stmt_exec_timestamp -“最大协调程序语句执行时间戳记”监视元素』

pkgcache_metrics 逻辑数据组

第 951 页的『partition_key -“分区键”监视元素』

第 754 页的『event_id -“事件标识”监视元素』

第 755 页的『event_timestamp -“事件时间戳记”监视元素』

第 951 页的『partition_number -“分区号”』

第 1337 页的『wlm_queue_time_total -“工作负载管理器队列时间总计”监视元素』

第 1336 页的『wlm_queue_assignments_total -“工作负载管理器队列分配总次数”监视元素』

第 783 页的『fcm_tq_recv_wait_time -“FCM 表队列接收等待时间”监视元素』

第 768 页的『fcm_message_recv_wait_time -“接收 FCM 消息等待时间”监视元素』

第 787 页的『fcm_tq_send_wait_time -“FCM 表队列发送等待时间”监视元素』

第 772 页的『fcm_message_send_wait_time -“发送 FCM 消息等待时间”监视元素』

第 863 页的『lock_wait_time -“等待锁定时间”监视元素』

第 868 页的『lock_waits -“等待锁定次数”监视元素』

第 737 页的『direct_read_time -“直接读时间”监视元素』

第 735 页的『direct_read_reqs -“直接读请求数”监视元素』

第 742 页的『direct_write_time -“直接写时间”监视元素』

第 740 页的『direct_write_reqs -“直接写请求数”监视元素』

第 872 页的『log_buffer_wait_time -“日志缓冲区等待时间”监视元素』

第 913 页的『num_log_buffer_full -“日志缓冲区变满而导致代理程序等待的次数”监视元素』

第 874 页的『log_disk_wait_time -“日志磁盘等待时间”监视元素』

第 875 页的『log_disk_waits_total -“日志磁盘等待总次数”监视元素』

第 1058 页的『pool_write_time -“缓冲池物理写时间总计”监视元素』

第 1043 页的『pool_read_time -“缓冲池物理读时间总计”监视元素』

第 629 页的『audit_file_write_wait_time -“审计文件写等待时间”监视元素』

第 631 页的『audit_file_writes_total -“写审计文件总次数”监视元素』

第 632 页的『audit_subsystem_wait_time -“审计子系统等待时间”监视元素』

第 634 页的『audit_subsystem_waits_total -“审计子系统等待总次数”监视元素』

第 732 页的『diaglog_write_wait_time -“诊断日志文件写等待时间”监视元素』

第 734 页的『diaglog_writes_total -“写诊断日志文件总次数”监视元素』

第 779 页的『fcm_send_wait_time -“FCM 发送等待时间”监视元素』

第 775 页的『fcm_recv_wait_time -“FCM 接收等待时间”监视元素』

第 1232 页的『total_act_wait_time -“活动等待时间总计”监视元素』

第 1285 页的『total_section_sort_proc_time -“节排序处理时间总计”监视元素』

第 1288 页的『total_section_sorts -“节排序总次数”监视元素』

第 1287 页的『total_section_sort_time -“节排序时间总计”监视元素』

第 1231 页的『total_act_time -“活动时间总计”监视元素』

第 1118 页的『rows_read -“读取行数”监视元素』

第 1117 页的『rows_modified -“修改的行数”监视元素』

第 982 页的『pool_data_l_reads -“缓冲池数据逻辑读取数”监视元素』
第 1012 页的『pool_index_l_reads -“缓冲池索引逻辑读取数”监视元素』
第 1046 页的『pool_temp_data_l_reads -“缓冲池临时数据逻辑读取数”监视元素』
第 1050 页的『pool_temp_index_l_reads -“缓冲池临时索引逻辑读取数”监视元素』
第 1065 页的『pool_xda_l_reads -“缓冲池 XDA 数据逻辑读取数”监视元素』
第 1053 页的『pool_temp_xda_l_reads -“缓冲池临时 XDA 数据逻辑读取数”监视元素』
第 1249 页的『total_cpu_time -“CPU 时间总计”监视元素』
第 984 页的『pool_data_p_reads -“缓冲池数据物理读取数”监视元素』
第 1048 页的『pool_temp_data_p_reads -“缓冲池临时数据物理读取数”监视元素』
第 1069 页的『pool_xda_p_reads -“缓冲池 XDA 数据物理读取数”监视元素』
第 1055 页的『pool_temp_xda_p_reads -“缓冲池临时 XDA 数据物理读取数”监视元素』
第 1014 页的『pool_index_p_reads -“缓冲池索引物理读取数”监视元素』
第 1052 页的『pool_temp_index_p_reads -“缓冲池临时索引物理读取数”监视元素』
第 986 页的『pool_data_writes -“缓冲池数据写次数”监视元素』
第 1071 页的『pool_xda_writes -“缓冲池 XDA 数据写次数”监视元素』
第 1016 页的『pool_index_writes -“缓冲池索引写次数”监视元素』
第 738 页的『direct_reads -“直接读数据库数目”监视元素』
第 744 页的『direct_writes -“直接写数据库数目”监视元素』
第 1120 页的『rows_returned -“返回的行数”监视元素』
第 728 页的『deadlocks -“检测到的死锁数”监视元素』
第 860 页的『lock_timeouts -“锁定超时次数”监视元素』
第 845 页的『lock_escals -“锁定升级次数”监视元素』
第 780 页的『fcm_sends_total -“FCM 发送总计”监视元素』
第 777 页的『fcm_recvs_total -“FCM 接收总计”监视元素』
第 778 页的『fcm_send_volume -“FCM 发送量”监视元素』
第 774 页的『fcm_recv_volume -“FCM 接收量”监视元素』
第 773 页的『fcm_message_sends_total -“发送 FCM 消息总数”监视元素』
第 769 页的『fcm_message_recvs_total -“接收 FCM 消息总数”监视元素』
第 770 页的『fcm_message_send_volume -“发送 FCM 消息量”监视元素』
第 766 页的『fcm_message_recv_volume -“接收 FCM 消息量”监视元素』
第 788 页的『fcm_tq_sends_total -“FCM 表队列发送总次数”监视元素』
第 784 页的『fcm_tq_recvs_total -“FCM 表队列接收总量”监视元素』
第 786 页的『fcm_tq_send_volume -“FCM 表队列发送量”监视元素』
第 782 页的『fcm_tq_recv_volume -“FCM 表队列接收量”监视元素』
第 1311 页的『tq_tot_send_spills -“溢出表队列缓冲区总数”监视元素』
第 1079 页的『post_threshold_sorts -“超出阈值后的排序次数”监视元素』
第 1073 页的『post_shrthreshold_sorts -“共享阈值后排序数”监视元素』
第 1149 页的『sort_overflows -“排序溢出数”监视元素』
第 628 页的『audit_events_total -“审计事件总数”监视元素』

第 1292 页的『total_sorts -“排序总数”监视元素』

第 1168 页的『stmt_exec_time -“语句执行时间”监视元素』

第 696 页的『coord_stmt_exec_time -“协调代理程序执行语句的时间”监视元素』

第 1274 页的『total_routine_non_sect_proc_time -“非部分处理时间” 监视元素』

第 1274 页的『total_routine_non_sect_time -“非部分例程执行时间” 监视元素』

第 1284 页的『total_section_proc_time -“部分处理时间总计”监视元素』

第 1235 页的『total_app_section_executions -“应用程序执行部分执行的总次数”监视元素』

第 1289 页的『total_section_time -“部分时间总计”监视元素』

第 1276 页的『total_routine_user_code_proc_time -“例程用户代码处理时间总计”监视元素』

第 1278 页的『total_routine_user_code_time -“例程用户代码时间总计”监视元素』

第 1275 页的『total_routine_time -“例程时间总计”监视元素』

第 1223 页的『thresh_violations -“阈值违例次数”监视元素』

第 917 页的『num_lw_thresh_exceeded -“超过锁定等待阈值的次数”监视元素』

第 1272 页的『total_routine_invocations -“例程调用总计”监视元素』

第 865 页的『lock_wait_time_global -“锁定等待时间全局”监视元素』

第 869 页的『lock_waits_global -“锁定等待全局”监视元素』

第 1099 页的『reclaim_wait_time -“回收等待时间”监视元素』

第 1160 页的『spacemappage_reclaim_wait_time -“空间映射页回收等待时间”监视元素』

第 861 页的『lock_timeouts_global -“锁定超时全局”监视元素』

第 850 页的『lock_escals_maxlocks -“maxlocks 锁定升级数”监视元素』

第 849 页的『lock_escals_locklist -“locklist 锁定升级数”监视元素』

第 848 页的『lock_escals_global -“全局锁定升级数”监视元素』

第 652 页的『cf_wait_time -“集群高速缓存设施等待时间”监视元素』

第 651 页的『cf_waits -“集群高速缓存设施等待次数”监视元素』

第 978 页的『pool_data_gbp_l_reads -“组缓冲池数据逻辑读取数”监视元素』

第 979 页的『pool_data_gbp_p_reads -“组缓冲池数据物理读取数”监视元素』

第 981 页的『pool_data_lbp_pages_found -“本地缓冲池发现的数据页数”监视元素』

第 977 页的『pool_data_gbp_invalid_pages -“组缓冲池无效数据页数”监视元素』

第 1008 页的『pool_index_gbp_l_reads -“组缓冲池索引逻辑读取数”监视元素』

第 1009 页的『pool_index_gbp_p_reads -“组缓冲池索引物理读取数”监视元素』

第 1010 页的『pool_index_lbp_pages_found -“发现的本地缓冲池索引页数”监视元素』

第 1006 页的『pool_index_gbp_invalid_pages -“组缓冲池无效索引页数”监视元素』

第 1062 页的『pool_xda_gbp_l_reads -“组缓冲池 XDA 数据逻辑读取请求数”监视元素』

第 1064 页的『pool_xda_gbp_p_reads -“组缓冲池 XDA 数据物理读取请求数”监视元素』

第 1067 页的『pool_xda_lbp_pages_found -“发现的本地缓冲池 XDA 数据页数”监视元素』

第 1061 页的『pool_xda_gbp_invalid_pages -“组缓冲池无效 XDA 数据页数”监视元素』

第 758 页的『evmon_wait_time -“事件监视器等待时间”监视元素』

第 760 页的『evmon_waits_total -“事件监视器总等待次数”监视元素』

第 1253 页的『total_extended_latch_wait_time -“扩展锁存器等待时间总计”监视元素』

第 1254 页的『total_extended_latch_waits -“扩展锁存器等待总计”监视元素』

第 1251 页的『total_disp_run_queue_time -“分派器运行队列时间总计”监视元素』

第 1021 页的『pool_queued_async_data_reqs -“数据预取请求数”监视元素』

第 1025 页的『pool_queued_async_index_reqs -“索引预取请求数”监视元素』

第 1041 页的『pool_queued_async_xda_reqs -“XDA 预取请求数”监视元素』

第 1019 页的『pool_queued_async_data_pages -“预取请求的数据页数”监视元素』

第 1023 页的『pool_queued_async_index_pages -“预取请求的索引页数”监视元素』

第 1039 页的『pool_queued_async_xda_pages -“预取请求的 XDA 页数”监视元素』

pkgcache_stmt_args 逻辑数据组

第 754 页的『event_id -“事件标识”监视元素』

第 755 页的『event_timestamp -“事件时间戳记”监视元素』

第 1182 页的『stmt_value_index -“值索引”』

第 1183 页的『stmt_value_isreopt -“用于语句重新优化的变量”监视元素』

第 1182 页的『stmt_value_isnull -“包含空值”监视元素』

第 1184 页的『stmt_value_type -“值类型”监视元素』

第 1181 页的『stmt_value_data -“值数据”』

第 896 页的『member -“数据库成员”监视元素』

regvar 逻辑数据组

第 754 页的『event_id -“事件标识”监视元素』

第 755 页的『event_timestamp -“事件时间戳记”监视元素』

第 896 页的『member -“数据库成员”监视元素』

第 756 页的『event_type -“事件类型”监视元素』

第 1101 页的『regvar_name -“注册表变量名称”』

第 1101 页的『regvar_value -“注册表变量值”』

第 1101 页的『regvar_old_value -“注册表变量旧值”』

第 1100 页的『regvar_level -“注册表变量级别”』

第 1100 页的『regvar_collection_type -“注册表变量收集类型”』

sqlca 逻辑数据组

sqlcabc

sqlcaid

sqlcode

sqlerrd

sqlerrmc

sqlerrml

sqlerrp

sqlstate

sqlwarn

txncompletion 逻辑数据组

第 754 页的『event_id -“事件标识”监视元素』

第 755 页的『event_timestamp -“事件时间戳记”监视元素』

第 896 页的『member -“数据库成员”监视元素』

第 756 页的『event_type -“事件类型”监视元素』

第 794 页的『global_transaction_id -“全局事务标识”监视元素』

第 841 页的『local_transaction_id -“本地事务标识”监视元素』

第 1125 页的『savepoint_id -“保存点标识”』

第 1316 页的『uow_id -“工作单元标识”监视元素』

第 725 页的『ddl_classification -“DDL 分类”』

第 1313 页的『txn_completion_status -“事务完成状态”』

uow 逻辑数据组

第 951 页的『partition_key -“分区键”监视元素』

第 625 页的『application_handle -“应用程序句柄”监视元素』

第 616 页的『appl_id -“应用程序标识”监视元素』

第 620 页的『appl_name -“应用程序名称”监视元素』

第 657 页的『client_acctng -“客户机记帐字符串”监视元素』

第 658 页的『client_applname -“客户机应用程序名称”监视元素』

第 659 页的『client_hostname -“客户机主机名”监视元素』

第 661 页的『client_pid -“客户机进程标识”监视元素』

第 662 页的『client_port_number -“客户机端口号”监视元素』

第 663 页的『client_prdid -“客户机产品和版本标识”监视元素』

第 664 页的『client_userid -“客户机用户标识”监视元素』

第 665 页的『client_wrkstnname -“客户机工作站名称”监视元素』

第 670 页的『completion_status -“完成状态”监视元素』

第 682 页的『conn_time -“数据库连接时间”监视元素』

第 695 页的『coord_member -“协调程序成员”监视元素』

第 754 页的『event_id -“事件标识”监视元素』

第 755 页的『event_timestamp -“事件时间戳记”监视元素』

第 762 页的『executable_list_size -“可执行列表大小”监视元素』

第 794 页的『global_transaction_id -“全局事务标识”监视元素』

第 827 页的『intra_parallel_state -“分区内并行性的当前状态”监视元素』

第 841 页的『local_transaction_id -“本地事务标识”监视元素』

第 896 页的『member -“数据库成员”监视元素』

第 717 页的『db_conn_time -“数据库激活时间戳记”监视元素』

第 905 页的『mon_interval_id -“监视时间间隔标识”监视元素』

第 941 页的『package_list_exceeded -“超过了程序包列表的容量”监视元素』

第 941 页的『package_list_size -“程序包列表大小”监视元素』

service_class_id - 服务类标识

service_subclass_name - 服务子类名

service_superclass_name - 服务超类名

第 1137 页的『session_auth_id -“会话授权标识”监视元素』

start_time - 事件启动时间

stop_time - 事件停止时间

第 1188 页的『system_auth_id -“系统授权标识”监视元素』

UOW_CLIENT_PLATFORM

UOW_CLIENT_PROTOCOL

uow_id - 工作单元标识

第 1318 页的『uow_log_space_used -“使用的工作单元日志空间”监视元素』

workload_id - 工作负载标识

workload_name - 工作负载名称

workload_occurrence_id - 工作负载项标识

第 662 页的『client_platform -“客户机操作平台”监视元素』

第 664 页的『client_protocol -“客户机通信协议”监视元素』

第 763 页的『executable_list_truncated -“截断可执行列表”监视元素』

第 683 页的『connection_start_time -“连接开始时间”监视元素』

uow_executable_list 逻辑数据组

第 951 页的『partition_key -“分区键”监视元素』

第 616 页的『appl_id -“应用程序标识”监视元素』

第 762 页的『executable_id -“可执行文件标识”监视元素』

第 863 页的『lock_wait_time -“等待锁定时间”监视元素』

第 868 页的『lock_waits -“等待锁定次数”监视元素』

第 911 页的『num_executions -“语句执行次数”监视元素』

第 951 页的『partition_number -“分区号”』

第 1073 页的『post_shrthreshold_sorts -“共享阈值后排序数”监视元素』

第 1079 页的『post_threshold_sorts -“超出阈值后的排序次数”监视元素』

第 1118 页的『rows_read -“读取行数”监视元素』

第 1149 页的『sort_overflows -“排序溢出数”监视元素』

第 1231 页的『total_act_time -“活动时间总计”监视元素』

第 1232 页的『total_act_wait_time -“活动等待时间总计”监视元素』

第 1249 页的『total_cpu_time -“CPU 时间总计”监视元素』

第 1292 页的『total_sorts -“排序总数”监视元素』

uow_id - 工作单元标识

uow_metrics 逻辑数据组

第 951 页的『partition_key -“分区键”监视元素』

第 616 页的『appl_id -“应用程序标识”监视元素』

第 951 页的『partition_number -“分区号”』

第 1316 页的『uow_id -“工作单元标识”监视元素』

第 1337 页的『wlm_queue_time_total -“工作负载管理器队列时间总计”监视元素』

第 1336 页的『wlm_queue_assignments_total -“工作负载管理器队列分配总次数”监视元素』

第 783 页的『fcm_tq_recv_wait_time -“FCM 表队列接收等待时间”监视元素』

第 768 页的『fcm_message_recv_wait_time -“接收 FCM 消息等待时间”监视元素』

第 787 页的『fcm_tq_send_wait_time -“FCM 表队列发送等待时间”监视元素』

第 772 页的『fcm_message_send_wait_time -“发送 FCM 消息等待时间”监视元素』

第 605 页的『agent_wait_time -“代理程序等待时间”监视元素』

第 606 页的『agent_waits_total -“等待代理程序总次数”监视元素』

第 863 页的『lock_wait_time -“等待锁定时间”监视元素』

第 868 页的『lock_waits -“等待锁定次数”监视元素』

第 737 页的『direct_read_time -“直接读时间”监视元素』

第 735 页的『direct_read_reqs -“直接读请求数”监视元素』

第 742 页的『direct_write_time -“直接写时间”监视元素』

第 740 页的『direct_write_reqs -“直接写请求数”监视元素』

第 872 页的『log_buffer_wait_time -“日志缓冲区等待时间”监视元素』

第 913 页的『num_log_buffer_full -“日志缓冲区变满而导致代理程序等待的次数”监视元素』

第 874 页的『log_disk_wait_time -“日志磁盘等待时间”监视元素』

第 875 页的『log_disk_waits_total -“日志磁盘等待总次数”监视元素』

第 1218 页的『tcpip_recv_wait_time -“TCP/IP 接收等待时间”监视元素』

第 1219 页的『tcpip_recvs_total -“TCP/IP 接收总次数”监视元素』

第 660 页的『client_idle_wait_time -“客户机空闲等待时间”监视元素』

第 829 页的『ipc_recv_wait_time -“进程间通信接收等待时间”监视元素』

第 830 页的『ipc_recvs_total -“进程间通信接收总次数”监视元素』

第 831 页的『ipc_send_wait_time -“进程间通信发送等待时间”监视元素』

第 832 页的『ipc_sends_total -“进程间通信发送总次数”监视元素』

第 1221 页的『tcpip_send_wait_time -“TCP/IP 发送等待时间”监视元素』

第 1222 页的『tcpip_sends_total -“TCP/IP 发送总次数”监视元素』

第 1058 页的『pool_write_time -“缓冲池物理写时间总计”监视元素』

第 1043 页的『pool_read_time -“缓冲池物理读时间总计”监视元素』

第 629 页的『audit_file_write_wait_time -“审计文件写等待时间”监视元素』

第 631 页的『audit_file_writes_total -“写审计文件总次数”监视元素』

第 632 页的『audit_subsystem_wait_time -“审计子系统等待时间”监视元素』

第 634 页的『audit_subsystem_waits_total -“审计子系统等待总次数”监视元素』

第 732 页的『diaglog_write_wait_time -“诊断日志文件写等待时间”监视元素』
第 734 页的『diaglog_writes_total -“写诊断日志文件总次数”监视元素』
第 779 页的『fcm_send_wait_time -“FCM 发送等待时间”监视元素』
第 775 页的『fcm_recv_wait_time -“FCM 接收等待时间”监视元素』
第 1302 页的『total_wait_time -“等待时间总计”监视元素』
第 1124 页的『rqsts_completed_total -“完成请求总数”监视元素』
第 1280 页的『total_rqst_time -“请求时间总计”监视元素』
第 615 页的『app_rqsts_completed_total -“完成应用程序请求总数”监视元素』
第 1235 页的『total_app_rqst_time -“应用程序请求时间总计”监视元素』
第 1285 页的『total_section_sort_proc_time -“节排序处理时间总计”监视元素』
第 1288 页的『total_section_sorts -“节排序总次数”监视元素』
第 1287 页的『total_section_sort_time -“节排序时间总计”监视元素』
第 1118 页的『rows_read -“读取行数”监视元素』
第 1117 页的『rows_modified -“修改的行数”监视元素』
第 982 页的『pool_data_l_reads -“缓冲池数据逻辑读取数”监视元素』
第 1012 页的『pool_index_l_reads -“缓冲池索引逻辑读取数”监视元素』
第 1046 页的『pool_temp_data_l_reads -“缓冲池临时数据逻辑读取数”监视元素』
第 1050 页的『pool_temp_index_l_reads -“缓冲池临时索引逻辑读取数”监视元素』
第 1065 页的『pool_xda_l_reads -“缓冲池 XDA 数据逻辑读取数”监视元素』
第 1053 页的『pool_temp_xda_l_reads -“缓冲池临时 XDA 数据逻辑读取数”监视元素』
第 1249 页的『total_cpu_time -“CPU 时间总计”监视元素』
第 589 页的『act_completed_total -“完成活动总数”监视元素』
第 984 页的『pool_data_p_reads -“缓冲池数据物理读取数”监视元素』
第 1048 页的『pool_temp_data_p_reads -“缓冲池临时数据物理读取数”监视元素』
第 1069 页的『pool_xda_p_reads -“缓冲池 XDA 数据物理读取数”监视元素』
第 1055 页的『pool_temp_xda_p_reads -“缓冲池临时 XDA 数据物理读取数”监视元素』
第 1014 页的『pool_index_p_reads -“缓冲池索引物理读取数”监视元素』
第 1052 页的『pool_temp_index_p_reads -“缓冲池临时索引物理读取数”监视元素』
第 986 页的『pool_data_writes -“缓冲池数据写次数”监视元素』
第 1071 页的『pool_xda_writes -“缓冲池 XDA 数据写次数”监视元素』
第 1016 页的『pool_index_writes -“缓冲池索引写次数”监视元素』
第 738 页的『direct_reads -“直接读数据库数目”监视元素』
第 744 页的『direct_writes -“直接写数据库数目”监视元素』
第 1120 页的『rows_returned -“返回的行数”监视元素』
第 728 页的『deadlocks -“检测到的死锁数”监视元素』
第 860 页的『lock_timeouts -“锁定超时次数”监视元素』
第 845 页的『lock_escals -“锁定升级次数”监视元素』
第 780 页的『fcm_sends_total -“FCM 发送总计”监视元素』
第 777 页的『fcm_recvs_total -“FCM 接收总计”监视元素』

第 778 页的『fcm_send_volume -“FCM 发送量”监视元素』
第 774 页的『fcm_recv_volume -“FCM 接收量”监视元素』
第 773 页的『fcm_message_sends_total -“发送 FCM 消息总数”监视元素』
第 769 页的『fcm_message_recvs_total -“接收 FCM 消息总数”监视元素』
第 770 页的『fcm_message_send_volume -“发送 FCM 消息量”监视元素』
第 766 页的『fcm_message_recv_volume -“接收 FCM 消息量”监视元素』
第 788 页的『fcm_tq_sends_total -“FCM 表队列发送总次数”监视元素』
第 784 页的『fcm_tq_recvs_total -“FCM 表队列接收总量”监视元素』
第 786 页的『fcm_tq_send_volume -“FCM 表队列发送量”监视元素』
第 782 页的『fcm_tq_recv_volume -“FCM 表队列接收量”监视元素』
第 1311 页的『tq_tot_send_spills -“溢出表队列缓冲区总数”监视元素』
第 1220 页的『tcpip_send_volume -“TCP/IP 发送量”监视元素』
第 1217 页的『tcpip_recv_volume -“TCP/IP 接收量”监视元素』
第 831 页的『ipc_send_volume -“进程间通信发送量”监视元素』
第 828 页的『ipc_recv_volume -“进程间通信接收量”监视元素』
第 1079 页的『post_threshold_sorts -“超出阈值后的排序次数”监视元素』
第 1073 页的『post_shrthreshold_sorts -“共享阈值后排序数”监视元素』
第 1149 页的『sort_overflows -“排序溢出数”监视元素』
第 628 页的『audit_events_total -“审计事件总数”监视元素』
第 591 页的『act_rejected_total -“被拒绝活动总数”监视元素』
第 588 页的『act_aborted_total -“异常终止活动总数”监视元素』
第 1292 页的『total_sorts -“排序总数”监视元素』
第 1275 页的『total_routine_time -“例程时间总计”监视元素』
第 1241 页的『total_compile_proc_time -“编译处理时间总计”监视元素』
第 1240 页的『total_compilations -“编译次数总计”监视元素』
第 1242 页的『total_compile_time -“编译时间总计”监视元素』
第 1258 页的『total_implicit_compile_proc_time -“隐式编译处理时间总计”监视元素』
第 1257 页的『total_implicit_compilations -“隐式编译总数”监视元素』
第 1259 页的『total_implicit_compile_time -“隐式编译时间总计”监视元素』
第 1282 页的『total_runstats_proc_time -“运行时统计信息处理时间总计”监视元素』
第 1281 页的『total_runstats -“运行时统计信息总计”监视元素』
第 1283 页的『total_runstats_time -“运行时统计信息时间总计”监视元素』
第 1268 页的『total_reorg_proc_time -“重组处理时间总计”监视元素』
第 1270 页的『total_reorgs -“重组操作总数”监视元素』
第 1269 页的『total_reorg_time -“重组时间总计”监视元素』
第 1260 页的『total_load_proc_time -“装入处理时间总计”监视元素』
第 1262 页的『total_loads -“装入操作总数”监视元素』
第 1261 页的『total_load_time -“装入时间总计”监视元素』
第 1284 页的『total_section_proc_time -“部分处理时间总计”监视元素』

第 1235 页的『total_app_section_executions -“应用程序执行部分执行的总次数”监视元素』

第 1289 页的『total_section_time -“部分时间总计”监视元素』

第 1238 页的『total_commit_proc_time -“落实处理时间总计”监视元素』

第 1233 页的『total_app_commits -“应用程序落实次数总计”监视元素』

第 1239 页的『total_commit_time -“落实时间总计”监视元素』

第 1270 页的『total_rollback_proc_time -“回滚处理时间总计”监视元素』

第 1234 页的『total_app_rollbacks -“应用程序回滚次数总计”监视元素』

第 1271 页的『total_rollback_time -“回滚时间总计”监视元素』

第 1276 页的『total_routine_user_code_proc_time -“例程用户代码处理时间总计”监视元素』

第 1278 页的『total_routine_user_code_time -“例程用户代码时间总计”监视元素』

第 1223 页的『thresh_violations -“阈值违例次数”监视元素』

第 917 页的『num_lw_thresh_exceeded -“超过锁定等待阈值的次数”监视元素』

第 1272 页的『total_routine_invocations -“例程调用总计”监视元素』

第 821 页的『int_commits -“内部落实数”监视元素』

第 823 页的『int_rollbacks -“内部回滚数”监视元素』

第 646 页的『cat_cache_inserts -“目录高速缓存插入数”监视元素』

第 647 页的『cat_cache_lookups -“目录高速缓存查询数”监视元素』

第 955 页的『pkg_cache_inserts -“程序包高速缓存插入数”监视元素』

第 956 页的『pkg_cache_lookups -“程序包高速缓存查询数”监视元素』

第 593 页的『act_rqsts_total -“活动请求总数”监视元素』

第 1232 页的『total_act_wait_time -“活动等待时间总计”监视元素』

第 1231 页的『total_act_time -“活动时间总计”监视元素』

第 865 页的『lock_wait_time_global -“锁定等待时间全局”监视元素』

第 869 页的『lock_waits_global -“锁定等待全局”监视元素』

第 1099 页的『reclaim_wait_time -“回收等待时间”监视元素』

第 1160 页的『spacemappage_reclaim_wait_time -“空间映射页回收等待时间”监视元素』

第 861 页的『lock_timeouts_global -“锁定超时全局”监视元素』

第 850 页的『lock_escals_maxlocks -“maxlocks 锁定升级数”监视元素』

第 849 页的『lock_escals_locklist -“locklist 锁定升级数”监视元素』

第 848 页的『lock_escals_global -“全局锁定升级数”监视元素』

第 652 页的『cf_wait_time -“集群高速缓存设施等待时间”监视元素』

第 651 页的『cf_waits -“集群高速缓存设施等待次数”监视元素』

第 978 页的『pool_data_gbp_l_reads -“组缓冲池数据逻辑读取数”监视元素』

第 979 页的『pool_data_gbp_p_reads -“组缓冲池数据物理读取数”监视元素』

第 981 页的『pool_data_lbp_pages_found -“本地缓冲池发现的数据页数”监视元素』

第 977 页的『pool_data_gbp_invalid_pages -“组缓冲池无效数据页数”监视元素』

第 1008 页的『pool_index_gbp_l_reads -“组缓冲池索引逻辑读取数”监视元素』

第 1009 页的『pool_index_gbp_p_reads -“组缓冲池索引物理读取数”监视元素』

第 1010 页的『pool_index_lbp_pages_found -“发现的本地缓冲池索引页数”监视元素』

第 1006 页的『pool_index_gbp_invalid_pages -“组缓冲池无效索引页数”监视元素』

第 1062 页的『pool_xda_gbp_l_reads -“组缓冲池 XDA 数据逻辑读取请求数”监视元素』

第 1064 页的『pool_xda_gbp_p_reads -“组缓冲池 XDA 数据物理读取请求数”监视元素』

第 1067 页的『pool_xda_lbp_pages_found -“发现的本地缓冲池 XDA 数据页数”监视元素』

第 1061 页的『pool_xda_gbp_invalid_pages -“组缓冲池无效 XDA 数据页数”监视元素』

第 758 页的『evmon_wait_time -“事件监视器等待时间”监视元素』

第 760 页的『evmon_waits_total -“事件监视器总等待次数”监视元素』

第 1253 页的『total_extended_latch_wait_time -“扩展锁存器等待时间总计”监视元素』

第 1254 页的『total_extended_latch_waits -“扩展锁存器等待总计”监视元素』

第 1293 页的『total_stats_fabrication_proc_time -“统计信息生成处理时间总计”监视元素』

第 1295 页的『total_stats_fabrications -“统计信息生成总计”监视元素』

第 1294 页的『total_stats_fabrication_time -“统计信息生成时间总计”监视元素』

第 1298 页的『total_sync_runstats_proc_time -“同步 RUNSTATS 处理时间总计”监视元素』

第 1299 页的『total_sync_runstats -“同步 RUNSTATS 活动总数”监视元素』

第 1296 页的『total_sync_runstats_time -“同步 RUNSTATS 时间总计”监视元素』

第 1251 页的『total_disp_run_queue_time -“分派器运行队列时间总计”监视元素』

第 1021 页的『pool_queued_async_data_reqs -“数据预取请求数”监视元素』

第 1025 页的『pool_queued_async_index_reqs -“索引预取请求数”监视元素』

第 1041 页的『pool_queued_async_xda_reqs -“XDA 预取请求数”监视元素』

第 1019 页的『pool_queued_async_data_pages -“预取请求的数据页数”监视元素』

第 1023 页的『pool_queued_async_index_pages -“预取请求的索引页数”监视元素』

第 1039 页的『pool_queued_async_xda_pages -“预取请求的 XDA 页数”监视元素』

第 613 页的『app_act_completed_total -“成功的外部协调程序活动总数”监视元素』

第 612 页的『app_act_aborted_total -“失败的外部协调程序活动总数”监视元素』

第 614 页的『app_act_rejected_total -“拒绝的外部协调程序活动总数”监视元素』

uow_package_list 逻辑数据组

第 951 页的『partition_key -“分区键”监视元素』

第 616 页的『appl_id -“应用程序标识”监视元素』

第 827 页的『invocation_id -“调用标识”监视元素』

第 906 页的『nesting_level -“嵌套级别”监视元素』

第 941 页的『package_elapsed_time -“程序包耗用时间”监视元素』

第 940 页的『package_id -“程序包标识”监视元素』

第 1115 页的『routine_id -“例程标识”监视元素』

第 1316 页的『uow_id -“工作单元标识”监视元素』

第 896 页的『member -“数据库成员”监视元素』

utillocation 逻辑数据组

第 754 页的『event_id -“事件标识”监视元素』

第 755 页的『event_timestamp -“事件时间戳记”监视元素』

第 896 页的『member -“数据库成员”监视元素』

第 756 页的『event_type -“事件类型”监视元素』

第 1328 页的『utility_invocation_id -“实用程序调用标识”』

第 1333 页的『utility_type -“实用程序类型”』

第 732 页的『device_type -“设备类型”』

第 841 页的『location_type -“位置类型”』

第 841 页的『location -“位置”』

utilphase 逻辑数据组

第 754 页的『event_id -“事件标识”监视元素』

第 755 页的『event_timestamp -“事件时间戳记”监视元素』

第 896 页的『member -“数据库成员”监视元素』

第 756 页的『event_type -“事件类型”监视元素』

第 1328 页的『utility_invocation_id -“实用程序调用标识”』

第 1333 页的『utility_type -“实用程序类型”』

第 1330 页的『utility_phase_type -“实用程序阶段类型”』

第 953 页的『phase_start_event_id -“阶段开始事件标识”』

第 953 页的『phase_start_event_timestamp -“阶段开始事件时间戳记”』

第 932 页的『objtype -“对象类型”监视元素』

第 928 页的『object_schema -“对象模式”监视元素』

第 927 页的『object_name -“对象名”监视元素』

第 1330 页的『utility_phase_detail -“实用程序阶段详细信息”』

utilstart 逻辑数据组

第 754 页的『event_id -“事件标识”监视元素』

第 755 页的『event_timestamp -“事件时间戳记”监视元素』

第 896 页的『member -“数据库成员”监视元素』

第 756 页的『event_type -“事件类型”监视元素』

第 1328 页的『utility_invocation_id -“实用程序调用标识”』

第 1333 页的『utility_type -“实用程序类型”』

第 1329 页的『utility_operation_type -“实用程序操作类型”』

第 1328 页的『utility_invoker_type -“实用程序调用者类型”』

第 1331 页的『utility_priority -“实用程序优先级”』

第 1331 页的『utility_start_type -“实用程序启动类型”』

第 932 页的『objtype -“对象类型”监视元素』

第 928 页的『object_schema -“对象模式”监视元素』

第 927 页的『object_name -“对象名”监视元素』

第 919 页的『num_tbsps -“表空间数”监视元素』

第 1216 页的『tbsp_names -“表空间名称”』

第 1327 页的『utility_detail -“实用程序详细信息”』

utilstop 逻辑数据组

第 754 页的『event_id -“事件标识”监视元素』

第 755 页的『event_timestamp -“事件时间戳记”监视元素』

第 896 页的『member -“数据库成员”监视元素』

第 756 页的『event_type -“事件类型”监视元素』

第 1328 页的『utility_invocation_id -“实用程序调用标识”』

第 1333 页的『utility_type -“实用程序类型”』

第 1332 页的『utility_stop_type -“实用程序停止类型”』

第 1164 页的『start_event_id -“开始事件标识”』

第 1164 页的『start_event_timestamp -“开始事件时间戳记”』

第 1154 页的『sqlca -“SQL 通信区（SQLCA）”』

管理目标表、控制表和事件监视器表:

可定义事件监视器以使它将其事件记录存储在 SQL 表中。为此，应将 CREATE EVENT MONITOR 语句与 WRITE TO TABLE 子句配合使用。

如果创建“写至表”事件监视器，那么该事件监视器将创建目标表以存储每个返回数据的逻辑数据组的记录。在每个表中，列名必须与它们表示的监视元素名称相匹配。缺省情况下，该事件监视器会在事件监视器创建者的模式下创建表，并通过将它们的对应逻辑数据组名称与事件监视器名称并置来命名这些表。

例如，考虑以下语句，它创建用于捕获 STATEMENTS 事件的事件监视器:

```
CREATE EVENT MONITOR test FOR STATEMENTS WRITE TO TABLE
```

使用 STATEMENTS 事件类型的事件监视器从 event_connheader、event_stmt 和 event_subsection 逻辑数据组收集数据。系统会创建表示特定于各事件类型的逻辑数据组的表，并为每个“写至表”事件监视器创建一个控制表。对于用户 `riihi` 创建的事件监视器 `test`，数据库管理器会创建以下表:

- riihi.connheader_test
- riihi.stmt_test
- riihi.subsection_test
- riihi.control_test

前三个表分别对应于逻辑数据组 event_connheader、event_stmt 和 event_subsection 中的每一个。最后一个表 riihi.control_test 是控制表。具体来说，控制表中包含来自 event_start、event_dbheader（仅适用于 conn_time 监视元素）和 event_overflow 逻辑数据组的事件监视器元数据。

仅对于*非受阻*事件监视器，才会将监视元素写至溢出组。对于非分块事件监视器，如果事件缓冲区已满，那么生成事件的代理程序不会等待将事件缓冲区写入表。如果从

代理程序接收监视器数据的速度超过了事件监视器写入数据的速度，那么生成事件的代理程序将废弃这些监视器数据。在这种情况下，事件监视器会在控制表中记录信息，以指出发生了溢出。此信息中包括 **message** 监视元素，如果发生了溢出，那么此监视元素中包含文本 OVERFLOW:n，其中 *n* 表示由于事件缓冲区已满而被废弃的事件记录数。

每当“写至表”事件监视器激活时，它都会获取针对每个目标表的 IN 或 IX 表锁定，以避免事件监视器处于活动状态时修改该表。在事件监视器处于活动状态时会维护所有表上的表锁定。如果需要针对任何目标表的互斥访问（例如，以运行实用程序），那么系统会在尝试这类访问前取消激活事件监视器以释放表锁定。

目标表中的每个列名与一个事件监视元素标识相匹配。将忽略没有相应目标表列的事件监视元素。

必须以手动方式修剪“写至表”事件监视器的目标表，包括无格式事件 (UE) 表。在很多时间处于活动状态的系统上，事件监视器会迅速填满磁盘空间，因为它们记录的数据量很大。与定义那些写至文件或命名管道的事件监视器不同，可定义“写至表”事件监视器以仅记录某些逻辑数据组或监视元素中的信息。可使用此功能以仅收集所需数据，从而减少事件监视器生成的数据量。例如，以下语句定义一个事件监视器，该事件监视器仅捕获来自 event_conn 逻辑数据组的连接事件，并且仅包括 **lock_waits** 监视元素:

```
CREATE EVENT MONITOR conn_monitor FOR CONNECTIONS WRITE TO TABLE
                    CONN(INCLUDES(lock_waits))
```

您可能不希望缺省表空间内的缺省模式中包括某个事件监视器的具有缺省表名的目标表。如果预期会有大量监视数据时，那么您可能想要目标表放在它们自己的表空间中。可对 CREATE EVENT MONITOR 语句指定模式名、表名和表空间名。模式名和表名形成表的派生名称。可使用可选 IN 子句在表名后添加表空间名。与 DB2 数据库管理器自动创建的目标表不同，如果事件监视器定义中包含某个表空间，那么该表空间必须已存在。如果未指定表空间，那么系统将指定您对其具有 USE 特权的表空间。

一个目标表只能由一个事件监视器使用。如果对另一个事件监视器定义了目标表，或者因为任何其他原因而不能创建该目标表，那么 CREATE EVENT MONITOR 语句失败。

可使用可选 IN 子句在表名后添加表空间名。与 DB2 数据库管理器自动创建的目标表不同，如果事件监视器定义中包含某个表空间，那么该表空间必须已存在。如果没有指定表空间，那么将指定定义者对其具有 USE 特权的表空间。

在分区数据库环境中，“写至表”事件监视器只有在包含该事件监视器表的表空间所在的数据库分区上才处于活动状态。如果特定数据库分区上没有处于活动状态的事件监视器的目标表空间，那么系统将对该数据库分区取消激活此事件监视器，同时会向 **db2diag** 命令日志文件写入错误。

为了在检索事件监视器数据时提高性能，可为事件表创建索引。如果添加了触发器、关系完整性和约束之类的表属性，那么该事件监视器会将它们忽略。

例如，以下语句定义一个事件监视器，该事件监视器使用 event_connheader、event_stmt 和 event_subsection 逻辑数据组来捕获 STATEMENTS 事件。三个目标表中的每一个都有不同的模式、表和表空间组合:

```
CREATE EVENT MONITOR test FOR STATEMENTS
WRITE TO TABLE CONNHEADER,
STMT (TABLE mydept.statements),
SUBSECTION (TABLE subsections, IN mytablespace)
```

假定用户 riihi 发出了上述语句，那么目标表的派生名称和表空间如下所示:

- CONNHEADER: 缺省表空间中的 riihi.connheader_test
- STMT: 缺省表空间中的 mydept.statements
- SUBSECTION: mytablespace 表空间中的 riihi.subsections

如果事件监视器激活时目标表不存在，那么激活将继续进行，并且会忽略本来会插入至目标表的数据。相应的，如果监视元素在目标表中没有专用的列，就会被忽略。

对于活动的“写至表”事件监视器，可能会有存储事件记录的表空间用完容量的风险。为针对 DMS 表空间控制此风险，可定义一个百分比，表空间容量达到此百分比时会取消激活事件监视器。可在 CREATE EVENT MONITOR 语句的 PCTDEACTIVATE 子句中指定此值。对于 SMS 表空间，此值设置为 100。如果已对目标表空间启用 Autoresize 功能，那么应将 PCTDEACTIVATE 值设置为 100。

在非分区数据库环境中，所有“写至表”事件监视器都会在最后一个应用程序终止（并且数据库未显式激活）时取消激活。在分区数据库环境中，“写至表”事件监视器将在目录分区取消激活时取消激活。

逻辑数据组和事件监视器输出表:

经常一起使用的监视元素被划分到*逻辑数据组*中。写至表的事件监视器通常为它们捕获的监视元素的每个逻辑数据组生成一个输出表。

下表按事件类型提供缺省目标表名。

表 7. 写至表事件监视器逻辑数据组

事件类型	逻辑数据组	逻辑组中的信息	属于逻辑组的元素写至的表的名称
DEADLOCKS[1]	event_connheader	连接元数据。	CONNHEADER_*evmon-name*
	event_deadlock	死锁数据。	DEADLOCK_*evmon-name*
	event_dlconn	死锁涉及的应用程序和锁定。	DLCONN_*evmon-name*
	CONTROL[2]	事件监视器元数据。	CONTROL_*evmon-name*
DEADLOCKS WITH DETAILS[1]	event_connheader	连接元数据。	CONNHEADER_*evmon-name*
	event_deadlock	死锁数据。	DEADLOCK_*evmon-name*
	event_detailed_dlconn	死锁涉及的应用程序。	DLCONN_*evmon-name*
	dllock	死锁涉及的锁定。	DLLOCK_*evmon-name*
	CONTROL[2]	事件监视器元数据。	CONTROL_*evmon-name*
DEADLOCKS WITH DETAILS HISTORY[1]	event_connheader	连接元数据。	CONNHEADER_*evmon-name*
	event_deadlock	死锁数据。	DEADLOCK_*evmon-name*
	event_detailed_dlconn	死锁涉及的应用程序。	DLCONN_*evmon-name*
	dllock	死锁涉及的锁定。	DLLOCK_*evmon-name*
	event_stmt	工作单元中的先前语句列表。	STMTHIST_*evmon-name*
	CONTROL[2]	事件监视器元数据。	CONTROL_*evmon-name*

表 7. 写至表事件监视器逻辑数据组 (续)

事件类型	逻辑数据组	逻辑组中的信息	属于逻辑组的元素写至的表的名称
DEADLOCKS WITH DETAILS HISTORY VALUES[1]	event_connheader	连接元数据。	CONNHEADER_*evmon-name*
	event_deadlock	死锁数据。	DEADLOCK_*evmon-name*
	event_detailed_dlconn	死锁涉及的应用程序。	DLCONN_*evmon-name*
	dllock	死锁涉及的锁定。	DLLOCK_*evmon-name*
	event_stmt_history	工作单元中的先前语句列表。	STMTHIST_*evmon-name*
	STMTVALS	STMTHIST 表中的语句的输入数据值。	STMTVALS_*evmon-name*
	CONTROL[2]	事件监视器元数据。	CONTROL_*evmon-name*
STATEMENT	event_connheader	连接元数据。	CONNHEADER_*evmon-name*
	event_stmt	语句数据。	STMT_*evmon-name*
	event_subsection	特定于子节的语句数据。	SUBSECTION_*evmon-name*
	CONTROL[2]	事件监视器元数据。	CONTROL_*evmon-name*
TRANSACTIONS[3]	event_connheader	连接元数据。	CONNHEADER_*evmon-name*
	event_xact	事务数据。	XACT_*evmon-name*
	CONTROL[2]	事件监视器元数据。	CONTROL_*evmon-name*
CONNECTIONS	event_connheader	连接元数据。	CONNHEADER_*evmon-name*
	event_conn	连接数据。	CONN_*evmon-name*
	CONTROL[2]	事件监视器元数据。	CONTROL_*evmon-name*
	event_connmemuse	内存池元数据。	CONNMEMUSE_*evmon-name*
DATABASE	event_db	数据库管理器数据。	DB_*evmon-name*
	CONTROL[2]	事件监视器元数据。	CONTROL_*evmon-name*
	event_dbmemuse	内存池元数据。	DBMEMUSE_*evmon-name*
BUFFERPOOLS	event_bufferpool	缓冲池数据。	BUFFERPOOL_*evmon-name*
	CONTROL[2]	事件监视器元数据。	CONTROL_*evmon-name*
TABLESPACES	event_tablespace	表空间数据。	TABLESPACE_*evmon-name*
	CONTROL[2]	事件监视器元数据。	CONTROL_*evmon-name*
TABLES	event_table	表数据。	TABLE_*evmon-name*
	CONTROL[2]	事件监视器元数据。	CONTROL_*evmon-name*

表 7. 写至表事件监视器逻辑数据组 (续)

事件类型	逻辑数据组	逻辑组中的信息	属于逻辑组的元素写至的表的名称
ACTIVITIES	event_activity	完成执行的或执行过程中捕获的活动。	ACTIVITY_*evmon-name*
	event_activitystmt	语句活动的语句信息。	ACTIVITYSTMT_*evmon-name*
	event_activityvals	具有输入数据的活动的输入数据值。不报告以下数据类型：CLOB、REF、BOOLEAN、STRUCT、DATALINK、LONG VARGRAPHIC、LONG、XMLLOB 和 DBCLOB。	ACTIVITYVALS_*evmon-name*
	activity_metrics	活动度量值。	ACTIVITYMETRICS_*evmon-name*
	CONTROL[2]	事件监视器元数据。	CONTROL_*evmon-name*
STATISTICS	event_scstats	根据系统中每个服务类、工作类或工作负载内执行的活动计算的统计信息。	SCSTATS_*evmon-name*
	event_wcstats		WCSTATS_*evmon-name*
	event_wlstats		WLSTATS_*evmon-name*
	event_histogrambin		HISTOGRAMBIN_*evmon-name*
	event_qstats		QSTATS_*evmon-name*
	CONTROL[2]	事件监视器元数据。	CONTROL_*evmon-name*
THRESHOLD VIOLATIONS	event_thresholdviolations	违反的阈值及违例次数的列表。	THRESHOLDVIOLATIONS_*evmon-name*
	CONTROL[2]	事件监视器元数据。	CONTROL_*evmon-name*
LOCKING	lock	锁定等待、锁定超时或死锁事件信息的摘要。	LOCK_EVENT*evmon-name*
	lock_participants	有关锁定参与者的信息。	LOCK_PARTICIPANTS_*evmon-name*
	lock_participant_activities	每个锁定参与者的活动数据。	LOCK_PARTICIPANT_ACTIVITIES_*evmon-name*
	lock_activity_values	有关正由特定活动处理的特定数据的详细信息。	LOCK_ACTIVITY_VALUES_*evmon-name*
	CONTROL[2]	事件监视器元数据。	CONTROL_*evmon-name*
PACKAGE CACHE	pkgcache	程序包高速缓存事件信息的摘要。此信息包括 METRICS 列中的详细度量值（使用 XML 格式）。	PKGCACHE_EVENT*evmon-name*
	pkgcache_metrics	包含 PKGCACHE 表中的 METRICS 列内包括的度量值的表。	PKGCACHE_METRICS_*evmon-name*
	CONTROL[2]	事件监视器元数据。	CONTROL_*evmon-name*

表 7. 写至表事件监视器逻辑数据组 (续)

事件类型	逻辑数据组	逻辑组中的信息	属于逻辑组的元素写至的表的名称
UNIT OF WORK	uow	工作单元事件信息的摘要。此信息包括 METRICS 列中的详细度量值（使用 XML 格式）。	UOW_EVENT*evmon-name*
	uow_metrics	包含 PKGCACHE 表中的 METRICS 列内包括的度量值的表。	UOW_METRICS_*evmon-name*
	uow_package_list	包列表详细信息。[4]	UOW_PACKAGE_LIST_*evmon-name*
	uow_executable_list	可执行列表信息。[4]	UOW_EXECUTABLE_LIST_*evmon-name*
	CONTROL[2]	事件监视器元数据。	CONTROL_*evmon-name*

1 建议不要使用此选项，将来的发行版中可能会将其除去。请使用 CREATE EVENT MONITOR FOR LOCKING 语句来监视与锁定相关的事件，例如锁定超时、锁定等待和死锁。

2 CONTROL 逻辑组由来自 event_dbheader、event_start 和 event_overflow 逻辑数据组中的一个或多个的所选元素组成。

3 建议不要使用此选项，将来的发行版中可能会将其除去。请使用 CREATE EVENT MONITOR FOR UNIT OF WORK 语句来监视事务事件。

4 除非您显式指定要为工作单元事件监视器创建的输出表，否则缺省情况下会包括此表。如果未将用于收集相关信息的配置参数（**mon_uow_pkglist** 或 **mon_uow_execlist**）设置为 ON，那么会创建此表，但此表不包含任何数据。

未对“写至表”事件监视器收集下列逻辑数据组:

- log_stream_header
- log_header
- dbheader（仅收集 **conn_time** 监视元素）

事件监视器表中的每一列的数据类型对应于该列表示的监视元素的数据类型。下表包含一组数据类型映射，这些映射使监视元素（在 sqlmon.h 文件中）的原始系统监视器数据类型与表列的 SQL 数据类型相对应。

表 8. 系统监视器数据类型映射

系统监视器数据类型	SQL 数据类型
SQLM_TYPE_STRING	CHAR[n]、VARCHAR[n] 或 CLOB[n]
SQLM_TYPE_U8BIT 和 SQLM_TYPE_8BIT	SMALLINT、INTEGER 或 BIGINT
SQLM_TYPE_U16BIT 和 SQLM_TYPE_16BIT	SMALLINT、INTEGER 或 BIGINT
SQLM_TYPE_U32BIT 和 SQLM_TYPE_32BIT	INTEGER 或 BIGINT
SQLM_TYPE_U64BIT 和 SQLM_TYPE_64BIT	BIGINT
SQLM_TIMESTAMP	TIMESTAMP
SQLM_TIME	BIGINT
SQLCA: SQLERRMC	VARCHAR[72]
SQLCA: SQLSTATE	CHAR[5]
SQLCA: SQLWARN	CHAR[11]

表 8. 系统监视器数据类型映射 (续)

系统监视器数据类型	SQL 数据类型
SQLCA: 其他字段	INTEGER 或 BIGINT
SQLM_TYPE_HANDLE	BLOB[n]

注:

1. 并非所有列都是空值。
2. 因为带有 CLOB 列的表的性能低于带有 VARCHAR 列的表，所以在指定 stmt evmGroup（或在使用带有详细信息的死锁时指定 dlconn evmGroup）时考虑使用 TRUNC 关键字。
3. SQLM_TYPE_HANDLE 用于表示编译环境句柄对象。

创建写至无格式事件 (UE) 表的事件监视器

如果事件监视器数据收集的性能特别重要，那么您可选择让事件监视器将其输出写至无格式事件 (UE) 表。写至 UE 表的大部分数据作为直接插入二进制数据写入，这可以提高收集数据时的 I/O 速度。

对于事件监视器，UE 表优胜于常规表的另一点是，您通常不必在事件监视器创建时考虑不同选项，例如，要使用的缓冲区大小、是阻止还是取消阻止事件监视器或者必须收集的数据类型（逻辑组）。但是，因为大部分所收集数据为二进制格式，所以必须对 UE 表执行后处理以便能检查事件数据。

注: 从 IBM DB2 10.1 V10.1 开始，以下与 UE 表相关的功能可用:

- 可使用过程 EVMON_UPGRADE_TABLES 来升级事件监视器在之前发行版中生成的 UE 表。此功能使您能更轻松地在升级 DB2 产品时保留事件监视器数据。
- 可将输出写至 UE 表的所有事件监视器也可将输出写至常规表。
- 可使用过程 EVMON_FORMAT_UE_TO_TABLES 的选项 PRUNE_UE_TABLE 从 UE 表中剪除不必要的数据。

开始之前

创建写至无格式事件表的事件监视器时，应记住以下注意事项:

- 您需要 SQLADM 或 DBADM 权限才能创建写至 UE 表的事件监视器。
- 对无格式事件表使用已优化以提高性能的表空间。创建该表空间时，请记住以下准则:
  - 指定尽可能大的页大小 (PAGESIZE)，最大 32KB。较大的页大小可确保包含该事件数据的 BLOB 可直接写入表行。如果因为页大小太小而不允许直接插入 BLOB，那么事件监视器的性能可能会下降。数据库管理器会尝试在无格式事件表中直接插入 BLOB 列 event_data，但这并非始终有可能实现。要检查无格式事件表中的行是否直接插入型的行，请使用 ADMIN_IS_INLINED 函数。如果这些行不是直接插入型的行，请使用 ADMIN_EST_INLINE_LENGTH 函数来确定这些行所需的空间量。
  - 指定 NO FILE CACHING SYSTEM 选项。

- 在分区数据库环境中，考虑用于存放表空间的分区。如果目标无格式事件表的表空间在某个数据库分区中不存在，那么该目标无格式事件表的数据将被忽略。此行为允许用户创建只在特定数据库分区中存在的表空间，从而选择要监视的部分数据库分区。

关于此任务

以下事件监视器类型支持使用 UE 表:

- 工作单元
- 程序包高速缓存
- 锁定

注: 尽管名称不一样，但无格式事件表仍为关系表。锁定事件监视器生成的 UE 表与锁定事件监视器生成的常规表之间的主要差别在于 UE 表中的大部分数据以二进制格式写至 EVENT_DATA 列。有关 UE 表的结构的更多信息，请参阅第 89 页的『无格式事件表的列定义』。

过程

要创建写至 UE 表的事件监视器，请执行以下操作:

- 使用 WRITE TO UNFORMATTED EVENT TABLE 子句构造 CREATE EVENT MONITOR 语句。 例如，要创建名为 uowmon 的工作单元事件监视器，您可使用类似如下的语句:

  ```
  CREATE EVENT MONITOR uowmon FOR UNIT OF WORK
                      WRITE TO UNFORMATTED EVENT TABLE
  ```

  缺省情况下，该事件监视器创建的 UE 表的名称与该事件监视器的名称相同。

- 要指定缺省表名的替代项，请使用 TABLE 子句。 例如，如果要将该 UE 表命名为 myunitsofwork，请构造类似如下的语句:

  ```
  CREATE EVENT MONITOR uowmon FOR UNIT OF WORK
                      WRITE TO UNFORMATTED EVENT TABLE
                      TABLE myunitsofwork
  ```

  还可使用 IN *tablespace-name* 子句指定用于存储该 UE 表的表空间:

  ```
  CREATE EVENT MONITOR uowmon FOR UNIT OF WORK
                      WRITE TO UNFORMATTED EVENT TABLE
                      TABLE myunitsofwork
                      IN mytablespace
  ```

  或者

  ```
  CREATE EVENT MONITOR uowmon FOR UNIT OF WORK
                      WRITE TO UNFORMATTED EVENT TABLE
                      IN mytablespace
  ```

  第一个示例将 UE 表 myunitsofwork 放在表空间 mytablespace 中；第二个示例将名为 uowmon 的 UE 表（缺省情况，因为不会指定任何表名）放在表空间 mytablespace 中。

- 缺省情况下，会创建写至 UE 表的任何事件监视器以在数据库激活时自动激活。可使用 MANUALSTART 子句来覆盖此行为:

```
CREATE EVENT MONITOR uowmon FOR UNIT OF WORK
                    WRITE TO UNFORMATTED EVENT TABLE
                    MANUALSTART
```

在以上示例中，必须始终使用 SET EVENT MONITOR STATE 语句手动激活 uowmon 事件监视器。

下一步做什么

缺省情况下，V9.7 或更高版本中引入的事件监视器会创建为 AUTOSTART 事件监视器。数据库下次激活时以及此后后续数据库激活时，它们会自动激活。如果要立即激活该事件监视器，请在下一次数据库激活之前使用 SET EVENT MONITOR STATE 语句来手动启动该事件监视器。此外，对于锁定事件监视器、工作单元事件监视器和程序包高速缓存事件监视器中的每一个，还必须启用数据收集。

无格式事件表的列定义:

当您发出包括 WRITE TO UNFORMATTED EVENT TABLE 子句的 CREATE EVENT MONITOR 语句时，将创建无格式事件表。当您想要抽取数据以进行分析或者要从表中修剪掉不需要的数据时，列定义非常有用。

当您想要使用下列其中一个例程从无格式事件表中抽取数据时，无格式事件表的列定义非常有用:

- EVMON_FORMAT_UE_TO_XML - 将无格式事件表中的数据抽取到 XML 文档。
- EVMON_FORMAT_UE_TO_TABLES - 将无格式事件表中的数据抽取到一组关系表。

对这些例程的调用接受一个 SELECT 语句，该语句指定要抽取的行。您可以使用无格式事件表的列定义来帮助构造 SELECT 语句。

写入无格式事件表的事件数据不会被自动清除。您必须以手动方式从该表中清除数据。当您想要清除特定的一组记录时，无格式事件表的列定义非常有用。另一个选项是，使用 TRUNCATE TABLE 语句来除去所有表行。

在 CREATE EVENT MONITOR 语句中，可以指定相关联无格式事件表的名称。如果未指定此名称，那么缺省名称将是事件监视器的名称。SYSCAT.EVENTTABLES 目录视图列示了事件监视器、与其相关联的无格式表以及其他详细信息。

下表描述无格式事件表中的列。键列是 event_data 列。其他列表示可用于查找您感兴趣的事件的标识。要了解表列的其他属性，请发出 DESCRIBE 语句。

表 9. 无格式事件表的列定义

列名	列数据类型	列描述
appl_id	VARCHAR	appl_id -“应用程序标识”监视元素
appl_name	VARCHAR	appl_name -“应用程序名称”监视元素

表 9. 无格式事件表的列定义 (续)

列名	列数据类型	列描述
event_correlation_id	BIT DATA	可选的事件相关标识。空值表示无法获得事件相关标识。 此值基于事件监视器类型: • LOCKING - 保留供将来使用 • UOW - 保留供将来使用
event_data	BLOB	事件监视器所捕获的事件的完整事件记录数据，以原始二进制格式存储。
event_id	INTEGER	event_id -“事件标识”监视元素
event_timestamp	TIMESTAMP	event_timestamp -“事件时间戳记”监视元素
event_type	VARCHAR	event_type -“事件类型”监视元素
member	SMALLINT	member -“数据库成员”监视元素
partitioning_key	INTEGER	表的分区键，用于使插入操作在运行事件监视器的数据库分区中以局部方式执行。
record_seq_num	INTEGER	在 event_data 列中存储的记录的序号。
record_type	INTEGER	在 event_data 列中存储的记录的类型。
service_subclass_name	VARCHAR	service_subclass_name -“服务子类名称”监视元素
service_superclass_name	VARCHAR	service_superclass_name -“服务超类名称”监视元素
workload_name	VARCHAR	workload_name -“工作负载名称”监视元素
mon_interval_id	BIGINT	mon_interval_id -“监视时间间隔标识”监视元素

常规表输出与 UE 表输出之间的差别:

通常来讲，可写至常规表和无格式事件 (UE) 表的事件监视器捕获相同数据。但是，有些细微差别需要注意。

列顺序

第一个差别与表的列顺序有关。事件监视器生成常规表时，与通过对 UES 表运行 EVMON_FORMAT_UE_TO_TABLES 生成的输出相比，列通常按字母顺序展示，以下两种情况例外:

- 如果输出中包括 PARTITION_KEY 列，那么它为第一列。
- 对于报告度量值的表，相关列组合到一起。例如，报告花在系统上的时间的列组合到一起。

返回的列

另一个差别与列的数据类型有关。在大多数情况下，写至表事件监视器中的列与通过对 UE 表运行 EVMON_FORMAT_UE_TO_TABLES 生成的列相同。但存在一些差别。表 10中概述了这些差别。

表 10.

逻辑数据组	常规表中返回的列	从 **EVMON_FORMAT_UE_TO** 返回的列 **_TABLES**
所有组	包括 PARTITION_KEY 列	
uow	包括 TYPE 列	不包括 TYPE 列
uow_package_list	ROUTINE_ID 数据类型为 BIGINT	ROUTINE_ID 数据类型为 INTEGER
pkgcache	不包括 XMLID 列	包括 XMLID 列
lock	DL_CONNS 数据类型为 BIGINT	DL_CONNS 数据类型为 INTEGER
	ROLLED_BACK_PARTICIPANT_NO 数据类型为 SMALLINT	ROLLED_BACK_PARTICIPANT_NO 数据类型为 INTEGER
	不包括 XMLID 列	包括 XMLID 列
lock_participants	AGENT_STATUS 数据类型为 BIGINT	AGENT_STATUS 数据类型为 INTEGER
	APPL_ID 数据类型为 VARCHAR(64)	APPL_ID 数据类型为 VARCHAR(128)
	APPL_NAME 数据类型为 VARCHAR(255)	APPL_NAME 数据类型为 VARCHAR(128)
	CLIENT_ACCTING 数据类型为 VARCHAR(200)	CLIENT_ACCTNG 数据类型为 VARCHAR(255)
	TABLESPACE_NAME 数据类型为 VARCHAR(18)	TABLESPACE_NAME 数据类型为 VARCHAR(128)
	不包括 XMLID 列	包括 XMLID 列
	包括 INTERNAL_DATA 列	不包括 INTERNAL_DATA 中包含的数据。
lock_participant_activities	ACTIVITY_ID 数据类型为 BIGINT	ACTIVITY_ID 数据类型为 INTEGER
	包括 EVENT_ID、EVENT_TYPE 和 EVENT_TIMESTAMP 而不包括 XMLID[1]	包括 XMLID 列
	CONSISTENCY_TOKEN 为 CHAR(8)	CONSISTENCY_TOKEN 为 VARCHAR(8)
lock_activity_values	ACTIVITY_ID 数据类型为 BIGINT	ACTIVITY_ID 数据类型为 INTEGER
	PARTICIPANT_NO 数据类型为 SMALLINT	PARTICIPANT_NO 数据类型为 INTEGER
	包括 EVENT_ID、EVENT_TYPE 和 EVENT_TIMESTAMP 而不包括 XMLID。[1]	包括 XMLID 列

表 10. (续)

逻辑数据组	常规表中返回的列	从 **EVMON_FORMAT_UE_TO** 返回的列 **_TABLES**
1. XMLID 列表示构成 event_header、event_id、event_type、event_timestamp 和分区监视元素的并置的复合监视元素。		

创建文件事件监视器

在创建事件监视器时，必须确定所收集信息的存储位置。文件事件监视器将事件记录存储在文件中。文件事件监视器及其选项将由 CREATE EVENT MONITOR 语句定义。

开始之前

您需要 SQLADM 或 DBADM 权限才能创建文件事件监视器。

关于此任务

文件事件监视器将事件记录流输送至一系列 8 字符编号的文件，这些文件的扩展名为 "EVT"（如 00000000.evt、00000001.evt 和 00000002.evt）。即使数据被分为较小的数据块，也应考虑将数据作为一个逻辑文件（即，数据流的开头是文件 00000000.evt 中的第一个字节；数据流的结尾是文件 nnnnnnnn.evt 的最后一个字节）。事件监视器永远不会让单个事件记录跨越两个文件。

过程

1. 指定会将事件监视器数据收集在一个文件或一组文件中，并提供存储事件文件的目录位置。

   ```
   CREATE EVENT MONITOR dlmon FOR eventtype
                       WRITE TO FILE '/tmp/dlevents'
   ```

   dlmon 是事件监视器的名称。

   /tmp/dlevents 是目录路径（在 UNIX 系统上）的名称，事件监视器会将事件文件写至该目录。

2. 指定要监视的事件的类型。可使用单个事件监视器来监视一个或多个事件类型。

   ```
   CREATE EVENT MONITOR dlmon FOR CONNECTIONS, DEADLOCKS WITH DETAILS
                       WRITE TO FILE '/tmp/dlevents'
   ```

   此事件监视器将监视 CONNECTIONS 和 DEADLOCKS WITH DETAILS 事件类型。

3. 通过调整 BUFFERSIZE 值来指定文件事件监视器缓冲区的大小（以 4K 页计）：

   ```
   CREATE EVENT MONITOR dlmon FOR CONNECTIONS, DEADLOCKS WITH DETAILS
                       WRITE TO FILE '/tmp/dlevents' BUFFERSIZE 8
   ```

   8 是两个事件文件缓冲区的容量（以 4K 页计）。

   每个缓冲区的缺省大小为 4 页（分配两个 16K 缓冲区）。最小大小为 1 页。由于缓冲区是根据监视器堆进行分配，所以缓冲区的最大大小受该监视器堆大小限制。由于性能原因，高可用事件监视器的缓冲区应该比不活动事件监视器的缓冲区大。

4. 指示是需要分块事件监视器还是非分块事件监视器。对于分块事件监视器，如果事件缓冲区已满，那么生成事件的每个代理程序将等待事件缓冲区写至文件。这可能

会导致数据库性能降低，原因是暂挂的代理程序和所有从属代理程序在缓冲区清空之前不能运行。使用 BLOCKED 子句来确保事件数据不会丢失:

```
CREATE EVENT MONITOR dlmon FOR CONNECTIONS, DEADLOCKS WITH DETAILS
                    WRITE TO FILE '/tmp/dlevents' BUFFERSIZE 8
                    BLOCKED
```

缺省情况下，事件监视器处于受阻状态。如果数据库性能比收集每个事件记录更重要，那么使用非分块事件监视器。在此情况下，如果事件缓冲区已满，那么生成事件的每个代理程序将不会等待事件缓冲区写至文件。因此，非分块事件监视器可能会导致活动频繁的系统上的数据丢失。使用 NONBLOCKED 子句来使事件监视导致的额外处理时间降至最少:

```
CREATE EVENT MONITOR dlmon FOR CONNECTIONS, DEADLOCKS WITH DETAILS
                    WRITE TO FILE '/tmp/dlevents' BUFFERSIZE 8
                    NONBLOCKED
```

5. 指定可对每个事件监视器收集的事件文件的最大数目。如果达到此限制，那么事件监视器将释放自身。

   ```
   CREATE EVENT MONITOR dlmon FOR CONNECTIONS, DEADLOCKS WITH DETAILS
                       WRITE TO FILE '/tmp/dlevents' BUFFERSIZE 8
                       NONBLOCKED MAXFILES 5
   ```

   5 是将要创建的事件文件的最大数目。

   还可指定不限制事件监视器可创建的事件文件数:

   ```
   CREATE EVENT MONITOR dlmon FOR CONNECTIONS, DEADLOCKS WITH DETAILS
                       WRITE TO FILE '/tmp/dlevents' BUFFERSIZE 8
                       NONBLOCKED MAXFILES NONE
   ```

6. 指定事件监视器创建的每个事件文件的最大大小（以 4K 页计）。如果达到此限制，那么创建新文件。

   ```
   CREATE EVENT MONITOR dlmon FOR CONNECTIONS, DEADLOCKS WITH DETAILS
                       WRITE TO FILE '/tmp/dlevents' BUFFERSIZE 8
                       NONBLOCKED MAXFILES 5 MAXFILESIZE 32
   ```

   32 是事件文件可包含的最大 4K 页数。

   此值必须大于 BUFFERSIZE 参数指定的值。还可指定不限制事件文件的大小:

   ```
   CREATE EVENT MONITOR dlmon FOR CONNECTIONS, DEADLOCKS WITH DETAILS
                       WRITE TO FILE '/tmp/dlevents' BUFFERSIZE 8
                       NONBLOCKED MAXFILES NONE MAXFILESIZE NONE
   ```

7. 指定每次数据库启动时是否自动激活事件监视器。在缺省情况下，数据库启动时不会自动激活事件监视器（WLM 事件监视器例外）。

   - 要创建在数据库启动时自动启动的事件监视器，请发出以下语句:

     ```
     CREATE EVENT MONITOR dlmon FOR CONNECTIONS, DEADLOCKS WITH DETAILS
                         WRITE TO FILE '/tmp/dlevents' BUFFERSIZE 8
                         NONBLOCKED AUTOSTART
     ```

   - 要创建在数据库启动时不自动启动的事件监视器，请发出以下语句:

     ```
     CREATE EVENT MONITOR dlmon FOR CONNECTIONS, DEADLOCKS WITH DETAILS
                         WRITE TO FILE '/tmp/dlevents' BUFFERSIZE 8
                         NONBLOCKED MANUALSTART
     ```

8. 要激活或释放事件监视器，请使用 SET EVENT MONITOR STATE 语句。

结果

一旦创建并激活文件事件监视器，该事件监视器就会在指定的事件发生时记录监视数据。

事件监视器文件管理:

使用某些事件监视器，您可以将事件数据写入到文本文件中。可以配置某些所创建文件数量的上限，及其 CREATE 或 ALTER EVENT MONITOR 语句上选项的大小。

文件事件监视器允许事件监视器将其事件记录存储在文件中。事件监视器的所有输出将存储在 CREATE EVENT MONITOR 语句的 FILE 参数提供的目录中。在激活监视器之前，此目录必须存在，否则 SET EVENT MONITOR 命令将返回错误；如果此目录还不存在，那么数据库管理器不会创建此目录。

要点: 当第一次激活文件事件监视器时，将在此目录中创建控制文件 `db2event.ctl`。不要除去或修改此文件。

在缺省情况下，事件监视器会将跟踪写至称为 `00000000.evt` 的单个文件。只要文件系统上还有空间，此文件就会一直增长。如果使用 CREATE EVENT MONITOR 语句的 **MAXFILESIZE** 参数指定了文件大小限制，那么在文件已满时，输出将自动导向下一个文件。每次创建一个新文件时，构成文件名的数字便加 1。因此，活动文件就是具有最高编号的文件。

还可通过使用 CREATE EVENT MONITOR 语句的 **MAXFILES** 参数来限制整个事件监视器跟踪的最大大小。当文件数达到 MAXFILES 定义的最大数目时，事件监视器将释放自身，并且会向管理通知日志写入以下消息。

```
DIA1601I 事件监视器 monitor-name 在达到预设的 MAXFILES 和 MAXFILESIZE 限制时释放。
```

如果您收到此消息，那么请勿删除任何事件监视器文件。如果您删除了任何文件，将不能使用 **db2evmon** 命令查看任何事件监视器信息（即使是任何其余文件中包含的此类信息）。请改为执行下列其中一项操作:

- 在没有 **MAXFILES** 和 **MAXFILESIZE** 限制的情况下重新创建事件监视器。
- 将 **MAXFILES** 和 **MAXFILESIZE** 参数所实施的限制保留不变，但是将目录中的所有文件（最新的 *.evt 文件除外）移动到其他目录或文件系统。然后您便可在新目录的文件中查看事件监视器信息。如果需要，可创建一个脚本来自动执行该操作。

无论是何种方法，均必须使用语句 **SET EVENT MONITOR** *event-monitor-name* **STATE 1** 重新激活事件监视器，从而在收到 DIA1601I 消息后再次开始收集信息。

重新启动文件事件监视器时，它会擦除任何现有数据或追加的新数据。在 CREATE EVENT MONITOR 语句中指定此选项，并可创建 APPEND 监视器或 REPLACE 监视器。APPEND 是缺省选项。APPEND 事件监视器开始在上次使用的文件结尾写入。如果已除去该文件，那么会使用下一个顺序文件编号。重新启动追加事件监视器时，仅生成 start_event。仅对第一次激活生成事件日志头和数据库头。REPLACE 事件监视器总是会删除现有事件文件并开始在 `00000000.evt` 中写入。

注: 如果您未对事件监视器使用 REPLACE 选项，那么可执行以下步骤来强制事件监视器开始收集新数据集:

1. 使用 **SET EVENT MONITOR** *event-monitor-name* **STATE 0** 命令停用事件监视器。
2. 删除 CREATE EVENT MONITOR 语句的 FILE 选项所指定目录中的所有文件。

3. 使用 **SET EVENT MONITOR** *event-monitor-name* **STATE 1** 命令重新激活事件监视器。

如果文件事件监视器用完了磁盘空间，那么在管理通知日志中记录系统错误级别消息后，它会关闭自身。

您可能想要在事件监视器活动时处理监视器数据。这是可行的，而且在处理完文件后可以删除它，从而释放空间以供后续监视数据使用。除非停止并重新启动事件监视器，否则不能强制事件监视器切换至下一个文件。它必须也处于 APPEND 方式。为跟踪在活动文件中处理的事件，可创建一个应用程序，它只跟踪已处理的上一个记录的文件编号和位置。下一次处理跟踪时，该应用程序可搜索该文件的位置。

创建管道事件监视器

在创建事件监视器时，必须确定所收集信息的存储位置。管道事件监视器直接将事件记录从事件监视器传输至命名管道。

开始之前

- 您需要 SQLADM 或 DBADM 权限才能创建管道事件监视器。
- 此任务假定已创建命名管道。要在 UNIX 或 Linux 系统上创建命名管道，请使用这些系统上提供的 **mkfifo** 命令。

关于此任务

在事件监视器写入事件数据时，将由监视应用程序迅速读取管道中的数据。如果事件监视器无法将数据写至管道（例如，如果管道已满），那么监视器数据将会丢失。

管道事件监视器将使用 CREATE EVENT MONITOR 语句定义。

过程

1. 指示会将事件监视器数据引导至命名管道。

   ```
   CREATE EVENT MONITOR myevmon FOR eventtype
                        WRITE TO PIPE '/home/dbadmin/dlevents'
   ```

   myevmon 是事件监视器的名称。

   /home/dbadmin/dlevents 是事件监视器会将事件记录引导至的命名管道的名称（在 UNIX 上）。CREATE EVENT MONITOR 语句支持 UNIX 和 Windows 管道命名语法。

   激活事件监视器时，CREATE EVENT MONITOR 语句中指定的命名管道必须存在并且打开。如果指定事件监视器将自动启动，那么在创建事件监视器之前命名管道必须存在。

2. 指定要监视的事件的类型。可使用单个事件监视器来监视一个或多个事件类型。

   ```
   CREATE EVENT MONITOR myevmon FOR BUFFERPOOLS, TABLESPACES
                        WRITE TO PIPE '/home/dbadmin/myevents'
   ```

   此事件监视器将监视 BUFFERPOOLS 和 TABLESPACES 事件类型。

3. 指定每次数据库启动时是否自动激活事件监视器。在缺省情况下，数据库启动时不会自动激活事件监视器。
   - 要创建在数据库启动时自动启动的事件监视器，请发出以下语句:

     ```
     CREATE EVENT MONITOR myevmon FOR BUFFERPOOLS, TABLESPACES
                          WRITE TO PIPE '/home/dbadmin/myevents'
                                        AUTOSTART
     ```

- 要创建在数据库启动时不自动启动的事件监视器，请发出以下语句:

```
CREATE EVENT MONITOR myevmon FOR BUFFERPOOLS, TABLESPACES
                    WRITE TO PIPE '/home/dbadmin/myevents
                                          MANUALSTART
```

4. 启动从命名管道读取的客户机应用程序。例如，可启动 db2evmon 工具以在数据传送至管道时处理该数据。
5. 要激活或释放事件监视器，请使用 SET EVENT MONITOR STATE 语句。

结果

创建并激活管道事件监视器后，该事件监视器将在其指定事件发生时记录监视数据。

事件监视器命名管道管理:

使用某些事件监视器，您可以将事件数据写入到命名管道中。接下来的一些指南是关于如何高效使用命名管道事件监视器。

管道事件监视器允许通过命名管道处理事件监视器数据流。如果需要实时处理事件记录，那么最好使用管道事件监视器。另一个很重要的优点是，应用程序在读完管道时可以忽略不想要的数据，从而可以大幅降低存储器要求。

在 AIX® 上，可使用 mkfifo 命令来创建命名管道。在 Linux 和其他 UNIX 类型（如 Solaris 操作系统）上，使用 pipe() 例程。在 Windows 上，可通过使用 CreateNamedPipe() 例程来创建命名管道。

将数据引导至管道时，I/O 总是会分块并且唯一的缓冲将由管道执行。在事件监视器写入事件数据时，将由监视应用程序迅速读取管道中的数据。如果事件监视器无法将数据写至管道（例如，因为管道已满），那么监视器数据将会丢失。

此外，命名管道中必须有足够的空间用来处理入局事件记录。如果应用程序从命名管道读取数据时不够快，管道将填满并溢出。管道缓冲区越小，溢出的机率越大。

当发生管道溢出时，监视器将创建溢出事件记录以指示发生溢出。事件监视器不会关闭，但监视器数据会丢失。如果释放监视器时存在明显的溢出事件记录，那么会记录诊断消息。否则，溢出事件记录可能时将写至管道。

一次可写至管道的数据量由底层操作系统确定。如果操作系统允许定义管道缓冲区的大小，那么使用的管道缓冲区至少应该为 32K。对于高容量事件监视器，应将监视应用程序的进程优先级设置为等于或高于代理进程优先级。

来自活动事件监视器或统计信息事件监视器的单次写操作的数据流包含的数据可多于写至命名管道的数据。在这类情况下，该数据流会分割为可装入到缓冲区中的块，每个块用头标识: 第一个块用带有元素标识 SQLM_ELM_EVENT_STARTPIPEBLOCK 的逻辑头标识。最后一个块用带有元素标识 SQLM_ELM_EVENT_ENDPIPEBLOCK 的逻辑头标识。之间的所有块用带有元素标识 SQLM_ELM_EVENT_MIDPIPEBLOCK 的逻辑头标识。正读取该管道的监视应用程序必须识别这些头，并且将这些块重新聚集成完整数据流（根据需要剥离块头并重新聚集这些块以形成完整的有效数据流）。db2evmon 工具提供此功能；它提供写至命名管道的事件监视器为所有事件生成的有格式输出（根据需要重新聚集这些块）。如果只想处理所选事件或监视元素，那么可编写您自己的应用程序来完成此任务。

“写至表”和文件事件监视器缓存

对于写至表和文件事件监视器而言，事件监视器在将输出写至文件或表之前将输出存储在缓冲区中。表 11显示哪些事件监视器使用这类输出缓冲区。

表 11. 事件监视器和输出缓冲区

事件监视器类型	将输出写至磁盘前是否先写至缓冲区?
活动	否
缓冲池	是
变更历史记录	否
连接	是
数据库	是
死锁（所有版本）	是
锁定	否
程序包高速缓存	否
语句	是
统计信息	是
表空间	是
表	是
事务	是
工作单元	否

不使用缓冲区的事件监视器会使用更新更快速的机制将输出写至磁盘，从而不需要缓冲区。

对于使用缓冲区的事件监视器，缓冲区已满时记录会自动写至磁盘。因此，通过指定较大的缓冲区以减少磁盘访问次数，可以改善高吞吐量事件监视器的监视性能。为了强制事件监视器清空其缓冲区，必须释放该事件监视器，或者使用 FLUSH EVENT MONITOR 语句来清空缓冲区。

使用缓冲区的事件监视器允许您指定事件监视器输出是*受阻*还是*非受阻*。对于受阻事件监视器来说，当它的两个缓冲区都变满时，它将暂挂正在发送监视器数据的数据库进程。这是为了确保在受阻事件监视器活动期间不会删除任何事件记录。在将缓冲区内容写入文件或表之前，暂挂的数据库进程以及任何从属数据库进程都无法运行。根据工作负载类型以及 I/O 设备速度的不同，这会造成极大的性能下降。缺省情况下，事件监视器处于受阻状态。

对于非受阻事件监视器来说，如果从代理程序接收监视器数据的速度高于事件监视器写数据的速度，它将删除那些数据。这样就可以避免事件监视操作影响其他数据库活动的性能。

删除了事件记录的事件监视器会生成溢出事件。此事件指定了监视器删除事件的开始时间和停止时间以及在该时间段内删除的事件数。事件监视器有可能对要报告的暂挂溢出终止或释放。如果发生这种情况，就会将以下消息写入管理日志:

```
DIA2503I 事件监视器 monitor-name 在释放时有暂挂的溢出记录。
```

各个事件记录也会发生丢失事件监视数据的情况。如果事件记录长度超过事件缓冲区大小，在缓冲区中装不下的数据就会被截断。例如，如果正在捕获 stmt_text 监视元素

并且连接到被监视数据库的应用程序发出了很长的 SQL 语句，就会发生数据截断的情况。如果必须捕获所有事件记录信息，请指定较大的缓冲区。请记住，较大的缓冲区会降低写入文件或表的频率。

事件监视器自描述数据流

写入管道或文件的事件监视器将输出逻辑数据分组的二进制流；对于管道事件监视器和文件事件监视器，此二进制流完全相同。可通过使用 **db2evmon** 命令或开发客户机应用程序来格式化数据流。此数据流显示为自描述格式。图 4 显示数据流的结构，而 表 12 提供可能返回的逻辑数据组和监视元素的一些示例。

注: 在示例和表中，将对标识使用描述性名称。在实际数据流中，将在这些名称前加上 **SQLM_ELM_** 前缀。例如，**db_event** 在事件监视器输出中显示为 SQLM_ELM_DB_EVENT。在实际数据流中，将在类型前加上 **SQLM_TYPE_** 前缀。例如，HEADER 在数据流中将显示为 SQLM_TYPE_HEADER。

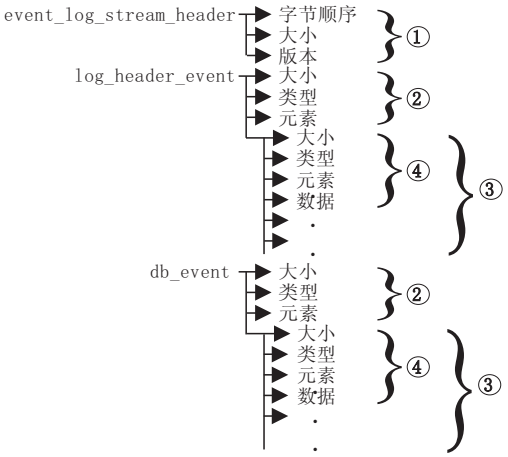


图 4. 管道或文件事件监视器数据流

1. sqlm_event_log_data_stream_header 的结构与数据流中的其他头不同。版本字段确定是否能作为自描述数据流来处理输出。

   此头的大小和类型与 V6 之前的事件监视器流的大小和类型相同。这允许应用程序确定事件监视器输出是自描述格式还是 V6 之前的静态格式。

   注: 将通过从数据流读取 sizeof(sqlm_event_log_data_stream) 字节来抽取此监视元素。
2. 每个逻辑数据组以指示其大小和元素名称的头开始。因为大小元素包含哑元值以保留向后兼容性，所以这不适用于 event_log_stream_header。
3. 头中的大小元素指示该逻辑数据组中的所有数据的大小。
4. 监视元素信息在逻辑数据组头后面，并且也是自描述格式。

表 12. 样本事件数据流

逻辑数据组	数据流	描述
event_log_stream_header	┌►sqlm_little_endian	未使用（因为与前发行版的兼容性）。
	├►200	未使用（因为与前发行版的兼容性）。
	└►sqlm_dbmon_version9	返回数据的数据库管理器的版本。事件监视器以自描述格式写入数据。

表 12. 样本事件数据流 (续)

逻辑数据组	数据流	描述
log_header_event	┬►100	逻辑数据组的大小。
	├►header	指示逻辑数据组的开头。
	├►log_header	逻辑数据组的名称。
	├ →►4	此监视元素中存储的数据的大小。
	│ ├►u32bit	监视元素类型 - 32 位数字。
	│ ├►byte_order	收集的监视元素的名称。
	│ └►little_endian	此元素的已收集值。
	└ ┬►2	此监视元素中存储的数据的大小。
	├►u16bit	监视元素类型 - 不带符号的 16 位数字。
	├►codepage_id	收集的监视元素的名称。
	└►850	此元素的已收集值。
db_event	┬►100	逻辑数据组的大小。
	├►header	指示逻辑数据组的开头。
	├►db_event	逻辑数据组的名称。
	└ →►4	此监视元素中存储的数据的大小。
	├►u32bit	监视元素类型 - 不带符号的 32 位数字。
	├►lock_waits	收集的监视元素的名称。
	└►2	此元素的已收集值。

event_log_stream_header 标识返回数据的数据库管理器的版本。事件监视器以自描述格式写入它们的数据。与快照监视器不同，事件监视器没有用来返回跟踪总大小的**大小**元素。event_log_stream_header 中显示的数字是为向后兼容性提供的哑元值。写入 event_log_stream_header 时，事件跟踪的总大小未知。通常可读取事件监视器跟踪直到文件或管道末尾。

日志头描述跟踪的特征，它包含各种信息，如在其中收集跟踪的服务器的内存模型（如小尾数法）和数据库的代码页。如果在其中读取跟踪的系统的内存模型与服务器的内存模型不同（例如，如果在 Windows 2000 系统上从 UNIX 服务器读取跟踪），那么您可能必须对数字值执行字节交换。如果配置数据库时使用的语言与在其中读取跟踪的机器使用的语言不同，那么可能还需要进行代码页转换。读取跟踪时，可使用**大小**元素来跳过跟踪中的逻辑数据组。

事件类型至逻辑数据组的映射

对于文件和管道事件监视器，事件监视器输出由一个有序的逻辑数据分组序列组成。不管事件监视器类型如何，输出记录总是包含相同的起始逻辑数据组。它们为逻辑数据组设置了框架，这些逻辑数据组的存在与否取决于事件监视器记录的事件类型。

对于文件和管道事件监视器，可能会对任何连接生成事件记录，并且事件记录可能会因此以混合顺序出现在流中。这意味着您可能获得连接 1 的事务事件，之后紧跟连接 2 的连接事件。但是，属于单个连接或单个事件的记录将以逻辑顺序出现。例如，语句记录（语句结尾）总是在事务记录（UOW 结尾）之前，如果有。同样，死锁事件总是在死锁中涉及的每个连接的死锁连接事件记录之前。**应用程序标识**或**应用程序句柄（agent_id）**可用来将记录与连接相匹配。

通常会对与数据库的每个连接写入连接头事件。对于带有详细信息的死锁事件监视器，仅当发生死锁时才将它们写入。在此情况下，将仅对死锁参与者（而不是与数据库的所有连接）写入连接头事件。

逻辑数据分组将按以下四个不同级别排序：监视器、序言、内容和结尾。下面详细描述每个级别，包括对应的事件类型和逻辑数据组。

监视器

将对所有事件监视器生成监视器级别的信息。它包含事件监视器元数据。

表 13. 事件监视器数据流：监视器部分

事件类型	逻辑数据组	可用信息
监视器级别	event_log_stream_header	标识事件监视器的版本级别和字节顺序。应用程序可使用此头来确定是否能够处理 evmon 输出流。

序言

在激活事件监视器时将生成序言信息。

表 14. 事件监视器数据流：序言部分

事件类型	逻辑数据组	可用信息
日志头	event_log_header	跟踪的特征，如服务器类型和内存布局。
数据库头	event_db_header	数据库名称、路径和激活时间。
事件监视器启动	event_start	启动或重新启动监视器的时间。
连接头	event_connheader	每个当前事件对应一个连接头，包括连接时间和应用程序名称。仅对连接、语句、事务和死锁事件监视器生成事件连接头。带有详细信息的死锁事件监视器仅在发生死锁时生成连接头。

内容

特定于事件监视器的指定事件类型的信息出现在内容部分。

表 15. 事件监视器数据流：内容部分

事件类型	逻辑数据组	可用信息
语句事件	event_stmt	语句级别数据，包括动态语句的文本。语句事件监视器不记录访存。
子节事件	event_subsection	子节级别数据。
事务事件[1]	event_xact	事务级别数据。
连接事件	event_conn	连接级别数据。
死锁事件	event_deadlock	死锁级别数据。
死锁的连接事件	event_dlconn	死锁涉及的每个连接对应一个死锁的连接事件，包括涉及的应用程序和处于争用状态的锁定。

表 15. 事件监视器数据流：内容部分 (续)

事件类型	逻辑数据组	可用信息
带有详细信息的死锁的连接事件	event_detailed_dlconn, lock	死锁涉及的每个连接对应一个带有详细信息的死锁的连接事件，包括涉及的应用程序、处于争用状态的锁定、当前语句信息和应用程序争用挂起的其他锁定。
溢出	event_overflow	丢失的记录的个数 - 在写程序不能与（非分块）事件监视器保持一致时生成。
带有详细信息的死锁的历史纪录 [2]	event_stmt_history	列示死锁涉及的任何工作单元中执行的语句列表。
带有详细信息的死锁的历史记录值 [2]	event_data_value	event_stmt_history 列表中的语句的参数标记。
活动	event_activity	系统上已完成执行的或在完成前捕获的活动列表。
	event_activitystmt	活动类型为语句时，有关活动正在执行的语句的信息。
	event_activityvals	用作每个 SQL 语句活动的输入变量的数据值。这些数据值不包括 LOB 数据、长数据或结构化类型数据。
统计信息	event_scstats	从在系统中每个服务类、工作类或工作负载内执行的活动计算出来的统计信息，以及从阈值队列计算出来的统计信息。
	event_wcstats	
	event_wlstats	
	event_qstats	
	event_histogrambin	
阈值违例	event_thresholdviolations	标识阈值违例及违例时间的信息。

[1] 建议不要使用此选项。建议不要再使用此选项，将来的发行版可能会将其除去。请使用 CREATE EVENT MONITOR FOR UNIT OF WORK 语句来监视事务事件。

[2] 建议不要使用此选项。建议不要再使用此选项，将来的发行版可能会将其除去。请使用 CREATE EVENT MONITOR FOR LOCKING 语句来监视与锁定相关的事件，例如锁定超时、锁定等待和死锁。

结尾

结尾信息将在数据库释放时（最后一个应用程序断开连接后）生成:

表 16. 事件监视器数据流：结尾部分

事件类型	逻辑数据组	可用信息
数据库事件	event_db	数据库管理器级别数据。
缓冲池事件	event_bufferpool	缓冲池级别数据。
表空间事件	event_tablespace	表空间级别数据。
表事件	event_table	表级别数据。

显示数据库中创建的事件监视器的列表

可通过使用目录视图 SYSCAT.EVENTMONITORS 来查看数据库中已定义的事件监视器。

过程

要查看您在系统上定义的事件监视器的列表，请查询目录视图 SYSCAT.EVENTMONITORS。 例如，要查看包括事件监视器名称、目标输出类型（即，常规表、文件、命名管道或无格式事件表）和所有者的事件监视器列表，可使用类似如下的查询:

```
SELECT SUBSTR(EVMONNAME,1,20) AS EVMON_NAME, TARGET_TYPE, OWNER
FROM SYSCAT.EVENTMONITORS
```

上述查询返回类似如下的结果:

```
EVMON_NAME           TARGET_TYPE OWNER
-------------------- ----------- ----------------------------------
DB2DETAILDEADLOCK    F           DBADMIN1
CACHEEVMON           T           DBADMIN1
INVTLOCK             T           DBADMIN1
INVTUOW              T           DBADMIN1
INVTACT              T           DBADMIN1
INVTSTATS            T           DBADMIN1
INVTTHRESHOLD        T           DBADMIN1
TABLE_INVTTABLE      T           DBADMIN1
BUFFER_INVT          T           DBADMIN1
TABLESPACES_INVT     T           DBADMIN1
CONNECTIONS_INVT     T           DBADMIN1
TRANSAC_INVT         T           DBADMIN1
DEADLOCK_INVT        T           DBADMIN1
QUINNJN_LOC_UNF      U           DBADMIN1
UNFORM               U           DBADMIN1
RM                   U           DBADMIN1
UOWINVT              U           DBADMIN1
LOCK_UP_STAFF        U           DBADMIN1
INVTLOCK2            T           DBADMIN1
STAFF_UOW            T           DBADMIN1
STAFFSTATS           T           DBADMIN1

  21 record(s) selected.
```

示例

还可使用目录视图来查看有哪些用于监视特定类型的事件的事件监视器。 SYSCAT.EVENTS 视图返回事件监视器列表和这些事件监视器记录其数据的事件类型。

```
SELECT SUBSTR(TYPE,1,20) AS EVENT_TYPE,
   SUBSTR(EVMONNAME,1,20) AS EVENT_MONITOR_NAME
   FROM SYSCAT.EVENTS
   ORDER BY TYPE

EVENT_TYPE           EVENT_MONITOR_NAME
-------------------- --------------------
ACTIVITIES           INVTACT
BUFFERPOOLS          BUFFER_INVT
CONNECTIONS          CONNECTIONS_INVT
DEADLOCKS            DEADLOCK_INVT
DETAILDEADLOCKS      DB2DETAILDEADLOCK
LOCKING              INVTLOCK
LOCKING              QUINNJN_LOC_UNF
LOCKING              UNFORM
LOCKING              RM
```

```
LOCKING              LOCK_UP_STAFF
LOCKING              INVTLOCK2
PKGCACHEBASE         CACHEEVMON
STATISTICS           INVTSTATS
STATISTICS           STAFFSTATS
TABLES               TABLE_INVTTABLE
TABLESPACES          TABLESPACES_INVT
THRESHOLDVIOLATIONS  INVTTHRESHOLD
TRANSACTIONS         TRANSAC_INVT
UOW                  INVTUOW
UOW                  UOWINVT
UOW                  STAFF_UOW

  21 record(s) selected.
```

分区数据库和 DB2 pureScale 环境中的数据库的事件监视器

通常，分区数据库系统或 DB2 pureScale® 环境中的事件监视器的工作方式与在非分区单成员数据库上运行的事件监视器的工作方式相似。但是，有些差别需要注意。

分区数据库环境

写至常规表和无格式事件 (UE) 表的事件监视器

不能在特定分区上创建写至常规表和无格式事件 (UE) 表的事件监视器。反而，对于分区数据库环境，一个事件监视器进程在每个分区上运行。更具体地说，该事件监视器进程在属于数据库分区组的每个分区（目标表在这些分区上）的成员上运行。

对于特定事件监视器，运行事件监视器进程的每个分区具有同一组目标表。这些表中的数据因分区不同而不同，因为特定分区的数据仅反映该分区上发生的事件。对于表事件监视器，可通过发出 SQL 语句以从每个分区的事件监视器表中收集数据来从所有分区检索聚集值。对于 UE 表事件监视器，可通过使用您对 EVMON_FORMAT_UE_TO_TABLE 存储过程指定的 SQL 语句或使用 EVMON_FORMAT_UE_TO_XML 表函数来跨分区聚集数据。

每个事件监视器表的第一列名为 PARTITION_KEY，并且用作该表的分区键。此列的值将被选中，以便每个事件监视器进程将数据插入到运行该进程的数据库分区中。即，在运行该事件监视器进程的数据库分区本地执行插入操作。在任何数据库分区上，PARTITION_KEY 字段包含相同的值。因此，如果删除数据分区并且执行数据重新分发，那么所删除数据库分区上的所有数据将转至另一数据库分区而不是平均分发。因此，在删除数据库分区之前，请考虑删除该数据库分区上的所有表行。

此外，在分区数据库环境中，可对每个表定义名为 PARTITION_NUMBER 或 MEMBER 的列。此列包含插入了数据的分区或成员的编号。

事件被写至目标表的表空间存在的那些分区上的事件监视器目标表。如果事件监视器目标表的表空间在运行该事件监视器的任何分区上不存在，那么系统不会在这些分区上收集任何数据，也不返回任何错误。而且，不会在不存在表空间的位置写入这些事件的日志记录。此行为意味着您可通过创建仅存在于某些分区上的表空间来选择要监视的一部分分区。

在“写至表”事件监视器激活期间，FIRST_CONNECT 和 EVMON_START 的 CONTROL 表行将插入至目标表的表空间存在的所有数据库分区。

如果激活事件监视器后分区仍未处于活动状态，那么该事件监视器将在该分区下一次激活时激活。

写至文件和命名管道的事件监视器

文件和管道事件监视器仅捕获运行它们的数据库分区（*监视分区*）上发生的事件，但有一个例外。这类事件监视器称为*局部事件监视器*。该例外是 DEADLOCK 事件监视器；可将其创建为局部或全局事件监视器。如果将其创建为全局事件监视器，那么系统会在所有数据库分区上收集死锁信息并报告给运行该事件监视器进程的特定数据库分区。[3]

如果在分区数据库环境中创建文件或管道事件监视器，那么可指定要将其作为 CREATE EVENT MONITOR 语句一部分运行的分区。如果省略分区编号，那么该事件监视器在创建该事件监视器时连接的数据库分区上运行。

仅当监视分区处于活动状态时，才能激活事件监视器。如果使用 SET EVENT MONITOR 语句来激活事件监视器但监视分区尚未处于活动状态，那么该事件监视器将在监视分区下一次启动时激活。而且，直到您显式取消激活该事件监视器或实例，该事件监视器才不会自动激活。例如，考虑以下语句序列:

```
DB2 CONNECT TO PAYROLL
DB2 CREATE EVENT MONITOR ABC ... ON DBPARTITIONNUM 2
DB2 SET EVENT MONITOR ABC STATE 1
```

运行这些语句后，每当在数据库分区 2 上激活数据库 PAYROLL 时，事件监视器 ABC 都会自动激活。直到发出 `DB2 SET EVENT MONITOR ABC STATE 0` 语句或停止分区 2，才不会进行此自动激活。

如果添加数据库分区，那么现有全局表或 UE 表事件监视器不会自动开始收集新创建分区的数据。要收集并记录关于新分区的数据，您必须执行下列其中一个步骤:

- 对于全局事件监视器（即，DEADLOCKS 事件监视器），请重新启动这些事件监视器。
- 对于表或 UE 表事件监视器，请删除、重新创建和重新启动这些事件监视器。

DB2 pureScale 环境

在 DB2 pureScale 环境中，实际上有一个数据分区及两个或更多处理数据的成员。因此，如果创建事件监视器，那么事件监视器进程会在所有成员上运行，不管它们写至文件、管道、表还是 UE 表。

事件数据是针对每个成员报告的。因此，与成员相关联的监视元素或度量值（例如，**total_cpu_time** 监视元素）报告特定于该成员的数据。但是，不管哪个成员报告，与该数据本身相关的监视元素（例如，**tablespace_total_pages** 监视元素）都反映相同的值。

示例

示例 1: 在分区数据库环境中创建“写至文件”事件监视器

以下示例说明如何创建事件监视器，该事件监视器运行并收集分区 3 上与缓冲池相关的事件的数据，然后将其输出写至文件:

```
CREATE EVENT MONITOR bpmon FOR BUFFERPOOLS
                    WRITE TO FILE '/tmp/dlevents'
                    ON DBPARTITION 3
```

3. 建议不要使用此事件监视器。 LOCKING 事件监视器是捕获锁定和死锁事件信息的首选事件监视器。

示例 2: 在分区数据库环境中创建表事件监视器

以下示例说明如何创建表事件监视器，该表事件监视器运行并收集与活动相关的事件的数据，然后将其输出写至表:

```
CREATE EVENT MONITOR myacts FOR ACTIVITIES
                    WRITE TO TABLE
```

在此示例中，因为未对事件监视器指定逻辑数据组，所以系统为与此类型的事件监视器相关联的所有逻辑数据组创建表。如果每个分区上存在缺省表空间，那么系统会在缺省表空间中的每个分区上创建其中每个表。在每个数据库分区上的表中收集的数据与该分区上发生的事件相关。

要查看所选分区中的事件监视器数据，请发出查询这些分区的 SELECT 语句:

```
SELECT TOTAL_CPU_TIME FROM myacts WHERE PARTITION_NUMBER = 3
```

启用事件监视器数据收集

根据您正在使用的事件监视器的类型，您可能需要在创建该事件监视器后配置收集。缺省情况下，某些事件监视器会在激活后立即收集某些数据。其他事件监视器要求您以独立于创建事件监视器的方式显式配置数据收集。这些类型的事件监视器有时又称为*被动*事件监视器。

开始之前

必须先激活所有事件监视器，所有数据才会写至其目标输出表（常规表或 UE 表）、文件或管道。某些事件监视器在缺省情况下配置为 AUTOSTART 事件监视器。这意味着它们在数据库激活时自动激活。其他事件监视器在缺省情况下配置为需要您手动激活。在任一方式下，您都可覆盖缺省启动选项。但是，要在您创建自动事件监视器后下次数据库激活前启动该事件监视器，必须使用 SET EVENT MONITOR STATE 语句以手动激活该事件监视器。

关于此任务

某些事件监视器支持在 CREATE 或 ALTER EVENT MONITOR 语句上使用 WHERE 子句以选择性地捕获事件信息。但是，以下事件监视器能够控制哪些事件数据以独立于事件监视器定义的方式进行收集:

- 活动
- 变更历史记录
- 锁定
- 统计信息
- 工作单元

缺省情况下，某些列示的事件监视器在事件监视器激活后收集某些类型的数据；其他事件监视器要求您显式启用数据收集。在任一方式下，都可以两种方式的其中一种来启用数据收集，这取决于您要为其收集数据的活动的作用域:

数据库中的所有活动

要收集数据库中的所有活动的监视数据，应针对您关心的数据类型修改适当的配置参数。例如，要收集在数据库中运行的所有工作单元的工作单元数据，请将 **mon_uow_data** 设置为 BASE。在某些情况下，配置参数的缺省设置为: 如果存在处于活动状态的适当事件监视器可接收某种类型的数据，那么始终会收

集该数据。例如，**mon_req_metrics** 的缺省设置为 BASE（除非您覆盖此设置）；所有处于活动状态的统计事件监视器或工作单元事件监视器都会记录请求监视元素的 BASE 集合的值。

切记: 支持使用 WHERE 谓词的事件监视器仅收集满足该谓词中指定条件的数据，不管任何相关配置参数的设置如何都是如此。

所选活动

某些事件监视器 - 特别是工作负载管理事件监视器（阈值违例、统计信息和活动）- 能够控制特定工作负载管理对象的数据收集。例如，您可选择收集正在特定服务超类中运行的活动的活动信息。在此级别配置收集通常包括向 CREATE 或 ALTER WORKLOAD（或者，SERVICE CLASS 或 WORK ACTION）语句添加 COLLECT 子句，以指定要对在该 WLM 对象的保护下运行的活动收集的信息类型。例如，为了对服务类 urgent 启用扩展统计信息收集，可使用以下语句:

```
ALTER SERVICE CLASS urgent
      COLLECT AGGREGATE ACTIVITY DATA EXTENDED
```

注: 如果在 WLM CREATE 或 ALTER 语句中指定 COLLECT 子句，那么对于该 WLM 对象，在该子句中指定的设置优先于使用配置参数配置的任何数据库范围的设置。例如，如果 **mon_req_metrics** 设置为 EXTENDED 并且工作负载 payroll 配置为收集 BASE 请求度量值（例如，CREATE WORKLOAD payroll COLLECT REQUEST METRICS BASE），那么会针对数据库中的所有活动（payroll 工作负载除外）收集扩展请求度量值。

过程

要对本节开头显示的其中一种类型的事件监视器启用数据收集，请执行以下步骤:

1. 确定缺省情况下是否已收集任何数据。可能不需要您更改任何设置就会收集您关心的数据。
2. 决定要对其收集数据的活动的作用域。您是要对整个数据库收集数据，还是仅对特定工作负载、服务类或工作操作收集数据？
3. 决定要收集的监视元素的类型。某些事件监视器支持收集不同类型的监视数据，例如，请求监视元素、活动数据等。
4. 对于所收集监视数据的不同集合，决定要在每个集合内收集的数据的范围。您通常可选择不收集任何数据 (NONE)、收集基本数据 (BASE) 或扩展数据 (EXTENDED)。请进行查看以确定要针对每个设置收集的数据。
5. 根据先前步骤中所做的决定，使用配置参数或 COLLECT 子句来配置数据收集。
   a. 要针对整个数据库配置收集，请设置适当的配置参数。例如，要启用通过数据库 SALES 上的锁定事件监视器对带有历史记录的锁定等待信息的收集，请运行以下命令。

      ```
      UPDATE DATABASE CONFIGURATION for SALES USING mon_lockwait HISTORY
      ```

   b. 要配置对特定工作负载的收集，请创建或修改该工作负载并包括适当的 COLLECT 子句。 例如，要对 MANAGERS 工作负载中锁定等待长于 5 秒的情况配置带有语句历史记录的锁定等待数据的收集，请运行类似如下的语句:

      ```
      ALTER WORKLOAD MANAGERS
        COLLECT LOCK WAIT DATA FOR LOCKS WAITING MORE THAN 5 SECONDS
          WITH HISTORY
      ```

下一步做什么

现在事件监视器已创建并处于活动状态，并且数据收集已启用，请运行应用程序或工作负载。

用于访问事件监视器信息的方法

根据您要使用的事件监视器类型及其生成的输出类型，有不同选项可用于访问和查看事件监视器数据。

例如:

- 可直接使用 SQL 查询表事件监视器生成的数据。
- 可在写至管道的事件监视器中的数据生成时查看该数据。
- 可通过在取消激活事件监视器后打开输出文件来查看文件事件监视器中的数据。
- 还可使用 **db2evmon** 命令将文件事件监视器和管道事件监视器中的数据的格式变为报告。
- 必须先对写至 UE 表的数据执行后处理，才能检查该数据。UE 事件监视器数据可转换为表或 XML，这使得能够使用 SQL 或 XML 查询技巧来查询该数据。或者，可将 UE 表中的数据的格式变为有格式报告而不必完成转换过程。

下面的几节描述可用于访问事件监视器生成的信息的不同方法。

存取常规表中的事件监视器数据

可使用 SQL 直接访问写至常规关系表的事件监视器数据。

开始之前

访问数据之前，必须执行以下任务:

- 创建和激活事件监视器
- 如果必要，对您正在使用的事件监视器类型和您要收集的数据类型启用数据收集
- 运行要对其收集监视数据的工作负载或应用程序

（可选）根据您使用该事件监视器数据的方式，在开始检查事件数据之前取消激活数据收集。如果事件监视器仍处于活动状态，那么它会继续将数据写至输出表。因此，一个查询的结果可能不同于以后运行同一查询时获取的结果。

关于此任务

存取关系表中的事件监视器数据包括使用 SQL 来构造查询以从该事件监视器生成的表中检索数据。

过程

要从写至表的事件监视器生成的表中检索信息，请执行以下操作:

1. 构造 SELECT 语句以显示您要查看的监视元素数据。 例如，要请求针对来自名为 mylocks 的锁定事件监视器的 payroll 工作负载数据的锁定，可使用类似如下的查询:

```
SELECT DISTINCT CAST(STMT_TEXT AS VARCHAR(25)) STMT, LP.PARTICIPANT_NO,
   VARCHAR(LP.APPL_NAME,10) APPL_NAME, LP.LOCK_MODE_REQUESTED,
     LP.PARTICIPANT_TYPE
```

```
FROM LOCK_PARTICIPANT_ACTIVITIES_LOCK_MYLOCKS AS LPA
JOIN LOCK_PARTICIPANTS_LOCK_MYLOCKS AS LP
   ON LPA.EVENT_ID = LP.EVENT_ID
   WHERE LP.WORKLOAD_NAME = 'PAYROLL'
```

在此示例中，来自事件监视器 mylocks 的 LOCK_PARTICIPANTS 表的数据与来自 LOCK_PARTICIPANTS_ACTIVITIES 表的信息连接到一起，从而返回以下结果。

2. 运行此 SQL 语句。

结果

```
STMT                       PARTICIPANT_NO APPL_NAME  LOCK_WAIT_VAL
-------------------------- -------------- ---------- --------------------
select * from staff                     2 db2bp                         0
select * from staff                     1 db2bp                      1000

LOCK_MODE_REQUESTED  PARTICIPANT_TYPE
-------------------- ----------------
                   0 OWNER
                   1 REQUESTER


  2 record(s) selected.
```

用于访问无格式事件表中的信息的方法

可使用不同方法来访问无格式事件 (UE) 表中的信息。可生成计划读取的文本报告。或者，可将数据抽取到关系表或 XML 中；此方法允许您使用 SQL 或 pureXML® 查询该数据。

写至 UE 表的事件监视器以二进制格式写入事件数据。您可以使用 db2evmonfmt 命令或例程来访问此数据。

借助 db2evmonfmt 命令，可以执行以下操作:

- 根据下列属性来选择感兴趣的事件: 事件标识、事件类型、时间段、应用程序、工作负载或服务类。
- 选择是以文本报告形式还是格式化 XML 文档形式来接收输出。
- 通过创建您自己的 XSLT 样式表代替使用 db2evmonfmt 附带提供的 XSLT 样式表，对输出格式进行全面控制。

您还可以使用下列例程从无格式事件表中抽取数据:

- EVMON_FORMAT_UE_TO_XML - 将无格式事件表中的数据抽取到 XML 文档。
- EVMON_FORMAT_UE_TO_TABLES - 将无格式事件表中的数据抽取到一组关系表。

借助这两个例程，可使用 SELECT 语句指定要从无格式事件表中抽取的确切行。

用于读取事件监视器数据的 db2evmonfmt 工具:

根据使用无格式事件表的事件监视器所生成的数据，基于 Java 的通用 XML 解析器工具 **db2evmonfmt** 将生成可读的纯文本输出（文本版本）或格式化 XML 输出。根据您指定的参数，**db2evmonfmt** 工具将确定事件监视器数据的解析方式以及所要创建的输出类型。

db2evmonfmt 工具以 Java 源代码形式提供。在使用此工具之前，您必须通过执行下列步骤来设置和编译此工具:

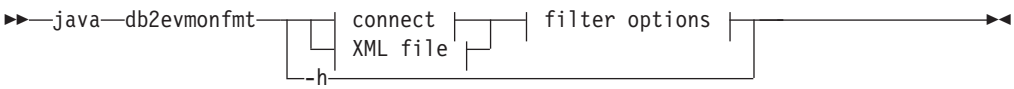
1. 在 sqllib/samples/java/jdbc 目录中找到源代码
2. 按照 Java 源文件中嵌入的指示信息来设置和编译此工具

您可以随意修改源代码以更改输出。

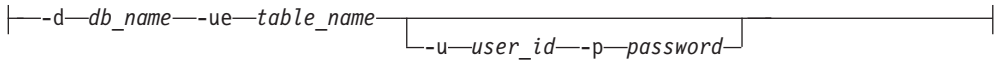
此工具使用 XSLT 样式表将事件数据变换为带格式文本。您不需要理解这些样式表。此工具将根据事件监视器类型自动装入正确的样式表并变换事件数据。每个事件监视器都将在 sqllib/samples/xml/data 目录中提供缺省样式表。此工具还提供了下列过滤选项:

- 事件标识
- 事件时间戳记
- 事件类型
- 工作负载名
- 服务类名
- 应用程序名

工具语法

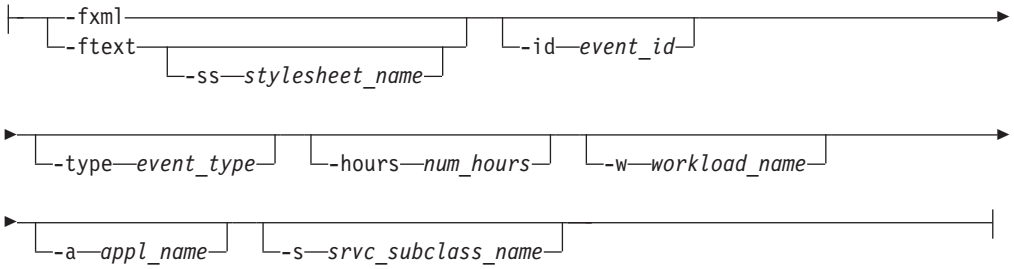


connect:



XML file:

filter options:



工具参数

java

要成功地运行基于 Java 的 **db2evmonfmt** 工具，必须在工具名前面指定 **java** 关键字。在 DB2 产品的安装期间，将从 sqllib/java/jdk64 目录中安装成功地运行此工具所需的正确 Java 版本。

-d *db_name*
指定所要连接的数据库的名称。

-ue *table_name*
指定无格式事件表的名称。

-u *user_id*
指定用户标识。

-p *password*
指定密码。

-f *xml_filename*
指定要格式化的输入 XML 文件的名称。

-fxml
生成格式化 XML 文档（输出到标准输出）。

-ftext
将 XML 文档格式化为文本文档（输出到标准输出）。

-ss *stylesheet_name*
指定用于变换 XML 文档的 XSLT 样式表。

-id *event_id*
显示所有与指定事件标识匹配的事件。

-type *event_type*
显示所有与指定事件类型匹配的事件。

-hours *num_hours*
显示所有在过去指定数目的小时内发生的事件。

-w *workload_name*
显示所有属于所指定工作负载的事件。

-a *appl_name*
显示所有属于所指定应用程序的事件。

-s *srvc_subclass_name*
显示所有属于所指定服务子类的事件。

XSLT 样式表

DB2 数据库管理器提供了缺省的 XSLT 样式表（参见表 1），这些样式表在 `sqllib/samples/java/jdbc` 目录中。您可以对这些样式表进行更改，以便生成需要的输出。

表 17. 事件监视器的缺省 XSLT 样式表

事件监视器	缺省 XSLT 样式表
锁定	DB2EvmonLocking.xsl
工作单元	DB2EvmonUOW.xsl
程序包高速缓存	DB2EvmonPkgCache.xsl

您可以创建自己的 XSLT 样式表以变换 XML 文档。可以使用 `-ss stylesheet_name` 选项将这些样式表传递给基于 Java 的工具。

示例

示例 1

要从数据库 SAMPLE 中的程序包高速缓存无格式事件表 PKG 中获取过去 32 小时内发生的所有事件的带格式文本输出，请发出以下命令:

```
java db2evmonfmt -d sample -ue pkg -ftext -hours 32
```

示例 2

要从数据库 SAMPLE 中的无格式事件表 LOCK 中获取过去 24 小时内发生的所有 LOCKTIMEOUT 类型事件的带格式文本输出，请发出以下命令:

```
java db2evmonfmt -d sample -ue LOCK -ftext -hours 24 –type locktimeout
```

示例 3

要从 XML 源文件 LOCK.XML 中获取带格式文本输出，以便抽取过去 5 小时内发生的所有与事件类型 LOCKWAIT 匹配的事件，请发出以下命令:

```
java db2evmonfmt -f lock.xml -ftext -type lockwait -hours 5
```

示例 4

要使用已创建的 XSLT 样式表 SUMMARY.XSL 从数据库 SAMPLE 中的无格式事件表 UOW 中获取所有事件的带格式文本输出，请发出以下命令:

```
java db2evmonfmt -d sample -ue uow -ftext -ss summary.xsl
```

格式化纯文本输出样本

以下格式化纯文本输出样本是使用锁定事件监视器 XSLT 样式表生成的:

```
------------------------------------------------
Event Entry     : 0
Event ID        : 1
Event Type      : Locktimeout
Event Timestamp : 2008-05-23-12.00.14.132329000
------------------------------------------------

Lock Details
------------
Lock Name       : 02000401000000000000000054
Lock Type       : Table
Lock Attributes : 00000000
Lock Count      : 1
Lock Hold Count : 0
Lock rrIID      : 0
Lock Status     : Waiting
Cursor Bitmap   : 00000000
Tablespace Name : USERSPACE1
Table Name      : NEWTON  .SARAH

Attributes           Requestor                      Holder
----------           ---------                      ------
Application Handle   [0-35]                         [0-16]
Application ID       *LOCAL.horton.080523160016     *LOCAL.horton.080523155938
Application Name     xaplus0001                     db2bp
Authentication ID    NEWTON                         HORTON
Requesting Agent     65                             21
Coordinating Agent   65                             21
Application Status   SQLM_CONNECTPEND               SQLM_CONNECTPEND
Lock Timeout         5000                           0
Workload Name        XAPLUS0010_WL02                SYSDEFAULTUSERWORKLOAD
Service Subclass     XAPLUS0010_SC02                SYSDEFAULTSUBCLASS
Current Request      Execute                        Execute Immediate
Lock Mode            Intent Exclusive               Exclusive
tpmon Userid
```

```
tpmon Wkstn
tpmon App
tpmon Accstring

Lock Requestor Current Activities
---------------------------------
Activity ID     : 2
Uow ID          : 1
Package ID      : 65426E4D4B584659
Package SectNo  : 3
Package Name    : NEWTON
Package Schema  : AKINTERF
Package Version :
Reopt           : always
Eff Isolation   : Cursor Stability
Eff Locktimeout : 5
Eff Degree      : 0
Nesting Level   : 0
Stmt Unicode    : No
Stmt Flag       : Dynamic
Stmt Type       : DML, Insert/Update/Delete
Stmt Text       : INSERT INTO SARAH VALUES(:H00008, :H00013, :H00014)


Lock Requestor Past Activities
------------------------------
Activity ID     : 1
Uow ID          : 1
Package ID      : 65426E4D4B584659
Package SectNo  : 2
Package Name    : NEWTON
Package Schema  : AKINTERF
Package Version :
Reopt           : always
Eff Isolation   : Cursor Stability
Eff Locktimeout : 5
Eff Degree      : 0
Nesting Level   : 0
Stmt Unicode    : No
Stmt Flag       : Dynamic
Stmt Type       : DML, Insert/Update/Delete
Stmt Text       : INSERT INTO NADIA VALUES(:H00007)


Lock Holder Current Activities
------------------------------

Lock Holder Past Activities
------------------------------
Activity ID     : 1
Uow ID          : 2
Package ID      : 41414141414E4758
Package SectNo  : 201
Package Name    : NULLID
Package Schema  : SQLC2G13
Package Version :
Reopt           : none
Eff Isolation   : Cursor Stability
Eff Locktimeout : 5
Eff Degree      : 0
Nesting Level   : 0
Stmt Unicode    : No
Stmt Flag       : Dynamic
Stmt Type       : DML, Select (blockable)
Stmt Text       : select * from newton.sarah

Activity ID     : 2
```

```
Uow ID          : 2
Package ID      : 414141414E4758
Package SectNo  : 203
Package Name    : NULLID
Package Schema  : SQLC2G13
Package Version :
Reopt           : none
Eff Isolation   : Cursor Stability
Eff Locktimeout : 5
Eff Degree      : 0
Nesting Level   : 0
Stmt Unicode    : No
Stmt Flag       : Dynamic
Stmt Type       : DML, Lock Table
Stmt Text       : lock table newton.sarah in exclusive mode


------------------------------------------------
Event Entry     : 1
Event ID        : 2
Event Type      : Locktimeout
Event Timestamp : 2008-05-23-12.04.42.144896000
------------------------------------------------

...
...
...
```

使用说明

db2evmonfmt 实用程序是基于 Java 的工具，要成功地运行此工具，必须在它前面指定 **java** 关键字。所需的 Java 版本是从 sqllib/java/jdk64 目录中随 DB2 产品一起安装的版本。

注: 您还可以使用 EVMON_FORMAT_UE_TO_XML 表函数将无格式事件表 BLOB 列中包含的二进制事件格式化为 XML 文档。

用于从无格式事件表抽取数据的例程:

如果要对事件监视器收集并写至无格式事件 (UE) 表的数据执行查询，必须先使用为此用途提供的两个例程中的其中一个从 UE 表抽取该数据。EVMON_FORMAT_UE_TO_TABLES 过程从 UE 表中抽取数据以创建关系表。EVMON_FORMAT_UE_TO_XML 表函数创建 XML 文档。

EVMON_FORMAT_UE_TO_TABLES

EVMON_FORMAT_UE_TO_TABLES 过程检查事件监视器生成的 UE 表，并将它包含的数据抽取到您可查询的关系表中。生成的表的数目取决于事件监视器类型以及该事件监视器针对其收集数据的逻辑数据组。一般来讲，每个逻辑数据组中的数据将写至另一个表。例如，程序包高速缓存事件监视器从三个逻辑数据组收集事件数据: pkgcache、pkgcache_metrics 和 pkgcache_stmt_args。因此，EVMON_FORMAT_UE_TO_TABLES 会生成三个表。

注: EVMON_FORMAT_UE_TO_TABLES 不会为控制逻辑数据组创建表。

除通过 UE 表创建关系表以外，从 V10.1 开始，EVMON_FORMAT_UE_TO_TABLES 过程还能够从 UE 表剪除数据。如果对 EVMON_FORMAT_UE_TO_TABLES 使用 PRUNE_UE_TABLES 选项，那么会从无格式事件 (UE) 表中删除成功插入至关系表的数据。

EVMON_FORMAT_UE_TO_XML

EVMON_FORMAT_UE_TO_XML 表函数检查事件监视器生成的 UE 表，并将它包含的数据抽取到 XML 文档中。然后，可根据需要经常使用 pureXML 来查询此文档。

注意:

- 此表函数的工作方式与 **db2evmonfmt** 实用程序（该实用程序与 **-fxml** 选项配合使用时）类似。使用 EVMON_FORMAT_UE_TO_XML 代替 **db2evmonfmt** 的差别如下:
  - EVMON_FORMAT_UE_TO_XML 是表函数。因此，它作为 SQL 语句的一部分调用。**db2evmonfmt** 作为单独的实用程序运行。
  - EVMON_FORMAT_UE_TO_XML 允许您指定带 WHERE 子句的 SELECT 语句以从 UE 表中过滤事件。**db2evmonfmt** 仅限制用于过滤事件数据的功能。
- **db2evmonfmt** 可格式化 EVMON_FORMAT_UE_TO_XML 的输出 XML 文档以创建平面文本文件。

通过两个例程，必须在对例程的调用中包括 SELECT 语句以指定要针对其抽取数据的条件。

从 UE 表中剪除数据:

如果使用 EVMON_FORMAT_UE_TO_TABLES 过程从 UE 表中抽取数据，那么可使用 PRUNE_UE_TABLE 选项来除去您不再需要的数据。

开始之前

必须先对写至 UE 表的事件监视器创建、激活和启用数据收集，才能从 UE 表中抽取数据。

关于此任务

除 UE 表提供的性能优势外，将 UE 表用作事件监视器的输出允许您使用 EVMON_FORMAT_UE_TO_TABLES 的自动剪除功能。如果使用此过程，那么可自动从 UE 表中除去从 UE 表抽取并写至常规表的任何数据。此过程使您能更轻松地管理 UE 表。例如，假定您要使用工作单元事件监视器来捕获信息以生成日常报告以便记帐，例如，针对应用程序或查询使用 CPU 的时间向部门收费）。在此情况下，生成报告后，您可能想要修剪数据。

过程

要从 UE 表中抽取数据然后剪除，请执行以下操作:

发出以下 SQL 语句，该语句调用带 PRUNE_UE_TABLE 选项的 EVMON_FORMAT_UE_TO_TABLES 过程以将数据抽取到常规表中。 例如，如果已有名为 TRACKWORK 的工作单元事件监视器，那么可创建类似如下的语句:

```
CALL EVMON_FORMAT_UE_TO_TABLES
   ('UOW', NULL, NULL, NULL, NULL, NULL, 'PRUNE_UE_TABLE', -1,
      'SELECT * FROM TRACKWORK')
```

所有事件数据将从 UE 表复制到 UOW_EVENT_TRACKWORK 表和 UOW_METRICS_TRACKWORK 表。此外，将从 UE 表中除去复制的所有记录。

从命令行格式化文件或管道事件监视器输出

文件或管道事件监视器的输出是一个逻辑数据分组二进制流。可使用 **db2evmon** 命令从命令行格式化此数据流。此高效工具从事件监视器的文件或管道读取事件记录，然后将它们写至屏幕（标准输出）。

开始之前

除非连接至数据库，否则不需要任何权限，如果连接至数据库，那么需要具有下列其中一个权限:

- SYSADM
- SYSCTRL
- SYSMAINT
- DBADM

关于此任务

可通过提供事件文件的路径或提供数据库名称和事件监视器名称，以指示想要格式化的事件监视器输出。

过程

要格式化事件监视器输出:

- 指定包含事件监视器文件的目录:

  ```
  db2evmon -path '/tmp/dlevents'
  ```

  /tmp/dlevents 表示（UNIX）路径。
- 指定数据库和事件监视器名称:

  ```
  db2evmon -db 'sample' -evm 'dlmon'
  ```

  sample 表示事件监视器所属的数据库。

  dlmon 表示事件监视器。

改变事件监视器

不能更改事件监视器，但下面是一个例外情况: 可向该事件监视器收集的逻辑数据组集合添加一个或多个逻辑数据组。应使用 ALTER EVENT MONITOR 语句来添加逻辑组。

关于此任务

缺省情况下，创建写至表的事件监视器时，将捕获与该事件监视器相关联的监视元素的所有逻辑数据组。但是，如果在 CREATE EVENT MONITOR 语句中添加了逻辑数据组的名称，那么仅捕获这些组。例如，可创建仅捕获来自 event_activity 和 event_activity_metrics 逻辑数据组的数据的活动事件监视器，如以下示例所示:

```
CREATE EVENT MONITOR myacts FOR ACTIVITIES
   WRITE TO TABLE
   event_activity, event_activity_metrics
```

以上 DDL 语句会创建事件监视器，该事件监视器写至以下两个表：ACTIVITY_myacts 和 ACTIVITY_METRICS_myacts。

限制

只能使用 ALTER EVENT MONITOR 语句向事件监视器添加逻辑数据组。不能除去逻辑数据组。也不能更改与下表相关联的名称、目标表空间或 PCTDEACTIVATE 值：该表用于捕获属于数据组的监视元素中的数据。

过程

要向事件监视器添加其他逻辑数据组，请执行以下操作:

1. 决定要添加的逻辑数据组。 假定您要添加 event_activitystmt 和 event_activityvals 逻辑数据组（使用以上仅捕获两个逻辑数据组的锁定事件监视器示例）。
2. 创建 ALTER EVENT MONITOR 语句以添加这些新逻辑数据组。

```
ALTER EVENT MONITOR mylacts
   ADD LOGICAL GROUP event_activitystmt
   ADD LOGICAL GROUP event_activityvals
```

3. 执行该语句。

结果

ALTER EVENT MONITOR 语句执行完后，系统将为事件监视器 myacts 创建两个其他表:

ACTIVITYSTMT_myacts

ACTIVITYVALS_myacts

下一次激活该事件监视器时，系统会使用来自这些表的对应逻辑数据组中的数据来填充这些表。

切记: 如果向事件监视器添加新逻辑数据组，那么以前该表中的逻辑数据组存在的任何数据将不会在新添加逻辑组的表中有任何对应行。根据需要调整查询，或考虑在添加逻辑组后从该表中剪除旧数据。

示例

数据库管理员使用以下 SQL 语句创建名为 mylocks 的锁定事件监视器:

```
CREATE EVENT MONITOR mylocks FOR LOCKING WRITE TO TABLE LOCK, LOCK_PARTICIPANTS
```

此语句收集锁定逻辑数据组和 lock_participants 逻辑数据组中的监视元素的信息。已创建具有缺省表名 LOCK_MYLOCKS 和 LOCK_PARTICIPANTS_MYLOCKS 的表，它们用于写入监视元素数据。

稍后，数据库管理员决定要收集 LOCK_PARTICIPANT_ACTIVITIES 逻辑数据组中的信息。她使用以下语句来修改该事件监视器:

```
ALTER EVENT MONITOR mylocks ADD LOGICAL GROUP LOCK_PARTICIPANT_ACTIVITIES
```

此语句导致收集 lock_participant_activities 中的监视元素以及其他已收集的元素。这一组新的监视元素被写至表 LOCK_PARTICIPANT_ACTIVITIES_MYLOCKS。

稍后，数据库管理员决定还需要来自控制逻辑数据组的数据。但是，她希望此数据写至其名称不同于缺省名称并且其表空间不同于缺省表空间的表。她使用以下语句:

```
ALTER EVENT MONITOR mylocks ADD LOGICAL GROUP CONTROL TABLE ctl_mylocks IN mytbsp3
```

此语句将该控制逻辑数据组添加至该事件监视器的输出。此语句将该控制逻辑数据组添加至该事件监视器的输出。数据被写至 CTL_MYLOCKS 表，表被写至表空间 mytbsp3 而不是缺省表空间。

监视不同类型的事件

锁定和死锁事件监视

在大型 DB2 环境中，诊断和排除锁定争用情况的工作可能相当复杂且耗时。锁定事件监视器旨在通过收集锁定数据简化此任务。

注: 建议不要使用死锁事件监视器，它提供的功能包括在锁定事件监视器中。此外，也建议不要使用 DB2DETAILDEADLOCK 事件监视器。有关此事件监视器的重要用法信息，请参阅第 119 页的『不推荐使用的锁定监视功能』。

锁定事件监视器用于在发生锁定事件时捕获关于那些事件的描述性信息。捕获的信息将标识锁定事件引起的锁定争用情况所涉及的关键应用程序。将同时捕获关于锁定请求者（接收到死锁或锁定超时错误或者等待锁定时的耗用时间超出指定时间长度的应用程序）和当前锁定所有者的信息。

锁定事件监视器收集的信息将以二进制格式写至数据库中的无格式事件表，在此情况下必须在后捕获步骤中处理所捕获数据。或者，可将锁定事件信息写至一组常规表。有关如何选择最适当的输出格式的更多信息，请参阅第 31 页的『事件监视器的输出选项』。

您还可以使用动态或静态 SQL 来直接访问 DB2 关系监视接口（表函数），以便收集锁定事件信息。

确定是否已发生死锁或锁定超时的过程也有所简化。消息将在其中任何一个事件发生时被写入管理通知日志；这将对返回给应用程序的 SQL0911N（sqlcode 为 -911）错误进行补充。此外，还会将锁定升级通知写入管理通知日志；此信息对于您调整锁定表大小以及应用程序能够使用的锁定表空间量而言十分有用。另外，还可以检查有关锁定超时（**lock_timeouts**）、锁定等待（**lock_waits**）和死锁（**deadlocks**）的计数器。

可以捕获其锁定数据的活动的类型如下所示:

- SQL 语句，例如:
  - DML
  - DDL
  - CALL
- **LOAD** 命令
- **REORG** 命令
- **BACKUP DATABASE** 命令
- 实用程序请求

锁定事件监视器将不推荐使用的死锁事件监视器（CREATE EVENT MONITOR FOR DEADLOCKS 语句和 DB2DETAILDEADLOCK）以及不推荐使用的锁定超时报告功能（DB2_CAPTURE_LOCKTIMEOUT 注册表变量）替换为简化而一致的锁定事件数据收集接口，并添加了捕获锁定等待数据的功能。

功能概述

要使用锁定事件监视器来捕获锁定事件数据，需要执行两个步骤:

1. 必须使用 CREATE EVENT MONITOR FOR LOCKING 语句来创建 LOCK EVENT 监视器。您应提供监视器的名称和（如果要将 UE 表作为输出格式）锁定事件数据写至的无格式事件表的名称。

   注: 如果选择对事件监视器输出使用常规表，那么会指定缺省表名。如果愿意，可覆盖 CREATE EVENT MONITOR 语句中的缺省值。

2. 必须使用下列其中一种方法来指定锁定事件数据的捕获级别:
   - 您可以通过更改现有工作负载或者使用 CREATE 或 ALTER WORKLOAD 语句创建新工作负载来指定特定的工作负载。在工作负载级别，您必须指定要捕获的锁定事件数据的类型（死锁、锁定超时或锁定等待），并指定锁定是否涉及应用程序的 SQL 语句历史记录和输入值。对于锁定等待，还必须指定应用程序将等待锁定的时间长度（在这段时间过后，将捕获锁定等待数据）。
   - 通过设置适当的数据库配置参数，可以在数据库级别收集数据并影响所有 DB2 工作负载:

     mon_lockwait

     此参数控制锁定等待事件的生成方式。

     最佳实践是，在工作负载级别启用锁定等待数据收集功能。

     mon_locktimeout

     此参数控制锁定超时事件的生成方式。

     最佳实践是，如果这些事件并不是应用程序所预期的事件，请在数据库级别启用锁定超时数据收集功能。否则，请在工作负载级别启用此功能。

     mon_deadlock

     此参数控制死锁事件的生成方式。

     最佳实践是，在数据库级别启用死锁数据收集功能。

     mon_lw_thresh

     此参数控制在生成 **mon_lockwait** 的事件之前等待锁定时耗用的时间。

捕获 SQL 语句历史记录和输入值将使用额外的处理器时间、内存和存储，但要成功地调试锁定问题，通常需要此级别的详细信息。

发生锁定事件后，可查看该事件监视器生成的输出中的事件数据。如果使用 UE 表，那么通过使用所提供的基于 Java 的应用程序 **db2evmonfmt**，可以将无格式事件表中的二进制数据变换为 XML 或文本文档。另外，可以使用 EVMON_FORMAT_UE_TO_XML 表函数将无格式事件表 BLOB 列中的二进制事件数据格式化为 XML 报告文档，也可以使用 EVMON_FORMAT_UE_TO_TABLES 过程将其格式化为关系表。

如果使用常规表作为输出格式，那么可使用 SQL 直接查询该数据。

为了帮助确定应该对哪些工作负载监视锁定事件，您可以查看管理通知日志。每当遇到死锁或锁定超时情况时，系统都会将一条消息写入日志。这些消息将标识正在其中运行锁定请求者和锁定所有者的工作负载以及锁定事件的类型。另外，还可以在工作负载级别检查有关锁定超时（**lock_timeouts**）、锁定等待（**lock_waits**）和死锁（**deadlocks**）的计数器。

为锁定事件收集的信息

锁定事件监视器所收集的一些锁定事件信息包括:

- 导致发生事件的锁定
- 正在挂起导致发生锁定事件的锁定的应用程序
- 正在等待或请求导致发生锁定事件的锁定的应用程序
- 应用程序在锁定事件期间执行的操作

不推荐使用的锁定监视功能

缺省情况下，将为每个数据库创建不推荐使用的详细死锁事件监视器 DB2DETAILDEADLOCK，并在激活数据库时启动该监视器。如果使用锁定事件监视器来检测死锁，请考虑禁用 DB2DETAILDEADLOCK 事件监视器。如果锁定事件监视器也在收集死锁信息期间 DB2DETAILDEADLOCK 事件监视器仍处于活动状态，那么两个事件监视器都将收集数据，这会严重影响性能。

要除去 DB2DETAILDEADLOCK 事件监视器，请发出下列 SQL 语句:

```
SET EVENT MONITOR DB2DETAILDEADLOCK state 0
DROP EVENT MONITOR DB2DETAILDEADLOCK
```

锁定事件监视器生成的数据

锁定事件监视器生成有关系统中的锁定和死锁的数据。可选择将锁定事件监视器的输出写至常规表或无格式事件 (UE) 表。如果数据写至 UE 表，那么必须对该数据执行后处理才能查看该数据。

不管您选择什么输出格式，锁定事件数据都来自下列四个逻辑组的其中一个:

- lock
- lock_participants
- lock_participant_activities
- lock_activity_values

如果选择将锁定事件数据写至常规表，那么另一个组 (CONTROL) 中的数据用于生成有关事件监视器本身的元数据。

注: 缺省情况下，仅对锁定事件监视器生成死锁信息。要生成其他类型的锁定数据，必须显式启用该数据的收集。

锁定事件监视器写至表的信息:

在指定了 WRITE TO TABLE 选项的情况下锁定事件监视器写入的信息。

选择 WRITE TO TABLE 作为锁定事件监视器的输出类型时，缺省情况下，会生成五个表，每个表包含一个或多个逻辑数据组中的监视元素:

表 18. 锁定写至表事件监视器生成的表

缺省表名	报告的逻辑数据组
LOCK_*evmon-name*	lock
LOCK_PARTICIPANTS_*evmon-name*	lock_participants
LOCK_PARTICIPANT_ACTIVITIES_*evmon-name*	lock_participant_activities
LOCK_ACTIVITY_VALUES_*evmon-name*	lock_activity_values
CONTROL_*evmon-name*	CONTROL 逻辑组由来自 event_dbheader、event_start 和 event_overflow 逻辑数据组中的一个或多个的所选元素组成。

要点：即使缺省情况下生成全部五个表，仍然必须确保已启用对要收集的锁定信息种类启用了数据收集。否则，其中一些列包含空值。

要将事件监视器的输出范围限制为特定表，请对 CREATE EVENT MONITOR 或 ALTER EVENT MONITOR 语句指定要对其生成表的逻辑组的名称。有关详细信息，请参阅这些语句的参考主题。

生成的表

表 19. 对锁定事件监视器返回的信息：缺省表名：LOCK_evmon-name

列名	数据类型	描述
PARTITION_KEY	INTEGER	第 951 页的『partition_key -“分区键”监视元素』
DEADLOCK_TYPE	VARCHAR(10)	第 728 页的『deadlock_type -“死锁类型”监视元素』
DL_CONNS	INTEGER	dl_conns - 死锁中涉及的连接数
EVENT_ID	BIGINT NOT NULL	event_id -“事件标识”监视元素
EVENT_TIMESTAMP	TIMESTAMP NOT NULL	event_timestamp -“事件时间戳记”监视元素
EVENT_TYPE	VARCHAR(128) NOT NULL	event_type -“事件类型”监视元素
MEMBER	SMALLINT NOT NULL	member - 数据库成员
ROLLED_BACK_PARTICIPANT_NO	INTEGER	rolled_back_participant_no - 回滚的应用程序参与者

表 20. 对锁定事件监视器返回的信息：缺省表名：LOCK_PARTICIPANTS_evmon-name

列名	数据类型	描述
PARTITION_KEY	INTEGER	第 951 页的『partition_key -“分区键”监视元素』
AGENT_STATUS	INTEGER	agent_status - DCS 应用程序代理程序数
AGENT_TID	BIGINT	第 604 页的『agent_tid -“代理程序线程标识”监视元素』
APPL_ACTION	VARCHAR(64)	第 615 页的『appl_action -“应用程序操作”监视元素』
APPL_ID	VARCHAR(128)	appl_id - 应用程序标识

表 20. 对锁定事件监视器返回的信息：缺省表名：*LOCK_PARTICIPANTS_evmon-name*（续）

列名	数据类型	描述
APPL_NAME	VARCHAR(128)	appl_name - 应用程序名称
APPLICATION_HANDLE	BIGINT	application_handle - 应用程序句柄
AUTH_ID	VARCHAR(128)	auth_id - 授权标识
CLIENT_ACCTNG	VARCHAR(255)	client_acctng - 客户机记帐字符串
CLIENT_APPLNAME	VARCHAR(255)	client_applname - 客户机应用程序名称
CLIENT_USERID	VARCHAR(255)	client_userid - 客户机用户标识
CLIENT_WRKSTNNAME	VARCHAR(255)	client_wrkstnname - 客户机工作站名称
COORD_AGENT_TID	BIGINT	第 692 页的『coord_agent_tid -“协调代理程序引擎可分派单元标识”监视元素』
CURRENT_REQUEST	VARCHAR(32)	第 711 页的『current_request -“当前操作请求”监视元素』
DEADLOCK_MEMBER	SMALLINT	第 727 页的『deadlock_member -“死锁成员”监视元素』
EVENT_ID	BIGINT	event_id -“事件标识”监视元素
EVENT_TIMESTAMP	TIMESTAMP	event_timestamp -“事件时间戳记”监视元素
EVENT_TYPE	VARCHAR(128)	event_type -“事件类型”监视元素
INTERNAL_DATA	VARCHAR(255)	
LOCK_ATTRIBUTES	CHAR(8)	lock_attributes - 锁定属性
LOCK_COUNT	BIGINT	lock_count - 锁定计数
LOCK_CURRENT_MODE	BIGINT	lock_current_mode - 转换前的原始锁定方式
LOCK_ESCALATION	CHAR(3)	lock_escalation - 锁定升级
LOCK_HOLD_COUNT	BIGINT	lock_hold_count - 锁定持有计数
LOCK_MODE	BIGINT	lock_mode - 锁定方式
LOCK_MODE_REQUESTED	BIGINT	lock_mode_requested - 请求的锁定方式
LOCK_NAME	CHAR(32)	lock_name - 锁定名称
LOCK_OBJECT_TYPE	BIGINT	lock_object_type - 等待的锁定对象类型
LOCK_OBJECT_TYPE_ID	CHAR(1)	保留以供将来使用
LOCK_RELEASE_FLAGS	CHAR(8)	lock_release_flags - 锁定释放标志
LOCK_RRIID	BIGINT	
LOCK_STATUS	BIGINT	lock_status - 锁定状态
LOCK_TIMEOUT_VAL	BIGINT	lock_timeout_val - 锁定超时值
LOCK_WAIT_END_TIME	TIMESTAMP	第 863 页的『lock_wait_end_time -“锁定等待结束时间戳记”监视元素』
LOCK_WAIT_START_TIME	TIMESTAMP	lock_wait_start_time - 锁定等待开始时间戳记
LOCK_WAIT_VAL	BIGINT	第 867 页的『lock_wait_val -“锁定等待值”监视元素』
MEMBER	SMALLINT	member - 数据库成员
OBJECT_REQUESTED	VARCHAR(10)	第 928 页的『object_requested -“所请求对象”监视元素』
PARTICIPANT_NO	INTEGER	participant_no - 死锁参与者

表 20. 对锁定事件监视器返回的信息：缺省表名：LOCK_PARTICIPANTS_evmon-name（续）

列名	数据类型	描述
PARTICIPANT_NO_HOLDING_LK	INTEGER	participant_no_holding_lk - 对应用程序所需对象持有锁定的参与者
PARTICIPANT_TYPE	VARCHAR(10)	第 950 页的『participant_type -“参与者类型”监视元素』
PAST_ACTIVITIES_WRAPPED	CHAR(3)	第 953 页的『past_activities_wrapped -“合并过去活动列表”监视元素』
QUEUE_START_TIME	TIMESTAMP	第 1094 页的『queue_start_time -“队列开始时间戳记”监视元素』
QUEUED_AGENTS	BIGINT	第 1095 页的『queued_agents -“已排队阈值代理程序数”监视元素』
SERVICE_CLASS_ID	INTEGER	service_class_id - 服务类标识
SERVICE_SUBCLASS_NAME	VARCHAR(128)	service_subclass_name - 服务子类名
SERVICE_SUPERCLASS_NAME	VARCHAR(128)	service_superclass_name - 服务超类名
TABLE_FILE_ID	BIGINT	table_file_id - 表文件标识
TABLE_NAME	VARCHAR(128)	table_name - 表名
TABLE_SCHEMA	VARCHAR(128)	table_schema - 表模式名
TABLESPACE_NAME	VARCHAR(128)	tablespace_name - 表空间名称
THRESHOLD_ID	INTEGER	第 1228 页的『thresholdid -“阈值标识”监视元素』
THRESHOLD_NAME	VARCHAR(128)	threshold_name - 阈值名称
UTILITY_INVOCATION_ID	VARCHAR(32) FOR BIT DATA	
WORKLOAD_ID	INTEGER	workload_id - 工作负载标识
WORKLOAD_NAME	VARCHAR(128)	workload_name - 工作负载名称
XID	VARCHAR(140)	xid - 事务标识

表 21. 对锁定事件监视器返回的信息：缺省表名：LOCK_PARTICIPANT_ACTIVITIES_evmon-name

列名	数据类型	描述
PARTITION_KEY	INTEGER	第 951 页的『partition_key -“分区键”监视元素』
ACTIVITY_ID	BIGINT	activity_id - 活动标识
ACTIVITY_TYPE	VARCHAR(10)	activity_type - 活动类型
CONSISTENCY_TOKEN	CHARACTER(8)	consistency_token - 程序包一致性标记
EFFECTIVE_ISOLATION	CHARACTER(2)	effective_isolation - 有效隔离级别
EFFECTIVE_QUERY_DEGREE	BIGINT	effective_query_degree - 有效查询并行度
EVENT_ID	BIGINT	event_id -“事件标识”监视元素
EVENT_TIMESTAMP	TIMESTAMP	event_timestamp -“事件时间戳记”监视元素
EVENT_TYPE	VARCHAR(128)	event_type -“事件类型”监视元素
INCREMENTAL_BIND	CHARACTER(3)	第 816 页的『incremental_bind -“增量绑定”监视元素』
MEMBER	SMALLINT	member - 数据库成员
PACKAGE_NAME	VARCHAR(128)	package_name - 程序包名
PACKAGE_SCHEMA	VARCHAR(128)	package_schema - 程序包模式

表 21. 对锁定事件监视器返回的信息：缺省表名：LOCK_PARTICIPANT_ACTIVITIES_evmon-name（续）

列名	数据类型	描述
PACKAGE_VERSION_ID	VARCHAR(64)	package_version_id - 程序包版本
PARTICIPANT_NO	SMALLINT	participant_no - 死锁参与者
QUERY_ACTUAL_DEGREE	INTEGER	query_actual_degree - 实际运行时分区内并行度
REOPT	VARCHAR(10)	第 1104 页的『reopt -“REOPT 绑定选项”监视元素』
SECTION_NUMBER	BIGINT	section_number - 节号
STMT_FIRST_USE_TIME	TIMESTAMP	stmt_first_use_time - 第一次使用语句时的时间戳记
STMT_INVOCATION_ID	BIGINT	stmt_invocation_id - 语句调用标识
STMT_LAST_USE_TIME	TIMESTAMP	stmt_last_use_time - 上一次使用语句时的时间戳记
STMT_LOCK_TIMEOUT	INTEGER	stmt_lock_timeout - 语句锁定超时
STMT_NEST_LEVEL	BIGINT	stmt_nest_level - 语句嵌套级别
STMT_OPERATION	VARCHAR(128)	第 1173 页的『stmt_operation/operation -“语句操作”监视元素』
STMT_PKGCACHE_ID	BIGINT	stmt_pkgcache_id - 语句程序包高速缓存标识
STMT_QUERY_ID	BIGINT	stmt_query_id - 语句查询标识
STMT_SOURCE_ID	BIGINT	stmt_source_id - 语句源标识
STMT_TEXT	CLOB	stmt_text - SQL 语句文本
STMT_TYPE	BIGINT	stmt_type - 语句类型
STMT_UNICODE	CHARACTER(3)	第 1181 页的『stmt_unicode -“语句 Unicode 标志”监视元素』
UOW_ID	INTEGER	uow_id - 工作单元标识

表 22. 对锁定事件监视器返回的信息：缺省表名：LOCK_ACTIVITY_VALUES_evmon-name

列名	数据类型	描述
PARTITION_KEY	INTEGER	第 951 页的『partition_key -“分区键”监视元素』
ACTIVITY_ID	BIGINT	activity_id - 活动标识
EVENT_ID	BIGINT	event_id -“事件标识”监视元素
EVENT_TIMESTAMP	TIMESTAMP	event_timestamp -“事件时间戳记”监视元素
EVENT_TYPE	VARCHAR(128)	event_type -“事件类型”监视元素
MEMBER	SMALLINT	member - 数据库成员
PARTICIPANT_NO	SMALLINT	participant_no - 死锁参与者
STMT_VALUE_DATA	CLOB	stmt_value_data - 值数据
STMT_VALUE_INDEX	INTEGER	stmt_value_index - 值索引
STMT_VALUE_ISNULL	INTEGER	stmt_value_isnull - 包含空值
STMT_VALUE_ISREOPT	INTEGER	stmt_value_isreopt - 用于语句重新优化的变量
STMT_VALUE_TYPE	CHARACTER(16)	stmt_value_type - 值类型
UOW_ID	INTEGER	uow_id - 工作单元标识

表 23. 对锁定事件监视器返回的信息: 缺省表名: CONTROL_evmon-name

列名	数据类型	描述
PARTITION_KEY	INTEGER	第 951 页的『partition_key -“分区键”监视元素』
EVENT_MONITOR_NAME	VARCHAR(128)	event_monitor_name - 事件监视器名称
MESSAGE	VARCHAR(128)	message - 控制表消息
MESSAGE_TIME	TIMESTAMP	message_time - 时间戳记控制表消息
PARTITION_NUMBER	SMALLINT	partition_number - 分区号

EVMON_FORMAT_UE_TO_TABLES 为锁定事件监视器写至关系表的信息:

EVMON_FORMAT_UE_TO_TABLES 表函数为锁定事件监视器写入的信息。`sqllib/misc/DB2EvmonLocking.xsd` 文件中也记录了此信息。

表 24. 为锁定事件监视器返回的信息: 表名: LOCK_EVENT

列名	数据类型	描述
XMLID	VARCHAR(256) NOT NULL	第 1346 页的『xmlid -“XML 标识”监视元素』
DEADLOCK_TYPE	VARCHAR(10)	第 728 页的『deadlock_type -“死锁类型”监视元素』
EVENT_ID	BIGINT NOT NULL	event_id -“事件标识”监视元素
EVENT_TYPE	VARCHAR(128) NOT NULL	event_type -“事件类型”监视元素
EVENT_TIMESTAMP	TIMESTAMP NOT NULL	event_timestamp -“事件时间戳记”监视元素
MEMBER	SMALLINT NOT NULL	member - 数据库成员
DL_CONNS	INTEGER	dl_conns - 死锁中涉及的连接数
ROLLED_BACK_PARTICIPANT_NO	INTEGER	rolled_back_participant_no - 回滚的应用程序参与者

表 25. 为锁定事件监视器返回的信息: 表名: LOCK_PARTICIPANTS

列名	数据类型	描述
XMLID	VARCHAR(256) NOT NULL	第 1346 页的『xmlid -“XML 标识”监视元素』
PARTICIPANT_NO	INTEGER	participant_no - 死锁参与者
PARTICIPANT_TYPE	VARCHAR(10)	第 950 页的『participant_type -“参与者类型”监视元素』
PARTICIPANT_NO_HOLDING_LK	INTEGER	participant_no_holding_lk - 对应用程序所需对象持有锁定的参与者
APPLICATION_HANDLE	BIGINT	application_handle - 应用程序句柄
APPL_ACTION	VARCHAR(64)	第 615 页的『appl_action -“应用程序操作”监视元素』
APPL_ID	VARCHAR(128)	appl_id - 应用程序标识
APPL_NAME	VARCHAR(128)	appl_name - 应用程序名称
AUTH_ID	VARCHAR(128)	auth_id - 授权标识

表 25. 为锁定事件监视器返回的信息: 表名: LOCK_PARTICIPANTS (续)

列名	数据类型	描述
AGENT_TID	BIGINT	第 604 页的『agent_tid -“代理程序线程标识”监视元素』
COORD_AGENT_TID	BIGINT	第 692 页的『coord_agent_tid -“协调代理程序引擎可分派单元标识”监视元素』
AGENT_STATUS	INTEGER	agent_status - DCS 应用程序代理程序数
DEADLOCK_MEMBER	SMALLINT	第 727 页的『deadlock_member -“死锁成员”监视元素』
LOCK_TIMEOUT_VAL	BIGINT	lock_timeout_val - 锁定超时值
LOCK_WAIT_VAL	BIGINT	第 867 页的『lock_wait_val -“锁定等待值”监视元素』
WORKLOAD_ID	INTEGER	workload_id - 工作负载标识
WORKLOAD_NAME	VARCHAR(128)	workload_name - 工作负载名称
SERVICE_CLASS_ID	INTEGER	service_class_id - 服务类标识
SERVICE_SUPERCLASS_NAME	VARCHAR(128)	service_superclass_name - 服务超类名
SERVICE_SUBCLASS_NAME	VARCHAR(128)	service_subclass_name - 服务子类名
CURRENT_REQUEST	VARCHAR(32)	第 711 页的『current_request -“当前操作请求”监视元素』
LOCK_ESCALATION	CHAR(3)	lock_escalation - 锁定升级
PAST_ACTIVITIES_WRAPPED	CHAR(3)	第 953 页的『past_activities_wrapped -“合并过去活动列表”监视元素』
CLIENT_USERID	VARCHAR(255)	client_userid - 客户机用户标识
CLIENT_WRKSTNNAME	VARCHAR(255)	client_wrkstnname - 客户机工作站名称
CLIENT_APPLNAME	VARCHAR(255)	client_applname - 客户机应用程序名称
CLIENT_ACCTNG	VARCHAR(255)	client_acctng - 客户机记帐字符串
OBJECT_REQUESTED	VARCHAR(10)	第 928 页的『object_requested -“所请求对象”监视元素』
LOCK_NAME	CHAR(32)	lock_name - 锁定名称
LOCK_OBJECT_TYPE	VARCHAR(32)	lock_object_type - 等待的锁定对象类型
LOCK_OBJECT_TYPE_ID	CHAR(1) FOR BIT DATA	保留以供将来使用
LOCK_ATTRIBUTES	CHAR(8)	lock_attributes - 锁定属性
LOCK_CURRENT_MODE	BIGINT	lock_current_mode - 转换前的原始锁定方式
LOCK_MODE_REQUESTED	BIGINT	lock_mode_requested - 请求的锁定方式
LOCK_MODE	BIGINT	lock_mode - 锁定方式

表 25. 为锁定事件监视器返回的信息: 表名: LOCK_PARTICIPANTS (续)

列名	数据类型	描述
LOCK_COUNT	BIGINT	lock_count - 锁定计数
LOCK_HOLD_COUNT	BIGINT	lock_hold_count - 锁定持有计数
LOCK_RRIID	BIGINT	
LOCK_STATUS	BIGINT	lock_status - 锁定状态
LOCK_RELEASE_FLAGS	CHAR(8)	lock_release_flags - 锁定释放标志
LOCK_WAIT_START_TIME	TIMESTAMP	lock_wait_start_time - 锁定等待开始时间戳记
LOCK_WAIT_END_TIME	TIMESTAMP	第 863 页的『lock_wait_end_time -"锁定等待结束时间戳记"监视元素』
QUEUED_AGENTS	BIGINT	第 1095 页的『queued_agents -"已排队阈值代理程序数"监视元素』
QUEUE_START_TIME	TIMESTAMP	第 1094 页的『queue_start_time -"队列开始时间戳记"监视元素』
TABLE_FILE_ID	BIGINT	table_file_id - 表文件标识
TABLE_NAME	VARCHAR(128)	table_name - 表名
TABLE_SCHEMA	VARCHAR(128)	table_schema - 表模式名
TABLESPACE_NAME	VARCHAR(128)	tablespace_name - 表空间名称
THRESHOLD_ID	INTEGER	第 1228 页的『thresholdid -"阈值标识"监视元素』
THRESHOLD_NAME	VARCHAR(128)	threshold_name - 阈值名称
UTILITY_INVOCATION_ID	VARCHAR(32) FOR BIT DATA	
XID	VARCHAR(140) FOR BIT DATA)	xid - 事务标识

表 26. 为锁定事件监视器返回的信息: 表名: LOCK_PARTICIPANT_ACTIVITIES

列名	数据类型	描述
XMLID	VARCHAR(256) NOT NULL	第 1346 页的『xmlid -"XML 标识"监视元素』
PARTICIPANT_NO	INTEGER	participant_no - 死锁参与者
ACTIVITY_ID	INTEGER	activity_id - 活动标识
ACTIVITY_TYPE	VARCHAR(10)	activity_type - 活动类型
UOW_ID	INTEGER	uow_id - 工作单元标识
PACKAGE_NAME	VARCHAR(128)	package_name - 程序包名
PACKAGE_SCHEMA	VARCHAR(128)	package_schema - 程序包模式
PACKAGE_VERSION_ID	VARCHAR(64)	package_version_id - 程序包版本
CONSISTENCY_TOKEN	VARCHAR(8)	consistency_token - 程序包一致性标记
SECTION_NUMBER	BIGINT	section_number - 节号
REOPT	VARCHAR(10)	第 1104 页的『reopt -"REOPT 绑定选项"监视元素』
INCREMENTAL_BIND	CHAR(3)	第 816 页的『incremental_bind -"增量绑定"监视元素』

表 26. 为锁定事件监视器返回的信息: 表名: LOCK_PARTICIPANT_ACTIVITIES (续)

列名	数据类型	描述
EFFECTIVE_ISOLATION	CHAR(2)	effective_isolation - 有效隔离级别
EFFECTIVE_QUERY_DEGREE	BIGINT	effective_query_degree - 有效查询并行度
STMT_LOCK_TIMEOUT	INTEGER	stmt_lock_timeout - 语句锁定超时
STMT_TYPE	BIGINT	stmt_type - 语句类型
STMT_QUERY_ID	BIGINT	stmt_query_id - 语句查询标识
STMT_NEST_LEVEL	BIGINT	stmt_nest_level - 语句嵌套级别
STMT_INVOCATION_ID	BIGINT	stmt_invocation_id - 语句调用标识
STMT_OPERATION	VARCHAR(128)	第 1173 页的『stmt_operation/operation -"语句操作"监视元素』
STMT_SOURCE_ID	BIGINT	stmt_source_id - 语句源标识
STMT_PKGCACHE_ID	BIGINT	stmt_pkgcache_id - 语句程序包高速缓存标识
STMT_FIRST_USE_TIME	TIMESTAMP	stmt_first_use_time - 第一次使用语句时的时间戳记
STMT_LAST_USE_TIME	TIMESTAMP	stmt_last_use_time - 上一次使用语句时的时间戳记
STMT_TEXT	CLOB(2097152)	stmt_text - SQL 语句文本
STMT_UNICODE	CHAR(3)	第 1181 页的『stmt_unicode -"语句 Unicode 标志"监视元素』
QUERY_ACTUAL_DEGREE	INTEGER	query_actual_degree - 实际运行时分区内并行度

表 27. 为锁定事件监视器返回的信息: 表名: LOCK_ACTIVITY_VALUES

列名	数据类型	描述
XMLID	VARCHAR(256) NOT NULL	第 1346 页的『xmlid -"XML 标识"监视元素』
PARTICIPANT_NO	INTEGER	participant_no - 死锁参与者
ACTIVITY_ID	INTEGER	activity_id - 活动标识
UOW_ID	INTEGER	uow_id - 工作单元标识
STMT_VALUE_INDEX	INTEGER	stmt_value_index - 值索引
STMT_VALUE_ISREOPT	INTEGER	stmt_value_isreopt - 用于语句重新优化的变量
STMT_VALUE_ISNULL	INTEGER	stmt_value_isnull - 包含空值
STMT_VALUE_TYPE	CHAR(16)	stmt_value_type - 值类型
STMT_VALUE_DATA	CLOB (32K)	stmt_value_data - 值数据

EVMON_FORMAT_UE_TO_XML 为锁定事件监视器写至 XML 的信息:

EVMON_FORMAT_UE_TO_XML 表函数为锁定事件监视器写入的信息。`sqllib/misc/DB2EvmonLocking.xsd` 文件中也记录了此信息。

db2_lock_event

用于详细描述锁定超时、锁定等待或死锁事件的主要模式。

元素内容: ((『db2_deadlock_graph』 {zero or one times (?)} , 『db2_participant』 {one or more (+)}) | (第 129 页的『db2_message』, 第 129 页的『db2_event_file』))

属性:

QName	类型	修订时间	缺省值	用法	注释
id	xs:long			必需	
类型				必需	
时间戳记	xs:dateTime			必需	
成员				必需	
release	xs:long			必需	
任何名称空间中的任何属性					

db2_deadlock_graph

此模式元素表示 DB2 死锁图。此图概述死锁涉及的所有参与者。

包含者: 『db2_lock_event』

元素内容: (第 139 页的『db2_participant』 {one or more (+)})

属性:

QName	类型	修订时间	缺省值	用法	注释
dl_conns	xs:int			必需	
rolled_back _participant_no	xs:int			必需	
类型			必需		
任何名称空间中的任何属性					

db2_participant

此模式元素表示锁定事件所涉及的所有参与者的应用程序信息。

包含者: 『db2_lock_event』 『db2_deadlock_graph』

元素内容: (第 133 页的『db2_object_requested』 {zero or one times (?)} , 第 134 页的『db2_app_details』, 第 134 页的『db2_activity』 {zero or more (*)})

属性:

QName	类型	修订时间	缺省值	用法	注释
否	xs:int			必需	
类型			必需		
participant_no _holding_lk	xs:int			可选	
deadlock_member	xs:int			可选	
任何名称空间中的任何属性					

db2_message

错误消息

包含者: 第 128 页的『db2_lock_event』

db2_event_file

事件已被写入的文件的标准路径。

包含者: 第 128 页的『db2_lock_event』

application_handle

应用程序在系统范围内的唯一标识。有关更多详细信息，请参阅监视元素 第 601 页的『agent_id -“应用程序句柄（代理程序标识）”监视元素』。

包含者: 第 134 页的『db2_app_details』

appl_id

当应用程序连接到数据库管理器上的数据库时，将生成此标识。有关更多详细信息，请参阅监视元素 第 616 页的『appl_id -“应用程序标识”监视元素』。

包含者: 第 134 页的『db2_app_details』

appl_name

客户机上运行的应用程序在数据库中的名称。有关更多详细信息，请参阅监视元素 第 620 页的『appl_name -“应用程序名称”监视元素』。

包含者: 第 134 页的『db2_app_details』

auth_id

调用受监视应用程序的用户的授权标识。有关更多详细信息，请参阅监视元素 第 635 页的『auth_id -“授权标识”』。

包含者: 第 134 页的『db2_app_details』

agent_tid

包含者: 第 134 页的『db2_app_details』

元素内容:

类型	构面
xs:long	

coord_agent_tid

包含者: 第 134 页的『db2_app_details』

元素内容:

类型	构面
xs:long	

agent_status

应用程序的当前状态。有关更多详细信息，请参阅监视元素 第 623 页的『appl_status - 应用程序状态监视元素』。

包含者: 第 134 页的『db2_app_details』

属性:

QName	类型	修订时间	缺省值	用法	注释
id	xs:int			可选	

appl_action

客户机应用程序正在执行的操作/请求

包含者: 第 134 页的『db2_app_details』

属性:

QName	类型	修订时间	缺省值	用法	注释
id	xs:int			可选	

lock_timeout_val

数据库配置参数“锁定超时”。值以秒计。有关更多详细信息，请参阅监视元素 第 859 页的『lock_timeout_val -“锁定超时值”监视元素』。

包含者: 第 134 页的『db2_app_details』

元素内容:

类型	构面
xs:long	

lock_wait_val

在锁定事件发生期间起作用的锁定等待参数。这是数据库配置参数 MON_LKWAIT_THRSH 或者在工作负载级别指定的 COLLECT LOCK WAIT DATA 设置。值以毫秒计。

包含者: 第 134 页的『db2_app_details』

元素内容:

类型	构面
xs:long	

tentry_state

TEntry 状态。仅供内部使用。

包含者: 第 134 页的『db2_app_details』

属性:

QName	类型	修订时间	缺省值	用法	注释
id	xs:int			可选	

tentry_flag1

TEntry 标志 1。仅供内部使用。

包含者: 第 134 页的『db2_app_details』

tentry_flag2

TEntry 标志 2。仅供内部使用。

包含者: 第 134 页的『db2_app_details』

xid

XID - 全局事务标识。

包含者: 第 134 页的『db2_app_details』

workload_id

此应用程序所属的工作负载的标识。有关更多详细信息，请参阅监视元素 第 1341 页的『workload_id -“工作负载标识”监视元素』。

包含者: 第 134 页的『db2_app_details』

workload_name

此应用程序所属的工作负载的名称。有关更多详细信息，请参阅监视元素 第 1342 页的『workload_name -“工作负载名称”监视元素』。

包含者: 第 134 页的『db2_app_details』

service_class_id

此应用程序所属的服务子类的标识。有关更多详细信息，请参阅监视元素 第 1134 页的『service_class_id -“服务类标识”监视元素』。

包含者: 第 134 页的『db2_app_details』

service_subclass_name

此应用程序所属的服务子类的名称。有关更多详细信息，请参阅监视元素 第 1135 页的『service_subclass_name -“服务子类名”监视元素』。

包含者: 第 134 页的『db2_app_details』

current_request

当前正在处理或最近处理的操作。

包含者: 第 134 页的『db2_app_details』

lock_escalation

指示锁定请求是否作为锁定升级的组成部分。有关更多详细信息，请参阅监视元素 第 845 页的『lock_escalation -“锁定升级”监视元素』。可能的值包括: Yes 或 No。

包含者: 第 134 页的『db2_app_details』

past_activities_wrapped

指示活动列表是否已合并。对任何一个应用程序所保留的先前活动数的缺省限制是 250。可使用注册表变量 DB2_MAX_INACT_STMTS 覆盖此缺省值。用户可能希望选择另一个限制值，以便增加或减少用于不活动语句信息的系统监视器堆空间量。

包含者: 第 134 页的『db2_app_details』

client_userid

由事务管理器生成并提供给服务器的客户机用户标识。有关更多详细信息，请参阅监视元素 第 664 页的『client_userid -“客户机用户标识”监视元素』。

包含者: 第 134 页的『db2_app_details』

client_wrkstnname

如果在此连接中发出了 sqleseti API，那么此元素标识客户机系统或工作站。有关更多详细信息，请参阅监视元素 第 665 页的『client_wrkstnname -“客户机工作站名称”监视元素』。

包含者: 第 134 页的『db2_app_details』

client_applname

如果在此连接中发出了 sqleseti API，那么此元素标识执行事务的服务器事务程序。有关更多详细信息，请参阅监视元素 第 658 页的『client_applname -“客户机应用程序名称”监视元素』。

包含者: 第 134 页的『db2_app_details』

client_acctng

如果在此连接中发出了 sqleseti API，那么此元素是为了进行日志记录和诊断而传递到目标数据库的数据。有关更多详细信息，请参阅监视元素 第 657 页的『client_acctng -“客户机记帐字符串”监视元素』。

包含者: 第 134 页的『db2_app_details』

utility_invocation_id

包含者: 第 134 页的『db2_app_details』

service_superclass_name

此应用程序所属的服务超类的名称。有关更多详细信息，请参阅监视元素 第 1136 页的『service_superclass_name -“服务超类名”监视元素』。

包含者: 第 134 页的『db2_app_details』

db2_object_requested

此模式元素表示请求者正在尝试获取但被所有者挂起的 DB2 锁定。

包含者: 第 128 页的『db2_participant』

元素内容: ((第 134 页的『lock_name』, 第 134 页的『lock_object_type』, 第 135 页的『lock_specifics』, 第 135 页的『lock_attributes』, 第 135 页的『lock_current_mode』, 第 135 页的『lock_mode_requested』, 第 135 页的『lock_mode』, 第 136 页的『lock_count』, 第 136 页的『lock_hold_count』, 第 136 页的『lock_rriid』, 第 136 页的『lock_status』, 第 137 页的『lock_release_flags』, 第 137 页的『tablespace_name』, 第 137 页的『table_name』, 第 137 页的『table_schema』, 第 137 页的『lock_object_type_id』, 第 138 页的『lock_wait_start_time』, 第 138 页的『lock_wait_end_time』, ANY content (skip) {zero or more (*)}) | (第 138 页的『threshold_name』, 第 138 页的『threshold_id』, 第 138 页的『queued_agents』, 第 139 页的『queue_start_time』, ANY content (skip) {zero or more (*)}))

属性:

QName	类型	修订时间	缺省值	用法	注释
类型			必需		

db2_app_details

此模式元素表示关于此参与者的详细信息。

包含者: 第 128 页的『db2_participant』

元素内容: (第 129 页的『application_handle』, 第 129 页的『appl_id』, 第 129 页的『appl_name』, 第 129 页的『auth_id』, 第 130 页的『agent_tid』, 第 130 页的『coord_agent_tid』, 第 130 页的『agent_status』, 第 130 页的『appl_action』, 第 130 页的『lock_timeout_val』, 第 131 页的『lock_wait_val』, 第 131 页的『tentry_state』, 第 131 页的『tentry_flag1』, 第 131 页的『tentry_flag2』, 第 131 页的『xid』, 第 131 页的『workload_id』, 第 132 页的『workload_name』, 第 132 页的『service_class_id』, 第 132 页的『service_subclass_name』, 第 132 页的『current_request』, 第 132 页的『lock_escalation』, 第 132 页的『past_activities_wrapped』, 第 132 页的『client_userid』, 第 132 页的『client_wrkstnname』, 第 133 页的『client_applname』, 第 133 页的『client_acctng』, 第 133 页的『utility_invocation_id』, 第 133 页的『service_superclass_name』, ANY content (skip) {zero or more (*)})

db2_activity

应用程序当前正在执行或已执行的所有 DB2 活动的列表。

包含者: 第 128 页的『db2_participant』

元素内容: (第 144 页的『db2_activity_details』, 第 144 页的『db2_input_variable』 {zero or more (*)})

属性:

QName	类型	修订时间	缺省值	用法	注释
类型			必需		
任何名称空间中的任何属性					

lock_name

内部二进制锁定名称。此元素将充当锁定的唯一标识。有关更多详细信息，请参阅监视元素 第 854 页的『lock_name -“锁定名称”监视元素』。

包含者: 第 133 页的『db2_object_requested』

lock_object_type

应用程序正在等待锁定的对象的类型。有关更多详细信息，请参阅监视元素 第 856 页的『lock_object_type -“等待的锁定对象类型”监视元素』。

包含者: 第 133 页的『db2_object_requested』

属性:

QName	类型	修订时间	缺省值	用法	注释
id	xs:long			可选	

lock_specifics

关于锁定的内部具体信息。仅供参考。

包含者: 第 133 页的『db2_object_requested』

lock_attributes

锁定属性。有关更多详细信息，请参阅监视元素 第 842 页的『lock_attributes -“锁定属性”监视元素』。

包含者: 第 133 页的『db2_object_requested』

lock_current_mode

转换之前的原始锁定。有关更多详细信息，请参阅监视元素 第 844 页的『lock_current_mode -“转换前的原始锁定方式”监视元素』。

包含者: 第 133 页的『db2_object_requested』

属性:

QName	类型	修订时间	缺省值	用法	注释
id	xs:long			可选	
mode				可选	

lock_mode_requested

此参与者所请求的锁定方式。有关更多详细信息，请参阅监视元素 第 853 页的『lock_mode_requested -“请求的锁定方式”监视元素』。

包含者: 第 133 页的『db2_object_requested』

属性:

QName	类型	修订时间	缺省值	用法	注释
id	xs:long			可选	
mode				可选	

lock_mode

正在挂起的锁定的类型。有关更多详细信息，请参阅监视元素 第 852 页的『lock_mode -“锁定方式”监视元素』。

包含者: 第 133 页的『db2_object_requested』

属性:

QName	类型	修订时间	缺省值	用法	注释
id	xs:long			可选	
mode				可选	

lock_count

正在挂起的锁定上的锁定数目。有关更多详细信息，请参阅监视元素 第 843 页的『lock_count -“锁定计数”监视元素』。

包含者: 第 133 页的『db2_object_requested』

元素内容:

类型	构面
xs:long	

lock_hold_count

挂起锁定的次数。有关更多详细信息，请参阅监视元素 第 851 页的『lock_hold_count -“锁定挂起计数”监视元素』。

包含者: 第 133 页的『db2_object_requested』

元素内容:

类型	构面
xs:long	

lock_rriid

行锁定的 IID。仅供内部使用。

包含者: 第 133 页的『db2_object_requested』

元素内容:

类型	构面
xs:long	

lock_status

指示锁定的内部状态。有关更多详细信息，请参阅监视元素 第 858 页的『lock_status -“锁定状态”监视元素』。

包含者: 第 133 页的『db2_object_requested』

属性:

QName	类型	修订时间	缺省值	用法	注释
id	xs:int			可选	

lock_release_flags

锁定释放标志。有关更多详细信息，请参阅监视元素 第 858 页的『lock_release_flags -“锁定释放标志”监视元素』。

包含者: 第 133 页的『db2_object_requested』

tablespace_name

在其中挂起锁定的表空间的名称。有关更多详细信息，请参阅监视元素 第 1200 页的『tablespace_name -“表空间名称”监视元素』。

包含者: 第 133 页的『db2_object_requested』

属性:

QName	类型	修订时间	缺省值	用法	注释
id	xs:long			可选	

table_name

在其中挂起锁定的表的名称。有关更多详细信息，请参阅监视元素 第 1191 页的『table_name -“表名”监视元素』。

包含者: 第 133 页的『db2_object_requested』

属性:

QName	类型	修订时间	缺省值	用法	注释
id	xs:long			可选	
data_member_id				可选	为其返回信息的数据成员的标识。

table_schema

表的模式。有关更多详细信息，请参阅监视元素 第 1193 页的『table_schema -“表模式名”监视元素』。

包含者: 第 133 页的『db2_object_requested』

lock_object_type_id

应用程序正在等待锁定的对象的类型。有关更多详细信息，请参阅监视元素 第 856 页的『lock_object_type -“等待的锁定对象类型”监视元素』。

包含者: 第 133 页的『db2_object_requested』

lock_wait_start_time

应用程序开始等待获取对于某个对象（该对象当前已被锁定所有者锁定）的锁定的日期和时间。有关更多详细信息，请参阅监视元素 第 863 页的『lock_wait_start_time -“锁定等待开始时间戳记”监视元素』。

包含者: 第 133 页的『db2_object_requested』

元素内容:

类型	构面
xs:dateTime	

lock_wait_end_time

应用程序停止等待获取对于某个对象（该对象当前已被锁定所有者锁定）的锁定的日期和时间。

包含者: 第 133 页的『db2_object_requested』

元素内容:

类型	构面
xs:dateTime	

threshold_name

阈值队列的名称。

包含者: 第 133 页的『db2_object_requested』

threshold_id

阈值队列的标识。

包含者: 第 133 页的『db2_object_requested』

queued_agents

当前正在阈值队列中进行排队的代理程序的总数。

包含者: 第 133 页的『db2_object_requested』

元素内容:

类型	构面
xs:long	

queue_start_time

应用程序开始在队列中等待获取阈值凭单的日期和时间。

包含者: 第 133 页的『db2_object_requested』

元素内容:

类型	构面
xs:dateTime	

db2_participant

此模式元素表示死锁图中的单一堆栈条目。

包含者: 第 128 页的『db2_lock_event』 第 128 页的『db2_deadlock_graph』

属性:

QName	类型	修订时间	缺省值	用法	注释
否	xs:int			必需	
deadlock_member				必需	
participant_no _holding_lk	xs:int			必需	
application_handle				必需	
任何名称空间中的任何属性					

activity_id

用于唯一地标识给定工作单元中某个应用程序的活动的计数器。有关更多详细信息，请参阅监视元素 第 597 页的『activity_id -“活动标识”监视元素』。

包含者: 第 144 页的『db2_activity_details』

uow_id

此活动记录所应用于的工作单元标识。有关更多详细信息，请参阅监视元素 第 1316 页的『uow_id -“工作单元标识”监视元素』。

包含者: 第 144 页的『db2_activity_details』

package_name

当前正在执行的 SQL 语句所在程序包的名称。有关更多详细信息，请参阅监视元素 第 941 页的『package_name -“程序包名”监视元素』。

包含者: 第 144 页的『db2_activity_details』

package_schema

与 SQL 语句相关联的程序包的模式名。有关更多详细信息，请参阅监视元素 第 942 页的『package_schema -“程序包模式”监视元素』。

包含者: 第 144 页的『db2_activity_details』

package_version_id

程序包版本指定当前正在执行的 SQL 语句所在程序包的版本标识。有关更多详细信息，请参阅监视元素 第 943 页的『package_version_id -“程序包版本”监视元素』。

包含者: 第 144 页的『db2_activity_details』

consistency_token

程序包一致性标记帮助标识当前正在执行的 SQL 语句所在程序包的版本。有关更多详细信息，请参阅监视元素 第 684 页的『consistency_token -“程序包一致性标记”监视元素』。

包含者: 第 144 页的『db2_activity_details』

section_number

程序包中用于当前正在处理或最新处理的 SQL 语句的内部节号。有关更多详细信息，请参阅监视元素 第 1128 页的『section_number -“节号”监视元素』。

包含者: 第 144 页的『db2_activity_details』

元素内容:

类型	构面
xs:long	

reopt

用于预编译此包的 REOPT 绑定选项。可能的值包括: NONE、ONCE 和 ALWAYS。有关更多详细信息，请参阅 REOPT 绑定选项。

包含者: 第 144 页的『db2_activity_details』

incremental_bind

此程序包在执行时以增量方式进行绑定。可能的值包括: Yes 或 No。

包含者: 第 144 页的『db2_activity_details』

effective_isolation

运行 SQL 语句时作用于该语句的隔离值。有关更多详细信息，请参阅监视元素 第 750 页的『effective_isolation -“有效隔离级别”监视元素』。

包含者: 第 144 页的『db2_activity_details』

属性:

QName	类型	修订时间	缺省值	用法	注释
id	xs:long			可选	

effective_query_degree

运行 SQL 语句时作用于该语句的程度值。有关更多详细信息，请参阅监视元素 第 750 页的『effective_query_degree -“有效查询并行度”监视元素』。

包含者: 第 144 页的『db2_activity_details』

元素内容:

类型	构面
xs:long	

stmt_unicode

SQL 语句 Unicode 标志。可能的值包括: Yes 或 No。

包含者: 第 144 页的『db2_activity_details』

stmt_lock_timeout

运行 SQL 语句时作用于该语句的锁定超时值。有关更多详细信息，请参阅监视元素 第 1172 页的『stmt_lock_timeout -“语句锁定超时”监视元素』。

包含者: 第 144 页的『db2_activity_details』

元素内容:

类型	构面
xs:int	

stmt_type

所处理的 SQL 语句的类型。可能的值包括: Dynamic 或 Static。有关更多详细信息，请参阅监视元素 第 1179 页的『stmt_type -“语句类型”监视元素』。

包含者: 第 144 页的『db2_activity_details』

属性:

QName	类型	修订时间	缺省值	用法	注释
id	xs:long			必需	

stmt_operation

包含者: 第 144 页的『db2_activity_details』

stmt_query_id

对任何 SQL 语句指定的内部查询标识。有关更多详细信息，请参阅监视元素 第 1175 页的『stmt_query_id -“语句查询标识”监视元素』。

包含者: 第 144 页的『db2_activity_details』

元素内容:

类型	构面
xs:long	

stmt_nest_level

此元素包含运行语句时起作用的嵌套级别或递归级别。有关更多详细信息，请参阅监视元素 第 1172 页的『stmt_nest_level -“语句嵌套级别”监视元素』。

包含者: 第 144 页的『db2_activity_details』

元素内容:

类型	构面
xs:long	

stmt_invocation_id

此元素包含运行 SQL 语句的例程调用的标识。有关更多详细信息，请参阅监视元素 第 1170 页的『stmt_invocation_id -“语句调用标识”监视元素』。

包含者: 第 144 页的『db2_activity_details』

元素内容:

类型	构面
xs:long	

stmt_source_id

此元素显示为已运行的 SQL 语句的源代码指定的内部标识。有关更多详细信息，请参阅监视元素 第 1176 页的『stmt_source_id -“语句源标识”』。

包含者: 第 144 页的『db2_activity_details』

元素内容:

类型	构面
xs:long	

stmt_pkgcache_id

此元素显示动态 SQL 语句的内部程序包高速缓存标识。有关更多详细信息，请参阅监视元素 第 1174 页的『stmt_pkgcache_id -“语句程序包高速缓存标识”监视元素』。

包含者: 第 144 页的『db2_activity_details』

元素内容:

类型	构面
xs:long	

stmt_text

SQL 语句的文本。有关更多详细信息，请参阅监视元素 第 1178 页的『stmt_text -“SQL 语句文本”监视元素』。

包含者: 第 144 页的『db2_activity_details』

stmt_first_use_time

此元素显示第一次处理语句条目的时间。对于游标操作，第 1169 页的『stmt_first_use_time -“第一次使用语句时的时间戳记”监视元素』将显示打开游标的时间。在应用程序协调节点，此值反映应用程序请求；在非协调程序节点，此值反映从原始节点接收请求的时间。有关更多详细信息，请参阅监视元素 stmt_first_use_time。

包含者: 第 144 页的『db2_activity_details』

元素内容:

类型	构面
xs:dateTime	

stmt_last_use_time

此元素显示上一次处理语句条目的时间。对于游标操作，第 1171 页的『stmt_last_use_time -“上一次使用语句时的时间戳记”监视元素』显示游标上操作可能为打开、访存或关闭的上一次操作的时间。在应用程序协调节点，此值反映应用程序请求；在非协调程序节点，此值反映从原始节点接收请求的时间。有关更多详细信息，请参阅监视元素 stmt_last_use_time。

包含者: 第 144 页的『db2_activity_details』

元素内容:

类型	构面
xs:dateTime	

query_actual_degree

运行 SQL 语句时作用于该语句的实际运行时等级值。有关更多详细信息，请参阅监视元素 第 1091 页的『query_actual_degree -“实际运行时分区内并行度”监视元素』。

包含者: 『db2_activity_details』

元素内容:

类型	构面
xs:int	

db2_activity_details

模式表示有关此活动的详细信息

包含者: 第 134 页的『db2_activity』

元素内容: (第 139 页的『activity_id』, 第 139 页的『uow_id』, 第 139 页的『package_name』, 第 140 页的『package_schema』, 第 140 页的『package_version_id』, 第 140 页的『consistency_token』, 第 140 页的『section_number』, 第 140 页的『reopt』, 第 140 页的『incremental_bind』, 第 140 页的『effective_isolation』, 第 141 页的『effective_query_degree』, 第 141 页的『stmt_unicode』, 第 141 页的『stmt_lock_timeout』, 第 141 页的『stmt_type』, 第 141 页的『stmt_operation』, 第 142 页的『stmt_query_id』, 第 142 页的『stmt_nest_level』, 第 142 页的『stmt_invocation_id』, 第 142 页的『stmt_source_id』, 第 143 页的『stmt_pkgcache_id』, 第 143 页的『stmt_text』, 第 143 页的『stmt_first_use_time』, 第 143 页的『stmt_last_use_time』, 『query_actual_degree』, ANY content (skip) {zero or more (*)})

db2_input_variable

此模式元素表示与 SQL 语句相关联的输入变量的列表。

包含者: 第 134 页的『db2_activity』

元素内容: (『stmt_value_index』, 『stmt_value_isreopt』, 第 145 页的『stmt_value_isnull』, 第 145 页的『stmt_value_type』, 第 145 页的『stmt_value_data』, ANY content (skip) {zero or more (*)})

stmt_value_index

此元素表示 SQL 语句中使用的输入参数标记或主变量的位置。有关更多详细信息，请参阅监视元素 第 1182 页的『stmt_value_index -“值索引”』。

包含者: 『db2_input_variable』

stmt_value_isreopt

此元素指示该变量在语句重新优化期间是否已被使用。有关更多详细信息，请参阅监视元素 第 1183 页的『stmt_value_isreopt -“用于语句重新优化的变量”监视元素』。

包含者: 『db2_input_variable』

属性:

QName	类型	修订时间	缺省值	用法	注释
id	xs:int			必需	

stmt_value_isnull

此元素指示与 SQL 语句相关联的数据值是否为空值。有关更多详细信息，请参阅监视元素 第 1182 页的『stmt_value_isnull -“包含空值”监视元素』。

包含者: 第 144 页的『db2_input_variable』

属性:

QName	类型	修订时间	缺省值	用法	注释
id	xs:int			必需	

stmt_value_type

第 1184 页的『stmt_value_type -“值类型”监视元素』

包含者: 第 144 页的『db2_input_variable』

stmt_value_data

此元素包含与 SQL 语句相关联的数据值的字符串表示。有关更多详细信息，请参阅监视元素 第 1181 页的『stmt_value_data -“值数据”』。

包含者: 第 144 页的『db2_input_variable』

收集锁定事件数据并生成报告

您可以使用锁定事件监视器来收集锁定超时、锁定等待和死锁信息，以帮助确定和解决锁定问题。本任务描述以不可读格式在无格式事件表中收集锁定事件数据之后，如何获取可读的文本报告。

开始之前

要创建锁定事件监视器并收集锁定事件监视器数据，您必须具有 DBADM 或 SQLADM 权限。

关于此任务

锁定事件监视器用于收集有助于确定和解决锁定问题的相关信息。例如，锁定事件监视器收集的一些锁定事件信息如下:

- 导致锁定事件的锁定
- 正在请求或挂起导致锁定事件的锁定的应用程序
- 应用程序在锁定事件期间执行的操作

本任务提供有关收集指定工作负载的锁定事件数据的指示信息。在下列情况下，您可能想收集锁定事件数据:

- 使用 MON_GET_WORKLOAD 表函数时，您注意到锁定等待值超出正常情况。
- 应用程序在管理通知日志中返回 SQL 返回码 -911，原因码为 68，这表明“事务由于锁定超时而回滚”。有关更多详细信息，另请参阅 SQL0911N 消息。
- 请注意管理通知日志中的死锁事件消息（SQL 返回码 -911，原因码 2，这表明“事务由于死锁而回滚”）。此日志消息指出在两个应用程序之间发生锁定事件，例如在应用程序 A 与 B 之间发生此事件，其中 A 是工作负载 FINANCE 的组成部分，B 是工作负载 PAYROLL 的组成部分。有关更多详细信息，另请参阅 SQL0911N 消息。

限制

要查看数据值，您必须对 EVMON_FORMAT_UE_* 例程具有 EXECUTE 特权（SQLADM 和 DBADM 权限隐式地包含此特权）。您还必须对无格式事件表具有 SELECT 特权，缺省情况下，具有 DATAACCESS 权限的用户以及事件监视器及相关无格式事件表的创建者具有此特权。

过程

要收集关于将来有可能发生的锁定事件的详细信息，请执行以下步骤:

1. 使用 CREATE EVENT MONITOR FOR LOCKING 语句来创建名为 lockevmon 的锁定事件监视器，如以下示例所示:

   ```
   CREATE EVENT MONITOR lockevmon FOR LOCKING
      WRITE TO UNFORMATTED EVENT TABLE
   ```

   注: 以下列示创建事件监视器时要记住的要点:

   - 您可以提前创建事件监视器而不必担心耗用磁盘空间，这是因为，在数据库或工作负载级别激活数据收集操作之前不会写入任何内容
   - 在分区数据库环境中，请确保在分区表空间的所有节点上设置事件监视器。否则，在没有分区表空间的分区中，将丢失锁定事件。
   - 请确保设置表空间和缓冲池，以便最大程度地降低对表进行访问以获取数据期间执行的工作对高性能工作的影响。
2. 通过运行以下语句来激活名为 lockevmon 的锁定事件监视器:

   ```
   SET EVENT MONITOR lockevmon STATE 1
   ```

3. 要在工作负载级别启用锁定事件数据收集功能，请发出带有下列其中一个 COLLECT 子句的 ALTER WORKLOAD 语句: COLLECT LOCK TIMEOUT DATA、COLLECT DEADLOCK DATA 或 COLLECT LOCK WAIT DATA。对 COLLECT 子句指定 WITH HISTORY 选项。设置数据库配置参数将在数据库级别影响锁定事件数据收集，并且所有工作负载都将受影响。

   对于锁定等待事件

   要对 FINANCE 应用程序收集任何超过 5 秒钟才能获取的锁定的锁定等待数据，并且要对 PAYROLL 应用程序收集任何超过 10 秒才能获取的锁定的锁定等待数据，请发出下列语句:

   ```
   ALTER WORKLOAD finance COLLECT LOCK WAIT DATA WITH HISTORY AND VALUES
      FOR LOCKS WAITING MORE THAN 5 SECONDS
   ALTER WORKLOAD payroll COLLECT LOCK WAIT DATA
      FOR LOCKS WAITING MORE THAN 10 SECONDS WITH HISTORY
   ```

要使用 HIST_AND_VALUES 输入数据值对 SAMPLE 数据库设置 **mon_lockwait** 数据库配置参数，并且要将 **mon_lw_thresh** 数据库配置参数设置为 10 秒，请发出下列命令:

```
db2 update db cfg for sample using mon_lockwait hist_and_values
db2 update db cfg for sample using mon_lw_thresh 10000000
```

对于锁定超时事件

要对 FINANCE 和 PAYROLL 应用程序收集锁定超时数据，请发出下列语句:

```
ALTER WORKLOAD finance COLLECT LOCK TIMEOUT DATA WITH HISTORY
ALTER WORKLOAD payroll COLLECT LOCK TIMEOUT DATA WITH HISTORY
```

要使用 HIST_AND_VALUES 输入数据值对 SAMPLE 数据库设置 **mon_locktimeout** 数据库配置参数，请发出以下命令:

```
db2 update db cfg for sample using mon_locktimeout hist_and_values
```

对于死锁事件

要对 FINANCE 和 PAYROLL 应用程序收集数据，请发出下列语句:

```
ALTER WORKLOAD finance COLLECT DEADLOCK DATA WITH HISTORY
ALTER WORKLOAD payroll COLLECT DEADLOCK DATA WITH HISTORY
```

要使用 HIST_AND_VALUES 输入数据值对 SAMPLE 数据库设置 **mon_deadlock** 数据库配置参数，请发出以下命令:

```
db2 update db cfg for sample using mon_deadlock hist_and_values
```

4. 重新运行工作负载，以便接收另一个锁定事件通知。
5. 连接到数据库。
6. 使用下列其中一个过程来获取锁定事件报告:
   a. 使用 XML 解析器工具 **db2evmonfmt** 来生成基于无格式事件表中收集的事件数据并使用缺省样式表的纯文本报告，例如:

      ```
      java db2evmonfmt -d db_name -ue table_name -ftext -u user_id -p password
      ```

   b. 使用 EVMON_FORMAT_UE_TO_XML 表函数来获取 XML 文档。
   c. 使用 EVMON_FORMAT_UE_TO_TABLES 过程将该数据输出到关系表。
7. 分析报告，确定锁定事件问题的原因并解决该问题。
8. 通过运行下列语句或重置数据库配置参数，对 FINANCE 和 PAYROLL 应用程序关闭锁定数据收集功能:

对于锁定等待事件

```
ALTER WORKLOAD finance COLLECT LOCK WAIT DATA NONE
ALTER WORKLOAD payroll COLLECT LOCK WAIT DATA NONE
```

要使用缺省的 NONE 输入数据值对 SAMPLE 数据库重置 **mon_lockwait** 数据库配置参数，并且要将 **mon_lw_thresh** 数据库配置参数重置为缺省值（5 秒），请发出下列命令:

```
db2 update db cfg for sample using mon_lockwait none
db2 update db cfg for sample using mon_lw_thresh 5000000
```

对于锁定超时事件

```
ALTER WORKLOAD finance COLLECT LOCK TIMEOUT DATA NONE
ALTER WORKLOAD payroll COLLECT LOCK TIMEOUT DATA NONE
```

要使用缺省的 NONE 输入数据值对 SAMPLE 数据库重置 **mon_locktimeout** 数据库配置参数，请发出以下命令:

```
db2 update db cfg for sample using mon_locktimeout none
```

对于死锁事件

```
ALTER WORKLOAD finance COLLECT DEADLOCK DATA NONE
ALTER WORKLOAD payroll COLLECT DEADLOCK DATA NONE
```

要使用缺省的 WITHOUT_HIST 输入数据值对 SAMPLE 数据库重置 **mon_deadlock** 数据库配置参数，请发出以下命令:

```
db2 update db cfg for sample using mon_deadlock without_hist
```

下一步做什么

请重新运行应用程序，以确保锁定问题已不再存在。

工作单元事件监视

每当完成工作单元时（即，每当执行落实或回滚时），工作单元事件监视器都会记录一个事件。这些有关各工作单元的历史信息对于退款用途（按 CPU 使用情况付款）和监视是否符合响应时间服务级别目标很有用。

工作单元事件监视器是从系统角度监视请求度量值的一种方法。与工作单元事件监视器最紧密相关的替代项或补充项是统计信息事件监视器和 MON_GET_UNIT_OF_WORK 及 MON_GET_UNIT_OF_WORK_DETAILS 表函数。

可使用工作单元事件监视器来收集工作单元内使用的程序包列表以及使用该工作单元时所处的嵌套级别。此信息有助于顺利诊断存储过程。从 DB2 V10.1 开始，还可为工作单元内运行的语句生成可执行标识及关联语句级别度量值的列表。

要创建工作单元事件监视器并收集工作单元事件监视器数据，您必须具有 DBADM 或 SQLADM 权限。

创建工作单元事件监视器

从 DB2 V10.1 开始，可选择将工作单元事件监视器的输出写至无格式事件 (UE) 表或常规表。有关如何选择最适当的输出格式的更多信息，请参阅第 31 页的『事件监视器的输出选项』。

不管您使用哪种类型的表，在您创建工作单元事件监视器时，都应标识您计划存储包含该事件监视器的输出的表的表空间。建议的做法是配置专用表空间来存储该表。但是，创建事件监视器时可指定现有表空间。如果未指定表空间，那么系统将为您选择表空间。

要使用缺省值和最佳实践来创建工作单元事件监视器，请使用 CREATE EVENT MONITOR 语句。以下样本语句将尽可能使用缺省值，并指定输出将存储在 MY_EVMON_TABLESPACE 表空间内的 UE 表中:

```
CREATE EVENT MONITOR MY_UOW_EVMON
  FOR UNIT OF WORK
   WRITE TO UNFORMATTED EVENT TABLE (IN MY_EVMON_TABLESPACE)
```

配置数据收集功能

可对工作单元数据指定 4 个不同收集级别:

1. 无
2. 基本工作单元数据
   a. 有关在工作单元内运行的程序包的信息
   b. 在工作单元内运行的语句的可执行标识的列表。

可使用数据库配置参数来控制针对数据库中处于活动状态的所有工作单元事件监视器的工作单元数据收集。或者，要控制针对特定工作负载、服务类或工作操作的信息收集，可对适当的工作负载对象使用 CREATE 和 ALTER 语句。

要在数据库级别配置数据收集，请使用 **UPDATE DATABASE CONFIGURATION** 命令来设置 **mon_uow_data** 数据库配置参数及（可选）**mon_uow_pkglist** 和 **mon_uow_execlist** 数据库配置参数。这些参数值的可能组合显示在表 28中:

表 28. 工作单元事件监视器配置参数的可能值

要收集的数据	mon_uow_data	mon_uow_pkglist	mon_uow_execlist
不收集工作单元数据	NONE（缺省值）	OFF（缺省值）	OFF（缺省值）
仅收集基本工作单元数据	BASE	OFF（缺省值）	OFF（缺省值）
收集程序包列表信息，但不收集有关可执行标识的信息	BASE	ON	OFF（缺省值）
收集有关可执行标识的信息，但不收集程序包列表	BASE	OFF（缺省值）	ON
收集基本工作单元数据、程序包列表信息和有关可执行标识的信息	BASE	ON	ON

提示:

- 如果未设置任何配置参数，那么不会收集任何工作单元数据，除非您已启用针对特定工作负载对象的收集。可通过对适当工作负载对象类型使用 CREATE 或 ALTER 语句（例如，CREATE SERVICE CLASS 或 ALTER WORKLOAD 语句）来启用针对特定工作负载对象的收集。
- 要收集基本工作单元数据而不收集程序包列表或可执行标识信息，可将 **mon_uow_data** 配置参数设置为 BASE 并省略 **mon_uow_pkglist** 和 **mon_uow_execlist** 配置参数。如果未显式设置它们，那么会使用 OFF 的缺省值。
- 要收集程序包列表和/或可执行标识信息，还必须将 **mon_uow_data** 配置参数设置为 BASE。如果将 **mon_uow_data** 配置参数设置为 NONE，那么不会收集任何信息，不管 **mon_uow_pkglist** 和 **mon_uow_execlist** 配置参数的设置如何都是如此。

要控制针对特定工作负载对象的数据收集，请对您关心的特定工作负载对象类型使用 CREATE 或 ALTER 语句的 COLLECT UNIT OF WORK DATA 子句。例如，要收集工作负载 REPORTS 的基本工作单元事件数据和程序包列表信息，可发出类似如下的语句:

```
ALTER WORKLOAD REPORTS COLLECT UNIT OF WORK DATA BASE INCLUDE PACKAGE LIST
```

要收集程序包列表信息和工作单元中运行的语句的可执行标识列表，可使用以下语句:

```
ALTER WORKLOAD REPORTS COLLECT UNIT OF WORK DATA BASE INCLUDE PACKAGE LIST,
 EXECUTABLE LIST
```

第 149 页的表 28 中显示的设置将应用于系统中运行的所有工作负载，除非您使用 CREATE WORKLOAD 或 ALTER WORKLOAD 语句来覆盖特定工作负载的这些设置。如果不仅想要收集所有工作负载的基本级别信息，而且想要收集所选工作负载的程序包列表信息，请将 **mon_uow_data** 数据库配置参数设置为 BASE。然后，使用 CREATE WORKLOAD 或 ALTER WORKLOAD 语句对您关心的工作负载将级别设置为 BASE PACKAGE LIST。

缺省情况下，适用表函数和事件监视器（包括工作单元事件监视器）会收集并报告请求度量值。可按如下所示更改缺省设置:

- 通过使用 **mon_req_metrics** 数据库配置参数
- 通过对服务超类使用 CREATE SERVICE CLASS 或 ALTER SERVICE CLASS 语句的 COLLECT REQUEST METRICS 子句。

更改缺省设置会影响任何可报告请求度量值的表函数或事件监视器。

访问工作单元事件监视器捕获的事件数据

工作单元事件监视器可将数据写至常规表，也可将二进制格式的数据写至无格式事件 (UE) 表。可使用 SQL 来访问常规表中的数据。

要访问 UE 表中的数据，请使用下列其中一个表函数:

EVMON_FORMAT_UE_TO_XML
将无格式事件表中的数据抽取到 XML 文档中。

EVMON_FORMAT_UE_TO_TABLES
将无格式事件表中的数据抽取到一组关系表中。

如果使用这些表函数的其中一个，那么可通过包括 SELECT 语句作为该函数的其中一个参数来指定要抽取的数据。您可以对 SELECT 语句所提供的选择、排序和其他方面功能进行全面控制。

如果要生成程序包列表信息，那么可使用 EVMON_FORMAT_UE_TO_XML 表函数来生成单个 XML 文档，该文档包含基本工作单元事件监视器数据和程序包列表。EVMON_FORMAT_UE_TO_TABLES 过程将生成两个表，一个表用于保存基本工作单元事件监视器信息，另一个表用于保存程序包列表信息。可以使用 MEMBER、APPLICATION_ID 和 UOW_ID 列中的值来连接这两个表。

还可使用 **db2evmonfmt** 命令来帮助执行以下任务:

- 根据以下属性来选择您关心的事件: 事件标识、事件类型、时间段、应用程序、工作负载或服务类
- 选择是以文本报告形式还是带格式 XML 文档形式来接收输出
- 通过创建您自己的 XSLT 样式表（而不是使用 **db2evmonfmt** 命令所提供的 XSLT 样式表）来控制输出格式

例如，以下命令提供工作单元报告，它在数据库 SAMPLE 中选择过去 24 小时内发生的工作单元事件。可从名为 SAMPLE_UOW_EVENTS 的无格式事件表中获取这些事件记录。该命令通过使用 MyUOW.xsl 样式表来创建带格式文本输出。

```
java db2evmonfmt -d SAMPLE -ue SAMPLE_UOW_EVENTS -ftext -ss MyUOW.xsl -hours 24
```

工作单元事件监视器生成的数据

工作单元事件监视器生成有关系统中运行的工作单元（事务）的数据。可选择将工作单元事件监视器的输出写至常规表或无格式事件 (UE) 表。如果数据写至 UE 表，那么必须对该数据执行后处理才能查看该数据。

不管您选择什么输出格式，工作单元事件数据都来自下列四个逻辑组的其中一个:

- uow
- uow_metrics
- uow_package_list
- uow_exec_list

如果选择将工作单元事件数据写至常规表，那么另一个组 (CONTROL) 中的数据用于生成有关事件监视器本身的元数据。

注: 除非您另行指定，否则工作单元事件监视器仅收集请求度量值监视元素。为能够生成基本工作单元事件数据，或者要获取包列表数据或执行列表数据，必须显式启用数据收集。有关更多信息，请参阅第 105 页的『启用事件监视器数据收集』。

工作单元事件监视器写至表的信息:

在指定了 WRITE TO TABLE 选项的情况下工作单元事件监视器写入的信息。

选择 WRITE TO TABLE 作为工作单元事件监视器的输出类型时，缺省情况下，会生成五个表，每个表包含一个或多个逻辑数据组中的监视元素:

表 29. UNIT OF WORK 写至表事件监视器生成的表. 表名通过逻辑数据组（用于填充该表）的名称和 CREATE EVENT MONITOR 语句中给予事件监视器的名称（如下表的表名中的 *evmon-name* 所示）并置派生。

缺省表名	报告的逻辑数据组
UOW_*evmon-name*	uow
UOW_METRICS_*evmon-name*	uow_metrics
UOW_PACKAGE_LIST_*evmon-name*	uow_package_list
UOW_EXECUTABLE_LIST_*evmon-name*	uow_executable_list
CONTROL_*evmon-name*	CONTROL 逻辑组由来自 event_dbheader、event_start 和 event_overflow 逻辑数据组中的一个或多个的所选元素组成。

注: 即使缺省情况下生成全部五个表，也仍然必须确保已启用数据收集以获取要收集的锁定信息种类，否则某些列将报告空值。

要将事件监视器的输出限制为发送至特定表，请在 CREATE EVENT MONITOR 或 ALTER EVENT MONITOR 语句中指定要对其生成表的逻辑组的名称。有关详细信息，请参阅这些语句的参考主题。

生成的表

表 30. 对锁定事件监视器返回的信息: 缺省表名: UOW_evmon-name

列名	数据类型	描述
PARTITION_KEY	INTEGER	第 951 页的『partition_key -“分区键”监视元素』
APPLICATION_HANDLE	BIGINT	application_handle - 应用程序句柄
APPLICATION_ID	VARCHAR(128)	第 616 页的『appl_id -“应用程序标识”监视元素』
APPLICATION_NAME	VARCHAR(128)	第 620 页的『appl_name -“应用程序名称”监视元素』
CLIENT_ACCTNG	VARCHAR(255)	client_acctng - 客户机记帐字符串
CLIENT_APPLNAME	VARCHAR(255)	client_applname - 客户机应用程序名称
CLIENT_HOSTNAME	VARCHAR(255)	client_hostname - 客户机主机名
CLIENT_PID	BIGINT	client_pid - 客户机进程标识
CLIENT_PLATFORM	VARCHAR(12)	client_platform - 客户机操作平台
CLIENT_PORT_NUMBER	INTEGER	client_port_number - 客户机端口号
CLIENT_PRODUCT_ID	VARCHAR(128)	第 663 页的『client_prdid -“客户机产品和版本标识”监视元素』
CLIENT_PROTOCOL	VARCHAR(10)	client_protocol - 客户机通信协议
CLIENT_USERID	VARCHAR(255)	client_userid - 客户机用户标识
CLIENT_WRKSTNNAME	VARCHAR(255)	client_wrkstnname - 客户机工作站名称
COMPLETION_STATUS	VARCHAR(128)	completion_status - 完成状态
CONNECTION_TIME	TIMESTAMP	第 683 页的『connection_start_time -“连接开始时间”监视元素』
COORD_MEMBER	SMALLINT	coord_member - 协调程序成员
EVENT_ID	INTEGER NOT NULL	event_id -“事件标识”监视元素
EVENT_TIMESTAMP	TIMESTAMP NOT NULL	event_timestamp -“事件时间戳记”监视元素
EXECUTABLE_LIST_SIZE	BIGINT	第 762 页的『executable_list_size -“可执行列表大小”监视元素』
EXECUTABLE_LIST_TRUNCATED	CHAR(3)	第 763 页的『executable_list_truncated -“截断可执行列表”监视元素』
GLOBAL_TRANSACTION_ID	VARCHAR(40)	第 794 页的『global_transaction_id -“全局事务标识”监视元素』

表 30. 对锁定事件监视器返回的信息：缺省表名：UOW_evmon-name（续）

列名	数据类型	描述
INTRA_PARALLEL_STATE	VARCHAR(128)	intra_parallel_state - 分区内并行性的当前状态
LOCAL_TRANSACTION_ID	VARCHAR(16)	第 841 页的『local_transaction_id -“本地事务标识”监视元素』
MEMBER	SMALLINT	member - 数据库成员
MEMBER_ACTIVATION_TIME	TIMESTAMP	第 717 页的『db_conn_time -“数据库激活时间戳记”监视元素』
METRICS	BLOB	
MON_INTERVAL_ID	VARCHAR(128)	mon_interval_id - 监视时间间隔标识
PACKAGE_LIST_EXCEEDED	CHAR(3)	package_list_exceeded - 超过了程序包列表的容量
PACKAGE_LIST_SIZE	INTEGER	第 941 页的『package_list_size -“程序包列表大小”监视元素』
SERVICE_CLASS_ID	INTEGER	service_class_id - 服务类标识
SERVICE_SUBCLASS_NAME	VARCHAR(128)	service_subclass_name - 服务子类名
SERVICE_SUPERCLASS_NAME	VARCHAR(128)	service_superclass_name - 服务超类名
SESSION_AUTHID	VARCHAR(128)	第 1137 页的『session_auth_id -“会话授权标识”监视元素』
START_TIME	TIMESTAMP	start_time - 事件启动时间
STOP_TIME	TIMESTAMP	stop_time - 事件停止时间
SYSTEM_AUTHID	VARCHAR(128)	第 1188 页的『system_auth_id -“系统授权标识”监视元素』
UOW_ID	INTEGER	uow_id - 工作单元标识
UOW_LOG_SPACE_USED	BIGINT	uow_log_space_used - 使用的工作单元日志空间
WORKLOAD_ID	INTEGER	workload_id - 工作负载标识
WORKLOAD_NAME	VARCHAR(128)	workload_name - 工作负载名称
WORKLOAD_OCCURRENCE_ID	INTEGER	workload_occurrence_id - 工作负载项标识

表 31. 对工作单元事件监视器返回的信息：表名：UOW_METRICS_evmon-name

列名	数据类型	描述
PARTITION_KEY	INTEGER	第 951 页的『partition_key -“分区键”监视元素』
APPLICATION_ID	VARCHAR (128)	第 616 页的『appl_id -“应用程序标识”监视元素』
MEMBER	SMALLINT	member - 数据库成员
UOW_ID	INTEGER	uow_id - 工作单元标识
WLM_QUEUE_TIME_TOTAL	BIGINT	wlm_queue_time_total - 工作负载管理器队列时间总计

表 31. 对工作单元事件监视器返回的信息：表名：UOW_METRICS_evmon-name（续）

列名	数据类型	描述
WLM_QUEUE_ASSIGNMENTS _TOTAL	BIGINT	wlm_queue_assignments_total - 工作负载管理器队列分配总计
FCM_TQ_RECV_WAIT_TIME	BIGINT	fcm_tq_recv_wait_time - FCM 表队列接收等待时间
FCM_MESSAGE_RECV_WAIT_TIME	BIGINT	fcm_message_recv_wait_time - 接收 FCM 消息等待时间
FCM_TQ_SEND_WAIT_TIME	BIGINT	fcm_tq_send_wait_time - FCM 表队列发送等待时间
FCM_MESSAGE_SEND_WAIT_TIME	BIGINT	fcm_message_send_wait_time - 发送 FCM 消息等待时间
AGENT_WAIT_TIME	BIGINT	agent_wait_time - 代理程序等待时间
AGENT_WAITS_TOTAL	BIGINT	agent_waits_total - 等待代理程序总次数
LOCK_WAIT_TIME	BIGINT	lock_wait_time - 等待锁定时间
LOCK_WAITS	BIGINT	lock_waits - 等待锁定次数
DIRECT_READ_TIME	BIGINT	direct_read_time - 直接读时间
DIRECT_READ_REQS	BIGINT	direct_read_reqs - 直接读请求数
DIRECT_WRITE_TIME	BIGINT	direct_write_time - 直接写时间
DIRECT_WRITE_REQS	BIGINT	direct_write_reqs - 直接写请求数
LOG_BUFFER_WAIT_TIME	BIGINT	log_buffer_wait_time - 日志缓冲区等待时间
NUM_LOG_BUFFER_FULL	BIGINT	num_log_buffer_full - 日志缓冲区变满次数
LOG_DISK_WAIT_TIME	BIGINT	log_disk_wait_time - 日志磁盘等待时间
LOG_DISK_WAITS_TOTAL	BIGINT	log_disk_waits_total - 日志磁盘等待总次数
TCPIP_RECV_WAIT_TIME	BIGINT	tcpip_recv_wait_time - TCP/IP 接收等待时间
TCPIP_RECVS_TOTAL	BIGINT	tcpip_recvs_total - TCP/IP 接收总次数
CLIENT_IDLE_WAIT_TIME	BIGINT	client_idle_wait_time - 客户机空闲等待时间
IPC_RECV_WAIT_TIME	BIGINT	ipc_recv_wait_time - 进程间通信接收等待时间
IPC_RECVS_TOTAL	BIGINT	ipc_recvs_total - 进程间通信接收总次数
IPC_SEND_WAIT_TIME	BIGINT	ipc_send_wait_time - 进程间通信发送等待时间
IPC_SENDS_TOTAL	BIGINT	ipc_sends_total - 进程间通信发送总次数
TCPIP_SEND_WAIT_TIME	BIGINT	tcpip_send_wait_time - TCP/IP 发送等待时间
TCPIP_SENDS_TOTAL	BIGINT	tcpip_sends_total - TCP/IP 发送总次数
POOL_WRITE_TIME	BIGINT	pool_write_time - 缓冲池物理写时间总计
POOL_READ_TIME	BIGINT	pool_read_time - 缓冲池物理读时间总计
AUDIT_FILE_WRITE_WAIT_TIME	BIGINT	audit_file_write_wait_time - 审计文件写等待时间
AUDIT_FILE_WRITES_TOTAL	BIGINT	audit_file_writes_total - 写审计文件总次数
AUDIT_SUBSYSTEM_WAIT_TIME	BIGINT	audit_subsystem_wait_time - 审计子系统等待时间
AUDIT_SUBSYSTEM_WAITS_TOTAL	BIGINT	audit_subsystem_waits_total - 审计子系统等待总次数
DIAGLOG_WRITE_WAIT_TIME	BIGINT	diaglog_write_wait_time - 诊断日志文件写等待时间
DIAGLOG_WRITES_TOTAL	BIGINT	diaglog_writes_total - 写诊断日志文件总次数
FCM_SEND_WAIT_TIME	BIGINT	fcm_send_wait_time - FCM 发送等待时间
FCM_RECV_WAIT_TIME	BIGINT	fcm_recv_wait_time - FCM 接收等待时间
TOTAL_WAIT_TIME	BIGINT	total_wait_time - 等待时间总计
RQSTS_COMPLETED_TOTAL	BIGINT	rqsts_completed_total - 完成请求总数
TOTAL_RQST_TIME	BIGINT	total_rqst_time - 请求时间总计

表 31. 对工作单元事件监视器返回的信息: 表名: UOW_METRICS_evmon-name (续)

列名	数据类型	描述
APP_RQSTS_COMPLETED_TOTAL	BIGINT	app_rqsts_completed_total - 完成应用程序请求总数
TOTAL_APP_RQST_TIME	BIGINT	total_app_rqst_time - 应用程序请求时间总计
TOTAL_SECTION_SORT_PROC_TIME	BIGINT	total_section_sort_proc_time - 节排序处理时间总计
TOTAL_SECTION_SORTS	BIGINT	total_section_sorts - 节排序总次数
TOTAL_SECTION_SORT_TIME	BIGINT	total_section_sort_time - 节排序时间总计
ROWS_READ	BIGINT	rows_read - 读取行数
ROWS_MODIFIED	BIGINT	rows_modified - 修改的行数
POOL_DATA_L_READS	BIGINT	pool_data_l_reads - 缓冲池数据逻辑读取数
POOL_INDEX_L_READS	BIGINT	pool_index_l_reads - 缓冲池索引逻辑读取数
POOL_TEMP_DATA_L_READS	BIGINT	pool_temp_data_l_reads - 缓冲池临时数据逻辑读取数
POOL_TEMP_INDEX_L_READS	BIGINT	pool_temp_index_l_reads - 缓冲池临时索引逻辑读取数
POOL_XDA_L_READS	BIGINT	pool_xda_l_reads - 缓冲池 XDA 数据逻辑读取数
POOL_TEMP_XDA_L_READS	BIGINT	pool_temp_xda_l_reads - 缓冲池临时 XDA 数据逻辑读取数
TOTAL_CPU_TIME	BIGINT	total_cpu_time - CPU 时间总计
ACT_COMPLETED_TOTAL	BIGINT	act_completed_total - 完成活动总数
POOL_DATA_P_READS	BIGINT	pool_data_p_reads - 缓冲池数据物理读取数
POOL_TEMP_DATA_P_READS	BIGINT	pool_temp_data_p_reads - 缓冲池临时数据物理读取数
POOL_XDA_P_READS	BIGINT	pool_xda_p_reads - 缓冲池 XDA 数据物理读取数
POOL_TEMP_XDA_P_READS	BIGINT	pool_temp_xda_p_reads - 缓冲池临时 XDA 数据物理读取数
POOL_INDEX_P_READS	BIGINT	pool_index_p_reads - 缓冲池索引物理读取数
POOL_TEMP_INDEX_P_READS	BIGINT	pool_temp_index_p_reads - 缓冲池临时索引物理读取数
POOL_DATA_WRITES	BIGINT	pool_data_writes - 缓冲池数据写次数
POOL_XDA_WRITES	BIGINT	pool_xda_writes - 缓冲池 XDA 数据写次数
POOL_INDEX_WRITES	BIGINT	pool_index_writes - 缓冲池索引写次数
DIRECT_READS	BIGINT	direct_reads - 直接读数据库数目
DIRECT_WRITES	BIGINT	direct_writes - 直接写数据库数目
ROWS_RETURNED	BIGINT	rows_returned - 返回的行数
DEADLOCKS	BIGINT	deadlocks - 检测到的死锁数
LOCK_TIMEOUTS	BIGINT	lock_timeouts - 锁定超时次数
LOCK_ESCALS	BIGINT	lock_escals - 锁定升级次数
FCM_SENDS_TOTAL	BIGINT	fcm_sends_total - FCM 发送总计
FCM_RECVS_TOTAL	BIGINT	fcm_recvs_total - FCM 接收总计
FCM_SEND_VOLUME	BIGINT	fcm_send_volume - FCM 发送量
FCM_RECV_VOLUME	BIGINT	fcm_recv_volume - FCM 接收量
FCM_MESSAGE_SENDS_TOTAL	BIGINT	fcm_message_sends_total - 发送 FCM 消息总数
FCM_MESSAGE_RECVS_TOTAL	BIGINT	fcm_message_recvs_total - 接收 FCM 消息总数
FCM_MESSAGE_SEND_VOLUME	BIGINT	fcm_message_send_volume - 发送 FCM 消息量

表 31. 对工作单元事件监视器返回的信息：表名：UOW_METRICS_evmon-name（续）

列名	数据类型	描述
FCM_MESSAGE_RECV_VOLUME	BIGINT	fcm_message_recv_volume - 接收 FCM 消息量
FCM_TQ_SENDS_TOTAL	BIGINT	fcm_tq_sends_total - FCM 表队列发送总次数
FCM_TQ_RECVS_TOTAL	BIGINT	fcm_tq_recvs_total - FCM 表队列接收总量
FCM_TQ_SEND_VOLUME	BIGINT	fcm_tq_send_volume - FCM 表队列发送量
FCM_TQ_RECV_VOLUME	BIGINT	fcm_tq_recv_volume - FCM 表队列接收量
TQ_TOT_SEND_SPILLS	BIGINT	tq_tot_send_spills - 溢出表队列缓冲区总数
TCPIP_SEND_VOLUME	BIGINT	tcpip_send_volume - TCP/IP 发送量
TCPIP_RECV_VOLUME	BIGINT	tcpip_recv_volume - TCP/IP 接收量
IPC_SEND_VOLUME	BIGINT	ipc_send_volume - 进程间通信发送量
IPC_RECV_VOLUME	BIGINT	ipc_recv_volume - 进程间通信接收量
POST_THRESHOLD_SORTS	BIGINT	post_threshold_sorts - 超过阈值后的排序数
POST_SHRTHRESHOLD_SORTS	BIGINT	post_shrthreshold_sorts - 共享阈值后排序数
SORT_OVERFLOWS	BIGINT	sort_overflows - 排序溢出数
AUDIT_EVENTS_TOTAL	BIGINT	audit_events_total - 审计事件总数
ACT_REJECTED_TOTAL	BIGINT	act_rejected_total - 被拒绝活动总数
ACT_ABORTED_TOTAL	BIGINT	act_aborted_total - 异常终止活动总数
TOTAL_SORTS	BIGINT	total_sorts - 排序总数
TOTAL_ROUTINE_TIME	BIGINT	total_routine_time - 例程时间总计
TOTAL_COMPILE_PROC_TIME	BIGINT	total_compile_proc_time - 编译处理时间总计
TOTAL_COMPILATIONS	BIGINT	total_compilations - 编译次数总计
TOTAL_COMPILE_TIME	BIGINT	total_compile_time - 编译时间总计
TOTAL_IMPLICIT_COMPILATIONS	BIGINT	total_implicit_compilations - 隐式编译总数
TOTAL_IMPLICIT_COMPILE_TIME	BIGINT	total_implicit_compile_time - 隐式编译时间总计
TOTAL_RUNSTATS_PROC_TIME	BIGINT	total_runstats_proc_time - 运行时统计信息处理时间总计
TOTAL_RUNSTATS	BIGINT	total_runstats - 运行时统计信息总计
TOTAL_RUNSTATS_TIME	BIGINT	total_runstats_time - 运行时统计信息时间总计
TOTAL_REORG_PROC_TIME	BIGINT	total_reorg_proc_time - 重组处理时间总计
TOTAL_REORGS	BIGINT	total_reorgs - 重组操作总数
TOTAL_REORG_TIME	BIGINT	total_reorg_time - 重组时间总计
TOTAL_LOAD_PROC_TIME	BIGINT	total_load_proc_time - 装入处理时间总计
TOTAL_LOADS	BIGINT	total_loads - 装入操作总数
TOTAL_LOAD_TIME	BIGINT	total_load_time - 装入时间总计
TOTAL_SECTION_PROC_TIME	BIGINT	total_section_proc_time - 部分处理时间总计
TOTAL_APP_SECTION_EXECUTIONS	BIGINT	total_app_section_executions - 应用程序执行部分执行的总次数
TOTAL_SECTION_TIME	BIGINT	total_section_time - 部分时间总计
TOTAL_COMMIT_PROC_TIME	BIGINT	total_commit_proc_time - 落实处理时间总计
TOTAL_APP_COMMITS	BIGINT	total_app_commits - 应用程序落实次数总计
TOTAL_COMMIT_TIME	BIGINT	total_commit_time - 落实时间总计
TOTAL_ROLLBACK_PROC_TIME	BIGINT	total_rollback_proc_time - 回滚处理时间总计
TOTAL_APP_ROLLBACKS	BIGINT	total_app_rollbacks - 应用程序回滚次数总计
TOTAL_ROLLBACK_TIME	BIGINT	total_rollback_time - 回滚时间总计
TOTAL_ROUTINE_USER_CODE _TIME	BIGINT	total_routine_user_code_time - 例程用户代码时间总计

表 31. 对工作单元事件监视器返回的信息: 表名: UOW_METRICS_evmon-name (续)

列名	数据类型	描述
THRESH_VIOLATIONS	BIGINT	thresh_violations - 阈值违例次数
NUM_LW_THRESH_EXCEEDED	BIGINT	num_lw_thresh_exceeded - 超过锁定等待阈值的次数
TOTAL_ROUTINE_INVOCATIONS	BIGINT	total_routine_invocations - 例程调用总计
INT_COMMITS	BIGINT	int_commits - 内部落实数
INT_ROLLBACKS	BIGINT	int_rollbacks - 内部回滚
CAT_CACHE_INSERTS	BIGINT	cat_cache_inserts - 目录高速缓存插入数
CAT_CACHE_LOOKUPS	BIGINT	cat_cache_lookups - 目录高速缓存查询数
PKG_CACHE_INSERTS	BIGINT	pkg_cache_inserts - 程序包高速缓存插入数
PKG_CACHE_LOOKUPS	BIGINT	pkg_cache_lookups - 程序包高速缓存查询数
ACT_RQSTS_TOTAL	BIGINT	act_rqsts_total - 活动请求总数
TOTAL_ACT_WAIT_TIME	BIGINT	total_act_wait_time - 活动等待时间总计
TOTAL_ACT_TIME	BIGINT	total_act_time - 活动时间总计
LOCK_WAIT_TIME_GLOBAL	BIGINT	lock_wait_time_global - 锁定等待时间全局
LOCK_WAITS_GLOBAL	BIGINT	lock_waits_global - 锁定等待全局
RECLAIM_WAIT_TIME	BIGINT	reclaim_wait_time - 回收等待时间
SPACEMAPPAGE_RECLAIM_WAIT_TIME	BIGINT	spacemappage_reclaim_wait_time - 空间映射页回收等待时间
LOCK_TIMEOUTS_GLOBAL	BIGINT	lock_timeouts_global - 锁定超时全局
LOCK_ESCALS_MAXLOCKS	BIGINT	lock_escals_maxlocks - maxlocks 锁定升级数
LOCK_ESCALS_LOCKLIST	BIGINT	lock_escals_locklist - locklist 锁定升级数
LOCK_ESCALS_GLOBAL	BIGINT	lock_escals_global - 全局锁定升级数
CF_WAIT_TIME	BIGINT	cf_wait_time - 集群高速缓存工具等待时间
CF_WAITS	BIGINT	cf_waits - 集群高速缓存工具 DB2 pureScale 服务器等待数
POOL_DATA_GBP_L_READS	BIGINT	pool_data_gbp_l_reads - 组缓冲池数据逻辑读取数
POOL_DATA_GBP_P_READS	BIGINT	pool_data_gbp_p_reads - 组缓冲池数据物理读取数
POOL_DATA_LBP_PAGES_FOUND	BIGINT	pool_data_lbp_pages_found - 本地缓冲池发现的数据页数
POOL_DATA_GBP_INVALID_PAGES	BIGINT	pool_data_gbp_invalid_pages - 组缓冲池无效数据页数
POOL_INDEX_GBP_L_READS	BIGINT	pool_index_gbp_l_reads - 组缓冲池索引逻辑读取数
POOL_INDEX_GBP_P_READS	BIGINT	pool_index_gbp_p_reads - 组缓冲池索引物理读取数
POOL_INDEX_LBP_PAGES_FOUND	BIGINT	pool_index_lbp_pages_found - 发现的本地缓冲池索引页数
POOL_INDEX_GBP_INVALID_PAGES	BIGINT	pool_index_gbp_invalid_pages - 组缓冲池无效索引页数
POOL_XDA_GBP_L_READS	BIGINT	pool_xda_gbp_l_reads - 组缓冲池 XDA 数据逻辑读取请求数
POOL_XDA_GBP_P_READS	BIGINT	pool_xda_gbp_p_reads - 组缓冲池 XDA 数据物理读取请求数
POOL_XDA_LBP_PAGES_FOUND	BIGINT	pool_xda_lbp_pages_found - 发现的本地缓冲池 XDA 数据页数

表 31. 对工作单元事件监视器返回的信息：表名：UOW_METRICS_evmon-name（续）

列名	数据类型	描述
POOL_XDA_GBP_INVALID_PAGES	BIGINT	pool_xda_gbp_invalid_pages - 组缓冲池无效 XDA 数据页数
EVMON_WAIT_TIME	BIGINT	evmon_wait_time - 事件监视器等待时间
EVMON_WAITS_TOTAL	BIGINT	evmon_waits_total - 事件监视器总等待次数
TOTAL_EXTENDED_LATCH_WAIT _TIME	BIGINT	total_extended_latch_wait_time - 扩展锁存器等待时间总计
TOTAL_EXTENDED_LATCH_WAITS	BIGINT	total_extended_latch_waits - 扩展锁存器等待总计
TOTAL_STATS_FABRICATIONS	BIGINT	total_stats_fabrications - 统计信息生成总计
TOTAL_STATS_FABRICATION_TIME	BIGINT	total_stats_fabrication_time - 统计信息生成时间总计
TOTAL_SYNC_RUNSTATS_PROC _TIME	BIGINT	total_sync_runstats_proc_time - 同步 RUNSTATS 处理时间总计
TOTAL_SYNC_RUNSTATS	BIGINT	total_sync_runstats - 同步 RUNSTATS 活动总数
TOTAL_SYNC_RUNSTATS_TIME	BIGINT	total_sync_runstats_time - 同步 RUNSTATS 时间总计
TOTAL_DISP_RUN_QUEUE_TIME	BIGINT	total_disp_run_queue_time - 分派器运行队列时间总计
POOL_QUEUED_ASYNC_DATA _REQS	BIGINT	pool_queued_async_data_reqs - 数据预取请求数
POOL_QUEUED_ASYNC_INDEX _REQS	BIGINT	pool_queued_async_index_reqs - 索引预取请求数
POOL_QUEUED_ASYNC_XDA_REQS	BIGINT	pool_queued_async_xda_reqs - XDA 预取请求数
POOL_QUEUED_ASYNC_OTHER _REQS	BIGINT	pool_queued_async_other_reqs - 非预取请求数
POOL_QUEUED_ASYNC_DATA _PAGES	BIGINT	pool_queued_async_data_pages - 预取请求的数据页数
POOL_QUEUED_ASYNC_INDEX _PAGES	BIGINT	pool_queued_async_index_pages - 预取请求的索引页数
POOL_QUEUED_ASYNC_XDA _PAGES	BIGINT	pool_queued_async_xda_pages - 预取请求的 XDA 页数
POOL_FAILED_ASYNC_DATA_REQS	BIGINT	pool_failed_async_data_reqs - 失败数据预取请求数
POOL_FAILED_ASYNC_INDEX_REQS	BIGINT	pool_failed_async_index_reqs - 失败的索引预取请求数
POOL_FAILED_ASYNC_XDA_REQS	BIGINT	pool_failed_async_xda_reqs - 失败的 XDA 预取请求数
POOL_FAILED_ASYNC_OTHER _REQS	BIGINT	pool_failed_async_other_reqs - 失败的非预取请求数
APP_ACT_COMPLETED_TOTAL	BIGINT	app_act_completed_total - 成功的外部协调程序活动总数
APP_ACT_ABORTED_TOTAL	BIGINT	app_act_aborted_total - 失败的外部协调程序活动总数
APP_ACT_REJECTED_TOTAL	BIGINT	app_act_rejected_total - 拒绝的外部协调程序活动总数
TOTAL_PEDS	BIGINT	total_peds - 部分提前相异总数
DISABLED_PEDS	BIGINT	第 746 页的『disabled_peds -“已禁用部分提前相异数”监视元素』
POST_THRESHOLD_PEDS	BIGINT	post_threshold_peds - 部分提前相异数阈值
TOTAL_PEAS	BIGINT	total_peas - 部分提前聚集总数

表 31. 对工作单元事件监视器返回的信息: 表名: UOW_METRICS_evmon-name (续)

列名	数据类型	描述
POST_THRESHOLD_PEAS	BIGINT	post_threshold_peas - 部分提前聚集阈值
TQ_SORT_HEAP_REQUESTS	BIGINT	tq_sort_heap_requests - 表队列排序堆请求数
TQ_SORT_HEAP_REJECTIONS	BIGINT	tq_sort_heap_rejections - 表队列排序堆拒绝数
TOTAL_CONNECT_REQUESTS	BIGINT	total_connect_requests - 连接或交换机用户请求数
TOTAL_CONNECT_REQUEST_TIME	BIGINT	total_connect_request_time - 连接或交换机用户请求时间总计
TOTAL_CONNECT _AUTHENTICATIONS	BIGINT	total_connect_authentications - 执行的连接或交换机用户认证数
PREFETCH_WAIT_TIME	BIGINT	prefetch_wait_time - 等待预取的时间
PREFETCH_WAITS	BIGINT	prefetch_waits - 预取程序等待计数
POOL_DATA_GBP_INDEP_PAGES _FOUND_IN_LBP	BIGINT	第 975 页的『pool_data_gbp_indep_pages_found _in_lbp -"本地缓冲池中发现的独立于组缓冲池的数据页数"监视元素』
POOL_INDEX_GBP_INDEP_PAGES _FOUND_IN_LBP	BIGINT	第 1005 页的『pool_index_gbp_indep_pages _found_in_lbp -"本地缓冲池中发现的独立于组缓冲池的索引页数"监视元素』
POOL_XDA_GBP_INDEP_PAGES _FOUND_IN_LBP	BIGINT	第 1059 页的『pool_xda_gbp_indep_pages _found_in_lbp -"本地缓冲池中发现的独立于组缓冲池的 XDA 页数"监视元素』

表 32. 对锁定事件监视器返回的信息: 缺省表名: UOW_PACKAGE_LIST_evmon-name

列名	数据类型	描述
PARTITION_KEY	INTEGER	第 951 页的『partition_key -"分区键"监视元素』
APPLICATION_ID	VARCHAR(128)	第 616 页的『appl_id -"应用程序标识"监视元素』
INVOCATION_ID	INTEGER	invocation_id - 调用标识
MEMBER	SMALLINT	member - 数据库成员
NESTING_LEVEL	INTEGER	nesting_level - 嵌套级别
PACKAGE_ELAPSED_TIME	BIGINT	package_elapsed_time - 程序包耗用时间
PACKAGE_ID	BIGINT	package_id - 程序包标识
ROUTINE_ID	INTEGER	routine_id - 例程标识
UOW_ID	INTEGER	uow_id - 工作单元标识

表 33. 对锁定事件监视器返回的信息: 缺省表名: UOW_EXECUTABLE_LIST_evmon-name

列名	数据类型	描述
PARTITION_KEY	INTEGER	第 951 页的『partition_key -"分区键"监视元素』
APPLICATION_ID	VARCHAR(128)	第 616 页的『appl_id -"应用程序标识"监视元素』
EXECUTABLE_ID	VARCHAR(32)	executable_id - 可执行标识
LOCK_WAIT_TIME	BIGINT	lock_wait_time - 等待锁定时间
LOCK_WAITS	BIGINT	lock_waits - 等待锁定次数

表 33. 对锁定事件监视器返回的信息：缺省表名：UOW_EXECUTABLE_LIST_evmon-name（续）

列名	数据类型	描述
MEMBER	SMALLINT	member - 数据库成员
NUM_EXECUTIONS	BIGINT	num_executions - 语句执行次数
POST_SHRTHRESHOLD_SORTS	BIGINT	post_shrthreshold_sorts - 共享阈值后排序数
POST_THRESHOLD_SORTS	BIGINT	post_threshold_sorts - 超过阈值后的排序数
ROWS_READ	BIGINT	rows_read - 读取行数
SORT_OVERFLOWS	BIGINT	sort_overflows - 排序溢出数
TOTAL_ACT_TIME	BIGINT	total_act_time - 活动时间总计
TOTAL_ACT_WAIT_TIME	BIGINT	total_act_wait_time - 活动等待时间总计
TOTAL_CPU_TIME	BIGINT	total_cpu_time - CPU 时间总计
TOTAL_SORTS	BIGINT	total_sorts - 排序总数
UOW_ID	INTEGER	uow_id - 工作单元标识

表 34. 对工作单元事件监视器返回的信息：缺省表名：CONTROL_evmon-name

列名	数据类型	描述
PARTITION_KEY	INTEGER	第 951 页的『partition_key -“分区键”监视元素』
EVENT_MONITOR_NAME	VARCHAR(128)	event_monitor_name - 事件监视器名称
MESSAGE	VARCHAR(128)	message - 控制表消息
MESSAGE_TIME	TIMESTAMP	message_time - 时间戳记控制表消息
PARTITION_NUMBER	SMALLINT	partition_number - 分区号

EVMON_FORMAT_UE_TO_TABLES 为工作单元事件监视器写至关系表的信息：

EVMON_FORMAT_UE_TO_TABLES 表函数为工作单元事件监视器写入的信息。`sqllib/misc/DB2EvmonUOW.xsd` 文件中也记录了此信息。

表 35. 为工作单元事件监视器返回的信息：表名：UOW_EVENT

列名	数据类型	描述
EVENT_ID	BIGINT NOT NULL	event_id -“事件标识”监视元素
TYPE	VARCHAR(128) NOT NULL	
EVENT_TIMESTAMP	TIMESTAMP NOT NULL	event_timestamp -“事件时间戳记”监视元素
MEMBER	SMALLINT	member - 数据库成员
COORD_MEMBER	SMALLINT	coord_member - 协调程序成员
COMPLETION_STATUS	VARCHAR(128)	completion_status - 完成状态
START_TIME	TIMESTAMP	start_time - 事件启动时间
STOP_TIME	TIMESTAMP	stop_time - 事件停止时间

表 35. 为工作单元事件监视器返回的信息：表名：UOW_EVENT (续)

列名	数据类型	描述
WORKLOAD_NAME	VARCHAR(128)	workload_name - 工作负载名称
WORKLOAD_ID	INTEGER	workload_id - 工作负载标识
SERVICE_SUPERCLASS_NAME	VARCHAR(128)	service_superclass_name - 服务超类名
SERVICE_SUBCLASS_NAME	VARCHAR(128)	service_subclass_name - 服务子类名
SERVICE_CLASS_ID	INTEGER	service_class_id - 服务类标识
UOW_ID	INTEGER	uow_id - 工作单元标识
WORKLOAD_OCCURRENCE_ID	INTEGER	workload_occurrence_id - 工作负载项标识
CONNECTION_TIME	TIMESTAMP	第 683 页的『connection_start_time -“连接开始时间”监视元素』
MEMBER_ACTIVATION_TIME	TIMESTAMP	第 717 页的『db_conn_time -“数据库激活时间戳记”监视元素』
APPLICATION_ID	VARCHAR(128)	第 616 页的『appl_id -“应用程序标识”监视元素』
APPLICATION_HANDLE	BIGINT	application_handle - 应用程序句柄
APPLICATION_NAME	VARCHAR(128)	第 620 页的『appl_name -“应用程序名称”监视元素』
SYSTEM_AUTHID	VARCHAR(128)	第 1188 页的『system_auth_id -“系统授权标识”监视元素』
SESSION_AUTHID	VARCHAR(128)	第 1137 页的『session_auth_id -“会话授权标识”监视元素』
CLIENT_PLATFORM	VARCHAR(12)	client_platform - 客户机操作平台
CLIENT_PID	BIGINT	client_pid - 客户机进程标识
CLIENT_PRODUCT_ID	VARCHAR(128)	第 663 页的『client_prdid -“客户机产品和版本标识”监视元素』
CLIENT_PROTOCOL	VARCHAR(10)	client_protocol - 客户机通信协议
CLIENT_HOSTNAME	VARCHAR(255)	client_hostname - 客户机主机名
CLIENT_PORT_NUMBER	INTEGER	client_port_number - 客户机端口号
CLIENT_WRKSTNNAME	VARCHAR(255)	client_wrkstnname - 客户机工作站名称
CLIENT_ACCTNG	VARCHAR(255)	client_acctng - 客户机记帐字符串
CLIENT_USERID	VARCHAR(255)	client_userid - 客户机用户标识
CLIENT_APPLNAME	VARCHAR(255)	client_applname - 客户机应用程序名称
LOCAL_TRANSACTION_ID	VARCHAR(16)	第 841 页的『local_transaction_id -“本地事务标识”监视元素』
GLOBAL_TRANSACTION_ID	VARCHAR(40)	第 794 页的『global_transaction_id -“全局事务标识”监视元素』
UOW_LOG_SPACE_USED	BIGINT	uow_log_space_used - 使用的工作单元日志空间
PACKAGE_LIST_SIZE	INTEGER	第 941 页的『package_list_size -“程序包列表大小”监视元素』
PACKAGE_LIST_EXCEEDED	CHAR(3)	package_list_exceeded - 超过了程序包列表的容量

表 35. 为工作单元事件监视器返回的信息：表名：UOW_EVENT（续）

列名	数据类型	描述
EXECUTABLE_LIST_SIZE	BIGINT	第 762 页的『executable_list_size -“可执行列表大小”监视元素』
EXECUTABLE_LIST_TRUNCATED	CHAR(3)	第 763 页的『executable_list_truncated -“截断可执行列表”监视元素』
METRICS	BLOB(1M)	包含与度量值相关的监视元素的 XML 文档。此文档中的度量值与本主题后面出现的 UOW_METRICS 表描述的度量值相同。有关更多信息，请参阅第 12 页的『在 XML 文档中返回监视数据的接口』。
INTRA_PARALLEL_STATE	VARCHAR(3)	intra_parallel_state - 分区内并行性的当前状态
MON_INTERVAL_ID	BIGINT	mon_interval_id - 监视时间间隔标识

表 36. 为工作单元事件监视器返回的信息：表名：UOW_PACKAGE_LIST

列名	数据类型	描述
MEMBER	SMALLINT	member - 数据库成员
UOW_ID	INTEGER	uow_id - 工作单元标识
APPLICATION_ID	VARCHAR(128)	第 616 页的『appl_id -“应用程序标识”监视元素』
PACKAGE_ID	BIGINT	package_id - 程序包标识
NESTING_LEVEL	INTEGER	nesting_level - 嵌套级别
ROUTINE_ID	BIGINT	routine_id - 例程标识
INVOCATION_ID	INTEGER	invocation_id - 调用标识
PACKAGE_ELAPSED_TIME	BIGINT	package_elapsed_time - 程序包耗用时间

表 37. 为工作单元事件监视器返回的信息：表名：UOW_EXECUTABLE_LIST

列名	数据类型	描述
MEMBER	SMALLINT	member - 数据库成员
UOW_ID	INTEGER	uow_id - 工作单元标识
APPLICATION_ID	VARCHAR(128)	第 616 页的『appl_id -“应用程序标识”监视元素』
EXECUTABLE_ID	VARCHAR(32) FOR BIT DATA	executable_id - 可执行标识
NUM_EXECUTIONS	BIGINT	num_executions - 语句执行次数
ROWS_READ	BIGINT	rows_read - 读取行数
TOTAL_CPU_TIME	BIGINT	total_cpu_time - CPU 时间总计
TOTAL_ACT_TIME	BIGINT	total_act_time - 活动时间总计
TOTAL_ACT_WAIT_TIME	BIGINT	total_act_wait_time - 活动等待时间总计
LOCK_WAIT_TIME	BIGINT	lock_wait_time - 等待锁定时间

表 37. 为工作单元事件监视器返回的信息：表名：UOW_EXECUTABLE_LIST（续）

列名	数据类型	描述
LOCK_WAITS	BIGINT	lock_waits - 等待锁定次数
TOTAL_SORTS	BIGINT	total_sorts - 排序总数
POST_THRESHOLD_SORTS	BIGINT	post_threshold_sorts - 超过阈值后的排序数
POST_SHRTHRESHOLD_SORTS	BIGINT	post_shrthreshold_sorts - 共享阈值后排序数
SORT_OVERFLOWS	BIGINT	sort_overflows - 排序溢出数

表 38. 为工作单元事件监视器返回的信息：表名：UOW_METRICS。此表中的度量值与 UOW_EVENT 表中 METRICS 监视元素返回的度量值相同

列名	数据类型	描述
MEMBER	SMALLINT	member - 数据库成员
UOW_ID	INTEGER	uow_id - 工作单元标识
APPLICATION_ID	VARCHAR(128)	第 616 页的『appl_id -“应用程序标识”监视元素』
ACT_ABORTED_TOTAL	BIGINT	act_aborted_total - 异常终止活动总数
ACT_COMPLETED_TOTAL	BIGINT	act_completed_total - 完成活动总数
ACT_REJECTED_TOTAL	BIGINT	act_rejected_total - 被拒绝活动总数
AGENT_WAIT_TIME	BIGINT	agent_wait_time - 代理程序等待时间
AGENT_WAITS_TOTAL	BIGINT	agent_waits_total - 等待代理程序总次数
POOL_DATA_L_READS	BIGINT	pool_data_l_reads - 缓冲池数据逻辑读取数
POOL_INDEX_L_READS	BIGINT	pool_index_l_reads - 缓冲池索引逻辑读取数
POOL_TEMP_DATA_L_READS	BIGINT	pool_temp_data_l_reads - 缓冲池临时数据逻辑读取数
POOL_TEMP_INDEX_L_READS	BIGINT	pool_temp_index_l_reads - 缓冲池临时索引逻辑读取数
POOL_TEMP_XDA_L_READS	BIGINT	pool_temp_xda_l_reads - 缓冲池临时 XDA 数据逻辑读取数
POOL_XDA_L_READS	BIGINT	pool_xda_l_reads - 缓冲池 XDA 数据逻辑读取数
POOL_DATA_P_READS	BIGINT	pool_data_p_reads - 缓冲池数据物理读取数
POOL_INDEX_P_READS	BIGINT	pool_index_p_reads - 缓冲池索引物理读取数
POOL_TEMP_DATA_P_READS	BIGINT	pool_temp_data_p_reads - 缓冲池临时数据物理读取数
POOL_TEMP_INDEX_P_READS	BIGINT	pool_temp_index_p_reads - 缓冲池临时索引物理读取数
POOL_TEMP_XDA_P_READS	BIGINT	pool_temp_xda_p_reads - 缓冲池临时 XDA 数据物理读取数
POOL_XDA_P_READS	BIGINT	pool_xda_p_reads - 缓冲池 XDA 数据物理读取数
POOL_DATA_WRITES	BIGINT	pool_data_writes - 缓冲池数据写次数
POOL_INDEX_WRITES	BIGINT	pool_index_writes - 缓冲池索引写次数
POOL_XDA_WRITES	BIGINT	pool_xda_writes - 缓冲池 XDA 数据写次数

表 38. 为工作单元事件监视器返回的信息：表名：UOW_METRICS。此表中的度量值与 UOW_EVENT 表中 METRICS 监视元素返回的度量值相同 (续)

列名	数据类型	描述
POOL_READ_TIME	BIGINT	pool_read_time - 缓冲池物理读时间总计
POOL_WRITE_TIME	BIGINT	pool_write_time - 缓冲池物理写时间总计
CLIENT_IDLE_WAIT_TIME	BIGINT	client_idle_wait_time - 客户机空闲等待时间
DEADLOCKS	BIGINT	deadlocks - 检测到的死锁数
DIRECT_READS	BIGINT	direct_reads - 直接读数据库数目
DIRECT_READ_TIME	BIGINT	direct_read_time - 直接读时间
DIRECT_WRITES	BIGINT	direct_writes - 直接写数据库数目
DIRECT_WRITE_TIME	BIGINT	direct_write_time - 直接写时间
DIRECT_READ_REQS	BIGINT	direct_read_reqs - 直接读请求数
DIRECT_WRITE_REQS	BIGINT	direct_write_reqs - 直接写请求数
FCM_RECV_VOLUME	BIGINT	fcm_recv_volume - FCM 接收量
FCM_RECVS_TOTAL	BIGINT	fcm_recvs_total - FCM 接收总计
FCM_SEND_VOLUME	BIGINT	fcm_send_volume - FCM 发送量
FCM_SENDS_TOTAL	BIGINT	fcm_sends_total - FCM 发送总计
FCM_RECV_WAIT_TIME	BIGINT	fcm_recv_wait_time - FCM 接收等待时间
FCM_SEND_WAIT_TIME	BIGINT	fcm_send_wait_time - FCM 发送等待时间
IPC_RECV_VOLUME	BIGINT	ipc_recv_volume - 进程间通信接收量
IPC_RECV_WAIT_TIME	BIGINT	ipc_recv_wait_time - 进程间通信接收等待时间
IPC_RECVS_TOTAL	BIGINT	ipc_recvs_total - 进程间通信接收总次数
IPC_SEND_VOLUME	BIGINT	ipc_send_volume - 进程间通信发送量
IPC_SEND_WAIT_TIME	BIGINT	ipc_send_wait_time - 进程间通信发送等待时间
IPC_SENDS_TOTAL	BIGINT	ipc_sends_total - 进程间通信发送总次数
LOCK_ESCALS	BIGINT	lock_escals - 锁定升级次数
LOCK_TIMEOUTS	BIGINT	lock_timeouts - 锁定超时次数
LOCK_WAIT_TIME	BIGINT	lock_wait_time - 等待锁定时间
LOCK_WAITS	BIGINT	lock_waits - 等待锁定次数
LOG_BUFFER_WAIT_TIME	BIGINT	log_buffer_wait_time - 日志缓冲区等待时间
NUM_LOG_BUFFER_FULL	BIGINT	num_log_buffer_full - 日志缓冲区变满次数
LOG_DISK_WAIT_TIME	BIGINT	log_disk_wait_time - 日志磁盘等待时间
LOG_DISK_WAITS_TOTAL	BIGINT	log_disk_waits_total - 日志磁盘等待总次数
RQSTS_COMPLETED_TOTAL	BIGINT	rqsts_completed_total - 完成请求总数
ROWS_MODIFIED	BIGINT	rows_modified - 修改的行数
ROWS_READ	BIGINT	rows_read - 读取行数
ROWS_RETURNED	BIGINT	rows_returned - 返回的行数
TCPIP_RECV_VOLUME	BIGINT	tcpip_recv_volume - TCP/IP 接收量
TCPIP_SEND_VOLUME	BIGINT	tcpip_send_volume - TCP/IP 发送量
TCPIP_RECV_WAIT_TIME	BIGINT	tcpip_recv_wait_time - TCP/IP 接收等待时间
TCPIP_RECVS_TOTAL	BIGINT	tcpip_recvs_total - TCP/IP 接收总次数
TCPIP_SEND_WAIT_TIME	BIGINT	tcpip_send_wait_time - TCP/IP 发送等待时间

表 38. 为工作单元事件监视器返回的信息：表名：UOW_METRICS。此表中的度量值与 UOW_EVENT 表中 METRICS 监视元素返回的度量值相同 (续)

列名	数据类型	描述
TCPIP_SENDS_TOTAL	BIGINT	tcpip_sends_total - TCP/IP 发送总次数
TOTAL_APP_RQST_TIME	BIGINT	total_app_rqst_time - 应用程序请求时间总计
TOTAL_RQST_TIME	BIGINT	total_rqst_time - 请求时间总计
WLM_QUEUE_TIME_TOTAL	BIGINT	wlm_queue_time_total - 工作负载管理器队列时间总计
WLM_QUEUE_ASSIGNMENTS _TOTAL	BIGINT	wlm_queue_assignments_total - 工作负载管理器队列分配总计
TOTAL_CPU_TIME	BIGINT	total_cpu_time - CPU 时间总计
TOTAL_WAIT_TIME	BIGINT	total_wait_time - 等待时间总计
APP_RQSTS_COMPLETED_TOTAL	BIGINT	app_rqsts_completed_total - 完成应用程序请求总数
TOTAL_SECTION_SORT_TIME	BIGINT	total_section_sort_time - 节排序时间总计
TOTAL_SECTION_SORT_PROC _TIME	BIGINT	total_section_sort_proc_time - 节排序处理时间总计
TOTAL_SECTION_SORTS	BIGINT	total_section_sorts - 节排序总次数
TOTAL_SORTS	BIGINT	total_sorts - 排序总数
POST_THRESHOLD_SORTS	BIGINT	post_threshold_sorts - 超过阈值后的排序数
POST_SHRTHRESHOLD_SORTS	BIGINT	post_shrthreshold_sorts - 共享阈值后排序数
SORT_OVERFLOWS	BIGINT	sort_overflows - 排序溢出数
TOTAL_COMPILE_TIME	BIGINT	total_compile_time - 编译时间总计
TOTAL_COMPILE_PROC_TIME	BIGINT	total_compile_proc_time - 编译处理时间总计
TOTAL_COMPILATIONS	BIGINT	total_compilations - 编译次数总计
TOTAL_IMPLICIT_COMPILE_TIME	BIGINT	total_implicit_compile_time - 隐式编译时间总计
TOTAL_IMPLICIT_COMPILE_PROC _TIME	BIGINT	total_implicit_compile_proc_time - 隐式编译处理时间总计
TOTAL_IMPLICIT_COMPILATIONS	BIGINT	total_implicit_compilations - 隐式编译总数
TOTAL_SECTION_TIME	BIGINT	total_section_time - 部分时间总计
TOTAL_SECTION_PROC_TIME	BIGINT	total_section_proc_time - 部分处理时间总计
TOTAL_APP_SECTION _EXECUTIONS	BIGINT	total_app_section_executions - 应用程序执行部分执行的总次数
TOTAL_ACT_TIME	BIGINT	total_act_time - 活动时间总计
TOTAL_ACT_WAIT_TIME	BIGINT	total_act_wait_time - 活动等待时间总计
ACT_RQSTS_TOTAL	BIGINT	act_rqsts_total - 活动请求总数
TOTAL_ROUTINE_TIME	BIGINT	total_routine_time - 例程时间总计
TOTAL_ROUTINE_INVOCATIONS	BIGINT	total_routine_invocations - 例程调用总计
TOTAL_COMMIT_TIME	BIGINT	total_commit_time - 落实时间总计
TOTAL_COMMIT_PROC_TIME	BIGINT	total_commit_proc_time - 落实处理时间总计
TOTAL_APP_COMMITS	BIGINT	total_app_commits - 应用程序落实次数总计
INT_COMMITS	BIGINT	int_commits - 内部落实数
TOTAL_ROLLBACK_TIME	BIGINT	total_rollback_time - 回滚时间总计

表 38. 为工作单元事件监视器返回的信息：表名：UOW_METRICS。此表中的度量值与 UOW_EVENT 表中 METRICS 监视元素返回的度量值相同 (续)

列名	数据类型	描述
TOTAL_ROLLBACK_PROC_TIME	BIGINT	total_rollback_proc_time - 回滚处理时间总计
TOTAL_APP_ROLLBACKS	BIGINT	total_app_rollbacks - 应用程序回滚次数总计
INT_ROLLBACKS	BIGINT	int_rollbacks - 内部回滚
TOTAL_RUNSTATS_TIME	BIGINT	total_runstats_time - 运行时统计信息时间总计
TOTAL_RUNSTATS_PROC_TIME	BIGINT	total_runstats_proc_time - 运行时统计信息处理时间总计
TOTAL_RUNSTATS	BIGINT	total_runstats - 运行时统计信息总计
TOTAL_REORG_TIME	BIGINT	total_reorg_time - 重组时间总计
TOTAL_REORG_PROC_TIME	BIGINT	total_reorg_proc_time - 重组处理时间总计
TOTAL_REORGS	BIGINT	total_reorgs - 重组操作总数
TOTAL_LOAD_TIME	BIGINT	total_load_time - 装入时间总计
TOTAL_LOAD_PROC_TIME	BIGINT	total_load_proc_time - 装入处理时间总计
TOTAL_LOADS	BIGINT	total_loads - 装入操作总数
CAT_CACHE_INSERTS	BIGINT	cat_cache_inserts - 目录高速缓存插入数
CAT_CACHE_LOOKUPS	BIGINT	cat_cache_lookups - 目录高速缓存查询数
PKG_CACHE_INSERTS	BIGINT	pkg_cache_inserts - 程序包高速缓存插入数
PKG_CACHE_LOOKUPS	BIGINT	pkg_cache_lookups - 程序包高速缓存查询数
THRESH_VIOLATIONS	BIGINT	thresh_violations - 阈值违例次数
NUM_LW_THRESH_EXCEEDED	BIGINT	num_lw_thresh_exceeded - 超过锁定等待阈值的次数
FCM_TQ_RECV_WAIT_TIME	BIGINT	fcm_tq_recv_wait_time - FCM 表队列接收等待时间
FCM_MESSAGE_RECV_WAIT_TIME	BIGINT	fcm_message_recv_wait_time - 接收 FCM 消息等待时间
FCM_TQ_SEND_WAIT_TIME	BIGINT	fcm_tq_send_wait_time - FCM 表队列发送等待时间
FCM_MESSAGE_SEND_WAIT_TIME	BIGINT	fcm_message_send_wait_time - 发送 FCM 消息等待时间
AUDIT_FILE_WRITE_WAIT_TIME	BIGINT	audit_file_write_wait_time - 审计文件写等待时间
AUDIT_FILE_WRITES_TOTAL	BIGINT	audit_file_writes_total - 写审计文件总次数
AUDIT_SUBSYSTEM_WAIT_TIME	BIGINT	audit_subsystem_wait_time - 审计子系统等待时间
AUDIT_SUBSYSTEM_WAITS_TOTAL	BIGINT	audit_subsystem_waits_total - 审计子系统等待总次数
DIAGLOG_WRITE_WAIT_TIME	BIGINT	diaglog_write_wait_time - 诊断日志文件写等待时间
DIAGLOG_WRITES_TOTAL	BIGINT	diaglog_writes_total - 写诊断日志文件总次数
FCM_MESSAGE_SENDS_TOTAL	BIGINT	fcm_message_sends_total - 发送 FCM 消息总数
FCM_MESSAGE_RECVS_TOTAL	BIGINT	fcm_message_recvs_total - 接收 FCM 消息总数
FCM_MESSAGE_SEND_VOLUME	BIGINT	fcm_message_send_volume - 发送 FCM 消息量
FCM_MESSAGE_RECV_VOLUME	BIGINT	fcm_message_recv_volume - 接收 FCM 消息量

表 38. 为工作单元事件监视器返回的信息：表名：UOW_METRICS。此表中的度量值与 UOW_EVENT 表中 METRICS 监视元素返回的度量值相同 (续)

列名	数据类型	描述
FCM_TQ_SENDS_TOTAL	BIGINT	fcm_tq_sends_total - FCM 表队列发送总次数
FCM_TQ_RECVS_TOTAL	BIGINT	fcm_tq_recvs_total - FCM 表队列接收总量
FCM_TQ_SEND_VOLUME	BIGINT	fcm_tq_send_volume - FCM 表队列发送量
FCM_TQ_RECV_VOLUME	BIGINT	fcm_tq_recv_volume - FCM 表队列接收量
TQ_TOT_SEND_SPILLS	BIGINT	tq_tot_send_spills - 溢出表队列缓冲区总数
AUDIT_EVENTS_TOTAL	BIGINT	audit_events_total - 审计事件总数
TOTAL_ROUTINE_USER_CODE _PROC_TIME	BIGINT	total_routine_user_code_proc_time - 例程用户代码处理时间总计
TOTAL_ROUTINE_USER_CODE _TIME	BIGINT	total_routine_user_code_time - 例程用户代码时间总计
LOCK_WAITS_GLOBAL	BIGINT	lock_waits_global - 锁定等待全局
LOCK_WAIT_TIME_GLOBAL	BIGINT	lock_wait_time_global - 锁定等待时间全局
LOCK_TIMEOUTS_GLOBAL	BIGINT	lock_timeouts_global - 锁定超时全局
LOCK_ESCALS_MAXLOCKS	BIGINT	lock_escals_maxlocks - maxlocks 锁定升级数
LOCK_ESCALS_LOCKLIST	BIGINT	lock_escals_locklist - locklist 锁定升级数
LOCK_ESCALS_GLOBAL	BIGINT	lock_escals_global - 全局锁定升级数
RECLAIM_WAIT_TIME	BIGINT	reclaim_wait_time - 回收等待时间
SPACEMAPPAGE_RECLAIM_WAIT _TIME	BIGINT	spacemappage_reclaim_wait_time - 空间映射页回收等待时间
CF_WAITS	BIGINT	cf_waits - 集群高速缓存工具 DB2 pureScale 服务器等待数
CF_WAIT_TIME	BIGINT	cf_wait_time - 集群高速缓存工具等待时间
POOL_DATA_GBP_L_READS	BIGINT	pool_data_gbp_l_reads - 组缓冲池数据逻辑读取数
POOL_DATA_GBP_P_READS	BIGINT	pool_data_gbp_p_reads - 组缓冲池数据物理读取数
POOL_DATA_LBP_PAGES_FOUND	BIGINT	pool_data_lbp_pages_found - 本地缓冲池发现的数据页数
POOL_DATA_GBP_INVALID _PAGES	BIGINT	pool_data_gbp_invalid_pages - 组缓冲池无效数据页数
POOL_INDEX_GBP_L_READS	BIGINT	pool_index_gbp_l_reads - 组缓冲池索引逻辑读取数
POOL_INDEX_GBP_P_READS	BIGINT	pool_index_gbp_p_reads - 组缓冲池索引物理读取数
POOL_INDEX_LBP_PAGES_FOUND	BIGINT	pool_index_lbp_pages_found - 发现的本地缓冲池索引页数
POOL_INDEX_GBP_INVALID_PAGES	BIGINT	pool_index_gbp_invalid_pages - 组缓冲池无效索引页数
POOL_XDA_GBP_L_READS	BIGINT	pool_xda_gbp_l_reads - 组缓冲池 XDA 数据逻辑读取请求数
POOL_XDA_GBP_P_READS	BIGINT	pool_xda_gbp_p_reads - 组缓冲池 XDA 数据物理读取请求数

表 38. 为工作单元事件监视器返回的信息: 表名: UOW_METRICS。此表中的度量值与 UOW_EVENT 表中 METRICS 监视元素返回的度量值相同 (续)

列名	数据类型	描述
POOL_XDA_LBP_PAGES_FOUND	BIGINT	pool_xda_lbp_pages_found - 发现的本地缓冲池 XDA 数据页数
POOL_XDA_GBP_INVALID_PAGES	BIGINT	pool_xda_gbp_invalid_pages - 组缓冲池无效 XDA 数据页数
EVMON_WAIT_TIME	BIGINT	evmon_wait_time - 事件监视器等待时间
EVMON_WAITS_TOTAL	BIGINT	evmon_waits_total - 事件监视器总等待次数
TOTAL_EXTENDED_LATCH_WAIT _TIME	BIGINT	total_extended_latch_wait_time - 扩展锁存器等待时间总计
TOTAL_EXTENDED_LATCH_WAITS	BIGINT	total_extended_latch_waits - 扩展锁存器等待总计
TOTAL_STATS_FABRICATION_TIME	BIGINT	total_stats_fabrication_time - 统计信息生成时间总计
TOTAL_STATS_FABRICATION _PROC_TIME	BIGINT	total_stats_fabrication_proc_time - 统计信息生成处理时间总计
TOTAL_STATS_FABRICATIONS	BIGINT	total_stats_fabrications - 统计信息生成总计
TOTAL_SYNC_RUNSTATS_TIME	BIGINT	total_sync_runstats_time - 同步 RUNSTATS 时间总计
TOTAL_SYNC_RUNSTATS_PROC _TIME	BIGINT	total_sync_runstats_proc_time - 同步 RUNSTATS 处理时间总计
TOTAL_SYNC_RUNSTATS	BIGINT	total_sync_runstats - 同步 RUNSTATS 活动总数
TOTAL_DISP_RUN_QUEUE_TIME	BIGINT	total_disp_run_queue_time - 分派器运行队列时间总计
TOTAL_PEDS	BIGINT	total_peds - 部分提前相异总数
DISABLED_PEDS	BIGINT	第 746 页的『disabled_peds -“已禁用部分提前相异数”监视元素』
POST_THRESHOLD_PEDS	BIGINT	post_threshold_peds - 部分提前相异数阈值
TOTAL_PEAS	BIGINT	total_peas - 部分提前聚集总数
POST_THRESHOLD_PEAS	BIGINT	post_threshold_peas - 部分提前聚集阈值
TQ_SORT_HEAP_REQUESTS	BIGINT	tq_sort_heap_requests - 表队列排序堆请求数
TQ_SORT_HEAP_REJECTIONS	BIGINT	tq_sort_heap_rejections - 表队列排序堆拒绝数
POOL_QUEUED_ASYNC_DATA _REQS	BIGINT	pool_queued_async_data_reqs - 数据预取请求数
POOL_QUEUED_ASYNC_INDEX _REQS	BIGINT	pool_queued_async_index_reqs - 索引预取请求数
POOL_QUEUED_ASYNC_XDA _REQS	BIGINT	pool_queued_async_xda_reqs - XDA 预取请求数
POOL_QUEUED_ASYNC_TEMP _DATA_REQS	BIGINT	pool_queued_async_temp_data_reqs - 临时表空间数据预取请求数
POOL_QUEUED_ASYNC_TEMP _INDEX_REQS	BIGINT	pool_queued_async_temp_index_reqs - 临时表空间索引预取请求数
POOL_QUEUED_ASYNC_TEMP _XDA_REQS	BIGINT	pool_queued_async_temp_xda_reqs - 临时表空间 XDA 数据预取请求数

表 38. 为工作单元事件监视器返回的信息：表名：UOW_METRICS。此表中的度量值与 UOW_EVENT 表中 METRICS 监视元素返回的度量值相同 (续)

列名	数据类型	描述
POOL_QUEUED_ASYNC_OTHER _REQS	BIGINT	pool_queued_async_other_reqs - 非预取请求数
POOL_QUEUED_ASYNC_DATA _PAGES	BIGINT	pool_queued_async_data_pages - 预取请求的数据页数
POOL_QUEUED_ASYNC_INDEX _PAGES	BIGINT	pool_queued_async_index_pages - 预取请求的索引页数
POOL_QUEUED_ASYNC_XDA _PAGES	BIGINT	pool_queued_async_xda_pages - 预取请求的 XDA 页数
POOL_QUEUED_ASYNC_TEMP _DATA_PAGES	BIGINT	pool_queued_async_temp_data_pages - 预取请求的临时表空间数据页数
POOL_QUEUED_ASYNC_TEMP _INDEX_PAGES	BIGINT	pool_queued_async_temp_index_pages - 预取请求的临时表空间索引页数
POOL_QUEUED_ASYNC_TEMP _XDA_PAGES	BIGINT	pool_queued_async_temp_xda_pages - 预取请求的临时表空间 XDA 数据页数
POOL_FAILED_ASYNC_DATA_REQS	BIGINT	pool_failed_async_data_reqs - 失败数据预取请求数
POOL_FAILED_ASYNC_INDEX _REQS	BIGINT	pool_failed_async_index_reqs - 失败的索引预取请求数
POOL_FAILED_ASYNC_XDA_REQS	BIGINT	pool_failed_async_xda_reqs - 失败的 XDA 预取请求数
POOL_FAILED_ASYNC_TEMP _DATA_REQS	BIGINT	pool_failed_async_temp_data_reqs - 临时表空间的失败数据预取请求数
POOL_FAILED_ASYNC_TEMP _INDEX_REQS	BIGINT	pool_failed_async_temp_index_reqs - 临时表空间的失败索引预取请求数
POOL_FAILED_ASYNC_TEMP_XDA _REQS	BIGINT	pool_failed_async_temp_xda_reqs - 临时表空间的失败 XDA 预取请求数
POOL_FAILED_ASYNC_OTHER _REQS	BIGINT	pool_failed_async_other_reqs - 失败的非预取请求数
PREFETCH_WAIT_TIME	BIGINT	prefetch_wait_time - 等待预取的时间
PREFETCH_WAITS	BIGINT	prefetch_waits - 预取程序等待计数
APP_ACT_COMPLETED_TOTAL	BIGINT	app_act_completed_total - 成功的外部协调程序活动总数
APP_ACT_ABORTED_TOTAL	BIGINT	app_act_aborted_total - 失败的外部协调程序活动总数
APP_ACT_REJECTED_TOTAL	BIGINT	app_act_rejected_total - 拒绝的外部协调程序活动总数
TOTAL_CONNECT_REQUEST_TIME	BIGINT	total_connect_request_time - 连接或交换机用户请求时间总计
TOTAL_CONNECT_REQUEST _PROC_TIME	BIGINT	total_connect_request_proc_time - 连接或交换机用户请求处理时间总计
TOTAL_CONNECT_REQUESTS	BIGINT	total_connect_requests - 连接或交换机用户请求数
TOTAL_CONNECT _AUTHENTICATION_TIME	BIGINT	total_connect_authentication_time - 连接或交换机用户认证请求时间总计

表 38. 为工作单元事件监视器返回的信息：表名：UOW_METRICS。此表中的度量值与 UOW_EVENT 表中 METRICS 监视元素返回的度量值相同 (续)

列名	数据类型	描述
TOTAL_CONNECT _AUTHENTICATION_PROC _TIME	BIGINT	total_connect_authentication_proc_time - 连接认证处理时间总计
TOTAL_CONNECT _AUTHENTICATIONS	BIGINT	total_connect_authentications - 执行的连接或交换机用户认证数
POOL_DATA_GBP_INDEP_PAGES _FOUND_IN_LBP	BIGINT	第 975 页的『pool_data_gbp_indep_pages_found _in_lbp -“本地缓冲池中发现的独立于组缓冲池的数据页数”监视元素』
POOL_INDEX_GBP_INDEP_PAGES _FOUND_IN_LBP	BIGINT	第 1005 页的『pool_index_gbp_indep_pages _found_in_lbp -“本地缓冲池中发现的独立于组缓冲池的索引页数”监视元素』
POOL_XDA_GBP_INDEP_PAGES _FOUND_IN_LBP	BIGINT	第 1059 页的『pool_xda_gbp_indep_pages _found_in_lbp -“本地缓冲池中发现的独立于组缓冲池的 XDA 页数”监视元素』
COMM_EXIT_WAIT_TIME	BIGINT	comm_exit_wait_time -“通信缓冲区出口等待时间”监视元素
COMM_EXIT_WAITS	BIGINT	comm_exit_waits -“通信缓冲区出口等待数”监视元素

EVMON_FORMAT_UE_TO_XML 为工作单元事件监视器写至 XML 的信息:

EVMON_FORMAT_UE_TO_XML 表函数为工作单元事件监视器写入的信息。`sqllib/misc/DB2EvmonUOW.xsd` 文件中也记录了此信息。

db2_uow_event

用于描述工作单元事件的主模式。

元素内容: （第 176 页的『completion_status』，第 177 页的『start_time』，第 177 页的『stop_time』，第 177 页的『connection_time』，第 177 页的『application_name』，第 177 页的『application_handle』，第 178 页的『application_id』，第 178 页的『uow_id』，第 178 页的『workload_occurrence_id』，第 178 页的『coord_member』，第 178 页的『member_activation_time 』，第 178 页的『workload_name』，第 178 页的『workload_id』，第 179 页的『service_superclass_name』 {zero or one times (?)}，第 179 页的『service_subclass_name』 {zero or one times (?)}，第 179 页的『service_class_id』 {zero or one times (?)}，第 179 页的『session_authid』 {zero or one times (?)}，第 179 页的『system_authid』，第 179 页的『client_pid』，第 179 页的『client_product_id』，第 180 页的『client_platform』，第 180 页的『client_protocol』 {zero or one times (?)}，第 180 页的『client_userid』 {zero or one times (?)}，第 180 页的『client_wrkstnname』 {zero or one times (?)}，第 180 页的『client_applname』 {zero or one times (?)}，第 180 页的『client_acctng』 {zero or one times (?)}，第 181 页的『local_transaction_id』，第 181 页的『global_transaction_id 』，第 181 页的『system_metrics』，第 181 页的『client_hostname』，第 181 页的『client_port_number』，第 181 页的『uow_log_space_used』，第 182 页的『package_list』，第 182 页的『executable_list』，第 182 页的『intra_parallel_state』，ANY content（skip）{zero or more (*)}）

属性:

QName	类型	修订时间	缺省值	用法	注释
id	xs:long			必需	
类型				必需	
时间戳记	xs:dateTime			必需	
成员				必需	
release	xs:long			必需	
mon_interval_id	xs:long			必需	
任何名称空间中的任何属性					

package_id

有关更多详细信息，请参阅监视元素 第 940 页的『package_id -“程序包标识”监视元素』。

包含者: 第 172 页的『package_entry』

元素内容:

类型	构面
xs:long	

package_elapsed_time

有关更多详细信息，请参阅监视元素 第 941 页的『package_elapsed_time -“程序包耗用时间”监视元素』。

包含者: 第 172 页的『package_entry』

元素内容:

类型	构面
xs:long	

invocation_id

有关更多详细信息，请参阅监视元素 第 827 页的『invocation_id -“调用标识”监视元素』。

包含者: 第 172 页的『package_entry』

元素内容:

类型	构面
xs:int	

routine_id

有关更多详细信息，请参阅监视元素 第 1115 页的『routine_id -“例程标识”监视元素』。

包含者: 『package_entry』

元素内容:

类型	构面
xs:int	

nesting_level

有关更多详细信息，请参阅监视元素 第 906 页的『nesting_level -“嵌套级别”监视元素』。

包含者: 『package_entry』

元素内容:

类型	构面
xs:int	

package_entry

包含者: 第 173 页的『package_list_entries』

元素内容: （第 171 页的『package_id』，第 171 页的『package_elapsed_time』，第 171 页的『invocation_id』，『routine_id』，『nesting_level』，ANY content（skip）{zero or more (*)}）

属性:

QName	类型	修订时间	缺省值	用法	注释
任何名称空间中的任何属性					

package_list_size

包含者: 第 182 页的『package_list』

元素内容:

类型	构面
xs:int	

package_list_exceeded

有关更多详细信息，请参阅监视元素 第 941 页的『package_list_exceeded -“超过了程序包列表的容量”监视元素』。

包含者: 第 182 页的『package_list』

package_list_entries

包含者: 第 182 页的『package_list』

元素内容: (第 172 页的『package_entry』 {zero or more (*)})

属性:

QName	类型	修订时间	缺省值	用法	注释
任何名称空间中的任何属性					

executable_id

有关更多详细信息，请参阅监视元素 第 762 页的『executable_id -“可执行文件标识”监视元素』。

包含者: 第 176 页的『executable_entry』

num_executions

有关更多详细信息，请参阅监视元素 第 911 页的『num_executions -“语句执行次数”监视元素』。

包含者: 第 176 页的『executable_entry』

元素内容:

类型	构面
xs:long	

rows_read

有关更多详细信息，请参阅监视元素 第 1118 页的『rows_read -“读取行数”监视元素』。

包含者: 第 176 页的『executable_entry』

元素内容:

类型	构面
xs:long	

total_cpu_time

有关更多详细信息，请参阅监视元素 第 1249 页的『total_cpu_time -“CPU 时间总计”监视元素』。

包含者: 第 176 页的『executable_entry』

元素内容:

类型	构面
xs:long	

total_act_time

有关更多详细信息，请参阅监视元素 第 1231 页的『total_act_time -“活动时间总计”监视元素』。

包含者: 第 176 页的『executable_entry』

元素内容:

类型	构面
xs:long	

total_act_wait_time

有关更多详细信息，请参阅监视元素 第 1232 页的『total_act_wait_time -“活动等待时间总计”监视元素』。

包含者: 第 176 页的『executable_entry』

元素内容:

类型	构面
xs:long	

lock_wait_time

有关更多详细信息，请参阅监视元素 第 863 页的『lock_wait_time -“等待锁定时间”监视元素』。

包含者: 第 176 页的『executable_entry』

元素内容:

类型	构面
xs:long	

lock_waits

有关更多详细信息，请参阅监视元素 第 868 页的『lock_waits -“等待锁定次数”监视元素』。

包含者: 第 176 页的『executable_entry』

元素内容:

类型	构面
xs:long	

total_sorts

有关更多详细信息，请参阅监视元素 第 1292 页的『total_sorts -“排序总数”监视元素』。

包含者: 第 176 页的『executable_entry』

元素内容:

类型	构面
xs:long	

post_threshold_sorts

有关更多详细信息，请参阅监视元素 第 1079 页的『post_threshold_sorts -“超出阈值后的排序次数”监视元素』。

包含者: 第 176 页的『executable_entry』

元素内容:

类型	构面
xs:long	

post_shrthreshold_sorts

有关更多详细信息，请参阅监视元素 第 1073 页的『post_shrthreshold_sorts -“共享阈值后排序数”监视元素』。

包含者: 第 176 页的『executable_entry』

元素内容:

类型	构面
xs:long	

sort_overflows

有关更多详细信息，请参阅监视元素 第 1149 页的『sort_overflows -“排序溢出数”监视元素』。

包含者: 第 176 页的『executable_entry』

元素内容:

类型	构面
xs:long	

executable_entry

包含者: 『executable_list_entries』

元素内容: (第 173 页的『executable_id』, 第 173 页的『num_executions』, 第 173 页的『rows_read』, 第 173 页的『total_cpu_time』, 第 174 页的『total_act_time』, 第 174 页的『total_act_wait_time』, 第 174 页的『lock_wait_time』, 第 174 页的『lock_waits』, 第 175 页的『total_sorts』, 第 175 页的『post_threshold_sorts』, 第 175 页的『post_shrthreshold_sorts』, 第 175 页的『sort_overflows』, ANY content (skip) {zero or more (*)})

属性:

QName	类型	修订时间	缺省值	用法	注释
任何名称空间中的任何属性					

executable_list_size

包含者: 第 182 页的『executable_list』

元素内容:

类型	构面
xs:int	

executable_list_truncated

包含者: 第 182 页的『executable_list』

executable_list_entries

包含者: 第 182 页的『executable_list』

元素内容: (『executable_entry』 {zero or more (*)})

属性:

QName	类型	修订时间	缺省值	用法	注释
任何名称空间中的任何属性					

completion_status

工作单元的完成状态。可能的值包括:
UNKNOWN、COMMIT、ROLLBACK、GLOBAL_COMMIT、GLOBAL ROLLBACK、XA_END 或 XA_PREPARE

包含者: 第 170 页的『db2_uow_event』

start_time

工作单元的开始时间。有关更多详细信息，请参阅监视元素 第 1319 页的『uow_start_time -“工作单元开始时间戳记”监视元素』。

包含者: 第 170 页的『db2_uow_event』

元素内容:

类型	构面
xs:dateTime	

stop_time

工作单元的停止时间。有关更多详细信息，请参阅监视元素 第 1320 页的『uow_stop_time -“工作单元停止时间戳记”监视元素』。

包含者: 第 170 页的『db2_uow_event』

元素内容:

类型	构面
xs:dateTime	

connection_time

应用程序连接到数据库成员的时间。有关更多详细信息，请参阅监视元素 第 682 页的『conn_time -“数据库连接时间”监视元素』。

包含者: 第 170 页的『db2_uow_event』

元素内容:

类型	构面
xs:dateTime	

application_name

客户机上运行的应用程序在数据库中的名称。有关更多详细信息，请参阅监视元素 第 620 页的『appl_name -“应用程序名称”监视元素』。

包含者: 第 170 页的『db2_uow_event』

application_handle

应用程序在系统范围内的唯一标识。有关更多详细信息，请参阅监视元素 第 601 页的『agent_id -“应用程序句柄（代理程序标识）”监视元素』。

包含者: 第 170 页的『db2_uow_event』

application_id

当应用程序连接到数据库管理器上的数据库时，将生成此标识。有关更多详细信息，请参阅监视元素 第 616 页的『appl_id -“应用程序标识”监视元素』。

包含者: 第 170 页的『db2_uow_event』

uow_id

此活动记录所应用于的工作单元标识。有关更多详细信息，请参阅监视元素 第 1316 页的『uow_id -“工作单元标识”监视元素』。

包含者: 第 170 页的『db2_uow_event』

workload_occurrence_id

此活动记录所应用于的工作负载实例标识。有关更多详细信息，请参阅监视元素 第 1343 页的『workload_occurrence_id -“工作负载项标识”监视元素』。

包含者: 第 170 页的『db2_uow_event』

coord_member

包含者: 第 170 页的『db2_uow_event』

member_activation_time

此数据库成员的激活时间。有关更多详细信息，请参阅监视元素 第 717 页的『db_conn_time -“数据库激活时间戳记”监视元素』。

包含者: 第 170 页的『db2_uow_event』

元素内容:

类型	构面
xs:dateTime	

workload_name

在其中完成此工作单元的工作负载的名称。有关更多详细信息，请参阅监视元素 第 1342 页的『workload_name -“工作负载名称”监视元素』。

包含者: 第 170 页的『db2_uow_event』

workload_id

在其中完成此工作单元的工作负载的工作负载标识。有关更多详细信息，请参阅监视元素 第 1341 页的『workload_id -“工作负载标识”监视元素』。

包含者: 第 170 页的『db2_uow_event』

service_superclass_name

在其中完成此工作单元的服务超类的名称。有关更多详细信息，请参阅监视元素 第 1136 页的『service_superclass_name -“服务超类名”监视元素』。

包含者: 第 170 页的『db2_uow_event』

service_subclass_name

在其中完成此工作单元的服务子类的名称。有关更多详细信息，请参阅监视元素 第 1135 页的『service_subclass_name -“服务子类名”监视元素』。

包含者: 第 170 页的『db2_uow_event』

service_class_id

在其中完成此工作单元的服务类的服务类标识。有关更多详细信息，请参阅监视元素 第 1134 页的『service_class_id -“服务类标识”监视元素』。

包含者: 第 170 页的『db2_uow_event』

session_authid

调用所监视应用程序的用户的会话授权标识。有关更多详细信息，请参阅监视元素 第 1137 页的『session_auth_id -“会话授权标识”监视元素』。

包含者: 第 170 页的『db2_uow_event』

system_authid

调用所监视应用程序的用户的系统授权标识。有关更多详细信息，请参阅监视元素 第 1188 页的『system_auth_id -“系统授权标识”监视元素』。

包含者: 第 170 页的『db2_uow_event』

client_pid

客户机报告的进程标识。有关更多详细信息，请参阅监视元素 第 661 页的『client_pid -“客户机进程标识”监视元素』。

包含者: 第 170 页的『db2_uow_event』

元素内容:

类型	构面
xs:long	

client_product_id

客户机的产品标识。有关更多详细信息，请参阅监视元素 第 663 页的『client_prdid -“客户机产品和版本标识”监视元素』。

包含者: 第 170 页的『db2_uow_event』

client_platform

客户机的平台。有关更多详细信息，请参阅监视元素 第 662 页的『client_platform -“客户机操作平台”监视元素』。

包含者: 第 170 页的『db2_uow_event』

属性:

QName	类型	修订时间	缺省值	用法	注释
id	xs:short			可选	

client_protocol

客户机的产品标识。有关更多详细信息，请参阅监视元素 第 664 页的『client_protocol -“客户机通信协议”监视元素』。

包含者: 第 170 页的『db2_uow_event』

client_userid

由事务管理器生成并提供给服务器的客户机用户标识。有关更多详细信息，请参阅监视元素 第 664 页的『client_userid -“客户机用户标识”监视元素』。

包含者: 第 170 页的『db2_uow_event』

client_wrkstnname

如果在此连接中发出了 sqleseti API，那么此元素标识客户机系统或工作站。有关更多详细信息，请参阅监视元素 第 665 页的『client_wrkstnname -“客户机工作站名称”监视元素』。

包含者: 第 170 页的『db2_uow_event』

client_applname

如果在此连接中发出了 sqleseti API，那么此元素标识执行事务的服务器事务程序。有关更多详细信息，请参阅监视元素 第 658 页的『client_applname -“客户机应用程序名称”监视元素』。

包含者: 第 170 页的『db2_uow_event』

client_acctng

如果在此连接中发出了 sqleseti API，那么此元素是为了进行日志记录和诊断而传递到目标数据库的数据。有关更多详细信息，请参阅监视元素 第 657 页的『client_acctng -“客户机记帐字符串”监视元素』。

包含者: 第 170 页的『db2_uow_event』

local_transaction_id

工作单元的局部事务标识。

包含者: 第 170 页的『db2_uow_event』

global_transaction_id

工作单元的全局事务标识。

包含者: 第 170 页的『db2_uow_event』

system_metrics

工作单元的度量值。

包含者: 第 170 页的『db2_uow_event』

client_hostname

客户机的主机名。有关更多详细信息，请参阅监视元素 第 659 页的『client_hostname -“客户机主机名”监视元素』。

包含者: 第 170 页的『db2_uow_event』

client_port_number

客户机的端口号。有关更多详细信息，请参阅监视元素 第 662 页的『client_port_number -“客户机端口号”监视元素』。

包含者: 第 170 页的『db2_uow_event』

元素内容:

类型	构面
xs:int	

uow_log_space_used

在工作单元期间使用的日志空间量。有关更多详细信息，请参阅监视元素 第 1318 页的『uow_log_space_used -“使用的工作单元日志空间”监视元素』。

包含者: 第 170 页的『db2_uow_event』

元素内容:

类型	构面
xs:long	

package_list

工作单元的程序包列表。

包含者: 第 170 页的『db2_uow_event』

元素内容: (第 172 页的『package_list_size』, 第 172 页的『package_list_exceeded』, 第 173 页的『package_list_entries』, ANY content (skip) {zero or more (*)})

属性:

QName	类型	修订时间	缺省值	用法	注释
任何名称空间中的任何属性					

executable_list

工作单元的可执行列表。

包含者: 第 170 页的『db2_uow_event』

元素内容: (第 176 页的『executable_list_size』, 第 176 页的『executable_list_truncated』, 第 176 页的『executable_list_entries』, ANY content (skip) {zero or more (*)})

属性:

QName	类型	修订时间	缺省值	用法	注释
任何名称空间中的任何属性					

intra_parallel_state

工作单元的当前分区内并行性状态。可能的值为 YES 和 NO。

包含者: 第 170 页的『db2_uow_event』

工作单元事件监视器程序包列表信息:

工作单元事件监视器可以收集工作单元中使用的程序包的列表。可以使用此信息来确定应用程序中的哪些存储过程所花的时间可能比预期运行时间更长。

从 DB2 V9.7 FP1 开始，可以将有关工作单元中所使用的程序包的信息包括在事件监视器所收集的数据中。根据您为工作单元事件监视器选择的输出选项，此信息会在工作单元结束时与该事件的关联信息的余下部分一起写至无格式事件表或 UOW_PACKAGE_LIST_*evmon-name* 表（其中 *evmon-name* 是指定给事件监视器的名称）。

可以通过两种方法来控制如何捕获此信息:

- 在 CREATE 或 ALTER WORKLOAD 语句的 COLLECT UNIT OF WORK DATA 子句中，使用 PACKAGE LIST 选项来控制为特定工作负载收集此信息。如果指定了

此选项，那么会将 CREATE WORKLOAD 或 ALTER WORKLOAD 语句中所标识工作负载下执行的工作单元的信息（其中包括程序包列表信息）发送至任何处于活动状态的工作单元事件监视器。

- 可将 **mon_uow_pkglist** 配置参数设置为 ON，以便针对数据服务器执行的*所有*工作单元的包列表信息发送至任何处于活动状态的工作单元事件监视器。

  注： **mon_uow_data** 必须也设置为 BASE，才能收集包列表信息。

将为程序包列表收集以下数据：

程序包标识（第 940 页的『package_id -“程序包标识”监视元素』）
用于标识程序包的唯一标识。

嵌套级别（第 906 页的『nesting_level -“嵌套级别”监视元素』）
运行语句时有效的嵌套级别或递归级别。每个嵌套级别都对应于一个存储过程或用户定义的函数 (UDF) 的嵌套调用或递归调用。

例程标识（第 1115 页的『routine_id -“例程标识”监视元素』）
唯一的例程标识。如果此活动不是任何例程的组成部分，那么此元素将返回零。

调用标识（第 827 页的『invocation_id -“调用标识”监视元素』）
此标识用于将例程的一次调用与工作单元中位于同一嵌套级别的其他调用区分开。它在特定嵌套级别的工作单元中是唯一的。

程序包耗用时间（第 941 页的『package_elapsed_time -“程序包耗用时间”监视元素』） 执行程序包中的各个部分所耗用的时间。

按照为程序包列表收集的信息列表的建议，不仅要捕获每个程序包的信息，而且要捕获对于程序包中例程的每次调用的信息。

还要跟踪耗用时间。针对所给定调用计算的时间是从第一次执行程序包中的某个部分开始，直到数据库管理器切换到另一个程序包为止。请参阅第 186 页的『示例』，以了解有关如何跟踪耗用时间的更多信息。

如何将程序包列表写入无格式的事件表

当您允许收集程序包列表信息时，工作单元事件监视器会向每个工作单元的无格式事件 (UE) 表中写入两条记录。第一条记录包含基本工作单元事件监视器数据。下一条记录包含程序包列表信息。

程序包列表信息存储在 UE 表的 BLOB 列中。当表空间的页大小为 4k（缺省值）时，可以将具有 32 个条目的列表存储为直接插入 BLOB。由 **mon_pkglist_sz** 配置参数来控制可以写入程序包列表的条目数。此参数的缺省值为 32，这意味着程序包列表中最多可以包括 32 个条目。如果您希望增加程序包列表中可以包括的条目数，那么请确保在具有更大页大小的表空间中创建了用来存储事件监视器输出的 UE 表。假定程序包列表大小每增加 32 个条目需要将表空间的页大小增大 4k。例如，如果您希望程序包列表中有多达 64 个条目，那么应确保表空间的页大小至少为 8k。如果您增大 **mon_pkglist_sz** 而不增大表空间的页大小，那么虽然仍会创建程序包列表，但是不会直接将 BLOB 存储在表中，这可能会影响性能。

注： 可以使用 ADMIN_IS_INLINED 管理函数来确定是否会直接存储包含程序包列表信息的 BLOB。

包列表如何写至常规表

对事件监视器输出使用常规表时，包列表信息会作为第 79 页的『uow_package_list 逻辑数据组』的一部分捕获。每个工作单元完成时，会向表 UOW_PACKAGE_LIST_*evmon-name* 添加一行或多行，并为逻辑数据组中的每个监视元素添加一列。添加至该表的行数取决于有多少个包作为该工作单元的一部分运行。但是，可添加至此表的行数上限由 **mon_pkglist_sz** 配置参数控制。此参数的缺省值为 32，这意味着程序包列表中最多可以包括 32 个条目。如果要增加包列表中可包括的条目数，请增加 **mon_pkglist_sz**。

程序包列表输出

如前所述，如果事件监视器写至 UE 表，那么收集包信息时工作单元事件监视器会向 UE 表写入两个记录。用于显示 UE 表中的数据的每个界面提供了一种机制，以查看两个 UE 表记录中包含的信息。例如，**db2evmonfmt** 工具将每条记录中的信息组合成单个报告。如果使用 EVMON_FORMAT_UE_TO_TABLES 过程，那么它会生成您可连接的关系表；表 UOW_PACKAGE_LIST 包含包列表信息。EVMON_FORMAT_UE_TO_XML 会生成单个 XML 文档，该 XML 文档包含两个记录中的信息。有关更多信息，请参阅第 150 页的『访问工作单元事件监视器捕获的事件数据』。

如果事件监视器直接写至关系表，那么包列表信息会写至表 UOW_PACKAGE_LIST_*evmon-name*。

注: 在分区数据库环境中，仅在由*协调代理程序*生成的工作单元事件中报告程序包列表，并明确反映该代理程序在每个程序包中耗用的时间；它并不会反映任何其他代理程序在任何其他分区中用于这些程序包的时间。

第 185 页的图 5 显示了由工作单元事件监视器生成的信息，由 **db2evmonfmt** 工具调整其格式。

```
------------------------------------------------------
Event ID               : 12
Event Type             : UOW
Event Timestamp        : 2009-12-08-14.44.39.162707
Member                 : 0
Release                : 9070200
------------------------------------------------------

Database Level Details
----------------------
Database Member Activation Time : 2009-12-08-14.41.55.089416
Coordinator Member         : 0

Connection Level Details
------------------------
Application ID                 : *LOCAL.gstager.091208194155
Application Handle             : 21
Application Name               : db2bp
Session Authorization ID       :
System Authorization ID        :
Connection Timestamp           : 2009-12-08-14.41.55.089416
Client Process ID              : 13043
Client Platform                : LINUXX8664
Client Product ID              : SQL09072
Client Protocol                : LOCAL
Client Hostname                : HOSTX
Client Port Number             : 0

UOW Level Details
-------------------------
Start Time                     : 2009-12-08-14.44.39.160651
Stop Time                      : 2009-12-08-14.44.39.162707
Completion Status              : COMMIT
UOW ID                         : 12
Workoad Occurrence ID          : 1
Workload Name                  : SYSDEFAULTUSERWORKLOAD
Workoad ID                     : 1
Service Superclass Name        : SYSDEFAULTUSERCLASS
Service Subclass Name          : SYSDEFAULTSUBCLASS
Service Class ID               : 13
Client Userid                  :
Client Workstation Name        :
Client Application Name        :
Client Accounting String       :
Local Transaction ID           : 000000000000013B
Global Transaction ID          : 00000000000000000000000000000000000000000000
Log Space Used                 : 124

UOW Metrics
-------------------------
TOTAL_CPU_TIME            : 1591
TOTAL_WAIT_TIME           : 8363
ACT_ABORTED_TOTAL         : 0
ACT_COMPLETED_TOTAL       : 1
ACT_REJECTED_TOTAL:       : 0
AGENT_WAIT_TIME           : 87
AGENT_WAITS_TOTAL         : 1
APP_RQSTS_COMPLETED_TOTAL : 1
    .
    .
    .

Package List
-------------------------
Package List Size       : 2
Package List Exceeded   : no

PACKAGE_ID           NESTING_LEVEL ROUTINE_ID  INVOCATION_ID        PACKAGE_ELAPSED_TIME
-------------------- ------------- ----------- -------------------- --------------------
240                  0             0           0                    0
330                  1             66539       1                    1
```

注: 已经排除了“UOW 度量值”部分中的某些度量值。

图 5. 工作单元事件监视器的样本输出以及程序包列表信息

package_list_count 监视元素（前一个报告中的“程序包列表大小”）中反映了出现在所给定工作单元的程序包列表中的程序包数目，此数目与基本工作单元事件监视器数据包括在一起。如果与工作单元配合使用的程序包数目超过了在 **mon_pkglist_sz** 配置参数中所指定的值，那么不会将其他程序包包括在程序包列表中。但是，

package_list_exceeded 监视元素指示程序包数目是否超过了程序包列表可以容纳的数目。此监视元素与工作单元事件监视器的基本信息（第 185 页的图 5 中的“Package List Exceeded”）一起返回。如果此监视元素的值为 YES，那么可以增大 **mon_pkglist_sz** 的值，以使程序包列表中包括更多程序包。

示例

下面的每个示例都显示了为程序包列表返回的信息，此信息与由 **db2evmonfmt** 工具显示的信息相同。

示例 1：将执行单个程序包中的一个或多个部分的应用程序

在此示例中，为此工作单元运行了一个程序包标识为 300 的程序包。

```
PACKAGE_ID  NESTING_LEVEL  ROUTINE_ID INVOCATION_ID ELAPSED_TIME
----------  -------------  ---------- ------------- ------------
300         0              0          0             100
```

在此示例中，程序包列表中有一个条目，此条目反映执行程序包中的一个或多个部分的情况。所执行的同一程序包中的所有部分都被认为是同一程序包调用的一部分。

示例 2：应用程序将调用程序包中的存储过程

在此示例中，程序包标识为 300 的程序包将调用标识为 806 的存储过程。执行了此存储过程中的三个部分。

```
PACKAGE_ID  NESTING_LEVEL  ROUTINE_ID INOVATION_ID ELAPSED_TIME
----------  -------------  ---------- ------------ ------------
300         0              0          0            21
300         1              806        1            100
```

此输出显示了列表中的两个条目。一个条目用于调用此存储过程，另一个条目用于执行此存储过程中的三个部分。列表中的第二个条目的 NESTING_LEVEL 反映已从另一个程序包中调用了此存储过程这一事实。

示例 3：应用程序执行两个不同的程序包中的部分

在此示例中，应用程序将执行一个程序包中的部分，再执行另一个程序包中的部分，然后返回到第一个程序包。未调用存储过程。下列伪码表示此工作单元:

```
Application
    EXEC PACKAGEA
    EXEC PACKAGEB
    EXEC PACKAGEA
```

还假定调用 **PACKAGEA** 需要 100 ms，调用 **PACKAGEB** 需要 25 ms，且调用 **PACKAGEC** 需要 460 ms。以下输出显示程序包列表的显示效果:

```
PACKAGE_ID  NESTING_LEVEL  ROUTINE_ID INVOCATION_ID ELAPSED_TIME
----------  -------------  ---------- ------------- ------------
300         0              0          0             560
301         0              0          0             25
```

在此示例中，列表中有两个条目。PACKAGE_ID 为 300 的程序包 A 的各个部分总共运行 560 ms。程序包 B 会运行了 25 ms。通过单行来表示程序包 A，因为每个调用都具有相同的 INVOCATION_ID 和 NESTING_LEVEL。INVOCATION_ID 和 NESTING_LEVEL 保持为 0，因为在任一程序包中都未调用存储过程。

示例 4：应用程序执行多个程序包中的部分和存储过程

在此示例中，有三个程序包，程序包标识分别为 100、101 和 102。该应用程序位于程序包 100 中。有两个标识分别为 201 和 202 的存储过程。第一个存储过程 (SP1) 位于程序包 101 中，第二个存储过程 (SP2) 位于程序包 102 中。下列伪码表示此工作单元:

```
Application
   CALL SP1 a
      INSERT INTO T1 VALUES(7)  b
      CALL SP2 c
         INSERT INTO T2 VALUES(8)
      CALL SP2 d
         INSERT INTO T2 VALUES(8)
```

此工作单元的程序包列表将为如下所示:

```
PACKAGE_ID  NESTING_LEVEL  ROUTINE_ID INVOCATION_ID ELAPSED_TIME
----------  -------------  ---------- ------------- ------------
100         0              0          0             21
101         1 1            201        1             40
102         2 2            202        1 3           35
102         2              202        2 3           35
```

在前面的输出中有四个条目:

- 第一个条目对应于调用第一个程序包中的 SP1，该伪码中的 **a** 行表示此工作单元。
- 第二个条目对应于执行程序包 101 中标识为 201 的存储过程中的部分。这些部分包括 **b**、**c** 和 **d** 行。嵌套级别增大为 1，如 **1** 所示。
- 第三个条目表示执行 SP2 中的第一个 INSERT INTO T2 语句（从 SP1 中调用存储过程）。嵌套级别将再次增大（**2**）。
- 列表中的第四个条目表示执行 SP2 中的第二个 INSERT INTO T2 语句。嵌套级别将保持不变，因为与上一次调用 SP2 一样，也是从 SP1 调用此存储过程。但是，因为这两个语句用于此存储过程的不同调用中，所以它们具有不同的调用标识（**3**）。因此，程序包列表中有两个不同的条目。

可执行文件列表信息:

收集工作单元信息时，您可选择同时收集作为每个工作单元的一部分运行的语句的可执行标识列表。

有关可执行标识的信息将写至无格式事件 (UE) 表或常规表。有两种方法可用来捕获此信息:

- 对 CREATE WORKLOAD 或 ALTER WORKLOAD 语句的 COLLECT UNIT OF WORK DATA 子句使用 EXECUTABLE LIST 选项来收集特定工作负载的信息。有关您在语句中标识的工作负载下执行的工作单元的信息（包括可执行标识）将发送至处于活动状态的工作单元 (UOW) 事件监视器。
- 使用配置参数将有关数据服务器上执行的所有工作单元的信息（包括可执行信息）发送至处于活动状态的工作单元事件监视器。要收集可执行标识信息，请将 **mon_uow_data** 配置参数设置为 BASE，并将 **mon_uow_execlist** 配置参数设置为 ON。

针对可执行文件列表收集了以下数据:

工作单元级别

executable_list_size
特定工作单元的可执行标识列表中的条目数。

executable_list_truncated
YES 或 NO 值，用于指示该列表是否已截断。如果处理期间没有足够可用内存来存储整个可执行文件列表，那么该列表可能会被截断。

可执行标识列表

executable_id（第 762 页的『**executable_id -"可执行文件标识"监视元素**』）
在数据服务器上生成的加密二进制标记，用于唯一地标识已执行的 SQL 语句节。

num_executions（第 911 页的『**num_executions -"语句执行次数"监视元素**』） 已执行 SQL 语句的次数。

rows_read（第 1118 页的『**rows_read -"读取行数"监视元素**』）
从表中读取的行数。

total_cpu_time（第 1249 页的『**total_cpu_time -"CPU 时间总计"监视元素**』） 在 DB2 产品中耗用的 CPU 时间总计。此值代表用户 CPU 时间与系统 CPU 时间的总计。此值以微秒计。

total_act_time（第 1231 页的『**total_act_time -"活动时间总计"监视元素**』）
执行活动时的耗用时间总计。此值以毫秒计。

total_act_wait_time（第 1232 页的『**total_act_wait_time -"活动等待时间总计"监视元素**』）
处理活动期间，在 DB2 数据库服务器中进行等待时的耗用时间总计。此值以毫秒计。

lock_wait_time（第 863 页的『**lock_wait_time -"等待锁定时间"监视元素**』）
等待锁定的耗用时间总计。此值以毫秒计。

lock_waits（第 868 页的『**lock_waits -"等待锁定次数"监视元素**』）
应用程序或连接等待锁定的总次数。

total_sorts（第 1292 页的『**total_sorts -"排序总数"监视元素**』）
已执行的排序总数。

post_threshold_sorts（第 1079 页的『**post_threshold_sorts -"超出阈值后的排序次数"监视元素**』）
超过排序堆阈值后请求堆的排序数。

post_shrthreshold_sorts（第 1073 页的『**post_shrthreshold_sorts -"共享阈值后排序数"监视元素**』）
排序内存限量算法已限制的排序总数。受限排序是被授予内存量少于排序内存管理器所请求内存量的排序

sort_overflows（第 1149 页的『**sort_overflows -"排序溢出数"监视元素**』）
用完排序堆并且可能需要磁盘空间以供临时存储器使用的排序总数。

执行列表如何写至 UE 表

收集 UOW 事件监视器的基本数据和可执行标识列表数据时，至少可将两个不同记录写至 UE 表。第一个记录包含有关包含基本 UOW 数据的 UOW 事件的信息。第二个记录是包含可执行标识列表数据的 UOW_EXEC_LIST 事件。第二个记录可能由多个记录组成，因为单个 UOW 可能有大量的唯一可执行标识。这些记录会作为不同行写至 UE 表以确保每个事件包含在直接插入的可用 LOB 空间中。可使用用于设置 UE 表格式的

界面来合并这些事件的信息。如果未收集可执行标识列表，那么不会创建关联记录；该表不包含任何行。

执行列表如何写至常规表

将常规表用于事件监视器输出时，可执行列表信息作为第 74 页的『uow_executable_list 逻辑数据组』组的一部分捕获。每个工作单元完成时，会向表 UOW_EXECUTABLE_LIST_*evmon_name* 添加一行或多行，并为逻辑数据组中的每个监视元素添加一列。添加至该表的行数取决于有多少个唯一可执行标识作为该工作单元的一部分运行。

可执行列表输出

如果事件监视器写至 UE 表，那么收集执行信息时工作单元事件监视器会将两个记录写至 UE 表。用于显示 UE 表中的数据的每个界面提供了一种机制，以查看两个 UE 表记录中包含的信息。**db2evmonfmt** 工具将每个记录中的信息组合成单个报告。EVMON_FORMAT_UE_TO_TABLES 过程生成您可连接的关系表；表 UOW_EXECUTABLE_LIST 包含可执行列表信息。EVMON_FORMAT_UE_TO_XML 表函数会生成单个 XML 文档，该 XML 文档包含这两个记录中的信息。有关更多信息，请参阅第 150 页的『访问工作单元事件监视器捕获的事件数据』。

如果该事件监视器直接将数据写至关系表，那么可执行列表信息会写至表 UOW_EXECUTABLE_LIST_*evmon_name*。

在分区数据库环境中，将对每个成员（包括每个协调代理程序成员和数据成员）生成可执行标识列表。在 DB2 pureScale 环境中，将根据协调程序成员生成该列表（这类似于非分区配置中的情况）。

示例

以下样本信息是针对执行 UOW 内的 5 个不同 SQL 语句部分的应用程序收集的。此输出提供带有样本列的逻辑视图；实际输出取决于您运行的工具或查询。

```
EXECUTABLE_ID                             NUM_EXECUTIONS ROWS_READ TOTAL_CPU_TIME
----------------------------------------  -------------- --------- --------------
x'01007A00000020020081126171554951791'               1     23456          76888
x'01007900000020020081126171533551120'              55       345            768
x'01007C00000020020081126171720728997'             234        67            232
x'01007B00000020020081126171657272914'            3456       347           1223
x'01007D00000020020081126172409987719'           22242      2244         432444
```

在此示例中，可执行标识列表中有 5 个条目对应已执行的 5 个不同部分。如 NUM_EXECUTIONS 列中所示，这五个部分的执行次数不同，但只为每个唯一部分提供一个条目。第一行可能指示存在问题的活动语句，因为执行一次该语句就消耗了太多 CPU 时间。

收集工作单元事件数据并生成报告

可使用工作单元事件监视器来收集有关可用于退款的事务的数据。已收集事务事件数据在无格式事件表中为不可读格式。可使用此数据来创建可读文本报告。

开始之前

要收集工作单元事件监视器数据，您必须具有 SYSADM 或 SYSCTRL 权限。

关于此任务

本任务提供有关收集特定工作负载的工作单元事件数据的指示信息。

如果将 **mon_uow_pkglist** 和 **mon_uow_execlist** 配置参数都设置为 ON，那么还会收集程序包列表和执行列表信息。或者，可通过按如下所示改变 ALTER WORKLOAD 语句来收集工作负载的程序包列表和执行列表信息而不管 **mon_uow_pkglist** 和 **mon_uow_execlist** 配置参数的设置:

- 要获取程序包列表信息，请将 BASE 选项替换为 BASE INCLUDE PACKAGE LIST 选项。
- 要获取执行列表信息，请将 BASE 选项替换为 BASE INCLUDE EXECUTABLE LIST 选项。
- 要获取程序包列表和执行列表信息，请将 BASE 选项替换为 BASE INCLUDE PACKAGE LIST, EXECUTABLE LIST 选项。

工作单元事件监视器收集用于标识应用程序事务和相应 CPU 使用情况的信息。工作单元事件监视器针对事务事件收集的信息的示例如下所示:

- CPU 使用时间总计（TOTAL_CPU_TIME 监视元素）
- 应用程序句柄（APPLICATION_HANDLE 监视元素）

限制

如果您不具有 SYSADM 或 SYSCTRL 权限，那么输入数据值不可查看。

过程

要收集有关工作单元事件的详细信息，请执行以下操作:

1. 通过发出 CREATE EVENT MONITOR FOR UNIT OF WORK 语句来创建称为 UOWEVMON 的工作单元事件监视器，如以下示例所示:

   ```
   CREATE EVENT MONITOR UOWEVMON FOR UNIT OF WORK
      WRITE TO UNFORMATTED EVENT TABLE
   ```

2. 通过发出以下语句来激活 UOWEVMON 工作单元事件监视器:

   ```
   SET EVENT MONITOR UOWEVMON STATE 1
   ```

3. 通过发出带语句历史记录的 ALTER WORKLOAD 语句以在工作负载级别启用工作单元事件数据收集。 例如，要收集 FINANCE 和 PAYROLL 应用程序的工作单元数据，请发出以下语句:

   ```
   ALTER WORKLOAD finance COLLECT UNIT OF WORK DATA BASE
   ALTER WORKLOAD payroll COLLECT UNIT OF WORK DATA BASE
   ```

4. 要收集工作单元事务事件，请重新运行工作负载。
5. 连接到数据库。
6. 使用 XML 解析器工具 **db2evmonfmt** 来生成基于无格式事件表中收集的事件数据的纯文本报告，如以下示例所示:

   ```
   java db2evmonfmt -d db_name -ue table_name -ftext -u user_id -p password
   ```

7. 分析该报告，确定应用程序使用的 CPU 时间量，以便相应地进行扣费。
8. 如果要对 FINANCE 和 PAYROLL 应用程序关闭工作单元数据收集，请发出以下语句:

```
ALTER WORKLOAD finance COLLECT UNIT OF WORK DATA NONE
ALTER WORKLOAD payroll COLLECT UNIT OF WORK DATA NONE
```

示例

以下报告示例是通过使用 **db2evmonfmt** 工具转换工作单元事件监视器在无格式事件表中收集的数据来获取的:

```
------------------------------------------------------
Event ID        : 1
Event Type              : UOW
Event Timestamp         : 2008-10-31-13.29.04.130849
Member of detection     : 0
------------------------------------------------------

Database Level Details
----------------------
Member Activation Time     : 2008-10-31T13:28:48.538973
Coordinator Member         : 0

Connection Level Details
------------------------
Application ID             : *LOCAL.gstager.081031172848
Application Handle         : 20
Application Name :       db2bp
Session Authorization ID   : GSTAGER
System Authorization ID    : GSTAGER
Connection Timestamp       : 2008-10-31T13:28:48.538973
Client Process ID          : 28167
Client Platform            : 30
Client Product ID          : SQL09070
Client Hostname            : gilera
Client Port Number         : 30143

UOW Level Details
-----------------------
Start Time                 : 2008-10-31T13:28:51.560138
Stop Time                  : 2008-10-31T13:29:04.130849
Completion Status          : COMMIT
UOW ID                     : 5
Workoad Occurrence ID      : 1
Workload Name              : SYSDEFAULTUSERWORKLOAD
Workoad ID                 : 1
Client userid              :
Client Workstation Name    :
Client Application Name    :
Client Accounting String   :
Local Transaction ID       : 00000000000000EB
Global Transaction ID      : 00000000000000000000000000000000000000000000
Log Space Used             : 0

UOW Metrics
-----------------------
TOTAL_CPU_TIME                : 7459
TOTAL_WAIT_TIME               : 0
ACT_ABORTED_TOTAL             : 0
...
```

通过工作单元事件监视器来计算不同应用程序或工作负载所使用的 CPU 时间:

此主题显示在日常数据库操作中使用工作单元事件监视器的一种方法。

在某些业务环境中，将对部门的应用程序所使用的处理时间进行计费。您可使用工作单元事件来记录不同应用程序、工作负载或服务类所使用的 CPU 时间。此信息转而可用于为系统资源执行计费的记账应用程序。

开始之前

CREATE EVENT MONITOR 语句需要页大小至少为 8K 的表空间来存储事件监视器生成的未格式化事件 (UE) 表。除非在 CREATE EVENT MONITOR 语句中显式命名了表空间，否则使用数据库的缺省表空间。

关于此任务

此任务描述“拒付”记账的一个基本场景。在随后的示例中，将跟踪在系统上执行的所有工作。通过收集的数据，将创建一些显示不同应用程序使用的 CPU 时间的报告。

根据您的组织的建立方式，根据工作负载跟踪系统事件可能是恰当的。或者，您也可以按特定工作负载，甚至按不同用户查看不同服务超级类中使用的 CPU 时间。如果数据写入到关系表（如本任务中的示例所示），那么您可使用 SQL 以几乎无限制的方法来查询和显示数据。

注: 工作单元中的活动可运行在不同服务子类中。因此，按服务子类聚集工作单元信息是不恰当的。如果要按服务类聚集 CPU 时间，请改用活动事件监视器。

过程

1. 创建一个工作单元事件监视器以在工作单元完成时捕获其有关信息。 例如，要创建一个名为 TRACKWORK 的事件监视器，您可使用以下 SQL:

   ```
   CREATE EVENT MONITOR TRACKWORK FOR UNIT OF WORK WRITE TO UNFORMATTED EVENT TABLE
   ```

   此语句创建一个写入到无格式事件 (UE) 表的工作单元事件监视器。UE 表与事件监视器自身 TRACKWORK 具有相同名称，并且 UE 表存储在缺省表空间中。
2. 通过运行以下命令告知数据库管理器您要收集数据库上完成的所有工作单元的事件信息:

   ```
   UPDATE DATABASE CONFIGURATION FOR dbname USING MON_UOW_DATA BASE
   ```

   此命令将导致在工作单元完成时，有关在数据服务器上执行的所有工作单元的信息发送到活动的工作单元事件监视器。有关控制收集的工作单元数据范围的更多信息，请参阅第 149 页的『配置数据收集功能』。
3. 接着，激活事件监视器:

   ```
   SET EVENT MONITOR TRACKWORK STATE 1
   ```

   注: 缺省情况下，事件监视器在数据库激活时自动启动，因为缺省情况下应用 AUTOSTART 选项。但是，由于此事件监视器是在已经活动的数据库中创建的，因此您必须使用 **SET EVENT MONITOR** 命令手动启动此事件监视器。
   从此时起，工作单元事件监视器将在每个工作单元运行完成时捕获其信息。当每个工作单元完成时，事件监视器会将事件的记录添加到 UE 表 TRACKWORK。
4. 当您准备好收集用于报告的数据，必须从 UE 表 TRACKWORK 中提取记录。

   您可使用 EVMON_FORMAT_UE_TO_XML 或 EVMON_FORMAT_UE_TO_TABLES 过程来转换 UE 表中的数据，从而以 XML 格式或关系格式来查看此信息。或者，

也可使用 **db2evmonfmt** 工具创建事件监视器返回的信息的文本报告。此示例显示使用 EVMON_FORMAT_UE_TO_TABLES 创建关系表，您可以通过适合您的需求的任何方式来查询这些关系表。

```
CALL EVMON_FORMAT_UE_TO_TABLES
   ('UOW', NULL, NULL, NULL, NULL, NULL, NULL, -1, 'SELECT * FROM TRACKWORK')
```

EVMON_FORMAT_UE_TO_TABLES 过程检查由事件监视器生成的 UE 表 TRACKWORK；此过程从 UE 表中选择每个记录，然后从这些记录中，在两个关系表中创建包含工作单元事件监视器收集的数据的行:

- UOW_EVENT
- UOW_METRICS

第一个表包含最常用的监视元素以及与捕获的每个事件相关联的度量值。第二个表包含每个事件的详细度量值。

注意:

- 如果您在步骤 第 192 页的2 中为 **MON_UOW_DATA** 配置参数指定 PKGLIST 而非 BASE，那么 EVMON_FORMAT_UE_TO_TABLES 过程将创建第三个名为 UOW_PACKAGE_LIST 的表。此表包含与工作单元相关的程序包列表信息。但是，在此示例中，由于仅收集基本监视器元素（请参阅步骤 第 192 页的2），因而此表不包含任何数据。（有关如何使用程序包列表的更多信息，请参阅第 182 页的『工作单元事件监视器程序包列表信息』。）
- UOW_METRICS 列中的值还可在 UOW_METRICS 表的 METRICS 列中包含的 XML 文档中找到。为了更加方便以及面向列的访问，这些值在 UOW_METRICS 表中提供。

5. 对先前步骤中生成的表进行查询以查看应用程序使用 CPU 时间的方式。随后的语句将返回在工作单元事件监视器初始化后，由系统上不同用户使用的 CPU 时间总计的统计分析。（此示例假定客户机应用程序已经使用 sqleseti API 或通过您正在使用的任何应用程序开发环境（如 IBM Rational® Application Developer for WebSphere® Software）将其自身与数据库对应。

```
SELECT SUBSTR(E.CLIENT_USERID,1,10) AS CLIENT_ID,
       SUBSTR(E.CLIENT_APPLNAME,1,80) AS CLIENT_APP,
       SUBSTR(E.CLIENT_WRKSTNNAME,1,10) AS WKSTN,
       SUM(M.TOTAL_CPU_TIME) AS CPU_TIME
       FROM UOW_EVENT E, UOW_METRICS M
WHERE  M.APPLICATION_ID = E.APPLICATION_ID
   AND M.UOW_ID = E.UOW_ID
   AND M.MEMBER = E.MEMBER
GROUP BY E.CLIENT_USERID, E.CLIENT_APPLNAME, E.CLIENT_WRKSTNNAME
ORDER BY CPU_TIME DESC;
```

上述查询将返回下列结果:

```
CLIENT_ID  CLIENT_APP                                                                      WKSTN      CPU_TIME
---------- ------------------------------------------------------------------------------- ---------- --------------------
                                                                                                      987770013
           DB2BATCH                                                                                   249375000
           CLP C:\DOCUME~1\ALLUSE~1\APPLIC~1\IBM\DB2\DB2COPY1\DB2\TMP\CCSCRIPT1003021324173            91181678
           CLP C:\DOCUME~1\ALLUSE~1\APPLIC~1\IBM\DB2\DB2COPY1\DB2\TMP\CCSCRIPT1004201047173            66097348
           CLP C:\DOCUME~1\ALLUSE~1\APPLIC~1\IBM\DB2\DB2COPY1\DB2\TMP\CCSCRIPT1003191536588            28824420
           CLP C:\DOCUME~1\ALLUSE~1\APPLIC~1\IBM\DB2\DB2COPY1\DB2\TMP\CCSCRIPT1003191536434            27555568
           CLP C:\DOCUME~1\ALLUSE~1\APPLIC~1\IBM\DB2\DB2COPY1\DB2\TMP\CCSCRIPT1003221122075            16203116
           CLP C:\DOCUME~1\ALLUSE~1\APPLIC~1\IBM\DB2\DB2COPY1\DB2\TMP\CCSCRIPT1003221118191            15759227
           CLP C:\DOCUME~1\ALLUSE~1\APPLIC~1\IBM\DB2\DB2COPY1\DB2\TMP\CCSCRIPT1003221531062            15630121
           CLP C:\DOCUME~1\ALLUSE~1\APPLIC~1\IBM\DB2\DB2COPY1\DB2\TMP\CCSCRIPT1003221117466            15236718
           CLP C:\DOCUME~1\ALLUSE~1\APPLIC~1\IBM\DB2\DB2COPY1\DB2\TMP\CCSCRIPT1003221116141            14607249
           CLP C:\DOCUME~1\ALLUSE~1\APPLIC~1\IBM\DB2\DB2COPY1\DB2\TMP\CCSCRIPT1003251550366            14427883
```

```
CLP C:\DOCUME~1\ALLUSE~1\APPLIC~1\IBM\DB2\DB2COPY1\DB2\TMP\CCSCRIPT1003051054311                          1312500
CLP C:\DOCUME~1\ALLUSE~1\APPLIC~1\IBM\DB2\DB2COPY1\DB2\TMP\CCSCRIPT1003051053301                          1296875
CLP C:\DOCUME~1\ALLUSE~1\APPLIC~1\IBM\DB2\DB2COPY1\DB2\TMP\CCSCRIPT1003051139066                          1296875
CLP C:\DOCUME~1\ALLUSE~1\APPLIC~1\IBM\DB2\DB2COPY1\DB2\TMP\CCSCRIPT1003051152281                          1281250
CLP C:\DOCUME~1\ALLUSE~1\APPLIC~1\IBM\DB2\DB2COPY1\DB2\TMP\CCSCRIPT1003041230283                          1046875
                                                                                  asrisk2                 1031250
CLP C:\DOCUME~1\ALLUSE~1\APPLIC~1\IBM\DB2\DB2COPY1\DB2\TMP\CCSCRIPT1003291503479                           515625
CLP C:\DOCUME~1\ALLUSE~1\APPLIC~1\IBM\DB2\DB2COPY1\DB2\TMP\CCSCRIPT1003251506219                           484375
CLP C:\DOCUME~1\ALLUSE~1\APPLIC~1\IBM\DB2\DB2COPY1\DB2\TMP\CCSCRIPT1003221444488                           453125
CLP C:\DOCUME~1\ALLUSE~1\APPLIC~1\IBM\DB2\DB2COPY1\DB2\TMP\CCSCRIPT1003021323249                           406250
CLP C:\DOCUME~1\ALLUSE~1\APPLIC~1\IBM\DB2\DB2COPY1\DB2\TMP\CCSCRIPT1003251544498                           296875
CLP C:\DOCUME~1\ALLUSE~1\APPLIC~1\IBM\DB2\DB2COPY1\DB2\TMP\CCSCRIPT1003171431559                           171875
CLP C:\DOCUME~1\ALLUSE~1\APPLIC~1\IBM\DB2\DB2COPY1\DB2\TMP\CCSCRIPT1003041227488                           156250
CLP C:\DOCUME~1\ALLUSE~1\APPLIC~1\IBM\DB2\DB2COPY1\DB2\TMP\CCSCRIPT1003221117188                           109375
CLP C:\DOCUME~1\ALLUSE~1\APPLIC~1\IBM\DB2\DB2COPY1\DB2\TMP\CCSCRIPT1003021333329                            62500
CLP C:\DOCUME~1\ALLUSE~1\APPLIC~1\IBM\DB2\DB2COPY1\DB2\TMP\CCSCRIPT1003191502148                            62500
CLP C:\DOCUME~1\ALLUSE~1\APPLIC~1\IBM\DB2\DB2COPY1\DB2\TMP\CCSCRIPT1003191527385                            62500
CLP C:\DOCUME~1\ALLUSE~1\APPLIC~1\IBM\DB2\DB2COPY1\DB2\TMP\CCSCRIPT1003191528492                            62500
CLP C:\DOCUME~1\ALLUSE~1\APPLIC~1\IBM\DB2\DB2COPY1\DB2\TMP\CCSCRIPT1003191530518                            62500
CLP C:\DOCUME~1\ALLUSE~1\APPLIC~1\IBM\DB2\DB2COPY1\DB2\TMP\CCSCRIPT1003191533265                            62500
CLP C:\Documents and Settings\All Users\Application Data\IBM\DB2\DB2COPY1\DB2DAS                             62500
```

6. 此时，工作单元事件监视器 TRACKWORK 仍在收集信息。根据您要如何跟踪不同应用程序、用户或工作负载使用的 CPU 时间，可选择采取下列一项操作:
   - 如果您要每天计算 CPU 使用情况，那么可保留此工作单元事件监视器处于活动状态。每天运行 EVMON_FORMAT_UE_TO_TABLES 过程以仅检索前一天的耗用时间度量值:

     ```
     CALL EVMON_FORMAT_UE_TO_TABLES
          ('UOW', NULL, NULL, NULL, NULL, NULL, NULL, -1,
           'SELECT * FROM TRACKWORK
           WHERE (DATE(EVENT_TIMESTAMP)=(CURRENT DATE - 1 DAY))'
          )
     ```

     通过此方法，EVMON_FORMAT_UE_TO_TABLES 过程生成的三个关系表继续增长，从而不断提供 CPU 使用情况历史记录。步骤 第 193 页的 5 中的查询返回自使用 EVMON_FORMAT_UE_TO_TABLES 过程首次创建表后，累计的 CPU 时间总计。您可修改该查询以仅显示前一天的结果，如下所示:

     ```
     SELECT SUBSTR(E.CLIENT_USERID,1,10) AS CLIENT_ID,
            SUBSTR(E.CLIENT_APPLNAME,1,80) AS CLIENT_APP,
            SUBSTR(E.CLIENT_WRKSTNNAME,1,10) AS WKSTN,
            SUM(M.TOTAL_CPU_TIME) AS CPU_TIME
            FROM UOW_EVENT E, UOW_METRICS M
     WHERE  M.APPLICATION_ID = E.APPLICATION_ID
        AND M.UOW_ID = E.UOW_ID
        AND M.MEMBER = E.MEMBER
        AND(DATE(E.EVENT_TIMESTAMP)=(CURRENT DATE - 1 DAY))
     GROUP BY E.CLIENT_USERID, E.CLIENT_APPLNAME, E.CLIENT_WRKSTNNAME
     ORDER BY CPU_TIME DESC;
     ```

     提示: 如果您要每天跟踪 CPU 使用情况，但还要管理在系统上收集的数据量，那么在更新关系表后，从 UE 表中除去不再需要的数据。例如，要从 UE 表 TRACKWORK 中删除前一天收集的数据，请使用类似以下的 DELETE 语句:

     ```
     DELETE FROM TRACKWORK WHERE (DATE(EVENT_TIMESTAMP)=(CURRENT DATE - 1 DAY))
     ```

     当事件监视器处于活动状态时，其将挂起针对任何表（即事件监视器将信息写入到任何表）的意向互斥 (IX) 表锁定，从而避免在事件监视器仍在使用这些表的情况下删除这些表。当正在删除更大数量的行时，DELETE 语句将获取更大数量的行锁定。在此情况下，可能发生锁定升级，因为行锁定可能转换为表锁定。由于事件监视器已具有针对表的锁定，因此对表锁定的这一请求可导致 DELETE 语句挂起。

要避免这种情况，在发出 DELETE 语句前请首先考虑设置锁定超时:

```
SET CURRENT LOCK TIMEOUT 60
```

如果增加锁定超时时间段没有解决该问题，请尝试删除更小的数据子集，如更小时间段的记录（例如，6 或 12 个小时）。此方法要求更少的锁定，这将减少发生锁定升级的可能性。

您还可根据平衡存储需求以及查看历史数据的需要来修剪 EVMON_FORMAT_UE_TO_TABLES 生成的关系表。

- 如果您完成了 CPU 时间计算，可通过执行以下步骤来停止事件监视器信息的收集并删除事件监视器及其相关表:
  a. 使用 **SET EVENT MONITOR TRACKWORK STATE 0** 命令为此事件监视器信息禁用工作单元的收集。
  b. 使用 DROP EVENT MONITOR 语句删除事件监视器自身。
  c. 使用 DROP TABLE 语句删除与事件监视器相关的表。在此情况下，总共有四个要删除的表:
     – TRACKWORK，用于从事件监视器收集信息的 UE 表
     – UOW_EVENT
     – UOW_METRICS
     – UOW_PACKAGE_LIST
  d. 可选: 如果没有其余的活动事件监视器，那么您可能要使用以下命令更新数据库配置从而不收集任何工作单元事件信息:

     ```
     UPDATE DATABASE CONFIGURATION FOR dbname USING MON_UOW_DATA NONE
     ```

变体: 收集特定工作负载的度量值

前一个示例向您演示了如何捕获在系统上执行的所有工作的工作单元度量值。使用 **UPDATE DATABASE CONFIGURATION** 命令设置收集的数据的范围可能导致收集的信息超过您的预期。例如，您希望仅跟踪由特定工作负载完成的工作。在此情况下，不必启用对整个数据库上工作单元信息的收集（如步骤 第 192 页的 2 中所示），而是可使用 CREATE 或 ALTER WORKLOAD 语句指定 COLLECT UNIT OF WORK DATA 子句。此子句使得事件监视器仅收集指定工作负载的数据。例如，要收集名为 PAYROLL 的工作负载的工作单元数据，请使用以下语句:

```
ALTER WORKLOAD PAYROLL COLLNECT UNIT OF WORK DATA BASE
```

您可通过对每个工作负载运行 ALTER WORKLOAD 语句而收集多个工作负载的数据。

其余步骤是相同的，但步骤 第 193 页的 5 除外，您需要在该步骤中将查询更改为类似如下内容:

```
SELECT E.WORKLOAD_NAME,
       SUM(M.TOTAL_CPU_TIME) AS CPU_TIME
       FROM UOW_EVENT E, UOW_METRICS M
WHERE  M.APPLICATION_ID = E.APPLICATION_ID
   AND M.UOW_ID = E.UOW_ID
   AND M.MEMBER = E.MEMBER
GROUP  BY E.WORKLOAD_NAME
ORDER  BY CPU_TIME DESC
```

前置语句报告为其启用了度量值收集的每个工作负载 CPU 时间:

```
WORKLOAD                       CPU_TIME
------------------------------ --------------------
PAYROLL                                  2143292042
MARKETING                                492784916

  2 record(s) selected.
```

程序包高速缓存语句逐出事件监视

程序包高速缓存事件监视器将捕获与已经从数据库程序包高速缓存中写入的语句条目相关的数据。此事件监视器提供了程序包高速缓存内容的历史记录，这有助于解决 SQL 查询性能和问题确定方面的问题。

概述

程序包高速缓存事件监视器将与 MON_GET_PKG_CACHE_STMT 表函数收集相同的信息，其中包括所有可用的活动度量值以及条目的可执行部分信息。

从 V10.1 开始，可获取与运行时间最长的语句相关的输入自变量的相关信息。此语句是与监视元素 **max_coord_stmt_exec_time** 相关联的语句。与此语句相关联的输入自变量记录在 pkgcache_stmt_args 逻辑数据组中。

针对 CREATE EVENT MONITOR 语句的两种控制机制有助于限制可以捕获的数据量。这两种控制机制提供了下列功能:

1. 使用 WHERE 子句并根据下面的一个或多个条件来过滤条目:
   - 最近一次更新某个条目的度量是否是在除去该条目之前的特定时间之后进行的 (UPDATED_SINCE_BOUNDARY_TIME)。仅当最近一次更新度量的时间晚于为事件监视器定义的边界时间时，才会收集该条目。可以使用 MON_GET_PKG_CACHE_STMT 表函数来设置事件监视器的边界时间。如果尚未对事件监视器设置边界时间，那么 UPDATED_SINCE_BOUNDARY_TIME 子句将不起作用。
   - 执行某个条目的部分的次数 (NUM_EXECUTIONS)
   - 执行语句所耗用的时间总计 (STMT_EXEC_TIME)
2. COLLECT DATA 子句选项:
   - COLLECT BASE DATA

     与 MON_GET_PKG_CACHE_STMT 表函数收集相同的信息，以及所有可用的活动度量值
   - COLLECT DETAILED DATA

     收集的信息与使用 COLLECT BASE DATA 子句收集的信息相同，并且还包括条目的可执行部分

当您需要调查 SQL 语句某一次的执行情况时，如果条目仍然存在于程序包高速缓存中，那么可以使用 MON_GET_PKG_CACHE_STMT 表函数来比较已高速缓存的条目相对于其他条目的行为。可将已高速缓存的条目的执行度量值、编译环境和详细描述用于进行诊断。

如果已经从程序包高速缓存中写入条目，那么可以使用程序包高速缓存事件监视器来查看那些已经从程序包高速缓存中写入的已高速缓存条目的历史记录。历史记录数据中包含的信息与 MON_GET_PKG_CACHE_STMT 表函数所提供的信息相同。此外，此

事件监视器还提供了该语句的可执行部分。这全部都适用于动态和静态 SQL 语句。

创建程序包高速缓存事件监视器

要创建程序包高速缓存事件监视器并收集程序包高速缓存事件监视器数据，您必须具有 DBADM 或 SQLADM 权限。

程序包高速缓存事件监视器可将其输出写至常规表或无格式事件表。

在创建程序包高速缓存事件监视器之前，请标识要在其中存储事件监视器的输出的表空间。如果您未指定，那么 CREATE EVENT MONITOR 语句将使用缺省表空间。但是，建议的做法是，配置专用表空间来存储与任何事件监视器相关联的输出表。如果要使用无格式事件表，请在页大小至少为 8K 的表空间中创建程序包高速缓存事件监视器，以确保事件数据包含在 UE 表的直接插入 BLOB 列中。如果不直接插入 BLOB 列，那么对无格式事件表读写事件的效率可能不高。

要使用缺省值和最佳实践来设置程序包高速缓存事件监视器，请完成下列步骤:

- 通过发出 CREATE EVENT MONITOR 语句来创建事件监视器。以下示例将尽可能使用缺省值，并且指定将无格式事件表存储在现有表空间 MY_EVMON_TABLESPACE 中:

```
CREATE EVENT MONITOR MY_PKGCACHE_EVMON
  FOR PACKAGE CACHE
   WRITE TO UNFORMATTED EVENT TABLE (IN MY_EVMON_TABLESPACE)
```

启用数据收集功能

要启用数据收集功能，必须使用 SET EVENT MONITOR STATE 语句来激活事件监视器。程序包高速缓存事件监视器不是*被动型*监视器；激活此事件监视器之后，一旦从程序包高速缓存中写入了语句，并且满足在创建程序包高速缓存事件监视器时所设置的过滤条件，它就会自动开始收集数据。

访问程序包高速缓存事件监视器所捕获的事件数据

工作单元事件监视器可将数据写至常规表，也可将二进制格式的数据写至无格式事件 (UE) 表。可使用 SQL 来访问常规表中的数据。

要访问 UE 表中的数据，请使用下列其中一个表函数:

EVMON_FORMAT_UE_TO_XML
将无格式事件表中的数据抽取到 XML 文档中。

EVMON_FORMAT_UE_TO_TABLES
将无格式事件表中的数据抽取到一组关系表中。

如果使用这些表函数的其中一个，那么可通过包括 SELECT 语句作为该函数的其中一个参数来指定要抽取的数据。您可以对 SELECT 语句所提供的选择、排序和其他方面功能进行全面控制。

模式文件 ~/sqllib/misc/DB2EvmonPkgCache.xsd 用来将程序包高速缓存事件监视器报告的期望输出记录在 XML 文档中。此模式文件将引用一个公共监视器模式文件 (DB2MonCommon.xsd)，以避免复制公共内容。

~/sqllib/samples/jdbc/DB2EvmonPkgCache.xsl 文件中提供了 XML 样式表。

使用这些表函数来指定要使用 SELECT 语句抽取的数据。您可以对 SELECT 语句所提供的选择、排序和其他方面功能进行全面控制。

另外，还可以使用 **db2evmonfmt** 命令来执行下列任务:

- 根据下列属性来选择感兴趣的事件: 可执行文件标识、部分类型、估计查询成本、语句程序包高速缓存标识和写入时间。
- 选择是以文本报告形式还是格式化 XML 文档形式来接收输出。
- 通过创建您自己的 XSLT 样式表代替使用 **db2evmonfmt** 命令所提供的 XSLT 样式表，对输出格式进行控制。

例如，以下命令将提供一个具有下列特性的程序包高速缓存报告:

1. 选择过去 24 小时内在 SAMPLE 数据库中发生的程序包高速缓存事件。可从名为 SAMPLE_PKGCACHE_EVENTS 的无格式事件表中获取这些事件记录。
2. 使用 DB2EvmonPkgCache.xsl 样式表来提供带格式文本输出。

```
java db2evmonfmt -d SAMPLE -ue SAMPLE_PKGCACHE_EVENTS -ftext -ss DB2EvmonPkgCache.xsl -hours 24
```

程序包高速缓存事件监视器生成的数据

程序包高速缓存事件监视器生成有关从程序包高速缓存中取出的包的数据。可选择将程序包高速缓存事件监视器的输出写至常规表或无格式事件 (UE) 表。如果数据写至 UE 表，那么必须对该数据执行后处理才能查看该数据。

不管您选择什么输出格式，所有程序包高速缓存事件数据都来自下列三个逻辑组的其中一个:

- pkgcache
- pkgcache_metrics
- pkgcache_stmt_args

如果选择将程序包高速缓存事件数据写至常规表，那么另一个组 (CONTROL) 中的数据用于生成有关事件监视器本身的元数据。

注: 与锁定事件监视器和工作单元事件监视器不同，您不必在创建程序包高速缓存事件监视器后启用程序包高速缓存事件数据的生成；数据收集将在该事件监视器激活后立即开始。

程序包高速缓存事件监视器写至表的信息:

在指定了 WRITE TO TABLE 选项的情况下程序包高速缓存事件监视器写入的信息。

选择 WRITE TO TABLE 作为程序包高速缓存事件监视器的输出类型时，缺省情况下，会生成三个表，每个表包含一个或多个逻辑数据组中的监视元素。

表 39. 程序包高速缓存写至表事件监视器生成的表. 表名通过逻辑数据组（用于填充该表）的名称和 CREATE EVENT MONITOR 语句中给予事件监视器的名称（如下表的表名中的 *evmon-name* 所示）并置派生。

缺省表名	报告的逻辑数据组
PKGCACHE_*evmon-name*	pkgcache
PKGCACHE_METRICS_*evmon-name*	pkgcache_metrics
PKGCACHE_STMT_ARGS_*evmon-name*	第 72 页的『pkgcache_stmt_args 逻辑数据组』

表 39. 程序包高速缓存写至表事件监视器生成的表 (续). 表名通过逻辑数据组（用于填充该表）的名称和 CREATE EVENT MONITOR 语句中给予事件监视器的名称（如下表的表名中的 *evmon-name* 所示）并置派生。

缺省表名	报告的逻辑数据组
CONTROL_*evmon-name*	CONTROL 逻辑组由来自 event_dbheader、event_start 和 event_overflow 逻辑数据组中的一个或多个的所选元素组成。

要将事件监视器的输出限制为发送至特定表，请在 CREATE EVENT MONITOR 或 ALTER EVENT MONITOR 语句中指定要对其生成表的逻辑组的名称。有关详细信息，请参阅这些语句的参考主题。

生成的表

表 40. 对程序包高速缓存事件监视器返回的信息: 缺省表名: PKGCACHE_evmon-name

列名	数据类型	描述
PARTITION_KEY	INTEGER	第 951 页的『partition_key -“分区键”监视元素』
COMP_ENV_DESC	BLOB	comp_env_desc - 编译环境
EFFECTIVE_ISOLATION	CHARACTER (2)	effective_isolation - 有效隔离级别
EVENT_ID	BIGINT	event_id -“事件标识”监视元素
EVENT_TIMESTAMP	TIMESTAMP	event_timestamp -“事件时间戳记”监视元素
EXECUTABLE_ID	VARCHAR (32)	executable_id - 可执行标识
INSERT_TIMESTAMP	TIMESTAMP	insert_timestamp - 插入时间戳记
LAST_METRICS_UPDATE	TIMESTAMP	last_metrics_update - 最近一次更新度量的时间戳记
MAX_COORD_STMT_EXEC_TIME	BIGINT	max_coord_stmt_exec_time - 最长协调程序语句执行时间
MAX_COORD_STMT_EXEC _TIMESTAMP	TIMESTAMP	max_coord_stmt_exec_timestamp - 最大协调语句执行时间戳记
MEMBER	SMALLINT	member - 数据库成员
METRICS	BLOB	
NUM_COORD_EXEC	BIGINT	num_coord_exec - 协调代理程序执行的次数
NUM_COORD_EXEC_WITH _METRICS	BIGINT	num_coord_exec_with_metrics - 协调代理程序执行的次数以及度量
NUM_EXEC_WITH_METRICS	BIGINT	num_exec_with_metrics - 在收集度量值情况下的执行次数
NUM_EXECUTIONS	BIGINT	num_executions - 语句执行次数
PACKAGE_NAME	VARCHAR (128)	package_name - 程序包名
PACKAGE_SCHEMA	VARCHAR (128)	package_schema - 程序包模式
PACKAGE_VERSION_ID	VARCHAR (64)	package_version_id - 程序包版本
PREP_TIME	BIGINT	prep_time - 编译时间
QUERY_COST_ESTIMATE	BIGINT	query_cost_estimate - 查询估算成本

表 40. 对程序包高速缓存事件监视器返回的信息：缺省表名：PKGCACHE_evmon-name（续）

列名	数据类型	描述
QUERY_DATA_TAG_LIST	VARCHAR (32)	query_data_tag_list - 查询数据标记列表
ROUTINE_ID	BIGINT	routine_id - 例程标识
SECTION_ENV	BLOB(0)	section_env - 节环境
SECTION_NUMBER	BIGINT	section_number - 节号
SECTION_TYPE	CHARACTER (1)	section_type - 节类型指示器
STMT_PKG_CACHE_ID	BIGINT	第 1174 页的『stmt_pkgcache_id -“语句程序包高速缓存标识”监视元素』
STMT_TEXT	CLOB	stmt_text - SQL 语句文本
STMT_TYPE_ID	VARCHAR (32)	stmt_type_id - 语句类型标识
TOTAL_STATS_FABRICATION _TIME	BIGINT	total_stats_fabrication_time - 统计信息生成时间总计
TOTAL_STATS_FABRICATIONS	BIGINT	total_stats_fabrications - 统计信息生成总计
TOTAL_SYNC_RUNSTATS	BIGINT	total_sync_runstats - 同步 RUNSTATS 活动总数
TOTAL_SYNC_RUNSTATS_TIME	BIGINT	total_sync_runstats_time - 同步 RUNSTATS 时间总计

表 41. 对程序包高速缓存事件监视器返回的信息：表名：PKGCACHE_METRICS_evmon-name

列名	数据类型	描述
PARTITION_KEY	INTEGER	第 951 页的『partition_key -“分区键”监视元素』
EVENT_ID	BIGINT	event_id -“事件标识”监视元素
EVENT_TIMESTAMP	TIMESTAMP	event_timestamp -“事件时间戳记”监视元素
MEMBER	SMALLINT	member - 数据库成员
WLM_QUEUE_TIME_TOTAL	BIGINT	wlm_queue_time_total - 工作负载管理器队列时间总计
WLM_QUEUE_ASSIGNMENTS _TOTAL	BIGINT	wlm_queue_assignments_total - 工作负载管理器队列分配总计
FCM_TQ_RECV_WAIT_TIME	BIGINT	fcm_tq_recv_wait_time - FCM 表队列接收等待时间
FCM_MESSAGE_RECV_WAIT_TIME	BIGINT	fcm_message_recv_wait_time - 接收 FCM 消息等待时间
FCM_TQ_SEND_WAIT_TIME	BIGINT	fcm_tq_send_wait_time - FCM 表队列发送等待时间
FCM_MESSAGE_SEND_WAIT_TIME	BIGINT	fcm_message_send_wait_time - 发送 FCM 消息等待时间
LOCK_WAIT_TIME	BIGINT	lock_wait_time - 等待锁定时间
LOCK_WAITS	BIGINT	lock_waits - 等待锁定次数
DIRECT_READ_TIME	BIGINT	direct_read_time - 直接读时间
DIRECT_READ_REQS	BIGINT	direct_read_reqs - 直接读请求数
DIRECT_WRITE_TIME	BIGINT	direct_write_time - 直接写时间
DIRECT_WRITE_REQS	BIGINT	direct_write_reqs - 直接写请求数
LOG_BUFFER_WAIT_TIME	BIGINT	log_buffer_wait_time - 日志缓冲区等待时间
NUM_LOG_BUFFER_FULL	BIGINT	num_log_buffer_full - 日志缓冲区变满次数
LOG_DISK_WAIT_TIME	BIGINT	log_disk_wait_time - 日志磁盘等待时间
LOG_DISK_WAITS_TOTAL	BIGINT	log_disk_waits_total - 日志磁盘等待总次数
POOL_WRITE_TIME	BIGINT	pool_write_time - 缓冲池物理写时间总计
POOL_READ_TIME	BIGINT	pool_read_time - 缓冲池物理读时间总计
AUDIT_FILE_WRITE_WAIT_TIME	BIGINT	audit_file_write_wait_time - 审计文件写等待时间

表 41. 对程序包高速缓存事件监视器返回的信息：表名：PKGCACHE_METRICS_evmon-name（续）

列名	数据类型	描述
AUDIT_FILE_WRITES_TOTAL	BIGINT	audit_file_writes_total - 写审计文件总次数
AUDIT_SUBSYSTEM_WAIT_TIME	BIGINT	audit_subsystem_wait_time - 审计子系统等待时间
AUDIT_SUBSYSTEM_WAITS_TOTAL	BIGINT	audit_subsystem_waits_total - 审计子系统等待总次数
DIAGLOG_WRITE_WAIT_TIME	BIGINT	diaglog_write_wait_time - 诊断日志文件写等待时间
DIAGLOG_WRITES_TOTAL	BIGINT	diaglog_writes_total - 写诊断日志文件总次数
FCM_SEND_WAIT_TIME	BIGINT	fcm_send_wait_time - FCM 发送等待时间
FCM_RECV_WAIT_TIME	BIGINT	fcm_recv_wait_time - FCM 接收等待时间
TOTAL_ACT_WAIT_TIME	BIGINT	total_act_wait_time - 活动等待时间总计
TOTAL_SECTION_SORT_PROC_TIME	BIGINT	total_section_sort_proc_time - 节排序处理时间总计
TOTAL_SECTION_SORTS	BIGINT	total_section_sorts - 节排序总次数
TOTAL_SECTION_SORT_TIME	BIGINT	total_section_sort_time - 节排序时间总计
TOTAL_ACT_TIME	BIGINT	total_act_time - 活动时间总计
ROWS_READ	BIGINT	rows_read - 读取行数
ROWS_MODIFIED	BIGINT	rows_modified - 修改的行数
POOL_DATA_L_READS	BIGINT	pool_data_l_reads - 缓冲池数据逻辑读取数
POOL_INDEX_L_READS	BIGINT	pool_index_l_reads - 缓冲池索引逻辑读取数
POOL_TEMP_DATA_L_READS	BIGINT	pool_temp_data_l_reads - 缓冲池临时数据逻辑读取数
POOL_TEMP_INDEX_L_READS	BIGINT	pool_temp_index_l_reads - 缓冲池临时索引逻辑读取数
POOL_XDA_L_READS	BIGINT	pool_xda_l_reads - 缓冲池 XDA 数据逻辑读取数
POOL_TEMP_XDA_L_READS	BIGINT	pool_temp_xda_l_reads - 缓冲池临时 XDA 数据逻辑读取数
TOTAL_CPU_TIME	BIGINT	total_cpu_time - CPU 时间总计
POOL_DATA_P_READS	BIGINT	pool_data_p_reads - 缓冲池数据物理读取数
POOL_TEMP_DATA_P_READS	BIGINT	pool_temp_data_p_reads - 缓冲池临时数据物理读取数
POOL_XDA_P_READS	BIGINT	pool_xda_p_reads - 缓冲池 XDA 数据物理读取数
POOL_TEMP_XDA_P_READS	BIGINT	pool_temp_xda_p_reads - 缓冲池临时 XDA 数据物理读取数
POOL_INDEX_P_READS	BIGINT	pool_index_p_reads - 缓冲池索引物理读取数
POOL_TEMP_INDEX_P_READS	BIGINT	pool_temp_index_p_reads - 缓冲池临时索引物理读取数
POOL_DATA_WRITES	BIGINT	pool_data_writes - 缓冲池数据写次数
POOL_XDA_WRITES	BIGINT	pool_xda_writes - 缓冲池 XDA 数据写次数
POOL_INDEX_WRITES	BIGINT	pool_index_writes - 缓冲池索引写次数
DIRECT_READS	BIGINT	direct_reads - 直接读数据库数目
DIRECT_WRITES	BIGINT	direct_writes - 直接写数据库数目
ROWS_RETURNED	BIGINT	rows_returned - 返回的行数
DEADLOCKS	BIGINT	deadlocks - 检测到的死锁数
LOCK_TIMEOUTS	BIGINT	lock_timeouts - 锁定超时次数
LOCK_ESCALS	BIGINT	lock_escals - 锁定升级次数
FCM_SENDS_TOTAL	BIGINT	fcm_sends_total - FCM 发送总计
FCM_RECVS_TOTAL	BIGINT	fcm_recvs_total - FCM 接收总计
FCM_SEND_VOLUME	BIGINT	fcm_send_volume - FCM 发送量
FCM_RECV_VOLUME	BIGINT	fcm_recv_volume - FCM 接收量
FCM_MESSAGE_SENDS_TOTAL	BIGINT	fcm_message_sends_total - 发送 FCM 消息总数
FCM_MESSAGE_RECVS_TOTAL	BIGINT	fcm_message_recvs_total - 接收 FCM 消息总数
FCM_MESSAGE_SEND_VOLUME	BIGINT	fcm_message_send_volume - 发送 FCM 消息量
FCM_MESSAGE_RECV_VOLUME	BIGINT	fcm_message_recv_volume - 接收 FCM 消息量
FCM_TQ_SENDS_TOTAL	BIGINT	fcm_tq_sends_total - FCM 表队列发送总次数

表 41. 对程序包高速缓存事件监视器返回的信息：表名：PKGCACHE_METRICS_evmon-name（续）

列名	数据类型	描述
FCM_TQ_RECVS_TOTAL	BIGINT	fcm_tq_recvs_total - FCM 表队列接收总量
FCM_TQ_SEND_VOLUME	BIGINT	fcm_tq_send_volume - FCM 表队列发送量
FCM_TQ_RECV_VOLUME	BIGINT	fcm_tq_recv_volume - FCM 表队列接收量
TQ_TOT_SEND_SPILLS	BIGINT	tq_tot_send_spills - 溢出表队列缓冲区总数
POST_THRESHOLD_SORTS	BIGINT	post_threshold_sorts - 超过阈值后的排序数
POST_SHRTHRESHOLD_SORTS	BIGINT	post_shrthreshold_sorts - 共享阈值后排序数
SORT_OVERFLOWS	BIGINT	sort_overflows - 排序溢出数
AUDIT_EVENTS_TOTAL	BIGINT	audit_events_total - 审计事件总数
TOTAL_SORTS	BIGINT	total_sorts - 排序总数
STMT_EXEC_TIME	BIGINT	stmt_exec_time - 语句执行时间
COORD_STMT_EXEC_TIME	BIGINT	coord_stmt_exec_time - 协调代理程序执行语句的时间
TOTAL_ROUTINE_NON_SECT_PROC _TIME	BIGINT	total_routine_non_sect_proc_time - 非部分处理时间
TOTAL_ROUTINE_NON_SECT_TIME	BIGINT	total_routine_non_sect_time - 非部分例程执行时间
TOTAL_SECTION_PROC_TIME	BIGINT	total_section_proc_time - 部分处理时间总计
TOTAL_APP_SECTION_EXECUTIONS	BIGINT	total_app_section_executions - 应用程序执行部分执行的总次数
TOTAL_SECTION_TIME	BIGINT	total_section_time - 部分时间总计
TOTAL_ROUTINE_USER_CODE _PROC_TIME	BIGINT	total_routine_user_code_proc_time - 例程用户代码处理时间总计
TOTAL_ROUTINE_USER_CODE_TIME	BIGINT	total_routine_user_code_time - 例程用户代码时间总计
TOTAL_ROUTINE_TIME	BIGINT	total_routine_time - 例程时间总计
THRESH_VIOLATIONS	BIGINT	thresh_violations - 阈值违例次数
NUM_LW_THRESH_EXCEEDED	BIGINT	num_lw_thresh_exceeded - 超过锁定等待阈值的次数
TOTAL_ROUTINE_INVOCATIONS	BIGINT	total_routine_invocations - 例程调用总计
LOCK_WAIT_TIME_GLOBAL	BIGINT	lock_wait_time_global - 锁定等待时间全局
LOCK_WAITS_GLOBAL	BIGINT	lock_waits_global - 锁定等待全局
RECLAIM_WAIT_TIME	BIGINT	reclaim_wait_time - 回收等待时间
SPACEMAPPAGE_RECLAIM_WAIT _TIME	BIGINT	spacemappage_reclaim_wait_time - 空间映射页回收等待时间
LOCK_TIMEOUTS_GLOBAL	BIGINT	lock_timeouts_global - 锁定超时全局
LOCK_ESCALS_MAXLOCKS	BIGINT	lock_escals_maxlocks - maxlocks 锁定升级数
LOCK_ESCALS_LOCKLIST	BIGINT	lock_escals_locklist - locklist 锁定升级数
LOCK_ESCALS_GLOBAL	BIGINT	lock_escals_global - 全局锁定升级数
CF_WAIT_TIME	BIGINT	cf_wait_time - 集群高速缓存工具等待时间
CF_WAITS	BIGINT	cf_waits - 集群高速缓存工具 DB2 pureScale 服务器等待数
POOL_DATA_GBP_L_READS	BIGINT	pool_data_gbp_l_reads - 组缓冲池数据逻辑读取数
POOL_DATA_GBP_P_READS	BIGINT	pool_data_gbp_p_reads - 组缓冲池数据物理读取数
POOL_DATA_LBP_PAGES_FOUND	BIGINT	pool_data_lbp_pages_found - 本地缓冲池发现的数据页数
POOL_DATA_GBP_INVALID_PAGES	BIGINT	pool_data_gbp_invalid_pages - 组缓冲池无效数据页数
POOL_INDEX_GBP_L_READS	BIGINT	pool_index_gbp_l_reads - 组缓冲池索引逻辑读取数
POOL_INDEX_GBP_P_READS	BIGINT	pool_index_gbp_p_reads - 组缓冲池索引物理读取数
POOL_INDEX_LBP_PAGES_FOUND	BIGINT	pool_index_lbp_pages_found - 发现的本地缓冲池索引页数
POOL_INDEX_GBP_INVALID_PAGES	BIGINT	pool_index_gbp_invalid_pages - 组缓冲池无效索引页数
POOL_XDA_GBP_L_READS	BIGINT	pool_xda_gbp_l_reads - 组缓冲池 XDA 数据逻辑读取请求数
POOL_XDA_GBP_P_READS	BIGINT	pool_xda_gbp_p_reads - 组缓冲池 XDA 数据物理读取请求数

表 41. 对程序包高速缓存事件监视器返回的信息：表名：PKGCACHE_METRICS_evmon-name（续）

列名	数据类型	描述
POOL_XDA_LBP_PAGES_FOUND	BIGINT	pool_xda_lbp_pages_found - 发现的本地缓冲池 XDA 数据页数
POOL_XDA_GBP_INVALID_PAGES	BIGINT	pool_xda_gbp_invalid_pages - 组缓冲池无效 XDA 数据页数
EVMON_WAIT_TIME	BIGINT	evmon_wait_time - 事件监视器等待时间
EVMON_WAITS_TOTAL	BIGINT	evmon_waits_total - 事件监视器总等待次数
TOTAL_EXTENDED_LATCH_WAIT _TIME	BIGINT	total_extended_latch_wait_time - 扩展锁存器等待时间总计
TOTAL_EXTENDED_LATCH_WAITS	BIGINT	total_extended_latch_waits - 扩展锁存器等待总计
TOTAL_DISP_RUN_QUEUE_TIME	BIGINT	total_disp_run_queue_time - 分派器运行队列时间总计
POOL_QUEUED_ASYNC_DATA _REQS	BIGINT	pool_queued_async_data_reqs - 数据预取请求数
POOL_QUEUED_ASYNC_INDEX _REQS	BIGINT	pool_queued_async_index_reqs - 索引预取请求数
POOL_QUEUED_ASYNC_XDA_REQS	BIGINT	pool_queued_async_xda_reqs - XDA 预取请求数
POOL_QUEUED_ASYNC_TEMP _DATA_REQS	BIGINT	pool_queued_async_temp_data_reqs - 临时表空间数据预取请求数
POOL_QUEUED_ASYNC_TEMP _INDEX_REQS	BIGINT	pool_queued_async_temp_index_reqs - 临时表空间索引预取请求数
POOL_QUEUED_ASYNC_TEMP _XDA_REQS	BIGINT	pool_queued_async_temp_xda_reqs - 临时表空间 XDA 数据预取请求数
POOL_QUEUED_ASYNC_OTHER _REQS	BIGINT	pool_queued_async_other_reqs - 非预取请求数
POOL_QUEUED_ASYNC_DATA _PAGES	BIGINT	pool_queued_async_data_pages - 预取请求的数据页数
POOL_QUEUED_ASYNC_INDEX _PAGES	BIGINT	pool_queued_async_index_pages - 预取请求的索引页数
POOL_QUEUED_ASYNC_XDA _PAGES	BIGINT	pool_queued_async_xda_pages - 预取请求的 XDA 页数
POOL_QUEUED_ASYNC_TEMP _DATA_PAGES	BIGINT	pool_queued_async_temp_data_pages - 预取请求的临时表空间数据页数
POOL_QUEUED_ASYNC_TEMP _INDEX_PAGES	BIGINT	pool_queued_async_temp_index_pages - 预取请求的临时表空间索引页数
POOL_QUEUED_ASYNC_TEMP_XDA _PAGES	BIGINT	pool_queued_async_temp_xda_pages - 预取请求的临时表空间 XDA 数据页数
POOL_FAILED_ASYNC_DATA_REQS	BIGINT	pool_failed_async_data_reqs - 失败数据预取请求数
POOL_FAILED_ASYNC_INDEX_REQS	BIGINT	pool_failed_async_index_reqs - 失败的索引预取请求数
POOL_FAILED_ASYNC_XDA_REQS	BIGINT	pool_failed_async_xda_reqs - 失败的 XDA 预取请求数
POOL_FAILED_ASYNC_TEMP_DATA _REQS	BIGINT	pool_failed_async_temp_data_reqs - 临时表空间的失败数据预取请求数
POOL_FAILED_ASYNC_TEMP _INDEX_REQS	BIGINT	pool_failed_async_temp_index_reqs - 临时表空间的失败索引预取请求数
POOL_FAILED_ASYNC_TEMP_XDA _REQS	BIGINT	pool_failed_async_temp_xda_reqs - 临时表空间的失败 XDA 预取请求数
POOL_FAILED_ASYNC_OTHER _REQS	BIGINT	pool_failed_async_other_reqs - 失败的非预取请求数
TOTAL_PEDS	BIGINT	total_peds - 部分提前相异总数
DISABLED_PEDS	BIGINT	第 746 页的『disabled_peds -“已禁用部分提前相异数”监视元素』

表 41. 对程序包高速缓存事件监视器返回的信息: 表名: PKGCACHE_METRICS_evmon-name (续)

列名	数据类型	描述
POST_THRESHOLD_PEDS	BIGINT	post_threshold_peds - 部分提前相异数阈值
TOTAL_PEAS	BIGINT	total_peas - 部分提前聚集总数
POST_THRESHOLD_PEAS	BIGINT	post_threshold_peas - 部分提前聚集阈值
TQ_SORT_HEAP_REQUESTS	BIGINT	tq_sort_heap_requests - 表队列排序堆请求数
TQ_SORT_HEAP_REJECTIONS	BIGINT	tq_sort_heap_rejections - 表队列排序堆拒绝数
PREFETCH_WAIT_TIME	BIGINT	prefetch_wait_time - 等待预取的时间
PREFETCH_WAITS	BIGINT	prefetch_waits - 预取程序等待计数
POOL_DATA_GBP_INDEP_PAGES _FOUND_IN_LBP	BIGINT	第 975 页的『pool_data_gbp_indep_pages_found _in_lbp -"本地缓冲池中发现的独立于组缓冲池的数据页数"监视元素』
POOL_INDEX_GBP_INDEP_PAGES _FOUND_IN_LBP	BIGINT	第 1005 页的『pool_index_gbp_indep_pages _found_in_lbp -"本地缓冲池中发现的独立于组缓冲池的索引页数"监视元素』
POOL_XDA_GBP_INDEP_PAGES _FOUND_IN_LBP	BIGINT	第 1059 页的『pool_xda_gbp_indep_pages _found_in_lbp -"本地缓冲池中发现的独立于组缓冲池的 XDA 页数"监视元素』

表 42. 对程序包高速缓存事件监视器返回的信息: 缺省表名: PKGCACHE_STMT_ARGS_evmon-name

列名	数据类型	描述
PARTITION_KEY	INTEGER	第 951 页的『partition_key -"分区键"监视元素』
EVENT_ID	BIGINT	event_id -"事件标识"监视元素
EVENT_TIMESTAMP	TIMESTAMP	event_timestamp -"事件时间戳记"监视元素
MEMBER	SMALLINT	member - 数据库成员
STMT_VALUE_DATA	CLOB	stmt_value_data - 值数据
STMT_VALUE_INDEX	INTEGER	stmt_value_index - 值索引
STMT_VALUE_ISNULL	INTEGER	stmt_value_isnull - 包含空值
STMT_VALUE_ISREOPT	INTEGER	stmt_value_isreopt - 用于语句重新优化的变量
STMT_VALUE_TYPE	CHARACTER(16)	stmt_value_type - 值类型

以下数据类型的条目记录在以上表中，但是，参数的实际值未记录在 STMT_VALUE_DATA 元素中:

- BLOB
- CLOB
- REF
- BOOLEAN
- 结构化数据类型
- DATALINK
- LONG VARGRAPHIC
- LONG VARCHAR
- XML 类型
- DBCLOB
- ARRAY 类型
- ROW 类型
- ROWID
- CURSOR 变量

表 43. 对程序包高速缓存事件监视器返回的信息: 缺省表名: CONTROL_evmon-name

列名	数据类型	描述
PARTITION_KEY	INTEGER	第 951 页的『partition_key -“分区键”监视元素』
EVENT_MONITOR_NAME	VARCHAR(128)	event_monitor_name - 事件监视器名称
MESSAGE	VARCHAR(128)	message - 控制表消息
MESSAGE_TIME	TIMESTAMP	message_time - 时间戳记控制表消息
PARTITION_NUMBER	SMALLINT	partition_number - 分区号

EVMON_FORMAT_UE_TO_TABLES 为程序包高速缓存事件监视器写至关系表的信息:

EVMON_FORMAT_UE_TO_TABLES 表函数为程序包高速缓存事件监视器写入的信息。DB2EvmonPkgCache.xsd 文件中也记录了此信息。

表 44. 为程序包高速缓存事件监视器返回的信息: 表名: PKGCACHE_EVENT

列名	数据类型	描述
XMLID	VARCHAR(256) NOT NULL	第 1346 页的『xmlid -“XML 标识”监视元素』
EVENT_ID	BIGINT NOT NULL	event_id -“事件标识”监视元素
EVENT_TYPE	VARCHAR(128) NOT NULL	event_type -“事件类型”监视元素
EVENT_TIMESTAMP	TIMESTAMP NOT NULL	event_timestamp -“事件时间戳记”监视元素
MEMBER	SMALLINT NOT NULL	member - 数据库成员
SECTION_TYPE	CHAR(1)	section_type - 节类型指示器
INSERT_TIMESTAMP	TIMESTAMP	insert_timestamp - 插入时间戳记
EXECUTABLE_ID	VARCHAR(32) FOR BIT DATA	executable_id - 可执行标识
PACKAGE_SCHEMA	VARCHAR(128)	package_schema - 程序包模式
PACKAGE_NAME	VARCHAR(128)	package_name - 程序包名
PACKAGE_VERSION_ID	VARCHAR(64)	package_version_id - 程序包版本
SECTION_NUMBER	BIGINT	section_number - 节号
EFFECTIVE_ISOLATION	CHAR(2)	effective_isolation - 有效隔离级别
NUM_EXECUTIONS	BIGINT	num_executions - 语句执行次数
NUM_EXEC_WITH_METRICS	BIGINT	num_exec_with_metrics - 在收集度量值情况下的执行次数
PREP_TIME	BIGINT	prep_time - 编译时间
LAST_METRICS_UPDATE	TIMESTAMP	last_metrics_update - 最近一次更新度量的时间戳记
NUM_COORD_EXEC	BIGINT	num_coord_exec - 协调代理程序执行的次数
NUM_COORD_EXEC_WITH_METRICS	BIGINT	num_coord_exec_with_metrics - 协调代理程序执行的次数以及度量
STMT_TYPE_ID	VARCHAR(32)	stmt_type_id - 语句类型标识
QUERY_COST_ESTIMATE	BIGINT	query_cost_estimate - 查询估算成本

表 44. 为程序包高速缓存事件监视器返回的信息：表名：PKGCACHE_EVENT (续)

列名	数据类型	描述
STMT_PKG_CACHE_ID	BIGINT	第 1174 页的『stmt_pkgcache_id -“语句程序包高速缓存标识”监视元素』
STMT_TEXT	CLOB(2M)	stmt_text - SQL 语句文本
COMP_ENV_DESC	BLOB(10K)	comp_env_desc - 编译环境
METRICS	BLOB(1M)	包含与度量值相关的监视元素的 XML 文档。此文档中的度量值与本主题后面出现的 PKGCACHE_METRICS 表描述的度量值相同。有关更多信息，请参阅第 12 页的『在 XML 文档中返回监视数据的接口』。
SECTION_ENV	BLOB(150M)	section_env - 节环境
ROUTINE_ID	BIGINT	routine_id - 例程标识
QUERY_DATA_TAG_LIST	VARCHAR(32)	query_data_tag_list - 查询数据标记列表
TOTAL_STATS_FABRICATION_TIME	BIGINT	total_stats_fabrication_time - 统计信息生成时间总计
TOTAL_STATS_FABRICATIONS	BIGINT	total_stats_fabrications - 统计信息生成总计
TOTAL_SYNC_RUNSTATS_TIME	BIGINT	total_sync_runstats_time - 同步 RUNSTATS 时间总计
TOTAL_SYNC_RUNSTATS	BIGINT	total_sync_runstats - 同步 RUNSTATS 活动总数
MAX_COORD_STMT_EXEC _TIMESTAMP	TIMESTAMP	max_coord_stmt_exec_timestamp - 最大协调语句执行 时间戳记
MAX_COORD_STMT_EXEC_TIME	BIGINT	max_coord_stmt_exec_time - 最长协调程序语句执行时间

表 45. 为程序包高速缓存事件监视器返回的信息：表名：PKGCACHE_METRICS。此表中的度量值与 PKGCACHE_EVENT 表中 METRICS 监视元素返回的度量值相同

列名	数据类型	描述
XMLID	VARCHAR(256) NOT NULL	第 1346 页的『xmlid -“XML 标识”监视元素』
TOTAL_ACT_TIME	BIGINT	total_act_time - 活动时间总计
TOTAL_ACT_WAIT_TIME	BIGINT	total_act_wait_time - 活动等待时间总计
TOTAL_CPU_TIME	BIGINT	total_cpu_time - CPU 时间总计
POOL_READ_TIME	BIGINT	pool_read_time - 缓冲池物理读时间总计
POOL_WRITE_TIME	BIGINT	pool_write_time - 缓冲池物理写时间总计
DIRECT_READ_TIME	BIGINT	direct_read_time - 直接读时间
DIRECT_WRITE_TIME	BIGINT	direct_write_time - 直接写时间
LOCK_WAIT_TIME	BIGINT	lock_wait_time - 等待锁定时间
TOTAL_SECTION_SORT_TIME	BIGINT	total_section_sort_time - 节排序时间总计

表 45. 为程序包高速缓存事件监视器返回的信息：表名：PKGCACHE_METRICS。此表中的度量值与 PKGCACHE_EVENT 表中 METRICS 监视元素返回的度量值相同 (续)

列名	数据类型	描述
TOTAL_SECTION_SORT_PROC _TIME	BIGINT	total_section_sort_proc_time - 节排序处理时间总计
TOTAL_SECTION_SORTS	BIGINT	total_section_sorts - 节排序总次数
LOCK_ESCALS	BIGINT	lock_escals - 锁定升级次数
LOCK_WAITS	BIGINT	lock_waits - 等待锁定次数
ROWS_MODIFIED	BIGINT	rows_modified - 修改的行数
ROWS_READ	BIGINT	rows_read - 读取行数
ROWS_RETURNED	BIGINT	rows_returned - 返回的行数
DIRECT_READS	BIGINT	direct_reads - 直接读数据库数目
DIRECT_READ_REQS	BIGINT	direct_read_reqs - 直接读请求数
DIRECT_WRITES	BIGINT	direct_writes - 直接写数据库数目
DIRECT_WRITE_REQS	BIGINT	direct_write_reqs - 直接写请求数
POOL_DATA_L_READS	BIGINT	pool_data_l_reads - 缓冲池数据逻辑读取数
POOL_TEMP_DATA_L_READS	BIGINT	pool_temp_data_l_reads - 缓冲池临时数据逻辑读取数
POOL_XDA_L_READS	BIGINT	pool_xda_l_reads - 缓冲池 XDA 数据逻辑读取数
POOL_TEMP_XDA_L_READS	BIGINT	pool_temp_xda_l_reads - 缓冲池临时 XDA 数据逻辑读取数
POOL_INDEX_L_READS	BIGINT	pool_index_l_reads - 缓冲池索引逻辑读取数
POOL_TEMP_INDEX_L_READS	BIGINT	pool_temp_index_l_reads - 缓冲池临时索引逻辑读取数
POOL_DATA_P_READS	BIGINT	pool_data_p_reads - 缓冲池数据物理读取数
POOL_TEMP_DATA_P_READS	BIGINT	pool_temp_data_p_reads - 缓冲池临时数据物理读取数
POOL_XDA_P_READS	BIGINT	pool_xda_p_reads - 缓冲池 XDA 数据物理读取数
POOL_TEMP_XDA_P_READS	BIGINT	pool_temp_xda_p_reads - 缓冲池临时 XDA 数据物理读取数
POOL_INDEX_P_READS	BIGINT	pool_index_p_reads - 缓冲池索引物理读取数
POOL_TEMP_INDEX_P_READS	BIGINT	pool_temp_index_p_reads - 缓冲池临时索引物理读取数
POOL_DATA_WRITES	BIGINT	pool_data_writes - 缓冲池数据写次数
POOL_XDA_WRITES	BIGINT	pool_xda_writes - 缓冲池 XDA 数据写次数
POOL_INDEX_WRITES	BIGINT	pool_index_writes - 缓冲池索引写次数
TOTAL_SORTS	BIGINT	total_sorts - 排序总数
POST_THRESHOLD_SORTS	BIGINT	post_threshold_sorts - 超过阈值后的排序数
POST_SHRTHRESHOLD_SORTS	BIGINT	post_shrthreshold_sorts - 共享阈值后排序数

表 45. 为程序包高速缓存事件监视器返回的信息：表名：PKGCACHE_METRICS。此表中的度量值与 PKGCACHE_EVENT 表中 METRICS 监视元素返回的度量值相同 (续)

列名	数据类型	描述
SORT_OVERFLOWS	BIGINT	sort_overflows - 排序溢出数
WLM_QUEUE_TIME_TOTAL	BIGINT	wlm_queue_time_total - 工作负载管理器队列时间总计
WLM_QUEUE_ASSIGNMENTS _TOTAL	BIGINT	wlm_queue_assignments_total - 工作负载管理器队列分配总计
DEADLOCKS	BIGINT	deadlocks - 检测到的死锁数
FCM_RECV_VOLUME	BIGINT	fcm_recv_volume - FCM 接收量
FCM_RECVS_TOTAL	BIGINT	fcm_recvs_total - FCM 接收总计
FCM_SEND_VOLUME	BIGINT	fcm_send_volume - FCM 发送量
FCM_SENDS_TOTAL	BIGINT	fcm_sends_total - FCM 发送总计
FCM_RECV_WAIT_TIME	BIGINT	fcm_recv_wait_time - FCM 接收等待时间
FCM_SEND_WAIT_TIME	BIGINT	fcm_send_wait_time - FCM 发送等待时间
LOCK_TIMEOUTS	BIGINT	lock_timeouts - 锁定超时次数
LOG_BUFFER_WAIT_TIME	BIGINT	log_buffer_wait_time - 日志缓冲区等待时间
NUM_LOG_BUFFER_FULL	BIGINT	num_log_buffer_full - 日志缓冲区变满次数
LOG_DISK_WAIT_TIME	BIGINT	log_disk_wait_time - 日志磁盘等待时间
LOG_DISK_WAITS_TOTAL	BIGINT	log_disk_waits_total - 日志磁盘等待总次数
TOTAL_ROUTINE_TIME	BIGINT	total_routine_time - 例程时间总计
TOTAL_ROUTINE_INVOCATIONS	BIGINT	total_routine_invocations - 例程调用总计
COORD_STMT_EXEC_TIME	BIGINT	coord_stmt_exec_time - 协调代理程序执行语句的时间
STMT_EXEC_TIME	BIGINT	stmt_exec_time - 语句执行时间
TOTAL_SECTION_TIME	BIGINT	total_section_time - 部分时间总计
TOTAL_SECTION_PROC_TIME	BIGINT	total_section_proc_time - 部分处理时间总计
TOTAL_ROUTINE_NON_SECT _TIME	BIGINT	total_routine_non_sect_time - 非部分例程执行时间
TOTAL_ROUTINE_NON_SECT _PROC_TIME	BIGINT	total_routine_non_sect_proc_time - 非部分处理时间
FCM_TQ_RECV_WAIT_TIME	BIGINT	fcm_tq_recv_wait_time - FCM 表队列接收等待时间
FCM_MESSAGE_RECV_WAIT_TIME	BIGINT	fcm_message_recv_wait_time - 接收 FCM 消息等待时间
FCM_TQ_SEND_WAIT_TIME	BIGINT	fcm_tq_send_wait_time - FCM 表队列发送等待时间
FCM_MESSAGE_SEND_WAIT_TIME	BIGINT	fcm_message_send_wait_time - 发送 FCM 消息等待时间
AUDIT_FILE_WRITE_WAIT_TIME	BIGINT	audit_file_write_wait_time - 审计文件写等待时间
AUDIT_FILE_WRITES_TOTAL	BIGINT	audit_file_writes_total - 写审计文件总次数
AUDIT_SUBSYSTEM_WAIT_TIME	BIGINT	audit_subsystem_wait_time - 审计子系统等待时间

表 45. 为程序包高速缓存事件监视器返回的信息：表名：PKGCACHE_METRICS。此表中的度量值与 PKGCACHE_EVENT 表中 METRICS 监视元素返回的度量值相同 (续)

列名	数据类型	描述
AUDIT_SUBSYSTEM_WAITS_TOTAL	BIGINT	audit_subsystem_waits_total - 审计子系统等待总次数
DIAGLOG_WRITE_WAIT_TIME	BIGINT	diaglog_write_wait_time - 诊断日志文件写等待时间
DIAGLOG_WRITES_TOTAL	BIGINT	diaglog_writes_total - 写诊断日志文件总次数
FCM_MESSAGE_SENDS_TOTAL	BIGINT	fcm_message_sends_total - 发送 FCM 消息总数
FCM_MESSAGE_RECVS_TOTAL	BIGINT	fcm_message_recvs_total - 接收 FCM 消息总数
FCM_MESSAGE_SEND_VOLUME	BIGINT	fcm_message_send_volume - 发送 FCM 消息量
FCM_MESSAGE_RECV_VOLUME	BIGINT	fcm_message_recv_volume - 接收 FCM 消息量
FCM_TQ_SENDS_TOTAL	BIGINT	fcm_tq_sends_total - FCM 表队列发送总次数
FCM_TQ_RECVS_TOTAL	BIGINT	fcm_tq_recvs_total - FCM 表队列接收总量
FCM_TQ_SEND_VOLUME	BIGINT	fcm_tq_send_volume - FCM 表队列发送量
FCM_TQ_RECV_VOLUME	BIGINT	fcm_tq_recv_volume - FCM 表队列接收量
TQ_TOT_SEND_SPILLS	BIGINT	tq_tot_send_spills - 溢出表队列缓冲区总数
AUDIT_EVENTS_TOTAL	BIGINT	audit_events_total - 审计事件总数
TOTAL_APP_SECTION _EXECUTIONS	BIGINT	total_app_section_executions - 应用程序执行部分执行的总次数
TOTAL_ROUTINE_USER_CODE _PROC_TIME	BIGINT	total_routine_user_code_proc_time - 例程用户代码处理时间总计
TOTAL_ROUTINE_USER_CODE _TIME	BIGINT	total_routine_user_code_time - 例程用户代码时间总计
THRESH_VIOLATIONS	BIGINT	thresh_violations - 阈值违例次数
NUM_LW_THRESH_EXCEEDED	BIGINT	num_lw_thresh_exceeded - 超过锁定等待阈值的次数
LOCK_WAITS_GLOBAL	BIGINT	lock_waits_global - 锁定等待全局
LOCK_WAIT_TIME_GLOBAL	BIGINT	lock_wait_time_global - 锁定等待时间全局
LOCK_TIMEOUTS_GLOBAL	BIGINT	lock_timeouts_global - 锁定超时全局
LOCK_ESCALS_MAXLOCKS	BIGINT	lock_escals_maxlocks - maxlocks 锁定升级数
LOCK_ESCALS_LOCKLIST	BIGINT	lock_escals_locklist - locklist 锁定升级数
LOCK_ESCALS_GLOBAL	BIGINT	lock_escals_global - 全局锁定升级数
RECLAIM_WAIT_TIME	BIGINT	reclaim_wait_time - 回收等待时间
SPACEMAPPAGE_RECLAIM_WAIT _TIME	BIGINT	spacemappage_reclaim_wait _time - 空间映射页回收等待时间
CF_WAITS	BIGINT	cf_waits - 集群高速缓存工具 DB2 pureScale 服务器等待数

表 45. 为程序包高速缓存事件监视器返回的信息：表名：PKGCACHE_METRICS。此表中的度量值与 PKGCACHE_EVENT 表中 METRICS 监视元素返回的度量值相同 (续)

列名	数据类型	描述
CF_WAIT_TIME	BIGINT	cf_wait_time - 集群高速缓存工具等待时间
POOL_DATA_GBP_L_READS	BIGINT	pool_data_gbp_l_reads - 组缓冲池数据逻辑读取数
POOL_DATA_GBP_P_READS	BIGINT	pool_data_gbp_p_reads - 组缓冲池数据物理读取数
POOL_DATA_LBP_PAGES_FOUND	BIGINT	pool_data_lbp_pages_found - 本地缓冲池发现的数据页数
POOL_DATA_GBP_INVALID_PAGES	BIGINT	pool_data_gbp_invalid_pages - 组缓冲池无效数据页数
POOL_INDEX_GBP_L_READS	BIGINT	pool_index_gbp_l_reads - 组缓冲池索引逻辑读取数
POOL_INDEX_GBP_P_READS	BIGINT	pool_index_gbp_p_reads - 组缓冲池索引物理读取数
POOL_INDEX_LBP_PAGES_FOUND	BIGINT	pool_index_lbp_pages_found - 发现的本地缓冲池索引页数
POOL_INDEX_GBP_INVALID _PAGES	BIGINT	pool_index_gbp_invalid_pages - 组缓冲池无效索引页数
POOL_XDA_GBP_L_READS	BIGINT	pool_xda_gbp_l_reads - 组缓冲池 XDA 数据逻辑读取请求数
POOL_XDA_GBP_P_READS	BIGINT	pool_xda_gbp_p_reads - 组缓冲池 XDA 数据物理读取请求数
POOL_XDA_LBP_PAGES_FOUND	BIGINT	pool_xda_lbp_pages_found - 发现的本地缓冲池 XDA 数据页数
POOL_XDA_GBP_INVALID_PAGES	BIGINT	pool_xda_gbp_invalid_pages - 组缓冲池无效 XDA 数据页数
EVMON_WAIT_TIME	BIGINT	evmon_wait_time - 事件监视器等待时间
EVMON_WAITS_TOTAL	BIGINT	evmon_waits_total - 事件监视器总等待次数
TOTAL_EXTENDED_LATCH_WAIT _TIME	BIGINT	total_extended_latch_wait_time - 扩展锁存器等待时间总计
TOTAL_EXTENDED_LATCH_WAITS	BIGINT	total_extended_latch_waits - 扩展锁存器等待总计
TOTAL_DISP_RUN_QUEUE_TIME	BIGINT	total_disp_run_queue_time - 分派器运行队列时间总计
TOTAL_PEDS	BIGINT	total_peds - 部分提前相异总数
DISABLED_PEDS	BIGINT	第 746 页的『disabled_peds -"已禁用部分提前相异数"监视元素』
POST_THRESHOLD_PEDS	BIGINT	post_threshold_peds - 部分提前相异数阈值
TOTAL_PEAS	BIGINT	total_peas - 部分提前聚集总数
POST_THRESHOLD_PEAS	BIGINT	post_threshold_peas - 部分提前聚集阈值
TQ_SORT_HEAP_REQUESTS	BIGINT	tq_sort_heap_requests - 表队列排序堆请求数

表 45. 为程序包高速缓存事件监视器返回的信息：表名：PKGCACHE_METRICS。此表中的度量值与 PKGCACHE_EVENT 表中 METRICS 监视元素返回的度量值相同 (续)

列名	数据类型	描述
TQ_SORT_HEAP_REJECTIONS	BIGINT	tq_sort_heap_rejections - 表队列排序堆拒绝数
POOL_QUEUED_ASYNC_DATA_REQS	BIGINT	pool_queued_async_data_reqs - 数据预取请求数
POOL_QUEUED_ASYNC_INDEX_REQS	BIGINT	pool_queued_async_index_reqs - 索引预取请求数
POOL_QUEUED_ASYNC_XDA_REQS	BIGINT	pool_queued_async_xda_reqs - XDA 预取请求数
POOL_QUEUED_ASYNC_TEMP_DATA_REQS	BIGINT	pool_queued_async_temp_data_reqs - 临时表空间数据预取请求数
POOL_QUEUED_ASYNC_TEMP_INDEX_REQS	BIGINT	pool_queued_async_temp_index_reqs - 临时表空间索引预取请求数
POOL_QUEUED_ASYNC_TEMP_XDA_REQS	BIGINT	pool_queued_async_temp_xda_reqs - 临时表空间 XDA 数据预取请求数
POOL_QUEUED_ASYNC_OTHER_REQS	BIGINT	pool_queued_async_other_reqs - 非预取请求数
POOL_QUEUED_ASYNC_DATA_PAGES	BIGINT	pool_queued_async_data_pages - 预取请求的数据页数
POOL_QUEUED_ASYNC_INDEX_PAGES	BIGINT	pool_queued_async_index_pages - 预取请求的索引页数
POOL_QUEUED_ASYNC_XDA_PAGES	BIGINT	pool_queued_async_xda_pages - 预取请求的 XDA 页数
POOL_QUEUED_ASYNC_TEMP_DATA_PAGES	BIGINT	pool_queued_async_temp_data_pages - 预取请求的临时表空间数据页数
POOL_QUEUED_ASYNC_TEMP_INDEX_PAGES	BIGINT	pool_queued_async_temp_index_pages - 预取请求的临时表空间索引页数
POOL_QUEUED_ASYNC_TEMP_XDA_PAGES	BIGINT	pool_queued_async_temp_xda_pages - 预取请求的临时表空间 XDA 数据页数
POOL_FAILED_ASYNC_DATA_REQS	BIGINT	pool_failed_async_data_reqs - 失败数据预取请求数
POOL_FAILED_ASYNC_INDEX_REQS	BIGINT	pool_failed_async_index_reqs - 失败的索引预取请求数
POOL_FAILED_ASYNC_XDA_REQS	BIGINT	pool_failed_async_xda_reqs - 失败的 XDA 预取请求数
POOL_FAILED_ASYNC_TEMP_DATA_REQS	BIGINT	pool_failed_async_temp_data_reqs - 临时表空间的失败数据预取请求数
POOL_FAILED_ASYNC_TEMP_INDEX_REQS	BIGINT	pool_failed_async_temp_index_reqs - 临时表空间的失败索引预取请求数
POOL_FAILED_ASYNC_TEMP_XDA_REQS	BIGINT	pool_failed_async_temp_xda_reqs - 临时表空间的失败 XDA 预取请求数
POOL_FAILED_ASYNC_OTHER_REQS	BIGINT	pool_failed_async_other_reqs - 失败的非预取请求数
PREFETCH_WAIT_TIME	BIGINT	prefetch_wait_time - 等待预取的时间

表 45. 为程序包高速缓存事件监视器返回的信息：表名：PKGCACHE_METRICS。此表中的度量值与 PKGCACHE_EVENT 表中 METRICS 监视元素返回的度量值相同 (续)

列名	数据类型	描述
PREFETCH_WAITS	BIGINT	prefetch_waits - 预取程序等待计数
POOL_DATA_GBP_INDEP_PAGES _FOUND_IN_LBP	BIGINT	第 975 页的『pool_data_gbp_indep_pages_found _in_lbp -“本地缓冲池中发现的独立于组缓冲池的数据页数”监视元素』
POOL_INDEX_GBP_INDEP_PAGES _FOUND_IN_LBP	BIGINT	第 1005 页的『pool_index_gbp_indep_pages _found_in_lbp -“本地缓冲池中发现的独立于组缓冲池的索引页数”监视元素』
POOL_XDA_GBP_INDEP_PAGES _FOUND_IN_LBP	BIGINT	第 1059 页的『pool_xda_gbp_indep_pages _found_in_lbp -“本地缓冲池中发现的独立于组缓冲池的 XDA 页数”监视元素』

表 46. 为程序包高速缓存事件监视器返回的信息：表名：PKGCACHE_STMT_ARGS。

列名	数据类型	描述
XMLID	VARCHAR(256) NOT NULL	第 1346 页的『xmlid -“XML 标识”监视元素』
STMT_VALUE_INDEX	INTEGER NOT NULL	stmt_value_index - 值索引
STMT_VALUE_ISREOPT	INTEGER	stmt_value_isreopt - 用于语句重新优化的变量
STMT_VALUE_ISNULL	INTEGER	stmt_value_isnull - 包含空值
STMT_VALUE_TYPE	CHAR(16)	stmt_value_type - 值类型
STMT_VALUE_DATA	CLOB (32K)	stmt_value_data - 值数据

以下数据类型的条目记录在以上表中，但是，参数的实际值未记录在 STMT_VALUE_DATA 元素中:

- BLOB
- CLOB
- REF
- BOOLEAN
- 结构化数据类型
- DATALINK
- LONG VARGRAPHIC
- LONG VARCHAR
- XML 类型
- DBCLOB
- ARRAY 类型
- ROW 类型
- ROWID
- CURSOR 变量

在语句中，最先出现的输入自变量在记录时从 1 开始，后续每个输入自变量依次加 1。此表中可记录的输入参数的数目仅受 UE 事件监视器用于捕获事件信息的 BLOB 文档的大小的上限所限制。实际上，捕获的输入自变量数目可能导致达到此限制。

为程序包高速缓存事件监视器写入 XML 的信息:

EVMON_FORMAT_UE_TO_XML 表函数为程序包高速缓存事件监视器写入的信息。DB2EvmonPkgCache.xsd 文件中也记录了此信息。

db2_pkgcache_event

用于详细描述程序包高速缓存事件的主模式。

元素内容: (『section_type』, 『insert_timestamp』, 第 214 页的『executable_id』, 第 214 页的『package_schema』, 第 214 页的『package_name』, 第 214 页的『package_version_id』, 第 214 页的『section_number』 {zero or one times (?)}, 第 214 页的『effective_isolation』, 第 215 页的『num_executions』, 第 215 页的『num_exec_with_metrics』, 第 215 页的『prep_time』, 第 215 页的『last_metrics_update』, 第 216 页的『num_coord_exec』, 第 216 页的『num_coord_exec_with_metrics』, 第 216 页的『stmt_type_id』, 第 216 页的『query_cost_estimate』, 第 217 页的『stmt_pkg_cache_id』, 第 217 页的『stmt_text』, 第 217 页的『comp_env_desc』, 第 217 页的『section_env』, 第 217 页的『activity_metrics』, 第 217 页的『routine_id』, 第 218 页的『query_data_tag_list』, 第 218 页的『total_stats_fabrication_time』, 第 218 页的『total_stats_fabrications』, 第 218 页的『total_sync_runstats_time』, 第 218 页的『total_sync_runstats』, 第 219 页的『max_coord_stmt_exec_timestamp』 {zero or one times (?)}, 第 219 页的『max_coord_stmt_exec_time_arg』 {zero or more (*)}, 第 219 页的『max_coord_stmt_exec_time』, ANY content (skip) {zero or more (*)})

属性:

QName	类型	修订时间	缺省值	用法	注释
id	xs:long			必需	
类型				必需	
时间戳记	xs:dateTime			必需	
成员				必需	
release	xs:long			必需	
任何名称空间中的任何属性					

section_type

所处理的 SQL 语句的类型。可能的值: D:Dynamic 或 S:Static。有关更多详细信息, 请参阅监视元素 第 1129 页的『section_type -"节类型指示器"监视元素』。

包含者: 『db2_pkgcache_event』

insert_timestamp

将变体或部分插入到高速缓存中的时间。有关更多详细信息, 请参阅监视元素 第 820 页的『insert_timestamp -"插入时间戳记"监视元素』。

包含者: 『db2_pkgcache_event』

元素内容:

类型	构面
xs:dateTime	

executable_id

在数据服务器上生成的二进制标记，用于唯一地标识已执行的 SQL 语句部分。有关更多详细信息，请参阅监视元素 第 762 页的『executable_id -“可执行文件标识”监视元素』。

包含者: 第 213 页的『db2_pkgcache_event』

package_schema

与 SQL 语句相关联的程序包的模式名。有关更多详细信息，请参阅监视元素 第 942 页的『package_schema -“程序包模式”监视元素』。

包含者: 第 213 页的『db2_pkgcache_event』

package_name

当前正在执行的 SQL 语句所在程序包的名称。有关更多详细信息，请参阅监视元素 第 941 页的『package_name -“程序包名”监视元素』。

包含者: 第 213 页的『db2_pkgcache_event』

package_version_id

程序包版本指定当前正在执行的 SQL 语句所在程序包的版本标识。有关更多详细信息，请参阅监视元素 第 943 页的『package_version_id -“程序包版本”监视元素』。

包含者: 第 213 页的『db2_pkgcache_event』

section_number

程序包中用于当前正在处理或最新处理的 SQL 语句的内部节号。有关更多详细信息，请参阅监视元素 第 1128 页的『section_number -“节号”监视元素』。

包含者: 第 213 页的『db2_pkgcache_event』

元素内容:

类型	构面
xs:long	

effective_isolation

运行 SQL 语句时作用于该语句的隔离值。有关更多详细信息，请参阅监视元素 第 750 页的『effective_isolation -“有效隔离级别”监视元素』。

包含者: 第 213 页的『db2_pkgcache_event』

属性:

QName	类型	修订时间	缺省值	用法	注释
id	xs:long			可选	

num_executions

已执行 SQL 语句的次数。有关更多详细信息，请参阅监视元素 第 911 页的『num_executions -“语句执行次数”监视元素』。

包含者: 第 213 页的『db2_pkgcache_event』

元素内容:

类型	构面
xs:long	

num_exec_with_metrics

已执行此 SQL 语句并且收集了度量值的次数。有关更多详细信息，请参阅监视元素 第 912 页的『num_exec_with_metrics -“在收集度量值情况下的执行次数”监视元素』。

包含者: 第 213 页的『db2_pkgcache_event』

元素内容:

类型	构面
xs:long	

prep_time

在活动为 SQL 语句的情况下编译 SQL 语句所需的时间（以毫秒计）。有关更多详细信息，请参阅监视元素 第 1083 页的『prep_time -“编译时间”监视元素』。

包含者: 第 213 页的『db2_pkgcache_event』

元素内容:

类型	构面
xs:long	

last_metrics_update

用于反映上一次更新此高速缓存条目的时间度量值的时间戳记。有关更多详细信息，请参阅监视元素 第 835 页的『last_metrics_update -“最近一次更新度量的时间戳记”监视元素』。

包含者: 第 213 页的『db2_pkgcache_event』

元素内容:

类型	构面
xs:dateTime	

num_coord_exec

协调代理程序执行此部分的次数。有关更多详细信息，请参阅监视元素 第 910 页的『num_coord_exec -“协调代理程序执行的次数”监视元素』。

包含者: 第 213 页的『db2_pkgcache_event』

元素内容:

类型	构面
xs:long	

num_coord_exec_with_metrics

协调代理程序执行此部分的次数以及所捕获的监视度量值。有关更多详细信息，请参阅监视元素 第 910 页的『num_coord_exec_with_metrics -“协调代理程序执行的次数以及度量”监视元素』。

包含者: 第 213 页的『db2_pkgcache_event』

元素内容:

类型	构面
xs:long	

stmt_type_id

语句类型标识。有关更多详细信息，请参阅监视元素 第 1180 页的『stmt_type_id -“语句类型标识”监视元素』。

包含者: 第 213 页的『db2_pkgcache_event』

属性:

QName	类型	修订时间	缺省值	用法	注释
id	xs:long			可选	

query_cost_estimate

由 SQL 编译器确定的查询估算成本。有关更多详细信息，请参阅监视元素 第 1092 页的『query_cost_estimate -“查询估算成本”监视元素』。

包含者: 第 213 页的『db2_pkgcache_event』

元素内容:

类型	构面
xs:long	

stmt_pkg_cache_id

包含者: 第 213 页的『db2_pkgcache_event』

元素内容:

类型	构面
xs:long	

stmt_text

SQL 语句的文本。有关更多详细信息，请参阅监视元素 第 1178 页的『stmt_text -“SQL 语句文本”监视元素』。

包含者: 第 213 页的『db2_pkgcache_event』

comp_env_desc

第 669 页的『comp_env_desc -“编译环境”监视元素』

包含者: 第 213 页的『db2_pkgcache_event』

section_env

包含 SQL 语句的部分的 BLOB。有关更多详细信息，请参阅监视元素 第 1128 页的『section_env -“节环境”监视元素』。

包含者: 第 213 页的『db2_pkgcache_event』

activity_metrics

此高速缓存条目的活动度量值。

包含者: 第 213 页的『db2_pkgcache_event』

routine_id

对于 CALL 语句，此元素存储与所调用存储过程相关联的例程标识。有关更多详细信息，请参阅监视元素 第 1115 页的『routine_id -“例程标识”监视元素』。

包含者: 第 213 页的『db2_pkgcache_event』

元素内容:

类型	构面
xs:long	

query_data_tag_list

此高速缓存条目的数据标记列表。有关更多详细信息，请参阅监视元素 第 1093 页的『query_data_tag_list -“估算的查询数据标记列表”监视元素』。

包含者: 第 213 页的『db2_pkgcache_event』

total_stats_fabrication_time

通过收集实时统计信息来生成统计信息时所花的总时间（以毫秒计）。有关更多详细信息，请参阅监视元素 第 1294 页的『total_stats_fabrication_time -“统计信息生成时间总计”监视元素』。

包含者: 第 213 页的『db2_pkgcache_event』

元素内容:

类型	构面
xs:long	

total_stats_fabrications

通过在查询编译期间由实时统计信息收集执行的统计信息生成的总次数。有关更多详细信息，请参阅监视元素 第 1295 页的『total_stats_fabrications -“统计信息生成总计”监视元素』。

包含者: 第 213 页的『db2_pkgcache_event』

元素内容:

类型	构面
xs:long	

total_sync_runstats_time

实时统计信息收集触发的同步 RUNSTATS 活动所花的总时间（以毫秒计）。有关更多详细信息，请参阅监视元素 第 1296 页的『total_sync_runstats_time -“同步 RUNSTATS 时间总计”监视元素』。

包含者: 第 213 页的『db2_pkgcache_event』

元素内容:

类型	构面
xs:long	

total_sync_runstats

实时统计信息收集触发同步 RUNSTATS 活动的总数。有关更多详细信息，请参阅监视元素 第 1299 页的『total_sync_runstats -“同步 RUNSTATS 活动总数”监视元素』。

包含者: 第 213 页的『db2_pkgcache_event』

元素内容:

类型	构面
xs:long	

max_coord_stmt_exec_timestamp

有关更多详细信息，请参阅监视元素 第 884 页的『max_coord_stmt_exec_timestamp -"最大协调程序语句执行时间戳"监视元素』。

包含者: 第 213 页的『db2_pkgcache_event』

元素内容:

类型	构面
xs:dateTime	

max_coord_stmt_exec_time_arg

描述生成 max_coord_stmt_exec_time 的语句的输入变量。

包含者: 第 213 页的『db2_pkgcache_event』

元素内容: (『stmt_value_index』, 第 220 页的『stmt_value_isreopt』, 第 220 页的『stmt_value_isnull』, 第 220 页的『stmt_value_type』, 第 220 页的『stmt_value_data』, ANY content (skip) {zero or more (*)})

max_coord_stmt_exec_time

有关更多详细信息，请参阅监视元素 第 881 页的『max_coord_stmt_exec_time -"最长协调程序语句执行时间"监视元素』。

包含者: 第 213 页的『db2_pkgcache_event』

元素内容:

类型	构面
xs:long	

stmt_value_index

此元素表示 SQL 语句中使用的输入参数标记或主变量的位置。有关更多详细信息，请参阅监视元素 第 1182 页的『stmt_value_index -"值索引"』。

包含者: 『max_coord_stmt_exec_time_arg』

stmt_value_isreopt

此元素指示该变量在语句重新优化期间是否已被使用。有关更多详细信息，请参阅监视元素 第 1183 页的『stmt_value_isreopt -“用于语句重新优化的变量”监视元素』。

包含者: 第 219 页的『max_coord_stmt_exec_time_arg』

属性:

QName	类型	修订时间	缺省值	用法	注释
id	xs:int			必需	

stmt_value_isnull

此元素指示与 SQL 语句相关联的数据值是否为空值。有关更多详细信息，请参阅监视元素 第 1182 页的『stmt_value_isnull -“包含空值”监视元素』。

包含者: 第 219 页的『max_coord_stmt_exec_time_arg』

属性:

QName	类型	修订时间	缺省值	用法	注释
id	xs:int			必需	

stmt_value_type

第 1184 页的『stmt_value_type -“值类型”监视元素』

包含者: 第 219 页的『max_coord_stmt_exec_time_arg』

stmt_value_data

此元素包含与 SQL 语句相关联的数据值的字符串表示。有关更多详细信息，请参阅监视元素 第 1181 页的『stmt_value_data -“值数据”』。

包含者: 第 219 页的『max_coord_stmt_exec_time_arg』

收集程序包高速缓存事件数据和生成报告

可以使用程序包高速缓存事件监视器来收集有关从数据库程序包高速缓存中写入的语句条目的数据。在无格式事件表中收集程序包高速缓存事件数据之后，遵循本任务中的指令来获取文本报告。

开始之前

要收集程序包高速缓存事件监视器数据，您必须具有 DBADM 或 SQLADM 权限。

关于此任务

程序包高速缓存事件监视器将收集有关此程序包高速缓存中先前所具有内容的相关历史记录信息，以帮助解决与 SQL 语句相关的查询性能和问题确定问题。例如，下面就是程序包高速缓存事件监视器从数据库程序包高速缓存中收集的一些信息:

- 可执行文件标识 (EXECUTABLE_ID)

- 估计查询成本 (QUERY_COST_ESTIMATE)
- 从程序包高速缓存中写入条目的时间（事件时间戳记）

本任务提供有关收集程序包高速缓存事件数据的指示信息。

限制

如果您不具备 DBADM 或 SQLADM 权限，那么将不能查看输入数据值。

过程

要收集关于程序包高速缓存事件的详细信息，请执行下列步骤:

1. 使用 CREATE EVENT MONITOR FOR PACKAGE CACHE 语句来创建名为 cachestmtevmon 的程序包高速缓存事件监视器，如以下示例中所示:

   ```
   CREATE EVENT MONITOR cachestmtevmon FOR PACKAGE CACHE
      WRITE TO UNFORMATTED EVENT TABLE
   ```

2. 通过运行以下语句来激活名为 cachestmtevmon 的程序包高速缓存事件监视器:

   ```
   SET EVENT MONITOR cachestmtevmon STATE 1
   ```

3. 与锁定事件监视器和工作单元事件监视器不同，程序包高速缓存事件监视器会在激活事件监视器之后自动开始收集数据。
4. 连接到数据库。
5. 运行您要为其收集事件监视器信息的应用程序、工作负载或 SQL 语句。
6. 如果您希望关闭程序包高速缓存数据收集，请通过运行以下命令来取消激活事件监视器:

   ```
   SET EVENT MONITOR cachestmtevmon STATE 0
   ```

7. 使用 XML 解析器工具 **db2evmonfmt** 来生成基于无格式事件表中收集的事件数据的纯文本报告从而获取程序包高速缓存事件报告，例如:

   ```
   java db2evmonfmt -d db_name -ue table_name -ftext -u user_id -p password
   ```

8. 分析此报告。

示例

对于程序包高速缓存事件监视器所收集的无格式事件表中的数据，下面是使用基于 Java 的报告工具 **db2evmonfmt** 对这些数据进行转换之后所获得的报告示例:

```
------------------------------------------------------
Event ID        : 1
Event Type            : PKGCACHEBASE
Event Timestamp       : 2009-11-06-12.32.06.442020
Member                : 0
Release               : 9070100
------------------------------------------------------

Package Cache Details
---------------------
Section Type                          : D
Insert Timestamp                      : 2009-11-06-12.28.04.246930
Executable ID                         : 0100000000000000010000000000000000000000000200200911061228042469 32
Package Schema                        :
Package Name                          :
Package Version ID                    :
Section Number                        :
Effective Isolation                   : CS
Number Of Executions                  : 1
Number Of Executions With Metrics     : 1
Prep Time                             : 9
Last Metrics Update                   : 2009-11-06-12.28.07.905942
Executions By Coordinator             : 1
Executions By Coordinator With Metrics : 1
Statement Type                        : DDL, (not Set Constraints)
```

```
Query Cost Estimate                    : 1
Statement Package Cache ID             : 1151051235329
Statement Text                         : create event monitor cachestmtevmon for package cache write to unformatted event table
Compilation Environment                : 47454E5F434D504C01000000E0000008001000000000000000000000000000010000000400000000010000
                                         0000000020000000400000008010000000000003000000080000001001000000000B0FA040000000100000018
                                         010000B804B0FA0500000001000000200100008040000060000000100000028010000B80400000700000010
                                         0000030010000B8040000080000000100000038010000B8040000090000000500000040010000B80400000A00
                                         0000080000004801000B80400000B0000000B0000005001000B8040000100000000A0000006001000B8040
                                         0001300000004000000700100000000000F0000000400000078010000000000001000000000000000500000
                                         0000000000000000000000000300000000000000030000000000000004E0000000000000004E0000000000000
                                         04E0000000000000312020202000000020204444444444440000000000000000000C000000000020091106
                                         1228040000000000000000000000000000000000000100000000000000
Section Environment                    :

Metrics
-------------------
WLM_QUEUE_TIME_TOTAL                : 0
WLM_QUEUE_ASSIGNMENTS_TOTAL         : 0
FCM_TQ_RECV_WAIT_TIME               : 0
FCM_MESSAGE_RECV_WAIT_TIME          : 0
FCM_TQ_SEND_WAIT_TIME               : 0
FCM_MESSAGE_SEND_WAIT_TIME          : 0
LOCK_WAIT_TIME                      : 0
LOCK_WAITS                          : 0
DIRECT_READ_TIME                    : 0
DIRECT_READ_REQS                    : 0
DIRECT_WRITE_TIME                   : 3
DIRECT_WRITE_REQS                   : 1
LOG_BUFFER_WAIT_TIME                : 0
NUM_LOG_BUFFER_FULL                 : 0
LOG_DISK_WAIT_TIME                  : 0
LOG_DISK_WAITS_TOTAL                : 0
POOL_WRITE_TIME                     : 0
POOL_READ_TIME                      : 33
AUDIT_FILE_WRITE_WAIT_TIME          : 0
AUDIT_FILE_WRITES_TOTAL             : 0
AUDIT_SUBSYSTEM_WAIT_TIME           : 0
AUDIT_SUBSYSTEM_WAITS_TOTAL         : 0
DIAGLOG_WRITE_WAIT_TIME             : 0
DIAGLOG_WRITES_TOTAL                : 0
FCM_SEND_WAIT_TIME                  : 0
FCM_RECV_WAIT_TIME                  : 0
TOTAL_ACT_WAIT_TIME                 : 36
TOTAL_SECTION_SORT_PROC_TIME        : 0
TOTAL_SECTION_SORTS                 : 0
TOTAL_SECTION_SORT_TIME             : 0
TOTAL_ACT_TIME                      : 37
TOTAL_ROUTINE_TIME                  : 0
STMT_EXEC_TIME                      : 3658
COORD_STMT_EXEC_TIME                : 3658
TOTAL_ROUTINE_NON_SECTION_PROC_TIME : 0
TOTAL_ROUTINE_NON_SECTION_TIME      : 0
TOTAL_SECTION_PROC_TIME             : 1
TOTAL_SECTION_TIME                  : 37
TOTAL_ROUTINE_USER_CODE_PROC_TIME   : 0
TOTAL_ROUTINE_USER_CODE_TIME        : 0
ROWS_READ                           : 19
ROWS_MODIFIED                       : 3
POOL_DATA_L_READS                   : 42
POOL_INDEX_L_READS                  : 83
POOL_TEMP_DATA_L_READS              : 0
POOL_TEMP_INDEX_L_READS             : 0
POOL_XDA_L_READS                    : 0
POOL_TEMP_XDA_L_READS               : 0
TOTAL_CPU_TIME                      : 2243
POOL_DATA_P_READS                   : 13
POOL_TEMP_DATA_P_READS              : 0
POOL_XDA_P_READS                    : 0
POOL_TEMP_XDA_P_READS               : 0
POOL_INDEX_P_READS                  : 33
POOL_TEMP_INDEX_P_READS             : 0
POOL_DATA_WRITES                    : 0
POOL_XDA_WRITES                     : 0
POOL_INDEX_WRITES                   : 0
DIRECT_READS                        : 0
DIRECT_WRITES                       : 2
ROWS_RETURNED                       : 0
DEADLOCKS                           : 0
LOCK_TIMEOUTS                       : 0
LOCK_ESCALS                         : 0
FCM_SENDS_TOTAL                     : 0
FCM_RECVS_TOTAL                     : 0
FCM_SEND_VOLUME                     : 0
FCM_RECV_VOLUME                     : 0
FCM_MESSAGE_SENDS_TOTAL             : 0
FCM_MESSAGE_RECVS_TOTAL             : 0
FCM_MESSAGE_SEND_VOLUME             : 0
FCM_MESSAGE_RECV_VOLUME             : 0
FCM_TQ_SENDS_TOTAL                  : 0
FCM_TQ_RECVS_TOTAL                  : 0
```

```
FCM_TQ_SEND_VOLUME                 : 0
FCM_TQ_RECV_VOLUME                 : 0
TQ_TOT_SEND_SPILLS                 : 0
POST_THRESHOLD_SORTS               : 0
POST_SHRTHRESHOLD_SORTS            : 0
SORT_OVERFLOWS                     : 0
AUDIT_EVENTS_TOTAL                 : 0
TOTAL_SORTS                        : 0
THRESH_VIOLATIONS                  : 0
NUM_LW_THRESH_EXCEEDED             : 0
TOTAL_ROUTINE_INVOCATIONS          : 0
```

使用程序包高速缓存信息确定用于进行性能调整的候选语句:

可以将程序包高速缓存事件监视器与内存中度量值配合使用以确定程序包高速缓存中运行成本较高的语句。一旦知道运行花费时间较长的语句后，就可对其进行性能调整。

开始之前

CREATE EVENT MONITOR 语句需要页大小至少为 8K 的表空间来存储事件监视器生成的未格式化事件 (UE) 表。除非在 CREATE EVENT MONITOR 语句中显式命名了表空间，否则使用数据库的缺省表空间。

关于此任务

此任务向您显示如何检查在两个时间点内系统上执行的所有工作，从而查找成本最高的语句（就 CPU 总时间方面而言）。将程序包高速缓存事件监视器与内存中监视器元素中反映的信息（如 MON_GET_PKG_CACHE_STMT 或 MON_GET_PKG_CACHE_STMT_DETAILS 表函数返回的信息）配合使用将很有用，因为您可同时查看高速缓存中语句以及从高速缓存逐出的语句。在确定高成本的语句后，然后可对这些语句执行性能调整。

注: 您可选择一些确定高运行成本的语句时要使用的监视元素。在此示例中，使用 CPU 时间 (第 1249 页的『total_cpu_time -“CPU 时间总计”监视元素』)。此度量显示所消耗的实际 CPU 资源；它不反映诸如锁定等待时间或语句执行期间消耗的其他时间之类的内容。您可能改为选择使用语句执行时间 (第 1168 页的『stmt_exec_time -“语句执行时间”监视元素』)，其包含段中所有代理程序所耗用的时间并包括等待时间等等。您还可从程序包高速缓存事件监视器返回的众多其他时间耗用元素中进行选择。有关您可从中选择的监视元素的更多信息，请参阅第 205 页的『EVMON_FORMAT_UE_TO_TABLES 为程序包高速缓存事件监视器写至关系表的信息』或第 213 页的『为程序包高速缓存事件监视器写入 XML 的信息』。

限制

在这一特定示例中，被分析的语句长度限制为 3000 个字符。这种限制是由于语句中使用了 GROUP BY 子句，而 GROUP BY 子句无法与 LOB 值（如 **stmt_text** 监视元素）配合使用。

过程

1. 创建一个程序包高速缓存事件监视器以在语句从程序包高速缓存中被除去（逐出）时将其捕获。 例如，要创建一个名为 EXPENSIVESTMTS 的事件监视器，您可使用以下 SQL:

   ```
   CREATE EVENT MONITOR EXPENSIVESTMTS FOR PACKAGE CACHE WRITE TO UNFORMATTED EVENT TABLE
   ```

此语句创建一个程序包高速缓存事件监视器，用于写入到与数据库的缺省表空间中的事件监视器 EXPENSIVESTMTS 同名的 UE 表中。可使用 TABLE *table-name* 子句覆盖 UE 表的缺省名称。还可使用 **IN** *tablespace-name* 子句覆盖用于 UE 表的表空间。

缺省情况下，程序包高速缓存事件监视器将捕获从程序包高速缓存中逐出的所有语句。要限制所收集的信息量，可将用于限制收集的信息的某些选项指定为 CREATE EVENT MONITOR 语句的一部分。有关更多信息，请参阅 CREATE EVENT MONITOR (程序包高速缓存) 语句的文档。

2. 接着，激活事件监视器:

```
SET EVENT MONITOR EXPENSIVESTMTS STATE 1
```

注: 缺省情况下，事件监视器在数据库激活时自动启动，因为缺省情况下应用 AUTOSTART 选项。但是，由于此事件监视器是在已经活动的数据库中创建的，因此您必须使用 **SET EVENT MONITOR** 命令手动启动此事件监视器。

3. 连接到数据库，并运行您有兴趣为其执行性能分析的任意语句、工作负载或应用程序。您可收集所需要的任意数量信息。但是，在您具有定期运行的应用程序或工作负载的情况下，这种类型的性能调整具有最佳效果；否则您为先前执行的语句进行的调整对于将来运行的语句可能没有任何影响。
4. 完成数据收集后，停用事件监视器:

```
SET EVENT MONITOR EXPENSIVESTMTS STATE 0
```

5. 使用 EVMON_FORMAT_UE_TO_TABLES 过程从事件监视器所填充的 UE 表中提取数据。

```
CALL EVMON_FORMAT_UE_TO_TABLES ('PKGCACHE', NULL, NULL, NULL, NULL, NULL,
 NULL, -1, 'SELECT * FROM EXPENSIVESTMTS')
```

此过程检查由事件监视器生成的 UE 表 TRACKSTMTS。此过程从 UE 表中选择所有记录，并且在这些记录中，通过程序包高速缓存事件监视器所收集的数据创建两个关系表:

- PKGCACHE_EVENT
- PCKCACHE_METRICS

第一个表包含最常用的监视元素以及与捕获的每个事件相关联的度量值。第二个表包含每个事件的详细度量值。

注: PKGCACHE_METRICS 列中的值还可在 PKGCACHE_EVENT 表的 METRICS 列中包含的 XML 文档中找到。为了更加方便以及面向列的访问，这些值在 PKGCACHE_METRICS 表中提供。

6. 对事件监视器中的输出进行查询以确定哪些语句的运行耗用时间最长。在此示例中，CPU 总时间 (第 1249 页的『total_cpu_time -“CPU 时间总计”监视元素』) 是用于确定总体开销的时间耗用监视元素:

```
WITH STMTS AS
     (
1    SELECT SUM(TOTAL_CPU_TIME) AS TOTAL_CPU_TIME, EXECUTABLE_ID, VARCHAR(STMT_TEXT, 3000) AS STMT_TEXT
     FROM TABLE(MON_GET_PKG_CACHE_STMT(NULL,NULL,NULL,-2)) AS T
     GROUP BY EXECUTABLE_ID, VARCHAR(STMT_TEXT, 3000)
     UNION ALL
2    SELECT SUM(TOTAL_CPU_TIME) AS TOTAL_CPU_TIME, EXECUTABLE_ID, VARCHAR(STMT_TEXT, 3000) AS STMT_TEXT
     FROM PKGCACHE_EVENT E, PKGCACHE_METRICS M WHERE E.XMLID = M.XMLID
     GROUP BY EXECUTABLE_ID, VARCHAR(STMT_TEXT, 3000)
     )
```

```
SELECT SUM(TOTAL_CPU_TIME) AS TOTAL_CPU_TIME, STMT_TEXT, EXECUTABLE_ID
FROM STMTS
GROUP BY EXECUTABLE_ID, STMT_TEXT
ORDER BY TOTAL_EXEC_TIME DESCFETCH FIRST 10 ROWS ONLY;
```

在上述示例中，将同时检索从 MON_GET_PKG_CACHE_STMT 表函数（请参见 **1** ）以及程序包高速缓存事件监视器（请参见 **2** ）返回的数据。通过同时查看这两个数据集，您可查看仍在程序包高速缓存中的语句的数据以及从程序包高速缓存逐出的语句的数据。这样做可确保您在计算高运行成本语句时，考虑了在两个时间点之间运行的所有语句。 上述查询将返回下列结果:

注: 为了便于打印，已减小了组成样本输出的字体大小。从本主题的联机版本（DB2 信息中心中的『使用程序包高速缓存信息确定用于进行性能调整的候选语句』）中可能更加易于阅读此输出。

```
TOTAL_CPU_TIME       STMT_TEXT                     EXECUTABLE_ID
-------------------- ----------------------------- --------------------------------------------------------------------------------
              656250 CALL EVMON_FORMAT_UE_TO_TABLES x'0100000000000000070000000000000000000000002002010120712575922100 0'
SQL0445W  Value "WITH STMTS AS ( SELECT SUM(TOTAL_CPU_TIME) AS TOTAL_CPU_TIME" has been truncated.  SQLSTATE=01004
              500000 CALL XSR_COMPLETE(?,?,NULL,1)  x'0100000000000000160000000000000000000000002002010120712580111200 4'
              156250 CALL XSR_ADDSCHEMADOC(?,?,?,?, x'0100000000000000090000000000000000000000002002010120712575987700 0'
              156250 CREATE INDEX PKGCACHE_EVENT_IN x'0100000000000000120000000000000000000000002002010120712580056500 3'
               93750 CALL XSR_REGISTER(?,?,?,?, NUL x'0100000000000000080000000000000000000000002002010120712575964300 0'
               93750 CALL XDB_DECOMP_XML_FROM_QUERY x'0100000000000000180000000000000000000000002002010120712580186200 1'
               78125 CREATE INDEX PKGCACHE_METRICS_ x'0100000000000000140000000000000000000000002002010120712580092400 0'
               46875 CREATE EVENT MONITOR EXPENSIVE x'0100000000000000010000000000000000000000002002010120712575829900 0'
               46875 SET EVENT MONITOR EXPENSIVESTM x'0100000000000000050000000000000000000000002002010120712575876800 1'
               46875 CALL SYSPROC.SYSINSTALLOBJECTS x'0100000000000000240000000000000000000000002002010120712593628600 2'

  10 record(s) selected with 1 warning messages printed.
```

注: 为了便于表示，已截断了 STMT_TEXT 列。

下一步做什么

使用步骤 第 224 页的 6 中显示的查询的输出来确定要调整的语句。

使用程序包高速缓存信息和 db2advis 来寻找提高性能的机会:

DB2 设计顾问程序可分析 SQL 语句以提供有关如何提高数据库性能的建议。可使用程序包高速缓存中的语句（包括由程序包高速缓存事件监视器捕获的语句）作为设计顾问程序的输入，从而确定对于某个给定工作负载（甚至是在两个时间点内运行的所有语句）可进行哪些更改来提高性能。

开始之前

- CREATE EVENT MONITOR 语句需要页大小至少为 8K 的表空间来存储事件监视器生成的未格式化事件 (UE) 表。除非在 CREATE EVENT MONITOR 语句中显式命名了表空间，否则使用数据库的缺省表空间。
- 您必须已创建了设计顾问程序所必需的说明表。

关于此任务

此任务显示如何使用程序包高速缓存事件监视器跟踪在两个时间点系统上执行的所有工作，然后使用 **db2advis** 命令分析在该期间内运行的高成本语句。db2advis 命令的输出会根据程序包高速缓存事件监视器活动时运行的语句，向您建议可对数据库进行的调整或更改以提高其性能。如果有疑问的语句已不在程序包高速缓存中，那么使用程序包高速缓存事件监视器捕获这些语句便很有用。

限制

在这一特定示例中，被分析的语句长度限制为 3000 个字符。这种限制是由于语句中使用了 GROUP BY 子句，而 GROUP BY 子句无法与 LOB 值（如 **stmt_text** 监视元素）配合使用。

过程

1. 创建一个程序包高速缓存事件监视器以在语句从程序包高速缓存中被除去（逐出）时将其捕获。 例如，要创建一个名为 TRACKSTMTS 的事件监视器，您可使用以下 SQL:

   ```
   CREATE EVENT MONITOR TRACKSTMTS FOR PACKAGE CACHE WRITE TO UNFORMATTED EVENT TABLE
   ```

   此语句创建一个程序包高速缓存事件监视器，用于写入到与事件监视器 TRACKSTMTS 同名的 UE 表中。

2. 接着，激活事件监视器:

   ```
   SET EVENT MONITOR TRACKSTMTS STATE 1
   ```

3. 连接到数据库，并运行您有兴趣为其执行性能分析的任意语句、工作负载或应用程序。您可收集所需要的任意数量信息。但是，在您具有定期运行的应用程序或工作负载的情况下，这种类型的性能调整具有最佳效果；否则您为先前执行的语句进行的调整对于将来运行的语句可能没有任何影响。

4. 完成数据收集后，停用事件监视器:

   ```
   SET EVENT MONITOR TRACKSTMTS STATE 0
   ```

5. 使用 EVMON_FORMAT_UE_TO_TABLES 过程从事件监视器所填充的 UE 表中提取数据。

   ```
   CALL EVMON_FORMAT_UE_TO_TABLES
       ('PKGCACHE', NULL, NULL, NULL, NULL, NULL, NULL, -1,
        'SELECT * FROM TRACKSTMTS')
   ```

   此过程通过程序包高速缓存事件监视器所收集的数据创建两个关系表:

   - PKGCACHE_EVENT
   - PCKCACHE_METRICS

   第一个表包含最常用的监视元素以及与捕获的每个事件相关联的度量值。第二个表包含每个事件的详细度量值。

   注: PKGCACHE_METRICS 列中的值还可在 PKGCACHE_EVENT 表的 METRICS 列中包含的 XML 文档中找到。为了更加方便以及面向列的访问，这些值在 PKGCACHE_METRICS 表中提供。

6. 对事件监视器中的输出进行查询以确定哪些语句的运行耗用时间最长。在此示例中，语句执行时间 (第 1168 页的『stmt_exec_time -“语句执行时间”监视元素』) 是用于确定总体开销的时间耗用监视元素。此监视元素会在所有数据库分区之间累加。

   提示: 将查询中的输出保存到某个文本文件。您将在下一步骤中使用此文件。

```
WITH STMTS AS
(
  SELECT SUM(TOTAL_STMT_EXEC_TIME)/SUM(TOTAL_NUM_COORD_EXEC_WITH_METRICS) AS AVG_TIME_PER_EXEC,
         STMT_TEXT, SUM(NUM_EXECUTIONS) AS NUM_EXECUTIONS, STMT_TYPE_ID
  FROM (
        (
         SELECT    SUM(STMT_EXEC_TIME) AS TOTAL_STMT_EXEC_TIME,
                   SUM(NUM_COORD_EXEC_WITH_METRICS) AS TOTAL_NUM_COORD_EXEC_WITH_METRICS,
                   SUM(NUM_COORD_EXEC) AS NUM_EXECUTIONS,
                   VARCHAR(STMT_TEXT, 3000) AS STMT_TEXT,
```

```
                STMT_TYPE_ID         FROM      PKGCACHE_EVENT AS E, PKGCACHE_METRICS AS M
         WHERE     E.XMLID = M.XMLID
          AND      NUM_COORD_EXEC_WITH_METRICS > 0
         GROUP BY  VARCHAR(STMT_TEXT, 3000),STMT_TYPE_ID
         ORDER BY TOTAL_NUM_COORD_EXEC_WITH_METRICS DESC
         FETCH FIRST 50 ROWS ONLY
        )
        UNION ALL
        (
         SELECT    SUM(STMT_EXEC_TIME) AS TOTAL_STMT_EXEC_TIME,
                   SUM(NUM_COORD_EXEC_WITH_METRICS) AS TOTAL_NUM_COORD_EXEC_WITH_METRICS,
                   SUM(NUM_COORD_EXEC) AS NUM_EXECUTIONS,
                   VARCHAR(STMT_TEXT, 3000) AS STMT_TEXT,
                   STMT_TYPE_ID         FROM TABLE(MON_GET_PKG_CACHE_STMT ( NULL, NULL, NULL, -2)) AS T
         WHERE     NUM_COORD_EXEC_WITH_METRICS > 0
         GROUP BY  VARCHAR(STMT_TEXT, 3000),STMT_TYPE_ID
         ORDER BY TOTAL_NUM_COORD_EXEC_WITH_METRICS DESC
         FETCH FIRST 50 ROWS ONLY
        )
       ) AS Q_UA
  GROUP BY STMT_TEXT, STMT_TYPE_ID
)
SELECT      '--# SET FREQUENCY ' || NUM_EXECUTIONS || X'0A' || STMT_TEXT || ';'
FROM        STMTS  WHERE STMT_TYPE_ID LIKE 'DML, Select%' OR STMT_TYPE_ID LIKE 'DML, Insert%' 1
ORDER BY    AVG_TIME_PER_EXEC DESC
FETCH FIRST 50 ROWS ONLY;
```

在上述示例语句中，将同时检索来自程序包高速缓存事件监视器的数据以及来自 MON_GET_PKG_CACHE_STMT 表函数的内存中信息。通过同时查看这两个数据集，您可查看从程序包高速缓存逐出的语句的数据，以及仍在程序包高速缓存中的语句的数据。这样做可确保您在计算高运行成本语句时，您还包含了尚未从高速缓存中逐出的语句。在每种情况下，根据语句运行的次数，查询均从活动的程序包高速缓存以及程序包高速缓存事件监视器中检索前 50 个语句。然后，根据语句运行的平均时间长度，从这些语句中选择前 50 个 SELECT 或 INSERT 语句 **1** 。

注: 您可选择一些确定高运行成本的语句时要使用的监视元素。在此示例中，使用语句执行时间。此度量包含由所有成员和代理程序执行此部分时在执行期间耗用的时间量，并且包含诸如等待时间之类的时间。您可能改为选择使用仅报告 CPU 处理语句所耗用的时间的 CPU 时间 (第 1249 页的『total_cpu_time -“CPU 时间总计”监视元素』)。您还可从程序包高速缓存事件监视器返回的众多其他时间耗用元素中进行选择。有关您可从中选择的监视元素的更多信息，请参阅第 205 页的『EVMON_FORMAT_UE_TO_TABLES 为程序包高速缓存事件监视器写至关系表的信息』或第 213 页的『为程序包高速缓存事件监视器写入 XML 的信息』。

此外，查询还以设计顾问程序用于其分析的 --# SET FREQUENCY 格式展现输出。 上述查询将返回类似以下的结果:

```
--------------------------------------------------------------------------------------------------------------------------------
--# SET FREQUENCY 1
WITH STMTS AS ( SELECT SUM(TOTAL_STMT_EXEC_TIME)/SUM(TOTAL_NUM_COORD_EXEC_WITH_METRICS) AS AVG_TIME_PER_EXEC, STMT
--# SET FREQUENCY 2
WITH STMTS AS ( SELECT SUM(TOTAL_CPU_TIME) AS TOTAL_CPU_TIME, EXECUTABLE_ID, VARCHAR(STMT_TEXT, 3000) AS STMT_TEXT
--# SET FREQUENCY 1055
SELECT POLICY FROM SYSTOOLS.POLICY WHERE MED='DB2CommonMED' AND DECISION='NOP' AND NAME='CommonPolicy';
--# SET FREQUENCY 99
SELECT CREATOR, NAME, CTIME FROM SYSIBM.SYSTABLES WHERE TYPE='T' OR TYPE='S' OR TYPE='N' WITH UR;
--# SET FREQUENCY 1
UPDATE SYSTOOLS.HMON_ATM_INFO SET STATS_LOCK = 'N', REORG_LOCK = 'N';
--# SET FREQUENCY 1
UPDATE SYSTOOLS.HMON_ATM_INFO AS ATM SET STATS_FLAG = 'N', REORG_FLAG = 'N' WHERE (ATM.SCHEMA, ATM.NAME) IN   (SEL
--# SET FREQUENCY 1
SELECT POLICY FROM SYSTOOLS.POLICY WHERE MED='DB2TableMaintenanceMED' AND DECISION='TableRunstatsDecision' AND NAM
--# SET FREQUENCY 83
WITH JTAB(JSCHEMA,JNAME) AS (VALUES(TABLE_SCHEMA(CAST(? AS varchar(128)), CAST(? AS varchar(128))), TABLE_NAME (CA
--# SET FREQUENCY 122
WITH VTYPED (NAME, SCHEMA) AS (VALUES(TABLE_NAME (CAST(? AS varchar(128)), CAST(? AS varchar(128))), TABLE_SCHEMA(
--# SET FREQUENCY 1210
SELECT COLNAME, TYPENAME FROM  SYSCAT.COLUMNS WHERE TABNAME='POLICY' AND TABSCHEMA='SYSTOOLS';
--# SET FREQUENCY 105
SELECT TABNAME FROM SYSCAT.TABLES  WHERE TABNAME='HMON_ATM_INFO' AND  TABSCHEMA='SYSTOOLS';
--# SET FREQUENCY 104
```

```
DELETE FROM SYSTOOLS.HMON_ATM_INFO AS ATM WHERE NOT EXISTS ( SELECT * FROM SYSIBM.SYSTABLES AS IBM WHERE ATM.NAME
--# SET FREQUENCY 1118
VALUES(SUBSTR(:H00003      ,:H00014,:H00015 ))         INTO :H00009:H00017    ;
--# SET FREQUENCY 274
INSERT INTO "ASRISK"."PKGCACHE_EVENT"("EVENT_ID","XMLID","EVENT_TYPE","EVENT_TIMESTAMP","MEMBER","SECTION_TYPE","I
--# SET FREQUENCY 1
SELECT IBM.TID, IBM.FID FROM SYSIBM.SYSTABLES AS IBM, SYSTOOLS.HMON_ATM_INFO AS ATM WHERE ATM.STATS_FLAG <> 'Y' AN
--# SET FREQUENCY 115
VALUES(SUBSTR(CAST(? AS CLOB(162)),CAST(? AS INTEGER),CAST(? AS INTEGER)));
--# SET FREQUENCY 8227

⋮

-# SET FREQUENCY 532
SELECT TBNAME, TBCREATOR FROM "ASRISK  ".SYSINDEXES WHERE NAME = 'INDCOLUMNS01' AND CREATOR = 'SYSIBM  ';
--# SET FREQUENCY 105
SELECT TABNAME FROM SYSCAT.TABLES  WHERE TABNAME='HMON_COLLECTION' AND  TABSCHEMA='SYSTOOLS';
--# SET FREQUENCY 4091
SELECT STATS_LOCK, REORG_LOCK FROM SYSTOOLS.HMON_ATM_INFO WHERE SCHEMA = ? AND NAME = ? AND CREATE_TIME = ? FOR UP
--# SET FREQUENCY 17100
SELECT CREATE_TIME FROM SYSTOOLS.HMON_ATM_INFO WHERE SCHEMA = ? AND NAME = ? FOR UPDATE;
--# SET FREQUENCY 524
SELECT COUNT(*) FROM "SYSIBM".SYSTABLES WHERE NAME = 'SYSDATAPARTITIONEXPRESSION' AND CREATOR = 'SYSIBM  ' AND TYP
--# SET FREQUENCY 532
SELECT COUNT(*) FROM "SYSIBM".SYSTABLES WHERE NAME = 'SYSCOLUMNS' AND CREATOR = 'SYSIBM  ' AND TYPE = 'S';

  47 record(s) selected
```

注: 为了便于显示，上述样本输出中的行已被截断。

7. 使用步骤 第 226 页的 6 中查询所返回的语句为 **db2advis** 命令创建一个输入文件。（有关为 **db2advis** 命令创建输入文件的更多信息，请参阅该命令的参考文档。）
8. 使用在步骤 7 中创建的输入文件来运行 **db2advis** 命令。 例如，如果您创建的输入文件名为 pkgcache_stmts.txt，那么运行如下所示的命令:

   ```
   db2advis -d customer -i pkgcache_stmts.txt -m MICP
   ```

   其中
   - **-d CUSTOMER** 标识您正在为其获取建议的数据库的名称
   - **-i pkgcache_stmts.txt** 标识 **db2advis** 的输入文件的名称
   - **-m MICP** 是 db2advis 命令的伪指令，用于生成以下提高性能的建议:
     - **M** 新的具体化查询表
     - **I** 新的索引
     - **C** 将标准表转换为多维集群表 (MQT)
     - **P** 对现有索引重新分区

结果

设计顾问程序返回类似以下的建议:

```
execution started at timestamp 2010-03-16-14.25.57.562000
Using the default table space name USERSPACE1
found [47] SQL statements from the input file
excluding statement [0] from the workload.
excluding statement [1] from the workload.
excluding statement [19] from the workload.
excluding statement [39] from the workload.
Recommending indexes...
Recommending MQTs...
Recommending Multi-Dimensional Clusterings...
Found 19 user defined views in the catalog table
Found [17] candidate MQTs
Getting cost of workload with MQTs
total disk space needed for initial set [   0.159] MB
total disk space constrained to         [  69.215] MB
  2  indexes in current solution
  0  MQTs in current solution
total disk space needed for initial set [   0.024] MB
total disk space constrained to         [ 103.822] MB
No useful Multi-dimensional Clustering dimensions for this workload
 [5651.8281] timerons  (without recommendations)
 [5519.8281] timerons  (with current solution)
 [2.34%] improvement


--
```

```
--
-- LIST OF MODIFIED CREATE-TABLE STATEMENTS WITH RECOMMENDED PARTITIONING KEYS AND TABLESPACES AND/OR RECOMMENDED MULTI-DIMENSIONAL CLUSTERINGS
-- ===========================
--  No new partitioning keys or tablespaces are recommended for this workload.


--
--
-- LIST OF RECOMMENDED MQTs
-- ===========================


--
--
-- RECOMMENDED EXISTING MQTs
-- ===========================


--
--
-- UNUSED EXISTING MQTs
-- ===========================
-- DROP TABLE "ASRISK  "."ADEFUSR";


--
--
-- RECOMMENDED CLUSTERING INDEXES
-- ===========================


--
--
-- LIST OF RECOMMENDED INDEXES
-- ===========================
-- index[1],    0.024MB
   CREATE INDEX "ASRISK  "."IDX003161830530000" ON "ASRISK  "."SYSINDEXES"
   ("CREATOR" ASC, "NAME" ASC, "TBCREATOR" ASC, "TBNAME"
   ASC) ALLOW REVERSE SCANS COLLECT SAMPLED DETAILED STATISTICS;
   COMMIT WORK;


--
--
-- RECOMMENDED EXISTING INDEXES
-- ===========================
-- RUNSTATS ON TABLE "SYSTOOLS"."POLICY" FOR SAMPLED DETAILED INDEX "SYSTOOLS"."POLICY_UNQ" ;
-- COMMIT WORK ;
-- RUNSTATS ON TABLE "SYSTOOLS"."HMON_ATM_INFO" FOR SAMPLED DETAILED INDEX "SYSTOOLS"."ATM_UNIQ" ;
-- COMMIT WORK ;
-- RUNSTATS ON TABLE "SYSIBM  "."SYSDATAPARTITIONS" FOR SAMPLED DETAILED INDEX "SYSIBM  "."INDDATAPARTITIONS03" ;
-- COMMIT WORK ;
-- RUNSTATS ON TABLE "SYSIBM  "."SYSTABLES" FOR SAMPLED DETAILED INDEX "SYSIBM  "."INDTABLES01" ;
-- COMMIT WORK ;
-- RUNSTATS ON TABLE "SYSIBM  "."SYSTABLESPACES" FOR SAMPLED DETAILED INDEX "SYSIBM  "."INDTABLESPACES04" ;
-- COMMIT WORK ;
-- RUNSTATS ON TABLE "SYSIBM  "."SYSCOLUMNS" FOR SAMPLED DETAILED INDEX "SYSIBM  "."INDCOLUMNS01" ;
-- COMMIT WORK ;
-- RUNSTATS ON TABLE "SYSIBM  "."SYSINDEXES" FOR SAMPLED DETAILED INDEX "SYSIBM  "."INDINDEXES02" ;
-- COMMIT WORK ;
-- RUNSTATS ON TABLE "SYSIBM  "."SYSTRIGGERS" FOR SAMPLED DETAILED INDEX "SYSIBM  "."INDTRIGGERS02" ;
-- COMMIT WORK ;


--
--
-- UNUSED EXISTING INDEXES
-- ===========================
-- DROP INDEX "ASRISK  "."PKGCACHE_EVENT_IND1";
-- ===========================
--

-- ====ADVISOR DETAILED XML OUTPUT=============
-- ==(Benefits do not include clustering recommendations)==

⋮
```

注: 为了便于表示，已截断了设计顾问程序的输出。

下一步做什么

当确定对数据库进行哪些更改来提高性能时，使用设计顾问程序中的输出来提供帮助。

活动事件监视

活动事件监视器捕获与在系统上运行的活动有关的数据。可使用此事件监视器来收集数据以帮助您更好地了解语句及系统上的常规负载的性能和行为。

系统中的每个活动完成后，活动事件监视器会记录信息。相比之下，工作单元事件监视器在每个事务完成时记录数据。通过使用活动事件监视器，可检查与各语句的执行相关的监视元素。

活动事件监视器返回的数据是对以下表函数返回的数据的补充:

- MON_GET_ACTIVITY_DETAILS
- MON_GET_PKG_CACHE_STMT
- MON_GET_PKG_CACHE_STMT_DETAILS

虽然事件监视器返回有关系统上运行的活动的历史信息，但表函数会提供有关系统上已运行或最新运行的活动的信息。

使用活动事件监视器

与其他事件监视器配合使用

活动事件监视器在与其他事件监视器配合使用时特别有用。例如，您可能想要捕获违反所定义阈值的语句执行的信息。在此情况下，您执行以下步骤:

1. 使用 CREATE THRESHOLD 语句来定义阈值。在定义阈值时，指定 COLLECT ACTIVITY DATA 子句以便让任何处于活动状态的活动事件监视器记录活动数据。
2. 创建阈值违例事件监视器以捕获有关阈值违例时间的详细信息以及有关当时系统中所发生情况的其他数据。
3. 创建用于捕获阈值违例生成的活动信息的活动事件监视器。
4. 运行应用程序或工作负载。
5. 查询事件监视器输出以查看有关违反阈值时所发生情况的信息。可连接来自阈值违例事件监视器的数据与来自活动事件监视器的数据以找出发生违例时正在运行的语句。

第 244 页的『示例: 捕获与语句的执行相关的活动信息』中较详细地说明了此过程。

该活动事件监视器的其他应用包括以下各项:

- 捕获有关长时间运行的查询的信息。在此情况下，您运行 WLM_CAPTURE_ACTIVITY_IN_PROGRESS 过程以在活动完成前强制收集有关活动的信息。如果您想要终止长时间运行的语句但又想捕获其相关信息，那么运行此过程很有用。
- 查看特定工作负载中的应用程序正在运行哪些语句。

命令、过程或工具的输入

活动事件监视器生成的数据可用作各种工具和存储过程（包括下列各项）的输入:

db2advis - DB2 设计顾问程序命令

db2advis 命令可使用活动事件监视器的输出来生成有关以下项目和活动的建议:

- 具体化查询表 (MQT)

- 索引
- 对表重新分区
- 对多维集群 (MDC) 表的转换
- 删除未使用对象

db2expln - SQL 和 XQuery 说明命令

db2expln 命令可使用活动事件监视器中的部分信息来描述与该部分相关的语句的存取方案。

EXPLAIN_FROM_ACTIVITY 存储过程

EXPLAIN_FROM_ACTIVITY 存储过程说明如何使用过程从活动事件监视器获取的部分的内容来具体执行语句。说明输出放在说明表中以便使用说明工具（例如，**db2exfmt** 命令）进行处理。说明输出包含（如果可用）存取方案和部分实际值（存取方案中的运算符的运行时统计信息）。

工作负载管理历史分析工具

wlmhist.pl 和 wlmhistrep.pl Perl 脚本通过使用活动数事件监视器捕获的信息来执行历史分析。

部分实际值

活动事件监视器的另一用法是捕获部分实际值。可使用此数据来比较活动事件监视器捕获的实际值与存取方案中的估算成本。执行此比较允许您查看该存取方案是否仍然有效。

创建活动事件监视器

要创建用于捕获活动事件的事件监视器，请使用 CREATE EVENT MONITOR FOR ACTIVITIES 语句。

开始之前

必须具有 DBADM 或 SQLADM 权限才能创建活动事件监视器。

过程

要创建活动事件监视器，请执行以下操作:

1. 构造 CREATE EVENT MONITOR FOR ACTIVITIES 语句。 例如，要创建名为 myactevmon 的事件监视器，可使用类似如下的语句:

```
CREATE EVENT MONITOR myactevmon FOR ACTIVITIES
    WRITE TO TABLE
```

2. 运行此语句。 此示例创建名为 myactevmon 并具有以下特征的活动事件监视器:
   - 收集适用于活动事件监视器的所有逻辑数据组的监视元素。
   - 输出写至关系表。每个表被指定缺省表名。
   - 该事件监视器配置为第一次激活数据库时自动启动。
   -

   这些结果是对活动事件监视器使用缺省设置导致的。必要时，可覆盖这些缺省值。
3. 激活事件监视器。 尽管事件监视器配置为自动启动，但直到创建该事件监视器后第一次激活数据库，它才会自动启动。要强制事件监视器立即开始收集数据，请使用 SET EVENT MONITOR STATE 语句，如以下示例所示:

```
SET EVENT MONITOR myactevmon STATE 1
```

下一步做什么

配置事件监视器以收集所需数据。

为活动事件监视器配置数据收集

必须先配置数据收集，才能收集与活动相关的事件数据。可使用的配置选项取决于收集数据的用途。

关于此任务

有两种不同方法可用来为活动事件监视器配置数据收集:

- 可在每当违反您使用 CREATE THRESHOLD 语句创建的阈值时生成活动事件数据。
- 可对 WLM 对象的 CREATE 或 ALTER 语句指定 COLLECT ACTIVITY METRICS 子句。

过程

要收集与活动相关的事件数据，请执行以下操作:

1. 确定要对其收集数据的活动。 下表概述各个选项:

要收集哪些活动的数据?	配置由以下各项控制:
与阈值违例相关的活动	对于与阈值违例相关的活动，请发出 CREATE THRESHOLD 或 ALTER THRESHOLD 语句并指定 COLLECT ACTIVITY DATA 子句。如果指定 COLLECT ACTIVITY DATA 子句，那么每当违反该阈值时，系统会将活动数据发送至所有处于活动状态的活动事件监视器。（可选）指定下列其中一个附加子句: WITH DETAILS、WITH DETAILS, SECTION 或 AND VALUES。 **提示:** 考虑将 **mon_act_metrics** 配置参数的值更改为 NONE。这样做会将对其收集数据的活动的范围限制为仅与阈值违例相关的活动。

要收集哪些活动的数据?	配置由以下各项控制:
与特定工作负载管理对象相关的活动	对于与工作负载管理相关的活动，请发出下列其中一个语句并指定 COLLECT ACTIVITY DATA 子句。（可选）指定下列其中一个附加子句: WITH DETAILS、WITH DETAILS, SECTION、WITH SECTION INCLUDE ACTUALS BASE 或 AND VALUES: • CREATE WORKLOAD • ALTER WORKLOAD • CREATE WORK ACTION SET • ALTER WORK ACTION SET • CREATE SERVICE CLASS • ALTER SERVICE CLASS 如果在上述任何语句中指定 COLLECT ACTIVITY DATA 子句，那么系统会将该语句中引用的工作负载对象的活动数据发送至任何处于活动状态的活动事件监视器。 例如，要收集 PAYROLL 工作负载的活动事件数据，可通过发出以下语句来改变该工作负载: `ALTER WORKLOAD PAYROLL COLLECT ACTIVITY DATA WITH SECTION INCLUDE ACTUALS BASE` 在此示例中，活动数据是与部分实际值数据一起收集的。

2. 使用上一步中描述的其中一个配置选项来设置收集级别。 例如，要收集 PAYROLL 工作负载的活动事件数据，可使用以下语句改变该工作负载:

   ```
   ALTER WORKLOAD PAYROLL COLLECT ACTIVITY DATA WITH SECTION INCLUDE ACTUALS BASE
   ```

   在最后一个示例中，收集了活动数据以及部分实际值的数据。

结果

已配置数据收集。

收集缺省用户工作负载的活动数据

如果要收集未映射至特定工作负载的活动的活动事件数据，那么可通过发出类似如下的语句来收集缺省用户工作负载的数据:

```
ALTER WORKLOAD SYSDEFAULTUSERWORKLOAD COLLECT ACTIVITY DATA ON COORDINATOR
 WITH DETAILS
```

此语句会导致对未与用户定义工作负载对象相关联的任何活动收集详细活动数据。缺省情况下，如果没有用户定义的工作负载对象，那么所有数据库连接与 SYSDEFAULTUSERWORKLOAD 相关联。

下一步做什么

运行应用程序或工作负载。满足收集条件的活动运行时，事件数据会发送至所有处于活动状态的活动事件监视器。

活动事件监视器生成的数据

可选择将活动事件监视器的输出写至常规表、文件或命名管道。

不管您选择什么输出格式，活动事件监视器捕获的所有数据都来自下列四个逻辑数据组中的一个:

- event_activity
- event_activitymetrics
- event_activitystmt
- event_activityvals

如果选择将统计信息事件监视器数据写至常规表，那么另一个组 (CONTROL) 中的数据用于生成有关事件监视器本身的元数据。

活动事件监视器写至表的信息:

在指定了 WRITE TO TABLE 选项的情况下活动事件监视器写入的信息。

选择 WRITE TO TABLE 作为活动事件监视器的输出类型时，缺省情况下，会生成五个表，每个表包含一个或多个逻辑数据组中的监视元素:

表 47. ACTIVITIES 写至表事件监视器生成的表. 表名通过逻辑数据组（用于填充该表）的名称和 CREATE EVENT MONITOR 语句中给予事件监视器的名称（如下表的表名中的 *evmon-name* 所示）并置派生。

缺省表名	报告的逻辑数据组
ACTIVITY_*evmon-name*	event_activity
ACTIVITYSTMT_*evmon-name*	event_activitystmt
ACTIVITYVALS_*evmon-name*	event_activityvals
ACTIVITYMETRICS_*evmon-name*	event_activitymetrics
CONTROL_*evmon-name*	CONTROL 逻辑组由来自 event_dbheader、event_start 和 event_overflow 逻辑数据组中的一个或多个的所选元素组成。

要将事件监视器的输出限制为发送至特定表，请在 CREATE EVENT MONITOR 或 ALTER EVENT MONITOR 语句中指定要对其生成表的逻辑组的名称。有关详细信息，请参阅这些语句的参考主题。

生成的表

表 48. 对活动事件监视器返回的信息: 缺省表名: ACTIVITY_evmon-name

列名	数据类型	描述
PARTITION_KEY	INTEGER	第 951 页的『partition_key -“分区键”监视元素』
ACT_EXEC_TIME	BIGINT	act_exec_time - 活动执行时间
ACTIVATE_TIMESTAMP	TIMESTAMP	activate_timestamp - 激活时间戳记
ACTIVITY_ID	BIGINT	activity_id - 活动标识

表 48. 对活动事件监视器返回的信息: 缺省表名: ACTIVITY_evmon-name (续)

列名	数据类型	描述
ACTIVITY_SECONDARY_ID	SMALLINT	activity_secondary_id - 活动辅助标识
ACTIVITY_TYPE	VARCHAR(64)	activity_type - 活动类型
ADDRESS	VARCHAR(128)	address - 从中发起连接的 IP 地址
AGENT_ID	BIGINT	agent_id - 应用程序句柄（代理程序标识）
APPL_ID	VARCHAR(64)	appl_id - 应用程序标识
APPL_NAME	VARCHAR(255)	appl_name - 应用程序名称
ARM_CORRELATOR	BLOB(0)	arm_correlator - 应用程序响应测量相关因子
COORD_PARTITION_NUM	INTEGER	coord_partition_num - 协调程序分区号
DB_WORK_ACTION_SET_ID	INTEGER	db_work_action_set_id - 数据库工作操作集标识
DB_WORK_CLASS_ID	INTEGER	db_work_class_id - 数据库工作类标识
DETAILS_XML	BLOB(0)	
MON_INTERVAL_ID	BIGINT	mon_interval_id - 监视时间间隔标识
NUM_REMAPS	BIGINT	num_remaps - 重新映射次数
PARENT_ACTIVITY_ID	BIGINT	parent_activity_id - 父活动标识
PARENT_UOW_ID	INTEGER	parent_uow_id - 父工作单元标识
PARTIAL_RECORD	SMALLINT	partial_record - 部分记录
PARTITION_NUMBER	SMALLINT	partition_number - 分区号
POOL_DATA_L_READS	BIGINT	pool_data_l_reads - 缓冲池数据逻辑读取数
POOL_DATA_P_READS	BIGINT	pool_data_p_reads - 缓冲池数据物理读取数
POOL_INDEX_L_READS	BIGINT	pool_index_l_reads - 缓冲池索引逻辑读取数
POOL_INDEX_P_READS	BIGINT	pool_index_p_reads - 缓冲池索引物理读取数
POOL_TEMP_DATA_L_READS	BIGINT	pool_temp_data_l_reads - 缓冲池临时数据逻辑读取数
POOL_TEMP_DATA_P_READS	BIGINT	pool_temp_data_p_reads - 缓冲池临时数据物理读取数
POOL_TEMP_INDEX_L_READS	BIGINT	pool_temp_index_l_reads - 缓冲池临时索引逻辑读取数
POOL_TEMP_INDEX_P_READS	BIGINT	pool_temp_index_p_reads - 缓冲池临时索引物理读取数
POOL_TEMP_XDA_L_READS	BIGINT	pool_temp_xda_l_reads - 缓冲池临时 XDA 数据逻辑读取数
POOL_TEMP_XDA_P_READS	BIGINT	pool_temp_xda_p_reads - 缓冲池临时 XDA 数据物理读取数
POOL_XDA_L_READS	BIGINT	pool_xda_l_reads - 缓冲池 XDA 数据逻辑读取数
POOL_XDA_P_READS	BIGINT	pool_xda_p_reads - 缓冲池 XDA 数据物理读取数
PREP_TIME	BIGINT	prep_time - 编译时间
QUERY_ACTUAL_DEGREE	INTEGER	query_actual_degree - 实际运行时分区内并行度
QUERY_CARD_ESTIMATE	BIGINT	query_card_estimate - 行查询数估计
QUERY_COST_ESTIMATE	BIGINT	query_cost_estimate - 查询估算成本
QUERY_DATA_TAG_LIST	VARCHAR(32)	query_data_tag_list - 查询数据标记列表
ROWS_FETCHED	BIGINT	rows_fetched - 访存的行数
ROWS_MODIFIED	BIGINT	rows_modified - 修改的行数
ROWS_RETURNED	BIGINT	rows_returned - 返回的行数

表 48. 对活动事件监视器返回的信息：缺省表名：ACTIVITY_evmon-name（续）

列名	数据类型	描述
SC_WORK_ACTION_SET_ID	INTEGER	sc_work_action_set_id - 服务类工作操作集标识
SC_WORK_CLASS_ID	INTEGER	sc_work_class_id - 服务类工作类标识
SECTION_ACTUALS	BLOB(0)	section_actuals - 部分实际值
SERVICE_SUBCLASS_NAME	VARCHAR(128)	service_subclass_name - 服务子类名
SERVICE_SUPERCLASS_NAME	VARCHAR(128)	service_superclass_name - 服务超类名
SESSION_AUTH_ID	VARCHAR(128)	session_auth_id - 会话授权标识
SORT_OVERFLOWS	BIGINT	sort_overflows - 排序溢出数
SQLCABC	INTEGER	请参阅《*SQL Reference Volume 1*》中的『SQLCA（SQL 通信区）』。
SQLCAID	CHARACTER(8)	请参阅《*SQL Reference Volume 1*》中的『SQLCA（SQL 通信区）』。
SQLCODE	INTEGER	请参阅《*SQL Reference Volume 1*》中的『SQLCA（SQL 通信区）』。
SQLERRD1	INTEGER	请参阅《*SQL Reference Volume 1*》中的『SQLCA（SQL 通信区）』。
SQLERRD2	INTEGER	请参阅《*SQL Reference Volume 1*》中的『SQLCA（SQL 通信区）』。
SQLERRD3	INTEGER	请参阅《*SQL Reference Volume 1*》中的『SQLCA（SQL 通信区）』。
SQLERRD4	INTEGER	请参阅《*SQL Reference Volume 1*》中的『SQLCA（SQL 通信区）』。
SQLERRD5	INTEGER	请参阅《*SQL Reference Volume 1*》中的『SQLCA（SQL 通信区）』。
SQLERRD6	INTEGER	请参阅《*SQL Reference Volume 1*》中的『SQLCA（SQL 通信区）』。
SQLERRM	VARCHAR(72)	请参阅《*SQL Reference Volume 1*》中的『SQLCA（SQL 通信区）』。
SQLERRP	CHARACTER(8)	请参阅《*SQL Reference Volume 1*》中的『SQLCA（SQL 通信区）』。
SQLSTATE	CHARACTER(5)	请参阅《*SQL Reference Volume 1*》中的『SQLCA（SQL 通信区）』。
SQLWARN	CHARACTER(11)	请参阅《*SQL Reference Volume 1*》中的『SQLCA（SQL 通信区）』。
SYSTEM_CPU_TIME	BIGINT	system_cpu_time - 系统 CPU 时间
TIME_COMPLETED	TIMESTAMP	time_completed - 完成时间
TIME_CREATED	TIMESTAMP	time_created - 创建时间
TIME_STARTED	TIMESTAMP	time_started - 开始时间
TOTAL_SORT_TIME	BIGINT	total_sort_time - 总排序时间
TOTAL_SORTS	BIGINT	total_sorts - 排序总数
TOTAL_STATS_FABRICATION_TIME	BIGINT	total_stats_fabrication_time - 统计信息生成时间总计
TOTAL_STATS_FABRICATIONS	BIGINT	total_stats_fabrications - 统计信息生成总计
TOTAL_SYNC_RUNSTATS	BIGINT	total_sync_runstats - 同步 RUNSTATS 活动总数

表 48. 对活动事件监视器返回的信息：缺省表名：ACTIVITY_evmon-name（续）

列名	数据类型	描述
TOTAL_SYNC_RUNSTATS_TIME	BIGINT	total_sync_runstats_time - 同步 RUNSTATS 时间总计
TPMON_ACC_STR	VARCHAR(200)	tpmon_acc_str - TP 监视器客户机记帐字符串
TPMON_CLIENT_APP	VARCHAR(255)	tpmon_client_app - TP 监视器客户机应用程序名称
TPMON_CLIENT_USERID	VARCHAR(255)	tpmon_client_userid - TP 监视器客户机用户标识
TPMON_CLIENT_WKSTN	VARCHAR(255)	tpmon_client_wkstn - TP 监视器客户机工作站名称
UOW_ID	INTEGER	uow_id - 工作单元标识
USER_CPU_TIME	BIGINT	user_cpu_time - 用户 CPU 时间
WL_WORK_ACTION_SET_ID	INTEGER	wl_work_action_set_id -“工作负载工作操作集标识”
WL_WORK_CLASS_ID	INTEGER	wl_work_class_id - 工作负载工作类标识
WORKLOAD_ID	INTEGER	workload_id - 工作负载标识
WORKLOAD_OCCURRENCE_ID	INTEGER	workload_occurrence_id - 工作负载项标识

表 49. 对活动事件监视器返回的信息：表名：ACTIVITYSTMT_evmon-name

列名	数据类型	描述
PARTITION_KEY	INTEGER	第 951 页的『partition_key -“分区键”监视元素』
ACTIVATE_TIMESTAMP	TIMESTAMP	activate_timestamp - 激活时间戳记
ACTIVITY_ID	BIGINT	activity_id - 活动标识
ACTIVITY_SECONDARY_ID	SMALLINT	activity_secondary_id - 活动辅助标识
APPL_ID	VARCHAR(64)	appl_id - 应用程序标识
COMP_ENV_DESC	BLOB(0)	comp_env_desc - 编译环境
CREATOR	VARCHAR(128)	creator - 应用程序创建者
EFF_STMT_TEXT	CLOB	eff_stmt_text - 有效语句文本
EXECUTABLE_ID	VARCHAR(32)	executable_id - 可执行标识
PACKAGE_NAME	VARCHAR(128)	package_name - 程序包名
PACKAGE_VERSION_ID	VARCHAR(64)	package_version_id - 程序包版本
PARTITION_NUMBER	SMALLINT	partition_number - 分区号
ROUTINE_ID	BIGINT	routine_id - 例程标识
SECTION_ENV	BLOB(0)	section_env - 节环境
SECTION_NUMBER	BIGINT	section_number - 节号
STMT_FIRST_USE_TIME	TIMESTAMP	stmt_first_use_time - 第一次使用语句时的时间戳记
STMT_INVOCATION_ID	BIGINT	stmt_invocation_id - 语句调用标识
STMT_ISOLATION	BIGINT	stmt_isolation - 语句隔离
STMT_LAST_USE_TIME	TIMESTAMP	stmt_last_use_time - 上一次使用语句时的时间戳记
STMT_LOCK_TIMEOUT	INTEGER	stmt_lock_timeout - 语句锁定超时
STMT_NEST_LEVEL	BIGINT	stmt_nest_level - 语句嵌套级别
STMT_PKGCACHE_ID	BIGINT	stmt_pkgcache_id - 语句程序包高速缓存标识
STMT_QUERY_ID	BIGINT	stmt_query_id - 语句查询标识
STMT_SOURCE_ID	BIGINT	stmt_source_id - 语句源标识
STMT_TEXT	CLOB	stmt_text - SQL 语句文本
STMT_TYPE	BIGINT	stmt_type - 语句类型

表 49. 对活动事件监视器返回的信息: 表名: ACTIVITYSTMT_evmon-name (续)

列名	数据类型	描述
UOW_ID	INTEGER	uow_id - 工作单元标识

表 50. 对活动事件监视器返回的信息: 表名: ACTIVITYMETRICS_evmon-name

列名	数据类型	描述
PARTITION_KEY	INTEGER	第 951 页的『partition_key -“分区键”监视元素』
ACTIVITY_ID	BIGINT	activity_id - 活动标识
ACTIVITY_SECONDARY_ID	SMALLINT	activity_secondary_id - 活动辅助标识
APPL_ID	VARCHAR(64)	appl_id - 应用程序标识
PARTITION_NUMBER	SMALLINT	partition_number - 分区号
UOW_ID	INTEGER	uow_id - 工作单元标识
WLM_QUEUE_TIME_TOTAL	BIGINT	wlm_queue_time_total - 工作负载管理器队列时间总计
WLM_QUEUE_ASSIGNMENTS_TOTAL	BIGINT	wlm_queue_assignments_total - 工作负载管理器队列分配总计
FCM_TQ_RECV_WAIT_TIME	BIGINT	fcm_tq_recv_wait_time - FCM 表队列接收等待时间
FCM_MESSAGE_RECV_WAIT_TIME	BIGINT	fcm_message_recv_wait_time - 接收 FCM 消息等待时间
FCM_TQ_SEND_WAIT_TIME	BIGINT	fcm_tq_send_wait_time - FCM 表队列发送等待时间
FCM_MESSAGE_SEND_WAIT_TIME	BIGINT	fcm_message_send_wait_time - 发送 FCM 消息等待时间
LOCK_WAIT_TIME	BIGINT	lock_wait_time - 等待锁定时间
LOCK_WAITS	BIGINT	lock_waits - 等待锁定次数
DIRECT_READ_TIME	BIGINT	direct_read_time - 直接读时间
DIRECT_READ_REQS	BIGINT	direct_read_reqs - 直接读请求数
DIRECT_WRITE_TIME	BIGINT	direct_write_time - 直接写时间
DIRECT_WRITE_REQS	BIGINT	direct_write_reqs - 直接写请求数
LOG_BUFFER_WAIT_TIME	BIGINT	log_buffer_wait_time - 日志缓冲区等待时间
NUM_LOG_BUFFER_FULL	BIGINT	num_log_buffer_full - 日志缓冲区变满次数
LOG_DISK_WAIT_TIME	BIGINT	log_disk_wait_time - 日志磁盘等待时间
LOG_DISK_WAITS_TOTAL	BIGINT	log_disk_waits_total - 日志磁盘等待总次数
POOL_WRITE_TIME	BIGINT	pool_write_time - 缓冲池物理写时间总计
POOL_READ_TIME	BIGINT	pool_read_time - 缓冲池物理读时间总计
AUDIT_FILE_WRITE_WAIT_TIME	BIGINT	audit_file_write_wait_time - 审计文件写等待时间
AUDIT_FILE_WRITES_TOTAL	BIGINT	audit_file_writes_total - 写审计文件总次数
AUDIT_SUBSYSTEM_WAIT_TIME	BIGINT	audit_subsystem_wait_time - 审计子系统等待时间
AUDIT_SUBSYSTEM_WAITS_TOTAL	BIGINT	audit_subsystem_waits_total - 审计子系统等待总次数
DIAGLOG_WRITE_WAIT_TIME	BIGINT	diaglog_write_wait_time - 诊断日志文件写等待时间
DIAGLOG_WRITES_TOTAL	BIGINT	diaglog_writes_total - 写诊断日志文件总次数
FCM_SEND_WAIT_TIME	BIGINT	fcm_send_wait_time - FCM 发送等待时间
FCM_RECV_WAIT_TIME	BIGINT	fcm_recv_wait_time - FCM 接收等待时间
TOTAL_ACT_WAIT_TIME	BIGINT	total_act_wait_time - 活动等待时间总计

表 50. 对活动事件监视器返回的信息: 表名: ACTIVITYMETRICS_evmon-name (续)

列名	数据类型	描述
TOTAL_SECTION_SORT_PROC_TIME	BIGINT	total_section_sort_proc_time - 节排序处理时间总计
TOTAL_SECTION_SORTS	BIGINT	total_section_sorts - 节排序总次数
TOTAL_SECTION_SORT_TIME	BIGINT	total_section_sort_time - 节排序时间总计
TOTAL_ACT_TIME	BIGINT	total_act_time - 活动时间总计
ROWS_READ	BIGINT	rows_read - 读取行数
ROWS_MODIFIED	BIGINT	rows_modified - 修改的行数
POOL_DATA_L_READS	BIGINT	pool_data_l_reads - 缓冲池数据逻辑读取数
POOL_INDEX_L_READS	BIGINT	pool_index_l_reads - 缓冲池索引逻辑读取数
POOL_TEMP_DATA_L_READS	BIGINT	pool_temp_data_l_reads - 缓冲池临时数据逻辑读取数
POOL_TEMP_INDEX_L_READS	BIGINT	pool_temp_index_l_reads - 缓冲池临时索引逻辑读取数
POOL_XDA_L_READS	BIGINT	pool_xda_l_reads - 缓冲池 XDA 数据逻辑读取数
POOL_TEMP_XDA_L_READS	BIGINT	pool_temp_xda_l_reads - 缓冲池临时 XDA 数据逻辑读取数
TOTAL_CPU_TIME	BIGINT	total_cpu_time - CPU 时间总计
POOL_DATA_P_READS	BIGINT	pool_data_p_reads - 缓冲池数据物理读取数
POOL_TEMP_DATA_P_READS	BIGINT	pool_temp_data_p_reads - 缓冲池临时数据物理读取数
POOL_XDA_P_READS	BIGINT	pool_xda_p_reads - 缓冲池 XDA 数据物理读取数
POOL_TEMP_XDA_P_READS	BIGINT	pool_temp_xda_p_reads - 缓冲池临时 XDA 数据物理读取数
POOL_INDEX_P_READS	BIGINT	pool_index_p_reads - 缓冲池索引物理读取数
POOL_TEMP_INDEX_P_READS	BIGINT	pool_temp_index_p_reads - 缓冲池临时索引物理读取数
POOL_DATA_WRITES	BIGINT	pool_data_writes - 缓冲池数据写次数
POOL_XDA_WRITES	BIGINT	pool_xda_writes - 缓冲池 XDA 数据写次数
POOL_INDEX_WRITES	BIGINT	pool_index_writes - 缓冲池索引写次数
DIRECT_READS	BIGINT	direct_reads - 直接读数据库数目
DIRECT_WRITES	BIGINT	direct_writes - 直接写数据库数目
ROWS_RETURNED	BIGINT	rows_returned - 返回的行数
DEADLOCKS	BIGINT	deadlocks - 检测到的死锁数
LOCK_TIMEOUTS	BIGINT	lock_timeouts - 锁定超时次数
LOCK_ESCALS	BIGINT	lock_escals - 锁定升级次数
FCM_SENDS_TOTAL	BIGINT	fcm_sends_total - FCM 发送总计
FCM_RECVS_TOTAL	BIGINT	fcm_recvs_total - FCM 接收总计
FCM_SEND_VOLUME	BIGINT	fcm_send_volume - FCM 发送量
FCM_RECV_VOLUME	BIGINT	fcm_recv_volume - FCM 接收量
FCM_MESSAGE_SENDS_TOTAL	BIGINT	fcm_message_sends_total - 发送 FCM 消息总数
FCM_MESSAGE_RECVS_TOTAL	BIGINT	fcm_message_recvs_total - 接收 FCM 消息总数
FCM_MESSAGE_SEND_VOLUME	BIGINT	fcm_message_send_volume - 发送 FCM 消息量
FCM_MESSAGE_RECV_VOLUME	BIGINT	fcm_message_recv_volume - 接收 FCM 消息量
FCM_TQ_SENDS_TOTAL	BIGINT	fcm_tq_sends_total - FCM 表队列发送总次数
FCM_TQ_RECVS_TOTAL	BIGINT	fcm_tq_recvs_total - FCM 表队列接收总量

表 50. 对活动事件监视器返回的信息: 表名: ACTIVITYMETRICS_evmon-name (续)

列名	数据类型	描述
FCM_TQ_SEND_VOLUME	BIGINT	fcm_tq_send_volume - FCM 表队列发送量
FCM_TQ_RECV_VOLUME	BIGINT	fcm_tq_recv_volume - FCM 表队列接收量
TQ_TOT_SEND_SPILLS	BIGINT	tq_tot_send_spills - 溢出表队列缓冲区总数
POST_THRESHOLD_SORTS	BIGINT	post_threshold_sorts - 超过阈值后的排序数
POST_SHRTHRESHOLD_SORTS	BIGINT	post_shrthreshold_sorts - 共享阈值后排序数
SORT_OVERFLOWS	BIGINT	sort_overflows - 排序溢出数
AUDIT_EVENTS_TOTAL	BIGINT	audit_events_total - 审计事件总数
TOTAL_SORTS	BIGINT	total_sorts - 排序总数
STMT_EXEC_TIME	BIGINT	stmt_exec_time - 语句执行时间
COORD_STMT_EXEC_TIME	BIGINT	coord_stmt_exec_time - 协调代理程序执行语句的时间
TOTAL_ROUTINE_NON_SECT_PROC _TIME	BIGINT	total_routine_non_sect_proc_time - 非部分处理时间
TOTAL_ROUTINE_NON_SECT_TIME	BIGINT	total_routine_non_sect_time - 非部分例程执行时间
TOTAL_SECTION_PROC_TIME	BIGINT	total_section_proc_time - 部分处理时间总计
TOTAL_APP_SECTION_EXECUTIONS	BIGINT	total_app_section_executions - 应用程序执行部分执行的总次数
TOTAL_SECTION_TIME	BIGINT	total_section_time - 部分时间总计
TOTAL_ROUTINE_USER_CODE_PROC _TIME	BIGINT	total_routine_user_code_proc_time - 例程用户代码处理时间总计
TOTAL_ROUTINE_USER_CODE_TIME	BIGINT	total_routine_user_code_time - 例程用户代码时间总计
TOTAL_ROUTINE_TIME	BIGINT	total_routine_time - 例程时间总计
THRESH_VIOLATIONS	BIGINT	thresh_violations - 阈值违例次数
NUM_LW_THRESH_EXCEEDED	BIGINT	num_lw_thresh_exceeded - 超过锁定等待阈值的次数
TOTAL_ROUTINE_INVOCATIONS	BIGINT	total_routine_invocations - 例程调用总计
LOCK_WAIT_TIME_GLOBAL	BIGINT	lock_wait_time_global - 锁定等待时间全局
LOCK_WAITS_GLOBAL	BIGINT	lock_waits_global - 锁定等待全局
RECLAIM_WAIT_TIME	BIGINT	reclaim_wait_time - 回收等待时间
SPACEMAPPAGE_RECLAIM_WAIT_TIME	BIGINT	spacemappage_reclaim_wait_time - 空间映射页回收等待时间
LOCK_TIMEOUTS_GLOBAL	BIGINT	lock_timeouts_global - 锁定超时全局
LOCK_ESCALS_MAXLOCKS	BIGINT	lock_escals_maxlocks - maxlocks 锁定升级数
LOCK_ESCALS_LOCKLIST	BIGINT	lock_escals_locklist - locklist 锁定升级数
LOCK_ESCALS_GLOBAL	BIGINT	lock_escals_global - 全局锁定升级数
CF_WAIT_TIME	BIGINT	cf_wait_time - 集群高速缓存工具等待时间
CF_WAITS	BIGINT	cf_waits - 集群高速缓存工具 DB2 pureScale 服务器等待数
POOL_DATA_GBP_L_READS	BIGINT	pool_data_gbp_l_reads - 组缓冲池数据逻辑读取数
POOL_DATA_GBP_P_READS	BIGINT	pool_data_gbp_p_reads - 组缓冲池数据物理读取数
POOL_DATA_LBP_PAGES_FOUND	BIGINT	pool_data_lbp_pages_found - 本地缓冲池发现的数据页数
POOL_DATA_GBP_INVALID_PAGES	BIGINT	pool_data_gbp_invalid_pages - 组缓冲池无效数据页数

表 50. 对活动事件监视器返回的信息：表名：ACTIVITYMETRICS_evmon-name（续）

列名	数据类型	描述
POOL_INDEX_GBP_L_READS	BIGINT	pool_index_gbp_l_reads - 组缓冲池索引逻辑读取数
POOL_INDEX_GBP_P_READS	BIGINT	pool_index_gbp_p_reads - 组缓冲池索引物理读取数
POOL_INDEX_LBP_PAGES_FOUND	BIGINT	pool_index_lbp_pages_found - 发现的本地缓冲池索引页数
POOL_INDEX_GBP_INVALID_PAGES	BIGINT	pool_index_gbp_invalid_pages - 组缓冲池无效索引页数
POOL_XDA_GBP_L_READS	BIGINT	pool_xda_gbp_l_reads - 组缓冲池 XDA 数据逻辑读取请求数
POOL_XDA_GBP_P_READS	BIGINT	pool_xda_gbp_p_reads - 组缓冲池 XDA 数据物理读取请求数
POOL_XDA_LBP_PAGES_FOUND	BIGINT	pool_xda_lbp_pages_found - 发现的本地缓冲池 XDA 数据页数
POOL_XDA_GBP_INVALID_PAGES	BIGINT	pool_xda_gbp_invalid_pages - 组缓冲池无效 XDA 数据页数
EVMON_WAIT_TIME	BIGINT	evmon_wait_time - 事件监视器等待时间
EVMON_WAITS_TOTAL	BIGINT	evmon_waits_total - 事件监视器总等待次数
TOTAL_EXTENDED_LATCH_WAIT_TIME	BIGINT	total_extended_latch_wait_time - 扩展锁存器等待时间总计
TOTAL_EXTENDED_LATCH_WAITS	BIGINT	total_extended_latch_waits - 扩展锁存器等待总计
TOTAL_DISP_RUN_QUEUE_TIME	BIGINT	total_disp_run_queue_time - 分派器运行队列时间总计
POOL_QUEUED_ASYNC_DATA_REQS	BIGINT	pool_queued_async_data_reqs - 数据预取请求数
POOL_QUEUED_ASYNC_INDEX_REQS	BIGINT	pool_queued_async_index_reqs - 索引预取请求数
POOL_QUEUED_ASYNC_XDA_REQS	BIGINT	pool_queued_async_xda_reqs - XDA 预取请求数
POOL_QUEUED_ASYNC_TEMP_DATA_REQS	BIGINT	pool_queued_async_temp_data_reqs - 临时表空间数据预取请求数
POOL_QUEUED_ASYNC_TEMP_INDEX_REQS	BIGINT	pool_queued_async_temp_index_reqs - 临时表空间索引预取请求数
POOL_QUEUED_ASYNC_TEMP_XDA_REQS	BIGINT	pool_queued_async_temp_xda_reqs - 临时表空间 XDA 数据预取请求数
POOL_QUEUED_ASYNC_OTHER_REQS	BIGINT	pool_queued_async_other_reqs - 非预取请求数
POOL_QUEUED_ASYNC_DATA_PAGES	BIGINT	pool_queued_async_data_pages - 预取请求的数据页数
POOL_QUEUED_ASYNC_INDEX_PAGES	BIGINT	pool_queued_async_index_pages - 预取请求的索引页数
POOL_QUEUED_ASYNC_XDA_PAGES	BIGINT	pool_queued_async_xda_pages - 预取请求的 XDA 页数
POOL_QUEUED_ASYNC_TEMP_DATA_PAGES	BIGINT	pool_queued_async_temp_data_pages - 预取请求的临时表空间数据页数
POOL_QUEUED_ASYNC_TEMP_INDEX_PAGES	BIGINT	pool_queued_async_temp_index_pages - 预取请求的临时表空间索引页数
POOL_QUEUED_ASYNC_TEMP_XDA_PAGES	BIGINT	pool_queued_async_temp_xda_pages - 预取请求的临时表空间 XDA 数据页数
POOL_FAILED_ASYNC_DATA_REQS	BIGINT	pool_failed_async_data_reqs - 失败数据预取请求数
POOL_FAILED_ASYNC_INDEX_REQS	BIGINT	pool_failed_async_index_reqs - 失败的索引预取请求数
POOL_FAILED_ASYNC_XDA_REQS	BIGINT	pool_failed_async_xda_reqs - 失败的 XDA 预取请求数

表 50. 对活动事件监视器返回的信息: 表名: ACTIVITYMETRICS_evmon-name (续)

列名	数据类型	描述
POOL_FAILED_ASYNC_TEMP_DATA _REQS	BIGINT	pool_failed_async_temp_data_reqs - 临时表空间的失败数据预取请求数
POOL_FAILED_ASYNC_TEMP_INDEX _REQS	BIGINT	pool_failed_async_temp_index_reqs - 临时表空间的失败索引预取请求数
POOL_FAILED_ASYNC_TEMP_XDA_REQS	BIGINT	pool_failed_async_temp_xda_reqs - 临时表空间的失败 XDA 预取请求数
POOL_FAILED_ASYNC_OTHER_REQS	BIGINT	pool_failed_async_other_reqs - 失败的非预取请求数
TOTAL_PEDS	BIGINT	total_peds - 部分提前相异总数
DISABLED_PEDS	BIGINT	第 746 页的『disabled_peds -"已禁用部分提前相异数"监视元素』
POST_THRESHOLD_PEDS	BIGINT	post_threshold_peds - 部分提前相异数阈值
TOTAL_PEAS	BIGINT	total_peas - 部分提前聚集总数
POST_THRESHOLD_PEAS	BIGINT	post_threshold_peas - 部分提前聚集阈值
TQ_SORT_HEAP_REQUESTS	BIGINT	tq_sort_heap_requests - 表队列排序堆请求数
TQ_SORT_HEAP_REJECTIONS	BIGINT	tq_sort_heap_rejections - 表队列排序堆拒绝数
PREFETCH_WAIT_TIME	BIGINT	prefetch_wait_time - 等待预取的时间
PREFETCH_WAITS	BIGINT	prefetch_waits - 预取程序等待计数
POOL_DATA_GBP_INDEP_PAGES_FOUND _IN_LBP	BIGINT	第 975 页的『pool_data_gbp_indep_pages_found _in_lbp -"本地缓冲池中发现的独立于组缓冲池的数据页数"监视元素』
POOL_INDEX_GBP_INDEP_PAGES _FOUND_IN_LBP	BIGINT	第 1005 页的『pool_index_gbp_indep_pages _found_in_lbp -"本地缓冲池中发现的独立于组缓冲池的索引页数"监视元素』
POOL_XDA_GBP_INDEP_PAGES _FOUND_IN_LBP	BIGINT	第 1059 页的『pool_xda_gbp_indep_pages _found_in_lbp -"本地缓冲池中发现的独立于组缓冲池的 XDA 页数"监视元素』

表 51. 对活动事件监视器返回的信息: 表名: ACTIVITYVALS_evmon-name

列名	数据类型	描述
PARTITION_KEY	INTEGER	第 951 页的『partition_key -"分区键"监视元素』
ACTIVATE_TIMESTAMP	TIMESTAMP	activate_timestamp - 激活时间戳记
ACTIVITY_ID	BIGINT	activity_id - 活动标识
ACTIVITY_SECONDARY_ID	SMALLINT	activity_secondary_id - 活动辅助标识
APPL_ID	VARCHAR(64)	appl_id - 应用程序标识
PARTITION_NUMBER	SMALLINT	partition_number - 分区号
STMT_VALUE_DATA	CLOB	stmt_value_data - 值数据
STMT_VALUE_INDEX	INTEGER	stmt_value_index - 值索引
STMT_VALUE_ISNULL	INTEGER	stmt_value_isnull - 包含空值
STMT_VALUE_ISREOPT	INTEGER	stmt_value_isreopt - 用于语句重新优化的变量
STMT_VALUE_TYPE	CHARACTER(16)	stmt_value_type - 值类型
UOW_ID	INTEGER	uow_id - 工作单元标识

表 52. 对活动事件监视器返回的信息: 缺省表名: CONTROL_evmon-name

列名	数据类型	描述
PARTITION_KEY	INTEGER	第 951 页的『partition_key -“分区键”监视元素』
EVENT_MONITOR_NAME	VARCHAR(128)	event_monitor_name - 事件监视器名称
MESSAGE	VARCHAR(128)	message - 控制表消息
MESSAGE_TIME	TIMESTAMP	message_time - 时间戳记控制表消息
PARTITION_NUMBER	SMALLINT	partition_number - 分区号

访问活动事件监视器写至表的事件信息

活动事件监视器可将其输出写至表、文件和管道。

有关使用写至文件和管道的数据的更多信息，请参阅第 98 页的『事件监视器自描述数据流』。

开始之前

必须已创建并激活活动事件监视器，同时已启用数据收集。

关于此任务

第 81 页的『管理目标表、控制表和事件监视器表』中描述了活动事件监视器生成的表。在 DB2 V10.1 之前，度量值监视元素会写至 ACTIVITIES 表的 DETAIL_XML 列中的 XML 文档。该 XML 文档的模式包含在 sqllib/misc/DB2MonCommon.xsd 文件中，并且顶级元素是 activity_metrics。从 V10.1 开始，以前仅在 details_xml XML 文档中提供的度量值现在也会在活动事件监视器生成的 ACTIVITYMETRICS 表中提供。

过程

要访问活动事件监视器生成的数据，请执行以下操作:

1. 构造一个返回您想要查看的列的查询。 例如，如果您关心与特定工作单元相关联的语句的文本的信息，那么可构造类似如下的查询:

   ```
   SELECT UOW_ID, SUBSTR(STMT_TEXT, 1,70) AS STMT_TEXT FROM ACTIVITYSTMT_ACTEVMON
     WHERE UOW_ID=11
   ```

   在此情况下，事件监视器名为 actevmon。
2. 运行此查询。 以上查询可能返回类似如下的结果:

   ```
   UOW_ID      STMT_TEXT
   --------- ------------------------------------------------------------------------
          11 select * from gosaleshr.employee_expense_detail order by expense_date

     1 record(s) selected.
   ```

结果

示例

如果要访问 ACTIVITY 表的 DETAILS_XML 列中的数据，可将随 DB2 产品提供的任何界面用于此用途。例如，要查看活动事件监视器为工作单元收集的度量值信息，可使用类似如下的语句:

```
SELECT  SUBSTR(B.METRIC_NAME, 1, 20) METRIC_NAME, B.VALUE
   FROM ACTIVITY_ACTEVMON AS A,
   TABLE(MON_FORMAT_XML_METRICS_BY_ROW(A.DETAILS_XML))AS B
   WHERE UOW_ID=23
   ORDER BY B.VALUE DESC
```

此语句返回为 UOW_ID 为 23 的工作单元收集的所有活动度量值:

```
METRIC_NAME               VALUE
------------------- --------------------
TOTAL_CPU_TIME                    140625
ROWS_READ                            977
TOTAL_ACT_TIME                       880
STMT_EXEC_TIME                       880
COORD_STMT_EXEC_TIME                 880
TOTAL_SECTION_PROC_T                 880
TOTAL_SECTION_TIME                   880

⋮

⋮
FCM_TQ_SEND_WAITS_TO                   0
FCM_MESSAGE_SEND_WAI                   0
FCM_SEND_WAITS_TOTAL                   0
FCM_RECV_WAITS_TOTAL                   0

  92 record(s) selected.
```

有关如何处理事件监视器返回的 XML 数据的更多信息，请参阅第 12 页的『在 XML 文档中返回监视数据的接口』。

示例: 捕获与语句的执行相关的活动信息

如果发现某个语句的执行时间很长，那么可定义阈值以在超出该阈值时让活动事件监视器捕获有关该语句的执行的信息。

然后，可将语句执行信息与活动事件监视器收集的信息相关联来查看活动度量值，它们可帮助您了解可能导致速度变慢的原因。

开始之前

在捕获活动信息之前，必须标识所讨论的语句；例如，用户或应用程序开发者可能抱怨特定语句的运行时间比预期长。或者，可使用程序包高速缓存事件监视器来标识运行时间较长的语句。

关于此任务

在此示例中，要调查的查询作为应用程序的一部分运行。查询如下所示:

```
SELECT DISTINCT PARTS_BIN FROM STOCK WHERE PART_NUMBER = ?
```

速度变慢的一个可能原因是数据分发不适宜。例如，如果 STOCK 表只有几行对应大部分部件号，但有几千行对应一个特定部件号，那么它会花较长时间来运行此 SELECT 语句。以下示例说明如何检索与先前查询相关联的活动针对参数标记（“?”）处理的实际值。

过程

要测试不适宜数据分发导致查询运行变慢的假设，可为所提到的语句创建阈值。然后，可使用阈值和活动事件监视器来捕获有关该特定语句的执行的信息。可通过此信息确定运行时间超过预期的查询处理的实际值。

1. 为所讨论的语句创建阈值，指定该语句运行超过 10 秒时发生的阈值违例事件:

```
CREATE THRESHOLD TH1
    FOR STATEMENT TEXT 'SELECT DISTINCT PARTS_BIN
    FROM STOCK WHERE PART_NUMBER = ?' ACTIVITIES
    ENFORCEMENT DATABASE
    WHEN ACTIVITYTOTALTIME > 10 SECONDS
    COLLECT ACTIVITY DATA WITH DETAILS, SECTION AND VALUES
    CONTINUE
```

2. 创建阈值事件监视器来记录阈值违例:

```
CREATE EVENT MONITOR STMT_THRESH_VIOLATIONS
FOR THRESHOLD VIOLATIONS
WRITE TO TABLE
                        AUTOSTART
```

3. 创建活动事件监视器来记录详细活动信息:

```
CREATE EVENT MONITOR ACTIVITIES
FOR ACTIVITIES
WRITE TO TABLE
```

4. 启用新事件监视器:

```
SET EVENT MONITOR ACTIVITIES STATE 1
SET EVENT MONITOR STMT_THRESH_VIOLATIONS STATE 1
```

5. 运行执行此语句的应用程序。如果发生阈值违例，那么阈值违例事件监视器 STMT_THRESH_VIOLATIONS 会记录有关阈值违例的信息；与阈值违例相关联的活动的信息由活动事件监视器 ACTIVITIES 记录。

6. 要确定是否发生了阈值违例，请查询阈值事件监视器针对步骤 1 中定义的阈值 TH1 记录的违例次数。要执行此查询，请连接视图 SYSCAT.THRESHOLDS 与阈值事件监视器生成的包含阈值违例信息的表。此连接是必需的，因为阈值名称 TH1 包含在 SYSCAT.THRESHOLDS 中:

```
SELECT COUNT(1) NUM_VIOLATIONS
        FROM THRESHOLDVIOLATIONS_DB2THRESHOLDVIOLATIONS T
        JOIN SYSCAT.THRESHOLDS S ON T.THRESHOLDID = S.THRESHOLDID
        WHERE S.THRESHOLDNAME = 'TH1';

NUM_VIOLATIONS
--------------
                        1

  1 record(s) selected.
```

   在此情况下，有一个阈值违例；步骤 1 中标识的语句的一次执行时间超过 10 秒。

7. 检查您在 1 中标识的语句内的参数标记 (?) 表示的数据（部件号）。 在以下示例中，SELECT 语句从活动事件监视器生成的其中一个 ACTIVITYVALS 表中检索参数标记（由下面的 SQL 中的 STMT_VALUE_DATA 表示）的值:

```
SELECT SUBSTR(V.STMT_VALUE_DATA, 1, 80) PARAM_MARKER_VALUE
        FROM ACTIVITYVALS_ACTIVITIES V
        JOIN THRESHOLDVIOLATIONS_STMT_THRESH_VIOLATIONS T
          ON T.APPL_ID = V.APPL_ID
          AND T.UOW_ID = V.UOW_ID
```

```
  AND T.ACTIVITY_ID = V.ACTIVITY_ID
JOIN SYSCAT.THRESHOLDS S
  ON T.THRESHOLDID = S.THRESHOLDID
WHERE S.THRESHOLDNAME = 'TH1';
```

在先前示例中，SELECT 语句从活动事件监视器生成的某个表中检索参数标记 (STMT_VALUE_DATA) 的值。

```
PARAM_MARKER_VALUE
------------------------------------------------------------------------------
475299
```

8. 既然您知道与长时间运行的语句相关联的 PART_NUMBER 的值，那么可检查 STOCK 表以了解表中是否存在任何与该部件号出现次数有关的因素可能导致查询时间变长。例如，包含 475299 作为 PART_NUBMER 的值的许多行（与其他部件号的行数相比）可能是查询运行时间变长的原因（如果遇到此值）。

变体: 通过使用可执行标识来为语句定义阈值

在以上示例中，该阈值在步骤 第 245 页的1 中通过使用该语句的实际文本显式标识。还可间接定义阈值，以标识程序包高速缓存中包含的语句的可执行标识。例如，可按如下所示定义阈值:

```
CREATE THRESHOLD TH1
   FOR STATEMENT REFERENCE
     x'0100000000000000002000000000000000000000000200201003041621585848 50' ACTIVITIES
   ENFORCEMENT DATABASE
   WHEN ACTIVITYTOTALTIME > 10 SECONDS
   COLLECT ACTIVITY DATA WITH DETAILS, SECTION AND VALUES
   CONTINUE;
```

在此示例中，跟在关键字 STATEMENT REFERENCE 后面的可执行标识用于在程序包高速缓存中查询相应语句文本。可通过检查程序包高速缓存来确定语句的可执行标识。有关如何查看程序包高速缓存中包含的信息的更多信息（包括语句的可执行标识），请参阅第 223 页的『使用程序包高速缓存信息确定用于进行性能调整的候选语句』。

如果在程序包高速缓存中发现可执行标识，那么会从程序包高速缓存中检索关联语句文本，并且此文本用于定义语句阈值。对于静态 SQL 片段中的语句，如果该可执行标识不在程序包高速缓存中，那么会从系统目录中检索语句文本。对于动态 SQL 片段中的语句，考虑使用 PREPARE 语句以通过语句字符串创建预编译语句。如果在程序包高速缓存或系统目录中找不到该可执行标识，那么会返回错误 (SQL4721N)。

使用统计信息事件监视器来捕获系统度量值

统计信息事件监视器提供可用于度量系统操作的不同方面的数据。

可使用统计信息事件监视器来收集两种类型的信息:

系统度量值

*系统度量值*是用于捕获有关系统的度量信息的请求监视元素。它们包括所花时间监视元素以及充当计数器的监视元素，例如，监视元素 deadlocks。可配置对整个数据库或特定服务类收集这些度量值。系统度量值监视元素中的值通常从 0 开始，并继续累积直到下次数据库激活。系统度量值作为 event_scstats 和 event_wlstats 逻辑数据组中 details_xml 监视元素的一部分收集。

工作负载管理统计信息

系统为工作负载管理器对象（包括服务类、工作类、工作负载和阈值队列）保留统计信息。这些统计信息驻留在内存中，并且可使用工作负载管理器统计信息表函数来实时查看，或者可收集统计信息并将其发送至统计信息事件监视器，以后可在此事件监视器中查看这些统计信息来分析历史。缺省情况下，为每个工作负载管理器对象收集最低级别的一组统计信息。可使用针对各种工作负载管理器对象的 CREATE 或 ALTER 语句的子句来修改统计信息收集的作用域。每个收集时间间隔后，工作负载管理统计信息重置为 0。

统计信息事件监视器收集的度量值与 MON_GET_SERVICE_SUBCLASS_DETAILS 和 MON_GET_WORKLOAD_DETAILS 表函数报告的度量值是同一组度量值。但是，请注意，除系统度量值监视元素以外，这两个表函数返回的表的 DETAILS 列中的 XML 文档还包含许多其他监视元素）。

第 248 页的『统计信息事件监视器写至表的信息』中描述了写至表统计信息事件监视器生成的表。这些系统度量值在 XML 文档 details_xml 的 SCSTATS 和 WLSTATS 表的 ETAILS_XML 列中收集。这些 XML 文档包括反映各种不同系统度量的一些度量值。文件 `sqllib/misc/DB2MonCommon.xsd` 包含 DETAILS_XML 列中返回的 XML 文档的模式。顶层元素是 system_metrics。

切记: 针对统计信息事件监视器在 details_xml 文档中生成的度量值监视元素返回的数据会从数据库激活起**持续**累积。相反，统计信息监视元素生成的表的其他列中出现的工作负载管理统计信息监视元素在每个收集时间间隔会**重置为 0**。

统计信息事件监视器收集的系统度量值监视元素通过两个机制控制:

- 要收集特定服务类的度量值，可通过指定 CREATE SERVICE CLASS 或 ALTER SERVICE CLASS 语句的 COLLECT REQUEST METRICS 子句来收集系统度量值。
- 要收集整个数据库的度量值，请使用 **mon_req_metrics** 数据库配置参数。例如，要收集 BASE 级别度量值，请使用以下命令:

```
db2 update db cfg using mon_req_metrics base
```

仅当请求由服务子类（其父代服务超类已启用请求监视元素收集）中的代理程序处理或已对整个数据库启用系统度量值收集时，才会针对请求收集系统度量值监视元素。如果已在数据库级别禁用系统度量值收集，并且对服务超类禁用了系统度量值收集，那么 DETAILS_XML 文档中报告的度量值将停止增加（或保持为零，如果在数据库激活时禁用了请求度量值）。

统计信息事件监视器生成的数据

可选择将统计信息事件监视器的输出写至常规表、文件或命名管道。

不管您选择什么输出格式，统计信息事件监视器捕获的所有数据都来自下列五个逻辑数据组中的一个:

- event_scstats
- event_wcstats
- event_wlstats
- event_qstats
- event_histogrambin

如果选择将统计信息事件监视器数据写至常规表，那么另一个组 (CONTROL) 中的数据用于生成有关事件监视器本身的元数据。

统计信息事件监视器写至表的信息:

在指定了 WRITE TO TABLE 选项的情况下统计信息事件监视器写入的信息。

选择 WRITE TO TABLE 作为统计信息事件监视器的输出类型时，缺省情况下，会生成五个表，每个表包含一个或多个逻辑数据组中的监视元素:

表 53. STATISTICS 写至表事件监视器生成的表. 表名通过逻辑数据组（用于填充该表）的名称和 CREATE EVENT MONITOR 语句中给予事件监视器的名称（如下表的表名中的 *evmon-name* 所示）并置派生。

缺省表名	报告的逻辑数据组
QSTATS_*evmon-name*	第 57 页的『event_qstats 逻辑数据组』
SCSTATS_*evmon-name*	第 57 页的『event_scstats 逻辑数据组』
HISTOGRAMBIN_*evmon-name*	第 56 页的『event_histogrambin 逻辑数据组』
WCSTATS_*evmon-name*	第 62 页的『event_wcstats 逻辑数据组』
WLSTATS_*evmon-name*	第 63 页的『event_wlstats 逻辑数据组』
CONTROL_*evmon-name*	CONTROL 逻辑组由来自 event_dbheader、event_start 和 event_overflow 逻辑数据组中的一个或多个的所选元素组成。

要将事件监视器的输出限制为发送至特定表，请在 CREATE EVENT MONITOR 或 ALTER EVENT MONITOR 语句中指定要对其生成表的逻辑组的名称。有关详细信息，请参阅这些语句的参考主题。

生成的表

表 54. 对统计信息事件监视器返回的信息: 缺省表名: QSTATS_evmon-name

列名	数据类型	描述
PARTITION_KEY	INTEGER	第 951 页的『partition_key -“分区键”监视元素』
LAST_WLM_RESET	TIMESTAMP	last_wlm_reset - 最后一次重置时间
MON_INTERVAL_ID	BIGINT	mon_interval_id - 监视时间间隔标识
PARTITION_NUMBER	SMALLINT	partition_number - 分区号
QUEUE_ASSIGNMENTS_TOTAL	BIGINT	queue_assignments_total - 队列分配总次数
QUEUE_SIZE_TOP	INTEGER	queue_size_top - 最大队列大小
QUEUE_TIME_TOTAL	BIGINT	queue_time_total - 总队列时间
SERVICE_SUBCLASS_NAME	VARCHAR(128)	service_subclass_name - 服务子类名
SERVICE_SUPERCLASS_NAME	VARCHAR(128)	service_superclass_name - 服务超类名
STATISTICS_TIMESTAMP	TIMESTAMP	statistics_timestamp - 统计信息时间戳记
THRESHOLD_DOMAIN	VARCHAR(64)	threshold_domain - 阈值域
THRESHOLD_NAME	VARCHAR(128)	threshold_name - 阈值名称
THRESHOLD_PREDICATE	VARCHAR(64)	threshold_predicate - 阈值谓词
THRESHOLDID	INTEGER	thresholdid - 阈值标识
WORK_ACTION_SET_NAME	VARCHAR(128)	work_action_set_name - 工作操作集名称
WORK_CLASS_NAME	VARCHAR(128)	work_class_name - 工作类名

表 55. 对统计信息事件监视器返回的信息: 表名: SCSTATS_evmon-name

列名	数据类型	描述
PARTITION_KEY	INTEGER	第 951 页的『partition_key -"分区键"监视元素』
ACT_CPU_TIME_TOP	BIGINT	act_cpu_time_top - 最长活动 CPU 时间
ACT_REMAPPED_IN	BIGINT	act_remapped_in - 重新映入的活动数
ACT_REMAPPED_OUT	BIGINT	act_remapped_out - 重新映出的活动数
ACT_ROWS_READ_TOP	BIGINT	act_rows_read_top - 最高活动读取行数
ACT_THROUGHPUT	BIGINT	act_throughput - 活动吞吐量
AGG_TEMP_TABLESPACE_TOP	BIGINT	agg_temp_tablespace_top - 最大聚集临时表空间
APP_ACT_ABORTED_TOTAL	BIGINT	app_act_aborted_total - 失败的外部协调程序活动总数
APP_ACT_COMPLETED_TOTAL	BIGINT	app_act_completed_total - 成功的外部协调程序活动总数
APP_ACT_REJECTED_TOTAL	BIGINT	app_act_rejected_total - 拒绝的外部协调程序活动总数
CONCURRENT_ACT_TOP	INTEGER	concurrent_act_top - 最高并行活动数
CONCURRENT_CONNECTION_TOP	INTEGER	concurrent_connection_top - 最高并行连接数
CONCURRENT_WLO_TOP	INTEGER	concurrent_wlo_top - 最大并行工作负载项数
COORD_ACT_ABORTED_TOTAL	BIGINT	coord_act_aborted_total - 异常终止的协调程序活动总数
COORD_ACT_COMPLETED_TOTAL	BIGINT	coord_act_completed_total - 完成的协调程序活动总数
COORD_ACT_EST_COST_AVG	BIGINT	coord_act_est_cost_avg - 平均协调程序活动估计成本
COORD_ACT_EXEC_TIME_AVG	BIGINT	coord_act_exec_time_avg - 平均协调程序活动执行时间
COORD_ACT_INTERARRIVAL_TIME_AVG	BIGINT	coord_act_interarrival_time_avg - 平均协调程序活动到达时间
COORD_ACT_LIFETIME_AVG	BIGINT	coord_act_lifetime_avg - 平均协调程序活动生存期
COORD_ACT_LIFETIME_TOP	BIGINT	coord_act_lifetime_top - 协调程序活动生存期项部
COORD_ACT_QUEUE_TIME_AVG	BIGINT	coord_act_queue_time_avg - 平均协调程序活动队列时间
COORD_ACT_REJECTED_TOTAL	BIGINT	coord_act_rejected_total - 被拒绝的协调程序活动总数
COST_ESTIMATE_TOP	BIGINT	cost_estimate_top - 最高估计成本
CPU_UTILIZATION	BIGINT	cpu_utilization - CPU 利用率
DETAILS_XML	BLOB(0)	**注**: 在此文档中为超类 SYSDEFAULTSYSTEMCLASS 下的缺省子类 SYSDEFAULTSUBCLASS 报告的度量的值为 0。
LAST_WLM_RESET	TIMESTAMP	last_wlm_reset - 最后一次重置时间
MON_INTERVAL_ID	BIGINT	mon_interval_id - 监视时间间隔标识
PARTITION_NUMBER	SMALLINT	partition_number - 分区号
REQUEST_EXEC_TIME_AVG	BIGINT	request_exec_time_avg - 平均请求执行时间
ROWS_RETURNED_TOP	BIGINT	rows_returned_top - 最高实际返回行数
SERVICE_CLASS_ID	INTEGER	service_class_id - 服务类标识
SERVICE_SUBCLASS_NAME	VARCHAR(128)	service_subclass_name - 服务子类名
SERVICE_SUPERCLASS_NAME	VARCHAR(128)	service_superclass_name - 服务超类名
STATISTICS_TIMESTAMP	TIMESTAMP	statistics_timestamp - 统计信息时间戳记
TEMP_TABLESPACE_TOP	BIGINT	temp_tablespace_top - 最大临时表空间
TOTAL_CPU_TIME	BIGINT	total_cpu_time - CPU 时间总计
TOTAL_DISP_RUN_QUEUE_TIME	BIGINT	total_disp_run_queue_time - 分派器运行队列时间总计

表 55. 对统计信息事件监视器返回的信息：表名：SCSTATS_evmon-name（续）

列名	数据类型	描述
UOW_COMPLETED_TOTAL	BIGINT	uow_completed_total - 完成的工作单元总数
UOW_LIFETIME_AVG	BIGINT	uow_lifetime_avg - 工作单元平均生存期
UOW_THROUGHPUT	BIGINT	uow_throughput - 工作单元吞吐量
UOW_TOTAL_TIME_TOP	BIGINT	uow_total_time_top - 最长 UOW 时间总计

表 56. 对统计信息事件监视器返回的信息：表名：HISTOGRAMBIN_evmon-name

列名	数据类型	描述
PARTITION_KEY	INTEGER	第 951 页的『partition_key -“分区键”监视元素』
BIN_ID	INTEGER	bin_id - 直方图条形标识
BOTTOM	BIGINT	bottom - 直方图类别底部
HISTOGRAM_TYPE	VARCHAR(64)	histogram_type - 直方图类型
MON_INTERVAL_ID	BIGINT	mon_interval_id - 监视时间间隔标识
NUMBER_IN_BIN	BIGINT	number_in_bin - 条形中的数目
PARTITION_NUMBER	SMALLINT	partition_number - 分区号
SERVICE_CLASS_ID	INTEGER	service_class_id - 服务类标识
STATISTICS_TIMESTAMP	TIMESTAMP	statistics_timestamp - 统计信息时间戳记
TOP	BIGINT	top - 最多直方图类别
WORK_ACTION_SET_ID	INTEGER	work_action_set_id - 工作操作集标识
WORK_CLASS_ID	INTEGER	work_class_id - 工作类标识
WORKLOAD_ID	INTEGER	workload_id - 工作负载标识

表 57. 对统计信息事件监视器返回的信息：表名：WCSTATS_evmon-name

列名	数据类型	描述
PARTITION_KEY	INTEGER	第 951 页的『partition_key -“分区键”监视元素』
ACT_CPU_TIME_TOP	BIGINT	act_cpu_time_top - 最长活动 CPU 时间
ACT_ROWS_READ_TOP	BIGINT	act_rows_read_top - 最高活动读取行数
ACT_TOTAL	BIGINT	act_total - 活动总数
COORD_ACT_EST_COST_AVG	BIGINT	coord_act_est_cost_avg - 平均协调程序活动估计成本
COORD_ACT_EXEC_TIME_AVG	BIGINT	coord_act_exec_time_avg - 平均协调程序活动执行时间
COORD_ACT_INTERARRIVAL_TIME_AVG	BIGINT	coord_act_exec_time_avg - 平均协调程序活动执行时间
COORD_ACT_LIFETIME_AVG	BIGINT	coord_act_lifetime_avg - 平均协调程序活动生存期
COORD_ACT_LIFETIME_TOP	BIGINT	coord_act_lifetime_top - 协调程序活动生存期项部
COORD_ACT_QUEUE_TIME_AVG	BIGINT	coord_act_queue_time_avg - 平均协调程序活动队列时间
COST_ESTIMATE_TOP	BIGINT	cost_estimate_top - 最高估计成本
LAST_WLM_RESET	TIMESTAMP	last_wlm_reset - 最后一次重置时间
MON_INTERVAL_ID	BIGINT	mon_interval_id - 监视时间间隔标识
PARTITION_NUMBER	SMALLINT	partition_number - 分区号
ROWS_RETURNED_TOP	BIGINT	rows_returned_top - 最高实际返回行数
STATISTICS_TIMESTAMP	TIMESTAMP	statistics_timestamp - 统计信息时间戳记
TEMP_TABLESPACE_TOP	BIGINT	temp_tablespace_top - 最大临时表空间

表 57. 对统计信息事件监视器返回的信息：表名：WCSTATS_evmon-name（续）

列名	数据类型	描述
WORK_ACTION_SET_ID	INTEGER	work_action_set_id - 工作操作集标识
WORK_ACTION_SET_NAME	VARCHAR(128)	work_action_set_name - 工作操作集名称
WORK_CLASS_ID	INTEGER	work_class_id - 工作类标识
WORK_CLASS_NAME	VARCHAR(128)	work_class_name - 工作类名

表 58. 对统计信息事件监视器返回的信息：表名：WLSTATS_evmon-name

列名	数据类型	描述
PARTITION_KEY	INTEGER	第 951 页的『partition_key -“分区键”监视元素』
ACT_CPU_TIME_TOP	BIGINT	act_cpu_time_top - 最长活动 CPU 时间
ACT_ROWS_READ_TOP	BIGINT	act_rows_read_top - 最高活动读取行数
ACT_THROUGHPUT	BIGINT	act_throughput - 活动吞吐量
APP_ACT_ABORTED_TOTAL	BIGINT	app_act_aborted_total - 失败的外部协调程序活动总数
APP_ACT_COMPLETED_TOTAL	BIGINT	app_act_completed_total - 成功的外部协调程序活动总数
APP_ACT_REJECTED_TOTAL	BIGINT	app_act_rejected_total - 拒绝的外部协调程序活动总数
CONCURRENT_WLO_ACT_TOP	INTEGER	concurrent_wlo_act_top - 最大并行 WLO 活动数
CONCURRENT_WLO_TOP	INTEGER	concurrent_wlo_top - 最大并行工作负载项数
COORD_ACT_ABORTED_TOTAL	BIGINT	coord_act_aborted_total - 异常终止的协调程序活动总数
COORD_ACT_COMPLETED_TOTAL	BIGINT	coord_act_completed_total - 完成的协调程序活动总数
COORD_ACT_EST_COST_AVG	BIGINT	coord_act_est_cost_avg - 平均协调程序活动估计成本
COORD_ACT_EXEC_TIME_AVG	BIGINT	coord_act_exec_time_avg - 平均协调程序活动执行时间
COORD_ACT_INTERARRIVAL_TIME_AVG	BIGINT	第 690 页的『coord_act_interarrival_time_avg -“平均协调程序活动到达时间”监视元素』
COORD_ACT_LIFETIME_AVG	BIGINT	coord_act_lifetime_avg - 平均协调程序活动生存期
COORD_ACT_LIFETIME_TOP	BIGINT	coord_act_lifetime_top - 协调程序活动生存期项部
COORD_ACT_QUEUE_TIME_AVG	BIGINT	coord_act_queue_time_avg - 平均协调程序活动队列时间
COORD_ACT_REJECTED_TOTAL	BIGINT	coord_act_rejected_total - 被拒绝的协调程序活动总数
COST_ESTIMATE_TOP	BIGINT	cost_estimate_top - 最高估计成本
CPU_UTILIZATION	BIGINT	cpu_utilization - CPU 利用率
DETAILS_XML	BLOB(0)	
LAST_WLM_RESET	TIMESTAMP	last_wlm_reset - 最后一次重置时间
LOCK_WAIT_TIME_GLOBAL_TOP	BIGINT	lock_wait_time_global_top - 最长全局锁定等待时间
LOCK_WAIT_TIME_TOP	BIGINT	lock_wait_time_top - 最长锁定等待时间
MON_INTERVAL_ID	BIGINT	mon_interval_id - 监视时间间隔标识
PARTITION_NUMBER	SMALLINT	partition_number - 分区号
ROWS_RETURNED_TOP	BIGINT	rows_returned_top - 最高实际返回行数
STATISTICS_TIMESTAMP	TIMESTAMP	statistics_timestamp - 统计信息时间戳记
TEMP_TABLESPACE_TOP	BIGINT	temp_tablespace_top - 最大临时表空间
TOTAL_CPU_TIME	BIGINT	total_cpu_time - CPU 时间总计
TOTAL_DISP_RUN_QUEUE_TIME	BIGINT	total_disp_run_queue_time - 分派器运行队列时间总计
UOW_COMPLETED_TOTAL	BIGINT	uow_completed_total - 完成的工作单元总数

表 58. 对统计信息事件监视器返回的信息: 表名: WLSTATS_evmon-name (续)

列名	数据类型	描述
UOW_LIFETIME_AVG	BIGINT	uow_lifetime_avg - 工作单元平均生存期
UOW_THROUGHPUT	BIGINT	uow_throughput - 工作单元吞吐量
UOW_TOTAL_TIME_TOP	BIGINT	uow_total_time_top - 最长 UOW 时间总计
WLO_COMPLETED_TOTAL	BIGINT	wlo_completed_total - 完成的工作负载项总数
WORKLOAD_ID	INTEGER	workload_id - 工作负载标识
WORKLOAD_NAME	VARCHAR(128)	workload_name - 工作负载名称

表 59. 对统计信息事件监视器返回的信息: 缺省表名: CONTROL_evmon-name

列名	数据类型	描述
PARTITION_KEY	INTEGER	第 951 页的『partition_key -"分区键"监视元素』
EVENT_MONITOR_NAME	VARCHAR(128)	event_monitor_name - 事件监视器名称
MESSAGE	VARCHAR(128)	message - 控制表消息
MESSAGE_TIME	TIMESTAMP	message_time - 时间戳记控制表消息
PARTITION_NUMBER	SMALLINT	partition_number - 分区号

为 system_metrics 和 activity_metrics 监视元素写入 XML 的信息:

在 MON_GET_ACTIVITY_DETAILS 表函数、MON_GET_PKG_CACHE_STMT_DETAILS 表函数和活动事件监视器中报告了 **activity_metrics** 监视元素。在 MON_GET_CONNECTION_DETAILS、MON_GET_UNIT_OF_WORK_DETAILS、MON_GET_SERVICE_SUBCLASS_DETAILS、MON_GET_WORKLOAD_DETAILS 表函数和统计信息事件监视器中报告了 **system_metrics** 监视元素。sqllib/misc/DB2MonCommon.xsd 文件中也记录了此信息。

system_metrics

系统级度量值。

元素内容: (第 274 页的『wlm_queue_time_total』, 第 275 页的『wlm_queue_assignments_total』, 第 275 页的『fcm_tq_recv_wait_time』, 第 275 页的『fcm_message_recv_wait_time』, 第 275 页的『fcm_tq_send_wait_time』, 第 276 页的『fcm_message_send_wait_time』, 第 257 页的『agent_wait_time』, 第 258 页的『agent_waits_total』, 第 276 页的『lock_wait_time』, 第 276 页的『lock_waits』, 第 276 页的『direct_read_time』, 第 277 页的『direct_read_reqs』, 第 277 页的『direct_write_time』, 第 277 页的『direct_write_reqs』, 第 277 页的『log_buffer_wait_time』, 第 278 页的『num_log_buffer_full』, 第 278 页的『log_disk_wait_time』, 第 278 页的『log_disk_waits_total』, 第 258 页的『tcpip_recv_wait_time』, 第 258 页的『tcpip_recvs_total』, 第 258 页的『client_idle_wait_time』, 第 259 页的『ipc_recv_wait_time』, 第 259 页的『ipc_recvs_total』, 第 259 页的『ipc_send_wait_time』, 第 259 页的『ipc_sends_total』, 第 260 页的『tcpip_send_wait_time』, 第 260 页的『tcpip_sends_total』, 第 278 页的『pool_write_time』, 第 279 页的『pool_read_time』, 第 279 页的『audit_file_write_wait_time』, 第 279 页的『audit_file_writes_total』, 第 279 页的『audit_subsystem_wait_time』, 第 280 页的『audit_subsystem_waits_total』, 第 280 页的

的『diaglog_write_wait_time』，第 280 页的『diaglog_writes_total』，第 280 页的『fcm_send_wait_time』，第 281 页的『fcm_recv_wait_time』，第 260 页的『total_wait_time』，第 260 页的『total_rqst_time』，第 261 页的『rqsts_completed_total』，第 261 页的『total_app_rqst_time』，第 261 页的『app_rqsts_completed_total』，第 281 页的『total_section_sort_proc_time』，第 281 页的『total_section_sort_time』，第 282 页的『total_section_sorts』，第 282 页的『rows_read』，第 282 页的『rows_modified』，第 282 页的『pool_data_l_reads』，第 283 页的『pool_index_l_reads』，第 283 页的『pool_temp_data_l_reads』，第 283 页的『pool_temp_index_l_reads』，第 283 页的『pool_xda_l_reads』，第 284 页的『pool_temp_xda_l_reads』，第 284 页的『total_cpu_time』，第 261 页的『act_completed_total』，第 284 页的『pool_data_p_reads』，第 284 页的『pool_temp_data_p_reads』，第 285 页的『pool_xda_p_reads』，第 285 页的『pool_temp_xda_p_reads』，第 285 页的『pool_index_p_reads』，第 285 页的『pool_temp_index_p_reads』，第 286 页的『pool_data_writes』，第 286 页的『pool_xda_writes』，第 286 页的『pool_index_writes』，第 286 页的『direct_reads』，第 287 页的『direct_writes』，第 287 页的『rows_returned』，第 287 页的『死锁』，第 287 页的『lock_timeouts』，第 288 页的『lock_escals』，第 288 页的『fcm_sends_total』，第 288 页的『fcm_recvs_total』，第 288 页的『fcm_send_volume』，第 289 页的『fcm_recv_volume』，第 289 页的『fcm_message_sends_total』，第 289 页的『fcm_message_recvs_total』，第 289 页的『fcm_message_send_volume』，第 290 页的『fcm_message_recv_volume』，第 290 页的『fcm_tq_sends_total』，第 290 页的『fcm_tq_recvs_total』，第 290 页的『fcm_tq_send_volume』，第 291 页的『fcm_tq_recv_volume』，第 291 页的『tq_tot_send_spills』，第 262 页的『tcpip_send_volume』，第 262 页的『tcpip_recv_volume』，第 262 页的『ipc_send_volume』，第 262 页的『ipc_recv_volume』，第 291 页的『post_threshold_sorts』，第 291 页的『post_shrthreshold_sorts』，第 292 页的『sort_overflows』，第 292 页的『audit_events_total』，第 263 页的『total_rqst_mapped_in』{zero or one times (?)}，第 263 页的『total_rqst_mapped_out』{zero or one times (?)}，第 263 页的『act_rejected_total』，第 263 页的『act_aborted_total』，第 292 页的『total_sorts』，第 294 页的『total_routine_time』，第 264 页的『total_compile_proc_time』，第 264 页的『total_compile_time』，第 264 页的『total_compilations』，第 264 页的『total_implicit_compile_proc_time』，第 265 页的『total_implicit_compile_time』，第 265 页的『total_implicit_compilations』，第 265 页的『total_runstats_proc_time』，第 265 页的『total_runstats_time』，第 266 页的『total_runstats』，第 266 页的『total_reorg_proc_time』，第 266 页的『total_reorg_time』，第 266 页的『total_reorgs』，第 267 页的『total_load_proc_time』，第 267 页的『total_load_time』，第 267 页的『total_loads』，第 293 页的『total_section_proc_time』，第 293 页的『total_section_time』，第 294 页的『total_app_section_executions』，第 267 页的『total_commit_proc_time』，第 268 页的『total_commit_time』，第 268 页的『total_app_commits』，第 268 页的『total_rollback_proc_time』，第 268 页的『total_rollback_time』，第 269 页的『total_app_rollbacks』，第 294 页的『total_routine_user_code_proc_time』，第 294 页的『total_routine_user_code_time』，第 295 页的『thresh_violations』，第 295 页的『num_lw_thresh_exceeded』，第 295 页的『total_routine_invocations』，第 269 页的『int_commits』，第 269 页的『int_rollbacks』，第 269 页的『cat_cache_inserts』，第 270 页的『cat_cache_lookups』，第 270 页的『pkg_cache_inserts』，第 270 页的『pkg_cache_lookups』，第 270 页的『act_rqsts_total』，第 281 页的『total_act_wait_time』，第 282 页的『total_act_time』，第 295 页的『lock_wait_time_global』，第 296 页的『lock_waits_global』，第 296 页的『reclaim_wait_time』，第 296 页的『spacemappage_reclaim_wait_time』，第 296 页的『lock_timeouts_global』，第 297 页的

『lock_escals_maxlocks』，第 297 页的『lock_escals_locklist』，第 297 页的『lock_escals_global』，第 297 页的『cf_wait_time』，第 298 页的『cf_waits』，第 298 页的『pool_data_gbp_l_reads』，第 298 页的『pool_data_gbp_p_reads』，第 298 页的『pool_data_lbp_pages_found』，第 299 页的『pool_data_gbp_invalid_pages』，第 299 页的『pool_index_gbp_l_reads』，第 299 页的『pool_index_gbp_p_reads』，第 299 页的『pool_index_lbp_pages_found』，第 300 页的『pool_index_gbp_invalid_pages』，第 300 页的『pool_xda_gbp_l_reads』，第 300 页的『pool_xda_gbp_p_reads』，第 300 页的『pool_xda_lbp_pages_found』，第 301 页的『pool_xda_gbp_invalid_pages』，第 301 页的『evmon_wait_time』，第 301 页的『evmon_waits_total』，第 301 页的『total_extended_latch_wait_time』，第 302 页的『total_extended_latch_waits』，第 271 页的『total_stats_fabrication_proc_time』，第 271 页的『total_stats_fabrication_time』，第 271 页的『total_stats_fabrications』，第 271 页的『total_sync_runstats_proc_time』，第 272 页的『total_sync_runstats_time』，第 272 页的『total_sync_runstats』，第 302 页的『total_disp_run_queue_time』，第 302 页的『pool_queued_async_data_reqs』，第 302 页的『pool_queued_async_index_reqs』，第 303 页的『pool_queued_async_xda_reqs』，第 303 页的『pool_queued_async_temp_data_reqs』，第 303 页的『pool_queued_async_temp_index_reqs』，第 303 页的『pool_queued_async_temp_xda_reqs』，第 304 页的『pool_queued_async_other_reqs』，第 304 页的『pool_queued_async_data_pages』，第 304 页的『pool_queued_async_index_pages』，第 304 页的『pool_queued_async_xda_pages』，第 305 页的『pool_queued_async_temp_data_pages』，第 305 页的『pool_queued_async_temp_index_pages』，第 305 页的『pool_queued_async_temp_xda_pages』，第 305 页的『pool_failed_async_data_reqs』，第 306 页的『pool_failed_async_index_reqs』，第 306 页的『pool_failed_async_xda_reqs』，第 306 页的『pool_failed_async_temp_data_reqs』，第 306 页的『pool_failed_async_temp_index_reqs』，第 307 页的『pool_failed_async_temp_xda_reqs』，第 307 页的『pool_failed_async_other_reqs』，第 272 页的『app_act_completed_total』，第 272 页的『app_act_aborted_total』，第 273 页的『app_act_rejected_total』，第 307 页的『total_peds』，第 307 页的『disabled_peds』，第 307 页的『post_threshold_peds』，第 308 页的『total_peas』，第 308 页的『post_threshold_peas』，第 308 页的『tq_sort_heap_requests』，第 308 页的『tq_sort_heap_rejections』，第 273 页的『total_connect_request_proc_time』，第 273 页的『total_connect_request_time』，第 273 页的『total_connect_requests』，第 274 页的『total_connect_authentication_proc_time』，第 274 页的『total_connect_authentication_time』，第 274 页的『total_connect_authentications』，第 309 页的『prefetch_wait_time』，第 309 页的『prefetch_waits』，第 309 页的『pool_data_gbp_indep_pages_found_in_lbp』，第 309 页的『pool_index_gbp_indep_pages_found_in_lbp』，第 310 页的『pool_xda_gbp_indep_pages_found_in_lbp』，ANY content (skip) {zero or more (*)})

属性:

QName	类型	修订时间	缺省值	用法	注释
release	xs:long			必需	
任何名称空间中的任何属性					

activity_metrics

活动级别度量值。

元素内容: (第 274 页的『wlm_queue_time_total』, 第 275 页的『wlm_queue_assignments_total』, 第 275 页的『fcm_tq_recv_wait_time』, 第 275 页的『fcm_message_recv_wait_time』, 第 275 页的『fcm_tq_send_wait_time』, 第 276 页的『fcm_message_send_wait_time』, 第 276 页的『lock_wait_time』, 第 276 页的『lock_waits』, 第 276 页的『direct_read_time』, 第 277 页的『direct_read_reqs』, 第 277 页的『direct_write_time』, 第 277 页的『direct_write_reqs』, 第 277 页的『log_buffer_wait_time』, 第 278 页的『num_log_buffer_full』, 第 278 页的『log_disk_wait_time』, 第 278 页的『log_disk_waits_total』, 第 278 页的『pool_write_time』, 第 279 页的『pool_read_time』, 第 279 页的『audit_file_write_wait_time』, 第 279 页的『audit_file_writes_total』, 第 279 页的『audit_subsystem_wait_time』, 第 280 页的『audit_subsystem_waits_total』, 第 280 页的『diaglog_write_wait_time』, 第 280 页的『diaglog_writes_total』, 第 280 页的『fcm_send_wait_time』, 第 281 页的『fcm_recv_wait_time』 , 第 281 页的『total_act_wait_time』, 第 281 页的『total_section_sort_proc_time』, 第 281 页的『total_section_sort_time』, 第 282 页的『total_section_sorts』, 第 282 页的『total_act_time』, 第 282 页的『rows_read』, 第 282 页的『rows_modified』, 第 282 页的『pool_data_l_reads』, 第 283 页的『pool_index_l_reads』, 第 283 页的『pool_temp_data_l_reads』, 第 283 页的『pool_temp_index_l_reads』, 第 283 页的『pool_xda_l_reads』, 第 284 页的『pool_temp_xda_l_reads』, 第 284 页的『total_cpu_time』, 第 284 页的『pool_data_p_reads』, 第 284 页的『pool_temp_data_p_reads』, 第 285 页的『pool_xda_p_reads』, 第 285 页的『pool_temp_xda_p_reads』, 第 285 页的『pool_index_p_reads』, 第 285 页的『pool_temp_index_p_reads』, 第 286 页的『pool_data_writes』, 第 286 页的『pool_xda_writes』, 第 286 页的『pool_index_writes』, 第 286 页的『direct_reads』, 第 287 页的『direct_writes』, 第 287 页的『rows_returned』, 第 287 页的『死锁』 , 第 287 页的『lock_timeouts』, 第 288 页的『lock_escals』, 第 288 页的『fcm_sends_total』, 第 288 页的『fcm_recvs_total』, 第 288 页的『fcm_send_volume』, 第 289 页的『fcm_recv_volume』, 第 289 页的『fcm_message_sends_total』, 第 289 页的『fcm_message_recvs_total』, 第 289 页的『fcm_message_send_volume』, 第 290 页的『fcm_message_recv_volume』, 第 290 页的『fcm_tq_sends_total』, 第 290 页的『fcm_tq_recvs_total』, 第 290 页的『fcm_tq_send_volume』, 第 291 页的『fcm_tq_recv_volume』, 第 291 页的『tq_tot_send_spills』, 第 291 页的『post_threshold_sorts』, 第 291 页的『post_shrthreshold_sorts』, 第 292 页的『sort_overflows』, 第 292 页的『audit_events_total』, 第 292 页的『total_sorts』, 第 292 页的『stmt_exec_time』, 第 292 页的『coord_stmt_exec_time』 {zero or one times (?)} , 第 293 页的『total_routine_non_sect_proc_time』 , 第 293 页的『total_routine_non_sect_time』, 第 293 页的『total_section_proc_time』, 第 293 页的『total_section_time』, 第 294 页的『total_app_section_executions』, 第 294 页的『total_routine_user_code_proc_time』, 第 294 页的『total_routine_user_code_time』, 第 294 页的『total_routine_time』, 第 295 页的『thresh_violations』, 第 295 页的『num_lw_thresh_exceeded』, 第 295 页的『total_routine_invocations』, 第 295 页的『lock_wait_time_global』, 第 296 页的『lock_waits_global』, 第 296 页的『reclaim_wait_time』, 第 296 页的『spacemappage_reclaim_wait_time』, 第 296 页的『lock_timeouts_global』, 第 297 页的『lock_escals_maxlocks』, 第 297 页的『lock_escals_locklist』, 第 297 页的『lock_escals_global』, 第 297 页的『cf_wait_time』,

第 298 页的『cf_waits』，第 298 页的『pool_data_gbp_l_reads』，第 298 页的『pool_data_gbp_p_reads』，第 298 页的『pool_data_lbp_pages_found』，第 299 页的『pool_data_gbp_invalid_pages』，第 299 页的『pool_index_gbp_l_reads』，第 299 页的『pool_index_gbp_p_reads』，第 299 页的『pool_index_lbp_pages_found』，第 300 页的『pool_index_gbp_invalid_pages』，第 300 页的『pool_xda_gbp_l_reads』，第 300 页的『pool_xda_gbp_p_reads』，第 300 页的『pool_xda_lbp_pages_found』，第 301 页的『pool_xda_gbp_invalid_pages』，第 301 页的『evmon_wait_time』，第 301 页的『evmon_waits_total』，第 301 页的『total_extended_latch_wait_time』，第 302 页的『total_extended_latch_waits』，第 302 页的『total_disp_run_queue_time』，第 302 页的『pool_queued_async_data_reqs』，第 302 页的『pool_queued_async_index_reqs』，第 303 页的『pool_queued_async_xda_reqs』，第 303 页的『pool_queued_async_temp_data_reqs』，第 303 页的『pool_queued_async_temp_index_reqs』，第 303 页的『pool_queued_async_temp_xda_reqs』，第 304 页的『pool_queued_async_other_reqs』，第 304 页的『pool_queued_async_data_pages』，第 304 页的『pool_queued_async_index_pages』，第 304 页的『pool_queued_async_xda_pages』，第 305 页的『pool_queued_async_temp_data_pages』，第 305 页的『pool_queued_async_temp_index_pages』，第 305 页的『pool_queued_async_temp_xda_pages』，第 305 页的『pool_failed_async_data_reqs』，第 306 页的『pool_failed_async_index_reqs』，第 306 页的『pool_failed_async_xda_reqs』，第 306 页的『pool_failed_async_temp_data_reqs』，第 306 页的『pool_failed_async_temp_index_reqs』，第 307 页的『pool_failed_async_temp_xda_reqs』，第 307 页的『pool_failed_async_other_reqs』，第 307 页的『total_peds』，第 307 页的『disabled_peds』，第 307 页的『post_threshold_peds』，第 308 页的『total_peas』，第 308 页的『post_threshold_peas』，第 308 页的『tq_sort_heap_requests』，第 308 页的『tq_sort_heap_rejections』，第 309 页的『prefetch_wait_time』，第 309 页的『prefetch_waits』，第 309 页的『pool_data_gbp_indep_pages_found_in_lbp』，第 309 页的『pool_index_gbp_indep_pages_found_in_lbp』，第 310 页的『pool_xda_gbp_indep_pages_found_in_lbp』，ANY content (skip) {zero or more (*)})

属性:

QName	类型	修订时间	缺省值	用法	注释
release	xs:long			必需	
任何名称空间中的任何属性					

stmt_value_index

此元素表示 SQL 语句中使用的输入参数标记或主变量的位置。有关更多详细信息，请参阅监视元素 第 1182 页的『stmt_value_index -“值索引”』。

包含者:

元素内容:

类型	构面
xs:int	

stmt_value_isnull

包含者:

属性:

QName	类型	修订时间	缺省值	用法	注释
id	xs:int			必需	

stmt_value_isreopt

包含者:

属性:

QName	类型	修订时间	缺省值	用法	注释
id	xs:int			必需	

stmt_value_type

第 1184 页的『stmt_value_type -“值类型”监视元素』

包含者:

元素内容:

类型	构面
xs:string	Max length: 16

stmt_value_data

此元素包含与 SQL 语句相关联的数据值的字符串表示。有关更多详细信息，请参阅监视元素 第 1181 页的『stmt_value_data -“值数据”』。

包含者:

元素内容:

类型	构面
xs:string	Max length: 32768

agent_wait_time

有关更多详细信息，请参阅监视元素 第 605 页的『agent_wait_time -“代理程序等待时间”监视元素』。

包含者: 第 252 页的『system_metrics』

元素内容:

类型	构面
xs:long	

agent_waits_total

有关更多详细信息，请参阅监视元素 第 606 页的『agent_waits_total -“等待代理程序总次数”监视元素』。

包含者: 第 252 页的『system_metrics』

元素内容:

类型	构面
xs:long	

tcpip_recv_wait_time

有关更多详细信息，请参阅监视元素 第 1218 页的『tcpip_recv_wait_time -“TCP/IP 接收等待时间”监视元素』。

包含者: 第 252 页的『system_metrics』

元素内容:

类型	构面
xs:long	

tcpip_recvs_total

有关更多详细信息，请参阅监视元素 第 1219 页的『tcpip_recvs_total -“TCP/IP 接收总次数”监视元素』。

包含者: 第 252 页的『system_metrics』

元素内容:

类型	构面
xs:long	

client_idle_wait_time

有关更多详细信息，请参阅监视元素 第 660 页的『client_idle_wait_time -“客户机空闲等待时间”监视元素』。

包含者: 第 252 页的『system_metrics』

元素内容:

类型	构面
xs:long	

ipc_recv_wait_time

有关更多详细信息，请参阅监视元素 第 829 页的『ipc_recv_wait_time -“进程间通信接收等待时间”监视元素』。

包含者: 第 252 页的『system_metrics』

元素内容:

类型	构面
xs:long	

ipc_recvs_total

有关更多详细信息，请参阅监视元素 第 830 页的『ipc_recvs_total -“进程间通信接收总次数”监视元素』。

包含者: 第 252 页的『system_metrics』

元素内容:

类型	构面
xs:long	

ipc_send_wait_time

有关更多详细信息，请参阅监视元素 第 831 页的『ipc_send_wait_time -“进程间通信发送等待时间”监视元素』。

包含者: 第 252 页的『system_metrics』

元素内容:

类型	构面
xs:long	

ipc_sends_total

有关更多详细信息，请参阅监视元素 第 832 页的『ipc_sends_total -“进程间通信发送总次数”监视元素』。

包含者: 第 252 页的『system_metrics』

元素内容:

类型	构面
xs:long	

tcpip_send_wait_time

有关更多详细信息，请参阅监视元素 第 1221 页的『tcpip_send_wait_time -“TCP/IP 发送等待时间”监视元素』。

包含者: 第 252 页的『system_metrics』

元素内容:

类型	构面
xs:long	

tcpip_sends_total

有关更多详细信息，请参阅监视元素 第 1222 页的『tcpip_sends_total -“TCP/IP 发送总次数”监视元素』。

包含者: 第 252 页的『system_metrics』

元素内容:

类型	构面
xs:long	

total_wait_time

有关更多详细信息，请参阅监视元素 第 1302 页的『total_wait_time -“等待时间总计”监视元素』。

包含者: 第 252 页的『system_metrics』

元素内容:

类型	构面
xs:long	

total_rqst_time

有关更多详细信息，请参阅监视元素 第 1280 页的『total_rqst_time -“请求时间总计”监视元素』。

包含者: 第 252 页的『system_metrics』

元素内容:

类型	构面
xs:long	

rqsts_completed_total

有关更多详细信息，请参阅监视元素 第 1124 页的『rqsts_completed_total -“完成请求总数”监视元素』。

包含者: 第 252 页的『system_metrics』

元素内容:

类型	构面
xs:long	

total_app_rqst_time

有关更多详细信息，请参阅监视元素 第 1235 页的『total_app_rqst_time -“应用程序请求时间总计”监视元素』。

包含者: 第 252 页的『system_metrics』

元素内容:

类型	构面
xs:long	

app_rqsts_completed_total

有关更多详细信息，请参阅监视元素 第 615 页的『app_rqsts_completed_total -“完成应用程序请求总数”监视元素』。

包含者: 第 252 页的『system_metrics』

元素内容:

类型	构面
xs:long	

act_completed_total

有关更多详细信息，请参阅监视元素 第 589 页的『act_completed_total -“完成活动总数”监视元素』。

包含者: 第 252 页的『system_metrics』

元素内容:

类型	构面
xs:long	

tcpip_send_volume

有关更多详细信息，请参阅监视元素 第 1220 页的『tcpip_send_volume -“TCP/IP 发送量”监视元素』。

包含者: 第 252 页的『system_metrics』

元素内容:

类型	构面
xs:long	

tcpip_recv_volume

有关更多详细信息，请参阅监视元素 第 1217 页的『tcpip_recv_volume -“TCP/IP 接收量”监视元素』。

包含者: 第 252 页的『system_metrics』

元素内容:

类型	构面
xs:long	

ipc_send_volume

有关更多详细信息，请参阅监视元素 第 831 页的『ipc_send_volume -“进程间通信发送量”监视元素』。

包含者: 第 252 页的『system_metrics』

元素内容:

类型	构面
xs:long	

ipc_recv_volume

有关更多详细信息，请参阅监视元素 第 828 页的『ipc_recv_volume -“进程间通信接收量”监视元素』。

包含者: 第 252 页的『system_metrics』

元素内容:

类型	构面
xs:long	

total_rqst_mapped_in

有关更多详细信息，请参阅监视元素 第 1279 页的『total_rqst_mapped_in -“映入请求总数”监视元素』。

包含者: 第 252 页的『system_metrics』

元素内容:

类型	构面
xs:long	

total_rqst_mapped_out

有关更多详细信息，请参阅监视元素 第 1279 页的『total_rqst_mapped_out -“映出请求总数”监视元素』。

包含者: 第 252 页的『system_metrics』

元素内容:

类型	构面
xs:long	

act_rejected_total

有关更多详细信息，请参阅监视元素 第 591 页的『act_rejected_total -“被拒绝活动总数”监视元素』。

包含者: 第 252 页的『system_metrics』

元素内容:

类型	构面
xs:long	

act_aborted_total

有关更多详细信息，请参阅监视元素 第 588 页的『act_aborted_total -“异常终止活动总数”监视元素』。

包含者: 第 252 页的『system_metrics』

元素内容:

类型	构面
xs:long	

total_compile_proc_time

有关更多详细信息，请参阅监视元素 第 1241 页的『 total_compile_proc_time -“编译处理时间总计”监视元素 』。

包含者: 第 252 页的『 system_metrics 』

元素内容:

类型	构面
xs:long	

total_compile_time

有关更多详细信息，请参阅监视元素 第 1242 页的『 total_compile_time -“编译时间总计”监视元素 』。

包含者: 第 252 页的『 system_metrics 』

元素内容:

类型	构面
xs:long	

total_compilations

有关更多详细信息，请参阅监视元素 第 1240 页的『 total_compilations -“编译次数总计”监视元素 』。

包含者: 第 252 页的『 system_metrics 』

元素内容:

类型	构面
xs:long	

total_implicit_compile_proc_time

有关更多详细信息，请参阅监视元素 第 1258 页的『 total_implicit_compile_proc_time -“隐式编译处理时间总计”监视元素 』。

包含者: 第 252 页的『 system_metrics 』

元素内容:

类型	构面
xs:long	

total_implicit_compile_time

有关更多详细信息，请参阅监视元素 第 1259 页的『total_implicit_compile_time -“隐式编译时间总计”监视元素』。

包含者: 第 252 页的『system_metrics』

元素内容:

类型	构面
xs:long	

total_implicit_compilations

有关更多详细信息，请参阅监视元素 第 1257 页的『total_implicit_compilations -“隐式编译总数”监视元素』。

包含者: 第 252 页的『system_metrics』

元素内容:

类型	构面
xs:long	

total_runstats_proc_time

有关更多详细信息，请参阅监视元素 第 1282 页的『total_runstats_proc_time -“运行时统计信息处理时间总计”监视元素』。

包含者: 第 252 页的『system_metrics』

元素内容:

类型	构面
xs:long	

total_runstats_time

有关更多详细信息，请参阅监视元素 第 1283 页的『total_runstats_time -“运行时统计信息时间总计”监视元素』。

包含者: 第 252 页的『system_metrics』

元素内容:

类型	构面
xs:long	

total_runstats

有关更多详细信息，请参阅监视元素 第 1281 页的『total_runstats -“运行时统计信息总计”监视元素』。

包含者: 第 252 页的『system_metrics』

元素内容:

类型	构面
xs:long	

total_reorg_proc_time

有关更多详细信息，请参阅监视元素 第 1268 页的『total_reorg_proc_time -“重组处理时间总计”监视元素』。

包含者: 第 252 页的『system_metrics』

元素内容:

类型	构面
xs:long	

total_reorg_time

有关更多详细信息，请参阅监视元素 第 1269 页的『total_reorg_time -“重组时间总计”监视元素』。

包含者: 第 252 页的『system_metrics』

元素内容:

类型	构面
xs:long	

total_reorgs

有关更多详细信息，请参阅监视元素 第 1270 页的『total_reorgs -“重组操作总数”监视元素』。

包含者: 第 252 页的『system_metrics』

元素内容:

类型	构面
xs:long	

total_load_proc_time

有关更多详细信息，请参阅监视元素 第 1260 页的『total_load_proc_time -“装入处理时间总计”监视元素』。

包含者: 第 252 页的『system_metrics』

元素内容:

类型	构面
xs:long	

total_load_time

有关更多详细信息，请参阅监视元素 第 1261 页的『total_load_time -“装入时间总计”监视元素』。

包含者: 第 252 页的『system_metrics』

元素内容:

类型	构面
xs:long	

total_loads

有关更多详细信息，请参阅监视元素 第 1262 页的『total_loads -“装入操作总数”监视元素』。

包含者: 第 252 页的『system_metrics』

元素内容:

类型	构面
xs:long	

total_commit_proc_time

有关更多详细信息，请参阅监视元素 第 1238 页的『total_commit_proc_time -“落实处理时间总计”监视元素』。

包含者: 第 252 页的『system_metrics』

元素内容:

类型	构面
xs:long	

total_commit_time

有关更多详细信息，请参阅监视元素 第 1239 页的『total_commit_time -“落实时间总计”监视元素』。

包含者: 第 252 页的『system_metrics』

元素内容:

类型	构面
xs:long	

total_app_commits

有关更多详细信息，请参阅监视元素 第 1233 页的『total_app_commits -“应用程序落实次数总计”监视元素』。

包含者: 第 252 页的『system_metrics』

元素内容:

类型	构面
xs:long	

total_rollback_proc_time

有关更多详细信息，请参阅监视元素 第 1270 页的『total_rollback_proc_time -“回滚处理时间总计”监视元素』。

包含者: 第 252 页的『system_metrics』

元素内容:

类型	构面
xs:long	

total_rollback_time

有关更多详细信息，请参阅监视元素 第 1271 页的『total_rollback_time -“回滚时间总计”监视元素』。

包含者: 第 252 页的『system_metrics』

元素内容:

类型	构面
xs:long	

total_app_rollbacks

有关更多详细信息，请参阅监视元素 第 1234 页的『total_app_rollbacks -“应用程序回滚次数总计”监视元素』。

包含者: 第 252 页的『system_metrics』

元素内容:

类型	构面
xs:long	

int_commits

有关更多详细信息，请参阅监视元素 第 821 页的『int_commits -“内部落实数”监视元素』。

包含者: 第 252 页的『system_metrics』

元素内容:

类型	构面
xs:long	

int_rollbacks

有关更多详细信息，请参阅监视元素 第 823 页的『int_rollbacks -“内部回滚数”监视元素』。

包含者: 第 252 页的『system_metrics』

元素内容:

类型	构面
xs:long	

cat_cache_inserts

有关更多详细信息，请参阅监视元素 第 646 页的『cat_cache_inserts -“目录高速缓存插入数”监视元素』。

包含者: 第 252 页的『system_metrics』

元素内容:

类型	构面
xs:long	

cat_cache_lookups

有关更多详细信息，请参阅监视元素 第 647 页的『cat_cache_lookups -“目录高速缓存查询数”监视元素』。

包含者: 第 252 页的『system_metrics』

元素内容:

类型	构面
xs:long	

pkg_cache_inserts

有关更多详细信息，请参阅监视元素 第 955 页的『pkg_cache_inserts -“程序包高速缓存插入数”监视元素』。

包含者: 第 252 页的『system_metrics』

元素内容:

类型	构面
xs:long	

pkg_cache_lookups

有关更多详细信息，请参阅监视元素 第 956 页的『pkg_cache_lookups -“程序包高速缓存查询数”监视元素』。

包含者: 第 252 页的『system_metrics』

元素内容:

类型	构面
xs:long	

act_rqsts_total

有关更多详细信息，请参阅监视元素 第 593 页的『act_rqsts_total -“活动请求总数”监视元素』。

包含者: 第 252 页的『system_metrics』

元素内容:

类型	构面
xs:long	

total_stats_fabrication_proc_time

有关更多详细信息，请参阅监视元素 第 1293 页的『total_stats_fabrication_proc_time -“统计信息生成处理时间总计”监视元素』。

包含者: 第 252 页的『system_metrics』

元素内容:

类型	构面
xs:long	

total_stats_fabrication_time

有关更多详细信息，请参阅监视元素 第 1294 页的『total_stats_fabrication_time -“统计信息生成时间总计”监视元素』。

包含者: 第 252 页的『system_metrics』

元素内容:

类型	构面
xs:long	

total_stats_fabrications

有关更多详细信息，请参阅监视元素 第 1295 页的『total_stats_fabrications -“统计信息生成总计”监视元素』。

包含者: 第 252 页的『system_metrics』

元素内容:

类型	构面
xs:long	

total_sync_runstats_proc_time

有关更多详细信息，请参阅监视元素 第 1298 页的『total_sync_runstats_proc_time -“同步 RUNSTATS 处理时间总计”监视元素』。

包含者: 第 252 页的『system_metrics』

元素内容:

类型	构面
xs:long	

total_sync_runstats_time

有关更多详细信息，请参阅监视元素 第 1296 页的『total_sync_runstats_time -“同步 RUNSTATS 时间总计”监视元素』。

包含者: 第 252 页的『system_metrics』

元素内容:

类型	构面
xs:long	

total_sync_runstats

有关更多详细信息，请参阅监视元素 第 1299 页的『total_sync_runstats -“同步 RUNSTATS 活动总数”监视元素』。

包含者: 第 252 页的『system_metrics』

元素内容:

类型	构面
xs:long	

app_act_completed_total

有关更多详细信息，请参阅监视元素 第 613 页的『app_act_completed_total -“成功的外部协调程序活动总数”监视元素』。

包含者: 第 252 页的『system_metrics』

元素内容:

类型	构面
xs:long	

app_act_aborted_total

有关更多详细信息，请参阅监视元素 第 612 页的『app_act_aborted_total -“失败的外部协调程序活动总数”监视元素』。

包含者: 第 252 页的『system_metrics』

元素内容:

类型	构面
xs:long	

app_act_rejected_total

有关更多详细信息，请参阅监视元素 第 614 页的『app_act_rejected_total -“拒绝的外部协调程序活动总数”监视元素』。

包含者: 第 252 页的『system_metrics』

元素内容:

类型	构面
xs:long	

total_connect_request_proc_time

有关更多详细信息，请参阅监视元素 第 1246 页的『total_connect_request_proc_time -“连接或交换机用户请求处理时间总计”监视元素』。

包含者: 第 252 页的『system_metrics』

元素内容:

类型	构面
xs:long	

total_connect_request_time

有关更多详细信息，请参阅监视元素 第 1248 页的『total_connect_request_time -“连接或交换机用户请求时间总计”监视元素』。

包含者: 第 252 页的『system_metrics』

元素内容:

类型	构面
xs:long	

total_connect_requests

有关更多详细信息，请参阅监视元素 第 1247 页的『total_connect_requests -“连接或交换机用户请求数”监视元素』。

包含者: 第 252 页的『system_metrics』

元素内容:

类型	构面
xs:long	

total_connect_authentication_proc_time

有关更多详细信息，请参阅监视元素 第 1243 页的『total_connect_authentication_proc_time -“连接认证处理时间总计”监视元素』。

包含者: 第 252 页的『system_metrics』

元素内容:

类型	构面
xs:long	

total_connect_authentication_time

有关更多详细信息，请参阅监视元素 第 1245 页的『total_connect_authentication_time -“连接或交换机用户认证请求时间总计”监视元素』。

包含者: 第 252 页的『system_metrics』

元素内容:

类型	构面
xs:long	

total_connect_authentications

有关更多详细信息，请参阅监视元素 第 1244 页的『total_connect_authentications -“执行的连接或交换机用户认证数”监视元素』。

包含者: 第 252 页的『system_metrics』

元素内容:

类型	构面
xs:long	

wlm_queue_time_total

有关更多详细信息，请参阅监视元素 第 1337 页的『wlm_queue_time_total -“工作负载管理器队列时间总计”监视元素』。

包含者: 第 252 页的『system_metrics』 第 255 页的『activity_metrics』

元素内容:

类型	构面
xs:long	

wlm_queue_assignments_total

有关更多详细信息，请参阅监视元素 第 1336 页的『wlm_queue_assignments_total -“工作负载管理器队列分配总次数”监视元素』。

包含者: 第 252 页的『system_metrics』 第 255 页的『activity_metrics』

元素内容:

类型	构面
xs:long	

fcm_tq_recv_wait_time

有关更多详细信息，请参阅监视元素 第 783 页的『fcm_tq_recv_wait_time -“FCM 表队列接收等待时间”监视元素』。

包含者: 第 252 页的『system_metrics』 第 255 页的『activity_metrics』

元素内容:

类型	构面
xs:long	

fcm_message_recv_wait_time

有关更多详细信息，请参阅监视元素 第 768 页的『fcm_message_recv_wait_time -“接收 FCM 消息等待时间”监视元素』。

包含者: 第 252 页的『system_metrics』 第 255 页的『activity_metrics』

元素内容:

类型	构面
xs:long	

fcm_tq_send_wait_time

有关更多详细信息，请参阅监视元素 第 787 页的『fcm_tq_send_wait_time -“FCM 表队列发送等待时间”监视元素』。

包含者: 第 252 页的『system_metrics』 第 255 页的『activity_metrics』

元素内容:

类型	构面
xs:long	

fcm_message_send_wait_time

有关更多详细信息，请参阅监视元素 第 772 页的『fcm_message_send_wait_time -“发送 FCM 消息等待时间”监视元素』。

包含者: 第 252 页的『system_metrics』 第 255 页的『activity_metrics』

元素内容:

类型	构面
xs:long	

lock_wait_time

有关更多详细信息，请参阅监视元素 第 863 页的『lock_wait_time -“等待锁定时间”监视元素』。

包含者: 第 252 页的『system_metrics』 第 255 页的『activity_metrics』

元素内容:

类型	构面
xs:long	

lock_waits

有关更多详细信息，请参阅监视元素 第 868 页的『lock_waits -“等待锁定次数”监视元素』。

包含者: 第 252 页的『system_metrics』 第 255 页的『activity_metrics』

元素内容:

类型	构面
xs:long	

direct_read_time

有关更多详细信息，请参阅监视元素 第 737 页的『direct_read_time -“直接读时间”监视元素』。

包含者: 第 252 页的『system_metrics』 第 255 页的『activity_metrics』

元素内容:

类型	构面
xs:long	

direct_read_reqs

有关更多详细信息，请参阅监视元素 第 735 页的『direct_read_reqs -“直接读请求数”监视元素』。

包含者: 第 252 页的『system_metrics』 第 255 页的『activity_metrics』

元素内容:

类型	构面
xs:long	

direct_write_time

有关更多详细信息，请参阅监视元素 第 742 页的『direct_write_time -“直接写时间”监视元素』。

包含者: 第 252 页的『system_metrics』 第 255 页的『activity_metrics』

元素内容:

类型	构面
xs:long	

direct_write_reqs

有关更多详细信息，请参阅监视元素 第 740 页的『direct_write_reqs -“直接写请求数”监视元素』。

包含者: 第 252 页的『system_metrics』 第 255 页的『activity_metrics』

元素内容:

类型	构面
xs:long	

log_buffer_wait_time

有关更多详细信息，请参阅监视元素 第 872 页的『log_buffer_wait_time -“日志缓冲区等待时间”监视元素』。

包含者: 第 252 页的『system_metrics』 第 255 页的『activity_metrics』

元素内容:

类型	构面
xs:long	

num_log_buffer_full

有关更多详细信息，请参阅监视元素 第 913 页的『num_log_buffer_full -“日志缓冲区变满而导致代理程序等待的次数”监视元素』。

包含者: 第 252 页的『system_metrics』 第 255 页的『activity_metrics』

元素内容:

类型	构面
xs:long	

log_disk_wait_time

有关更多详细信息，请参阅监视元素 第 874 页的『log_disk_wait_time -“日志磁盘等待时间”监视元素』。

包含者: 第 252 页的『system_metrics』 第 255 页的『activity_metrics』

元素内容:

类型	构面
xs:long	

log_disk_waits_total

有关更多详细信息，请参阅监视元素 第 875 页的『log_disk_waits_total -“日志磁盘等待总次数”监视元素』。

包含者: 第 252 页的『system_metrics』 第 255 页的『activity_metrics』

元素内容:

类型	构面
xs:long	

pool_write_time

有关更多详细信息，请参阅监视元素 第 1058 页的『pool_write_time -“缓冲池物理写时间总计”监视元素』。

包含者: 第 252 页的『system_metrics』 第 255 页的『activity_metrics』

元素内容:

类型	构面
xs:long	

pool_read_time

有关更多详细信息，请参阅监视元素 第 1043 页的『pool_read_time -“缓冲池物理读时间总计”监视元素』。

包含者: 第 252 页的『system_metrics』 第 255 页的『activity_metrics』

元素内容:

类型	构面
xs:long	

audit_file_write_wait_time

有关更多详细信息，请参阅监视元素 第 629 页的『audit_file_write_wait_time -“审计文件写等待时间”监视元素』。

包含者: 第 252 页的『system_metrics』 第 255 页的『activity_metrics』

元素内容:

类型	构面
xs:long	

audit_file_writes_total

有关更多详细信息，请参阅监视元素 第 631 页的『audit_file_writes_total -“写审计文件总次数”监视元素』。

包含者: 第 252 页的『system_metrics』 第 255 页的『activity_metrics』

元素内容:

类型	构面
xs:long	

audit_subsystem_wait_time

有关更多详细信息，请参阅监视元素 第 632 页的『audit_subsystem_wait_time -“审计子系统等待时间”监视元素』。

包含者: 第 252 页的『system_metrics』 第 255 页的『activity_metrics』

元素内容:

类型	构面
xs:long	

audit_subsystem_waits_total

有关更多详细信息，请参阅监视元素 第 634 页的『audit_subsystem_waits_total -“审计子系统等待总次数”监视元素』。

包含者: 第 252 页的『system_metrics』 第 255 页的『activity_metrics』

元素内容:

类型	构面
xs:long	

diaglog_write_wait_time

有关更多详细信息，请参阅监视元素 第 732 页的『diaglog_write_wait_time -“诊断日志文件写等待时间”监视元素』。

包含者: 第 252 页的『system_metrics』 第 255 页的『activity_metrics』

元素内容:

类型	构面
xs:long	

diaglog_writes_total

有关更多详细信息，请参阅监视元素 第 734 页的『diaglog_writes_total -“写诊断日志文件总次数”监视元素』。

包含者: 第 252 页的『system_metrics』 第 255 页的『activity_metrics』

元素内容:

类型	构面
xs:long	

fcm_send_wait_time

有关更多详细信息，请参阅监视元素 第 779 页的『fcm_send_wait_time -“FCM 发送等待时间”监视元素』。

包含者: 第 252 页的『system_metrics』 第 255 页的『activity_metrics』

元素内容:

类型	构面
xs:long	

fcm_recv_wait_time

有关更多详细信息，请参阅监视元素 第 775 页的『fcm_recv_wait_time -“FCM 接收等待时间”监视元素』。

包含者: 第 252 页的『system_metrics』 第 255 页的『activity_metrics』

元素内容:

类型	构面
xs:long	

total_act_wait_time

有关更多详细信息，请参阅监视元素 第 1232 页的『total_act_wait_time -“活动等待时间总计”监视元素』。

包含者: 第 252 页的『system_metrics』 第 255 页的『activity_metrics』

元素内容:

类型	构面
xs:long	

total_section_sort_proc_time

有关更多详细信息，请参阅监视元素 第 1285 页的『total_section_sort_proc_time -“节排序处理时间总计”监视元素』。

包含者: 第 252 页的『system_metrics』 第 255 页的『activity_metrics』

元素内容:

类型	构面
xs:long	

total_section_sort_time

有关更多详细信息，请参阅监视元素 第 1287 页的『total_section_sort_time -“节排序时间总计”监视元素』。

包含者: 第 252 页的『system_metrics』 第 255 页的『activity_metrics』

元素内容:

类型	构面
xs:long	

total_section_sorts

有关更多详细信息，请参阅监视元素 第 1288 页的『total_section_sorts -“节排序总次数”监视元素』。

包含者: 第 252 页的『system_metrics』 第 255 页的『activity_metrics』

元素内容:

类型	构面
xs:long	

total_act_time

有关更多详细信息，请参阅监视元素 第 1231 页的『total_act_time -“活动时间总计”监视元素』。

包含者: 第 252 页的『system_metrics』 第 255 页的『activity_metrics』

元素内容:

类型	构面
xs:long	

rows_read

有关更多详细信息，请参阅监视元素 第 1118 页的『rows_read -“读取行数”监视元素』。

包含者: 第 252 页的『system_metrics』 第 255 页的『activity_metrics』

元素内容:

类型	构面
xs:long	

rows_modified

有关更多详细信息，请参阅监视元素 第 1117 页的『rows_modified -“修改的行数”监视元素』。

包含者: 第 252 页的『system_metrics』 第 255 页的『activity_metrics』

元素内容:

类型	构面
xs:long	

pool_data_l_reads

有关更多详细信息，请参阅监视元素 第 982 页的『pool_data_l_reads -“缓冲池数据逻辑读取数”监视元素』。

包含者: 第 252 页的『system_metrics』 第 255 页的『activity_metrics』

元素内容:

类型	构面
xs:long	

pool_index_l_reads

有关更多详细信息，请参阅监视元素 第 1012 页的『pool_index_l_reads -“缓冲池索引逻辑读取数”监视元素』。

包含者: 第 252 页的『system_metrics』 第 255 页的『activity_metrics』

元素内容:

类型	构面
xs:long	

pool_temp_data_l_reads

有关更多详细信息，请参阅监视元素 第 1046 页的『pool_temp_data_l_reads -“缓冲池临时数据逻辑读取数”监视元素』。

包含者: 第 252 页的『system_metrics』 第 255 页的『activity_metrics』

元素内容:

类型	构面
xs:long	

pool_temp_index_l_reads

有关更多详细信息，请参阅监视元素 第 1050 页的『pool_temp_index_l_reads -“缓冲池临时索引逻辑读取数”监视元素』。

包含者: 第 252 页的『system_metrics』 第 255 页的『activity_metrics』

元素内容:

类型	构面
xs:long	

pool_xda_l_reads

有关更多详细信息，请参阅监视元素 第 1065 页的『pool_xda_l_reads -“缓冲池 XDA 数据逻辑读取数”监视元素』。

包含者: 第 252 页的『system_metrics』 第 255 页的『activity_metrics』

元素内容:

类型	构面
xs:long	

pool_temp_xda_l_reads

有关更多详细信息，请参阅监视元素 第 1053 页的『pool_temp_xda_l_reads -“缓冲池临时 XDA 数据逻辑读取数”监视元素』。

包含者: 第 252 页的『system_metrics』 第 255 页的『activity_metrics』

元素内容:

类型	构面
xs:long	

total_cpu_time

有关更多详细信息，请参阅监视元素 第 1249 页的『total_cpu_time -“CPU 时间总计”监视元素』。

包含者: 第 252 页的『system_metrics』 第 255 页的『activity_metrics』

元素内容:

类型	构面
xs:long	

pool_data_p_reads

有关更多详细信息，请参阅监视元素 第 984 页的『pool_data_p_reads -“缓冲池数据物理读取数”监视元素』。

包含者: 第 252 页的『system_metrics』 第 255 页的『activity_metrics』

元素内容:

类型	构面
xs:long	

pool_temp_data_p_reads

有关更多详细信息，请参阅监视元素 第 1048 页的『pool_temp_data_p_reads -“缓冲池临时数据物理读取数”监视元素』。

包含者: 第 252 页的『system_metrics』 第 255 页的『activity_metrics』

元素内容:

类型	构面
xs:long	

pool_xda_p_reads

有关更多详细信息，请参阅监视元素 第 1069 页的『pool_xda_p_reads -“缓冲池 XDA 数据物理读取数”监视元素』。

包含者: 第 252 页的『system_metrics』 第 255 页的『activity_metrics』

元素内容:

类型	构面
xs:long	

pool_temp_xda_p_reads

有关更多详细信息，请参阅监视元素 第 1055 页的『pool_temp_xda_p_reads -“缓冲池临时 XDA 数据物理读取数”监视元素』。

包含者: 第 252 页的『system_metrics』 第 255 页的『activity_metrics』

元素内容:

类型	构面
xs:long	

pool_index_p_reads

有关更多详细信息，请参阅监视元素 第 1014 页的『pool_index_p_reads -“缓冲池索引物理读取数”监视元素』。

包含者: 第 252 页的『system_metrics』 第 255 页的『activity_metrics』

元素内容:

类型	构面
xs:long	

pool_temp_index_p_reads

有关更多详细信息，请参阅监视元素 第 1052 页的『pool_temp_index_p_reads -“缓冲池临时索引物理读取数”监视元素』。

包含者: 第 252 页的『system_metrics』 第 255 页的『activity_metrics』

元素内容:

类型	构面
xs:long	

pool_data_writes

有关更多详细信息，请参阅监视元素 第 986 页的『pool_data_writes -“缓冲池数据写次数”监视元素』。

包含者: 第 252 页的『system_metrics』 第 255 页的『activity_metrics』

元素内容:

类型	构面
xs:long	

pool_xda_writes

有关更多详细信息，请参阅监视元素 第 1071 页的『pool_xda_writes -“缓冲池 XDA 数据写次数”监视元素』。

包含者: 第 252 页的『system_metrics』 第 255 页的『activity_metrics』

元素内容:

类型	构面
xs:long	

pool_index_writes

有关更多详细信息，请参阅监视元素 第 1016 页的『pool_index_writes -“缓冲池索引写次数”监视元素』。

包含者: 第 252 页的『system_metrics』 第 255 页的『activity_metrics』

元素内容:

类型	构面
xs:long	

direct_reads

有关更多详细信息，请参阅监视元素 第 738 页的『direct_reads -“直接读数据库数目”监视元素』。

包含者: 第 252 页的『system_metrics』 第 255 页的『activity_metrics』

元素内容:

类型	构面
xs:long	

direct_writes

有关更多详细信息，请参阅监视元素 第 744 页的『direct_writes -“直接写数据库数目”监视元素』。

包含者: 第 252 页的『system_metrics』 第 255 页的『activity_metrics』

元素内容:

类型	构面
xs:long	

rows_returned

有关更多详细信息，请参阅监视元素 第 1120 页的『rows_returned -“返回的行数”监视元素』。

包含者: 第 252 页的『system_metrics』 第 255 页的『activity_metrics』

元素内容:

类型	构面
xs:long	

死锁

有关更多详细信息，请参阅监视元素 第 728 页的『deadlocks -“检测到的死锁数”监视元素』。

包含者: 第 252 页的『system_metrics』 第 255 页的『activity_metrics』

元素内容:

类型	构面
xs:long	

lock_timeouts

有关更多详细信息，请参阅监视元素 第 860 页的『lock_timeouts -“锁定超时次数”监视元素』。

包含者: 第 252 页的『system_metrics』 第 255 页的『activity_metrics』

元素内容:

类型	构面
xs:long	

lock_escals

有关更多详细信息，请参阅监视元素 第 845 页的『lock_escals -“锁定升级次数”监视元素』。

包含者: 第 252 页的『system_metrics』 第 255 页的『activity_metrics』

元素内容:

类型	构面
xs:long	

fcm_sends_total

有关更多详细信息，请参阅监视元素 第 780 页的『fcm_sends_total -“FCM 发送总计”监视元素』。

包含者: 第 252 页的『system_metrics』 第 255 页的『activity_metrics』

元素内容:

类型	构面
xs:long	

fcm_recvs_total

有关更多详细信息，请参阅监视元素 第 777 页的『fcm_recvs_total -“FCM 接收总计”监视元素』。

包含者: 第 252 页的『system_metrics』 第 255 页的『activity_metrics』

元素内容:

类型	构面
xs:long	

fcm_send_volume

有关更多详细信息，请参阅监视元素 第 778 页的『fcm_send_volume -“FCM 发送量”监视元素』。

包含者: 第 252 页的『system_metrics』 第 255 页的『activity_metrics』

元素内容:

类型	构面
xs:long	

fcm_recv_volume

有关更多详细信息，请参阅监视元素 第 774 页的『fcm_recv_volume -“FCM 接收量”监视元素』。

包含者: 第 252 页的『system_metrics』 第 255 页的『activity_metrics』

元素内容:

类型	构面
xs:long	

fcm_message_sends_total

有关更多详细信息，请参阅监视元素 第 773 页的『fcm_message_sends_total -“发送 FCM 消息总数”监视元素』。

包含者: 第 252 页的『system_metrics』 第 255 页的『activity_metrics』

元素内容:

类型	构面
xs:long	

fcm_message_recvs_total

有关更多详细信息，请参阅监视元素 第 769 页的『fcm_message_recvs_total -“接收 FCM 消息总数”监视元素』。

包含者: 第 252 页的『system_metrics』 第 255 页的『activity_metrics』

元素内容:

类型	构面
xs:long	

fcm_message_send_volume

有关更多详细信息，请参阅监视元素 第 770 页的『fcm_message_send_volume -“发送 FCM 消息量”监视元素』。

包含者: 第 252 页的『system_metrics』 第 255 页的『activity_metrics』

元素内容:

类型	构面
xs:long	

fcm_message_recv_volume

有关更多详细信息，请参阅监视元素 第 766 页的『fcm_message_recv_volume -“接收 FCM 消息量”监视元素』。

包含者: 第 252 页的『system_metrics』 第 255 页的『activity_metrics』

元素内容:

类型	构面
xs:long	

fcm_tq_sends_total

有关更多详细信息，请参阅监视元素 第 788 页的『fcm_tq_sends_total -“FCM 表队列发送总次数”监视元素』。

包含者: 第 252 页的『system_metrics』 第 255 页的『activity_metrics』

元素内容:

类型	构面
xs:long	

fcm_tq_recvs_total

有关更多详细信息，请参阅监视元素 第 784 页的『fcm_tq_recvs_total -“FCM 表队列接收总量”监视元素』。

包含者: 第 252 页的『system_metrics』 第 255 页的『activity_metrics』

元素内容:

类型	构面
xs:long	

fcm_tq_send_volume

有关更多详细信息，请参阅监视元素 第 786 页的『fcm_tq_send_volume -“FCM 表队列发送量”监视元素』。

包含者: 第 252 页的『system_metrics』 第 255 页的『activity_metrics』

元素内容:

类型	构面
xs:long	

fcm_tq_recv_volume

有关更多详细信息，请参阅监视元素 第 782 页的『fcm_tq_recv_volume -“FCM 表队列接收量”监视元素』。

包含者: 第 252 页的『system_metrics』 第 255 页的『activity_metrics』

元素内容:

类型	构面
xs:long	

tq_tot_send_spills

有关更多详细信息，请参阅监视元素 第 1311 页的『tq_tot_send_spills -“溢出表队列缓冲区总数”监视元素』。

包含者: 第 252 页的『system_metrics』 第 255 页的『activity_metrics』

元素内容:

类型	构面
xs:long	

post_threshold_sorts

有关更多详细信息，请参阅监视元素 第 1079 页的『post_threshold_sorts -“超出阈值后的排序次数”监视元素』。

包含者: 第 252 页的『system_metrics』 第 255 页的『activity_metrics』

元素内容:

类型	构面
xs:long	

post_shrthreshold_sorts

有关更多详细信息，请参阅监视元素 第 1073 页的『post_shrthreshold_sorts -“共享阈值后排序数”监视元素』。

包含者: 第 252 页的『system_metrics』 第 255 页的『activity_metrics』

元素内容:

类型	构面
xs:long	

sort_overflows

有关更多详细信息，请参阅监视元素 第 1149 页的『sort_overflows -“排序溢出数”监视元素』。

包含者: 第 252 页的『system_metrics』 第 255 页的『activity_metrics』

元素内容:

类型	构面
xs:long	

audit_events_total

有关更多详细信息，请参阅监视元素 第 628 页的『audit_events_total -“审计事件总数”监视元素』。

包含者: 第 252 页的『system_metrics』 第 255 页的『activity_metrics』

元素内容:

类型	构面
xs:long	

total_sorts

有关更多详细信息，请参阅监视元素 第 1292 页的『total_sorts -“排序总数”监视元素』。

包含者: 第 252 页的『system_metrics』 第 255 页的『activity_metrics』

元素内容:

类型	构面
xs:long	

stmt_exec_time

有关更多详细信息，请参阅监视元素 第 1168 页的『stmt_exec_time -“语句执行时间”监视元素』。

包含者: 第 255 页的『activity_metrics』

元素内容:

类型	构面
xs:long	

coord_stmt_exec_time

有关更多详细信息，请参阅监视元素 第 696 页的『coord_stmt_exec_time -“协调代理程序执行语句的时间”监视元素』。

包含者: 第 255 页的『activity_metrics』

元素内容:

类型	构面
xs:long	

total_routine_non_sect_proc_time

有关更多详细信息，请参阅监视元素 第 1274 页的『total_routine_non_sect_proc_time -“非部分处理时间” 监视元素』。

包含者: 第 255 页的『activity_metrics』

元素内容:

类型	构面
xs:long	

total_routine_non_sect_time

有关更多详细信息，请参阅监视元素 第 1274 页的『total_routine_non_sect_time -“非部分例程执行时间” 监视元素』。

包含者: 第 255 页的『activity_metrics』

元素内容:

类型	构面
xs:long	

total_section_proc_time

有关更多详细信息，请参阅监视元素 第 1284 页的『total_section_proc_time -“部分处理时间总计”监视元素』。

包含者: 第 252 页的『system_metrics』 第 255 页的『activity_metrics』

元素内容:

类型	构面
xs:long	

total_section_time

有关更多详细信息，请参阅监视元素 第 1289 页的『total_section_time -“部分时间总计” 监视元素』。

包含者: 第 252 页的『system_metrics』 第 255 页的『activity_metrics』

元素内容:

类型	构面
xs:long	

total_app_section_executions

有关更多详细信息，请参阅监视元素 第 1235 页的『total_app_section_executions -“应用程序执行部分执行的总次数”监视元素』。

包含者: 第 252 页的『system_metrics』 第 255 页的『activity_metrics』

元素内容:

类型	构面
xs:long	

total_routine_user_code_proc_time

有关更多详细信息，请参阅监视元素 第 1276 页的『total_routine_user_code_proc_time -“例程用户代码处理时间总计”监视元素』。

包含者: 第 252 页的『system_metrics』 第 255 页的『activity_metrics』

元素内容:

类型	构面
xs:long	

total_routine_user_code_time

有关更多详细信息，请参阅监视元素 第 1278 页的『total_routine_user_code_time -“例程用户代码时间总计”监视元素』。

包含者: 第 252 页的『system_metrics』 第 255 页的『activity_metrics』

元素内容:

类型	构面
xs:long	

total_routine_time

有关更多详细信息，请参阅监视元素 第 1275 页的『total_routine_time -“例程时间总计”监视元素』。

包含者: 第 252 页的『system_metrics』 第 255 页的『activity_metrics』

元素内容:

类型	构面
xs:long	

thresh_violations

有关更多详细信息，请参阅监视元素 第 1223 页的『thresh_violations -“阈值违例次数”监视元素』。

包含者: 第 252 页的『system_metrics』 第 255 页的『activity_metrics』

元素内容:

类型	构面
xs:long	

num_lw_thresh_exceeded

有关更多详细信息，请参阅监视元素 第 917 页的『num_lw_thresh_exceeded -“超过锁定等待阈值的次数”监视元素』。

包含者: 第 252 页的『system_metrics』 第 255 页的『activity_metrics』

元素内容:

类型	构面
xs:long	

total_routine_invocations

有关更多详细信息，请参阅监视元素 第 1272 页的『total_routine_invocations -“例程调用总计”监视元素』。

包含者: 第 252 页的『system_metrics』 第 255 页的『activity_metrics』

元素内容:

类型	构面
xs:long	

lock_wait_time_global

有关更多详细信息，请参阅监视元素 第 865 页的『lock_wait_time_global -“锁定等待时间全局”监视元素』。

包含者: 第 252 页的『system_metrics』 第 255 页的『activity_metrics』

元素内容:

类型	构面
xs:long	

lock_waits_global

有关更多详细信息，请参阅监视元素 第 869 页的『lock_waits_global -“锁定等待全局”监视元素』。

包含者: 第 252 页的『system_metrics』 第 255 页的『activity_metrics』

元素内容:

类型	构面
xs:long	

reclaim_wait_time

有关更多详细信息，请参阅监视元素 第 1099 页的『reclaim_wait_time -“回收等待时间”监视元素』。

包含者: 第 252 页的『system_metrics』 第 255 页的『activity_metrics』

元素内容:

类型	构面
xs:long	

spacemappage_reclaim_wait_time

有关更多详细信息，请参阅监视元素 第 1160 页的『spacemappage_reclaim_wait_time -“空间映射页回收等待时间”监视元素』。

包含者: 第 252 页的『system_metrics』 第 255 页的『activity_metrics』

元素内容:

类型	构面
xs:long	

lock_timeouts_global

有关更多详细信息，请参阅监视元素 第 861 页的『lock_timeouts_global -“锁定超时全局”监视元素』。

包含者: 第 252 页的『system_metrics』 第 255 页的『activity_metrics』

元素内容:

类型	构面
xs:long	

lock_escals_maxlocks

有关更多详细信息，请参阅监视元素 第 850 页的『lock_escals_maxlocks -“maxlocks 锁定升级数”监视元素』。

包含者: 第 252 页的『system_metrics』 第 255 页的『activity_metrics』

元素内容:

类型	构面
xs:long	

lock_escals_locklist

有关更多详细信息，请参阅监视元素 第 849 页的『lock_escals_locklist -“locklist 锁定升级数”监视元素』。

包含者: 第 252 页的『system_metrics』 第 255 页的『activity_metrics』

元素内容:

类型	构面
xs:long	

lock_escals_global

有关更多详细信息，请参阅监视元素 第 848 页的『lock_escals_global -“全局锁定升级数”监视元素』。

包含者: 第 252 页的『system_metrics』 第 255 页的『activity_metrics』

元素内容:

类型	构面
xs:long	

cf_wait_time

有关更多详细信息，请参阅监视元素 第 652 页的『cf_wait_time -“集群高速缓存设施等待时间”监视元素』。

包含者: 第 252 页的『system_metrics』 第 255 页的『activity_metrics』

元素内容:

类型	构面
xs:long	

cf_waits

有关更多详细信息，请参阅监视元素 第 651 页的『cf_waits -“集群高速缓存设施等待次数”监视元素』。

包含者: 第 252 页的『system_metrics』 第 255 页的『activity_metrics』

元素内容:

类型	构面
xs:long	

pool_data_gbp_l_reads

有关更多详细信息，请参阅监视元素 第 978 页的『pool_data_gbp_l_reads -“组缓冲池数据逻辑读取数”监视元素』。

包含者: 第 252 页的『system_metrics』 第 255 页的『activity_metrics』

元素内容:

类型	构面
xs:long	

pool_data_gbp_p_reads

有关更多详细信息，请参阅监视元素 第 979 页的『pool_data_gbp_p_reads -“组缓冲池数据物理读取数”监视元素』。

包含者: 第 252 页的『system_metrics』 第 255 页的『activity_metrics』

元素内容:

类型	构面
xs:long	

pool_data_lbp_pages_found

有关更多详细信息，请参阅监视元素 第 981 页的『pool_data_lbp_pages_found -“本地缓冲池发现的数据页数”监视元素』。

包含者: 第 252 页的『system_metrics』 第 255 页的『activity_metrics』

元素内容:

类型	构面
xs:long	

pool_data_gbp_invalid_pages

有关更多详细信息，请参阅监视元素 第 977 页的『pool_data_gbp_invalid_pages -“组缓冲池无效数据页数”监视元素』。

包含者: 第 252 页的『system_metrics』 第 255 页的『activity_metrics』

元素内容:

类型	构面
xs:long	

pool_index_gbp_l_reads

有关更多详细信息，请参阅监视元素 第 1008 页的『pool_index_gbp_l_reads -“组缓冲池索引逻辑读取数”监视元素』。

包含者: 第 252 页的『system_metrics』 第 255 页的『activity_metrics』

元素内容:

类型	构面
xs:long	

pool_index_gbp_p_reads

有关更多详细信息，请参阅监视元素 第 1009 页的『pool_index_gbp_p_reads -“组缓冲池索引物理读取数”监视元素』。

包含者: 第 252 页的『system_metrics』 第 255 页的『activity_metrics』

元素内容:

类型	构面
xs:long	

pool_index_lbp_pages_found

有关更多详细信息，请参阅监视元素 第 1010 页的『pool_index_lbp_pages_found -“发现的本地缓冲池索引页数”监视元素』。

包含者: 第 252 页的『system_metrics』 第 255 页的『activity_metrics』

元素内容:

类型	构面
xs:long	

pool_index_gbp_invalid_pages

有关更多详细信息，请参阅监视元素 第 1006 页的『pool_index_gbp_invalid_pages -“组缓冲池无效索引页数”监视元素』。

包含者: 第 252 页的『system_metrics』 第 255 页的『activity_metrics』

元素内容:

类型	构面
xs:long	

pool_xda_gbp_l_reads

有关更多详细信息，请参阅监视元素 第 1062 页的『pool_xda_gbp_l_reads -“组缓冲池 XDA 数据逻辑读取请求数”监视元素』。

包含者: 第 252 页的『system_metrics』 第 255 页的『activity_metrics』

元素内容:

类型	构面
xs:long	

pool_xda_gbp_p_reads

有关更多详细信息，请参阅监视元素 第 1064 页的『pool_xda_gbp_p_reads -“组缓冲池 XDA 数据物理读取请求数”监视元素』。

包含者: 第 252 页的『system_metrics』 第 255 页的『activity_metrics』

元素内容:

类型	构面
xs:long	

pool_xda_lbp_pages_found

有关更多详细信息，请参阅监视元素 第 1067 页的『pool_xda_lbp_pages_found -“发现的本地缓冲池 XDA 数据页数”监视元素』。

包含者: 第 252 页的『system_metrics』 第 255 页的『activity_metrics』

元素内容:

类型	构面
xs:long	

pool_xda_gbp_invalid_pages

有关更多详细信息，请参阅监视元素 第 1061 页的『pool_xda_gbp_invalid_pages -“组缓冲池无效 XDA 数据页数”监视元素』。

包含者: 第 252 页的『system_metrics』 第 255 页的『activity_metrics』

元素内容:

类型	构面
xs:long	

evmon_wait_time

有关更多详细信息，请参阅监视元素 第 758 页的『evmon_wait_time -“事件监视器等待时间”监视元素』。

包含者: 第 252 页的『system_metrics』 第 255 页的『activity_metrics』

元素内容:

类型	构面
xs:long	

evmon_waits_total

有关更多详细信息，请参阅监视元素 第 760 页的『evmon_waits_total -“事件监视器总等待次数”监视元素』。

包含者: 第 252 页的『system_metrics』 第 255 页的『activity_metrics』

元素内容:

类型	构面
xs:long	

total_extended_latch_wait_time

有关更多详细信息，请参阅监视元素 第 1253 页的『total_extended_latch_wait_time -“扩展锁存器等待时间总计”监视元素』。

包含者: 第 252 页的『system_metrics』 第 255 页的『activity_metrics』

元素内容:

类型	构面
xs:long	

total_extended_latch_waits

有关更多详细信息，请参阅监视元素 第 1254 页的『total_extended_latch_waits -“扩展锁存器等待总计”监视元素』。

包含者: 第 252 页的『system_metrics』 第 255 页的『activity_metrics』

元素内容:

类型	构面
xs:long	

total_disp_run_queue_time

有关更多详细信息，请参阅监视元素 第 1251 页的『total_disp_run_queue_time -“分派器运行队列时间总计”监视元素』。

包含者: 第 252 页的『system_metrics』 第 255 页的『activity_metrics』

元素内容:

类型	构面
xs:long	

pool_queued_async_data_reqs

有关更多详细信息，请参阅监视元素 第 1021 页的『pool_queued_async_data_reqs -“数据预取请求数”监视元素』。

包含者: 第 252 页的『system_metrics』 第 255 页的『activity_metrics』

元素内容:

类型	构面
xs:long	

pool_queued_async_index_reqs

有关更多详细信息，请参阅监视元素 第 1025 页的『pool_queued_async_index_reqs -“索引预取请求数”监视元素』。

包含者: 第 252 页的『system_metrics』 第 255 页的『activity_metrics』

元素内容:

类型	构面
xs:long	

pool_queued_async_xda_reqs

有关更多详细信息，请参阅监视元素 第 1041 页的『pool_queued_async_xda_reqs -“XDA 预取请求数”监视元素』。

包含者: 第 252 页的『system_metrics』 第 255 页的『activity_metrics』

元素内容:

类型	构面
xs:long	

pool_queued_async_temp_data_reqs

有关更多详细信息，请参阅监视元素 第 1030 页的『pool_queued_async_temp_data_reqs -“临时表空间数据预取请求数”监视元素』。

包含者: 第 252 页的『system_metrics』 第 255 页的『activity_metrics』

元素内容:

类型	构面
xs:long	

pool_queued_async_temp_index_reqs

有关更多详细信息，请参阅监视元素 第 1034 页的『pool_queued_async_temp_index_reqs -“临时表空间索引预取请求数”监视元素』。

包含者: 第 252 页的『system_metrics』 第 255 页的『activity_metrics』

元素内容:

类型	构面
xs:long	

pool_queued_async_temp_xda_reqs

有关更多详细信息，请参阅监视元素 第 1037 页的『pool_queued_async_temp_xda_reqs -“临时表空间 XDA 数据预取请求数”监视元素』。

包含者: 第 252 页的『system_metrics』 第 255 页的『activity_metrics』

元素内容:

类型	构面
xs:long	

pool_queued_async_other_reqs

有关更多详细信息，请参阅监视元素 第 1027 页的『pool_queued_async_other_reqs -“预取程序处理的其他请求数”监视元素』。

包含者: 第 252 页的『system_metrics』 第 255 页的『activity_metrics』

元素内容:

类型	构面
xs:long	

pool_queued_async_data_pages

有关更多详细信息，请参阅监视元素 第 1019 页的『pool_queued_async_data_pages -“预取请求的数据页数”监视元素』。

包含者: 第 252 页的『system_metrics』 第 255 页的『activity_metrics』

元素内容:

类型	构面
xs:long	

pool_queued_async_index_pages

有关更多详细信息，请参阅监视元素 第 1023 页的『pool_queued_async_index_pages -“预取请求的索引页数”监视元素』。

包含者: 第 252 页的『system_metrics』 第 255 页的『activity_metrics』

元素内容:

类型	构面
xs:long	

pool_queued_async_xda_pages

有关更多详细信息，请参阅监视元素 第 1039 页的『pool_queued_async_xda_pages -“预取请求的 XDA 页数”监视元素』。

包含者: 第 252 页的『system_metrics』 第 255 页的『activity_metrics』

元素内容:

类型	构面
xs:long	

pool_queued_async_temp_data_pages

有关更多详细信息，请参阅监视元素 第 1028 页的『pool_queued_async_temp_data_pages -“预取请求的临时表空间数据页数”监视元素』。

包含者: 第 252 页的『system_metrics』 第 255 页的『activity_metrics』

元素内容:

类型	构面
xs:long	

pool_queued_async_temp_index_pages

有关更多详细信息，请参阅监视元素 第 1032 页的『pool_queued_async_temp_index_pages -“预取请求的临时表空间索引页数”监视元素』。

包含者: 第 252 页的『system_metrics』 第 255 页的『activity_metrics』

元素内容:

类型	构面
xs:long	

pool_queued_async_temp_xda_pages

有关更多详细信息，请参阅监视元素 第 1036 页的『pool_queued_async_temp_xda_pages -“预取请求的临时表空间 XDA 数据页数”监视元素』。

包含者: 第 252 页的『system_metrics』 第 255 页的『activity_metrics』

元素内容:

类型	构面
xs:long	

pool_failed_async_data_reqs

有关更多详细信息，请参阅监视元素 第 990 页的『pool_failed_async_data_reqs -“失败数据预取请求数”监视元素』。

包含者: 第 252 页的『system_metrics』 第 255 页的『activity_metrics』

元素内容:

类型	构面
xs:long	

pool_failed_async_index_reqs

有关更多详细信息，请参阅监视元素 第 992 页的『pool_failed_async_index_reqs -“失败的索引预取请求数”监视元素』。

包含者: 第 252 页的『system_metrics』 第 255 页的『activity_metrics』

元素内容:

类型	构面
xs:long	

pool_failed_async_xda_reqs

有关更多详细信息，请参阅监视元素 第 1002 页的『pool_failed_async_xda_reqs -“失败的 XDA 预取请求数”监视元素』。

包含者: 第 252 页的『system_metrics』 第 255 页的『activity_metrics』

元素内容:

类型	构面
xs:long	

pool_failed_async_temp_data_reqs

有关更多详细信息，请参阅监视元素 第 996 页的『pool_failed_async_temp_data_reqs -“临时表空间的失败数据预取请求数”监视元素』。

包含者: 第 252 页的『system_metrics』 第 255 页的『activity_metrics』

元素内容:

类型	构面
xs:long	

pool_failed_async_temp_index_reqs

有关更多详细信息，请参阅监视元素 第 998 页的『pool_failed_async_temp_index_reqs -“临时表空间的失败索引预取请求数”监视元素』。

包含者: 第 252 页的『system_metrics』 第 255 页的『activity_metrics』

元素内容:

类型	构面
xs:long	

pool_failed_async_temp_xda_reqs

有关更多详细信息，请参阅监视元素 第 1000 页的『pool_failed_async_temp_xda_reqs -“临时表空间的失败 XDA 预取请求数”监视元素』。

包含者: 第 252 页的『system_metrics』 第 255 页的『activity_metrics』

元素内容:

类型	构面
xs:long	

pool_failed_async_other_reqs

有关更多详细信息，请参阅监视元素 第 994 页的『pool_failed_async_other_reqs -“失败的非预取请求数”监视元素』。

包含者: 第 252 页的『system_metrics』 第 255 页的『activity_metrics』

元素内容:

类型	构面
xs:long	

total_peds

有关更多详细信息，请参阅监视元素 第 1266 页的『total_peds -“部分提前相异总数”监视元素』。

包含者: 第 252 页的『system_metrics』 第 255 页的『activity_metrics』

元素内容:

类型	构面
xs:long	

disabled_peds

包含者: 第 252 页的『system_metrics』 第 255 页的『activity_metrics』

元素内容:

类型	构面
xs:long	

post_threshold_peds

有关更多详细信息，请参阅监视元素 第 1077 页的『post_threshold_peds -“部分提前相异数阈值”监视元素』。

包含者: 第 252 页的『system_metrics』 第 255 页的『activity_metrics』

元素内容:

类型	构面
xs:long	

total_peas

有关更多详细信息，请参阅监视元素 第 1264 页的『total_peas -“部分提前聚集总数”监视元素』。

包含者: 第 252 页的『system_metrics』 第 255 页的『activity_metrics』

元素内容:

类型	构面
xs:long	

post_threshold_peas

有关更多详细信息，请参阅监视元素 第 1075 页的『post_threshold_peas -“部分提前聚集阈值”监视元素』。

包含者: 第 252 页的『system_metrics』 第 255 页的『activity_metrics』

元素内容:

类型	构面
xs:long	

tq_sort_heap_requests

有关更多详细信息，请参阅监视元素 第 1310 页的『tq_sort_heap_requests -“表队列排序堆请求数”监视元素』。

包含者: 第 252 页的『system_metrics』 第 255 页的『activity_metrics』

元素内容:

类型	构面
xs:long	

tq_sort_heap_rejections

有关更多详细信息，请参阅监视元素 第 1308 页的『tq_sort_heap_rejections -“表队列排序堆拒绝数”监视元素』。

包含者: 第 252 页的『system_metrics』 第 255 页的『activity_metrics』

元素内容:

类型	构面
xs:long	

prefetch_wait_time

有关更多详细信息，请参阅监视元素 第 1080 页的『prefetch_wait_time -“等待预取的时间”监视元素』。

包含者: 第 252 页的『system_metrics』 第 255 页的『activity_metrics』

元素内容:

类型	构面
xs:long	

prefetch_waits

有关更多详细信息，请参阅监视元素 第 1082 页的『prefetch_waits -“预取程序等待计数”监视元素』。

包含者: 第 252 页的『system_metrics』 第 255 页的『activity_metrics』

元素内容:

类型	构面
xs:long	

pool_data_gbp_indep_pages_found_in_lbp

包含者: 第 252 页的『system_metrics』 第 255 页的『activity_metrics』

元素内容:

类型	构面
xs:long	

pool_index_gbp_indep_pages_found_in_lbp

包含者: 第 252 页的『system_metrics』 第 255 页的『activity_metrics』

元素内容:

类型	构面
xs:long	

pool_xda_gbp_indep_pages_found_in_lbp

包含者: 第 252 页的『system_metrics』 第 255 页的『activity_metrics』

元素内容:

类型	构面
xs:long	

数据库事件监视

数据库事件监视器生成的数据

数据库事件监视器生成有关数据库级别计数器的数据。可选择将数据库事件监视器的输出写至常规表、文件或命名管道。

不管您选择什么输出格式，所有数据库事件数据都来自下列两个逻辑组的其中一个:

- 第 50 页的『event_db 逻辑数据组』
- 第 53 页的『event_dbmemuse 逻辑数据组』

此外，如果选择将数据库事件数据写至表，那么另一个组 (CONTROL) 中的数据用于生成有关事件监视器本身的元数据。

数据库事件监视器写至表的信息:

在指定了 WRITE TO TABLE 选项的情况下数据库事件监视器写入的信息。

选择 WRITE TO TABLE 作为数据库事件监视器的输出类型时，缺省情况下，会生成三个表，每个表包含一个或多个逻辑数据组中的监视元素:

表 60. DATABASE 写至表事件监视器生成的表. 表名通过逻辑数据组（用于填充该表）的名称和 CREATE EVENT MONITOR 语句中给予事件监视器的名称（如下表的表名中的 *evmon-name* 所示）并置派生。

缺省表名	报告的逻辑数据组
DB_*evmon-name*	event_db
DBMEMUSE_*evmon-name*	event_dbmemuse
CONTROL_*evmon-name*	CONTROL 逻辑组由来自 event_dbheader、event_start 和 event_overflow 逻辑数据组中的一个或多个的所选元素组成。

要将事件监视器的输出限制为发送至特定表，请在 CREATE EVENT MONITOR 或 ALTER EVENT MONITOR 语句中指定要对其生成表的逻辑组的名称。有关详细信息，请参阅这些语句的参考主题。

有关该事件监视器写至文件或命名管道时返回的输出的信息，请参阅第 98 页的『事件监视器自描述数据流』。

生成的表

表 61. 对数据库事件监视器返回的信息: 缺省表名: DB_evmon-name

列名	数据类型	描述
APPL_SECTION_INSERTS	BIGINT	appl_section_inserts - 节插入数
APPL_SECTION_LOOKUPS	BIGINT	appl_section_lookups - 节查询数
ASYNC_RUNSTATS	BIGINT	async_runstats - 异步 RUNSTATS 请求总数
BINDS_PRECOMPILES	BIGINT	binds_precompiles - 尝试的绑定次数/预编译次数
BLOCKS_PENDING_CLEANUP	BIGINT	blocks_pending_cleanup - 暂挂清除的滚出块数
CAT_CACHE_HEAP_FULL	BIGINT	
CAT_CACHE_INSERTS	BIGINT	cat_cache_inserts - 目录高速缓存插入数
CAT_CACHE_LOOKUPS	BIGINT	cat_cache_lookups - 目录高速缓存查询数
CAT_CACHE_OVERFLOWS	BIGINT	cat_cache_overflows - 目录高速缓存溢出数
CAT_CACHE_SIZE_TOP	BIGINT	cat_cache_size_top - 目录高速缓存高水位标记
CATALOG_NODE	BIGINT	catalog_node - 目录节点号
CATALOG_NODE_NAME	VARCHAR(32)	catalog_node_name - 目录节点网络名
COMMIT_SQL_STMTS	BIGINT	commit_sql_stmts - 尝试的落实语句数
CONNECTIONS_TOP	BIGINT	connections_top - 最大并行连接数
DB_HEAP_TOP	BIGINT	db_heap_top - 分配的最大数据库堆
DDL_SQL_STMTS	BIGINT	ddl_sql_stmts - 数据定义语言（DDL）SQL 语句数
DEADLOCKS	BIGINT	deadlocks - 检测到的死锁数
DIRECT_READ_REQS	BIGINT	direct_read_reqs - 直接读请求数
DIRECT_READ_TIME	BIGINT	direct_read_time - 直接读时间
DIRECT_READS	BIGINT	direct_reads - 直接读数据库数目
DIRECT_WRITE_REQS	BIGINT	direct_write_reqs - 直接写请求数
DIRECT_WRITE_TIME	BIGINT	direct_write_time - 直接写时间
DIRECT_WRITES	BIGINT	direct_writes - 直接写数据库数目
DISCONN_TIME	TIMESTAMP	disconn_time - 数据库释放时间戳记
DYNAMIC_SQL_STMTS	BIGINT	dynamic_sql_stmts - 尝试的动态 SQL 语句数

表 61. 对数据库事件监视器返回的信息: 缺省表名: DB_evmon-name (续)

列名	数据类型	描述
ELAPSED_EXEC_TIME	BIGINT	elapsed_exec_time - 语句执行耗用时间
EVMON_ACTIVATES	BIGINT	evmon_activates - 事件监视器激活数
EVMON_FLUSHES	BIGINT	evmon_flushes - 事件监视器清空数
FAILED_SQL_STMTS	BIGINT	failed_sql_stmts - 失败的语句操作
FILES_CLOSED	BIGINT	files_closed - 关闭数据库文件数
HASH_JOIN_OVERFLOWS	BIGINT	hash_join_overflows - 散列连接溢出数
HASH_JOIN_SMALL_OVERFLOWS	BIGINT	hash_join_small_overflows - 散列连接小溢出数
INT_AUTO_REBINDS	BIGINT	int_auto_rebinds - 内部自动重新绑定次数
INT_COMMITS	BIGINT	int_commits - 内部落实数
INT_ROLLBACKS	BIGINT	int_rollbacks - 内部回滚
INT_ROWS_DELETED	BIGINT	int_rows_deleted - 删除的内部行数
INT_ROWS_INSERTED	BIGINT	int_rows_inserted - 插入的内部行数
INT_ROWS_UPDATED	BIGINT	int_rows_updated - 更新的内部行数
LOCK_ESCALS	BIGINT	lock_escals - 锁定升级次数
LOCK_TIMEOUTS	BIGINT	lock_timeouts - 锁定超时次数
LOCK_WAIT_TIME	BIGINT	lock_wait_time - 等待锁定时间
LOCK_WAITS	BIGINT	lock_waits - 等待锁定次数
LOG_FILE_ARCHIVE	BIGINT	
LOG_FILE_NUM_CURR	BIGINT	
LOG_FILE_NUM_FIRST	BIGINT	
LOG_FILE_NUM_LAST	BIGINT	
LOG_HELD_BY_DIRTY_PAGES	BIGINT	log_held_by_dirty_pages - 脏页占用的日志空间量
LOG_READ_TIME	BIGINT	log_read_time - 日志读取时间
LOG_READS	BIGINT	log_reads - 读取的日志页数
LOG_TO_REDO_FOR_RECOVERY	BIGINT	log_to_redo_for_recovery - 要为恢复重做的日志量
LOG_WRITE_TIME	BIGINT	log_write_time - 日志写入时间
LOG_WRITES	BIGINT	log_writes - 写入的日志页数
NUM_LOG_BUFF_FULL	BIGINT	
NUM_LOG_DATA_IN_BUFF	BIGINT	

表 61. 对数据库事件监视器返回的信息: 缺省表名: DB_evmon-name (续)

列名	数据类型	描述
NUM_LOG_PART_PAGE_IO	BIGINT	num_log_part_page_io - 部分日志页写入数
NUM_LOG_READ_IO	BIGINT	num_log_read_io - 日志读取数
NUM_LOG_WRITE_IO	BIGINT	num_log_write_io - 日志写入次数
NUM_THRESHOLD_VIOLATIONS	INTEGER	num_threshold_violations - 阈值违例次数
OLAP_FUNC_OVERFLOWS	BIGINT	olap_func_overflows - OLAP 函数溢出次数
PARTIAL_RECORD	SMALLINT	partial_record - 部分记录
PKG_CACHE_INSERTS	BIGINT	pkg_cache_inserts - 程序包高速缓存插入数
PKG_CACHE_LOOKUPS	BIGINT	pkg_cache_lookups - 程序包高速缓存查询数
PKG_CACHE_NUM_OVERFLOWS	BIGINT	pkg_cache_num_overflows - 程序包高速缓存溢出数
PKG_CACHE_SIZE_TOP	BIGINT	pkg_cache_size_top - 程序包高速缓存高水位标记
POOL_ASYNC_DATA_READ_REQS	BIGINT	pool_async_data_read_reqs - 缓冲池异步读请求数
POOL_ASYNC_DATA_READS	BIGINT	pool_async_data_reads - 缓冲池异步数据读次数
POOL_ASYNC_DATA_WRITES	BIGINT	pool_async_data_writes - 缓冲池异步数据写次数
POOL_ASYNC_INDEX_READ_REQS	BIGINT	pool_async_index_read_reqs - 缓冲池异步索引读请求数
POOL_ASYNC_INDEX_READS	BIGINT	pool_async_index_reads - 缓冲池异步索引读取数
POOL_ASYNC_INDEX_WRITES	BIGINT	pool_async_index_writes - 缓冲池异步索引写次数
POOL_ASYNC_READ_TIME	BIGINT	pool_async_read_time - 缓冲池异步读取时间
POOL_ASYNC_WRITE_TIME	BIGINT	pool_async_write_time - 缓冲池异步写入时间
POOL_ASYNC_XDA_READ_REQS	BIGINT	pool_async_xda_read_reqs - 缓冲池异步 XDA 读请求数
POOL_ASYNC_XDA_READS	BIGINT	pool_async_xda_reads - 缓冲池异步 XDA 数据读取数
POOL_ASYNC_XDA_WRITES	BIGINT	pool_async_xda_writes - 缓冲池异步 XDA 数据写次数
POOL_DATA_L_READS	BIGINT	pool_data_l_reads - 缓冲池数据逻辑读取数
POOL_DATA_P_READS	BIGINT	pool_data_p_reads - 缓冲池数据物理读取数

表 61. 对数据库事件监视器返回的信息: 缺省表名: DB_evmon-name (续)

列名	数据类型	描述
POOL_DATA_WRITES	BIGINT	pool_data_writes - 缓冲池数据写次数
POOL_DRTY_PG_STEAL_CLNS	BIGINT	pool_drty_pg_steal_clns - 触发缓冲池牺牲页清除程序次数
POOL_DRTY_PG_THRSH_CLNS	BIGINT	pool_drty_pg_thrsh_clns - 触发缓冲池阈值清除程序次数
POOL_INDEX_L_READS	BIGINT	pool_index_l_reads - 缓冲池索引逻辑读取数
POOL_INDEX_P_READS	BIGINT	pool_index_p_reads - 缓冲池索引物理读取数
POOL_INDEX_WRITES	BIGINT	pool_index_writes - 缓冲池索引写次数
POOL_LSN_GAP_CLNS	BIGINT	pool_lsn_gap_clns - 触发缓冲池日志空间清除程序次数
POOL_NO_VICTIM_BUFFER	BIGINT	pool_no_victim_buffer - 缓冲池无牺牲缓冲区次数
POOL_READ_TIME	BIGINT	pool_read_time - 缓冲池物理读时间总计
POOL_TEMP_DATA_L_READS	BIGINT	pool_temp_data_l_reads - 缓冲池临时数据逻辑读取数
POOL_TEMP_DATA_P_READS	BIGINT	pool_temp_data_p_reads - 缓冲池临时数据物理读取数
POOL_TEMP_INDEX_L_READS	BIGINT	pool_temp_index_l_reads - 缓冲池临时索引逻辑读取数
POOL_TEMP_INDEX_P_READS	BIGINT	pool_temp_index_p_reads - 缓冲池临时索引物理读取数
POOL_TEMP_XDA_L_READS	BIGINT	pool_temp_xda_l_reads - 缓冲池临时 XDA 数据逻辑读取数
POOL_TEMP_XDA_P_READS	BIGINT	pool_temp_xda_p_reads - 缓冲池临时 XDA 数据物理读取数
POOL_WRITE_TIME	BIGINT	pool_write_time - 缓冲池物理写时间总计
POOL_XDA_L_READS	BIGINT	pool_xda_l_reads - 缓冲池 XDA 数据逻辑读取数
POOL_XDA_P_READS	BIGINT	pool_xda_p_reads - 缓冲池 XDA 数据物理读取数
POOL_XDA_WRITES	BIGINT	pool_xda_writes - 缓冲池 XDA 数据写次数
POST_SHRTHRESHOLD_HASH_JOINS	BIGINT	post_shrthreshold_hash_joins - 阈值后散列连接数
POST_SHRTHRESHOLD_SORTS	BIGINT	post_shrthreshold_sorts - 共享阈值后排序数
PREFETCH_WAIT_TIME	BIGINT	prefetch_wait_time - 等待预取的时间

表 61. 对数据库事件监视器返回的信息: 缺省表名: DB_evmon-name (续)

列名	数据类型	描述
ROLLBACK_SQL_STMTS	BIGINT	rollback_sql_stmts - 尝试的回滚语句数
ROWS_DELETED	BIGINT	rows_deleted - 删除行数
ROWS_INSERTED	BIGINT	rows_inserted - 插入行数
ROWS_READ	BIGINT	rows_read - 读取行数
ROWS_SELECTED	BIGINT	rows_selected - 选择的行数
ROWS_UPDATED	BIGINT	rows_updated - 更新行数
SEC_LOG_USED_TOP	BIGINT	sec_log_used_top - 使用的最大辅助日志空间
SELECT_SQL_STMTS	BIGINT	select_sql_stmts - 执行的 Select SQL 语句数
SERVER_PLATFORM	INTEGER	server_platform - 服务器操作系统
SORT_OVERFLOWS	BIGINT	sort_overflows - 排序溢出数
SORT_SHRHEAP_TOP	BIGINT	sort_shrheap_top - 排序共享堆高水位标记
STATIC_SQL_STMTS	BIGINT	static_sql_stmts - 尝试的静态 SQL 语句数
STATS_CACHE_SIZE	BIGINT	stats_cache_size - 统计信息高速缓存大小
STATS_FABRICATE_TIME	BIGINT	stats_fabricate_time - 生成统计信息的活动所耗的总时间
STATS_FABRICATIONS	BIGINT	stats_fabrications - 生成统计信息的总次数
SYNC_RUNSTATS	BIGINT	sync_runstats - 同步 RUNSTATS 活动总数
SYNC_RUNSTATS_TIME	BIGINT	第 1188 页的『sync_runstats_time -“同步 RUNSTATS 活动所花的总时间”监视元素』
TOT_LOG_USED_TOP	BIGINT	tot_log_used_top - 使用的最大总日志空间
TOTAL_CONS	BIGINT	total_cons - 数据库激活以后的连接数
TOTAL_HASH_JOINS	BIGINT	total_hash_joins - 散列连接总数
TOTAL_HASH_LOOPS	BIGINT	total_hash_loops - 总散列循环数
TOTAL_OLAP_FUNCS	BIGINT	total_olap_funcs - OLAP 函数总数
TOTAL_SORT_TIME	BIGINT	total_sort_time - 总排序时间
TOTAL_SORTS	BIGINT	total_sorts - 排序总数
UID_SQL_STMTS	BIGINT	uid_sql_stmts - 执行的 UPDATE/INSERT/DELETE SQL 语句数

表 61. 对数据库事件监视器返回的信息: 缺省表名: DB_evmon-name (续)

列名	数据类型	描述
UNREAD_PREFETCH_PAGES	BIGINT	unread_prefetch_pages - 未读取的预取页数
X_LOCK_ESCALS	BIGINT	x_lock_escals - 互斥锁定升级数
XQUERY_STMTS	BIGINT	xquery_stmts - 尝试的 XQuery 语句数

表 62. 对数据库事件监视器返回的信息: 缺省表名: DBMEMUSE_evmon-name

列名	数据类型	描述
EVMON_ACTIVATES	BIGINT	evmon_activates - 事件监视器激活数
EVMON_FLUSHES	BIGINT	evmon_flushes - 事件监视器清空数
POOL_CUR_SIZE	BIGINT	pool_cur_size - 内存池的当前大小
POOL_ID	BIGINT	pool_id - 内存池标识
POOL_MAX_SIZE	BIGINT	
POOL_SECONDARY_ID	CHARACTER(32)	pool_secondary_id - 内存池辅助标记
POOL_WATERMARK	BIGINT	pool_watermark - 内存池水位标记

表 63. 对数据库事件监视器返回的信息: 缺省表名: CONTROL_evmon-name

列名	数据类型	描述
EVENT_MONITOR_NAME	VARCHAR(128)	event_monitor_name - 事件监视器名称
MESSAGE	VARCHAR(128)	message - 控制表消息
MESSAGE_TIME	TIMESTAMP	message_time - 时间戳记控制表消息

阈值违例事件监视

阈值违例事件监视器生成的数据

阈值违例事件监视器生成有关阈值违例的数据。可选择将数据库事件监视器的输出写至常规表、文件或命名管道。

不管您选择什么输出格式，所有阈值违例事件数据都来自 event_thresholdviolations 逻辑数据组。此外，如果选择将事件数据写至表，那么另一个组 (CONTROL) 中的数据用于生成有关事件监视器本身的元数据。

阈值违例事件监视器写至表的信息:

在指定了 WRITE TO TABLE 选项的情况下阈值违例事件监视器写入的信息。

下面的几节说明在 CREATE EVENT MONITOR 语句上使用 WRITE TO TABLE 选项时阈值违例事件监视器的输出。有关该事件监视器写至文件或命名管道时返回的输出的信息，请参阅第 98 页的『事件监视器自描述数据流』。

表 64. THRESHOLD 写至表事件监视器生成的表. 表名通过逻辑数据组（用于填充该表）的名称和 CREATE EVENT MONITOR 语句中给予事件监视器的名称（如下表的表名中的 *evmon-name* 所示）并置派生。

缺省表名	报告的逻辑数据组
THRESHOLDVIOLATIONS_*evmon-name*	event_thresholdviolations
CONTROL_*evmon-name*	CONTROL 逻辑组由来自 event_dbheader、event_start 和 event_overflow 逻辑数据组中的一个或多个的所选元素组成。

要将事件监视器的输出限制为发送至特定表，请在 CREATE EVENT MONITOR 或 ALTER EVENT MONITOR 语句中指定要对其生成表的逻辑组的名称。有关详细信息，请参阅这些语句的参考主题。

生成的表

表 65. 对阈值违例事件监视器返回的信息：缺省表名：THRESHOLDVIOLATIONS_evmon-name

列名	数据类型	描述
PARTITION_KEY	INTEGER	第 951 页的『partition_key -“分区键”监视元素』
ACTIVATE_TIMESTAMP	TIMESTAMP	activate_timestamp - 激活时间戳记
ACTIVITY_COLLECTED	CHARACTER(1)	activity_collected - 收集的活动
ACTIVITY_ID	BIGINT	activity_id - 活动标识
AGENT_ID	BIGINT	agent_id - 应用程序句柄（代理程序标识）
APPL_ID	VARCHAR(64)	appl_id - 应用程序标识
COORD_PARTITION_NUM	INTEGER	coord_partition_num - 协调程序分区号
DESTINATION_SERVICE_CLASS_ID	INTEGER	第 731 页的『destination_service_class_id -“目标服务类标识”监视元素』
PARTITION_NUMBER	SMALLINT	partition_number - 分区号
SOURCE_SERVICE_CLASS_ID	INTEGER	source_service_class_id - 源服务类标识
THRESHOLD_ACTION	VARCHAR(16)	threshold_action - 阈值操作
THRESHOLD_MAXVALUE	BIGINT	threshold_maxvalue - 阈值最大值
THRESHOLD_PREDICATE	VARCHAR(64)	threshold_predicate - 阈值谓词
THRESHOLD_QUEUESIZE	BIGINT	threshold_queuesize - 阈值队列大小
THRESHOLDID	INTEGER	thresholdid - 阈值标识
TIME_OF_VIOLATION	TIMESTAMP	time_of_violation - 违例时间
UOW_ID	INTEGER	uow_id - 工作单元标识

表 66. 对阈值违例事件监视器返回的信息：表名：CONTROL_evmon-name

列名	数据类型	描述
PARTITION_KEY	INTEGER	第 951 页的『partition_key -“分区键”监视元素』
EVENT_MONITOR_NAME	VARCHAR(128)	event_monitor_name - 事件监视器名称
MESSAGE	VARCHAR(128)	message - 控制表消息
MESSAGE_TIME	TIMESTAMP	message_time - 时间戳记控制表消息
PARTITION_NUMBER	SMALLINT	partition_number - 分区号

语句事件监视

语句事件监视器生成的数据

语句事件监视器生成有关在系统上运行的语句的数据。可选择将数据库事件监视器的输出写至常规表、文件或命名管道。

不管您选择什么输出格式，所有语句事件数据都来自下列三个逻辑组的其中一个:

- 第 58 页的『event_stmt 逻辑数据组』
- 第 49 页的『event_connheader 逻辑数据组』
- 第 60 页的『event_subsection 逻辑数据组』

此外，如果选择将语句事件数据写至表，那么另一个组 (CONTROL) 中的数据用于生成有关事件监视器本身的元数据。

语句事件监视器写至表的信息:

在指定了 WRITE TO TABLE 选项的情况下语句事件监视器写入的信息。

下面的几节说明在 CREATE EVENT MONITOR 语句上使用 WRITE TO TABLE 选项时语句事件监视器的输出。有关该事件监视器写至文件或命名管道时返回的输出的信息，请参阅第 98 页的『事件监视器自描述数据流』。

表 67. STATEMENT"写至表"事件监视器生成的表. 表名通过逻辑数据组（用于填充该表）的名称和 CREATE EVENT MONITOR 语句中给予事件监视器的名称（如下表的表名中的 *evmon-name* 所示）并置派生。

缺省表名	报告的逻辑数据组
STMT_*evmon-name*	第 58 页的『event_stmt 逻辑数据组』
CONNHEADER_*evmon-name*	第 49 页的『event_connheader 逻辑数据组』
CONNHEADER_*evmon-name*	第 49 页的『event_connheader 逻辑数据组』
SUBSECTION_*evmon-name*	第 60 页的『event_subsection 逻辑数据组』（仅在分区数据库环境中生成）
CONTROL_*evmon-name*	CONTROL 逻辑组由来自 event_dbheader、event_start 和 event_overflow 逻辑数据组中的一个或多个的所选元素组成。

要将事件监视器的输出限制为发送至特定表，请在 CREATE EVENT MONITOR 或 ALTER EVENT MONITOR 语句中指定要对其生成表的逻辑组的名称。有关详细信息，请参阅这些语句的参考主题。

生成的表

表 68. 对语句事件监视器返回的信息: 缺省表名: STMT_evmon-name

列名	数据类型	描述
AGENT_ID	BIGINT	agent_id - 应用程序句柄（代理程序标识）
AGENTS_TOP	BIGINT	agents_top - 创建的代理程序数
APPL_ID	VARCHAR(64)	appl_id - 应用程序标识
BLOCKING_CURSOR	SMALLINT	blocking_cursor - 分块游标

表 68. 对语句事件监视器返回的信息：缺省表名：STMT_evmon-name（续）

列名	数据类型	描述
CONSISTENCY_TOKEN	CHARACTER	consistency_token - 程序包一致性标记
CREATOR	VARCHAR(128)	creator - 应用程序创建者
CURSOR_NAME	VARCHAR(18)	cursor_name - 游标名称
EVMON_FLUSHES	BIGINT	evmon_flushes - 事件监视器清空数
FETCH_COUNT	BIGINT	fetch_count - 成功的访存数
INT_ROWS_DELETED	BIGINT	int_rows_deleted - 删除的内部行数
INT_ROWS_INSERTED	BIGINT	int_rows_inserted - 插入的内部行数
INT_ROWS_UPDATED	BIGINT	int_rows_updated - 更新的内部行数
PACKAGE_NAME	VARCHAR(128)	package_name - 程序包名
PACKAGE_VERSION_ID	VARCHAR(64)	package_version_id - 程序包版本
PARTIAL_RECORD	SMALLINT	partial_record - 部分记录
POOL_DATA_L_READS	BIGINT	pool_data_l_reads - 缓冲池数据逻辑读取数
POOL_DATA_P_READS	BIGINT	pool_data_p_reads - 缓冲池数据物理读取数
POOL_INDEX_L_READS	BIGINT	pool_index_l_reads - 缓冲池索引逻辑读取数
POOL_INDEX_P_READS	BIGINT	pool_index_p_reads - 缓冲池索引物理读取数
POOL_TEMP_DATA_L_READS	BIGINT	pool_temp_data_l_reads - 缓冲池临时数据逻辑读取数
POOL_TEMP_DATA_P_READS	BIGINT	pool_temp_data_p_reads - 缓冲池临时数据物理读取数
POOL_TEMP_INDEX_L_READS	BIGINT	pool_temp_index_l_reads - 缓冲池临时索引逻辑读取数
POOL_TEMP_INDEX_P_READS	BIGINT	pool_temp_index_p_reads - 缓冲池临时索引物理读取数
POOL_TEMP_XDA_L_READS	BIGINT	pool_temp_xda_l_reads - 缓冲池临时 XDA 数据逻辑读取数
POOL_TEMP_XDA_P_READS	BIGINT	pool_temp_xda_p_reads - 缓冲池临时 XDA 数据物理读取数
POOL_XDA_L_READS	BIGINT	pool_xda_l_reads - 缓冲池 XDA 数据逻辑读取数
POOL_XDA_P_READS	BIGINT	pool_xda_p_reads - 缓冲池 XDA 数据物理读取数
ROWS_READ	BIGINT	rows_read - 读取行数
ROWS_WRITTEN	BIGINT	rows_written - 写入的行数
SECTION_NUMBER	BIGINT	section_number - 节号

表 68. 对语句事件监视器返回的信息: 缺省表名: STMT_evmon-name (续)

列名	数据类型	描述
SEQUENCE_NO	CHARACTER	sequence_no - 序号
SORT_OVERFLOWS	BIGINT	sort_overflows - 排序溢出数
SQL_REQ_ID	BIGINT	sql_req_id - SQL 语句的请求标识
SQLCABC	INTEGER	请参阅《*SQL Reference Volume 1*》中的『SQLCA（SQL 通信区）』。
SQLCAID	CHARACTER	请参阅《*SQL Reference Volume 1*》中的『SQLCA（SQL 通信区）』。
SQLCODE	INTEGER	请参阅《*SQL Reference Volume 1*》中的『SQLCA（SQL 通信区）』。
SQLERRD1	INTEGER	请参阅《*SQL Reference Volume 1*》中的『SQLCA（SQL 通信区）』。
SQLERRD2	INTEGER	请参阅《*SQL Reference Volume 1*》中的『SQLCA（SQL 通信区）』。
SQLERRD3	INTEGER	请参阅《*SQL Reference Volume 1*》中的『SQLCA（SQL 通信区）』。
SQLERRD4	INTEGER	请参阅《*SQL Reference Volume 1*》中的『SQLCA（SQL 通信区）』。
SQLERRD5	INTEGER	请参阅《*SQL Reference Volume 1*》中的『SQLCA（SQL 通信区）』。
SQLERRD6	INTEGER	请参阅《*SQL Reference Volume 1*》中的『SQLCA（SQL 通信区）』。
SQLERRM	VARCHAR(72)	请参阅《*SQL Reference Volume 1*》中的『SQLCA（SQL 通信区）』。
SQLERRP	CHARACTER	请参阅《*SQL Reference Volume 1*》中的『SQLCA（SQL 通信区）』。
SQLSTATE	CHARACTER	请参阅《*SQL Reference Volume 1*》中的『SQLCA（SQL 通信区）』。
SQLWARN	CHARACTER	请参阅《*SQL Reference Volume 1*》中的『SQLCA（SQL 通信区）』。
START_TIME	TIMESTAMP	start_time - 事件启动时间

表 68. 对语句事件监视器返回的信息：缺省表名：STMT_evmon-name（续）

列名	数据类型	描述
STATS_FABRICATE_TIME	BIGINT	stats_fabricate_time - 生成统计信息的活动所耗的总时间
STMT_OPERATION	BIGINT	
STMT_TYPE	BIGINT	stmt_type - 语句类型
STOP_TIME	TIMESTAMP	stop_time - 事件停止时间
SYNC_RUNSTATS_TIME	BIGINT	第 1188 页的『sync_runstats_time -“同步 RUNSTATS 活动所花的总时间”监视元素』
SYSTEM_CPU_TIME	BIGINT	system_cpu_time - 系统 CPU 时间
TOTAL_SORT_TIME	BIGINT	total_sort_time - 总排序时间
TOTAL_SORTS	BIGINT	total_sorts - 排序总数
USER_CPU_TIME	BIGINT	user_cpu_time - 用户 CPU 时间
STMT_TEXT	CLOB(2097152)	stmt_text - SQL 语句文本

表 69. 对语句事件监视器返回的信息：缺省表名：CONNHEADER_evmon-name

列名	数据类型	描述
AGENT_ID	BIGINT	agent_id - 应用程序句柄（代理程序标识）
APPL_ID	VARCHAR(64)	appl_id - 应用程序标识
APPL_NAME	VARCHAR(255)	appl_name - 应用程序名称
AUTH_ID	VARCHAR(128)	auth_id - 授权标识
CLIENT_DB_ALIAS	CHARACTER	client_db_alias - 应用程序使用的数据库别名
CLIENT_NNAME	VARCHAR(20)	第 661 页的『client_nname -“客户机名称”监视元素』
CLIENT_PID	BIGINT	client_pid - 客户机进程标识
CLIENT_PLATFORM	INTEGER	client_platform - 客户机操作平台
CLIENT_PRDID	VARCHAR(20)	client_prdid - 客户机产品和版本标识
CLIENT_PROTOCOL	INTEGER	client_protocol - 客户机通信协议
CODEPAGE_ID	INTEGER	codepage_id - 应用程序使用的代码页标识
CONN_TIME	TIMESTAMP	conn_time - 数据库连接时间
CORR_TOKEN	VARCHAR(64)	corr_token - DRDA® 关联标记
EXECUTION_ID	VARCHAR(128)	execution_id - 用户登录标识
SEQUENCE_NO	CHARACTER	sequence_no - 序号
TERRITORY_CODE	INTEGER	territory_code - 数据库地域代码

表 70. 对语句事件监视器返回的信息：缺省表名：CONTROL_evmon-name

列名	数据类型	描述
EVENT_MONITOR_NAME	VARCHAR(128)	event_monitor_name - 事件监视器名称
MESSAGE	VARCHAR(128)	message - 控制表消息
MESSAGE_TIME	TIMESTAMP	message_time - 时间戳记控制表消息

表事件监视

表事件监视器生成的数据

表事件监视器为数据库中的表生成聚集度量值。可选择将数据库事件监视器的输出写至常规表、文件或命名管道。

注: 如果想要有关特定表对象的更多详细用法信息，请考虑为该对象创建用法列表。

不管您选择什么输出格式，所有表事件数据都来自 event_table 逻辑数据组。此外，如果选择将语句事件数据写至表，那么另一个组 (CONTROL) 中的数据用于生成有关事件监视器本身的元数据。

表事件监视器写至表的信息:

在指定了 WRITE TO TABLE 选项的情况下表事件监视器写入的信息。

下面的几节说明在 CREATE EVENT MONITOR 语句上使用 WRITE TO TABLE 选项时表事件监视器的输出。有关该事件监视器写至文件或命名管道时返回的输出的信息，请参阅第 98 页的『事件监视器自描述数据流』。

表 71. TABLE 写至表事件监视器生成的表. 表名通过逻辑数据组（用于填充该表）的名称和 CREATE EVENT MONITOR 语句中给予事件监视器的名称（如下表的表名中的 *evmon-name* 所示）并置派生。

缺省表名	报告的逻辑数据组
TABLE_*evmon-name*	event_table
CONTROL_*evmon-name*	CONTROL 逻辑组由来自 event_dbheader、event_start 和 event_overflow 逻辑数据组中的一个或多个的所选元素组成。

要将事件监视器的输出限制为发送至特定表，请在 CREATE EVENT MONITOR 或 ALTER EVENT MONITOR 语句中指定要对其生成表的逻辑组的名称。有关详细信息，请参阅这些语句的参考主题。

生成的表

表 72. 对表事件监视器返回的信息：缺省表名：TABLE_evmon-name

列名	数据类型	描述
DATA_OBJECT_PAGES	BIGINT	data_object_pages - 数据对象页数
DATA_PARTITION_ID	INTEGER	data_partition_id - 数据分区标识

表 72. 对表事件监视器返回的信息：缺省表名：TABLE_evmon-name（续）

列名	数据类型	描述
EVENT_TIME	TIMESTAMP	event_time - 事件时间
EVMON_ACTIVATES	BIGINT	evmon_activates - 事件监视器激活数
EVMON_FLUSHES	BIGINT	evmon_flushes - 事件监视器清空数
INDEX_OBJECT_PAGES	BIGINT	index_object_pages - 索引对象页数
LOB_OBJECT_PAGES	BIGINT	lob_object_pages - LOB 对象页数
LONG_OBJECT_PAGES	BIGINT	long_object_pages - 长对象页数
OVERFLOW_ACCESSES	BIGINT	overflow_accesses - 访问溢出记录次数
PAGE_REORGS	BIGINT	page_reorgs - 页重组
PARTIAL_RECORD	SMALLINT	partial_record - 部分记录
ROWS_READ	BIGINT	rows_read - 读取行数
ROWS_WRITTEN	BIGINT	rows_written - 写入的行数
TABLE_NAME	VARCHAR(128)	table_name - 缺省表名
TABLE_SCHEMA	VARCHAR(128)	table_schema - 表模式名
TABLE_TYPE	BIGINT	table_type - 表类型
TABLESPACE_ID	BIGINT	tablespace_id - 表空间标识
XDA_OBJECT_PAGES	BIGINT	xda_object_pages - XDA 对象页数

表 73. 对表事件监视器返回的信息：缺省表名：CONTROL_evmon-name

列名	数据类型	描述
EVENT_MONITOR_NAME	VARCHAR(128)	event_monitor_name - 事件监视器名称
MESSAGE	VARCHAR(128)	message - 控制表消息
MESSAGE_TIME	TIMESTAMP	message_time - 时间戳记控制表消息

缓冲池事件监视

缓冲池事件监视器生成的数据

缓冲池事件监视器生成有关缓冲池活动的聚集度量值。可选择将数据库事件监视器的输出写至常规表、文件或命名管道。

不管您选择什么输出格式，所有缓冲池事件数据都来自 event_bufferpool 逻辑数据组。此外，如果选择将语句事件数据写至表，那么另一个组 (CONTROL) 中的数据用于生成有关事件监视器本身的元数据。

缓冲池事件监视器写至表的信息:

在指定了 WRITE TO TABLE 选项的情况下缓冲池事件监视器写入的信息。

下面的几节说明在 CREATE EVENT MONITOR 语句上使用 WRITE TO TABLE 选项时缓冲池事件监视器的输出。有关该事件监视器写至文件或命名管道时返回的输出的信息，请参阅第 98 页的『事件监视器自描述数据流』。

表 74. BUFFERPOOL 写至表事件监视器生成的表. 表名通过逻辑数据组（用于填充该表）的名称和 CREATE EVENT MONITOR 语句中给予事件监视器的名称（如下表的表名中的 *evmon-name* 所示）并置派生。

缺省表名	报告的逻辑数据组
BUFFERPOOL_*evmon-name*	event_bufferpool
CONTROL_*evmon-name*	CONTROL 逻辑组由来自 event_dbheader、event_start 和 event_overflow 逻辑数据组中的一个或多个的所选元素组成。

要将事件监视器的输出限制为发送至特定表，请在 CREATE EVENT MONITOR 或 ALTER EVENT MONITOR 语句中指定要对其生成表的逻辑组的名称。有关详细信息，请参阅这些语句的参考主题。

生成的表

表 75. 对缓冲池事件监视器返回的信息：缺省表名：BUFFERPOOL_evmon-name

列名	数据类型	描述
BLOCK_IOS	BIGINT	block_ios - 块 I/O 请求数
BP_NAME	VARCHAR(20)	bp_name - 缓冲池名称
DB_NAME	CHARACTER(8)	db_name - 数据库名称
DB_PATH	VARCHAR(215)	db_path - 数据库路径
DIRECT_READ_REQS	BIGINT	direct_read_reqs - 直接读请求数
DIRECT_READ_TIME	BIGINT	direct_read_time - 直接读时间
DIRECT_READS	BIGINT	direct_reads - 直接读数据库数目
DIRECT_WRITE_REQS	BIGINT	direct_write_reqs - 直接写请求数
DIRECT_WRITE_TIME	BIGINT	direct_write_time - 直接写时间
DIRECT_WRITES	BIGINT	direct_writes - 直接写数据库数目
EVENT_TIME	TIMESTAMP	event_time - 事件时间
EVMON_ACTIVATES	BIGINT	evmon_activates - 事件监视器激活数
EVMON_FLUSHES	BIGINT	evmon_flushes - 事件监视器清空数
FILES_CLOSED	BIGINT	files_closed - 关闭数据库文件数
PAGES_FROM_BLOCK_IOS	BIGINT	pages_from_block_ios - 块 I/O 读取总页数
PAGES_FROM_VECTORED_IOS	BIGINT	pages_from_vectored_ios - 向量 I/O 读取总页数
PARTIAL_RECORD	SMALLINT	partial_record - 部分记录
POOL_ASYNC_DATA_READ_REQS	BIGINT	pool_async_data_read_reqs - 缓冲池异步读请求数
POOL_ASYNC_DATA_READS	BIGINT	pool_async_data_reads - 缓冲池异步数据读次数
POOL_ASYNC_DATA_WRITES	BIGINT	pool_async_data_writes - 缓冲池异步数据写次数
POOL_ASYNC_INDEX_READ_REQS	BIGINT	pool_async_index_read_reqs - 缓冲池异步索引读请求数
POOL_ASYNC_INDEX_READS	BIGINT	pool_async_index_reads - 缓冲池异步索引读取数

表 75. 对缓冲池事件监视器返回的信息：缺省表名：BUFFERPOOL_evmon-name（续）

列名	数据类型	描述
POOL_ASYNC_INDEX_WRITES	BIGINT	pool_async_index_writes - 缓冲池异步索引写次数
POOL_ASYNC_READ_TIME	BIGINT	pool_async_read_time - 缓冲池异步读取时间
POOL_ASYNC_WRITE_TIME	BIGINT	pool_async_write_time - 缓冲池异步写入时间
POOL_ASYNC_XDA_READ_REQS	BIGINT	pool_async_xda_read_reqs - 缓冲池异步 XDA 读请求数
POOL_ASYNC_XDA_READS	BIGINT	pool_async_xda_reads - 缓冲池异步 XDA 数据读取数
POOL_ASYNC_XDA_WRITES	BIGINT	pool_async_xda_writes - 缓冲池异步 XDA 数据写次数
POOL_DATA_L_READS	BIGINT	pool_data_l_reads - 缓冲池数据逻辑读取数
POOL_DATA_P_READS	BIGINT	pool_data_p_reads - 缓冲池数据物理读取数
POOL_DATA_WRITES	BIGINT	pool_data_writes - 缓冲池数据写次数
POOL_INDEX_L_READS	BIGINT	pool_index_l_reads - 缓冲池索引逻辑读取数
POOL_INDEX_P_READS	BIGINT	pool_index_p_reads - 缓冲池索引物理读取数
POOL_INDEX_WRITES	BIGINT	pool_index_writes - 缓冲池索引写次数
POOL_NO_VICTIM_BUFFER	BIGINT	pool_no_victim_buffer - 缓冲池无牺牲缓冲区次数
POOL_READ_TIME	BIGINT	pool_read_time - 缓冲池物理读时间总计
POOL_WRITE_TIME	BIGINT	pool_write_time - 缓冲池物理写时间总计
POOL_XDA_L_READS	BIGINT	pool_xda_l_reads - 缓冲池 XDA 数据逻辑读取数
POOL_XDA_P_READS	BIGINT	pool_xda_p_reads - 缓冲池 XDA 数据物理读取数
POOL_XDA_WRITES	BIGINT	pool_xda_writes - 缓冲池 XDA 数据写次数
UNREAD_PREFETCH_PAGES	BIGINT	unread_prefetch_pages - 未读取的预取页数
VECTORED_IOS	BIGINT	vectored_ios - 向量 I/O 请求数

表 76. 对缓冲池事件监视器返回的信息：缺省表名：CONTROL_evmon-name

列名	数据类型	描述
EVENT_MONITOR_NAME	VARCHAR(128)	event_monitor_name - 事件监视器名称
MESSAGE	VARCHAR(128)	message - 控制表消息
MESSAGE_TIME	TIMESTAMP	message_time - 时间戳记控制表消息

表空间事件监视

表空间事件监视器生成的数据

表空间事件监视器为数据库中的表空间生成聚集度量值。可选择将数据库事件监视器的输出写至常规表、文件或命名管道。

不管您选择什么输出格式，所有表空间事件数据都来自 event_tablespace 逻辑数据组。此外，如果选择将语句事件数据写至表，那么另一个组 (CONTROL) 中的数据用于生成有关事件监视器本身的元数据。

表空间事件监视器写至表的信息:

在指定了 WRITE TO TABLE 选项的情况下表空间事件监视器写入的信息。

下面的几节说明在 CREATE EVENT MONITOR 语句上使用 WRITE TO TABLE 选项时表空间事件监视器的输出。有关该事件监视器写至文件或命名管道时返回的输出的信息，请参阅第 98 页的『事件监视器自描述数据流』。

表 77. TABLESPACE 写至表事件监视器生成的表. 表名通过逻辑数据组（用于填充该表）的名称和 CREATE EVENT MONITOR 语句中给予事件监视器的名称（如下表的表名中的 *evmon-name* 所示）并置派生。

缺省表名	报告的逻辑数据组
TABLESPACE_*evmon-name*	event_tablespace
CONTROL_*evmon-name*	CONTROL 逻辑组由来自 event_dbheader、event_start 和 event_overflow 逻辑数据组中的一个或多个的所选元素组成。

要将事件监视器的输出限制为发送至特定表，请在 CREATE EVENT MONITOR 或 ALTER EVENT MONITOR 语句中指定要对其生成表的逻辑组的名称。有关详细信息，请参阅这些语句的参考主题。

生成的表

表 78. 对表空间事件监视器返回的信息: 缺省表名: TABLESPACE_evmon-name

列名	数据类型	描述
DIRECT_READ_REQS	BIGINT	direct_read_reqs - 直接读请求数
DIRECT_READ_TIME	BIGINT	direct_read_time - 直接读时间
DIRECT_READS	BIGINT	direct_reads - 直接读数据库数目
DIRECT_WRITE_REQS	BIGINT	direct_write_reqs - 直接写请求数
DIRECT_WRITE_TIME	BIGINT	direct_write_time - 直接写时间
DIRECT_WRITES	BIGINT	direct_writes - 直接写数据库数目
EVENT_TIME	TIMESTAMP	event_time - 事件时间
EVMON_ACTIVATES	BIGINT	evmon_activates - 事件监视器激活数
EVMON_FLUSHES	BIGINT	evmon_flushes - 事件监视器清空数
FILES_CLOSED	BIGINT	files_closed - 关闭数据库文件数
PARTIAL_RECORD	SMALLINT	partial_record - 部分记录
POOL_ASYNC_DATA_READ_REQS	BIGINT	pool_async_data_read_reqs - 缓冲池异步读请求数
POOL_ASYNC_DATA_READS	BIGINT	pool_async_data_reads - 缓冲池异步数据读次数
POOL_ASYNC_DATA_WRITES	BIGINT	pool_async_data_writes - 缓冲池异步数据写次数
POOL_ASYNC_INDEX_READ_REQS	BIGINT	pool_async_index_read_reqs - 缓冲池异步索引读请求数
POOL_ASYNC_INDEX_READS	BIGINT	pool_async_index_reads - 缓冲池异步索引读取数
POOL_ASYNC_INDEX_WRITES	BIGINT	pool_async_index_writes - 缓冲池异步索引写次数
POOL_ASYNC_READ_TIME	BIGINT	pool_async_read_time - 缓冲池异步读取时间
POOL_ASYNC_WRITE_TIME	BIGINT	pool_async_write_time - 缓冲池异步写入时间
POOL_ASYNC_XDA_READ_REQS	BIGINT	pool_async_xda_read_reqs - 缓冲池异步 XDA 读请求数
POOL_ASYNC_XDA_READS	BIGINT	pool_async_xda_reads - 缓冲池异步 XDA 数据读取数
POOL_ASYNC_XDA_WRITES	BIGINT	pool_async_xda_writes - 缓冲池异步 XDA 数据写次数
POOL_DATA_L_READS	BIGINT	pool_data_l_reads - 缓冲池数据逻辑读取数

表 78. 对表空间事件监视器返回的信息: 缺省表名: TABLESPACE_evmon-name (续)

列名	数据类型	描述
POOL_DATA_P_READS	BIGINT	pool_data_p_reads - 缓冲池数据物理读取数
POOL_DATA_WRITES	BIGINT	pool_data_writes - 缓冲池数据写次数
POOL_INDEX_L_READS	BIGINT	pool_index_l_reads - 缓冲池索引逻辑读取数
POOL_INDEX_P_READS	BIGINT	pool_index_p_reads - 缓冲池索引物理读取数
POOL_INDEX_WRITES	BIGINT	pool_index_writes - 缓冲池索引写次数
POOL_NO_VICTIM_BUFFER	BIGINT	pool_no_victim_buffer - 缓冲池无牺牲缓冲区次数
POOL_READ_TIME	BIGINT	pool_read_time - 缓冲池物理读时间总计
POOL_WRITE_TIME	BIGINT	pool_write_time - 缓冲池物理写时间总计
POOL_XDA_L_READS	BIGINT	pool_xda_l_reads - 缓冲池 XDA 数据逻辑读取数
POOL_XDA_P_READS	BIGINT	pool_xda_p_reads - 缓冲池 XDA 数据物理读取数
POOL_XDA_WRITES	BIGINT	pool_xda_writes - 缓冲池 XDA 数据写次数
TABLESPACE_FS_CACHING	SMALLINT	fs_caching - 文件系统高速缓存
TABLESPACE_NAME	VARCHAR(18)	tablespace_name - 表空间名称
UNREAD_PREFETCH_PAGES	BIGINT	unread_prefetch_pages - 未读取的预取页数

表 79. 对表空间事件监视器返回的信息: 缺省表名: CONTROL_evmon-name

列名	数据类型	描述
EVENT_MONITOR_NAME	VARCHAR(128)	event_monitor_name - 事件监视器名称
MESSAGE	VARCHAR(128)	message - 控制表消息
MESSAGE_TIME	TIMESTAMP	message_time - 时间戳记控制表消息

连接事件监视

连接事件监视器生成的数据

连接事件监视器捕获应用程序建立的与数据库的每个连接的度量值和其他监视元素。可选择将输出写至文件、命名管道或常规表。

不管您选择什么输出格式，所有连接事件数据都来自下列三个逻辑组的其中一个:

- event_connheader
- event_conn
- event_connmemuse

此外，如果选择将连接事件数据写至表，那么另一个组 (CONTROL) 中的数据用于生成有关事件监视器本身的元数据。

连接事件监视器写至表的信息:

在指定了 WRITE TO TABLE 选项的情况下连接事件监视器写入的信息。

选择 WRITE TO TABLE 作为连接事件监视器的输出类型时，缺省情况下，会生成四个表，每个表包含一个或多个逻辑数据组中的监视元素:

表 80. CONNECTIONS 写至表事件监视器生成的表. 表名通过逻辑数据组（用于填充该表）的名称和 CREATE EVENT MONITOR 语句中给予事件监视器的名称（如下表的表名中的 *evmon-name* 所示）并置派生。

缺省表名	报告的逻辑数据组
CONNHEADER_*evmon-name*	event_connheader
CONN_*evmon-name*	event_conn
CONMEMUSE_*evmon-name*	event_connmemuse
CONTROL_*evmon-name*	CONTROL 逻辑组由来自 event_dbheader、event_start 和 event_overflow 逻辑数据组中的一个或多个的所选元素组成。

要将事件监视器的输出限制为发送至特定表，请在 CREATE EVENT MONITOR 或 ALTER EVENT MONITOR 语句中指定要对其生成表的逻辑组的名称。有关详细信息，请参阅这些语句的参考主题。

有关该事件监视器写至文件或命名管道时返回的输出的信息，请参阅第 98 页的『事件监视器自描述数据流』。

生成的表

表 81. 对连接事件监视器返回的信息: 缺省表名: CONNHEADER_evmon-name

列名	数据类型	描述
AGENT_ID	BIGINT	agent_id - 应用程序句柄（代理程序标识）
APPL_ID	VARCHAR(64)	appl_id - 应用程序标识
APPL_NAME	VARCHAR(255)	appl_name - 应用程序名称
AUTH_ID	VARCHAR(128)	auth_id - 授权标识
CLIENT_DB_ALIAS	CHAR(8)	client_db_alias - 应用程序使用的数据库别名
CLIENT_NNAME	VARCHAR(20)	第 661 页的『client_nname -“客户机名称”监视元素』
CLIENT_PID	BIGINT	client_pid - 客户机进程标识
CLIENT_PLATFORM	INTEGER	client_platform - 客户机操作平台
CLIENT_PRDID	VARCHAR(20)	client_prdid - 客户机产品和版本标识
CLIENT_PROTOCOL	INTEGER	client_protocol - 客户机通信协议
CODEPAGE_ID	INTEGER	codepage_id - 应用程序使用的代码页标识
CONN_TIME	TIMESTAMP	conn_time - 数据库连接时间
CORR_TOKEN	VARCHAR(64)	corr_token - DRDA 关联标记
EXECUTION_ID	VARCHAR(128)	execution_id - 用户登录标识
SEQUENCE_NO	CHAR(5)	sequence_no - 序号
TERRITORY_CODE	INTEGER	territory_code - 数据库地域代码

表 82. 对连接事件监视器返回的信息: 表名: CONN_evmon-name

列名	数据类型	描述
ACC_CURS_BLK	BIGINT	acc_curs_blk - 接受的块游标请求数

表 82. 对连接事件监视器返回的信息: 表名: CONN_evmon-name (续)

列名	数据类型	描述
AGENT_ID	BIGINT	agent_id - 应用程序句柄（代理程序标识）
APPL_ID	VARCHAR(64)	appl_id - 应用程序标识
APPL_PRIORITY	BIGINT	appl_priority - 应用程序代理程序优先级
APPL_PRIORITY_TYPE	BIGINT	appl_priority_type - 应用程序优先级类型
APPL_SECTION_INSERTS	BIGINT	appl_section_inserts - 节插入数
APPL_SECTION_LOOKUPS	BIGINT	appl_section_lookups - 节查询数
APPL_STATUS	BIGINT	appl_status - 应用程序状态
AUTHORITY_BITMAP	CHARACTER(22)	authority_bitmap- 用户权限级别
AUTHORITY_LVL	BIGINT	authority_lvl - 用户权限级别
BINDS_PRECOMPILES	BIGINT	binds_precompiles - 尝试的绑定次数/预编译次数
CAT_CACHE_HEAP_FULL	BIGINT	
CAT_CACHE_INSERTS	BIGINT	cat_cache_inserts - 目录高速缓存插入数
CAT_CACHE_LOOKUPS	BIGINT	cat_cache_lookups - 目录高速缓存查询数
CAT_CACHE_OVERFLOWS	BIGINT	cat_cache_overflows - 目录高速缓存溢出数
CAT_CACHE_SIZE_TOP	BIGINT	cat_cache_size_top - 目录高速缓存高水位标记
COMMIT_SQL_STMTS	BIGINT	commit_sql_stmts - 尝试的落实语句数
COORD_NODE	BIGINT	coord_node - 协调节点
DDL_SQL_STMTS	BIGINT	ddl_sql_stmts - 数据定义语言（DDL）SQL 语句数
DEADLOCKS	BIGINT	deadlocks - 检测到的死锁数
DIRECT_READ_REQS	BIGINT	direct_read_reqs - 直接读请求数
DIRECT_READ_TIME	BIGINT	direct_read_time - 直接读时间
DIRECT_READS	BIGINT	direct_reads - 直接读数据库数目
DIRECT_WRITE_REQS	BIGINT	direct_write_reqs - 直接写请求数
DIRECT_WRITE_TIME	BIGINT	direct_write_time - 直接写时间
DIRECT_WRITES	BIGINT	direct_writes - 直接写数据库数目
DISCONN_TIME	TIMESTAMP	disconn_time - 数据库释放时间戳记
DYNAMIC_SQL_STMTS	BIGINT	dynamic_sql_stmts - 尝试的动态 SQL 语句数
ELAPSED_EXEC_TIME	BIGINT	elapsed_exec_time - 语句执行耗用时间
EVMON_FLUSHES	BIGINT	evmon_flushes - 事件监视器清空数
FAILED_SQL_STMTS	BIGINT	failed_sql_stmts - 失败的语句操作
HASH_JOIN_OVERFLOWS	BIGINT	hash_join_overflows - 散列连接溢出数
HASH_JOIN_SMALL_OVERFLOWS	BIGINT	hash_join_small_overflows - 散列连接小溢出数
INT_AUTO_REBINDS	BIGINT	int_auto_rebinds - 内部自动重新绑定次数
INT_COMMITS	BIGINT	int_commits - 内部落实数

表 82. 对连接事件监视器返回的信息: 表名: CONN_evmon-name (续)

列名	数据类型	描述
INT_DEADLOCK_ROLLBACKS	BIGINT	int_deadlock_rollbacks - 死锁导致的内部回滚数
INT_ROLLBACKS	BIGINT	int_rollbacks - 内部回滚
INT_ROWS_DELETED	BIGINT	int_rows_deleted - 删除的内部行数
INT_ROWS_INSERTED	BIGINT	int_rows_inserted - 插入的内部行数
INT_ROWS_UPDATED	BIGINT	int_rows_updated - 更新的内部行数
LOCK_ESCALS	BIGINT	lock_escals - 锁定升级次数
LOCK_TIMEOUTS	BIGINT	lock_timeouts - 锁定超时次数
LOCK_WAIT_TIME	BIGINT	lock_wait_time - 等待锁定时间
LOCK_WAITS	BIGINT	lock_waits - 等待锁定次数
OLAP_FUNC_OVERFLOWS	BIGINT	olap_func_overflows - OLAP 函数溢出次数
PARTIAL_RECORD	SMALLINT	partial_record - 部分记录
PKG_CACHE_INSERTS	BIGINT	pkg_cache_inserts - 程序包高速缓存插入数
PKG_CACHE_LOOKUPS	BIGINT	pkg_cache_lookups - 程序包高速缓存查询数
POOL_DATA_L_READS	BIGINT	pool_data_l_reads - 缓冲池数据逻辑读取数
POOL_DATA_P_READS	BIGINT	pool_data_p_reads - 缓冲池数据物理读取数
POOL_DATA_WRITES	BIGINT	pool_data_writes - 缓冲池数据写次数
POOL_INDEX_L_READS	BIGINT	pool_index_l_reads - 缓冲池索引逻辑读取数
POOL_INDEX_P_READS	BIGINT	pool_index_p_reads - 缓冲池索引物理读取数
POOL_INDEX_WRITES	BIGINT	pool_index_writes - 缓冲池索引写次数
POOL_READ_TIME	BIGINT	pool_read_time - 缓冲池物理读时间总计
POOL_TEMP_DATA_L_READS	BIGINT	pool_temp_data_l_reads - 缓冲池临时数据逻辑读取数
POOL_TEMP_DATA_P_READS	BIGINT	pool_temp_data_p_reads - 缓冲池临时数据物理读取数
POOL_TEMP_INDEX_L_READS	BIGINT	pool_temp_index_l_reads - 缓冲池临时索引逻辑读取数
POOL_TEMP_INDEX_P_READS	BIGINT	pool_temp_index_p_reads - 缓冲池临时索引物理读取数
POOL_TEMP_XDA_L_READS	BIGINT	pool_temp_xda_l_reads - 缓冲池临时 XDA 数据逻辑读取数
POOL_TEMP_XDA_P_READS	BIGINT	pool_temp_xda_p_reads - 缓冲池临时 XDA 数据物理读取数
POOL_WRITE_TIME	BIGINT	pool_write_time - 缓冲池物理写时间总计
POOL_XDA_L_READS	BIGINT	pool_xda_l_reads - 缓冲池 XDA 数据逻辑读取数

表 82. 对连接事件监视器返回的信息：表名：CONN_evmon-name（续）

列名	数据类型	描述
POOL_XDA_P_READS	BIGINT	pool_xda_p_reads - 缓冲池 XDA 数据物理读取数
POOL_XDA_WRITES	BIGINT	pool_xda_writes - 缓冲池 XDA 数据写次数
PREFETCH_WAIT_TIME	BIGINT	prefetch_wait_time - 等待预取的时间
REJ_CURS_BLK	BIGINT	rej_curs_blk - 拒绝的块游标请求数
ROLLBACK_SQL_STMTS	BIGINT	rollback_sql_stmts - 尝试的回滚语句数
ROWS_DELETED	BIGINT	rows_deleted - 删除行数
ROWS_INSERTED	BIGINT	rows_inserted - 插入行数
ROWS_READ	BIGINT	rows_read - 读取行数
ROWS_SELECTED	BIGINT	rows_selected - 选择的行数
ROWS_UPDATED	BIGINT	rows_updated - 更新行数
ROWS_WRITTEN	BIGINT	rows_written - 写入的行数
SELECT_SQL_STMTS	BIGINT	select_sql_stmts - 执行的 Select SQL 语句数
SEQUENCE_NO	CHARACTER(5)	sequence_no - 序号
SORT_OVERFLOWS	BIGINT	sort_overflows - 排序溢出数
STATIC_SQL_STMTS	BIGINT	static_sql_stmts - 尝试的静态 SQL 语句数
SYSTEM_CPU_TIME	BIGINT	system_cpu_time - 系统 CPU 时间
TOTAL_HASH_JOINS	BIGINT	total_hash_joins - 散列连接总数
TOTAL_HASH_LOOPS	BIGINT	total_hash_loops - 总散列循环数
TOTAL_OLAP_FUNCS	BIGINT	total_olap_funcs - OLAP 函数总数
TOTAL_SORT_TIME	BIGINT	total_sort_time - 总排序时间
TOTAL_SORTS	BIGINT	total_sorts - 排序总数
UID_SQL_STMTS	BIGINT	uid_sql_stmts - 执行的 UPDATE/INSERT/DELETE SQL 语句数
UNREAD_PREFETCH_PAGES	BIGINT	unread_prefetch_pages - 未读取的预取页数
USER_CPU_TIME	BIGINT	user_cpu_time - 用户 CPU 时间
X_LOCK_ESCALS	BIGINT	x_lock_escals - 互斥锁定升级数
XQUERY_STMTS	BIGINT	xquery_stmts - 尝试的 XQuery 语句数

表 83. 对连接事件监视器返回的信息：表名：CONMEMUSE_evmon-name

列名	数据类型	描述
APPL_ID	VARCHAR(64)	appl_id - 应用程序标识
EVMON_FLUSHES	BIGINT	evmon_flushes - 事件监视器清空数
POOL_CUR_SIZE	BIGINT	pool_cur_size - 内存池的当前大小
POOL_ID	BIGINT	pool_id - 内存池标识
POOL_LIST_ID	BIGINT	
POOL_MAX_SIZE	BIGINT	

表 83. 对连接事件监视器返回的信息: 表名: CONMEMUSE_evmon-name (续)

列名	数据类型	描述
POOL_WATERMARK	BIGINT	pool_watermark - 内存池水位标记

表 84. 对连接事件监视器返回的信息: 缺省表名: CONTROL_evmon-name

列名	数据类型	描述
APPL_ID	VARCHAR(64)	appl_id - 应用程序标识
EVMON_FLUSHES	BIGINT	evmon_flushes - 事件监视器清空数
POOL_CUR_SIZE	BIGINT	pool_cur_size - 内存池的当前大小
POOL_ID	BIGINT	pool_id - 内存池标识
POOL_LIST_ID	BIGINT	
POOL_MAX_SIZE	BIGINT	
POOL_WATERMARK	BIGINT	pool_watermark - 内存池水位标记

事务事件监视

事务事件监视器生成的数据

事务事件监视器记录有关数据库事务的信息。

注: 建议不要使用此事件监视器。不再建议使用此事件监视器，将来的发行版可能会将其除去。请改用工作单元事件监视器来监视工作单元。

不管您选择什么输出格式，所有事务事件数据都来自下列两个逻辑组:

- 第 64 页的『event_xact 逻辑数据组』
- 第 49 页的『event_connheader 逻辑数据组』

此外，如果选择将事务事件数据写至表，那么另一个组 (CONTROL) 中的数据用于生成有关事件监视器本身的元数据。

事务事件监视器写至表的信息:

在指定了 WRITE TO TABLE 的情况下事务事件监视器写入的信息。

下面的几节说明在 CREATE EVENT MONITOR 语句上使用 WRITE TO TABLE 选项时事务事件监视器的输出。有关该事件监视器写至文件或命名管道时返回的输出的信息，请参阅第 98 页的『事件监视器自描述数据流』。

表 85. TRANSACTION 写至表事件监视器生成的表. 表名通过逻辑数据组（用于填充该表）的名称和 CREATE EVENT MONITOR 语句中给予事件监视器的名称（如下表的表名中的 *evmon-name* 所示）并置派生。

缺省表名	报告的逻辑数据组
XACT_*evmon-name*	第 64 页的『event_xact 逻辑数据组』
CONNHEADER_*evmon-name*	第 49 页的『event_connheader 逻辑数据组』
CONTROL_*evmon-name*	CONTROL 逻辑组由来自 event_dbheader、event_start 和 event_overflow 逻辑数据组中的一个或多个的所选元素组成。

生成的表

表 86. 对事务事件监视器返回的信息: 缺省表名: XACT_evmon-name

列名	数据类型	描述
AGENT_ID	BIGINT	agent_id - 应用程序句柄（代理程序标识）
APPL_ID	VARCHAR(64)	appl_id - 应用程序标识
EVMON_FLUSHES	BIGINT	evmon_flushes - 事件监视器清空数
LOCK_ESCALS	BIGINT	lock_escals - 锁定升级次数
LOCK_WAIT_TIME	BIGINT	lock_wait_time - 等待锁定时间
LOCKS_HELD_TOP	BIGINT	locks_held_top - 挂起的最大锁定数
PARTIAL_RECORD	SMALLINT	partial_record - 部分记录
PREV_UOW_STOP_TIME	TIMESTAMP	prev_uow_stop_time - 上一个工作单元完成时间戳记
ROWS_READ	BIGINT	rows_read - 读取行数
ROWS_WRITTEN	BIGINT	rows_written - 写入的行数
SEQUENCE_NO	CHARACTER	sequence_no - 序号
STOP_TIME	TIMESTAMP	stop_time - 事件停止时间
SYSTEM_CPU_TIME	BIGINT	system_cpu_time - 系统 CPU 时间
TPMON_ACC_STR	VARCHAR(200)	tpmon_acc_str - TP 监视器客户机记帐字符串
TPMON_CLIENT_APP	VARCHAR(255)	tpmon_client_app - TP 监视器客户机应用程序名称
TPMON_CLIENT_USERID	VARCHAR(255)	tpmon_client_userid - TP 监视器客户机用户标识
TPMON_CLIENT_WKSTN	VARCHAR(255)	tpmon_client_wkstn - TP 监视器客户机工作站名称
UOW_LOG_SPACE_USED	BIGINT	uow_log_space_used - 使用的工作单元日志空间
UOW_START_TIME	TIMESTAMP	uow_start_time - 工作单元开始时间戳记
UOW_STATUS	BIGINT	uow_status - 工作单元状态
USER_CPU_TIME	BIGINT	user_cpu_time - 用户 CPU 时间
X_LOCK_ESCALS	BIGINT	x_lock_escals - 互斥锁定升级数

表 87. 对事务事件监视器返回的信息: 缺省表名: CONNHEADER_evmon-name

列名	数据类型	描述
AGENT_ID	BIGINT	agent_id - 应用程序句柄（代理程序标识）
APPL_ID	VARCHAR(64)	appl_id - 应用程序标识
APPL_NAME	VARCHAR(255)	appl_name - 应用程序名称
AUTH_ID	VARCHAR(128)	auth_id - 授权标识
CLIENT_DB_ALIAS	CHARACTER(8)	client_db_alias - 应用程序使用的数据库别名
CLIENT_NNAME	VARCHAR(20)	第 661 页的『client_nname -“客户机名称”监视元素』
CLIENT_PID	BIGINT	client_pid - 客户机进程标识
CLIENT_PLATFORM	INTEGER	client_platform - 客户机操作平台

表 87. 对事务事件监视器返回的信息: 缺省表名: CONNHEADER_evmon-name (续)

列名	数据类型	描述
CLIENT_PRDID	VARCHAR(20)	client_prdid - 客户机产品和版本标识
CLIENT_PROTOCOL	INTEGER	client_protocol - 客户机通信协议
CODEPAGE_ID	INTEGER	codepage_id - 应用程序使用的代码页标识
CONN_TIME	TIMESTAMP	conn_time - 数据库连接时间
CORR_TOKEN	VARCHAR(64)	corr_token - DRDA 关联标记
EXECUTION_ID	VARCHAR(128)	execution_id - 用户登录标识
SEQUENCE_NO	CHARACTER(5)	sequence_no - 序号
TERRITORY_CODE	INTEGER	territory_code - 数据库地域代码

表 88. 对事务事件监视器返回的信息: 缺省表名: CONTROL_evmon-name

列名	数据类型	描述
EVENT_MONITOR_NAME	VARCHAR(128)	event_monitor_name - 事件监视器名称
MESSAGE	VARCHAR(128)	message - 控制表消息
MESSAGE_TIME	TIMESTAMP	message_time - 时间戳记控制表消息

死锁事件监视

死锁事件监视器生成的数据

死锁事件监视器记录有关死锁情况的信息。

注: 建议不要使用此事件监视器。建议不要再使用此事件监视器，将来的发行版可能会将其除去。请改用锁定事件监视器来监视死锁。

不管您选择什么输出格式，所有死锁事件数据都来自下列三个逻辑组:

- 第 49 页的『event_connheader 逻辑数据组』
- 第 54 页的『event_deadlock 逻辑数据组』
- 第 55 页的『event_dlconn 逻辑数据组』

此外，如果选择将事务事件数据写至表，那么另一个组 (CONTROL) 中的数据用于生成有关事件监视器本身的元数据。

死锁事件监视器写至表的信息:

在指定了 WRITE TO TABLE 选项的情况下死锁事件监视器写入的信息。

下面的几节说明在 CREATE EVENT MONITOR 语句上使用 WRITE TO TABLE 选项时死锁事件监视器的输出。有关该事件监视器写至文件或命名管道时返回的输出的信息，请参阅第 98 页的『事件监视器自描述数据流』。

表 89. DEADLOCK"写至表"事件监视器生成的表. 表名通过逻辑数据组（用于填充该表）的名称和 CREATE EVENT MONITOR 语句中给予事件监视器的名称（如下表的表名中的 *evmon-name* 所示）并置派生。

缺省表名	报告的逻辑数据组
CONNHEADER_*evmon-name*	第 49 页的『event_connheader 逻辑数据组』

表 89. DEADLOCK“写至表”事件监视器生成的表 (续). 表名通过逻辑数据组（用于填充该表）的名称和 CREATE EVENT MONITOR 语句中给予事件监视器的名称（如下表的表名中的 *evmon-name* 所示）并置派生。

缺省表名	报告的逻辑数据组
DEADLOCK_*evmon-name*	第 54 页的『event_deadlock 逻辑数据组』
DLCONN_*evmon-name*	第 55 页的『event_dlconn 逻辑数据组』
CONTROL_*evmon-name*	CONTROL 逻辑组由来自 event_dbheader、event_start 和 event_overflow 逻辑数据组中的一个或多个的所选元素组成。

生成的表

表 90. 对死锁事件监视器返回的信息: 缺省表名: CONNHEADER_evmon-name

列名	数据类型	描述
AGENT_ID	BIGINT	agent_id -“应用程序句柄（代理程序标识）”监视元素
APPL_ID	VARCHAR(64)	appl_id -“应用程序标识”监视元素
APPL_NAME	VARCHAR(255)	appl_name -“应用程序名称”监视元素
AUTH_ID	VARCHAR(128)	auth_id -“授权标识”监视元素
CLIENT_DB_ALIAS	CHARACTER(8)	client_db_alias -“应用程序使用的数据库别名”监视元素
CLIENT_NNAME	VARCHAR(20)	第 661 页的『client_nname -“客户机名称”监视元素』
CLIENT_PID	BIGINT	client_pid -“客户机进程标识”监视元素
CLIENT_PLATFORM	INTEGER	client_platform -“客户机操作平台”监视元素
CLIENT_PRDID	VARCHAR(20)	client_prdid -“客户机产品和版本标识”监视元素
CLIENT_PROTOCOL	INTEGER	client_protocol -“客户机通信协议”监视元素
CODEPAGE_ID	INTEGER	codepage_id -“应用程序使用的代码页的标识”监视元素
CONN_TIME	TIMESTAMP	conn_time -“数据库连接时间”监视元素
CORR_TOKEN	VARCHAR(64)	corr_token -“DRDA 关联标记”监视元素
EXECUTION_ID	VARCHAR(128)	execution_id -“用户登录标识”监视元素
SEQUENCE_NO	CHARACTER(5)	sequence_no -“序号”监视元素
TERRITORY_CODE	INTEGER	territory_code -“数据库地域代码”监视元素

表 91. 对死锁事件监视器返回的信息: 缺省表名: DEADLOCK_evmon-name

列名	数据类型	描述
DEADLOCK_ID	BIGINT	deadlock_id -“死锁事件标识”监视元素
DL_CONNS	BIGINT	dl_conns -“死锁中涉及的连接数”监视元素
EVMON_ACTIVATES	BIGINT	evmon_activates -“事件监视器激活数”监视元素
ROLLED_BACK_AGENT_ID	BIGINT	rolled_back_agent_id -“回滚的代理程序”监视元素
ROLLED_BACK_APPL_ID	VARCHAR(64)	rolled_back_appl_id -“回滚的应用程序”监视元素
ROLLED_BACK_PARTICIPANT_NO	SMALLINT	rolled_back_participant_no -“回滚的应用程序参与者”监视元素
ROLLED_BACK_SEQUENCE_NO	CHARACTER(5)	rolled_back_sequence_no -“回滚的序号”监视元素
START_TIME	TIMESTAMP	start_time -“事件开始时间”监视元素

表 92. 对死锁事件监视器返回的信息: 缺省表名: DLCONN_evmon-name

列名	数据类型	描述
AGENT_ID	BIGINT	agent_id -“应用程序句柄（代理程序标识）”监视元素
APPL_ID	VARCHAR(64)	appl_id -“应用程序标识”监视元素
APPL_ID_HOLDING_LK	VARCHAR(64)	appl_id_holding_lk -“挂起锁定的应用程序标识”监视元素
DATA_PARTITION_ID	INTEGER	data_partition_id -“数据分区标识”监视元素
DEADLOCK_ID	BIGINT	deadlock_id -“死锁事件标识”监视元素
EVMON_ACTIVATES	BIGINT	evmon_activates -“事件监视器激活数”监视元素
LOCK_ATTRIBUTES	BIGINT	lock_attributes -“锁定属性”监视元素
LOCK_COUNT	BIGINT	lock_count -“锁定计数”监视元素
LOCK_CURRENT_MODE	BIGINT	lock_current_mode -“转换前的原始锁定方式”监视元素
LOCK_ESCALATION	SMALLINT	lock_escalation -“锁定升级”监视元素
LOCK_HOLD_COUNT	BIGINT	lock_hold_count -“锁定挂起计数”监视元素
LOCK_MODE	BIGINT	lock_mode -“锁定方式”监视元素
LOCK_MODE_REQUESTED	BIGINT	lock_mode_requested -“请求的锁定方式”监视元素
LOCK_NAME	CHARACTER(13)	lock_name -“锁定名称”监视元素
LOCK_NODE	BIGINT	lock_node -“锁定节点”监视元素
LOCK_OBJECT_NAME	BIGINT	lock_object_name -“锁定对象名称”监视元素
LOCK_OBJECT_TYPE	BIGINT	lock_object_type -“等待的锁定对象类型”监视元素
LOCK_RELEASE_FLAGS	BIGINT	lock_release_flags -“锁定释放标志”监视元素
LOCK_WAIT_START_TIME	TIMESTAMP	lock_wait_start_time -“锁定等待开始时间戳记”监视元素
PARTICIPANT_NO	SMALLINT	participant_no -“死锁参与者”监视元素
PARTICIPANT_NO_HOLDING_LK	SMALLINT	participant_no_holding_lk -“挂起对应用程序所需对象的锁定的参与者”监视元素
SEQUENCE_NO	CHARACTER(5)	sequence_no -“序号”监视元素
SEQUENCE_NO_HOLDING_LK	CHARACTER(5)	sequence_no_holding_lk -“挂起锁定的序号”监视元素
START_TIME	TIMESTAMP	start_time -“事件开始时间”监视元素
TABLE_NAME	VARCHAR(128)	table_name -“表名”监视元素
TABLE_SCHEMA	VARCHAR(128)	table_schema -“表模式名”监视元素
TABLESPACE_NAME	VARCHAR(18)	tablespace_name -“表空间名称”监视元素

表 93. 对死锁事件监视器返回的信息: 缺省表名: CONTROL_evmon-name

列名	数据类型	描述
AGENT_ID	BIGINT	agent_id -“应用程序句柄（代理程序标识）”监视元素
APPL_ID	VARCHAR(64)	appl_id -“应用程序标识”监视元素
APPL_ID_HOLDING_LK	VARCHAR(64)	appl_id_holding_lk -“挂起锁定的应用程序标识”监视元素
DATA_PARTITION_ID	INTEGER	data_partition_id -“数据分区标识”监视元素
DEADLOCK_ID	BIGINT	deadlock_id -“死锁事件标识”监视元素
EVMON_ACTIVATES	BIGINT	evmon_activates -“事件监视器激活数”监视元素
LOCK_ATTRIBUTES	BIGINT	lock_attributes -“锁定属性”监视元素
LOCK_COUNT	BIGINT	lock_count -“锁定计数”监视元素
LOCK_CURRENT_MODE	BIGINT	lock_current_mode -“转换前的原始锁定方式”监视元素

表 93. 对死锁事件监视器返回的信息：缺省表名：CONTROL_evmon-name（续）

列名	数据类型	描述
LOCK_ESCALATION	SMALLINT	lock_escalation -“锁定升级”监视元素
LOCK_HOLD_COUNT	BIGINT	lock_hold_count -“锁定挂起计数”监视元素
LOCK_MODE	BIGINT	lock_mode -“锁定方式”监视元素
LOCK_MODE_REQUESTED	BIGINT	lock_mode_requested -“请求的锁定方式”监视元素
LOCK_NAME	CHARACTER(13)	lock_name -“锁定名称”监视元素
LOCK_NODE	BIGINT	lock_node -“锁定节点”监视元素
LOCK_OBJECT_NAME	BIGINT	lock_object_name -“锁定对象名称”监视元素
LOCK_OBJECT_TYPE	BIGINT	lock_object_type -“等待的锁定对象类型”监视元素
LOCK_RELEASE_FLAGS	BIGINT	lock_release_flags -“锁定释放标志”监视元素
LOCK_WAIT_START_TIME	TIMESTAMP	lock_wait_start_time -“锁定等待开始时间戳记”监视元素
PARTICIPANT_NO	SMALLINT	participant_no -“死锁参与者”监视元素
PARTICIPANT_NO_HOLDING_LK	SMALLINT	participant_no_holding_lk -“挂起对应用程序所需对象的锁定的参与者”监视元素
SEQUENCE_NO	CHARACTER(5)	sequence_no -“序号”监视元素
SEQUENCE_NO_HOLDING_LK	CHARACTER(5)	sequence_no_holding_lk -“挂起锁定的序号”监视元素
START_TIME	TIMESTAMP	start_time -“事件开始时间”监视元素
TABLE_NAME	VARCHAR(128)	table_name -“表名”监视元素
TABLE_SCHEMA	VARCHAR(128)	table_schema -“表模式名”监视元素
TABLESPACE_NAME	VARCHAR(18)	tablespace_name -“表空间名称”监视元素

变更历史记录事件监视

变更历史记录事件监视器捕获数据库服务器上可能影响常规数据库工作负载运行的事件的信息。可使用此事件监视器捕获的数据来了解数据库和数据库管理系统的行为、性能或稳定性的更改。

常规工作负载遇到性能降低或您发现意外行为时，可使工作负载行为更改与变更历史记录事件监视器捕获的事件相关联。以下更改可能对数据库系统有负面影响。

- 意外创建或删除索引
- 安排维护运行失败
- 更改数据库配置参数或 DB2 注册表变量

更改可能由用户显式导致。例如，管理员可能运行删除索引的 DDL 语句。或者，更改可能在没有任何用户交互的情况下隐式发生或自动发生。例如，自调整内存管理器 (STMM) 可能会更改配置参数，或自动表重组可能会重组表。

手动跟踪数据库服务器更改可能是很难完成的任务。过去不同类型的更改的信息是通过不同接口捕获的。例如，配置更新会写至诊断日志文件（例如，db2diag 日志文件），实用程序进度在数据库历史记录文件中捕获。变更历史记录事件监视器为您提供单个界面来捕获更改数据库系统的行为和性能特征的事件。通过使用事件监视器表，可查看您关心的任何更改事件。

变更历史记录事件监视器可捕获许多操作的与更改相关的事件，包括:

- 数据库和数据库管理器配置参数更改

- 注册表变量更改
- DDL 语句的执行
- 变更历史记录事件监视器启动
- 执行以下 DB2 实用程序和命令:
  - LOAD
  - ADMIN_MOVE_TABLE 过程调用
  - BACKUP DATABASE（仅 ONLINE 选项）
  - RESTORE DATABASE（仅 ONLINE 选项）
  - ROLLFORWARD DATABASE（仅 ONLINE 选项）
  - REDISTRIBUTE DATABASE PARTITION GROUP
  - REORG
  - RUNSTATS

通常，不会捕获在变更历史记录事件监视器处于不活动状态或数据库脱机时发生的事件的相关信息。但是，可配置变更历史记录事件监视器以在事件监视器激活时捕获生效的注册表变量值。同样，可在变更历史记录事件监视器激活时捕获数据库配置参数和数据库管理器配置参数。捕获配置参数值时，事件监视器可检测它处于不活动状态时是否有任何配置参数发生更改，以便事件监视器仅捕获发生了更改的配置参数值。

变更历史记录事件监视器生成的数据

变更历史记录事件监视器捕获有关可能影响数据库和数据库管理系统的性能、行为和稳定性的活动的信息。变更历史记录的输出会写至一些逻辑数据组，其中每个逻辑数据组都有关联事件监视器表。

与更改相关的操作可在变更历史记录事件监视器中生成一个或多个事件。例如，数据库配置更新生成单个事件，而执行 REORG 实用程序会生成用于标记 REORG 操作的开始和结束的两个事件。事件与逻辑数据组之间存在一对多映射。一个事件可将信息写至多个逻辑数据组，并且可将多个条目（行）写至与给定逻辑数据组相关联的表。每个与更改相关的事件通常由以下 3 个关键字段唯一标识:

事件时间戳记
事件发生的时间。

事件标识
用于确保事件时间戳记相同时的唯一性的数字标记。

成员 发生该事件的数据库管理器进程。

所有逻辑组都包含这 3 个字段，对于这些字段，与同一事件相对应的所有记录或行包含相同的值。这些相同值会使不同逻辑数据组之间的信息连接顺利进行。针对不同成员的实用程序操作和配置参数更新会作为不同事件捕获，并且会导致这些关键字段的值不同。

变更历史记录事件监视器仅支持事件监视器逻辑数据组的 TABLE 目标。变更历史记录事件监视器不支持 UNFORMATTED EVENT TABLE、FILE 和 PIPE 目标。

下表包含变更历史记录事件监视器使用的逻辑数据组及关联表的列表。每个逻辑数据组的缺省表名通过将用于填充表的逻辑数据组的名称与使用 CREATE EVENT MONI-

TOR 语句创建事件监视器时给予该事件监视器的名称并置派生。所显示表名是未在 CREATE EVENT MONITOR 语句中指定名称时的缺省表名。

表 94. 变更历史记录事件监视器的逻辑数据组

逻辑数据组	缺省表名	包含
CHANGESUMMARY	CHANGESUMMARY_evmon-name (see 第 342 页的『CHANGESUMMARY 逻辑数据组』)	变更历史记录事件监视器捕获的所有事件的摘要
DBDBMCFG	DBDBMCFG_evmon-name (see 第 345 页的『DBDBMCFG 逻辑数据组』)	配置参数更改
REGVAR	REGVAR_evmon-name (see 第 347 页的『REGVAR 逻辑数据组』)	注册表变量更改
DDLSTMTEXEC	DDLSTMTEXEC_evmon-name (see 第 348 页的『DDLSTMTEXEC 逻辑数据组』)	DDL 执行
TXNCOMPLETION	TXNCOMPLETION_evmon-name (see 第 351 页的『TXNCOMPLETION 逻辑数据组』)	落实、回滚或回滚至保存点的实例
EVMONSTART	EVMONSTART_evmon-name (see 第 352 页的『EVMONSTART 逻辑数据组』)	事件监视器启动信息
UTILSTART	UTILSTART_evmon-name (see 第 352 页的『UTILSTART 逻辑数据组』)	实用程序启动信息
UTILLOCATION	UTILLOCATION_evmon-name (see 第 356 页的『UTILLOCATION 逻辑数据组』)	实用程序路径或文件信息
UTILSTOP	UTILSTOP_evmon-name (see 第 358 页的『UTILSTOP 逻辑数据组』)	实用程序停止信息
UTILPHASE	UTILPHASE_evmon-name (see 第 360 页的『UTILPHASE 逻辑数据组』)	实用程序阶段信息

变更历史记录事件监视器可捕获许多事件。并非所有用户会关心所有事件。当您创建事件监视器时，可以控制通过使用 CREATE EVENT MONITOR 语句中的 WHERE EVENT IN 子句更改历史记录事件监视器来捕获哪些事件类型。

下表显示数据库服务器上生成可供变更历史记录事件监视器捕获的事件的操作。它还指示 WHERE EVENT IN 子句中指定的哪些控制选项导致捕获这些事件并在生成事件时填充哪些逻辑数据组。

表 95. 操作生成的事件

操作	事件类型	WHERE EVENT IN 子句	逻辑数据组	详细信息
更改数据库配置参数	DBCFG	ALL CFGALL DBCFG	CHANGESUMMARY DBDBMCFG	向 CHANGESUMMARY 写入 1 个记录 对于每个已更改参数，向 DBDBMCFG 写入 1 个记录

表 95. 操作生成的事件 (续)

操作	事件类型	WHERE EVENT IN 子句	逻辑数据组	详细信息
在事件监视器启动时捕获所有数据库配置参数值（如果事件监视器处于不活动状态时数据库配置参数更改）	DBCFGVALUES	ALL CFGALL DBCFGVALUES	CHANGESUMMARY DBDBMCFG	向 CHANGESUMMARY 写入 1 个记录 对于每个参数，向 DBDBMCFG 写入 1 个记录
更改数据库管理器配置参数	DBMCFG	ALL CFGALL DBMCFG	CHANGESUMMARY DBDBMCFG	向 CHANGESUMMARY 写入 1 个记录 对于每个已更改参数，向 DBDBMCFG 写入 1 个记录
在事件监视器启动时捕获所有数据库管理器配置参数值（如果事件监视器处于不活动状态时数据库管理器配置参数更改）	DBMCFGVALUES	ALL CFGALL DBMCFGVALUES	CHANGESUMMARY DBDBMCFG	向 CHANGESUMMARY 写入 1 个记录 对于每个参数，向 DBDBMCFG 写入 1 个记录
更改注册表变量。只有立即更新（使用 **db2set** 命令上 -immediate 标志的注册表变量更改）生成事件。	REGVAR	ALL CFGALL REGVAR	CHANGESUMMARY REGVAR	向 CHANGESUMMARY 写入 1 个记录 对于每个已更改变量，向 REGVAR 写入 1 个记录。仅捕获显式设置的注册表变量。不会写入任何记录，因为变量是通过聚集注册表变量进行隐式设置。
在事件监视器启动时捕获注册表变量值	REGVARVALUES	ALL CFGALL REGVARVALUES	CHANGESUMMARY REGVAR	向 CHANGESUMMARY 写入 1 个记录 对于每个显式设置的变量，向 REGVAR 写入 1 个记录
成功执行 DDL 语句	DDLSTMTEXEC	ALL DDLALL DDLDATA DDLFEDERATED DDLMONITOR DDLSECURITY DDLSQL DDLSTORAGE DDLWLM DDLXML	CHANGESUMMARY DDLSTMTEXEC	向 CHANGESUMMARY 写入 1 个记录 向 DDLSTMTEXEC 写入 1 个记录

表 95. 操作生成的事件 (续)

操作	事件类型	WHERE EVENT IN 子句	逻辑数据组	详细信息
落实或回滚包含成功执行 DDL 语句的事务，或回滚到包含成功执行 DDL 语句的保存点	TXNCOMPLETION	ALL DDLALL DDLDATA DDLFEDERATED DDLMONITOR DDLSECURITY DDLSQL DDLSTORAGE DDLWLM DDLXML	CHANGESUMMARY TXNCOMPLETION	向 CHANGESUMMARY 写入 1 个记录 向 TXNCOMPLETION 写入 1 个记录
启动或停止事件监视器	EVMONSTART	不适用	CHANGESUMMARY EVMONSTART	向 CHANGESUMMARY 写入 1 个记录 向 EVMONSTART 写入 1 个记录
暂停后启动实用程序执行或恢复执行。此事件仅在协调程序成员上生成	UTILSTART	ALL BACKUP LOAD MOVETABLE REDISTRIBUTE REORG RESTORE ROLLFORWARD RUNSTATS UTILALL	CHANGESUMMARY UTILSTART UTILLOCATION	向 CHANGESUMMARY 写入 1 个记录 向 UTILSTART 写入 1 个记录 向 UTILLOCATION 写入 0 个或更多记录；对于与实用程序启动相关联的每个文件，写入 1 个记录。
完成实用程序执行或暂停执行。此事件仅在协调程序成员上生成	UTILSTOP	ALL BACKUP LOAD MOVETABLE REDISTRIBUTE REORG RESTORE ROLLFORWARD RUNSTATS UTILALL	CHANGESUMMARY UTILSTOP	向 CHANGESUMMARY 写入 1 个记录 向 UTILSTOP 写入 1 个记录
实用程序的处理在某个成员上启动。此事件仅在多成员环境中生成	UTILSTARTPROC	ALL BACKUP LOAD MOVETABLE REDISTRIBUTE REORG RESTORE ROLLFORWARD RUNSTATS UTILALL	CHANGESUMMARY UTILSTART	向 CHANGESUMMARY 写入 1 个记录 向 UTILSTART 写入 1 个记录

表 95. 操作生成的事件 (续)

操作	事件类型	WHERE EVENT IN 子句	逻辑数据组	详细信息
实用程序的处理在某个成员上停止。此事件仅在多成员环境中生成	UTILSTOPPROC	ALL BACKUP LOAD MOVETABLE REDISTRIBUTE REORG RESTORE ROLLFORWARD RUNSTATS UTILALL	CHANGESUMMARY UTILSTOP	向 CHANGESUMMARY 写入 1 个记录 向 UTILSTOP 写入 1 个记录
开始对成员执行实用程序的特定处理阶段	UTILPHASESTART	ALL BACKUP UTILALL	CHANGESUMMARY UTILPHASE	向 CHANGESUMMARY 写入 1 个记录 向 UTILPHASE 写入 1 个记录
停止对成员执行实用程序的特定处理阶段	UTILPHASESTOP	ALL BACKUP UTILALL	CHANGESUMMARY UTILPHASE	向 CHANGESUMMARY 写入 1 个记录 向 UTILPHASE 写入 1 个记录

CHANGESUMMARY 逻辑数据组:

CHANGESUMMARY 逻辑数据组的表由变更历史记录事件监视器生成，其中每行表示一个已发生的唯一变更历史记录事件。此表提供了一种快速方法，用于确定变更历史记录事件监视器是否捕获了所有更改以及哪些其他逻辑数据组包含这些更改的详细信息。

下表提供变更历史记录事件监视器收集的更改事件信息的摘要。表名通过将用于填充表的逻辑数据组的名称与 CREATE EVENT MONITOR 语句中给予该事件监视器的名称并置派生。

表 96. CHANGESUMMARY 逻辑数据组的变更历史记录事件监视器返回的信息。缺省表名为 CHANGESUMMARY_evmon-name。

列名	数据类型	描述
EVENT_ID	BIGINT	与事件相关联的唯一标记。
EVENT_TIMESTAMP	TIMESTAMP	生成事件的时间。
MEMBER	SMALLINT	发生此事件的成员。

表 96. CHANGESUMMARY 逻辑数据组的变更历史记录事件监视器返回的信息。缺省表名为 CHANGESUMMARY_evmon-name。 (续)

列名	数据类型	描述
EVENT_TYPE	VARCHAR(32)	所发生事件的类型。对于此逻辑数据组，类型为下列其中之一: • DBCFG • DBCFGVALUES • DBMCFG • DBMCFGVALUES • REGVAR • REGVARVALUES • DDLSTMTEXEC • TXNCOMPLETION • EVMONSTART • UTILSTART • UTILSTOP • UTILSTARTPROC • UTILSTOPPROC • UTILPHASESTART • UTILPHASESTOP
COORD_MEMBER	SMALLINT	给定工作单元或工作负载的协调成员。
UTILITY_INVOCATION_ID	VARCHAR(32) FOR BIT DATA	对应针对 EVENT_TYPE 捕获的实用程序调用的唯一标识: • UTILSTART • UTILSTOP • UTILSTARTPROC • UTILSTOPPROC • UTILPHASESTART • UTILPHASESTOP 其他 EVENT_TYPE 会导致空字符串。

表 96. CHANGESUMMARY 逻辑数据组的变更历史记录事件监视器返回的信息。缺省表名为 CHANGESUMMARY_evmon-name。 (续)

列名	数据类型	描述
UTILITY_TYPE	VARCHAR(16)	如 果 UTILITY_INVOCATION_ID 不为空，那么类型为下列其中一个: • BACKUP • LOAD • MOVETABLE • REDISTRIBUTE • REORG • RESTORE • ROLLFORWARD • RUNSTATS 否则为空字符串。
APPL_ID	VARCHAR(64)	应用程序连接至数据库时生成的标识。
APPL_NAME	VARCHAR(255)	客户机上运行的应用程序在数据库中的名称。
APPLICATION_HANDLE	BIGINT	应用程序的系统范围唯一标识
SYSTEM_AUTHID	VARCHAR(128)	连接的系统授权标识。这是 system_auth_id 监视元素的同义词。
SESSION_AUTHID	VARCHAR(128)	应用程序使用的会话的当前授权标识。这是 session_auth_id 监视元素的同义词。
CLIENT_PLATFORM	VARCHAR(12)	运行客户机应用程序的操作系统。
CLIENT_PROTOCOL	VARCHAR(10)	客户机应用程序用于与服务器通信的通信协议。
CLIENT_PORT_NUMBER	INTEGER	应用程序用来与数据库服务器通信的客户机端口号。
CLIENT_PID	BIGINT	建立与数据库的连接的客户机应用程序的进程标识。
CLIENT_HOSTNAME	VARCHAR(255)	与客户机应用程序相连的机器的主机名。
CLIENT_WRKSTNNAME	VARCHAR(255)	用于标识客户机系统或工作站的名称。
CLIENT_ACCTNG	VARCHAR(200)	传递至目标数据库以用于日志记录和诊断的数据。
CLIENT_USERID	VARCHAR(255)	提供给服务器的客户机用户标识。
CLIENT_APPLNAME	VARCHAR(255)	用于标识正执行事务的服务器事务程序的名称。

表 96. CHANGESUMMARY 逻辑数据组的变更历史记录事件监视器返回的信息。缺省表名为 CHANGESUMMARY_evmon-name。 (续)

列名	数据类型	描述
BACKUP_TIMESTAMP	VARCHAR(14)	如果 UTILITY_TYPE 为 BACKUP 且 EVENT_TYPE 为 UTILSTART，那么 BACKUP_TIMESTAMP 值是备份映像的时间戳记。 如果 UTILITY_TYPE 为 RESTORE 且 EVENT_TYPE 为 UTILSTOP，那么 BACKUP_TIMESTAMP 值是备份映像的时间戳记。 对于所有其他情况，BACKUP_TIMESTAMP 为空字符串。 BACKUP_TIMESTAMP 可通过使用 SYSIBMADM.DB_HISTORY 管理视图来与存储在数据库历史记录文件中的信息（例如，查询顺序信息）关联到一起。

DBDBMCFG 逻辑数据组:

DBDBMCFG 逻辑数据组的表由变更历史记录事件监视器生成，其中每行表示一个配置参数，此配置参数作为 DBCFG 或 DBMCFG 事件的一部分更新或作为 DBCFGVALUES 或 DBMCFGVALUES 事件的一部分在事件监视器启动时捕获。CFG_COLLECTION_TYPE 监视元素标识该记录是描述配置参数更新还是在事件监视器启动时记录的初始值。

下表显示变更历史记录事件监视器收集的配置参数更改。表名通过将用于填充表的逻辑数据组的名称与 CREATE EVENT MONITOR 语句中给予该事件监视器的名称并置派生。

表 97. DBDBMCFG 逻辑数据组的变更历史记录事件监视器返回的信息。缺省表名为 DBDBMCFG_evmon-name。

列名	数据类型	描述
EVENT_ID	BIGINT	与事件相关联的唯一标记。
EVENT_TIMESTAMP	TIMESTAMP	生成事件的时间。
MEMBER	SMALLINT	发生此事件的成员。

表 97. DBDBMCFG 逻辑数据组的变更历史记录事件监视器返回的信息。缺省表名为 DBDBMCFG_evmon-name。 (续)

列名	数据类型	描述
EVENT_TYPE	VARCHAR(32)	所发生事件的类型。对于此逻辑数据组，类型为下列其中之一: • DBCFG • DBCFGVALUES • DBMCFG • DBMCFGVALUES
CFG_NAME	VARCHAR(128)	配置参数的名称。
CFG_VALUE	VARCHAR(255)	如果 EVENT_TYPE 为 DBCFG 或 DBMCFG，那么这是配置参数的新值。 如果 EVENT_TYPE 为 DBCFGVALUES 或 DBMCFGVALUES，那么这是磁盘配置参数值。磁盘配置参数值是最新值，并且可能尚未生效。
CFG_VALUE_FLAGS	VARCHAR(32)	此标志指示如何确定新配置参数值: • AUTOMATIC • COMPUTED • NONE 如果 EVENT_TYPE 为 DBCFGVALUES 或 DBMCFGVALUES，那么这些标志表示该配置参数的最新磁盘值。
CFG_OLD_VALUE	VARCHAR(255)	如果 EVENT_TYPE 为 DBCFG 或 DBMCFG，那么这是旧配置参数值。 如果 EVENT_TYPE 为 DBCFGVALUES 或 DBMCFGVALUES，那么这是当前内存配置参数值。这是当前正使用的配置参数值。

表 97. DBDBMCFG 逻辑数据组的变更历史记录事件监视器返回的信息。缺省表名为 DBDBMCFG_evmon-name。 (续)

列名	数据类型	描述
CFG_OLD_VALUE_FLAGS	VARCHAR(32)	此标志指示如何确定旧配置参数值: • AUTOMATIC • COMPUTED • NONE 如果 EVENT_TYPE 为 DBCFGVALUES 或 DBMCFGVALUES，那么这些标志表示该配置参数的最新内存值。
CFG_COLLECTION_TYPE	CHAR(1)	指示收集配置参数值的时间: **I** 激活事件监视器时捕获的初始值。 **U** 已更新值
DEFERRED	CHAR(1)	指示对配置参数值的更改是否延迟: **Y** 更改延迟至下一次数据库激活 **N** 更改立即生效

REGVAR 逻辑数据组:

REGVAR 逻辑数据组的表由变更历史记录事件监视器生成，其中每行表示一个注册表变量，该注册表变量作为 REGVAR 事件的一部分更新或作为 RERVARVALUES 事件的一部分在事件监视器启动时捕获。REGVAR_COLLECTION_TYPE 监视元素标识该记录是描述即时注册表变量更新 (*U*) 还是在事件监视器启动时记录的初始值 (*I*)。

下表显示变更历史记录事件监视器收集的注册表变量更改。表名通过将用于填充表的逻辑数据组的名称与 CREATE EVENT MONITOR 语句中给予该事件监视器的名称并置派生。

表 98. REGVAR 逻辑数据组的变更历史记录事件监视器返回的信息。缺省表名为 REGVAR_evmon-name。

列名	数据类型	描述
EVENT_ID	BIGINT	与事件相关联的唯一标记。
EVENT_TIMESTAMP	TIMESTAMP	生成事件的时间。
MEMBER	SMALLINT	发生此事件的成员。
EVENT_TYPE	VARCHAR(32)	所发生事件的类型。对于此逻辑数据组，类型为下列其中之一: • REGVAR • REGVARVALUES
REGVAR_NAME	VARCHAR(256)	注册表变量的名称。

表 98. REGVAR 逻辑数据组的变更历史记录事件监视器返回的信息。缺省表名为 REGVAR_evmon-name。 (续)

列名	数据类型	描述
REGVAR_VALUE	CLOB(2k)	这是注册表变量的值。如果未设置，那么值为空字符串。
REGVAR_OLD_VALUE	CLOB(2k)	这是注册表变量的旧值。如果未设置值，那么此值为空字符串。
REGVAR_LEVEL	CHAR(1)	指示注册表变量的级别: **E** 环境 **G** 全局 **I** 实例级别 **P** 数据库分区
REGVAR_COLLECTION_TYPE	CHAR(1)	指示收集注册表变量值的时间: **I** 激活事件监视器时捕获的初始值。 **U** 已更新值

DDLSTMTEXEC 逻辑数据组:

DDLSTMTEXEC 逻辑数据组的表由变更历史记录事件监视器生成，其中每行表示一个已执行的 DDL 语句事件。在分区数据库环境中，系统会在协调程序分区上针对 DDL 执行捕获行。

每当向 DDLSTMTEXEC_evmon-name 表写入一行时，系统会对执行 DDL 语句后在同一工作单元中执行的每个关联事务状态更改（落实、回滚或回滚至保存点）向 TXNCOMPLETION_evmon-name 表写入一行。还可使用 DDLSTMTEXEC_evmon-name 表中的 GLOBAL_TRANSACTION_ID、LOCAL_TRANSACTION_ID 和 SAVEPOINT_ID 列在 TXNCOMPLETION_evmon-name 表中查找对应状态更改操作，并确定该 DDL 是否已落实。

下表显示变更历史记录事件监视器收集的 DDL 语句执行信息。表名通过将用于填充表的逻辑数据组的名称与 CREATE EVENT MONITOR 语句中给予该事件监视器的名称并置派生。

表 99. DDLSTMTEXEC 逻辑数据组的变更历史记录事件监视器返回的信息。缺省表名为 DDLSTMTEXEC_evmon-name。

列名	数据类型	描述
EVENT_ID	BIGINT	与事件相关联的唯一标记。
EVENT_TIMESTAMP	TIMESTAMP	生成事件的时间。
MEMBER	SMALLINT	发生此事件的成员。
EVENT_TYPE	VARCHAR(32)	所发生事件的类型。对于此逻辑数据组，类型为 DDLSTMTEXEC。

表 99. DDLSTMTEXEC 逻辑数据组的变更历史记录事件监视器返回的信息。缺省表名为 DDLSTMTEXEC_evmon-name。 (续)

列名	数据类型	描述
GLOBAL_TRANSACTION_ID	VARCHAR(40)	事件发生时正使用的全局事务标识。这是作为事务日志一部分的 SQLP_GXID 结构中的数据字段。
LOCAL_TRANSACTION_ID	VARCHAR(16)	事件发生时正使用的本地事务标识。这是作为事务日志一部分的 SQLU_TID 结构。
SAVEPOINT_ID	BIGINT	在工作单元内设置的保存点的名称。
UOW_ID	INTEGER	应用程序句柄内的工作单元标识的唯一标识。

表 99. DDLSTMTEXEC 逻辑数据组的变更历史记录事件监视器返回的信息。缺省表名为 DDLSTMTEXEC_evmon-name。 (续)

列名	数据类型	描述
DDL_CLASSIFICATION	VARCHAR(30)	已捕获 DDL 的分类: **DDLSTORAGE** 改变数据库 DDL、缓冲池 DDL、分区组 DDL、存储器组 DDL 和表空间 DDL 的执行。 **DDLWLM** 直方图 DDL、服务类 DDL、阈值 DDL、工作操作集 DDL、工作类集 DDL 和工作负载 DDL 的执行。 **DDLMONITOR** 事件监视器 DDL 和用法列表 DDL 的执行。 **DDLSECURITY** 审计策略 DDL、授权 DDL、掩码 DDL、许可权角色 DDL、撤销 DDL、安全标号 DDL、安全标号组件 DDL、安全策略 DDL 和可信上下文 DDL 的执行。 **DDLSQL** 别名 DDL、函数 DDL、方法 DDL、模块 DDL、程序包 DDL、过程 DDL、模式 DDL、同义词 DDL、变换 DDL、触发器 DDL、类型 DDL、变量 DDL 和视图 DDL 的执行。 **DDLDATA** 索引 DDL、序列 DDL、表 DDL 和临时表 DDL 的执行。

表 99. DDLSTMTEXEC 逻辑数据组的变更历史记录事件监视器返回的信息。缺省表名为 DDLSTMTEXEC_evmon-name。 (续)

列名	数据类型	描述
DDL_CLASSIFICATION（继续使用）	VARCHAR(30)	**DDLXML** XSROBJECT DDL 的执行。 **DDLFEDERATED** 昵称/服务器 DDL、类型/用户映射 DDL 和包装器 DDL 的执行。
STMT_TEXT	CLOB(2MB)	SQL 语句的文本。

TXNCOMPLETION 逻辑数据组:

TXNCOMPLETION 逻辑数据组的表由变更历史记录事件监视器生成，其中每行表示一个已完成的事务事件。每当落实、回滚或回滚至保存点与对应 DDLSTMTEXEC_evmon-name 表事件发生在同一工作单元中时，会写入行。

使用 TXNCOMPLETION_evmon-name 表中的信息来确定事务中的 DDL 语句是否已落实。然后可使用 TXNCOMPLETION_evmon-name 表中的 GLOBAL_TRANSACTION_ID、LOCAL_TRANSACTION_ID 和 SAVEPOINT_ID 列在 DDLSTMTEXEC_evmon-name 表中查找受事务完成事件影响的 DDL 语句。

下表显示变更历史记录事件监视器收集的事务完成信息。表名通过将用于填充表的逻辑数据组的名称与 CREATE EVENT MONITOR 语句中给予该事件监视器的名称并置派生。

表 100. TXNCOMPLETION 逻辑数据组的变更历史记录事件监视器返回的信息。缺省表名为 TXNCOMPLETION_evmon-name。

列名	数据类型	描述
EVENT_ID	BIGINT	与事件相关联的唯一标记。
EVENT_TIMESTAMP	TIMESTAMP	生成事件的时间。
MEMBER	SMALLINT	发生此事件的成员。
EVENT_TYPE	VARCHAR(32)	所发生事件的类型。对于此逻辑数据组，类型为 TXNCOMPLETION。
GLOBAL_TRANSACTION_ID	VARCHAR(40)	事件发生时正使用的全局事务标识。这是作为事务日志一部分的 SQLP_GXID 结构中的数据字段。
LOCAL_TRANSACTION_ID	VARCHAR(16)	事件发生时正使用的本地事务标识。这是作为事务日志一部分的 SQLU_TID 结构。
SAVEPOINT_ID	BIGINT	工作单元内设置的保存点的标识。
UOW_ID	INTEGER	应用程序句柄内的工作单元标识的唯一标识。

表 100. TXNCOMPLETION 逻辑数据组的变更历史记录事件监视器返回的信息。缺省表名为 TXNCOMPLETION_evmon-name。 (续)

列名	数据类型	描述
TXN_COMPLETION_STATUS	CHAR(1)	指示事务的状态: **C** 落实 **R** 回滚 **S** 回滚至保存点

EVMONSTART 逻辑数据组:

EVMONSTART 逻辑数据组的表由变更历史记录事件监视器生成，其中每行表示该变更历史记录事件监视器的一次启动。尽管事件监视器启动与系统性能无直接关系，但此信息为该监视器捕获的其他信息提供上下文。

事件监视器启动事件可帮助您了解更改何时开始生效。例如，激活时间戳记可帮助您跟踪任何延迟数据库或数据库管理器配置参数更新何时生效。知道事件监视器何时激活还可帮助您了解事件监视器表中捕获的信息的完整性。事件监视器显式或隐式取消激活时发生的任何事件不会被捕获。如果数据库未激活，那么该事件监视器处于隐式不活动状态。

下表显示变更历史记录事件监视器收集的事件监视器启动信息。表名通过将用于填充表的逻辑数据组的名称与 CREATE EVENT MONITOR 语句中给予该事件监视器的名称并置派生。

表 101. EVMONSTART 逻辑数据组的变更历史记录事件监视器返回的信息。缺省表名为 EVMONSTART_evmon-name。

列名	数据类型	描述
EVENT_ID	BIGINT	与事件相关联的唯一标记。
EVENT_TIMESTAMP	TIMESTAMP	生成事件的时间。
MEMBER	SMALLINT	发生此事件的成员。
EVENT_TYPE	VARCHAR(32)	所发生事件的类型。对于此逻辑数据组，类型为 EVMONSTART。
DB2START_TIME	TIMESTAMP	数据库成员激活时间戳记，可用于跟踪延迟的数据库管理器配置参数更新何时生效。
DB_CONN_TIME	TIMESTAMP	数据库激活时间戳记，可用于跟踪延迟数据库配置参数更新何时生效。

UTILSTART 逻辑数据组:

UTILSTART 逻辑数据组的表由变更历史记录事件监视器生成，其中每行表示一个已启动的实用程序。

下表显示变更历史记录事件监视器收集的实用程序详细信息。表名通过将用于填充表的逻辑数据组的名称与 CREATE EVENT MONITOR 语句中给予该事件监视器的名称并置派生。

表 102. UTILSTART 逻辑数据组的变更历史记录事件监视器返回的信息。缺省表名为 UTILSTART_evmon-name。

列名	数据类型	描述
EVENT_ID	BIGINT	与事件相关联的唯一标记。
EVENT_TIMESTAMP	TIMESTAMP	生成事件的时间。
MEMBER	SMALLINT	发生此事件的成员。
EVENT_TYPE	VARCHAR(32)	所发生事件的类型。对于此逻辑数据组，类型为下列其中之一: • UTILSTART • UTILSTARTPROC
UTILITY_INVOCATION_ID	VARCHAR(32) FOR BIT DATA	对应于实用程序调用的唯一标识。
UTILITY_TYPE	VARCHAR(16)	实用程序类型是下列其中一项: • BACKUP • LOAD • MOVETABLE • REDISTRIBUTE • REORG • RESTORE • ROLLFORWARD • RUNSTATS

表 102. UTILSTART 逻辑数据组的变更历史记录事件监视器返回的信息。缺省表名为 UTILSTART_evmon-name。 (续)

列名	数据类型	描述
UTILITY_OPERATION_TYPE	CHAR(1)	如果 UTILITY_TYPE 为 BACKUP，那么为下列其中一个: **D** 变化量 **I** 增量 **F** 完全 如果 UTILITY_TYPE 为 LOAD，那么为下列其中一个: **I** 插入 **R** 替换 **S** 重新启动 **T** 终止 如果 UTILITY_TYPE 为 MOVETABLE，那么为下列其中一个: **A** 取消 **C** 复制 **I** 初始化 **L** 清除 **M** 移动 **R** 重演 **S** 交换 **V** 验证 如果 UTILITY_TYPE 为 REDISTRIBUTE，那么为下列其中一个: **A** 中止 **C** 继续 **D** 缺省值 **T** 目标映射 如果 UTILITY_TYPE 为 REORG，那么为下列其中一个: **A** 重组所有表索引 **I** 索引重组 **N** 原地表重组 **R** 重组表回收扩展数据块 **T** 典型表重组

表 102. UTILSTART 逻辑数据组的变更历史记录事件监视器返回的信息。缺省表名为 UTILSTART_evmon-name。 (续)

列名	数据类型	描述
UTILITY_OPERATION_TYPE（继续使用）	CHAR(1)	如果 UTILITY_TYPE 为 RESTORE，那么为下列其中一个: **A** 增量自动 **B** 增量中止 **F** 完全 **M** 增量手动 如果 UTILITY_TYPE 为 ROLLFORWARD，那么为下列其中一个: **E** 日志结束 **P** 时间点 如果 UTILITY_TYPE 为 RUNSTATS，那么为下列其中一个: **A** 表的所有索引 **I** 索引 **T** 表
UTILITY_INVOKER_TYPE	VARCHAR(4)	指示实用程序的调用方式。下列其中一个: • AUTO • USER
UTILITY_PRIORITY	INTEGER	指定调速实用程序相对其调速层而言的相对重要性。优先级值介于 0 到 100 之间，其中 0 指示实用程序以非调速方式执行。
UTILITY_START_TYPE	VARCHAR(8)	指示实用程序的启动方式。下列其中一个: • RESUME • START
OBJECT_TYPE	VARCHAR(16)	该实用程序作用于的对象类型。下列其中一个: • DATABASE • INDEX • PARTITIONGROUP • TABLE • TABLESPACE 这是 objtype 监视元素的同义词。

表 102. UTILSTART 逻辑数据组的变更历史记录事件监视器返回的信息。缺省表名为 UTILSTART_evmon-name。 (续)

列名	数据类型	描述
OBJECT_SCHEMA	VARCHAR(128)	如果 OBJECT_TYPE 为 INDEX 或 TABLE，那么为索引或表的模式，否则为空字符串。
OBJECT_NAME	VARCHAR(128)	如果 OBJECT_TYPE 为 INDEX、PARTIONGROUP 或 TABLE，那么这是索引、分区组或表的名称。
NUM_TBSPS	INTEGER	如果 OBJECT_TYPE 是 DATABASE 或 TABLESPACE，那么为表空间数。
TBSP_NAMES	CLOB(5M)	如果 OBJECT_TYPE 为 DATABASE 或 TABLESPACE，那么为实用程序作用于的表空间名称的逗号分隔列表。
UTILITY_DETAIL	CLOB(2M)	实用程序正执行的工作的简短描述，包括对该实用程序指定的一些选项。例如，针对 REORG 调用的记录将包括部分重构的命令字符串，包括该实用程序使用的某些不同选项，例如，访问方式。此字段的格式取决于实用程序的类型，并且在不同发行版中可能变化。

UTILLOCATION 逻辑数据组:

UTILLOCATION 逻辑数据组的表由变更历史记录事件监视器生成，其中每行表示与实用程序的启动相关联的每个文件或路径。

下表显示变更历史记录事件监视器收集的实用程序详细信息。表名通过将用于填充表的逻辑数据组的名称与 CREATE EVENT MONITOR 语句中给予该事件监视器的名称并置派生。

表 103. UTILLOCATION 逻辑数据组的变更历史记录事件监视器返回的信息。缺省表名为 UTILLOCATION_evmon-name。

列名	数据类型	描述
EVENT_ID	BIGINT	与事件相关联的唯一标记。
EVENT_TIMESTAMP	TIMESTAMP	生成事件的时间。
MEMBER	SMALLINT	发生此事件的成员。
EVENT_TYPE	VARCHAR(32)	所发生事件的类型。对于此逻辑数据组，类型为 UTILSTART。

表 103. UTILLOCATION 逻辑数据组的变更历史记录事件监视器返回的信息。缺省表名为 UTILLOCATION_evmon-name。 (续)

列名	数据类型	描述
UTILITY_INVOCATION_ID	VARCHAR(32) FOR BIT DATA	对应于实用程序调用的唯一标识。
UTILITY_TYPE	VARCHAR(16)	实用程序类型是下列其中一项: • BACKUP • LOAD • RESTORE • ROLLFORWARD
DEVICE_TYPE	CHAR(1)	与 UTILSTART 事件相关联的设备类型的标识。此字段确定 LOCATION 字段的解释。下列其中一个: **A** TSM **C** 客户机 **D** 磁盘 **F** 快照备份 **L** 本地 **N** 由 DB2 内部生成 **O** 其他供应商设备支持 **P** 管道 **Q** 游标 **R** 除去访存数据 **S** 服务器 **T** 磁带 **U** 用户出口 **X** X/开放式 XBSA 接口

表 103. UTILLOCATION 逻辑数据组的变更历史记录事件监视器返回的信息。缺省表名为 UTILLOCATION_evmon-name。 (续)

列名	数据类型	描述
LOCATION_TYPE	CHAR(1)	此位置的用途描述。 如果 UTILITY_TYPE 为 LOAD，那么为下列其中一个: **C** 复制目标 **D** 输入数据 **L** LOB 路径 **X** XML 路径 如果 UTILITY_TYPE 为 BACKUP，那么为下列其中一个: **B** 备份目标位置 如果 UTILITY_TYPE 为 RESTORE，那么为下列其中一个: **S** 恢复源位置 如果 UTILITY_TYPE 为 ROLLFORWARD，那么为下列其中一个: **O** 作为 ROLLFORWARD DATABASE 命令的一部分捕获的备用溢出日志路径。请注意，如果使用缺省溢出日志路径，那么不会捕获任何位置记录。 否则为空白字符。
LOCATION	VARCHAR(1024)	与事件相关联的位置。位置取决于 UTILITY_TYPE。例如，装入输入文件或备份目标路径名。

UTILSTOP 逻辑数据组:

UTILSTOP 逻辑数据组的表由变更历史记录事件监视器生成，其中每行表示一个已停止的实用程序。

下表显示变更历史记录事件监视器收集的实用程序详细信息。表名通过将用于填充表的逻辑数据组的名称与 CREATE EVENT MONITOR 语句中给予该事件监视器的名称并置派生。

表 104. UTILSTOP 逻辑数据组的变更历史记录事件监视器返回的信息。缺省表名为 UTILSTOP_evmon-name。

列名	数据类型	描述
EVENT_ID	BIGINT	与事件相关联的唯一标记。
EVENT_TIMESTAMP	TIMESTAMP	生成事件的时间。
MEMBER	SMALLINT	发生此事件的成员。
EVENT_TYPE	VARCHAR(32)	所发生事件的类型。对于此逻辑数据组，类型为下列其中之一: • UTILSTOP • UTILSTOPPROC
UTILITY_INVOCATION_ID	VARCHAR(32) FOR BIT DATA	对应于实用程序调用的唯一标识。
UTILITY_TYPE	VARCHAR(16)	实用程序类型是下列其中一项: • BACKUP • LOAD • MOVETABLE • REDISTRIBUTE • REORG • RESTORE • ROLLFORWARD • RUNSTATS
UTIL_STOP_TYPE	VARCHAR(8)	指示实用程序的停止方式。下列其中一个: • PAUSE • STOP
START_EVENT_ID	BIGINT	对应 UTILSTART 或 UTILSTARTPROC 事件的唯一标识。与 START_EVENT_TIMESTAMP 和成员元素配合使用以将停止记录与对应开始记录相关联。
START_EVENT_TIMESTAMP	TIMESTAMP	对应 UTILSTART 或 UTILSTARTPROC 事件的时间。与 START_EVENT_ID 和成员元素配合使用以将停止记录与对应开始记录关联到一起。
SQLCA（SQL 通信区）（请参阅 *SQL Reference Volume 1*）	VARCHAR(8)	一个字符串，用于标识 SQL 通信区 (SQLCA) 的开始。
SQLABC	INTEGER	SQL 通信区 (SQLCA) 的长度。
SQLCODE	INTEGER	在 SQLCA 结构中，最新执行的 SQL 语句的 SQL 返回码。
SQLERRMC	VARCHAR(72)	在 SQLCA 结构中，用 X'FF' 分隔的一个或多个标记，它们将被错误消息（提供有关错误情况的具体信息）中的变量替代。

表 104. UTILSTOP 逻辑数据组的变更历史记录事件监视器返回的信息。缺省表名为 UTILSTOP_evmon-name。 (续)

列名	数据类型	描述
SQLERRP	VARCHAR(8)	在 SQLCA 结构中，由 3 个字母组成的标识，用于指示产品，后跟指示产品版本、发行版和修改级别的 5 个字母数字字符。
SQLERRD1	INTEGER	请参阅 SQLCA（SQL 通信区）。
SQLERRD2	INTEGER	请参阅 SQLCA（SQL 通信区）。
SQLERRD3	INTEGER	请参阅 SQLCA（SQL 通信区）。
SQLERRD4	INTEGER	请参阅 SQLCA（SQL 通信区）。
SQLERRD5	INTEGER	请参阅 SQLCA（SQL 通信区）。
SQLERRD6	INTEGER	请参阅 SQLCA（SQL 通信区）。
SQLWARN	VARCHAR(11)	在 SQLCA 结构中，一组警告指示符，每个指示符包含空白或“W”。
SQLSTATE	VARCHAR(5)	在 SQLCA 结构中，用于指示最新执行的 SQL 语句的结果的返回码。

UTILPHASE 逻辑数据组:

UTILPHASE 逻辑数据组的表由变更历史记录事件监视器生成，其中每行包含有关正在启动或停止的实用程序阶段的信息。

实用程序执行分为若干阶段或处理阶段。目前变更历史记录事件监视器仅捕获表空间备份的开始阶段和停止阶段。

下表显示变更历史记录事件监视器收集的实用程序阶段详细信息。表名通过将用于填充表的逻辑数据组的名称与 CREATE EVENT MONITOR 语句中给予该事件监视器的名称并置派生。

表 105. UTILPHASE 逻辑数据组的变更历史记录事件监视器返回的信息。缺省表名为 UTILPHASE_evmon-name。

列名	数据类型	描述
EVENT_ID	BIGINT	与事件相关联的唯一标记。
EVENT_TIMESTAMP	TIMESTAMP	生成事件的时间。
MEMBER	SMALLINT	发生此事件的成员。
EVENT_TYPE	VARCHAR(32)	所发生事件的类型。对于此逻辑数据组，类型为下列其中之一: • UTILPHASESTART • UTILPHASESTOP
UTILITY_INVOCATION_ID	VARCHAR(32) FOR BIT DATA	对应于实用程序调用的唯一标识。
UTILITY_TYPE	VARCHAR(16)	要启动或停止的实用程序的类型。对于此逻辑数据组，类型为 BACKUP。
UTILITY_PHASE_TYPE	VARCHAR(16)	如果 UTILITY_TYPE 为 BACKUP，那么阶段类型为: **BACKUPTS** 备份表空间

表 105. UTILPHASE 逻辑数据组的变更历史记录事件监视器返回的信息。缺省表名为 UTILPHASE_evmon-name。 (续)

列名	数据类型	描述
PHASE_START_EVENT_ID	BIGINT	如果 EVENT_TYPE 为 UTILPHASESTOP，那么这是对应 UTILPHASESTART 的 EVENT_ID，否则为 -1。与 PHASE_START_EVENT_TIMESTAMP 和成员元素配合使用以将阶段停止记录与对应开始记录关联到一起。
PHASE_START_EVENT_TIMESTAMP	TIMESTAMP	如果 EVENT_TYPE 为 UTILPHASESTOP，那么这是对应 UTILPHASESTART 的时间，否则为空。与 PHASE_START_EVENT_ID 和成员元素配合使用以将阶段停止记录与对应开始记录关联到一起。
OBJECT_TYPE	VARCHAR(16)	该实用程序作用于的对象类型。类型为 TABLESPACE。 这是 objtype 监视元素的同义词。
OBJECT_SCHEMA	VARCHAR(128)	保留以供将来使用
OBJECT_NAME	VARCHAR(128)	如果 OBJECT_TYPE 为 TABLESPACE，那么为表名。
UTILITY_PHASE_DETAIL	CLOB(2M)	保留以供将来使用

使用变更历史记录事件监视器来监视实用程序历史记录

变更历史记录事件监视器可捕获与实用程序执行相关的一些事件。可使用这些事件来监视数据库服务器上的实用程序执行的历史记录。此事件历史记录会写至逻辑数据组，其中每个逻辑数据组都有关联事件监视器表。

执行实用程序可在变更历史记录事件监视器中生成一个或多个事件。例如，执行 REORG 实用程序会生成两个事件，分别标记 REORG 操作的开始和结束。事件与逻辑数据组之间存在一对多映射。一个事件可将信息写至多个逻辑数据组，并且可将多个条目（行）写至与给定逻辑数据组相关联的表。对应实用程序的特定调用的每个事件由 *utility_invocation_id* 元素标识。*utility_invocation_id* 实用程序是二进制标记，用于唯一标识实用程序的给定调用。*utility_invocation_id* 在执行该实用程序的每个成员上相同。*utility_invocation_id* 在数据库取消激活、重新激活和成员关闭期间保持其唯一性，以允许快速标识对应实用程序的给定调用的所有事件监视器记录。不需要连接其他字段或担心重复标识。

通过使用 *utility_invocation_id*，可标识用于描述实用程序的特定调用的所有事件。例如，如果对表发出 REORG 命令，那么实用程序开始执行时生成 UTILSTART 事件，在实用程序完成执行时生成 UTILSTOP 事件。UTILSTART 和 UTILSTOP 事件将具有相同 *utility_invocation_id*，因为它们描述 REORG 命令的同一调用。*utility_invocation_id* 可用来连接这些事件以计算实用程序的耗用时间。

变更历史记录事件监视器可监视以下实用程序类型的执行:

- BACKUP
- LOAD

- MOVETABLE
- REDISTRIBUTE
- REORG
- RESTORE
- ROLLFORWARD
- RUNSTATS

变更历史记录事件监视器不会捕获脱机备份、复原或前滚的执行。请注意，仅当变更历史记录事件监视器在实用程序执行期间处于活动状态，才会捕获实用程序事件。如果在实用程序执行之前取消激活事件监视器，那么不会针对该实用程序的执行捕获任何事件。例如，如果实用程序需要对事件监视器目标表所在的表空间执行互斥存取。

下表列示与实用程序执行事件相关联的变更历史记录事件监视器逻辑数据组及关联表。表名通过将用于填充表的逻辑数据组的名称与使用 CREATE EVENT MONITOR 语句创建事件监视器时给予该事件监视器的名称并置派生。所显示表名是未在 CREATE EVENT MONITOR 语句中指定名称时的缺省表名。

表 106. 实用程序执行期间填充的逻辑数据组

逻辑数据组	缺省表名	包含
CHANGESUMMARY	CHANGESUMMARY_evmon-name (see 第 342 页的『CHANGESUMMARY 逻辑数据组』)	变更历史记录事件监视器捕获的所有事件的摘要
UTILSTART	UTILSTART_evmon-name (see 第 352 页的『UTILSTART 逻辑数据组』)	实用程序启动信息
UTILLOCATION	UTILLOCATION_evmon-name (see 第 356 页的『UTILLOCATION 逻辑数据组』)	实用程序路径或文件信息
UTILSTOP	UTILSTOP_evmon-name (see 第 358 页的『UTILSTOP 逻辑数据组』)	实用程序停止信息
UTILPHASE	UTILPHASE_evmon-name (see 第 360 页的『UTILPHASE 逻辑数据组』)	实用程序阶段信息

CREATE EVENT MONITOR （变更历史记录）的 WHERE EVENT IN 子句控制变更历史记录事件监视器监视的实用程序。以下列表指示哪些控件启用了哪些实用程序的捕获:

UTILALL
捕获装入、移动表、联机备份、联机复原、联机前滚、重新分发、reorg 和 runstats 实用程序的执行。

BACKUP
捕获联机备份实用程序的执行。

LOAD 捕获装入实用程序的执行。

MOVETABLE
捕获表移动实用程序的执行（ADMIN_MOVE_TABLE 存储过程的调用）。

REDISTRIBUTE
捕获重新分发分区组实用程序的执行。

REORG
捕获 reorg 实用程序的执行。

RESTORE
捕获联机复原实用程序的执行。

ROLLFORWARD
捕获联机前滚实用程序的执行。

RUNSTATS
捕获 runstats 实用程序的执行。

收集变更历史记录事件数据

可使用变更历史记录事件监视器来收集有关可能影响数据库和数据库管理系统性能、行为和稳定性的活动的信息。

开始之前

要创建变更历史记录事件监视器并收集变更历史记录事件监视器数据，您必须具有 DBADM 或 SQLADM 权限。

关于此任务

变更历史记录事件监视器捕获可能影响常规数据库工作负载运行的更改。常规工作负载遇到性能降低或您发现意外行为时，必须确定发生了哪些可能导致该问题的更改。每个与更改相关的事件通常由以下 3 个关键字段唯一标识:

事件时间戳记
事件发生的时间。

事件标识
用于确保事件时间戳记相同时的唯一性的数字标记。

成员 发生该事件的数据库管理器进程。成员确保全局唯一性，因为事件时间戳记和事件标识仅对于每个成员是唯一的。

所有逻辑组都包含这 3 个字段，对于这些字段，与同一事件相对应的所有记录或行包含相同的值。这些相同值会使不同逻辑数据组之间的信息连接顺利进行。针对不同成员的实用程序操作和配置参数更新会作为不同事件捕获，并且会导致这些关键字段的值不同。

限制

变更历史记录事件监视器数据只能写至与逻辑数据组相关联的表。这些变更历史记录事件监视器不会写至无格式事件表、文件或命名管道。

过程

要收集有关可能影响数据库性能、行为或稳定性的活动的详细信息，请执行以下步骤:

1. 决定您所关心的变更历史记录事件。变更历史记录事件监视器能够捕获描述下列各项的事件:
   - 配置参数更改
   - 注册表变量更改

- DDL 执行
- 落实、回滚或回滚至保存点的实例
- 事件监视器启动信息
- 实用程序启动信息
- 实用程序路径或文件信息
- 实用程序停止信息
- 实用程序阶段信息

2. 使用 CREATE EVENT MONITOR FOR CHANGE HISTORY 语句来创建名为 whats_changed 的变更历史记录事件监视器。使用 WHERE EVENT IN 子句来指定应捕获的变更历史记录事件。以下事件说明如何创建用于捕获所有事件类型的变更历史记录事件监视器:

```
CREATE EVENT MONITOR whats_changed
   FOR CHANGE HISTORY WHERE EVENT IN (ALL)
   WRITE TO TABLE
```

3. 通过运行以下语句来激活名为 whats_changed 的变更历史记录事件监视器:

```
SET EVENT MONITOR whats_changed STATE 1
```

结果

每当在变更历史记录事件监视器处于活动状态时发生变更历史记录事件（例如，数据库配置更新），都将在变更历史记录事件监视器表中捕获有关该事件的信息。变更历史记录事件监视器只会捕获该事件监视器的 WHERE EVENT IN 子句中标识的事件。

示例

示例: 使用变更历史记录事件监视器来调查锁定升级增加情况:

可使用变更历史记录事件监视器来检测哪些更改可能导致数据库性能下降。

方案

在此示例中，用户将报告数据库性能下降。数据库管理员 (DBA) 注意到过去 24 小时锁定升级数目过高。DBA 还注意到同一时间段内应用程序锁定等待时间相应增加。

DBA 已在使用变更历史记录事件监视器来监视配置更改、索引更改和装入操作。该事件监视器是使用以下语句创建的:

```
CREATE EVENT MONITOR CFGHIST
FOR CHANGE HISTORY WHERE EVENT IN (DBCFG, DBMCFG, DBCFGVALUES,
     DBMCFGVALUES,REGVAR,REGVARVALUES, DDLDATA, LOAD)
WRITE TO TABLE
```

该事件监视器是使用以下语句激活的:

```
SET EVENT MONITOR CFGHIST STATE=1
```

下表显示 CFGHIST 变更历史记录事件监视器可能写至 CHANGESUMMARY_CFGHIST 表的一些样本事件监视器数据。所有变更历史记录事件监视器都将数据写至 CHANGESUMMARY 逻辑数据组。按“CHANGESUMMARY 逻辑数据组”中的描述，CHANGESUMMARY 逻辑数据组返回一些概述所捕获事件的事件监视元素，以下输出

仅显示了这些元素中的一部分。表名通过将用于填充表 (CHANGESUMMARY) 的逻辑数据组的名称与 CREATE EVENT MONITOR 语句 (CFGHIST) 中给予该事件监视器的名称并置派生。

```
APPL_ID                       APPL_NAME .... EVENT_ID EVENT_TIMESTAMP
----------------------------- --------- .... -------- ------------------
*LOCAL.tripathy.111028110756 db2bp     .... 1        28/10/2011 07:12:02

EVENT_TYPE MEMBER ....
---------- ------ ....
EVMONSTART 0      ....
```

因为性能以前不存在问题，所以 DBA 怀疑该问题可能由某个最新更改导致，于是执行以下步骤:

1. 检查 CHANGESUMMARY 逻辑数据组以查找过去 24 小时内所做的任何更改。对于此示例，假定当前时间为 2011 年 10 月 31 日 06:00:00。

   ```
   SELECT EVENT_TYPE FROM CHANGESUMMARY_CFGHIST
      WHERE EVENT_TIMESTAMP > CURRENT TIMESTAMP - 24 HOURS
   ```

   查询返回下列结果:

   ```
   EVENT_TYPE
   ----------
   DBCFG
   DBCFG
   ```

   输出指示过去 24 小时内进行了两次数据库配置更新。

2. 查询 DBDBMCFG 逻辑数据组以获取这些配置更改的详细信息。

3. ```
   SELECT EVENT_TIMESTAMP, CFG_NAME, CFG_VALUE, CFG_OLD_VALUE, DB_DEFERRED
      FROM DBDBMCFG_CHGHIST
   ```

   查询返回下列结果:

   ```
   EVENT_TIMESTAMP       CFG_NAME      CFG_VALUE   CFG_OLD_VALUE   DB_DEFERRED
   --------------------  ------------  ---------   -------------   -----------
   30/10/2011 08:41:39   LOCKLIST           1024            2048   N
   30/10/2011 08:42:35   LOCKTIMEOUT           0              -1   Y
   ```

   输出指示在性能下降的时间段内进行了锁定更改。

DBA 注意到 LOCKTIMEOUT 更改已延迟，于是发出查询以检查数据库是否在进行此配置更改后激活。此检查确定数据库是否实现了此配置更改。如果未实现此更改，那么不太可能是此更改导致性能问题。数据库激活时间记录在 EVMONSTART 逻辑数据组中。缺省情况下，所有变更历史记录事件监视器都将数据写至 EVMONSTART 逻辑数据组。

```
SELECT COUNT (*)as POST_CFG_ACTIVATIONS FROM EVMONSTART_CHGHIST
   WHERE DB_CONN_TIME > TIMESTAMP(2011-10-30-08:42:35)
```

查询返回非零值。

```
POST_CFG_ACTIVATIONS
--------------------
                   1
```

此非零值确认数据库在 LOCKTIMEOUT 配置参数更改后激活，这意味着新值已生效。DBA 现在了解了系统上进行了哪些更改，并尝试将与锁定相关的配置参数调整回其原始值以查看此操作是否解决了该问题。
```

注: 如果更改配置参数时变更历史记录事件监视器处于不活动状态，那么该事件监视器将不捕获 DBCFG 事件。反而，变更历史记录事件监视器将在该事件监视器启动后捕获 DBCFGVALUES 事件。在 DBDBMCFG 逻辑数据组中，每行表示一个配置参数，此配置参数作为 DBCFG 或 DBMCFG 事件的一部分更新或作为 DBCFGVALUES 或 DBMCFGVALUES 事件的一部分在事件监视器启动时捕获。CFG_COLLECTION_TYPE 监视元素标识该记录是描述配置参数更新还是在事件监视器启动时记录的初始值。DBA 将需要比较当前变更历史记录事件监视器启动时捕获的值与先前捕获的值，以查找可能导致该问题的已更改值。检查诊断日志也会有所帮助。

示例: 使用变更历史记录事件监视器来标识配置更改和实用程序执行:

可使用变更历史记录事件监视器来确定是否有任何最新配置更改或最新实用程序执行。

方案

在此示例中，数据库管理员 (DBA) 注意到过去 24 小时数据库性能发生了变化。DBA 先前创建了名为 HIST 的变更历史记录事件监视器，他已使用此事件监视器了解数据库和数据库管理系统的行为、性能或稳定性方面的更改。

DBA 针对 CHANGESUMMARY 逻辑数据组发出以下查询，以汇总过去 24 小时内发生的所有更改事件或实用程序执行。

```
SELECT EVENT_TIMESTAMP,
       EVENT_TYPE,
       UTILITY_TYPE,
       COORD_MEMBER,
       MEMBER
  FROM CHANGESUMMARY_HIST
  WHERE EVENT_TIMESTAMP > CURRENT TIMESTAMP – 24 HOURS
  ORDER BY BY EVENT_TIMESTAMP ASC
```

此查询可能返回类似如下的输出:

```
EVENT_TIMESTAMP            EVENT_TYPE     UTILITY_TYPE COORD_MEMBER MEMBER
-------------------------- -------------- ------------ ------------ ------
2010-10-31-17.29.04.545210 DBCFG                                 0      0
2010-10-31-18.29.04.545210 UTILSTART      LOAD                   0      0
2010-10-31-18.40.04.545210 UTILSTARTPROC  LOAD                   0      0
2010-10-31-18.50.04.545210 UTILSTOPPROC   LOAD                   0      0
2010-10-31-18.40.04.545210 UTILSTARTPROC  LOAD                   0      1
2010-10-31-18.50.04.545210 UTILSTOPPROC   LOAD                   0      1
2010-10-31-19.29.04.545210 UTILSTOP       LOAD                   0      0
2010-10-31-19.56.04.545210 UTILSTART      BACKUP                 0      0
2010-10-31-20.09.04.545210 UTILPHASESTART BACKUP                 0      0
2010-10-31-20.29.04.545210 UTILPHASESTOP  BACKUP                 0      0
2010-10-31-21.29.04.545210 UTILSTOP       BACKUP                 0      0

9 record(s) selected.
```

在此输出中，DBA 确定过去 24 小时内发生了配置更改和实用程序执行。他现在可查询其他变更历史记录事件监视器逻辑数据组并获取有关 CHANGESUMMARY 逻辑数据组中返回的事件的更多信息。例如，要获取有关 UTILSTART 事件的更多信息，DBA 可查询 UTILSTART 逻辑数据组以了解实用程序要作用于哪些对象以及启动实用程序时使用了哪些选项。

示例: 使用变更历史记录事件监视器来列示 LOAD 操作:

可使用变更历史记录事件监视器来跟踪针对数据库执行的所有 LOAD 操作。

方案

在此示例中，数据库管理员 (DBA) 要捕获和列示针对数据库的所有装入实用程序执行的历史记录。要跟踪装入实用程序事件，请执行以下操作:

1. 创建用于跟踪 LOAD 事件的变更历史记录事件监视器。例如:

```
CREATE EVENT MONITOR MON_LOAD
   FOR CHANGE HISTORY WHERE EVENT IN (LOAD)
   WRITE TO TABLE
      CHANGESUMMARY (TABLE UTIL_COMMON),
      UTILSTART (TABLE LOAD_START),
      UTILSTOP (TABLE LOAD_STOP)
      UTILLOCATION (TABLE LOAD_INPUT_FILES)
      UTILPHASE (TABLE LOAD_PHASES);
```

2. 激活事件监视器。

```
SET EVENT MONITOR MON_LOAD STATE=1
```

3. 查询逻辑数据组以查找有关针对数据库执行的 LOAD 操作的信息。例如，以下查询列示所有已执行装入实用程序的启动和停止时间。此查询仅显示协调程序启动和停止时间。它会忽略暂停和恢复记录以显示实用程序执行的完整耗用时间。

```
SELECT A.APPL_ID,
       A.COORD_MEMBER,
       A.EVENT_TIMESTAMP AS START_TIME,
       B.EVENT_TIMESTAMP AS STOP_TIME,
       A.TABLE_SCHEMA,
       A.TABLE_NAME,
       SQLCODE,
       VARCHAR(A.UTILITY_DETAIL, 200) AS DETAIL
   FROM LOAD_START AS A
        LOAD_STOP AS B
        UTIL_COMMON AS C
   WHERE A.UTILITY_INVOCATION_ID = B.UTILITY_INVOCATION_ID AND
         A.UTILITY_START_TYPE = 'START' AND
         B.UTILITY_STOP_TYPE = 'STOP' AND
         A.MEMBER = B.MEMBER AND
         A.MEMBER = A.COORD_MEMBER
   ORDER BY A.EVENT_TIMESTAMP ASC
```

查询结果显示执行了两个装入实用程序，并且提供有关它们的启动和停止时间的详细信息、装入的目标（表名）以及所执行装入的详细信息。

```
APPL_ID                  START_TIME                 STOP_TIME
------------------------ -------------------------- --------------------------
*LOCAL.test.110131213809 2010-10-31-17.29.04.545210 2010-10-31-17.29.04.545210
*LOCAL.test.110131213809 2010-10-31-17.29.04.545210 2010-10-31-17.29.04.545210

TABLES_SCHEMA TABLE_NAME SQLCODE DETAIL
------------- ---------- ------- --------.....
TEST          T1               0 LOAD CURSOR..
TEST          T3               0 LOAD DEL...

2 record(s) selected.
```

示例: 使用变更历史记录事件监视器来报告实用程序执行的历史记录:

可使用变更历史记录事件监视器来跟踪针对数据库执行的实用程序操作。

方案

在此示例中，数据库管理员 (DBA) 要捕获和列示针对数据库的实用程序事件执行。要报告实用程序事件，请执行以下操作:

1. 创建用于跟踪实用程序事件的变更历史记录事件监视器。例如:

```
CREATE EVENT MONITOR MON_UTIL
   FOR CHANGE HISTORY WHERE EVENT IN (UTILALL)
   WRITE TO TABLE
      CHANGESUMMARY (TABLE UTIL_COMMON),
      UTILSTART (TABLE UTIL_START),
      UTILSTOP (TABLE UTIL_STOP)
      UTILLOCATION (TABLE UTIL_LOCATION)
      UTILPHASE (TABLE UTIL_PHASES) AUTOSTART;
```

2. 启用事件监视器。

```
SET EVENT MONITOR MON_UTIL STATE=1
```

3. 查询逻辑数据组以查找有关针对数据库执行的实用程序操作的信息。例如，以下查询列示针对每个成员的每个实用程序调用的历史记录。

```
WITH UTIL_HIST(TIMESTAMP, UTIL_TYPE, ACTION, PHASE_TYPE, UTILITY_INVOCATION_ID,
   MEMBER, SQLCODE) AS
 (SELECT A.EVENT_TIMESTAMP,
        A.UTILITY_TYPE,
        CAST('START' AS VARCHAR(32)),
        CAST(NULL AS VARCHAR(16)),
        A.UTILITY_INVOCATION_ID,
        A.MEMBER,
        CAST(NULL as INTEGER)
 FROM UTIL_START AS A
 UNION ALL
  SELECT A.EVENT_TIMESTAMP,
         A.UTILITY_TYPE,
         CASE WHEN EVENT_TYPE IN ('UTILPHASESTART') THEN
            CAST('PHASE START' AS VARCHAR(32))
         ELSE
            CAST('PHASE STOP' AS VARCHAR(32))
         END CASE,
         CAST(UTILITY_PHASE_TYPE AS VARCHAR(16)),
         A.UTILITY_INVOCATION_ID,
         A.MEMBER,
         CAST(NULL as INTEGER)
  FROM UTIL_PHASE AS B
 UNION ALL
  SELECT A.EVENT_TIMESTAMP,
         A.UTILITY_TYPE,
         CAST('STOP' AS VARCHAR(32))
         CAST(NULL AS VARCHAR(16)),
         A.UTILITY_INVOCATION_ID,
         A.MEMBER,
         A.SQLCODE
  FROM UTIL_STOP AS C)
SELECT * FROM UTIL_HIST
ORDER BY UTILITY_INVOCATION_ID, MEMBER, TIMESTAMP ASC
```

生成的实用程序事件报告可用于:

- 标识重叠的任何实用程序。例如，一个实用程序尚未在分区上停止，另一个实用程序就在同一分区上启动。
- 确定实用程序将其时间消耗在何处。例如，在每个阶段消耗的时间。注意: 在 V10.1 中，它仅对联机备份的表空间备份阶段可用。

```
TIMESTAMP                  UTIL_TYPE ACTION      PHASE_TYPE    UTILITY_INVOCATION_ID MEMBER SQLCODE
-------------------------- --------- ----------  ------------- --------------------- ------ -------
2010-10-31-17.29.04.545210 LOAD      START       -             x'18A901F...621'      0      -
2010-10-31-17.50.04.344230 LOAD      STOP        -             x'18A901F...621'      0      0
2010-10-31-17.29.04.545211 LOAD      START       -             x'18A901F...633'      1      -
2010-10-31-17.50.04.344229 LOAD      STOP        -             x'18A901F...633'      1      0
2010-10-31-17.29.04.344210 BACKUP    START       -             x'18A901F...645'      0      -
2010-10-31-17.50.04.344211 BACKUP    PHASE START BACKUPTS      x'18A901F...645'      0      0
2010-10-31-17.51.04.545214 BACKUP    PHASE STOP  BACKUPTS      x'18A901F...645'      0      -
2010-10-31-17.52.04.344218 BACKUP    STOP        -             x'18A901F...645'      0      0
8 record(s) selected.
```

示例：使用变更历史记录事件监视器来列示所有已落实 DDL 语句：

可使用变更历史记录事件监视器来快速列示所有已执行的已落实 DDL 语句，以确定任何所做更改是否可能正在影响工作负载。

方案

在此示例中，数据库管理员 (DBA) 注意到过去 24 小时许多查询性能下降。DBA 使用变更历史记录事件监视器来快速检查该时间段执行的 DDL，以确定是否进行了可能对工作负载有很大影响的任何更改（例如，删除的索引）。DBA 先前创建了名为 CHGHIST 的变更历史记录事件监视器，它用于跟踪 DDL 语句。DBA 发出了以下语句以列示变更历史记录事件监视器在过去 24 小时内捕获的所有已落实 DDL 语句。所发出语句排除了通过 ROLLBACK 语句或 ROLLBACK TO SAVEPOINT 语句回滚的语句。

请注意，变更历史记录事件监视器记录执行 DDL 时的 DDL 事件。DDL 是否导致数据库中发生任何更改取决于要落实的 DDL。

```
WITH savepoint_rollbacks (global_tran_id, local_tran_id, savepoint_id) AS
   (SELECT DISTINCT T.global_transaction_id, T.local_transaction_id, T.savepoint_id
      FROM DDLSTMTEXEC_CHGHIST as D, TXNCOMPLETION_CHGHIST as T
      WHERE T.txn_completion_status='S' AND
            D.savepoint_id >= T.savepoint_id AND
            D.event_timestamp <= T.event_timestamp)
   SELECT VARCHAR(D.STMT_TEXT, 70) AS STMT_TEXT FROM DDLSTMTEXEC_CHGHIST as D,
            TXNCOMPLETION_CHGHIST as T
      WHERE D.global_transaction_id = T.global_transaction_id AND
            D.local_transaction_id = T.local_transaction_id AND
            T.txn_completion_status = 'C' AND
            (D.global_transaction_id, D.local_transaction_id, D.savepoint_id)
   NOT IN (SELECT * FROM savepoint_rollbacks) AND
            D.EVENT_TIMESTAMP > CURRENT TIMESTAMP - 24 HOURS;

STMT_TEXT
----------------------------------------------------------------------
CREATE INDEX I1 ON T1 (ONE)

1 record(s) selected.
```

示例：使用变更历史记录事件监视器来列示 STMM 执行的更改：

可使用变更历史记录事件监视器来列示自调整内存管理器 (STMM) 执行的更改。

方案

在此示例中，数据库管理员 (DBA) 要监视 STMM 执行的任何更改。因为 STMM 可修改配置参数和缓冲池大小，所以 DBA 已创建名为 HIST 的事件监视器来捕获配置和 DDL 更改。

可通过查询该事件监视器来查找包含下列其中一个信息的记录以找到 STMM 发起的更改：

- 应用程序名称 (appl_name) 为 db2stmm。
- DDL 语句文本 (stmt_text) 包含带关键字 db2stmm 的注释。请注意，某些 DDL 更改是其他应用程序代表 STMM 执行的。

```
SELECT A.EVENT_TIMESTAMP,
      VARCHAR(A.EVENT_TYPE, 20) AS EVENT_TYPE, A.MEMBER
   FROM CHANGESUMMARY_HIST A LEFT OUTER JOIN
      DDLSTMTEXEC_HIST B
   ON A.EVENT_TIMESTAMP = B.EVENT_TIMESTAMP AND
      A.MEMBER = B.MEMBER AND
      A.EVENT_ID = B.EVENT_ID
   WHERE (A.APPL_NAME = 'db2stmm' OR
      B.STMT_TEXT LIKE '%db2stmm%');
```

此查询可能返回类似以下示例的输出:

```
EVENT_TIMESTAMP            EVENT_TYPE           MEMBER
-------------------------- -------------------- ------
2011-04-22-12.12.17.832316 DBCFG                     0
2011-04-22-12.22.35.227550 DBCFG                     0
2011-04-22-12.12.17.530274 DBCFG                     0
2011-04-22-12.12.17.721403 DBCFG                     0
2011-04-22-12.12.17.776889 DBCFG                     0
2011-04-22-12.22.35.172119 DBCFG                     0
2011-04-22-12.12.17.665098 DBCFG                     0
2011-04-22-12.22.35.116343 DBCFG                     0
2011-04-22-12.29.47.092822 DBCFG                     0
2011-04-22-12.29.47.037709 DBCFG                     0
2011-04-22-12.12.17.600511 DBCFG                     0
2011-04-22-12.22.35.283320 DBCFG                     0
2011-04-22-12.29.46.752477 DBCFG                     0
2011-04-22-12.29.47.148562 DBCFG                     0

 14 record(s) selected.
```

跨发行版保留事件监视器数据

从 DB2 V10.1 开始，可在升级 DB2 产品后升级事件监视器输出表。此功能允许您保留升级前拥有的事件监视器表中可能存在的任何数据。

DB2 产品中的事件监视器增强后，它们生成的表可能会更改。例如，可能会向表添加新列以报告新的监视元素。在 V10.1 之前，如果现有事件监视器已写至包含您要保留的数据的表，并且您想要收集新添加列中的数据，那么升级至新发行版后您需要手动改变这些表。此改变包括添加您可能要使用的任何新列。如果未添加这些新列，那么事件监视器将按它在先前发行版中的方式工作，仅捕获该事件监视器在该发行版中支持的数据。

已更改的无格式事件表根本不能升级；您必须删除这些表然后重新创建。

EVMON_UPGRADE_TABLES 存储过程会升级现有事件监视器表的定义以与当前级别的 DB2 产品生成的事件监视器表相匹配。此功能允许您保留您可能拥有的任何现有表以及它们包含的所有数据，从而使您不必手动改变表或者删除表然后重新创建。

注: 从 V10.1 开始，还可使用 ALTER EVENT MONITOR 语句向事件监视器添加新的逻辑组。可使用此方法作为 EVMON_UPGRADE_TABLES 的替代品来添加新发行版中添加的逻辑数据组。但是，不能使用 ALTER EVENT MONITOR 来修改已与该事件监视器相关联的逻辑组；如果已与该事件监视器相关联的逻辑数据组已更改，那么修改该事件监视器的唯一方法是使用 EVMON_UPGRADE_TABLES 过程。

EVMON_UPGRADE_TABLES 过程可处理常规表和 UE 表。对于常规表，该过程添加任何所需新列，删除不再需要的旧列并根据需要改变任何列。对于 UE 表，该过程根据需要添加新列和修改现有列，以允许该 UE 表供 **db2evmonfmt** 工具、EVMON_FORMAT_UE_TO_TABLES 或 EVMON_FORMAT_UE_TO_XML 例程处理。

要点：必须取消激活任何处于活动状态的事件监视器，升级过程才能正常工作。EVMON_UPGRADE_TABLES 过程在开始升级表之前会自动取消激活任何处于活动状态的事件监视器。不要重新激活任何带有 EVMON_UPGRADE_TABLES 正处理的表的事件监视器，否则升级过程将失败。升级之前处于活动状态的所有事件监视器在升级完成之后再次被激活。

不升级事件监视器表的隐含意义

与过去发行版中一样，可选择不升级事件监视器表。但是，已添加至新发行版中该事件监视器的所有新列将不会填充数据，并且不能用于查询。而且，先前存在于旧发行版中并且在新发行版中大小已增加的所有监视元素的值可能被截断。例如，如果监视元素的大小在新发行版中从 VARCHAR(20) 增加至 VARCHAR(128)，并且您未升级先前存在的表，那么包含这些监视元素值的列仍然仅存储 20 个字符的数据，即使系统可能正将该监视元素的 128 字节数据发送至事件监视器也是如此。

升级由 EVMON_FORMAT_UE_TO_TABLES 生成的表

与 UE 表配合使用时，EVMON_UPGRADE_TABLES 过程升级 UE 表本身；它对您可能已使用 EVMON_FORMAT_UE_TO_TABLES 过程创建的所有常规表不起作用。使用 EVMON_UPGRADE_TABLES 升级 UE 表之后，还可升级 EVMON_FORMAT_UE_TO_TABLES 生成的输出表。从 DB2 V10.1 开始，EVMON_FORMAT_UE_TO_TABLES 过程支持新选项 UPGRADE_TABLES。运行带此选项的 EVMON_FORMAT_UE_TO_TABLES 过程时，此过程生成的任何现有表会改变，以便表列与新版本的 EVMON_FORMAT_UE_TO_TABLES 过程生成的输出相匹配。

有关更多信息，请参阅 EVMON_FORMAT_UE_TO_TABLES 的参考信息。

第 4 章 其他监视接口

使用 MONREPORT 模块生成的报告

MONREPORT 模块将生成监视数据的文本报告，然后可以使用这些报告对 SQL 性能问题进行故障诊断。

可以使用 MONREPORT 模块生成下列报告:

表 107. 使用 MONREPORT 模块生成的报告列表

报告名称	用来创建报告的过程	主要数据源/表函数
摘要报告	MONREPORT.DBSUMMARY	MON_GET_SERVICE_SUBCLASS 以及从 MON_GET_CONNECTION 和 MON_GET_WORKLOAD 中选择的详细信息
连接报告	MONREPORT.CONNECTION	MON_GET_CONNECTION
当前应用程序报告	MONREPORT.CURRENTAPPS	包括 MON_GET_CONNECTION、MON_GET_UNIT_OF_WORK、WLM_GET_SERVICE_CLASS_AGENTS 和 WLM_GET_WORKLOAD_OCCURRENCE_ACTIVITIES 中的字段
当前 SQL 报告	MONREPORT.CURRENTSQL	MON_GET_PKG_CACHE_STMT（对于从 WLM_GET_WORKLOAD_OCCURRENCE_ACTIVITIES 表函数获取的 **executable_id**。）
程序包高速缓存报告	MONREPORT.PKGCACHE	MON_GET_PKG_CACHE_STMT
当前锁定等待报告	MONREPORT.LOCKWAIT	MON_GET_APPL_LOCKWAIT 中的大部分数据；MON_GET_CONNECTION、WLM_GET_SERVICE_CLASS_AGENTS、WLM_GET_WORKLOAD_OCCURRENCE_ACTIVITIES、MON_GET_PKG_CACHE_STMT 和 MON_GET_TABLE 中的其他数据

大多数报告都是从摘要部分开始，摘要部分为此报告中的每一项都提供了一行重要信息。例如，“连接”报告的摘要部分为每个连接都提供了一行信息。对于摘要中的每一项，在报告的主体中都提供了一个详细部分。

报告中的每个度量值都标有底层监视元素的名称（例如: CLIENT_IDLE_WAIT_TIME = 44）。要确定该度量值所表示的内容，请在信息中心中搜索该监视元素的名称。

您可以定制由 MONREPORT 模块所生成的报告。MONREPORT 模块完全是使用 SQL 实现的，可以从数据库目录中获取该模块的代码，然后创建一个定制版本。

初始诊断报告

这些报告的一个重要用途是对 SQL 性能下降问题进行故障诊断。每个报告旨在回答某些诊断问题。某些报告支持进行初始诊断，而其他报告则支持稍后对特定类型的问题进行详细诊断。

初始诊断涉及到下列事项:

- 要确定问题所属的类别，可将问题的范围缩小到出现速度下降的处理所属的方面或阶段。
- 确定该问题所涉及到的 SQL 语句，并收集有关这些 SQL 语句的信息以便进一步分析。

表 108. 适合于初始诊断的 MONREPORT 模块报告

过程名称	所提供的信息及其用途
MONREPORT.DBSUMMARY 第 1 部分: 系统性能	在摘要报告的第 1 部分，提供了整个数据库中聚集的处理的大多数方面的监视数据。 要回答有关出现速度下降的处理所属方面或阶段的问题，此信息将很有用。例如: • 是在数据服务器内部还是外部发生了此问题？ • 计算资源是否遇到了瓶颈？ • 请求是否处于等待状态？如果处于等待状态，那么是在等待哪个资源？ • 是特定数据服务器处理组件的速度下降吗？
MONREPORT.DBSUMMARY 第 2 部分: 应用程序性能	摘要报告的第 2 部分提供了每个连接、工作负载和服务类的关键性能指标。 要回答有关涉及到速度下降问题的应用程序请求的作用域的问题，此信息将很有用。例如: • 此速度下降是会影响很多工作负载或者所有工作负载的常规系统速度下降吗？ • 此速度下降问题是否仅限于从特定源（例如，特定连接、DB2 工作负载或者 DB2 服务类）发出的 SQL 语句？
MONREPORT.DBSUMMARY 第 3 部分: 成员级别的信息	摘要报告的第 3 部分提供了每个成员的关键性能指标。 要确定速度下降问题是否与某个成员或者某些成员无关，此信息很有用。
MONREPORT.CURRENTSQL	当前 SQL 报告采用前 *N* 个活动的多个列表的形式，提供了有关当前正在运行的语句的信息。这些语句按不同的度量值进行排序: 处理资源、已处理的行、直接读和直接写。 要确定速度下降问题是否与某个 SQL 语句或者某些 SQL 语句无关，此信息很有用。如果速度下降问题与某个 SQL 语句或者某些 SQL 语句无关，那么这些语句可能是出现在此报告中最前面的语句。

表 108. 适合于初始诊断的 MONREPORT 模块报告 (续)

过程名称	所提供的信息及其用途
MONREPORT.PKGCACHE	程序包高速缓存报告提供了有关最近已经运行并且存储在程序包高速缓存中的语句的信息。此报告显示了多个摘要，每个摘要列示了前 *N* 个活动。这些活动是按下列监视元素排序的: • CPU • 等待时间 • 已处理的行 • num_coord_exec_with_metrics -“协调代理程序已执行的次数以及度量值”监视元素（如果未指定成员），或者 num_exec_with_metrics -“执行次数以及已收集的度量值”监视元素（如果指定了成员） • I/O 等待时间 此报告包含其中每个度量值的摘要以及关于每次执行的报告。 要确定速度下降问题是否与某个 SQL 语句或者某些 SQL 语句无关，此信息很有用。如果无关，那么这些语句可能会显示在此报告顶部。关于每次执行的信息可以帮助确定成本最高的语句，而针对所有执行汇总的信息可帮助确定对系统造成的影响最大的语句，此处所说的影响是从语句成本和执行频率两方面综合考虑的。
MONREPORT.CURRENTAPPS	当前应用程序报告显示工作单元、代理程序和活动的当前处理状态。此报告以一个显示了当前连接数和活动数的摘要部分开头，接下来还有一系列摘要，例如，按工作负载状态列示的当前工作单元的摘要。在此报告的主体中，每个连接都有相应的一个部分，此部分提供了有关该连接的详细信息。 此信息对于查看当前正在系统中运行的所有工作很有用。这使您可以检查可确定问题类别的模式。

详细诊断报告

完成初始诊断之后，可能需要对您在初始诊断阶段所确定的问题类别进行一系列专门或详细的故障诊断分析。

表 109. 适合于详细诊断的 MONREPORT 模块报告

过程名称	所提供的信息及其用途
MONREPORT.CONNECTION	如果 MONREPORT.DBSUMMARY 报告指出了速度下降问题仅限于从特定连接发出的 SQL 语句，那么您可以查看有关受影响的连接的详细信息。 此报告与 MONREPORT.DBSUMMARY 报告的第 1 部分包含相同的度量值，但是此报告为每个连接都提供了此信息。
MONREPORT.LOCKWAIT	如果在初始诊断期间查看的报告指出有一个锁定等待问题，那么您可以查看有关当前正在等待的每个锁定等待的详细信息。 此信息包括锁定持有者和锁定请求者的详细信息，还包括持有的锁定和请求的锁定的特征。

定制 MONREPORT 模块报告

可定制 MONREPORT 模块生成的现有报告或根据现有报告创建新报告。

关于此任务

可更改报告的措辞或组织；或者添加、除去或修改报告中包含的监视元素。此外，可根据现有报告创建新报告。

MONREPORT 模块是使用 SQL 语句（包括存储过程和数据类型）实现的。要创建 MONREPORT 模块的定制版本，请从数据库目录获取模块代码，然后进行修改和部署。SYSIBMADM 模式中提供了 MONREPORT 模块。

MONREPORT 模块中的每个过程用于生成一个报告。例如，CONNECTION 过程用于生成连接报告。调用过程（例如，CONNECTION 过程）时，要生成报告，此过程会调用其他内部例程来执行报告的最终组合和格式化。内部例程执行以下函数:

- 调用 DB2 监视表函数以获取监视数据。对于许多报告，这些例程将调用表函数，等待时间间隔，然后再次调用表函数。
- 生成变化量值，以记录监视时间间隔开头和结尾的监视值之差。
- 执行计算以获取百分比、比率、小计和聚集。

可定制以下内部例程:

表 110. 可定制的例程

名称	描述	由报告使用
CONNDELTAS	此存储过程从连接度量值返回变化量值的结果集。	此例程与连接报告一起使用。
COMMONREQMETRICS	此存储过程计算度量值并格式化连接和摘要报告的度量值的报告输出。	MONREPORT.CONNECTION 和 MONREPORT.DBSUMMARY

MONREPORT 模块包含其他对象，您不需要修改这些对象就可定制此模块。

表 111. 不需要定制的例程

名称	描述	由报告使用
INITMSGCACHE	此存储过程检索报告中出现的可转换字符串。	在所有报告中使用。
SAVE_EXEC_INFO	内部使用的存储过程，用于存储有关所执行的 SQL 语句部分的信息。	由 MONREPORT.PKGCACHE 使用
db2monreport.src	此文件为二进制格式，并且由 INITMSGCACHE 例程访问。此文件包含报告中出现的文本字符串。	在所有报告中使用。

表 112. 在例程中使用的不能定制的数据类型

名称	描述	由报告使用
名称为 MONMETRICS_CHAR255_TYPE 和 MONMETRICS_CHAR32_TYPE 之类的数据类型。	这些数组数据类型用于存储表函数返回的监视元素。	在大部分或所有报告中使用。
REPORT_TYPE	数组数据类型，用于存储在例程终止时返回并显示的文本输出。	在大部分或所有报告中使用。
MONEXEC_TYPE	数组数据类型，用于存储唯一标识所执行的 SQL 语句部分的可执行标识。	在大部分或所有报告中使用。

过程

1. 获取代码。
2. 定制代码以满足您的需要。 重命名此模块以区分定制版本与原始版本，并使用任何适用的 SQL 编辑器或存储过程构建器来修改代码。
   - 要向报告添加文本，请直接向报告过程执行添加操作。不要使用 INITMSGCACHE 例程来管理文本字符串。
   - 要除去使用 INITMSGCACHE 例程获取的报告文本，请除去使用已高速缓存的字符串的关联代码。

   例如，要更改 MONREPORT.CONNECTION 报告，请执行以下操作:
   - 要添加或除去度量值，请修改 CONNDELTAS 过程中的 SQL 查询。要添加度量值，请指定要选择的其他列。
   - 要更改连接报告输出的详细信息部分，请更新 COMMONREQMETRICS 例程以添加计算并确保输出中显示新度量值。
   - 要更改连接报告的摘要部分，请更新 CONNECTION 例程以确保输出中显示新度量值。
3. 部署 MONREPORT 模块的定制版本。 对模块及其例程写入一组 CREATE 或 ALTER 语句。如果正在使用图形 SQL 编辑器，那么此编辑器将自动完成某些部署步骤。

快照监视器

可使用快照监视器以在特定时间捕获有关数据库和所有已连接应用程序的信息。快照对于确定数据库系统的状态非常有用。

如果每隔一定时间间隔获取快照，那么快照在观察趋势和预测潜在问题方面时也很有用。快照监视器中的某些数据获取自系统监视器。系统监视器中的可用数据由系统监视开关确定。

系统监视器仅累积数据库处于活动状态时的信息。如果所有应用程序与数据库断开连接并且数据库释放，那么不再提供该数据库的系统监视器数据。可通过使用 ACTIVATE DATABASE 命令启动数据库或保持与数据库的永久连接，以便让数据库保持活动状态直到获取最终快照。

快照监视需要实例连接。如果没有实例连接，那么创建缺省实例连接。通常，对于应用程序调用第一个数据库系统监视器 API 时，将隐式建立与 DB2INSTANCE 环境变量指定的实例的连接。还可使用 ATTACH TO 命令显式建立连接。一旦连接了应用程序，它调用的所有系统监视器请求都将引导至该实例。这允许客户机只需要连接至远程服务器上的实例就可以监视该远程服务器。

在分区数据库环境中，可在实例的任何分区上获取快照，或者使用单个实例连接获取全局快照。全局快照聚集在每个分区上收集到的数据并返回一组值。

在 DB2 pureScale 环境中，可在任何成员上获取快照或全局获取快照。全局快照会聚集在每个成员上收集的数据并返回一组值。

可从 CLP、SQL 表函数捕获快照，也可以通过使用 C 或 C++ 应用程序中的快照监视器 API 来捕获快照。有若干不同快照请求类型可用，每种类型返回特定类型的监视数据。例如，可捕获仅返回缓冲池信息的快照，或者捕获返回数据库管理器信息的快照。在捕获快照前，请考虑是否需要由监视开关控制的监视元素提供的信息。如果特定监视开关处于关闭状态，就不会收集它控制的监视元素。

访问系统监视器数据: SYSMON 权限

作为 SYSMON 数据库管理器级别组的成员的用户有权获取对数据库系统监视器数据的访问权。系统监视器数据是通过使用快照监视器 API、CLP 命令或 SQL 表函数来访问的。

SYSMON 权限组提供了方法，使没有系统管理或系统控制权限的用户能够访问数据库系统监视器数据。

除了 SYSMON 权限之外，访问使用快照监视器的系统监视器数据的唯一方法是使用系统管理或系统控制权限。

属于 SYSMON 组或具有系统管理或系统控制权限的任何用户可以执行下列快照监视器函数:

- CLP 命令:
  - GET DATABASE MANAGER MONITOR SWITCHES
  - GET MONITOR SWITCHES
  - GET SNAPSHOT

  - LIST ACTIVE DATABASES
  - LIST APPLICATIONS
  - LIST DCS APPLICATIONS
  - LIST UTILITIES
  - RESET MONITOR
  - UPDATE MONITOR SWITCHES
- API:
  - db2GetSnapshot - 获取快照
  - db2GetSnapshotSize - 估计 db2GetSnapshot() 输出缓冲区所需的大小
  - db2MonitorSwitches - 获取/更新监视开关
  - db2ResetMonitor - 重置监视器
- 无需预先运行 SYSPROC.SNAP_WRITE_FILE 的快照 SQL 表函数

使用快照管理视图和表函数来捕获数据库系统快照

授权用户可使用快照管理视图或快照表函数来捕获 DB2 实例的监视器信息快照。快照管理视图提供了一种简单的方法，可用来访问已连接数据库的所有数据库分区的数据。快照表函数允许您请求特定数据库分区的数据，全局聚集数据或所有数据库分区中的数据。某些快照表函数允许您请求所有活动数据库中的数据。

开始之前

必须具有 SYSADM、SYSCTRL、SYSMAINT 或 SYSMON 权限才能捕获数据库快照。要获取远程实例的快照，必须先连接至属于该实例的本地数据库。

关于此任务

虽然新的监视器数据可用时在将来发行版中可能需要新的快照表函数，但在新列添加至视图时快照管理视图集合将保持不变，使得长期进行应用程序维护时管理视图成为一个很好的选择。

每个快照视图返回一个表，每个数据库分区每个被监视对象对应一行，每列表示一个监视元素。每个表函数返回一个表，指定分区的每个被监视对象对应一行。返回的表的列名与监视元素名称相关。

例如，SAMPLE 数据库的一般应用程序信息的快照将使用如下 SNAPAPPL 管理视图进行捕获:

```
SELECT * FROM SYSIBMADM.SNAPAPPL
```

您也可以从返回的表中选择个别监视元素。例如，以下语句仅返回 **agent_id** 和 **appl_id** 监视元素:

```
SELECT agent_id, appl_id FROM SYSIBMADM.SNAPAPPL
```

限制

快照管理视图和表函数不能与下列任一项一起使用:

- 监视开关命令或 API
- 监视器重置命令或 API

此限制包括:

- **GET MONITOR SWITCHES**
- **UPDATE MONITOR SWITCHES**
- **RESET MONITOR**

出现此限制是因为这类命令使用 ATTACH 命令，而快照表函数使用 CONNECT 语句。

过程

- 要使用快照管理视图捕获快照:
  1. 连接到数据库。 这可以是实例中任何需要监视的数据库。为了能够使用快照管理视图发出 SQL 查询，必须连接到数据库。
  2. 确定需要捕获的快照的类型。 如果想要捕获当前连接的数据库之外的数据库的快照，或者如果想要从单个数据库分区或全局聚集数据中检索数据，那么需要改为使用快照表函数。
  3. 使用适当的快照管理视图发出查询。 例如，以下查询将对当前连接的数据库捕获锁定信息的快照:

     ```
     SELECT * FROM SYSIBMADM.SNAPLOCK
     ```

- 要使用快照表函数捕获快照:
  1. 连接到数据库。 这可以是实例中任何需要监视的数据库。为了能够使用快照表函数发出 SQL 查询，必须连接到数据库。
  2. 确定需要捕获的快照的类型。
  3. 使用适当的快照表函数发出查询。 例如，以下查询将对当前连接的数据库分区捕获有关 SAMPLE 数据库的锁定信息的快照:

     ```
     SELECT * FROM TABLE(SNAP_GET_LOCK('SAMPLE',-1)) AS SNAPLOCK
     ```

     SQL 表函数有两个输入参数:

     数据库名称
     VARCHAR(255)。如果输入 NULL，就会使用当前连接的数据库的名称。

     分区号
     SMALLINT。对于数据库分区号参数，输入对应于需要监视的数据库分区号的整数（0 到 999 之间的值）。要捕获当前连接的数据库分区的快照，请输入值 -1。要捕获全局聚集快照，请输入值 -2。要从所有数据库分区捕获快照，不要对此参数指定值。

     注:

     a. 对于以下快照表函数列表，如果对当前连接的数据库输入 NULL，那么将获取实例中所有数据库的快照信息:
        - SNAP_GET_DB
        - SNAP_GET_DB_MEMORY_POOL
        - SNAP_GET_DETAILLOG
        - SNAP_GET_HADR
        - SNAP_GET_STORAGE_PATHS
        - SNAP_GET_APPL
        - SNAP_GET_APPL_INFO

- SNAP_GET_AGENT
- SNAP_GET_AGENT_MEMORY_POOL
- SNAP_GET_STMT
- SNAP_GET_SUBSECTION
- SNAP_GET_BP
- SNAP_GET_BP_PART

b. 数据库名称参数不适用于数据库管理器级别快照表函数；那些表函数只有数据库分区号参数。数据库分区号参数是可选的。

使用 SNAP_WRITE_FILE 存储过程将数据库系统快照信息捕获到文件中

通过使用 SNAP_WRITE_FILE 存储过程，可以捕获监视器数据快照并将此信息存储到数据库服务器上的文件中，并且可以允许未拥有 SYSADM、SYSCTRL、SYSMAINT 或 SYSMON 权限的用户访问该数据。于是，任何用户都可以发出带有快照表函数的查询以访问这些文件中的快照信息。如果开放对快照监视器数据的访问，所有对快照表函数具有执行特权的用户就能获得一些敏感信息，例如已连接用户列表以及他们对数据库提交的 SQL 语句等。缺省情况下，已将执行快照表函数的特权授予 PUBLIC。（但是，请注意，使用快照监视器表函数并不会泄漏表中的实际数据或用户密码。）

开始之前

您必须拥有 SYSADM、SYSCTRL、SYSMAINT 或 SYSMON 权限才能使用 SNAP_WRITE_FILE 存储过程来捕获数据库快照。

关于此任务

调用 SNAP_WRITE_FILE 存储过程时，除了标识所要监视的数据库和分区以外，还需要指定*快照请求类型*。每种快照请求类型都确定了所收集的监视器数据的范围。请根据用户需要运行的快照表函数来选择快照请求类型。下表列示了快照表函数及其对应的请求类型。

表 113. 快照请求类型

快照表函数	快照请求类型
SNAP_GET_AGENT	APPL_ALL
SNAP_GET_AGENT_MEMORY_POOL	APPL_ALL
SNAP_GET_APPL	APPL_ALL
SNAP_GET_APPL_INFO	APPL_ALL
SNAP_GET_STMT	APPL_ALL
SNAP_GET_SUBSECTION	APPL_ALL
SNAP_GET_BP_PART	BUFFERPOOLS_ALL
SNAP_GET_BP	BUFFERPOOLS_ALL
SNAP_GET_DB	DBASE_ALL
SNAP_GET_DETAILLOG	DBASE_ALL
SNAP_GET_DB_MEMORY_POOL	DBASE_ALL
SNAP_GET_HADR	DBASE_ALL

表 113. 快照请求类型 (续)

快照表函数	快照请求类型
SNAP_GET_STORAGE_PATHS	DBASE_ALL
SNAP_GET_DBM	DB2
SNAP_GET_DBM_MEMORY_POOL	DB2
SNAP_GET_FCM	DB2
SNAP_GET_FCM_PART	DB2
SNAP_GET_SWITCHES	DB2
SNAP_GET_DYN_SQL	DYNAMIC_SQL
SNAP_GET_LOCK	DBASE_LOCKS
SNAP_GET_LOCKWAIT	APPL_ALL
SNAP_GET_TAB	DBASE_TABLES
SNAP_GET_TAB_REORG	DBASE_TABLES
SNAP_GET_TBSP	DBASE_TABLESPACES
SNAP_GET_TBSP_PART	DBASE_TABLESPACES
SNAP_GET_CONTAINER	DBASE_TABLESPACES
SNAP_GET_TBSP_QUIESCER	DBASE_TABLESPACES
SNAP_GET_TBSP_RANGE	DBASE_TABLESPACES
SNAP_GET_UTIL	DB2
SNAP_GET_UTIL_PROGRESS	DB2

过程

1. 连接到数据库。这可以是实例中任何需要监视的数据库。为了能够调用存储过程，必须连接到数据库。
2. 确定快照请求类型以及需要监视的数据库和分区。
3. 调用 SNAP_WRITE_FILE 存储过程，对快照请求类型参数、数据库参数和分区参数指定适当的设置。例如，以下调用将捕获 SAMPLE 数据库的当前连接分区的应用程序信息快照:

   ```
   CALL SNAP_WRITE_FILE('APPL_ALL','SAMPLE',-1)
   ```

   SNAP_WRITE_FILE 存储过程有 3 个输入参数:

   - 快照请求类型（请参阅第 381 页的表 113，该表提供了快照表函数及其相应请求类型的交叉引用）
   - VARCHAR (128) 数据库名称。如果输入 NULL，就会使用当前连接的数据库的名称。

     注: 此参数不适用于数据库管理器级别快照表函数；那些表函数只有请求类型参数和分区号参数。
   - SMALLINT 分区号（0 到 999 之间的值）。对于分区号参数，输入与所要监视的分区号相对应的整数。要捕获当前连接的分区的快照，请输入值 -1 或 NULL。要捕获全局快照，请输入值 -2。

结果

在将快照数据保存到文件中之后，通过将 (NULL, NULL) 指定为数据库级别表函数的输入值并对数据库管理器级别表函数指定 (NULL)，所有用户都可以发出包含相应快照表函数的查询。他们接收到的监视器数据是从 SNAP_WRITE_FILE 存储过程生成的文件中得来的。

注: 虽然这种方法限制了用户对敏感监视器数据的访问，但这种方法有一些局限性:

- SNAP_WRITE_FILE 文件中的快照监视器数据仅仅是上次调用 SNAP_WRITE_FILE 存储过程时的最新数据。通过定期调用 SNAP_WRITE_FILE 存储过程，可以确保获得最新的快照监视器数据。例如，在 UNIX 系统上，可以设置 cron 作业来完成此操作。
- 发出包含快照表函数的查询的用户不能标识所要监视的数据库或分区。发出 SNAP_WRITE_FILE 调用的用户所标识的数据库名称和分区号确定了快照表函数可访问的文件内容。
- 如果用户发出包含快照表函数的 SQL 查询，但尚未针对该快照表函数运行相应的 SNAP_WRITE_FILE 请求类型，就会尝试对当前连接的数据库和分区创建直接快照。仅当用户具有 SYSADM、SYSCTRL、SYSMAINT 或 SYSMON 权限时，此操作才会成功。

使用 SQL 查询中的快照表函数来访问数据库系统快照（使用文件访问）

对于授权用户调用 SNAP_WRITE_FILE 存储过程的每个请求类型，任何用户都可使用相应快照表函数发出查询。他们接收到的监视器数据是从 SNAP_WRITE_FILE 存储过程生成的文件中检索到的。

开始之前

对于打算用于访问 SNAP_WRITE_FILE 文件的每个快照表函数，授权用户必须已经使用相应快照请求类型发出了 SNAP_WRITE_FILE 存储过程调用。如果发出包含快照表函数的 SQL 查询，但尚未针对该快照表函数运行相应的 SNAP_WRITE_FILE 请求类型，就会尝试对当前连接的数据库和分区创建直接快照。仅当用户具有 SYSADM、SYSCTRL、SYSMAINT 或 SYSMON 权限时，此操作才会成功。

关于此任务

使用快照表函数从 SNAP_WRITE_FILE 文件访问快照数据的用户不能标识要监视的数据库或分区。如果数据库名称和分区号是通过用户发出 SNAP_WRITE_FILE 调用标识的，那么该数据库名称和分区号将确定 SNAP_WRITE_FILE 文件的内容。SNAP_WRITE_FILE 文件中可用的快照监视器数据仅仅是上次 SNAP_WRITE_FILE 存储过程捕获快照时的最新数据。

过程

1. 连接到数据库。这可以是实例中任何需要监视的数据库。要使用快照表函数发出 SQL 查询，必须连接到数据库。
2. 确定需要捕获的快照的类型。
3. 使用适当的快照表函数发出查询。例如，以下查询将捕获表空间信息的快照。

```
SELECT * FROM TABLE(SNAP_GET_TBSP (CAST(NULL AS VARCHAR(1)),
                    CAST (NULL AS INTEGER))) AS SNAP_GET_TBSP
```

注：必须对数据库名称和分区号参数输入空值。快照的数据库名称和分区将在 SNAP_WRITE_FILE 存储过程的调用中确定。而且，数据库名称参数不适用于数据库管理器级别快照表函数；那些表函数只有分区号参数。

每个快照表函数返回带有一行或多行的一个表，每列表示一个监视元素。相应的，监视元素列名与监视元素名称相关。

4. 您也可以从返回的表中选择个别监视元素。例如，以下语句仅返回 **agent_id** 监视元素：

```
SELECT agent_id FROM TABLE(
                          SNAP_GET_APPL(CAST(NULL AS VARCHAR(1)),
                                        CAST (NULL AS INTEGER)))
                          as SNAP_GET_APPL
```

快照监视器 SQL 管理视图

提供了许多不同的快照监视器 SQL 管理视图，每个管理视图都返回有关特定数据库系统区域的监视器数据。例如，SYSIBMADM.SNAPBP SQL 管理视图捕获快照或缓冲池信息。下表列示了每个可用的快照监视器管理视图。

表 114. 快照监视器 SQL 管理视图

监视器级别	SQL 管理视图	返回的信息
数据库管理器	SYSIBMADM.SNAPDBM	数据库管理器级别信息。
数据库管理器	SYSIBMADM.SNAPFCM	关于快速通信管理器 (FCM) 的数据库管理器级别信息。
数据库管理器	SYSIBMADM.SNAPFCM_PART	某个分区的关于快速通信管理器 (FCM) 的数据库管理器级别信息。
数据库管理器	SYSIBMADM.SNAPSWITCHES	数据库管理器监视开关设置。
数据库管理器	SYSIBMADM.SNAPDBM_MEMORY_POOL	有关内存使用情况的数据库管理器级别信息。
数据库	SYSIBMADM.SNAPDB	数据库的数据库级别信息和计数器。仅当至少有一个应用程序已连接到数据库时，才会返回此信息。
数据库	SYSIBMADM.SNAPDB_MEMORY_POOL	仅与 UNIX 平台的内存使用情况有关的数据库级别信息。
应用程序	SYSIBMADM.SNAPAPPL	有关连接至数据库的每个应用程序的常规应用程序级别信息。此信息包括累积计数器、状态信息和最近执行的 SQL 语句（如果设置了语句开关）。
应用程序	SYSIBMADM.SNAPAPPL_INFO	有关连接至数据库的每个应用程序的常规应用程序级别标识信息。
应用程序	SYSIBMADM.SNAPLOCKWAIT	有关连接至数据库的应用程序的锁定等待数的应用程序级别信息。
应用程序	SYSIBMADM.SNAPSTMT	有关连接至数据库的应用程序的语句的应用程序级别信息。此信息包括最近执行的 SQL 语句（如果设置了语句开关）。
应用程序	SYSIBMADM.SNAPAGENT	有关连接至数据库的应用程序的关联代理程序的应用程序级别信息。

表 114. 快照监视器 SQL 管理视图 (续)

监视器级别	SQL 管理视图	返回的信息
应用程序	SYSIBMADM.SNAPSUBSECTION	有关连接至数据库的应用程序的存取方案子节的应用程序级别信息。
应用程序	SYSIBMADM.SNAPAGENT_MEMORY_POOL	有关代理程序级别的内存使用情况的信息。
表	SYSIBMADM.SNAPTAB	每个与数据库相连接的应用程序的数据库级别和应用程序级别表活动信息。与数据库相连接的应用程序*已访问的*每个表在表级别的活动信息。需要表开关。
表	SYSIBMADM.SNAPTAB_REORG	正在重组的数据库中每个表在表级别的重组信息。
锁定	SYSIBMADM.SNAPLOCK	每个与数据库相连接的应用程序的数据库级别和应用程序级别锁定信息。需要锁定开关。
表空间	SYSIBMADM.SNAPTBSP	有关以下级别的表空间活动信息：数据库级别、与数据库相连接的每个应用程序的应用程序级别以及与数据库相连接的应用程序已访问的每个表空间的表空间级别。需要缓冲池开关。
表空间	SYSIBMADM.SNAPTBSP_PART	表空间配置信息。
表空间	SYSIBMADM.SNAPTBSP_QUIESCER	表空间级别停顿者信息。
表空间	SYSIBMADM.SNAPCONTAINER	表空间级别表空间容器配置信息。
表空间	SYSIBMADM.SNAPTBSP_RANGE	表空间映射的范围信息。
缓冲池	SYSIBMADM.SNAPBP	指定数据库的缓冲池活动计数器。需要缓冲池开关。
缓冲池	SYSIBMADM.SNAPBP_PART	有关针对每个分区计算出来的缓冲区大小和使用情况的信息。
动态 SQL	SYSIBMADM.SNAPDYN_SQL	数据库的 SQL 语句高速缓存中的时间点语句信息。
数据库	SYSIBMADM.SNAPUTIL	关于实用程序的信息。
数据库	SYSIBMADM.SNAPUTIL_PROGRESS	关于实用程序进度的信息。
数据库	SYSIBMADM.SNAPDETAILLOG	有关日志文件的数据库级别信息。
数据库	SYSPROC.ADMIN_GET_STORAGE_PATHS	返回数据库的自动存储器路径列表，该列表包括每个存储器路径的文件系统信息。

在捕获快照前，请考虑是否需要由监视开关控制的监视元素提供的信息。如果特定监视开关处于关闭状态，就不会收集它控制的监视元素。请参阅各个监视元素以确定所需的元素是否在开关控制之下。

所有快照监视管理视图和关联表函数都使用单独的实例连接，该连接与当前会话使用的连接不同。因此，可能会建立隐式实例附件，并且只有缺省数据库管理器监视开关才有效。不起作用的监视开关包括任何在当前会话或应用程序中动态打开或关闭的开关。

DB2 V9.5 还为您提供了一组管理视图，它们不仅返回个别监视元素的值，而且还返回监视任务中通常需要的计算值。例如，SYSIBMADM.BP_HITRATIO 管理视图返回缓冲池命中率计算值，此值由各个监视元素的值累计得出的。

表 115. 快照监视器 SQL 管理公用视图

SQL 管理公用视图	返回的信息
SYSIBMADM.APPLICATIONS	关于已连接的数据库应用程序的信息。
SYSIBMADM.APPL_PERFORMANCE	有关选择的行数与应用程序已读取的行数的比率的信息。
SYSIBMADM.BP_HITRATIO	数据库中的缓冲池命中率，包括命中率总计、数据命中率和索引命中率。
SYSIBMADM.BP_READ_IO	关于缓冲池读性能的信息。
SYSIBMADM.BP_WRITE_IO	关于缓冲池写性能的信息。
SYSIBMADM.CONTAINER_UTILIZATION	关于表空间容器和利用率的信息。
SYSIBMADM.LOCKS_HELD	关于当前持有的锁定的信息。
SYSIBMADM.LOCKWAITS	有关代表等待获取锁定的应用程序工作的 DB2 代理程序的信息。
SYSIBMADM.LOG_UTILIZATION	有关当前连接的数据库的日志利用率的信息。
SYSIBMADM.LONG_RUNNING_SQL	有关在当前连接的数据库中花费最长时间运行的 SQL 的信息。
SYSIBMADM.QUERY_PREP_COST	关于准备不同 SQL 语句所需时间的信息。
SYSIBMADM.TBSP_UTILIZATION	表空间配置和利用率信息。
SYSIBMADM.TOP_DYNAMIC_SQL	可按执行次数、平均执行时间、排序数或每个语句的排序数进行排序的顶级动态 SQL 语句数。

对数据库系统快照的 SQL 访问

有两种方法可用来使用快照监视器 SQL 表函数（又称为*快照表函数*）访问快照监视器数据：直接访问和文件访问。

关于此任务

直接访问

授权用户可使用快照表函数发出查询并接收包含监视数据的结果集。如果使用此方法，那么只有具有 SYSADM、SYSCTRL、SYSMAINT 或 SYSMON 权限的用户才能访问快照监视器数据。

要使用直接访问来捕获快照信息:

1. 可选: 设置并检查监视开关的状态。请参阅第 404 页的『通过 CLP 设置系统监视开关』。
2. 使用 SQL 捕获数据库系统快照。请参阅第 379 页的『使用快照管理视图和表函数来捕获数据库系统快照』。

文件访问

授权用户调用 SNAPSHOT_FILEW 存储过程（标识快照请求类型）及受影响的分区和数据库。然后 SNAPSHOT_FILEW 存储过程会将监视器数据存储在数据库服务器上的文件中。

授权用户可对其调用 SNAPSHOT_FILEW 存储过程的每个请求类型

虽然这是允许所有用户访问快照监视器数据的安全方法，但此方法仍然有一些局限性:

- SNAPSHOT_FILEW 文件中的快照监视器数据仅仅是上次调用 SNAPSHOT_FILEW 存储过程时的最新数据。通过定期调用 SNAPSHOT_FILEW 存储过程，可以确保获得最新的快照监视器数据。例如，在 UNIX 操作系统上，可以设置 cron 作业来完成此操作。

- 发出包含快照表函数的查询的用户不能标识所要监视的数据库或分区。发出 SNAPSHOT_FILEW 调用的用户所标识的数据库名称和分区号确定了快照表函数可访问的文件内容。
- 如果用户发出包含快照表函数的 SQL 查询，但尚未针对该快照表函数运行相应的 SNAPSHOT_FILEW 请求类型，就会尝试对当前连接的数据库和分区创建直接快照。仅当用户具有 SYSADM、SYSCTRL、SYSMAINT 或 SYSMON 权限时，此操作才会成功。

捕获数据库系统快照信息并保存至文件的 SYSADM、SYSCTRL、SYSMAINT 或 SYSMON 用户将执行下列任务。

过程

1. 了解将发出快照请求的用户的需要。 具体而言，确定他们需要的监视器数据、要从中进行收集的数据库以及收集是否限制为特定分区。
2. 可选: 设置并检查监视开关的状态。 请参阅第 404 页的『通过 CLP 设置系统监视开关』。
3. 将数据库系统快照信息捕获到文件中。 请参阅第 381 页的『使用 SNAP_WRITE_FILE 存储过程将数据库系统快照信息捕获到文件中』。

下一步做什么

一旦 SYSADM、SYSCTRL、SYSMAINT 或 SYSMON 用户完成上述步骤，所有用户都可以使用 SQL 查询中的快照表函数来访问数据库系统快照信息。

从 CLP 捕获数据库快照

可使用 GET SNAPSHOT 命令从 CLP 捕获数据库快照。有若干不同快照请求类型可用，可通过对 GET SNAPSHOT 命令指定特定参数来访问它们。

开始之前

必须具有 SYSADM、SYSCTRL、SYSMAINT 或 SYSMON 权限才能捕获数据库快照。

必须具有实例连接才能捕获数据库快照。如果没有实例连接，那么创建缺省实例连接。要获取远程实例的快照，必须先连接至该实例。

过程

- 可选: 设置并检查监视开关的状态。
- 从 CLP 发出带有必需参数的 `GET SNAPSHOT` 命令。在以下示例中，快照会捕获所有数据库的信息:

  ```
  db2 get snapshot for all databases
  ```

  要捕获特定数据库的数据库快照，请使用以下命令:

  ```
  db2 get snapsot for database on db-name
  ```

  其中 *db-name* 是您所关心的数据库的名称。
- 以下示例捕获数据库管理器级别信息:

  ```
  db2 get snapshot for dbm
  ```

- 对于分区数据库系统，可为特定分区捕获专门的数据库快照，或者为所有分区捕获全局的数据库快照。要对特定分区（如分区号 2）上的所有应用程序捕获数据库快照，请发出以下命令:

  ```
  db2 get snapshot for all applications at dbpartitionnum 2
  ```

- 要对所有分区上的所有应用程序捕获数据库快照，请发出以下命令:

  ```
  db2 get snapshot for all applications global
  ```

  对于分区数据库上的全局快照，将聚集所有分区中的监视器数据。

快照监视器 CLP 命令

下表列示了所有受支持的快照请求类型。对于特定请求类型，仅当相关联的监视开关设置为 ON 时，才返回某些信息。请参阅各个监视元素以确定所需的元素是否在开关控制之下。

表 116. 快照监视器 CLP 命令

监视器级别	CLP 命令	返回的信息
连接列表	`list applications [show detail]`	当前连接至数据库的所有应用程序的标识信息，该数据库由在其上获取快照的分区上的 DB2 实例管理。
连接列表	`list applications for database` *dbname* `[show detail]`	当前连接至指定数据库的每个应用程序的标识信息。
连接列表	`list dcs applications`	当前连接至数据库的所有 DCS 应用程序的标识信息，该数据库由在其上获取快照的分区上的 DB2 实例管理。
数据库管理器	`get snapshot for dbm`	数据库管理器级别信息，包括实例级别监视开关设置。
数据库管理器	`get dbm monitor switches`	实例级别监视开关设置。
数据库	`get snapshot for database on` *dbname*	数据库的数据库级别信息和计数器。仅当至少有一个应用程序已连接到数据库时，才会返回此信息。
数据库	`get snapshot for all databases`	在分区上处于活动状态的每个数据库的数据库级别信息和计数器。仅当至少有一个应用程序已连接到数据库时，才会返回此信息。
数据库	`list active databases`	与每个活动数据库的连接的数目。包括使用 ACTIVATE DATABASE 命令启动但没有连接的数据库。
数据库	`get snapshot for dcs database on` *dbname*	特定 DCS 数据库的数据库级别信息和计数器。仅当至少有一个应用程序已连接到数据库时，才会返回此信息。
数据库	`get snapshot for remote database on` *dbname*	特定联合系统数据库的数据库级别信息和计数器。仅当至少有一个应用程序已连接到数据库时，才会返回此信息。
数据库	`get snapshot for all remote databases`	分区上的每个活动联合系统数据库的数据库级别信息和计数器。仅当至少有一个应用程序已连接到数据库时，才会返回此信息。
应用程序	`get snapshot for application applid` *appl-id*	应用程序级别信息，包括累积计数器、状态信息和最近执行的 SQL 语句（如果设置了语句开关）。
应用程序	`get snapshot for application agentid` *appl-handle*	应用程序级别信息，包括累积计数器、状态信息和最近执行的 SQL 语句（如果设置了语句开关）。
应用程序	`get snapshot for applications on` *dbname*	与分区中数据库相连接的每个应用程序的应用程序级别信息。此信息包括累积计数器、状态信息和最近执行的 SQL 语句（如果设置了语句开关）。

表 116. 快照监视器 CLP 命令 (续)

监视器级别	CLP 命令	返回的信息
应用程序	`get snapshot for all applications`	在分区中处于活动状态的每个应用程序的应用程序级别信息。此信息包括累积计数器、状态信息和最近执行的 SQL 语句（如果设置了语句开关）。
应用程序	`get snapshot for dcs application applid` *appl-id*	应用程序级别信息，包括累积计数器、状态信息和最近执行的 SQL 语句（如果设置了语句开关）。
应用程序	`get snapshot for all dcs applications`	在分区中处于活动状态的每个 DCS 应用程序的应用程序级别信息。此信息包括累积计数器、状态信息和最近执行的 SQL 语句（如果设置了语句开关）。
应用程序	`get snapshot for dcs application agentid` *appl-handle*	应用程序级别信息，包括累积计数器、状态信息和最近执行的 SQL 语句（如果设置了语句开关）。
应用程序	`get snapshot for dcs applications on` *dbname*	与分区中数据库相连接的每个 DCS 应用程序的应用程序级别信息。此信息包括累积计数器、状态信息和最近执行的 SQL 语句（如果设置了语句开关）。
应用程序	`get snapshot for remote applications on` *dbname*	应用程序级别信息，包括累积计数器、状态信息和最近执行的 SQL 语句（如果设置了语句开关）。
应用程序	`get snapshot for all remote applications`	在分区中处于活动状态的每个联合系统应用程序的应用程序级别信息。此信息包括累积计数器、状态信息和最近执行的 SQL 语句（如果设置了语句开关）。
表	`get snapshot for tables on` *dbname*	每个与数据库相连接的应用程序的数据库级别和应用程序级别表活动信息。与数据库相连接的应用程序**已访问的**每个表在表级别的活动信息。需要表开关。
锁定	`get snapshot for locks for application applid` *appl-id*	应用程序挂起的锁定列表。锁定等待信息需要锁定开关。
锁定	`get snapshot for locks for application agentid` *appl-handle*	应用程序挂起的锁定列表。锁定等待信息需要锁定开关。
锁定	`get snapshot for locks on` *dbname*	每个与数据库相连接的应用程序的数据库级别和应用程序级别锁定信息。需要锁定开关。
表空间	`get snapshot for tablespaces on` *dbname*	有关数据库的表空间活动的信息。需要缓冲池开关。还包括有关容器、停顿者和范围的信息。此信息不受开关控制。
缓冲池	`get snapshot for all bufferpools`	缓冲池活动计数器。需要缓冲池开关。
缓冲池	`get snapshot for bufferpools on` *dbname*	指定数据库的缓冲池活动计数器。需要缓冲池开关。
动态 SQL	`get snapshot for dynamic sql on` *dbname*	数据库的 SQL 语句高速缓存中的时间点语句信息。该信息也可能来自远程数据源。

从客户机应用程序捕获数据库快照

可使用 C、C++ 或 COBOL 应用程序中的快照监视器 API 来捕获数据库快照。在 C 和 C++ 中，可通过在 db2GetSnapshot() 参数中指定特定参数来访问若干不同快照请求类型。

开始之前

必须具有 SYSADM、SYSCTRL、SYSMAINT 或 SYSMON 权限才能使用 db2MonitorSwitches API。

必须具有实例连接才能捕获数据库快照。如果没有实例连接，那么创建缺省实例连接。要获取远程实例的快照，必须先连接至该实例。

过程

1. 可选: 设置并检查监视开关的状态。
2. 包括下列 DB2 库: sqlmon.h 和 db2ApiDf.h。可在 sqllib 中的 include 子目录下找到它们。

   ```
   #include <db2ApiDf.h>
   #include <sqlmon.h>
   ```

3. 将快照缓冲区单元大小设置为 100 KB。

   ```
   #define SNAPSHOT_BUFFER_UNIT_SZ 102400
   ```

4. 声明 sqlca、sqlma、db2GetSnapshotData 和 sqlm_collected 结构。还应初始化指针以包含快照缓冲区并确定缓冲区大小。

   ```
   struct sqlma *pRequestedDataGroups;
   struct sqlca sqlca;
   memset (&sqlca, '\0', sizeof(struct sqlca));
   struct sqlm_collected collectedData;
   memset (&collectedData, '\0', sizeof(collectedData));
   db2GetSnapshotData getSnapshotParam;
   memset (&getSnapshotParam, '\0', sizeof(getSnapshotParam));

   static sqluint32 snapshotBufferSize = SNAPSHOT_BUFFER_UNIT_SZ;
   sqluint32 outputFormat;
   char *snapshotBuffer;
   ```

5. 初始化 sqlma 结构，并指定要捕获的快照属于数据库管理器级别信息。

   ```
   pRequestedDataGroups = (struct sqlma *)malloc(SQLMASIZE(1));
   memset(pRequestedDataGroups, '\0', SQLMASIZE(1));
   pRequestedDataGroups->obj_num = 1;
   pRequestedDataGroups->obj_var[0].obj_type = SQLMA_DB2;
   ```

6. 初始化用来容纳快照输出的缓冲区。

   ```
     snapshotBuffer = (char *)malloc(snapshotBufferSize);
   memset (snapshotBuffer, '\0', snapshotBufferSize);
   ```

7. 使用快照请求类型（来自 sqlma 结构）、缓冲区信息和捕获快照所需的其他信息来填充 db2GetSnapshotData 结构。

   ```
   getSnapshotParam.piSqlmaData = pRequestedDataGroups;
   getSnapshotParam.poCollectedData = &collectedData;
     getSnapshotParam.poBuffer = snapshotBuffer;
   getSnapshotParam.iVersion = SQLM_DBMON_VERSION9_5;
     getSnapshotParam.iBufferSize = snapshotBufferSize;
   getSnapshotParam.iStoreResult = 0;
   getSnapshotParam.iNodeNumber = SQLM_CURRENT_NODE;
   getSnapshotParam.poOutputFormat = &outputFormat;
   getSnapshotParam.iSnapshotClass = SQLM_CLASS_DEFAULT;
   ```

8. 捕获快照。传递 db2GetSnapshotData 结构和对缓冲区的引用，db2GetSnapshotData 结构包含捕获快照所需的信息，快照输出将引导至该缓冲区。

   ```
   db2GetSnapshot(db2Version810, &getSnapshotParam, &sqlca);
   ```

9. 包括用于处理缓冲区溢出的逻辑。获取快照后，将检查 sqlcode 是否存在缓冲区溢出。如果发生了缓冲区溢出，那么将清除并重新初始化缓冲区，然后再次获取快照。

```
while (sqlca.sqlcode == 1606)
{
  free(snapshotBuffer);
    snapshotBufferSize = snapshotBufferSize +
    SNAPSHOT_BUFFER_UNIT_SZ;
    snapshotBuffer = (char *)malloc(snapshotBufferSize);
    if (snapshotBuffer == NULL)
  {
        printf("\nMemory allocation error.\n");
        return 1;
  }
    getSnapshotParam.iBufferSize = snapshotBufferSize;
    getSnapshotParam.poBuffer = snapshotBuffer;
    db2GetSnapshot(db2Version810, &getSnapshotParam, &sqlca);
}
```

10. 处理快照监视器数据流。
11. 清除缓冲区。

```
free(snapshotBuffer);
free(pRequestedDataGroups);
```

快照监视器 API 请求类型

下表列示了所有受支持的快照请求类型。对于特定请求类型，仅当相关联的监视开关设置为 ON 时，才返回某些信息。请参阅各个监视元素以确定所需的元素是否在开关控制之下。

表 117. 快照监视器 API 请求类型

监视器级别	API 请求类型	返回的信息
连接列表	SQLMA_APPLINFO_ALL	当前连接至数据库的所有应用程序的标识信息，该数据库由在其上获取快照的分区上的 DB2 实例管理。
连接列表	SQLMA_DBASE_APPLINFO	当前连接至指定数据库的每个应用程序的标识信息。
连接列表	SQLMA_DCS_APPLINFO_ALL	当前连接至数据库的所有 DCS 应用程序的标识信息，该数据库由在其上获取快照的分区上的 DB2 实例管理。
数据库管理器	SQLMA_DB2	数据库管理器级别信息，包括实例级别监视开关设置。
数据库	SQLMA_DBASE	数据库的数据库级别信息和计数器。仅当至少有一个应用程序已连接到数据库时，才会返回此信息。
数据库	SQLMA_DBASE_ALL	在分区上处于活动状态的每个数据库的数据库级别信息和计数器。与每个活动数据库的连接的数目。包括使用 ACTIVATE DATABASE 命令启动但没有连接的数据库。仅当至少有一个应用程序已连接到数据库时，才会返回此信息。
数据库	SQLMA_DCS_DBASE	特定 DCS 数据库的数据库级别信息和计数器。仅当至少有一个应用程序已连接到数据库时，才会返回此信息。

表 117. 快照监视器 API 请求类型 (续)

监视器级别	API 请求类型	返回的信息
数据库	SQLMA_DCS_DBASE_ALL	在分区上处于活动状态的每个 DCS 数据库的数据库级别信息和计数器。仅当至少有一个应用程序已连接到数据库时，才会返回此信息。
数据库	SQLMA_DBASE_REMOTE	特定联合系统数据库的数据库级别信息和计数器。仅当至少有一个应用程序已连接到数据库时，才会返回此信息。
数据库	SQLMA_DBASE_REMOTE_ALL	分区上的每个活动联合系统数据库的数据库级别信息和计数器。仅当至少有一个应用程序已连接到数据库时，才会返回此信息。
应用程序	SQLMA_APPL	应用程序级别信息，包括累积计数器、状态信息和最近执行的 SQL 语句（如果设置了语句开关）。
应用程序	SQLMA_AGENT_ID	应用程序级别信息，包括累积计数器、状态信息和最近执行的 SQL 语句（如果设置了语句开关）。
应用程序	SQLMA_DBASE_APPLS	与分区中数据库相连接的每个应用程序的应用程序级别信息。此信息包括累积计数器、状态信息和最近执行的 SQL 语句（如果设置了语句开关）。
应用程序	SQLMA_APPL_ALL	在分区中处于活动状态的每个应用程序的应用程序级别信息。此信息包括累积计数器、状态信息和最近执行的 SQL 语句（如果设置了语句开关）。
应用程序	SQLMA_DCS_APPL	应用程序级别信息，包括累积计数器、状态信息和最近执行的 SQL 语句（如果设置了语句开关）。
应用程序	SQLMA_DCS_APPL_ALL	在分区中处于活动状态的每个 DCS 应用程序的应用程序级别信息。此信息包括累积计数器、状态信息和最近执行的 SQL 语句（如果设置了语句开关）。
应用程序	SQLMA_DCS_APPL_HANDLE	应用程序级别信息，包括累积计数器、状态信息和最近执行的 SQL 语句（如果设置了语句开关）。
应用程序	SQLMA_DCS_DBASE_APPLS	与分区中数据库相连接的每个 DCS 应用程序的应用程序级别信息。此信息包括累积计数器、状态信息和最近执行的 SQL 语句（如果设置了语句开关）。
应用程序	SQLMA_DBASE_APPLS_REMOTE	应用程序级别信息，包括累积计数器、状态信息和最近执行的 SQL 语句（如果设置了语句开关）。
应用程序	SQLMA_APPL_REMOTE_ALL	在分区中处于活动状态的每个联合系统应用程序的应用程序级别信息。此信息包括累积计数器、状态信息和最近执行的 SQL 语句（如果设置了语句开关）。
表	SQLMA_DBASE_TABLES	每个与数据库相连接的应用程序的数据库级别和应用程序级别表活动信息。与数据库相连接的应用程序**已访问的**每个表在表级别的活动信息。需要表开关。
锁定	SQLMA_APPL_LOCKS	应用程序挂起的锁定列表。锁定等待信息需要锁定开关。
锁定	SQLMA_APPL_LOCKS_AGENT_ID	应用程序挂起的锁定列表。锁定等待信息需要锁定开关。
锁定	SQLMA_DBASE_LOCKS	每个与数据库相连接的应用程序的数据库级别和应用程序级别锁定信息。需要锁定开关。

表 117. 快照监视器 API 请求类型 (续)

监视器级别	API 请求类型	返回的信息
表空间	SQLMA_DBASE_TABLESPACES	有关以下级别的表空间活动信息：数据库级别、与数据库相连接的每个应用程序的应用程序级别以及与数据库相连接的应用程序已访问的每个表空间的表空间级别。需要缓冲池开关。
缓冲池	SQLMA_BUFFERPOOLS_ALL	缓冲池活动计数器。需要缓冲池开关。
缓冲池	SQLMA_DBASE_BUFFERPOOLS	指定数据库的缓冲池活动计数器。需要缓冲池开关。
动态 SQL	SQLMA_DYNAMIC_SQL	数据库的 SQL 语句高速缓存中的时间点语句信息。

快照监视器样本输出

为说明快照监视器的特征，下面提供了使用 CLP 获取快照的示例及相应输出。此示例的目的在于获取连接至 SAMPLE 数据库的应用程序挂起的锁定列表。采取的步骤如下所示:

1. 连接至样本数据库:

   ```
   db2 connect to sample
   ```

2. 使用 UPDATE MONITOR SWITCHES 命令将 LOCK 开关设置为 ON，这样就会收集等待锁定所花的时间:

   ```
   db2 update monitor switches using LOCK on
   ```

3. 发出需要针对数据库目录的锁定的命令或语句。在此情况下，我们将声明、打开和访存游标:

   ```
   db2 -c- declare c1 cursor for
                          select * from staff where job='Sales' for update
   db2 -c- open c1
   db2 -c- fetch c1
   ```

4. 使用 GET SNAPSHOT 命令获取数据库锁定快照:

   ```
   db2 get snapshot for locks on sample
   ```

在从 CLP 发出 GET SNAPSHOT 命令后，快照输出将引导至屏幕。

```
                        数据库锁定快照

数据库名称                                  = SAMPLE
数据库路径                                  = C:\DB2\NODE0000\SQL00001\
输入数据库别名                              = SAMPLE
挂起的锁定数                                    = 5
当前连接的应用程序数                        = 1
当前正在等待锁定的代理程序数                = 0
快照时间戳记                                = 06-05-2002 17:08:25.048027

应用程序句柄                                = 8
应用程序标识                                = *LOCAL.DB2.0098C5210749
序号                                        = 0001
应用程序名称                                = db2bp.exe
连接授权标识                                = DB2ADMIN
应用程序状态                                = UOW 正在等待
状态更改时间                                = 未收集
应用程序代码页                              = 1252
挂起的锁定数                                    = 5
总等待时间（ms）                            = 0

锁定列表
```

```
锁定名称                    = 0x020003000500000000000000052
锁定属性                    = 0x00000000
释放标志                    = 0x00000001
锁定计数                    = 1
挂起计数                    = 0
锁定对象名                  = 5
对象类型                    = 行
表空间名称                  = USERSPACE1
表模式                      = DB2ADMIN
表名                        = STAFF
方式                        = U

锁定名称                    = 0x020003000000000000000000054
锁定属性                    = 0x00000000
释放标志                    = 0x00000001
锁定计数                    = 1
挂起计数                    = 0
锁定对象名                  = 3
对象类型                    = 表
表空间名称                  = USERSPACE1
表模式                      = DB2ADMIN
表名                        = STAFF
方式                        = IX

锁定名称                    = 0x010000000100000000100810056
锁定属性                    = 0x00000000
释放标志                    = 0x40000000
锁定计数                    = 1
挂起计数                    = 0
锁定对象名                  = 0
对象类型                    = 内部偏差锁定
方式                        = S

锁定名称                    = 0x41414141414A48520000000041
锁定属性                    = 0x00000000
释放标志                    = 0x40000000
锁定计数                    = 1
挂起计数                    = 0
锁定对象名                  = 0
对象类型                    = 内部计划锁定
方式                        = S

锁定名称                    = 0x434F4E544F4B4E310000000041
锁定属性                    = 0x00000000
释放标志                    = 0x40000000
锁定计数                    = 1
挂起计数                    = 0
锁定对象名                  = 0
对象类型                    = 内部计划锁定
方式                        = S
```

可从此快照中看到，当前有一个应用程序连接至 SAMPLE 数据库，并且该应用程序挂起 5 个锁定。

```
挂起的锁定数                                = 5
    当前连接的应用程序数                        = 1
```

注意，应用程序状态成为 UOW 正在锁定的时间（状态更改时间）返回为未收集。这是因为 UOW 开关为 OFF。

锁定快照还将返回直到目前为止连接至此数据库的应用程序等待锁定所花的总时间。

```
总等待时间（ms）                    = 0
```

子节快照

在使用分区间并行性的系统上，SQL 编译器将 SQL 语句的访问方案分为若干子节。每个子节由一个不同的 DB2 代理程序（或 SMP 的代理程序）执行。

通过使用 db2expln 命令，可获取 DB2 代码生成器在编译期间生成的 SQL 语句的存取方案。例如，选择分布在若干分区上的表中的所有行可能导致存取方案有两个子节:

1. 子节 0，协调程序子节，作用是收集其他 DB2 代理程序（子代理程序）访存的行并将其返回至应用程序。
2. 子节 1，作用是执行表扫描并将行返回至协调代理程序。

在此简单示例中，子节 1 将分布在所有数据库分区上。数据库分区组的每个物理分区上都有执行此子节的子代理程序，此表属于该数据库分区组。

数据库系统监视器允许您将运行时信息与存取方案相关联，这是编译时信息。借助分区间并行性，监视器会中断信息并下行至子节级别。例如，当语句监视开关设置为 ON 时，GET SNAPSHOT FOR APPLICATION 将返回在此分区上执行的每个子节的信息以及语句总数。

对应用程序快照返回的子节信息包括:

- 读取/写入的表行数
- CPU 消耗
- 耗用时间
- 发送至其他代理程序和从其他代理程序接收的表队列行数，这些代理程序处理此语句。这允许您通过获取一系列快照来跟踪长时间运行的查询的执行情况。
- 子节状态。如果子节因为等待另一代理程序发送或接收数据而处于 WAIT 状态，那么该信息还会标识阻止子节继续执行的分区。然后可获取这些分区的快照来调查情况。

在每个子节执行完之后，语句事件监视器记录的有关该子节的信息包括：CPU 消耗、总执行时间和若干其他计数器。

分区数据库系统上的全局快照

在分区数据库系统上，可以使用快照监视器来获取当前分区、指定分区或所有分区的快照。对分区数据库的所有分区获取全局快照时，会先聚集数据，然后返回结果。

对不同元素类型聚集数据的方式如下所示:

- **计数器、时间和标尺**

  包含从实例中的每个分区收集的所有可能值的总和。例如，GET SNAPSHOT FOR DATABASE XYZ ON TEST GLOBAL 对分区数据库实例中的所有分区返回从数据库读取的行数（rows_read）。

- **水位标记**

  返回分区数据库系统中的任何分区的最高（高水位）或最低（低水位）值。如果返回的值值得关注，那么可以获取各个分区的快照以确定特定分区是否使用过度或者问题是否为实例范围内的问题。

- **时间戳记**

设置为连接快照监视器实例代理程序的分区的时间戳记值。注意，所有时间戳记值都在 timestamp 监视开关控制之下。

- **信息**

  返回可能妨碍工作的分区的最重要信息。例如，对于元素 appl_status，如果一个分区上的状态为“正在执行 UOW”，而另一个分区上的状态为“等待锁定”，那么返回“等待锁定”，原因是这是挂起应用程序的执行的状态。

您也可以重置计数器，设置监视开关，以及检索分区数据库中的个别分区或所有分区的监视开关设置。

注: 获取全局快照时，如果一个或多个分区遇到错误，那么将从成功获取快照的分区收集数据，同时返回一个警告（sqlcode 1629）。如果以全局方式获取或更新监视开关，或者计数器在一个或多个分区上重置失败，那么不会设置这些分区的监视开关或进行数据重置。

快照监视器自描述数据流

在使用 db2GetSnapshot API 捕获快照之后，API 以自描述数据流的形式返回快照输出。图 6 显示数据流的结构，而 第 397 页的表 118 提供可能返回的逻辑数据组和监视元素的一些示例。

注: 描述性名称用于示例和表中的标识。在实际数据流中，将在这些名称前加上 **SQLM_ELM_** 前缀。例如，COLLECTED 在快照监视器输出中将显示为 SQLM_ELM_COLLECTED。在实际数据流中，将在类型前加上 **SQLM_TYPE_** 前缀。例如，HEADER 在数据流中将显示为 SQLM_TYPE_HEADER。

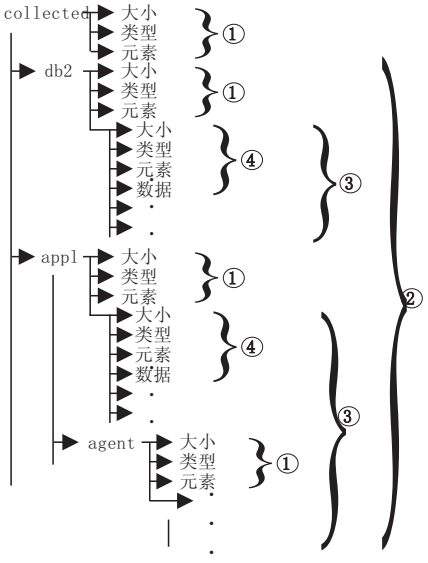


图 6. 快照监视器数据流

1. 每个逻辑数据组以指示其大小和名称的头开始。此大小不包括头本身占用的数据量。
2. 已收集头中的大小返回快照的总大小。
3. 其他头中的大小元素指示该逻辑数据组中的所有数据的大小，包括所有下级分组。

4. 监视元素信息在逻辑数据组头后面，并且也是自描述格式。

表 118. 样本快照数据流

逻辑数据组	数据流	描述
collected		
	1000	快照数据的大小（以字节计）。
	header	指示逻辑数据组的开头。
	collected	逻辑数据组的名称。
	4	此监视元素中存储的数据的大小。
	u32bit	监视元素类型 - 不带符号的 32 位数字。
	server_db2_type	收集的监视元素的名称。
	sqlf_nt_server	此元素的已收集值。
	2	此监视元素中存储的数据的大小。
	u16bit	监视元素类型 - 不带符号的 16 位数字。
	node_number	收集的监视元素的名称。
	3	此元素的已收集值。
db2		
	200	快照中 DB2 级别数据部分的大小。
	header	指示逻辑数据组的开头。
	db2	逻辑数据组的名称。
	4	此监视元素中存储的数据的大小。
	u32bit	监视元素类型 - 不带符号的 32 位数字。
	sort_heap_allocated	收集的监视元素的名称。
	16	此元素的已收集值。
	4	此监视元素中存储的数据的大小。
	u32bit	监视元素类型 - 不带符号的 32 位数字。
	local_cons	收集的监视元素的名称。
	3	此元素的已收集值。
	. . .	. . .
appl		
	100	快照中的应用程序元素数据的大小。
	header	指示逻辑数据组的开头。
	appl	逻辑数据组的名称。
	4	此监视元素中存储的数据的大小。
	u32bit	监视元素类型 - 不带符号的 32 位数字。
	locks_held	收集的监视元素的名称。
	3	此元素的已收集值。
	. . .	. . .

表 118. 样本快照数据流 (续)

逻辑数据组	数据流	描述
agent		
	50	应用程序结构的代理程序部分的大小。
	header	指示逻辑数据组的开头。
	agent	逻辑数据组的名称。
	4	此监视元素中存储的数据的大小。
	u32bit	监视元素类型 - 32 位数字。
	agent_pid	收集的监视元素的名称。
	12	此元素的已收集值。
	. . .	. . .

db2GetSnapshot() 例程在用户提供的缓冲区中返回自描述格式快照数据。将在与要捕获的快照类型相关联的逻辑数据分组中返回数据。

快照请求返回的每一项都包含指定大小和类型的字段。该字段可用来对返回的数据进行语法分析。字段的大小还可用来跳过逻辑数据组。例如，要跳过 DB2 记录，需要确定数据流中的字节数。使用以下公式来计算要跳过的字节数:

```
DB2 逻辑数据分组的大小 + sizeof(sqlm_header_info)
```

使用 db2top 以交互方式进行监视的命令

db2top 监视实用程序快速高效地监视复杂的 DB2 环境。它结合来自所有数据库分区的 DB2 快照信息，使用基于文本的用户界面提供正在运行的 DB2 系统的动态实时视图。

关于此任务

以交互方式运行 **db2top** 时，您可以发出下列命令:

A 监视 HADR 集群中的主数据库或辅助数据库。

a 转至代理程序的应用程序详细信息（或在声明屏幕上限制代理程序）。**db2top** 命令将提示您输入代理程序标识。

B 显示关键服务器资源的主要使用者（瓶颈分析）。

c 此选项允许您更改屏幕上显示的列的顺序。语法采用下列格式: 1,2,3,...，其中 1,2,3 对应于所显示的第 1 列、第 2 列和第 3 列。这些是指定排序条件时要使用的列数。

当使用 c 交换关键字时，将显示屏幕，指定屏幕上显示的列的顺序。屏幕的左侧部分显示缺省顺序和列数；屏幕右侧部分显示当前排序。要更改列的顺序，在屏幕底部文本字段中输入新的列顺序。接着，如左侧显示的那样，输入相对的列位置，用逗号对其分隔。不需要指定所有列。对于后续的 **db2top** 监视会话，可以通过选择 w 将此列排序保存在 *$DB2TOPRC* 中。您可以进行排序，并选择采用哪种顺序在屏幕上显示列。`.db2toprc` 文件中列排序的有效关键字是:

- sessions=
- tables=
- tablespaces=

- bufferpools=
- dynsql=
- statements=
- locks=
- utilities=
- federation=

b 转至缓冲池屏幕。

C 打开或关闭快照数据收集器。

d 转至数据库屏幕。

D 转至动态 SQL 屏幕。

f 冻结屏幕。

F 在主服务器上监视联合查询。

G 打开或关闭图表。

h 转至帮助屏幕

H 转至历史记录屏幕

i 打开或关闭闲置会话。

k 切换实际值与增量值。

l 转至会话屏幕。

L 允许显示来自 SQL 屏幕的完整查询文本。然后，可以使用 e 或 X 选项来运行常规 DB2 说明。

m 显示内存池。

o 显示会话设置。

p 转至分区屏幕。

P 选择要发出快照的数据库分区。

q 退出 db2top。

R 重置快照数据。

s 转至语句屏幕。

S 运行本机 DB2 快照。

t 转至表空间屏幕。

T 转至表屏幕

u 显示活动的实用程序，并且跨数据库分区将它们聚集起来。

U 转至锁定屏幕。

V 设置缺省说明模式。

w 将会话设置写至 .db2toprc。

W agent_id、os_user、db_user、应用程序或网络名的观看方式。会话快照（选项 1）返回的语句将写至 agent.sql、os_user-agent.sql、db_user-agent.sql、application-

agent.sql 或 netname-agent.sql。当从动态 SQL 屏幕（选项 D）发出时，语句将采用与 db2advis 兼容的格式写至 db2adv.sql。

X 打开或关闭扩展方式。

z|Z 按升序或降序方式进行排序。

/ 将表达式输入至过滤器数据。表达式必须符合正则表达式。您可以采用不同方法过滤每个函数（屏幕）。可对整行应用 regexp 检查。

<|> 移至屏幕的左侧或右侧。

下列切换只适用于应用程序屏幕:

r 返回至上一函数。

R 切换自动刷新。

g 打开或关闭图表。

X 打开或关闭扩展方式。

d 显示代理程序。

要以交互方式启动 **db2top**，可发出下列命令:

```
db2top -d <database name>
```

当输入

```
db2top -d sample
```

时，将显示下列输出:

```
[\]11:57:10,refresh=2secs(0.000) Inactive,part=[1/1],<instanceName>:sample
[d=Y,a=N,e=N,p=ALL] [qp=off]

[/]: 当旋转时，它表示 db2top 在两个快照之间等待，否则，它表示 db2top 在等待 DB2 的答复
11:57:10: 当前时间
refresh=2secs: 时间间隔
refresh=!secs: 感叹号表示 DB2 处理快照所需的时间超过时间间隔。
在此情况下，db2top 将按 50% 增加时间间隔。
如果由于系统太忙而频繁发生此问题，那么您可以增加快照时间间隔（选项 I）、监视单一数据库分区（选项 P）或关闭扩展显示方式
（选项 x）
0.000: DB2 内部处理快照所花费的时间
d=Y/N: 增量或累积快照指示器（命令选项 -k 或选项 k）。
a=Y/N: 仅限于活动对象指示器的或所有对象指示器（-a 命令选项集或 i）
e=Y/N: 扩展显示指示器
p=ALL: 所有数据库分区
p=CUR: 当前数据库分区（-P 命令选项，未指定分区数）
p=3: 目标数据库分区数: 例如，3

Inactive: 如果 DB2 没有在运行，那么会显示不活动，否则会显示运行 DB2 的平台
part=[1/1]: 活动数据库分区数与总计数据库分区数。例如，part=[2,3] 表示总共有 3 个
数据库分区，其中有一个数据库分区停机（2 个数据库分区处于活动状态，共有 3 个）
<instanceName>: 实例名
sample: 数据库名称
```

示例

下列示例演示在分区数据库环境中以交互方式运行 **db2top** 监视实用程序:

```
db2top -d TEST -n mynode -u user -p passwd -V skm4 -B -i 1
命令参数如下所示:
-d TEST     # 数据库名称
-n mynode   #  节点名
-u user     #  用户标识
```

```
-p passwd   #  密码
-V skm4     #  模式名称
-B          #  启用粗体
-i 1        #  屏幕更新时间间隔: 1 秒
```

.db2toprc 配置文件

.db2toprc 配置文件是用户生成的文件，用于在初始化时为 **db2top** 监视实用程序设置参数。

db2top 实用程序将使用用户定义的变量 *$db2topRC* 搜索 .db2toprc 文件的位置。如果该变量尚未设置，那么 **db2top** 将首先在当前目录中搜索 .db2toprc 文件，然后再在 home 目录中搜索该文件。.db2toprc 文件是用户生成的文件。

环境变量

您可以设置下列环境变量:

- **DB2TOPRC**

  存储 .db2toprc 文件位置的用户定义的环境变量。例如，在 Linux 上，您可以将 **DB2TOPRC** 定义为: export db2topRC="~/db2top"。

  如果用户未设置该变量，那么 **db2top** 将首先在当前目录中搜索 .db2toprc 文件，然后再在 home 目录中搜索该文件。

- **DB2DBDFT**

  此变量指定要用于隐式连接的数据库的数据库别名。当命令行或 .db2toprc 配置文件中未指定数据库名称时，将会使用此变量。

- **EDITOR**

  此系统环境变量指定用于启动文本编辑器的命令，该文本编辑器用于显示说明结果或本机快照。

  如果此变量未设置，那么将使用 **vi**。

结构

此处描述了 .db2toprc 文件中的一些条目。

cpu=command

使用此条目在屏幕输出右侧第二行中显示 CPU 活动的结果。例如:

cpu=vmstat 2 2 | tail -1 | awk '{printf("%d(usr+sys)",$14+$15);}' 将在屏幕右侧显示 Cpu=2(usr+sys)。

io=command

使用此条目来指定命令并在屏幕输出左侧的第二行中显示结果。例如:

io=vmstat 2 2 | tail -1 | awk '{printf("%d(bi+bo)",$10+$11);}' 将在屏幕左侧显示 Disk=76(bi+bo)。

两个命令均作为后台进程运行，并且屏幕上的字段会以异步方式更新。

shell alias=command

使用此 shell 条目来指定用户定义的命令，例如: 当输入 M 时，shell M=top 会从 **db2top** 会话衍生在顶部。

function alias=command

使用此条目来指定用户定义的命令，例如: `function N=netstat` 创建名为 N 的新函数，该函数重复地显示 **netstat** 的输出。可以存在多个 `function` 条目。您必须将它们置于单独的行。例如:

```
function Q=netstat
function N=df -k
```

sort=command

使用此条目来指定排序顺序，例如: `sort=command` 为此函数创建缺省排序顺序，其中 command 是列数。该排序顺序可以是升序或降序。排序对会话、表、表空间、缓冲池、dynsql、语句、锁、实用程序和联合有效。

样本 .db2toprc 文件

没有缺省 .db2toprc 配置文件。但是，您可以按“W”来为当前设置创建 .db2toprc。使用下列样本 **.db2toprc** 文件作为参考。注释已添加到所有条目。

```
# db2top 配置文件
# 在 UNIX 上，应该位于 $HOME/.db2toprc 中
# db2top-1.0a 生成的文件
#
node= # [-n] 节点名
database=sample # [-d] 数据库名称
user= # [-u] 数据库用户
password= # [-p] 用户密码（加密）
schema= # [-V] 说明的缺省模式
interval=2 # [-i] 采样时间间隔
active=OFF # [-a] 仅显示活动会话（打开/关闭）
reset=OFF # [-R] 在启动时重置快照（打开/关闭）
delta=ON # [-k] 切换增量值/累积值的显示（打开/关闭）
gauge=ON # 在会话列表上显示图表（打开/关闭）
colors=ON  # 如果终端支持色彩，那么为 True。如果它可以用色彩显示信息，那么通知 GE_WRS
graphic=ON  # 如果终端支持半图解字符，那么为 True（打开/关闭）。
port= # 用于网络收集的端口
streamsize=size # 每小时的最大收集大小（例如，1024 或 1K : K、M 或 G）
# 从操作系统获取 cpu 使用信息的命令
cpu=vmstat 2 2 | tail -1 | awk '{printf("%d(usr+sys)",$14+$15);}'
# 从操作系统获取 IO 使用信息的命令
io=vmstat 2 2 | tail -1 | awk '{printf("%d(bi+bo)",$10+$11);}'
# 会话屏幕中信息的排序
# 会话屏幕的列顺序（选项 l）
sessions=0,1,18,2,3,4,5,6,7,8,9,10,11,12,13,14,15,16,17,19,20,21,22,23
# 表屏幕的列顺序（选项 T）
tables=0,1,2,4,3,5,6,7
# 表空间屏幕的列顺序（选项 t）。
# 将在列 #22 按升序顺序对显示内容进行排序
tablespaces=0,1,18,2,3,4,5,6,7,8, sort=22a
# 缓冲池屏幕的列顺序（选项 b）
bufferpools=0,1,18,2,3,4,5,6,7,8,9,10
# 动态 SQL 屏幕的列顺序（选项 D）
dynsql=0,1,18,2,3,4,5,6,7,8,9
statements=0,1
locks=0,1
utilities=0 # 包含实用程序屏幕的缺省列和排序顺序
federation=0,2,4  # 包含联合屏幕的缺省列和排序顺序

# 用户定义的命令
shell P=top
function N=date && netstat -t tcp
```

基于开关的监视概念

系统监视开关

系统监视开关用于控制快照监视器和某些事件监视器收集数据的方式。

注: 这些系统监视开关不影响 DB2 V9.7 中引入的工作单元事件监视器和锁定事件监视器。

快照监视器和某些事件监视器将报告系统监视器所收集的数据。收集系统监视器数据将引入针对数据库管理器的额外处理。例如，为了计算 SQL 语句的执行时间，数据库管理器必须对操作系统进行调用，以获取每个语句执行之前和执行之后的时间戳记。这些系统调用类型的成本通常很高。系统监视器也会增加内存消耗。对于系统监视器跟踪的每个监视元素，数据库管理器将使用内存来存储收集的数据。

为了将维护监视信息所涉及的额外处理时间降至最低，监视开关将控制数据库管理器可能进行的高成本数据收集。每个开关只有两个设置: ON 或 OFF。如果监视开关为 OFF，那么受开关控制的监视元素不会收集任何信息。有很大一部分基本监视数据不受开关控制，并且不管开关设置如何，始终会收集这些信息。

每个监视应用程序都有自己的监视开关（和系统监视器数据）逻辑视图。在启动时，每个应用程序将从数据库管理器配置文件中的 dft_monswitches 参数继承其监视开关设置（在实例级别）。监视应用程序可使用 UPDATE MONITOR SWITCHES USING MONSWITCH OFF/ON 命令来更改其监视开关设置。MONSWITCH 参数包含下一部分显示的“快照监视开关”表的“监视开关”列中的值。在应用程序级别对开关设置的更改仅影响从中更改开关的应用程序。

可在不停止数据库管理系统的情况下更改实例级别的监视开关。为此，请使用 UPDATE DBM CFG USING DBMSWITCH OFF/ON 命令。DBMSWITCH 参数包含下一部分显示的“快照监视开关”表的“DBM 参数”列中的值。开关的动态更新要求执行更新的应用程序显式连接至实例，以使更改动态生效。动态更新不会影响其他现有快照应用程序。新的监视应用程序将继承已更新实例级别监视开关设置。要让现有监视应用程序继承新的缺省监视开关值，它必须终止并重新建立连接。如果在数据库管理器配置文件中更新开关，那么会更新分区数据库中的所有分区的开关。

数据库管理器记录所有快照监视应用程序及其开关设置。如果开关在应用程序的配置中设置为 ON，那么数据库管理器总是会收集该监视器数据。如果之后同一开关在应用程序配置中设置为 OFF，那么只要至少有一个此开关设置为 ON 的应用程序，数据库管理器仍将收集数据。

时间和时间戳记元素的收集由 TIMESTAMP 开关控制。如果将此开关设置为 OFF（在缺省情况下设置为 ON），那么指示在确定与时间或时间戳记相关的监视元素时，数据库管理器将跳过所有时间戳记操作系统调用。在 CPU 利用率达到 100% 时，应将此开关设置为 OFF。当此情况发生时，发出时间戳记导致的性能下降的幅度会急剧增长。对于可由 TIMESTAMP 开关和另一开关控制的监视元素，如果任一开关设置为 OFF，那么不会收集数据。因此，如果 TIMESTAMP 开关设置为 OFF，那么受另一监视开关控制的整体数据成本会大大降低。

事件监视器不会以与快照监视应用程序相同的方式受监视开关的影响。定义事件监视器时，它会将指定事件类型所需的实例级别监视开关自动设置为 ON。例如，死锁事件监视器会将 LOCK 监视开关自动设置为 ON。在激活事件监视器时，必需的监视开关将设置为 ON。释放事件监视器时，监视开关将设置为 OFF。

事件监视器不会自动设置 TIMESTAMP 监视开关。这是控制所有监视元素的收集的唯一监视开关，这些监视元素属于事件监视器逻辑数据分组。如果 TIMESTAMP 开关设

置为 OFF，那么不会收集事件监视器收集的大多数时间戳记和时间监视元素。这些元素仍将写至指定的表、文件或管道，但必须带有值零。

表 119. 快照监视开关

监视开关	DBM 参数	提供的信息
BUFFERPOOL	DFT_MON_BUFPOOL	读写次数，所花时间
LOCK	DFT_MON_LOCK	等待锁定次数，死锁数
SORT	DFT_MON_SORT	使用的堆数，排序性能
STATEMENT	DFT_MON_STMT	启动/停止时间，语句标识
TABLE	DFT_MON_TABLE	活动量度（读/写行数）
UOW	DFT_MON_UOW	开始/结束时间，完成状态
TIMESTAMP	DFT_MON_TIMESTAMP	时间戳记

在捕获快照或使用事件监视器之前，必须确定需要数据库管理器收集的数据。如果想要在快照中收集下列任何特殊类型的数据，那么需要设置适当的监视开关。

- 缓冲池活动信息
- 锁定、锁定等待及与时间有关的锁定信息
- 排序信息
- SQL 语句信息
- 表活动信息
- 时间和时间戳记信息
- 工作单元信息

在缺省情况下，与以上列出的信息类型相对应的开关全部设置为“OFF”，但对应于时间和时间戳记信息的开关在缺省情况下设置为“ON”。

事件监视器仅受时间和时间戳记信息开关影响。所有其他开关设置对事件监视器收集的数据没有影响。

通过 CLP 设置系统监视开关

系统监视开关用于控制系统监视器的数据收集操作。通过将特定监视开关设置为 ON，可以收集特定类型的监视器数据。

开始之前

执行任何监视开关更新的应用程序必须具有实例连接。必须具有 SYSADM、SYSCTRL、SYSMAINT 或 SYSMON 权限中的一种才能使用下列命令:

- UPDATE MONITOR SWITCHES
- GET MONITOR SWITCHES
- GET DATABASE MANAGER MONITOR SWITCHES

必须具有 SYSADM 权限才能使用 UPDATE DBM CFG 命令。

关于此任务

过程

- 要激活任何局部监视开关，请使用 UPDATE MONITOR SWITCHES命令。开关将保持活动状态，直到应用程序 (CLP) 拆离，或者直到再次使用 UPDATE MONITOR SWITCHES 命令释放它们。以下示例将所有局部监视开关更新为 ON:

```
db2 update monitor switches using BUFFERPOOL on LOCK on
    SORT on STATEMENT on TIMESTAMP on TABLE on UOW on
```

- 要释放任何局部监视开关，请使用 UPDATE MONITOR SWITCHES 命令。以下示例将所有局部监视开关更新为 OFF:

```
db2 update monitor switches using BUFFERPOOL off, LOCK off,
        SORT off, STATEMENT off, TIMESTAMP off, TABLE off, UOW off
```

  以下是在发出上面显示的 UPDATE MONITOR SWITCH 命令后希望看到的输出的示例:

```
          监视器记录开关

数据库分区号 1 的开关列表
缓冲池活动信息 (BUFFERPOOL)   = OFF
锁定信息 (LOCK)               = OFF
排序信息 (SORT)               = OFF
SQL 语句信息 (STATEMENT)      = OFF
表活动信息 (TABLE)            = OFF
工作单元信息 (UOW)            = OFF
获取时间戳记信息 (TIMESTAMP)  = OFF
```

- 还可以在数据库管理器级别操作监视开关。这涉及使用 UPDATE DBM CFG 命令在数据库管理器配置文件中更改 dft_monswitches 参数。在以下示例中，除了基本信息之外，仅收集锁定开关控制的信息。

```
db2 update dbm cfg using DFT_MON_LOCK on
```

  每次启动监视应用程序时，它将从数据库管理器继承监视开关设置。对数据库管理器的监视开关设置所作的任何更改不会影响任何正在运行的监视应用程序。监视应用程序必须重新连接至实例以获取对监视开关设置的所有更改。

- 对于分区数据库系统，可专门为特定分区设置监视开关，或者为所有分区设置全局监视开关。
  1. 要为特定分区（如分区号 3）设置监视开关（如 BUFFERPOOL），请发出以下命令:

     ```
     db2 update monitor switches using BUFFERPOOL on
             at dbpartitionnum 3
     ```

  2. 要为所有分区设置监视开关（如 SORT），请发出以下命令:

     ```
     db2 update monitor switches using SORT on global
     ```

- 要检查局部监视开关的状态，请使用 GET MONITOR SWITCHES 命令。

```
db2 get monitor switches
```

- 对于分区数据库系统，可专门查看特定分区的监视开关设置，或者查看所有分区的全局监视开关设置。
  1. 要查看特定分区（如分区号 2）的监视开关设置，请发出以下命令:

     ```
     db2 get monitor switches at dbpartitionnum 2
     ```

  2. 要查看所有分区的监视开关设置，请发出以下命令:

     ```
     db2 get monitor switches global
     ```

- 要在数据库管理器级别（或实例级别）检查监视开关的状态，请使用 GET DATABASE MANAGER MONITOR SWITCHES 命令。此命令将显示要监视的实例的整体开关设置。

  ```
  db2 get database manager monitor switches
  ```

  以下是发出以上显示的命令后希望看到的输出的示例:

  ```
           收集的 DBM 系统监视器信息

  数据库分区号 1 的开关列表
  缓冲池活动信息 (BUFFERPOOL)   = OFF
  锁定信息 (LOCK)               = ON    10-25-2001 16:04:39
  排序信息 (SORT)               = OFF
  SQL 语句信息 (STATEMENT)      = OFF
  表活动信息 (TABLE)            = OFF
  工作单元信息 (UOW)            = OFF
  获取时间戳记信息 (TIMESTAMP)  = OFF
  ```

结果

既然已经设置了需要的监视开关并且确认了开关设置，就可以捕获并收集监视器数据了。

通过客户机应用程序设置系统监视开关

系统监视开关用于控制系统监视器的数据收集操作。通过将特定监视开关设置为 ON，可以收集特定类型的监视器数据。

开始之前

执行任何监视开关更新的应用程序必须具有实例连接。必须具有 SYSADM、SYSCTRL、SYSMAINT 或 SYSMON 权限才能使用 db2MonitorSwitches API。

过程

1. 包括下列 DB2 库: sqlutil.h 和 db2ApiDf.h。可在 sqllib 下的 include 子目录中找到它们。

   ```
   #include <sqlutil.h>
   #include <db2ApiDf.h>
   #include <string.h>
   #include <sqlmon.h>
   ```

2. 将开关列表缓冲区单元大小设置为 1 KB。

   ```
   #define SWITCHES_BUFFER_UNIT_SZ 1024
   ```

3. 初始化 sqlca、db2MonitorSwitches 和 sqlm_recording_group 结构。还应初始化指针以包含开关列表缓冲区并确定缓冲区大小。

   ```
   struct sqlca sqlca;
   memset (&sqlca, '\0', sizeof(struct sqlca));
   db2MonitorSwitchesData switchesData;
   memset (&switchesData, '\0', sizeof(switchesData));
   struct sqlm_recording_group switchesList[SQLM_NUM_GROUPS];
   memset(switchesList, '\0', sizeof(switchesList));
   sqluint32 outputFormat;
   static sqluint32 switchesBufferSize = SWITCHES_BUFFER_UNIT_SZ;
   char *switchesBuffer;
   ```

4. 初始化缓冲区以容纳开关列表输出。

```
switchesBuffer = (char *)malloc(switchesBufferSize);
memset(switchesBuffer, '\0', switchesBufferSize));
```

5. 要更改局部监视开关的状态，请更改 sqlm_recording_group 结构中的元素（按先前步骤中的指示命名为 switchesList）。对于要设置为“ON”的监视开关，参数 input_state 应设置为 SQLM_ON。对于要设置为“OFF”的监视开关，参数 input_state 必须设置为 SQLM_OFF。

```
switchesList[SQLM_UOW_SW].input_state = SQLM_ON;
switchesList[SQLM_STATEMENT_SW].input_state = SQLM_ON;
switchesList[SQLM_TABLE_SW].input_state = SQLM_ON;
switchesList[SQLM_BUFFER_POOL_SW].input_state = SQLM_OFF;
switchesList[SQLM_LOCK_SW].input_state = SQLM_OFF;
switchesList[SQLM_SORT_SW].input_state = SQLM_OFF;
switchesList[SQLM_TIMESTAMP_SW].input_state = SQLM_OFF;
switchesData.piGroupStates = switchesList;
switchesData.poBuffer = switchesBuffer;
switchesData.iVersion = SQLM_DBMON_VERSION9_5;
switchesData.iBufferSize = switchesBufferSize;
switchesData.iReturnData = 0;
switchesData.iNodeNumber = SQLM_CURRENT_NODE;
switchesData.poOutputFormat = &outputFormat;
```

注: 如果 iVersion 小于 SQLM_DBMON_VERSION8，那么 SQLM_TIMESTAMP_SW 不可用。

6. 要提交对开关设置的更改，请调用 db2MonitorSwitches() 函数。将 db2MonitorSwitchesData 结构（在此示例中命名为 switchesData）作为参数传递至 db2MonitorSwitches API。switchesData 以参数的形式包含 sqlm_recording_group 结构。

```
db2MonitorSwitches(db2Version810, &switchesData, &sqlca);
```

7. 从开关列表缓冲区处理开关列表数据流。
8. 清除开关列表缓冲区。

```
free(switchesBuffer);
free(pRequestedDataGroups);
```

结果

既然已经设置了需要的监视开关并且确认了开关设置，就可以捕获并收集监视器数据了。

系统监视开关自描述数据流

在使用 db2MonitorSwitches API 更新或查看当前系统监视开关设置之后，API 将以自描述数据流的形式返回开关设置。第 408 页的图 7 显示可能对分区数据库环境返回的开关列表信息的结构。

注:

1. 描述性名称用于示例和表中的标识。在实际数据流中，将在这些名称前加上 **SQLM_ELM_** 前缀。例如，db_event 在事件监视器输出中显示为 SQLM_ELM_DB_EVENT。在实际数据流中，将在类型前加上 **SQLM_TYPE_** 前缀。例如，HEADER 在数据流中将显示为 SQLM_TYPE_HEADER。
2. 对于全局开关请求，返回的信息的分区顺序在每个开关请求中可能有所不同。在此情况下，分区标识包括在数据流中。

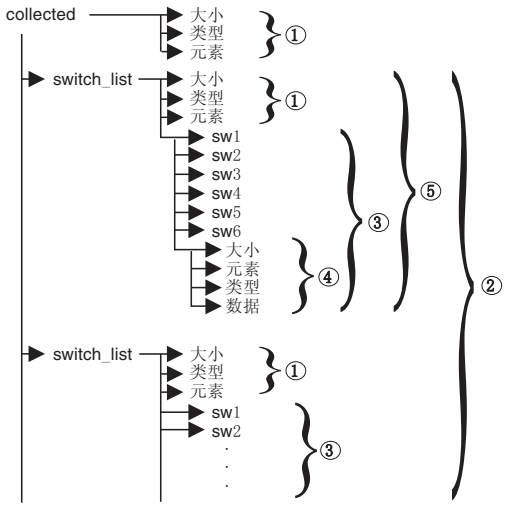


图 7. 开关列表监视器数据流

1. 每个逻辑数据组以指示其大小和名称的头开始。此大小不包括头本身占用的数据量。
2. 已收集头中的大小返回所有分区的所有监视开关列表的总大小。
3. 开关列表头中的大小元素指示该分区的开关数据的大小。
4. 开关信息是自描述格式。
5. 对于非分区数据库，将返回独立分区的开关设置。即只返回一个开关列表。

数据库系统监视器数据结构

系统监视器使用快照监视器和某些事件监视器的接口来收集和存储供您访问的信息。数据库系统监视器将收集的信息存储在称为*监视元素*（先前称为数据元素）的实体中。每个监视元素存储有关数据库系统状态的一个特定方面的信息。

此外，监视元素由唯一名称标识并且存储特定类型的信息。

以下元素类型可供系统监视器存储数据:

计数器 计算活动的发生次数。在监视期间，计数器值会增大。大多数计数器元素可以重置。

标尺 指示某个项的当前值。根据数据库活动的不同，标尺值会增加或减小（例如，挂起的锁定数）。标尺元素不能重置。

水位标记
指示自开始监视以来元素所达到的最高值（最大值）或最低值（最小值）。水位标记元素不能重置。

信息 提供参考类型的监视活动详细信息。这可以包括诸如分区名称、别名和路径详细信息之类的项。信息元素不能重置。

时间戳记
通过提供自 1970 年 1 月 1 日后经历的秒数和微秒数，以指示活动发生的日期和时间。对于快照监视器和事件监视器，时间戳记元素的收集由 TIME-

STAMP 监视开关控制。此开关在缺省情况下设置为 ON，如果数据库实例的 CPU 利用率达到 100%，那么考虑性能方面的原因应将其设置为 OFF。时间戳记元素不能重置。

时间戳记元素的值为零意味着“不可用”。如果尝试导入此数据，这样的值将生成超过范围错误（SQL0181）。为避免此错误，在导出数据前将该值更新为有效时间戳记值。

时间 返回执行活动时耗用的秒数和微秒数。对于快照监视器和事件监视器，大多数时间元素的收集由 TIMESTAMP 监视开关控制。此开关在缺省情况下设置为 ON，如果数据库实例的 CPU 利用率达到 100%，那么考虑性能方面的原因应将其设置为 OFF。某些时间元素可以重置。

监视元素收集一个或多个逻辑数据组的数据。逻辑数据组是一组监视元素，它们用来收集特定数据库活动作用域的数据库系统监视信息。监视元素在逻辑数据组中按它们提供的信息级别排序。例如，在进行快照监视时，“总排序时间”监视元素将返回数据库（dbase）、应用程序（appl）和语句（stmt）信息；因此，它将出现在已列示并且用圆括号括起来的每个逻辑数据组中。

尽管许多监视元素同时被快照监视器和事件监视器使用，但每个监视器使用一个不同的逻辑数据组集合。这是因为可对其捕获快照的数据库活动作用域与可对其收集事件数据的数据库活动作用域不同。从实际上说，可从快照监视器访问的完整监视元素集合不同于可从事件监视器访问的那些监视元素。

计数器状态和可视性

系统监视器收集的监视元素包括若干累加计数器。这些计数器将在数据库或数据库管理器操作期间递增，如每次应用程序落实事务时。

将在适用对象可用时初始化计数器。例如，对数据库读取的缓冲池页数（基本监视元素）在数据库激活时设置为零。

某些可以由系统监视器收集的计数器由监视开关控制。如果特定监视开关设置为“OFF”，那么受其控制的监视元素不会收集数据。当监视开关设置为“ON”时，所有关联计数器都将重置为零。

当事件监视器激活时，事件监视器返回的计数器将重置为零。

事件监视器计数表示自下列其中一个起始点之后的计数:

- 对数据库、表空间和表启动事件监视器时。
- 对现有连接启动事件监视器时。
- 应用程序连接，对于在启动监视器之后建立的连接。
- 在启动监视器之后又启动下一个事务（工作单元）或语句时。
- 在启动监视器之后发生死锁时。

每个事件监视器和任意监视应用程序（使用快照监视器 API 的应用程序）有自己的系统监视器数据逻辑视图。这意味着重置或初始化计数器时，将仅影响重置或初始化计数器的事件监视器或应用程序。除非关闭事件监视器然后再打开它，否则事件监视器计数器不能重置。获取快照的应用程序可使用 RESET MONITOR 命令随时重置其计数器视图。

如果在语句启动之后启动语句事件监视器，那么监视器将在启动下一条 SQL 语句时开始收集信息。因此，事件监视器不会返回有关启动监视器时数据库管理器正在执行的语句的信息。对于事务信息也如此。

系统监视器输出: 自描述数据流

除了在屏幕上显示系统监视器数据或将其存储在 SQL 表中之外，还可以开发客户机应用程序来进行处理。系统监视器通过自描述数据流同时对快照监视器和事件监视器返回监视器数据。

在快照监视应用程序中，可调用快照 API 以捕获快照，然后直接处理数据流。

处理事件监视器数据的不同之处在于发生数据库事件时事件数据将发送至应用程序。对于管道事件监视器，应用程序将等待事件数据到达，并在事件数据到达时进行处理。对于文件事件监视器，应用程序将对事件文件进行语法分析，因此会分批处理事件记录。

此自描述数据流允许您对返回的数据进行语法分析，一次分析一个元素。这样就可以进行多方面的监视，包括查找有关特定应用程序或特定数据库状态的信息。

返回的监视器数据具有以下格式:

大小 存储在监视元素或逻辑数据分组中的数据的大小（以字节计）。如果是逻辑数据分组，那么此项为逻辑组中的所有数据的大小。例如，数据库逻辑分组 (*db*) 包含各个监视元素（如 *total_log_used*）及其他逻辑数据分组，如前滚信息 (*rollforward*)。这不包括“大小”、“类型”和“元素”信息占用的大小。

类型 存储在数据中的元素的类型（如可变长度字符串或带符号的 32 位数字值）。元素类型头指的是元素的逻辑数据分组。

元素标识
监视器捕获的监视元素的标识。如果是逻辑数据分组，那么此项为该组的标识（如 *collected*、*dbase* 或 *event_db*）。

数据 监视器对监视元素收集的值。如果是逻辑数据分组，那么该数据由属于它的监视元素组成。

监视元素中的所有时间戳记将以两个不带符号的 4 字节监视元素（秒和微秒）的形式返回。它们表示 GMT 时间 1970 年 1 月 1 日。

监视元素中的字符串的大小元素表示该字符串元素的实际数据大小。因为此字符串不是以 NULL 结束的，所以此大小不包括 NULL 终止符。

监视器数据的内存需求

将从监视器堆分配监视器数据所需的内存。监视器堆大小由 **mon_heap_sz** 数据库配置参数控制。此参数的缺省值为 AUTOMATIC，这表示监视器堆可以根据需要增大，直到达到 instance_memory 限制为止。

如果以手动方式配置 **mon_heap_sz** 参数，请考虑下列因素:

- 监视应用程序的数目
- 事件监视器的数目和特征
- 设置的监视开关

- 数据库活动级别

如果监视器命令失败并且 SQLCODE 为 -973，那么考虑增大 **mon_heap_sz** 参数的值。

以下公式提供监视器堆所需的大致页数:

```
(应用程序使用的内存量                    +
 事件监视器使用的内存量                  +
 监视应用程序使用的内存量                +
 网关应用程序使用的内存量)              / 4096
```

每个应用程序使用的内存量

- 如果 STATEMENT 开关设置为 OFF 或零
- 如果 STATEMENT 开关设置为 ON:
  - 为将要同时运行的每个语句加上 400 字节。（即，应用程序可能具有的打开游标数）。这*不是*应用程序已经运行的累积语句总数。
  - 如果是分区数据库，那么为每个语句加上以下大小:
    - 200 字节 * (平均子节数)
- 如果应用程序发出 sqleseti() info，那么加上用户标识、应用程序名称、工作站名称和记帐字符串大小。

每个事件监视器使用的内存量

对于每个类型为 ACTIVITIES 的事件监视器:

- 3500 字节
- 如果事件监视器的类型为 TABLES，那么加上 36K * (CPU 核心数 + 1)
- 如果事件监视器的类型为 FILE 或 PIPE，那么加上 2K * (CPU 核心数 + 1)

如果您预计工作量较大，那么为每个记录加上 250 兆字节。否则，根据您估计的工作量加上相应的数目。

对于每个类型为 LOCKING 或 UOW 的事件监视器:

- 3500 字节
- 3K * (CPU 核心数 + 1)

如果您预计工作量较大，那么为每个记录加上 250 兆字节。否则，根据您估计的工作量加上相应的数目。

对于每个具有以下类型的事件监视器:
DATABASE、TABLES、TABLESPACES、BUFFERPOOLS、CONNECTIONS 或 DEADLOCK:

- 4100 字节
- 2 * BUFFERSIZE
- 如果事件监视器写至文件，那么加上 550 字节。
- 如果事件监视器的类型为 DATABASE，那么:
  - 加上 6000 字节
  - 对语句高速缓存中的每个语句加上 100 字节
- 如果事件监视器的类型为 TABLES，那么:
  - 加上 1500 字节

  - 对访问的每个表加上 70 字节
- 如果事件监视器的类型为 TABLESPACES，那么:
  - 加上 450 字节
  - 对每个表空间加上 350 字节
- 如果事件监视器的类型为 BUFFERPOOLS，那么:
  - 加上 450 字节
  - 对每个缓冲池加上 340 字节
- 如果事件监视器的类型为 CONNECTIONS:
  - 加上 1500 字节
  - 对于每个已连接应用程序:
    - 加上 750 字节
  - 记住加上出自第 411 页的『每个应用程序使用的内存量』的值。
- 如果事件监视器的类型为 DEADLOCK:
  - 并且 WITH DETAILS HISTORY 正在运行:
    - 加上 X*475 字节，加上次数为希望运行的并行应用程序的最大数目，其中 X 是应用程序的工作单元中的最大期望语句数。
  - 并且 WITH DETAILS HISTORY VALUES 正在运行:
    - 同时加上 X*Y 字节，加上次数为希望运行的并行应用程序的最大数目，其中 Y 是将绑定至 SQL 语句的参数值的最大期望大小。

每个监视应用程序使用的内存量

- 250 字节
- 对于要重置的每个数据库:
  - 350 字节
  - 对每个远程数据库加上 200 字节。
  - 如果 SORT 开关设置为 ON，那么加上 25 字节。
  - 如果 LOCK 开关设置为 ON，那么加上 25 字节。
  - 如果 TABLE 开关设置为 ON:
    - 加上 600 字节
    - 对访问的每个表加上 75 字节
  - 如果 BUFFERPOOL 开关设置为 ON:
    - 加上 300 字节
    - 对访问的每个表空间加上 250 字节
    - 对访问的每个缓冲池加上 250 字节
  - 如果 STATEMENT 开关设置为 ON:
    - 加上 2100 字节
    - 对每个语句加上 100 字节
  - 对连接至数据库的每个应用程序:
    - 加上 600 字节
    - 对应用程序连接至的每个远程数据库加上 200 字节
    - 如果 SORT 开关设置为 ON，那么加上 25 字节

- 如果 LOCK 开关设置为 ON，那么加上 25 字节
- 如果 BUFFERPOOL 开关设置为 ON，那么加上 250 字节

- 对于要重置的每个 DCS 数据库:
  - 对数据库加上 200 字节
  - 对连接至数据库的每个应用程序加上 200 字节
  - 如果 STATEMENT 开关设置为 ON，那么必须重置传输级别数据:
    - 对于每个数据库，对每个传输级别加上 200 字节
    - 对于每个应用程序，对每个传输级别加上 200 字节

网关应用程序使用的内存量

- 对每个主机数据库加上 250 字节（即使所有开关设置为 OFF）
- 对每个应用程序加上 400 字节（即使所有开关设置为 OFF）
- 如果 STATEMENT 开关设置为 ON:
  - 对于每个应用程序，对同时运行的每个语句加上 200 字节（即应用程序可能具有的打开游标数）。这不是应用程序已经运行的累积语句总数。
  - 必须计算传输级别数据:
    - 对于每个数据库，对每个传输级别加上 200 字节
    - 对于每个应用程序，对每个传输级别加上 200 字节
- 如果 UOW 开关设置为 ON:
  - 则对每个应用程序加上 50 字节
- 对于使用 TMDB 的每个应用程序（用于 SYNCPOINT TWOPHASE 活动）:
  - 加上 20 字节并加上 XID 本身的大小
- 对于已发出 sqleseti 以设置客户机名称、应用程序名称、wkstn 或记帐的任何应用程序:
  - 加上 800 字节并加上记帐字符串本身的大小

监视缓冲池活动

数据库服务器执行的所有数据读取和更新操作都是对缓冲池进行的。当应用程序需要数据时，就会将数据从磁盘复制到缓冲池。

下列程序将页放入缓冲池:

- 代理程序。这是同步 I/O。
- I/O 服务器（预取程序）。这是异步 I/O。

下列程序将页从缓冲池写入磁盘:

- 代理程序（同步方式）
- 页清除程序（异步方式）

如果服务器需要读取一页数据，并且该页已在缓冲池中，那么访问该页的速度要比从磁盘读取该页快。如果在缓冲池中能命中尽可能多的页，那就最好不过了。避免磁盘 I/O 操作对于提高数据库性能来说十分重要，因此，在调整性能时，正确地配置缓冲池是其中一项最重要的考虑事项。

对于页请求，如果该页已在缓冲池中，数据库管理器就不需要从磁盘装入该页便可以为该请求提供服务，缓冲池*命中率*指的就是此类情况所占的时间百分比。缓冲池命中率越高，磁盘 I/O 操作频率就会越低。

注：下面的信息讨论 DB2 pureScale 环境以外的环境中的缓冲池。缓冲池在 DB2 pureScale 环境中以不同方式工作。有关更多信息，请参阅 *数据库监视指南和参考* 中的“在 DB2 pureScale 环境中监视缓冲池”。

例如，可按如下所示计算整体缓冲池命中率:

```
     ((pool_data_lbp_pages_found + pool_index_lbp_pages_found
+ pool_xda_lbp_pages_found
-
pool_async_data_lbp_pages_found - pool_async_index_lbp_pages_found
- pool_async_xda_lbp_pages_found)
/ (pool_data_l_reads
+ pool_index_l_reads + pool_xda_l_reads + pool_temp_data_l_reads +
pool_temp_xda_l_reads + pool_temp_index_l_reads)) × 100
```

此计算公式考虑缓冲池高速缓存的所有页（索引和数据）。

另一种方便的方法是使用 BP_HITRATIO 管理视图来监视缓冲池命中率。

对于大型数据库来说，增加缓冲池大小对缓冲池命中率的影响极小。它的数据页数可能非常多，增加缓冲池大小并不会提高命中统计机率。而是，您会发现通过调整索引缓冲池命中率能获得需要的结果。这可以通过两种方法实现:

1. 将数据和索引分到两个不同的缓冲池中并分别对它们进行调整。
2. 使用一个缓冲池，但增加其大小，直到索引命中率不再提高为止。索引缓冲池命中率的计算公式如下:

   ```
   ((pool_index_lbp_pages_found
   - pool_async_index_lbp_pages_found - pool_temp_index_l_reads)
   / pool_index_l_reads) × 100
   ```

第一种方法通常效率更高，但由于它要求索引和数据在不同的表空间中，所以对于现有数据库来说不能选择此方法。此方法还要求调整两个缓冲池（而不是只调整一个缓冲池），此任务可能更难以完成，当内存有限时尤其如此。

您还应该考虑预取程序对命中率的影响。预取程序将数据页读入缓冲池，即预计应用程序需要这些数据页（异步方式）。在大多数情况下，这些页刚好是在需要它们之前读取的（需要的情况）。但是，预取程序会将不会使用到的页读入缓冲池，从而执行不必要的 I/O 操作。例如，应用程序开始读整个表。这种情况被检测到，预取开始，但应用程序在填满应用程序缓冲区后停止读取。同时，预取操作已经读取了许多其他的页。已经为不会使用到的页执行了 I/O，那些页占用了缓冲池的部分空间。

页清除程序监视缓冲池并以异步方式将页写入磁盘。它们的目标是:

- 确保代理程序在缓冲池中总能找到可用页。如果代理程序在缓冲池中找不到可用页，它就必须自己清除它们，相关应用程序的响应速度就会变慢。
- 加快系统崩溃时的数据库恢复速度。已写入磁盘的页数越多，恢复数据库时必须处理的日志文件记录数就越少。

虽然会将脏页写入磁盘，但除非需要空间来读入新页，否则不会立即从缓冲池中除去这些页。

注: 缓冲池信息通常是在表空间级别收集的，但数据库系统监视器的工具可以提高到缓冲池级别和数据库级别来收集信息。根据分析类型的不同，可能需要在任何或全部这些级别检查此数据。

数据库系统监视器接口

监视任务	API
捕获快照	db2GetSnapshot
转换自描述数据流	db2ConvMonStream
显示数据库系统监视开关	db2MonitorSwitches
估计快照大小	db2GetSnapshotSize
获取/更新监视开关	db2MonitorSwitches
重置监视器计数器	db2ResetMonitor
更新数据库系统监视开关	db2MonitorSwitches

监视任务	CLP 命令
捕获快照	GET SNAPSHOT
显示数据库管理器系统监视开关	GET DATABASE MANAGER MONITOR SWITCHES
显示监视应用程序的监视开关	GET MONITOR SWITCHES
格式化事件监视器跟踪	db2evmon
对写至表 CREATE EVENT MONITOR 语句生成样本 SQL	db2evtbl
列示活动数据库	LIST ACTIVE DATABASES
列示连接至数据库的应用程序	LIST APPLICATIONS
列示 DCS 应用程序	LIST DCS APPLICATIONS
重置监视器计数器	RESET MONITOR
更新数据库系统监视开关	UPDATE MONITOR SWITCHES

监视任务	SQL 语句
激活事件监视器	SET EVENT MONITOR STATE
创建事件监视器	CREATE EVENT MONITOR
释放事件监视器	SET EVENT MONITOR STATE
除去事件监视器	DROP
写入事件监视器值	FLUSH EVENT MONITOR

监视任务	SQL 函数
确定事件监视器的状态	EVENT_MON_STATE 标量函数
获取数据库管理器级别快照	SNAPDBM 管理视图和 SNAP_GET_DBM 表函数
在数据库管理器级别获取当前监视开关设置	SNAPSWITCHES 管理视图和 SNAP_GET_SWITCHES 表函数
获取快速通信管理器快照	SNAPFCM 管理视图和 SNAP_GET_FCM 表函数
获取给定分区的快速通信管理器快照	SNAPFCM_PART 管理视图和 SNAP_GET_FCM_PART 表函数
获取数据库级别快照	SNAPDB 管理视图和 SNAP_GET_DB 表函数

监视任务	SQL 函数
获取应用程序级别快照	SNAPAPPL 管理视图和 SNAP_GET_APPL 表函数
获取应用程序级别快照	SNAPAPPL_INFO 管理视图和 SNAP_GET_APPL_INFO 表函数
获取锁定等待信息的应用程序级别快照	SNAPLOCKWAIT 管理视图和 SNAP_GET_LOCKWAIT 表函数
获取语句信息的应用程序级别快照	SNAPSTMT 管理视图和 SNAP_GET_STMT 表函数
获取代理程序信息的应用程序级别快照	SNAPAGENT 管理视图和 SNAP_GET_AGENT 表函数
获取子节信息的应用程序级别快照	SNAPSUBSECTION 管理视图和 SNAP_GET_SUBSECTION 表函数
获取缓冲池级别快照	SNAPBP 管理视图和 SNAP_GET_BP 表函数
获取表空间级别快照	SNAPTBSP 管理视图和 SNAP_GET_TBSP 表函数
获取配置信息的表空间级别快照	SNAPTBSP_PART 管理视图和 SNAP_GET_TBSP_PART 表函数
获取容器信息的表空间级别快照	SNAPCONTAINER 管理视图和 SNAP_GET_CONTAINER 表函数
获取停顿者信息的表空间级别快照	SNAPTBSP_QUIESCER 管理视图和 SNAP_GET_TBSP_QUIESCER 表函数
获取表空间映射范围的表空间级别快照	SNAPTBSP_RANGE 管理视图和 SNAP_GET_TBSP_RANGE 表函数
获取表级别快照	SNAPTAB 管理视图和 SNAP_GET_TAB 表函数
获取锁定级别快照	SNAPLOCK 管理视图和 SNAP_GET_LOCK 表函数
获取 SQL 语句高速缓存信息的快照	SNAPDYN_SQL 管理视图和 SNAP_GET_DYN_SQL 表函数

确定上次使用某个数据库对象的日期

上次使用某个对象的日期由上次引用的日期（又称为上次使用的日期）来指示。可提供上次引用索引、程序包、表、表数据分区和具体化查询表 (MQT) 的日期。可以使用上次引用的日期来标识连续长时间未使用的对象，这些对象可能会被视作除去候选项。

上次引用的日期存储在该对象的相应目录表的 LASTUSED 列中，可通过该表的目录视图来访问此日期。目录中的用法信息由一个名为 **db2lused**（运行在数据库目录分区上的 LASTUSED 守护程序）的引擎可分派单元 (EDU) 进行更新。LASTUSED 守护程序每隔 15 分钟便收集所有分区中所有对象的使用信息，并更新相应目录表中的 LASTUSED 列以将信息写入到磁盘。给定对象的目录条目最多每天更新一次，这意味着在经过 24 小时的时间间隔之前，将不会再次检查相同对象。选择 15 分钟时间间隔是为了在最小程度上影响数据库服务器的性能，且此值不能由用户配置。对上次引用日期的更新是异步执行的，因此对象访问不会立即记录在目录中。

注: 如果目录表中的相应行已锁定，那么可能会延迟使用信息的更新，直至下一个 15 分钟收集时间间隔。此外，当数据库取消激活时，在取消激活之前，LASTUSED 守护程序未收集的任何用法信息（例如，自守护程序进行最后轮询后首次访问的任何对象）均无法写入到磁盘。请显式激活数据库，以使此功能行为正常。

当已经连续一段时间（例如，几个月）未使用某个对象时，使用上次引用的日期就有意义。上次引用的日期在下列情况下很有用:

- 表和表数据分区: 可以帮助回收未使用的空间
- 索引: 可以帮助回收未使用的空间，以避免进行不必要的插入和维护操作；它还可以减少索引要考虑的选项数，从而缩短编译时间
- 程序包: 可以帮助检测未使用的程序包版本，可以释放这些版本

- MQT: 可以帮助检测未使用的 MQT 以回收未使用的空间; 或者帮助调查和了解未使用某个 MQT 的原因

下列示例描述了一些特定情况，上次引用的日期在这些情况下很有用:

- 为了能够节省空间和降低维护时间，每年都可以通过检查 SYSCAT.INDEXES 目录视图中的 LASTUSED 列，来检查有关索引的上次使用信息。如果去年未使用某个索引，那么可以将该索引视作待删除索引。因为在某些情况下可能并不需要删除索引，所以最终还是由您决定是否删除某个索引。例如，您可能有一个表，大家都知道只有在紧急情况下或者在很少情况下才访问此表（在这些情况下，能够快速访问此表显得尤其重要）；或者，表的索引可能是唯一的，用来强制施加唯一性约束，即使从未显式使用此索引也是如此。可以将上次使用的日期信息用来帮助您决定是否除去索引。
- 贵公司有一些内部应用程序，先前已将它们部署在数据库中，几个月或几年之后，它们已被替换或者不再使用。已引退的应用程序将创造节省空间的机会。可以使用上次使用的日期信息来标识不再使用、但是在应用程序已引退之后未清除的数据库对象。例如，这些数据库对象可能是一些表，存储用来填充 GUI 的值。可以在 SYSCAT.TABLES 目录视图的 LASTUSED 列中找到这些表的上次使用的日期，并且可以从此日期开始调查可除去以回收空间的表对象。

有关特定数据库对象的目录视图的 LASTUSED 列的更多信息（尤其是哪些操作导致了更新），请参阅下列主题:

- SYSCAT.DATAPARTITIONS 目录视图
- SYSCAT.INDEXES 目录视图
- SYSCAT.PACKAGES 目录视图
- SYSCAT.TABLES 目录视图

第 5 章 建议不要使用的监视工具

建议不要使用运行状况监视器和 Windows Management Instrumentation (WMI)，将来的发行版可能会将其除去。

开始使用 IBM InfoSphere Optim 工具。IBM InfoSphere Optim Performance Manager 提供了一个 Web 界面，可用来隔离和分析典型数据库性能问题。还可查看数据库运行状况摘要并深入了解。有关更多详细信息，请参阅使用 Optim Performance Manager 进行监视（网址为 http://publib.boulder.ibm.com/infocenter/idm/docv3/topic/com.ibm.datatools.perfmgmt.monitor.doc/p_monitor.html）。

运行状况监视器简介

运行状况监视器是一个服务器端工具，它增添的“按异常情况进行管理”功能是通过不断监视实例和活动数据库的运行状况实现的。运行状况监视器还可向数据库管理员 (DBA) 发出有关潜在系统运行状况问题的警报。

运行状况监视器以前摄方式检测可能导致硬件故障或不可接受的系统系统性能或功能的问题。运行状况监视器的前摄特征使用户能够在发现的问题影响系统性能之前解决该问题。

要点: V9.7 中已经不推荐使用“运行状况监视器”、“运行状况指示器”及其相关组件，在以后的发行版中可能会将它们除去。运行状况监视器在 DB2 pureScale 环境中不受支持。 有关更多信息，请参阅『已经不推荐使用运行状况监视器』主题（网址为 http://publib.boulder.ibm.com/infocenter/db2luw/v9r7/topic/com.ibm.db2.luw.wn.doc/doc/i0055045.html）。

运行状况监视器使用运行状况指示器来检查系统状态，以确定是否应该发出警报。可对应警报执行先前配置的操作。运行状况监视器还会将警报记录在管理通知日志中，并通过电子邮件或寻呼机发送通知。这一“按异常情况进行管理”模型对潜在系统运行状况问题生成警报而不需要活动监视，从而让 DBA 从事更有价值的工作。

运行状况监视器定期收集有关系统运行状况的信息，这对整体性能的影响非常小。它不会打开任何快照监视开关来收集信息。

运行状况指示器

运行状况监视器使用运行状况指示器来评估数据库管理器性能或数据库性能特定方面的运行状况。运行状况指示器测量特定种类数据库对象（例如表空间）某些方面的运行状况。对度量应用条件以确定运行状况。应用的条件视运行状况指示器类型而定。运行状况是否正常是根据生成警报的条件确定的。

要点: V9.7 中已经不推荐使用“运行状况监视器”、“运行状况指示器”及其相关组件，在以后的发行版中可能会将它们除去。运行状况监视器在 DB2 pureScale 环境中不受支持。 有关更多信息，请参阅『已经不推荐使用运行状况监视器』主题（网址为 http://publib.boulder.ibm.com/infocenter/db2luw/v9r7/topic/com.ibm.db2.luw.wn.doc/doc/i0055045.html）。

运行状况监视器返回三种类型的运行状况指示器:

- **基于阈值**的指示器是对象行为的（连续值范围）统计信息度量。警告阈值和警报阈值定义了正常、警告和警报范围的边界或区域。基于阈值的运行状况指示器有三种有效状态: “正常”、“警告”或“警报”。
- **基于状态**的指示器是有限状态集（包含一个对象的两种或更多种不同状态）度量，该状态集定义了数据库对象或资源的工作是否正常。其中一种状态表示情况正常，所有其他状态都被视为不正常。基于状态的运行状况指示器有两种有效状态: “正常”和“注意”。
- **基于集合状态**的指示器是数据库级度量，它们代表数据库中一个或多个对象的聚集状态。为该集合中的每个对象捕获数据，那些对象中最严重的情况以聚集状态表示。如果该集合中有一个或多个对象处于要求发出警报的状态，运行状况指示器就会显示“注意”状态。基于集合状态的运行状况指示器有两种有效状态: “正常”和“注意”。

在实例级、数据库级、表空间级和表空间容器级都存在运行状况指示器。

可通过 CLP 或 API 来访问运行状况监视器信息。可以通过这些工具来配置运行状况指示器。

当从正常状态切换到非正常状态，或者运行状况指示器值进入警告或警报区域（基于已定义的阈值边界），就会生成警报。有三种类型的警报: 注意、警告和警报。

- 对于量度不同状态的运行状况指示器来说，如果注册了非正常状态，就会发出注意警报。
- 对于量度连续值范围的运行状况指示器来说，阈值定义了正常状态、警告状态和警报状态的边界或区域。例如，如果值进入定义警报区域的阈值范围，就会发出警报类型的警报以指示问题需要立即加以注意。

对于给定的运行状况指示器，运行状况监视器仅在特定警报条件第一次出现时发送通知并运行操作。如果运行状况指示器一直处于特定警报条件下，就不会发送更多通知，也不进一步运行操作。如果运行状况指示器更改了警报条件，或者回到正常状态并重新进入警报条件，就会发送新通知并运行操作。

下表显示了不同刷新时间间隔的运行状况指示器示例以及运行状况监视器对运行状况指示器状态的响应。此示例使用缺省警告阈值和警报阈值，它们分别是 80% 和 90%。

表 120. 不同刷新时间间隔的运行状况指示器条件

刷新时间间隔	ts.ts_util（表空间利用率）运行状况指示器的值	ts.ts_util 运行状况指示器状态	运行状况监视器响应
1	80	警告	发送警告通知，运行警告警报条件的操作
2	81	警告	不发送通知，不运行操作
3	75	正常	不发送通知，不运行操作
4	85	警告	发送警告通知，运行警告警报条件的操作
5	90	警报	发送警报通知，运行警报条件的操作

运行状况指示器处理周期

下图说明运行状况指示器的求值过程。每次经历运行状况指示器特定的刷新时间间隔时，就会运行这一组步骤。

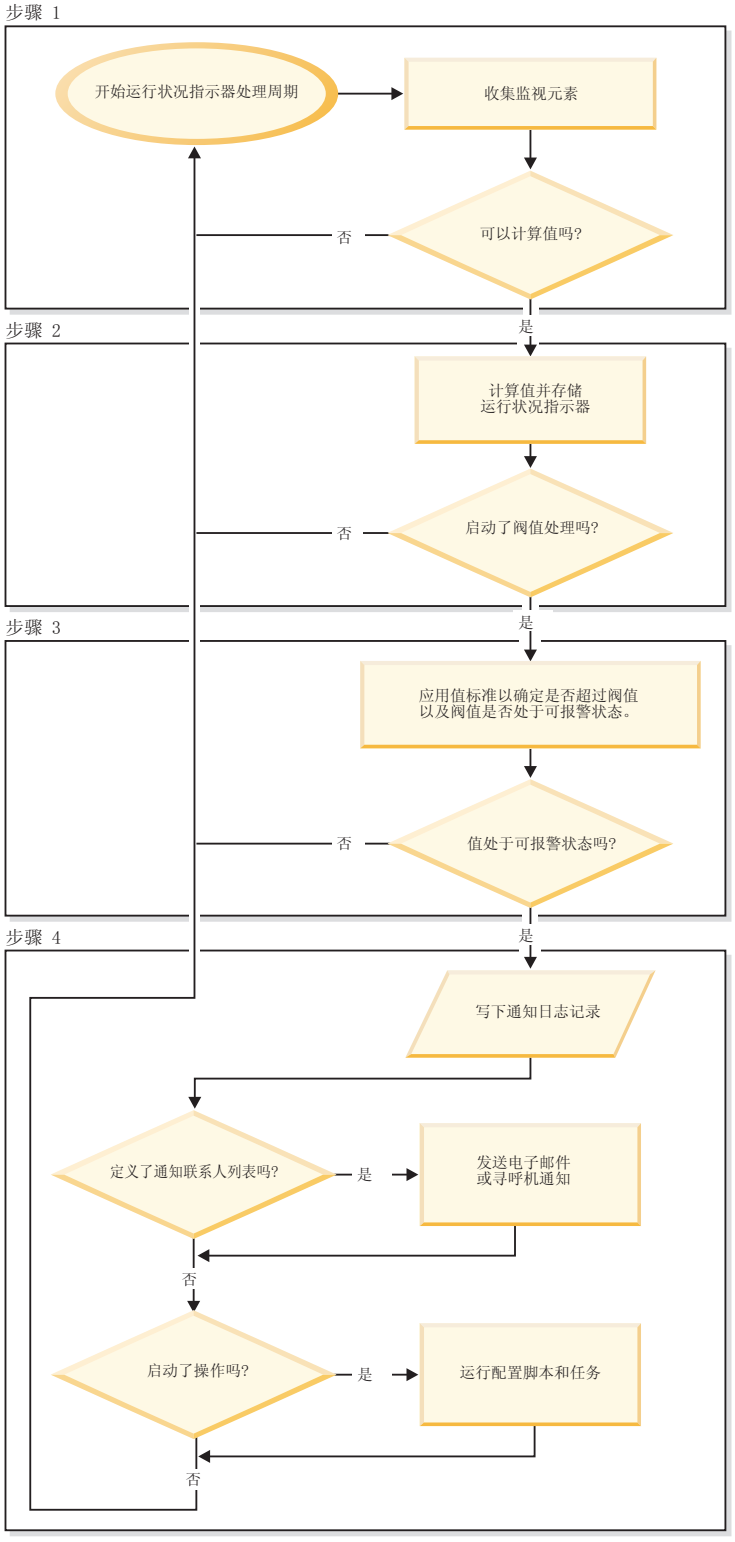


图 8. 运行状况指示器处理周期

注:

1. NOTIFYLEVEL 数据库管理器配置参数控制是将警报通知发送至 DB2 管理通知日志还是发送至所有定义的联系人。警报通知的最低严重性级别需要为 2。要发送警告和注意警报，最低严重性级别需要为 3。

运行状况指示器格式

对运行状况指示器所收集数据的描述。

运行状况指示器的记录是以如下标准格式描述的:

标识 运行状况指示器的名称。此标识用于 CLP 中的配置。

运行状况监视器级别
运行状况监视器捕获运行状况指示器的级别。

类别 运行状况指示器的类别。

类型 运行状况指示器的类型。该类型有四个可能的值:

- 基于阈值上限，其中警报级数为: 正常，警告，警报
- 基于阈值下限
- 基于状态，其中一种状态表示情况正常，所有其他状态都被视为不正常
- 基于集合状态，其中状态基于集合中对象的状态聚集

单位 运行状况指示器中度量的数据单位，如百分比。此项对基于状态或集合状态的运行状况指示器不适用。

运行状况指示器摘要

下表按类别分组列示所有运行状况指示器。

表 121. 数据库自动存储器利用率运行状况指示器

名称	标识	其他信息
数据库自动存储器利用率	db.auto_storage_util	第 426 页的『db.auto_storage_util -“数据库自动存储器利用率”运行状况指示器』

表 122. 表空间存储器运行状况指示器

名称	标识	其他信息
表空间自动调整大小状态	ts.ts_auto_resize_status	第 427 页的『ts.ts_auto_resize_status -“表空间自动调整大小状态”运行状况指示器』
自动调整大小表空间利用率	ts.ts_util_auto_resize	第 428 页的『ts.ts_util_auto_resize -“自动调整表空间大小利用率”运行状况指示器』
表空间利用率	ts.ts_util	第 428 页的『ts.ts_util -“表空间利用率”』
表空间容器利用率	tsc.tscont_util	第 429 页的『tsc.tscont_util -“表空间容器利用率”』
表空间运作状态	ts.ts_op_status	第 430 页的『ts.ts_op_status -“表空间操作状态”』
表空间容器运作状态	tsc.tscont_op_status	第 430 页的『tsc.tscont_op_status -“表空间容器操作状态”』

表 122. 表空间存储器运行状况指示器 (续)

名称	标识	其他信息
表空间自动调整大小状态	ts.ts_auto_resize_status	第 427 页的『ts.ts_auto_resize_status -“表空间自动调整大小状态”运行状况指示器』

表 123. 排序运行状况指示器

名称	标识	其他信息
专用排序内存利用率	db2.sort_privmem_util	第 431 页的『db2.sort_privmem_util -“专用排序内存利用率”』
共享排序内存利用率	db.sort_shrmem_util	第 432 页的『db.sort_shrmem_util -“共享排序内存利用率”』
溢出排序百分比	db.spilled_sorts	第 432 页的『db.spilled_sorts -“溢出排序百分比”』
长期共享排序内存利用率	db.max_sort_shrmem_util	第 433 页的『db.max_sort_shrmem_util -“长期共享排序内存利用率”』

表 124. 数据库管理器运行状况指示器

名称	标识	其他信息
实例运作状态	db2.db2_op_status	第 433 页的『db2.db2_op_status -“实例操作状态”』
实例最高严重性警报状态	–	第 434 页的『实例最高级别严重性警报状态』

表 125. 数据库运行状况指示器

名称	标识	其他信息
数据库运作状态	db.db_op_status	第 435 页的『db.db_op_status -“数据库操作状态”』
数据库最高严重性警报状态	–	第 435 页的『“数据库最高级别严重性警报状态”』

表 126. 维护运行状况指示器

名称	标识	其他信息
需要重组	db.tb_reorg_req	第 435 页的『db.tb_reorg_req -“需要重组”』
必需的统计信息收集运行状况指示器	db.tb_runstats_req	第 436 页的『db.tb_runstats_req -“需要收集统计信息”』
必需的数据库备份	db.db_backup_req	第 437 页的『db.db_backup_req -“需要备份数据库”』

表 127. 高可用性灾难恢复运行状况指示器

名称	标识	其他信息
HADR 运作状态运行状况指示器	db.hadr_op_status	第 437 页的『db.hadr_op_status -“HADR 操作状态”』

表 127. 高可用性灾难恢复运行状况指示器 (续)

名称	标识	其他信息
HADR 日志延迟运行状况指示器	db.hadr_delay	第 438 页的『db.hadr_delay -“HADR 日志延迟”』

表 128. 日志记录运行状况指示器

名称	标识	其他信息
日志利用率	db.log_util	第 438 页的『db.log_util -“日志利用率”』
日志文件系统利用率	db.log_fs_util	第 438 页的『db.log_fs_util -“日志文件系统利用率”』

表 129. 应用程序并行运行状况指示器

名称	标识	其他信息
死锁率	db.deadlock_rate	第 439 页的『db.deadlock_rate -“死锁率”』
锁定列表利用率	db.locklist_util	第 440 页的『db.locklist_util -“锁定列表利用率”』
锁定升级率	db.lock_escal_rate	第 440 页的『db.lock_escal_rate -“锁定升级率”』
等待锁定的应用程序的百分比	db.apps_waiting_locks	第 441 页的『db.apps_waiting_locks -“等待锁定的应用程序的百分比”』

表 130. 程序包高速缓存、目录高速缓存以及工作空间运行状况指示器

名称	标识	其他信息
目录高速缓存命中率	db.catcache_hitratio	第 442 页的『db.catcache_hitratio -“目录高速缓存命中率”』
程序包高速缓存命中率	db.pkgcache_hitratio	第 442 页的『db.pkgcache_hitratio -“程序包高速缓存命中率”』
共享工作空间命中率	db.shrworkspace_hitratio	第 443 页的『db.shrworkspace_hitratio -“共享工作空间命中率”』

表 131. 内存运行状况指示器

名称	标识	其他信息
监视器堆利用率	db2.mon_heap_util	第 443 页的『db2.mon_heap_util -“监视器堆利用率”』
数据库堆利用率	db.db_heap_util	第 444 页的『db.db_heap_util -“数据库堆利用率”』

表 132. 联合运行状况指示器

名称	标识	其他信息
昵称状态	db.fed_nicknames_op_status	第 444 页的『db.fed_nicknames_op_status -“昵称状态”』

表 132. 联合运行状况指示器 (续)

名称	标识	其他信息
数据源服务器状态	db.fed_servers_op_status	第 445 页的『db.fed_servers_op_status -“数据源服务器状态”』

表空间存储器运行状况指示器:

DMS 表空间的运行状况指示器:

可根据表空间的特征来确定与 DMS 表空间有关的运行状况指示器。

要点: V9.7 中已经不推荐使用“运行状况监视器”、“运行状况指示器”及其相关组件，在以后的发行版中可能会将它们除去。运行状况监视器在 DB2 pureScale 环境中不受支持。 有关更多信息，请参阅『已经不推荐使用运行状况监视器』主题（网址为 http://publib.boulder.ibm.com/infocenter/db2luw/v9r7/topic/com.ibm.db2.luw.wn.doc/doc/i0055045.html）。

此表描述根据表空间的特征来看与 DMS 表空间有关的表空间运行状况指示器:

表 133. DMS 表空间的相关表空间运行状况指示器

表空间特征	定义的最大表空间大小	未定义的最大表空间大小
Automatic resize enabled = Yes	ts.ts_util_auto_resize - 跟踪相对于定义的最大值而言使用的空间在表空间中所占的百分比。警报指示表空间很快就会变满并且需要您的干预。只要将最大大小设置为合理的值（即最大大小指定的空间量存在），它就是此配置最重要的运行状况指示器。 ts.ts_util - 跟踪当前分配的表空间存储器使用情况。警报可能不需要您的干预就能够解决任何问题，原因是表空间在变满时将尝试增加大小。 ts.ts_auto_resize_status - 跟踪调整大小尝试的运行状况。警报指示表空间调整大小失败（即表空间已满）。	ts.ts_util_auto_resize - 不适用。未对表空间大小指定上限。 ts.ts_util - 跟踪当前分配的表空间存储器使用情况。警报可能不需要您的干预就能够解决任何问题，原因是表空间将尝试增加大小。 ts.ts_auto_resize_status - 跟踪调整大小尝试的运行状况。警报指示表空间调整大小失败（即表空间已满）。 **注**: 如果使用自动存储器定义了 DMS 表空间并且未指定最大大小，那么还应注意 db.auto_storage_util 运行状况指示器。此运行状况指示器跟踪与数据库存储器路径相关联的空间的利用率。当此空间变满时，表空间将无法增长。这可能导致出现表空间已满的情况。
Automatic resize enabled = No	配置无效。最大表空间大小仅对能够自动调整大小的表空间有效。	ts.ts_util_auto_resize - 不适用。表空间将不尝试调整大小。 ts.ts_util - 跟踪当前分配的表空间存储器使用情况。警报指示表空间已满情况并且需要您立即干预。表空间自身不会尝试调整大小。 ts.ts_auto_resize_status - 不适用。表空间将不尝试调整大小。

db.auto_storage_util -“数据库自动存储器利用率”运行状况指示器:

指示已定义的数据库存储器路径的存储器消耗情况。

要点: V9.7 中已经不推荐使用“运行状况监视器”、“运行状况指示器”及其相关组件，在以后的发行版中可能会将它们除去。运行状况监视器在 DB2 pureScale 环境中不受支持。 有关更多信息，请参阅『已经不推荐使用运行状况监视器』主题（网址为 http://publib.boulder.ibm.com/infocenter/db2luw/v9r7/topic/com.ibm.db2.luw.wn.doc/doc/i0055045.html）。

标识 db.auto_storage_util

运行状况监视器级别
数据库

类别 数据库

类型 基于阈值上限

单位 百分比

创建自动存储器表空间时，将在数据库存储器路径上为这些表空间自动分配容器。如果定义了数据库存储器路径的任何文件系统上没有更多空间，那么自动存储器表空间将无法增加大小，从而可能会变满。

使用以下公式计算指示器:

```
(db.auto_storage_used / db.auto_storage_total) * 100
```

其中:

- *db.auto_storage_used* 是数据库存储器路径中标识的所有物理文件系统中使用的空间总和
- *db.auto_storage_total* 是数据库存储器路径列表中标识的所有物理文件系统中使用的总空间总和

数据库自动存储器路径利用率以数据库存储器路径文件系统上消耗空间的百分比来度量，其中高百分比指示未达到此指示器的最优运行状况。

对此运行状况指示器返回的“其他信息”行中的“表空间将满的时间”是对达到表空间的最大大小还剩余多少时间的预测。

用法说明
如果使用存储器组，那么此运行状况指示器仅指示缺省存储器组中的已定义数据库存储器路径的存储器消耗情况。

ts.ts_auto_resize_status -“表空间自动调整大小状态”运行状况指示器:

此运行状况指示器标识对能够自动调整大小的 DMS 表空间执行表空间调整大小操作是否成功。当能够自动调整大小的 DMS 表空间未能增加大小时，它会很快变满。此情况可能是因为定义了表空间容器的文件系统上缺少可用空间造成的，也可能是表空间自动调整大小设置造成的。例如，可能已经达到定义的最大大小，或者增加量可能设置得太高而导致余下可用空间无法容纳。

要点: V9.7 中已经不推荐使用“运行状况监视器”、“运行状况指示器”及其相关组件，在以后的发行版中可能会将它们除去。运行状况监视器在 DB2 pureScale 环境中不受支

持。 有关更多信息，请参阅『已经不推荐使用运行状况监视器』主题（网址为 http://publib.boulder.ibm.com/infocenter/db2luw/v9r7/topic/com.ibm.db2.luw.wn.doc/doc/i0055045.html）。

标识 ts.ts_auto_resize_status

运行状况监视器级别
表空间

类别 表空间存储器

类型 基于状态

单位 不适用

ts.ts_util_auto_resize -"自动调整表空间大小利用率"运行状况指示器:

此运行状况指示器跟踪每个 DMS 表空间的存储器消耗情况，这些 DMS 表空间已经定义了最大大小，并且可以自动调整大小。达到最大大小时，那么认为 DMS 表空间已满。

要点: V9.7 中已经不推荐使用"运行状况监视器"、"运行状况指示器"及其相关组件，在以后的发行版中可能会将它们除去。运行状况监视器在 DB2 pureScale 环境中不受支持。 有关更多信息，请参阅『已经不推荐使用运行状况监视器』主题（网址为 http://publib.boulder.ibm.com/infocenter/db2luw/v9r7/topic/com.ibm.db2.luw.wn.doc/doc/i0055045.html）。

标识 ts.ts_util_auto_resize

运行状况监视器级别
表空间

类别 表空间存储器

类型 基于阈值上限

单位 百分比

使用以下公式计算指示器:

```
((ts.used * ts.page_size) / ts.max_size) * 100
```

其中:

- *ts.used* 是 第 1214 页的『tablespace_used_pages -"表空间中的已使用页数"监视元素』的值
- *ts.page_size* 是 第 1202 页的『tablespace_page_size -"表空间页大小"监视元素』 的值
- *ts.max_size* 是 第 1199 页的『tablespace_max_size -"最大表空间大小"』 的值

自动调整大小 DMS 表空间利用率是用消耗的最大表空间存储器所占的百分比度量的。高百分比指示表空间接近已满程度。此指示符的附加信息中包括的短期增长率和长期增长率可用来确定，当前增长率是短期畸变还是与长期增长一致。

对此运行状况指示器返回的"其他信息"行中的"表空间将满的时间"是对达到表空间的最大大小还剩余多少时间的预测。

ts.ts_util -"表空间利用率":

此运行状况指示器跟踪每个 DMS 表空间的存储器消耗情况。

要点: V9.7 中已经不推荐使用“运行状况监视器”、“运行状况指示器”及其相关组件，在以后的发行版中可能会将它们除去。运行状况监视器在 DB2 pureScale 环境中不受支持。 有关更多信息，请参阅『已经不推荐使用运行状况监视器』主题（网址为 http://publib.boulder.ibm.com/infocenter/db2luw/v9r7/topic/com.ibm.db2.luw.wn.doc/doc/i0055045.html）。

标识 ts.ts_util

运行状况监视器级别
表空间

类别 表空间存储器

类型 基于阈值上限

单位 百分比

当所有容器已满时，那么认为 DMS 表空间已满。

如果在表空间上启用自动调整大小，那么不会评估此运行状况指示器。相反，对于表空间存储器监视，数据库自动存储器利用率 **db.auto_storage_util** 运行状况指示器和表空间自动调整大小状态 **ts.ts_auto_resize_status** 运行状况指示器是相关的。如果对此表空间定义了最大大小，那么自动调整大小表空间利用率 **ts.ts_util_auto_resize** 运行状况指示器还将可用。如果需要，还可从 TBSP_UTILIZATION 管理视图的 TBSP_UTILIZATION_PERCENT 列检索表空间利用率百分比。

使用以下公式计算指示器:

```
(ts.used / ts.usable) * 100
```

其中:

- *ts.used* 是 第 1214 页的『tablespace_used_pages -“表空间中的已使用页数”监视元素』的值
- *ts.usable* 是 第 1213 页的『tablespace_usable_pages -“表空间中的可用页数”监视元素』的值

表空间利用率以消耗空间的百分比来度量，其中高百分比指示未达到此指示器的最优运行状况。

此指示符的附加信息中包括的短期增长率和长期增长率可用来确定，当前增长率是短期畸变还是与长期增长一致。

对此运行状况指示器返回的“其他信息”行中的“表空间将满的时间”是对达到表空间的最大大小还剩余多少时间的预测。

tsc.tscont_util -“表空间容器利用率”:

此运行状况指示器跟踪未使用自动存储器的每个 SMS 表空间的存储器消耗情况。

要点: V9.7 中已经不推荐使用“运行状况监视器”、“运行状况指示器”及其相关组件，在以后的发行版中可能会将它们除去。运行状况监视器在 DB2 pureScale 环境中不受支

持。 有关更多信息，请参阅『已经不推荐使用运行状况监视器』主题（网址为 http://publib.boulder.ibm.com/infocenter/db2luw/v9r7/topic/com.ibm.db2.luw.wn.doc/doc/i0055045.html）。

标识 tsc.tscont_util

运行状况监视器级别
表空间容器

类别 表空间存储器

类型 基于阈值上限

单位 百分比

如果对其定义容器的任何文件系统上都没有更多空间，那么认为 SMS 表空间已满。

如果文件系统上没有可用空间可供扩展 SMS 容器，那么表示关联表空间已满。

可对在用完可用空间的文件系统上定义的每个容器发出警报。

使用以下公式计算指示器:

```
(fs.used / fs.total)*100
```

其中 `fs` 是容器所在的文件系统。

SMS 表空间利用率以消耗空间的百分比来度量，其中高百分比指示未达到此指示器的最优运行状况。

此指示符的附加信息中包括的短期增长率和长期增长率可用来确定，当前增长率是短期畸变还是与长期增长一致。

对此运行状况指示器返回的“其他信息”行中的“表空间将满的时间”是对达到表空间的最大大小还剩余多少时间的预测。

ts.ts_op_status -“表空间操作状态”:

表空间的状态可以限制可执行的活动或任务。从正常状态切换至另一状态可能生成“注意”警报。

要点: V9.7 中已经不推荐使用“运行状况监视器”、“运行状况指示器”及其相关组件，在以后的发行版中可能会将它们除去。运行状况监视器在 DB2 pureScale 环境中不受支持。 有关更多信息，请参阅『已经不推荐使用运行状况监视器』主题（网址为 http://publib.boulder.ibm.com/infocenter/db2luw/v9r7/topic/com.ibm.db2.luw.wn.doc/doc/i0055045.html）。

标识 ts.ts_op_status

运行状况监视器级别
表空间

类别 表空间存储器

类型 基于状态

单位 不适用

tsc.tscont_op_status -“表空间容器操作状态”:

此运行状况指示器跟踪表空间容器的可访问性。容器的可访问性可以限制可执行的活动或任务。如果容器不可访问，那么可能生成“注意”警报。

要点: V9.7 中已经不推荐使用“运行状况监视器”、“运行状况指示器”及其相关组件，在以后的发行版中可能会将它们除去。运行状况监视器在 DB2 pureScale 环境中不受支持。 有关更多信息，请参阅『已经不推荐使用运行状况监视器』主题（网址为 http://publib.boulder.ibm.com/infocenter/db2luw/v9r7/topic/com.ibm.db2.luw.wn.doc/doc/i0055045.html）。

标识 tsc.tscont_op_status

运行状况监视器级别
表空间容器

类别 表空间存储器

类型 基于状态

单位 不适用

排序运行状况指示器:

db2.sort_privmem_util -“专用排序内存利用率”:

此指示器跟踪专用排序内存的利用率。如果 db2.sort_heap_allocated（系统监视元素）>= *sheapthres*（DBM 配置参数），那么排序可能未成为 *sortheap* 参数定义的已满排序堆，并且可能会生成警报。

要点: V9.7 中已经不推荐使用“运行状况监视器”、“运行状况指示器”及其相关组件，在以后的发行版中可能会将它们除去。运行状况监视器在 DB2 pureScale 环境中不受支持。 有关更多信息，请参阅『已经不推荐使用运行状况监视器』主题（网址为 http://publib.boulder.ibm.com/infocenter/db2luw/v9r7/topic/com.ibm.db2.luw.wn.doc/doc/i0055045.html）。

标识 db2.sort_privmem_util

运行状况监视器级别
数据库

类别 排序

类型 基于阈值上限

单位 百分比

如果有足够的堆空间来执行排序，并且排序没有不必要的溢出，那么认为排序状态良好。

使用以下公式计算指示器:

```
(db2.sort_heap_allocated / sheapthres)*100
```

“阈值后排序数”快照监视元素测量超过排序堆阈值后请求堆的排序数。此指示器的值显示在“其他详细信息”中，对此运行状况指示器指示问题的严重程度。

“使用的最大专用排序内存”快照监视元素保留实例的专用排序内存高水位标记。此指示器的值显示在“其他信息”中，指示自上次回收实例后在任一时间点使用的最大专用排序内存量。此值可用来帮助确定 *sheapthres* 的适当值。

db.sort_shrmem_util -"共享排序内存利用率":

此指示器跟踪共享排序内存的利用率。*sheapthres_shr* 数据库配置参数是硬限制。如果分配接近该限制，那么会生成警报。

要点: V9.7 中已经不推荐使用"运行状况监视器"、"运行状况指示器"及其相关组件，在以后的发行版中可能会将它们除去。运行状况监视器在 DB2 pureScale 环境中不受支持。 有关更多信息，请参阅『已经不推荐使用运行状况监视器』主题（网址为 http://publib.boulder.ibm.com/infocenter/db2luw/v9r7/topic/com.ibm.db2.luw.wn.doc/doc/i0055045.html）。

标识 db.sort_shrmem_util

运行状况监视器级别
数据库

类别 排序

类型 基于阈值上限

单位 百分比

如果有足够的堆空间来执行排序，并且排序没有不必要的溢出，那么认为排序状态良好。

使用以下公式计算指示器:

```
(db.sort_shrheap_allocated / sheapthres_shr)*100
```

注意，如果 *sheapthres_shr* 设置为 0，那么 *sheapthres* 充当共享排序堆阈值。

"使用的最大共享排序内存"快照监视元素保留数据库的共享排序内存高水位标记。此指示器的值显示在"其他信息"中，指示自数据库处于活动状态后在任一时间点使用的最大共享排序内存量。此值可用来帮助确定共享排序内存阈值的适当值。

考虑使用自调整内存功能，以根据当前工作负载的需要自动分配排序内存资源。如果对排序内存区启用自调整内存功能，那么应配置此运行状况指示器以禁用阈值检查。

db.spilled_sorts -"溢出排序百分比":

溢出至磁盘的排序可能导致严重的性能下降。如果发生这种情况，那么会生成警报。

要点: V9.7 中已经不推荐使用"运行状况监视器"、"运行状况指示器"及其相关组件，在以后的发行版中可能会将它们除去。运行状况监视器在 DB2 pureScale 环境中不受支持。 有关更多信息，请参阅『已经不推荐使用运行状况监视器』主题（网址为 http://publib.boulder.ibm.com/infocenter/db2luw/v9r7/topic/com.ibm.db2.luw.wn.doc/doc/i0055045.html）。

标识 db.spilled_sorts

运行状况监视器级别
数据库

类别 排序

类型 基于阈值上限

单位 百分比

如果有足够的堆空间来执行排序，并且排序没有不必要的溢出，那么认为排序状态良好。

使用以下公式计算指示器:

```
(db.sort_overflows_t - db.sort_overflows_t-1) /
(db.total_sorts_t - db.total_sorts_t-1) * 100
```

其中 *t* 是当前快照，而 *t-1* 是 1 小时以前的快照。系统监视元素 db.sort_overflows（基于 sort_overflows 监视元素）是用完排序堆并且可能需要磁盘空间以供临时存储器使用的排序总数。元素 db.total_sorts（基于 total_sorts 监视元素）是已执行的排序总数。

考虑使用自调整内存功能，以根据当前工作负载的需要自动分配排序内存资源。如果对排序内存区启用自调整内存功能，那么应配置此运行状况指示器以禁用阈值检查。

db.max_sort_shrmem_util -"长期共享排序内存利用率":

此指示器跟踪配置过度的共享排序堆，了解能否释放某些资源以用于 DB2 数据库系统中的其他位置。

要点: V9.7 中已经不推荐使用"运行状况监视器"、"运行状况指示器"及其相关组件，在以后的发行版中可能会将它们除去。运行状况监视器在 DB2 pureScale 环境中不受支持。 有关更多信息，请参阅『已经不推荐使用运行状况监视器』主题（网址为 http://publib.boulder.ibm.com/infocenter/db2luw/v9r7/topic/com.ibm.db2.luw.wn.doc/doc/i0055045.html）。

标识 db.max_sort_shrmem_util

运行状况监视器级别
数据库

类别 排序

类型 基于阈值下限

单位 百分比

如果有足够的堆空间来执行排序，并且排序没有不必要的溢出，那么认为排序状态良好。

使用百分比很低时，可能会生成警报。

使用以下公式计算指示器:

```
(db.max_shr_sort_mem / sheapthres_shr)*100
```

系统监视元素 db.max_shr_sort_mem（基于 sort_shrheap_top 监视元素）是共享排序内存使用情况的高水位标记。

考虑使用自调整内存功能，以根据当前工作负载的需要自动分配排序内存资源。如果对排序内存区启用自调整内存功能，那么应配置此运行状况指示器以禁用阈值检查。

数据库管理器（DBMS）运行状况指示器:

db2.db2_op_status -"实例操作状态":

如果实例状态未对正在执行的活动或任务产生限制，那么认为该实例的运行状况正常。

要点: V9.7 中已经不推荐使用“运行状况监视器”、“运行状况指示器”及其相关组件，在以后的发行版中可能会将它们除去。运行状况监视器在 DB2 pureScale 环境中不受支持。 有关更多信息，请参阅『已经不推荐使用运行状况监视器』主题（网址为 http://publib.boulder.ibm.com/infocenter/db2luw/v9r7/topic/com.ibm.db2.luw.wn.doc/doc/i0055045.html）。

标识 db2.db2_op_status

运行状况监视器级别
实例

类别 DBMS

类型 基于状态

单位 不适用

状态可以是下列值中的一个: “活动”、“停顿暂挂”、“已停顿”或“已关闭”。非“活动”状态可能会生成“注意”警报。

如果运行状况指示器进入“关闭”状态，运行状况监视器就无法执行 db2.db2_op_status 运行状况指示器的操作。例如，当指示器所监视的实例由于显式的停止请求或异常终止而进入不活动状态时，指示器就会进入“关闭”状态。如果要让实例在任何异常终止发生后自动重新启动，那么可以配置故障监视器（**db2fm**）以保持该实例的高可用性。

实例最高级别严重性警报状态:

此指示器表示要监视的实例的累积警报状态。实例的警报状态是要监视的实例、实例数据库及数据库对象的最高警报状态。

要点: V9.7 中已经不推荐使用“运行状况监视器”、“运行状况指示器”及其相关组件，在以后的发行版中可能会将它们除去。运行状况监视器在 DB2 pureScale 环境中不受支持。 有关更多信息，请参阅『已经不推荐使用运行状况监视器』主题（网址为 http://publib.boulder.ibm.com/infocenter/db2luw/v9r7/topic/com.ibm.db2.luw.wn.doc/doc/i0055045.html）。

标识 不适用。此运行状况指示器没有配置或建议支持。

运行状况监视器级别
实例

类别 DBMS

类型 基于状态

单位 不适用

警报状态的顺序如下所示:

- 警报
- 警告
- 注意
- 正常

实例的警报状态确定 DB2 数据库系统的整体运行状况。

数据库运行状况指示器:

db.db_op_status -"数据库操作状态":

数据库的状态可以限制可执行的活动或任务。状态可以是下列值中的一个: "活动"、"停顿暂挂"、"已停顿"或"已前滚"。从活动切换至另一状态可能会生成"注意"警报。

要点: V9.7 中已经不推荐使用"运行状况监视器"、"运行状况指示器"及其相关组件，在以后的发行版中可能会将它们除去。运行状况监视器在 DB2 pureScale 环境中不受支持。 有关更多信息，请参阅『已经不推荐使用运行状况监视器』主题（网址为 http://publib.boulder.ibm.com/infocenter/db2luw/v9r7/topic/com.ibm.db2.luw.wn.doc/doc/i0055045.html）。

标识 db.db_op_status

运行状况监视器级别
数据库

类别 数据库

类型 基于状态

单位 不适用

"数据库最高级别严重性警报状态":

此指示器表示要监视的数据库的累积警报状态。数据库的警报状态是数据库及其对象的最高警报状态。

要点: V9.7 中已经不推荐使用"运行状况监视器"、"运行状况指示器"及其相关组件，在以后的发行版中可能会将它们除去。运行状况监视器在 DB2 pureScale 环境中不受支持。 有关更多信息，请参阅『已经不推荐使用运行状况监视器』主题（网址为 http://publib.boulder.ibm.com/infocenter/db2luw/v9r7/topic/com.ibm.db2.luw.wn.doc/doc/i0055045.html）。

标识 不适用。此运行状况指示器没有配置或建议支持。

运行状况监视器级别
数据库

类别 数据库

类型 基于状态

单位 不适用

警报状态的顺序如下所示:

- 警报
- 警告
- 注意
- 正常

维护运行状况指示器:

db.tb_reorg_req -"需要重组":

此运行状况指示器跟踪在数据库中重组表或索引的需要。表或对表定义的所有索引需要重组以消除碎片数据。重组将通过压缩信息并重构行或索引数据完成。此操作的结果是性能得到提高并且释放了表或索引中的空间。

要点: V9.7 中已经不推荐使用“运行状况监视器”、“运行状况指示器”及其相关组件，在以后的发行版中可能会将它们除去。运行状况监视器在 DB2 pureScale 环境中不受支持。 有关更多信息，请参阅『已经不推荐使用运行状况监视器』主题（网址为 http://publib.boulder.ibm.com/infocenter/db2luw/v9r7/topic/com.ibm.db2.luw.wn.doc/doc/i0055045.html）。

标识 db.tb_reorg_req

运行状况监视器级别
数据库

类别 数据库维护

类型 基于集合状态

单位 不适用

通过在自动维护策略中指定要评估的表名，可以过滤此运行状况指示器评估的一组表。可通过使用“自动维护”向导来执行此操作。

可能会生成“注意”警报来指示需要重组。可通过将 AUTO_REORG 数据库配置参数设置为 ON 来自动进行重组。如果已经启用自动重组，那么“注意”警报表示有一个或多个自动重组无法成功完成，或者有需要重组的表，但由于每个数据库分区的表大小超过可以进行脱机重组的最大表大小限定，使重组工作无法执行。有关需要注意的对象的列表，请参阅此运行状况指示器收集的详细信息。

db.tb_runstats_req -“需要收集统计信息”:

此运行状况指示器跟踪收集数据库中的表及其索引的统计信息的需要。需要收集表和对表定义的所有索引的统计信息以缩短查询执行时间。

要点: V9.7 中已经不推荐使用“运行状况监视器”、“运行状况指示器”及其相关组件，在以后的发行版中可能会将它们除去。运行状况监视器在 DB2 pureScale 环境中不受支持。 有关更多信息，请参阅『已经不推荐使用运行状况监视器』主题（网址为 http://publib.boulder.ibm.com/infocenter/db2luw/v9r7/topic/com.ibm.db2.luw.wn.doc/doc/i0055045.html）。

标识 db.tb_runstats_req

运行状况监视器级别
数据库

类别 数据库维护

类型 基于集合状态

单位 不适用

此运行状况指示器认为可使用 SQL 查询来限制这些表。附加信息中的作用域显示此查询用于系统表的子查询子句。

可能会生成“注意”警报来指示需要收集统计信息。可通过将 AUTO_RUNSTATS 数据库配置参数设置为 ON 以自动收集统计信息。如果启用了自动统计信息收集，那么“注意”警报将指示未成功完成一个或多个自动统计信息收集。

db.db_backup_req -“需要备份数据库”:

此运行状况指示器跟踪在数据库中执行备份的需要。恢复策略中应包含定期备份以保护数据，从而避免在硬件或软件故障时丢失数据。

要点: V9.7 中已经不推荐使用“运行状况监视器”、“运行状况指示器”及其相关组件，在以后的发行版中可能会将它们除去。运行状况监视器在 DB2 pureScale 环境中不受支持。 有关更多信息，请参阅『已经不推荐使用运行状况监视器』主题（网址为 http://publib.boulder.ibm.com/infocenter/db2luw/v9r7/topic/com.ibm.db2.luw.wn.doc/doc/i0055045.html）。

标识 db.db_backup_req

运行状况监视器级别
数据库

类别 数据库维护

类型 基于状态

单位 不适用

此运行状况指示器根据上次备份后的经历时间和更改的数据量来确定需要进行数据库备份的时间。

可能会生成“注意”警报来指示需要进行数据库备份。可通过将 AUTO_DB_BACKUP 数据库配置参数设置为 ON 来自动进行数据库备份。如果启用了自动数据库备份，那么“注意”警报将指示未成功完成一个或多个自动数据库备份。

高可用性灾难恢复 (HADR) 运行状况指示器:

db.hadr_op_status -“HADR 操作状态”:

此运行状况指示器跟踪数据库的高可用性灾难恢复 (HADR) 操作状态。主服务器与备用服务器之间的状态可以是下列值中的一个: “已连接”、“拥塞”或“已断开连接”。从“已连接”状态切换至另一状态可能生成“注意”警报。

要点: V9.7 中已经不推荐使用“运行状况监视器”、“运行状况指示器”及其相关组件，在以后的发行版中可能会将它们除去。运行状况监视器在 DB2 pureScale 环境中不受支持。 有关更多信息，请参阅『已经不推荐使用运行状况监视器』主题（网址为 http://publib.boulder.ibm.com/infocenter/db2luw/v9r7/topic/com.ibm.db2.luw.wn.doc/doc/i0055045.html）。

标识 db.hadr_op_status

运行状况监视器级别
数据库

类别 高可用性灾难恢复

类型 基于状态

单位 不适用

db.hadr_delay -"HADR 日志延迟":

此运行状况指示器跟踪主数据库上的数据更改与备用数据库上的更改复制之间的当前平均延迟（以分钟计）。如果延迟值很高，那么在主数据库发生故障后转移至备用数据库时可能导致数据丢失。如果延迟值很高，还可能意味着因为主数据库先于备用数据库而导致需要接管时所花的停机时间更长。

要点: V9.7 中已经不推荐使用"运行状况监视器"、"运行状况指示器"及其相关组件，在以后的发行版中可能会将它们除去。运行状况监视器在 DB2 pureScale 环境中不受支持。 有关更多信息，请参阅『已经不推荐使用运行状况监视器』主题（网址为 http://publib.boulder.ibm.com/infocenter/db2luw/v9r7/topic/com.ibm.db2.luw.wn.doc/doc/i0055045.html）。

标识 db.hadr_delay

运行状况监视器级别
数据库

类别 高可用性灾难恢复

类型 基于阈值上限

单位 分钟

日志记录运行状况指示器:

db.log_util -"日志利用率":

此指示器跟踪在数据库中使用的总活动日志空间量。

要点: V9.7 中已经不推荐使用"运行状况监视器"、"运行状况指示器"及其相关组件，在以后的发行版中可能会将它们除去。运行状况监视器在 DB2 pureScale 环境中不受支持。 有关更多信息，请参阅『已经不推荐使用运行状况监视器』主题（网址为 http://publib.boulder.ibm.com/infocenter/db2luw/v9r7/topic/com.ibm.db2.luw.wn.doc/doc/i0055045.html）。

标识 db.log_util

运行状况监视器级别
数据库

类别 日志记录

类型 基于阈值上限

单位 百分比

日志利用率以消耗空间的百分比来度量，出现高百分比时可能会生成警报。

使用以下公式计算指示器:

```
(db.total_log_used / (db.total_log_used + db.total_log_available)) * 100
```

与日志有关的数据库配置参数的值显示在附加信息中，这些值显示日志的当前分配。附加信息还包括具有最旧活动事务的应用程序的标识。可强制此应用程序释放日志空间。

db.log_fs_util -"日志文件系统利用率":

日志文件系统利用率跟踪事务日志所在的文件系统的充满程度。

要点: V9.7 中已经不推荐使用“运行状况监视器”、“运行状况指示器”及其相关组件，在以后的发行版中可能会将它们除去。运行状况监视器在 DB2 pureScale 环境中不受支持。 有关更多信息，请参阅『已经不推荐使用运行状况监视器』主题（网址为 http://publib.boulder.ibm.com/infocenter/db2luw/v9r7/topic/com.ibm.db2.luw.wn.doc/doc/i0055045.html）。

标识 db.log_fs_util

运行状况监视器级别
数据库

类别 日志记录

类型 基于阈值上限

单位 百分比

如果文件系统上没有空间，那么 DB2 数据库系统可能无法创建新的日志文件。

日志利用率以消耗空间的百分比来度量。如果文件系统中的可用空间量降至最低（即高百分比利用率），那么可能生成警报。

使用以下公式计算此指示器: *(fs.log_fs_used / fs.log_fs_total)*100*，其中 fs 是日志所在的文件系统。

与日志有关的数据库配置参数的值显示在附加信息中，这些值显示日志的当前分配。如果启用了用户出口，那么还会显示其他详细信息。

如果显示在附加详细信息中的“在日志磁盘已满时停止”设置为“是”并且利用率达到 100%，那么应尽快解决所有警报，以限制对不能落实事务的应用程序的影响，直到成功创建日志文件。

应用程序并行运行状况指示器:

db.deadlock_rate -“死锁率”:

死锁率跟踪死锁出现在数据库上的比率以及应用程序遇到争用问题的等级。

要点: V9.7 中已经不推荐使用“运行状况监视器”、“运行状况指示器”及其相关组件，在以后的发行版中可能会将它们除去。运行状况监视器在 DB2 pureScale 环境中不受支持。 有关更多信息，请参阅『已经不推荐使用运行状况监视器』主题（网址为 http://publib.boulder.ibm.com/infocenter/db2luw/v9r7/topic/com.ibm.db2.luw.wn.doc/doc/i0055045.html）。

标识 db.deadlock_rate

运行状况监视器级别
数据库

类别 应用程序并行性

类型 基于阈值上限

单位 每小时产生的死锁数

死锁可能是由下列情况导致的:

- 数据库发生锁定升级
- 在系统生成的行锁定已足够的情况下应用程序显式锁定了表
- 绑定时应用程序使用了不适当的隔离级别
- 目录表已被锁定以供可重复读
- 应用程序正以不同的顺序获取相同的锁定，从而导致死锁。

使用以下公式计算指示器:

$(db.deadlocks_t - db.deadlocks_{t-1})$

其中 *t* 是当前快照，而 *t-1* 是上一个快照，即距获取当前快照之前 60 分钟时获取的快照。

死锁率越高，可能生成警报的争用等级就越高。

db.locklist_util -"锁定列表利用率":

此指示器跟踪要使用的锁定列表内存量。

要点: V9.7 中已经不推荐使用"运行状况监视器"、"运行状况指示器"及其相关组件，在以后的发行版中可能会将它们除去。运行状况监视器在 DB2 pureScale 环境中不受支持。 有关更多信息，请参阅『已经不推荐使用运行状况监视器』主题（网址为 http://publib.boulder.ibm.com/infocenter/db2luw/v9r7/topic/com.ibm.db2.luw.wn.doc/doc/i0055045.html）。

标识 db.locklist_util

运行状况监视器级别
数据库

类别 应用程序并行性

类型 基于阈值上限

单位 百分比

每个数据库有一个锁定列表，锁定列表包含由同时连接至数据库的所有应用程序挂起的锁定。这是对锁定列表内存设置的限制。一旦达到该限制，就会因为下列情况而使得性能下降:

- 锁定升级将行锁定转换为表锁定，从而降低了数据库中的共享对象的并行性。
- 因为应用程序等待有限数目的表锁定，所以应用程序间会出现更多死锁。因此将回滚事务。

当最大锁定请求数达到对数据库设置的限制时，将对应用程序返回错误。

使用以下公式计算指示器:

```
(db.lock_list_in_use / (locklist * 4096)) * 100
```

利用率以消耗内存的百分比来度量，出现高百分比表示状况不良。

考虑使用自调整内存功能，以根据当前工作负载的需要自动分配锁定内存资源。如果对锁定内存区启用自调整内存功能，那么应配置此运行状况指示器以禁用阈值检查。

db.lock_escal_rate -"锁定升级率":

此指示器跟踪锁定已从行锁定升级至表锁定从而影响事务并行性的次数。

要点: V9.7 中已经不推荐使用“运行状况监视器”、“运行状况指示器”及其相关组件，在以后的发行版中可能会将它们除去。运行状况监视器在 DB2 pureScale 环境中不受支持。 有关更多信息，请参阅『已经不推荐使用运行状况监视器』主题（网址为 http://publib.boulder.ibm.com/infocenter/db2luw/v9r7/topic/com.ibm.db2.luw.wn.doc/doc/i0055045.html）。

标识 db.lock_escal_rate

运行状况监视器级别
数据库

类别 应用程序并行性

类型 基于阈值上限

单位 每小时产生的锁定升级数

当应用程序挂起的锁定总数达到可供应用程序使用的最大锁定列表空间量，或者所有应用程序消耗的锁定列表空间达到总锁定列表空间时，锁定将会升级。可用锁定列表空间量由 *maxlocks* 和 *locklist* 数据库配置参数确定。

当应用程序达到允许的最大锁定数并且没有其他要升级的锁定时，应用程序将使用锁定列表中为其他应用程序分配的空间。每个数据库有一个锁定列表，锁定列表包含由同时连接至数据库的所有应用程序挂起的锁定。当整个锁定列表已满时，将发生错误。

使用以下公式计算指示器:

$(db.lock\_escals_{t} - db.lock\_escals_{t-1})$

其中“t”是当前快照，而“t-1”是上一个快照，即距获取当前快照之前 60 分钟时获取的快照。

死锁率越高，可能生成警报的争用等级就越高。

考虑使用自调整内存功能，以根据当前工作负载的需要自动分配锁定内存资源。如果对锁定内存区启用自调整内存功能，那么应配置此运行状况指示器以禁用阈值检查。

db.apps_waiting_locks -“等待锁定的应用程序的百分比”:

此指示器度量所有当前执行的等待锁定的应用程序所占的百分比。

要点: V9.7 中已经不推荐使用“运行状况监视器”、“运行状况指示器”及其相关组件，在以后的发行版中可能会将它们除去。运行状况监视器在 DB2 pureScale 环境中不受支持。 有关更多信息，请参阅『已经不推荐使用运行状况监视器』主题（网址为 http://publib.boulder.ibm.com/infocenter/db2luw/v9r7/topic/com.ibm.db2.luw.wn.doc/doc/i0055045.html）。

标识 db.apps_waiting_locks

运行状况监视器级别
数据库

类别 应用程序并行性

类型 基于阈值上限

单位 百分比

高百分比可能指示应用程序遇到并行性问题，这对性能有负面影响。

使用以下公式计算指示器:

```
(db.locks_waiting / db.appls_cur_cons) *100)
```

程序包高速缓存、目录高速缓存以及工作空间运行状况指示器:

db.catcache_hitratio -"目录高速缓存命中率":

命中率是一个百分比，用于指示目录高速缓存对避免对磁盘上的目录的实际访问所起到的帮助作用。高命中率指示在避免实际磁盘 I/O 访问方面很成功。

要点: V9.7 中已经不推荐使用"运行状况监视器"、"运行状况指示器"及其相关组件，在以后的发行版中可能会将它们除去。运行状况监视器在 DB2 pureScale 环境中不受支持。 有关更多信息，请参阅『已经不推荐使用运行状况监视器』主题（网址为 http://publib.boulder.ibm.com/infocenter/db2luw/v9r7/topic/com.ibm.db2.luw.wn.doc/doc/i0055045.html）。

标识 db.catcache_hitratio

运行状况监视器级别
数据库

类别 程序包和目录高速缓存，以及工作空间

类型 基于阈值下限

单位 百分比

使用以下公式计算指示器:

```
(1-(db.cat_cache_inserts/db.cat_cache_lookups))*100
```

db.pkgcache_hitratio -"程序包高速缓存命中率":

命中率是一个百分比，用于指示程序包高速缓存对避免从系统目录重新装入静态 SQL 的程序包和段以及避免重新编译动态 SQL 语句所起到的帮助作用。高命中率指示在避免这些活动方面很成功。

要点: V9.7 中已经不推荐使用"运行状况监视器"、"运行状况指示器"及其相关组件，在以后的发行版中可能会将它们除去。运行状况监视器在 DB2 pureScale 环境中不受支持。 有关更多信息，请参阅『已经不推荐使用运行状况监视器』主题（网址为 http://publib.boulder.ibm.com/infocenter/db2luw/v9r7/topic/com.ibm.db2.luw.wn.doc/doc/i0055045.html）。

标识 db.pkgcache_hitratio

运行状况监视器级别
数据库

类别 程序包和目录高速缓存，以及工作空间

类型 基于阈值下限

单位 百分比

使用以下公式计算指示器:

```
(1-(db.pkg_cache_inserts/db.pkg_cache_lookups))*100
```

考虑使用自调整内存功能，以根据当前工作负载的需要自动分配程序包高速缓存内存资源。如果对程序包高速缓存内存区启用自调整内存功能，那么应配置此运行状况指示器以禁用阈值检查。

db.shrworkspace_hitratio -"共享工作空间命中率":

命中率是一个百分比，用于指示共享 SQL 工作空间对避免初始化要执行的 SQL 语句的各段所起到的帮助作用。高命中率指示在避免此操作方面很成功。

要点: V9.7 中已经不推荐使用"运行状况监视器"、"运行状况指示器"及其相关组件，在以后的发行版中可能会将它们除去。运行状况监视器在 DB2 pureScale 环境中不受支持。 有关更多信息，请参阅『已经不推荐使用运行状况监视器』主题（网址为 http://publib.boulder.ibm.com/infocenter/db2luw/v9r7/topic/com.ibm.db2.luw.wn.doc/doc/i0055045.html）。

注: 从 DB2 V9.5 开始，不推荐使用 **db.shrworkspace_hitratio** 运行状况指示器。使用此运行状况指示器不会生成错误。但是不会返回有效值。建议不要再使用此指示器，将来的发行版中可能会将其除去。

标识 db.shrworkspace_hitratio

运行状况监视器级别
数据库

类别 程序包和目录高速缓存，以及工作空间

类型 基于阈值下限

单位 百分比

使用以下公式计算指示器:

```
(1-(db.shr_workspace_section_inserts/db.shr_workspace_section_lookups))*100
```

内存运行状况指示器:

db2.mon_heap_util -"监视器堆利用率":

此指示器跟踪基于带有标识 SQLM_HEAP_MONITOR 的内存池的监视器堆内存的消耗。

要点: V9.7 中已经不推荐使用"运行状况监视器"、"运行状况指示器"及其相关组件，在以后的发行版中可能会将它们除去。运行状况监视器在 DB2 pureScale 环境中不受支持。 有关更多信息，请参阅『已经不推荐使用运行状况监视器』主题（网址为 http://publib.boulder.ibm.com/infocenter/db2luw/v9r7/topic/com.ibm.db2.luw.wn.doc/doc/i0055045.html）。

标识 db2.mon_heap_util

运行状况监视器级别
实例

类别 内存

类型 基于阈值上限

单位 百分比

对内存池标识 SQLM_HEAP_MONITOR 使用以下公式计算利用率:

```
(db2.pool_cur_size / db2.pool_config_size) * 100
```

一旦此百分比达到最大值 100%，监视器操作可能会失败。

db.db_heap_util -"数据库堆利用率":

此指示器跟踪基于带有标识 SQLM_HEAP_DATABASE 的内存池的监视器堆内存的消耗。

要点: V9.7 中已经不推荐使用"运行状况监视器"、"运行状况指示器"及其相关组件，在以后的发行版中可能会将它们除去。运行状况监视器在 DB2 pureScale 环境中不受支持。 有关更多信息，请参阅『已经不推荐使用运行状况监视器』主题（网址为 http://publib.boulder.ibm.com/infocenter/db2luw/v9r7/topic/com.ibm.db2.luw.wn.doc/doc/i0055045.html）。

标识 db.db_heap_util

运行状况监视器级别
数据库

类别 内存

类型 基于阈值上限

单位 百分比

对于内存池标识 SQLM_HEAP_DATABASE，使用以下公式计算利用率:

```
(db.pool_cur_size / db.pool_config_size) * 100
```

一旦此百分比达到最大值 100%，查询和操作可能会因为没有堆可用而失败。

联合运行状况指示器:

db.fed_nicknames_op_status -"昵称状态":

此运行状况指示器检查联合数据库中定义的所有昵称以确定是否存在无效昵称。如果数据源对象已删除或者已更改，又或者如果用户映射不正确，那么昵称可能会无效。

要点: V9.7 中已经不推荐使用"运行状况监视器"、"运行状况指示器"及其相关组件，在以后的发行版中可能会将它们除去。运行状况监视器在 DB2 pureScale 环境中不受支持。 有关更多信息，请参阅『已经不推荐使用运行状况监视器』主题（网址为 http://publib.boulder.ibm.com/infocenter/db2luw/v9r7/topic/com.ibm.db2.luw.wn.doc/doc/i0055045.html）。

标识 db.fed_nicknames_op_status

运行状况监视器级别
数据库

类别 联合

类型 基于集合状态

单位 不适用

如果在联合数据库中定义的任何昵称无效，那么可能会生成"注意"警报。有关需要注意的对象的列表，请参阅此运行状况指示器收集的详细信息。

如果此运行状况指示器要检查昵称状态，那么 FEDERATED 数据库管理器参数必须设置为 YES。

db.fed_servers_op_status -"数据源服务器状态":

此运行状况指示器检查联合数据库中定义的所有数据源服务器以确定是否存在不可用的数据源服务器。如果数据源服务器停止、不再存在或者进行了错误的配置，那么数据源服务器可能不可用。

要点: V9.7 中已经不推荐使用"运行状况监视器"、"运行状况指示器"及其相关组件，在以后的发行版中可能会将它们除去。运行状况监视器在 DB2 pureScale 环境中不受支持。 有关更多信息，请参阅『已经不推荐使用运行状况监视器』主题（网址为 http://publib.boulder.ibm.com/infocenter/db2luw/v9r7/topic/com.ibm.db2.luw.wn.doc/doc/i0055045.html）。

标识 db.fed_servers_op_status

运行状况监视器级别
数据库

类别 联合

类型 基于集合状态

单位 不适用

如果在联合数据库中定义的任何昵称无效，那么可能会生成"注意"警报。有关需要注意的对象的列表，请参阅此运行状况指示器收集的详细信息。

如果此运行状况指示器要检查数据源服务器状态，那么 FEDERATED 数据库管理器参数必须设置为 YES。

运行状况监视器接口

要点: V9.7 中已经不推荐使用"运行状况监视器"、"运行状况指示器"及其相关组件，在以后的发行版中可能会将它们除去。运行状况监视器在 DB2 pureScale 环境中不受支持。 有关更多信息，请参阅『已经不推荐使用运行状况监视器』主题（网址为 http://publib.boulder.ibm.com/infocenter/db2luw/v9r7/topic/com.ibm.db2.luw.wn.doc/doc/i0055045.html）。

下表列示 API 的运行状况监视器接口:

表 134. 运行状况监视器: API

监视任务	API
捕获运行状况快照	db2GetSnapshot - 获取快照类为 SQLM_CLASS_HEALTH 的快照
捕获带有集合对象的完整列表的运行状况快照	db2GetSnapshot - 获取快照类为 SQLM_CLASS_HEALTH 并且 agent_id 为 SQLM_HMON_OPT_COLL_FULL 的快照
捕获带有公式、附加信息和历史记录的运行状况快照	db2GetSnapshot - 获取快照类为 SQLM_CLASS_HEALTH_WITH_DETAIL 的快照
捕获带有公式、附加信息、历史记录和集合对象的完整列表的运行状况快照	db2GetSnapshot - 获取快照类为 SQLM_CLASS_HEALTH_WITH_DETAIL 并且 agent_id 为 SQLM_HMON_OPT_COLL_FULL 的快照

表 134. 运行状况监视器: API (续)

监视任务	API
转换自描述数据流	db2ConvMonStream - 转换监视器流
估计运行状况快照的大小	db2GetSnapshotSize - 估计 db2GetSnapshot 输出缓冲区所需的大小

GET HEALTH SNAPSHOT 命令包含在不推荐使用的运行状况监视器组件中。

下表列示 CLP 命令的运行状况监视器接口:

表 135. 运行状况监视器接口: CLP 命令

监视任务	CLP 命令
捕获运行状况快照	GET HEALTH SNAPSHOT 命令
捕获带有公式、附加信息和历史记录的运行状况快照	GET HEALTH SNAPSHOT WITH DETAILS 命令

运行状况监视器 SQL 函数包含在不推荐使用的运行状况监视器组件中。

下表列示 SQL 函数的运行状况监视器接口:

表 136. 运行状况监视器接口: SQL 函数

监视任务	SQL 函数
数据库管理器级别运行状况信息快照	HEALTH_DBM_INFO
数据库管理器级别运行状况指示器快照	HEALTH_DBM_HI
数据库管理器级别运行状况指示器历史记录快照	HEALTH_DBM_HI_HIS
数据库级别运行状况信息快照	HEALTH_DB_INFO
数据库级别运行状况指示器快照	HEALTH_DB_HI
数据库级别运行状况指示器历史记录快照	HEALTH_DB_HI_HIS
数据库级别运行状况指示器集合快照	HEALTH_DB_HIC
数据库级别运行状况指示器集合历史记录快照	HEALTH_DB_HIC_HIS
表空间级别运行状况信息快照	HEALTH_TBS_INFO
表空间级别运行状况指示器快照	HEALTH_TBS_HI
表空间级别运行状况指示器历史记录快照	HEALTH_TBS_HI_HIS
表空间容器级别运行状况信息快照	HEALTH_CONT_INFO
表空间容器级别运行状况指示器快照	HEALTH_CONT_HI
表空间容器级别运行状况指示器历史记录快照	HEALTH_CONT_HI_HIS

运行状况监视器 SQL 表函数:

建议不要使用运行状况监视器 SQL 表函数。

要点: V9.7 中已经不推荐使用“运行状况监视器”、“运行状况指示器”及其相关组件，在以后的发行版中可能会将它们除去。运行状况监视器在 DB2 pureScale 环境中不受支持。 有关更多信息，请参阅『已经不推荐使用运行状况监视器』主题（网址为 http://publib.boulder.ibm.com/infocenter/db2luw/v9r7/topic/com.ibm.db2.luw.wn.doc/doc/i0055045.html）。

下表列示所有快照表函数。每个表函数对应一个运行状况快照请求类型。

表 137. 快照监视器 SQL 表函数

监视器级别	SQL 表函数	返回的信息
数据库管理器	HEALTH_DBM_INFO	有关来自数据库管理器级别的运行状况快照的基本信息
数据库管理器	HEALTH_DBM_HI	来自数据库管理器级别的运行状况指示器信息
数据库管理器	HEALTH_DBM_HI_HIS	来自数据库管理器级别的运行状况指示器历史记录信息
数据库	HEALTH_DB_INFO	有关来自数据库的运行状况快照的基本信息
数据库	HEALTH_DB_HI	来自数据库的运行状况指示器信息
数据库	HEALTH_DB_HI_HIS	来自数据库的运行状况指示器历史记录信息
数据库	HEALTH_DB_HIC	数据库的集合运行状况指示器的集合信息
数据库	HEALTH_DB_HIC_HIS	数据库的集合运行状况指示器的集合历史记录信息
表空间	HEALTH_TBS_INFO	有关数据库的表空间的运行状况快照的基本信息
表空间	HEALTH_TBS_HI	有关数据库的表空间的运行状况指示器信息
表空间	HEALTH_TBS_HI_HIS	有关数据库的表空间的运行状况指示器历史记录信息
表空间	HEALTH_CONT_INFO	有关数据库的容器的运行状况快照的基本信息
表空间	HEALTH_CONT_HI	有关数据库的容器的运行状况指示器信息
表空间	HEALTH_CONT_HI_HIS	有关数据库的容器的运行状况指示器历史记录信息

运行状况监视器 CL 命令:

可通过发出运行状况监视器数据库来获取数据库管理器及其数据库的运行状况信息。

所返回信息表示发出命令时的运行状况状态的快照。

要点: V9.7 中已经不推荐使用“运行状况监视器”、“运行状况指示器”及其相关组件，在以后的发行版中可能会将它们除去。运行状况监视器在 DB2 pureScale 环境中不受支持。 有关更多信息，请参阅『已经不推荐使用运行状况监视器』主题（网址为 http://publib.boulder.ibm.com/infocenter/db2luw/v9r7/topic/com.ibm.db2.luw.wn.doc/doc/i0055045.html）。

下表列示了所有受支持的快照请求类型。

表 138. 快照监视器 CLP 命令

监视器级别	CLP 命令	返回的信息
数据库管理器	`get health snapshot for dbm`	数据库管理器级别信息。
数据库	`get health snapshot for all databases`	数据库级别信息。仅当激活数据库后才会返回信息。

表 138. 快照监视器 CLP 命令 (续)

监视器级别	CLP 命令	返回的信息
数据库	`get health snapshot for database on database-alias`	数据库级别信息。仅当激活数据库后才会返回信息。
数据库	`get health snapshot for all on database-alias`	数据库、表空间和表空间容器信息。仅当激活数据库后才会返回信息。
表空间	`get snapshot for tablespaces on database-alias`	连接到数据库的应用程序已访问的每个表空间的表空间级别信息。并且，返回的信息还包括该表空间中每个表空间容器的运行状况信息。

运行状况监视器 API 请求类型:

下表列示了所有受支持的快照请求类型。

表 139. 快照监视器 API 请求类型

监视器级别	API 请求类型	返回的信息
数据库管理器	SQLMA_DB2	数据库管理器级别信息。
数据库	SQLMA_DBASE_ALL	数据库级别信息。仅当激活数据库后才会返回信息。
数据库	SQLMA_DBASE	数据库级别信息。仅当激活数据库后才会返回信息。
表空间	SQLMA_DBASE_TABLESPACES	连接到数据库的应用程序已访问的每个表空间的表空间级别信息。并且，返回的信息还包括该表空间中每个表空间容器的运行状况信息。

运行状况监视器接口至逻辑数据组的映射

下表列示了所有受支持的运行状况快照请求类型。

表 140. 运行状况监视器接口至逻辑数据组的映射

API 请求类型	CLP 命令	SQL 表函数	逻辑数据组
SQLMA_DB2	`get health snapshot for dbm`	HEALTH_DBM_INFO	db2
		HEALTH_DBM_HI	health_indicator
	`get health snapshot for dbm show detail`	HEALTH_DBM_HI_HIS	health_indicator_history
SQLMA_DBASE	`get health snapshot for database on dbname`	HEALTH_DB_INFO	dbase
		HEALTH_DB_HI	health_indicator
	`get health snapshot for database on dbname show detail`	HEALTH_DB_HI_HIS	health_indicator_history
SQLMA_DBASE with SQLM_HMON_OPT_COLL_FULL in the agent_id	`get health snapshot for database on dbname with full collection`	HEALTH_DB_HIC	health_indicator, hi_obj_list
	`get health snapshot for database on dbname show detail with full collection`	HEALTH_DB_HIC_HIST	health_indicator_history, hi_obj_list

表 140. 运行状况监视器接口至逻辑数据组的映射 (续)

API 请求类型	CLP 命令	SQL 表函数	逻辑数据组
SQLMA_DBASE_ALL	get health snapshot for all databases	HEALTH_DB_INFO	dbase
		HEALTH_DB_HI	health_indicator
	get health snapshot for all databases show detail	HEALTH_DB_HI_HIS	health_indicator_history
SQLMA_DBASE_TABLESPACES	get health snapshot for tablespaces on *dbname*	HEALTH_TS_INFO	tablespace
		HEALTH_TS_HI	health_indicator
		HEALTH_CONT_INFO	tablespace_container
		HEALTH_CONT_HI	health_indicator
	get health snapshot for tablespaces on *dbname* show detail	HEALTH_TS_HI_HIS	health_indicator_history
		HEALTH_CONT_HI_HIS	health_indicator_history

下图显示逻辑数据分组在运行状况快照数据流中可能出现的顺序。

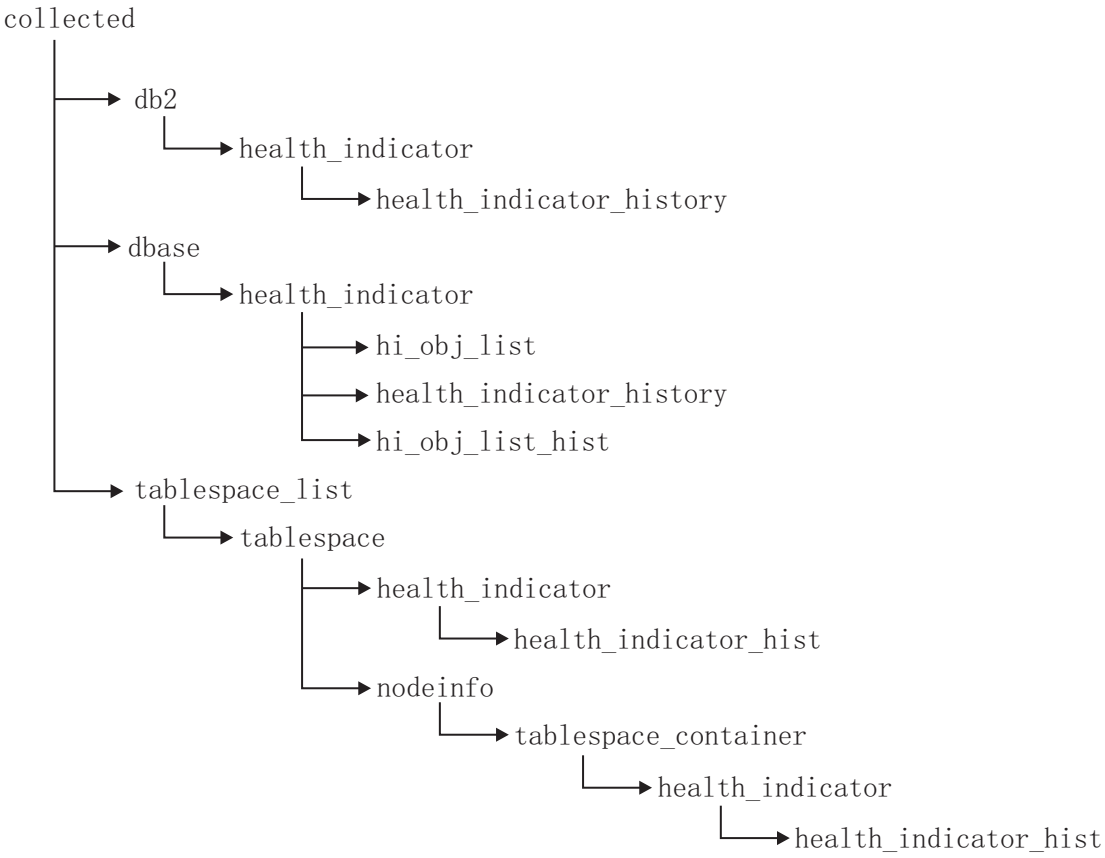


图 9. 运行状况快照逻辑数据分组

启用运行状况警报通知

要在生成警报时启用电子邮件或寻呼机通知，必须设置配置参数并指定联系人信息。

开始之前

DB2 管理服务器（DAS）必须正在联系人列表所在的系统上运行。例如，如果 CONTACT_HOST 配置参数设置为远程系统，那么为了获取要通知有关警报的联系人，DAS 必须在远程系统上运行。

关于此任务

要启用运行状况警报通知:

过程

1. 指定 SMTP_SERVER 参数。 DAS 配置参数 SMTP_SERVER 指定发送电子邮件和寻呼机通知消息时使用的邮件服务器的位置。如果安装 DB2 数据库的系统是作为未授权 SMTP 服务器启用的，那么省略此步骤。
2. 指定 CONTACT_HOST 参数。 DAS 配置参数 CONTACT_HOST 指定本地系统上所有实例的联系人列表的远程位置。通过设置此参数，可在多个系统间共享单个联系人列表。如果想要将联系人列表保留在安装 DB2 数据库的本地系统上，那么省略此步骤。
3. 指定运行状况监视器通知的缺省联系人。为了能够在生成警报时从运行状况监视器启用电子邮件或寻呼机通知，必须指定缺省管理联系人。如果选择不提供此信息，那么不会对警报条件发送通知消息。可在安装期间提供缺省管理联系人信息，也可以将此任务延迟至安装完成之后进行。 如果选择延迟该任务或者想要将更多联系人或组添加至通知列表，那么可以通过 CLP 或 C API 来指定联系人:

   - **要使用 CLP 指定联系人**:

     要将电子邮件联系人定义为运行状况监视器通知的缺省联系人，请发出以下命令:

     ```
     DB2 ADD CONTACT contact_name TYPE EMAIL ADDRESS
        email_address DESCRIPTION 'Default Contact'

     DB2 UPDATE NOTIFICATION LIST ADD CONTACT contact_name
     ```

     有关完整的语法详细信息，请参阅 Command Reference。

   - **要使用 C API 指定联系人**:

     下列 C 代码摘录说明如何定义运行状况通知联系人:

     ```
     ...
     #include <db2ApiDf.h>

     SQL_API_RC rc = 0;
     struct db2AddContactData addContactData;
     struct sqlca sqlca;

     char* userid = "myuser";
     char* password = "pwd";
     char* contact = "DBA1";
     char* email = "dba1@mail.com";
     char* desc = "Default contact";

     memset(&addContactData, '\0', sizeof(addContactData));
     memset (&sqlca, '\0', sizeof(struct sqlca));
     addContactData.piUserid = userid;
     addContactData.piPassword = password;
     ```

```
addContactData.piName = contact;
addContactData.iType = DB2CONTACT_EMAIL;
addContactData.piAddress = email;
addContactData.iMaxPageLength = 0;
addContactData.piDescription = desc;

rc = db2AddContact(db2Version810, &addContactData, &sqlca);

if (rc == 0) {
      db2HealthNotificationListUpdate update;
      db2UpdateHealthNotificationListData data;
      db2ContactTypeData contact;

      contact.pName = contact;
      contact.contactType = DB2CONTACT_EMAIL;

      update.iUpdateType = DB2HEALTHNOTIFICATIONLIST_ADD;
      update.piContact = &contact;

      data.iNumUpdates = 1;
      data.piUpdates = &update;

   rc = db2UpdateHealthNotificationList (db2Version810, &data, &ca);
}
...
```

-

运行状况监视器

运行状况监视器捕获有关数据库管理器、数据库、表空间和表空间容器的信息。

要点: V9.7 中已经不推荐使用“运行状况监视器”、“运行状况指示器”及其相关组件，在以后的发行版中可能会将它们除去。运行状况监视器在 DB2 pureScale 环境中不受支持。 有关更多信息，请参阅『已经不推荐使用运行状况监视器』主题（网址为 http://publib.boulder.ibm.com/infocenter/db2luw/v9r7/topic/com.ibm.db2.luw.wn.doc/doc/i0055045.html）。

运行状况监视器根据从数据库系统监视元素、操作系统和 DB2 数据库检索到的信息计算运行状况指示器的各项指标。仅当数据库活动时，运行状况监视器才能对数据库及其对象计算运行状况指示器的各项指标。可通过使用 **ACTIVATE DATABASE** 命令启动数据库或保持与数据库的永久连接，以便让数据库保持活动状态。

运行状况监视器对每个运行状况指示器保留最多 10 个历史记录。此历史记录存储在 *instance_path*\hmonCache 目录中并且在停止运行状况监视器时除去。当达到最大记录数时，运行状况监视器将自动删除过时的历史记录。

运行状况监视器数据可通过运行状况快照访问。每个运行状况快照按最新刷新时间间隔报告每个运行状况指示器的状态。在检测现有数据库运行状况问题和预测数据库环境可能出现的不良运行状况时，快照非常有用。可以使用 C 或 C++ 应用程序中的 API 或者图形管理工具来从 CLP 捕获运行状况快照。

运行状况监视需要实例连接。如果未使用 **ATTACH TO** 命令建立与实例的连接，那么将创建与本地实例的缺省连接。

在分区数据库环境中，可在实例的任何分区上获取快照，或者使用单个实例连接获取全局快照。全局快照聚集在每个分区上收集到的数据并返回一组值。

使用说明

DB2 数据库的所有版本都支持运行状况监视器。

在 Windows 上，DB2 实例的服务需要在具有 SYSADM 权限的帐户下运行。可在 **db2icrt** 命令上使用 **-u** 选项，或在 Windows 上使用“服务”文件夹然后编辑“登录”属性以使用带有管理员特权的帐户。

运行状况监视器将作为 DB2 受防护方式进程运行。这些进程在 Windows 上显示为 DB2FMP。在其他平台上，运行状况监视器进程显示为 DB2ACD。

DB2 管理服务器必须正在运行状况监视器所在的系统上运行，才能发送通知和运行警报操作。如果使用远程脚本、任务或联系人列表，那么必须同时启动远程系统上的 DB2 管理服务器。

只有创建任务时才需要工具目录数据库。如果不对任何运行状况指示器使用警报任务操作，那么运行状况监视器不需要工具目录数据库。

运行状况指示器数据

运行状况监视器对每个数据库分区上的每个运行状况指示器记录一组数据，包括:

- 运行状况指示器名称
- 值
- 求值时间戳记
- 警报状态
- 公式（如果适用）
- 其他信息（如果适用）
- 最多 10 个最新运行状况指示器求值历史记录。每个历史记录条目捕获导致当前运行状况指示器输出的下列运行状况指示器求值:
  - 值
  - 公式（如果适用）
  - 警报状态
  - 时间戳记

运行状况监视器还会跟踪实例、数据库和表空间级别的最高严重性警报状态。在每一级别，此运行状况指示器表示运行状况指示器在该级别或该级别之下的任何级别存在的最高严重性警报。例如，实例的最高严重性警报状态包括实例、该实例的任何数据库和每个数据库的任何表空间容器上的运行状况指示器。

捕获数据库运行状况快照

使用 SQL 表函数捕获数据库运行状况快照:

可使用 SQL 表函数来捕获数据库运行状况快照。每个可用的运行状况快照表函数对应一个运行状况快照请求类型。

要点: V9.7 中已经不推荐使用“运行状况监视器”、“运行状况指示器”及其相关组件，在以后的发行版中可能会将它们除去。运行状况监视器在 DB2 pureScale 环境中不受支

持。 有关更多信息，请参阅『已经不推荐使用运行状况监视器』主题（网址为 http://publib.boulder.ibm.com/infocenter/db2luw/v9r7/topic/com.ibm.db2.luw.wn.doc/doc/i0055045.html）。

关于此任务

要使用 SQL 表函数来捕获数据库运行状况快照:

过程

1. 标识计划使用的 SQL 表函数。

   SQL 表函数有两个输入参数:

   - VARCHAR(255)，用于数据库名称
   - INT，用于分区号（0 到 999 之间的值）。输入与要监视的分区号相对应的整数。要捕获当前连接的分区的快照，请输入值 -1。要捕获全局快照，请输入值 -2。

   注: 对于此规则，数据库管理器快照 SQL 表函数是一个例外，原因是它们只有一个参数。单一参数用于分区号。如果对数据库名称参数输入 NULL，那么监视器使用连接定义的数据库，表函数就是通过该连接调用的。

2. 发出 SQL 语句。

   以下示例捕获当前连接的分区的基本运行状况快照，以及连接定义的数据库上的基本运行状况快照（通过该连接调用此表函数）:

   ```
   SELECT * FROM TABLE( HEALTH_DB_INFO( cast (NULL as VARCHAR(1)), -1))
                as HEALTH_DB_INFO
   ```

   您也可以从返回的表中选择个别监视元素。返回的表中的每一列对应一个监视元素。相应地，监视元素列名直接对应监视元素名称。以下语句仅返回数据库路径和服务器平台监视元素:

   ```
   SELECT db_path, server_platform
                FROM TABLE( HEALTH_DB_INFO( cast (NULL as VARCHAR(1)), -1 ) )
                as HEALTH_DB_INFO
   ```

使用 CLP 捕获数据库运行状况快照:

可从 CLP 使用 GET HEALTH SNAPSHOT 命令来捕获运行状况快照。该命令语法支持检索运行状况监视器监视的不同对象类型的运行状况快照信息。

要点: V9.7 中已经不推荐使用“运行状况监视器”、“运行状况指示器”及其相关组件，在以后的发行版中可能会将它们除去。运行状况监视器在 DB2 pureScale 环境中不受支持。 有关更多信息，请参阅『已经不推荐使用运行状况监视器』主题（网址为 http://publib.boulder.ibm.com/infocenter/db2luw/v9r7/topic/com.ibm.db2.luw.wn.doc/doc/i0055045.html）。

开始之前

必须具有实例连接才能捕获运行状况快照。如果没有实例连接，那么创建缺省实例连接。要获取远程实例的快照，必须先连接至该实例。

关于此任务

要使用 CLP 捕获数据库运行状况快照

过程

1. 从 CLP 发出带有必需参数的 GET HEALTH SNAPSHOT 命令。

   在以下示例中，将在启动数据库管理器之后立即捕获数据库管理器级别运行状况快照。

   ```
   db2 get health snapshot for dbm
   ```

2. 对于分区数据库系统，可为特定分区捕获专门的数据库快照，或者为所有分区捕获全局的数据库快照。要对特定分区（如分区号 2）上的数据库捕获运行状况快照，请发出以下命令:

   ```
   db2 get health snapshot for db on sample at dbpartitionnum 2
   ```

   要对所有分区上的所有应用程序捕获数据库快照，请发出以下命令:

   ```
   db2 get health snapshot for db on sample global
   ```

   以下命令捕获的运行状况快照带有附加详细信息，包括公式、附加信息和运行状况指示器历史记录:

   ```
   db2 get health snapshot for db on sample show detail
   ```

3. 对于基于集合状态的运行状况指示器，可对所有集合对象捕获数据库快照，而不考虑这些对象的状态。常规 GET HEALTH SNAPSHOT FOR DB 命令返回所有集合对象，这些对象需要针对所有基于集合状态的运行状况指示器的警报。

   要对列示了所有集合对象的数据库捕获运行状况快照，请发出以下命令:

   ```
   db2 get health snapshot for db on sample with full collection
   ```

从客户机应用程序捕获数据库运行状况快照:

可使用 C 或 C++ 应用程序中的快照监视器 API 来捕获运行状况快照。可通过在 db2GetSnapshot API 中指定参数来访问若干不同运行状况快照请求类型。

要点: V9.7 中已经不推荐使用“运行状况监视器”、“运行状况指示器”及其相关组件，在以后的发行版中可能会将它们除去。运行状况监视器在 DB2 pureScale 环境中不受支持。 有关更多信息，请参阅『已经不推荐使用运行状况监视器』主题（网址为 http://publib.boulder.ibm.com/infocenter/db2luw/v9r7/topic/com.ibm.db2.luw.wn.doc/doc/i0055045.html）。

开始之前

必须连接至实例才能捕获运行状况快照。如果没有实例连接，那么创建缺省实例连接。要获取远程实例的快照，必须先连接至该实例。

过程

1. 将 sqlmon.h 和 db2ApiDf.h DB2 库包括在代码中。这些库可在 sqllib\include 目录中找到。

   ```
   #include <db2ApiDf.h>
   #include <sqlmon.h>
   ```

2. 将快照缓冲区单元大小设置为 50 KB。

   ```
   #define SNAPSHOT_BUFFER_UNIT_SZ 51200
   ```

3. 声明 sqlma、sqlca、sqlm_collected 和 db2GetSnapshotData 结构。

```
struct sqlma *pRequestedDataGroups;
struct sqlca sqlca;
memset (&sqlca, '\0', sizeof(struct sqlca));
struct sqlm_collected collectedData;
memset (&sqlm_collected, '\0', sizeof(struct sqlm_collected));
db2GetSnapshotData getSnapshotParam;
memset(&db2GetSnapshotData, '\0', sizeof(db2GetSnapshotData));
```

4. 初始化指针以包含快照缓冲区并确定缓冲区大小。

```
static sqluint32 snapshotBufferSize = SNAPSHOT_BUFFER_UNIT_SZ;
sqluint32 outputFormat;
char *snapshotBuffer;
```

5. 初始化 sqlma 结构，并指定要捕获的快照属于数据库管理器级别信息。

```
pRequestedDataGroups = (struct sqlma *)malloc(SQLMASIZE(1));
memset(&pRequestedDataGroups, '\0', sizeof(struct pRequestedDataGroups));
pRequestedDataGroups->obj_num = 1;
pRequestedDataGroups->obj_var[0].obj_type = SQLMA_DB2;
```

6. 初始化缓冲区以容纳快照输出。

```
snapshotBuffer = (char *)malloc(snapshotBufferSize);
memset (&snapshotBuffer, '\0', sizeof(snapshotBuffer));
```

7. 使用快照请求类型（来自 sqlma 结构）、缓冲区信息和捕获快照所需的其他信息来填充 db2GetSnapshotData 结构。

```
getSnapshotParam.piSqlmaData = pRequestedDataGroups;
getSnapshotParam.poCollectedData = &collectedData;
getSnapshotParam.poBuffer = snapshotBuffer;
getSnapshotParam.iVersion = SQLM_DBMON_VERSION9_5;
getSnapshotParam.iBufferSize = snapshotBufferSize;
getSnapshotParam.iStoreResult = 0;
getSnapshotParam.iNodeNumber = SQLM_CURRENT_NODE;
getSnapshotParam.poOutputFormat = &outputFormat;
getSnapshotParam.iSnapshotClass = SQLM_CLASS_HEALTH;
```

8. 捕获运行状况快照。传递下列参数:
   - db2GetSnapshotData 结构，它包含捕获快照所需的信息
   - 对缓冲区的引用，快照输出将引导至该缓冲区。

```
db2GetSnapshot(db2Version810, &getSnapshotParam, &sqlca);
```

9. 包括用于处理缓冲区溢出的逻辑。获取快照后，将检查 sqlcode 是否存在缓冲区溢出。如果发生了缓冲区溢出，那么将清除并重新初始化缓冲区，然后再次获取快照。

```
while (sqlca.sqlcode == 1606)
{
  free(snapshotBuffer);
  snapshotBufferSize += SNAPSHOT_BUFFER_UNIT_SZ;
  snapshotBuffer = (char *)malloc(snapshotBufferSize);
  if (snapshotBuffer == NULL)
  {
    printf("\nMemory allocation error.\n");
    return;
  }

  getSnapshotParam.iBufferSize = snapshotBufferSize;
  getSnapshotParam.poBuffer = snapshotBuffer;
  db2GetSnapshot(db2Version810, &getSnapshotParam, &sqlca);
}
```

10. 处理快照监视器数据流。参考遵循这些步骤之后的图形就会看到快照监视器数据流。

11. 清除缓冲区。

```
free(snapshotBuffer);
free(pRequestedDataGroups);
```

结果

在使用 db2GetSnapshot API 捕获运行状况快照之后，该 API 以自描述数据流的形式返回运行状况快照输出。以下示例显示数据流结构:

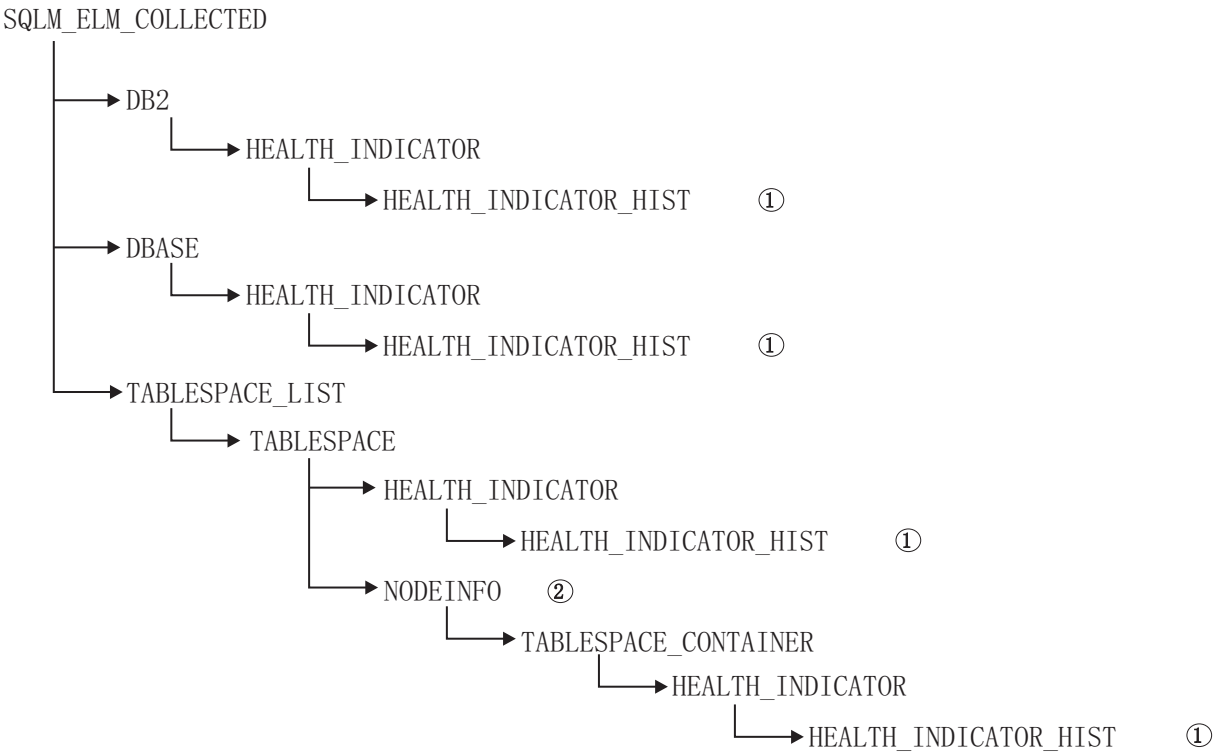


图 10. 运行状况快照自描述数据流

图注:

1. 只有在使用 SQLM_CLASS_HEALTH_WITH_DETAIL 快照类时才可用。
2. 只有在 DB2 Enterprise Server Edition中才可用。否则为表空间容器流。

下列层次结构显示运行状况快照自描述数据流中的特定元素。

SQLM_ELM_HI 中的各个元素的层次结构:

```
SQLM_ELM_HI
   SQLM_ELM_HI_ID
   SQLM_ELM_HI_VALUE
      SQLM_ELM_HI_TIMESTAMP
         SQLM_ELM_SECONDS
         SQLM_ELM_MICROSEC
   SQLM_ELM_HI_ALERT_STATE
```

SQLM_ELM_HI_HIST 中的各个元素的层次结构，仅对 SQLM_CLASS_HEALTH_WITH_DETAIL 快照类可用:

```
SQLM_ELM_HI_HIST
   SQLM_ELM_HI_FORMULA
   SQLM_ELM_HI_ADDITIONAL_INFO
```

```
SQLM_ELM_HEALTH_INDICATOR_HIST
   SQLM_ELM_HI_ID
   SQLM_ELM_HI_VALUE
      SQLM_ELM_HI_TIMESTAMP
         SQLM_ELM_SECONDS
         SQLM_ELM_MICROSEC
   SQLM_ELM_HI_ALERT_STATE
   SQLM_ELM_HI_FORMULA
   SQLM_ELM_HI_ADDITIONAL_INFO
```

SQLM_ELM_OBJ_LIST 中的各个元素的层次结构:

```
SQLM_ELM_HI_OBJ_LIST
   SQLM_ELM_HI_OBJ_NAME
   SQLM_ELM_HI_OBJ_DETAIL
   SQLM_ELM_HI_OBJ_STATE
      SQLM_ELM_HI_TIMESTAMP
         SQLM_ELM_SECONDS
         SQLM_ELM_MICROSEC
```

SQLM_ELM_OBJ_LIST_HIST 中的各个元素的层次结构，仅对 SQLM_CLASS_HEALTH_WITH_DETAIL 快照类可用:

```
SQLM_ELM_HI_OBJ_LIST_HIST
   SQLM_ELM_HI_OBJ_NAME
   SQLM_ELM_HI_OBJ_STATE
      SQLM_ELM_HI_TIMESTAMP
         SQLM_ELM_SECONDS
         SQLM_ELM_MICROSEC
```

运行状况监视器样本输出

下列示例显示使用 CLP 获取的运行状况快照及其相应输出，并说明运行状况监视器的特征。这些示例的目的是启动数据库管理器后立即检查整体运行状况状态。

1. 使用 GET HEALTH SNAPSHOT 命令获取数据库管理器快照:

   ```
   db2 get health snapshot for dbm
   ```

   在从 CLP 发出 GET HEALTH SNAPSHOT 命令后，快照输出将引导至屏幕。

   ```
   节点名                                  =
   节点类型                                = 带有本地客户机和远程客户机的
                                         数据库服务器
   实例名                                  = DB2
   快照时间戳记                            = 11-07-2002 12:43:23.613425

   DB2 实例中的数据库分区数                = 1
   启动数据库管理器时间戳记                = 11-07-2002 12:43:18.000108
   实例最高严重性警报状态                  = 尚未求值

   运行状况指示器:

         尚未求值
   ```

2. 分析输出。从此运行状况快照中，可以看到实例最高严重性警报状态为“Not yet evaluated”。实例处于此状态的原因是运行状况监视器刚刚启动，还没有计算任何运行状况指示器的各项指标。

   如果实例最高严重性警报状态未更改，那么:

   - 检查 HEALTH_MON 数据库管理器配置参数的值以确定运行状况监视器是否已启动。

- 如果 HEALTH_MON=OFF，那么表示运行状况监视器未启动。要启动运行状况监视器，请发出 UPDATE DBM CFG USING HEALTH_MON ON 命令。
- 如果 HEALTH_MON=ON，那么连接至实例以激活运行状况监视器。如果存在实例连接，那么可能不能将运行状况监视器装入到内存中。

使用 CLP 获取数据库运行状况快照的另一示例如下所述。

1. 开始之前应确保数据库连接存在并且数据库已停顿。
2. 使用 GET HEALTH SNAPSHOT 命令获取数据库管理器快照:

   ```
   db2 get health snapshot for db on sample
   ```

3. 在从 CLP 发出 GET HEALTH SNAPSHOT 命令后，快照输出将引导至屏幕。

   ```
                数据库运行状况快照

   快照时间戳记                                = 12-09-2002 11:44:37.793184

   数据库名称                                  = SAMPLE
   数据库路径                                  = E:\DB2\NODE0000\SQL00002\
   输入数据库别名                              = SAMPLE
   在数据库服务器上运行的操作系统              = NT
   数据库的位置                                = 本地
   数据库最高严重性警报状态                    = 注意

   运行状况指示器:

    ...
      指示器名称                                = db.log_util
         值                                     = 60
         单位                                   = %
         求值时间戳记                           = 12-09-2002 11:44:00.095000
         警报状态                               = 正常

      指示器名称                                = db.db_op_status
         值                                     = 2
         求值时间戳记                           = 12-09-2002 11:44:00.095000
         警报状态                               = 注意
   ```

4. 分析输出。

   此运行状况快照显示 *db.db_op_status* 运行状况指示器上存在注意警报。值 2 指示数据库处于停顿状态。

全局运行状况快照

在分区数据库系统上，可获取当前分区、指定分区或所有分区的运行状况快照。对分区数据库的所有分区获取全局运行状况快照时，会尽可能地先聚集数据，然后返回结果。

运行状况指示器的聚集警报状态相当于所有数据库分区上的最高严重性警报状态。不能聚集各个数据库分区上的附加信息和历史记录数据，因此不会提供它们。运行状况指示器的余下数据的聚集将在下表中作详细描述。

表 141. 运行状况指示器值、时间戳记和公式数据的聚集

运行状况指示器	聚集详细信息
• db2.db2_op_status • db2.sort_privmem_util • db2.mon_heap_util • db.db_op_status • db.sort_shrmem_util • db.spilled_sorts • db.log_util • db.log_fs_util • db.locklist_util • db.apps_waiting_locks • db.db_heap_util • db.db_backup_req • ts.ts_util	从包含最高值的分区获取运行状况指示器值。 从同一分区获取求值时间戳记和公式。
• db.max_sort_shrmem_util • db.pkgcache_hitratio • db.catcache_hitratio • db.shrworkspace_hitratio	从包含最低值的分区获取运行状况指示器值。 从同一分区获取求值时间戳记和公式。
• db.deadlock_rate • db.lock_escal_rate	运行状况指示器值是所有数据库分区上的值的总和。 不能聚集求值时间戳记和公式，所以不提供它们。
• ts.ts_op_status • tsc.tscont_op_status • tsc.tscont_util	不聚集这些运行状况指示器。
• db.hadr_op_status • db.hadr_log_delay	多分区数据库不支持这些运行状况指示器。
• db.tb_reorg_req • db.tb_runstats_req • db.fed_nicknames_op_status • db.fed_servers_op_status	仅在一个分区上对此运行状况指示器求值，所以不需要聚集。从对运行状况指示器求值的分区返回数据。

注: 对单个分区对象获取全局快照时，输出将包括所有属性，这是因为没有要聚集的分区。

检索运行状况建议

使用 SQL 执行的运行状态建议查询:

可借助使用 SYSPROC.HEALTH_HI_REC 存储过程的 SQL 来查询建议。

使用 SYSPROC.HEALTH_HI_REC 存储过程时，将在如下 XML 文档中返回建议:

- 该文档根据 `sqllib\misc` 目录中的运行状况建议 XML 模式 DB2RecommendationSchema.xsd 进行了格式化。

- 该文档以 UTF-8 进行编码并且包含客户机语言文本。
- 该文档的结构为一组建议集合，其中每个建议集合描述一个要解决的问题（运行状况指示器），并且包含用于解决该运行状况指示器的一个或多个建议。有关可从该文档检索的信息的特定详细信息，请参阅模式定义。

使用 SQL 查询时返回的 XML 建议文档也会提供可通过 CLP 获取的所有信息。

SYSPROC.HEALTH_HI_REC 存储过程采用下列自变量:

- 运行状况指示器
- 运行状况指示器进入警报状态的对象的定义

输出建议文档返回为 BLOB。因此，从命令行使用此存储过程没什么帮助，原因是 CLP 将限制显示的输出量。建议使用高级语言（如 C 或 Java）来调用此存储过程，高级语言能够正确地对返回的 XML 文档进行语法分析以检索任何必需的元素和属性。

使用命令行处理器来检索运行状况建议:

可通过命令行处理器 (CLP) 使用 GET RECOMMENDATIONS 命令来检索建议。命令语法支持查询建议以解决特定运行状况警报，例如，针对特定对象并且已进入警报状态的运行状况指示器。

开始之前

必须具有实例连接才能从运行状况监视器检索建议。如果没有实例连接，那么创建缺省实例连接。要从远程实例上的运行状况监视器获取建议，必须先连接至该实例。从运行状况监视器检索建议不需要任何特权。

关于此任务

命令语法还支持检索特定运行状况指示器的完整建议集合，执行命令时该运行状况指示器不一定要处于警报状态。可在单一分区级别或全局级别查询用于解决特定运行状况指示器上的警报的建议。

查询针对特定对象的运行状况警报的建议时，运行状况监视器将解决特定警报，并且能够在输出的问题部分提供有关要解决的警报的详细信息。

运行状况监视器还可以提供建议的排名，并且在某些情况下，它能够生成一些脚本，执行这些脚本就可以解决警报。此外，如果某些建议不适用于特定问题情况，运行状况监视器可能拒绝并且不显示这些建议。另外，如果仅通过运行状况指示器名称查询建议（就像第一个示例所示那样），那么将总是返回可能建议的总集合。在这样的情况下，CLP 命令仅显示有关用户看到警报时应考虑采用的操作的信息。

要点: V9.7 中已经不推荐使用“运行状况监视器”、“运行状况指示器”及其相关组件，在以后的发行版中可能会将它们除去。运行状况监视器在 DB2 pureScale 环境中不受支持。 有关更多信息，请参阅『已经不推荐使用运行状况监视器』主题（网址为 http://publib.boulder.ibm.com/infocenter/db2luw/v9r7/topic/com.ibm.db2.luw.wn.doc/doc/i0055045.html）。

过程

使用 GET RECOMMENDATIONS 命令检索建议:

- 您可以想要发出以下命令以查看完整的操作集合，该操作集合是为了解决 **db.db_op_status** 运行状况指示器上的警报而建议的。

  ```
  db2 get recommendations for health indicator db.db_op_status
  ```

  在此示例中，将对 **db.db_op_status** 运行状况指示器返回完整的建议集合。运行状况指示器不必处于警报状态就可以发出此命令。

  此输出显示针对此运行状况指示器的两个可能建议：取消停顿数据库或调查数据库上的前滚进度。因为该命令将用于查询所有可能的建议而不是询问如何解决特定警报，所以运行状况监视器不能标识此情况下的最佳建议。因此，将返回完整的建议集合。

  ```
  建议:

  建议：调查前滚进度。

  因为管理员提交显式请求，所以前滚正在数据库上进行。您必须等待前滚完成，实例才能
  返回至活动状态。

  在命令行处理器中发出以下示例中显示的命令，
  以查看 ROLLFORWARD 实用程序的
  进度:

     LIST UTILITIES SHOW DETAIL

  建议：取消停顿数据库。

  已通过来自管理员的显式请求将数据库置于 QUIESCE PENDING 或 QUIESCE 状态。如果您有
  QUIESCE_CONNECT 权限或者是 DBADM 或 SYSADM，那么表示您仍然可以访问该数据库，并可
  正常使用该数据库。对于所有其他用户，不允许与数据库建立新连接，也不能
  启动新的工作单元。而且，根据停顿请求，将允许活动工作单元完成或立即回滚。
  您可发出取消停顿请求以返回活动状态。

  在命令行处理器中发出以下示例中显示的命令:
  CONNECT TO DATABASE database-alias
  UNQUIESCE DATABASE
  ```

- 假定您观察到数据库 SAMPLE 的运行状况指示器 **db.db_heap_util** 已进入警报状态，并且您想要确定如何解决该警报。在此情况下，您可能想要解决特定问题，因此您可按以下方式发出 GET RECOMMENDATIONS 命令:

  ```
  db2 get recommendations for health indicator db.db_heap_util
      for database on sample
  ```

  此输出显示问题摘要及用于解决问题的一组建议。运行状况监视器按其首选顺序对这些建议排名。每个建议包含描述和一组操作以指示如何执行建议操作。

  ```
  问题:

      指示器名称      = db.db_heap_util       值   = 42
         求值时间戳记 = 11/25/2003 19:04:54
         警报状态     = 警报
         其他信息     =

  建议:

  建议：增加数据库堆大小。
  排名：1

  充分增大数据库配置参数 dbheap 的大小，以使利用率达到正常操作级别。要增大该值，请
  将 dbheap 的新值设置为等于 (pool_cur_size / (4096*U))，其中 U 是期望
  的利用率。例如，如果期望的利用率为警告阈值级别的
  60%（以前您将其设置为 75%），那么 U = 0.6 * 0.75 = 0.45（或 45%）。
  在 DB2 服务器上执行以下命令（此操作可通过使用 EXEC_DB2_CMD 存储过程完成）:
  ```

```
CONNECT TO DATABASE SAMPLE;
UPDATE DB CFG USING DBHEAP 149333;
CONNECT_RESET;
建议: 调查数据库堆的内存使用情况。
排名: 2

每个数据库都有一个数据库堆，并且数据库管理器会代表连接至该数据库
的所有应用程序使用该数据库堆。
数据区将根据需要扩展，最高可扩展至 dbheap 指定的最大大小。

有关数据库堆的更多信息，请参阅 DB2 信息中心。

有关数据库堆的更多信息，请参阅 DB2 信息中心。
调查一段时间内用于数据库堆的内存量，以确定最适当的数据库堆配置参数值。
数据库系统监视器会记录用于数据库堆的最大内存量。
```

- 对于分区数据库系统，可查询针对在特定分区上进入警报状态的运行状况指示器的建议，或者查询针对所有分区的全局建议。以全局方式查询建议时，将返回适用于所有分区上的运行状况指示器的一组建议。例如，如果运行状况指示器在分区 1 和分区 3 上处于警报状态，那么可能返回包含两个脚本的集合，其中每个脚本适用于不同分区。

  以下示例显示如何查询针对特定分区上的运行状况指示器的建议（在此示例中为分区号 2）：

  ```
  db2 get recommendations for health indicator db.db_heap_util
    for database on sample at dbpartitionnum 2
  ```

  以下示例显示如何检索一组建议以解决在若干分区上处于警报状态的运行状况指示器:

  ```
  db2 get recommendations for health indicator db.db_heap_util
    for database on sample global
  ```

使用客户机应用程序检索运行状况建议:

可使用 C 或 C++ 应用程序中的 db2GetRecommendations API 来查询建议。

开始之前

必须具有实例连接才能捕获运行状况快照。如果没有实例连接，那么创建缺省实例连接。要查询远程实例上的建议，必须先连接至该实例。

关于此任务

使用 db2GetRecommendations API 时，将会在如下 XML 文档中返回建议:

- 该文档根据 SQLLIB 目录的 MISC 子目录中的运行状况建议 XML 模式 DB2RecommendationSchema.xsd 进行了格式化。
- 该文档以 UTF-8 进行编码并且包含客户机语言文本。
- 该文档的结构为一组建议集合，其中每个建议集合描述一个要解决的问题（运行状况指示器），并且包含用于解决该运行状况指示器的一个或多个建议。有关可从该文档检索的信息的特定详细信息，请参阅模式定义。

返回的 XML 建议文档也会提供可通过 CLP 获取的所有信息。

要使用客户机应用程序检索运行状况建议:

过程

1. 包括 sqlmon.h 和 db2ApiDf.h DB2 头文件。可在 sqllib\include 目录中找到这些头文件。

   ```
   #include <db2ApiDf.h>
   #include <sqlmon.h>
   ```

2. 声明 sqlca 和 db2GetRecommendationsData 结构。

   ```
   struct sqlca sqlca ;
   db2GetRecommendationsData recData ;

   memset( &sqlca, '\0', sizeof( struct sqlca ) ) ;
   memset( &recData, '\0', sizeof( db2GetRecommendationsData ) ) ;
   ```

3. 使用有关想要对其检索建议的警报的信息来填充 db2GetRecommendationsData 结构。在如下代码摘录中，将针对样本数据库上的 **db2.db_heap_util** 运行状况指示器查询建议。

   ```
   recData.iSchemaVersion = DB2HEALTH_RECSCHEMA_VERSION8_2 ;
   recData.iNodeNumber = SQLM_CURRENT_NODE ;
   recData.iIndicatorID = SQLM_HI_DATABASE_HEAP_UTILIZATION  ;
   recData.iObjType = DB2HEALTH_OBJTYPE_DATABASE ;
   recData.piDbName = "SAMPLE" ;
   ```

4. 调用 db2GetRecommendations API 以检索针对指定数据库上此运行状况指示器的警报的建议。

   ```
   db2GetRecommendations(  db2Version820, &recData, &sqlca ) ;
   ```

5. 检查 sqlca 中返回的 sqlcode 以了解是否发生任何错误。如果 API 调用成功，那么处理 db2GetRecommendationsData 结构的 poRecommendation 字段中返回的建议 XML 文档。使用您选择的 XML 解析器来抽取必需的元素或属性。请参阅 sqllib\misc 目录中的 DB2RecommendationSchema.xsd XML 模式，以了解有关可从 XML 文档检索的信息的详细信息。

6. 释放 db2GetRecommendations API 分配的所有内存。这将释放 db2GetRecommendationsData 结构的 poRecommendation 字段中返回的建议文档。

   ```
   db2GetRecommendationsFree( db2Version820, &recData, &sqlca );
   ```

结果

一般来说，当您检测到运行状况指示器进入警报状态时通常会查询建议，所以应将先前的代码与对快照 API 的调用组合在一起来获取运行状况快照。

运行状况指示器配置

在安装期间将提供缺省运行状况监视器配置。此配置将确保一启动 DB2，运行状况监视器就可以评估数据库环境的运行状况。但是，在对运行状况指示器求值和对警报状态作出反应时，可通过配置特定用户环境来对运行状况监视器的行为作细微的调整

建议不要使用运行状况监视器 SQL 表函数。

要点: V9.7 中已经不推荐使用“运行状况监视器”、“运行状况指示器”及其相关组件，在以后的发行版中可能会将它们除去。运行状况监视器在 DB2 pureScale 环境中不受支持。 有关更多信息，请参阅『已经不推荐使用运行状况监视器』主题（网址为 http://publib.boulder.ibm.com/infocenter/db2luw/v9r7/topic/com.ibm.db2.luw.wn.doc/doc/i0055045.html）。

可在不同级别定义配置。安装 DB2 时为每个运行状况指示器提供了工厂设置的缺省配置。第一次启动运行状况监视器时，工厂设置的副本将提供实例和全局设置的缺省值。

实例设置将应用于实例。全局设置将应用于实例中未定义定制设置的对象，如数据库、表空间和表空间容器。

如果对特定数据库、表空间或表空间容器更新运行状况指示器设置，那么会对更新后的运行状况指示器创建对象设置。对象设置的缺省值是全局设置。

运行状况监视器在对特定数据库、表空间或表空间容器处理运行状况指示器时将检查对象设置。如果特定运行状况指示器的设置未更新，那么会使用缺省全局设置来处理运行状况指示器。当运行状况监视器对实例处理运行状况指示器时，将使用实例设置。

通过使用可对每个运行状况指示器配置的一些属性，可以更改运行状况监视器行为。第一组参数（求值标志、阈值和灵敏度）定义运行状况监视器对运行状况指示器生成警报的条件。第二组参数（操作标志和操作）定义生成警报时运行状况监视器要执行的操作。

求值标志

每个运行状况指示器都有一个求值标志，可用来启用或禁用警报状态的求值。

警告和警报阈值

基于阈值的运行状况指示器包含一些设置，它们用来定义运行状况指示器值的警告和警报区域。可对特定数据库环境修改这些警告和警报阈值。

灵敏度参数

灵敏度参数定义运行状况指示器值必须至少保持警报状态多长时间（以秒计）才会生成警报。与灵敏度值相关联的等待时间从运行状况指示器值进入警报状态期间的第一次刷新时间间隔开始算起。可使用此值来消除因为资源使用中的暂时峰值而生成的错误警报。

考虑使用日志利用率（*db.log_util*）运行状况指示器的示例。假定您每星期复查一次 DB2 通知日志。在第一个星期，*db.log_util* 的某个条目处于警报状态。您想起曾经收到过有关此情况的通知，但在从 CLP 检查警报状态时，运行状况指示器已回复正常状态。第二个星期以后，您在这一个星期的同一时间收到针对同一运行状况指示器的第二个警报通知条目。您在数据库环境中调查两次生成警报的活动，发现有一个应用程序的落实时间很长并且每星期运行一次。此应用程序导致日志利用率在大约 8 到 9 分钟的一小段时间内处于峰值，直到应用程序落实。可查看通知日志的警报通知记录中的历史记录条目，您会发现 *db.log_util* 运行状况指示器每 10 分钟求值一次。因为将生成警报，所以应用程序时间一定会跨越刷新时间间隔。将 *db.log_util* 参数的灵敏度设置为 10 分钟。现在每次 *db.log_util* 的值第一次进入警告或警报阈值区域时，该值必须在该区域中保持至少 10 分钟，才会生成警报。因为应用程序只持续 8 到 9 分钟，所以通知日志中不会再针对此情况记录通知条目。

操作标志

生成警报时的操作运行由操作标志控制。仅当操作标志启用时，才会配置要运行的警报操作。

操作 可将脚本或任务操作配置为在发生警报时运行。对于基于阈值的运行状况指示

器，可将操作配置为针对警告或警报阈值运行。对于基于状态的运行状况指示器，可将操作配置为针对任何可能的非正常状态运行。DB2 管理服务器必须正在运行，这些操作才能运行。

下列输入参数将传递至每个操作系统命令脚本:

- <运行状况指示器短名称>
- <对象名>
- <value | state>
- <警报类型>

脚本操作使用操作系统上的缺省解释器。如果想要使用非缺省解释器，请将 ADMIN_TASK_ADD 过程与脚本内容配合使用以创建任务。在多分区环境中，在脚本操作中定义的脚本必须可供所有分区访问。

不能配置运行状况监视器检查每个运行状况指示器的刷新时间间隔。运行状况监视器认为不能配置建议操作。

运行状况监视器配置存储在二进制文件 HealthRules.reg 中:

- 在 Windows 上，HealthRules.reg 存储在 x:\<SQLLIB_PATH>\<INSTANCE_NAME> 中。例如 d:\sqllib\DB2。
- 在 UNIX 上，HealthRules.reg 存储在 ~/<SQLLIB_PATH>/cfg 中。例如，~/home/sqllib/cfg。

可将运行状况监视器配置复制至其他 Linux、UNIX 或 Windows 服务器上的 DB2 V8 实例。可通过将二进制配置文件复制至目标实例上的适当目录位置以完成此复制。

使用 CLP 检索运行状况指示器配置:

GET ALERT CONFIGURATION 命令允许您查看工厂设置和实例设置、全局设置和对象设置。

过程

1. 要查看数据库级别运行状况指示器的全局设置（在运行状况指示器没有定制设置的情况下，这些设置适用于所有数据库），请发出以下命令:

   ```
   DB2 GET ALERT CONFIGURATION FOR DATABASES
   ```

2. 要查看数据库级别运行状况指示器的全局设置（在运行状况指示器没有定制设置的情况下，这些设置适用于所有数据库），请发出以下命令:

   ```
   DB2 GET ALERT CONFIGURATION FOR DATABASES
   ```

   每个运行状况指示器设置的输出指示它是否更改了缺省值。在以下输出中，全局设置尚未更新；因此，它们与缺省工厂设置完全相同。要查看数据库级别运行状况指示器的工厂设置，使用 DEFAULT 关键字发出在先前示例中发出的命令:

   ```
                        警报配置

   指示器名称                  = db.db_op_status
   缺省值                      = 是
   类型                        = 基于状态
   灵敏度                      = 0
   公式                        = db.db_status;
   操作                        = 已禁用
   阈值或状态检查              = 已启用

   指示器名称                  = db.sort_shrmem_util
   缺省值                      = 是
   类型                        = 基于阈值
   ```

```
警告                                = 70
警报                                = 85
单位                                = %
灵敏度                              = 0
公式                                = ((db.sort_shrheap_allocated/sheapthres_shr)*100);
操作                                = 已禁用
阈值或状态检查                      = 已启用
...
```

3. 要查看 SAMPLE 数据库的定制设置，请发出以下命令:

```
DB2 GET ALERT CONFIGURATION FOR DATABASE ON SAMPLE
```

如果指定的对象上的特定运行状况指示器没有特定设置，那么显示所有数据库的全局设置。要查看特定运行状况指示器的设置，将 USING *health-indicator-name* 子句添加在所有先前示例的前面。

使用 CLP 进行运行状况指示器配置更新: 可对全局设置或特定对象的设置更新特定运行状况指示器的配置。

UPDATE ALERT CONFIGURATION 命令有四个小子句，覆盖了不同的更新选项。每个 UPDATE ALERT CONFIGURATION 命令中只能使用一个小子句。要使用多个选项，必须发出多个 UPDATE ALERT CONFIGURATION 命令。

第一个小子句 SET *parameter-name value* 提供对下列更新的支持:

- 求值标志
- 警告和警报阈值（如果适用）
- 灵敏度标志
- 操作标志

这些设置的相应参数名称是:

- THRESHOLDSCHECKED
- WARNING 和 ALARM
- SENSITIVITY
- ACTIONSENABLED

其他三个小子句提供对添加、更新和删除脚本或任务操作的支持。

下列命令更新 SAMPLE 数据库上的 *db.spilled_sorts* 运行状况指示器的基于阈值的配置。该更新会将警告阈值更改为 25，以启用操作和添加脚本操作:

```
DB2 UPDATE ALERT CONFIGURATION FOR DATABASE ON SAMPLE USING DB.SPILLED_SORTS
        SET WARNING 25, ACTIONSENABLED YES
DB2 UPDATE ALERT CONFIGURATION FOR DATABASE ON SAMPLE USING DB.SPILLED_SORTS
        ADD ACTION SCRIPT c:\myscript TYPE OS COMMAND LINE PARAMETERS 'space'
        WORKING DIRECTORY c:\ ON ALARM USER dba1 PASSWORD dba1
```

下列命令更新全局设置的 *ts.ts_util* 运行状况指示器的基于状态的配置。该更新定义任何表空间处于备份暂挂状态时要运行的操作。

```
DB2 UPDATE ALERT CONFIGURATION FOR TABLESPACES USING TS.TS_UTIL
    SET ACTIONSENABLED YES
DB2 UPDATE ALERT CONFIGURATION FOR TABLESPACES USING TS.TS_UTIL
    ADD ACTION TASK 0.1 ON ATTENTION 32 ON localhost USER dba1 PASSWORD dba1
```

此更新将应用于此运行状况指示器没有定制设置的实例的所有表空间。

将操作添加至运行状况指示器配置时，ON *condition* 子句的选项将基于运行状况指示器的类型:

- 对于基于阈值的运行状况指示器，WARNING 和 ALARM 是有效条件。
- 对于基于状态的运行状况指示器，必须使用 ON ATTENTION *state* 选项。应该使用对运行状况指示器定义的有效数字状态。可在 sqllib\include\sqlmon.h 中找到数据库管理器和数据库操作状态值。表空间和表空间容器操作值列示在 sqllib\include\sqlutil.h 中。注意，不能对处于停机状态的数据库管理器执行这些操作。有关详细信息，请参阅对 db2.db2_op_status 运行状况指示器的描述。

使用 CLP 重置运行状况指示器配置:

CLP 提供对将全局设置重置至工厂设置的支持。特定对象的对象设置还可重置为该对象类型的定制设置。

过程

- 要将 SAMPLE 数据库的对象设置重置至数据库的当前全局设置，请发出以下命令:

  ```
  DB2 RESET ALERT CONFIGURATION FOR DATABASE ON SAMPLE
  ```

- 要将数据库的全局设置重置至工厂设置，请发出以下命令:

  ```
  DB2 RESET ALERT CONFIGURATION FOR DATABASES
  ```

- 要重置特定运行状况指示器的配置，请将 USING *health-indicator-name* 子句添加在所有先前示例的前面。

使用客户机应用程序配置运行状况指示器:

可通过 C 或 C++ 应用程序中的 db2GetAlertCfg、db2UpdateAlertCfg 和 db2ResetAlertCfg API 访问运行状况监视器配置。其中的每个 API 都可访问工厂设置、实例设置、全局设置和对象设置。

开始之前

必须具有实例连接才能访问运行状况监视器配置。如果没有实例连接，那么创建缺省实例连接。要访问远程实例的运行状况监视器配置，必须先连接至该实例。

关于此任务

db2GetAlertCfgData 结构中的 **objType** 和 **defaultType** 参数组合允许您访问各种级别的运行状况指示器配置。

表 142. objType 和 defaultType 用于访问配置级别的设置

设置	objType 和 defaultType
工厂设置	objType = DB2ALERTCFG_OBJTYPE_{DBM | DATABASES |TABLESPACES | CONTAINERS} 和 defaultType = DB2ALERTCFG_DEFAULT

表 142. *objType 和 defaultType 用于访问配置级别的设置 (续)*

设置	objType 和 defaultType
全局设置	objType = DB2ALERTCFG_OBJTYPE_{DBM | DATABASES | TABLESPACES | CONTAINERS} 和 defaultType = DB2ALERTCFG_NOT_DEFAULT 或者 objType = DB2ALERTCFG_OBJTYPE_{DATABASE | TABLESPACE | CONTAINER} 和 defaultType = DB2ALERTCFG_DEFAULT
对象设置	objType = DB2ALERTCFG_OBJTYPE_{DATABASE | TABLESPACE | CONTAINER} 和 defaultType = DB2ALERTCFG_NOT_DEFAULT

过程

1. 要获取 SAMPLE 数据库上的运行状况指示器的特定对象设置:
   a. 包括 sqllib\include 目录中的 db2ApiDf.h DB2 头文件。

```
#include <db2ApiDf.h>
```

   b. 声明并初始化 sqlca 和 db2GetAlertCfgData 结构。

```
struct sqlca ca;
memset (&sqlca, '\0', sizeof(struct sqlca));
char* objName = NULL;
char* dbName = "SAMPLE";
db2Uint32 objType = DB2ALERTCFG_OBJTYPE_DATABASE;
db2Uint32 defaultType = DB2ALERTCFG_NOT_DEFAULT;

db2GetAlertCfgData data = {objType, objName, defaultType, dbName, 0, NULL} ;
```

   c. 调用 db2GetAlertCfg API。

```
rc = db2GetAlertCfg (db2Version810, &data, &ca);
```

   d. 处理返回的配置并释放 API 分配的缓冲区。

```
if (rc >= SQLO_OK) {
  if ((data.ioNumIndicators > 0) && (data.pioIndicators != NULL)) {
    db2GetAlertCfgInd *pIndicators = data.pioIndicators;

    for (db2Uint32 i=0; i  data.ioNumIndicators; i++) {
 //process the entry as necessary using fields defined in db2ApiDf.h
    }
  }

  db2GetAlertCfgFree (db2Version810, &data, &ca);
}
```

2. 下列步骤详细描述更新数据库对象全局设置的 **db.sort_shrmem_util** 运行状况指示器的警报配置的过程，将警告阈值设置为 80 并且添加任务操作 1.1:
   a. 包括 sqllib\include 目录中的 db2ApiDf.h DB2 头文件。

```
#include <db2ApiDf.h>
```

   b. 声明并初始化 sqlca 和 db2AlertTaskAction 结构。

```
struct sqlca ca;
memset (&sqlca, '\0', sizeof(struct sqlca));
db2Uint32 objType = DB2ALERTCFG_OBJTYPE_DATABASES;

db2Uint32 taskCondition = DB2ALERTCFG_CONDITION_WARNING;
char* taskname = "1.1";
char* hostname = NULL;
char* userid = "nobody";
```

```
char* password = "nothing";

db2AlertTaskAction newTask={taskname,taskCondition,userid,password,hostname};
```

c. 声明并初始化 db2UpdateAlertCfgData 结构。

```
struct db2UpdateAlertCfgData setData;

setData.iObjType = objType;
setData.piObjName = NULL;
setData.piDbName = NULL;

setData.iIndicatorID = 1002;

setData.iNumIndAttribUpdates = 1;
setData.piIndAttribUpdates[0].iAttribID = DB2ALERTCFG_WARNING;
setData.piIndAttribUpdates[0].piAttribValue == 80;

setData.iNumActionUpdates = 0;
setData.piActionUpdates = NULL;

setData.iNumActionDeletes = 0;
setData.piActionDeletes = NULL;

setData.iNumNewActions = 1;
setData.piNewActions[0].iActionType = DB2ALERTCFG_ACTIONTYPE_TASK;
setData.piNewActions[0].piScriptAttribs = NULL;
setData.piNewActions[0].piTaskAttribs = &newTask;
```

d. 调用 db2UpdateAlertCfg API。

```
rc = db2UpdateAlertCfg(db2Version810, &setData, &ca);
```

3. 下列步骤详细描述重置 SAMPLE 数据库的 MYTS 表空间的定制设置的过程。

a. 包括 sqllib\include 目录中的 db2ApiDf.h DB2 头文件。

```
#include <db2ApiDf.h>
```

b. 声明并初始化 sqlca 和 db2ResetAlertCfgData structures。

```
struct sqlca ca;
memset (&sqlca, '\0', sizeof(struct sqlca));
char* objName = "MYTS";
char* dbName = "SAMPLE";
db2Uint32 objType = DB2ALERTCFG_OBJTYPE_TABLESPACE;

db2ResetAlertCfgData data = {objType, objName, dbName};
```

c. 调用 db2ResetAlertCfg API。

```
rc = db2ResetAlertCfg (db2Version810, &data, &ca);
```

针对组合状态的运行状况监视器警报操作:

警报操作是在运行状况指示器进入警报状态时运行的任务或脚本。

从 DB2 V9.1 开始，无论其他组合状态如何，每次将表空间的状态设置为单个警报状态时，针对该状态为运行状况指示器 **ts.ts_op_status** 定义的运行状况监视器警报操作都会运行。这使得可以对特定表空间状态运行警报操作，即使该状态与其他状态一起设置。

在以下示例中，即使在表空间状态同时为 QUIESCED:share 和 QUIESCE:update 时，针对注意状态 QUIESCED:share 定义的警报操作 script1 也会运行。

```
db2 call SYSPROC.ADMCMD('update alert cfg for tablespaces using ts.ts_op_status set
actionsenabled yes')

db2 call SYSPROC.ADMCMD('update alert cfg for tablespaces using ts.ts_op_status add
action script /home/guest001/script1 type operating system command line parameters
userParam working directory /home/guest001/ on attention QUIESCED_SHARE on aix1
user guest001 using passw0rd')
```

在以下示例中，当且仅当表空间状态同时为 QUIESCED:share 和 QUIESCED:update 时，使用组合状态（QUIESCED:share + QUIESCED:update = 3）定义的警报操作才执行。

```
db2 call SYSPROC.ADMCMD('update alert cfg for tablespaces using ts.ts_op_status set
actionsenabled yes')

db2 call SYSPROC.ADMCMD('update alert cfg for tablespaces using ts.ts_op_status add
action script /home/guest001/script1 type operating system command line parameters
userParam working directory /home/guest001/ on attention 3 on aix1 user guest001
using passw0rd')
```

从 DB2 V9.1 开始，使用相同操作属性（名称、工作目录、命令行参数、主机、用户和密码）对某个对象定义的运行状况监视器警报操作仅运行一次，即使该操作是针对多个警报状态定义的也是如此。

在以下示例中，对两个不同的警报状态定义了同一操作。即使表空间状态同时为 QUIESCED:share 和 QUIESCED:update，也只能对给定表空间执行一次该操作。

```
db2 call SYSPROC.ADMCMD('update alert cfg for tablespaces using ts.ts_op_status add
action script /home/guest001/script1 type operating system command line parameters
userParam working directory /home/guest001/ on attention QUIESCED_SHARE on aix1
user guest001 using passw0rd')

db2 call SYSPROC.ADMCMD('update alert cfg for tablespaces using ts.ts_op_status add
action script /home/guest001/script1 type operating system command line parameters
userParam working directory /home/guest001/ on attention QUIESCED_UPDATE on aix1
user guest001 using passw0rd')
```

“Windows 管理规范”（WMI）简介

有一个建立管理基础结构标准的业界开端，它提供结合各种硬件和软件管理系统的信息的一种方法。此开端称为“基于 Web 的企业管理”（WBEM）。WBEM 是基于“公共信息模型”（CIM）模式的，它是由 Desktop Management Task Force（DMTF）派生的业界标准。

“Microsoft Windows 管理规范”（WMI）实现了受支持的 Windows 平台的 WBEM 开端。WMI 在 Windows 企业网络中非常有用，使用它可以减少维护和管理企业网络组件的成本。WMI 提供:

- Windows 操作、配置和状态的一致模型。
- 允许访问管理信息的 COM API。
- 允许使用其他 Windows 管理服务。
- 一个灵活且可扩展的体系结构，它允许供应商编写其他 WMI 提供程序以支持新设备、应用程序和其他增强功能。
- 用来创建信息的详细查询的“WMI 查询语言”（WQL）。
- 一个 API，管理应用程序开发者可使用它来编写 Visual Basic 或“Windows 脚本编制主机”（WSH）脚本。

WMI 体系结构分为两个部分:

1. 包括“CIM 对象管理器”（CIMOM）的管理基础结构以及用来管理数据的中央存储区（称为 CIMOM 对象库）。CIMOM 允许应用程序以统一的方式访问管理数据。
2. WMI 提供程序。WMI 提供程序是 CIMOM 与受管对象之间的媒介。通过使用 WMI API，WMI 提供程序向 CIMOM 提供来自受管对象的数据、代表管理应用程序处理请求并生成事件通知。

“Windows 管理规范”（WMI）提供程序是标准的 COM 或 DCOM 服务器，它们充当受管对象与“CIM 对象管理器”（CIMOM）之间的介体。如果 CIMOM 接收到管理应用程序对 CIMOM 对象库中不存在的数据或对事件的请求，那么 CIMOM 将请求转发至 WMI 提供程序。WMI 提供程序为特定于其特定域的受管对象提供数据和事件通知。

DB2 数据库系统与 Windows 管理规范集成

“Windows 管理规范”（WMI）可通过 DB2 性能计数器并使用内置 PerfMon 提供程序来访问快照监视器。

WMI 可通过使用内置“注册表”提供程序访问 DB2 概要文件注册表变量。

“WMI 软件开发包”（WMI SDK）包括几个内置提供程序:

- PerfMon 提供程序
- 注册表事件提供程序
- 注册表提供程序
- Windows 事件日志提供程序
- Win32 提供程序
- WDM 提供程序

WMI 可使用内置“Windows 事件日志”提供程序来访问“事件日志”中的 DB2 错误。

DB2 数据库系统具有“DB2 WMI 管理”提供程序和样本 WMI 脚本文件，用于访问下列受管对象:

1. 数据库服务器的实例，包括分发的那些实例。可以执行以下操作:
   - 枚举实例
   - 配置数据库管理器参数
   - 启动/停止/查询 DB2 服务器服务的状态
   - 设置或建立通信
2. 数据库。可以执行以下操作:
   - 枚举数据库
   - 配置数据库参数
   - 创建/删除数据库
   - 备份/复原/前滚数据库

运行 WMI 应用程序之前需要向系统注册 DB2 WMI 提供程序。通过输入下列命令完成注册:

- mofcomp %DB2PATH%\bin\db2wmi.mof

  此命令将 DB2 WMI 模式的定义装入到系统中。
- regsvr %DB2PATH%\bin\db2wmi.dll

  此命令向 Windows 注册 DB2 WMI 提供程序 COM DLL。

在两个命令中，%DB2PATH% 是安装 DB2 的路径。另外，db2wmi.mof 是包含 the DB2 WMI 模式定义的 .MOF 文件。

与 WMI 基础结构集成有几个好处:

1. 您可以在基于 Windows 的环境中使用 WMI 提供的工具很容易地编写脚本来管理 DB2 服务器。提供了样本 Visual Basic（VBS）脚本，可用它来执行简单任务，例如，列示实例、创建和删除数据库以及更新配置参数。样本脚本包括在 DB2 应用程序开发 Windows 版产品中。
2. 可以创建功能强大的管理应用程序，它们使用 WMI 执行许多任务。这些任务可包括:
   - 显示系统信息
   - 监视 DB2 性能
   - 监视 DB2 系统资源消耗情况

   通过此类型的管理应用程序监视系统事件和 DB2 事件，可以更好地管理数据库。
3. 可使用现有 COM 和 Visual Basic 编程知识和技巧。通过提供 COM 或 Visual Basic 接口，程序员可以在开发企业管理应用程序时节省时间。

监视 Windows 平台上的性能

当使用 Windows 的 DB2 数据库管理器时，可以使用下列工具来监视性能。

要点: 建议不要使用 Windows Performance Monitor，并且将来的发行版可能会将其除去。在 V10.1 中，应改用 IBM InfoSphere Optim Performance Manager。

- **Windows 性能监视器**

  “Windows 性能监视器”使您能够监视数据库和系统性能，可从向系统注册的任何性能数据提供程序检索信息。Windows 还提供有关计算机运行的所有方面的性能数据，包括下列方面:

  – CPU 使用情况
  – 内存使用率
  – 磁盘活动
  – 网络活动

向 Windows 性能监视器注册 DB2

关于此任务

安装程序自动向“Windows 性能监视器”注册 DB2。

要使 Windows 性能监视器可访问 DB2 数据库和 DB2 Connect™ 性能信息，那么必须注册 DB2 for Windows 性能计数器的 DLL。这也允许任何其他使用 Win32 性能 API 的 Windows 应用程序获取性能数据。要安装和注册“DB2 性能计数器”的 DLL（DB2Perf.DLL）并向“Windows 性能监视器”注册此 DDL，请输入:

```
db2perfi -i
```

注册该 DLL 的同时会在注册表的 services 选项中创建一个新键。其中一个条目给出 DLL 的名称，它提供计数器支持。另外三个条目给出该 DLL 中提供的函数的名称。这些函数包括:

Open 在一个进程中系统首次装入该 DLL 时调用。

Collect
从 DLL 请求性能信息时调用。

Close 卸装 DLL 时调用。

启用对 DB2 性能信息的远程访问

要从另一台 DB2 for Windows 计算机查看 Windows 性能对象，必须向 DB2 数据库管理器注册管理员用户名和密码。“Windows 性能监视器”的缺省用户名 SYSTEM 是 DB2 数据库保留字，因此不能使用。

关于此任务

如果 DB2 for Windows 工作站与其他 Windows 计算机联网，那么可使用本节中描述的功能部件。

要注册该用户名，请输入:

```
db2perfr -r username password
```

注: 使用的 username 必须符合 DB2 数据库命名规则。

用户名和密码数据保存在注册表内的一个键中，并设置了安全性，它只允许管理员和 SYSTEM 帐户访问。编码该数据，以避免在注册表中存储管理员密码出现安全性问题。

注:

1. 向 DB2 数据库系统注册了用户名和密码组合之后，即使“性能监视器”的本地实例也将使用该用户名和密码显式登录。这就表示，如果向 DB2 数据库系统注册的用户名信息不匹配，那么“性能监视器”的本地会话将不显示 DB2 数据库性能信息。
2. 必须维护该用户名和密码组合，以便与 Windows 安全性数据库中存储的用户名和密码值相匹配。如果在 Windows 安全性数据库中更改了用户名或密码，那么必须重置用于远程性能监视的用户名和密码组合。
3. 要注销，请输入:

   ```
   db2perfr -u <username> <password>
   ```

显示 DB2 数据库和 DB2 Connect 性能值

关于此任务

要使用“性能监视器”显示 DB2 数据库和 DB2 Connect 性能值，只需从**添加到**框中选择您想要显示其值的性能计数器。此框显示性能对象的列表及其性能数据。选择一个对象，查看该对象提供的计数器列表。

一个性能对象也可以有多个实例。例如，LogicalDisk 对象提供诸如“% 磁盘读时间”和“磁盘字节/秒”之类的计数器；它还为计算机上的每个逻辑驱动器提供一个实例，包括“C:”和“D:”。

Windows 性能对象

Windows 提供了下列性能对象:

- **DB2 数据库管理器**

  此对象提供了单个 Windows 实例的常规信息。受到监视的 DB2 数据库实例显示为对象实例。

由于实际原因和性能原因，每次只能从一个 DB2 数据库实例获取性能信息。“性能监视器”显示的 DB2 数据库实例受“性能监视器”进程中的 db2instance 注册表变量控制。如果您有多个 DB2 数据库实例在同时运行，并且想要查看多个实例的性能信息，对于每个要监视的 DB2 数据库实例，必须将 db2instance 设置为相关值，然后启动一个单独的“性能监视器”会话。

如果运行的是分区数据库环境，那么一次只能从一个数据库分区服务器获取性能信息。在缺省情况下，显示缺省数据库分区（即带有逻辑端口 0 的数据库分区）的性能信息。要查看另一数据库分区的性能信息，必须启动一个单独的“性能监视器”会话，并将 DB2NODE 环境变量设置为要监视的数据库分区的数据库分区号。

- **DB2 数据库**

  此对象提供特定数据库的信息。每个当前活动的数据库都有可用的信息。

- **DB2 应用程序**

  此对象提供特定的 DB2 数据库应用程序的信息。每个当前活动的 DB2 数据库应用程序都有可用的信息。

- **DB2 DCS 数据库**

  此对象提供特定的 DCS 数据库的信息。每个当前活动的数据库都有可用的信息。

- **DB2 DCS 应用程序**

  此对象提供特定的 DB2 DCS 应用程序的信息。每个当前活动的 DB2 DCS 应用程序都有可用的信息。

“Windows 性能监视器”将列示的对象取决于 Windows 计算机上安装了什么内容以及哪些应用程序是活动的。例如，如果安装了 DB2 数据库管理器并且已启动它，将列示“DB2 数据库管理器”对象。如果此计算机上还有一些 DB2 数据库和应用程序当前是活动的，也将列示那些 DB2 数据库和 DB2 应用程序对象。如果将 Windows 系统用作 DB2 Connect 网关，且有一些 DCS 数据库和应用程序当前是活动的，将列示 DB2 DCS 数据库和 DB2 DCS 应用程序对象。

访问远程 DB2 数据库性能信息

关于此任务

前面已讨论过允许远程访问“DB2 性能信息”。在**添加到**框中选择要监视的另一台计算机。这将显示一个列表，列示该计算机上所有可用的性能对象。

为了能够监视远程计算机上的 DB2 性能对象，安装在那台计算机上的 DB2 数据库或 DB2 Connect 代码的级别必须是 V6 或更高版本。

重置 DB2 性能值

关于此任务

当应用程序调用 DB2 监视器 API 时，由于启动了 DB2 数据库服务器，因此返回的信息通常是累积值。但它经常可用于:

- 重置性能值
- 运行测试
- 再次重置值

- 重新运行测试

要重置数据库性能值，可使用 **db2perfc** 程序。请输入:

```
db2perfc
```

缺省情况下，此命令将重置所有活动 DB2 数据库的性能值。但也可指定要重置的数据库的列表。还可使用 -d 选项指定应重置 DCS 数据库的性能值。例如:

```
    db2perfc
db2perfc dbalias1 dbalias2 ... dbaliasn

    db2perfc  -d
db2perfc  -d  dbalias1 dbalias2 ... dbaliasn
```

第一个示例重置所有活动 DB2 数据库的性能值。第二个示例重置特定 DB2 数据库的性能值。第三个示例重置所有活动的 DB2 DCS 数据库的性能值。最后一个示例重置特定的 DB2 DCS 数据库的性能值。

db2perfc 程序重置当前访问相关 DB2 数据库服务器实例（即在运行 **db2perfc** 的会话中的 DB2INSTANCE 中的服务器实例）的数据库性能信息的所有程序的值。

当执行 **db2perfc** 命令时，调用 **db2perfc** 也能重置远程访问 DB2 数据库性能信息的人员所见到的值。

注: 有一个 DB2 数据库 API **sqlmrset**，它允许应用程序重置其在本地（而非全局）看到的特定数据库的值。

第 2 部分 监视元素

监视元素是一种数据结构，用于存储数据库系统状态的特定方面的信息。例如，监视元素 **direct_reads** 反映所发生的直接从磁盘执行（而不是从任何缓冲池执行）的读操作数。

每个监视元素反映下列其中一种数据类型:

计数器 计数器跟踪发生某情况的次数。例如，**deadlocks** 监视元素记录已发生的死锁总数。计数器的其他示例包括 **commit_sql_stmts (commit statements attempted)**、**rows_deleted** 和 **total_sorts**。

标尺 标尺反映发生某情况的程度或使用某对象的次数的度量。例如，所花时间监视元素，例如，**total_section_proc_time** 或 **total_sort_time** 度量不同处理阶段使用的时间。标尺的其他示例包括：**locks_held**、**num_extent_moved** 和 **sort_heap_allocated**。与只能随时间变化增加的计数器相比，根据数据库中发生的情况，标尺中的值可能增加或减少。

水位标记

水位标记反映针对给定度量达到的最高值。例如，**uow_total_time_top** 显示数据库激活后运行时间最长的工作单元的生存期。水位标记的其他示例包括: **pkg_cache_size_top** 和 **sort_heap_top**。

文本 许多监视元素报告文本值。例如，**stmt_text** 包含 SQL 语句的文本。文本监视元素的其他示例包括：**table_name**、**tablespace_type** 和 **db_storage_path_state**。

时间戳记

时间戳记监视元素显示发生某情况的时间。例如，**conn_time** 显示与数据库建立连接的时间。时间戳记监视元素的其他示例包括：**lock_wait_start_time**、**stmt_first_use_time** 和 **uow_stop_time**。与度量耗用时间的标尺相比，时间戳记度量某情况开始或结束的准确时间点。

可使用随 DB2 产品提供的各种监视界面（例如，表函数或事件监视器）的其中一个或多个来检查监视元素。

第 6 章 请求监视元素

请求监视元素（也称为*请求度量值*）用于度量数据库服务器处理不同类型的请求时完成的工作量，其中包括整体系统处理、与特定类型的处理相关的请求以及与特定数据服务器环境相关的请求。

使用请求监视元素来监视数据库系统，尤其是数据服务器为了处理应用程序请求而完成的工作量。

*请求*是对数据库代理程序发出的伪指令，用于执行某些需要耗用数据库资源的工作。请求的来源包括:

- 由外部应用程序直接发出的伪指令，例如 OPEN 或 EXECUTE 伪指令。这些请求被称为“应用程序请求”。
- 协调代理程序向同一个或另一个数据库成员上的子代理程序发出的伪指令。
- 由另一个数据库成员上的代理程序发出的伪指令。

用于度量整体系统处理信息的一些典型监视元素:

- **rqsts_completed_total** 监视元素用于度量系统已完成的请求数。
- **total_rqst_time** 监视元素用于度量数据服务器中的请求所耗用的时间，其中包括等待时间和处理时间。
- **total_wait_time** 监视元素用于度量整体等待时间。
- **total_cpu_time** 监视元素用于度量 CPU 使用时间。

用于度量客户机/服务器处理信息的一些典型监视元素:

- **client_idle_wait_time** 监视元素用于度量等待下一个来自已打开连接的请求时耗用的时间。
- **tcpip_recv_volume** 监视元素用于度量数据服务器通过 TCP/IP 从客户机接收的数据量。

用于度量常用数据服务器处理操作的一些典型监视元素:

- **pool_data_l_reads** 是其中一个用于提供缓冲池资源使用情况信息的监视元素。
- **pool_read_time** 是其中一个用于提供 I/O 处理信息的监视元素。
- **lock_wait_time** 是其中一个用于提供锁定信息的监视元素。
- **total_section_sorts** 是其中一个用于提供排序信息的监视元素。

用于监视与所选类型的数据服务器环境相关的处理的一些典型监视元素:

- **fcm_recv_wait_time** 是其中一个用于测量快速通信管理器 (FCM) 处理的监视元素。
- **wlm_queue_time_total** 是其中一个用于测量工作负载管理控制操作的监视元素。

使用表函数来访问请求度量值

可以使用下列表函数来访问请求度量值:

- MON_GET_SERVICE_SUBCLASS 和 MON_GET_SERVICE_SUBCLASS_DETAILS
- MON_GET_WORKLOAD 和 MON_GET_WORKLOAD_DETAILS

- MON_GET_CONNECTION 和 MON_GET_CONNECTION_DETAILS
- MON_GET_UNIT_OF_WORK 和 MON_GET_UNIT_OF_WORK_DETAILS

这组监视表函数中的每个表函数都有两种格式，其中一种格式的名称以“DETAILS”结尾。未以“DETAILS”结尾的函数提供了用于返回最常用数据的 SQL 关系接口。另一个函数以基于 XML 的方式来访问监视数据并返回一组更详尽的数据。

这组表函数使您能够侧重于特定聚集级别的请求度量值。您可以选择表函数，以便侧重于您在给定情况下关注的部分系统工作负载（或者系统工作负载的聚集）。所有这些表函数都包括一组公共的请求度量值监视元素。每个表函数都可以返回几项并非所有表函数都返回的附加详细信息。

在不存在用户定义的工作负载或服务类的数据库中，数据库管理器执行的所有用户工作都在缺省用户工作负载和用户服务类中发生。对每个服务类（或工作负载）都返回数据的表函数将返回单一服务类（或工作负载）的数据，该服务类（或工作负载）代表整个数据库的用户工作负载的处理。

在存在用户定义的工作负载和服务类的数据库中，对每个服务类（或工作负载）都返回数据的表函数使您能够对每个服务类（或工作负载）的处理进行比较。通过使用 SQL，可以对所有服务类（或工作负载）的值进行求和以获取一个监视元素的值，该监视元素代表整个数据库的用户工作负载的处理。

使用事件监视器来访问请求度量值

请求度量值由下列事件监视器报告:

- 统计信息事件监视器 - 请求度量值是此事件监视器所报告的多种信息的其中一种。
- UoW 事件监视器 - 此事件监视器所报告的字段与 MON_GET_UNIT_OF_WORK 表函数所报告的字段类似或等同。

活动监视元素

活动监视元素（又称为*活动度量值*）是请求监视元素的一部分。您可以使用活动度量值来监视与执行活动（尤其是为了执行 SQL 语句节而完成的处理）相关的数据服务器处理子集。

请求监视元素用于监视数据服务器为了处理应用程序请求而执行的全部工作。活动监视元素用于监视为了执行 SQL 语句节而完成的工作，其中包括锁定、排序和行处理。

要访问活动监视元素的当前值，请使用下列表函数:

MON_GET_ACTIVITY_DETAILS

返回关于进行中的一项或多项活动的详细信息。请在输入参数中指定您所关注的活动。返回的数据包括活动度量值监视元素、许多其他监视元素以及语句文本。数据以 XML 格式返回。

MON_GET_PKG_CACHE_STMT

返回数据库程序包高速缓存中某些或全部 SQL 语句节的详细信息，这些语句既包括静态 SQL 语句也包括动态 SQL 语句。返回的数据包括针对该节被添加到程序包高速缓存后全部各次执行进行聚集的活动度量值监视元素。数据以关系格式返回。

使用活动事件监视器来访问关于活动的历史数据。此监视器将捕获有关每个活动的每次执行的数据。此活动事件监视器与 MON_GET_ACTIVITY_DETAILS 表函数捕获相同的活动监视元素。它还会捕获一些其他信息。

第 7 章 数据对象监视元素

数据对象监视元素提供关于对特定数据对象（其中包括表、索引、缓冲池、表空间和容器）执行的操作的信息。

每种数据对象类型都有一组可以监视的监视元素。例如，缓冲池的元素可用于计算缓冲池命中率。

请使用下列表函数来访问数据对象监视元素的当前值。这些监视器表函数以关系格式返回数据:

- MON_GET_BUFFERPOOL
- MON_GET_TABLESPACE
- MON_GET_CONTAINER
- MON_GET_TABLE
- MON_GET_INDEX

第 8 章 监视元素收集级别

监视元素的*监视元素收集级别*是指对于要针对该元素收集的数据必须处于活动状态的设置（如果存在）。对于许多监视元素，数据收集由配置参数和/或 DDL 中用于定义工作负载管理对象的子句控制。

收集级别设置

使用数据库配置参数设置收集级别会设置整个数据库的特定监视元素类别的缺省收集级别。例如，将配置参数 **mon_req_metrics** 设置为 BASE 会导致针对数据库中运行的所有代理程序收集请求度量值。对于请求和活动度量值以及部分实际值监视元素，还可对特定 WLM 对象指定不同于用于数据库整体的级别的收集级别。这样一来，给定监视元素的*生效收集级别*通过对收集该元素的作用域应用针对该元素指定的最广（最高）收集级别确定。有关不同收集范围如何一起工作的示例，请参阅标题为第 486 页的『示例』的章节。

收集级别在主题中用于描述大部分监视元素。（快照接口返回的监视元素使用监视开关而不是收集级别）。例如，表 143显示的表描述返回监视元素 **skipped_prefetch_data_p_reads** 的接口以及对于要针对该元素收集的数据必须处于活动状态的监视元素收集级别。

大部分主题包括这类表以显示哪些接口返回该监视元素，以及对于要针对主题中描述的监视元素收集的数据，必须达到的最低收集级别。

表 143. 表函数监视信息

表函数	监视元素收集级别
MON_GET_BUFFERPOOL 表函数 - 获取缓冲池度量值	DATA OBJECT METRICS BASE
MON_GET_TABLESPACE 表函数 - 获取表空间度量值	DATA OBJECT METRICS BASE

监视元素参考主题中使用的收集级别如下所示:

始终收集
始终收集此监视元素的数据。没有任何配置参数或 SQL 语句选项用来控制此信息的收集。

DATA OBJECT METRICS BASE 和 DATA OBJECT METRICS EXTENDED
如果数据库配置参数 **mon_obj_metrics** 设置为 BASE 或 EXTENDED，那么收集具有此收集级别的监视元素。如果 **mon_obj_metrics** 设置为 NONE，那么不收集任何数据。

REQUEST METRICS BASE 和 REQUEST METRICS EXTENDED
如果生效收集级别设置为 BASE 或 EXTENDED，那么会收集具有此收集级别的监视元素。请求度量值的生效收集级别通过检查数据库配置参数 **mon_req_metrics** 的当前设置和针对 WLM 服务超类的 COLLECT REQUEST METRICS 子句指定的设置确定。

ACTIVITY METRICS BASE 和 ACTIVITY METRICS EXTENDED

如果生效收集级别设置为 BASE 或 EXTENDED，那么会收集具有此收集级别的监视元素。活动度量值的收集级别通过检查数据库配置参数 **mon_act_metrics** 的当前设置和针对 WLM 工作负载的 COLLECT ACTIVITY METRICS 子句指定的设置确定。

SECTION ACTUALS BASE

如果生效收集级别设置为 BASE，那么会收集具有此收集级别的监视元素。部分实际值的生效收集级别通过检查数据库配置参数 **section_actuals** 的当前设置及以下各项的设置确定:

- CREATE 或 ALTER WORKLOAD、CREATE 或 ALTER WORK ACTION SET 或者 CREATE 或 ALTER SERVICE CLASS 语句中的 INCLUDE ACTUALS 子句。
- WLM_SET_CONN_ENV 例程上的 <collectsectionactuals> 设置。

COLLECT AGGREGATE ACTIVITY DATA 和 COLLECT AGGREGATE REQUEST DATA 如果特定类型的 WLM 对象的 CREATE 或 ALTER 语句中包括子句 COLLECT AGGREGATE ACTIVITY DATA 或 COLLECT AGGREGATE REQUEST DATA，那么会收集具有此收集级别的监视元素。

某些元素被标记为“不适用；报告 XML 文档中作为格式化函数的输入提供的所有元素”。在这些情况下，仅格式化已收集事件数据的输出的函数返回该元素。

示例

示例 1：数据库整体的缺省收集级别为 NONE，特定服务超类的缺省收集级别为 EXTENDED。

此示例说明在此情况下如何对针对数据库整体的请求度量值监视元素指定收集级别 NONE，但仍收集在服务超类中运行的代理程序的 EXTENDED 度量值。

通常，要禁用在数据库中运行的代理程序的请求度量值的收集，但收集特定服务超类的扩展度量值，请执行以下步骤:

1. 通过发出以下命令将针对数据库整体的请求度量值设置收集级别:

   ```
   DB2 UPDATE DB CFG FOR database-name USING MON_REQ_METRICS NONE
   ```

2. 通过执行以下语句来改变要对其收集请求度量值的服务超类:

   ```
   ALTER SERVICE CLASS service-class-name COLLECT REQUEST METRICS EXTENDED
   ```

结果：针对名为 *service-class-name* 的服务超类中运行的所有代理程序收集请求度量值。不收集在该服务超类外部运行的代理程序的请求度量值。

示例 2：数据库整体的缺省收集级别为 EXTENDED。

此示例说明如何在收集监视元素数据时应用最广收集级别（可能产生意外结果）。

- 使用以下命令来指定要对数据库中运行的所有活动捕获活动度量值:

  ```
  DB2 UPDATE DB CFG FOR database-name USING MON_ACT_METRICS EXTENDED
  ```

- 修改 WLM 工作负载以便不收集活动度量值:

  ```
  ALTER WORKLOAD workload-name COLLECT ACTIVITY METRICS NONE
  ```

结果: 对数据库中运行的所有代理程序（包括工作负载 *workload-name* 中运行的代理程序）收集活动度量值。在此情况下，生效收集级别通过在 **mon_act_metrics** 配置参数中对数据库整体指定的更广收集级别 (EXTENDED) 确定。

第 9 章 “耗用时间”监视元素

“耗用时间”监视元素用于跟踪系统中耗用时间的情况。可以查询这些监视元素以了解进行等待所花的时间或者执行不同类型的处理所花的时间。还可以查看在特定系统组件中所耗用的时间。

图 11 显示了一个示例，用来说明您可以如何查看用于等待的相对时间以及处理请求所花的时间。

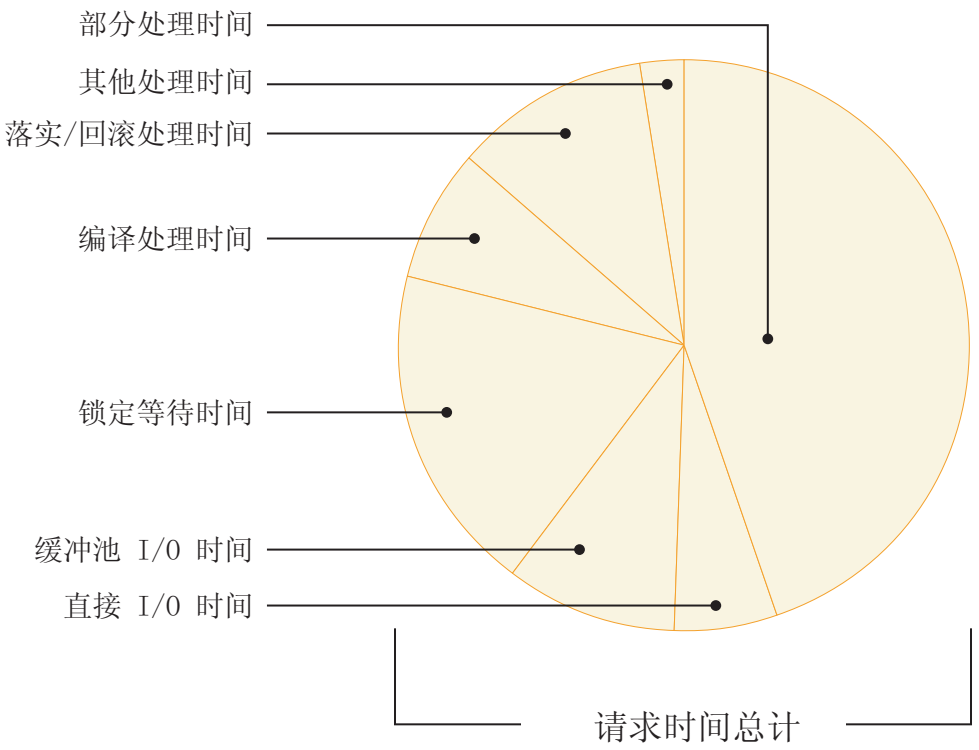


图 11. 耗用时间度量值可以如何提供系统中的时间耗用情况的全景图. 将时间划分为用于等待的时间（锁定等待时间、缓冲池 I/O 时间、直接 I/O 时间）以及实际执行处理所花的时间。

数据库管理器可以采用三种方法来监视系统中耗用的时间:

- 等待时间
- 组件处理时间
- 组件耗用时间。

等待时间

“等待时间”监视元素反映数据库管理器在可以继续进行处理之前等待完成某一事项所花的时间。某些等待时间的示例包括等待完成以下服务所花的时间:

- 入局客户机请求
- 释放对于对象的锁定
- 写入诊断日志
- 从缓冲池中读取或者写入缓冲池。

用于跟踪等待时间的监视元素示例包括：**lock_wait_time** 和 **pool_read_time**。

组件处理时间

这些时间表示在数据库的特定逻辑组件中实际执行处理所花的时间。某些组件处理时间的示例包括执行以下服务所花的时间:

- 落实或回滚事务
- 执行数据库重组
- 编译 SQL
- 装入数据
- 执行 RUNSTATS 操作。

用于跟踪组件处理时间的监视元素示例包括：**total_compile_proc_time** 和 **total_commit_proc_time**。

组件耗用时间

组件耗用时间反映数据库的某个逻辑组件中的耗用时间总计。它们*同时*包括在整个处理阶段可能耗用的处理时间和各种类型的等待时间。例如，执行落实所耗用的总时间既包括实际的落实处理时间，可能还包括各种类型的等待时间（例如，等待完成 I/O 操作或者日志文件操作所耗用的时间）。

注：耗用时间与通过时钟测量的耗用时间不相等；如果耗用的总时间划分为多个线程耗用的时间，那么此数目表示的是每个线程所耗用的时间。

用来说明可以如何使用组件时间的一些示例包括:

- 了解何处在为所给定的工作负载执行成本相对较高的处理（例如，与执行查询相比，成本较高的 SQL 编译）
- 确定特定组件区域的成本是否可以归因于实际的处理，或者确定等待时间在降低吞吐量方面是否扮演重要角色
- 了解在系统中耗用的总时间中，特定组件区域的成本（例如，执行回滚处理）。

用于跟踪组件时间总计的监视元素示例包括：**total_compile_time** 和 **total_commit_time**

您可以查询组件处理时间和等待时间，以获得相对于处理时间对特定等待时间进行的统计分析。第 489 页的图 11 是一个示例，用来说明在这两种类型的耗用时间度量值中，如何相对于一种类型的值来查看另一种类型的值。

虽然组件耗用时间不能用来获得特定类型的等待时间（例如，锁定等待以及与 I/O 相关的等待）的统计分析，但是它们确实提供了一个备用视图，可以用来查看相对于给定逻辑数据库组件中耗用的总时间而提供的处理时间。一个示例是检查表或索引重组实际耗用的处理时间（**total_reorg_proc_time**）与执行重组所耗用的总时间（**total_reorg_time**）之间的比率；而执行重组所耗用的总时间可能包括不是直接与重组本身相关的多种其他处理和等待所耗用的时间。

“耗用时间”监视元素的层次结构

“耗用时间”监视元素中的信息是在更多常规监视元素中累积的。

例如，各等待时间元素（例如，用于表示等待接收表队列中的下一个缓冲区中的信息所耗用时间的元素 (**fcm_tq_recv_wait_time**) 以及用于表示等待 FCM 应答消息所耗用时间的元素 (**fcm_message_recv_wait_time**)）都包括在整个 **fcm_recv_wait_time** 元素中。“耗用时间”监视元素的分层组织使得可以选择具有最适合级别的元素以满足您的需要。

用于查看“耗用时间”监视元素的维和透视图

可以采用不同的方法来查看“耗用时间”监视元素的层次结构。一种方法是从系统角度将它们作为一个整体来查看；还可以在系统中的特定活动的上下文中查看这些监视元素的层次结构。

系统级别的视图或*系统维*包括一些元素，可以使用这些元素来整体了解系统在执行的操作。还可使用系统维中的元素来查看特定工作负载的耗用时间信息。

活动级别的视图或活动维包括一些元素，可以使用这些元素来查看系统耗用时间完成的特定活动（例如，SQL 语句的执行）。活动维中的所有监视元素都包括在更高级别的系统维中。

在这两个维中，每个维都有两个不同的*透视图*，可以用来查看“耗用时间”监视元素:

- 与等待时间相比较的组件处理时间
- 与组件处理时间相比较的组件耗用时间

在第一个透视图中，等待时间元素的值既独立于组件处理时间元素的值，又与组件处理时间元素的值互补。如果您将所报告的所有等待时间之总和加到所有组件处理时间的总和中，那么所获得的值将与 **total_rqst_time** 监视元素所报告的值非常接近。因为任何监视元素都未跟踪少量的其他组件处理时间，所以这两个值之间存在细微差别。

在第二个透视图中，组件耗用时间是组件处理时间的超集。例如，对于执行落实的组件之类的数据库逻辑组件，**total_commit_proc_time** 监视元素报告的落实处理时间总量包括在 **total_commit_time** 监视元素报告的整体落实耗用时间中。耗用时间总计与处理时间总计之间的差值由组件耗用时间监视元素未单独跟踪的其他等待时间或处理时间组成。

查看相对于等待时间的组件耗用时间没有意义，这是因为组件耗用时间已经包括了系统的该部分所造成的等待时间，它是该部分的耗用时间的一部分。如果您创建了一个由组件耗用时间和等待时间组成的饼图，那么因为您重复计算了各种类型的等待时间，所以它将不能准确表示系统中所耗用的时间。

下面几节描述各种维（系统和活动）及透视图（组件处理、等待时间、组件耗用时间和组件处理时间），可从中查看“耗用时间”监视元素中的信息。

提示：并不是所有接口都会报告“耗用时间”元素中的所有信息。例如，**client_idle_wait_time** 监视元素仅对 MON_GET_SERVICE_SUBCLASS 表函数之类的系统级别接口适用。请参阅每个监视元素的参考主题以获取报告该元素的接口的列表。

- 第 492 页的『系统维』
- 第 496 页的『活动维』

系统维

第 493 页的图 12 显示了一个全景图，它从系统维的角度说明了等待时间与组件处理时间的监视元素的相互关系。

- 第 660 页的『client_idle_wait_time -“客户机空闲等待时间”监视元素』
- 第 1280 页的『total_rqst_time -“请求时间总计”监视元素』
  - 第 1302 页的『total_wait_time -“等待时间总计”监视元素』
    - 第 605 页的『agent_wait_time -“代理程序等待时间”监视元素』
    - 第 1337 页的『wlm_queue_time_total -“工作负载管理器队列时间总计”监视元素』
    - 第 863 页的『lock_wait_time -“等待锁定时间”监视元素』
    - 第 872 页的『log_buffer_wait_time -“日志缓冲区等待时间”监视元素』
    - 第 874 页的『log_disk_wait_time -“日志磁盘等待时间”监视元素』
    - 第 1218 页的『tcpip_recv_wait_time -“TCP/IP 接收等待时间”监视元素』
    - 第 1221 页的『tcpip_send_wait_time -“TCP/IP 发送等待时间”监视元素』
    - 第 829 页的『ipc_recv_wait_time -“进程间通信接收等待时间”监视元素』
    - 第 831 页的『ipc_send_wait_time -“进程间通信发送等待时间”监视元素』
    - 第 775 页的『fcm_recv_wait_time -“FCM 接收等待时间”监视元素』[1]
      - 第 783 页的『fcm_tq_recv_wait_time -“FCM 表队列接收等待时间”监视元素』[1]
      - 第 768 页的『fcm_message_recv_wait_time -“接收 FCM 消息等待时间”监视元素』[1]
    - 第 779 页的『fcm_send_wait_time -“FCM 发送等待时间”监视元素』[1]
      - 第 787 页的『fcm_tq_send_wait_time -“FCM 表队列发送等待时间”监视元素』[1]
      - 第 772 页的『fcm_message_send_wait_time -“发送 FCM 消息等待时间”监视元素』[1]
    - 第 632 页的『audit_subsystem_wait_time -“审计子系统等待时间”监视元素』
    - 第 629 页的『audit_file_write_wait_time -“审计文件写等待时间”监视元素』
    - 第 732 页的『diaglog_write_wait_time -“诊断日志文件写等待时间”监视元素』
    - 第 1043 页的『pool_read_time -“缓冲池物理读时间总计”监视元素』
    - 第 1058 页的『pool_write_time -“缓冲池物理写时间总计”监视元素』
    - 第 737 页的『direct_read_time -“直接读时间”监视元素』
    - 第 742 页的『direct_write_time -“直接写时间”监视元素』
    - 第 758 页的『evmon_wait_time -“事件监视器等待时间”监视元素』
    - 第 1253 页的『total_extended_latch_wait_time -“扩展锁存器等待时间总计”监视元素』
    - 第 1080 页的『prefetch_wait_time -“等待预取的时间”监视元素』
    - 第 667 页的『comm_exit_wait_time -“通信缓冲区出口等待时间”监视元素』
  - 第 1241 页的『total_compile_proc_time -“编译处理时间总计”监视元素』
    - 第 1298 页的『total_sync_runstats_proc_time -“同步 RUNSTATS 处理时间总计”监视元素』
    - 第 1293 页的『total_stats_fabrication_proc_time -“统计信息生成处理时间总计”监视元素』
    - *其他监视元素*[2]
  - 第 1258 页的『total_implicit_compile_proc_time -“隐式编译处理时间总计”监视元素』
  - 第 1276 页的『total_routine_user_code_proc_time -“例程用户代码处理时间总计”监视元素』
  - 第 1284 页的『total_section_proc_time -“部分处理时间总计”监视元素』
    - 第 1285 页的『total_section_sort_proc_time -“节排序处理时间总计”监视元素』
    - *其他监视元素*[2]
  - 第 1238 页的『total_commit_proc_time -“落实处理时间总计”监视元素』
  - 第 1270 页的『total_rollback_proc_time -“回滚处理时间总计”监视元素』
  - 第 1282 页的『total_runstats_proc_time -“运行时统计信息处理时间总计”监视元素』
  - 第 1268 页的『total_reorg_proc_time -“重组处理时间总计”监视元素』
  - 第 1260 页的『total_load_proc_time -“装入处理时间总计”监视元素』
  - 第 1246 页的『total_connect_request_proc_time -“连接或交换机用户请求处理时间总计”监视元素』
    - 第 1243 页的『total_connect_authentication_proc_time -“连接认证处理时间总计”监视元素』
    - *其他监视元素*[2]
  - *其他监视元素*[3]

[1]在成员中聚集时，这些与 FCM 相关的等待时间不会获取有意义的信息。有关更多信息，请参阅“FCM 通信的等待时间”

第 495 页的图 13 显示了在各组件区域耗用的时间的监视元素的详细视图。每个组件时间由两个不同的监视元素表示:

- 一个监视元素，它报告某个组件或某个处理阶段的处理时间总计
- 一个监视，它报告在该组件上耗用的整体耗用时间。此整体时间包括该组件的处理时间和可能涉及的任何其他处理时间或等待时间。

- 第 1280 页的『total_rqst_time -“请求时间总计”监视元素』
  - 第 1242 页的『total_compile_time -“编译时间总计”监视元素』
    - 第 1294 页的『total_stats_fabrication_time -“统计信息生成时间总计”监视元素』
      - 第 1298 页的『total_sync_runstats_proc_time -“同步 RUNSTATS 处理时间总计”监视元素』
    - 第 1296 页的『total_sync_runstats_time -“同步 RUNSTATS 时间总计”监视元素』
      - 第 1298 页的『total_sync_runstats_proc_time -“同步 RUNSTATS 处理时间总计”监视元素』
    - *其他监视元素*[1]
  - 第 1259 页的『total_implicit_compile_time -“隐式编译时间总计”监视元素』
    - 第 1258 页的『total_implicit_compile_proc_time -“隐式编译处理时间总计”监视元素』
    - *其他监视元素*[1]
  - 第 1278 页的『total_routine_user_code_time -“例程用户代码时间总计”监视元素』
    - 第 1276 页的『total_routine_user_code_proc_time -“例程用户代码处理时间总计”监视元素』
  - 第 1289 页的『total_section_time -“部分时间总计”监视元素』
    - 第 1287 页的『total_section_sort_time -“节排序时间总计”监视元素』
      - 第 1285 页的『total_section_sort_proc_time -“节排序处理时间总计”监视元素』
      - *其他监视元素*[1]
    - *其他监视元素*[1]
  - 第 1239 页的『total_commit_time -“落实时间总计”监视元素』
    - 第 1238 页的『total_commit_proc_time -“落实处理时间总计”监视元素』
    - *其他监视元素*[1]
  - 第 1271 页的『total_rollback_time -“回滚时间总计”监视元素』
    - 第 1270 页的『total_rollback_proc_time -“回滚处理时间总计”监视元素』
    - *其他监视元素*[1]
  - 第 1281 页的『total_runstats -“运行时统计信息总计”监视元素』
    - 第 1282 页的『total_runstats_proc_time -“运行时统计信息处理时间总计”监视元素』
    - *其他监视元素*[1]
  - 第 1269 页的『total_reorg_time -“重组时间总计”监视元素』
    - 第 1268 页的『total_reorg_proc_time -“重组处理时间总计”监视元素』
    - *其他监视元素*[1]
  - 第 1261 页的『total_load_time -“装入时间总计”监视元素』
    - 第 1260 页的『total_load_proc_time -“装入处理时间总计”监视元素』
    - *其他监视元素*[1]
  - 第 1248 页的『total_connect_request_time -“连接或交换机用户请求时间总计”监视元素』
    - 第 1245 页的『total_connect_authentication_time -“连接或交换机用户认证请求时间总计”监视元素』
      - 第 1243 页的『total_connect_authentication_proc_time -“连接认证处理时间总计”监视元素』
    - *其他监视元素*[1]
  - *其他监视元素*[2]

[1]这些监视元素包括一个或多个不同类型的等待时间。

[2]这些监视元素包括少量当前未监视的其他耗用时间类型（处理时间和等待时间）。

图 13. 组件处理耗用时间监视元素 - 系统维. 缩进的监视元素的值包括在位于层次结构的次高级别的那些元素之前的元素中。

活动维

图 14 显示了一些监视元素，可以通过这些监视元素从等待时间（与组件处理时间相比较）的透视图来查看活动。

- 第 1168 页的『stmt_exec_time -“语句执行时间”监视元素』
  - 第 1232 页的『total_act_wait_time -“活动等待时间总计”监视元素』 [1]
    - 第 863 页的『lock_wait_time -“等待锁定时间”监视元素』
    - 第 872 页的『log_buffer_wait_time -“日志缓冲区等待时间”监视元素』
    - 第 874 页的『log_disk_wait_time -“日志磁盘等待时间”监视元素』
    - 第 775 页的『fcm_recv_wait_time -“FCM 接收等待时间”监视元素』[2]
      - 第 783 页的『fcm_tq_recv_wait_time -“FCM 表队列接收等待时间”监视元素』[2]
      - 第 768 页的『fcm_message_recv_wait_time -“接收 FCM 消息等待时间”监视元素』[2]
    - 第 779 页的『fcm_send_wait_time -“FCM 发送等待时间”监视元素』[2]
      - 第 787 页的『fcm_tq_send_wait_time -“FCM 表队列发送等待时间”监视元素』[2]
      - 第 772 页的『fcm_message_send_wait_time -“发送 FCM 消息等待时间”监视元素』[2]
    - 第 632 页的『audit_subsystem_wait_time -“审计子系统等待时间”监视元素』
    - 第 758 页的『evmon_wait_time -“事件监视器等待时间”监视元素』
    - 第 629 页的『audit_file_write_wait_time -“审计文件写等待时间”监视元素』
    - 第 732 页的『diaglog_write_wait_time -“诊断日志文件写等待时间”监视元素』
    - 第 1043 页的『pool_read_time -“缓冲池物理读时间总计”监视元素』
    - 第 1058 页的『pool_write_time -“缓冲池物理写时间总计”监视元素』
    - 第 737 页的『direct_read_time -“直接读时间”监视元素』
    - 第 742 页的『direct_write_time -“直接写时间”监视元素』
    - 第 1253 页的『total_extended_latch_wait_time -“扩展锁存器等待时间总计”监视元素』
    - 第 1080 页的『prefetch_wait_time -“等待预取的时间”监视元素』
  - 第 1274 页的『total_routine_non_sect_proc_time -“非部分处理时间” 监视元素』
    - 第 1276 页的『total_routine_user_code_proc_time -“例程用户代码处理时间总计”监视元素』
    - *其他监视元素*[3]
  - 第 1284 页的『total_section_proc_time -“部分处理时间总计”监视元素』
    - 第 1285 页的『total_section_sort_proc_time -“节排序处理时间总计”监视元素』
    - *其他监视元素*[3]
  - *其他监视元素*[4]

[1]此监视元素不包括语句执行的嵌套（子）活动导致的任何等待时间。

[2]在成员中聚集时，这些与 FCM 相关的等待时间不会获取有意义的信息。有关更多信息，请参阅“FCM 通信的等待时间”。

[3]包括未与此组件明确相关的其他处理时间。

[4]包括当前未监视的其他类型的少量耗用时间（处理时间和等待时间）。此外，此时间包括子活动导致的任何处理时间和等待时间。

图 14. 等待耗用时间和组件处理耗用时间监视元素 - 活动维. 缩进的监视元素的值包括在位于层次结构的次高级别的那些元素之前的元素中。

第 497 页的图 15 显示了一些监视元素，可以通过这些监视元素从组件耗用时间（其中包括组件处理时间）的透视图来查看活动。

- 第 1168 页的『stmt_exec_time -“语句执行时间”监视元素』
  - 第 1289 页的『total_section_time -“部分时间总计”监视元素』
    - 第 1287 页的『total_section_sort_time -“节排序时间总计”监视元素』
      - 第 1285 页的『total_section_sort_proc_time -“节排序处理时间总计”监视元素』
      - *其他*[1]
    - *其他*[2]
  - 第 1275 页的『total_routine_time -“例程时间总计”监视元素』
    - 第 1274 页的『total_routine_non_sect_time -“非部分例程执行时间” 监视元素』
      - 第 1278 页的『total_routine_user_code_time -“例程用户代码时间总计”监视元素』
      - *其他*[2]
    - *其他*[2]

[1]这些监视元素包括一个或多个不同类型的等待时间。

[2]这些处理时间元素包括未专门与此组件相关其他处理时间和等待时间。

图 15. 组件耗用时间和组件处理时间监视元素 - 活动维. 缩进的监视元素的值包括在位于层次结构的次高级别的那些元素之前的元素中。

FCM 通信的等待时间

在多分区数据库中，或者在具有分区内并行性的环境中，快速通信管理器 (FCM) 管理处理同一语句的不同代理程序之间的通信，无论这些代理程序是否在同一成员中。当一个代理程序等待另一个代理程序完成工作或等待数据从一个代理程序传输到另一个代理程序时，所有 FCM 通信都会涉及等待时间的可能性。

与 FCM 相关的等待时间不一定指示跨成员的处理被阻止；对于给定语句，可针对跨成员的子代理程序并行或串行执行工作。与 FCM 相关的等待时间显示某个代理程序因等待另一个代理程序而在单个成员上被阻塞的时间；但是，工作可能会在其他成员上顺利进行。

例如，成员 0 上的代理程序 A 可能会因等待成员 1 上的代理程序 B 读取发送到代理程序 B 的数据而被阻塞。如果代理程序 B 正忙并且未立即从表队列中检索数据，那么代理程序 A 在被强制等待来自代理程序 B 的确认前，仅被允许发送有限的数据量，之后才能发送其余数据。这一等待时间由代理程序 A 计为 **fcm_tq_send_wait_time**。

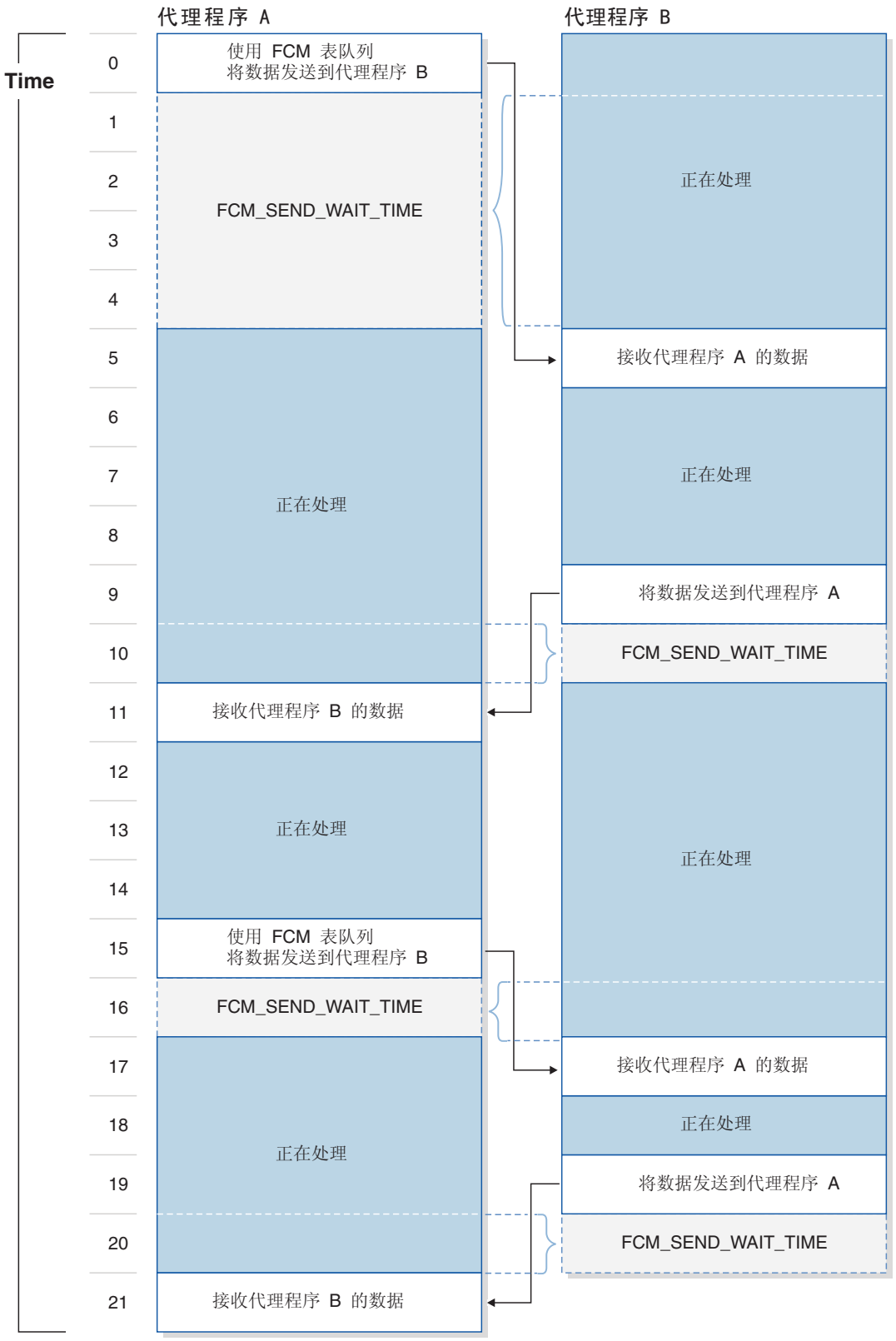


图 16. FCM 通信中的等待时间

另一个场景可能涉及成员上的代理程序将请求分派到另一个成员上的代理程序。如果发生以下情况之一，将产生 **fcm_message_recv_wait_time**:

- 代理程序 A 将一个很长的请求发送到代理程序 B，且代理程序 B 被强制等待接收完整请求。在此情况下，代理程序 B 会产生 **fcm_message_recv_wait_time**。
- 代理程序 A 将请求发送到代理程序 B 并等待代理程序 B 的应答。在此情况下，代理程序 A 会产生 **fcm_message_recv_wait_time**。

如果发生以下情况之一，将产生 **fcm_message_send_wait_time**:

- 代理程序 A 将一个很长的请求发送到代理程序 B，且该请求由于某些原因被阻止。例如，当正在被发送的请求的第一部分由本地 FCM 守护程序处理时，代理程序 A 可能需要等待。在此情况下，代理程序 A 会产生 **fcm_message_send_wait_time**。
- 代理程序 B 发送对来自代理程序 A 的请求的应答。如果在发送整个消息之前，代理程序 B 由于某些原因而被阻止，那么代理程序 B 会产生 **fcm_message_send_wait_time**。

根据您要进行度量的内容，如果您在聚集多个分区之间所耗用时间的度量值，从总时间中减去 FCM 等待时间可能才是正确的。

检索和处理“耗用时间”监视元素数据

可以几乎不受限制地使用“耗用时间”监视元素数据。例如，可以自动生成一些图表，这些图表清楚地显示了系统中的时间耗用情况。或者，可以使用数据在一段时间内跟踪系统中的某些类型的等待时间。

接下来的主题提供了一些基本示例，用来说明如何使用“耗用时间”监视元素以及您用来访问它们包含的数据的表函数。

查看系统中耗用时间的位置

可以使用“耗用时间”监视元素来了解系统中耗用时间的位置。可以使用“耗用时间”监视元素来针对特定工作单元、服务子类、工作负载或连接进行报告。

关于此任务

一旦您检索用于报告系统中耗用时间的位置的各种监视元素，就可以采用多种方法来查看这些监视元素。在最基本的级别，可以将所报告的值作为列表来查看。您可能想使用值来创建一些比率，例如，锁定等待时间与总的请求时间之间的比率。您也可以使用检索到的值来创建图表，以帮助您将“耗用时间”监视元素相对于另一个监视元素进行可视化。

注意:

- 这些查询的输出中所显示的值只是为了便于说明，不应将它们理解为表示您在自己的系统中可能看到的内容。
- 此任务说明如何检索特定的“耗用时间”监视元素。还可以使用 V9.7 修订包 1 中引入的新的格式化函数来检索满足特定条件的“耗用时间”监视元素，例如，具有非零值的那些监视元素，在您指定的某一范围内的值的监视元素，或者最前面的 *n* 个监视元素（例如，前 5 个等待时间）。示例 4 说明了这些功能的工作方式。

过程

1. 首先，确定您对哪些“耗用时间”元素感兴趣。 例如，您可能希望查看与系统中的所有连接的总请求时间相比较的等待时间总计。

2. 明确阐明一个将使用其中一个监视表函数的 SQL 查询，而此表函数将检索您感兴趣的元素。 在此示例中，可以使用 MON_GET_CONNECTION 表函数来检索某个连接的 **total_request_time** 和 **total_wait_time** 监视元素:

```
SELECT APPLICATION_HANDLE,
       TOTAL_WAIT_TIME,
       TOTAL_RQST_TIME
FROM TABLE(MON_GET_CONNECTION(NULL,NULL))
```

上述查询将返回以下输出（所有时间都是按毫秒进行报告）:

```
APPLICATION_HANDLE   TOTAL_WAIT_TIME      TOTAL_RQST_TIME
-------------------- -------------------- --------------------
                  39                  179                  269
                  78                    0                    0
                  51                  207                  316
                  77                    0                   21
                  50                 1014                 1408
                  40                  109                  351
                  79                   89                  167

  7 record(s) selected.
```

3. 在此示例中有 7 个应用程序连接；可以使用第二列和第三列中的结果来确定等待每个应用程序所耗用时间所占的百分比。 例如，对于应用程序 50，与总的请求时间相比，该应用程序所耗用的等待时间为 (1014 ÷ 1408) × 100 ≈ 72%。

示例

示例 1: 确定所有连接中用于等待的时间相对于请求时间总计的平均值。

此示例与前一个示例相似，只不过这次是通过 SQL 来计算等待时间百分比的平均值:

```
WITH PCTWAIT AS (
     SELECT SUM(TOTAL_WAIT_TIME)AS WAIT_TIME,
            SUM(TOTAL_RQST_TIME)AS RQST_TIME
     FROM TABLE(MON_GET_CONNECTION(NULL,NULL)) AS METRICS)
     SELECT WAIT_TIME,
            RQST_TIME,
     CASE WHEN RQST_TIME > 0
     THEN DEC((FLOAT(WAIT_TIME))/FLOAT(RQST_TIME) * 100,5,2)
     ELSE NULL END AS WAIT_PCT FROM PCTWAIT
```

运行上述查询的结果将类似如下内容:

```
WAIT_TIME            RQST_TIME            WAIT_PCT
-------------------- -------------------- --------
                1515                 2439    62.11

  1 record(s) selected.
```

示例 2: 对特定服务子类的等待时间总计与所选组件处理时间进行比较

此示例说明了可以如何将特定类型的组件处理所耗用的时间与用于等待的时间进行比较:

```
SELECT SUM(TOTAL_WAIT_TIME) AS WAIT,
       SUM(TOTAL_COMPILE_PROC_TIME) AS COMPILE,
       SUM(TOTAL_IMPLICIT_COMPILE_PROC_TIME) AS IMP_COMPILE,
       SUM(TOTAL_SECTION_PROC_TIME) AS SECTION,
       SUM(TOTAL_COMMIT_PROC_TIME) AS COMMIT,
       SUM(TOTAL_REORG_PROC_TIME) AS REORG,
       SUM(TOTAL_RUNSTATS_PROC_TIME) AS RUNSTATS,
       SUM(TOTAL_ROLLBACK_PROC_TIME) AS ROLLBACK,
       SUM(TOTAL_LOAD_PROC_TIME) AS LOAD
 FROM TABLE(MON_GET_SERVICE_SUBCLASS( 'SYSDEFAULTUSERCLASS','SYSDEFAULTSUBCLASS',NULL))
```

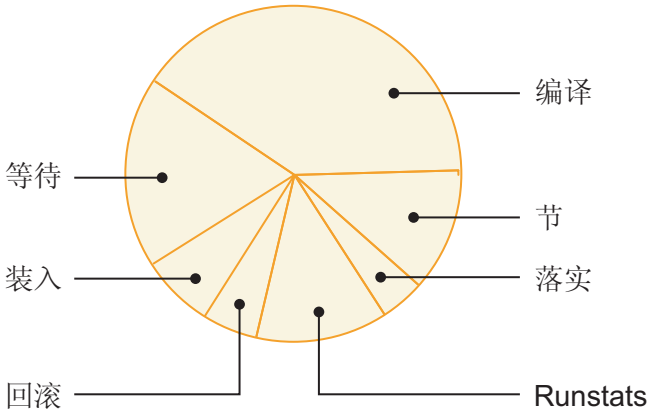
运行上述查询的结果将类似如下内容（为了便于显示，已经分割了该查询的输出行）：

```
WAIT                 COMPILE              IMP_COMPILE          SECTION              COMMIT
-------------------- -------------------- -------------------- -------------------- --------------------
                 611                 1931                    0                  395                   15

REORG                RUNSTATS             ROLLBACK             LOAD
-------------------- -------------------- -------------------- --------------------
                   0                  432                   18                    0

  1 record(s) selected.
```

可以使用所报告的数目来构造一个饼图，以显示与不同处理阶段所耗用时间进行比较的相对等待时间（不包括组件时间为 0）。



示例 3: 查看耗用的时间总计与不同组件中的处理时间相比较而得到的比率

此示例说明了可以如何获得在不同处理阶段（组件）执行操作所耗用的时间相对于该组件中耗用的时间总计的概述。以下查询将计算执行实际处理所耗用的时间与特定组件中的耗用时间总计相比较而获得的比率（用百分比表示）。

```
WITH PCTPROC AS (
     SELECT SUM(TOTAL_SECTION_TIME) AS SECT_TIME, SUM(TOTAL_SECTION_PROC_TIME) AS SECT_PROC_TIME,
            SUM(TOTAL_COMPILE_TIME) AS COMP_TIME, SUM(TOTAL_COMPILE_PROC_TIME) AS COMP_PROC_TIME,
            SUM(TOTAL_IMPLICIT_COMPILE_TIME) AS IMP_C_TIME, SUM(TOTAL_IMPLICIT_COMPILE_PROC_TIME) AS IMP_C_PROC_TIME,
            SUM(TOTAL_COMMIT_TIME) AS COMMIT_TIME, SUM(TOTAL_COMMIT_PROC_TIME) AS COMMIT_PROC_TIME,
            SUM(TOTAL_ROLLBACK_TIME) AS ROLLBACK_TIME, SUM(TOTAL_ROLLBACK_PROC_TIME) AS ROLLBACK_PROC_TIME,
            SUM(TOTAL_RUNSTATS_TIME) AS RUNSTATS_TIME, SUM(TOTAL_RUNSTATS_PROC_TIME)AS RUNSTATS_PROC_TIME,
            SUM(TOTAL_REORG_TIME) AS REORG_TIME, SUM(TOTAL_REORG_PROC_TIME) AS REORG_PROC_TIME,
            SUM(TOTAL_LOAD_TIME) AS LOAD_TIME, SUM(TOTAL_LOAD_PROC_TIME) AS LOAD_PROC_TIME
     FROM TABLE(MON_GET_CONNECTION(NULL, -2)) AS METRICS)
     SELECT CASE WHEN SECT_TIME > 0
                 THEN DEC((FLOAT(SECT_PROC_TIME) / FLOAT(SECT_TIME)) * 100,5,1)
                 ELSE NULL END AS SECT_PROC_PCT,
            CASE WHEN COMP_TIME > 0
                 THEN DEC((FLOAT(COMP_PROC_TIME) / FLOAT(COMP_TIME)) * 100,5,1)
                 ELSE NULL END AS COMPILE_PROC_PCT,
            CASE WHEN IMP_C_TIME > 0
                 THEN DEC((FLOAT(IMP_C_PROC_TIME) / FLOAT(IMP_C_TIME)) * 100,5,1)
                 ELSE NULL END AS IMPL_COMPILE_PROC_PCT,
              CASE WHEN ROLLBACK_TIME > 0
                 THEN DEC((FLOAT(ROLLBACK_PROC_TIME) / FLOAT(ROLLBACK_TIME)) * 100,5,1)
                 ELSE NULL END AS ROLLBACK_PROC_PCT,
              CASE WHEN COMMIT_TIME > 0
                 THEN DEC((FLOAT(COMMIT_PROC_TIME) / FLOAT(COMMIT_TIME)) * 100,5,1)
                 ELSE NULL END AS COMMIT_PROC_PCT,
              CASE WHEN RUNSTATS_TIME > 0
                 THEN DEC((FLOAT(RUNSTATS_PROC_TIME) / FLOAT(RUNSTATS_TIME)) * 100,5,1)
                 ELSE NULL END AS RUNSTATS_PROC_PCT,
            CASE WHEN REORG_TIME > 0
                 THEN DEC((FLOAT(REORG_PROC_TIME) / FLOAT(REORG_TIME)) * 100,5,1)
                 ELSE NULL END AS REORG_PROC_PCT,
            CASE WHEN LOAD_TIME > 0
                 THEN DEC((FLOAT(LOAD_PROC_TIME) / FLOAT(LOAD_TIME)) * 100,5,1)
                 ELSE NULL END AS LOAD_PROC_PCT
       FROM PCTPROC
```

此查询将生成以下输出:

```
SECT_PROC_PCT COMPILE_PROC_PCT IMPL_COMPILE_PROC_PCT ROLLBACK_PROC_PCT COMMIT_PROC_PCT RUNSTATS_PROC_PCT REORG_PROC_PCT LOAD_PROC_PCT
------------- ---------------- ---------------------- ----------------- ---------------- ------------------ -------------- -------------
         57.6              0.1                      -              96.9             95.6                0.0           71.1          84.6

  1 record(s) selected.
```

用图形来表示此数据时，看起来可能与 图 17 中所显示的内容相似:

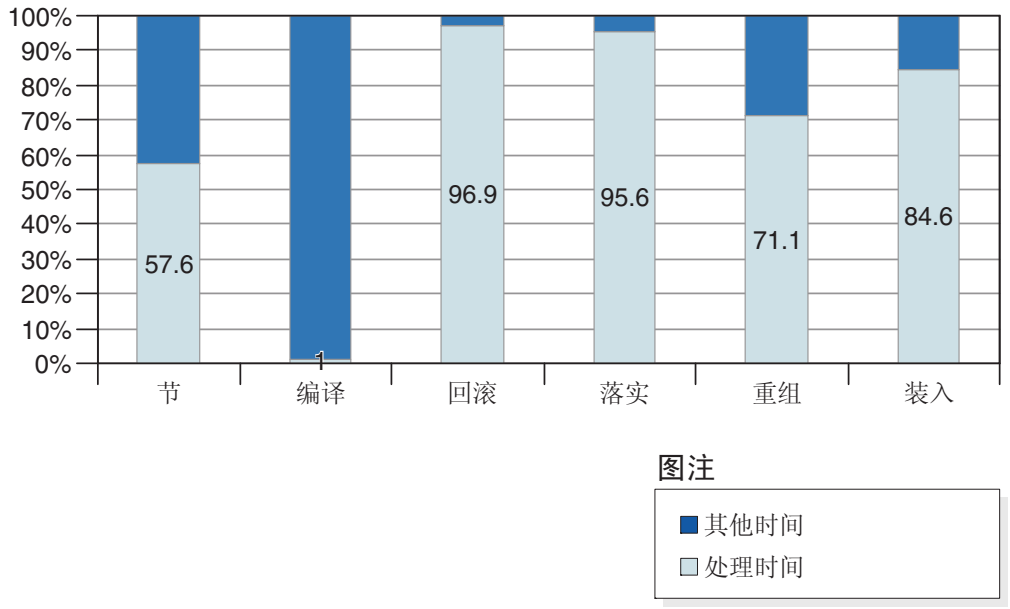


图 17. 用耗用时间总计的百分比来表示的组件处理时间

示例 4: 将“耗用时间”监视元素排名

在前面的示例中，所显示的所有监视元素都是在用于查询的 SQL 中显式指定

的；在查询结果中，每个监视元素都显示在它自己的那一列中。但是，有时候您可能不知道要检查哪些“耗用时间”监视元素（例如，您想查看前 10 个等待时间监视元素，或者只想查看非零的“耗用时间”监视元素）。

在 DB2 V9.7 修订包 1 中添加了多个表函数，可以将它们用来按面向行的格式显示监视元素，这种情况下，每个元素显示在单独的一行中。可以用来完成此任务的表函数的名称格式为 MON_FORMAT_XML_* _BY_ROW。这些函数将从某些监视界面返回的 XML 文档中抽取度量值。（请参阅第 12 页的『在 XML 文档中返回监视数据的接口』以了解更多信息。）

当您不知道要查看哪些元素时，MON_FORMAT_XML_* _BY_ROW 函数就很有用。例如，您可能想查看名为 CLPWORKLOAD 的工作负载的前 10 个等待时间监视元素。要收集此信息，可以创建一个称为 DBSTATS 的统计信息事件监视器（event_wlstats 逻辑数据组）。假定您设置此事件监视器以写入某个表，那么它会将度量值记录在 DETAILS_XML 列中。一旦使用监视数据填充了事件监视器的输出表，就可以构造一个使用 MON_FORMAT_XML_WAIT_TIMES_BY_ROW 函数来抽取您要查看的监视元素的查询。

```
SELECT SUBSTR(STATS.WORKLOAD_NAME,1,15) AS WORKLOAD_NAME,
       SUBSTR(METRICS.METRIC_NAME,1,30) AS METRIC_NAME,
       SUM(METRICS.TOTAL_TIME_VALUE) AS TOTAL_TIME_VALUE
FROM   WLSTATS_DBSTATS AS STATS,
       TABLE(MON_FORMAT_XML_WAIT_TIMES_BY_ROW(STATS.DETAILS_XML)) AS METRICS
WHERE  WORKLOAD_NAME='CLPWORKLOAD' AND (PARENT_METRIC_NAME='TOTAL_WAIT_TIME')
GROUP  BY WORKLOAD_NAME,METRIC_NAME
ORDER  BY TOTAL_TIME_VALUE DESC
FETCH FIRST 10 ROWS ONLY
```

切记：“耗用时间”监视元素按层次结构进行组织。在此示例中，为了避免存在重复计算的等待时间，将只包括会累积到 **total_wait_time** 中的监视元素（请参阅前一个 SQL 语句中的 WHERE 子句）。否则，**total_wait_time** 本身也会包括在结果中，total_wait_time 包括若干个单独的等待时间。

下面的输出显示上述查询的结果可能为如下所示：

```
WORKLOAD_NAME   METRIC_NAME                    TOTAL_TIME_VALUE
--------------- ------------------------------ --------------------
CLPWORKLOAD     LOCK_WAIT_TIME                        15138541
CLPWORKLOAD     DIRECT_READ_TIME                       6116231
CLPWORKLOAD     POOL_READ_TIME                         6079458
CLPWORKLOAD     DIRECT_WRITE_TIME                       452627
CLPWORKLOAD     POOL_WRITE_TIME                         386208
CLPWORKLOAD     IPC_SEND_WAIT_TIME                      283172
CLPWORKLOAD     LOG_DISK_WAIT_TIME                      103888
CLPWORKLOAD     DIAGLOG_WRITE_WAIT_TIME                  78198
CLPWORKLOAD     IPC_RECV_WAIT_TIME                       15612
CLPWORKLOAD     TCPIP_SEND_WAIT_TIME                      3291

  10 record(s) selected.
```

确定在执行 SQL 语句期间耗用时间的位置

检索活动级别的耗用时间信息的一个示例是查看特定 SQL 语句的“耗用时间”监视元素。可以使用 MON_GET_PKG_CACHE_STMT 表函数来检索此信息。

关于此任务

此任务显示了一个示例，用来说明如何检索有关程序包高速缓存中的 SQL 语句的所选耗用时间详细信息。

注：

- 为程序包高速缓存中的给定语句报告的耗用时间度量值，是执行该语句的所有次数的耗用时间度量值的总计。

- 这些查询的输出中所显示的值只是为了便于说明，不应将它们理解为表示您在自己的系统中可能看到的内容。

过程

1. 明确阐述一个 SQL 语句，它使用 MON_GET_PKG_CACHE_STMT 表函数来检索有关程序包高速缓存中的语句的信息。 例如，假定您想确定相对于语句的执行时间总计的等待时间总计。用于检索此信息的查询可能为如下所示:

```
SELECT SUM(STMT_EXEC_TIME) AS TOTAL_EXEC_TIME,
       SUM(TOTAL_ACT_WAIT_TIME) AS TOTAL_WAIT_TIME,
       EXECUTABLE_ID
FROM TABLE(MON_GET_PKG_CACHE_STMT ( NULL, NULL, NULL, -2)) AS T
WHERE STMT_EXEC_TIME <> 0
GROUP BY EXECUTABLE_ID
ORDER BY TOTAL_EXEC_TIME DESC
```

2. 运行此查询。 结果可能与下列输出相似:

```
TOTAL_EXEC_TIME      TOTAL_WAIT_TIME      EXECUTABLE_ID
-------------------- -------------------- --------------------------------------------------------------------------------
                9021                 9021 x'0100000000000000032000000000000000000000002002009111112032014000'
                3017                  372 x'0100000000000000033000000000000000000000002002009111111543806200'
                 591                    0 x'0100000000000000010000000000000000000000002002009111111525226500'
                 203                  192 x'0100000000000000027000000000000000000000002002009111111593675000'
                 142                    0 x'010000000000000002B000000000000000000000002002009111111594400000'
                 111                   48 x'0100000000000000070000000000000000000000002002009111111544135900'
                 108                   35 x'010000000000000000B000000000000000000000002002009111111544175000'
                  55                    0 x'010000000000000000D000000000000000000000002002009111111544206200'
                  50                    0 x'010000000000000000C000000000000000000000002002009111111544192100'
                  38                    0 x'0100000000000000026000000000000000000000002002009111111593660900'
                  35                    2 x'010000000000000000A000000000000000000000002002009111111544160900'
                  35                   35 x'0100000000000000013000000000000000000000002002009111111544259300'
                  33                    0 x'0100000000000000012000000000000000000000002002009111111544253100'
                  32                    0 x'0100000000000000024000000000000000000000002002009111111593657800'
                  29                    0 x'010000000000000000E000000000000000000000002002009111111544220300'
                  24                   23 x'0100000000000000040000000000000000000000002002009111111544064000'
                  24                    0 x'0100000000000000011000000000000000000000002002009111111544248400'
                  20                    0 x'0100000000000000030000000000000000000000002002009111112024182800'
                  15                    0 x'0100000000000000050000000000000000000000002002009111111544098400'
                  14                    0 x'0100000000000000080000000000000000000000002002009111111544143700'
                  13                   13 x'010000000000000000F000000000000000000000002002009111111544240600'
                   4                    0 x'0100000000000000010000000000000000000000002002009111111544248400'
                   3                    0 x'0100000000000000018000000000000000000000002002009111111544282800'
                   3                    3 x'010000000000000001F000000000000000000000002002009111111593651500'
                   3                    0 x'0100000000000000029000000000000000000000002002009111111594396800'
                   2                    0 x'0100000000000000015000000000000000000000002002009111111544265600'
                   2                    0 x'0100000000000000017000000000000000000000002002009111111544275000'
                   1                    0 x'0100000000000000016000000000000000000000002002009111111544273400'
                   1                    0 x'0100000000000000028000000000000000000000002002009111111593700000'
                   1                    0 x'010000000000000002A000000000000000000000002002009111111594398400'

  30 record(s) selected.
```

结果

此时，您可以再次使用 MON_GET_PKG_CACHE_STMT 表函数来检索您特别感兴趣的任何语句的语句文本。例如，可以使用以下查询来确定以上所显示的具有最长等待时间的语句:

```
SELECT VARCHAR(STMT_TEXT, 80) AS STMT_TEXT
FROM TABLE(MON_GET_PKG_CACHE_STMT(NULL,x'0100000000000000032000000000000000000000002002009111112032014000',NULL,-2))
     AS T
```

上述查询的输出将类似如下内容:

```
STMT_TEXT
--------------------------------------------------------------------------------
UPDATE EMPLOYEE SET BONUS=10000 WHERE PERF_RATING=1
  1 record(s) selected.
```

第 10 章 逻辑数据组概述

检查监视元素数据时，同时查看多个相关元素通常很有用。*逻辑数据组*是相辅相成的元素的分组。

例如，uow 逻辑数据组包括 **appl_id**（应用程序标识）和 **appl_name**（应用程序名称）之类的元素，您可轻松想像这两个元素如何相互查看。

逻辑数据组通过快照监视界面和（特别是）事件监视器使用。DB2 产品中有超过 1000 个监视元素。检查监视元素时，被强迫一直考虑并专门指定要查看的元素非常讨厌。例如，如果要创建事件监视器，考虑并指定要针对其捕获数据的元素可能非常讨厌。DB2 产品改为将每个事件监视器与一组缺省逻辑数据组相关联。这意味着您不必在 CREATE EVENT MONITOR 语句中指定任何设置，只需要包括与要捕获的事件相关的监视元素。对于写至常规表的事件监视器，您能够灵活地指定要针对其捕获监视元素数据的逻辑数据组。

写至无格式事件（UE）表的事件监视器还会捕获一组缺省监视元素；使用 EVMON_FORMAT_UE_TO_TABLES 过程来生成关联表时，逻辑数据组用于将不同表中的相关元素组合到一起。例如，`lock` 逻辑数据组包含 LOCK_EVENT 表中使用的元素；`participant` 逻辑数据组包含 LOCK_PARTICIPANT 表中使用的元素。

事件监视器逻辑数据组和监视元素

一起检查时通常很有用的监视元素被组合到*逻辑数据组*中。

所有事件监视器按一种方式或另一种方式使用逻辑数据组。对于某些事件监视器类型，可通过指定要记录其信息的逻辑数据组来指定要收集什么信息。逻辑数据组还用于将数据组合到事件监视器生成的输出中；例如，写至表的事件监视器通常为每个监视元素逻辑数据组创建一个表。

下表列示事件监视可能返回的逻辑数据分组和监视元素。

- 第 50 页的『event_db 逻辑数据组』
- 第 53 页的『event_dbheader 逻辑数据组』
- 第 53 页的『event_dbmemuse 逻辑数据组』
- 第 54 页的『event_deadlock 逻辑数据组』
- 第 54 页的『event_detailed_dlconn 逻辑数据组』
- 第 55 页的『event_dlconn 逻辑数据组』
- 第 56 页的『event_histogrambin 逻辑数据组』
- 第 56 页的『event_log_header 逻辑数据组』
- 第 56 页的『event_overflow 逻辑数据组』
- 第 57 页的『event_qstats 逻辑数据组』
- 第 57 页的『event_scstats 逻辑数据组』
- 第 58 页的『event_start 逻辑数据组』
- 第 58 页的『event_stmt 逻辑数据组』
- 第 59 页的『event_stmt_history 逻辑数据组』
- 第 60 页的『event_subsection 逻辑数据组』
- 第 60 页的『event_table 逻辑数据组』
- 第 61 页的『event_tablespace 逻辑数据组』
- 第 62 页的『event_thresholdviolations 逻辑数据组』
- 第 63 页的『event_wlstats 逻辑数据组』
- 第 62 页的『event_wcstats 逻辑数据组』
- 第 64 页的『event_xact 逻辑数据组』
- 第 64 页的『evmonstart 逻辑数据组』
- 第 65 页的『lock 逻辑数据组』
- 第 66 页的『lock_participants 逻辑数据组』
- 第 65 页的『lock_participant_activities 逻辑数据组』
- 第 65 页的『lock_activity_values 逻辑数据组』
- 第 68 页的『pkgcache 逻辑数据组』
- 第 69 页的『pkgcache_metrics 逻辑数据组』
- 第 72 页的『pkgcache_stmt_args 逻辑数据组』
- 第 72 页的『regvar 逻辑数据组』
- 第 72 页的『sqlca 逻辑数据组』
- 第 73 页的『txncompletion 逻辑数据组』
- 第 73 页的『uow 逻辑数据组』
- 第 75 页的『uow_metrics 逻辑数据组』
- 第 79 页的『uow_package_list 逻辑数据组』
- 第 74 页的『uow_executable_list 逻辑数据组』
- 第 80 页的『utillocation 逻辑数据组』
- 第 80 页的『utilphase 逻辑数据组』
- 第 80 页的『utilstart 逻辑数据组』
- 第 81 页的『utilstop 逻辑数据组』

changesummary 逻辑数据组

第 754 页的『event_id -“事件标识”监视元素』

第 756 页的『event_type -“事件类型”监视元素』

第 755 页的『event_timestamp -“事件时间戳记”监视元素』

第 896 页的『member -“数据库成员”监视元素』

第 695 页的『coord_member -“协调程序成员”监视元素』

第 1328 页的『utility_invocation_id -“实用程序调用标识”』

第 1333 页的『utility_type -“实用程序类型”』

第 616 页的『appl_id -“应用程序标识”监视元素』

第 620 页的『appl_name -“应用程序名称”监视元素』

第 625 页的『application_handle -“应用程序句柄”监视元素』

第 1188 页的『system_auth_id -“系统授权标识”监视元素』

第 1137 页的『session_auth_id -“会话授权标识”监视元素』

第 662 页的『client_platform -“客户机操作平台”监视元素』

第 664 页的『client_protocol -“客户机通信协议”监视元素』

第 662 页的『client_port_number -“客户机端口号”监视元素』

第 661 页的『client_pid -“客户机进程标识”监视元素』

第 659 页的『client_hostname -“客户机主机名”监视元素』

第 665 页的『client_wrkstnname -“客户机工作站名称”监视元素』

第 657 页的『client_acctng -“客户机记帐字符串”监视元素』

第 664 页的『client_userid -“客户机用户标识”监视元素』

第 658 页的『client_applname -“客户机应用程序名称”监视元素』

第 638 页的『backup_timestamp -“备份时间戳记”』

dbdbmcfg 逻辑数据组

第 754 页的『event_id -“事件标识”监视元素』

第 755 页的『event_timestamp -“事件时间戳记”监视元素』

第 896 页的『member -“数据库成员”监视元素』

第 756 页的『event_type -“事件类型”监视元素』

第 653 页的『cfg_name -“配置名称”』

第 654 页的『cfg_value -“配置值”』

第 654 页的『cfg_value_flags -“配置值标志”』

第 653 页的『cfg_old_value -“配置旧值”』

第 654 页的『cfg_old_value_flags -“配置旧值标志”』

第 652 页的『cfg_collection_type -“配置收集类型”』

第 730 页的『deferred -“延迟”』

ddlstmtexec 逻辑数据组

第 754 页的『event_id -“事件标识”监视元素』

第 755 页的『event_timestamp -“事件时间戳记”监视元素』

第 896 页的『member -“数据库成员”监视元素』

第 756 页的『event_type -“事件类型”监视元素』

第 794 页的『global_transaction_id -“全局事务标识”监视元素』

第 841 页的『local_transaction_id -“本地事务标识”监视元素』

第 1125 页的『savepoint_id -“保存点标识”』

第 1316 页的『uow_id -“工作单元标识”监视元素』

第 725 页的『ddl_classification -“DDL 分类”』

第 1178 页的『stmt_text -“SQL 语句文本”监视元素』

dllock 逻辑数据组

第 713 页的『data_partition_id -“数据分区标识”监视元素』

第 842 页的『lock_attributes -“锁定属性”监视元素』

第 843 页的『lock_count -“锁定计数”监视元素』

第 844 页的『lock_current_mode -“转换前的原始锁定方式”监视元素』

第 845 页的『lock_escalation -“锁定升级”监视元素』

第 851 页的『lock_hold_count -“锁定挂起计数”监视元素』

第 852 页的『lock_mode -“锁定方式”监视元素』

第 854 页的『lock_name -“锁定名称”监视元素』

第 855 页的『lock_object_name -“锁定对象名称”』

第 856 页的『lock_object_type -“等待的锁定对象类型”监视元素』

第 858 页的『lock_release_flags -“锁定释放标志”监视元素』

第 858 页的『lock_status -“锁定状态”监视元素』

第 908 页的『node_number -“节点号”』

第 1190 页的『table_file_id -“表文件标识”监视元素』

第 1191 页的『table_name -“表名”监视元素』

第 1193 页的『table_schema -“表模式名”监视元素』

第 1200 页的『tablespace_name -“表空间名称”监视元素』

注: 此逻辑数据组的底层实现是快照监视器 LOCK 逻辑数据组。如果在用于文件和管道输出选项的自描述流中检查此逻辑组的输出，那么可见到 LOCK 组用于生成该输出。

event_activity 逻辑数据组

第 591 页的『act_exec_time -“活动执行时间”监视元素』

第 595 页的『activate_timestamp -“激活时间戳记”监视元素』

第 597 页的『activity_id -“活动标识”监视元素』

第 597 页的『activity_secondary_id -“活动辅助标识”监视元素』

第 598 页的『activity_type -“活动类型”监视元素』

第 600 页的『address - 从中发起连接的 IP 地址』

第 601 页的『agent_id -“应用程序句柄（代理程序标识）”监视元素』

第 616 页的『appl_id -“应用程序标识”监视元素』

第 620 页的『appl_name -“应用程序名称”监视元素』

第 626 页的『arm_correlator -“应用程序响应测量相关因子”监视元素』

第 696 页的『coord_partition_num -“协调程序分区号”监视元素』

第 722 页的『db_work_action_set_id -“数据库工作操作集标识”监视元素』

第 723 页的『db_work_class_id -“数据库工作类标识”监视元素』

details_xml（此 XML 文档是一个 activity_metrics 类型的度量值文档，如 XML 模式文档 sqllib/misc/DB2MonCommon.xsd 中所述。）还可通过 event_activitymetrics 逻辑数据组访问此文档中报告的度量值。

第 827 页的『intra_parallel_state -“分区内并行性的当前状态”监视元素』

第 919 页的『num_remaps -“重新映射次数”监视元素』

第 948 页的『parent_activity_id -“父活动标识”监视元素』

第 948 页的『parent_uow_id -“父工作单元标识”监视元素』

第 949 页的『partial_record -“部分记录”监视元素』

第 982 页的『pool_data_l_reads -“缓冲池数据逻辑读取数”监视元素』

第 984 页的『pool_data_p_reads -“缓冲池数据物理读取数”监视元素』

第 1091 页的『query_actual_degree -“实际运行时分区内并行度”监视元素』

第 1125 页的『sc_work_action_set_id -“服务类工作操作集标识”监视元素』

第 1125 页的『sc_work_class_id -“服务类工作类标识”监视元素』

第 1127 页的『section_actuals -“部分实际值”监视元素』

第 1135 页的『service_subclass_name -“服务子类名”监视元素』

第 1136 页的『service_superclass_name -“服务超类名”监视元素』

第 1137 页的『session_auth_id -“会话授权标识”监视元素』

第 1149 页的『sort_overflows -“排序溢出数”监视元素』

第 1154 页的『sqlca -“SQL 通信区（SQLCA）”』

第 1228 页的『time_completed -“完成时间”监视元素』

第 1228 页的『time_created -“创建时间”监视元素』

第 1229 页的『time_started -“开始时间”监视元素』

第 1291 页的『total_sort_time -“排序时间总计”监视元素』

第 1292 页的『total_sorts -“排序总数”监视元素』

第 1303 页的『tpmon_acc_str -“TP 监视器客户机记帐字符串”监视元素』

第 1304 页的『tpmon_client_app -“TP 监视器客户机应用程序名称”监视元素』

第 1304 页的『tpmon_client_userid -“TP 监视器客户机用户标识”监视元素』

第 1305 页的『tpmon_client_wkstn -“TP 监视器客户机工作站名称”监视元素』

第 1316 页的『uow_id -“工作单元标识”监视元素』

第 1341 页的『workload_id -“工作负载标识”监视元素』

第 1343 页的『workload_occurrence_id -“工作负载项标识”监视元素』

第 1012 页的『pool_index_l_reads -“缓冲池索引逻辑读取数”监视元素』

第 1014 页的『pool_index_p_reads -“缓冲池索引物理读取数”监视元素』

第 1046 页的『pool_temp_data_l_reads -“缓冲池临时数据逻辑读取数”监视元素』

第 1048 页的『pool_temp_data_p_reads -“缓冲池临时数据物理读取数”监视元素』

第 1050 页的『pool_temp_index_l_reads -“缓冲池临时索引逻辑读取数”监视元素』

第 1052 页的『pool_temp_index_p_reads -“缓冲池临时索引物理读取数”监视元素』

第 1053 页的『pool_temp_xda_l_reads -“缓冲池临时 XDA 数据逻辑读取数”监视元素』

第 1055 页的『pool_temp_xda_p_reads -“缓冲池临时 XDA 数据物理读取数”监视元素』

第 1065 页的『pool_xda_l_reads -“缓冲池 XDA 数据逻辑读取数”监视元素』

第 1069 页的『pool_xda_p_reads -“缓冲池 XDA 数据物理读取数”监视元素』

第 1083 页的『prep_time -“编译时间”监视元素』

第 1091 页的『query_card_estimate -“行查询数估计”』

第 1092 页的『query_cost_estimate -“查询估算成本”监视元素』

第 1116 页的『rows_fetched -“访存的行数”监视元素』

第 1117 页的『rows_modified -“修改的行数”监视元素』

第 1120 页的『rows_returned -“返回的行数”监视元素』

第 1189 页的『system_cpu_time -“系统 CPU 时间”监视元素』

第 1326 页的『user_cpu_time -“用户 CPU 时间”监视元素』

第 1335 页的『wl_work_action_set_id -“工作负载工作操作集标识”监视元素』

第 1336 页的『wl_work_class_id -“工作负载工作类标识”监视元素』

第 905 页的『mon_interval_id -“监视时间间隔标识”监视元素』

第 896 页的『member -“数据库成员”监视元素』

第 1093 页的『query_data_tag_list -“估算的查询数据标记列表”监视元素』

第 1294 页的『total_stats_fabrication_time -“统计信息生成时间总计”监视元素』

第 1295 页的『total_stats_fabrications -“统计信息生成总计”监视元素』

第 1299 页的『total_sync_runstats -“同步 RUNSTATS 活动总数”监视元素』

第 1296 页的『total_sync_runstats_time -“同步 RUNSTATS 时间总计”监视元素』

event_activitymetrics 逻辑数据组

第 628 页的『audit_events_total -“审计事件总数”监视元素』

第 631 页的『audit_file_writes_total -“写审计文件总次数”监视元素』

第 632 页的『audit_subsystem_wait_time -“审计子系统等待时间”监视元素』

第 634 页的『audit_subsystem_waits_total -“审计子系统等待总次数”监视元素』

第 696 页的『coord_stmt_exec_time -“协调代理程序执行语句的时间”监视元素』

第 728 页的『deadlocks -“检测到的死锁数”监视元素』

第 732 页的『diaglog_write_wait_time -“诊断日志文件写等待时间”监视元素』

第 734 页的『diaglog_writes_total -“写诊断日志文件总次数”监视元素』

第 735 页的『direct_read_reqs -“直接读请求数”监视元素』

第 737 页的『direct_read_time -“直接读时间”监视元素』

第 738 页的『direct_reads -“直接读数据库数目”监视元素』

第 740 页的『direct_write_reqs -“直接写请求数”监视元素』

第 742 页的『direct_write_time -“直接写时间”监视元素』

第 744 页的『direct_writes -“直接写数据库数目”监视元素』

第 766 页的『fcm_message_recv_volume -“接收 FCM 消息量”监视元素』

第 768 页的『fcm_message_recv_wait_time -“接收 FCM 消息等待时间”监视元素』

第 769 页的『fcm_message_recvs_total -“接收 FCM 消息总数”监视元素』

第 770 页的『fcm_message_send_volume -“发送 FCM 消息量”监视元素』
第 772 页的『fcm_message_send_wait_time -“发送 FCM 消息等待时间”监视元素』
第 773 页的『fcm_message_sends_total -“发送 FCM 消息总数”监视元素』
第 774 页的『fcm_recv_volume -“FCM 接收量”监视元素』
第 775 页的『fcm_recv_wait_time -“FCM 接收等待时间”监视元素』
第 777 页的『fcm_recvs_total -“FCM 接收总计”监视元素』
第 778 页的『fcm_send_volume -“FCM 发送量”监视元素』
第 779 页的『fcm_send_wait_time -“FCM 发送等待时间”监视元素』
第 780 页的『fcm_sends_total -“FCM 发送总计”监视元素』
第 782 页的『fcm_tq_recv_volume -“FCM 表队列接收量”监视元素』
第 783 页的『fcm_tq_recv_wait_time -“FCM 表队列接收等待时间”监视元素』
第 784 页的『fcm_tq_recvs_total -“FCM 表队列接收总量”监视元素』
第 786 页的『fcm_tq_send_volume -“FCM 表队列发送量”监视元素』
第 787 页的『fcm_tq_send_wait_time -“FCM 表队列发送等待时间”监视元素』
第 788 页的『fcm_tq_sends_total -“FCM 表队列发送总次数”监视元素』
第 845 页的『lock_escals -“锁定升级次数”监视元素』
第 860 页的『lock_timeouts -“锁定超时次数”监视元素』
第 863 页的『lock_wait_time -“等待锁定时间”监视元素』
第 868 页的『lock_waits -“等待锁定次数”监视元素』
第 872 页的『log_buffer_wait_time -“日志缓冲区等待时间”监视元素』
第 874 页的『log_disk_wait_time -“日志磁盘等待时间”监视元素』
第 875 页的『log_disk_waits_total -“日志磁盘等待总次数”监视元素』
第 913 页的『num_log_buffer_full -“日志缓冲区变满而导致代理程序等待的次数”监视元素』
第 917 页的『num_lw_thresh_exceeded -“超过锁定等待阈值的次数”监视元素』
第 982 页的『pool_data_l_reads -“缓冲池数据逻辑读取数”监视元素』
第 984 页的『pool_data_p_reads -“缓冲池数据物理读取数”监视元素』
第 986 页的『pool_data_writes -“缓冲池数据写次数”监视元素』
第 1012 页的『pool_index_l_reads -“缓冲池索引逻辑读取数”监视元素』
第 1014 页的『pool_index_p_reads -“缓冲池索引物理读取数”监视元素』
第 1016 页的『pool_index_writes -“缓冲池索引写次数”监视元素』
第 1043 页的『pool_read_time -“缓冲池物理读时间总计”监视元素』
第 1046 页的『pool_temp_data_l_reads -“缓冲池临时数据逻辑读取数”监视元素』
第 1048 页的『pool_temp_data_p_reads -“缓冲池临时数据物理读取数”监视元素』
第 1050 页的『pool_temp_index_l_reads -“缓冲池临时索引逻辑读取数”监视元素』
第 1052 页的『pool_temp_index_p_reads -“缓冲池临时索引物理读取数”监视元素』
第 1053 页的『pool_temp_xda_l_reads -“缓冲池临时 XDA 数据逻辑读取数”监视元素』
第 1055 页的『pool_temp_xda_p_reads -“缓冲池临时 XDA 数据物理读取数”监视元素』
第 1058 页的『pool_write_time -“缓冲池物理写时间总计”监视元素』

第 1065 页的『pool_xda_l_reads -“缓冲池 XDA 数据逻辑读取数”监视元素』

第 1069 页的『pool_xda_p_reads -“缓冲池 XDA 数据物理读取数”监视元素』

第 1071 页的『pool_xda_writes -“缓冲池 XDA 数据写次数”监视元素』

第 1073 页的『post_shrthreshold_sorts -“共享阈值后排序数”监视元素』

第 1079 页的『post_threshold_sorts -“超出阈值后的排序次数”监视元素』

第 1117 页的『rows_modified -“修改的行数”监视元素』

第 1118 页的『rows_read -“读取行数”监视元素』

第 1120 页的『rows_returned -“返回的行数”监视元素』

第 1149 页的『sort_overflows -“排序溢出数”监视元素』

第 1168 页的『stmt_exec_time -“语句执行时间”监视元素』

第 1223 页的『thresh_violations -“阈值违例次数”监视元素』

第 1231 页的『total_act_time -“活动时间总计”监视元素』

第 1232 页的『total_act_wait_time -“活动等待时间总计”监视元素』

第 1235 页的『total_app_section_executions -“应用程序执行部分执行的总次数”监视元素』

第 1249 页的『total_cpu_time -“CPU 时间总计”监视元素』

第 1272 页的『total_routine_invocations -“例程调用总计”监视元素』

第 1274 页的『total_routine_non_sect_proc_time -“非部分处理时间” 监视元素』

第 1274 页的『total_routine_non_sect_time -“非部分例程执行时间” 监视元素』

第 1275 页的『total_routine_time -“例程时间总计”监视元素』

第 1276 页的『total_routine_user_code_proc_time -“例程用户代码处理时间总计”监视元素』

第 1278 页的『total_routine_user_code_time -“例程用户代码时间总计”监视元素』

第 1284 页的『total_section_proc_time -“部分处理时间总计”监视元素』

第 1285 页的『total_section_sort_proc_time -“节排序处理时间总计”监视元素』

第 1287 页的『total_section_sort_time -“节排序时间总计”监视元素』

第 1288 页的『total_section_sorts -“节排序总次数”监视元素』

第 1289 页的『total_section_time -“部分时间总计”监视元素』

第 1292 页的『total_sorts -“排序总数”监视元素』

第 1311 页的『tq_tot_send_spills -“溢出表队列缓冲区总数”监视元素』

第 1336 页的『wlm_queue_assignments_total -“工作负载管理器队列分配总次数”监视元素』

第 1337 页的『wlm_queue_time_total -“工作负载管理器队列时间总计”监视元素』

event_activitystmt 逻辑数据组

第 595 页的『activate_timestamp -“激活时间戳记”监视元素』

第 597 页的『activity_id -“活动标识”监视元素』

第 597 页的『activity_secondary_id -“活动辅助标识”监视元素』

第 616 页的『appl_id -“应用程序标识”监视元素』

第 669 页的『comp_env_desc -“编译环境”监视元素』

第 709 页的『creator -“应用程序创建者”』

第 749 页的『eff_stmt_text -“有效语句文本”监视元素』

第 762 页的『executable_id -“可执行文件标识”监视元素』

第 827 页的『intra_parallel_state -“分区内并行性的当前状态”监视元素』

第 941 页的『package_name -“程序包名”监视元素』

第 943 页的『package_version_id -“程序包版本”监视元素』

第 1091 页的『query_actual_degree -“实际运行时分区内并行度”监视元素』

第 1115 页的『routine_id -“例程标识”监视元素』

第 1128 页的『section_env -“节环境”监视元素』

第 1128 页的『section_number -“节号”监视元素』

第 1169 页的『stmt_first_use_time -“第一次使用语句时的时间戳记”监视元素』

第 1170 页的『stmt_invocation_id -“语句调用标识”监视元素』

第 1171 页的『stmt_isolation -“语句隔离”』

第 1171 页的『stmt_last_use_time -“上一次使用语句时的时间戳记”监视元素』

第 1172 页的『stmt_lock_timeout -“语句锁定超时”监视元素』

第 1172 页的『stmt_nest_level -“语句嵌套级别”监视元素』

第 1174 页的『stmt_pkgcache_id -“语句程序包高速缓存标识”监视元素』

第 1175 页的『stmt_query_id -“语句查询标识”监视元素』

第 1176 页的『stmt_source_id -“语句源标识”』

第 1178 页的『stmt_text -“SQL 语句文本”监视元素』

第 1179 页的『stmt_type -“语句类型”监视元素』

第 1316 页的『uow_id -“工作单元标识”监视元素』

第 896 页的『member -“数据库成员”监视元素』

event_activityvals 逻辑数据组

第 595 页的『activate_timestamp -“激活时间戳记”监视元素』

第 597 页的『activity_id -“活动标识”监视元素』

第 597 页的『activity_secondary_id -“活动辅助标识”监视元素』

第 616 页的『appl_id -“应用程序标识”监视元素』

第 827 页的『intra_parallel_state -“分区内并行性的当前状态”监视元素』

第 1091 页的『query_actual_degree -“实际运行时分区内并行度”监视元素』

第 1181 页的『stmt_value_data -“值数据”』

第 1182 页的『stmt_value_index -“值索引”』

第 1182 页的『stmt_value_isnull -“包含空值”监视元素』

第 1183 页的『stmt_value_isreopt -“用于语句重新优化的变量”监视元素』

第 1184 页的『stmt_value_type -“值类型”监视元素』

第 1316 页的『uow_id -“工作单元标识”监视元素』

第 896 页的『member -“数据库成员”监视元素』

event_bufferpool 逻辑数据组

第 642 页的『bp_id -“缓冲池标识”监视元素』

第 642 页的『bp_name -“缓冲池名称”监视元素』

第 1334 页的『vectored_ios -“向量 I/O 请求数”监视元素』

event_conn 逻辑数据组

第 588 页的『acc_curs_blk -“接受的块游标请求数”』
第 601 页的『agent_id -“应用程序句柄（代理程序标识）”监视元素』
第 616 页的『appl_id -“应用程序标识”监视元素』
第 621 页的『appl_priority -“应用程序代理程序优先级”』
第 621 页的『appl_priority_type -“应用程序优先级类型”』
第 622 页的『appl_section_inserts -“节插入数”监视元素』
第 622 页的『appl_section_lookups -“节查询数”』
第 636 页的『authority_bitmap -“用户权限级别”监视元素』
第 636 页的『authority_lvl -“用户权限级别”监视元素』
第 639 页的『binds_precompiles -“尝试的绑定次数/预编译次数”』
第 646 页的『cat_cache_inserts -“目录高速缓存插入数”监视元素』
第 647 页的『cat_cache_lookups -“目录高速缓存查询数”监视元素』
第 649 页的『cat_cache_overflows -“目录高速缓存溢出数”』
第 668 页的『commit_sql_stmts -“尝试的落实语句数”』
第 726 页的『ddl_sql_stmts -“数据定义语言（DDL）SQL 语句数”』
第 728 页的『deadlocks -“检测到的死锁数”监视元素』
第 735 页的『direct_read_reqs -“直接读请求数”监视元素』
第 737 页的『direct_read_time -“直接读时间”监视元素』
第 738 页的『direct_reads -“直接读数据库数目”监视元素』
第 740 页的『direct_write_reqs -“直接写请求数”监视元素』
第 742 页的『direct_write_time -“直接写时间”监视元素』
第 744 页的『direct_writes -“直接写数据库数目”监视元素』
第 747 页的『disconn_time -“数据库释放时间戳记”』
第 748 页的『dynamic_sql_stmts -“尝试的动态 SQL 语句数”』
第 764 页的『failed_sql_stmts -“失败的语句操作”』
第 809 页的『hash_join_overflows -“散列连接溢出数”』
第 809 页的『hash_join_small_overflows -“散列连接小溢出数”』
第 820 页的『int_auto_rebinds -“内部自动重新绑定次数”』
第 821 页的『int_commits -“内部落实数”监视元素』
第 823 页的『int_deadlock_rollbacks -“死锁导致的内部回滚数”』
第 823 页的『int_rollbacks -“内部回滚数”监视元素』
第 825 页的『int_rows_deleted -“删除的内部行数”』
第 826 页的『int_rows_inserted -“插入的内部行数”』
第 826 页的『int_rows_updated -“更新的内部行数”』
第 845 页的『lock_escalation -“锁定升级”监视元素』
第 860 页的『lock_timeouts -“锁定超时次数”监视元素』
第 863 页的『lock_wait_time -“等待锁定时间”监视元素』

第 1283 页的『total_sec_cons -“辅助连接数”』

第 1291 页的『total_sort_time -“排序时间总计”监视元素』

第 1292 页的『total_sorts -“排序总数”监视元素』

第 1313 页的『uid_sql_stmts -“执行的 Update/Insert/Delete SQL 语句数”』

第 1314 页的『unread_prefetch_pages -“未读取的预取页数”监视元素』

第 1326 页的『user_cpu_time -“用户 CPU 时间”监视元素』

第 1344 页的『x_lock_escals -“互斥锁定升级数”监视元素』

第 1346 页的『xquery_stmts -“尝试的 XQuery 语句数”』

第 623 页的『appl_status - 应用程序状态监视元素』

第 650 页的『cat_cache_size_top -“目录高速缓存高水位标记”监视元素』

第 695 页的『coord_node -“协调节点”』

第 751 页的『elapsed_exec_time -“语句执行耗用时间”』

第 763 页的『evmon_flushes -“事件监视器清空数”』

第 845 页的『lock_escals -“锁定升级次数”监视元素』

第 1053 页的『pool_temp_xda_l_reads -“缓冲池临时 XDA 数据逻辑读取数”监视元素』

第 1055 页的『pool_temp_xda_p_reads -“缓冲池临时 XDA 数据物理读取数”监视元素』

第 1065 页的『pool_xda_l_reads -“缓冲池 XDA 数据逻辑读取数”监视元素』

第 1069 页的『pool_xda_p_reads -“缓冲池 XDA 数据物理读取数”监视元素』

第 1071 页的『pool_xda_writes -“缓冲池 XDA 数据写次数”监视元素』

第 1115 页的『rows_deleted -“删除行数”监视元素』

第 1116 页的『rows_inserted -“插入行数”监视元素』

第 1123 页的『rows_updated -“更新行数”监视元素』

CAT_CACHE_HEAP_FULL

event_connheader 逻辑数据组

第 601 页的『agent_id -“应用程序句柄（代理程序标识）”监视元素』

第 616 页的『appl_id -“应用程序标识”监视元素』

第 620 页的『appl_name -“应用程序名称”监视元素』

第 635 页的『auth_id -“授权标识”』

第 659 页的『client_db_alias -“应用程序使用的数据库别名”』

第 661 页的『client_pid -“客户机进程标识”监视元素』

第 662 页的『client_platform -“客户机操作平台”监视元素』

第 663 页的『client_prdid -“客户机产品和版本标识”监视元素』

第 664 页的『client_protocol -“客户机通信协议”监视元素』

第 666 页的『codepage_id -“应用程序使用的代码页标识”』

第 682 页的『conn_time -“数据库连接时间”监视元素』

第 697 页的『corr_token -“DRDA 关联标记”』

第 764 页的『execution_id -“用户登录标识”』

第 908 页的『node_number -“节点号”』

第 1130 页的『sequence_no -“序号”监视元素』

第 1223 页的『territory_code -“数据库地域代码”』

第 661 页的『client_nname -“客户机名称”监视元素』

event_connmemuse 逻辑数据组

第 908 页的『node_number -“节点号”』

第 974 页的『pool_config_size -“内存池的已配置大小”』

第 975 页的『pool_cur_size -“内存池的当前大小”』

第 1004 页的『pool_id -“内存池标识”』

第 1045 页的『pool_secondary_id -“内存池辅助标记”』

第 1057 页的『pool_watermark -“内存池水位标记”』

第 616 页的『appl_id -“应用程序标识”监视元素』

第 763 页的『evmon_flushes -“事件监视器清空数”』

POOL_LIST_ID

POOL_MAX_SIZE

event_data_value 逻辑数据组

第 727 页的『deadlock_id -“死锁事件标识”』

第 727 页的『deadlock_node -“发生死锁的分区号”』

第 757 页的『evmon_activates -“事件监视器激活数”』

第 950 页的『participant_no -“死锁参与者”』

第 1169 页的『stmt_history_id -“语句历史记录标识”』

第 1181 页的『stmt_value_data -“值数据”』

第 1182 页的『stmt_value_index -“值索引”』

第 1182 页的『stmt_value_isnull -“包含空值”监视元素』

第 1183 页的『stmt_value_isreopt -“用于语句重新优化的变量”监视元素』

第 1184 页的『stmt_value_type -“值类型”监视元素』

event_db 逻辑数据组

第 596 页的『active_hash_joins -“活动散列连接数”』

第 622 页的『appl_section_inserts -“节插入数”监视元素』

第 622 页的『appl_section_lookups -“节查询数”』

第 627 页的『async_runstats -“异步 RUNSTATS 请求总数”监视元素』

第 639 页的『binds_precompiles -“尝试的绑定次数/预编译次数”』

第 641 页的『blocks_pending_cleanup -“暂挂清除已转出块”监视元素』

第 646 页的『cat_cache_inserts -“目录高速缓存插入数”监视元素』

第 647 页的『cat_cache_lookups -“目录高速缓存查询数”监视元素』

第 649 页的『cat_cache_overflows -“目录高速缓存溢出数”』

第 650 页的『cat_cache_size_top -“目录高速缓存高水位标记”监视元素』

第 650 页的『catalog_node -“目录节点号”』

第 651 页的『catalog_node_name -“目录节点网络名”』

第 668 页的『commit_sql_stmts -“尝试的落实语句数”』
第 684 页的『connections_top -“最大并行连接数”』
第 718 页的『db_heap_top -“分配的最大数据库堆”』
第 726 页的『ddl_sql_stmts -“数据定义语言（DDL）SQL 语句数”』
第 728 页的『deadlocks -“检测到的死锁数”监视元素』
第 735 页的『direct_read_reqs -“直接读请求数”监视元素』
第 737 页的『direct_read_time -“直接读时间”监视元素』
第 738 页的『direct_reads -“直接读数据库数目”监视元素』
第 740 页的『direct_write_reqs -“直接写请求数”监视元素』
第 742 页的『direct_write_time -“直接写时间”监视元素』
第 744 页的『direct_writes -“直接写数据库数目”监视元素』
第 747 页的『disconn_time -“数据库释放时间戳记”』
第 748 页的『dynamic_sql_stmts -“尝试的动态 SQL 语句数”』
第 757 页的『evmon_activates -“事件监视器激活数”』
第 763 页的『evmon_flushes -“事件监视器清空数”』
第 764 页的『failed_sql_stmts -“失败的语句操作”』
第 790 页的『files_closed -“关闭数据库文件数”监视元素』
第 809 页的『hash_join_overflows -“散列连接溢出数”』
第 809 页的『hash_join_small_overflows -“散列连接小溢出数”』
第 820 页的『int_auto_rebinds -“内部自动重新绑定次数”』
第 821 页的『int_commits -“内部落实数”监视元素』
第 823 页的『int_rollbacks -“内部回滚数”监视元素』
第 825 页的『int_rows_deleted -“删除的内部行数”』
第 826 页的『int_rows_inserted -“插入的内部行数”』
第 826 页的『int_rows_updated -“更新的内部行数”』
第 845 页的『lock_escals -“锁定升级次数”监视元素』
第 860 页的『lock_timeouts -“锁定超时次数”监视元素』
第 863 页的『lock_wait_time -“等待锁定时间”监视元素』
第 868 页的『lock_waits -“等待锁定次数”监视元素』
第 876 页的『log_held_by_dirty_pages -“脏页占用的日志空间量”』
第 877 页的『log_read_time -“日志读取时间”』
第 877 页的『log_reads -“读取的日志页数”』
第 878 页的『log_to_redo_for_recovery -“要为恢复重做的日志量”』
第 878 页的『log_write_time -“日志写入时间”』
第 879 页的『log_writes -“写入的日志页数”』
第 916 页的『num_log_read_io -“日志读取数”』
第 916 页的『num_log_write_io -“日志写入次数”』
第 919 页的『num_threshold_violations -“阈值违例次数”监视元素』
第 933 页的『olap_func_overflows -“OLAP 函数溢出次数”监视元素』
第 949 页的『partial_record -“部分记录”监视元素』

第 1123 页的『rows_updated -“更新行数”监视元素』
第 1126 页的『sec_log_used_top -“使用的最大辅助日志空间”』
第 1129 页的『select_sql_stmts -“执行的 Select SQL 语句数”』
第 1132 页的『server_platform -“服务器操作系统”』
第 1138 页的『shr_workspace_num_overflows -“共享工作空间溢出数”』
第 1138 页的『shr_workspace_section_inserts -“共享工作空间节插入数”』
第 1139 页的『shr_workspace_section_lookups -“共享工作空间节查询数”』
第 1140 页的『shr_workspace_size_top -“最大共享工作空间大小”』
第 1149 页的『sort_overflows -“排序溢出数”监视元素』
第 1165 页的『static_sql_stmts -“尝试的静态 SQL 语句数”』
第 1165 页的『stats_cache_size -“统计信息高速缓存大小”监视元素』
第 1166 页的『stats_fabricate_time -“生成统计信息的活动所花的总时间”监视元素』
第 1167 页的『stats_fabrications -“生成统计信息的次数”监视元素』
第 1187 页的『sync_runstats -“同步 RUNSTATS 活动总数”监视元素』
第 1188 页的『sync_runstats_time -“同步 RUNSTATS 活动所花的总时间”监视元素』
第 1230 页的『tot_log_used_top -“使用的最大总日志空间”』
第 1243 页的『total_cons -“数据库激活以后的连接数”』
第 1256 页的『total_hash_joins -“散列连接总数”』
第 1256 页的『total_hash_loops -“总散列循环数”』
第 1264 页的『total_olap_funcs -“OLAP 函数总数”监视元素』
第 1291 页的『total_sort_time -“排序时间总计”监视元素』
第 1292 页的『total_sorts -“排序总数”监视元素』
第 1313 页的『uid_sql_stmts -“执行的 Update/Insert/Delete SQL 语句数”』
第 1314 页的『unread_prefetch_pages -“未读取的预取页数”监视元素』
第 1344 页的『x_lock_escals -“互斥锁定升级数”监视元素』
第 1346 页的『xquery_stmts -“尝试的 XQuery 语句数”』
第 751 页的『elapsed_exec_time -“语句执行耗用时间”』
第 915 页的『num_log_part_page_io -“部分日志页写入数”』
第 972 页的『pool_async_xda_read_reqs -“缓冲池异步 XDA 读请求数”监视元素』
第 973 页的『pool_async_xda_reads -“缓冲池异步 XDA 数据读取数”监视元素』
第 974 页的『pool_async_xda_writes -“缓冲池异步 XDA 数据写次数”监视元素』
第 1053 页的『pool_temp_xda_l_reads -“缓冲池临时 XDA 数据逻辑读取数”监视元素』
第 1055 页的『pool_temp_xda_p_reads -“缓冲池临时 XDA 数据物理读取数”监视元素』
第 1065 页的『pool_xda_l_reads -“缓冲池 XDA 数据逻辑读取数”监视元素』
第 1069 页的『pool_xda_p_reads -“缓冲池 XDA 数据物理读取数”监视元素』
第 1071 页的『pool_xda_writes -“缓冲池 XDA 数据写次数”监视元素』
第 1151 页的『sort_shrheap_top -“排序共享堆高水位标记”』
CAT_CACHE_HEAP_FULL
LOG_FILE_ARCHIVE

LOG_FILE_NUM_CURR

LOG_FILE_NUM_FIRST

LOG_FILE_NUM_LAST

NUM_LOG_BUFF_FULL

NUM_LOG_DATA_IN_BUFF

event_dbheader 逻辑数据组

第 682 页的『conn_time -“数据库连接时间”监视元素』

第 719 页的『db_name - 数据库名称监视元素』

第 720 页的『db_path -“数据库路径”』

event_dbmemuse 逻辑数据组

第 908 页的『node_number -“节点号”』

第 974 页的『pool_config_size -“内存池的已配置大小”』

第 975 页的『pool_cur_size -“内存池的当前大小”』

第 1004 页的『pool_id -“内存池标识”』

第 1057 页的『pool_watermark -“内存池水位标记”』

第 757 页的『evmon_activates -“事件监视器激活数”』

第 763 页的『evmon_flushes -“事件监视器清空数”』

第 1045 页的『pool_secondary_id -“内存池辅助标记”』

POOL_MAX_SIZE

event_deadlock 逻辑数据组

第 727 页的『deadlock_id -“死锁事件标识”』

第 727 页的『deadlock_node -“发生死锁的分区号”』

第 748 页的『dl_conns -“死锁中涉及的连接数”监视元素』

第 757 页的『evmon_activates -“事件监视器激活数”』

第 1113 页的『rolled_back_agent_id -“回滚的代理程序”』

第 1114 页的『rolled_back_appl_id -“回滚的应用程序”』

第 1114 页的『rolled_back_participant_no -“回滚的应用程序参与者”监视元素』

第 1114 页的『rolled_back_sequence_no -“回滚的序号”』

第 1164 页的『start_time -“事件启动时间”』

event_detailed_dlconn 逻辑数据组

第 601 页的『agent_id -“应用程序句柄（代理程序标识）”监视元素』

第 616 页的『appl_id -“应用程序标识”监视元素』

第 618 页的『appl_id_holding_lk -“挂起锁定的应用程序标识”』

第 640 页的『blocking_cursor -“分块游标”』

第 684 页的『consistency_token -“程序包一致性标记”监视元素』

第 709 页的『creator -“应用程序创建者”』

第 711 页的『cursor_name -“游标名称”』

第 713 页的『data_partition_id -“数据分区标识”监视元素』

第 727 页的『deadlock_id -“死锁事件标识”』
第 727 页的『deadlock_node -“发生死锁的分区号”』
第 757 页的『evmon_activates -“事件监视器激活数”』
第 845 页的『lock_escalation -“锁定升级”监视元素』
第 852 页的『lock_mode -“锁定方式”监视元素』
第 853 页的『lock_mode_requested -“请求的锁定方式”监视元素』
第 855 页的『lock_node -“锁定节点”』
第 855 页的『lock_object_name -“锁定对象名称”』
第 856 页的『lock_object_type -“等待的锁定对象类型”监视元素』
第 863 页的『lock_wait_start_time -“锁定等待开始时间戳记”监视元素』
第 871 页的『locks_held -“挂起的锁定数”监视元素』
第 872 页的『locks_in_list -“报告的锁定数”』
第 941 页的『package_name -“程序包名”监视元素』
第 943 页的『package_version_id -“程序包版本”监视元素』
第 950 页的『participant_no -“死锁参与者”』
第 950 页的『participant_no_holding_lk -“对应用程序所需对象挂起锁定的参与者”』
第 1128 页的『section_number -“节号”监视元素』
第 1130 页的『sequence_no -“序号”监视元素』
第 1131 页的『sequence_no_holding_lk -“挂起锁定的序号”』
第 1164 页的『start_time -“事件启动时间”』
第 1173 页的『stmt_operation/operation -“语句操作”监视元素』
第 1178 页的『stmt_text -“SQL 语句文本”监视元素』
第 1179 页的『stmt_type -“语句类型”监视元素』
第 1191 页的『table_name -“表名”监视元素』
第 1193 页的『table_schema -“表模式名”监视元素』
第 1200 页的『tablespace_name -“表空间名称”监视元素』
第 842 页的『lock_attributes -“锁定属性”监视元素』
第 843 页的『lock_count -“锁定计数”监视元素』
第 844 页的『lock_current_mode -“转换前的原始锁定方式”监视元素』
第 851 页的『lock_hold_count -“锁定挂起计数”监视元素』
第 854 页的『lock_name -“锁定名称”监视元素』
第 858 页的『lock_release_flags -“锁定释放标志”监视元素』
第 1303 页的『tpmon_acc_str -“TP 监视器客户机记帐字符串”监视元素』
第 1304 页的『tpmon_client_app -“TP 监视器客户机应用程序名称”监视元素』
第 1304 页的『tpmon_client_userid -“TP 监视器客户机用户标识”监视元素』
第 1305 页的『tpmon_client_wkstn -“TP 监视器客户机工作站名称”监视元素』

event_dlconn 逻辑数据组

第 601 页的『agent_id -“应用程序句柄（代理程序标识）”监视元素』
第 616 页的『appl_id -“应用程序标识”监视元素』

第 618 页的『appl_id_holding_lk -“挂起锁定的应用程序标识”』

第 713 页的『data_partition_id -“数据分区标识”监视元素』

第 727 页的『deadlock_id -“死锁事件标识”』

第 727 页的『deadlock_node -“发生死锁的分区号”』

第 757 页的『evmon_activates -“事件监视器激活数”』

第 842 页的『lock_attributes -“锁定属性”监视元素』

第 843 页的『lock_count -“锁定计数”监视元素』

第 844 页的『lock_current_mode -“转换前的原始锁定方式”监视元素』

第 845 页的『lock_escalation -“锁定升级”监视元素』

第 851 页的『lock_hold_count -“锁定挂起计数”监视元素』

第 852 页的『lock_mode -“锁定方式”监视元素』

第 853 页的『lock_mode_requested -“请求的锁定方式”监视元素』

第 854 页的『lock_name -“锁定名称”监视元素』

第 855 页的『lock_node -“锁定节点”』

第 855 页的『lock_object_name -“锁定对象名称”』

第 856 页的『lock_object_type -“等待的锁定对象类型”监视元素』

第 858 页的『lock_release_flags -“锁定释放标志”监视元素』

第 863 页的『lock_wait_start_time -“锁定等待开始时间戳记”监视元素』

第 950 页的『participant_no -“死锁参与者”』

第 950 页的『participant_no_holding_lk -“对应用程序所需对象挂起锁定的参与者”』

第 1130 页的『sequence_no -“序号”监视元素』

第 1131 页的『sequence_no_holding_lk -“挂起锁定的序号”』

第 1164 页的『start_time -“事件启动时间”』

第 1191 页的『table_name -“表名”监视元素』

第 1193 页的『table_schema -“表模式名”监视元素』

第 1200 页的『tablespace_name -“表空间名称”监视元素』

第 1303 页的『tpmon_acc_str -“TP 监视器客户机记帐字符串”监视元素』

第 1304 页的『tpmon_client_app -“TP 监视器客户机应用程序名称”监视元素』

第 1304 页的『tpmon_client_userid -“TP 监视器客户机用户标识”监视元素』

第 1305 页的『tpmon_client_wkstn -“TP 监视器客户机工作站名称”监视元素』

event_histogrambin 逻辑数据组

第 638 页的『bin_id -“直方图条形标识”监视元素』

第 641 页的『bottom -“直方图类别底部”监视元素』

第 810 页的『histogram_type -“直方图类型”监视元素』

第 921 页的『number_in_bin -“条形中的数目”监视元素』

第 1134 页的『service_class_id -“服务类标识”监视元素』

第 1165 页的『statistics_timestamp -“统计信息时间戳记”监视元素』

第 1230 页的『top -“最大直方图类别数”监视元素』

第 1339 页的『work_action_set_id -“工作操作集标识”监视元素』

第 1340 页的『work_class_id -“工作类标识”监视元素』

第 1341 页的『workload_id -“工作负载标识”监视元素』

第 905 页的『mon_interval_id -“监视时间间隔标识”监视元素』

第 896 页的『member -“数据库成员”监视元素』

event_log_header 逻辑数据组

第 646 页的『byte_order -“事件数据的字节顺序”』

第 666 页的『codepage_id -“应用程序使用的代码页标识”』

第 754 页的『event_monitor_name -“事件监视器名称”』

第 918 页的『num_nodes_in_db2_instance -“分区中的节点数”』

第 1132 页的『server_instance_name -“服务器实例名称”』

第 1133 页的『server_prdid -“服务器产品/版本标识”』

第 1223 页的『territory_code -“数据库地域代码”』

第 1334 页的『version -“监视器数据版本”』

event_overflow 逻辑数据组

第 697 页的『count -“事件监视器溢出数”』

第 791 页的『first_overflow_time -“第一次事件溢出时间”』

第 836 页的『last_overflow_time -“最后一次事件溢出时间”』

第 908 页的『node_number -“节点号”』

event_qstats 逻辑数据组

第 838 页的『last_wlm_reset -“最后一次重置时间”监视元素』

第 1093 页的『queue_assignments_total -“队列分配总次数”监视元素』

第 1094 页的『queue_size_top -“最大队列大小”监视元素』

第 1094 页的『queue_time_total -“总队列时间”监视元素』

第 1135 页的『service_subclass_name -“服务子类名”监视元素』

第 1136 页的『service_superclass_name -“服务超类名”监视元素』

第 1165 页的『statistics_timestamp -“统计信息时间戳记”监视元素』

第 1225 页的『threshold_domain -“阈值域”监视元素』

第 1226 页的『threshold_name -“阈值名称”监视元素』

第 1226 页的『threshold_predicate -“阈值谓词”监视元素』

第 1228 页的『thresholdid -“阈值标识”监视元素』

第 1339 页的『work_action_set_name -“工作操作集名称”监视元素』

第 1340 页的『work_class_name -“工作类名”监视元素』

第 905 页的『mon_interval_id -“监视时间间隔标识”监视元素』

第 896 页的『member -“数据库成员”监视元素』

event_scstats 逻辑数据组

第 590 页的『act_cpu_time_top -“最长活动 CPU 时间”监视元素』

第 592 页的『act_remapped_in -“重新映入的活动数”监视元素』

第 593 页的『act_remapped_out -“重新映出的活动数”监视元素』

第 593 页的『act_rows_read_top -“最大活动读取行数”监视元素』

第 594 页的『act_throughput -“活动吞吐量”监视元素』

第 610 页的『agg_temp_tablespace_top -“最大聚集临时表空间”监视元素』

第 672 页的『concurrent_act_top -“最大并行活动数”监视元素』

第 674 页的『concurrent_wlo_top -“最大并行工作负载项数”监视元素』

第 673 页的『concurrent_connection_top -“最大并行连接数”监视元素』

第 687 页的『coord_act_aborted_total -“异常终止的协调程序活动总数”监视元素』

第 688 页的『coord_act_completed_total -“完成的协调程序活动总数”监视元素』

第 689 页的『coord_act_est_cost_avg -“平均协调程序活动估计成本”监视元素』

第 689 页的『coord_act_exec_time_avg -“平均协调程序活动执行时间”监视元素』

第 690 页的『coord_act_interarrival_time_avg -“平均协调程序活动到达时间”监视元素』

第 691 页的『coord_act_lifetime_avg -“平均协调程序活动生存期”监视元素』

第 692 页的『coord_act_lifetime_top -“协调程序活动生存期项部”监视元素』

第 692 页的『coord_act_queue_time_avg -“平均协调程序活动队列时间”监视元素』

第 693 页的『coord_act_rejected_total -“被拒绝的协调程序活动总数”监视元素』

第 697 页的『cost_estimate_top -“最高估计成本”监视元素』

第 704 页的『cpu_utilization -“CPU 利用率”监视元素』

details_xml（此 XML 文档是一个 system_metrics 类型的文档，如 XML 模式文档 sqllib/misc/DB2MonCommon.xsd 中所述。 为超类 SYSDEFAULTSYSTEMCLASS 下的缺省子类 SYSDEFAULTSUBCLASS 报告的度量的值为 0。）

第 838 页的『last_wlm_reset -“最后一次重置时间”监视元素』

第 1111 页的『request_exec_time_avg -“平均请求执行时间”监视元素』

第 1122 页的『rows_returned_top -“最高实际返回行数”监视元素』

第 1134 页的『service_class_id -“服务类标识”监视元素』

第 1135 页的『service_subclass_name -“服务子类名”监视元素』

第 1136 页的『service_superclass_name -“服务超类名”监视元素』

第 1165 页的『statistics_timestamp -“统计信息时间戳记”监视元素』

第 1222 页的『temp_tablespace_top -“最大临时表空间”监视元素』

第 1249 页的『total_cpu_time -“CPU 时间总计”监视元素』

第 1251 页的『total_disp_run_queue_time -“分派器运行队列时间总计”监视元素』

第 1315 页的『uow_completed_total -“完成的工作单元总数”监视元素』

第 1317 页的『uow_lifetime_avg -“工作单元平均生存期”监视元素』

第 1321 页的『uow_throughput -“工作单元吞吐量”监视元素』

第 1321 页的『uow_total_time_top -“最长 UOW 时间总计”监视元素』

第 612 页的『app_act_aborted_total -“失败的外部协调程序活动总数”监视元素』

第 613 页的『app_act_completed_total -“成功的外部协调程序活动总数”监视元素』

第 614 页的『app_act_rejected_total -“拒绝的外部协调程序活动总数”监视元素』

第 905 页的『mon_interval_id -“监视时间间隔标识”监视元素』

第 896 页的『member -“数据库成员”监视元素』

event_start 逻辑数据组

第 1164 页的『start_time -“事件启动时间”』

event_stmt 逻辑数据组

第 601 页的『agent_id -“应用程序句柄（代理程序标识）”监视元素』

第 609 页的『agents_top -“创建的代理程序数”』

第 616 页的『appl_id -“应用程序标识”监视元素』

第 640 页的『blocking_cursor -“分块游标”』

第 684 页的『consistency_token -“程序包一致性标记”监视元素』

第 709 页的『creator -“应用程序创建者”』

第 711 页的『cursor_name -“游标名称”』

第 789 页的『fetch_count -“成功的访存数”』

第 825 页的『int_rows_deleted -“删除的内部行数”』

第 826 页的『int_rows_inserted -“插入的内部行数”』

第 826 页的『int_rows_updated -“更新的内部行数”』

第 941 页的『package_name -“程序包名”监视元素』

第 943 页的『package_version_id -“程序包版本”监视元素』

第 949 页的『partial_record -“部分记录”监视元素』

第 982 页的『pool_data_l_reads -“缓冲池数据逻辑读取数”监视元素』

第 984 页的『pool_data_p_reads -“缓冲池数据物理读取数”监视元素』

第 1012 页的『pool_index_l_reads -“缓冲池索引逻辑读取数”监视元素』

第 1014 页的『pool_index_p_reads -“缓冲池索引物理读取数”监视元素』

第 1046 页的『pool_temp_data_l_reads -“缓冲池临时数据逻辑读取数”监视元素』

第 1048 页的『pool_temp_data_p_reads -“缓冲池临时数据物理读取数”监视元素』

第 1050 页的『pool_temp_index_l_reads -“缓冲池临时索引逻辑读取数”监视元素』

第 1052 页的『pool_temp_index_p_reads -“缓冲池临时索引物理读取数”监视元素』

第 1118 页的『rows_read -“读取行数”监视元素』

第 1123 页的『rows_written -“写入的行数”』

第 1128 页的『section_number -“节号”监视元素』

第 1130 页的『sequence_no -“序号”监视元素』

第 1149 页的『sort_overflows -“排序溢出数”监视元素』

第 1153 页的『sql_req_id -“SQL 语句的请求标识”』

第 1154 页的『sqlca -“SQL 通信区（SQLCA）”』

第 1164 页的『start_time -“事件启动时间”』

第 1166 页的『stats_fabricate_time -“生成统计信息的活动所花的总时间”监视元素』

第 1173 页的『stmt_operation/operation -“语句操作”监视元素』

第 1178 页的『stmt_text -“SQL 语句文本”监视元素』

第 1179 页的『stmt_type -“语句类型”监视元素』

第 1185 页的『stop_time -“事件停止时间”』

第 1188 页的『sync_runstats_time -“同步 RUNSTATS 活动所花的总时间”监视元素』

第 1189 页的『system_cpu_time -“系统 CPU 时间”监视元素』

第 1291 页的『total_sort_time -“排序时间总计”监视元素』

第 1292 页的『total_sorts -“排序总数”监视元素』

第 1326 页的『user_cpu_time -“用户 CPU 时间”监视元素』

第 763 页的『evmon_flushes -“事件监视器清空数”』

第 1053 页的『pool_temp_xda_l_reads -“缓冲池临时 XDA 数据逻辑读取数”监视元素』

第 1055 页的『pool_temp_xda_p_reads -“缓冲池临时 XDA 数据物理读取数”监视元素』

第 1065 页的『pool_xda_l_reads -“缓冲池 XDA 数据逻辑读取数”监视元素』

第 1069 页的『pool_xda_p_reads -“缓冲池 XDA 数据物理读取数”监视元素』

event_stmt_history 逻辑数据组

第 669 页的『comp_env_desc -“编译环境”监视元素』

第 709 页的『creator -“应用程序创建者”』

第 727 页的『deadlock_id -“死锁事件标识”』

第 727 页的『deadlock_node -“发生死锁的分区号”』

第 757 页的『evmon_activates -“事件监视器激活数”』

第 941 页的『package_name -“程序包名”监视元素』

第 943 页的『package_version_id -“程序包版本”监视元素』

第 950 页的『participant_no -“死锁参与者”』

第 1128 页的『section_number -“节号”监视元素』

第 1130 页的『sequence_no -“序号”监视元素』

第 1169 页的『stmt_first_use_time -“第一次使用语句时的时间戳记”监视元素』

第 1169 页的『stmt_history_id -“语句历史记录标识”』

第 1170 页的『stmt_invocation_id -“语句调用标识”监视元素』

第 1171 页的『stmt_isolation -“语句隔离”』

第 1171 页的『stmt_last_use_time -“上一次使用语句时的时间戳记”监视元素』

第 1172 页的『stmt_lock_timeout -“语句锁定超时”监视元素』

第 1172 页的『stmt_nest_level -“语句嵌套级别”监视元素』

第 1174 页的『stmt_pkgcache_id -“语句程序包高速缓存标识”监视元素』

第 1175 页的『stmt_query_id -“语句查询标识”监视元素』

第 1176 页的『stmt_source_id -“语句源标识”』

第 1178 页的『stmt_text -“SQL 语句文本”监视元素』

第 1179 页的『stmt_type -“语句类型”监视元素』

第 616 页的『appl_id -“应用程序标识”监视元素』

event_subsection 逻辑数据组

第 601 页的『agent_id -“应用程序句柄（代理程序标识）”监视元素』

第 909 页的『num_agents -“正在处理语句的代理程序数”』

第 949 页的『partial_record -“部分记录”监视元素』

第 1161 页的『ss_exec_time -“子节执行耗用时间”』

第 1161 页的『ss_node_number -“子节节点号”』

第 1162 页的『ss_number -“子节号”监视元素』

第 1162 页的『ss_sys_cpu_time -“子节使用的系统 CPU 时间”』

第 1163 页的『ss_usr_cpu_time -“子节使用的用户 CPU 时间”』

第 1306 页的『tq_max_send_spills -“最大表队列缓冲区溢出数”』

第 1307 页的『tq_rows_read -“从表队列读取的行数”』

第 1307 页的『tq_rows_written -“写至表队列的行数”』

第 1311 页的『tq_tot_send_spills -“溢出表队列缓冲区总数”监视元素』

第 616 页的『appl_id -“应用程序标识”监视元素』

第 763 页的『evmon_flushes -“事件监视器清空数”』

第 1153 页的『sql_req_id -“SQL 语句的请求标识”』

event_table 逻辑数据组

第 712 页的『data_object_pages -“数据对象页数”』

第 713 页的『data_partition_id -“数据分区标识”监视元素』

第 755 页的『event_time -“事件时间”』

第 757 页的『evmon_activates -“事件监视器激活数”』

第 763 页的『evmon_flushes -“事件监视器清空数”』

第 817 页的『index_object_pages -“索引对象页数”』

第 839 页的『lob_object_pages -“LOB 对象页数”』

第 880 页的『long_object_pages -“长对象页数”』

第 940 页的『overflow_accesses -“访问溢出记录次数”监视元素』

第 944 页的『page_reorgs -“页重组”监视元素』

第 949 页的『partial_record -“部分记录”监视元素』

第 1118 页的『rows_read -“读取行数”监视元素』

第 1123 页的『rows_written -“写入的行数”』

第 1191 页的『table_name -“表名”监视元素』

第 1193 页的『table_schema -“表模式名”监视元素』

第 1194 页的『table_type -“表类型”监视元素』

第 1197 页的『tablespace_id -“表空间标识”监视元素』

第 1345 页的『xda_object_pages -“XDA 对象页数”』

event_tablespace 逻辑数据组

第 735 页的『direct_read_reqs -“直接读请求数”监视元素』

第 737 页的『direct_read_time -“直接读时间”监视元素』

第 738 页的『direct_reads -“直接读数据库数目”监视元素』

第 740 页的『direct_write_reqs -“直接写请求数”监视元素』

第 742 页的『direct_write_time -“直接写时间”监视元素』

第 744 页的『direct_writes -“直接写数据库数目”监视元素』

第 755 页的『event_time -“事件时间”』

第 757 页的『evmon_activates -“事件监视器激活数”』

第 763 页的『evmon_flushes -“事件监视器清空数”』
第 790 页的『files_closed -“关闭数据库文件数”监视元素』
第 949 页的『partial_record -“部分记录”监视元素』
第 961 页的『pool_async_data_read_reqs -“缓冲池异步读请求数”监视元素』
第 962 页的『pool_async_data_reads -“缓冲池异步数据读次数”监视元素』
第 963 页的『pool_async_data_writes -“缓冲池异步数据写次数”监视元素』
第 966 页的『pool_async_index_read_reqs -“缓冲池异步索引读请求数”监视元素』
第 966 页的『pool_async_index_reads -“缓冲池异步索引读取数”监视元素』
第 967 页的『pool_async_index_writes -“缓冲池异步索引写次数”监视元素』
第 968 页的『pool_async_read_time -“缓冲池异步读取时间”』
第 969 页的『pool_async_write_time -“缓冲池异步写入时间”监视元素』
第 982 页的『pool_data_l_reads -“缓冲池数据逻辑读取数”监视元素』
第 984 页的『pool_data_p_reads -“缓冲池数据物理读取数”监视元素』
第 986 页的『pool_data_writes -“缓冲池数据写次数”监视元素』
第 1012 页的『pool_index_l_reads -“缓冲池索引逻辑读取数”监视元素』
第 1014 页的『pool_index_p_reads -“缓冲池索引物理读取数”监视元素』
第 1016 页的『pool_index_writes -“缓冲池索引写次数”监视元素』
第 1018 页的『pool_no_victim_buffer -“缓冲池无牺牲缓冲区次数”监视元素』
第 1043 页的『pool_read_time -“缓冲池物理读时间总计”监视元素』
第 1046 页的『pool_temp_data_l_reads -“缓冲池临时数据逻辑读取数”监视元素』
第 1048 页的『pool_temp_data_p_reads -“缓冲池临时数据物理读取数”监视元素』
第 1050 页的『pool_temp_index_l_reads -“缓冲池临时索引逻辑读取数”监视元素』
第 1052 页的『pool_temp_index_p_reads -“缓冲池临时索引物理读取数”监视元素』
第 1058 页的『pool_write_time -“缓冲池物理写时间总计”监视元素』
第 1200 页的『tablespace_name -“表空间名称”监视元素』
第 972 页的『pool_async_xda_read_reqs -“缓冲池异步 XDA 读请求数”监视元素』
第 973 页的『pool_async_xda_reads -“缓冲池异步 XDA 数据读取数”监视元素』
第 974 页的『pool_async_xda_writes -“缓冲池异步 XDA 数据写次数”监视元素』
第 1065 页的『pool_xda_l_reads -“缓冲池 XDA 数据逻辑读取数”监视元素』
第 1069 页的『pool_xda_p_reads -“缓冲池 XDA 数据物理读取数”监视元素』
第 1071 页的『pool_xda_writes -“缓冲池 XDA 数据写次数”监视元素』
TABLESPACE_FS_CACHING
第 1314 页的『unread_prefetch_pages -“未读取的预取页数”监视元素』

event_thresholdviolations 逻辑数据组

第 595 页的『activate_timestamp -“激活时间戳记”监视元素』
第 596 页的『activity_collected -“收集的活动”监视元素』
第 597 页的『activity_id -“活动标识”监视元素』
第 601 页的『agent_id -“应用程序句柄（代理程序标识）”监视元素』
第 616 页的『appl_id -“应用程序标识”监视元素』

第 696 页的『coord_partition_num -“协调程序分区号”监视元素』

第 731 页的『destination_service_class_id -“目标服务类标识”监视元素』

第 1152 页的『source_service_class_id -“源服务类标识”监视元素』

第 1225 页的『threshold_action -“阈值操作”监视元素』

第 1226 页的『threshold_maxvalue -“阈值最大值”监视元素』

第 1226 页的『threshold_predicate -“阈值谓词”监视元素』

第 1227 页的『threshold_queuesize -“阈值队列大小”监视元素』

第 1228 页的『thresholdid -“阈值标识”监视元素』

第 1229 页的『time_of_violation -“违例时间”监视元素』

第 1316 页的『uow_id -“工作单元标识”监视元素』

第 896 页的『member -“数据库成员”监视元素』

event_wcstats 逻辑数据组

第 590 页的『act_cpu_time_top -“最长活动 CPU 时间”监视元素』

第 593 页的『act_rows_read_top -“最大活动读取行数”监视元素』

第 595 页的『act_total -“活动总数”监视元素』

第 689 页的『coord_act_est_cost_avg -“平均协调程序活动估计成本”监视元素』

第 689 页的『coord_act_exec_time_avg -“平均协调程序活动执行时间”监视元素』

第 690 页的『coord_act_interarrival_time_avg -“平均协调程序活动到达时间”监视元素』

第 691 页的『coord_act_lifetime_avg -“平均协调程序活动生存期”监视元素』

第 692 页的『coord_act_lifetime_top -“协调程序活动生存期项部”监视元素』

第 692 页的『coord_act_queue_time_avg -“平均协调程序活动队列时间”监视元素』

第 697 页的『cost_estimate_top -“最高估计成本”监视元素』

第 838 页的『last_wlm_reset -“最后一次重置时间”监视元素』

第 1122 页的『rows_returned_top -“最高实际返回行数”监视元素』

第 1165 页的『statistics_timestamp -“统计信息时间戳记”监视元素』

第 1222 页的『temp_tablespace_top -“最大临时表空间”监视元素』

第 1339 页的『work_action_set_id -“工作操作集标识”监视元素』

第 1339 页的『work_action_set_name -“工作操作集名称”监视元素』

第 1340 页的『work_class_id -“工作类标识”监视元素』

第 1340 页的『work_class_name -“工作类名”监视元素』

第 905 页的『mon_interval_id -“监视时间间隔标识”监视元素』

第 896 页的『member -“数据库成员”监视元素』

event_wlstats 逻辑数据组

第 590 页的『act_cpu_time_top -“最长活动 CPU 时间”监视元素』

第 593 页的『act_rows_read_top -“最大活动读取行数”监视元素』

第 594 页的『act_throughput -“活动吞吐量”监视元素』

第 673 页的『concurrent_wlo_act_top -“最大并行 WLO 活动数”监视元素』

第 674 页的『concurrent_wlo_top -“最大并行工作负载项数”监视元素』

第 687 页的『coord_act_aborted_total -“异常终止的协调程序活动总数”监视元素』

第 688 页的『coord_act_completed_total -“完成的协调程序活动总数”监视元素』

第 689 页的『coord_act_est_cost_avg -“平均协调程序活动估计成本”监视元素』

第 689 页的『coord_act_exec_time_avg -“平均协调程序活动执行时间”监视元素』

第 690 页的『coord_act_interarrival_time_avg -“平均协调程序活动到达时间”监视元素』

第 691 页的『coord_act_lifetime_avg -“平均协调程序活动生存期”监视元素』

第 692 页的『coord_act_lifetime_top -“协调程序活动生存期顶部”监视元素』

第 692 页的『coord_act_queue_time_avg -“平均协调程序活动队列时间”监视元素』

第 693 页的『coord_act_rejected_total -“被拒绝的协调程序活动总数”监视元素』

第 697 页的『cost_estimate_top -“最高估计成本”监视元素』

第 704 页的『cpu_utilization -“CPU 利用率”监视元素』

details_xml（此 XML 文档是一个 system_metrics 类型的度量值文档，如 XML 模式文档 `sqllib/misc/DB2MonCommon.xsd` 中所述。）

第 838 页的『last_wlm_reset -“最后一次重置时间”监视元素』

第 867 页的『lock_wait_time_top -“最长锁定等待时间”监视元素』

第 1122 页的『rows_returned_top -“最高实际返回行数”监视元素』

第 1165 页的『statistics_timestamp -“统计信息时间戳记”监视元素』

第 1222 页的『temp_tablespace_top -“最大临时表空间”监视元素』

第 1249 页的『total_cpu_time -“CPU 时间总计”监视元素』

第 1251 页的『total_disp_run_queue_time -“分派器运行队列时间总计”监视元素』

第 1315 页的『uow_completed_total -“完成的工作单元总数”监视元素』

第 1317 页的『uow_lifetime_avg -“工作单元平均生存期”监视元素』

第 1321 页的『uow_throughput -“工作单元吞吐量”监视元素』

第 1321 页的『uow_total_time_top -“最长 UOW 时间总计”监视元素』

第 1338 页的『wlo_completed_total -“完成的工作负载项总数”监视元素』

第 1341 页的『workload_id -“工作负载标识”监视元素』

第 1342 页的『workload_name -“工作负载名称”监视元素』

第 612 页的『app_act_aborted_total -“失败的外部协调程序活动总数”监视元素』

第 613 页的『app_act_completed_total -“成功的外部协调程序活动总数”监视元素』

第 614 页的『app_act_rejected_total -“拒绝的外部协调程序活动总数”监视元素』

第 867 页的『lock_wait_time_global_top -“最长全局锁定等待时间”监视元素』

第 905 页的『mon_interval_id -“监视时间间隔标识”监视元素』

第 896 页的『member -“数据库成员”监视元素』

event_xact 逻辑数据组

第 601 页的『agent_id -“应用程序句柄（代理程序标识）”监视元素』

第 616 页的『appl_id -“应用程序标识”监视元素』

第 845 页的『lock_escals -“锁定升级次数”监视元素』

第 863 页的『lock_wait_time -“等待锁定时间”监视元素』

第 871 页的『locks_held_top -“挂起的最大锁定数”监视元素』

第 949 页的『partial_record -“部分记录”监视元素』

第 1084 页的『prev_uow_stop_time -“上一个工作单元完成时间戳记”』

第 1118 页的『rows_read -“读取行数”监视元素』

第 1123 页的『rows_written -“写入的行数”』

第 1130 页的『sequence_no -“序号”监视元素』

第 1189 页的『system_cpu_time -“系统 CPU 时间”监视元素』

第 1303 页的『tpmon_acc_str -“TP 监视器客户机记帐字符串”监视元素』

第 1304 页的『tpmon_client_app -“TP 监视器客户机应用程序名称”监视元素』

第 1304 页的『tpmon_client_userid -“TP 监视器客户机用户标识”监视元素』

第 1305 页的『tpmon_client_wkstn -“TP 监视器客户机工作站名称”监视元素』

第 1318 页的『uow_log_space_used -“使用的工作单元日志空间”监视元素』

第 1319 页的『uow_start_time -“工作单元开始时间戳记”监视元素』

第 1320 页的『uow_status -“工作单元状态”』

第 1185 页的『stop_time -“事件停止时间”』

第 1326 页的『user_cpu_time -“用户 CPU 时间”监视元素』

第 1344 页的『x_lock_escals -“互斥锁定升级数”监视元素』

第 763 页的『evmon_flushes -“事件监视器清空数”』

evmonstart 逻辑数据组

第 754 页的『event_id -“事件标识”监视元素』

第 755 页的『event_timestamp -“事件时间戳记”监视元素』

第 896 页的『member -“数据库成员”监视元素』

第 756 页的『event_type -“事件类型”监视元素』

第 717 页的『db2start_time -“启动数据库管理器时间戳记”』

第 717 页的『db_conn_time -“数据库激活时间戳记”监视元素』

lock 逻辑数据组

第 951 页的『partition_key -“分区键”监视元素』

第 748 页的『dl_conns -“死锁中涉及的连接数”监视元素』

第 754 页的『event_id -“事件标识”监视元素』

第 755 页的『event_timestamp -“事件时间戳记”监视元素』

第 756 页的『event_type -“事件类型”监视元素』

第 951 页的『partition_number -“分区号”』

第 1114 页的『rolled_back_participant_no -“回滚的应用程序参与者”监视元素』

第 728 页的『deadlock_type -“死锁类型”监视元素』

lock_activity_values 逻辑数据组

第 951 页的『partition_key -“分区键”监视元素』

第 597 页的『activity_id -“活动标识”监视元素』

第 754 页的『event_id -“事件标识”监视元素』

第 755 页的『event_timestamp -“事件时间戳记”监视元素』

第 756 页的『event_type -“事件类型”监视元素』

第 950 页的『participant_no -“死锁参与者”』

第 951 页的『partition_number -“分区号”』

第 1182 页的『stmt_value_index -“值索引”』

第 1182 页的『stmt_value_isnull -“包含空值”监视元素』

第 1183 页的『stmt_value_isreopt -“用于语句重新优化的变量”监视元素』

第 1184 页的『stmt_value_type -“值类型”监视元素』

第 1316 页的『uow_id -“工作单元标识”监视元素』

第 1181 页的『stmt_value_data -“值数据”』

第 754 页的『event_id -“事件标识”监视元素』

lock_participant_activities 逻辑数据组

第 951 页的『partition_key -“分区键”监视元素』

第 597 页的『activity_id -“活动标识”监视元素』

第 598 页的『activity_type -“活动类型”监视元素』

第 684 页的『consistency_token -“程序包一致性标记”监视元素』

第 750 页的『effective_isolation -“有效隔离级别”监视元素』

第 750 页的『effective_query_degree -“有效查询并行度”监视元素』

第 754 页的『event_id -“事件标识”监视元素』

第 755 页的『event_timestamp -“事件时间戳记”监视元素』

第 756 页的『event_type -“事件类型”监视元素』

第 816 页的『incremental_bind -“增量绑定”监视元素』

第 941 页的『package_name -“程序包名”监视元素』

第 942 页的『package_schema -“程序包模式”监视元素』

第 943 页的『package_version_id -“程序包版本”监视元素』

第 950 页的『participant_no -“死锁参与者”』

第 951 页的『partition_number -“分区号”』

第 1091 页的『query_actual_degree -“实际运行时分区内并行度”监视元素』

第 1104 页的『reopt -“REOPT 绑定选项”监视元素』

第 1128 页的『section_number -“节号”监视元素』

第 1169 页的『stmt_first_use_time -“第一次使用语句时的时间戳记”监视元素』

第 1170 页的『stmt_invocation_id -“语句调用标识”监视元素』

第 1171 页的『stmt_last_use_time -“上一次使用语句时的时间戳记”监视元素』

第 1172 页的『stmt_lock_timeout -“语句锁定超时”监视元素』

第 1172 页的『stmt_nest_level -“语句嵌套级别”监视元素』

第 1174 页的『stmt_pkgcache_id -“语句程序包高速缓存标识”监视元素』

第 1175 页的『stmt_query_id -“语句查询标识”监视元素』

第 1176 页的『stmt_source_id -“语句源标识”』

第 1179 页的『stmt_type -“语句类型”监视元素』

第 1316 页的『uow_id -“工作单元标识”监视元素』

第 1178 页的『stmt_text -“SQL 语句文本”监视元素』

第 1181 页的『stmt_unicode -“语句 Unicode 标志”监视元素』

第 1173 页的『stmt_operation/operation -“语句操作”监视元素』

lock_participants 逻辑数据组

第 951 页的『partition_key -“分区键”监视元素』

第 603 页的『agent_status -“DCS 应用程序代理程序数”』

第 601 页的『agent_id -“应用程序句柄（代理程序标识）”监视元素』

第 616 页的『appl_id -“应用程序标识”监视元素』

第 620 页的『appl_name -“应用程序名称”监视元素』

第 625 页的『application_handle -“应用程序句柄”监视元素』

第 635 页的『auth_id -“授权标识”』

第 657 页的『client_acctng -“客户机记帐字符串”监视元素』

第 658 页的『client_applname -“客户机应用程序名称”监视元素』

第 664 页的『client_userid -“客户机用户标识”监视元素』

第 665 页的『client_wrkstnname -“客户机工作站名称”监视元素』

第 692 页的『coord_agent_tid -“协调代理程序引擎可分派单元标识”监视元素』

第 711 页的『current_request -“当前操作请求”监视元素』

第 754 页的『event_id -“事件标识”监视元素』

第 755 页的『event_timestamp -“事件时间戳记”监视元素』

第 756 页的『event_type -“事件类型”监视元素』

第 842 页的『lock_attributes -“锁定属性”监视元素』

第 843 页的『lock_count -“锁定计数”监视元素』

第 844 页的『lock_current_mode -“转换前的原始锁定方式”监视元素』

第 845 页的『lock_escalation -“锁定升级”监视元素』

第 851 页的『lock_hold_count -“锁定挂起计数”监视元素』

第 852 页的『lock_mode -“锁定方式”监视元素』

第 853 页的『lock_mode_requested -“请求的锁定方式”监视元素』

第 854 页的『lock_name -“锁定名称”监视元素』

第 856 页的『lock_object_type -“等待的锁定对象类型”监视元素』

LOCK_OBJECT_TYPE_ID

第 858 页的『lock_release_flags -“锁定释放标志”监视元素』

LOCK_RRIID

第 858 页的『lock_status -“锁定状态”监视元素』

第 859 页的『lock_timeout_val -“锁定超时值”监视元素』

第 863 页的『lock_wait_end_time -“锁定等待结束时间戳记”监视元素』

第 863 页的『lock_wait_start_time -“锁定等待开始时间戳记”监视元素』

第 867 页的『lock_wait_val -“锁定等待值”监视元素』

第 896 页的『member -“数据库成员”监视元素』

第 928 页的『object_requested -“所请求对象”监视元素』

第 950 页的『participant_no -“死锁参与者”』

第 950 页的『participant_no_holding_lk -“对应用程序所需对象挂起锁定的参与者”』

第 950 页的『participant_type -“参与者类型”监视元素』

第 953 页的『past_activities_wrapped -“合并过去活动列表”监视元素』

第 1134 页的『service_class_id -“服务类标识”监视元素』

第 1135 页的『service_subclass_name -“服务子类名”监视元素』

第 1190 页的『table_file_id -“表文件标识”监视元素』

第 1191 页的『table_name -“表名”监视元素』

第 1193 页的『table_schema -“表模式名”监视元素』

第 1228 页的『thresholdid -“阈值标识”监视元素』

第 1226 页的『threshold_name -“阈值名称”监视元素』

第 1341 页的『workload_id -“工作负载标识”监视元素』

第 1342 页的『workload_name -“工作负载名称”监视元素』

第 604 页的『agent_tid -“代理程序线程标识”监视元素』

第 615 页的『appl_action -“应用程序操作”监视元素』

第 727 页的『deadlock_member -“死锁成员”监视元素』

第 1094 页的『queue_start_time -“队列开始时间戳记”监视元素』

第 1095 页的『queued_agents -“已排队阈值代理程序数”监视元素』

第 1136 页的『service_superclass_name -“服务超类名”监视元素』

第 1200 页的『tablespace_name -“表空间名称”监视元素』

第 1346 页的『xid -“事务标识”』

第 1328 页的『utility_invocation_id -“实用程序调用标识”』

pkgcache 逻辑数据组

第 951 页的『partition_key -“分区键”监视元素』

第 669 页的『comp_env_desc -“编译环境”监视元素』

第 750 页的『effective_isolation -“有效隔离级别”监视元素』

第 754 页的『event_id -“事件标识”监视元素』

第 755 页的『event_timestamp -“事件时间戳记”监视元素』

第 762 页的『executable_id -“可执行文件标识”监视元素』

第 820 页的『insert_timestamp -“插入时间戳记”监视元素』

第 835 页的『last_metrics_update -“最近一次更新度量的时间戳记”监视元素』

第 896 页的『member -“数据库成员”监视元素』

第 910 页的『num_coord_exec -“协调代理程序执行的次数”监视元素』

第 910 页的『num_coord_exec_with_metrics -“协调代理程序执行的次数以及度量”监视元素』

第 912 页的『num_exec_with_metrics -“在收集度量值情况下的执行次数”监视元素』

第 911 页的『num_executions -“语句执行次数”监视元素』

第 941 页的『package_name -“程序包名”监视元素』

第 942 页的『package_schema -“程序包模式”监视元素』

第 943 页的『package_version_id -“程序包版本”监视元素』

第 951 页的『partition_number -“分区号”』

第 1083 页的『prep_time -“编译时间”监视元素』

第 1092 页的『query_cost_estimate -“查询估算成本”监视元素』

第 1093 页的『query_data_tag_list -“估算的查询数据标记列表”监视元素』

第 1115 页的『routine_id -“例程标识”监视元素』

第 1128 页的『section_env -“节环境”监视元素』

第 1128 页的『section_number -“节号”监视元素』

第 1129 页的『section_type -“节类型指示器”监视元素』

第 1174 页的『stmt_pkgcache_id -“语句程序包高速缓存标识”监视元素』

第 1180 页的『stmt_type_id -“语句类型标识”监视元素』

第 1294 页的『total_stats_fabrication_time -“统计信息生成时间总计”监视元素』

第 1295 页的『total_stats_fabrications -“统计信息生成总计”监视元素』

第 1299 页的『total_sync_runstats -“同步 RUNSTATS 活动总数”监视元素』

第 1296 页的『total_sync_runstats_time -“同步 RUNSTATS 时间总计”监视元素』

第 1178 页的『stmt_text -“SQL 语句文本”监视元素』

第 881 页的『max_coord_stmt_exec_time -“最长协调程序语句执行时间”监视元素』

第 884 页的『max_coord_stmt_exec_timestamp -“最大协调程序语句执行时间戳记”监视元素』

pkgcache_metrics 逻辑数据组

第 951 页的『partition_key -“分区键”监视元素』

第 754 页的『event_id -“事件标识”监视元素』

第 755 页的『event_timestamp -“事件时间戳记”监视元素』

第 951 页的『partition_number -“分区号”』

第 1337 页的『wlm_queue_time_total -“工作负载管理器队列时间总计”监视元素』

第 1336 页的『wlm_queue_assignments_total -“工作负载管理器队列分配总次数”监视元素』

第 783 页的『fcm_tq_recv_wait_time -“FCM 表队列接收等待时间”监视元素』

第 768 页的『fcm_message_recv_wait_time -“接收 FCM 消息等待时间”监视元素』

第 787 页的『fcm_tq_send_wait_time -“FCM 表队列发送等待时间”监视元素』

第 772 页的『fcm_message_send_wait_time -“发送 FCM 消息等待时间”监视元素』

第 863 页的『lock_wait_time -“等待锁定时间”监视元素』

第 868 页的『lock_waits -“等待锁定次数”监视元素』

第 737 页的『direct_read_time -“直接读时间”监视元素』

第 735 页的『direct_read_reqs -“直接读请求数”监视元素』

第 742 页的『direct_write_time -“直接写时间”监视元素』

第 740 页的『direct_write_reqs -“直接写请求数”监视元素』

第 872 页的『log_buffer_wait_time -“日志缓冲区等待时间”监视元素』

第 913 页的『num_log_buffer_full -“日志缓冲区变满而导致代理程序等待的次数”监视元素』

第 874 页的『log_disk_wait_time -“日志磁盘等待时间”监视元素』
第 875 页的『log_disk_waits_total -“日志磁盘等待总次数”监视元素』
第 1058 页的『pool_write_time -“缓冲池物理写时间总计”监视元素』
第 1043 页的『pool_read_time -“缓冲池物理读时间总计”监视元素』
第 629 页的『audit_file_write_wait_time -“审计文件写等待时间”监视元素』
第 631 页的『audit_file_writes_total -“写审计文件总次数”监视元素』
第 632 页的『audit_subsystem_wait_time -“审计子系统等待时间”监视元素』
第 634 页的『audit_subsystem_waits_total -“审计子系统等待总次数”监视元素』
第 732 页的『diaglog_write_wait_time -“诊断日志文件写等待时间”监视元素』
第 734 页的『diaglog_writes_total -“写诊断日志文件总次数”监视元素』
第 779 页的『fcm_send_wait_time -“FCM 发送等待时间”监视元素』
第 775 页的『fcm_recv_wait_time -“FCM 接收等待时间”监视元素』
第 1232 页的『total_act_wait_time -“活动等待时间总计”监视元素』
第 1285 页的『total_section_sort_proc_time -“节排序处理时间总计”监视元素』
第 1288 页的『total_section_sorts -“节排序总次数”监视元素』
第 1287 页的『total_section_sort_time -“节排序时间总计”监视元素』
第 1231 页的『total_act_time -“活动时间总计”监视元素』
第 1118 页的『rows_read -“读取行数”监视元素』
第 1117 页的『rows_modified -“修改的行数”监视元素』
第 982 页的『pool_data_l_reads -“缓冲池数据逻辑读取数”监视元素』
第 1012 页的『pool_index_l_reads -“缓冲池索引逻辑读取数”监视元素』
第 1046 页的『pool_temp_data_l_reads -“缓冲池临时数据逻辑读取数”监视元素』
第 1050 页的『pool_temp_index_l_reads -“缓冲池临时索引逻辑读取数”监视元素』
第 1065 页的『pool_xda_l_reads -“缓冲池 XDA 数据逻辑读取数”监视元素』
第 1053 页的『pool_temp_xda_l_reads -“缓冲池临时 XDA 数据逻辑读取数”监视元素』
第 1249 页的『total_cpu_time -“CPU 时间总计”监视元素』
第 984 页的『pool_data_p_reads -“缓冲池数据物理读取数”监视元素』
第 1048 页的『pool_temp_data_p_reads -“缓冲池临时数据物理读取数”监视元素』
第 1069 页的『pool_xda_p_reads -“缓冲池 XDA 数据物理读取数”监视元素』
第 1055 页的『pool_temp_xda_p_reads -“缓冲池临时 XDA 数据物理读取数”监视元素』
第 1014 页的『pool_index_p_reads -“缓冲池索引物理读取数”监视元素』
第 1052 页的『pool_temp_index_p_reads -“缓冲池临时索引物理读取数”监视元素』
第 986 页的『pool_data_writes -“缓冲池数据写次数”监视元素』
第 1071 页的『pool_xda_writes -“缓冲池 XDA 数据写次数”监视元素』
第 1016 页的『pool_index_writes -“缓冲池索引写次数”监视元素』
第 738 页的『direct_reads -“直接读数据库数目”监视元素』
第 744 页的『direct_writes -“直接写数据库数目”监视元素』
第 1120 页的『rows_returned -“返回的行数”监视元素』
第 728 页的『deadlocks -“检测到的死锁数”监视元素』

第 860 页的『lock_timeouts -“锁定超时次数”监视元素』

第 845 页的『lock_escals -“锁定升级次数”监视元素』

第 780 页的『fcm_sends_total -“FCM 发送总计”监视元素』

第 777 页的『fcm_recvs_total -“FCM 接收总计”监视元素』

第 778 页的『fcm_send_volume -“FCM 发送量”监视元素』

第 774 页的『fcm_recv_volume -“FCM 接收量”监视元素』

第 773 页的『fcm_message_sends_total -“发送 FCM 消息总数”监视元素』

第 769 页的『fcm_message_recvs_total -“接收 FCM 消息总数”监视元素』

第 770 页的『fcm_message_send_volume -“发送 FCM 消息量”监视元素』

第 766 页的『fcm_message_recv_volume -“接收 FCM 消息量”监视元素』

第 788 页的『fcm_tq_sends_total -“FCM 表队列发送总次数”监视元素』

第 784 页的『fcm_tq_recvs_total -“FCM 表队列接收总量”监视元素』

第 786 页的『fcm_tq_send_volume -“FCM 表队列发送量”监视元素』

第 782 页的『fcm_tq_recv_volume -“FCM 表队列接收量”监视元素』

第 1311 页的『tq_tot_send_spills -“溢出表队列缓冲区总数”监视元素』

第 1079 页的『post_threshold_sorts -“超出阈值后的排序次数”监视元素』

第 1073 页的『post_shrthreshold_sorts -“共享阈值后排序数”监视元素』

第 1149 页的『sort_overflows -“排序溢出数”监视元素』

第 628 页的『audit_events_total -“审计事件总数”监视元素』

第 1292 页的『total_sorts -“排序总数”监视元素』

第 1168 页的『stmt_exec_time -“语句执行时间”监视元素』

第 696 页的『coord_stmt_exec_time -“协调代理程序执行语句的时间”监视元素』

第 1274 页的『total_routine_non_sect_proc_time -“非部分处理时间” 监视元素』

第 1274 页的『total_routine_non_sect_time -“非部分例程执行时间” 监视元素』

第 1284 页的『total_section_proc_time -“部分处理时间总计”监视元素』

第 1235 页的『total_app_section_executions -“应用程序执行部分执行的总次数”监视元素』

第 1289 页的『total_section_time -“部分时间总计”监视元素』

第 1276 页的『total_routine_user_code_proc_time -“例程用户代码处理时间总计”监视元素』

第 1278 页的『total_routine_user_code_time -“例程用户代码时间总计”监视元素』

第 1275 页的『total_routine_time -“例程时间总计”监视元素』

第 1223 页的『thresh_violations -“阈值违例次数”监视元素』

第 917 页的『num_lw_thresh_exceeded -“超过锁定等待阈值的次数”监视元素』

第 1272 页的『total_routine_invocations -“例程调用总计”监视元素』

第 865 页的『lock_wait_time_global -“锁定等待时间全局”监视元素』

第 869 页的『lock_waits_global -“锁定等待全局”监视元素』

第 1099 页的『reclaim_wait_time -“回收等待时间”监视元素』

第 1160 页的『spacemappage_reclaim_wait_time -“空间映射页回收等待时间”监视元素』

第 861 页的『lock_timeouts_global -“锁定超时全局”监视元素』

第 850 页的『lock_escals_maxlocks -“maxlocks 锁定升级数”监视元素』

第 849 页的『lock_escals_locklist -“locklist 锁定升级数”监视元素』

第 848 页的『lock_escals_global -“全局锁定升级数”监视元素』

第 652 页的『cf_wait_time -“集群高速缓存设施等待时间”监视元素』

第 651 页的『cf_waits -“集群高速缓存设施等待次数”监视元素』

第 978 页的『pool_data_gbp_l_reads -“组缓冲池数据逻辑读取数”监视元素』

第 979 页的『pool_data_gbp_p_reads -“组缓冲池数据物理读取数”监视元素』

第 981 页的『pool_data_lbp_pages_found -“本地缓冲池发现的数据页数”监视元素』

第 977 页的『pool_data_gbp_invalid_pages -“组缓冲池无效数据页数”监视元素』

第 1008 页的『pool_index_gbp_l_reads -“组缓冲池索引逻辑读取数”监视元素』

第 1009 页的『pool_index_gbp_p_reads -“组缓冲池索引物理读取数”监视元素』

第 1010 页的『pool_index_lbp_pages_found -“发现的本地缓冲池索引页数”监视元素』

第 1006 页的『pool_index_gbp_invalid_pages -“组缓冲池无效索引页数”监视元素』

第 1062 页的『pool_xda_gbp_l_reads -“组缓冲池 XDA 数据逻辑读取请求数”监视元素』

第 1064 页的『pool_xda_gbp_p_reads -“组缓冲池 XDA 数据物理读取请求数”监视元素』

第 1067 页的『pool_xda_lbp_pages_found -“发现的本地缓冲池 XDA 数据页数”监视元素』

第 1061 页的『pool_xda_gbp_invalid_pages -“组缓冲池无效 XDA 数据页数”监视元素』

第 758 页的『evmon_wait_time -“事件监视器等待时间”监视元素』

第 760 页的『evmon_waits_total -“事件监视器总等待次数”监视元素』

第 1253 页的『total_extended_latch_wait_time -“扩展锁存器等待时间总计”监视元素』

第 1254 页的『total_extended_latch_waits -“扩展锁存器等待总计”监视元素』

第 1251 页的『total_disp_run_queue_time -“分派器运行队列时间总计”监视元素』

第 1021 页的『pool_queued_async_data_reqs -“数据预取请求数”监视元素』

第 1025 页的『pool_queued_async_index_reqs -“索引预取请求数”监视元素』

第 1041 页的『pool_queued_async_xda_reqs -“XDA 预取请求数”监视元素』

第 1019 页的『pool_queued_async_data_pages -“预取请求的数据页数”监视元素』

第 1023 页的『pool_queued_async_index_pages -“预取请求的索引页数”监视元素』

第 1039 页的『pool_queued_async_xda_pages -“预取请求的 XDA 页数”监视元素』

pkgcache_stmt_args 逻辑数据组

第 754 页的『event_id -“事件标识”监视元素』

第 755 页的『event_timestamp -“事件时间戳记”监视元素』

第 1182 页的『stmt_value_index -“值索引”』

第 1183 页的『stmt_value_isreopt -“用于语句重新优化的变量”监视元素』

第 1182 页的『stmt_value_isnull -“包含空值”监视元素』

第 1184 页的『stmt_value_type -“值类型”监视元素』

第 1181 页的『stmt_value_data -“值数据”』

第 896 页的『member -“数据库成员”监视元素』

regvar 逻辑数据组

第 754 页的『event_id -“事件标识”监视元素』

第 755 页的『event_timestamp -“事件时间戳记”监视元素』

第 896 页的『member -“数据库成员”监视元素』

第 756 页的『event_type -“事件类型”监视元素』

第 1101 页的『regvar_name -“注册表变量名称”』

第 1101 页的『regvar_value -“注册表变量值”』

第 1101 页的『regvar_old_value -“注册表变量旧值”』

第 1100 页的『regvar_level -“注册表变量级别”』

第 1100 页的『regvar_collection_type -“注册表变量收集类型”』

sqlca 逻辑数据组

sqlcabc

sqlcaid

sqlcode

sqlerrd

sqlerrmc

sqlerrml

sqlerrp

sqlstate

sqlwarn

txncompletion 逻辑数据组

第 754 页的『event_id -“事件标识”监视元素』

第 755 页的『event_timestamp -“事件时间戳记”监视元素』

第 896 页的『member -“数据库成员”监视元素』

第 756 页的『event_type -“事件类型”监视元素』

第 794 页的『global_transaction_id -“全局事务标识”监视元素』

第 841 页的『local_transaction_id -“本地事务标识”监视元素』

第 1125 页的『savepoint_id -“保存点标识”』

第 1316 页的『uow_id -“工作单元标识”监视元素』

第 725 页的『ddl_classification -“DDL 分类”』

第 1313 页的『txn_completion_status -“事务完成状态”』

uow 逻辑数据组

第 951 页的『partition_key -“分区键”监视元素』

第 625 页的『application_handle -“应用程序句柄”监视元素』

第 616 页的『appl_id -“应用程序标识”监视元素』

第 620 页的『appl_name -“应用程序名称”监视元素』

第 657 页的『client_acctng -“客户机记帐字符串”监视元素』

第 658 页的『client_applname -“客户机应用程序名称”监视元素』
第 659 页的『client_hostname -“客户机主机名”监视元素』
第 661 页的『client_pid -“客户机进程标识”监视元素』
第 662 页的『client_port_number -“客户机端口号”监视元素』
第 663 页的『client_prdid -“客户机产品和版本标识”监视元素』
第 664 页的『client_userid -“客户机用户标识”监视元素』
第 665 页的『client_wrkstnname -“客户机工作站名称”监视元素』
第 670 页的『completion_status -“完成状态”监视元素』
第 682 页的『conn_time -“数据库连接时间”监视元素』
第 695 页的『coord_member -“协调程序成员”监视元素』
第 754 页的『event_id -“事件标识”监视元素』
第 755 页的『event_timestamp -“事件时间戳记”监视元素』
第 762 页的『executable_list_size -“可执行列表大小”监视元素』
第 794 页的『global_transaction_id -“全局事务标识”监视元素』
第 827 页的『intra_parallel_state -“分区内并行性的当前状态”监视元素』
第 841 页的『local_transaction_id -“本地事务标识”监视元素』
第 896 页的『member -“数据库成员”监视元素』
第 717 页的『db_conn_time -“数据库激活时间戳记”监视元素』
第 905 页的『mon_interval_id -“监视时间间隔标识”监视元素』
第 941 页的『package_list_exceeded -“超过了程序包列表的容量”监视元素』
第 941 页的『package_list_size -“程序包列表大小”监视元素』
service_class_id - 服务类标识
service_subclass_name - 服务子类名
service_superclass_name - 服务超类名
第 1137 页的『session_auth_id -“会话授权标识”监视元素』
start_time - 事件启动时间
stop_time - 事件停止时间
第 1188 页的『system_auth_id -“系统授权标识”监视元素』
UOW_CLIENT_PLATFORM
UOW_CLIENT_PROTOCOL
uow_id - 工作单元标识
第 1318 页的『uow_log_space_used -“使用的工作单元日志空间”监视元素』
workload_id - 工作负载标识
workload_name - 工作负载名称
workload_occurrence_id - 工作负载项标识
第 662 页的『client_platform -“客户机操作平台”监视元素』
第 664 页的『client_protocol -“客户机通信协议”监视元素』
第 763 页的『executable_list_truncated -“截断可执行列表”监视元素』
第 683 页的『connection_start_time -“连接开始时间”监视元素』

uow_executable_list 逻辑数据组

第 951 页的『partition_key -“分区键”监视元素』

第 616 页的『appl_id -“应用程序标识”监视元素』

第 762 页的『executable_id -“可执行文件标识”监视元素』

第 863 页的『lock_wait_time -“等待锁定时间”监视元素』

第 868 页的『lock_waits -“等待锁定次数”监视元素』

第 911 页的『num_executions -“语句执行次数”监视元素』

第 951 页的『partition_number -“分区号”』

第 1073 页的『post_shrthreshold_sorts -“共享阈值后排序数”监视元素』

第 1079 页的『post_threshold_sorts -“超出阈值后的排序次数”监视元素』

第 1118 页的『rows_read -“读取行数”监视元素』

第 1149 页的『sort_overflows -“排序溢出数”监视元素』

第 1231 页的『total_act_time -“活动时间总计”监视元素』

第 1232 页的『total_act_wait_time -“活动等待时间总计”监视元素』

第 1249 页的『total_cpu_time -“CPU 时间总计”监视元素』

第 1292 页的『total_sorts -“排序总数”监视元素』

uow_id - 工作单元标识

uow_metrics 逻辑数据组

第 951 页的『partition_key -“分区键”监视元素』

第 616 页的『appl_id -“应用程序标识”监视元素』

第 951 页的『partition_number -“分区号”』

第 1316 页的『uow_id -“工作单元标识”监视元素』

第 1337 页的『wlm_queue_time_total -“工作负载管理器队列时间总计”监视元素』

第 1336 页的『wlm_queue_assignments_total -“工作负载管理器队列分配总次数”监视元素』

第 783 页的『fcm_tq_recv_wait_time -“FCM 表队列接收等待时间”监视元素』

第 768 页的『fcm_message_recv_wait_time -“接收 FCM 消息等待时间”监视元素』

第 787 页的『fcm_tq_send_wait_time -“FCM 表队列发送等待时间”监视元素』

第 772 页的『fcm_message_send_wait_time -“发送 FCM 消息等待时间”监视元素』

第 605 页的『agent_wait_time -“代理程序等待时间”监视元素』

第 606 页的『agent_waits_total -“等待代理程序总次数”监视元素』

第 863 页的『lock_wait_time -“等待锁定时间”监视元素』

第 868 页的『lock_waits -“等待锁定次数”监视元素』

第 737 页的『direct_read_time -“直接读时间”监视元素』

第 735 页的『direct_read_reqs -“直接读请求数”监视元素』

第 742 页的『direct_write_time -“直接写时间”监视元素』

第 740 页的『direct_write_reqs -“直接写请求数”监视元素』

第 872 页的『log_buffer_wait_time -“日志缓冲区等待时间”监视元素』

第 913 页的『num_log_buffer_full -“日志缓冲区变满而导致代理程序等待的次数”监视元素』

第 874 页的『log_disk_wait_time -“日志磁盘等待时间”监视元素』
第 875 页的『log_disk_waits_total -“日志磁盘等待总次数”监视元素』
第 1218 页的『tcpip_recv_wait_time -“TCP/IP 接收等待时间”监视元素』
第 1219 页的『tcpip_recvs_total -“TCP/IP 接收总次数”监视元素』
第 660 页的『client_idle_wait_time -“客户机空闲等待时间”监视元素』
第 829 页的『ipc_recv_wait_time -“进程间通信接收等待时间”监视元素』
第 830 页的『ipc_recvs_total -“进程间通信接收总次数”监视元素』
第 831 页的『ipc_send_wait_time -“进程间通信发送等待时间”监视元素』
第 832 页的『ipc_sends_total -“进程间通信发送总次数”监视元素』
第 1221 页的『tcpip_send_wait_time -“TCP/IP 发送等待时间”监视元素』
第 1222 页的『tcpip_sends_total -“TCP/IP 发送总次数”监视元素』
第 1058 页的『pool_write_time -“缓冲池物理写时间总计”监视元素』
第 1043 页的『pool_read_time -“缓冲池物理读时间总计”监视元素』
第 629 页的『audit_file_write_wait_time -“审计文件写等待时间”监视元素』
第 631 页的『audit_file_writes_total -“写审计文件总次数”监视元素』
第 632 页的『audit_subsystem_wait_time -“审计子系统等待时间”监视元素』
第 634 页的『audit_subsystem_waits_total -“审计子系统等待总次数”监视元素』
第 732 页的『diaglog_write_wait_time -“诊断日志文件写等待时间”监视元素』
第 734 页的『diaglog_writes_total -“写诊断日志文件总次数”监视元素』
第 779 页的『fcm_send_wait_time -“FCM 发送等待时间”监视元素』
第 775 页的『fcm_recv_wait_time -“FCM 接收等待时间”监视元素』
第 1302 页的『total_wait_time -“等待时间总计”监视元素』
第 1124 页的『rqsts_completed_total -“完成请求总数”监视元素』
第 1280 页的『total_rqst_time -“请求时间总计”监视元素』
第 615 页的『app_rqsts_completed_total -“完成应用程序请求总数”监视元素』
第 1235 页的『total_app_rqst_time -“应用程序请求时间总计”监视元素』
第 1285 页的『total_section_sort_proc_time -“节排序处理时间总计”监视元素』
第 1288 页的『total_section_sorts -“节排序总次数”监视元素』
第 1287 页的『total_section_sort_time -“节排序时间总计”监视元素』
第 1118 页的『rows_read -“读取行数”监视元素』
第 1117 页的『rows_modified -“修改的行数”监视元素』
第 982 页的『pool_data_l_reads -“缓冲池数据逻辑读取数”监视元素』
第 1012 页的『pool_index_l_reads -“缓冲池索引逻辑读取数”监视元素』
第 1046 页的『pool_temp_data_l_reads -“缓冲池临时数据逻辑读取数”监视元素』
第 1050 页的『pool_temp_index_l_reads -“缓冲池临时索引逻辑读取数”监视元素』
第 1065 页的『pool_xda_l_reads -“缓冲池 XDA 数据逻辑读取数”监视元素』
第 1053 页的『pool_temp_xda_l_reads -“缓冲池临时 XDA 数据逻辑读取数”监视元素』
第 1249 页的『total_cpu_time -“CPU 时间总计”监视元素』
第 589 页的『act_completed_total -“完成活动总数”监视元素』
第 984 页的『pool_data_p_reads -“缓冲池数据物理读取数”监视元素』

第 1048 页的『pool_temp_data_p_reads -“缓冲池临时数据物理读取数”监视元素』
第 1069 页的『pool_xda_p_reads -“缓冲池 XDA 数据物理读取数”监视元素』
第 1055 页的『pool_temp_xda_p_reads -“缓冲池临时 XDA 数据物理读取数”监视元素』
第 1014 页的『pool_index_p_reads -“缓冲池索引物理读取数”监视元素』
第 1052 页的『pool_temp_index_p_reads -“缓冲池临时索引物理读取数”监视元素』
第 986 页的『pool_data_writes -“缓冲池数据写次数”监视元素』
第 1071 页的『pool_xda_writes -“缓冲池 XDA 数据写次数”监视元素』
第 1016 页的『pool_index_writes -“缓冲池索引写次数”监视元素』
第 738 页的『direct_reads -“直接读数据库数目”监视元素』
第 744 页的『direct_writes -“直接写数据库数目”监视元素』
第 1120 页的『rows_returned -“返回的行数”监视元素』
第 728 页的『deadlocks -“检测到的死锁数”监视元素』
第 860 页的『lock_timeouts -“锁定超时次数”监视元素』
第 845 页的『lock_escals -“锁定升级次数”监视元素』
第 780 页的『fcm_sends_total -“FCM 发送总计”监视元素』
第 777 页的『fcm_recvs_total -“FCM 接收总计”监视元素』
第 778 页的『fcm_send_volume -“FCM 发送量”监视元素』
第 774 页的『fcm_recv_volume -“FCM 接收量”监视元素』
第 773 页的『fcm_message_sends_total -“发送 FCM 消息总数”监视元素』
第 769 页的『fcm_message_recvs_total -“接收 FCM 消息总数”监视元素』
第 770 页的『fcm_message_send_volume -“发送 FCM 消息量”监视元素』
第 766 页的『fcm_message_recv_volume -“接收 FCM 消息量”监视元素』
第 788 页的『fcm_tq_sends_total -“FCM 表队列发送总次数”监视元素』
第 784 页的『fcm_tq_recvs_total -“FCM 表队列接收总量”监视元素』
第 786 页的『fcm_tq_send_volume -“FCM 表队列发送量”监视元素』
第 782 页的『fcm_tq_recv_volume -“FCM 表队列接收量”监视元素』
第 1311 页的『tq_tot_send_spills -“溢出表队列缓冲区总数”监视元素』
第 1220 页的『tcpip_send_volume -“TCP/IP 发送量”监视元素』
第 1217 页的『tcpip_recv_volume -“TCP/IP 接收量”监视元素』
第 831 页的『ipc_send_volume -“进程间通信发送量”监视元素』
第 828 页的『ipc_recv_volume -“进程间通信接收量”监视元素』
第 1079 页的『post_threshold_sorts -“超出阈值后的排序次数”监视元素』
第 1073 页的『post_shrthreshold_sorts -“共享阈值后排序数”监视元素』
第 1149 页的『sort_overflows -“排序溢出数”监视元素』
第 628 页的『audit_events_total -“审计事件总数”监视元素』
第 591 页的『act_rejected_total -“被拒绝活动总数”监视元素』
第 588 页的『act_aborted_total -“异常终止活动总数”监视元素』
第 1292 页的『total_sorts -“排序总数”监视元素』
第 1275 页的『total_routine_time -“例程时间总计”监视元素』

第 1241 页的『total_compile_proc_time -“编译处理时间总计”监视元素』

第 1240 页的『total_compilations -“编译次数总计”监视元素』

第 1242 页的『total_compile_time -“编译时间总计”监视元素』

第 1258 页的『total_implicit_compile_proc_time -“隐式编译处理时间总计”监视元素』

第 1257 页的『total_implicit_compilations -“隐式编译总数”监视元素』

第 1259 页的『total_implicit_compile_time -“隐式编译时间总计”监视元素』

第 1282 页的『total_runstats_proc_time -“运行时统计信息处理时间总计”监视元素』

第 1281 页的『total_runstats -“运行时统计信息总计”监视元素』

第 1283 页的『total_runstats_time -“运行时统计信息时间总计”监视元素』

第 1268 页的『total_reorg_proc_time -“重组处理时间总计”监视元素』

第 1270 页的『total_reorgs -“重组操作总数”监视元素』

第 1269 页的『total_reorg_time -“重组时间总计”监视元素』

第 1260 页的『total_load_proc_time -“装入处理时间总计”监视元素』

第 1262 页的『total_loads -“装入操作总数”监视元素』

第 1261 页的『total_load_time -“装入时间总计”监视元素』

第 1284 页的『total_section_proc_time -“部分处理时间总计”监视元素』

第 1235 页的『total_app_section_executions -“应用程序执行部分执行的总次数”监视元素』

第 1289 页的『total_section_time -“部分时间总计”监视元素』

第 1238 页的『total_commit_proc_time -“落实处理时间总计”监视元素』

第 1233 页的『total_app_commits -“应用程序落实次数总计”监视元素』

第 1239 页的『total_commit_time -“落实时间总计”监视元素』

第 1270 页的『total_rollback_proc_time -“回滚处理时间总计”监视元素』

第 1234 页的『total_app_rollbacks -“应用程序回滚次数总计”监视元素』

第 1271 页的『total_rollback_time -“回滚时间总计”监视元素』

第 1276 页的『total_routine_user_code_proc_time -“例程用户代码处理时间总计”监视元素』

第 1278 页的『total_routine_user_code_time -“例程用户代码时间总计”监视元素』

第 1223 页的『thresh_violations -“阈值违例次数”监视元素』

第 917 页的『num_lw_thresh_exceeded -“超过锁定等待阈值的次数”监视元素』

第 1272 页的『total_routine_invocations -“例程调用总计”监视元素』

第 821 页的『int_commits -“内部落实数”监视元素』

第 823 页的『int_rollbacks -“内部回滚数”监视元素』

第 646 页的『cat_cache_inserts -“目录高速缓存插入数”监视元素』

第 647 页的『cat_cache_lookups -“目录高速缓存查询数”监视元素』

第 955 页的『pkg_cache_inserts -“程序包高速缓存插入数”监视元素』

第 956 页的『pkg_cache_lookups -“程序包高速缓存查询数”监视元素』

第 593 页的『act_rqsts_total -“活动请求总数”监视元素』

第 1232 页的『total_act_wait_time -“活动等待时间总计”监视元素』

第 1231 页的『total_act_time -“活动时间总计”监视元素』

第 865 页的『lock_wait_time_global -“锁定等待时间全局”监视元素』

第 869 页的『lock_waits_global -“锁定等待全局”监视元素』

第 1099 页的『reclaim_wait_time -“回收等待时间”监视元素』

第 1160 页的『spacemappage_reclaim_wait_time -“空间映射页回收等待时间”监视元素』

第 861 页的『lock_timeouts_global -“锁定超时全局”监视元素』

第 850 页的『lock_escals_maxlocks -“maxlocks 锁定升级数”监视元素』

第 849 页的『lock_escals_locklist -“locklist 锁定升级数”监视元素』

第 848 页的『lock_escals_global -“全局锁定升级数”监视元素』

第 652 页的『cf_wait_time -“集群高速缓存设施等待时间”监视元素』

第 651 页的『cf_waits -“集群高速缓存设施等待次数”监视元素』

第 978 页的『pool_data_gbp_l_reads -“组缓冲池数据逻辑读取数”监视元素』

第 979 页的『pool_data_gbp_p_reads -“组缓冲池数据物理读取数”监视元素』

第 981 页的『pool_data_lbp_pages_found -“本地缓冲池发现的数据页数”监视元素』

第 977 页的『pool_data_gbp_invalid_pages -“组缓冲池无效数据页数”监视元素』

第 1008 页的『pool_index_gbp_l_reads -“组缓冲池索引逻辑读取数”监视元素』

第 1009 页的『pool_index_gbp_p_reads -“组缓冲池索引物理读取数”监视元素』

第 1010 页的『pool_index_lbp_pages_found -“发现的本地缓冲池索引页数”监视元素』

第 1006 页的『pool_index_gbp_invalid_pages -“组缓冲池无效索引页数”监视元素』

第 1062 页的『pool_xda_gbp_l_reads -“组缓冲池 XDA 数据逻辑读取请求数”监视元素』

第 1064 页的『pool_xda_gbp_p_reads -“组缓冲池 XDA 数据物理读取请求数”监视元素』

第 1067 页的『pool_xda_lbp_pages_found -“发现的本地缓冲池 XDA 数据页数”监视元素』

第 1061 页的『pool_xda_gbp_invalid_pages -“组缓冲池无效 XDA 数据页数”监视元素』

第 758 页的『evmon_wait_time -“事件监视器等待时间”监视元素』

第 760 页的『evmon_waits_total -“事件监视器总等待次数”监视元素』

第 1253 页的『total_extended_latch_wait_time -“扩展锁存器等待时间总计”监视元素』

第 1254 页的『total_extended_latch_waits -“扩展锁存器等待总计”监视元素』

第 1293 页的『total_stats_fabrication_proc_time -“统计信息生成处理时间总计”监视元素』

第 1295 页的『total_stats_fabrications -“统计信息生成总计”监视元素』

第 1294 页的『total_stats_fabrication_time -“统计信息生成时间总计”监视元素』

第 1298 页的『total_sync_runstats_proc_time -“同步 RUNSTATS 处理时间总计”监视元素』

第 1299 页的『total_sync_runstats -“同步 RUNSTATS 活动总数”监视元素』

第 1296 页的『total_sync_runstats_time -“同步 RUNSTATS 时间总计”监视元素』

第 1251 页的『total_disp_run_queue_time -“分派器运行队列时间总计”监视元素』

第 1021 页的『pool_queued_async_data_reqs -“数据预取请求数”监视元素』

第 1025 页的『pool_queued_async_index_reqs -“索引预取请求数”监视元素』

第 1041 页的『pool_queued_async_xda_reqs -“XDA 预取请求数”监视元素』

第 1019 页的『pool_queued_async_data_pages -“预取请求的数据页数”监视元素』

第 1023 页的『pool_queued_async_index_pages -“预取请求的索引页数”监视元素』

第 1039 页的『pool_queued_async_xda_pages -“预取请求的 XDA 页数”监视元素』

第 613 页的『app_act_completed_total -“成功的外部协调程序活动总数”监视元素』

第 612 页的『app_act_aborted_total -“失败的外部协调程序活动总数”监视元素』

第 614 页的『app_act_rejected_total -“拒绝的外部协调程序活动总数”监视元素』

uow_package_list 逻辑数据组

第 951 页的『partition_key -“分区键”监视元素』

第 616 页的『appl_id -“应用程序标识”监视元素』

第 827 页的『invocation_id -“调用标识”监视元素』

第 906 页的『nesting_level -“嵌套级别”监视元素』

第 941 页的『package_elapsed_time -“程序包耗用时间”监视元素』

第 940 页的『package_id -“程序包标识”监视元素』

第 1115 页的『routine_id -“例程标识”监视元素』

第 1316 页的『uow_id -“工作单元标识”监视元素』

第 896 页的『member -“数据库成员”监视元素』

utillocation 逻辑数据组

第 754 页的『event_id -“事件标识”监视元素』

第 755 页的『event_timestamp -“事件时间戳记”监视元素』

第 896 页的『member -“数据库成员”监视元素』

第 756 页的『event_type -“事件类型”监视元素』

第 1328 页的『utility_invocation_id -“实用程序调用标识”』

第 1333 页的『utility_type -“实用程序类型”』

第 732 页的『device_type -“设备类型”』

第 841 页的『location_type -“位置类型”』

第 841 页的『location -“位置”』

utilphase 逻辑数据组

第 754 页的『event_id -“事件标识”监视元素』

第 755 页的『event_timestamp -“事件时间戳记”监视元素』

第 896 页的『member -“数据库成员”监视元素』

第 756 页的『event_type -“事件类型”监视元素』

第 1328 页的『utility_invocation_id -“实用程序调用标识”』

第 1333 页的『utility_type -“实用程序类型”』

第 1330 页的『utility_phase_type -“实用程序阶段类型”』

第 953 页的『phase_start_event_id -“阶段开始事件标识”』

第 953 页的『phase_start_event_timestamp -“阶段开始事件时间戳记”』

第 932 页的『objtype -“对象类型”监视元素』

第 928 页的『object_schema -“对象模式”监视元素』

第 927 页的『object_name -“对象名”监视元素』

第 1330 页的『utility_phase_detail -“实用程序阶段详细信息”』

utilstart 逻辑数据组

第 754 页的『event_id -“事件标识”监视元素』

第 755 页的『event_timestamp -“事件时间戳记”监视元素』

第 896 页的『member -“数据库成员”监视元素』

第 756 页的『event_type -“事件类型”监视元素』

第 1328 页的『utility_invocation_id -“实用程序调用标识”』

第 1333 页的『utility_type -“实用程序类型”』

第 1329 页的『utility_operation_type -“实用程序操作类型”』

第 1328 页的『utility_invoker_type -“实用程序调用者类型”』

第 1331 页的『utility_priority -“实用程序优先级”』

第 1331 页的『utility_start_type -“实用程序启动类型”』

第 932 页的『objtype -“对象类型”监视元素』

第 928 页的『object_schema -“对象模式”监视元素』

第 927 页的『object_name -“对象名”监视元素』

第 919 页的『num_tbsps -“表空间数”监视元素』

第 1216 页的『tbsp_names -“表空间名称”』

第 1327 页的『utility_detail -“实用程序详细信息”』

utilstop 逻辑数据组

第 754 页的『event_id -“事件标识”监视元素』

第 755 页的『event_timestamp -“事件时间戳记”监视元素』

第 896 页的『member -“数据库成员”监视元素』

第 756 页的『event_type -“事件类型”监视元素』

第 1328 页的『utility_invocation_id -“实用程序调用标识”』

第 1333 页的『utility_type -“实用程序类型”』

第 1332 页的『utility_stop_type -“实用程序停止类型”』

第 1164 页的『start_event_id -“开始事件标识”』

第 1164 页的『start_event_timestamp -“开始事件时间戳记”』

第 1154 页的『sqlca -“SQL 通信区（SQLCA）”』

事件类型至逻辑数据组的映射

对于文件和管道事件监视器，事件监视器输出由一个有序的逻辑数据分组序列组成。不管事件监视器类型如何，输出记录总是包含相同的起始逻辑数据组。它们为逻辑数据组设置了框架，这些逻辑数据组的存在与否取决于事件监视器记录的事件类型。

对于文件和管道事件监视器，可能会对任何连接生成事件记录，并且事件记录可能会因此以混合顺序出现在流中。这意味着您可能获得连接 1 的事务事件，之后紧跟连接 2 的连接事件。但是，属于单个连接或单个事件的记录将以逻辑顺序出现。例如，语句

记录（语句结尾）总是在事务记录（UOW 结尾）之前，如果有。同样，死锁事件总是在死锁中涉及的每个连接的死锁连接事件记录之前。**应用程序标识**或**应用程序句柄（agent_id）**可用来将记录与连接相匹配。

通常会对与数据库的每个连接写入连接头事件。对于带有详细信息的死锁事件监视器，仅当发生死锁时才将它们写入。在此情况下，将仅对死锁参与者（而不是与数据库的所有连接）写入连接头事件。

逻辑数据分组将按以下四个不同级别排序：监视器、序言、内容和结尾。下面详细描述每个级别，包括对应的事件类型和逻辑数据组。

监视器

将对所有事件监视器生成监视器级别的信息。它包含事件监视器元数据。

表 144. 事件监视器数据流：监视器部分

事件类型	逻辑数据组	可用信息
监视器级别	event_log_stream_header	标识事件监视器的版本级别和字节顺序。应用程序可使用此头来确定是否能够处理 evmon 输出流。

序言

在激活事件监视器时将生成序言信息。

表 145. 事件监视器数据流：序言部分

事件类型	逻辑数据组	可用信息
日志头	event_log_header	跟踪的特征，如服务器类型和内存布局。
数据库头	event_db_header	数据库名称、路径和激活时间。
事件监视器启动	event_start	启动或重新启动监视器的时间。
连接头	event_connheader	每个当前事件对应一个连接头，包括连接时间和应用程序名称。仅对连接、语句、事务和死锁事件监视器生成事件连接头。带有详细信息的死锁事件监视器仅在发生死锁时生成连接头。

内容

特定于事件监视器的指定事件类型的信息出现在内容部分。

表 146. 事件监视器数据流：内容部分

事件类型	逻辑数据组	可用信息
语句事件	event_stmt	语句级别数据，包括动态语句的文本。语句事件监视器不记录访存。
子节事件	event_subsection	子节级别数据。
事务事件[1]	event_xact	事务级别数据。
连接事件	event_conn	连接级别数据。

表 146. 事件监视器数据流：内容部分 (续)

事件类型	逻辑数据组	可用信息
死锁事件	event_deadlock	死锁级别数据。
死锁的连接事件	event_dlconn	死锁涉及的每个连接对应一个死锁的连接事件，包括涉及的应用程序和处于争用状态的锁定。
带有详细信息的死锁的连接事件	event_detailed_dlconn，lock	死锁涉及的每个连接对应一个带有详细信息的死锁的连接事件，包括涉及的应用程序、处于争用状态的锁定、当前语句信息和应用程序争用挂起的其他锁定。
溢出	event_overflow	丢失的记录的个数 - 在写程序不能与（非分块）事件监视器保持一致时生成。
带有详细信息的死锁的历史纪录 [2]	event_stmt_history	列示死锁涉及的任何工作单元中执行的语句列表。
带有详细信息的死锁的历史记录值 [2]	event_data_value	event_stmt_history 列表中的语句的参数标记。
活动	event_activity	系统上已完成执行的或在完成前捕获的活动列表。
	event_activitystmt	活动类型为语句时，有关活动正在执行的语句的信息。
	event_activityvals	用作每个 SQL 语句活动的输入变量的数据值。这些数据值不包括 LOB 数据、长数据或结构化类型数据。
统计信息	event_scstats	从在系统中每个服务类、工作类或工作负载内执行的活动计算出来的统计信息，以及从阈值队列计算出来的统计信息。
	event_wcstats	
	event_wlstats	
	event_qstats	
	event_histogrambin	
阈值违例	event_thresholdviolations	标识阈值违例及违例时间的信息。

1 建议不要使用此选项。建议不要再使用此选项，将来的发行版可能会将其除去。请使用 CREATE EVENT MONITOR FOR UNIT OF WORK 语句来监视事务事件。

2 建议不要使用此选项。建议不要再使用此选项，将来的发行版可能会将其除去。请使用 CREATE EVENT MONITOR FOR LOCKING 语句来监视与锁定相关的事件，例如锁定超时、锁定等待和死锁。

结尾

结尾信息将在数据库释放时（最后一个应用程序断开连接后）生成:

表 147. 事件监视器数据流：结尾部分

事件类型	逻辑数据组	可用信息
数据库事件	event_db	数据库管理器级别数据。
缓冲池事件	event_bufferpool	缓冲池级别数据。

表 147. 事件监视器数据流：结尾部分 (续)

事件类型	逻辑数据组	可用信息
表空间事件	event_tablespace	表空间级别数据。
表事件	event_table	表级别数据。

受 COLLECT ACTIVITY DATA 设置影响的逻辑数据组

下表显示为不同的 COLLECT ACTIVITY DATA 选项指定所有类型的 WLM 对象时收集的逻辑数据组，这些对象包括“服务子类”、“工作负载”、“工作类”（通过“工作操作”）和“阈值”。

表 148. COLLECT ACTIVITY DATA 设置

COLLECT ACTIVITY DATA 的设置	收集的逻辑数据组
NONE	无
WITHOUT DETAILS	event_activity event_activitymetrics
WITH DETAILS	event_activity event_activitymetrics event_activitystmt
WITH DETAILS AND VALUES	event_activity event_activitymetrics event_activitystmt event_activityvals

快照监视器接口至逻辑数据组的映射

下表列示用来访问快照监视器数据的一些方法。所有快照监视器数据都存储在监视元素中，这些监视元素按逻辑数据组分类。每个 API 请求类型、CLP 命令和 SQL 管理视图仅从所有逻辑数据组的某个子集捕获监视器数据。

此表中列示的每个 API 请求类型、CLP 命令和 SQL 管理视图返回最右列中列示的逻辑数据组中的监视元素。

注:

1. 有一些 API 请求类型和 CLP 命令没有相应的 SQL 管理视图。对于其他 API 请求类型和 CLP 命令，各个 SQL 管理视图捕获关联逻辑数据组的子集。
2. 仅当关联监视开关设置为 ON 时，才返回某些监视元素。请参阅各个监视元素以确定所需的元素是否在开关控制之下。

表 149. 快照监视器接口至逻辑数据组的映射

db2GetSnapshot API 请求类型	CLP 命令	SQL 管理视图	逻辑数据组
SQLMA_APPLINFO_ALL	`list applications [show detail]`	APPLICATIONS	appl_info

表 149. 快照监视器接口至逻辑数据组的映射 (续)

db2GetSnapshot API 请求类型	CLP 命令	SQL 管理视图	逻辑数据组
SQLMA_DBASE_APPLINFO	list applications for database *dbname* [show detail]	APPLICATIONS	appl_info
SQLMA_DCS_APPLINFO_ALL	list dcs applications [show detail]		dcs_appl_info
SQLMA_DB2	get snapshot for dbm	SNAPDBM	db2
		SNAPFCM	fcm
		SNAPFCMPART	fcm_node
		SNAPUTIL	utility_info
		SNAPUTIL_PROGRESS	progress 和 progress_info
		SNAPDBM_MEMORY_POOL	memory_pool
	get dbm monitor switches	SNAPSWITCHES	switch_list
SQLMA_DBASE	get snapshot for database on *dbname*	SNAPDB	dbase
		SNAPDETAILLOG	detail_log
		ADMIN_GET_STORAGE_PATHS	db_storage_group
			rollforward
			db_sto_path_info
		SNAPTBSP	tablespace
		SNAPDB_MEMORY_POOL	memory_pool
SQLMA_DBASE_ALL	get snapshot for all databases	SNAPDB	dbase
		ADMIN_GET_STORAGE_PATHS	db_storage_group
			rollforward
			db_sto_path_info
		SNAPTBSP	tablespace
		SNAPDB_MEMORY_POOL	memory_pool
	list active databases		dbase
SQLMA_DCS_DBASE	get snapshot for dcs database on *dbname*		dcs_dbase 和 stmt_transmissions
SQLMA_DCS_DBASE_ALL	get snapshot for all dcs databases		dcs_dbase 和 stmt_transmissions
SQLMA_DBASE_REMOTE	get snapshot for remote database on *dbname*		dbase_remote
SQLMA_DBASE_REMOTE_ALL	get snapshot for all remote databases		dbase_remote

表 149. 快照监视器接口至逻辑数据组的映射 (续)

db2GetSnapshot API 请求类型	CLP 命令	SQL 管理视图	逻辑数据组
SQLMA_APPL	get snapshot for application applid *appl-id*	SNAPAPPL	appl
		SNAPAGENT	agent
		SNAPAPPL_INFO	appl_info
		SNAPLOCKWAIT	lock_wait
		SNAPSTMT	stmt
		SNAPSUBSECTION	subsection
		SNAPAGENT_MEMORY_POOL	memory_pool
SQLMA_AGENT_ID	get snapshot for application agentid *appl-handle*	SNAPAGENT	appl
		SNAPAGENT	agent
		SNAPAPPL_INFO	appl_info
		SNAPLOCKWAIT	lock_wait
		SNAPSTMT	stmt
		SNAPSUBSECTION	subsection
		SNAPAGENT_MEMORY_POOL	memory_pool
SQLMA_DBASE_APPLS	get snapshot for applications on *dbname*	SNAPAPPL	appl
		SNAPAGENT	agent
		SNAPAPPL_INFO	appl_info
		SNAPLOCKWAIT	lock_wait
		SNAPSTMT	stmt
		SNAPSUBSECTION	subsection
		SNAPAGENT_MEMORY_POOL	memory_pool
SQLMA_APPL_ALL	get snapshot for all applications	SNAPAPPL	appl
		SNAPAPPL_INFO	appl_info
		SNAPLOCKWAIT	lock_wait
		SNAPSTATEMENT	stmt
		SNAPAGENT	agent
		SNAPSUBSECTION	subsection
		SNAPAGENT_MEMORY_POOL	memory_pool
SQLMA_DCS_APPL	get snapshot for dcs application applid *appl-id*		dcs_appl、dcs_stmt、dcs_appl_info 和 stmt_transmissions
SQLMA_DCS_APPL_ALL	get snapshot for all dcs applications		dcs_appl、dcs_stmt、dcs_appl_info 和 stmt_transmissions
SQLMA_DCS_APPL_HANDLE	get snapshot for dcs application agentid *appl-handle*		dcs_appl、dcs_stmt、dcs_appl_info 和 stmt_transmissions
SQLMA_DCS_DBASE_APPLS	get snapshot for dcs applications on *dbname*		dcs_appl、dcs_stmt、dcs_appl_info 和 stmt_transmissions

表 149. 快照监视器接口至逻辑数据组的映射 (续)

db2GetSnapshot API 请求类型	CLP 命令	SQL 管理视图	逻辑数据组
SQLMA_DBASE_APPLS_REMOTE	get snapshot for remote applications on *dbname*		dbase_appl
SQLMA_APPL_REMOTE_ALL	get snapshot for all remote applications		dbase_appl
SQLMA_DBASE_TABLES	get snapshot for tables on *dbname*	SNAPTAB	表
		SNAPTAB_REORG	table_reorg
			table_list
SQLMA_APPL_LOCKS	get snapshot for locks for application applid *appl-id*	SNAPLOCK、SNAPAPPL 和 SNAPLOCKWAIT	appl_lock_list、lock_wait 和 lock
SQLMA_APPL_LOCKS_AGENT_ID	get snapshot for locks for application agentid *appl-handle*	SNAPLOCK、SNAPAPPL 和 SNAPLOCKWAIT	appl_lock_list、lock_wait 和 lock
SQLMA_DBASE_LOCKS	get snapshot for locks on *dbname*	SNAPLOCK	appl_lock_list 和 lock
		SNAPLOCK 和 SNAPLOCKWAIT	db_lock_list 和 lock_wait
SQLMA_DBASE_TABLESPACES	get snapshot for tablespaces on *dbname*	SNAPTBSP	tablespace
		SNAPTBSPPART	tablespace 和 tablespace_nodeinfo
		SNAPTBSP_QUIESCER	tablespace_quiescer 和 tablespace_nodeinfo
		SNAPCONTAINER	tablespace_container 和 tablespace_nodeinfo
		SNAPTBSP_RANGE	tablespace_ranges 和 tablespace_nodeinfo
			tablespace_list 和 tablespace_nodeinfo
SQLMA_BUFFERPOOLS_ALL	get snapshot for all bufferpools	SNAPBP	bufferpool
SQLMA_DBASE_BUFFERPOOLS	get snapshot for bufferpools on *dbname*	SNAPBP	bufferpool
SQLMA_DYNAMIC_SQL	get snapshot for dynamic sql on *dbname*	SNAPDYN_SQL	dynsql
			dynsql_list

下图显示逻辑数据分组在快照数据流中可能出现的顺序。

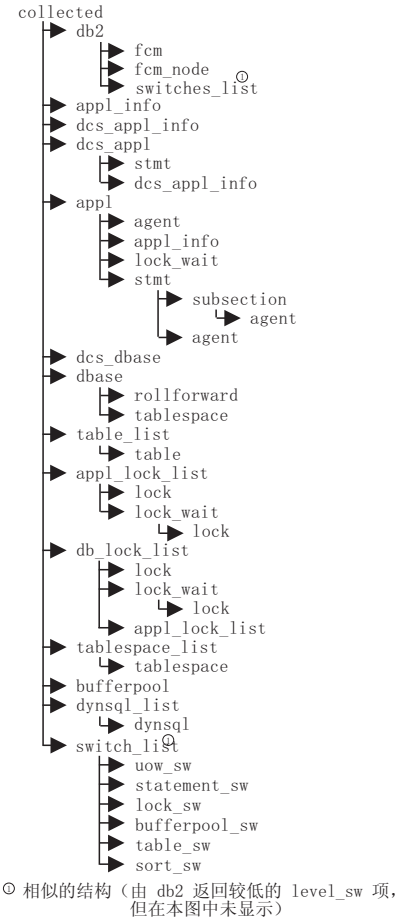


图 18. 数据流层次结构

注: 时间可能作为任何逻辑数据分组的一部分返回。

快照监视器逻辑数据组和监视元素

以下各部分列示快照监视可能返回的逻辑数据分组和监视元素。

- 第 557 页的『代理程序逻辑数据组』
- 第 557 页的『应用程序逻辑数据组』
- 第 560 页的『appl_id_info 逻辑数据组』
- 第 561 页的『appl_info 逻辑数据组』
- 第 561 页的『appl_lock_list 逻辑数据组』
- 第 562 页的『appl_remote 逻辑数据组』
- 第 562 页的『bufferpool 逻辑数据组』
- 第 564 页的『bufferpool_nodeinfo 逻辑数据组』
- 第 564 页的『collected 逻辑数据组』
- 第 564 页的『db2 逻辑数据组』
- 第 565 页的『db_lock_list 逻辑数据组』
- 第 565 页的『dbase 逻辑数据组』
- 第 569 页的『dbase_remote 逻辑数据组』

- 第 570 页的『db_storage_group 逻辑数据组』
- 第 570 页的『dcs_appl 逻辑数据组』
- 第 572 页的『dcs_appl_info 逻辑数据组』
- 第 573 页的『dcs_dbase 逻辑数据组』
- 第 574 页的『dcs_stmt 逻辑数据组』
- 第 575 页的『detail_log 逻辑数据组』
- 第 575 页的『dynsql 逻辑数据组』
- 第 576 页的『dynsql_list 逻辑数据组』
- 第 576 页的『fcm 逻辑数据组』
- 第 576 页的『fcm_node 逻辑数据组』
- 第 576 页的『hadr 逻辑数据组』
- 第 577 页的『lock 逻辑数据组』
- 第 577 页的『lock_wait 逻辑数据组』
- 第 578 页的『memory_pool 逻辑数据组』
- 第 578 页的『progress 逻辑数据组』
- 第 578 页的『progress_list 逻辑数据组』
- 第 578 页的『rollforward 逻辑数据组』
- 第 579 页的『stmt 逻辑数据组』
- 第 580 页的『stmt_transmissions 逻辑数据组』
- 第 581 页的『subsection 逻辑数据组』
- 第 582 页的『table 逻辑数据组』
- 第 582 页的『table_list 逻辑数据组』
- 第 582 页的『table_reorg 逻辑数据组』
- 第 583 页的『tablespace 逻辑数据组』
- 第 584 页的『tablespace_container 逻辑数据组』
- 第 584 页的『tablespace_list 逻辑数据组』
- 第 584 页的『tablespace_nodeinfo 逻辑数据组』
- 第 585 页的『tablespace_quiescer 逻辑数据组』
- 第 585 页的『tablespace_range 逻辑数据组』
- 第 586 页的『utility_info 逻辑数据组』

代理程序逻辑数据组

第 603 页的『agent_pid -“引擎可分派单元 (EDU) 标识”监视元素』

第 859 页的『lock_timeout_val -“锁定超时值”监视元素』

应用程序逻辑数据组

第 588 页的『acc_curs_blk -“接受的块游标请求数”』

第 603 页的『agent_sys_cpu_time -“代理程序使用的系统 CPU 时间”』

第 604 页的『agent_usr_cpu_time -“代理程序使用的用户 CPU 时间”』

第 609 页的『agents_stolen -“失窃代理程序数”』

第 616 页的『appl_con_time -“连接请求启动时间戳记”』

第 909 页的『num_agents -“正在处理语句的代理程序数”』
第 933 页的『olap_func_overflows -“OLAP 函数溢出次数”监视元素』
第 934 页的『open_loc_curs -“打开的本地游标数”』
第 934 页的『open_loc_curs_blk -“打开的本地分块游标数”』
第 935 页的『open_rem_curs -“打开的远程游标数”』
第 935 页的『open_rem_curs_blk -“打开的远程分块游标数”』
第 955 页的『pkg_cache_inserts -“程序包高速缓存插入数”监视元素』
第 956 页的『pkg_cache_lookups -“程序包高速缓存查询数”监视元素』
第 982 页的『pool_data_l_reads -“缓冲池数据逻辑读取数”监视元素』
第 984 页的『pool_data_p_reads -“缓冲池数据物理读取数”监视元素』
第 986 页的『pool_data_writes -“缓冲池数据写次数”监视元素』
第 1012 页的『pool_index_l_reads -“缓冲池索引逻辑读取数”监视元素』
第 1014 页的『pool_index_p_reads -“缓冲池索引物理读取数”监视元素』
第 1016 页的『pool_index_writes -“缓冲池索引写次数”监视元素』
第 1043 页的『pool_read_time -“缓冲池物理读时间总计”监视元素』
第 1046 页的『pool_temp_data_l_reads -“缓冲池临时数据逻辑读取数”监视元素』
第 1048 页的『pool_temp_data_p_reads -“缓冲池临时数据物理读取数”监视元素』
第 1050 页的『pool_temp_index_l_reads -“缓冲池临时索引逻辑读取数”监视元素』
第 1052 页的『pool_temp_index_p_reads -“缓冲池临时索引物理读取数”监视元素』
第 1053 页的『pool_temp_xda_l_reads -“缓冲池临时 XDA 数据逻辑读取数”监视元素』
第 1055 页的『pool_temp_xda_p_reads -“缓冲池临时 XDA 数据物理读取数”监视元素』
第 1058 页的『pool_write_time -“缓冲池物理写时间总计”监视元素』
第 1065 页的『pool_xda_l_reads -“缓冲池 XDA 数据逻辑读取数”监视元素』
第 1069 页的『pool_xda_p_reads -“缓冲池 XDA 数据物理读取数”监视元素』
第 1071 页的『pool_xda_writes -“缓冲池 XDA 数据写次数”监视元素』
第 1080 页的『prefetch_wait_time -“等待预取的时间”监视元素』
第 1084 页的『prev_uow_stop_time -“上一个工作单元完成时间戳记”』
第 1085 页的『priv_workspace_num_overflows -“专用工作空间溢出数”』
第 1086 页的『priv_workspace_section_inserts -“专用工作空间节插入数”』
第 1086 页的『priv_workspace_section_lookups -“专用工作空间节查询数”』
第 1087 页的『priv_workspace_size_top -“最大专用工作空间大小”』
第 1102 页的『rej_curs_blk -“拒绝的块游标请求数”』
第 1112 页的『rollback_sql_stmts -“尝试的回滚语句数”』
第 1115 页的『rows_deleted -“删除行数”监视元素』
第 1116 页的『rows_inserted -“插入行数”监视元素』
第 1118 页的『rows_read -“读取行数”监视元素』
第 1122 页的『rows_selected -“选择的行数”』
第 1123 页的『rows_updated -“更新行数”监视元素』
第 1123 页的『rows_written -“写入的行数”』

第 1129 页的『select_sql_stmts -“执行的 Select SQL 语句数”』
第 1138 页的『shr_workspace_num_overflows -“共享工作空间溢出数”』
第 1138 页的『shr_workspace_section_inserts -“共享工作空间节插入数”』
第 1139 页的『shr_workspace_section_lookups -“共享工作空间节查询数”』
第 1140 页的『shr_workspace_size_top -“最大共享工作空间大小”』
第 1149 页的『sort_overflows -“排序溢出数”监视元素』
第 1153 页的『sql_reqs_since_commit -“上次落实后的 SQL 请求数”』
第 1165 页的『static_sql_stmts -“尝试的静态 SQL 语句数”』
第 1256 页的『total_hash_joins -“散列连接总数”』
第 1256 页的『total_hash_loops -“总散列循环数”』
第 1264 页的『total_olap_funcs -“OLAP 函数总数”监视元素』
第 1291 页的『total_sort_time -“排序时间总计”监视元素』
第 1292 页的『total_sorts -“排序总数”监视元素』
第 1313 页的『uid_sql_stmts -“执行的 Update/Insert/Delete SQL 语句数”』
第 1314 页的『unread_prefetch_pages -“未读取的预取页数”监视元素』
第 1315 页的『uow_comp_status -“工作单元完成状态”』
第 1316 页的『uow_elapsed_time -“最新工作单元耗用时间”』
第 1318 页的『uow_lock_wait_time -“工作单元等待锁定的总时间”监视元素』
第 1318 页的『uow_log_space_used -“使用的工作单元日志空间”监视元素』
第 1319 页的『uow_start_time -“工作单元开始时间戳记”监视元素』
第 1320 页的『uow_stop_time -“工作单元停止时间戳记”监视元素』
第 1344 页的『x_lock_escals -“互斥锁定升级数”监视元素』
第 1346 页的『xquery_stmts -“尝试的 XQuery 语句数”』

appl_id_info 逻辑数据组

第 601 页的『agent_id -“应用程序句柄（代理程序标识）”监视元素』
第 616 页的『appl_id -“应用程序标识”监视元素』
第 620 页的『appl_name -“应用程序名称”监视元素』
第 623 页的『appl_status - 应用程序状态监视元素』
第 635 页的『auth_id -“授权标识”』
第 659 页的『client_db_alias -“应用程序使用的数据库别名”』
第 663 页的『client_prdid -“客户机产品和版本标识”监视元素』
第 666 页的『codepage_id -“应用程序使用的代码页标识”』
第 719 页的『db_name - 数据库名称监视元素』
第 720 页的『db_path -“数据库路径”』
第 819 页的『input_db_alias -“输入数据库别名”』
第 1130 页的『sequence_no -“序号”监视元素』
第 1167 页的『status_change_time -“应用程序状态更改时间”』

appl_info 逻辑数据组

第 601 页的『agent_id -“应用程序句柄（代理程序标识）”监视元素』

第 616 页的『appl_id -“应用程序标识”监视元素』

第 620 页的『appl_name -“应用程序名称”监视元素』

第 622 页的『appl_section_inserts -“节插入数”监视元素』

第 622 页的『appl_section_lookups -“节查询数”』

第 623 页的『appl_status - 应用程序状态监视元素』

第 635 页的『auth_id -“授权标识”』

第 636 页的『authority_bitmap -“用户权限级别”监视元素』

第 636 页的『authority_lvl -“用户权限级别”监视元素』

第 659 页的『client_db_alias -“应用程序使用的数据库别名”』

第 661 页的『client_pid -“客户机进程标识”监视元素』

第 662 页的『client_platform -“客户机操作平台”监视元素』

第 663 页的『client_prdid -“客户机产品和版本标识”监视元素』

第 664 页的『client_protocol -“客户机通信协议”监视元素』

第 666 页的『codepage_id -“应用程序使用的代码页标识”』

第 694 页的『coord_agent_pid -“协调代理程序标识”监视元素』

第 695 页的『coord_node -“协调节点”』

第 697 页的『corr_token -“DRDA 关联标记”』

第 719 页的『db_name - 数据库名称监视元素』

第 720 页的『db_path -“数据库路径”』

第 764 页的『execution_id -“用户登录标识”』

第 819 页的『input_db_alias -“输入数据库别名”』

第 833 页的『is_system_appl -“是系统应用程序”监视元素』

第 909 页的『num_assoc_agents -“关联代理程序数”』

第 1130 页的『sequence_no -“序号”监视元素』

第 1137 页的『session_auth_id -“会话授权标识”监视元素』

第 1167 页的『status_change_time -“应用程序状态更改时间”』

第 1223 页的『territory_code -“数据库地域代码”』

第 1303 页的『tpmon_acc_str -“TP 监视器客户机记帐字符串”监视元素』

第 1304 页的『tpmon_client_app -“TP 监视器客户机应用程序名称”监视元素』

第 1304 页的『tpmon_client_userid -“TP 监视器客户机用户标识”监视元素』

第 1305 页的『tpmon_client_wkstn -“TP 监视器客户机工作站名称”监视元素』

第 1341 页的『workload_id -“工作负载标识”监视元素』

appl_lock_list 逻辑数据组

第 601 页的『agent_id -“应用程序句柄（代理程序标识）”监视元素』

第 616 页的『appl_id -“应用程序标识”监视元素』

第 620 页的『appl_name -“应用程序名称”监视元素』

第 623 页的『appl_status - 应用程序状态监视元素』

第 635 页的『auth_id -“授权标识”』

第 659 页的『client_db_alias -“应用程序使用的数据库别名”』

第 666 页的『codepage_id -“应用程序使用的代码页标识”』

第 863 页的『lock_wait_time -“等待锁定时间”监视元素』

第 871 页的『locks_held -“挂起的锁定数”监视元素』

第 872 页的『locks_waiting -“当前正在等待锁定的代理程序数”监视元素』

第 1130 页的『sequence_no -“序号”监视元素』

第 1137 页的『session_auth_id -“会话授权标识”监视元素』

第 1167 页的『status_change_time -“应用程序状态更改时间”』

appl_remote 逻辑数据组

第 668 页的『commit_sql_stmts -“尝试的落实语句数”』

第 708 页的『create_nickname -“创建昵称数”』

第 708 页的『create_nickname_time -“创建昵称响应时间”』

第 714 页的『datasource_name -“数据源名称”』

第 719 页的『db_name - 数据库名称监视元素』

第 731 页的『delete_sql_stmts -“删除数”』

第 731 页的『delete_time -“删除响应时间”』

第 764 页的『failed_sql_stmts -“失败的语句操作”』

第 819 页的『insert_sql_stmts -“插入数”』

第 819 页的『insert_time -“插入响应时间”』

第 952 页的『passthru_time -“传递时间”』

第 952 页的『passthrus -“传递数”』

第 1103 页的『remote_lock_time -“远程锁定时间”』

第 1103 页的『remote_locks -“远程锁定”』

第 1112 页的『rollback_sql_stmts -“尝试的回滚语句数”』

第 1115 页的『rows_deleted -“删除行数”监视元素』

第 1116 页的『rows_inserted -“插入行数”监视元素』

第 1122 页的『rows_selected -“选择的行数”』

第 1123 页的『rows_updated -“更新行数”监视元素』

第 1129 页的『select_sql_stmts -“执行的 Select SQL 语句数”』

第 1130 页的『select_time -“查询响应时间”』

第 1152 页的『sp_rows_selected -“存储过程返回的行数”』

第 1186 页的『stored_proc_time -“存储过程时间”』

第 1186 页的『stored_procs -“存储过程”』

第 1322 页的『update_sql_stmts -“更新数”』

第 1322 页的『update_time -“更新响应时间”』

bufferpool 逻辑数据组

第 639 页的『block_ios -“块 I/O 请求数”监视元素』

第 642 页的『bp_id -“缓冲池标识”监视元素』

第 642 页的『bp_name -“缓冲池名称”监视元素』
第 719 页的『db_name - 数据库名称监视元素』
第 720 页的『db_path -“数据库路径”』
第 735 页的『direct_read_reqs -“直接读请求数”监视元素』
第 737 页的『direct_read_time -“直接读时间”监视元素』
第 738 页的『direct_reads -“直接读数据库数目”监视元素』
第 740 页的『direct_write_reqs -“直接写请求数”监视元素』
第 742 页的『direct_write_time -“直接写时间”监视元素』
第 744 页的『direct_writes -“直接写数据库数目”监视元素』
第 790 页的『files_closed -“关闭数据库文件数”监视元素』
第 819 页的『input_db_alias -“输入数据库别名”』
第 946 页的『pages_from_block_ios -“块 I/O 读取总页数”监视元素』
第 947 页的『pages_from_vectored_ios -“向量 I/O 读取总页数”监视元素』
第 961 页的『pool_async_data_read_reqs -“缓冲池异步读请求数”监视元素』
第 962 页的『pool_async_data_reads -“缓冲池异步数据读次数”监视元素』
第 963 页的『pool_async_data_writes -“缓冲池异步数据写次数”监视元素』
第 966 页的『pool_async_index_read_reqs -“缓冲池异步索引读请求数”监视元素』
第 966 页的『pool_async_index_reads -“缓冲池异步索引读取数”监视元素』
第 967 页的『pool_async_index_writes -“缓冲池异步索引写次数”监视元素』
第 968 页的『pool_async_read_time -“缓冲池异步读取时间”』
第 969 页的『pool_async_write_time -“缓冲池异步写入时间”监视元素』
第 972 页的『pool_async_xda_read_reqs -“缓冲池异步 XDA 读请求数”监视元素』
第 973 页的『pool_async_xda_reads -“缓冲池异步 XDA 数据读取数”监视元素』
第 974 页的『pool_async_xda_writes -“缓冲池异步 XDA 数据写次数”监视元素』
第 982 页的『pool_data_l_reads -“缓冲池数据逻辑读取数”监视元素』
第 984 页的『pool_data_p_reads -“缓冲池数据物理读取数”监视元素』
第 986 页的『pool_data_writes -“缓冲池数据写次数”监视元素』
第 1012 页的『pool_index_l_reads -“缓冲池索引逻辑读取数”监视元素』
第 1014 页的『pool_index_p_reads -“缓冲池索引物理读取数”监视元素』
第 1016 页的『pool_index_writes -“缓冲池索引写次数”监视元素』
第 1018 页的『pool_no_victim_buffer -“缓冲池无牺牲缓冲区次数”监视元素』
第 1043 页的『pool_read_time -“缓冲池物理读时间总计”监视元素』
第 1046 页的『pool_temp_data_l_reads -“缓冲池临时数据逻辑读取数”监视元素』
第 1048 页的『pool_temp_data_p_reads -“缓冲池临时数据物理读取数”监视元素』
第 1050 页的『pool_temp_index_l_reads -“缓冲池临时索引逻辑读取数”监视元素』
第 1052 页的『pool_temp_index_p_reads -“缓冲池临时索引物理读取数”监视元素』
第 1053 页的『pool_temp_xda_l_reads -“缓冲池临时 XDA 数据逻辑读取数”监视元素』
第 1055 页的『pool_temp_xda_p_reads -“缓冲池临时 XDA 数据物理读取数”监视元素』
第 1058 页的『pool_write_time -“缓冲池物理写时间总计”监视元素』

第 1065 页的『pool_xda_l_reads -“缓冲池 XDA 数据逻辑读取数”监视元素』

第 1069 页的『pool_xda_p_reads -“缓冲池 XDA 数据物理读取数”监视元素』

第 1071 页的『pool_xda_writes -“缓冲池 XDA 数据写次数”监视元素』

第 1334 页的『vectored_ios -“向量 I/O 请求数”监视元素』

bufferpool_nodeinfo 逻辑数据组

第 642 页的『bp_cur_buffsz -“缓冲池的当前大小”』

第 643 页的『bp_new_buffsz -“新的缓冲池大小”』

第 643 页的『bp_pages_left_to_remove -“要除去的余下页数”』

第 643 页的『bp_tbsp_use_count -“映射至缓冲池的表空间数”』

第 908 页的『node_number -“节点号”』

collected 逻辑数据组

第 908 页的『node_number -“节点号”』

第 1132 页的『server_db2_type -“受监视的（服务器）节点上的数据库管理器类型”』

第 1132 页的『server_instance_name -“服务器实例名称”』

第 1133 页的『server_prdid -“服务器产品/版本标识”』

第 1133 页的『server_version -“服务器版本”』

第 1229 页的『time_stamp -“快照时间”』

第 1229 页的『time_zone_disp -“时区偏移”』

db2 逻辑数据组

第 607 页的『agents_created_empty_pool -“由于空的代理程序池而创建的代理程序数”』

第 607 页的『agents_from_pool -“从池中分配的代理程序数”』

第 608 页的『agents_registered -“已注册的代理程序数”』

第 608 页的『agents_registered_top -“已注册的最大代理程序数”』

第 609 页的『agents_stolen -“失窃代理程序数”』

第 609 页的『agents_waiting_on_token -“正在等待令牌的代理程序数”』

第 610 页的『agents_waiting_top -“正在等待的最大代理程序数”监视元素』

第 668 页的『comm_private_mem -“已落实的专用内存”』

第 671 页的『con_local_dbases -“带有当前连接的本地数据库”』

第 694 页的『coord_agents_top -“最大协调代理程序数”』

第 717 页的『db2start_time -“启动数据库管理器时间戳记”』

第 720 页的『db_status - 数据库状态监视元素』

第 795 页的『gw_cons_wait_client -“等待客户机发送请求的连接数”』

第 795 页的『gw_cons_wait_host -“等待主机应答的连接数”』

第 796 页的『gw_cur_cons -“DB2 Connect 的当前连接数”』

第 797 页的『gw_total_cons -“对 DB2 Connect 尝试连接的总数”』

第 814 页的『idle_agents -“空闲代理程序数”』

第 837 页的『last_reset -“最后重置时间戳记”』

第 839 页的『local_cons -“本地连接数”』

第 840 页的『local_cons_in_exec -“数据库管理器中正在执行的本地连接数”』

第 881 页的『max_agent_overflows -“最大代理程序溢出次数”』

第 913 页的『num_gw_conn_switches -“连接交换次数”』

第 918 页的『num_nodes_in_db2_instance -“分区中的节点数”』

第 954 页的『piped_sorts_accepted -“接受的管道排序数”』

第 954 页的『piped_sorts_requested -“请求的管道排序数”』

第 1074 页的『post_threshold_hash_joins -“散列连接阈值”』

第 1075 页的『post_threshold_olap_funcs -“OLAP 函数阈值”监视元素』

第 1079 页的『post_threshold_sorts -“超出阈值后的排序次数”监视元素』

第 1087 页的『product_name -“产品名称”』

第 1102 页的『rem_cons_in -“与数据库管理器的远程连接数”』

第 1103 页的『rem_cons_in_exec -“数据库管理器中正在执行的远程连接数”』

第 1135 页的『service_level -“服务级别”』

第 1148 页的『smallest_log_avail_node -“带有最少可用日志空间的节点”』

第 1148 页的『sort_heap_allocated -“分配的总排序堆”』

第 1149 页的『sort_heap_top -“排序专用堆高水位标记”』

db_lock_list 逻辑数据组

第 626 页的『appls_cur_cons -“当前连接的应用程序数”』

第 719 页的『db_name - 数据库名称监视元素』

第 720 页的『db_path -“数据库路径”』

第 819 页的『input_db_alias -“输入数据库别名”』

第 871 页的『locks_held -“挂起的锁定数”监视元素』

第 872 页的『locks_waiting -“当前正在等待锁定的代理程序数”监视元素』

dbase 逻辑数据组

第 596 页的『active_hash_joins -“活动散列连接数”』

第 596 页的『active_olap_funcs -“活动 OLAP 函数”监视元素』

第 596 页的『active_sorts -“活动排序次数”』

第 609 页的『agents_top -“创建的代理程序数”』

第 619 页的『appl_id_oldest_xact -“带有最旧事务的应用程序”』

第 622 页的『appl_section_inserts -“节插入数”监视元素』

第 622 页的『appl_section_lookups -“节查询数”』

第 626 页的『appls_cur_cons -“当前连接的应用程序数”』

第 626 页的『appls_in_db2 -“数据库中当前执行的应用程序数”』

第 627 页的『async_runstats -“异步 RUNSTATS 请求总数”监视元素』

第 639 页的『binds_precompiles -“尝试的绑定次数/预编译次数”』

第 641 页的『blocks_pending_cleanup -“暂挂清除已转出块”监视元素』

第 646 页的『cat_cache_inserts -“目录高速缓存插入数”监视元素』

第 647 页的『cat_cache_lookups -“目录高速缓存查询数”监视元素』

第 649 页的『cat_cache_overflows -“目录高速缓存溢出数”』

第 871 页的『locks_held -“挂起的锁定数”监视元素』

第 872 页的『locks_waiting -“当前正在等待锁定的代理程序数”监视元素』

第 876 页的『log_held_by_dirty_pages -“脏页占用的日志空间量”』

第 877 页的『log_read_time -“日志读取时间”』

第 877 页的『log_reads -“读取的日志页数”』

第 878 页的『log_to_redo_for_recovery -“要为恢复重做的日志量”』

第 878 页的『log_write_time -“日志写入时间”』

第 879 页的『log_writes -“写入的日志页数”』

第 909 页的『num_assoc_agents -“关联代理程序数”』

第 911 页的『num_db_storage_paths -“自动存储器路径数”』

第 913 页的『num_indoubt_trans -“不确定事务数”』

第 913 页的『num_log_buffer_full -“日志缓冲区变满而导致代理程序等待的次数”监视元素』

第 915 页的『num_log_data_found_in_buffer -“在缓冲区中找到日志数据的次数”』

第 915 页的『num_log_part_page_io -“部分日志页写入数”』

第 916 页的『num_log_read_io -“日志读取数”』

第 916 页的『num_log_write_io -“日志写入次数”』

第 933 页的『olap_func_overflows -“OLAP 函数溢出次数”监视元素』

第 955 页的『pkg_cache_inserts -“程序包高速缓存插入数”监视元素』

第 956 页的『pkg_cache_lookups -“程序包高速缓存查询数”监视元素』

第 958 页的『pkg_cache_num_overflows -“程序包高速缓存溢出数”』

第 958 页的『pkg_cache_size_top -“程序包高速缓存高水位标记”』

第 961 页的『pool_async_data_read_reqs -“缓冲池异步读请求数”监视元素』

第 962 页的『pool_async_data_reads -“缓冲池异步数据读次数”监视元素』

第 963 页的『pool_async_data_writes -“缓冲池异步数据写次数”监视元素』

第 966 页的『pool_async_index_read_reqs -“缓冲池异步索引读请求数”监视元素』

第 966 页的『pool_async_index_reads -“缓冲池异步索引读取数”监视元素』

第 967 页的『pool_async_index_writes -“缓冲池异步索引写次数”监视元素』

第 968 页的『pool_async_read_time -“缓冲池异步读取时间”』

第 969 页的『pool_async_write_time -“缓冲池异步写入时间”监视元素』

第 972 页的『pool_async_xda_read_reqs -“缓冲池异步 XDA 读请求数”监视元素』

第 973 页的『pool_async_xda_reads -“缓冲池异步 XDA 数据读取数”监视元素』

第 974 页的『pool_async_xda_writes -“缓冲池异步 XDA 数据写次数”监视元素』

第 982 页的『pool_data_l_reads -“缓冲池数据逻辑读取数”监视元素』

第 984 页的『pool_data_p_reads -“缓冲池数据物理读取数”监视元素』

第 986 页的『pool_data_writes -“缓冲池数据写次数”监视元素』

第 988 页的『pool_drty_pg_steal_clns -“触发缓冲池牺牲页清除程序次数”监视元素』

第 989 页的『pool_drty_pg_thrsh_clns -“触发缓冲池阈值清除程序次数”监视元素』

第 1012 页的『pool_index_l_reads -“缓冲池索引逻辑读取数”监视元素』

第 1014 页的『pool_index_p_reads -“缓冲池索引物理读取数”监视元素』

第 1165 页的『stats_cache_size -“统计信息高速缓存大小”监视元素』

第 1166 页的『stats_fabricate_time -“生成统计信息的活动所花的总时间”监视元素』

第 1167 页的『stats_fabrications -“生成统计信息的次数”监视元素』

第 1187 页的『sync_runstats -“同步 RUNSTATS 活动总数”监视元素』

第 1188 页的『sync_runstats_time -“同步 RUNSTATS 活动所花的总时间”监视元素』

第 1230 页的『tot_log_used_top -“使用的最大总日志空间”』

第 1243 页的『total_cons -“数据库激活以后的连接数”』

第 1256 页的『total_hash_joins -“散列连接总数”』

第 1256 页的『total_hash_loops -“总散列循环数”』

第 1262 页的『total_log_available -“可用的总日志量”』

第 1263 页的『total_log_used -“使用的总日志空间”』

第 1264 页的『total_olap_funcs -“OLAP 函数总数”监视元素』

第 1283 页的『total_sec_cons -“辅助连接数”』

第 1291 页的『total_sort_time -“排序时间总计”监视元素』

第 1292 页的『total_sorts -“排序总数”监视元素』

第 1313 页的『uid_sql_stmts -“执行的 Update/Insert/Delete SQL 语句数”』

第 1314 页的『unread_prefetch_pages -“未读取的预取页数”监视元素』

第 1344 页的『x_lock_escals -“互斥锁定升级数”监视元素』

第 1346 页的『xquery_stmts -“尝试的 XQuery 语句数”』

dbase_remote 逻辑数据组

第 668 页的『commit_sql_stmts -“尝试的落实语句数”』

第 708 页的『create_nickname -“创建昵称数”』

第 708 页的『create_nickname_time -“创建昵称响应时间”』

第 714 页的『datasource_name -“数据源名称”』

第 719 页的『db_name - 数据库名称监视元素』

第 731 页的『delete_sql_stmts -“删除数”』

第 731 页的『delete_time -“删除响应时间”』

第 748 页的『disconnects -“断开连接次数”』

第 764 页的『failed_sql_stmts -“失败的语句操作”』

第 819 页的『insert_sql_stmts -“插入数”』

第 819 页的『insert_time -“插入响应时间”』

第 952 页的『passthru_time -“传递时间”』

第 952 页的『passthrus -“传递数”』

第 1103 页的『remote_lock_time -“远程锁定时间”』

第 1103 页的『remote_locks -“远程锁定”』

第 1112 页的『rollback_sql_stmts -“尝试的回滚语句数”』

第 1115 页的『rows_deleted -“删除行数”监视元素』

第 1116 页的『rows_inserted -“插入行数”监视元素』

第 1122 页的『rows_selected -“选择的行数”』

第 1123 页的『rows_updated -“更新行数”监视元素』

第 1129 页的『select_sql_stmts -“执行的 Select SQL 语句数”』

第 1130 页的『select_time -“查询响应时间”』

第 1152 页的『sp_rows_selected -“存储过程返回的行数”』

第 1186 页的『stored_proc_time -“存储过程时间”』

第 1186 页的『stored_procs -“存储过程”』

第 1243 页的『total_cons -“数据库激活以后的连接数”』

第 1322 页的『update_sql_stmts -“更新数”』

第 1322 页的『update_time -“更新响应时间”』

db_storage_group 逻辑数据组

第 792 页的『fs_id -“唯一文件系统标识号”监视元素』

第 793 页的『fs_total_size -“文件系统总大小”监视元素』

第 793 页的『fs_used_size -“文件系统中的已用空间量”监视元素』

第 908 页的『node_number -“节点号”』

第 1184 页的『sto_path_free_sz -“自动存储器路径可用空间量”监视元素』

dcs_appl 逻辑数据组

第 619 页的『appl_idle_time -“应用程序空闲时间”』

第 668 页的『commit_sql_stmts -“尝试的落实语句数”』

第 751 页的『elapsed_exec_time -“语句执行耗用时间”』

第 764 页的『failed_sql_stmts -“失败的语句操作”』

第 795 页的『gw_con_time -“DB2 Connect 网关首次启动的连接”』

第 796 页的『gw_exec_time -“DB2 Connect 网关处理所耗用的时间”』

第 813 页的『host_response_time -“主机响应时间”』

第 815 页的『inbound_bytes_received -“接收的入站字节数”』

第 815 页的『inbound_bytes_sent -“发送的入站字节数”』

第 837 页的『last_reset -“最后重置时间戳记”』

第 884 页的『max_data_received_1024 -“接收的出站字节数在 513 到 1024 字节之间的语句数”』

第 885 页的『max_data_received_128 -“接收的出站字节数在 1 到 128 字节之间的语句数”』

第 885 页的『max_data_received_16384 -“接收的出站字节数在 8193 到 16384 字节之间的语句数”』

第 886 页的『max_data_received_2048 -“接收的出站字节数在 1025 到 2048 字节之间的语句数”』

第 886 页的『max_data_received_256 -“接收的出站字节数在 129 到 256 字节之间的语句数”』

第 887 页的『max_data_received_31999 -“接收的出站字节数在 16385 到 31999 字节之间的语句数”监视元素』

第 887 页的『max_data_received_4096 -“接收的出站字节数在 2049 到 4096 字节之间的语句数”』

第 887 页的『max_data_received_512 -“接收的出站字节数在 257 到 512 字节之间的语句数”』

第 888 页的『max_data_received_64000 -“接收的出站字节数在 32000 到 64000 字节之间的语句数”监视元素』

第 888 页的『max_data_received_8192 -“接收的出站字节数在 4097 到 8192 字节之间的语句”』

第 889 页的『max_data_received_gt64000 -“接收的出站字节数高于 64000 的语句数”』

第 889 页的『max_data_sent_1024 -“发送的出站字节数在 513 到 1024 字节之间的语句数”』

第 889 页的『max_data_sent_128 -“发送的出站字节数在 1 到 128 字节之间的语句数”』

第 890 页的『max_data_sent_16384 -“发送的出站字节数在 8193 到 16384 字节之间的语句数”』

第 890 页的『max_data_sent_2048 -“发送的出站字节数在 1025 到 2048 字节之间的语句数”』

第 891 页的『max_data_sent_256 -“发送的出站字节数在 129 到 256 字节之间的语句数”』

第 891 页的『max_data_sent_31999 -“发送的出站字节数在 16385 到 31999 字节之间的语句数”』

第 892 页的『max_data_sent_4096 -“发送的出站字节数在 2049 到 4096 字节之间的语句数”』

第 892 页的『max_data_sent_512 -“发送的出站字节数在 257 到 512 字节之间的语句数”』

第 892 页的『max_data_sent_64000 -“发送的出站字节数在 32000 到 64000 字节之间的语句数”』

第 893 页的『max_data_sent_8192 -“发送的出站字节数在 4097 到 8192 字节之间的语句数”』

第 893 页的『max_data_sent_gt64000 -“发送的出站字节数高于 64000 的语句数”』

第 894 页的『max_network_time_100_ms -“网络时间在 16 到 100 毫秒之间的语句数”』

第 894 页的『max_network_time_16_ms -“网络时间在 4 到 16 毫秒之间的语句数”』

第 895 页的『max_network_time_1_ms -“网络时间最多为 1 毫秒的语句数”』

第 895 页的『max_network_time_4_ms -“网络时间在 1 到 4 毫秒之间的语句数”』

第 895 页的『max_network_time_500_ms -“网络时间在 100 到 500 毫秒之间的语句数”』

第 896 页的『max_network_time_gt500_ms -“网络时间大于 500 毫秒的语句数”』

第 906 页的『network_time_bottom -“语句的最短网络时间”』

第 907 页的『network_time_top -“语句的最长网络时间”』

第 933 页的『open_cursors -“打开的游标数”』

第 937 页的『outbound_bytes_received -“接收的出站字节数”』

第 938 页的『outbound_bytes_sent -“发送的出站字节数”』

第 1084 页的『prev_uow_stop_time -“上一个工作单元完成时间戳记”』

第 1112 页的『rollback_sql_stmts -“尝试的回滚语句数”』

第 1122 页的『rows_selected -“选择的行数”』

第 1153 页的『sql_stmts -“尝试的 SQL 语句数”』

第 1303 页的『tpmon_acc_str -“TP 监视器客户机记帐字符串”监视元素』

第 1304 页的『tpmon_client_app -“TP 监视器客户机应用程序名称”监视元素』

第 1304 页的『tpmon_client_userid -“TP 监视器客户机用户标识”监视元素』

第 1305 页的『tpmon_client_wkstn -“TP 监视器客户机工作站名称”监视元素』

第 1315 页的『uow_comp_status -“工作单元完成状态”』

第 1316 页的『uow_elapsed_time -“最新工作单元耗用时间”』

第 1319 页的『uow_start_time -“工作单元开始时间戳记”监视元素』

第 1320 页的『uow_stop_time -“工作单元停止时间戳记”监视元素』

第 1346 页的『xid -“事务标识”』

dcs_appl_info 逻辑数据组

第 601 页的『agent_id -“应用程序句柄（代理程序标识）”监视元素』

第 603 页的『agent_status -“DCS 应用程序代理程序数”』

第 616 页的『appl_id -“应用程序标识”监视元素』

第 620 页的『appl_name -“应用程序名称”监视元素』

第 635 页的『auth_id -“授权标识”』

第 661 页的『client_pid -“客户机进程标识”监视元素』

第 662 页的『client_platform -“客户机操作平台”监视元素』

第 663 页的『client_prdid -“客户机产品和版本标识”监视元素』

第 664 页的『client_protocol -“客户机通信协议”监视元素』

第 666 页的『codepage_id -“应用程序使用的代码页标识”』

第 725 页的『dcs_appl_status -“DCS 应用程序状态”监视元素』

第 725 页的『dcs_db_name -“DCS 数据库名称”』

第 764 页的『execution_id -“用户登录标识”』

第 796 页的『gw_db_alias -“网关上的数据库别名”』

第 811 页的『host_ccsid -“主机编码字符集标识”』

第 812 页的『host_db_name -“主机数据库名称”』

第 813 页的『host_prdid -“主机产品/版本标识”』

第 815 页的『inbound_comm_address -“入站通信地址”』

第 936 页的『outbound_appl_id -“出站应用程序标识”』

第 939 页的『outbound_comm_address -“出站通信地址”』

第 939 页的『outbound_comm_protocol -“出站通信协议”』

第 939 页的『outbound_sequence_no -“出站序号”』

第 1130 页的『sequence_no -“序号”监视元素』

第 1167 页的『status_change_time -“应用程序状态更改时间”』

dcs_dbase 逻辑数据组

第 889 页的『max_data_sent_1024 -“发送的出站字节数在 513 到 1024 字节之间的语句数”』

第 889 页的『max_data_sent_128 -“发送的出站字节数在 1 到 128 字节之间的语句数”』

第 890 页的『max_data_sent_16384 -“发送的出站字节数在 8193 到 16384 字节之间的语句数”』

第 890 页的『max_data_sent_2048 -“发送的出站字节数在 1025 到 2048 字节之间的语句数”』

第 891 页的『max_data_sent_256 -“发送的出站字节数在 129 到 256 字节之间的语句数”』

第 891 页的『max_data_sent_31999 -“发送的出站字节数在 16385 到 31999 字节之间的语句数”』

第 892 页的『max_data_sent_4096 -“发送的出站字节数在 2049 到 4096 字节之间的语句数”』

第 892 页的『max_data_sent_512 -“发送的出站字节数在 257 到 512 字节之间的语句数”』

第 892 页的『max_data_sent_64000 -“发送的出站字节数在 32000 到 64000 字节之间的语句数”』

第 893 页的『max_data_sent_8192 -“发送的出站字节数在 4097 到 8192 字节之间的语句数”』

第 893 页的『max_data_sent_gt64000 -“发送的出站字节数高于 64000 的语句数”』

第 894 页的『max_network_time_100_ms -“网络时间在 16 到 100 毫秒之间的语句数”』

第 894 页的『max_network_time_16_ms -“网络时间在 4 到 16 毫秒之间的语句数”』

第 895 页的『max_network_time_1_ms -“网络时间最多为 1 毫秒的语句数”』

第 895 页的『max_network_time_4_ms -“网络时间在 1 到 4 毫秒之间的语句数”』

第 895 页的『max_network_time_500_ms -“网络时间在 100 到 500 毫秒之间的语句数”』

第 896 页的『max_network_time_gt500_ms -“网络时间大于 500 毫秒的语句数”』

第 906 页的『network_time_bottom -“语句的最短网络时间”』

第 907 页的『network_time_top -“语句的最长网络时间”』

第 938 页的『outbound_bytes_sent -“发送的出站字节数”』

第 1112 页的『rollback_sql_stmts -“尝试的回滚语句数”』

第 1122 页的『rows_selected -“选择的行数”』

第 1153 页的『sql_stmts -“尝试的 SQL 语句数”』

dcs_stmt 逻辑数据组

第 640 页的『blocking_cursor -“分块游标”』

第 709 页的『creator -“应用程序创建者”』

第 751 页的『elapsed_exec_time -“语句执行耗用时间”』

第 789 页的『fetch_count -“成功的访存数”』

第 796 页的『gw_exec_time -“DB2 Connect 网关处理所耗用的时间”』

第 813 页的『host_response_time -“主机响应时间”』
第 815 页的『inbound_bytes_received -“接收的入站字节数”』
第 815 页的『inbound_bytes_sent -“发送的入站字节数”』
第 920 页的『num_transmissions -“传输次数”』
第 920 页的『num_transmissions_group -“传输组数目”』
第 937 页的『outbound_bytes_received -“接收的出站字节数”』
第 938 页的『outbound_bytes_sent -“发送的出站字节数”』
第 941 页的『package_name -“程序包名”监视元素』
第 1091 页的『query_card_estimate -“行查询数估计”』
第 1092 页的『query_cost_estimate -“查询估算成本”监视元素』
第 1128 页的『section_number -“节号”监视元素』
第 1168 页的『stmt_elapsed_time -“最新语句耗用时间”』
第 1173 页的『stmt_operation/operation -“语句操作”监视元素』
第 1177 页的『stmt_start -“语句操作开始时间戳记”』
第 1177 页的『stmt_stop -“语句操作停止时间戳记”』
第 1178 页的『stmt_text -“SQL 语句文本”监视元素』

detail_log 逻辑数据组

第 710 页的『current_active_log -“当前活动日志文件编号”』
第 711 页的『current_archive_log -“当前归档日志文件编号”』
第 791 页的『first_active_log -“第一个活动日志文件编号”』
第 834 页的『last_active_log -“最后一个活动日志文件编号”』
第 908 页的『node_number -“节点号”』

dynsql 逻辑数据组

第 789 页的『fetch_count -“成功的访存数”』
第 820 页的『insert_timestamp -“插入时间戳记”监视元素』
第 825 页的『int_rows_deleted -“删除的内部行数”』
第 826 页的『int_rows_inserted -“插入的内部行数”』
第 826 页的『int_rows_updated -“更新的内部行数”』
第 910 页的『num_compilations -“语句编译次数”』
第 911 页的『num_executions -“语句执行次数”监视元素』
第 982 页的『pool_data_l_reads -“缓冲池数据逻辑读取数”监视元素』
第 984 页的『pool_data_p_reads -“缓冲池数据物理读取数”监视元素』
第 1012 页的『pool_index_l_reads -“缓冲池索引逻辑读取数”监视元素』
第 1014 页的『pool_index_p_reads -“缓冲池索引物理读取数”监视元素』
第 1046 页的『pool_temp_data_l_reads -“缓冲池临时数据逻辑读取数”监视元素』
第 1048 页的『pool_temp_data_p_reads -“缓冲池临时数据物理读取数”监视元素』
第 1050 页的『pool_temp_index_l_reads -“缓冲池临时索引逻辑读取数”监视元素』
第 1052 页的『pool_temp_index_p_reads -“缓冲池临时索引物理读取数”监视元素』
第 1053 页的『pool_temp_xda_l_reads -“缓冲池临时 XDA 数据逻辑读取数”监视元素』

第 1055 页的『pool_temp_xda_p_reads -“缓冲池临时 XDA 数据物理读取数”监视元素』

第 1065 页的『pool_xda_l_reads -“缓冲池 XDA 数据逻辑读取数”监视元素』

第 1069 页的『pool_xda_p_reads -“缓冲池 XDA 数据物理读取数”监视元素』

第 1084 页的『prep_time_best -“语句最短编译时间”监视元素』

第 1084 页的『prep_time_worst -“语句最长编译时间”监视元素』

第 1118 页的『rows_read -“读取行数”监视元素』

第 1123 页的『rows_written -“写入的行数”』

第 1149 页的『sort_overflows -“排序溢出数”监视元素』

第 1166 页的『stats_fabricate_time -“生成统计信息的活动所花的总时间”监视元素』

第 1174 页的『stmt_pkgcache_id -“语句程序包高速缓存标识”监视元素』

第 1176 页的『stmt_sorts -“语句排序数”』

第 1178 页的『stmt_text -“SQL 语句文本”监视元素』

第 1188 页的『sync_runstats_time -“同步 RUNSTATS 活动所花的总时间”监视元素』

第 1252 页的『total_exec_time -“执行语句所耗用的时间”监视元素』

第 1291 页的『total_sort_time -“排序时间总计”监视元素』

第 1300 页的『total_sys_cpu_time -“语句的系统 CPU 时间总计”监视元素』

第 1302 页的『total_usr_cpu_time -“语句的用户 CPU 时间总计”监视元素』

dynsql_list 逻辑数据组

第 719 页的『db_name - 数据库名称监视元素』

第 720 页的『db_path -“数据库路径”』

fcm 逻辑数据组

第 643 页的『buff_free -“当前可用的 FCM 缓冲区数”』

第 644 页的『buff_free_bottom -“最少可用 FCM 缓冲区数”』

第 645 页的『buff_max -“FCM 缓冲区可能达到的最大数目”监视元素』

第 645 页的『buff_total -“当前已分配的 FCM 缓冲区数目”监视元素』

第 655 页的『ch_free -“当前可用的通道数”』

第 656 页的『ch_free_bottom -“最低可用通道数”』

第 656 页的『ch_max -“FCM 通道可能达到的最大数目”监视元素』

第 656 页的『ch_total -“当前已分配的 FCM 通道数”监视元素』

fcm_node 逻辑数据组

第 683 页的『connection_status -“连接状态”』

第 908 页的『node_number -“节点号”』

第 1237 页的『total_buffers_rcvd -“接收到的 FCM 缓冲区总数”』

第 1237 页的『total_buffers_sent -“发送的 FCM 缓冲区总数”』

hadr 逻辑数据组

第 797 页的『hadr_connect_status -“HADR 连接状态”监视元素』

第 798 页的『hadr_connect_time -“HADR 连接时间”监视元素』

第 798 页的『hadr_heartbeat -“HADR 脉动信号”监视元素』

第 799 页的『hadr_local_host -“HADR 本地主机”监视元素』

第 800 页的『hadr_local_service -“HADR 本地服务”监视元素』

第 800 页的『hadr_log_gap -“HADR 日志间隔”』

第 802 页的『hadr_primary_log_file -“HADR 主日志文件”监视元素』

第 802 页的『hadr_primary_log_lsn -“HADR 主日志 LSN”监视元素』

第 803 页的『hadr_primary_log_page -“HADR 主日志页”监视元素』

第 803 页的『hadr_remote_host -“HADR 远程主机”监视元素』

第 804 页的『hadr_remote_instance -“HADR 远程实例”监视元素』

第 804 页的『hadr_remote_service -“HADR 远程服务”监视元素』

第 805 页的『hadr_role -“HADR 角色”』

第 805 页的『hadr_standby_log_file -“HADR 备用日志文件”监视元素』

第 806 页的『hadr_standby_log_lsn -“HADR 备用日志 LSN”监视元素』

第 806 页的『hadr_standby_log_page -“HADR 备用日志页”监视元素』

第 807 页的『hadr_state -“HADR 状态”监视元素』

第 807 页的『hadr_syncmode -“HADR 同步方式”监视元素』

第 808 页的『hadr_timeout -“HADR 超时”监视元素』

lock 逻辑数据组

第 713 页的『data_partition_id -“数据分区标识”监视元素』

第 842 页的『lock_attributes -“锁定属性”监视元素』

第 843 页的『lock_count -“锁定计数”监视元素』

第 844 页的『lock_current_mode -“转换前的原始锁定方式”监视元素』

第 845 页的『lock_escalation -“锁定升级”监视元素』

第 851 页的『lock_hold_count -“锁定挂起计数”监视元素』

第 852 页的『lock_mode -“锁定方式”监视元素』

第 854 页的『lock_name -“锁定名称”监视元素』

第 855 页的『lock_object_name -“锁定对象名称”』

第 856 页的『lock_object_type -“等待的锁定对象类型”监视元素』

第 858 页的『lock_release_flags -“锁定释放标志”监视元素』

第 858 页的『lock_status -“锁定状态”监视元素』

第 908 页的『node_number -“节点号”』

第 1190 页的『table_file_id -“表文件标识”监视元素』

第 1191 页的『table_name -“表名”监视元素』

第 1193 页的『table_schema -“表模式名”监视元素』

第 1200 页的『tablespace_name -“表空间名称”监视元素』

lock_wait 逻辑数据组

第 602 页的『agent_id_holding_lock -“挂起锁定的代理程序标识”』

第 618 页的『appl_id_holding_lk -“挂起锁定的应用程序标识”』

第 713 页的『data_partition_id -“数据分区标识”监视元素』

第 842 页的『lock_attributes -“锁定属性”监视元素』

第 844 页的『lock_current_mode -“转换前的原始锁定方式”监视元素』

第 845 页的『lock_escalation -“锁定升级”监视元素』

第 852 页的『lock_mode -“锁定方式”监视元素』

第 853 页的『lock_mode_requested -“请求的锁定方式”监视元素』

第 854 页的『lock_name -“锁定名称”监视元素』

第 856 页的『lock_object_type -“等待的锁定对象类型”监视元素』

第 858 页的『lock_release_flags -“锁定释放标志”监视元素』

第 863 页的『lock_wait_start_time -“锁定等待开始时间戳记”监视元素』

第 908 页的『node_number -“节点号”』

第 1162 页的『ss_number -“子节号”监视元素』

第 1191 页的『table_name -“表名”监视元素』

第 1193 页的『table_schema -“表模式名”监视元素』

第 1200 页的『tablespace_name -“表空间名称”监视元素』

memory_pool 逻辑数据组

第 908 页的『node_number -“节点号”』

第 974 页的『pool_config_size -“内存池的已配置大小”』

第 975 页的『pool_cur_size -“内存池的当前大小”』

第 1004 页的『pool_id -“内存池标识”』

第 1045 页的『pool_secondary_id -“内存池辅助标记”』

第 1057 页的『pool_watermark -“内存池水位标记”』

progress 逻辑数据组

第 1088 页的『progress_completed_units -“完成的进度工作单元数”』

第 1088 页的『progress_description -“进度描述”』

第 1089 页的『progress_seq_num -“进度序号”』

第 1089 页的『progress_start_time -“进度开始时间”』

第 1090 页的『progress_total_units -“进度工作单元总数”』

第 1090 页的『progress_work_metric -“进度工作度量”』

progress_list 逻辑数据组

第 1088 页的『progress_list_attr -“当前进度列表属性”』

第 1089 页的『progress_list_cur_seq_num -“当前进度列表序号”』

rollforward 逻辑数据组

第 908 页的『node_number -“节点号”』

第 1111 页的『rf_log_num -“正在前滚的日志”监视元素』

第 1111 页的『rf_status -“日志阶段”』

第 1112 页的『rf_timestamp -“前滚时间戳记”』

第 1112 页的『rf_type -“前滚类型“』

第 1313 页的『ts_name -“正在前滚的表空间”监视元素』

stmt 逻辑数据组

第 609 页的『agents_top -“创建的代理程序数”』

第 640 页的『blocking_cursor -“分块游标”』

第 684 页的『consistency_token -“程序包一致性标记”监视元素』

第 709 页的『creator -“应用程序创建者”』

第 711 页的『cursor_name -“游标名称”』

第 730 页的『degree_parallelism -“并行度”』

第 789 页的『fetch_count -“成功的访存数”』

第 825 页的『int_rows_deleted -“删除的内部行数”』

第 826 页的『int_rows_inserted -“插入的内部行数”』

第 826 页的『int_rows_updated -“更新的内部行数”』

第 909 页的『num_agents -“正在处理语句的代理程序数”』

第 941 页的『package_name -“程序包名”监视元素』

第 943 页的『package_version_id -“程序包版本”监视元素』

第 982 页的『pool_data_l_reads -“缓冲池数据逻辑读取数”监视元素』

第 984 页的『pool_data_p_reads -“缓冲池数据物理读取数”监视元素』

第 1012 页的『pool_index_l_reads -“缓冲池索引逻辑读取数”监视元素』

第 1014 页的『pool_index_p_reads -“缓冲池索引物理读取数”监视元素』

第 1046 页的『pool_temp_data_l_reads -“缓冲池临时数据逻辑读取数”监视元素』

第 1048 页的『pool_temp_data_p_reads -“缓冲池临时数据物理读取数”监视元素』

第 1050 页的『pool_temp_index_l_reads -“缓冲池临时索引逻辑读取数”监视元素』

第 1052 页的『pool_temp_index_p_reads -“缓冲池临时索引物理读取数”监视元素』

第 1053 页的『pool_temp_xda_l_reads -“缓冲池临时 XDA 数据逻辑读取数”监视元素』

第 1055 页的『pool_temp_xda_p_reads -“缓冲池临时 XDA 数据物理读取数”监视元素』

第 1065 页的『pool_xda_l_reads -“缓冲池 XDA 数据逻辑读取数”监视元素』

第 1069 页的『pool_xda_p_reads -“缓冲池 XDA 数据物理读取数”监视元素』

第 1091 页的『query_card_estimate -“行查询数估计”』

第 1092 页的『query_cost_estimate -“查询估算成本”监视元素』

第 1118 页的『rows_read -“读取行数”监视元素』

第 1123 页的『rows_written -“写入的行数”』

第 1128 页的『section_number -“节号”监视元素』

第 1149 页的『sort_overflows -“排序溢出数”监视元素』

第 1168 页的『stmt_elapsed_time -“最新语句耗用时间”』

第 1173 页的『stmt_node_number -“语句节点”』

第 1173 页的『stmt_operation/operation -“语句操作”监视元素』

第 1176 页的『stmt_sorts -“语句排序数”』

第 1177 页的『stmt_start -“语句操作开始时间戳记”』

第 1177 页的『stmt_stop -“语句操作停止时间戳记”』

第 1177 页的『stmt_sys_cpu_time -“语句使用的系统 CPU 时间”』

第 1178 页的『stmt_text -“SQL 语句文本”监视元素』

第 1179 页的『stmt_type -“语句类型”监视元素』

第 1181 页的『stmt_usr_cpu_time -“语句使用的用户 CPU 时间”』

第 1291 页的『total_sort_time -“排序时间总计”监视元素』

stmt_transmissions 逻辑数据组

第 751 页的『elapsed_exec_time -“语句执行耗用时间”』

第 813 页的『host_response_time -“主机响应时间”』

第 884 页的『max_data_received_1024 -“接收的出站字节数在 513 到 1024 字节之间的语句数”』

第 885 页的『max_data_received_128 -“接收的出站字节数在 1 到 128 字节之间的语句数”』

第 885 页的『max_data_received_16384 -“接收的出站字节数在 8193 到 16384 字节之间的语句数”』

第 886 页的『max_data_received_2048 -“接收的出站字节数在 1025 到 2048 字节之间的语句数”』

第 886 页的『max_data_received_256 -“接收的出站字节数在 129 到 256 字节之间的语句数”』

第 887 页的『max_data_received_31999 -“接收的出站字节数在 16385 到 31999 字节之间的语句数”监视元素』

第 887 页的『max_data_received_4096 -“接收的出站字节数在 2049 到 4096 字节之间的语句数”』

第 887 页的『max_data_received_512 -“接收的出站字节数在 257 到 512 字节之间的语句数”』

第 888 页的『max_data_received_64000 -“接收的出站字节数在 32000 到 64000 字节之间的语句数”监视元素』

第 888 页的『max_data_received_8192 -“接收的出站字节数在 4097 到 8192 字节之间的语句”』

第 889 页的『max_data_received_gt64000 -“接收的出站字节数高于 64000 的语句数”』

第 889 页的『max_data_sent_1024 -“发送的出站字节数在 513 到 1024 字节之间的语句数”』

第 889 页的『max_data_sent_128 -“发送的出站字节数在 1 到 128 字节之间的语句数”』

第 890 页的『max_data_sent_16384 -“发送的出站字节数在 8193 到 16384 字节之间的语句数”』

第 890 页的『max_data_sent_2048 -“发送的出站字节数在 1025 到 2048 字节之间的语句数”』

第 891 页的『max_data_sent_256 -“发送的出站字节数在 129 到 256 字节之间的语句数”』

第 891 页的『max_data_sent_31999 -“发送的出站字节数在 16385 到 31999 字节之间的语句数”』

第 892 页的『max_data_sent_4096 -“发送的出站字节数在 2049 到 4096 字节之间的语句数”』

第 892 页的『max_data_sent_512 -“发送的出站字节数在 257 到 512 字节之间的语句数”』

第 892 页的『max_data_sent_64000 -“发送的出站字节数在 32000 到 64000 字节之间的语句数”』

第 893 页的『max_data_sent_8192 -“发送的出站字节数在 4097 到 8192 字节之间的语句数”』

第 893 页的『max_data_sent_gt64000 -“发送的出站字节数高于 64000 的语句数”』

第 894 页的『max_network_time_100_ms -“网络时间在 16 到 100 毫秒之间的语句数”』

第 894 页的『max_network_time_16_ms -“网络时间在 4 到 16 毫秒之间的语句数”』

第 895 页的『max_network_time_1_ms -“网络时间最多为 1 毫秒的语句数”』

第 895 页的『max_network_time_4_ms -“网络时间在 1 到 4 毫秒之间的语句数”』

第 895 页的『max_network_time_500_ms -“网络时间在 100 到 500 毫秒之间的语句数”』

第 896 页的『max_network_time_gt500_ms -“网络时间大于 500 毫秒的语句数”』

第 906 页的『network_time_bottom -“语句的最短网络时间”』

第 907 页的『network_time_top -“语句的最长网络时间”』

第 937 页的『outbound_bytes_received -“接收的出站字节数”』

第 937 页的『outbound_bytes_received_bottom -“接收的最小出站字节数”』

第 938 页的『outbound_bytes_received_top -“接收的最大出站字节数”』

第 938 页的『outbound_bytes_sent -“发送的出站字节数”』

第 938 页的『outbound_bytes_sent_bottom -“发送的最小出站字节数”』

第 938 页的『outbound_bytes_sent_top -“发送的最大出站字节数”』

第 1152 页的『sql_chains -“尝试的 SQL 链数”』

第 1153 页的『sql_stmts -“尝试的 SQL 语句数”』

subsection 逻辑数据组

第 1118 页的『rows_read -“读取行数”监视元素』

第 1123 页的『rows_written -“写入的行数”』

第 1161 页的『ss_exec_time -“子节执行耗用时间”』

第 1161 页的『ss_node_number -“子节节点号”』

第 1162 页的『ss_number -“子节号”监视元素』

第 1162 页的『ss_status -“子节状态”监视元素』

第 1162 页的『ss_sys_cpu_time -“子节使用的系统 CPU 时间”』

第 1163 页的『ss_usr_cpu_time -“子节使用的用户 CPU 时间”』

第 1305 页的『tq_cur_send_spills -“当前溢出的表队列缓冲区数”监视元素』

第 1306 页的『tq_id_waiting_on -“在表队列的节点上等待”监视元素』

第 1306 页的『tq_max_send_spills -“最大表队列缓冲区溢出数”』

第 1306 页的『tq_node_waited_for -“在表队列上等待节点”』

table 逻辑数据组

table_list 逻辑数据组

table_reorg 逻辑数据组

第 1108 页的『reorg_start -“表重组开始时间”』

第 1108 页的『reorg_status -“表重组状态”』

第 1108 页的『reorg_tbspc_id -“用来重组表或数据分区的表空间”』

第 1109 页的『reorg_type -“表重组属性”』

第 1109 页的『reorg_xml_regions_compressed -“已压缩的 XML 区域数”监视元素』

第 1110 页的『reorg_xml_regions_rejected_for_compression -“拒绝压缩的 XML 区域数”监视元素』

tablespace 逻辑数据组

第 735 页的『direct_read_reqs -“直接读请求数”监视元素』

第 737 页的『direct_read_time -“直接读时间”监视元素』

第 738 页的『direct_reads -“直接读数据库数目”监视元素』

第 740 页的『direct_write_reqs -“直接写请求数”监视元素』

第 742 页的『direct_write_time -“直接写时间”监视元素』

第 744 页的『direct_writes -“直接写数据库数目”监视元素』

第 790 页的『files_closed -“关闭数据库文件数”监视元素』

第 792 页的『fs_caching -“文件系统高速缓存”监视元素』

第 961 页的『pool_async_data_read_reqs -“缓冲池异步读请求数”监视元素』

第 962 页的『pool_async_data_reads -“缓冲池异步数据读次数”监视元素』

第 963 页的『pool_async_data_writes -“缓冲池异步数据写次数”监视元素』

第 966 页的『pool_async_index_read_reqs -“缓冲池异步索引读请求数”监视元素』

第 966 页的『pool_async_index_reads -“缓冲池异步索引读取数”监视元素』

第 967 页的『pool_async_index_writes -“缓冲池异步索引写次数”监视元素』

第 968 页的『pool_async_read_time -“缓冲池异步读取时间”』

第 969 页的『pool_async_write_time -“缓冲池异步写入时间”监视元素』

第 972 页的『pool_async_xda_read_reqs -“缓冲池异步 XDA 读请求数”监视元素』

第 973 页的『pool_async_xda_reads -“缓冲池异步 XDA 数据读取数”监视元素』

第 974 页的『pool_async_xda_writes -“缓冲池异步 XDA 数据写次数”监视元素』

第 982 页的『pool_data_l_reads -“缓冲池数据逻辑读取数”监视元素』

第 984 页的『pool_data_p_reads -“缓冲池数据物理读取数”监视元素』

第 986 页的『pool_data_writes -“缓冲池数据写次数”监视元素』

第 1012 页的『pool_index_l_reads -“缓冲池索引逻辑读取数”监视元素』

第 1014 页的『pool_index_p_reads -“缓冲池索引物理读取数”监视元素』

第 1016 页的『pool_index_writes -“缓冲池索引写次数”监视元素』

第 1018 页的『pool_no_victim_buffer -“缓冲池无牺牲缓冲区次数”监视元素』

第 1043 页的『pool_read_time -“缓冲池物理读时间总计”监视元素』

第 1046 页的『pool_temp_data_l_reads -“缓冲池临时数据逻辑读取数”监视元素』

第 1048 页的『pool_temp_data_p_reads -“缓冲池临时数据物理读取数”监视元素』

第 1050 页的『pool_temp_index_l_reads -“缓冲池临时索引逻辑读取数”监视元素』

第 1052 页的『pool_temp_index_p_reads -“缓冲池临时索引物理读取数”监视元素』

第 1053 页的『pool_temp_xda_l_reads -“缓冲池临时 XDA 数据逻辑读取数”监视元素』

第 1055 页的『pool_temp_xda_p_reads -“缓冲池临时 XDA 数据物理读取数”监视元素』

第 1058 页的『pool_write_time -“缓冲池物理写时间总计”监视元素』

第 1065 页的『pool_xda_l_reads -“缓冲池 XDA 数据逻辑读取数”监视元素』

第 1069 页的『pool_xda_p_reads -“缓冲池 XDA 数据物理读取数”监视元素』

第 1071 页的『pool_xda_writes -“缓冲池 XDA 数据写次数”监视元素』

第 1195 页的『tablespace_auto_resize_enabled -“允许自动调整表空间大小”监视元素』

第 1195 页的『tablespace_content_type -“表空间内容类型”监视元素』

第 1196 页的『tablespace_cur_pool_id -“当前使用的缓冲池”监视元素』

第 1197 页的『tablespace_extent_size -“表空间扩展数据块大小”监视元素』

第 1197 页的『tablespace_id -“表空间标识”监视元素』

第 1200 页的『tablespace_name -“表空间名称”监视元素』

第 1201 页的『tablespace_next_pool_id -“下次启动时使用的缓冲池”监视元素』

第 1202 页的『tablespace_page_size -“表空间页大小”监视元素』

第 1204 页的『tablespace_prefetch_size -“表空间预取大小”监视元素』

第 1206 页的『tablespace_rebalancer_mode -“重新平衡程序方式”监视元素』

第 1213 页的『tablespace_type -“表空间类型”监视元素』

第 1214 页的『tablespace_using_auto_storage -“已对表空间启用自动存储器”监视元素』

tablespace_container 逻辑数据组

第 685 页的『container_accessible -“容器可访问”监视元素』

第 685 页的『container_id -“容器标识”监视元素』

第 685 页的『container_name -“容器名称”监视元素』

第 686 页的『container_stripe_set -“容器分割集”监视元素』

第 686 页的『container_total_pages -“容器中的总页数”监视元素』

第 687 页的『container_type -“容器类型”监视元素』

第 687 页的『container_usable_pages -“容器中的可用页数”监视元素』

tablespace_list 逻辑数据组

第 717 页的『db_conn_time -“数据库激活时间戳记”监视元素』

第 719 页的『db_name - 数据库名称监视元素』

第 720 页的『db_path -“数据库路径”』

第 819 页的『input_db_alias -“输入数据库别名”』

第 837 页的『last_reset -“最后重置时间戳记”』

tablespace_nodeinfo 逻辑数据组

第 1196 页的『tablespace_current_size -“当前表空间大小”』

第 1197 页的『tablespace_free_pages -“表空间中的空闲页数”监视元素』

第 1198 页的『tablespace_increase_size -“增加字节大小”』

第 1198 页的『tablespace_increase_size_percent -“增加大小（以百分比计）”监视元素』

第 1199 页的『tablespace_initial_size -“初始表空间大小”』

tablespace_quiescer 逻辑数据组

tablespace_range 逻辑数据组

第 1098 页的『range_number -“范围编号”』

第 1098 页的『range_offset -“范围偏移”』

第 1098 页的『range_start_stripe -“起始分割区”』

第 1098 页的『range_stripe_set_number -“分割集编号”』

utility_info 逻辑数据组

第 908 页的『node_number -“节点号”』

第 1326 页的『utility_dbname -“实用程序操作的数据库”』

第 1327 页的『utility_description -“实用程序描述”』

第 1327 页的『utility_id -“实用程序标识”』

第 1328 页的『utility_invoker_type -“实用程序调用者类型”』

第 1331 页的『utility_priority -“实用程序优先级”』

第 1331 页的『utility_start_time -“实用程序启动时间”』

第 1331 页的『utility_state -“实用程序状态”』

第 1333 页的『utility_type -“实用程序类型”』

第 11 章 监视元素参考

对监视元素所收集数据的描述。

系统监视器返回的监视元素分为下列几个类别:

- **标识**，用于要监视的数据库管理器、应用程序或数据库连接。
- 这些数据主要用于帮助您**配置**系统。
- 各个级别的数据库**活动**，包括数据库、应用程序、表或语句。此信息可用于活动监视、问题确定和性能分析。此信息还可用于配置。
- 有关 **DB2 Connect** 应用程序的信息。包括在网关上运行的 DCS 应用程序、要执行的 SQL 语句和数据库连接的信息。
- 有关**联合数据库系统**的信息。这包括有关由在 DB2 联合系统中运行的应用程序对数据源的总体访问的信息，以及由在联合服务器实例中运行的给定应用程序对数据源的访问的信息。

监视元素是以如下标准格式描述的:

元素标识

元素的名称。如果直接对数据流进行语法分析，那么元素标识将是大写的，并且加上 SQLM_ELM_ 前缀。

元素类型

监视元素返回的信息类型。例如，db2start_time 监视元素返回时间戳记。

表函数监视信息

如果监视元素是由表函数返回，那么将显示带有下列字段的表。

- *表函数*: 返回监视元素的表函数的名称。
- *监视元素收集级别*: 有关监视元素收集级别的更多信息，请参阅 第 485 页的第 8 章，『监视元素收集级别』。

快照监视信息

如果监视元素返回快照监视信息，将显示带有下列字段的表。

- *快照级别*: 快照监视器可捕获的信息的级别。例如，appl_status 监视元素返回应用程序级别和锁定级别的信息。
- *逻辑数据分组*: 返回捕获的快照信息的逻辑数据组。如果直接对数据流进行语法分析，那么逻辑数据组标识将是大写的，并且加上 SQLM_ELM_ 前缀。例如，appl_status 监视元素返回有关 appl_id_info 分组和 appl_lock_list 分组的信息。
- *监视开关*: 为获取此信息而必须设置的系统监视开关。如果开关为“基本”，那么总是收集有关该监视元素的数据。

事件监视信息

如果监视元素是由事件监视器收集的，将显示带有下列字段的表。

- *事件类型*: 事件监视器可收集的信息的级别。必须使用此事件类型来创建事件监视器以收集此信息。例如，将为 CONNECTIONS 事件监视器收集 appl_status 监视元素。

- *逻辑数据分组*: 在其中返回捕获的事件信息的逻辑数据组。如果直接对数据流进行语法分析，那么逻辑数据组标识将是大写的，并且加上 SQLM_ELM_ 前缀。例如，appl_status 监视元素返回有关 event_conn 分组的信息。
- *监视开关*: 为获取此信息而必须设置的系统监视开关。对于事件监视器，TIMESTAMP 开关是唯一能够限制事件数据收集的监视开关。如果此字段显示为虚线，那么总是收集有关该监视元素的数据。

用法 有关在监视数据库系统时能够如何使用监视元素收集的信息的信息。

acc_curs_blk -"接受的块游标请求数"

接受 I/O 块请求的次数。

元素标识

acc_curs_blk

元素类型

计数器

表 150. 快照监视信息

快照级别	逻辑数据分组	监视开关
应用程序	appl	基本

表 151. 事件监视信息

事件类型	逻辑数据分组	监视开关
连接	event_conn	始终收集

用法 可将此元素与 *rej_curs_blk* 一起使用来计算接受的和/或拒绝的分块请求百分比。

有关如何使用此信息来调整配置参数的建议，请参阅 *rej_curs_blk*。

act_aborted_total -"异常终止活动总数"监视元素

在任何嵌套级别以出错情况完成的协调程序活动的总数。对于服务类而言，如果在活动异常终止前通过 REMAP ACTIVITY 操作将其重新映射到另一个服务子类，那么此活动将仅计入在其中异常终止此活动的子类的总数。

表 152. 表函数监视信息

表函数	监视元素收集级别
MON_FORMAT_XML_METRICS_BY_ROW - 获取所有度量值的基于已格式化行的输出	不适用; 报告 XML 文档中作为格式化函数的输入提供的所有元素
MON_GET_CONNECTION 表函数 - 获取连接度量值	REQUEST METRICS BASE
MON_GET_CONNECTION_DETAILS 表函数 - 获取详细的连接度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE
MON_GET_SERVICE_SUBCLASS 表函数 - 获取服务子类度量值	REQUEST METRICS BASE

表 152. 表函数监视信息 (续)

表函数	监视元素收集级别
MON_GET_SERVICE_SUBCLASS_DETAILS 表函数 - 获取详细的服务子类度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE
MON_GET_UNIT_OF_WORK 表函数 - 获取工作单元度量值	REQUEST METRICS BASE
MON_GET_UNIT_OF_WORK 表函数 - 获取详细的工作单元度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE
MON_GET_WORKLOAD 表函数 - 获取工作负载度量值	REQUEST METRICS BASE
MON_GET_WORKLOAD_DETAILS 表函数 - 获取详细的工作负载度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE

表 153. 事件监视信息

事件类型	逻辑数据分组	监视开关
统计信息	event_scstats（在 details_xml 文档中报告）	REQUEST METRICS BASE
统计信息	event_wlstats（在 details_xml 文档中报告）	REQUEST METRICS BASE
工作单元	在 system_metrics 文档中报告。	-

用法

使用此元素来了解系统上的活动是否成功完成。活动可能因取消、错误或反应性阈值而异常终止。

act_completed_total -“完成活动总数”监视元素

在任何嵌套级别成功完成的协调程序活动的总数。对于服务类而言，如果在活动完成前通过 REMAP ACTIVITY 操作将其重新映射到另一个子类，那么此活动将仅计入在其中完成此活动的子类的总数。

表 154. 表函数监视信息

表函数	监视元素收集级别
MON_FORMAT_XML_METRICS_BY_ROW - 获取所有度量值的基于已格式化行的输出	不适用；报告 XML 文档中作为格式化函数的输入提供的所有元素
MON_GET_CONNECTION 表函数 - 获取连接度量值	REQUEST METRICS BASE
MON_GET_CONNECTION_DETAILS 表函数 - 获取详细的连接度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE
MON_GET_SERVICE_SUBCLASS 表函数 - 获取服务子类度量值	REQUEST METRICS BASE

表 154. 表函数监视信息 (续)

表函数	监视元素收集级别
MON_GET_SERVICE_SUBCLASS_DETAILS 表函数 - 获取详细的服务子类度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE
MON_GET_UNIT_OF_WORK 表函数 - 获取工作单元度量值	REQUEST METRICS BASE
MON_GET_UNIT_OF_WORK 表函数 - 获取详细的工作单元度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE
MON_GET_WORKLOAD 表函数 - 获取工作负载度量值	REQUEST METRICS BASE
MON_GET_WORKLOAD_DETAILS 表函数 - 获取详细的工作负载度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE
MON_SAMPLE_SERVICE_CLASS_METRICS - 获取样本服务类度量值	REQUEST METRICS BASE
MON_SAMPLE_WORKLOAD_METRICS - 获取样本	REQUEST METRICS BASE

表 155. 事件监视信息

事件类型	逻辑数据分组	监视开关
统计信息	event_scstats（在 details_xml 文档中报告）	REQUEST METRICS BASE
统计信息	event_wlstats（在 details_xml 文档中报告）	REQUEST METRICS BASE
工作单元	在 system_metrics 文档中报告。	-

用法

使用此元素来确定系统中的活动的吞吐量。

act_cpu_time_top -“最长活动 CPU 时间”监视元素

服务类、工作负载或工作类中所有嵌套级别的活动所使用的处理器时间的高水位标记。此值是按微秒报告的。

当此活动运行所在的服务类或工作负载的 COLLECT AGGREGATE ACTIVITY DATA 设置为 NONE 时，此监视元素将返回 -1。仅当启用了请求度量值时，活动才会影响此高水位标记。如果未启用活动度量值收集，那么会返回值 0。

对于服务类而言，使用 REMAP ACTIVITY 操作在服务子类之间重新映射活动时，将仅更新完成该活动的服务子类的 act_cpu_time_top 高水位标记（如果达到新的高水位标记的话）。该活动所映射到但未在其中完成该活动的其他服务子类的 act_cpu_time_top 高水位标记不受影响。

表 156. 事件监视信息

事件类型	逻辑数据分组	监视开关
统计信息	event_scstats	始终收集
统计信息	event_wcstats	始终收集
统计信息	event_wlstats	始终收集

用法

使用此元素来确定在收集时间间隔内，成员在服务类、工作负载或工作类上的活动所使用的最大处理器时间量。

act_exec_time -“活动执行时间”监视元素

在此成员执行所耗的时间（以毫秒计）。对于游标来说，执行时间是打开、访存和关闭的综合时间。游标闲置的时间不会计入执行时间。对于例程来说，执行时间是指例程调用开始到结束这段时间。在例程结束之后由例程将其保持为打开状态以返回结果集的任何游标的生存期不会计入例程执行时间。对于其他所有活动来说，执行时间为开始时间与停止时间之间的时间差。在所有情况下，执行时间均不包括初始化或排队所花的时间。

表 157. 事件监视信息

事件类型	逻辑数据分组	监视开关
活动	event_activity	始终收集

用法

可单独使用此元素来了解 DB2 在每个成员上执行活动所消耗的时间。还可以将此元素与协调程序成员上的 **time_started** 和 **time_completed** 监视元素配合使用来计算游标活动的空闲时间。您可以使用以下公式:

```
Cursor idle time = (time_completed - time_started) - act_exec_time
```

act_rejected_total -“被拒绝活动总数”监视元素

在任何嵌套级别由于被拒绝而未被允许执行的协调程序活动的总数。

表 158. 表函数监视信息

表函数	监视元素收集级别
MON_FORMAT_XML_METRICS_BY_ROW - 获取所有度量值的基于已格式化行的输出	不适用; 报告 XML 文档中作为格式化函数的输入提供的所有元素
MON_GET_CONNECTION 表函数 - 获取连接度量值	REQUEST METRICS BASE
MON_GET_CONNECTION_DETAILS 表函数 - 获取详细的连接度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE
MON_GET_SERVICE_SUBCLASS 表函数 - 获取服务子类度量值	REQUEST METRICS BASE

表 158. 表函数监视信息 (续)

表函数	监视元素收集级别
MON_GET_SERVICE_SUBCLASS_DETAILS 表函数 - 获取详细的服务子类度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE
MON_GET_UNIT_OF_WORK 表函数 - 获取工作单元度量值	REQUEST METRICS BASE
MON_GET_UNIT_OF_WORK 表函数 - 获取详细的工作单元度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE
MON_GET_WORKLOAD 表函数 - 获取工作负载度量值	REQUEST METRICS BASE
MON_GET_WORKLOAD_DETAILS 表函数 - 获取详细的工作负载度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE

表 159. 事件监视信息

事件类型	逻辑数据分组	监视开关
统计信息	event_scstats（在 details_xml 文档中报告）	REQUEST METRICS BASE
统计信息	event_wlstats（在 details_xml 文档中报告）	REQUEST METRICS BASE
工作单元	在 system_metrics 文档中报告。	-

用法

使用此元素来帮助确定导致无法执行活动的预测性阈值或工作操作是否有效以及它们的限制是否过于严格。

act_remapped_in -“重新映入的活动数”监视元素

自从上次重置之后重新映射到此服务子类的活动数。

表 160. 事件监视信息

事件类型	逻辑数据分组	监视开关
统计信息	event_scstats	-

用法

使用此计数来确定是否已按预期将活动重新映射到此服务子类。

act_remapped_out -“重新映出的活动数”监视元素

自从上次重置之后从此服务子类中向外重新映射的活动数。

表 161. 事件监视信息

事件类型	逻辑数据分组	监视开关
统计信息	event_scstats	-

用法

使用此计数来确定是否已按预期从此服务子类中向外重新映射活动。

act_rows_read_top -“最大活动读取行数”监视元素

服务类、工作负载或工作类中所有嵌套级别的活动所读取的行数的高水位标记。

当此活动运行所在的服务类或工作负载的 COLLECT AGGREGATE ACTIVITY DATA 设置为 NONE 时，此监视元素将返回 -1。仅当启用了请求度量值时，活动才会影响此高水位标记。如果未启用活动度量值收集，那么会返回值 0。

对于服务类而言，使用 REMAP ACTIVITY 操作在服务子类之间重新映射活动时，将仅更新完成该活动的服务子类的 act_rows_read_top 高水位标记（如果达到新的高水位标记的话）。该活动所映射到但未在其中完成该活动的服务子类的 act_rows_read_top 高水位标记不受影响。

表 162. 事件监视信息

事件类型	逻辑数据分组	监视开关
统计信息	event_scstats	始终收集
统计信息	event_wcstats	始终收集
统计信息	event_wlstats	始终收集

用法

使用此元素来确定在收集时间间隔内，成员在服务类、工作负载或工作类上的活动所读取的最大行数。

act_rqsts_total -“活动请求总数”监视元素

在活动中完成的各个协调程序和子代理程序请求的数目。例如，访存游标活动。

表 163. 表函数监视信息

表函数	监视元素收集级别
MON_FORMAT_XML_METRICS_BY_ROW - 获取所有度量值的基于已格式化行的输出	不适用；报告 XML 文档中作为格式化函数的输入提供的所有元素
MON_GET_CONNECTION 表函数 - 获取连接度量值	REQUEST METRICS BASE
MON_GET_CONNECTION_DETAILS 表函数 - 获取详细的连接度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE

表 163. 表函数监视信息 (续)

表函数	监视元素收集级别
MON_GET_SERVICE_SUBCLASS 表函数 - 获取服务子类度量值	REQUEST METRICS BASE
MON_GET_SERVICE_SUBCLASS_DETAILS 表函数 - 获取详细的服务子类度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE
MON_GET_UNIT_OF_WORK 表函数 - 获取工作单元度量值	REQUEST METRICS BASE
MON_GET_UNIT_OF_WORK 表函数 - 获取详细的工作单元度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE
MON_GET_WORKLOAD 表函数 - 获取工作负载度量值	REQUEST METRICS BASE
MON_GET_WORKLOAD_DETAILS 表函数 - 获取详细的工作负载度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE

表 164. 事件监视信息

事件类型	逻辑数据分组	监视开关
统计信息	event_scstats（在 details_xml 文档中报告）	REQUEST METRICS BASE
统计信息	event_wlstats（在 details_xml 文档中报告）	REQUEST METRICS BASE
工作单元	在 system_metrics 文档中报告。	REQUEST METRICS BASE

act_throughput -“活动吞吐量”监视元素

协调程序活动在任何嵌套级别完成的速率。此监视元素以每秒协调程序活动数计。

表 165. 表函数监视信息

表函数	监视元素收集级别
MON_SAMPLE_SERVICE_CLASS_METRICS - 获取样本服务类度量值	REQUEST METRICS BASE
MON_SAMPLE_WORKLOAD_METRICS - 获取样本工作负载度量值	REQUEST METRICS BASE
WLM_GET_SERVICE_SUBCLASS_STATS 表函数 - 返回服务子类的统计信息	ACTIVITY METRICS BASE
WLM_GET_WORKLOAD_STATS 表函数 - 返回工作负载统计信息	ACTIVITY METRICS BASE

表 166. 事件监视信息

事件类型	逻辑数据分组	监视开关
统计信息	event_scstats（在 details_xml 文档中报告）	始终收集

表 166. 事件监视信息 (续)

事件类型	逻辑数据分组	监视开关
统计信息	event_wlstats（在 details_xml 文档中报告）	始终收集

用法

如果由 WLM_GET_SERVICE_SUBCLASS_STATS 或 WLM_GET_WORKLOAD_STATS 函数返回此监视元素，那么此监视元素表示自上次重置统计信息以来的活动吞吐量。

如果由 MON_SAMPLE_SERVICE_CLASS_METRICS 或 MON_SAMPLE_WORKLOAD_METRICS 函数返回此监视元素，那么此监视元素表示自执行该函数以来的活动吞吐量。

act_total -“活动总数”监视元素

自最后一次重置以后在任何嵌套级别处应用与指定工作类相对应的工作操作的活动总数。

表 167. 表函数监视信息

表函数	监视元素收集级别
WLM_GET_WORK_ACTION_SET_STATS 表函数 - 返回工作操作集统计信息	ACTIVITY METRICS BASE

表 168. 事件监视信息

事件类型	逻辑数据分组	监视开关
统计信息	event_wcstats	-

用法

每次活动应用一个或多个与工作类相关联的工作操作时，均会更新工作类的计数器。此计数器通过使用 **act_total** 监视元素来展示。计数器可用于判断工作操作集的效用（例如，判断有多少活动已应用操作）。另外，它还可用于了解系统上不同类别的活动。

activate_timestamp -“激活时间戳记”监视元素

激活事件监视器的时间。

表 169. 事件监视信息

事件类型	逻辑数据分组	监视开关
活动	event_activity	-
活动	event_activitystmt	-
活动	event_activityvals	-
阈值违例	event_thresholdviolations	-

用法

使用此元素以使上述事件类型返回的信息相关。

active_hash_joins -“活动散列连接数”

当前正在运行并消耗内存的散列连接的总数。

表 170. 快照监视信息

快照级别	逻辑数据分组	监视开关
数据库	dbase	-

active_olap_funcs -“活动 OLAP 函数”监视元素

当前正在运行并消耗排序堆内存的 OLAP 函数总数。

表 171. 快照监视信息

快照级别	逻辑数据分组	监视开关
数据库	dbase	-

可将快照监视的计数器重置。

active_sorts -“活动排序次数”

数据库中当前分配了排序堆的排序数。

表 172. 快照监视信息

快照级别	逻辑数据分组	监视开关
数据库	dbase	基本

用法 将此值与 *sort_heap_allocated* 一起使用来确定每个排序使用的平均排序堆空间。如果 *sortheap* 配置参数实际上大于使用的平均排序堆，那么您可以降低此参数的值。

此值包括用于相关操作期间创建的临时表的排序堆。

activity_collected -“收集的活动”监视元素

此元素指示是否对违例的阈值收集活动事件监视记录。

表 173. 事件监视信息

事件类型	逻辑数据分组	监视开关
阈值违例	event_thresholdviolations	-

用法

使用此元素来确定是否要将违反阈值的活动的活动事件写入活动事件监视器。

当活动完成或异常终止且活动事件监视器处于活动状态时，如果此监视元素的值为“Y”，那么将收集违反此阈值的活动。如果此监视元素的值为“N”，那么将不收集。

activity_id -“活动标识”监视元素

用于唯一地标识给定工作单元中某个应用程序的活动的计数器。

表 174. 表函数监视信息

表函数	监视元素收集级别
MON_GET_ACTIVITY_DETAILS 表函数 - 获取完整的活动详细信息（在 XML 文档 DETAILS 中报告）	ACTIVITY METRICS BASE
WLM_GET_SERVICE_CLASS_AGENTS 表函数 - 列示正在服务类中运行的代理程序	ACTIVITY METRICS BASE
WLM_GET_WORKLOAD_OCCURRENCE_ACTIVITIES 表函数 - 返回活动列表	ACTIVITY METRICS BASE

表 175. 事件监视信息

事件类型	逻辑数据分组	监视开关
锁定	-	始终收集
活动	event_activity	始终收集
活动	event_activitystmt	始终收集
活动	event_activityvals	始终收集
活动	event_activitymetrics	ACTIVITY METRICS BASE
阈值违例	event_thresholdviolations	始终收集

用法

将此元素与其他活动历史元素配合使用来分析活动的行为。

要在活动的工作单元外部唯一地标识该活动，请将 **activity_id** 和 **uow_id** 的组合与下列其中一个监视元素配合使用：**appl_id** 或 **agent_id**。

activity_secondary_id -“活动辅助标识”监视元素

此元素的值在每次对同一活动写入活动记录时增加。例如，如果调用 WLM_CAPTURE_ACTIVITY_IN_PROGRESS 过程后写入活动记录一次，活动结束时再写入一次，那么第一个记录的元素的值为 0，第二个记录的元素值为 1。

表 176. 事件监视信息

事件类型	逻辑数据分组	监视开关
活动	event_activity	-
活动	event_activitystmt	-
活动	event_activityvals	-
活动	event_activitymetrics	ACTIVITY METRICS BASE

用法

当有关相同活动的信息被多次写入活动事件监视器时，将此元素与 **activity_id**、**uow_id** 和 **appl_id** 监视元素配合使用，以唯一地标识活动记录。

例如，在以下情况下，会将有关活动的信息发送至活动事件监视器两次:

- 在活动运行时，使用 WLM_CAPTURE_ACTIVITY_IN_PROGRESS 存储过程来捕获有关活动的信息
- 因为对与活动关联的服务类指定了 COLLECT ACTIVITY DATA 子句，所以在完成活动时收集有关活动的信息

activity_state -“活动状态”监视元素

活动的当前状态。

表 177. 表函数监视信息

表函数	监视元素收集命令和级别
MON_GET_ACTIVITY_DETAILS 表函数 - 获取完整的活动详细信息（在 XML 文档 DETAILS 中报告）	ACTIVITY METRICS BASE
WLM_GET_WORKLOAD_OCCURRENCE_ACTIVITIES 表函数 - 返回活动列表	ACTIVITY METRICS BASE

用法

使用此监视元素来确定此活动当前正在执行的操作（例如，此活动是停滞在队列中还是正在等待来自客户机的输入）。可能的值包括:

- CANCEL_PENDING
- EXECUTING
- IDLE
- INITIALIZING
- QP_CANCEL_PENDING
- QP_QUEUED
- QUEUED
- TERMINATING
- UNKNOWN

activity_type -“活动类型”监视元素

活动的类型。

表 178. 表函数监视信息

表函数	监视元素收集级别
MON_GET_ACTIVITY_DETAILS 表函数 - 获取完整的活动详细信息（在 XML 文档 DETAILS 中报告）	ACTIVITY METRICS BASE

表 179. 事件监视信息

事件类型	逻辑数据分组	监视开关
活动	event_activity	始终收集

用法

可能的值包括:

- LOAD
- READ_DML
- WRITE_DML
- DDL
- CALL
- OTHER

对于不执行 SQL 的 SET 语句（例如，SET 专用寄存器或 SET EVENT MONITOR STATE）和 LOCK TABLE 语句，将返回值 OTHER。

activitytotaltime_threshold_id -“活动时间总计阈值标识”监视元素

应用于此活动的 ACTIVITYTOTALTIME 阈值的标识。

表 180. 表函数监视信息

表函数	监视元素收集命令和级别
MON_GET_ACTIVITY_DETAILS 表函数 - 获取完整的活动详细信息（在 XML 文档 DETAILS 中报告）	ACTIVITY METRICS BASE

用法

使用此元素来确定应用于此活动的 ACTIVITYTOTALTIME 阈值（如果有的话）。

activitytotaltime_threshold_value -“活动时间总计阈值”监视元素

通过将 ACTIVITYTOTALTIME 阈值持续时间与活动进入时间相加计算而得的时间戳记。达到此时间戳记时，如果此活动仍在执行，那么将违反该阈值。

表 181. 表函数监视信息

表函数	监视元素收集命令和级别
MON_GET_ACTIVITY_DETAILS 表函数 - 获取完整的活动详细信息（在 XML 文档 DETAILS 中报告）	ACTIVITY METRICS BASE

用法

使用此元素来确定应用于此活动的 ACTIVITYTOTALTIME 阈值（如果有的话）。

activitytotaltime_threshold_violated -“违反活动时间总计阈值”监视元素

此监视元素返回“Yes”表明此活动已违反 ACTIVITYTOTALTIME 阈值。“No”表明此活动尚未违反该阈值。

表 182. 表函数监视信息

表函数	监视元素收集命令和级别
MON_GET_ACTIVITY_DETAILS 表函数 - 获取完整的活动详细信息（在 XML 文档 DETAILS 中报告）	ACTIVITY METRICS BASE

用法

使用此元素来确定此活动是否已违反应用于此活动的 ACTIVITYTOTALTIME 阈值。

adapter_name -“适配器名称”监视元素

此主机上网络适配器的名称。

表 183. 表函数监视信息

表函数	监视元素收集级别
ENV_GET_NETWORK_RESOURCES 表函数 - 返回网络适配器信息	ACTIVITY METRICS BASE

address - 从中发起连接的 IP 地址

从中发起活动连接的 IP 地址。

表 184. 表函数监视信息

表函数	监视元素收集级别
WLM_GET_SERVICE_CLASS_WORKLOAD_OCCURRENCES 表函数 - 列示工作负载出现次数	ACTIVITY METRICS BASE

表 185. 事件监视信息

事件类型	逻辑数据分组	监视开关
活动	event_activity	-

用法

使用此元素来确定从中发起活动连接的 IP 地址。显示的安全域名将转换为 IP 地址。

agent_id -“应用程序句柄（代理程序标识）”监视元素

应用程序在系统范围内的唯一标识。在单成员数据库配置中，此标识由一个 16 位的计数器构成。在多成员配置中，此标识由协调成员号与 16 位计数器的并置构成。并且，对于应用程序从中建立辅助连接的每个成员，此标识都相同。

表 186. 表函数监视信息

表函数	监视元素收集级别
MON_GET_ACTIVITY_DETAILS 表函数 - 获取完整的活动详细信息（在 XML 文档 DETAILS 中报告）	ACTIVITY METRICS BASE
MON_GET_CONNECTION 表函数 - 获取连接度量值	ACTIVITY METRICS BASE
MON_GET_CONNECTION_DETAILS 表函数 - 获取详细的连接度量值（在 XML 文档 DETAILS 中报告）	ACTIVITY METRICS BASE
MON_GET_LOCKS 表函数 - 列示当前所连接的数据库中的所有锁定	ACTIVITY METRICS BASE
MON_GET_MEMORY_POOL 表函数 - 获取内存池信息	ACTIVITY METRICS BASE
MON_GET_UNIT_OF_WORK 表函数 - 获取工作单元度量值	ACTIVITY METRICS BASE
MON_GET_UNIT_OF_WORK 表函数 - 获取详细的工作单元度量值（在 XML 文档 DETAILS 中报告）	ACTIVITY METRICS BASE

表 187. 快照监视信息

快照级别	逻辑数据分组	监视开关
应用程序	appl_id_info	基本
锁定	appl_lock_list	基本
DCS 应用程序	dcs_appl_info	基本
事务	event_xact	-

表 188. 事件监视信息

事件类型	逻辑数据分组	监视元素收集级别
锁定	-	始终收集
工作单元	-	始终收集
连接	event_connheader	始终收集
语句	event_stmt	始终收集
语句	event_subsection	始终收集
死锁 [1]	event_dlconn	始终收集
带有详细信息的死锁 [1]	event_detailed_dlconn	始终收集
阈值违例	event_thresholdviolations	始终收集
活动	event_activity	始终收集
变更历史记录	changesummary	始终收集

1 建议不要使用此事件监视器。建议不要再使用此选项，将来的发行版可能会将其除去。请使用 CREATE EVENT MONITOR FOR LOCKING 语句来监视与锁定相关的事件，例如锁定超时、锁定等待和死锁。

用法

应用程序句柄，也称为代理程序标识，可用来唯一标识活动应用程序。

注: 根据您使用的 DB2 的版本，**agent_id** 监视元素会有不同的行为。从版本为 SQLM_DBMON_VERSION1 或 SQLM_DBMON_VERSION2 的 DB2 获取快照并发送至 DB2（版本 5 或更高版本）数据库时，返回的 **agent_id** 不能用作应用程序标识，而是作为应用程序提供服务的代理程序的 **agent_pid**。在这些情况下，仍然会返回 **agent_id** 以便与先前发行版兼容，但 DB2 数据库服务器内部不再将该值识别为 **agent_id**。

此值可用作需要代理程序标识的 GET SNAPSHOT 命令的输入或者需要应用程序句柄的监视器表函数的输入。

读取事件跟踪时，它可用来将事件记录与给定应用程序相匹配。

它还可用作 FORCE APPLICATION 命令或 API 的输入。在多节点系统上，可从应用程序具有连接的任何节点发出此命令。它的影响是全局性的。

agent_id_holding_lock -“挂起锁定的代理程序标识”

代理程序的应用程序句柄，该代理程序持有该应用程序正在等待的锁定。必须开启锁定监视器组，才能获取此信息。

表 189. 快照监视信息

快照级别	逻辑数据分组	监视开关
应用程序	appl	锁定
锁定	appl_lock_list	锁定
锁定	lock_wait	锁定

用法 此元素可帮助您确定存在资源争用的应用程序。

如果此元素为 0（零）并且应用程序正在等待锁定，那么这指示锁定被不确定事务所挂起。可使用 appl_id_holding_lk 或 命令行处理器 LIST INDOUBT TRANSACTIONS 命令（当事务变得不确定时，它将显示处理该事务的 CICS® 代理程序的应用程序标识）来确定不确定事务，然后对其执行落实或回滚操作。

注意，多个应用程序可挂起针对此应用程序正在等待的对象的共享锁定。有关该应用程序挂起的锁定类型的信息，请参阅 lock_mode。如果正在获取应用程序快照，那么将只返回对该对象挂起锁定的其中一个代理程序标识。如果正在获取锁定快照，那么将标识对该对象挂起锁定的所有代理程序标识。

agent_pid -“引擎可分派单元 (EDU) 标识”监视元素

代理程序的引擎可分派单元 (EDU) 的唯一标识。除在 Linux 操作系统上以外，EDU 标识与线程标识映射。在 Linux 操作系统上，EDU 标识是 DB2 生成的唯一标识。

表 190. 快照监视信息

快照级别	逻辑数据分组	监视开关
应用程序	agent	语句

表 191. 事件监视信息

事件类型	逻辑数据分组	监视开关
锁定	-	-

用法

可以使用此元素来将数据库系统监视器信息链接至其他诊断信息源，如系统跟踪。还可使用它来监视为数据库应用程序工作的代理程序使用系统资源的方式。

agent_status -“DCS 应用程序代理程序数”

在连接集中器环境中，此值显示当前具有关联代理程序的应用程序。

元素标识

agent_status

元素类型

信息

表 192. 快照监视信息

快照级别	逻辑数据分组	监视开关
DCS 应用程序	dcs_appl_info	基本

用法 值包括:

- SQLM_AGENT_ASSOCIATED

  代表此应用程序工作的代理程序与其相关联。

- SQLM_AGENT_NOT_ASSOCIATED

  代表此应用程序工作的代理程序不再与其相关联并且正被另一应用程序使用。没有关联代理程序的应用程序下一次完成工作时，将重新关联代理程序。

agent_sys_cpu_time -“代理程序使用的系统 CPU 时间”

数据库管理器代理进程使用的总系统 CPU 时间（以秒和微秒计）。

元素标识

agent_sys_cpu_time

元素类型

时间

表 193. 快照监视信息

快照级别	逻辑数据分组	监视开关
应用程序	appl	时间戳记

对于应用程序级别的快照监视，可重置此计数器。不能在其他级别重置此计数器。

用法 此元素与其他相关 CPU 时间元素一起使用可帮助您了解应用程序内的活动级别，还可以帮助您标识可能因为调整而受益的应用程序。

此元素包括花费在 SQL 和非 SQL 语句上的 CPU 时间，同时包括花费在所有不受防护的用户定义的函数（UDF）上的 CPU 时间。

系统 CPU 表示花费在系统调用上的时间。用户 CPU 表示花费在执行数据库管理器代码上的时间。

注: 如果此信息对您的操作系统不可用，那么此元素将设置为 0。

agent_tid -“代理程序线程标识”监视元素

代理程序或系统实体的线程标识。如果此标识不可用，那么此列的值为空。

表 194. 表函数监视信息

表函数	监视元素收集级别
MON_GET_APPL_LOCKWAIT 表函数 - 获取有关应用程序正等待的锁定的信息	ACTIVITY METRICS BASE
WLM_GET_SERVICE_CLASS_AGENTS 表函数 - 列示正在服务类中运行的代理程序	ACTIVITY METRICS BASE

表 195. 事件监视信息

事件类型	逻辑数据分组	监视开关
锁定	lock_participants	

agent_usr_cpu_time -“代理程序使用的用户 CPU 时间”

数据库管理器代理进程使用的总 CPU 时间（以秒和微秒计）。

元素标识

agent_usr_cpu_time

元素类型

时间

表 196. 快照监视信息

快照级别	逻辑数据分组	监视开关
应用程序	appl	时间戳记

可将快照监视的计数器重置。

用法 此元素与其他相关 CPU 时间元素一起使用可帮助您标识消耗大量 CPU 的应用程序或查询。

此计数器包括花费在 SQL 和非 SQL 语句上的时间，同时包括花费在应用程序执行的所有不受防护的用户定义的函数（UDF）或存储过程上的时间。

系统 CPU 表示花费在系统调用上的时间。用户 CPU 表示花费在执行数据库管理器代码上的时间。

注: 如果此信息对您的操作系统不可用，那么此元素将返回为 0。

agent_wait_time -“代理程序等待时间”监视元素

在集中器配置中，已排队的应用程序在等待代理程序时耗用的时间。此值以毫秒计。

表 197. 表函数监视信息

表函数	监视元素收集级别
MON_FORMAT_XML_METRICS_BY_ROW - 获取所有度量值的基于已格式化行的输出	不适用; 报告 XML 文档中作为格式化函数的输入提供的所有元素
MON_FORMAT_XML_TIMES_BY_ROW - - 获取基于已格式化行的组合层次结构等待时间和处理时间	不适用; 报告 XML 文档中作为格式化函数的输入提供的所有元素
MON_FORMAT_XML_WAIT_TIMES_BY _ROW - 获取等待次数的已格式化的基于行的输出	不适用; 报告 XML 文档中作为格式化函数的输入提供的所有元素
MON_GET_CONNECTION 表函数 - 获取连接度量值	REQUEST METRICS BASE
MON_GET_CONNECTION_DETAILS 表函数 - 获取详细的连接度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE
MON_GET_SERVICE_SUBCLASS 表函数 - 获取服务子类度量值	REQUEST METRICS BASE
MON_GET_SERVICE_SUBCLASS_DETAILS 表函数 - 获取详细的服务子类度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE
MON_GET_UNIT_OF_WORK 表函数 - 获取工作单元度量值	REQUEST METRICS BASE
MON_GET_UNIT_OF_WORK 表函数 - 获取详细的工作单元度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE
MON_GET_WORKLOAD 表函数 - 获取工作负载度量值	REQUEST METRICS BASE
MON_GET_WORKLOAD_DETAILS 表函数 - 获取详细的工作负载度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE

表 198. 事件监视信息

事件类型	逻辑数据分组	监视开关
统计信息	event_scstats（在 details_xml 文档中报告）	REQUEST METRICS BASE

表 198. 事件监视信息 (续)

事件类型	逻辑数据分组	监视开关
统计信息	event_wlstats（在 details_xml 文档中报告）	REQUEST METRICS BASE
工作单元	在 system_metrics 文档中报告。	始终收集

用法

agent_wait_time 监视元素可用于帮助评估系统在集中器环境中的运行效率。如果代理程序等待时间相对于 **total_request_time** 监视元素的值而言过长，那么表明请求在进行排队以等待代理程序时耗用时间过多，即，表明发生下列一个或多个事件:

- 所配置的 **max_coordagents** 配置参数相对于工作负载而言太小。您可能需要增大 **max_coordagents** 配置参数的值，或者需要增大 **max_coordagents** 配置参数相对于 **max_connections** 配置参数的比率（如果这两个参数都设置为 AUTOMATIC 的话），以确保提供足够的协调代理程序为应用程序请求及时地提供服务。
- 落实工作负载的频率不够高。为了使集中器高效地工作，应用程序应该相对频繁地发出落实请求，从而确保它们的代理程序可以被释放以便为其他应用程序的请求提供服务。如果应用程序执行落实操作的频率不够高，那么可能需要配置相对较大的协调代理程序数目，以便缩短等待代理程序变为可用时耗用的时间。

agent_waits_total -“等待代理程序总次数”监视元素

在集中器配置中，应用程序被迫等待代理程序被分配的次数。

表 199. 表函数监视信息

表函数	监视元素收集级别
MON_FORMAT_XML_METRICS_BY_ROW - 获取所有度量值的基于已格式化行的输出	不适用; 报告 XML 文档中作为格式化函数的输入提供的所有元素
MON_GET_CONNECTION 表函数 - 获取连接度量值	REQUEST METRICS BASE
MON_GET_CONNECTION_DETAILS 表函数 - 获取详细的连接度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE
MON_GET_SERVICE_SUBCLASS 表函数 - 获取服务子类度量值	REQUEST METRICS BASE
MON_GET_SERVICE_SUBCLASS_DETAILS 表函数 - 获取详细的服务子类度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE
MON_GET_UNIT_OF_WORK 表函数 - 获取工作单元度量值	REQUEST METRICS BASE
MON_GET_UNIT_OF_WORK 表函数 - 获取详细的工作单元度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE
MON_GET_WORKLOAD 表函数 - 获取工作负载度量值	REQUEST METRICS BASE

表 199. 表函数监视信息 (续)

表函数	监视元素收集级别
MON_GET_WORKLOAD_DETAILS 表函数 - 获取详细的工作负载度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE

表 200. 事件监视信息

事件类型	逻辑数据分组	监视开关
统计信息	event_scstats（在 details_xml 文档中报告）	REQUEST METRICS BASE
统计信息	event_wlstats（在 details_xml 文档中报告）	REQUEST METRICS BASE
工作单元	在 system_metrics 文档中报告。	-

用法

通过将此元素与 **agent_wait_time** 监视元素配合使用，可以确定集中器环境中应用程序请求等待代理程序时的平均耗用时间。

agents_created_empty_pool -“由于空的代理程序池而创建的代理程序数”

由于空的代理程序池而创建的代理程序数。它包括在 **db2start** 时启动的代理程序数 (*num_initagents*)。

表 201. 快照监视信息

快照级别	逻辑数据分组	监视开关
数据库管理器	db2	基本

用法 与 agents_from_pool 一起使用来计算

由于空的代理程序池而创建的代理程序数 / 从池中分配的代理程序数

有关使用此元素的信息，请参阅 agents_from_pool。

agents_from_pool -“从池中分配的代理程序数”

从代理程序池中分配的代理程序数。

表 202. 快照监视信息

快照级别	逻辑数据分组	监视开关
数据库管理器	db2	基本

用法

此元素可与 **agents_created_empty_pool** 监视元素配合使用以确定因为池是空的而必须创建代理程序的频率。

以下比率

由于空的代理程序池而创建的代理程序数 / 从池中分配的代理程序数

可用来帮助为 **num_poolagents** 配置参数设置适当的值。

对于大多数用户而言，缺省值 100 与 AUTOMATIC 将确保最优性能。

此比率可能随工作负载而有所波动。当系统上的活动量低时，可能会出现额外的代理程序创建和终止。而当系统上的活动量高时，将出现更多代理程序复用。比率低表示代理程序复用量高，这种情况一般出现在活动量高的系统上。比率高表示发生的代理程序创建量比复用量高。如果有问题，请增加 **num_poolagents** 配置参数的值以降低比率。但是，这样会导致系统产生额外的资源消耗。

agents_registered -“已注册的代理程序数”

正在监视的数据库管理器实例中的已注册代理程序的数目（协调代理程序和子代理程序）。

表 203. 快照监视信息

快照级别	逻辑数据分组	监视开关
数据库管理器	db2	基本

用法

使用此元素来帮助评估 **max_coordagents** 和 **max_connections** 配置参数的设置以及查询内并行性的设置。

agents_registered_top -“已注册的最大代理程序数”

数据库管理器自启动后曾经同时注册的代理程序（协调代理程序和子代理程序）的最大数目。

表 204. 快照监视信息

快照级别	逻辑数据分组	监视开关
数据库管理器	db2	基本

用法

可以使用此元素来帮助您评估 **max_coordagents** 和 **max_connections** 配置参数的设置以及查询内并行性的设置。

agents_registered 监视元素将记录捕获快照时注册的代理程序数。

agents_stolen -“失窃代理程序数”

在数据库管理器快照级别，此监视元素表示与应用程序关联的重新分配到其他应用程序工作的空闲代理数。在应用程序快照级别，此监视元素表示与其他应用程序关联的重新分配到此应用程序工作的空闲代理数。

表 205. 快照监视信息

快照级别	逻辑数据分组	监视开关
数据库管理器	db2	基本
应用程序	appl	基本

可将快照监视的计数器重置。

用法

缺省情况下，**num_poolagents** 配置参数设置为 AUTOMATIC。这意味着 DB2 自动管理空闲代理的组合，包括对与其他应用程序关联的空闲代理分配工作。

agents_top -“创建的代理程序数”

在应用程序级别，此项是执行语句时使用的代理程序的最大数目。在数据库级别，此项是用于所有应用程序的代理程序的最大数目。

元素标识

agents_top

元素类型

水位标记

表 206. 快照监视信息

快照级别	逻辑数据分组	监视开关
数据库	dbase	语句
应用程序	stmt	语句

用法 指示查询内并行性实现情况的指示符。

agents_waiting_on_token -“正在等待令牌的代理程序数”

等待令牌以便在数据库管理器中执行事务的代理程序数。

注: 从 DB2 V9.5 开始，不推荐使用 **agents_waiting_on_token** 监视元素。使用此监视元素不会生成错误。但是不会返回有效值。建议不要再使用此监视元素，将来的发行版中可能会将其除去。

表 207. 快照监视信息

快照级别	逻辑数据分组	监视开关
数据库管理器	db2	基本

用法

可使用此元素来帮助您评估 **maxcagents** 配置参数的设置。

每个应用程序都有一个专用协调代理程序来处理数据库管理器中的数据库请求。每个代理程序都必须获取令牌才能执行事务。可以执行 数据库管理器 事务的最大代理程序数由配置参数 **maxcagents** 限制。

agents_waiting_top -“正在等待的最大代理程序数”监视元素

自数据库管理器启动后曾经同时等待令牌的代理程序的最大数目。

注: 从 DB2 V9.5 开始，不推荐使用 **agents_waiting_top** 监视元素。使用此监视元素不会生成错误。但是不会返回有效值。建议不要再使用此监视元素，将来的发行版中可能会将其除去。

表 208. 快照监视信息

快照级别	逻辑数据分组	监视开关
数据库管理器	db2	基本

用法

使用此元素来帮助您评估 **maxcagents** 配置参数的设置。

获取快照时等待令牌的代理程序数由 **agents_waiting_on_token** 监视元素将记录。

如果 **maxcagents** 参数设置为其缺省值（-1），那么不应有任何代理程序等待令牌并且此监视元素的值应该为零。

agg_temp_tablespace_top -“最大聚集临时表空间”监视元素

服务类中所有嵌套级别的 DML 活动使用的聚集临时表空间的高水位标记（以 KB 计）。此聚集值是通过对服务子类中所有活动的临时表空间使用量进行求和计算而得的，此高水位标记表示此聚集值在上次重置之后达到的最大值。当服务类的 COLLECT AGGREGATE ACTIVITY DATA 设置为 NONE 时，此监视元素将返回 -1。必须至少对此记录所属子类的超类中的一个服务子类定义并启用 AGGSQLTEMPSPACE 阈值，否则将返回 0 值。

表 209. 事件监视信息

事件类型	逻辑数据分组	监视开关
统计信息	event_scstats	始终收集

用法

使用此元素来确定在收集时间间隔内，成员对服务子类上达到的最高聚集 DML 活动系统临时表空间使用量。

aggsqltempspace_threshold_id -“聚集 SQL 临时空间阈值标识”监视元素

应用于此活动的 AGGSQLTEMPSPACE 阈值的数字标识。

表 210. 表函数监视信息

表函数	监视元素收集级别
MON_GET_ACTIVITY_DETAILS 表函数 - 获取完整的活动详细信息（在 XML 文档 DETAILS 中报告）	ACTIVITY METRICS BASE

用法

使用此元素来确定应用于此活动的 AGGSQLTEMPSPACE 阈值（如果有的话）。

aggsqltempspace_threshold_value -“AggSQL 临时空间阈值”监视元素

应用于此活动的 AGGSQLTEMPSPACE 阈值的上限。

表 211. 表函数监视信息

表函数	监视元素收集级别
MON_GET_ACTIVITY_DETAILS 表函数 - 获取完整的活动详细信息（在 XML 文档 DETAILS 中报告）	ACTIVITY METRICS BASE

用法

使用此元素来确定应用于此活动的 AGGSQLTEMPSPACE 阈值（如果有的话）。

aggsqltempspace_threshold_violated -“违反 AggSQL 临时空间阈值”监视元素

当这个可选的监视元素设置为“Yes”时，表明此活动已违反对其应用的 AGGSQLTEMPSPACE 阈值。“No”表明此活动尚未违反该阈值。

表 212. 表函数监视信息

表函数	监视元素收集级别
MON_GET_ACTIVITY_DETAILS 表函数 - 获取完整的活动详细信息（在 XML 文档 DETAILS 中报告）	ACTIVITY METRICS BASE

用法

使用此元素来确定此活动是否已违反应用于此活动的 AGGSQLTEMPSPACE 阈值。

app_act_aborted_total -“失败的外部协调程序活动总数”监视元素

已完成但带有错误的外部非嵌套协调程序活动总数。对于服务类而言，如果在活动异常终止前通过 REMAP ACTIVITY 操作将其重新映射到另一个服务子类，那么此活动将仅计入在其中异常终止此活动的子类的总数。

表 213. 表函数监视信息

表函数	监视元素收集级别
MON_FORMAT_XML_METRICS_BY_ROW - 获取所有度量值的基于已格式化行的输出	不适用; 报告 XML 文档中作为格式化函数的输入提供的所有元素
MON_GET_CONNECTION 表函数 - 获取连接度量值	REQUEST METRICS BASE
MON_GET_CONNECTION_DETAILS 表函数 - 获取详细的连接度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE
MON_GET_SERVICE_SUBCLASS 表函数 - 获取服务子类度量值	REQUEST METRICS BASE
MON_GET_SERVICE_SUBCLASS_DETAILS 表函数 - 获取详细的服务子类度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE
MON_GET_UNIT_OF_WORK 表函数 - 获取工作单元度量值	REQUEST METRICS BASE
MON_GET_UNIT_OF_WORK 表函数 - 获取详细的工作单元度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE
MON_GET_WORKLOAD 表函数 - 获取工作负载度量值	REQUEST METRICS BASE
MON_GET_WORKLOAD_DETAILS 表函数 - 获取详细的工作负载度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE
WLM_GET_SERVICE_SUBCLASS_STATS 表函数 - 返回服务子类的统计信息	ACTIVITY METRICS BASE
WLM_GET_WORKLOAD_STATS 表函数 - 返回工作负载统计信息	ACTIVITY METRICS BASE

表 214. 事件监视信息

事件类型	逻辑数据分组	监视开关
统计信息	event_scstats（在 details_xml 文档中报告）	REQUEST METRICS BASE
统计信息	event_wlstats（在 details_xml 文档中报告）	REQUEST METRICS BASE
工作单元	uow（在 metrics.xml 文档中报告） uow_metrics	REQUEST METRICS BASE

app_act_completed_total -“成功的外部协调程序活动总数”监视元素

已成功完成的外部非嵌套协调程序活动总数。

对于服务类而言，如果在活动完成前通过 REMAP ACTIVITY 操作将其重新映射到另一个子类，那么此活动将仅计入在其中完成此活动的子类的总数。

表 215. 表函数监视信息

表函数	监视元素收集级别
MON_FORMAT_XML_METRICS_BY_ROW - 获取所有度量值的基于已格式化行的输出	不适用；报告 XML 文档中作为格式化函数的输入提供的所有元素
MON_GET_CONNECTION 表函数 - 获取连接度量值	REQUEST METRICS BASE
MON_GET_CONNECTION_DETAILS 表函数 - 获取详细的连接度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE
MON_GET_SERVICE_SUBCLASS 表函数 - 获取服务子类度量值	REQUEST METRICS BASE
MON_GET_SERVICE_SUBCLASS_DETAILS 表函数 - 获取详细的服务子类度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE
MON_GET_UNIT_OF_WORK 表函数 - 获取工作单元度量值	REQUEST METRICS BASE
MON_GET_UNIT_OF_WORK 表函数 - 获取详细的工作单元度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE
MON_GET_WORKLOAD 表函数 - 获取工作负载度量值	REQUEST METRICS BASE
MON_GET_WORKLOAD_DETAILS 表函数 - 获取详细的工作负载度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE
WLM_GET_SERVICE_SUBCLASS_STATS 表函数 - 返回服务子类的统计信息	ACTIVITY METRICS BASE
WLM_GET_WORKLOAD_STATS 表函数 - 返回工作负载统计信息	ACTIVITY METRICS BASE

表 216. 事件监视信息

事件类型	逻辑数据分组	监视开关
统计信息	event_scstats（在 details_xml 文档中报告）	REQUEST METRICS BASE
统计信息	event_wlstats（在 details_xml 文档中报告）	REQUEST METRICS BASE
工作单元	uow（在 metrics.xml 文档中报告） uow_metrics	REQUEST METRICS BASE

app_act_rejected_total -“拒绝的外部协调程序活动总数”监视元素

在任何嵌套级别由于被拒绝而未被允许执行的外部非嵌套协调程序活动的总数。

表 217. 表函数监视信息

表函数	监视元素收集级别
MON_FORMAT_XML_METRICS_BY_ROW - 获取所有度量值的基于已格式化行的输出	不适用; 报告 XML 文档中作为格式化函数的输入提供的所有元素
MON_GET_CONNECTION 表函数 - 获取连接度量值	REQUEST METRICS BASE
MON_GET_CONNECTION_DETAILS 表函数 - 获取详细的连接度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE
MON_GET_SERVICE_SUBCLASS 表函数 - 获取服务子类度量值	REQUEST METRICS BASE
MON_GET_SERVICE_SUBCLASS_DETAILS 表函数 - 获取详细的服务子类度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE
MON_GET_UNIT_OF_WORK 表函数 - 获取工作单元度量值	REQUEST METRICS BASE
MON_GET_UNIT_OF_WORK 表函数 - 获取详细的工作单元度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE
MON_GET_WORKLOAD 表函数 - 获取工作负载度量值	REQUEST METRICS BASE
MON_GET_WORKLOAD_DETAILS 表函数 - 获取详细的工作负载度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE
WLM_GET_SERVICE_SUBCLASS_STATS 表函数 - 返回服务子类的统计信息	ACTIVITY METRICS BASE
WLM_GET_WORKLOAD_STATS 表函数 - 返回工作负载统计信息	ACTIVITY METRICS BASE

表 218. 事件监视信息

事件类型	逻辑数据分组	监视开关
统计信息	event_scstats（在 details_xml 文档中报告）	REQUEST METRICS BASE
统计信息	event_wlstats（在 details_xml 文档中报告）	REQUEST METRICS BASE
工作单元	uow（在 metrics.xml 文档中报告） uow_metrics	REQUEST METRICS BASE

appl_action -“应用程序操作”监视元素

客户机应用程序正在执行的操作或请求。

表 219. 事件监视信息

事件类型	逻辑数据分组	监视开关
锁定	lock_participants	

app_rqsts_completed_total -“完成应用程序请求总数”监视元素

协调程序所执行的外部（应用程序）请求的总数。对于服务子类而言，将仅对完成此应用程序请求的子类更新此监视元素。

表 220. 表函数监视信息

表函数	监视元素收集级别
MON_FORMAT_XML_METRICS_BY_ROW - 获取所有度量值的基于已格式化行的输出	不适用; 报告 XML 文档中作为格式化函数的输入提供的所有元素
MON_GET_CONNECTION 表函数 - 获取连接度量值	REQUEST METRICS BASE
MON_GET_CONNECTION_DETAILS 表函数 - 获取详细的连接度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE
MON_GET_SERVICE_SUBCLASS 表函数 - 获取服务子类度量值	REQUEST METRICS BASE
MON_GET_SERVICE_SUBCLASS_DETAILS 表函数 - 获取详细的服务子类度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE
MON_GET_UNIT_OF_WORK 表函数 - 获取工作单元度量值	REQUEST METRICS BASE
MON_GET_UNIT_OF_WORK 表函数 - 获取详细的工作单元度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE
MON_GET_WORKLOAD 表函数 - 获取工作负载度量值	REQUEST METRICS BASE
MON_GET_WORKLOAD_DETAILS 表函数 - 获取详细的工作负载度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE

表 221. 事件监视信息

事件类型	逻辑数据分组	监视开关
统计信息	event_scstats（在 details_xml 文档中报告）	REQUEST METRICS BASE
统计信息	event_wlstats（在 details_xml 文档中报告）	REQUEST METRICS BASE
工作单元	在 system_metrics 文档中报告。	REQUEST METRICS BASE

用法

使用此监视元素来确定应用程序向系统提交的请求数。

appl_con_time -“连接请求启动时间戳记”

应用程序启动连接请求的日期和时间。

元素标识

appl_con_time

元素类型

时间戳记

表 222. 快照监视信息

快照级别	逻辑数据分组	监视开关
应用程序	appl	时间戳记

用法 使用此元素来确定应用程序启动连接至数据库的请求的时间。

appl_id -“应用程序标识”监视元素

此标识是在应用程序连接至数据库管理器上的数据库或 DB2 Connect 接收到连接至 DRDA 数据库的请求时生成的。

表 223. 表函数监视信息

表函数	监视元素收集级别
MON_GET_CONNECTION 表函数 - 获取连接度量值	ACTIVITY METRICS BASE
MON_GET_CONNECTION_DETAILS 表函数 - 获取详细的连接度量值（在 XML 文档 DETAILS 中报告）	ACTIVITY METRICS BASE
MON_GET_UNIT_OF_WORK 表函数 - 获取工作单元度量值	ACTIVITY METRICS BASE
MON_GET_UNIT_OF_WORK 表函数 - 获取详细的工作单元度量值（在 XML 文档 DETAILS 中报告）	ACTIVITY METRICS BASE
PD_GET_DIAG_HIST 表函数 - 从给定设施返回记录	ACTIVITY METRICS BASE
PDLOGMSGS_LAST24HOURS 管理视图和 PD_GET_LOG_MSGS 表函数 - 检索问题确定消息	ACTIVITY METRICS BASE
WLM_GET_SERVICE_CLASS_AGENTS 表函数 - 列示正在服务类中运行的代理程序	ACTIVITY METRICS BASE
WLM_GET_SERVICE_CLASS_WORKLOAD_OCCURRENCES 表函数 - 列示工作负载出现次数	ACTIVITY METRICS BASE

表 224. 快照监视信息

快照级别	逻辑数据分组	监视开关
应用程序	appl_id_info	基本
DCS 应用程序	dcs_appl_info	基本
锁定	appl_lock_list	基本

表 225. 事件监视信息

事件类型	逻辑数据分组	监视开关
锁定 [1]	lock_participants	始终收集
工作单元 [1]	uow, uow_executable_list, uow_metrics	始终收集
连接	event_conn	始终收集
连接	event_connheader	始终收集
语句	event_stmt	始终收集
事务 [2]	event_xact	始终收集
死锁 [3]	event_dlconn	始终收集
带有详细信息的死锁 [3]	event_detailed_dlconn	始终收集
活动	event_activitystmt	始终收集
活动	event_activity	始终收集
活动	event_activityvals	始终收集
活动	event_activitymetrics	始终收集
阈值违例	event_thresholdviolations	始终收集
变更历史记录	changesummary	始终收集

1. 对于此事件监视器，此监视元素在列 APPLICATION_ID 中返回。
2. 建议不要使用此事件监视器。建议不要再使用此选项，将来的发行版可能会将其除去。请使用 CREATE EVENT MONITOR FOR UNIT OF WORK 语句来监视事务事件。
3. 建议不要使用此事件监视器。建议不要再使用此选项，将来的发行版可能会将其除去。请使用 CREATE EVENT MONITOR FOR LOCKING 语句来监视与锁定相关的事件，例如锁定超时、锁定等待和死锁。

用法

此标识在客户机和服务器上都是已知的，所以可使用它来使应用程序的客户机部分与服务器部分相关。对于 DB2 Connect 应用程序，那么还需要使用 **outbound_appl_id** 监视元素来使应用程序的客户机部分与服务器部分相关。

此标识在网络上是唯一的。应用程序标识有不同的格式，这取决于运行数据库管理器和/或 DB2 Connect 的客户机与服务器之间的通信协议。每种格式由用逗号隔开的三个部分构成。

1. TCP/IP

   格式 `IPAddr.Port.Timestamp`

   IPv4

示例 9.26.120.63.43538.090924175700

详细信息

在 IPv4 中，TCP/IP 生成的应用程序标识由三个部分组成。第一部分是 IP 地址。它表示为格式为 a.b.c.d 的 4 个十进制数。第二部分是端口号，它表示为 5 个十进制字符。第三部分是 近似时间戳记，表示为 12 个十进制字符。

IPv6

示例 2002:91a:519:13:20d:60ff:feef:cc64.5309.090924175700

详细信息

在 IPv6 中，TCP/IP 生成的应用程序标识由三部分组成。第一部分包含格式为 a:b:c:d:e:f:g:h 的 IPv6 地址，其中 a-h 中的每一个都是 4 位十六进制数字。第二部分是端口号。第三部分是此应用程序的实例的 近似时间戳记标识。

2. 本地应用程序

格式 *LOCAL.DB2 instance.Application instance

示例

*LOCAL.DB2INST1.930131235945

详细信息

为本地应用程序生成的应用程序标识是通过并置字符串 *LOCAL、DB2 实例的名称和此应用程序的实例的唯一标识构成的。

对于多数据库分区实例，LOCAL 将替换为 Nx，其中 x 是客户机用来连接至数据库的分区号。例如，*N2.DB2INST1.0B5A12222841。

使用 **client_protocol** 监视元素来确定连接使用的通信协议，并因此确定 **appl_id** 监视元素的格式。

appl_id_holding_lk -“挂起锁定的应用程序标识”

已经持有所需的对象锁定的应用程序的标识。

元素标识

appl_id_holding_lk

元素类型

信息

表 226. 快照监视信息

快照级别	逻辑数据分组	监视开关
应用程序	appl	锁定
锁定	appl_lock_list	锁定
锁定	lock_wait	锁定

表 227. 事件监视信息

事件类型	逻辑数据分组	监视开关
死锁	event_dlconn	始终收集

表 227. 事件监视信息 (续)

事件类型	逻辑数据分组	监视开关
带有详细信息的死锁	event_detailed_dlconn	始终收集

用法 此元素可帮助您确定存在资源争用的应用程序。具体而言，它可以帮助您标识挂起锁定的应用程序句柄（代理程序标识）和表标识。注意，可使用 LIST APPLICATIONS 命令来获取有关带有代理程序标识的应用程序标识的信息。但是，最好在获取快照时收集此类型的信息，原因是应用程序在运行 LIST APPLICATIONS 命令之前结束时此项将变得不可用。

注意，多个应用程序可挂起某个对象的共享锁定，此应用程序正在等待对该对象获取锁定。有关该应用程序挂起的锁定类型的信息，请参阅 lock_mode。如果正在获取应用程序快照，那么将只返回对该对象挂起锁定的其中一个应用程序标识。如果正在获取锁定快照，那么将返回对该对象挂起锁定的所有应用程序标识。

appl_id_oldest_xact -“带有最旧事务的应用程序”

具有最旧事务的应用程序的标识（对应于应用程序快照中的 *agent_id* 值）。

元素标识

appl_id_oldest_xact

元素类型

信息

表 228. 快照监视信息

快照级别	逻辑数据分组	监视开关
数据库	dbase	基本

用法 此元素可帮助您确定具有最旧活动事务的应用程序。可强制此应用程序释放日志空间。如果它占用了大量日志空间，那么应检查应用程序以确定是否可以修改它以提高执行落实操作的频率。

有时没有事务停止记录或者最旧的事务没有应用程序标识（例如，不确定事务或不活动事务）。在这类情况下，数据流中不会返回此应用程序的标识。

appl_idle_time -“应用程序空闲时间”

应用程序对服务器发出任何请求后经历的秒数。这包括未终止事务的应用程序，如未发出落实或回滚的应用程序。

元素标识

appl_idle_time

元素类型

信息

表 229. 快照监视信息

快照级别	逻辑数据分组	监视开关
应用程序	appl	语句

表 229. 快照监视信息 (续)

快照级别	逻辑数据分组	监视开关
DCS 应用程序	dcs_appl	语句

用法 此信息可用来实现强制用户空闲指定秒数的应用程序。

appl_name -“应用程序名称”监视元素

正在客户机上运行的并且数据库或 DB2 Connect 服务器所知道的应用程序的名称。

表 230. 表函数监视信息

表函数	监视元素收集级别
ADMINTEMPCOLUMNS 管理视图和 ADMIN_GET_TEMP_COLUMNS 表函数 - 检索临时表的列信息	ACTIVITY METRICS BASE
ADMINTEMPTABLES 管理视图和 ADMIN_GET_TEMP_TABLES 表函数 - 检索临时表的信息	ACTIVITY METRICS BASE
MON_GET_CONNECTION 表函数 - 获取连接度量值	ACTIVITY METRICS BASE
MON_GET_CONNECTION_DETAILS 表函数 - 获取详细的连接度量值（在 XML 文档 DETAILS 中报告）	ACTIVITY METRICS BASE
WLM_GET_SERVICE_CLASS_AGENTS 表函数 - 列示正在服务类中运行的代理程序	ACTIVITY METRICS BASE
WLM_GET_SERVICE_CLASS_WORKLOAD_OCCURRENCES 表函数 - 列示工作负载出现次数	ACTIVITY METRICS BASE

表 231. 快照监视信息

快照级别	逻辑数据分组	监视开关
应用程序	appl_id_info	基本
锁定	appl_lock_list	基本
DCS 应用程序	dcs_appl_info	基本

表 232. 事件监视信息

事件类型	逻辑数据分组	监视开关
锁定	-	始终收集
工作单元	-	始终收集
连接	event_connheader	始终收集
活动	event_activity	始终收集
变更历史记录	changesummary	始终收集

用法

此元素可与 **appl_id** 配合使用以使数据项与应用程序相关。

在客户机/服务器环境中建立数据库连接时此名称将从客户机传递至服务器。应用程序名称中的任何非英语字符将被除去。CLI 应用程序可通过对 SQLSetConnectAttr 的调用来设置 SQL_ATTR_INFO_PROGRAMNAME 属性。如果 SQL_ATTR_INFO_PROGRAMNAME 是在建立与服务器的连接前设置的，那么指定的值将覆盖实际客户机应用程序名称并且成为 **appl_name** 监视元素中显示的值。

如果客户机应用程序代码页与运行 数据库系统监视器 的代码页不同，可使用 **codepage_id** 来帮助转换 **appl_name**。

appl_priority -“应用程序代理程序优先级”

为此应用程序工作的代理程序的优先级。

元素标识

appl_priority

元素类型

信息

表 233. 快照监视信息

快照级别	逻辑数据分组	监视开关
应用程序	appl	基本

表 234. 事件监视信息

事件类型	逻辑数据分组	监视开关
连接	event_conn	始终收集

用法 可使用此元素来检查应用程序是否以期望的优先级运行。应用程序优先级可由管理员设置。可通过控制器实用程序（**db2gov**）来更改优先级。

DB2 使用控制器来监视和更改对数据库运行的应用程序的行为。此信息用来调度应用程序和平衡系统资源。

控制器守护程序通过获取快照来收集有关应用程序的统计信息。它将对照管理对该数据库运行的应用程序的规则来检查这些统计信息。如果控制器检测到规则违例，那么采取适当的操作。这些规则和操作是由您在控制器配置文件中指定的。

如果与某个规则相关联的操作将更改应用程序的优先级，那么控制器将在检测到违例的分区中更改代理程序的优先级。

appl_priority_type -“应用程序优先级类型”

代表应用程序工作的代理程序的操作系统优先级类型。

元素标识

appl_priority_type

元素类型

信息

表 235. 快照监视信息

快照级别	逻辑数据分组	监视开关
应用程序	appl	基本

表 236. 事件监视信息

事件类型	逻辑数据分组	监视开关
连接	event_conn	始终收集

用法 动态优先级由操作系统根据使用情况重新计算。静态优先级不会更改。

appl_section_inserts -“节插入数”监视元素

应用程序从其共享 SQL 工作空间插入 SQL 节的次数。

表 237. 快照监视信息

快照级别	逻辑数据分组	监视开关
数据库	dbase	基本
应用程序	appl	基本

表 238. 事件监视信息

事件类型	逻辑数据分组	监视开关
数据库	event_db	始终收集
连接	event_conn	始终收集

用法

任何可执行部分的工作副本都存储在共享 SQL 工作空间中。这是出现副本不可用并且必须插入的情况的计数。

appl_section_lookups -“节查询数”

应用程序从其共享 SQL 工作空间查询 SQL 节的次数。

表 239. 快照监视信息

快照级别	逻辑数据分组	监视开关
数据库	dbase	基本
应用程序	appl	基本

可将快照监视的计数器重置。

表 240. 事件监视信息

事件类型	逻辑数据分组	监视开关
数据库	event_db	始终收集
连接	event_conn	始终收集

用法

每个代理程序都可以访问保留任何可执行部分的工作副本的共享 SQL 工作空间。此计数器指示应用程序的代理程序访问 SQL 工作区的次数。

appl_status - 应用程序状态监视元素

应用程序的当前状态。

表 241. 快照监视信息

快照级别	逻辑数据分组	监视开关
应用程序	appl_id_info	基本
锁定	appl_lock_list	基本

表 242. 事件监视信息

事件类型	逻辑数据分组	监视开关
锁定	-	始终收集
连接	event_conn	始终收集

用法

此元素可以帮助您诊断潜在的应用程序问题。下表列示了此字段的值。

API 常量	描述
SQLM_AUTONOMOUS_WAIT	**自主等待**: 应用程序等待自主例程完成。
SQLM_BACKUP	**正在备份数据库**: 应用程序正在执行数据库备份。
SQLM_COMMIT_ACT	**正在落实**: 工作单元正在落实对其数据库所作的更改。
SQLM_COMP	**正在编译**: 数据库管理器正在为应用程序编译 SQL 语句或者预编译方案。
SQLM_CONNECTED	**数据库连接已完成**: 应用程序已启动数据库连接请求，并且该请求已完成。
SQLM_CONNECTPEND	**数据库连接暂挂**: 应用程序已启动数据库连接请求，但该请求尚未完成。
SQLM_CREATE_DB	**正在创建数据库**: 代理程序已启动了数据库创建请求，该请求尚未完成。
SQLM_DECOUPLED	**已经与代理程序解耦**: 当前没有与应用程序相关联的代理程序。这是一种正常状态。当连接集中器处于启用状态时，没有专用的协调代理程序，因此可以在协调程序分区中将应用程序解耦。在非集中器环境中，由于始终有专用的协调代理程序，所以无法在协调程序分区中将应用程序解耦。
SQLM_DISCONNECTPEND	**数据库断开连接暂挂**: 应用程序已启动数据库断开连接命令，但该命令尚未完成执行。可能是应用程序未显式执行数据库断开连接命令。如果应用程序在未执行断开连接命令的情况下结束，数据库管理器就会断开与数据库的连接。
SQLM_INTR	**请求已中断**: 正在处理请求中断。

API 常量	描述
SQLM_IOERROR_WAIT	**等待以禁用表空间**：应用程序检测到 I/O 错误，并且正在尝试禁用特定表空间。应用程序必须先等待对该表空间执行的所有其他活动事务完成，然后才能禁用该表空间。
SQLM_LOAD	**数据快速装入**：应用程序正在执行“快速装入”以便将数据装入到数据库中。
SQLM_LOCKWAIT	**等待锁定**：工作单元正在等待锁定。获取锁定之后，状态将复原为其先前值。
SQLM_QUIESCE_TABLESPACE	**正在停顿表空间**：应用程序正在执行停顿表空间请求。
SQLM_RECOMP	**正在重新编译**：数据库管理器正在为应用程序重新编译（即，重新绑定）方案。
SQLM_REMOTE_RQST	**联合请求暂挂**：应用程序正在等待来自联合数据源的结果。
SQLM_RESTART	**正在重新启动数据库**：应用程序正在重新启动数据库以执行崩溃恢复。
SQLM_RESTORE	**正在恢复数据库**：应用程序正在将备份映像恢复到数据库。
SQLM_ROLLBACK_ACT	**正在回滚**：工作单元正在回滚对其数据库所作的更改。
SQLM_ROLLBACK_TO_SAVEPOINT	**回滚到保存点**：应用程序正在回滚到保存点。
SQLM_TEND	**事务已结束**：该工作单元是已结束但尚未进入两阶段落实协议准备阶段的全局事务的一部分。
SQLM_THABRT	**事务已试探性回滚**：该工作单元是已试探性回滚的全局事务的一部分。
SQLM_THCOMT	**事务已试探性落实**：该工作单元是已试探性落实的全局事务的一部分。
SQLM_TPREP	**事务已准备好**：该工作单元是已进入两阶段落实协议准备阶段的全局事务的一部分。
SQLM_UNLOAD	**数据快速卸装**：应用程序正在执行“快速卸装”以便从数据库中卸装数据。
SQLM_UOWEXEC	**工作单元正在执行**：数据库管理器正在代表工作单元执行请求。
SQLM_UOWQUEUED	**工作单元 已排队**：工作单元已排队，正在等待另一个活动执行完毕。工作单元已排队，因为已经达到了可以同时执行的活动数的阈值。
SQLM_UOWWAIT	**工作单元正在等待**：数据库管理器代表应用程序中的工作单元正在等待。此状态通常表示系统正在执行应用程序代码。
SQLM_WAITFOR_REMOTE	**暂挂远程请求**：应用程序正在等待来自分区数据库实例中的远程分区的响应。

application_handle -“应用程序句柄”监视元素

应用程序在系统范围内的唯一标识。在单成员数据库配置中，此标识由一个 16 位的计数器构成。在多成员配置中，此标识由协调成员号与 16 位计数器的并置构成。并且，对于应用程序从中建立辅助连接的每个成员，此标识都相同。

表 243. 表函数监视信息

表函数	监视元素收集级别
MON_GET_ACTIVITY_DETAILS 表函数 - 获取完整的活动详细信息（在 XML 文档 DETAILS 中报告）	ACTIVITY METRICS BASE
MON_GET_CONNECTION 表函数 - 获取连接度量值	ACTIVITY METRICS BASE
MON_GET_CONNECTION_DETAILS 表函数 - 获取详细的连接度量值（在 XML 文档 DETAILS 中报告）	ACTIVITY METRICS BASE
MON_GET_LOCKS 表函数 - 列示当前所连接的数据库中的所有锁定	ACTIVITY METRICS BASE
MON_GET_MEMORY_POOL 表函数 - 获取内存池信息	ACTIVITY METRICS BASE
MON_GET_UNIT_OF_WORK 表函数 - 获取工作单元度量值	ACTIVITY METRICS BASE
MON_GET_UNIT_OF_WORK 表函数 - 获取详细的工作单元度量值（在 XML 文档 DETAILS 中报告）	ACTIVITY METRICS BASE

表 244. 快照监视信息

快照级别	逻辑数据分组	监视开关
应用程序	appl_id_info	基本
锁定	appl_lock_list	基本
DCS 应用程序	dcs_appl_info	基本
事务	event_xact	-

表 245. 事件监视信息

事件类型	逻辑数据分组	监视元素收集级别
锁定	-	始终收集
工作单元	-	始终收集
连接	event_connheader	始终收集
语句	event_stmt	始终收集
语句	event_subsection	始终收集
死锁 [1]	event_dlconn	始终收集
带有详细信息的死锁 [1]	event_detailed_dlconn	始终收集
阈值违例	event_thresholdviolations	始终收集
活动	event_activity	始终收集
变更历史记录	changesummary	始终收集

1 建议不要使用此事件监视器。建议不要再使用此选项，将来的发行版可能会将其除去。请使用 CREATE EVENT MONITOR FOR LOCKING 语句来监视与锁定相关的事件，例如锁定超时、锁定等待和死锁。

用法

此监视元素是 **agent_id** 监视元素的别名。

除了所描述的内存池是下列其中一种类型时之外，当 MON_GET_MEMORY_POOL 返回此监视元素时，此监视元素为 NULL:

- APPLICATION
- STATISTICS
- STATEMENT
- SORT_PRIVATE。

appls_cur_cons -“当前连接的应用程序数”

指示当前连接至数据库的应用程序数。

表 246. 快照监视信息

快照级别	逻辑数据分组	监视开关
数据库	dbase	基本
锁定	db_lock_list	基本

用法 可使用此元素来帮助您了解数据库内的活动级别以及正在使用的系统资源量。

它可帮助您调整 *maxappls* 和 *max_coordagents* 配置参数的设置。例如，它的值总是与 *maxappls* 相同，您可能想要提高 *maxappls* 的值。有关更多信息，请参阅 *rem_cons_in* 和 *local_cons* 监视元素。

appls_in_db2 -“数据库中当前执行的应用程序数”

指示当前连接至数据库并且数据库管理器正对其处理请求的应用程序的数目。

表 247. 快照监视信息

快照级别	逻辑数据分组	监视开关
数据库	dbase	基本

arm_correlator -“应用程序响应测量相关因子”监视元素

符合应用程序响应测量（ARM）标准的事务标识。

表 248. 事件监视信息

事件类型	逻辑数据分组	监视开关
活动	event_activity	-

用法

如果与此活动关联的应用程序也支持应用程序响应测量（ARM）标准，那么可以使用此元素来将活动事件监视器收集的活动链接至此类应用程序。

associated_agents_top -“最大关联代理程序数”

与此应用程序相关联的最大子代理程序数。

表 249. 快照监视信息

快照级别	逻辑数据分组	监视开关
应用程序	appl	基本

async_read_time -“异步读时间”监视元素

异步引擎可分派单元 (EDU) 读取缓冲池或表空间时所耗用的总时间。此值以毫秒计。

表 250. 表函数监视信息

表函数	监视元素收集级别
MON_GET_BUFFERPOOL 表函数 - 获取缓冲池度量值	DATA OBJECT METRICS BASE
MON_GET_TABLESPACE 表函数 - 获取表空间度量值	DATA OBJECT METRICS BASE

async_write_time -“异步写时间”监视元素

异步引擎可分派单元 (EDU) 写至缓冲池或表空间时所耗用的总时间。此值以毫秒计。

表 251. 表函数监视信息

表函数	监视元素收集级别
MON_GET_BUFFERPOOL 表函数 - 获取缓冲池度量值	DATA OBJECT METRICS BASE
MON_GET_TABLESPACE 表函数 - 获取表空间度量值	DATA OBJECT METRICS BASE

async_runstats -“异步 RUNSTATS 请求总数”监视元素

实时统计信息收集对数据库中所有应用程序执行的成功异步 RUNSTATS 活动总数。所有数据库分区报告的值将汇总合计。

表 252. 表函数监视信息

表函数	监视元素收集级别
SNAP_GET_DB_V97 表函数 - 从 dbase 逻辑组检索快照信息	
SNAPDB 管理视图和 SNAP_GET_DB 表函数 - 从 dbase 逻辑组检索快照信息	

表 253. 快照监视信息

快照级别	逻辑数据分组	监视开关
数据库	dbase	语句

可将快照监视的计数器重置。

表 254. 事件监视信息

事件类型	逻辑数据分组	监视开关
数据库	event_db	始终收集

用法

使用此元素来确定实时统计信息收集执行的成功异步 RUNSTATS 活动数。此值经常更改。为更好地了解系统使用情况，请长期在特定时间间隔捕获快照。与 **sync_runstats** 和 **stats_fabrications** 监视元素配合使用时，此元素可帮助您跟踪与实时统计信息收集相关的不同统计信息收集活动类型并分析它们对性能的影响。

audit_events_total -“审计事件总数”监视元素

所生成的审计事件的总数。

表 255. 表函数监视信息

表函数	监视元素收集级别
MON_FORMAT_XML_METRICS_BY_ROW - 获取所有度量值的基于已格式化行的输出	不适用; 报告 XML 文档中作为格式化函数的输入提供的所有元素
MON_GET_ACTIVITY_DETAILS 表函数 - 获取完整的活动详细信息（在 XML 文档 DETAILS 中报告）	ACTIVITY METRICS BASE
MON_GET_CONNECTION 表函数 - 获取连接度量值	REQUEST METRICS BASE
MON_GET_CONNECTION_DETAILS 表函数 - 获取详细的连接度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE
MON_GET_PKG_CACHE_STMT 表函数 - 获取程序包高速缓存中的 SQL 语句活动度量值	ACTIVITY METRICS BASE
MON_GET_PKG_CACHE_STMT_DETAILS 表函数 - 获取程序包高速缓存条目的详细度量值	ACTIVITY METRICS BASE
MON_GET_SERVICE_SUBCLASS 表函数 - 获取服务子类度量值	REQUEST METRICS BASE
MON_GET_SERVICE_SUBCLASS_DETAILS 表函数 - 获取详细的服务子类度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE
MON_GET_UNIT_OF_WORK 表函数 - 获取工作单元度量值	REQUEST METRICS BASE
MON_GET_UNIT_OF_WORK 表函数 - 获取详细的工作单元度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE

表 255. 表函数监视信息 (续)

表函数	监视元素收集级别
MON_GET_WORKLOAD 表函数 - 获取工作负载度量值	REQUEST METRICS BASE
MON_GET_WORKLOAD_DETAILS 表函数 - 获取详细的工作负载度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE

表 256. 事件监视信息

事件类型	逻辑数据分组	监视开关
活动	event_activity（在 details_xml 文档中报告）	ACTIVITY METRICS BASE
活动	event_activitymetrics	ACTIVITY METRICS BASE
统计信息	event_scstats（在 details_xml 文档中报告）	REQUEST METRICS BASE
统计信息	event_wlstats（在 details_xml 文档中报告）	REQUEST METRICS BASE
工作单元	在 system_metrics 文档中报告。	REQUEST METRICS BASE
程序包高速缓存	在 activity_metrics 文档中报告。	ACTIVITY METRICS BASE

audit_file_write_wait_time -“审计文件写等待时间”监视元素

等待写审计记录时耗用的时间。此值以毫秒计。

表 257. 表函数监视信息

表函数	监视元素收集级别
MON_FORMAT_XML_METRICS_BY_ROW - 获取所有度量值的基于已格式化行的输出	不适用; 报告 XML 文档中作为格式化函数的输入提供的所有元素
MON_FORMAT_XML_TIMES_BY_ROW - - 获取基于已格式化行的组合层次结构等待时间和处理时间	不适用; 报告 XML 文档中作为格式化函数的输入提供的所有元素
MON_FORMAT_XML_WAIT_TIMES_BY _ROW - 获取等待时间的已格式化的基于行的输出	不适用; 报告 XML 文档中作为格式化函数的输入提供的所有元素
MON_GET_ACTIVITY_DETAILS 表函数 - 获取完整的活动详细信息（在 XML 文档 DETAILS 中报告）	ACTIVITY METRICS BASE
MON_GET_CONNECTION 表函数 - 获取连接度量值	REQUEST METRICS BASE
MON_GET_CONNECTION_DETAILS 表函数 - 获取详细的连接度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE
MON_GET_PKG_CACHE_STMT 表函数 - 获取程序包高速缓存中的 SQL 语句活动度量值	ACTIVITY METRICS BASE

表 257. 表函数监视信息 (续)

表函数	监视元素收集级别
MON_GET_PKG_CACHE_STMT_DETAILS 表函数 - 获取程序包高速缓存条目的详细度量值	ACTIVITY METRICS BASE
MON_GET_SERVICE_SUBCLASS 表函数 - 获取服务子类度量值	REQUEST METRICS BASE
MON_GET_SERVICE_SUBCLASS_DETAILS 表函数 - 获取详细的服务子类度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE
MON_GET_UNIT_OF_WORK 表函数 - 获取工作单元度量值	REQUEST METRICS BASE
MON_GET_UNIT_OF_WORK 表函数 - 获取详细的工作单元度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE
MON_GET_WORKLOAD 表函数 - 获取工作负载度量值	REQUEST METRICS BASE
MON_GET_WORKLOAD_DETAILS 表函数 - 获取详细的工作负载度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE

表 258. 事件监视信息

事件类型	逻辑数据分组	监视开关
活动	event_activity（在 details_xml 文档中报告）	ACTIVITY METRICS BASE
活动	event_activitymetrics	ACTIVITY METRICS BASE
统计信息	event_scstats（在 details_xml 文档中报告）	REQUEST METRICS BASE
统计信息	event_wlstats（在 details_xml 文档中报告）	REQUEST METRICS BASE
工作单元	在 system_metrics 文档中报告。	REQUEST METRICS BASE
程序包高速缓存	在 activity_metrics 文档中报告。	ACTIVITY METRICS BASE

用法

使用此监视元素来确定代理程序等待以同步方式打开审计事件并将其写入磁盘时耗用的时间。

在典型情况下，每次将只有一个代理程序尝试打开审计日志文件，这是因为，其他代理程序在打开该文件前将等待访问公共审计子系统。因此，此等待时间通常代表等待操作系统将该文件写入磁盘时耗用的时间。审计实用程序可能会在执行期间锁定审计日志文件，这将导致代理程序打开并写审计日志文件时的等待时间超出正常情况。如果已启用异步审计功能，那么大于异步审计缓冲区大小的审计事件将被直接写入磁盘而不是写入缓冲区，这将导致等待时间延长。

除特殊的审计实用程序情况以外，等待时间取决于磁盘速度以及操作系统将数据写入磁盘的及时性。对于给定的应用程序和审计配置，要缩短此等待时间，您可以调整操作系统或者使用更快的磁盘。

audit_file_writes_total -“写审计文件总次数”监视元素

代理程序被迫等待将审计事件直接写入磁盘的总次数。

表 259. 表函数监视信息

表函数	监视元素收集级别
MON_FORMAT_XML_METRICS_BY_ROW - 获取所有度量值的基于已格式化行的输出	不适用；报告 XML 文档中作为格式化函数的输入提供的所有元素
MON_GET_ACTIVITY_DETAILS 表函数 - 获取完整的活动详细信息（在 XML 文档 DETAILS 中报告）	ACTIVITY METRICS BASE
MON_GET_CONNECTION 表函数 - 获取连接度量值	REQUEST METRICS BASE
MON_GET_CONNECTION_DETAILS 表函数 - 获取详细的连接度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE
MON_GET_PKG_CACHE_STMT 表函数 - 获取程序包高速缓存中的 SQL 语句活动度量值	ACTIVITY METRICS BASE
MON_GET_PKG_CACHE_STMT_DETAILS 表函数 - 获取程序包高速缓存条目的详细度量值	ACTIVITY METRICS BASE
MON_GET_SERVICE_SUBCLASS 表函数 - 获取服务子类度量值	REQUEST METRICS BASE
MON_GET_SERVICE_SUBCLASS_DETAILS 表函数 - 获取详细的服务子类度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE
MON_GET_UNIT_OF_WORK 表函数 - 获取工作单元度量值	REQUEST METRICS BASE
MON_GET_UNIT_OF_WORK 表函数 - 获取详细的工作单元度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE
MON_GET_WORKLOAD 表函数 - 获取工作负载度量值	REQUEST METRICS BASE
MON_GET_WORKLOAD_DETAILS 表函数 - 获取详细的工作负载度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE

表 260. 事件监视信息

事件类型	逻辑数据分组	监视开关
活动	event_activity（在 details_xml 文档中报告）	ACTIVITY METRICS BASE
活动	event_activitymetrics	ACTIVITY METRICS BASE
统计信息	event_scstats（在 details_xml 文档中报告）	REQUEST METRICS BASE

表 260. 事件监视信息 (续)

事件类型	逻辑数据分组	监视开关
统计信息	event_wlstats（在 details_xml 文档中报告）	REQUEST METRICS BASE
工作单元	在 system_metrics 文档中报告。	REQUEST METRICS BASE
程序包高速缓存	在 activity_metrics 文档中报告。	ACTIVITY METRICS BASE

用法

通过将此监视元素与 **audit_file_write_wait_time** 监视元素配合使用，可以确定应用程序请求等待以同步方式打开审计事件并将其写入磁盘时的平均耗用时间。

audit_subsystem_wait_time -“审计子系统等待时间”监视元素

等待审计缓冲区空间时耗用的时间。当审计缓冲区已满，并且代理程序必须等待审计守护程序将缓冲区内容写入磁盘时，就会发生这种等待情况。此值以毫秒计。

表 261. 表函数监视信息

表函数	监视元素收集级别
MON_FORMAT_XML_METRICS_BY_ROW - 获取所有度量值的基于已格式化行的输出	不适用; 报告 XML 文档中作为格式化函数的输入提供的所有元素
MON_FORMAT_XML_TIMES_BY_ROW - - 获取基于已格式化行的组合层次结构等待时间和处理时间	不适用; 报告 XML 文档中作为格式化函数的输入提供的所有元素
MON_FORMAT_XML_WAIT_TIMES_BY _ROW - 获取等待时间的已格式化的基于行的输出	不适用; 报告 XML 文档中作为格式化函数的输入提供的所有元素
MON_GET_ACTIVITY_DETAILS 表函数 - 获取完整的活动详细信息（在 XML 文档 DETAILS 中报告）	ACTIVITY METRICS BASE
MON_GET_CONNECTION 表函数 - 获取连接度量值	REQUEST METRICS BASE
MON_GET_CONNECTION_DETAILS 表函数 - 获取详细的连接度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE
MON_GET_PKG_CACHE_STMT 表函数 - 获取程序包高速缓存中的 SQL 语句活动度量值	ACTIVITY METRICS BASE
MON_GET_PKG_CACHE_STMT_DETAILS 表函数 - 获取程序包高速缓存条目的详细度量值	ACTIVITY METRICS BASE
MON_GET_SERVICE_SUBCLASS 表函数 - 获取服务子类度量值	REQUEST METRICS BASE
MON_GET_SERVICE_SUBCLASS_DETAILS 表函数 - 获取详细的服务子类度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE
MON_GET_UNIT_OF_WORK 表函数 - 获取工作单元度量值	REQUEST METRICS BASE

表 261. 表函数监视信息 (续)

表函数	监视元素收集级别
MON_GET_UNIT_OF_WORK 表函数 - 获取详细的工作单元度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE
MON_GET_WORKLOAD 表函数 - 获取工作负载度量值	REQUEST METRICS BASE
MON_GET_WORKLOAD_DETAILS 表函数 - 获取详细的工作负载度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE

表 262. 事件监视信息

事件类型	逻辑数据分组	监视开关
活动	event_activity（在 details_xml 文档中报告）	ACTIVITY METRICS BASE
活动	event_activitymetrics	ACTIVITY METRICS BASE
统计信息	event_scstats（在 details_xml 文档中报告）	REQUEST METRICS BASE
统计信息	event_wlstats（在 details_xml 文档中报告）	REQUEST METRICS BASE
工作单元	在 system_metrics 文档中报告。	REQUEST METRICS BASE
程序包高速缓存	在 activity_metrics 文档中报告。	ACTIVITY METRICS BASE

用法

使用此监视元素来确定代理程序等待访问公共审计子系统时耗用的时间，在这段时间内，公共审计子系统正忙于为其他代理程序处理事件。

审计子系统的某些公共部分每次只能由一个代理程序访问。此监视元素的值指示代理程序访问公共审计子系统时必须等待的时间。这包括填充当前异步缓冲区的代理程序在等待审计守护程序将先前异步缓冲区写入磁盘完成时耗用的时间。其他正在等待写审计日志文件或者正在等待发出审计守护程序请求的代理程序也已访问公共审计子系统，并且其等待时间也将在此值中反映。

如果正在使用异步审计功能，那么可以通过更改 **audit_buf_sz** 配置参数的值来缩短此等待时间。您可以不断增大 **audit_buf_sz** 配置参数的值，直到进一步增大此值不再能够缩短公共审计子系统等待时间为止。在这个点，异步缓冲区大小已足以保证守护程序能够在下一个缓冲区变满前将一个完整的缓冲区写入磁盘，因此守护程序不再是瓶颈。如果必须将 **audit_buf_sz** 配置参数的值增大到发生系统故障可能会导致过多审计记录丢失的程度，那么可以通过调整操作系统或使用更高速的磁盘来缩短等待时间。如果有必要进一步缩短等待时间，请使用审计策略来减少所生成的审计事件的数目。

audit_subsystem_waits_total -“审计子系统等待总次数”监视元素

审计子系统等待缓冲区写操作完成的次数。

表 263. 表函数监视信息

表函数	监视元素收集级别
MON_FORMAT_XML_METRICS_BY_ROW - 获取所有度量值的基于已格式化行的输出	不适用; 报告 XML 文档中作为格式化函数的输入提供的所有元素
MON_GET_ACTIVITY_DETAILS 表函数 - 获取完整的活动详细信息（在 XML 文档 DETAILS 中报告）	ACTIVITY METRICS BASE
MON_GET_CONNECTION 表函数 - 获取连接度量值	REQUEST METRICS BASE
MON_GET_CONNECTION_DETAILS 表函数 - 获取详细的连接度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE
MON_GET_PKG_CACHE_STMT 表函数 - 获取程序包高速缓存中的 SQL 语句活动度量值	ACTIVITY METRICS BASE
MON_GET_PKG_CACHE_STMT_DETAILS 表函数 - 获取程序包高速缓存条目的详细度量值	ACTIVITY METRICS BASE
MON_GET_SERVICE_SUBCLASS 表函数 - 获取服务子类度量值	REQUEST METRICS BASE
MON_GET_SERVICE_SUBCLASS_DETAILS 表函数 - 获取详细的服务子类度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE
MON_GET_UNIT_OF_WORK 表函数 - 获取工作单元度量值	REQUEST METRICS BASE
MON_GET_UNIT_OF_WORK 表函数 - 获取详细的工作单元度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE
MON_GET_WORKLOAD 表函数 - 获取工作负载度量值	REQUEST METRICS BASE
MON_GET_WORKLOAD_DETAILS 表函数 - 获取详细的工作负载度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE

表 264. 事件监视信息

事件类型	逻辑数据分组	监视开关
活动	event_activity（在 details_xml 文档中报告）	ACTIVITY METRICS BASE
活动	event_activitymetrics	ACTIVITY METRICS BASE
统计信息	event_scstats（在 details_xml 文档中报告）	REQUEST METRICS BASE
统计信息	event_wlstats（在 details_xml 文档中报告）	REQUEST METRICS BASE
工作单元	在 system_metrics 文档中报告。	REQUEST METRICS BASE

表 264. 事件监视信息 (续)

事件类型	逻辑数据分组	监视开关
程序包高速缓存	在 activity_metrics 文档中报告。	ACTIVITY METRICS BASE

用法

使用此监视元素来确定代理程序在访问公共审计子系统时被迫等待的总次数。在生成一个审计事件时，可能不需要访问公共审计子系统，也可以需要访问该子系统一次或多次以记录该事件。请使用 **audit_events_total** 监视元素来确定所生成的审计事件的准确数目。

auth_id -“授权标识”

调用受监视应用程序的用户的授权标识。在 DB2 Connect 网关节点上，这是用户在主机上的授权标识。

表 265. 表函数监视信息

表函数	监视元素收集级别
PD_GET_DIAG_HIST 表函数 - 从给定设施返回记录	ACTIVITY METRICS BASE

表 266. 快照监视信息

快照级别	逻辑数据分组	监视开关
应用程序	appl_id_info	基本
锁定	appl_lock_list	基本
DCS 应用程序	dcs_appl_info	基本

表 267. 事件监视信息

事件类型	逻辑数据分组	监视开关
锁定	-	始终收集
工作单元	-	始终收集
连接	event_connheader	始终收集

用法

在显式的可信连接中，当您切换用户时，**auth_id** 值不会立即更改。而是在您切换用户之后首次访问数据库时，才会更新 **auth_id**。这是因为切换用户操作始终会影响到后续操作。

可使用此元素来确定调用该应用程序的人员。

authority_bitmap -“用户权限级别”监视元素

对用户及用户所属的组授予的权限。这些权限包括授予特定角色的权限，该角色是授予该用户及其所属组的角色。对用户或授予用户的角色授予的权限被视为用户权限。对用户所属组或授予用户所属组的角色授予的权限被视为组权限。

表 268. 快照监视信息

快照级别	逻辑数据分组	监视开关
应用程序	appl	基本
应用程序	appl_info	基本

表 269. 事件监视信息

事件类型	逻辑数据分组	监视开关
连接	event_conn	始终收集

用法

authority_bitmap 监视元素的格式为数组格式。每个数组元素为一个字符，表示是否对用户标识授予了特定权限以及用户获得该权限的方式。

各个数组元素通过在 `sql.h` 文件中定义的下标值来创建下标。authority_bitmap 数组中的下标值称为*权限下标*。例如，SQL_DBAUTH_SYSADM 是确定用户是否拥有 SYSADM 权限的下标。

authority_bitmap 数组中由权限下标标识的一个元素的值表示授权标识是否拥有该权限。要确定授权标识如何拥有权限，对由授权下标标识的每个数组元素使用来自 sql.h 的下列定义:

SQL_AUTH_ORIGIN_USER
如果此位为 on，那么表示该授权标识拥有授予该用户或其角色的权限。

SQL_AUTH_ORIGIN_GROUP
如果此位为 on，那么表示该授权标识拥有授予该用户或其角色的权限。

例如，要确定用户是否具有 DBADM 权限，验证以下值:

```
authority_bitmap[SQL_DBAUTH_DBADM]
```

要确定用户是否直接拥有 DBADM 权限，验证:

```
authority_bitmap[SQL_DBAUTH_DBADM] & SQL_AUTH_ORIGIN_USER
```

authority_lvl -“用户权限级别”监视元素

授予应用程序的最高权限级别。

注: 从 DB2 数据库 V9.5 开始，不推荐使用 authority_lvl 监视元素。请改为使用 authority_bitmap 监视元素。请参阅『authority_bitmap -“用户权限级别”监视元素』。

表 270. 快照监视信息

快照级别	逻辑数据分组	监视开关
应用程序	appl	基本

表 270. 快照监视信息 (续)

快照级别	逻辑数据分组	监视开关
应用程序	appl_info	基本

表 271. 事件监视信息

事件类型	逻辑数据分组	监视开关
连接	event_conn	始终收集

用法 直接或间接授权进行应用程序允许的操作。

下面的定义来自 sql.h，可用来确定显式授予用户的权限:

- SQL_SYSADM
- SQL_DBADM
- SQL_CREATETAB
- SQL_BINDADD
- SQL_CONNECT
- SQL_CREATE_EXT_RT
- SQL_CREATE_NOT_FENC
- SQL_SYSCTRL
- SQL_SYSMAINT

下面的定义来自 sql.h，可用来确定从组或公用继承的间接权限:

- SQL_SYSADM_GRP
- SQL_DBADM_GRP
- SQL_CREATETAB_GRP
- SQL_BINDADD_GRP
- SQL_CONNECT_GRP
- SQL_CREATE_EXT_RT_GRP
- SQL_CREATE_NOT_FENC_GRP
- SQL_SYSCTRL_GRP
- SQL_SYSMAINT_GRP

auto_storage_hybrid -“混合自动存储器表空间指示器”监视元素

如果表空间是包含一些非自动存储器容器的自动存储器表空间，那么此监视元素将返回值 1。否则，它将返回值 0。

表 272. 表函数监视信息

表函数	监视元素收集级别
MON_GET_TABLESPACE 表函数 - 获取表空间度量值	ACTIVITY METRICS BASE

用法

混合自动存储器表空间是已使用 ALTER TABLESPACE 命令进行转换以便由自动存储器管理，但尚未进行重新平衡的表空间。此表空间仍包含非自动存储器容器。对此表空间进行重新平衡之后，它将只包含自动存储器容器，并且不再被视为混合表空间。

automatic -“自动调整缓冲池”监视元素

指示是否已对特定缓冲池启用自调整功能。如果已对此缓冲池启用自调整功能，那么此元素设置为 1，否则设置为 0。

表 273. 表函数监视信息

表函数	监视元素收集命令和级别
MON_GET_BUFFERPOOL 表函数 - 获取缓冲池度量值	DATA OBJECT METRICS BASE

backup_timestamp -“备份时间戳记”

备份映像的时间戳记。

表 274. 事件监视信息

事件类型	逻辑数据分组	监视元素收集级别
变更历史记录	changesummary	始终收集

用法

对于变更历史记录事件监视器:

- 如果 UTILITY_TYPE 为 BACKUP 且 EVENT_TYPE 为 UTILSTART，那么 BACKUP_TIMESTAMP 值是备份映像的时间戳记。如果 UTILITY_TYPE 为 RESTORE 且 EVENT_TYPE 为 UTILSTOP，那么 BACKUP_TIMESTAMP 值是备份映像的时间戳记。对于所有其他情况，BACKUP_TIMESTAMP 为空字符串。
- 对于 RESTORE，映像时间戳记在实用程序启动时并非始终已知。

BACKUP_TIMESTAMP 可通过使用 SYSIBMADM.DB_HISTORY 管理视图来与存储在数据库历史记录文件中的信息（例如，查询顺序信息）相关联

bin_id -“直方图条形标识”监视元素

直方图条形标识。**bin_id** 在直方图内是唯一的。

表 275. 事件监视信息

事件类型	逻辑数据分组	监视开关
统计信息	event_histogrambin	-

用法

使用此元素来区分同一直方图内的条形。

binds_precompiles -“尝试的绑定次数/预编译次数”

尝试的绑定次数和预编译次数。

元素标识

binds_precompiles

元素类型

计数器

表 276. 快照监视信息

快照级别	逻辑数据分组	监视开关
数据库	dbase	基本
应用程序	appl	基本

可将快照监视的计数器重置。

表 277. 事件监视信息

事件类型	逻辑数据分组	监视开关
数据库	event_db	始终收集
连接	event_conn	始终收集

用法 可使用此元素来了解数据库管理器内的当前活动级别。

此值不包括 *int_auto_rebinds* 的计数，但它包括因为 REBIND PACKAGE 命令而产生的绑定次数。

block_ios -“块 I/O 请求数”监视元素

块 I/O 请求的数目。更具体而言，就是 DB2 在缓冲池的块区域中执行顺序页预取的次数。

表 278. 表函数监视信息

表函数	监视元素收集级别
MON_GET_BUFFERPOOL 表函数 - 获取缓冲池度量值	DATA OBJECT METRICS BASE
MON_GET_CONTAINER 表函数 - 获取表空间容器度量值	REQUEST METRICS BASE
MON_GET_TABLESPACE 表函数 - 获取表空间度量值	DATA OBJECT METRICS BASE

表 279. 快照监视信息

快照级别	逻辑数据分组	监视开关
缓冲池	bufferpool	缓冲池

用法

如果启用了基于块的缓冲池，那么此监视元素将报告执行块 I/O 的频率。否则，此监视元素将返回 0。在使用基于块的缓冲池时，只有在顺序预取期间才监视块 I/O 请求的数目。

如果已启用基于块的缓冲池，并且此数目很低或者接近于向量 I/O 数（**vectored_ios** 监视元素的值），那么请考虑更改块大小。此状态可能指示下列事件中的一件:

- 一个或多个与缓冲池绑定的表空间的扩展数据块大小小于对缓冲池指定的块大小。
- 预取请求中请求的某些页已存在于缓冲池的页区域中。

预取程度允许在每个缓冲池块中浪费一些页，但如果浪费的页数过多，那么预取程序将决定在缓冲池的页区域中执行向量 I/O。

为了更好地利用基于块的缓冲池提供的顺序预取性能改进，应对块大小选择适当的值。但是，因为带有不同扩展数据块大小的多个表空间可能与同一个基于块的缓冲池绑定，所以这一点可能比较难以做到。为了获取最佳性能，建议将具有相同扩展数据块大小的表空间与一个基于块的缓冲池绑定，该缓冲池的块大小等于扩展数据块大小。如果表空间的扩展数据块大小大于块大小，那么可以获得较好的性能，扩展数据块大小小于块大小时情况则相反。

例如，如果扩展数据块大小为 2 而块大小为 8，那么将使用向量 I/O 而不是块 I/O（块 I/O 会浪费 6 页）。将块大小降低至 2 将解决此问题。

blocking_cursor -“分块游标”

此元素指示要执行的语句是否在使用分块游标。

元素标识

blocking_cursor

元素类型

信息

表 280. 快照监视信息

快照级别	逻辑数据分组	监视开关
应用程序	stmt	语句
DCS 语句	dcs_stmt	语句

表 281. 事件监视信息

事件类型	逻辑数据分组	监视开关
带有详细信息的死锁	event_detailed_dlconn	始终收集
语句	event_stmt	始终收集

用法 对查询的数据传输使用分块可以改进性能。用于查询的 SQL 会影响分块的使用并且可能需要一些修改。

blocks_pending_cleanup -“暂挂清除已转出块”监视元素

数据库中继滚出删除后暂挂异步清除的 MDC 表块数。

表 282. 表函数监视信息

表函数	监视元素收集级别
ADMINTABINFO 管理视图和 ADMIN_GET_TAB_INFO 表函数 - 检索表大小和状态信息	ACTIVITY METRICS BASE

表 283. 快照监视信息

快照级别	逻辑数据分组	监视开关
数据库	dbase	-
数据库	event_db	-

用法

使用此元素来确定删除延迟清除滚出后，未作为可用存储释放回系统的 MDC 表块数。

bottom -“直方图类别底部”监视元素

直方图类别范围的底部（该范围不含该底部值）。此监视元素的值也是上一直方图类别（如果有）的范围的包含顶端。

表 284. 事件监视信息

事件类型	逻辑数据分组	监视开关
统计信息	event_histogrambin	-

用法

将此元素与相应的 **top** 元素配合使用来确定直方图内的类别范围。

boundary_leaf_node_splits -“边界叶节点分割次数”监视元素

插入操作期间分割边界叶节点的次数。

表 285. 表函数监视信息

表函数	监视元素收集级别
MON_GET_INDEX 表函数 - 获取索引度量值	ACTIVITY METRICS BASE

bp_cur_buffsz -“缓冲池的当前大小”

当前缓冲池大小（以页计）。

表 286. 表函数监视信息

表函数	监视元素收集级别
MON_GET_BUFFERPOOL 表函数 - 获取缓冲池度量值	ACTIVITY METRICS BASE

表 287. 快照监视信息

快照级别	逻辑数据分组	监视开关
缓冲池	bufferpool_nodeinfo	缓冲池

bp_id -“缓冲池标识”监视元素

此元素包含正在监视的缓冲池的缓冲池标识。

表 288. 快照监视信息

快照级别	逻辑数据分组	监视开关
缓冲池	bufferpool	基本

bp_name -“缓冲池名称”监视元素

缓冲池的名称。

表 289. 表函数监视信息

表函数	监视元素收集命令和级别
MON_GET_BUFFERPOOL 表函数 - 获取缓冲池度量值	ACTIVITY METRICS BASE

表 290. 快照监视信息

快照级别	逻辑数据分组	监视开关
缓冲池	bufferpool	基本

用法 每个数据库都至少需要一个缓冲池。根据您的需要，可以选择对单个数据库创建若干个大小不同的缓冲池。CREATE、ALTER 和 DROP BUFFERPOOL 语句允许您创建、更改或除去缓冲池。

创建新数据库后，它将具有缺省缓冲池 IBMDEFAULTBP，其大小将由平台确定。它还会具有一组系统缓冲池，每个系统缓冲池对应不同页大小:

- IBMSYSTEMBP4K
- IBMSYSTEMBP8K
- IBMSYSTEMBP16K
- IBMSYSTEMBP32K

不能更改这些系统缓冲池。

bp_new_buffsz -“新的缓冲池大小”

一旦重新启动数据库后缓冲池将更改至的大小。当以 DEFERRED 方式执行 ALTER BUFFERPOOL 语句时，在停止并重新启动数据库之前，缓冲池大小不会更改。

表 291. 快照监视信息

快照级别	逻辑数据分组	监视开关
缓冲池	bufferpool_nodeinfo	缓冲池

bp_pages_left_to_remove -“要除去的余下页数”

在完成缓冲池调整大小之前，缓冲池中要除去的余下页数。此项仅适用于以 IMMEDIATE 方式执行的 ALTER BUFFERPOOL 语句调用的缓冲池调整大小操作。

表 292. 快照监视信息

快照级别	逻辑数据分组	监视开关
缓冲池	bufferpool_nodeinfo	缓冲池

bp_tbsp_use_count -“映射至缓冲池的表空间数”

使用此缓冲池的表空间数。

表 293. 快照监视信息

快照级别	逻辑数据分组	监视开关
缓冲池	bufferpool_nodeinfo	缓冲池

buff_auto_tuning -“FCM 缓冲区自动调整指示器”监视元素

指示是否自动设置和调整快速通信管理器 (FCM) 缓冲区的数目。值为 1 表示“是”，值为 0 表示“否”。

表 294. 表函数监视信息

表函数	监视元素收集级别
MON_GET_FCM - 获取 FCM 度量值	ACTIVITY METRICS BASE

用法

通过将 **fcm_num_buffers** 配置参数设置为 AUTOMATIC，即可启用自动调整 FCM 缓冲区。

buff_free -“当前可用的 FCM 缓冲区数”

此元素指示当前可用的 FCM 缓冲区数。

元素标识

buff_free

元素类型

标尺

表 295. 表函数监视信息

表函数	监视元素收集级别
MON_GET_FCM - 获取 FCM 度量值	ACTIVITY METRICS BASE

表 296. 快照监视信息

快照级别	逻辑数据分组	监视开关
数据库管理器	fcm	基本

用法

要计算可用 FCM 缓冲区百分比，请使用以下公式:

```
(buff_free/buff_total) * 100
```

如果可用 FCM 缓冲区的百分比低于 20% 且如果启用了 FCM 缓冲区自动调整，那么 DB2 数据库管理器将调整 FCM 缓冲区数。

如果可用 FCM 缓冲区的百分比低于 20% 且如果未启用 FCM 缓冲区自动调整，那么您需要调整 **fcm_num_buffers** 配置参数。

buff_free_bottom -“最少可用 FCM 缓冲区数”

处理期间达到的可用 FCM 缓冲区的最小数目。

表 297. 表函数监视信息

表函数	监视元素收集级别
MON_GET_FCM - 获取 FCM 度量值	ACTIVITY METRICS BASE

表 298. 快照监视信息

快照级别	逻辑数据分组	监视开关
数据库管理器	fcm	基本

用法

将此元素与 **fcm_num_buffers** 配置参数一起使用来确定最大 FCM 缓冲池利用率。如果 **buff_free_bottom** 监视元素的值较小，那么请增大 **fcm_num_buffers** 配置参数的值，以确保操作不会用尽 FCM 缓冲区。如果 **buff_free_bottom** 监视元素的值较大，那么请减小 **fcm_num_buffers** 配置参数的值，以节省系统资源。

buff_max -“FCM 缓冲区可能达到的最大数目”监视元素

当实例启动时，可以根据保留的虚拟内存量来分配的快速通信管理器 (FCM) 缓冲区的最大数目。

表 299. 表函数监视信息

表函数	监视元素收集级别
MON_GET_FCM - 获取 FCM 度量值	ACTIVITY METRICS BASE

表 300. 快照监视信息

快照级别	逻辑数据分组	监视开关
数据库管理器	fcm	基本

用法

此内部监视元素仅由 IBM 支持机构使用。

buff_total -“当前已分配的 FCM 缓冲区数目”监视元素

当前已分配的快速通信管理器 (FCM) 缓冲区的数目。此数目既包括正在使用的缓冲区，又包括可用缓冲区。

表 301. 表函数监视信息

表函数	监视元素收集级别
MON_GET_FCM - 获取 FCM 度量值	ACTIVITY METRICS BASE

表 302. 快照监视信息

快照级别	逻辑数据分组	监视开关
数据库管理器	fcm	基本

用法

如果 **buff_auto_tuning** 监视元素指示要自动调整 FCM，那么会根据对于 FCM 缓冲区的需求来调整 **buff_total** 监视元素的值。

要确定当前使用的 FCM 缓冲区的数量，请使用以下公式:

```
buff_total - buff_free
```

要计算可用 FCM 缓冲区百分比，请使用以下公式:

```
(buff_free/buff_total) * 100
```

如果可用 FCM 缓冲区的百分比低于 20% 且如果启用了 FCM 缓冲区自动调整，那么 DB2 数据库管理器将调整 FCM 缓冲区数。

如果可用 FCM 缓冲区的百分比低于 20% 且如果未启用 FCM 缓冲区自动调整，那么您需要调整 **fcm_num_buffers** 配置参数。

byte_order -“事件数据的字节顺序”

数字数据的字节定序，具体而言是在“大尾数法”服务器（如 RS/6000®）还是“小尾数法”服务器（如基于 Intel 并且运行 Windows 2000 的 PC）上生成事件数据流。

表 303. 事件监视信息

事件类型	逻辑数据分组	监视开关
事件日志头	event_log_header	始终收集

用法 因为“大尾数法”服务器上的整数字节顺序与“小尾数法”服务器上的字节顺序方向相反，所以必须使用此信息以允许您解释数据流中的数字数据。

如果处理数据的应用程序识别它在一种类型的计算机硬件（如大尾数法计算机）上运行，而事件数据是在另一种类型的计算机硬件（如小尾数法计算机）上生成的，那么监视应用程序必须先使数字数据字段的字节反向，然后再解释它们。否则不需要进行字节定向。

此元素可设置为下列其中一种 API 常量:

- SQLM_BIG_ENDIAN
- SQLM_LITTLE_ENDIAN

cached_timestamp -“高速缓存时间戳记”监视元素

服务器列表进行高速缓存的时间。

表 304. 表函数监视信息

表函数	监视元素收集级别
MON_GET_SERVERLIST 表函数 - 获取成员优先级详细信息	始终收集

cat_cache_inserts -“目录高速缓存插入数”监视元素

系统尝试将表描述符或权限信息插入到目录高速缓存中的次数。

表 305. 表函数监视信息

表函数	监视元素收集级别
MON_FORMAT_XML_METRICS_BY_ROW - 获取所有度量值的基于已格式化行的输出	不适用; 报告 XML 文档中作为格式化函数的输入提供的所有元素
MON_GET_CONNECTION 表函数 - 获取连接度量值	REQUEST METRICS BASE
MON_GET_CONNECTION_DETAILS 表函数 - 获取详细的连接度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE
MON_GET_SERVICE_SUBCLASS 表函数 - 获取服务子类度量值	REQUEST METRICS BASE
MON_GET_SERVICE_SUBCLASS_DETAILS 表函数 - 获取详细的服务子类度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE

表 305. 表函数监视信息 (续)

表函数	监视元素收集级别
MON_GET_UNIT_OF_WORK 表函数 - 获取工作单元度量值	REQUEST METRICS BASE
MON_GET_UNIT_OF_WORK 表函数 - 获取详细的工作单元度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE
MON_GET_WORKLOAD 表函数 - 获取工作负载度量值	REQUEST METRICS BASE
MON_GET_WORKLOAD_DETAILS 表函数 - 获取详细的工作负载度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE

表 306. 快照监视信息

快照级别	逻辑数据分组	监视开关
数据库	dbase	基本
应用程序	appl	基本

可将快照监视的计数器重置。

表 307. 事件监视信息

事件类型	逻辑数据分组	监视开关
数据库	event_db	始终收集
连接	event_conn	始终收集
统计信息	event_scstats（在 details_xml 文档中报告）	REQUEST METRICS BASE
统计信息	event_wlstats（在 details_xml 文档中报告）	REQUEST METRICS BASE
工作单元	在 system_metrics 文档中报告。	REQUEST METRICS BASE

用法

通过与“目录高速缓存查询”一起使用，可借助以下公式来计算目录高速缓存命中率:

```
1 - (目录高速缓存插入数 / 目录高速缓存查询数)
```

有关使用此元素的更多信息，请参阅 **cat_cache_lookups** 监视元素。

cat_cache_lookups -“目录高速缓存查询数”监视元素

为获取表描述符信息或权限信息而引用目录高速缓存的次数。

表 308. 表函数监视信息

表函数	监视元素收集级别
MON_FORMAT_XML_METRICS_BY_ROW - 获取所有度量值的基于已格式化行的输出	不适用; 报告 XML 文档中作为格式化函数的输入提供的所有元素

表 308. 表函数监视信息 (续)

表函数	监视元素收集级别
MON_GET_CONNECTION 表函数 - 获取连接度量值	REQUEST METRICS BASE
MON_GET_CONNECTION_DETAILS 表函数 - 获取详细的连接度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE
MON_GET_SERVICE_SUBCLASS 表函数 - 获取服务子类度量值	REQUEST METRICS BASE
MON_GET_SERVICE_SUBCLASS_DETAILS 表函数 - 获取详细的服务子类度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE
MON_GET_UNIT_OF_WORK 表函数 - 获取工作单元度量值	REQUEST METRICS BASE
MON_GET_UNIT_OF_WORK 表函数 - 获取详细的工作单元度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE
MON_GET_WORKLOAD 表函数 - 获取工作负载度量值	REQUEST METRICS BASE
MON_GET_WORKLOAD_DETAILS 表函数 - 获取详细的工作负载度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE

表 309. 快照监视信息

快照级别	逻辑数据分组	监视开关
数据库	dbase	基本
应用程序	appl	基本

可将快照监视的计数器重置。

表 310. 事件监视信息

事件类型	逻辑数据分组	监视开关
数据库	event_db	始终收集
连接	event_conn	始终收集
统计信息	event_scstats（在 details_xml 文档中报告）	REQUEST METRICS BASE
统计信息	event_wlstats（在 details_xml 文档中报告）	REQUEST METRICS BASE
工作单元	在 system_metrics 文档中报告。	REQUEST METRICS BASE

用法

此元素包括对目录高速缓存的成功访问和不成功访问。每当出现下列情况，就会引用目录高速缓存:

- 在编译 SQL 语句期间处理表、视图或别名
- 访问数据库权限信息

- 在编译 SQL 语句期间处理例程

要计算目录高速缓存命中率，请使用以下公式:

```
(1 - (cat_cache_inserts / cat_cache_lookups))
```

来指示目录高速缓存避免目录访问的效果如何。如果比率很高（超过 0.8），那么表示高速缓存确实起到作用。如果比率偏低，那么表示应增大 **catalogcache_sz** 配置参数。首次连接至数据库之后应该有较大的比率。

执行涉及表、视图或别名的数据定义语言（DDL）SQL 语句会从目录高速缓存中除去该对象的表描述符信息，从而导致这些信息在下一次引用时重新插入。此外，用于数据库权限和例程执行特权的 GRANT 和 REVOKE 语句会从目录高速缓存中除去主题权限信息。因此，大量使用 DDL 语句和 GRANT/REVOKE 语句也会提高该比率。

cat_cache_overflows -“目录高速缓存溢出数”

目录高速缓存溢出其分配内存边界的次数。

表 311. 快照监视信息

快照级别	逻辑数据分组	监视开关
数据库	dbase	基本
应用程序	appl	基本

可将快照监视的计数器重置。

表 312. 事件监视信息

事件类型	逻辑数据分组	监视开关
数据库	event_db	始终收集
连接	event_conn	始终收集

用法

将此元素与 **cat_cache_size_top** 监视元素配合使用来确定是否需要增加目录高速缓存的大小以避免溢出。

目录高速缓存空间是通过除去表、视图或别名的表描述符信息或当前未被任何事务使用的权限信息来回收的。

如果 **cat_cache_overflows** 监视元素的值很大，那么相对工作负载而言目录高速缓存可能会太小。扩大目录高速缓存可以改进性能。如果工作负载包括的事务将编译大量 SQL 语句，而这些语句又引用单个工作单元中的多个表、视图、别名、用户定义的函数或存储过程，那么在单个事务中编译较少的 SQL 语句可以改进目录高速缓存的性能。或者，如果工作负载包括绑定包含许多 SQL 语句的程序包，而这些语句又引用多个表、视图、别名、用户定义的函数或存储过程，那么可以尝试分割程序包以使这些程序包包括较少的 SQL 语句从而改进性能。

cat_cache_size_top -“目录高速缓存高水位标记”监视元素

目录高速缓存达到最大逻辑大小。

表 313. 快照监视信息

快照级别	逻辑数据分组	监视开关
数据库	dbase	基本

表 314. 事件监视信息

事件类型	逻辑数据分组	监视开关
数据库	event_db	始终收集

用法

此元素指示在激活数据库之后，对其运行的工作负载在逻辑上所需的目录高速缓存的最大字节数。

目录高速缓存通过逻辑大小进行管理，此大小不包括内存管理使用情况。数据库快照中的 **pool_watermark** 元素提供了目录高速缓存所使用的内存的物理高水位标记值。执行目录高速缓存监视和调整工作时，应该使用逻辑大小而不是物理大小。

如果目录高速缓存溢出，那么表示此元素包含溢出期间目录高速缓存达到的最大大小。检查 **cat_cache_overflows** 监视元素以确定是否发生此类情况。

可通过以下公式来确定工作负载需要的目录高速缓存最小大小:

```
最大目录高速缓存大小 / 4096
```

通过将结果四舍五入为整数，指示为避免溢出目录高速缓存最少需要的页数（每页 4K 字节）。

catalog_node -“目录节点号”

存储数据库目录表的节点的编号。

元素标识

catalog_node

元素类型

信息

表 315. 快照监视信息

快照级别	逻辑数据分组	监视开关
数据库	dbase	基本

表 316. 事件监视信息

事件类型	逻辑数据分组	监视开关
数据库	event_db	始终收集

用法 目录节点是存储所有系统目录表的节点。对系统目录表的所有访问都必须通过此节点进行。

catalog_node_name -“目录节点网络名”

目录节点的网络名。

元素标识

catalog_node_name

元素类型

信息

表 317. 快照监视信息

快照级别	逻辑数据分组	监视开关
数据库	dbase	基本

表 318. 事件监视信息

事件类型	逻辑数据分组	监视开关
数据库	event_db	始终收集

用法 使用此元素来确定数据库的位置。

cf_waits -“集群高速缓存设施等待次数”监视元素

DB2 数据库系统与集群高速缓存设施通信时等待的次数。

表 319. 表函数监视信息

表函数	监视元素收集级别
MON_FORMAT_XML_METRICS_BY_ROW - 获取所有度量值的基于已格式化行的输出	不适用; 报告 XML 文档中作为格式化函数的输入提供的所有元素
MON_GET_CONNECTION 表函数 - 获取连接度量值	REQUEST METRICS BASE
MON_GET_PKG_CACHE_STMT 表函数 - 获取程序包高速缓存中的 SQL 语句活动度量值	ACTIVITY METRICS BASE
MON_GET_SERVICE_SUBCLASS 表函数 - 获取服务子类度量值	REQUEST METRICS BASE
MON_GET_UNIT_OF_WORK 表函数 - 获取工作单元度量值	REQUEST METRICS BASE
MON_GET_WORKLOAD 表函数 - 获取工作负载度量值	REQUEST METRICS BASE

表 320. 事件监视信息

事件类型	逻辑数据分组	监视开关
活动	event_activity（在 details_xml 文档中报告）	ACTIVITY METRICS BASE
统计信息	event_scstats（在 details_xml 文档中报告）	REQUEST METRICS BASE
统计信息	event_wlstats（在 details_xml 文档中报告）	REQUEST METRICS BASE
工作单元	-	始终收集

表 320. 事件监视信息 (续)

事件类型	逻辑数据分组	监视开关
程序包高速缓存	-	始终收集
锁定	-	始终收集

cf_wait_time -“集群高速缓存设施等待时间”监视元素

与集群高速缓存设施通信所耗的时间量。此时间未包括可能因为授予锁定或执行页回收之类的操作而请求的或因为通信而发生的任何处理所耗的时间。时间以毫秒计。

表 321. 表函数监视信息

表函数	监视元素收集级别
MON_FORMAT_XML_METRICS_BY_ROW - 获取所有度量值的基于已格式化行的输出	不适用; 报告 XML 文档中作为格式化函数的输入提供的所有元素
MON_FORMAT_XML_TIMES_BY_ROW - - 获取基于已格式化行的组合层次结构等待时间和处理时间	不适用; 报告 XML 文档中作为格式化函数的输入提供的所有元素
MON_FORMAT_XML_WAIT_TIMES_BY _ROW - 获取等待次数的已格式化的基于行的输出	不适用; 报告 XML 文档中作为格式化函数的输入提供的所有元素
MON_GET_CONNECTION 表函数 - 获取连接度量值	REQUEST METRICS BASE
MON_GET_PKG_CACHE_STMT 表函数 - 获取程序包高速缓存中的 SQL 语句活动度量值	ACTIVITY METRICS BASE
MON_GET_SERVICE_SUBCLASS 表函数 - 获取服务子类度量值	REQUEST METRICS BASE
MON_GET_UNIT_OF_WORK 表函数 - 获取工作单元度量值	REQUEST METRICS BASE
MON_GET_WORKLOAD 表函数 - 获取工作负载度量值	REQUEST METRICS BASE

用法

此值是 DB2 与集群高速缓存设施通信时消耗在等待上的时间量的指示符。

cfg_collection_type -“配置收集类型”

指示收集配置参数值的时间:

表 322. 事件监视信息

事件类型	逻辑数据分组	监视开关
变更历史记录	DBDBMCFG	始终收集

用法

变更历史记录事件监视器将此值收集为:

I 激活事件监视器时捕获的初始值。

U 已更新值

cfg_name -“配置名称”

配置参数的名称。

表 323. 事件监视信息

事件类型	逻辑数据分组	监视开关
变更历史记录	DBDBMCFG	始终收集

用法

对于变更历史记录事件监视器，此元素标识作为 DBCFG 或 DBMCFG 事件一部分更新的或作为 DBCFGVALUES 或 DBMCFGVALUES 事件一部分在事件监视器启动时捕获的配置参数。这些事件表示发生以下事件:

DBCFG
更改数据库配置参数

DBMCFG
更改数据库管理器配置参数

DBCFGVALUES
在事件监视器启动时捕获数据库配置参数值（如果事件监视器处于不活动状态时数据库配置参数更改）

DBMCFGVALUES
在事件监视器启动时捕获数据库管理器配置参数值（如果事件监视器处于不活动状态时数据库管理器配置参数更改）

cfg_old_value -“配置旧值”

配置参数的旧值或内存中的配置参数值。

表 324. 事件监视信息

事件类型	逻辑数据分组	监视开关
变更历史记录	DBDBMCFG	始终收集

用法

对于变更历史记录事件监视器:

- 如果事件是更改数据库配置参数 (DBCFG) 或数据库管理器配置参数 (DBMCFG)，那么这是旧配置参数值。
- 如果事件是捕获事件监视器处于不活动状态时更改的数据库配置参数值 (DBCFGVALUES) 或数据库管理器配置参数值 (DBMCFGVALUES)，那么这是当前内存中的配置参数值。这是当前正使用的配置参数值。

cfg_old_value_flags -“配置旧值标志”

此标志指示如何确定旧配置参数值。

表 325. 事件监视信息

事件类型	逻辑数据分组	监视开关
变更历史记录	DBDBMCFG	始终收集

用法

对于变更历史记录事件监视器，此元素指示如何确定旧配置参数值:

- AUTOMATIC
- COMPUTED
- NONE

如果事件是捕获事件监视器处于不活动状态时更改的数据库配置参数值 (DBCFGVALUES) 或数据库管理器配置参数值 (DBMCFGVALUES)，那么这些标志表示该配置参数的当前内存中的值。

cfg_value -“配置值”

配置参数的新值或磁盘上的配置参数值。

表 326. 事件监视信息

事件类型	逻辑数据分组	监视开关
变更历史记录	DBDBMCFG	始终收集

用法

对于变更历史记录事件监视器:

- 如果事件是更改数据库配置参数 (DBCFG) 或数据库管理器配置参数 (DBMCFG)，那么这是配置参数的新值。
- 如果事件是捕获事件监视器处于不活动状态时更改的数据库配置参数值 (DBCFGVALUES) 或数据库管理器配置参数值 (DBMCFGVALUES)，那么这是磁盘上的配置参数值。磁盘配置参数值是最新值，并且可能尚未生效。

cfg_value_flags -“配置值标志”

此标志指示如何确定新配置参数值。

表 327. 事件监视信息

事件类型	逻辑数据分组	监视开关
变更历史记录	DBDBMCFG	始终收集

用法

对于变更历史记录事件监视器，此元素指示如何确定新配置参数值:

- AUTOMATIC
- COMPUTED
- NONE

如果事件是捕获事件监视器处于不活动状态时更改的数据库配置参数值 (DBCFGVALUES) 或数据库管理器配置参数值 (DBMCFGVALUES)，那么这些标志表示该配置参数的当前磁盘上的值。

ch_auto_tuning -“FCM 通道自动调整指示器”监视元素

指示是否自动设置和调整快速通信管理器 (FCM) 通道的数目。值为 1 表示“是”，值为 0 表示“否”。

表 328. 表函数监视信息

表函数	监视元素收集级别
MON_GET_FCM - 获取 FCM 度量值	ACTIVITY METRICS BASE

用法

通过将 **fcm_num_channels** 配置参数设置为 AUTOMATIC，即可启用自动调整 FCM 通道。

ch_free -“当前可用的通道数”

此元素指示当前可用的 FCM 通信通道数。

表 329. 表函数监视信息

表函数	监视元素收集级别
MON_GET_FCM - 获取 FCM 度量值	ACTIVITY METRICS BASE

表 330. 快照监视信息

快照级别	逻辑数据分组	监视开关
数据库管理器	fcm	基本

用法

要计算可用 FCM 通道百分比，请使用以下公式:

```
(ch_free/ch_total) * 100
```

如果可用 FCM 通道的百分比低于 20% 且如果启用了 FCM 通道自动调整，那么 DB2 数据库管理器将调整 FCM 通道数。

如果可用 FCM 通道的百分比低于 20% 且如果未启用 FCM 通道自动调整，那么您需要调整 **fcm_num_channels** 配置参数。

ch_free_bottom -“最低可用通道数”

处理期间达到的可用 FCM 通信信道的最小数目。

表 331. 表函数监视信息

表函数	监视元素收集级别
MON_GET_FCM - 获取 FCM 度量值	ACTIVITY METRICS BASE

表 332. 快照监视信息

快照级别	逻辑数据分组	监视开关
数据库管理器	fcm	基本

用法

将此监视元素与 **fcm_num_channels** 配置参数一起使用来确定最大连接条目利用率。

ch_max -“FCM 通道可能达到的最大数目”监视元素

当实例启动时，可以根据保留的虚拟内存量来分配的快速通信管理器 (FCM) 通道的最大数目。

表 333. 表函数监视信息

表函数	监视元素收集级别
MON_GET_FCM - 获取 FCM 度量值	ACTIVITY METRICS BASE

表 334. 快照监视信息

快照级别	逻辑数据分组	监视开关
数据库管理器	fcm	基本

用法

此内部监视元素仅由 IBM 支持机构使用。

ch_total -“当前已分配的 FCM 通道数”监视元素

当前已分配的快速通信管理器 (FCM) 通道的数目。此数目既包括正在使用的通道，又包括可用通道。

表 335. 表函数监视信息

表函数	监视元素收集级别
MON_GET_FCM - 获取 FCM 度量值	ACTIVITY METRICS BASE

表 336. 快照监视信息

快照级别	逻辑数据分组	监视开关
数据库管理器	fcm	基本

用法

如果 **ch_auto_tuning** 监视元素指示要自动调整 FCM，那么会根据对于 FCM 通道的需求来调整 **ch_total** 监视元素的值。

要确定当前使用的 FCM 通道的数量，请使用以下公式:

```
ch_total - ch_free
```

要计算可用 FCM 通道百分比，请使用以下公式:

```
(ch_free/ch_total) * 100
```

如果可用 FCM 通道的百分比低于 20% 且如果启用了 FCM 通道自动调整，那么 DB2 数据库管理器将调整 FCM 通道数。

如果可用 FCM 通道的百分比低于 20% 且如果未启用 FCM 通道自动调整，那么您需要调整 **fcm_num_channels** 配置参数。

client_acctng -“客户机记帐字符串”监视元素

如果在此连接中发出了 sqleseti API，那么此项是为了用于记录和诊断而传递至目标数据库的数据。 此连接、工作单元或活动的 CLIENT_ACCTNG 专用寄存器的当前值。

注: 仅对协调成员报告此元素。在远程成员上，报告的值是长度为 0 的字符串。

此监视元素与 **tpmon_acc_str** 监视元素同义。**client_acctng** 监视元素用于监视写入 DB2 V9.7 所引入的无格式表的表函数和事件监视器。**tpmon_acc_str** 监视元素用于写表、文件和管道的快照监视器和事件监视器。

表 337. 表函数监视信息

表函数	监视元素收集级别
MON_GET_ACTIVITY_DETAILS 表函数 - 获取完整的活动详细信息（在 XML 文档 DETAILS 中报告）	ACTIVITY METRICS BASE
MON_GET_CONNECTION 表函数 - 获取连接度量值	ACTIVITY METRICS BASE
MON_GET_CONNECTION_DETAILS 表函数 - 获取详细的连接度量值（在 XML 文档 DETAILS 中报告）	ACTIVITY METRICS BASE
MON_GET_UNIT_OF_WORK 表函数 - 获取工作单元度量值	ACTIVITY METRICS BASE
MON_GET_UNIT_OF_WORK 表函数 - 获取详细的工作单元度量值（在 XML 文档 DETAILS 中报告）	ACTIVITY METRICS BASE
WLM_GET_SERVICE_CLASS_WORKLOAD_OCCURRENCES 表函数 - 列示工作负载出现次数	ACTIVITY METRICS BASE

表 338. 事件监视信息

事件类型	逻辑数据分组	监视元素收集级别
锁定	-	-
工作单元	-	-
变更历史记录	changesummary	始终收集

用法

此元素用于问题确定和记帐目的。

client_applname -“客户机应用程序名称”监视元素

如果在此连接中发出 sqleseti API，那么此项标识执行事务时出现的服务器事务程序问题。 此连接、工作单元或活动的 CLIENT_APPLNAME 专用寄存器的当前值。

注: 仅对协调成员报告此元素。在远程成员上，报告的值是长度为 0 的字符串。

此监视元素与 **tpmon_client_app** 监视元素同义。**client_applname** 监视元素用于监视写入 DB2 V9.7 所引入的无格式表的表函数和事件监视器。**tpmon_client_app** 监视元素用于写表、文件和管道的快照监视器和事件监视器。

表 339. 表函数监视信息

表函数	监视元素收集级别
MON_GET_ACTIVITY_DETAILS 表函数 - 获取完整的活动详细信息（在 XML 文档 DETAILS 中报告）	ACTIVITY METRICS BASE
MON_GET_CONNECTION 表函数 - 获取连接度量值	ACTIVITY METRICS BASE
MON_GET_CONNECTION_DETAILS 表函数 - 获取详细的连接度量值（在 XML 文档 DETAILS 中报告）	ACTIVITY METRICS BASE
MON_GET_UNIT_OF_WORK 表函数 - 获取工作单元度量值	ACTIVITY METRICS BASE
MON_GET_UNIT_OF_WORK 表函数 - 获取详细的工作单元度量值（在 XML 文档 DETAILS 中报告）	ACTIVITY METRICS BASE
WLM_GET_SERVICE_CLASS_WORKLOAD_OCCURRENCES 表函数 - 列示工作负载出现次数	ACTIVITY METRICS BASE

表 340. 事件监视信息

事件类型	逻辑数据分组	监视元素收集级别
锁定	-	-
工作单元	-	-
变更历史记录	changesummary	始终收集

用法

此元素用于问题确定和记帐目的。

client_db_alias -“应用程序使用的数据库别名”

由要连接至数据库的应用程序提供的数据库别名。

元素标识

client_db_alias

元素类型

信息

表 341. 快照监视信息

快照级别	逻辑数据分组	监视开关
应用程序	appl_id_info	基本
锁定	appl_lock_list	基本

表 342. 事件监视信息

事件类型	逻辑数据分组	监视开关
连接	event_connheader	始终收集

用法 此元素可用来标识应用程序正在访问的实际数据库。此名称与 *db_name* 之间的映射可通过在客户机节点和数据库管理器服务器节点上使用数据库目录来实现。

这是在发出数据库连接请求的数据库管理器中定义的别名。

此元素还可用于帮助您确定认证类型，原因是不同数据库别名可能具有不同的认证类型。

client_hostname -“客户机主机名”监视元素

与客户机应用程序相连的机器的主机名。

表 343. 表函数监视信息

表函数	监视元素收集级别
MON_GET_CONNECTION 表函数 - 获取连接度量值	ACTIVITY METRICS BASE
MON_GET_CONNECTION_DETAILS 表函数 - 获取详细的连接度量值（在 XML 文档 DETAILS 中报告）	ACTIVITY METRICS BASE
MON_GET_UNIT_OF_WORK 表函数 - 获取工作单元度量值	ACTIVITY METRICS BASE
MON_GET_UNIT_OF_WORK 表函数 - 获取详细的工作单元度量值（在 XML 文档 DETAILS 中报告）	ACTIVITY METRICS BASE

表 344. 事件监视信息

事件类型	逻辑数据分组	监视元素收集级别
工作单元	-	始终收集
变更历史记录	changesummary	始终收集

client_idle_wait_time -“客户机空闲等待时间”监视元素

此监视元素用于记录等待客户机发送下一个请求时耗用的时间。此值以毫秒计。

表 345. 表函数监视信息

表函数	监视元素收集级别
MON_FORMAT_XML_METRICS_BY_ROW - 获取所有度量值的基于已格式化行的输出	不适用; 报告 XML 文档中作为格式化函数的输入提供的所有元素
MON_FORMAT_XML_TIMES_BY_ROW - - 获取基于已格式化行的组合层次结构等待时间和处理时间	不适用; 报告 XML 文档中作为格式化函数的输入提供的所有元素
MON_FORMAT_XML_WAIT_TIMES_BY _ROW - 获取等待时间的已格式化的基于行的输出	不适用; 报告 XML 文档中作为格式化函数的输入提供的所有元素
MON_GET_CONNECTION 表函数 - 获取连接度量值	REQUEST METRICS BASE
MON_GET_CONNECTION_DETAILS 表函数 - 获取详细的连接度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE
MON_GET_SERVICE_SUBCLASS 表函数 - 获取服务子类度量值	REQUEST METRICS BASE
MON_GET_SERVICE_SUBCLASS_DETAILS 表函数 - 获取详细的服务子类度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE
MON_GET_UNIT_OF_WORK 表函数 - 获取工作单元度量值	REQUEST METRICS BASE
MON_GET_UNIT_OF_WORK 表函数 - 获取详细的工作单元度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE
MON_GET_WORKLOAD 表函数 - 获取工作负载度量值	REQUEST METRICS BASE
MON_GET_WORKLOAD_DETAILS 表函数 - 获取详细的工作负载度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE

表 346. 事件监视信息

事件类型	逻辑数据分组	监视开关
统计信息	event_scstats（在 details_xml 文档中报告）	REQUEST METRICS BASE
统计信息	event_wlstats（在 details_xml 文档中报告）	REQUEST METRICS BASE

表 346. 事件监视信息 (续)

事件类型	逻辑数据分组	监视开关
工作单元	在 system_metrics 文档中报告。	始终收集

用法

使用此监视元素来确定等待来自客户机的请求（而不是处理请求）所耗用的时间。如果客户机空闲时间过长，那么表明客户机端（而不是服务器端）存在需要解决的性能问题。

client_nname -“客户机名称”监视元素

建议不要使用此监视元素。返回的值不是有效值。

表 347. 事件监视信息

事件类型	逻辑数据分组	监视开关
事务	event_connheader	
语句	event_connheader	
死锁	event_connheader	
连接	event_connheader	

client_pid -“客户机进程标识”监视元素

建立与数据库的连接的客户机应用程序的进程标识。

表 348. 表函数监视信息

表函数	监视元素收集级别
MON_GET_CONNECTION 表函数 - 获取连接度量值	ACTIVITY METRICS BASE
MON_GET_CONNECTION_DETAILS 表函数 - 获取详细的连接度量值（在 XML 文档 DETAILS 中报告）	ACTIVITY METRICS BASE
WLM_GET_SERVICE_CLASS_AGENTS 表函数 - 列示正在服务类中运行的代理程序	ACTIVITY METRICS BASE

表 349. 快照监视信息

快照级别	逻辑数据分组	监视开关
应用程序	appl_info	基本
应用程序	appl	基本
DCS 应用程序	dcs_appl_info	基本

表 350. 事件监视信息

事件类型	逻辑数据分组	监视元素收集级别
工作单元	-	始终收集
连接	event_connheader	始终收集

表 350. 事件监视信息 (续)

事件类型	逻辑数据分组	监视元素收集级别
变更历史记录	changesummary	始终收集

用法

可使用此元素使 CPU 和 I/O 时间之类的监视器信息与客户机应用程序相关。

如果是 DRDA AS 连接，那么此元素将设置为 0。

client_platform -“客户机操作平台”监视元素

运行客户机应用程序的操作系统。

表 351. 表函数监视信息

表函数	监视元素收集级别
MON_GET_CONNECTION 表函数 - 获取连接度量值	ACTIVITY METRICS BASE
MON_GET_CONNECTION_DETAILS 表函数 - 获取详细的连接度量值（在 XML 文档 DETAILS 中报告）	ACTIVITY METRICS BASE

表 352. 快照监视信息

快照级别	逻辑数据分组	监视开关
应用程序	appl_info	基本
应用程序	appl	基本
DCS 应用程序	dcs_appl_info	基本

表 353. 事件监视信息

事件类型	逻辑数据分组	监视元素收集级别
工作单元	-	始终收集
连接	event_connheader	始终收集
变更历史记录	changesummary	始终收集

用法

此元素可用于远程应用程序的问题确定。可在头文件 `sqlmon.h` 中找到此字段的值。

client_port_number -“客户机端口号”监视元素

对于 TCP/IP 连接，这是应用程序用来与数据库服务器进行通信的客户机上的端口号。

表 354. 表函数监视信息

表函数	监视元素收集级别
MON_GET_CONNECTION 表函数 - 获取连接度量值	ACTIVITY METRICS BASE

表 354. 表函数监视信息 (续)

表函数	监视元素收集级别
MON_GET_CONNECTION_DETAILS 表函数 - 获取详细的连接度量值（在 XML 文档 DETAILS 中报告）	ACTIVITY METRICS BASE
MON_GET_UNIT_OF_WORK 表函数 - 获取工作单元度量值	ACTIVITY METRICS BASE
MON_GET_UNIT_OF_WORK 表函数 - 获取详细的工作单元度量值（在 XML 文档 DETAILS 中报告）	ACTIVITY METRICS BASE

表 355. 事件监视信息

事件类型	逻辑数据分组	监视元素收集级别
工作单元	-	始终收集
变更历史记录	changesummary	始终收集

client_prdid -“客户机产品和版本标识”监视元素

正在客户机上运行的产品和版本。此监视元素是 client_product_id 监视元素的同义词。

表 356. 表函数监视信息

表函数	监视元素收集级别
MON_GET_CONNECTION 表函数 - 获取连接度量值	ACTIVITY METRICS BASE
MON_GET_CONNECTION_DETAILS 表函数 - 获取详细的连接度量值（在 XML 文档 DETAILS 中报告）	ACTIVITY METRICS BASE

表 357. 快照监视信息

快照级别	逻辑数据分组	监视开关
应用程序	appl_id_info	基本
DCS 应用程序	dcs_appl_info	基本

表 358. 事件监视信息

事件类型	逻辑数据分组	监视开关
工作单元	-	始终收集
连接	event_connheader	始终收集

用法

可使用此元素来标识IBM 数据服务器客户机的产品和代码版本。其格式为 PPPVVRRM，其中:

- PPP 标识产品，对于 DB2 产品为“SQL”。
- VV 标识两位版本号（版本只有一位时则高位为 0）

- RR 标识两位发行版号（发行版只有一位时高位为 0）
- M 标识 1 个字符的修改级别（0-9 或 A-Z）。

client_protocol -“客户机通信协议”监视元素

客户机应用程序用于与服务器通信的通信协议。

表 359. 表函数监视信息

表函数	监视元素收集级别
MON_GET_CONNECTION 表函数 - 获取连接度量值	ACTIVITY METRICS BASE
MON_GET_CONNECTION_DETAILS 表函数 - 获取详细的连接度量值（在 XML 文档 DETAILS 中报告）	ACTIVITY METRICS BASE

表 360. 快照监视信息

快照级别	逻辑数据分组	监视开关
应用程序	appl_info	基本
应用程序	appl	基本
DCS 应用程序	dcs_appl_info	基本

表 361. 事件监视信息

事件类型	逻辑数据分组	监视元素收集级别
工作单元	-	始终收集
连接	event_connheader	始终收集
变更历史记录	changesummary	始终收集

用法

此元素可用于远程应用程序的问题确定。此字段的值是:

SQLM_PROT_UNKNOWN
客户机使用未知协议进行通信。仅当将来客户机与较早级别的服务器连接时，才返回此值。

SQLM_PROT_LOCAL
客户机与服务器在同一节点上运行，未使用任何通信协议。

SQLM_PROT_TCPIP
TCP/IP

client_userid -“客户机用户标识”监视元素

如果使用 sqleseti API，那么此项是由事务管理器生成的并且提供给服务器的客户机用户标识。 此连接、工作单元或活动的 CLIENT_USERID 专用寄存器的当前值。

注: 仅对协调成员报告此元素。在远程成员上，报告的值是长度为 0 的字符串。

此监视元素与 **tpmon_client_userid** 监视元素同义。**client_userid** 监视元素用于监视写入 DB2 V9.7 所引入的无格式表的表函数和事件监视器。**tpmon_client_userid** 监视元素用于写表、文件和管道的快照监视器和事件监视器。

表 362. 表函数监视信息

表函数	监视元素收集级别
MON_GET_ACTIVITY_DETAILS 表函数 - 获取完整的活动详细信息（在 XML 文档 DETAILS 中报告）	ACTIVITY METRICS BASE
MON_GET_CONNECTION 表函数 - 获取连接度量值	ACTIVITY METRICS BASE
MON_GET_CONNECTION_DETAILS 表函数 - 获取详细的连接度量值（在 XML 文档 DETAILS 中报告）	ACTIVITY METRICS BASE
MON_GET_UNIT_OF_WORK 表函数 - 获取工作单元度量值	ACTIVITY METRICS BASE
MON_GET_UNIT_OF_WORK 表函数 - 获取详细的工作单元度量值（在 XML 文档 DETAILS 中报告）	ACTIVITY METRICS BASE

表 363. 事件监视信息

事件类型	逻辑数据分组	监视元素收集级别
锁定	-	-
工作单元	-	-
变更历史记录	changesummary	始终收集

用法

通过在应用程序服务器或事务处理监视器环境中使用此元素，可以标识为其执行事务的最终用户。

client_wrkstnname -“客户机工作站名称”监视元素

如果在此连接中发出了 sqleseti API，那么此项标识客户机的系统或工作站（如 CICS EITERMID）。此连接、工作单元或活动的 CLIENT_WRKSTNNAME 专用寄存器的当前值。

注: 仅对协调成员报告此元素。在远程成员上，报告的值是长度为 0 的字符串。

此监视元素与 **tpmon_client_wkstn** 监视元素同义。**client_wrkstnname** 监视元素用于监视写入 DB2 V9.7 所引入的无格式表的表函数和事件监视器。**tpmon_client_wkstn** 监视元素用于写表、文件和管道的快照监视器和事件监视器。

表 364. 表函数监视信息

表函数	监视元素收集级别
MON_GET_ACTIVITY_DETAILS 表函数 - 获取完整的活动详细信息（在 XML 文档 DETAILS 中报告）	ACTIVITY METRICS BASE

表 364. 表函数监视信息 (续)

表函数	监视元素收集级别
MON_GET_CONNECTION 表函数 - 获取连接度量值	ACTIVITY METRICS BASE
MON_GET_CONNECTION_DETAILS 表函数 - 获取详细的连接度量值（在 XML 文档 DETAILS 中报告）	ACTIVITY METRICS BASE
MON_GET_UNIT_OF_WORK 表函数 - 获取工作单元度量值	ACTIVITY METRICS BASE
MON_GET_UNIT_OF_WORK 表函数 - 获取详细的工作单元度量值（在 XML 文档 DETAILS 中报告）	ACTIVITY METRICS BASE
WLM_GET_SERVICE_CLASS_WORKLOAD_OCCURRENCES 表函数 - 列示工作负载出现次数	ACTIVITY METRICS BASE

表 365. 事件监视信息

事件类型	逻辑数据分组	监视元素收集级别
锁定	-	-
工作单元	-	-
变更历史记录	changesummary	始终收集

用法

使用此元素并借助节点标识、终端标识或类似标识来标识用户的机器。

codepage_id -“应用程序使用的代码页标识”

代码页标识。

表 366. 快照监视信息

快照级别	逻辑数据分组	监视开关
应用程序	appl_id_info	基本
锁定	appl_lock_list	基本
DCS 应用程序	dcs_appl_info	基本

表 367. 事件监视信息

事件类型	逻辑数据分组	监视开关
事件日志头	event_log_header	始终收集
连接	event_connheader	始终收集

用法 对于快照监视器数据，这是启动被监视应用程序的分区的代码页。此标识可用于远程应用程序的问题确定。可以使用此信息来确保应用程序代码页与数据库代码页（或者，对于 DRDA 主机数据库则为主机 CCSID）之间的数据转换是受支持的。有关受支持代码页的信息，请参阅 *管理指南* 。

对于事件监视器数据，这是对其收集事件数据的数据库的代码页。可使用此元素来确定事件监视器应用程序是否在与数据库使用的代码页不同的代码页中运行。事件监视器写下的数据使用数据库代码页。如果事件监视器应用程序使用另一代码页，那么可能需要执行一些字符转换来使数据可读。

comm_exit_wait_time -“通信缓冲区出口等待时间”监视元素

等待从通信缓冲区出口库 API 函数返回结果时耗用的时间。此值以毫秒计。

表 368. 表函数监视信息

表函数	监视元素收集级别
MON_GET_CONNECTION 表函数 - 获取连接度量值	REQUEST METRICS BASE
MON_GET_CONNECTION_DETAILS 表函数 - 获取详细的连接度量值	REQUEST METRICS BASE
MON_GET_SERVICE_SUBCLASS 表函数 - 获取服务子类度量值	REQUEST METRICS BASE
MON_GET_SERVICE_SUBCLASS_DETAILS 表函数 - 获取详细的服务子类度量值	REQUEST METRICS BASE
MON_GET_UNIT_OF_WORK 表函数 - 获取工作单元度量值	REQUEST METRICS BASE
MON_GET_UNIT_OF_WORK_DETAILS 表函数 - 获取详细的工作单元度量值	REQUEST METRICS BASE
MON_GET_WORKLOAD 表函数 - 获取工作负载度量值	REQUEST METRICS BASE
MON_GET_WORKLOAD_DETAILS 表函数 - 获取详细的工作负载度量值	REQUEST METRICS BASE

表 369. 事件监视信息

事件类型	逻辑数据分组	监视开关
工作单元	在 system_metrics 文档中报告。	始终收集
统计信息	Event_scstats（在 details_xml 文档中报告）。	REQUEST METRICS BASE
统计信息	Event_wlstats（在 details_xml 文档中报告）。	REQUEST METRICS BASE

comm_exit_waits -“通信缓冲区出口等待数”监视元素

调用通信缓冲区出口库 API 函数的次数。

表 370. 表函数监视信息

表函数	监视元素收集级别
MON_GET_CONNECTION 表函数 - 获取连接度量值	REQUEST METRICS BASE
MON_GET_CONNECTION_DETAILS 表函数 - 获取详细的连接度量值	REQUEST METRICS BASE

表 370. 表函数监视信息 (续)

表函数	监视元素收集级别
MON_GET_SERVICE_SUBCLASS 表函数 - 获取服务子类度量值	REQUEST METRICS BASE
MON_GET_SERVICE_SUBCLASS_DETAILS 表函数 - 获取详细的服务子类度量值	REQUEST METRICS BASE
MON_GET_UNIT_OF_WORK 表函数 - 获取工作单元度量值	REQUEST METRICS BASE
MON_GET_UNIT_OF_WORK_DETAILS 表函数 - 获取详细的工作单元度量值	REQUEST METRICS BASE
MON_GET_WORKLOAD 表函数 - 获取工作负载度量值	REQUEST METRICS BASE
MON_GET_WORKLOAD_DETAILS 表函数 - 获取详细的工作负载度量值	REQUEST METRICS BASE

表 371. 事件监视信息

事件类型	逻辑数据分组	监视开关
工作单元	在 system_metrics 文档中报告。	始终收集
统计信息	Event_scstats（在 details_xml 文档中报告）。	REQUEST METRICS BASE
统计信息	Event_wlstats（在 details_xml 文档中报告）。	REQUEST METRICS BASE

comm_private_mem -“已落实的专用内存”

目前数据库管理器的实例在获取快照时已落实的专用内存量。 返回的 comm_private_mem 值仅在 Windows 操作系统上有用。

表 372. 快照监视信息

快照级别	逻辑数据分组	监视开关
数据库管理器	db2	基本

commit_sql_stmts -“尝试的落实语句数”

尝试的 SQL COMMIT 语句总数。

表 373. 快照监视信息

快照级别	逻辑数据分组	监视开关
数据库	dbase	基本
数据库	dbase_remote	基本
应用程序	appl	基本
应用程序	appl_remote	基本
DCS 数据库	dcs_dbase	基本
DCS 应用程序	dcs_appl	基本

可将快照监视的计数器重置。

表 374. 事件监视信息

事件类型	逻辑数据分组	监视开关
数据库	event_db	始终收集
连接	event_conn	始终收集

用法 如果监视期间此计数器数字很少变化，那么指示应用程序执行的落实操作较少，这可能导致登录和数据并行性出现问题。

还可使用此元素并使用下列表达式来计算工作单元总数:

```
    commit_sql_stmts
+ int_commits
+ rollback_sql_stmts
+ int_rollbacks
```

注: 计算的工作单元数将仅包括发生以下情况之后出现的工作单元:

- 与数据库的连接（对于数据库级别信息，这是第一次连接的时间）
- 数据库监视器计数器的最后一次重置。

此计算可在数据库级别或应用程序级别完成。

comp_env_desc -“编译环境”监视元素

此元素存储关于编译 SQL 语句时使用的编译环境的信息。

表 375. 表函数监视信息

表函数	监视元素收集级别
MON_GET_PKG_CACHE_STMT 表函数 - 获取程序包高速缓存中的 SQL 语句活动度量值	ACTIVITY METRICS BASE
MON_GET_PKG_CACHE_STMT_DETAILS 表函数 - 获取程序包高速缓存条目的详细度量值	ACTIVITY METRICS BASE

表 376. 事件监视信息

事件类型	逻辑数据分组	监视开关
带有详细信息的死锁的历史记录值	event_stmt_history	始终收集
带有详细信息的死锁的历史纪录	event_stmt_history	始终收集
活动	event_activitystmt	始终收集
程序包高速缓存	-	COLLECT BASE DATA

表 377. 事件监视信息

事件类型	逻辑数据分组	监视开关
程序包高速缓存	-	COLLECT BASE DATA

用法

此监视元素将编译环境描述存储在二进制大对象中。要以可读格式查看此信息，请使用 COMPILATION_ENV 表函数。

可提供此元素作为 COMPILATION_ENV 表函数的输入，或者作为 SET COMPILATION ENVIRONMENT SQL 语句的输入。

completion_status -“完成状态”监视元素

工作单元的状态。

表 378. 事件监视信息

事件类型	逻辑数据分组	监视开关
工作单元	-	始终收集

用法

使用此元素来确定工作单元是否因为死锁或异常终止而结束。 `sqllib/misc/DB2EvmonUOW.xsd` 文件列示了可能的值:

- UNKNOWN
- COMMIT
- ROLLBACK
- GLOBAL_COMMIT
- GLOBAL_ROLLBACK
- XA_END
- XA_PREPARE

configured_cf_gbp_size -“已配置的集群高速缓存设施组缓冲池大小”监视元素

使用 **cf_gbp_sz** 配置参数指定的已分配并保留的组缓冲池内存（以大小为 4 KB 的页计）。

表 379. 表函数监视信息

表函数	监视元素收集级别
MON_GET_CF 表函数 - 获取 CF 度量值	ACTIVITY METRICS BASE

configured_cf_lock_size -“已配置的集群高速缓存设施锁定大小”监视元素

已配置的全局锁定内存（以大小为 4 KB 的页计）。此值是使用 **cf_lock_sz** 配置参数指定的。

表 380. 表函数监视信息

表函数	监视元素收集级别
MON_GET_CF 表函数 - 获取 CF 度量值	ACTIVITY METRICS BASE

configured_cf_sca_size -“已配置的集群高速缓存设施共享通信区大小”监视元素

当前分配并保留的共享通信区内存（以大小为 4 KB 的页计）。此值是使用 **cf_sca_sz** 配置参数指定的。

表 381. 表函数监视信息

表函数	监视元素收集级别
MON_GET_CF 表函数 - 获取 CF 度量值	ACTIVITY METRICS BASE

configured_cf_mem_size -“已配置的集群高速缓存设施内存大小”监视元素

为集群高速缓存设施配置的总内存大小（以大小为 4 KB 的页计）。此值是使用 **cf_mem_sz** 配置参数指定的。

表 382. 表函数监视信息

表函数	监视元素收集级别
MON_GET_CF 表函数 - 获取 CF 度量值	ACTIVITY METRICS BASE

con_elapsed_time -“最新连接耗用时间”

连接最新与此主机数据库断开连接的 DCS 应用程序所耗用的时间。

元素标识

con_elapsed_time

元素类型

时间

表 383. 快照监视信息

快照级别	逻辑数据分组	监视开关
DCS 数据库	dcs_dbase	时间戳记

用法

将此元素用作应用程序保持与主机数据库的连接的时间长度的指示符。

此元素由两个子元素组成，它们报告耗用时间的秒数和微秒（一秒的百万分之一）数。这些子元素的名称可通过将“_s”和“_ms”添加至此监视元素的名称派生而成。要检索此监视元素耗用的总时间，必须将这两个子元素的值加在一起。例如，如果“_s”子元素值为 3，“_ms”子元素值为 20，那么此监视元素耗用的总时间为 3.00002 秒。

con_local_dbases -“带有当前连接的本地数据库”

连接了应用程序的本地数据库的数目。

表 384. 快照监视信息

快照级别	逻辑数据分组	监视开关
数据库管理器	db2	基本

用法 此值指示在数据库级别收集数据时您可以期望的数据库信息记录数。

应用程序可在本地或远程运行，并且可能执行也可能不执行 数据库管理器 中的工作单元

con_response_time -“连接的最新响应时间”

对于连接至此数据库的最新 DCS 应用程序，此项为连接处理开始与实际建立连接之间所耗用的时间。

元素标识

con_response_time

元素类型

时间

表 385. 快照监视信息

快照级别	逻辑数据分组	监视开关
DCS 数据库	dcs_dbase	时间戳记

用法

将此元素用作当前应用程序连接至特定主机数据库所花时间的指示符。

此元素由两个子元素组成，它们报告耗用时间的秒数和微秒（一秒的百万分之一）数。这些子元素的名称可通过将“_s”和“_ms”添加至此监视元素的名称派生而成。要检索此监视元素耗用的总时间，必须将这两个子元素的值加在一起。例如，如果“_s”子元素值为 3，“_ms”子元素值为 20，那么此监视元素耗用的总时间为 3.00002 秒。

concurrent_act_top -“最大并行活动数”监视元素

自最后一次重置以后服务子类中并行活动的高水位标记（在任何嵌套级别）。

注: 此元素监视所有活动（包括未参与 CONCURRENTDBCOORDACTIVITIES 阈值的那些活动）的最高并行执行。例如，尽管 CALL 语句未针对 CONCURRENTDBCOORDACTIVITIES 阈值实施的并行计数，但它们会包括在并行活动高水位标记度量中。

表 386. 表函数监视信息

表函数	监视元素收集级别
WLM_GET_SERVICE_CLASS_WORKLOAD_OCCURRENCES 表函数 - 列示工作负载出现次数	ACTIVITY METRICS BASE
WLM_GET_SERVICE_SUBCLASS_STATS 表函数 - 返回服务子类的统计信息	ACTIVITY METRICS BASE

表 387. 事件监视信息

事件类型	逻辑数据分组	监视开关
统计信息	event_scstats	-

用法

使用此元素来了解在收集的时间间隔内，成员对服务子类上达到的最高活动（包括嵌套活动）并行数。

concurrent_connection_top -“最大并行连接数”监视元素

自最后一次重置后此服务类中的并行协调程序连接的高水位标记。在同一超类的每个子类中，此字段的值都相同。

表 388. 表函数监视信息

表函数	监视元素收集级别
WLM_GET_SERVICE_SUPERCLASS_STATS 表函数 - 返回服务超类的统计信息	ACTIVITY METRICS BASE

表 389. 事件监视信息

事件类型	逻辑数据分组	监视开关
统计信息	event_scstats	-

用法

通过显示当前高水位标记所在，此元素可帮助确定在何处对连接并行性设置阈值。此元素还可帮助验证该阈值是否正确配置且正常工作。

concurrent_wlo_act_top -“最大并行 WLO 活动数”监视元素

自最后一次重置后此工作负载的任何项的并行活动高水位标记（在任何嵌套级别）。

表 390. 表函数监视信息

表函数	监视元素收集级别
WLM_GET_WORKLOAD_STATS 表函数 - 返回工作负载统计信息	ACTIVITY METRICS BASE

表 391. 事件监视信息

事件类型	逻辑数据分组	监视开关
统计信息	event_wlstats	-

用法

使用此元素来了解在收集的时间间隔内，成员对此工作负载的任何项上达到的最高并行活动数。

concurrent_wlo_top -“最大并行工作负载项数”监视元素

自最后一次重置后工作负载并行项的高水位标记。

表 392. 表函数监视信息

表函数	监视元素收集级别
WLM_GET_SERVICE_SUBCLASS_STATS 表函数 - 返回服务子类的统计信息	ACTIVITY METRICS BASE
WLM_GET_WORKLOAD_STATS 表函数 - 返回工作负载统计信息	ACTIVITY METRICS BASE

表 393. 事件监视信息

事件类型	逻辑数据分组	监视开关
统计信息	event_wlstats	-
统计信息	event_scstats	-

用法

使用此元素来了解在收集的时间间隔内，成员对工作负载上达到的最高工作负载项并行数。

concurrentdbcoordactivities_db_ threshold_id -“并行数据库协调程序活动数的数据库阈值标识”监视元素

应用于此活动的 CONCURRENTDBCOORDACTIVITIES 数据库阈值的标识。

表 394. 表函数监视信息

表函数	监视元素收集级别
MON_GET_ACTIVITY_DETAILS 表函数 - 获取完整的活动详细信息（在 XML 文档 DETAILS 中报告）	ACTIVITY METRICS BASE

用法

使用此元素来确定应用于此活动的 CONCURRENTDBCOORDACTIVITIES 数据库阈值（如果有的话）。

concurrentdbcoordactivities_db_threshold _queued -“已由并行数据库协调程序活动数的数据库阈值排队”监视元素

此监视元素返回“Yes”表明 CONCURRENTDBCOORDACTIVITIES 数据库阈值已对此活动进行排队。“No”表明未对此活动进行排队。

表 395. 表函数监视信息

表函数	监视元素收集级别
MON_GET_ACTIVITY_DETAILS 表函数 - 获取完整的活动详细信息（在 XML 文档 DETAILS 中报告）	ACTIVITY METRICS BASE

用法

使用此元素来确定应用于此活动的 CONCURRENTDBCOORDACTIVITIES 数据库阈值是否已对此活动进行排队。

concurrentdbcoordactivities_db_ threshold_value -“并行数据库协调程序活动数的数据库阈值”监视元素

此监视元素返回应用于此活动的 CONCURRENTDBCOORDACTIVITIES 数据库阈值的上限。

表 396. 表函数监视信息

表函数	监视元素收集级别
MON_GET_ACTIVITY_DETAILS 表函数 - 获取完整的活动详细信息（在 XML 文档 DETAILS 中报告）	ACTIVITY METRICS BASE

用法

使用此元素来确定应用于此活动的 CONCURRENTDBCOORDACTIVITIES 数据库阈值（如果有的话）。

concurrentdbcoordactivities_db_ threshold_violated -“违反并行数据库协调程序活动数的数据库阈值”监视元素

此监视元素返回“Yes”表明此活动已违反 CONCURRENTDBCOORDACTIVITIES 数据库阈值。“No”表明此活动尚未违反该阈值。

表 397. 表函数监视信息

表函数	监视元素收集级别
MON_GET_ACTIVITY_DETAILS 表函数 - 获取完整的活动详细信息（在 XML 文档 DETAILS 中报告）	ACTIVITY METRICS BASE

用法

使用此元素来确定此活动是否已违反应用于此活动的 CONCURRENTDBCOORDACTIVITIES 数据库阈值。

concurrentdbcoordactivities_subclass_ threshold_id -“并行数据库协调程序活动数的服务子类阈值标识”监视元素

此监视元素返回应用于此活动的 CONCURRENTDBCOORDACTIVITIES 服务子类阈值的标识。

表 398. 表函数监视信息

表函数	监视元素收集级别
MON_GET_ACTIVITY_DETAILS 表函数 - 获取完整的活动详细信息（在 XML 文档 DETAILS 中报告）	ACTIVITY METRICS BASE

用法

使用此元素来确定应用于此活动的 CONCURRENTDBCOORDACTIVITIES 服务子类阈值（如果有的话）。

concurrentdbcoordactivities_subclass_ threshold_queued -“已由并行数据库协调程序活动数的服务子类阈值排队”监视元素

此监视元素返回“Yes”表明 CONCURRENTDBCOORDACTIVITIES 服务子类阈值已对此活动进行排队。“No”表明未对此活动进行排队。

表 399. 表函数监视信息

表函数	监视元素收集级别
MON_GET_ACTIVITY_DETAILS 表函数 - 获取完整的活动详细信息（在 XML 文档 DETAILS 中报告）	ACTIVITY METRICS BASE

用法

使用此元素来确定应用于此活动的 CONCURRENTDBCOORDACTIVITIES 服务子类阈值是否已对此活动进行排队。

concurrentdbcoordactivities_subclass_ threshold_value -“并行数据库协调程序活动数的服务子类阈值”监视元素

此监视元素返回应用于此活动的 CONCURRENTDBCOORDACTIVITIES 服务子类阈值的上限。

表 400. 表函数监视信息

表函数	监视元素收集级别
MON_GET_ACTIVITY_DETAILS 表函数 - 获取完整的活动详细信息（在 XML 文档 DETAILS 中报告）	ACTIVITY METRICS BASE
WLM_GET_ACTIVITY_DETAILS 表函数- 返回有关特定活动的详细信息	ACTIVITY METRICS BASE

用法

使用此元素来确定应用于此活动的 CONCURRENTDBCOORDACTIVITIES 服务子类阈值（如果有的话）。

concurrentdbcoordactivities_subclass_ threshold_violated -“违反并行数据库协调程序活动数的服务子类阈值”监视元素

此监视元素返回“Yes”表明此活动已违反 CONCURRENTDBCOORDACTIVITIES 服务子类阈值。“No”表明此活动尚未违反该阈值。

表 401. 表函数监视信息

表函数	监视元素收集级别
MON_GET_ACTIVITY_DETAILS 表函数 - 获取完整的活动详细信息（在 XML 文档 DETAILS 中报告）	ACTIVITY METRICS BASE

用法

使用此元素来确定此活动是否已违反应用于此活动的 CONCURRENTDBCOORDACTIVITIES 服务子类阈值。

concurrentdbcoordactivities_superclass_ threshold_id -“并行数据库协调程序活动数的服务超类阈值标识”监视元素

应用于此活动的 CONCURRENTDBCOORDACTIVITIES_SUPERCLASS 阈值的标识。

表 402. 表函数监视信息

表函数	监视元素收集级别
MON_GET_ACTIVITY_DETAILS 表函数 - 获取完整的活动详细信息（在 XML 文档 DETAILS 中报告）	ACTIVITY METRICS BASE

用法

使用此元素来确定应用于此活动的 CONCURRENTDBCOORDACTIVITIES 服务超类阈值（如果有的话）。

concurrentdbcoordactivities_superclass_ threshold_queued -“已由并行数据库协调程序活动数的服务超类阈值排队”监视元素

此监视元素返回“Yes”表明 CONCURRENTDBCOORDACTIVITIES 服务超类阈值已对此活动进行排队。“No”表明未对此活动进行排队。

表 403. 表函数监视信息

表函数	监视元素收集级别
MON_GET_ACTIVITY_DETAILS 表函数 - 获取完整的活动详细信息（在 XML 文档 DETAILS 中报告）	ACTIVITY METRICS BASE

用法

使用此元素来确定应用于此活动的 CONCURRENTDBCOORDACTIVITIES 服务超类阈值是否已对此活动进行排队。

concurrentdbcoordactivities_superclass_ threshold_value -“并行数据库协调程序活动数的服务超类阈值”监视元素

应用于此活动的 CONCURRENTDBCOORDACTIVITIES 服务超类阈值的上限。

表 404. 表函数监视信息

表函数	监视元素收集级别
MON_GET_ACTIVITY_DETAILS 表函数 - 获取完整的活动详细信息（在 XML 文档 DETAILS 中报告）	ACTIVITY METRICS BASE

用法

使用此元素来确定应用于此活动的 CONCURRENTDBCOORDACTIVITIES 服务超类阈值（如果有的话）。

concurrentdbcoordactivities_superclass_ threshold_violated -“违反并行数据库协调程序活动数的服务超类阈值”监视元素

此监视元素返回“Yes”表明此活动已违反 CONCURRENTDBCOORDACTIVITIES 服务超类阈值。“No”表明此活动尚未违反该阈值。

表 405. 表函数监视信息

表函数	监视元素收集级别
MON_GET_ACTIVITY_DETAILS 表函数 - 获取完整的活动详细信息（在 XML 文档 DETAILS 中报告）	ACTIVITY METRICS BASE

用法

使用此元素来确定此活动是否已违反应用于此活动的 CONCURRENTDBCOORDACTIVITIES 服务超类阈值。

concurrentdbcoordactivities_wl_was_threshold _id -“并行数据库协调程序活动数的工作负载工作操作集阈值标识”监视元素

应用于此活动的 CONCURRENTDBCOORDACTIVITIES 工作负载工作操作集阈值的标识。

表 406. 表函数监视信息

表函数	监视元素收集级别
MON_GET_ACTIVITY_DETAILS 表函数 - 获取完整的活动详细信息（在 XML 文档 DETAILS 中报告）	ACTIVITY METRICS BASE

用法

使用此元素来了解应用于此活动的 CONCURRENTDBCOORDACTIVITIES 工作负载工作操作集阈值（如果有的话）。

concurrentdbcoordactivities_wl_was_threshold _queued -“已由并行数据库协调程序活动数的工作负载工作操作集阈值排队”监视元素

此监视元素返回“Yes”表明 CONCURRENTDBCOORDACTIVITIES 工作负载工作操作集阈值已对此活动进行排队。“No”表明未对此活动进行排队。

表 407. 表函数监视信息

表函数	监视元素收集级别
MON_GET_ACTIVITY_DETAILS 表函数 - 获取完整的活动详细信息（在 XML 文档 DETAILS 中报告）	ACTIVITY METRICS BASE

用法

使用此元素来了解应用于此活动的 CONCURRENTDBCOORDACTIVITIES 工作负载工作操作集阈值是否已对此活动进行排队。

concurrentdbcoordactivities_wl_was_threshold _value -“并行数据库协调程序活动数的工作负载工作操作集阈值”监视元素

应用于此活动的 CONCURRENTDBCOORDACTIVITIES 工作负载工作操作集阈值的上限。

表 408. 表函数监视信息

表函数	监视元素收集级别
MON_GET_ACTIVITY_DETAILS 表函数 - 获取完整的活动详细信息（在 XML 文档 DETAILS 中报告）	ACTIVITY METRICS BASE

用法

使用此元素来了解应用于此活动的 CONCURRENTDBCOORDACTIVITIES 工作负载工作操作集阈值的值。

concurrentdbcoordactivities_wl_was_threshold _violated -“违反并行数据库协调程序活动数的工作负载工作操作集阈值”监视元素

此监视元素返回“Yes”表明此活动已违反 CONCURRENTDBCOORDACTIVITIES 工作负载工作操作集阈值。“No”表明此活动尚未违反该阈值。

表 409. 表函数监视信息

表函数	监视元素收集级别
MON_GET_ACTIVITY_DETAILS 表函数 - 获取完整的活动详细信息（在 XML 文档 DETAILS 中报告）	ACTIVITY METRICS BASE

用法

使用此元素来确定此活动是否已违反应用于此活动的 CONCURRENTDBCOORDACTIVITIES 工作负载工作操作集阈值。

concurrentdbcoordactivities_work_action_set_ threshold_id -“并行数据库协调程序活动数的工作操作集阈值标识”监视元素

应用于此活动的 CONCURRENTDBCOORDACTIVITIES 工作操作集阈值的标识。

表 410. 表函数监视信息

表函数	监视元素收集级别
MON_GET_ACTIVITY_DETAILS 表函数 - 获取完整的活动详细信息（在 XML 文档 DETAILS 中报告）	ACTIVITY METRICS BASE

用法

使用此元素来确定应用于此活动的 CONCURRENTDBCOORDACTIVITIES 工作操作集阈值（如果有的话）。

concurrentdbcoordactivities_work_action_set_ threshold_queued -“已由并行数据库协调程序活动数的工作操作集阈值排队”监视元素

此监视元素返回“Yes”表明 CONCURRENTDBCOORDACTIVITIES_WORK_ACTION_SET 阈值已对此活动进行排队。“No”表明未对此活动进行排队。

表 411. 表函数监视信息

表函数	监视元素收集级别
MON_GET_ACTIVITY_DETAILS 表函数 - 获取完整的活动详细信息（在 XML 文档 DETAILS 中报告）	ACTIVITY METRICS BASE

用法

使用此元素来确定应用于此活动的 CONCURRENTDBCOORDACTIVITIES_WORK_ACTION_SET 阈值是否已对此活动进行排队。

concurrentdbcoordactivities_work_action_set_ threshold_value -“并行数据库协调程序活动数的工作操作集阈值”监视元素

应用于此活动的 CONCURRENTDBCOORDACTIVITIES_WORK_ACTION_SET 阈值的上限。

表 412. 表函数监视信息

表函数	监视元素收集级别
MON_GET_ACTIVITY_DETAILS 表函数 - 获取完整的活动详细信息（在 XML 文档 DETAILS 中报告）	ACTIVITY METRICS BASE

用法

使用此元素来确定应用于此活动的 CONCURRENTDBCOORDACTIVITIES_WORK 阈值（如果有的话）。

concurrentdbcoordactivities_work_action_set_ threshold_violated -“违反并行数据库协调程序活动数的工作操作集阈值”监视元素

此监视元素返回“Yes”表明此活动已违反 CONCURRENTDBCOORDACTIVITIES_WORK_ACTION_SET 阈值。“No”表明此活动尚未违反该阈值。

表 413. 表函数监视信息

表函数	监视元素收集级别
MON_GET_ACTIVITY_DETAILS 表函数 - 获取完整的活动详细信息（在 XML 文档 DETAILS 中报告）	ACTIVITY METRICS BASE

用法

使用此元素来确定此活动是否已违反应用于此活动的 CONCURRENTDBCOORDACTIVITIES_WORK_ACTION_SET 阈值。

conn_complete_time -“连接请求完成时间戳记”

对连接请求授权的日期和时间。

元素标识

conn_complete_time

元素类型

时间戳记

表 414. 快照监视信息

快照级别	逻辑数据分组	监视开关
应用程序	appl	时间戳记

用法 使用此元素来确定对数据库的连接请求授权的时间。

conn_time -“数据库连接时间”监视元素

在数据库级别第一次连接至数据库的日期和时间，或者发出激活数据库命令的日期和时间。

表 415. 事件监视信息

事件类型	逻辑数据分组	监视开关
工作单元	-	始终收集
数据库	event_dbheader	始终收集
连接	event_connheader	始终收集

用法

通过将此元素与 **disconn_time** 监视元素配合使用，可以计算出现以下情况之后耗用的时间:

- 数据库处于活动状态（用于数据库级别的信息）。
- 连接处于活动状态（用于连接级别的信息）。

connection_start_time -“连接开始时间”监视元素

与数据库服务器建立连接的时间。connection_time 监视元素是 connection_start_time 监视元素的别名。

表 416. 表函数监视信息

表函数	监视元素收集级别
MON_GET_CONNECTION 表函数 - 获取连接度量值	ACTIVITY METRICS BASE
MON_GET_CONNECTION_DETAILS 表函数 - 获取详细的连接度量值（在 XML 文档 DETAILS 中报告）	ACTIVITY METRICS BASE

connection_status -“连接状态”

对于快照监视器，此监视元素报告发出 GET SNAPSHOT 命令的节点与 db2nodes.cfg 文件中所列示的其他节点之间的通信连接状态。对于表函数监视器，此监视元素报告用于指示 FCM 连接状态的文本标识。

表 417. 表函数监视信息

表函数	监视元素收集级别
MON_GET_FCM_CONNECTION_LIST - 获取有关所有 FCM 连接的详细信息	ACTIVITY METRICS BASE

表 418. 快照监视信息

快照级别	逻辑数据分组	监视开关
数据库管理器	fcm_node	基本

用法

对于快照监视器，连接值为:

SQLM_FCM_CONNECT_INACTIVE
当前无连接

SQLM_FCM_CONNECT_ACTIVE
连接处于活动状态

对于表函数监视，可用值是:

活动 当前无连接

不活动 连接处于活动状态

两个成员可以处于活动状态，但是在这两个成员之间存在一些通信之前，它们之间的通信连接将保持不活动状态。

connections_top -“最大并行连接数”

自数据库激活后同时与数据库进行的连接的最大数目。

元素标识

connections_top

元素类型

水位标记

表 419. 快照监视信息

快照级别	逻辑数据分组	监视开关
数据库	dbase	基本

表 420. 事件监视信息

事件类型	逻辑数据分组	监视开关
数据库	event_db	始终收集

用法 可使用此元素来评估 *maxappls* 配置参数的设置。

如果此元素的值与 *maxappls* 参数相同，那么可能拒绝了某些数据库连接请求，原因是 *maxappls* 限制了允许的数据库连接数。

可使用以下公式来计算获取快照时的当前连接数:

```
rem_cons_in + local_cons
```

consistency_token -“程序包一致性标记”监视元素

对于给定程序包名称和创建者，可以有（从 DB2 V8 开始）多个版本。程序包一致性标记帮助标识包含 SQL 当前执行的程序包的版本。

表 421. 快照监视信息

快照级别	逻辑数据分组	监视开关
应用程序	stmt	语句

表 422. 事件监视信息

事件类型	逻辑数据分组	监视开关
锁定	-	始终收集
语句	event_stmt	始终收集

用法

可使用此元素来帮助标识程序包和正在执行的 SQL 语句。

container_accessible -“容器可访问”监视元素

此元素指示容器是否可访问。值为 1 表示“是”，值为 0 表示“否”。

表 423. 表函数监视信息

表函数	监视元素收集级别
MON_GET_CONTAINER 表函数 - 获取表空间容器度量值	ACTIVITY METRICS BASE

表 424. 快照监视信息

快照级别	逻辑数据分组	监视开关
表空间	tablespace_container	基本

用法 通过将此元素与 **container_id**、**container_name**、**container_type**、**container_total_pages**、**container_usable_pages** 和 **container_stripe_set** 元素配合使用，可以描述容器。

container_id -“容器标识”监视元素

在表空间中唯一定义容器的整数。

表 425. 表函数监视信息

表函数	监视元素收集级别
MON_FORMAT_LOCK_NAME 表函数 - 设置内部锁定名称的格式并返回详细信息	ACTIVITY METRICS BASE
MON_GET_CONTAINER 表函数 - 获取容器度量值	ACTIVITY METRICS BASE

表 426. 快照监视信息

快照级别	逻辑数据分组	监视开关
表空间	tablespace_container	基本

用法 此元素可与元素 container_name、container_type、container_total_pages、container_usable_pages、container_stripe_set 和 container_accessible 一起使用来描述容器。

container_name -“容器名称”监视元素

容器的名称。

表 427. 表函数监视信息

表函数	监视元素收集级别
MON_GET_CONTAINER 表函数 - 获取容器度量值	ACTIVITY METRICS BASE

表 428. 快照监视信息

快照级别	逻辑数据分组	监视开关
表空间	tablespace_container	基本

用法 此元素可与元素 container_id、container_type、container_total_pages、container_usable_pages、container_stripe_set 和 container_accessible 一起使用来描述容器。

container_stripe_set -“容器分割集”监视元素

容器所属的分割集。

表 429. 表函数监视信息

表函数	监视元素收集命令和级别
MON_GET_CONTAINER 表函数 - 获取表空间容器度量值	ACTIVITY METRICS BASE

表 430. 快照监视信息

快照级别	逻辑数据分组	监视开关
表空间	tablespace_container	基本

用法

通过将此监视元素与 **container_id**、**container_name**、**container_type**、**container_total_pages**、**container_usable_pages** 和 **container_accessible** 元素配合使用，可以描述容器。此元素仅适用于 DMS 表空间。

container_total_pages -“容器中的总页数”监视元素

容器占用的总页数。

表 431. 表函数监视信息

表函数	监视元素收集命令和级别
MON_GET_CONTAINER 表函数 - 获取表空间容器度量值	ACTIVITY METRICS BASE

表 432. 快照监视信息

快照级别	逻辑数据分组	监视开关
表空间	tablespace_container	基本（DMS 表空间） 缓冲池（SMS 表空间）

用法 此元素可与元素 container_id、container_name、container_type、container_usable_pages、container_stripe_set 和 container_accessible 一起使用来描述容器。

container_type -“容器类型”监视元素

容器的类型。

表 433. 表函数监视信息

表函数	监视元素收集级别
MON_GET_CONTAINER 表函数 - 获取容器度量值	ACTIVITY METRICS BASE

表 434. 快照监视信息

快照级别	逻辑数据分组	监视开关
表空间	tablespace_container	基本

用法

此元素返回容器的类型，它可能是目录路径（仅适用于 SMS）、文件（适用于 DMS）或原始设备（适用于 DMS）。通过将此监视元素与 **container_id**、**container_name**、**container_type**、**container_total_pages**、**container_usable_pages** 和 **container_accessible** 元素配合使用，可以描述容器。

此监视元素的有效值由 sqlutil.h 文件定义。

container_usable_pages -“容器中的可用页数”监视元素

容器中的可用页的总数。

表 435. 表函数监视信息

表函数	监视元素收集命令和级别
MON_GET_CONTAINER 表函数 - 获取表空间容器度量值	ACTIVITY METRICS BASE

表 436. 快照监视信息

快照级别	逻辑数据分组	监视开关
表空间	tablespace_container	基本（DMS 表空间） 缓冲池（SMS 表空间）

用法 此元素可与元素 container_id、container_name、container_type、container_total_pages、container_stripe_set 和 container_accessible 一起使用来描述容器。对于 SMS 表空间，此值与 container_total_pages 相同。

coord_act_aborted_total -“异常终止的协调程序活动总数”监视元素

自最后一次重置后在任何嵌套级别完成时出错的协调程序活动总数。对于服务类，此值在活动完成时更新。对于工作负载，此值在其工作单元结束时由每个工作负载项更新。

对于服务类而言，如果在活动异常终止前通过 REMAP ACTIVITY 操作将其重新映射到另一个子类，那么此活动将仅计入在其中异常终止此活动的子类的总数。

表 437. 表函数监视信息

表函数	监视元素收集级别
WLM_GET_SERVICE_CLASS_WORKLOAD_OCCURRENCES 表函数 - 列示工作负载出现次数	ACTIVITY METRICS BASE
WLM_GET_SERVICE_SUBCLASS_STATS 表函数 - 返回服务子类的统计信息	ACTIVITY METRICS BASE
WLM_GET_WORKLOAD_STATS 表函数 - 返回工作负载统计信息	ACTIVITY METRICS BASE

表 438. 事件监视信息

事件类型	逻辑数据分组	监视开关
统计信息	event_scstats	-
统计信息	event_wlstats	-

用法

使用此元素来了解系统上的活动是否成功完成。活动可能因取消、错误或反应性阈值而异常终止。

coord_act_completed_total -“完成的协调程序活动总数”监视元素

自最后一次重置后在任何嵌套级别成功完成的协调程序活动总数。对于服务类，此值在活动完成时更新。对于工作负载，此值在其工作单元结束时由每个工作负载项更新。

对于服务类而言，如果在活动完成前通过 REMAP ACTIVITY 操作将其重新映射到另一个子类，那么此活动将仅计入在其中完成此活动的子类的总数。

表 439. 表函数监视信息

表函数	监视元素收集级别
WLM_GET_SERVICE_CLASS_WORKLOAD_OCCURRENCES 表函数 - 列示工作负载出现次数	ACTIVITY METRICS BASE
WLM_GET_SERVICE_SUBCLASS_STATS 表函数 - 返回服务子类的统计信息	ACTIVITY METRICS BASE
WLM_GET_WORKLOAD_STATS 表函数 - 返回工作负载统计信息	ACTIVITY METRICS BASE

表 440. 事件监视信息

事件类型	逻辑数据分组	监视开关
统计信息	event_wlstats	-
统计信息	event_scstats	-

用法

此元素可用来确定系统中活动的吞吐量或用来帮助计算多用户间的平均活动生存期。

coord_act_est_cost_avg -“平均协调程序活动估计成本”监视元素

自最后一次重置以来，在嵌套级别 0 处的协调程序 DML 活动估计成本的算术平均值与此服务子类或工作类相关联。如果内部跟踪的平均值已溢出，那么将返回值 -2。对于服务子类，当服务子类的 COLLECT AGGREGATE ACTIVITY DATA 设置为 NONE 或 BASE 时，此监视元素返回 -1。对于工作类，如果未对该工作类指定 COLLECT AGGREGATE ACTIVITY DATA EXTENDED 工作操作，那么此监视元素将返回 -1。对于工作负载而言，当工作负载的 COLLECT AGGREGATE ACTIVITY DATA 设置为 NONE 或 BASE 时，此监视元素返回 -1。单位为 timeron。

对于服务类而言，活动的估计成本只计入活动从其中进入系统的服务子类。使用 REMAP ACTIVITY 操作在服务子类之间重新映射活动时，活动重新映射到的服务子类的 coord_act_est_cost_avg 平均值不受影响。

表 441. 事件监视信息

事件类型	逻辑数据分组	监视开关
统计信息	event_scstats	始终收集
统计信息	event_wcstats	始终收集
统计信息	event_wlstats	始终收集

用法

使用此统计信息来确定自上次统计信息重置以来嵌套级别 0 处的协调程序 DML 活动（与已完成或中止的此服务子类、、工作负载或工作类相关联）的估计成本的算术平均值。

另外，还可以使用此平均值来确定用于活动估计成本直方图的直方图模板是否合适。根据活动估计成本直方图计算平均活动估计成本。将计算出来的平均值与此监视元素进行比较。如果计算出来的平均值偏离了此监视元素报告的真实平均值，那么考虑修改活动估计成本直方图的直方图模板并使用更为适合您的数据的一组 bin 值。

coord_act_exec_time_avg -“平均协调程序活动执行时间”监视元素

自最后一次重置以来，在嵌套级别 0 处的协调程序活动执行时间的算术平均值与此服务子类或工作类相关联。如果内部跟踪的平均值已溢出，那么将返回值 -2。对于服务子类，当服务子类的 COLLECT AGGREGATE ACTIVITY DATA 设置为 NONE 时，此监视元素返回 -1。对于工作类，如果未对该工作类指定 COLLECT AGGREGATE ACTIVITY DATA 工作操作，那么此监视元素将返回 -1。对于工作负载而言，当工作负载的 COLLECT AGGREGATE ACTIVITY DATA 设置为 NONE 时，此监视元素返回 -1。 单位为毫秒。

对于服务类而言，使用 REMAP ACTIVITY 操作在服务子类之间重新映射活动时，活动所映射到但未完成的服务子类的 coord_act_exec_time_avg 平均值不受影响。

表 442. 表函数监视信息

表函数	监视元素收集级别
WLM_GET_SERVICE_SUBCLASS_STATS 表函数 - 返回服务子类的统计信息	COLLECT AGGREGATE ACTIVITY DATA
WLM_GET_WORKLOAD_STATS 表函数 - 返回工作负载统计信息	COLLECT AGGREGATE ACTIVITY DATA

表 443. 事件监视信息

事件类型	逻辑数据分组	监视开关
统计信息	event_scstats	-
统计信息	event_wcstats	-
统计信息	event_wlstats	-

用法

使用此统计信息来确定与已完成或中止的服务子类、工作负载或工作类相关联的协调程序活动执行时间的算术平均值。

另外，还可以使用此平均值来确定用于活动执行时间直方图的直方图模板是否合适。根据活动执行时间直方图来计算平均活动执行时间。将计算出来的平均值与此监视元素进行比较。如果计算出来的平均值偏离了此监视元素报告的真实平均值，那么考虑修改活动执行时间直方图的直方图模板并使用更为适合您的数据的一组 bin 值。

coord_act_interarrival_time_avg -“平均协调程序活动到达时间”监视元素

自最后一次重置以来，在嵌套级别 0 处的协调程序活动到达间隔时间的算术平均值与此服务子类或工作类相关联。如果内部跟踪的平均值已溢出，那么将返回值 -2。对于服务子类，当服务子类的 COLLECT AGGREGATE ACTIVITY DATA 设置为 NONE 或 BASE 时，此监视元素返回 -1。对于工作类，如果未对该工作类指定 COLLECT AGGREGATE ACTIVITY DATA EXTENDED 工作操作，那么此监视元素将返回 -1。对于工作负载而言，当工作负载的 COLLECT AGGREGATE ACTIVITY DATA 设置为 NONE 或 BASE 时，此监视元素返回 -1。 单位为毫秒。

对于服务类而言，将对活动进入系统时所借助于的服务子类计算到达之间时间平均值。使用 REMAP ACTIVITY 操作在服务子类之间重新映射活动时，活动重新映射到的服务子类的 coord_act_interarrival_time_avg 不受影响。

表 444. 事件监视信息

事件类型	逻辑数据分组	监视开关
统计信息	event_scstats	-
统计信息	event_wcstats	-
统计信息	event_wlstats	-

用法

使用此统计信息来确定在与此服务子类、工作负载或工作类相关联的嵌套级别 0 处协调程序活动到达间隔时间的算术平均值。

到达间隔时间可用于确定到达速率，即到达间隔时间的倒数。另外，还可以使用此平均值来确定用于活动到达间隔时间直方图的直方图模板是否合适。根据活动到达间隔时间直方图来计算平均活动到达间隔时间。将计算出来的平均值与此监视元素进行比较。如果计算出来的平均值偏离了此监视元素报告的真实平均值，那么考虑修改活动到达间隔时间直方图的直方图模板并使用更为适合您的数据的一组 bin 值。

coord_act_lifetime_avg -“平均协调程序活动生存期”监视元素

自从上次重置以来，在嵌套级别 0 与此服务子类、工作负载或工作类相关联的协调程序活动的生存期算术平均值。如果内部跟踪的平均值已溢出，那么将返回值 -2。对于服务子类，当服务子类的 COLLECT AGGREGATE ACTIVITY DATA 设置为 NONE 时，此监视元素返回 -1。对于工作类，如果未对该工作类指定 COLLECT AGGREGATE ACTIVITY DATA 工作操作，那么此监视元素将返回 -1。对于工作负载而言，当工作负载的 COLLECT AGGREGATE ACTIVITY DATA 设置为 NONE 时，此监视元素返回 -1。 单位为毫秒。

对于服务类而言，使用 REMAP ACTIVITY 操作在服务子类之间重新映射活动时，只有完成活动的最终服务类的 coord_act_lifetime_avg 平均值受影响。

表 445. 表函数监视信息

表函数	监视元素收集级别
WLM_GET_SERVICE_SUBCLASS_STATS 表函数 - 返回服务子类的统计信息	COLLECT AGGREGATE ACTIVITY DATA
WLM_GET_WORKLOAD_STATS 表函数 - 返回工作负载统计信息	COLLECT AGGREGATE ACTIVITY DATA

表 446. 事件监视信息

事件类型	逻辑数据分组	监视开关
统计信息	event_scstats	-
统计信息	event_wcstats	-
统计信息	event_wlstats	-

用法

使用此统计信息来确定与已完成或中止的服务子类、工作负载或工作类相关联的协调程序活动生存期的算术平均值。

另外，还可以使用此统计信息来确定用于活动生存期直方图的直方图模板是否合适。根据活动生存期直方图计算平均活动生存期。将计算出来的平均值与此监视元素进行比较。如果计算出来的平均值偏离了此监视元素报告的真实平均值，那么考虑修改活动生存期直方图的直方图模板并使用更为适合您的数据的一组 bin 值。

coord_act_lifetime_top -“协调程序活动生存期顶部”监视元素

在所有嵌套级别计量的协调程序活动生存期的高水位标记。单位为毫秒。对于服务类，当服务类的 COLLECT AGGREGATE ACTIVITY DATA 设置为 NONE 时，此监视元素返回 -1。对于工作类，如果未对该工作类指定 COLLECT AGGREGATE ACTIVITY DATA 工作操作，那么此监视元素将返回 -1。对于工作负载而言，当工作负载的 COLLECT AGGREGATE ACTIVITY DATA 设置为 NONE 时，此监视元素返回 -1。

在同时使用 REMAP ACTIVITY 操作在服务子类之间重新映射活动的情况下，要有效地将此统计信息与服务类配合使用，必须将任何给定服务子类的 coord_act_lifetime_top 高水位标记与相同重新映射阈值所影响的其他子类的该高水位标记进行聚集。这是因为，活动在被重新映射阈值重新映射到另一服务子类后将完成，该活动被重新映射前在其他服务子类中的耗用时间只计入在其中完成该活动的服务类。

表 447. 表函数监视信息

表函数	监视元素收集级别
WLM_GET_SERVICE_SUBCLASS_STATS 表函数 - 返回服务子类的统计信息	COLLECT AGGREGATE ACTIVITY DATA
WLM_GET_WORKLOAD_STATS 表函数 - 返回工作负载统计信息	COLLECT AGGREGATE ACTIVITY DATA

表 448. 事件监视信息

事件类型	逻辑数据分组	监视开关
统计信息	event_wcstats	-
统计信息	event_scstats	-
统计信息	event_wlstats	-

用法

此元素可用来帮助确定活动生存期的域值是否有效，还可以帮助确定如何配置这些阈值。

coord_agent_tid -“协调代理程序引擎可分派单元标识”监视元素

应用程序的协调代理程序的引擎可分派单元 (EDU) 标识。

表 449. 事件监视信息

事件类型	逻辑数据分组	监视开关
锁定	lock_participants	

coord_act_queue_time_avg -“平均协调程序活动队列时间”监视元素

自最后一次重置以来，在嵌套级别 0 处的协调程序 DML 活动估计成本的算术平均值与此服务子类或工作类相关联。如果内部跟踪的平均值已溢出，那么将返回值 -2。对于服务子类，当服务子类的 COLLECT AGGREGATE ACTIVITY DATA 设置为 NONE 时，此监视元素返回 -1。对于工作类，如果未对该工作类指定 COLLECT AGGRE-

GATE ACTIVITY DATA 工作操作，那么此监视元素将返回 -1。对于工作负载而言，当工作负载的 COLLECT AGGREGATE ACTIVITY DATA 设置为 NONE 时，此监视元素返回 -1。 单位为毫秒。

对于服务类而言，队列时间只计入在其中完成或中止该活动的服务子类。使用 REMAP ACTIVITY 操作在服务子类之间重新映射活动时，活动所映射到但未完成的服务子类的 coord_act_queue_time_avg 平均值不受影响。

表 450. 表函数监视信息

表函数	监视元素收集级别
WLM_GET_SERVICE_SUBCLASS_STATS 表函数 - 返回服务子类的统计信息	COLLECT AGGREGATE ACTIVITY DATA
WLM_GET_WORKLOAD_STATS 表函数 - 返回工作负载统计信息	COLLECT AGGREGATE ACTIVITY DATA

表 451. 事件监视信息

事件类型	逻辑数据分组	监视开关
统计信息	event_scstats	始终收集
统计信息	event_wcstats	始终收集
统计信息	event_wlstats	始终收集

用法

使用此统计信息来确定与已完成或中止的服务子类、工作负载或工作类相关联的协调程序活动队列时间的算术平均值。

另外，还可以使用此统计信息来确定用于活动队列时间直方图的直方图模板是否合适。根据活动队列时间直方图来计算平均活动队列时间。将计算出来的平均值与此监视元素进行比较。如果计算出来的平均值偏离了此监视元素报告的真实平均值，那么考虑修改活动队列时间直方图的直方图模板并使用更为适合您的数据的一组 bin 值。

coord_act_rejected_total -“被拒绝的协调程序活动总数”监视元素

自最后一次重置后，在任何嵌套级别被拒绝而未被允许执行的协调程序活动总数。此计数器在活动被预测性阈值或阻止执行的工作操作阻止执行时更新。对于服务类，此值在活动完成时更新。对于工作负载，此值在其工作单元结束时由每个工作负载项更新。

表 452. 表函数监视信息

表函数	监视元素收集级别
WLM_GET_SERVICE_CLASS_WORKLOAD_OCCURRENCES 表函数 - 列示工作负载出现次数	ACTIVITY METRICS BASE
WLM_GET_SERVICE_SUBCLASS_STATS 表函数 - 返回服务子类的统计信息	COLLECT AGGREGATE ACTIVITY DATA
WLM_GET_WORKLOAD_STATS 表函数 - 返回工作负载统计信息	ACTIVITY METRICS BASE

表 453. 事件监视信息

事件类型	逻辑数据分组	监视开关
统计信息	event_scstats	-
统计信息	event_wlstats	-

用法

此元素可用来帮助确定阻止执行的预测性阈值或工作操作是否有效，以及它们的限制是否太严格。

coord_agent_pid -“协调代理程序标识”监视元素

应用程序的协调代理程序的引擎可分派单元 (EDU) 标识。除在 Linux 操作系统上以外，EDU 标识与线程标识映射。在 Linux 操作系统上，EDU 标识是 DB2 生成的唯一标识。

表 454. 快照监视信息

快照级别	逻辑数据分组	监视开关
应用程序	appl_info	基本

表 455. 事件监视信息

事件类型	逻辑数据分组	监视开关
锁定	-	-

用法

可以使用此元素来将数据库系统监视器信息链接至其他诊断信息源，如系统跟踪。

coord_agents_top -“最大协调代理程序数”

同时工作的最大协调代理程序数。

表 456. 快照监视信息

快照级别	逻辑数据分组	监视开关
数据库管理器	db2	基本
数据库	dbase	基本

用法

如果协调代理程序的峰值数目显示此节点的工作负载过高，可通过更改 **max_coordagents** 配置参数来降低此上限。

coord_member -“协调程序成员”监视元素

应用程序的协调成员。

表 457. 表函数监视信息

表函数	监视元素收集级别
MON_GET_ACTIVITY_DETAILS 表函数 - 获取完整的活动详细信息（在 XML 文档 DETAILS 中报告）	ACTIVITY METRICS BASE
MON_GET_CONNECTION 表函数 - 获取连接度量值	ACTIVITY METRICS BASE
MON_GET_CONNECTION_DETAILS 表函数 - 获取详细的连接度量值（在 XML 文档 DETAILS 中报告）	ACTIVITY METRICS BASE
MON_GET_UNIT_OF_WORK 表函数 - 获取工作单元度量值	ACTIVITY METRICS BASE
MON_GET_UNIT_OF_WORK_DETAILS 表函数 - 获取详细的工作单元度量值	ACTIVITY METRICS BASE
WLM_GET_SERVICE_CLASS_WORKLOAD_OCCURRENCES 表函数 - 列示工作负载出现次数	ACTIVITY METRICS BASE
WLM_GET_WORKLOAD_OCCURRENCE_ACTIVITIES 表函数 - 返回活动列表	ACTIVITY METRICS BASE

表 458. 事件监视信息

事件类型	逻辑数据分组	监视开关
工作单元	-	始终收集
变更历史记录	changesummary	始终收集

coord_node -“协调节点”

在多节点系统中，这是应用程序连接至实例的节点的节点号。

表 459. 快照监视信息

快照级别	逻辑数据分组	监视开关
应用程序	appl	基本

表 460. 事件监视信息

事件类型	逻辑数据分组	监视开关
连接	event_conn	始终收集

用法 每个连接的应用程序都有一个协调程序节点为其提供服务。

coord_partition_num -“协调程序分区号”监视元素

工作单元或活动的协调程序分区。在多分区系统中，应用程序将从协调程序分区中连接到数据库。

表 461. 表函数监视信息

表函数	监视元素收集级别
WLM_GET_SERVICE_CLASS_WORKLOAD_OCCURRENCES 表函数 - 列示工作负载出现次数	ACTIVITY METRICS BASE
WLM_GET_WORKLOAD_OCCURRENCE_ACTIVITIES 表函数 - 返回活动列表	ACTIVITY METRICS BASE

表 462. 事件监视信息

事件类型	逻辑数据分组	监视开关
工作单元	-	始终收集
活动	event_activity	始终收集
阈值违例	event_thresholdviolations	始终收集

用法

对于那些在除协调程序分区以外的分区中有记录的活动或工作单元，此元素用来标识它们的协调程序分区。

coord_stmt_exec_time -“协调代理程序执行语句的时间”监视元素

此成员上的协调代理程序执行此语句所花的总时间。此值以毫秒计。

表 463. 表函数监视信息

表函数	监视元素收集级别
MON_FORMAT_XML_METRICS_BY_ROW - 获取所有度量值的基于已格式化行的输出	不适用; 报告 XML 文档中作为格式化函数的输入提供的所有元素
MON_GET_ACTIVITY_DETAILS 表函数 - 获取完整的活动详细信息（在 XML 文档 DETAILS 中报告）	ACTIVITY METRICS BASE
MON_GET_PKG_CACHE_STMT 表函数 - 获取程序包高速缓存中的 SQL 语句活动度量值	ACTIVITY METRICS BASE
MON_GET_PKG_CACHE_STMT_DETAILS 表函数 - 获取程序包高速缓存条目的详细度量值	ACTIVITY METRICS BASE

表 464. 事件监视信息

事件类型	逻辑数据分组	监视开关
活动	event_activitymetrics	ACTIVITY METRICS BASE
程序包高速缓存	在 activity_metrics 文档中报告。	ACTIVITY METRICS BASE

corr_token -“DRDA 关联标记”

DRDA AS 关联标记。

表 465. 快照监视信息

快照级别	逻辑数据分组	监视开关
应用程序	appl_info	基本
应用程序	appl	基本

表 466. 事件监视信息

事件类型	逻辑数据分组	监视开关
连接	event_connheader	始终收集

用法 DRDA 关联标记用于关联应用程序服务器与应用程序请求器之间的处理。它是出现错误时转储至日志的标识，可使用该标识来标识存在错误的对话。在某些情况下，它将成为对话的 LUWID。

如果通信未使用 DRDA，那么此元素返回 *appl_id*（请参阅 appl_id）。

如果要使用数据库系统监视器 API，那么注意 API 常量 SQLM_APPLID_SZ 用于定义此元素的长度。

cost_estimate_top -“最高估计成本”监视元素

服务子类或工作类中所有嵌套级别的 DML 活动估计成本的高水位标记。对于服务子类，当服务子类的 COLLECT AGGREGATE ACTIVITY DATA 设置为 NONE 时，此监视元素返回 -1。对于工作类，如果未对该工作类指定 COLLECT AGGREGATE ACTIVITY DATA 工作操作，那么此监视元素将返回 -1。

对于服务类而言，DML 活动的估计成本只计入活动从其中进入系统的服务子类。使用 REMAP ACTIVITY 操作在服务子类之间重新映射活动时，活动重新映射到的服务子类的 cost_estimate_top 不受影响。

表 467. 事件监视信息

事件类型	逻辑数据分组	监视开关
统计信息	event_scstats	-
统计信息	event_wcstats	-
统计信息	event_wlstats	-

用法

使用此元素可确定在收集的时间间隔内，成员对服务类、工作负载或工作类达到的最高 DML 活动估计成本。

count -“事件监视器溢出数”

发生的连续溢出数。

元素标识

计数

元素类型

计数器

表 468. 表函数监视信息

表函数	监视元素收集级别
MON_FORMAT_XML_COMPONENT _TIMES_BY_ROW - 获取基于已格式化行的组件次数	不适用; 报告 XML 文档中作为格式化函数的输入提供的所有元素
MON_FORMAT_XML_TIMES_BY_ROW - - 获取基于已格式化行的组合层次结构等待时间和处理时间	不适用; 报告 XML 文档中作为格式化函数的输入提供的所有元素
MON_FORMAT_XML_WAIT_TIMES_BY _ROW - 获取等待次数的已格式化的基于行的输出	不适用; 报告 XML 文档中作为格式化函数的输入提供的所有元素

表 469. 事件监视信息

事件类型	逻辑数据分组	监视开关
溢出记录	event_overflow	始终收集

用法 可使用此元素来了解丢失的监视器数据量。

事件监视器对一组连续溢出发送一个溢出记录。

cpu_configured -“已配置的 CPU 数”监视元素

此主机上操作系统知道的处理器数。

表 470. 表函数监视信息

表函数	监视元素收集级别
ENV_GET_SYSTEM_RESOURCES 表函数 - 返回系统信息	ACTIVITY METRICS BASE

cpu_cores_per_socket -“每个套接字的 CPU 核心数”监视元素

此主机上的处理器数。在单核心系统上，此值为 1。

表 471. 表函数监视信息

表函数	监视元素收集级别
ENV_GET_SYSTEM_RESOURCES 表函数 - 返回系统信息	ACTIVITY METRICS BASE

cpu_hmt_degree -“逻辑 CPU 数”监视元素

在支持硬件多线程的系统上，似乎因为多线程而显示的逻辑处理器数。在不支持多线程的系统上，此值为 1。

表 472. 表函数监视信息

表函数	监视元素收集级别
ENV_GET_SYSTEM_RESOURCES 表函数 - 返回系统信息	ACTIVITY METRICS BASE

cpu_idle -“处理器空闲时间”监视元素

处理器空闲时间，以处理器 tick 数表示。仅对 Windows、AIX 和 Linux 系统报告。此计量表示系统上所有处理器的聚集。

表 473. 表函数监视信息

表函数	监视元素收集级别
ENV_GET_SYSTEM_RESOURCES 表函数 - 返回系统信息	ACTIVITY METRICS BASE

用法

- 此计量表示系统上所有处理器的聚集。
- 在 AIX 上，此度量是针对运行 DB2 服务器的工作负载分区 (WPAR) 和逻辑分区 (LPAR) 报告的。
- 可使用此监视元素和相关处理器计时器元素来计算主机系统上特定时间间隔的处理器利用率。要以百分比来计算处理器利用率，请执行以下步骤:
  1. 在时间间隔的开始使用 ENV_GET_SYSTEM_RESOURCES 函数来检索以下度量值:
     - cpu_user_{t1} = **cpu_user**
     - cpu_system_{t1} = **cpu_system**
     - cpu_idle_{t1} = **cpu_idle**
     - cpu_wait_{t1} = **cpu_wait**
  2. 重复以上步骤以在要对其计算处理器利用率的时间间隔结束时确定相同度量的时间戳记:
     - cpu_user_{t2} = **cpu_user**
     - cpu_system_{t2} = **cpu_system**
     - cpu_idle_{t2} = **cpu_idle**
     - cpu_iowait_{t2} = **cpu_iowait**
  3. 使用以下公式来计算处理器利用率:

$$100 \times \frac{(cpu\_system_{t2} - cpu\_system_{t1}) + (cpu\_user_{t2} - cpu\_user_{t1})}{(cpu\_system_{t2} - cpu\_system_{t1}) + (cpu\_user_{t2} - cpu\_user_{t1}) + (cpu\_idle_{t2} - cpu\_idle_{t1}) + (cpu\_iowait_{t2} - cpu\_iowait_{t1})}$$

cpu_iowait -“IO 等待时间”监视元素

等待 IO 所消耗的时间（Linux 和 UNIX）；接收和处理硬件中断所消耗的时间 (Windows)，以处理器 tick 数表示。仅对 Windows、AIX 和 Linux 系统报告。此计量表示系统上所有处理器的聚集。

表 474. 表函数监视信息

表函数	监视元素收集级别
ENV_GET_SYSTEM_RESOURCES 表函数 - 返回系统信息	ACTIVITY METRICS BASE

用法

- 此计量表示系统上所有处理器的聚集。
- 在 AIX 上，此度量是针对运行 DB2 服务器的工作负载分区 (WPAR) 和逻辑分区 (LPAR) 报告的。
- 可使用此监视元素和相关处理器计时器元素来计算主机系统上特定时间间隔的处理器利用率。要以百分比来计算处理器利用率，请执行以下步骤:
  1. 在时间间隔的开始使用 ENV_GET_SYSTEM_RESOURCES 函数来检索以下度量值:
     - cpu_user_{t1} = **cpu_user**
     - cpu_system_{t1} = **cpu_system**
     - cpu_idle_{t1} = **cpu_idle**
     - cpu_wait_{t1} = **cpu_wait**
  2. 重复以上步骤以在要对其计算处理器利用率的时间间隔结束时确定相同度量的时间戳记:
     - cpu_user_{t2} = **cpu_user**
     - cpu_system_{t2} = **cpu_system**
     - cpu_idle_{t2} = **cpu_idle**
     - cpu_iowait_{t2} = **cpu_iowait**
  3. 使用以下公式来计算处理器利用率:

$$100 \times \frac{(cpu\_system_{t2} - cpu\_system_{t1}) + (cpu\_user_{t2} - cpu\_user_{t1})}{(cpu\_system_{t2} - cpu\_system_{t1}) + (cpu\_user_{t2} - cpu\_user_{t1}) + (cpu\_idle_{t2} - cpu\_idle_{t1}) + (cpu\_iowait_{t2} - cpu\_iowait_{t1})}$$

cpu_limit -“WLM 分派器 CPU 限制”监视元素

为服务类配置的 WLM 分派器 CPU 限制。

表 475. 表函数监视信息

表函数	监视元素收集级别
MON_SAMPLE_SERVICE_CLASS_METRICS - 获取样本服务类度量值	ACTIVITY METRICS BASE

cpu_load_long -“处理器负载（长时间窗）”监视元素

由系统定义的长期的处理器负载。例如，过去 10 分钟或 15 分钟的平均处理器负载。针对除 Windows 以外的所有平台报告。

表 476. 表函数监视信息

表函数	监视元素收集级别
ENV_GET_SYSTEM_RESOURCES 表函数 - 返回系统信息	ACTIVITY METRICS BASE

cpu_load_medium -“处理器负载（中时间窗）”监视元素

由系统定义的中期的处理器负载。例如，过去 5 分钟或 10 分钟的平均处理器负载。针对除 Windows 以外的所有平台报告。

表 477. 表函数监视信息

表函数	监视元素收集级别
ENV_GET_SYSTEM_RESOURCES 表函数 - 返回系统信息	ACTIVITY METRICS BASE

cpu_load_short -“处理器负载（短时间窗）”监视元素

由系统定义的短期的处理器负载。例如，过去 1 分钟或 5 分钟的平均处理器负载。针对除 Windows 以外的所有平台报告。

表 478. 表函数监视信息

表函数	监视元素收集级别
ENV_GET_SYSTEM_RESOURCES 表函数 - 返回系统信息	ACTIVITY METRICS BASE

cpu_online -“联机 CPU 数”监视元素

此主机上当前联机的处理器数。

表 479. 表函数监视信息

表函数	监视元素收集级别
ENV_GET_SYSTEM_RESOURCES 表函数 - 返回系统信息	ACTIVITY METRICS BASE

cpu_share_type -“WLM 分派器 CPU 份额类型”监视元素

为服务类配置的 WLM 分派器 CPU 份额类型。可能的值为 soft 和 hard。

表 480. 表函数监视信息

表函数	监视元素收集级别
MON_SAMPLE_SERVICE_CLASS_METRICS - 获取样本服务类度量值	ACTIVITY METRICS BASE

cpu_shares -“WLM 分派器 CPU 共享”监视元素

为服务类配置的 WLM 分派器 CPU 共享数。

表 481. 表函数监视信息

表函数	监视元素收集级别
MON_SAMPLE_SERVICE_CLASS_METRICS - 获取样本服务类度量值	ACTIVITY METRICS BASE

cpu_speed -“CPU 时钟速度”监视元素

此主机上处理器的时钟速度，以 MHz 计。

表 482. 表函数监视信息

表函数	监视元素收集级别
ENV_GET_SYSTEM_RESOURCES 表函数 - 返回系统信息	ACTIVITY METRICS BASE

cpu_system -“内核时间”监视元素

运行内核代码所消耗的时间，以处理器 tick 数表示。仅对 Windows、AIX 和 Linux 系统报告。此计量表示系统上所有处理器的聚集。

表 483. 表函数监视信息

表函数	监视元素收集级别
ENV_GET_SYSTEM_RESOURCES 表函数 - 返回系统信息	ACTIVITY METRICS BASE
ENV_GET_DB2_SYSTEM_RESOURCES 表函数 - 返回 DB2(r) 系统信息	ACTIVITY METRICS BASE

用法

- 此计量表示系统上所有处理器的聚集。
- 在 AIX 上，此度量是针对运行 DB2 服务器的工作负载分区 (WPAR) 和逻辑分区 (LPAR) 报告的。
- 可使用此监视元素和相关处理器计时器元素来计算主机系统上特定时间间隔的处理器利用率。要以百分比来计算处理器利用率，请执行以下步骤:
  1. 在时间间隔的开始使用 ENV_GET_SYSTEM_RESOURCES 函数来检索以下度量值:
     - cpu_user_{t1} = **cpu_user**
     - cpu_system_{t1} = **cpu_system**
     - cpu_idle_{t1} = **cpu_idle**
     - cpu_wait_{t1} = **cpu_wait**

2. 重复以上步骤以在要对其计算处理器利用率的时间间隔结束时确定相同度量的时间戳记:
   - cpu_user_{t2} = **cpu_user**
   - cpu_system_{t2} = **cpu_system**
   - cpu_idle_{t2} = **cpu_idle**
   - cpu_iowait_{t2} = **cpu_iowait**
3. 使用以下公式来计算处理器利用率:

$$100 \times \frac{(cpu\_system_{t2} - cpu\_system_{t1}) + (cpu\_user_{t2} - cpu\_user_{t1})}{(cpu\_system_{t2} - cpu\_system_{t1}) + (cpu\_user_{t2} - cpu\_user_{t1}) + (cpu\_idle_{t2} - cpu\_idle_{t1}) + (cpu\_iowait_{t2} - cpu\_iowait_{t1})}$$

cpu_timebase -“时基寄存器递增频率”监视元素

时基寄存器的递增频率，以 Hz 计。仅适用于 Linux 和 PowerPC® 系统

表 484. 表函数监视信息

表函数	监视元素收集级别
ENV_GET_SYSTEM_RESOURCES 表函数 - 返回系统信息	ACTIVITY METRICS BASE

cpu_total -“CPU 数”监视元素

此主机上的处理器数。

表 485. 表函数监视信息

表函数	监视元素收集级别
ENV_GET_SYSTEM_RESOURCES 表函数 - 返回系统信息	ACTIVITY METRICS BASE

用法

针对此监视元素报告的数目在不同操作环境中有不同意义。例如，从 Windows 系统返回时，**cpu_total** 是指已安装的处理器总数；在 AIX 上，它表示已配置的处理器数。

cpu_usage_total -“处理器使用情况”监视元素

此主机上的整体处理器使用情况（包括内核处理时间），以百分比表示。仅对 AIX、Linux 和 Windows 系统报告。

表 486. 表函数监视信息

表函数	监视元素收集级别
ENV_GET_SYSTEM_RESOURCES 表函数 - 返回系统信息	ACTIVITY METRICS BASE

cpu_user -“非内核处理时间”监视元素

运行用户（非内核）代码所消耗的时间，以处理器 tick 数表示。仅对 Windows、AIX 和 Linux 系统报告。此计量表示系统上所有处理器的聚集。

表 487. 表函数监视信息

表函数	监视元素收集级别
ENV_GET_SYSTEM_RESOURCES 表函数 - 返回系统信息	ACTIVITY METRICS BASE
ENV_GET_DB2_SYSTEM_RESOURCES 表函数 - 返回 DB2(r) 系统信息	ACTIVITY METRICS BASE

用法

- 此计量表示系统上所有处理器的聚集。
- 在 AIX 上，此度量是针对运行 DB2 服务器的工作负载分区 (WPAR) 和逻辑分区 (LPAR) 报告的。
- 可使用此监视元素和相关处理器计时器元素来计算主机系统上特定时间间隔的处理器利用率。要以百分比来计算处理器利用率，请执行以下步骤:
  1. 在时间间隔的开始使用 ENV_GET_SYSTEM_RESOURCES 函数来检索以下度量值:
     - cpu_user_{t1} = **cpu_user**
     - cpu_system_{t1} = **cpu_system**
     - cpu_idle_{t1} = **cpu_idle**
     - cpu_wait_{t1} = **cpu_wait**
  2. 重复以上步骤以在要对其计算处理器利用率的时间间隔结束时确定相同度量的时间戳记:
     - cpu_user_{t2} = **cpu_user**
     - cpu_system_{t2} = **cpu_system**
     - cpu_idle_{t2} = **cpu_idle**
     - cpu_iowait_{t2} = **cpu_iowait**
  3. 使用以下公式来计算处理器利用率:

$$100 \times \frac{(cpu\_system_{t2} - cpu\_system_{t1}) + (cpu\_user_{t2} - cpu\_user_{t1})}{(cpu\_system_{t2} - cpu\_system_{t1}) + (cpu\_user_{t2} - cpu\_user_{t1}) + (cpu\_idle_{t2} - cpu\_idle_{t1}) + (cpu\_iowait_{t2} - cpu\_iowait_{t1})}$$

cpu_utilization -“CPU 利用率”监视元素

特定逻辑分区上的服务类或工作负载消耗的总 CPU 时间除以给定时间段内主机或 LPAR 上可用的 CPU 时间量。

表 488. 表函数监视信息

表函数	监视元素收集级别
MON_SAMPLE_SERVICE_CLASS_METRICS - 获取样本服务类度量值	REQUEST METRICS BASE

表 488. 表函数监视信息 (续)

表函数	监视元素收集级别
MON_SAMPLE_WORKLOAD_METRICS - 获取样本工作负载度量值	REQUEST METRICS BASE
WLM_GET_SERVICE_SUBCLASS_STATS 表函数 - 返回服务子类的统计信息	REQUEST METRICS BASE
WLM_GET_WORKLOAD_STATS 表函数 - 返回工作负载统计信息	REQUEST METRICS BASE

表 489. 事件监视信息

事件类型	逻辑数据分组	监视开关
统计信息	event_scstats（在 details_xml 文档中报告）	REQUEST METRICS BASE
统计信息	event_wlstats（在 details_xml 文档中报告）	REQUEST METRICS BASE
工作单元	在 system_metrics 文档中报告	REQUEST METRICS BASE

用法

如果由 WLM_GET_WORKLOAD_STATS 或 WLM_GET_SERVICE_SUBCLASS_STATS 函数返回此监视元素，那么此监视元素表示自上次重置统计信息以来的 CPU 利用率。

如果由 MON_SAMPLE_SERVICE_CLASS_METRICS 或 MON_SAMPLE_WORKLOAD_METRICS 函数返回此监视元素，那么此监视元素表示自执行该函数以来的 CPU 利用率。

cpu_velocity -“CPU 速率”监视元素

CPU 资源争用量的度量（以 0 到 1 的标度来度量），值越低意味着争用程度越高。

CPU 速率通过将服务类中的工作可访问 CPU 的时间量除以访问 CPU 或等待访问 CPU 时所耗的时间来计算的。它给出正在执行的工作的执行效率相对于可达到的执行效率（这类工作永远不需要等待 CPU 时）的度量值。公式如下所示:

```
cpu_velocity = total_cpu_time / (total_cpu_time + total_disp_run_queue_time)
```

wlm_dispatcher 数据库管理器配置参数必须设置为 ON，才能收集 **cpu_velocity**。

表 490. 表函数监视信息

表函数	监视元素收集级别
MON_SAMPLE_SERVICE_CLASS_METRICS - 样本服务类度量值	REQUEST METRICS BASE
MON_SAMPLE_WORKLOAD_METRICS - 样本工作负载度量值	REQUEST METRICS BASE

用法

该分派器在确定服务类或工作负载的优先级时生效（如果该服务类或工作负载在给定时间需要的 CPU 资源量超过所能提供的资源量）。在这类情况下，在该服务类或工作

负载中执行的工作会花时间排队访问 CPU 资源。如果发生此情况，那么分派器可给予这类服务类或工作负载更多 CPU 资源（通过减少它给予另一服务类或工作负载的 CPU 资源）。如果 CPU 速率很高，那么对于此服务类的当前级别的 CPU 需求，此分派器对此服务类的响应时间或吞吐量的改进的影响很小，因为此需求已经得到满足。如果 CPU 速率很低，那么对于此服务类或工作负载的当前级别的 CPU 需求，此分派器对此服务类或工作负载的响应时间或吞吐量的改进有重大影响。

使用此元素来确定服务类或工作负载中执行的工作花在排队使用 CPU 资源上的时间是否占很大比例。如果服务类的 CPU 速率很低，并且您想要提高 CPU 速率，那么可调整 CPU 资源的 WLM 分派器控制，方法是增加 CPU 份额数或增加指定给展示低 CPU 速率的服务类的 CPU 限额。

cputime_threshold_id -“CPU 时间阈值标识”监视元素

应用于此活动的 CPUTIME 阈值的标识。

表 491. 表函数监视信息

表函数	监视元素收集级别
MON_GET_ACTIVITY_DETAILS 表函数 - 获取完整的活动详细信息（在 XML 文档 DETAILS 中报告）	ACTIVITY METRICS BASE

用法

使用此元素来确定应用于此活动的 CPUTIME 阈值（如果有的话）。

cputime_threshold_value -“CPU 时间阈值”监视元素

应用于此活动的 CPUTIME 阈值的上限。

表 492. 表函数监视信息

表函数	监视元素收集级别
MON_GET_ACTIVITY_DETAILS 表函数 - 获取完整的活动详细信息（在 XML 文档 DETAILS 中报告）	ACTIVITY METRICS BASE

用法

使用此元素来确定应用于此活动的 CPUTIME 阈值（如果有的话）。

cputime_threshold_violated -“违反 CPU 时间阈值”监视元素

此监视元素返回“Yes”表明此活动已违反 CPUTIME 阈值。“No”表明此活动尚未违反该阈值。

表 493. 表函数监视信息

表函数	监视元素收集级别
MON_GET_ACTIVITY_DETAILS 表函数 - 获取完整的活动详细信息（在 XML 文档 DETAILS 中报告）	ACTIVITY METRICS BASE

用法

使用此元素来确定此活动是否已违反应用于此活动的 CPUTIME 阈值。

cputimeinsc_threshold_id -“服务类中 CPU 时间阈值标识”监视元素

应用于此活动的 CPUTIMEINSC 阈值的标识。

表 494. 表函数监视信息

表函数	监视元素收集级别
MON_GET_ACTIVITY_DETAILS 表函数 - 获取完整的活动详细信息（在 XML 文档 DETAILS 中报告）	ACTIVITY METRICS BASE

用法

使用此元素来确定应用于此活动的 CPUTIMEINSC 阈值（如果有的话）。

cputimeinsc_threshold_value -“服务类中 CPU 时间阈值”监视元素

应用于此活动的 CPUTIMEINSC 阈值的上限。

表 495. 表函数监视信息

表函数	监视元素收集级别
MON_GET_ACTIVITY_DETAILS 表函数 - 获取完整的活动详细信息（在 XML 文档 DETAILS 中报告）	ACTIVITY METRICS BASE

用法

使用此元素来确定应用于此活动的 CPUTIMEINSC 阈值（如果有的话）。

cputimeinsc_threshold_violated -“违反服务类中 CPU 时间阈值”监视元素

此监视元素返回“Yes”表明此活动已违反 CPUTIMEINSC 阈值。“No”表明此活动尚未违反该阈值。

表 496. 表函数监视信息

表函数	监视元素收集级别
MON_GET_ACTIVITY_DETAILS 表函数 - 获取完整的活动详细信息（在 XML 文档 DETAILS 中报告）	ACTIVITY METRICS BASE

用法

使用此元素来确定此活动是否已违反应用于此活动的 CPUTIMEINSC 阈值。

create_nickname -“创建昵称数”

此元素包含自联合服务器实例启动或数据库监视计数器最后一次重置以后（取较晚者），联合服务器代表任何应用程序对驻留在此数据源上的对象创建昵称的总次数。

表 497. 快照监视信息

快照级别	逻辑数据分组	监视开关
数据库	dbase_remote	基本
应用程序	appl_remote	基本

可将快照监视的计数器重置。

用法

使用此元素来确定此联合服务器实例或应用程序对此数据源执行的 CREATE NICKNAME 活动的级别。CREATE NICKNAME 处理将导致对数据源目录运行多个查询；因此，如果此元素的值很高，那么应确定原因并在可能的情况下限制此活动的执行。

create_nickname_time -“创建昵称响应时间”

此元素包含此数据源处理 CREATE NICKNAME 语句所花的总时间（以毫秒计），这些 CREATE NICKNAME 语句来自在此联合服务器实例上运行的所有应用程序或单个应用程序。响应时间自联合服务器实例启动或数据库监视计数器最后一次重置（取较晚者）起计。 the latest. 响应时间是以联合服务器开始从数据源接收信息以处理 CREATE NICKNAME 语句的时间与检索数据源中的所有必需数据所花时间之差量度的。

表 498. 快照监视信息

快照级别	逻辑数据分组	监视开关
数据库	dbase_remote	时间戳记
应用程序	appl_remote	时间戳记

可将快照监视的计数器重置。

用法 使用此元素来确定用于为此数据源创建昵称所花的实际时间。

creator -“应用程序创建者”

预编译应用程序的用户的授权标识。

表 499. 快照监视信息

快照级别	逻辑数据分组	监视开关
应用程序	stmt	语句
DCS 语句	dcs_stmt	语句

表 500. 事件监视信息

事件类型	逻辑数据分组	监视开关
死锁	event_dlconn	-
语句	event_stmt	-
活动	event_activitystmt	-

用法 将此元素与目录中的程序包段信息的 CREATOR 列一起使用来帮助标识正在处理的 SQL 语句。

如果设置了 CURRENT PACKAGE PATH 专用寄存器，那么 *creator* 值会在 SQL 语句有效期限内反映不同的值。如果快照或事件监视器记录是在解析 PACKAGE PATH 之前获取的，那么 *creator* 值将反映客户机请求中体现的值。如果快照或事件监视器记录是在解析 PACKAGE PATH 之后获取的，那么 *creator* 值将反映已解析程序包的创建者。已解析程序包将是这样一个程序包，其 *creator* 值在 CURRENT PACKAGE PATH SPECIAL REGISTER 中最早出现，并且其程序包名称和唯一标识与客户机请求的程序包名称和唯一标识相匹配。

current_cf_gbp_size -“当前集群高速缓存设施组缓冲池大小”监视元素

当前正在集群高速缓存设施中使用的组缓冲池内存（以大小为 4 KB 的页计）。

表 501. 表函数监视信息

表函数	监视元素收集级别
MON_GET_CF 表函数 - 获取 CF 度量值	ACTIVITY METRICS BASE

current_cf_lock_size -“当前集群高速缓存设施锁定大小”监视元素

当前正在使用的全局锁定内存（以大小为 4 KB 的页计）。

表 502. 表函数监视信息

表函数	监视元素收集级别
MON_GET_CF 表函数 - 获取 CF 度量值	ACTIVITY METRICS BASE

current_cf_sca_size -“当前集群高速缓存设施共享通信区大小”监视元素

当前正在使用的共享通信区内存（以大小为 4 KB 的页计）。

表 503. 表函数监视信息

表函数	监视元素收集级别
MON_GET_CF 表函数 - 获取 CF 度量值	ACTIVITY METRICS BASE

current_cf_mem_size -“当前集群高速缓存设施内存大小”监视元素

当前正在使用的内存总计（以大小为 4 KB 的页计）。

表 504. 表函数监视信息

表函数	监视元素收集级别
MON_GET_CF 表函数 - 获取 CF 度量值	ACTIVITY METRICS BASE

current_active_log -“当前活动日志文件编号”

DB2 数据库系统当前写入的活动日志文件的编号。

表 505. 表函数监视信息

表函数	监视元素收集级别
MON_GET_TRANSACTION_LOG 表函数 - 获取日志信息	ACTIVITY METRICS BASE

表 506. 快照监视信息

快照级别	逻辑数据分组	监视开关
数据库	detail_log	基本

表 507. 事件监视信息

事件类型	逻辑数据分组	监视开关
数据库	event_db	始终收集

用法 将此元素与 *first_active_log* 和 *last_active_log* 元素一起使用以确定活动日志文件的范围。知道活动日志文件的范围可帮助您确定日志文件所需的磁盘空间。

您也可以使用此元素来确定哪些日志文件包含数据，以帮助您标识分割镜像支持所需的日志文件。

current_archive_log -“当前归档日志文件编号”

DB2 数据库系统当前归档的日志文件的文件号。如果 DB2 数据库系统未归档日志文件，那么此元素的值为 SQLM_LOGFILE_NUM_UNKNOWN。

表 508. 表函数监视信息

表函数	监视元素收集级别
MON_GET_TRANSACTION_LOG 表函数 - 获取日志信息	ACTIVITY METRICS BASE

表 509. 快照监视信息

快照级别	逻辑数据分组	监视开关
数据库	detail_log	基本

表 510. 事件监视信息

事件类型	逻辑数据分组	监视开关
数据库	event_db	始终收集

用法 使用此元素来确定归档日志文件时是否存在问题。这类问题包括:

- 归档介质速度太慢
- 归档介质不可用

current_extent -“当前正在移动的扩展数据块”监视元素

表空间重新平衡过程当前正在移动的扩展数据块的数字标识。

表 511. 表函数监视信息

表函数	监视元素收集命令和级别
MON_GET_EXTENT_MOVEMENT_STATUS - 获取扩展数据块移动进度状态度量值	ACTIVITY METRICS BASE

current_request -“当前操作请求”监视元素

代理程序当前正在处理或最近处理的操作。

表 512. 事件监视信息

事件类型	逻辑数据分组	监视开关
锁定	lock_participants	

cursor_name -“游标名称”

对应此 SQL 语句的游标的名称。

元素标识

cursor_name

元素类型

信息

表 513. 快照监视信息

快照级别	逻辑数据分组	监视开关
应用程序	stmt	语句

表 514. 事件监视信息

事件类型	逻辑数据分组	监视开关
带有详细信息的死锁	event_detailed_dlconn	始终收集
语句	event_stmt	始终收集

用法 可使用此元素来标识正在处理的 SQL 语句。将在 SQL SELECT 语句的 OPEN、FETCH、CLOSE 和 PREPARE 上使用此名称。如果未使用游标，那么此字段将为空白。

data_object_pages -“数据对象页数”

表消耗的磁盘页数。此大小仅表示基本表大小。消耗空间的对象为：由 *index_object_pages* 报告的索引对象、由 *lob_object_pages* 报告的 LOB 数据，及由 *long_object_pages* 报告的长数据。

表 515. 快照监视信息

快照级别	逻辑数据分组	监视开关
表	表	基本

表 516. 事件监视信息

事件类型	逻辑数据分组	监视开关
表	event_table	始终收集

用法 此元素提供了一种机制，可用来查看特定表消耗的实际空间量。此元素可与表事件监视器配合使用以跟踪一段时间内表的增长率。

data_object_l_pages -“表数据逻辑页数”监视元素

此表中包含的数据在磁盘上使用的逻辑页数。

表 517. 表函数监视信息

表函数	监视元素收集级别
MON_GET_TABLE 表函数 - 获取表度量值	ACTIVITY METRICS BASE

用法

- 此值可能小于为该对象分配的空间量。将 RECLAIM EXTENTS ONLY 选项与 REORG TABLE 命令配合使用时，可能会发生这种情况。在此情况下，回收的扩展数据块数包括在 MON_GET_TABLE 返回的逻辑页数内。

- 此值可能小于实际上为该对象分配的空间量。使用 TRUNCATE 语句的 REUSE STORAGE 选项时，可能会发生这种情况。此选项导致为该表分配的存储空间继续被分配，尽管该存储空间被视为空。此外，此监视元素的值可能小于逻辑上为该对象分配的空间量，因为逻辑上分配的空间总量包括少量附加元数据。

  要检索对象的逻辑大小或物理大小的准确度量，请使用 ADMIN_GET_TAB_INFO_V97 函数。此函数提供的有关对象大小的信息比您可（通过将针对此监视元素报告的页数乘以页大小）获取的对象大小更准确。

data_partition_id -“数据分区标识”监视元素

与返回的信息相对应的数据分区的标识。

表 518. 表函数监视信息

表函数	监视元素收集级别
ADMINTABINFO 管理视图和 ADMIN_GET_TAB_INFO 表函数 - 检索表大小和状态信息	ACTIVITY METRICS BASE
MON_FORMAT_LOCK_NAME 表函数 - 设置内部锁定名称的格式并返回详细信息	ACTIVITY METRICS BASE
MON_GET_INDEX 表函数 - 获取索引度量值	ACTIVITY METRICS BASE
MON_GET_INDEX_USAGE_LIST 表函数 - 返回索引用法列表中的信息	ACTIVITY METRICS BASE
MON_GET_PAGE_ACCESS_INFO 表函数 - 获取缓冲池页等待信息	ACTIVITY METRICS BASE
MON_GET_TABLE 表函数 - 获取表度量值	ACTIVITY METRICS BASE
MON_GET_TABLE_USAGE_LIST 表函数 - 从表用法列表返回信息	ACTIVITY METRICS BASE
MON_GET_USAGE_LIST_STATUS 表函数 - 返回用法列表的状态	ACTIVITY METRICS BASE

表 519. 快照监视信息

快照级别	逻辑数据分组	监视开关
表	表	基本
锁定	锁定	锁定
锁定	lock_wait	锁定

表 520. 事件监视信息

事件类型	逻辑数据分组	监视开关
表	event_table	-
死锁	event_dlconn	-
带有详细信息的死锁	event_detailed_dlconn	-
死锁	锁定	-

用法

此元素仅适用于分区表和分区索引。否则，此监视元素的值是 NULL。

当返回锁定级别信息时，值 -1 代表一个控制对整个表进行访问的锁定。

datasource_name -“数据源名称”

此元素包含其远程访问信息由联合服务器显示的数据源的名称。此元素对应于 SYSCAT.SERVERS 中的 'SERVER' 列。

元素标识

datasource_name

元素类型

信息

表 521. 快照监视信息

快照级别	逻辑数据分组	监视开关
数据库	dbase_remote	基本
应用程序	appl_remote	基本

用法 使用此元素来标识已收集并正在返回其访问信息的数据源。

datataginsc_threshold_id -“服务类阈值（IN 条件）标识中的数据标记”

应用于此活动的 DATATAGINSC IN 阈值的标识。

表 522. 表函数监视信息

表函数	监视元素收集级别
MON_GET_ACTIVITY_DETAILS 表函数 - 获取完整的活动详细信息	ACTIVITY METRICS BASE

使用说明

- 使用此元素来了解应用于活动的 DATATAGINSC IN 阈值（如果存在）。

datataginsc_threshold_value -“服务类阈值（IN 条件）中的数据标记”

应用于此活动的 DATATAGINSC IN 阈值的数据标记的逗号分隔列表。

表 523. 表函数监视信息

表函数	监视元素收集级别
MON_GET_ACTIVITY_DETAILS 表函数 - 获取完整的活动详细信息	ACTIVITY METRICS BASE

使用说明

- 使用此元素来确定应用于此活动的 DATATAGINSC IN 阈值（如果存在）。

datataginsc_threshold_violated -“违反的服务类阈值（IN 条件）中的数据标记”

指示此活动是否违反了 DATATAGINSC IN 阈值。如果此活动违反了 DATATAGINSC IN 阈值，那么返回 1。如果此活动未违反阈值，那么返回 0。

表 524. 表函数监视信息

表函数	监视元素收集级别
MON_GET_ACTIVITY_DETAILS 表函数 - 获取完整的活动详细信息	ACTIVITY METRICS BASE

使用说明

- 使用此元素来确定此活动是否已违反应用于此活动的 DATATAGINSC IN 阈值。

datatagnotinsc_threshold_id -“服务类阈值（NOT IN 条件）标识中的数据标记”

应用于此活动的 DATATAGINSC NOT IN 阈值的标识。

表 525. 表函数监视信息

表函数	监视元素收集级别
MON_GET_ACTIVITY_DETAILS 表函数 - 获取完整的活动详细信息	ACTIVITY METRICS BASE

使用说明

- 使用此元素来了解应用于活动的 DATATAGINSC NOT IN 阈值（如果存在）。

datatagnotinsc_threshold_value -“服务类阈值（NOT IN 条件）中的数据标记”

应用于此活动的 DATATAGINSC NOT IN 阈值的数据标记的逗号分隔列表。

表 526. 表函数监视信息

表函数	监视元素收集级别
MON_GET_ACTIVITY_DETAILS 表函数 - 获取完整的活动详细信息	ACTIVITY METRICS BASE

使用说明

- 使用此元素来确定应用于此活动的 DATATAGINSC NOT IN 阈值（如果存在）。

datatagnotinsc_threshold_violated -“违反的服务类阈值（NOT IN 条件）中的数据标记”

指示此活动是否违反了 DATATAGINSC NOT IN 阈值。如果此活动违反了 DATATAGINSC NOT IN 阈值，那么返回 1。如果此活动未违反阈值，那么返回 0。

表 527. 表函数监视信息

表函数	监视元素收集级别
MON_GET_ACTIVITY_DETAILS 表函数 - 获取完整的活动详细信息	ACTIVITY METRICS BASE

使用说明

- 使用此元素来确定此活动是否已违反应用于此活动的 DATATAGINSC NOT IN 阈值。

db2_process_id -“DB2 进程标识”监视元素

在所报告成员上运行的 DB2 进程的数字标识。

表 528. 表函数监视信息

表函数	监视元素收集级别
ENV_GET_DB2_SYSTEM_RESOURCES 表函数 - 返回 DB2(r) 系统信息	ACTIVITY METRICS BASE

db2_process_name -“DB2 进程名称”监视元素

在所报告成员上运行的 DB2 进程的名称。

表 529. 表函数监视信息

表函数	监视元素收集级别
ENV_GET_DB2_SYSTEM_RESOURCES 表函数 - 返回 DB2(r) 系统信息	ACTIVITY METRICS BASE

db2_status -“DB2 实例的状态”监视元素

数据库管理器的实例的当前状态。

表 530. 快照监视信息

快照级别	逻辑数据分组	监视开关
数据库管理器	db2	基本

用法

可使用此元素来确定数据库管理器实例的状态。

此元素的值包括:

API 常量	值	描述
SQLM_DB2_ACTIVE	0	数据库管理器实例是活动的。
SQLM_DB2_QUIESCE_PEND	1	实例和实例中的数据库处于停顿-暂挂状态。不允许新建与任何实例数据库的连接，也不能启动新的工作单元。根据停顿请求，将允许活动工作单元完成或立即回滚。
SQLM_DB2_QUIESCED	2	实例和实例中的数据库已停顿。不允许新建与任何实例数据库的连接，也不能启动新的工作单元。

db2start_time -“启动数据库管理器时间戳记”

使用 db2start 命令启动数据库管理器的日期和时间。

元素标识

db2start_time

元素类型

时间戳记

表 531. 快照监视信息

快照级别	逻辑数据分组	监视开关
数据库管理器	db2	基本

表 532. 事件监视信息

事件类型	逻辑数据分组	监视开关
变更历史记录	evmonstart	始终收集

用法

此元素可与 *time_stamp* 监视元素配合使用，以计算自 数据库管理器 启动直到获取快照所耗用的时间。

对于变更历史记录事件监视器，此元素可用于跟踪延迟的数据库管理器配置参数更新的生效时间。

db_conn_time -“数据库激活时间戳记”监视元素

在数据库级别第一次连接至数据库的日期和时间，或者发出激活数据库命令的日期和时间。

表 533. 快照监视信息

快照级别	逻辑数据分组	监视开关
数据库	dbase	时间戳记
表空间	tablespace_list	缓冲池，时间戳记
表	table_list	时间戳记

表 534. 事件监视信息

事件类型	逻辑数据分组	监视开关
工作单元	-	-
变更历史记录	evmonstart	始终收集

用法

将此元素与 **disconn_time** 监视元素配合使用以计算总连接时间。

对于变更历史记录事件监视器，此元素可用于跟踪延迟的数据库配置参数更新时生效。

db_heap_top -“分配的最大数据库堆”

此元素将保留以获取 DB2 版本兼容性。它现在量度内存使用情况，但并非由数据库堆专用。

注: 从 DB2 V9.5 开始，不推荐使用 **db_heap_top** 监视元素。使用此监视元素不会生成错误。但是不会返回有效值。建议不要再使用此监视元素，将来的发行版中可能会将其除去。

表 535. 快照监视信息

快照级别	逻辑数据分组	监视开关
数据库	dbase	基本

表 536. 事件监视信息

事件类型	逻辑数据分组	监视开关
数据库	event_db	始终收集

db_location -“数据库位置”

与应用程序相关的数据库的位置。

表 537. 快照监视信息

快照级别	逻辑数据分组	监视开关
数据库	dbase	基本

用法 确定有关获取快照的应用程序的数据库服务器的相对位置。值包括:

- SQLM_LOCAL
- SQLM_REMOTE

db_name - 数据库名称监视元素

对其收集信息或应用程序连接至的数据库的真实名称。这是创建时给定的数据库名称。

表 538. 表函数监视信息

表函数	监视元素收集级别
MON_GET_AUTO_MAINT_QUEUE 表函数 - 获取有关自动维护作业的信息	ACTIVITY METRICS BASE
MON_GET_CF 表函数 - 获取 CF 度量值	ACTIVITY METRICS BASE
MON_GET_MEMORY_POOL 表函数 - 获取内存池信息	ACTIVITY METRICS BASE
MON_GET_MEMORY_SET 表函数 - 获取内存集合信息	ACTIVITY METRICS BASE
MON_SAMPLE_SERVICE_CLASS_METRICS - 获取样本服务类度量值	ACTIVITY METRICS BASE
MON_SAMPLE_WORKLOAD_METRICS - 获取样本工作负载度量值	ACTIVITY METRICS BASE

表 539. 快照监视信息

快照级别	逻辑数据分组	监视开关
数据库	dbase	基本
数据库	dbase_remote	基本
应用程序	appl_id_info	基本
应用程序	appl_remote	基本
表空间	tablespace_list	缓冲池
缓冲池	bufferpool	缓冲池
表	table_list	表
锁定	db_lock_list	基本
动态 SQL	dynsql_list	基本
DCS 数据库	dcs_dbase	基本
DCS 应用程序	dcs_appl_info	基本

表 540. 事件监视信息

事件类型	逻辑数据分组	监视开关
数据库	event_dbheader	始终收集

用法

可使用此元素来标识应用数据的特定数据库。

对于未使用 DB2 Connect 连接至主机或 System i® 数据库服务器的应用程序，可将此元素与 **db_path** 监视元素配合使用来唯一标识该数据库，并协助使监视器提供的信息的不同级别相关。

db_path -“数据库路径”

数据库在被监视系统上的存储位置的完整路径。

表 541. 快照监视信息

快照级别	逻辑数据分组	监视开关
数据库	dbase	基本
应用程序	appl_id_info	基本
表空间	tablespace_list	缓冲池
缓冲池	bufferpool	缓冲池
表	table_list	表
锁定	db_lock_list	基本
动态 SQL	dynsql_list	基本

表 542. 事件监视信息

事件类型	逻辑数据分组	监视开关
数据库	event_dbheader	始终收集

用法 此元素可与 *db_name* 监视元素配合使用以标识应用数据的特定数据库。

db_status - 数据库状态监视元素

数据库的当前状态。

表 543. 快照监视信息

快照级别	逻辑数据分组	监视开关
数据库	dbase	基本

用法

可使用此元素来确定数据库的状态。

此字段的值是:

API 常量	值	描述
SQLM_DB_ACTIVE	0	数据库是活动的。
SQLM_DB_QUIESCE_PEND	1	数据库处于停顿-暂挂状态。不允许新建与数据库的连接，也不能启动新的工作单元。根据停顿请求，将允许活动工作单元完成或立即回滚。
SQLM_DB_QUIESCED	2	数据库已停顿。不允许新建与数据库的连接，也不能启动新的工作单元。
SQLM_DB_ROLLFWD	3	数据库正在进行前滚。
SQLM_DB_ACTIVE_STANDBY	4	数据库是一个支持读操作的 HADR 备用数据库。
SQLM_DB_STANDBY	5	数据库是 HADR 备用数据库。

db_storage_path -“自动存储器路径”监视元素

此元素显示数据库用于放置自动存储器表空间的位置的完整路径。一个数据库可以有 0 个或多个相关联的存储器路径。

表 544. 表函数监视信息

表函数	监视元素收集级别
ADMIN_GET_STORAGE_PATHS 表函数 - 获取存储器组的存储器路径信息	ACTIVITY METRICS BASE

表 545. 快照监视信息

快照级别	逻辑数据分组	监视开关
数据库	db_sto_path_info	基本

用法

将此元素与 **num_db_storage_paths** 监视元素配合使用来标识与此数据库相关联的存储器路径。

db_storage_path_id -“存储器路径标识”

存储器组中的存储器路径的每个实例的唯一标识。

表 546. 表函数监视信息

表函数	监视元素收集级别
ADMIN_GET_STORAGE_PATHS 表函数 - 获取存储器组的存储器路径信息	ACTIVITY METRICS BASE
MON_GET_CONTAINER 表函数 - 获取表空间容器度量值	ACTIVITY METRICS BASE

db_storage_path_state -“存储器路径状态”监视元素

自动存储器路径状态指示存储器路径是否正被数据库使用。

表 547. 表函数监视信息

表函数	监视元素收集级别
ADMIN_GET_STORAGE_PATHS 表函数 - 获取存储器组的存储器路径信息	ACTIVITY METRICS BASE

表 548. 快照监视信息

快照级别	逻辑数据分组	监视开关
数据库	db_sto_path_info	基本

用法

使用此监视元素来确定存储器路径是否正被数据库使用。可能的值如下所示:

NOT_IN_USE
在指定的数据库分区中，没有任何表空间正在使用此存储器路径。

IN_USE
在指定的数据库分区中，有一些表空间正在使用此存储器路径。

DROP_PENDING
此存储器路径已被删除，但某些表空间仍在使用此路径。以物理方式从数据库中删除存储器路径之前，所有表空间都必须停止使用这些路径。要停止使用已删除的存储器路径，请删除该表空间，或者使用 ALTER TABLESPACE 语句的 REBALANCE 子句对该表空间进行重新平衡。

db_storage_path_with_dpe -“包含数据库分区表达式的存储器路径”监视元素

包含尚未进行求值的数据库分区表达式的自动存储器路径。

表 549. 表函数监视信息

表函数	监视元素收集级别
ADMIN_GET_STORAGE_PATHS 表函数 - 获取存储器组的存储器路径信息	ACTIVITY METRICS BASE

表 550. 快照监视信息

快照级别	逻辑数据分组	监视开关
数据库	db_sto_path_info	基本

用法

当存储器路径包含数据库分区表达式时，请使用此监视元素来确定 CREATE DATABASE 命令或 ALTER DATABASE 语句对数据库指定的存储器路径。

如果存储器路径未包含数据库分区表达式，那么此监视元素将返回空值。

db_work_action_set_id -“数据库工作操作集标识”监视元素

如果此活动已划分到数据库作用域的某个工作类，那么此监视元素显示与该工作类所属的工作类集相关联的工作操作集的标识。否则，此监视元素将显示值 0。

表 551. 表函数监视信息

表函数	监视元素收集命令和级别
WLM_GET_ACTIVITY_DETAILS_COMPLETE（在 XML 文档 DETAILS 中报告）	ACTIVITY METRICS BASE
MON_GET_ACTIVITY_DETAILS 表函数 - 获取完整的活动详细信息	ACTIVITY METRICS BASE

表 552. 事件监视信息

事件类型	逻辑数据分组	监视开关
活动	event_activity	始终收集

用法

可将此元素与 **db_work_class_id** 元素配合使用，以唯一地标识活动的数据库工作类（如果有）。

db_work_class_id -“数据库工作类标识”监视元素

如果此活动已划分到数据库作用域的某个工作类，那么此监视元素将显示该工作类的标识。否则，此监视元素将显示值 0。

表 553. 表函数监视信息

表函数	监视元素收集级别
WLM_GET_ACTIVITY_DETAILS_COMPLETE 表函数 - 获取完整的活动详细信息（在 XML 文档 DETAILS 中报告）	ACTIVITY METRICS BASE

表 554. 事件监视信息

事件类型	逻辑数据分组	监视开关
活动	event_activity	始终收集

用法

可将此元素与 **db_work_action_set_id** 元素配合使用，以唯一地标识活动的数据库工作类（如果有）。

dbpartitionnum -“数据库分区号”监视元素

在分区数据库环境中，此监视元素是数据库成员的数字标识。对于 DB2 Enterprise Server Edition 和在 DB2 pureScale环境中，此值为 0。

表 555. 表函数监视信息

表函数	监视元素收集级别
ADMIN_GET_INDEX_COMPRESS_INFO 表函数 - 返回压缩索引信息	ACTIVITY METRICS BASE
ADMIN_GET_INDEX_INFO 表函数 - 返回索引信息	ACTIVITY METRICS BASE
ADMIN_GET_MSGS 表函数 - 检索由通过 ADMIN_CMD 过程执行的数据移动实用程序生成的消息	ACTIVITY METRICS BASE
ADMIN_GET_STORAGE_PATHS 表函数 - 获取存储器组的存储器路径信息	ACTIVITY METRICS BASE
ADMIN_GET_TAB_COMPRESS_INFO 表函数 - 估算压缩节省量	ACTIVITY METRICS BASE
ADMIN_GET_TAB_DICTIONARY_INFO 表函数 - 报告现有表字典的属性	ACTIVITY METRICS BASE
ADMINTABINFO 管理视图和 ADMIN_GET_TAB_INFO 表函数 - 检索表大小和状态信息	ACTIVITY METRICS BASE

表 555. 表函数监视信息 (续)

表函数	监视元素收集级别
AUDIT_ARCHIVE 过程和表函数 - 归档审计日志文件	ACTIVITY METRICS BASE
DBCFG 管理视图和 DB_GET_CFG 表函数 - 检索数据库配置参数信息	ACTIVITY METRICS BASE
DBPATHS 管理视图和 ADMIN_LIST_DB_PATHS 表函数 - 检索数据库路径	ACTIVITY METRICS BASE
MON_GET_CONTAINER 表函数 - 获取表空间容器度量值	ACTIVITY METRICS BASE
MON_GET_REBALANCE_STATUS 表函数 - 获取表空间的重新平衡进度	ACTIVITY METRICS BASE
MON_GET_TABLE 表函数 - 获取表度量值	ACTIVITY METRICS BASE
MON_GET_TABLESPACE 表函数 - 获取表空间度量值	ACTIVITY METRICS BASE
PD_GET_DIAG_HIST 表函数 - 从给定设施返回记录	ACTIVITY METRICS BASE
PDLOGMSGS_LAST24HOURS 管理视图和 PD_GET_LOG_MSGS 表函数 - 检索问题确定消息	ACTIVITY METRICS BASE
WLM_GET_QUEUE_STATS 表函数 - 返回阈值队列统计信息	ACTIVITY METRICS BASE
WLM_GET_SERVICE_CLASS_AGENTS 表函数 - 列示正在服务类中运行的代理程序	ACTIVITY METRICS BASE
WLM_GET_SERVICE_CLASS_WORKLOAD_OCCURRENCES 表函数 - 列示工作负载出现次数	ACTIVITY METRICS BASE
WLM_GET_SERVICE_SUBCLASS_STATS 表函数 - 返回服务子类的统计信息	ACTIVITY METRICS BASE
WLM_GET_SERVICE_SUPERCLASS_STATS 表函数 - 返回服务超类的统计信息	ACTIVITY METRICS BASE
WLM_GET_WORK_ACTION_SET_STATS 表函数 - 返回工作操作集统计信息	ACTIVITY METRICS BASE
WLM_GET_WORKLOAD_OCCURRENCE_ACTIVITIES 表函数 - 返回活动列表	ACTIVITY METRICS BASE
WLM_GET_WORKLOAD_STATS 表函数 - 返回工作负载统计信息	ACTIVITY METRICS BASE

用法

在 DB2 pureScale环境中，多个成员在单个分区上运行。在这类配置中运行时，系统中所有成员上的存储器物理属性（例如，表空间中的可用页数）是重复的。每个成员报告系统的总准确大小。在多分区配置中，每个分区中的值必须由用户关联才能了解系统的整体值。

dbpartitionnum 监视元素与 **data_partition_id** 监视元素不同，后者用于标识通过根据值对表中数据进行细分所创建的数据分区。

dcs_appl_status -“DCS 应用程序状态”监视元素

DB2 Connect 网关上的 DCS 应用程序的状态。

表 556. 快照监视信息

快照级别	逻辑数据分组	监视开关
DCS 应用程序	dcs_appl_info	基本

用法

此元素用于 DCS 应用程序的问题确定。值包括:

- SQLM_DCS_CONNECTPEND_OUTBOUND

  应用程序已启动从 DB2 Connect 网关至主机数据库的数据库连接，但请求尚未完成。
- SQLM_DCS_UOWWAIT_OUTBOUND

  DB2 Connect 网关正在等待主机数据库应答应用程序的请求。
- SQLM_DCS_UOWWAIT_INBOUND

  已建立从 DB2 Connect 网关至主机数据库的连接并且网关正在等待来自应用程序的 SQL 请求。或者 DB2 Connect 网关正在代表应用程序中的工作单元等待。这通常意味着正在执行应用程序的代码。

dcs_db_name -“DCS 数据库名称”

DCS 目录中编目的 DCS 数据库的名称。

表 557. 快照监视信息

快照级别	逻辑数据分组	监视开关
DCS 数据库	dcs_dbase	基本
DCS 应用程序	dcs_appl_info	基本

用法 此元素用于 DCS 应用程序的问题确定。

ddl_classification -“DDL 分类”

用于描述所执行 DDL 类型的分类。

表 558. 事件监视信息

事件类型	逻辑数据分组	监视开关
变更历史记录	DDLSTMTEXEC	始终收集

用法

DDL 可分为下列其中一个类型:

DDLSTORAGE

改变数据库 DDL、缓冲池 DDL、分区组 DDL、存储器组 DDL 和表空间 DDL 的执行。

DDLWLM

直方图 DDL、服务类 DDL、阈值 DDL、工作操作集 DDL、工作类集 DDL 和工作负载 DDL 的执行。

DDLMONITOR

事件监视器 DDL 和用法列表 DDL 的执行。

DDLSECURITY

审计策略 DDL、授权 DDL、掩码 DDL、许可权角色 DDL、撤销 DDL、安全标号 DDL、安全标号组件 DDL、安全策略 DDL 和可信上下文 DDL 的执行。

DDLSQL

别名 DDL、函数 DDL、方法 DDL、模块 DDL、程序包 DDL、过程 DDL、模式 DDL、同义词 DDL、变换 DDL、触发器 DDL、类型 DDL、变量 DDL 和视图 DDL 的执行。

DDLDATA

索引 DDL、序列 DDL、表 DDL 和临时表 DDL 的执行。

DDLXML

XSROBJECT DDL 的执行。

DDLFEDERATED

昵称/服务器 DDL、类型/用户映射 DDL 和包装器 DDL 的执行。

ddl_sql_stmts -“数据定义语言（DDL）SQL 语句数”

此元素指示执行的 SQL 数据定义语言（DDL）语句数。

表 559. 快照监视信息

快照级别	逻辑数据分组	监视开关
数据库	dbase	基本
应用程序	appl	基本

可将快照监视的计数器重置。

表 560. 事件监视信息

事件类型	逻辑数据分组	监视开关
数据库	event_db	始终收集
连接	event_conn	始终收集

用法 可使用此元素来确定应用程序或数据库级别的数据库活动的级别。DDL 语句的运行成本很高，这是因为它们对系统目录表有影响。因此，如果此元素的值很高，那么应确定原因并在可能的情况下限制此活动的执行。

还可使用此元素并借助以下公式来确定 DDL 活动的百分比:

```
ddl_sql_stmts / 语句总数
```

此信息对于分析应用程序活动和吞吐量非常有用。DDL 语句还会影响:

- 目录高速缓存（通过使该处存储的表描述符信息和权限信息无效并导致因为从系统目录检索信息而产生附加系统使用率）
- 程序包高速缓存（通过使该处存储的段无效并导致因为段重新编译而产生附加系统处理时间）。

DDL 语句的示例包括 CREATE TABLE、CREATE VIEW、ALTER TABLE 和 DROP INDEX。

deadlock_id -“死锁事件标识”

死锁的标识。

元素标识

deadlock_id

元素类型

信息

表 561. 事件监视信息

事件类型	逻辑数据分组	监视开关
死锁	event_deadlock	始终收集
死锁	event_dlconn	始终收集
带有详细信息的死锁	event_detailed_dlconn	始终收集
带有详细信息的死锁的历史纪录	event_detailed_dlconn	始终收集
带有详细信息的死锁的历史纪录	event_stmt_history	始终收集
带有详细信息的死锁的历史记录值	event_data_value	始终收集
带有详细信息的死锁的历史记录值	event_detailed_dlconn	始终收集
带有详细信息的死锁的历史记录值	event_stmt_history	始终收集

用法 在监视应用程序中使用此元素，以将死锁连接和语句历史记录事件记录与死锁事件记录相关联。

deadlock_member -“死锁成员”监视元素

参与者正请求对其锁定的成员。

表 562. 事件监视信息

事件类型	逻辑数据分组	监视开关
锁定	lock_participants	

deadlock_node -“发生死锁的分区号”

发生死锁的分区号。

元素标识

deadlock_node

元素类型

信息

表 563. 事件监视信息

事件类型	逻辑数据分组	监视开关
死锁	event_deadlock	始终收集
死锁	event_dlconn	始终收集
带有详细信息的死锁	event_detailed_dlconn	始终收集

用法 此元素仅与分区数据库有关。在监视应用程序中使用此元素，以将死锁连接事件记录与死锁事件记录相关联。

deadlock_type -“死锁类型”监视元素

所发生的死锁的类型。此值可以是 LOCAL 或 GLOBAL。在局部死锁中，所有参与者在同一成员上运行。在全局死锁中，至少一个死锁参与者在远程成员上运行。

表 564. 事件监视信息

事件类型	逻辑数据分组	监视开关
锁定	lock	

deadlocks -“检测到的死锁数”监视元素

发生的死锁总数。

表 565. 表函数监视信息

表函数	监视元素收集级别
MON_FORMAT_XML_METRICS_BY_ROW - 获取所有度量值的基于已格式化行的输出	不适用；报告 XML 文档中作为格式化函数的输入提供的所有元素
MON_GET_ACTIVITY_DETAILS 表函数 - 获取完整的活动详细信息（在 XML 文档 DETAILS 中报告）	ACTIVITY METRICS BASE
MON_GET_CONNECTION 表函数 - 获取连接度量值	REQUEST METRICS BASE
MON_GET_CONNECTION_DETAILS 表函数 - 获取详细的连接度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE
MON_GET_PKG_CACHE_STMT 表函数 - 获取程序包高速缓存中的 SQL 语句活动度量值	ACTIVITY METRICS BASE
MON_GET_PKG_CACHE_STMT_DETAILS 表函数 - 获取程序包高速缓存条目的详细度量值	ACTIVITY METRICS BASE
MON_GET_SERVICE_SUBCLASS 表函数 - 获取服务子类度量值	REQUEST METRICS BASE

表 565. 表函数监视信息 (续)

表函数	监视元素收集级别
MON_GET_SERVICE_SUBCLASS_DETAILS 表函数 - 获取详细的服务子类度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE
MON_GET_UNIT_OF_WORK 表函数 - 获取工作单元度量值	REQUEST METRICS BASE
MON_GET_UNIT_OF_WORK 表函数 - 获取详细的工作单元度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE
MON_GET_WORKLOAD 表函数 - 获取工作负载度量值	REQUEST METRICS BASE
MON_GET_WORKLOAD_DETAILS 表函数 - 获取详细的工作负载度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE

表 566. 快照监视信息

快照级别	逻辑数据分组	监视开关
数据库	dbase	基本
应用程序	appl	锁定

可将快照监视的计数器重置。

表 567. 事件监视信息

事件类型	逻辑数据分组	监视开关
活动	event_activity（在 details_xml 文档中报告）	ACTIVITY METRICS BASE
活动	event_activitymetrics	ACTIVITY METRICS BASE
统计信息	event_scstats（在 details_xml 文档中报告）	REQUEST METRICS BASE
统计信息	event_wlstats（在 details_xml 文档中报告）	REQUEST METRICS BASE
工作单元	在 system_metrics 文档中报告。	REQUEST METRICS BASE
数据库	event_db	始终收集
连接	event_conn	始终收集
程序包高速缓存	在 activity_metrics 文档中报告。	ACTIVITY METRICS BASE

用法

此元素可指示应用程序遇到争用问题。这些问题可能是由下列情况导致的:

- 数据库发生锁定升级
- 在系统生成的行锁定已足够的情况下应用程序显式锁定了表
- 绑定时应用程序使用了不适当的隔离级别
- 目录表已被锁定以供可重复读

- 应用程序正以不同的顺序获取相同的锁定，从而导致死锁。

可通过确定发生死锁的应用程序（或应用程序进程）来解决问题。然后，您可以修改该应用程序，以使它能够更好地并行运行。但某些应用程序可能无法并行运行。

可以使用连接时间戳记监视元素（**last_reset**、**db_conn_time** 和 **appl_con_time**）来确定死锁的严重性。例如，5 分钟内发生 10 个死锁比 5 小时内发生 10 个死锁要严重得多。

以上列示的相关元素的描述还可提供其他调整建议。

deferred -“延迟”

指示是否延迟对配置参数值的更改。

表 568. 事件监视信息

事件类型	逻辑数据分组	监视开关
变更历史记录	DBDBMCFG	始终收集

用法

变更历史记录事件监视器将此值收集为:

Y 更改延迟至下一次数据库激活

N 更改立即生效

degree_parallelism -“并行度”

绑定查询时请求的并行度。

元素标识

degree_parallelism

元素类型

信息

表 569. 快照监视信息

快照级别	逻辑数据分组	监视开关
应用程序	stmt	语句

用法 与 agents_top 配合使用来确定查询是否达到最大级别的并行性。

del_keys_cleaned -“清除伪删除键数目”监视元素

已清除的伪删除键的数目。

表 570. 表函数监视信息

表函数	监视元素收集级别
MON_GET_INDEX 表函数 - 获取索引度量值	ACTIVITY METRICS BASE

delete_sql_stmts -“删除数”

此元素包含自联合服务器实例启动或数据库监视计数器最后一次重置以后（取较晚者），联合服务器代表任何应用程序对此数据源发出 DELETE 语句的总次数。

表 571. 快照监视信息

快照级别	逻辑数据分组	监视开关
数据库	dbase_remote	基本
应用程序	appl_remote	基本

可将快照监视的计数器重置。

用法 使用此元素来确定联合服务器或应用程序对此数据源执行的数据库活动的级别。

还可使用此元素并借助以下公式来确定联合服务器或应用程序对此数据源执行的写入活动的百分比:

```
write_activity =
        (INSERT 语句数 + UPDATE 语句数 + DELETE 语句数 ) /
        (SELECT 语句数 + INSERT 语句数 + UPDATE 语句数 +
        DELETE 语句数)
```

delete_time -“删除响应时间”

此元素包含此数据源响应 DELETE 所花的总时间（以毫秒计），这些 DELETE 来自联合服务器启动后或数据库监视计数器上一次重置后（取较晚者）在此联合服务器实例上运行的所有应用程序或单个应用程序。

响应时间是以联合服务器将 DELETE 语句提交给数据源的时间与数据源响应联合服务器以指示已处理 DELETE 的时间之差量度的。

表 572. 快照监视信息

快照级别	逻辑数据分组	监视开关
数据库	dbase_remote	时间戳记
应用程序	appl_remote	时间戳记

可将快照监视的计数器重置。

用法 使用此元素来确定等待处理对此数据源执行 DELETE 语句所花的实际时间。此信息对于容量规划和容量调整非常有用。

destination_service_class_id -“目标服务类标识”监视元素

生成此元素所属的阈值违例记录时此活动重新映射到的服务子类的标识。对于除 REMAP ACTIVITY 以外的任何阈值操作，此元素的值为零。

表 573. 事件监视信息

事件类型	逻辑数据分组	监视开关
阈值违例	event_thresholdviolations	-

用法

使用此元素来跟踪活动在它重新映射到的各个服务类之间的路径。此元素还可用于计算映射到给定服务子类的活动数的聚集值。

device_type -“设备类型”

此元素是与 UTILSTART 事件相关联的设备类型的标识。

表 574. 事件监视信息

事件类型	逻辑数据分组	监视开关
变更历史记录	UTILLOCATION	始终收集

用法

对于变更历史记录事件监视器，此字段指示 LOCATION 字段的解释:

A TSM

C 客户机

D 磁盘

F 快照备份

L 本地

N 由 DB2 内部生成

O 其他供应商设备支持

P 管道

Q 游标

R 除去访存数据

S 服务器

T 磁带

U 用户出口

X X/开放式 XBSA 接口

diaglog_write_wait_time -“诊断日志文件写等待时间”监视元素

等待写 db2diag 日志文件时耗用的时间。此值以毫秒计。

表 575. 表函数监视信息

表函数	监视元素收集级别
MON_FORMAT_XML_METRICS_BY_ROW - 获取所有度量值的基于已格式化行的输出	不适用; 报告 XML 文档中作为格式化函数的输入提供的所有元素
MON_FORMAT_XML_TIMES_BY_ROW - - 获取基于已格式化行的组合层次结构等待时间和处理时间	不适用; 报告 XML 文档中作为格式化函数的输入提供的所有元素

表 575. 表函数监视信息 (续)

表函数	监视元素收集级别
MON_FORMAT_XML_WAIT_TIMES_BY _ROW - 获取等待时间的已格式化的基于行的输出	不适用; 报告 XML 文档中作为格式化函数的输入提供的所有元素
MON_GET_ACTIVITY_DETAILS 表函数 - 获取完整的活动详细信息（在 XML 文档 DETAILS 中报告）	ACTIVITY METRICS BASE
MON_GET_CONNECTION 表函数 - 获取连接度量值	REQUEST METRICS BASE
MON_GET_CONNECTION_DETAILS 表函数 - 获取详细的连接度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE
MON_GET_PKG_CACHE_STMT 表函数 - 获取程序包高速缓存中的 SQL 语句活动度量值	ACTIVITY METRICS BASE
MON_GET_PKG_CACHE_STMT_DETAILS 表函数 - 获取程序包高速缓存条目的详细度量值	ACTIVITY METRICS BASE
MON_GET_SERVICE_SUBCLASS 表函数 - 获取服务子类度量值	REQUEST METRICS BASE
MON_GET_SERVICE_SUBCLASS_DETAILS 表函数 - 获取详细的服务子类度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE
MON_GET_UNIT_OF_WORK 表函数 - 获取工作单元度量值	REQUEST METRICS BASE
MON_GET_UNIT_OF_WORK 表函数 - 获取详细的工作单元度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE
MON_GET_WORKLOAD 表函数 - 获取工作负载度量值	REQUEST METRICS BASE
MON_GET_WORKLOAD_DETAILS 表函数 - 获取详细的工作负载度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE

表 576. 事件监视信息

事件类型	逻辑数据分组	监视开关
活动	event_activity（在 details_xml 文档中报告）	ACTIVITY METRICS BASE
活动	event_activitymetrics	ACTIVITY METRICS BASE
统计信息	event_scstats（在 details_xml 文档中报告）	REQUEST METRICS BASE
统计信息	event_wlstats（在 details_xml 文档中报告）	REQUEST METRICS BASE
工作单元	在 system_metrics 文档中报告。	REQUEST METRICS BASE
程序包高速缓存	在 activity_metrics 文档中报告。	ACTIVITY METRICS BASE

用法

通过使用此元素，可以确定写 db2diag 日志文件时的耗用时间。在分区数据库环境中，如果此时间过长，并且正在将共享存储器用于诊断目录路径（diagpath），那么可能表明 db2diag 日志文件被争用。较大的值也可能表明日志记录量过大，例如，已将 **diaglevel** 设置为记录所有参考消息。

diaglog_writes_total -“写诊断日志文件总次数”监视元素

代理程序写 db2diag 日志文件的次数。

表 577. 表函数监视信息

表函数	监视元素收集级别
MON_FORMAT_XML_METRICS_BY_ROW - 获取所有度量值的基于已格式化行的输出	不适用; 报告 XML 文档中作为格式化函数的输入提供的所有元素
MON_GET_ACTIVITY_DETAILS 表函数 - 获取完整的活动详细信息（在 XML 文档 DETAILS 中报告）	ACTIVITY METRICS BASE
MON_GET_CONNECTION 表函数 - 获取连接度量值	REQUEST METRICS BASE
MON_GET_CONNECTION_DETAILS 表函数 - 获取详细的连接度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE
MON_GET_PKG_CACHE_STMT 表函数 - 获取程序包高速缓存中的 SQL 语句活动度量值	ACTIVITY METRICS BASE
MON_GET_PKG_CACHE_STMT_DETAILS 表函数 - 获取程序包高速缓存条目的详细度量值	ACTIVITY METRICS BASE
MON_GET_SERVICE_SUBCLASS 表函数 - 获取服务子类度量值	REQUEST METRICS BASE
MON_GET_SERVICE_SUBCLASS_DETAILS 表函数 - 获取详细的服务子类度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE
MON_GET_UNIT_OF_WORK 表函数 - 获取工作单元度量值	REQUEST METRICS BASE
MON_GET_UNIT_OF_WORK 表函数 - 获取详细的工作单元度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE
MON_GET_WORKLOAD 表函数 - 获取工作负载度量值	REQUEST METRICS BASE
MON_GET_WORKLOAD_DETAILS 表函数 - 获取详细的工作负载度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE

表 578. 事件监视信息

事件类型	逻辑数据分组	监视开关
活动	event_activity（在 details_xml 文档中报告）	ACTIVITY METRICS BASE

表 578. 事件监视信息 (续)

事件类型	逻辑数据分组	监视开关
活动	event_activitymetrics	ACTIVITY METRICS BASE
统计信息	event_scstats（在 details_xml 文档中报告）	REQUEST METRICS BASE
统计信息	event_wlstats（在 details_xml 文档中报告）	REQUEST METRICS BASE
工作单元	在 system_metrics 文档中报告。	REQUEST METRICS BASE
程序包高速缓存	在 activity_metrics 文档中报告。	ACTIVITY METRICS BASE

用法

通过将此元素与 **diaglog_write_wait_time** 监视元素配合使用，可以了解写入到 db2diag 日志文件时平均耗用的时间。

direct_read_reqs -“直接读请求数”监视元素

对一个或多个扇区的数据执行直接读取的请求数。

表 579. 表函数监视信息

表函数	监视元素收集级别
MON_FORMAT_XML_METRICS_BY_ROW - 获取所有度量值的基于已格式化行的输出	不适用；报告 XML 文档中作为格式化函数的输入提供的所有元素
MON_GET_ACTIVITY_DETAILS 表函数 - 获取完整的活动详细信息（在 XML 文档 DETAILS 中报告）	ACTIVITY METRICS BASE
MON_GET_BUFFERPOOL 表函数 - 获取缓冲池度量值	DATA OBJECT METRICS BASE
MON_GET_CONNECTION 表函数 - 获取连接度量值	REQUEST METRICS BASE
MON_GET_CONNECTION_DETAILS 表函数 - 获取详细的连接度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE
MON_GET_PKG_CACHE_STMT 表函数 - 获取程序包高速缓存中的 SQL 语句活动度量值	ACTIVITY METRICS BASE
MON_GET_PKG_CACHE_STMT_DETAILS 表函数 - 获取程序包高速缓存条目的详细度量值	ACTIVITY METRICS BASE
MON_GET_SERVICE_SUBCLASS 表函数 - 获取服务子类度量值	REQUEST METRICS BASE
MON_GET_SERVICE_SUBCLASS_DETAILS 表函数 - 获取详细的服务子类度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE
MON_GET_TABLE 表函数 - 获取表度量值	DATA OBJECT METRICS EXTENDED
MON_GET_TABLE_USAGE_LIST 表函数 - 从表用法列表返回信息	DATA OBJECT METRICS EXTENDED

表 579. 表函数监视信息 (续)

表函数	监视元素收集级别
MON_GET_TABLESPACE 表函数 - 获取表空间度量值	DATA OBJECT METRICS BASE
MON_GET_UNIT_OF_WORK 表函数 - 获取工作单元度量值	REQUEST METRICS BASE
MON_GET_UNIT_OF_WORK 表函数 - 获取详细的工作单元度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE
MON_GET_WORKLOAD 表函数 - 获取工作负载度量值	REQUEST METRICS BASE
MON_GET_WORKLOAD_DETAILS 表函数 - 获取详细的工作负载度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE

表 580. 快照监视信息

快照级别	逻辑数据分组	监视开关
数据库	dbase	缓冲池
表空间	tablespace	缓冲池
缓冲池	bufferpool	缓冲池
应用程序	appl	缓冲池

可将快照监视的计数器重置。

表 581. 事件监视信息

事件类型	逻辑数据分组	监视开关
活动	event_activity（在 details_xml 文档中报告）	ACTIVITY METRICS BASE
活动	event_activitymetrics	ACTIVITY METRICS BASE
统计信息	event_scstats（在 details_xml 文档中报告）	REQUEST METRICS BASE
统计信息	event_wlstats（在 details_xml 文档中报告）	REQUEST METRICS BASE
工作单元	在 system_metrics 文档中报告。	REQUEST METRICS BASE
数据库	event_db	始终收集
连接	event_conn	始终收集
表空间	event_tablespace	始终收集
程序包高速缓存	在 activity_metrics 文档中报告。	ACTIVITY METRICS BASE

用法

使用以下公式来计算直接读操作读取的平均扇区数:

```
direct_reads / direct_read_reqs
```

direct_read_time -“直接读时间”监视元素

执行直接读操作时耗用的时间。此值以毫秒计。

表 582. 表函数监视信息

表函数	监视元素收集级别
MON_FORMAT_XML_METRICS_BY_ROW - 获取所有度量值的基于已格式化行的输出	不适用; 报告 XML 文档中作为格式化函数的输入提供的所有元素
MON_FORMAT_XML_TIMES_BY_ROW - - 获取基于已格式化行的组合层次结构等待时间和处理时间	不适用; 报告 XML 文档中作为格式化函数的输入提供的所有元素
MON_FORMAT_XML_WAIT_TIMES_BY _ROW - 获取等待时间的已格式化的基于行的输出	不适用; 报告 XML 文档中作为格式化函数的输入提供的所有元素
MON_GET_ACTIVITY_DETAILS 表函数 - 获取完整的活动详细信息（在 XML 文档 DETAILS 中报告）	ACTIVITY METRICS BASE
MON_GET_BUFFERPOOL 表函数 - 获取缓冲池度量值	DATA OBJECT METRICS BASE
MON_GET_CONNECTION 表函数 - 获取连接度量值	REQUEST METRICS BASE
MON_GET_CONNECTION_DETAILS 表函数 - 获取详细的连接度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE
MON_GET_CONTAINER 表函数 - 获取表空间容器度量值	DATA OBJECT METRICS BASE
MON_GET_PKG_CACHE_STMT 表函数 - 获取程序包高速缓存中的 SQL 语句活动度量值	ACTIVITY METRICS BASE
MON_GET_PKG_CACHE_STMT_DETAILS 表函数 - 获取程序包高速缓存条目的详细度量值	ACTIVITY METRICS BASE
MON_GET_SERVICE_SUBCLASS 表函数 - 获取服务子类度量值	REQUEST METRICS BASE
MON_GET_SERVICE_SUBCLASS_DETAILS 表函数 - 获取详细的服务子类度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE
MON_GET_TABLESPACE 表函数 - 获取表空间度量值	DATA OBJECT METRICS BASE
MON_GET_UNIT_OF_WORK 表函数 - 获取工作单元度量值	REQUEST METRICS BASE
MON_GET_UNIT_OF_WORK 表函数 - 获取详细的工作单元度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE
MON_GET_WORKLOAD 表函数 - 获取工作负载度量值	REQUEST METRICS BASE
MON_GET_WORKLOAD_DETAILS 表函数 - 获取详细的工作负载度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE

表 583. 快照监视信息

快照级别	逻辑数据分组	监视开关
数据库	dbase	缓冲池
表空间	tablespace	缓冲池
缓冲池	bufferpool	缓冲池
应用程序	appl	缓冲池

可将快照监视的计数器重置。

表 584. 事件监视信息

事件类型	逻辑数据分组	监视开关
活动	event_activity（在 details_xml 文档中报告）	ACTIVITY METRICS BASE
活动	event_activitymetrics	ACTIVITY METRICS BASE
统计信息	event_scstats（在 details_xml 文档中报告）	REQUEST METRICS BASE
统计信息	event_wlstats（在 details_xml 文档中报告）	REQUEST METRICS BASE
工作单元	在 system_metrics 文档中报告。	REQUEST METRICS BASE
数据库	event_db	始终收集
连接	event_conn	始终收集
表空间	event_tablespace	始终收集
程序包高速缓存	在 activity_metrics 文档中报告。	ACTIVITY METRICS BASE

用法

使用以下公式来计算每扇区平均直接读取时间:

```
direct_read_time / direct_reads
```

如果平均时间很长，那么指示可能存在 I/O 冲突。

direct_reads -“直接读数据库数目”监视元素

未使用缓冲池的读操作的数目。

表 585. 表函数监视信息

表函数	监视元素收集级别
MON_FORMAT_XML_METRICS_BY_ROW - 获取所有度量值的基于已格式化行的输出	不适用; 报告 XML 文档中作为格式化函数的输入提供的所有元素
MON_GET_ACTIVITY_DETAILS 表函数 - 获取完整的活动详细信息（在 XML 文档 DETAILS 中报告）	ACTIVITY METRICS BASE
MON_GET_BUFFERPOOL 表函数 - 获取缓冲池度量值	DATA OBJECT METRICS BASE

表 585. 表函数监视信息 (续)

表函数	监视元素收集级别
MON_GET_CONNECTION 表函数 - 获取连接度量值	REQUEST METRICS BASE
MON_GET_CONNECTION_DETAILS 表函数 - 获取详细的连接度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE
MON_GET_CONTAINER 表函数 - 获取表空间容器度量值	DATA OBJECT METRICS BASE
MON_GET_PKG_CACHE_STMT 表函数 - 获取程序包高速缓存中的 SQL 语句活动度量值	ACTIVITY METRICS BASE
MON_GET_PKG_CACHE_STMT_DETAILS 表函数 - 获取程序包高速缓存条目的详细度量值	ACTIVITY METRICS BASE
MON_GET_SERVICE_SUBCLASS 表函数 - 获取服务子类度量值	REQUEST METRICS BASE
MON_GET_SERVICE_SUBCLASS_DETAILS 表函数 - 获取详细的服务子类度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE
MON_GET_TABLE 表函数 - 获取表度量值	DATA OBJECT METRICS EXTENDED
MON_GET_TABLE_USAGE_LIST 表函数 - 从表用法列表返回信息	DATA OBJECT METRICS EXTENDED
MON_GET_TABLESPACE 表函数 - 获取表空间度量值	DATA OBJECT METRICS BASE
MON_GET_UNIT_OF_WORK 表函数 - 获取工作单元度量值	REQUEST METRICS BASE
MON_GET_UNIT_OF_WORK 表函数 - 获取详细的工作单元度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE
MON_GET_WORKLOAD 表函数 - 获取工作负载度量值	REQUEST METRICS BASE
MON_GET_WORKLOAD_DETAILS 表函数 - 获取详细的工作负载度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE
WLM_GET_WORKLOAD_OCCURRENCE_ACTIVITIES 表函数 - 返回活动列表	REQUEST METRICS BASE

表 586. 快照监视信息

快照级别	逻辑数据分组	监视开关
数据库	dbase	缓冲池
表空间	tablespace	缓冲池
缓冲池	bufferpool	缓冲池
应用程序	appl	缓冲池

可将快照监视的计数器重置。

表 587. 事件监视信息

事件类型	逻辑数据分组	监视开关
活动	event_activity（在 details_xml 文档中报告）	ACTIVITY METRICS BASE
活动	event_activitymetrics	ACTIVITY METRICS BASE
统计信息	event_scstats（在 details_xml 文档中报告）	REQUEST METRICS BASE
统计信息	event_wlstats（在 details_xml 文档中报告）	REQUEST METRICS BASE
工作单元	在 system_metrics 文档中报告。	REQUEST METRICS BASE
数据库	event_db	始终收集
连接	event_conn	始终收集
表空间	event_tablespace	始终收集
程序包高速缓存	在 activity_metrics 文档中报告。	ACTIVITY METRICS BASE

用法

使用以下公式来计算直接读操作读取的平均扇区数:

```
direct_reads / direct_read_reqs
```

使用系统监视器跟踪 I/O 时，此元素可帮助您区分设备上的数据库 I/O 与非数据库 I/O。

直接读操作是以单元为单位执行的，最小单元为 512 字节扇区。在下列情况下将使用它们:

- 读取 LONG VARCHAR 列
- 读取 LOB（大对象）列
- 执行备份

direct_write_reqs -“直接写请求数”监视元素

对一个或多个扇区的数据执行直接写入的请求数。

表 588. 表函数监视信息

表函数	监视元素收集级别
MON_FORMAT_XML_METRICS_BY_ROW - 获取所有度量值的基于已格式化行的输出	不适用; 报告 XML 文档中作为格式化函数的输入提供的所有元素
MON_GET_ACTIVITY_DETAILS 表函数 - 获取完整的活动详细信息（在 XML 文档 DETAILS 中报告）	ACTIVITY METRICS BASE
MON_GET_BUFFERPOOL 表函数 - 获取缓冲池度量值	DATA OBJECT METRICS BASE
MON_GET_CONNECTION 表函数 - 获取连接度量值	REQUEST METRICS BASE

表 588. 表函数监视信息 (续)

表函数	监视元素收集级别
MON_GET_CONNECTION_DETAILS 表函数 - 获取详细的连接度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE
MON_GET_PKG_CACHE_STMT 表函数 - 获取程序包高速缓存中的 SQL 语句活动度量值	ACTIVITY METRICS BASE
MON_GET_PKG_CACHE_STMT_DETAILS 表函数 - 获取程序包高速缓存条目的详细度量值	ACTIVITY METRICS BASE
MON_GET_SERVICE_SUBCLASS 表函数 - 获取服务子类度量值	REQUEST METRICS BASE
MON_GET_SERVICE_SUBCLASS_DETAILS 表函数 - 获取详细的服务子类度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE
MON_GET_TABLE 表函数 - 获取表度量值	DATA OBJECT METRICS EXTENDED
MON_GET_TABLE_USAGE_LIST 表函数 - 从表用法列表返回信息	DATA OBJECT METRICS EXTENDED
MON_GET_TABLESPACE 表函数 - 获取表空间度量值	DATA OBJECT METRICS BASE
MON_GET_UNIT_OF_WORK 表函数 - 获取工作单元度量值	REQUEST METRICS BASE
MON_GET_UNIT_OF_WORK 表函数 - 获取详细的工作单元度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE
MON_GET_WORKLOAD 表函数 - 获取工作负载度量值	REQUEST METRICS BASE
MON_GET_WORKLOAD_DETAILS 表函数 - 获取详细的工作负载度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE

表 589. 快照监视信息

快照级别	逻辑数据分组	监视开关
数据库	dbase	缓冲池
表空间	tablespace	缓冲池
缓冲池	bufferpool	缓冲池
应用程序	appl	缓冲池

可将快照监视的计数器重置。

表 590. 事件监视信息

事件类型	逻辑数据分组	监视开关
活动	event_activity（在 details_xml 文档中报告）	ACTIVITY METRICS BASE
活动	event_activitymetrics	ACTIVITY METRICS BASE

表 590. 事件监视信息 (续)

事件类型	逻辑数据分组	监视开关
统计信息	event_scstats（在 details_xml 文档中报告）	REQUEST METRICS BASE
统计信息	event_wlstats（在 details_xml 文档中报告）	REQUEST METRICS BASE
工作单元	在 system_metrics 文档中报告。	REQUEST METRICS BASE
数据库	event_db	始终收集
连接	event_conn	始终收集
表空间	event_tablespace	始终收集
程序包高速缓存	在 activity_metrics 文档中报告。	ACTIVITY METRICS BASE

用法

使用以下公式来计算直接写入操作写入的平均扇区数:

```
direct_writes / direct_write_reqs
```

direct_write_time -“直接写时间”监视元素

执行直接写操作时耗用的时间。此值以毫秒计。

表 591. 表函数监视信息

表函数	监视元素收集级别
MON_FORMAT_XML_METRICS_BY_ROW - 获取所有度量值的基于已格式化行的输出	不适用; 报告 XML 文档中作为格式化函数的输入提供的所有元素
MON_FORMAT_XML_TIMES_BY_ROW - - 获取基于已格式化行的组合层次结构等待时间和处理时间	不适用; 报告 XML 文档中作为格式化函数的输入提供的所有元素
MON_FORMAT_XML_WAIT_TIMES_BY _ROW - 获取等待时间的已格式化的基于行的输出	不适用; 报告 XML 文档中作为格式化函数的输入提供的所有元素
MON_GET_ACTIVITY_DETAILS 表函数 - 获取完整的活动详细信息（在 XML 文档 DETAILS 中报告）	ACTIVITY METRICS BASE
MON_GET_BUFFERPOOL 表函数 - 获取缓冲池度量值	DATA OBJECT METRICS BASE
MON_GET_CONNECTION 表函数 - 获取连接度量值	REQUEST METRICS BASE
MON_GET_CONNECTION_DETAILS 表函数 - 获取详细的连接度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE
MON_GET_CONTAINER 表函数 - 获取表空间容器度量值	DATA OBJECT METRICS BASE
MON_GET_PKG_CACHE_STMT 表函数 - 获取程序包高速缓存中的 SQL 语句活动度量值	ACTIVITY METRICS BASE

表 591. 表函数监视信息 (续)

表函数	监视元素收集级别
MON_GET_PKG_CACHE_STMT_DETAILS 表函数 - 获取程序包高速缓存条目的详细度量值	ACTIVITY METRICS BASE
MON_GET_SERVICE_SUBCLASS 表函数 - 获取服务子类度量值	REQUEST METRICS BASE
MON_GET_SERVICE_SUBCLASS_DETAILS 表函数 - 获取详细的服务子类度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE
MON_GET_TABLESPACE 表函数 - 获取表空间度量值	DATA OBJECT METRICS BASE
MON_GET_UNIT_OF_WORK 表函数 - 获取工作单元度量值	REQUEST METRICS BASE
MON_GET_UNIT_OF_WORK 表函数 - 获取详细的工作单元度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE
MON_GET_WORKLOAD 表函数 - 获取工作负载度量值	REQUEST METRICS BASE
MON_GET_WORKLOAD_DETAILS 表函数 - 获取详细的工作负载度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE

表 592. 快照监视信息

快照级别	逻辑数据分组	监视开关
数据库	dbase	缓冲池
表空间	tablespace	缓冲池
缓冲池	bufferpool	缓冲池
应用程序	appl	缓冲池

可将快照监视的计数器重置。

表 593. 事件监视信息

事件类型	逻辑数据分组	监视开关
活动	event_activity（在 details_xml 文档中报告）	ACTIVITY METRICS BASE
活动	event_activitymetrics	ACTIVITY METRICS BASE
统计信息	event_scstats（在 details_xml 文档中报告）	REQUEST METRICS BASE
统计信息	event_wlstats（在 details_xml 文档中报告）	REQUEST METRICS BASE
工作单元	在 system_metrics 文档中报告。	REQUEST METRICS BASE
数据库	event_db	始终收集
连接	event_conn	始终收集
表空间	event_tablespace	始终收集

表 593. 事件监视信息 (续)

事件类型	逻辑数据分组	监视开关
程序包高速缓存	在 activity_metrics 文档中报告。	ACTIVITY METRICS BASE

用法

使用以下公式来计算每扇区平均直接写入时间:

```
direct_write_time / direct_writes
```

如果平均时间很长，那么指示可能存在 I/O 冲突。

direct_writes -“直接写数据库数目”监视元素

未使用缓冲池的写操作的数目。

表 594. 表函数监视信息

表函数	监视元素收集级别
MON_FORMAT_XML_METRICS_BY_ROW - 获取所有度量值的基于已格式化行的输出	不适用; 报告 XML 文档中作为格式化函数的输入提供的所有元素
MON_GET_ACTIVITY_DETAILS 表函数 - 获取完整的活动详细信息（在 XML 文档 DETAILS 中报告）	ACTIVITY METRICS BASE
MON_GET_BUFFERPOOL 表函数 - 获取缓冲池度量值	DATA OBJECT METRICS BASE
MON_GET_CONNECTION 表函数 - 获取连接度量值	REQUEST METRICS BASE
MON_GET_CONNECTION_DETAILS 表函数 - 获取详细的连接度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE
MON_GET_CONTAINER 表函数 - 获取表空间容器度量值	DATA OBJECT METRICS BASE
MON_GET_PKG_CACHE_STMT 表函数 - 获取程序包高速缓存中的 SQL 语句活动度量值	ACTIVITY METRICS BASE
MON_GET_PKG_CACHE_STMT_DETAILS 表函数 - 获取程序包高速缓存条目的详细度量值	ACTIVITY METRICS BASE
MON_GET_SERVICE_SUBCLASS 表函数 - 获取服务子类度量值	REQUEST METRICS BASE
MON_GET_SERVICE_SUBCLASS_DETAILS 表函数 - 获取详细的服务子类度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE
MON_GET_TABLE 表函数 - 获取表度量值	DATA OBJECT METRICS EXTENDED
MON_GET_TABLE_USAGE_LIST 表函数 - 从表用法列表返回信息	DATA OBJECT METRICS EXTENDED
MON_GET_TABLESPACE 表函数 - 获取表空间度量值	DATA OBJECT METRICS BASE

表 594. 表函数监视信息 (续)

表函数	监视元素收集级别
MON_GET_UNIT_OF_WORK 表函数 - 获取工作单元度量值	REQUEST METRICS BASE
MON_GET_UNIT_OF_WORK 表函数 - 获取详细的工作单元度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE
MON_GET_WORKLOAD 表函数 - 获取工作负载度量值	REQUEST METRICS BASE
MON_GET_WORKLOAD_DETAILS 表函数 - 获取详细的工作负载度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE
WLM_GET_WORKLOAD_OCCURRENCE_ACTIVITIES 表函数 - 返回活动列表	REQUEST METRICS BASE

表 595. 快照监视信息

快照级别	逻辑数据分组	监视开关
数据库	dbase	缓冲池
表空间	tablespace	缓冲池
缓冲池	bufferpool	缓冲池
应用程序	appl	缓冲池

可将快照监视的计数器重置。

表 596. 事件监视信息

事件类型	逻辑数据分组	监视开关
活动	event_activity（在 details_xml 文档中报告）	ACTIVITY METRICS BASE
活动	event_activitymetrics	ACTIVITY METRICS BASE
统计信息	event_scstats（在 details_xml 文档中报告）	REQUEST METRICS BASE
统计信息	event_wlstats（在 details_xml 文档中报告）	REQUEST METRICS BASE
工作单元	在 system_metrics 文档中报告。	REQUEST METRICS BASE
数据库	event_db	始终收集
连接	event_conn	始终收集
表空间	event_tablespace	始终收集
程序包高速缓存	在 activity_metrics 文档中报告。	ACTIVITY METRICS BASE

用法

使用以下公式来计算直接写入操作写入的平均扇区数:

```
direct_writes / direct_write_reqs
```

使用系统监视器跟踪 I/O 时，此元素可帮助您区分设备上的数据库 I/O 与非数据库 I/O。

直接写入操作是以单元为单位执行的，最小单元为 512 字节扇区。在下列情况下将使用它们:

- 写入 LONG VARCHAR 列
- 写入 LOB（大对象）列
- 执行复原
- 执行装入
- 如果启用了 MPFA（这是缺省情况），那么为 SMS 表空间分配新的扩展数据块

disabled_peds -“已禁用部分提前相异数”监视元素

因为可用排序堆不足而导致部分提前相异操作被禁用的次数。

表 597. 表函数监视信息

表函数	监视元素收集级别
MON_FORMAT_XML_METRICS_BY_ROW - 获取所有度量值的基于已格式化行的输出	不适用; 报告 XML 文档中作为格式化函数的输入提供的所有元素
MON_GET_ACTIVITY_DETAILS 表函数 - 获取完整的活动详细信息（在 XML 文档 DETAILS 中报告）	ACTIVITY METRICS BASE
MON_GET_CONNECTION 表函数 - 获取连接度量值	REQUEST METRICS BASE
MON_GET_CONNECTION_DETAILS 表函数 - 获取详细的连接度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE
MON_GET_PKG_CACHE_STMT 表函数 - 获取程序包高速缓存中的 SQL 语句活动度量值	ACTIVITY METRICS BASE
MON_GET_PKG_CACHE_STMT_DETAILS 表函数 - 获取程序包高速缓存条目的详细度量值（在 XML 文档 DETAILS 中报告）	ACTIVITY METRICS BASE
MON_GET_SERVICE_SUBCLASS 表函数 - 获取服务子类度量值	REQUEST METRICS BASE
MON_GET_SERVICE_SUBCLASS_DETAILS 表函数 - 获取详细的服务子类度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE
MON_GET_UNIT_OF_WORK 表函数 - 获取工作单元度量值	REQUEST METRICS BASE
MON_GET_UNIT_OF_WORK 表函数 - 获取详细的工作单元度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE
MON_GET_WORKLOAD 表函数 - 获取工作负载度量值	REQUEST METRICS BASE
MON_GET_WORKLOAD_DETAILS 表函数 - 获取详细的工作负载度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE

表 598. 事件监视信息

事件类型	逻辑数据分组	监视开关
活动	event_activity（在 details_xml 文档中报告）	ACTIVITY METRICS BASE
统计信息	event_scstats（在 details_xml 文档中报告）	REQUEST METRICS BASE
统计信息	event_wlstats（在 details_xml 文档中报告）	REQUEST METRICS BASE
工作单元	在 system_metrics 文档中报告。	REQUEST METRICS BASE
连接	event_conn	-
语句	event_stmt	-
事务	event_xact	-
程序包高速缓存	在 activity_metrics 文档中报告。	ACTIVITY METRICS BASE

用法

将此元素与 **total_peds** 监视元素配合使用来确定大多数时间部分提前相异操作是否获取足够的排序堆内存。如果 **disabled_peds** 监视元素与 **total_peds** 监视元素的比率很高，那么数据库性能可能欠佳。应考虑增加排序堆大小和/或排序堆阈值。

disconn_time -“数据库释放时间戳记”

应用程序与数据库在数据库级别断开连接的日期和时间，这是应用程序最后一次断开连接。

元素标识

disconn_time

元素类型

时间戳记

表 599. 事件监视信息

事件类型	逻辑数据分组	监视开关
数据库	event_db	始终收集
连接	event_conn	始终收集

用法 使用此元素来计算自出现以下情况以来耗用的时间:

- 数据库处于活动状态（用于数据库级别的信息）
- 连接处于活动状态（用于连接级别的信息）。

disconnects -“断开连接次数”

此元素包含自联合服务器实例启动或数据库监视计数器最后一次重置以后（取较晚者），联合服务器代表任何应用程序与此数据源断开连接的总次数。

表 600. 快照监视信息

快照级别	逻辑数据分组	监视开关
数据库	dbase_remote	基本

可将快照监视的计数器重置。

用法

使用此元素来确定联合服务器代表任何应用程序与此数据源断开连接的总次数。此元素与 CONNECT 计数一起使用可提供一种机制，您可以通过这种机制来确定此联合服务器实例相信且当前已连接至数据源的应用程序数。

dl_conns -“死锁中涉及的连接数”监视元素

死锁中涉及的连接数。

表 601. 事件监视信息

事件类型	逻辑数据分组	监视开关
锁定	-	始终收集
死锁 [1]	event_deadlock	始终收集

1 建议不要使用此事件监视器。建议不要再使用此选项，将来的发行版可能会将其除去。请使用 CREATE EVENT MONITOR FOR LOCKING 语句来监视与锁定相关的事件，例如锁定超时、锁定等待和死锁。

用法

在监视应用程序中使用此元素以标识事件监视器数据流中将会出现的死锁连接事件记录数。

dynamic_sql_stmts -“尝试的动态 SQL 语句数”

尝试的动态 SQL 语句数。

表 602. 快照监视信息

快照级别	逻辑数据分组	监视开关
数据库	dbase	基本
应用程序	appl	基本

可将快照监视的计数器重置。

表 603. 事件监视信息

事件类型	逻辑数据分组	监视开关
数据库	event_db	始终收集

表 603. 事件监视信息 (续)

事件类型	逻辑数据分组	监视开关
连接	event_conn	始终收集

用法 可使用此元素来计算在数据库或应用程序级别成功执行的 SQL 语句总数:

```
    dynamic_sql_stmts
+ static_sql_stmts
- failed_sql_stmts
= 监视期间的吞吐量
```

edu_ID -“引擎可分派单元标识”监视元素

与此内存池相关联的引擎可分派单元的标识。

表 604. 表函数监视信息

表函数	监视元素收集级别
MON_GET_MEMORY_POOL 表函数 - 获取内存池信息	ACTIVITY METRICS BASE
PD_GET_DIAG_HIST 表函数 - 从给定设施返回记录	ACTIVITY METRICS BASE

用法

除了所描述的内存池是 PRIVATE 时之外，当表函数 MON_GET_MEMORY_POOL 返回此监视元素时，此监视元素为 NULL。

eff_stmt_text -“有效语句文本”监视元素

如果 SQL 语句作为语句集中器的结果而被修改，那么此元素包含该语句的有效文本。

表 605. 表函数监视信息

表函数	监视元素收集级别
WLM_GET_ACTIVITY_DETAILS_COMPLETE 表函数 - 获取完整的活动详细信息（在 XML 文档 DETAILS 中报告）	ACTIVITY METRICS BASE

表 606. 事件监视信息

事件类型	逻辑数据分组	监视开关
活动	event_activitystmt	始终收集

用法

如果已启用语句集中器，并且语句文本已作为语句集中器的结果而被修改，那么此监视元素包含有效的语句文本。否则，此监视元素包含长度为 0 字节的文本字符串。

effective_isolation -“有效隔离级别”监视元素

此语句的有效隔离级别。

表 607. 表函数监视信息

表函数	监视元素收集级别
MON_GET_ACTIVITY_DETAILS 表函数 - 获取完整的活动详细信息（在 XML 文档 DETAILS 中报告）	ACTIVITY METRICS BASE
MON_GET_PKG_CACHE_STMT 表函数 - 获取程序包高速缓存中的 SQL 语句活动度量值	ACTIVITY METRICS BASE
MON_GET_PKG_CACHE_STMT_DETAILS 表函数 - 获取程序包高速缓存条目的详细度量值	ACTIVITY METRICS BASE

表 608. 事件监视信息

事件类型	逻辑数据分组	监视开关
锁定	-	-
程序包高速缓存	-	COLLECT BASE DATA

用法

通过使用此元素，可以确定执行语句期间使用的隔离级别。

effective_lock_timeout -“有效锁定超时”监视元素

此活动的有效锁定超时值。此值以秒计。

表 609. 表函数监视信息

表函数	监视元素收集级别
MON_GET_ACTIVITY_DETAILS 表函数 - 获取完整的活动详细信息（在 XML 文档 DETAILS 中报告）	ACTIVITY METRICS BASE

用法

effective_query_degree -“有效查询并行度”监视元素

此活动的有效查询并行度。

表 610. 表函数监视信息

表函数	监视元素收集级别
MON_GET_ACTIVITY_DETAILS 表函数 - 获取完整的活动详细信息（在 XML 文档 DETAILS 中报告）	ACTIVITY METRICS BASE

表 611. 事件监视信息

事件类型	逻辑数据分组	监视开关
锁定	-	-

elapsed_exec_time -“语句执行耗用时间”

在 DCS 语句级别，这是在主机数据库服务器上处理 SQL 请求所耗用的时间。此值由此服务器报告。对于写至表的事件监视器，此元素的值将通过使用 BIGINT 数据类型以微秒为单位给定。和 host_response_time 元素相比，此元素不包括 DB2 Connect 与主机数据库服务器之间的网络耗用时间。在其他级别，此值表示对特定数据库或应用程序执行的所有语句的主机执行时间的总和，或者是使用给定数目的数据传输的那些语句的主机执行时间的总和。

表 612. 快照监视信息

快照级别	逻辑数据分组	监视开关
数据库	dbase	语句，时间戳记
应用程序	appl	语句，时间戳记
DCS 数据库	dcs_dbase	语句，时间戳记
DCS 应用程序	dcs_appl	语句，时间戳记
DCS 语句	dcs_stmt	语句，时间戳记
数据传输	stmt_transmissions	语句，时间戳记

对于语句级别的快照监视，不能重置此计数器。可在其他级别重置此计数器。

用法 将此元素与其他耗用时间监视元素一起使用以评估数据库服务器的 SQL 请求处理情况和帮助隔离性能问题。

从 host_response_time 元素减去此元素，以计算 DB2 Connect 与主机数据库服务器之间的网络耗用时间。

注: 对于 dcs_dbase、dcs_appl、dcs_stmt 和 stmt_transmissions 级别，*elapsed_exec_time 元素*仅适用于 z/OS® 数据库。如果 DB2 Connect 网关连接至 Windows、Linux、AIX 或其他 UNIX 数据库，那么 *elapsed_exec_time* 报告为零。

empty_pages_deleted -“删除的空页数”监视元素

已删除的伪空页数。伪空页就是已将所有键伪删除的页。

表 613. 表函数监视信息

表函数	监视元素收集级别
MON_GET_INDEX 表函数 - 获取索引度量值	ACTIVITY METRICS BASE

empty_pages_reused -“复用的空页数”监视元素

已复用的伪空页数。伪空页就是已将所有键伪删除的页。

表 614. 表函数监视信息

表函数	监视元素收集级别
MON_GET_INDEX 表函数 - 获取索引度量值	ACTIVITY METRICS BASE

entry_time -“进入时间”监视元素

此活动进入系统时的时间。

表 615. 表函数监视信息

表函数	监视元素收集级别
MON_GET_ACTIVITY_DETAILS 表函数 - 获取完整的活动详细信息（在 XML 文档 DETAILS 中报告）	ACTIVITY METRICS BASE
WLM_GET_WORKLOAD_OCCURRENCE_ACTIVITIES 表函数 - 返回活动列表	ACTIVITY METRICS BASE

用法

estimated_cpu_entitlement -“估算的 CPU 使用量”监视元素

主机或 LPAR 上根据 CPU 份额配置给服务子类使用的总 CPU 百分比（假定它使用的 CPU 份额正好为配置给它使用的份额）。确定哪些服务类参与计算取决于抽样时间段度量的实际 CPU 利用率和 WLM_DISP_MIN_UTIL 数据库管理器配置设置。计算时不考虑 CPU 限额对服务类本身、与它竞争的服务类或父服务类（如果它有父服务类）的影响。

表 616. 表函数监视信息

表函数	监视元素收集级别
MON_SAMPLE_SERVICE_CLASS_METRICS - 获取样本服务类度量值	ACTIVITY METRICS BASE

estimatedsqlcost_threshold_id -“估计 SQL 成本阈值标识”监视元素

应用于此活动的 ESTIMATEDSQLCOST 阈值的标识。

表 617. 表函数监视信息

表函数	监视元素收集级别
MON_GET_ACTIVITY_DETAILS 表函数 - 获取完整的活动详细信息（在 XML 文档 DETAILS 中报告）	ACTIVITY METRICS BASE

用法

使用此元素来确定应用于此活动的 ESTIMATEDSQLCOST 阈值（如果有的话）。

estimatedsqlcost_threshold_value -“估计 SQL 成本阈值”监视元素

应用于此活动的 ESTIMATEDSQLCOST 阈值的上限。

表 618. 表函数监视信息

表函数	监视元素收集级别
MON_GET_ACTIVITY_DETAILS 表函数 - 获取完整的活动详细信息（在 XML 文档 DETAILS 中报告）	ACTIVITY METRICS BASE

用法

使用此元素来确定应用于此活动的 ESTIMATEDSQLCOST 阈值（如果有的话）。

estimatedsqlcost_threshold_violated -“违反估计 SQL 成本阈值”监视元素

此监视元素返回“Yes”表明此活动已违反 ESTIMATEDSQLCOST 阈值。“No”表明此活动尚未违反该阈值。

表 619. 表函数监视信息

表函数	监视元素收集级别
MON_GET_ACTIVITY_DETAILS 表函数 - 获取完整的活动详细信息（在 XML 文档 DETAILS 中报告）	ACTIVITY METRICS BASE

用法

使用此元素来确定此活动是否已违反应用于此活动的 ESTIMATEDSQLCOST 阈值。

event_id -“事件标识”监视元素

这是与事件相关联的标识，此标识与其他监视元素配合使用以唯一标识该事件。此监视元素在返回此元素的界面中组成 **xmlid** 监视元素的一部分。

表 620. 事件监视信息

事件类型	逻辑数据分组	监视元素收集级别
变更历史记录	changesummary dbdbmcfg ddlstmtexec evmonstart regvar txncompletion utillocation utilphase utilstart utilstop	始终收集
锁定	• event_lock • event_lock_participants • event_lock_participants _activities • event_lock_activity_values	始终收集
程序包高速缓存	• event_pkgcache • event_pkgcache_metrics • event_pkgcache_stmt_args	始终收集
工作单元	event_uow	始终收集

用法

根据出现此标识的事件记录的类型不同，此标识的值也会不同:

锁定事件监视器记录

事件的数字标识。此标识将在数据库激活时重置。其唯一性通过组合 **event_timestamp**、**event_id**、**member** 和 **event-type** 来保证。

工作单元事件监视器记录

对每个连接唯一的 UOW 标识的别名。其唯一性通过组合 **event_timestamp**、**event_id**、**event-type**、**member** 和 **appl_id** 来保证。

程序包高速缓存事件监视器记录

事件的数字标识。此标识将在数据库激活时重置。其唯一性通过组合 **event_timestamp**、**event_id**、**member** 和 **event-type** 来保证。

变更历史记录事件监视器记录

事件的数字标识。此标识将在数据库激活时重置。事件的唯一性通过组合 **event_timestamp**、**event_id**、**member** 和 **event-type** 来保证。

event_monitor_name -“事件监视器名称”

创建事件数据流的事件监视器的名称。

元素标识

event_monitor_name

元素类型

信息

表 621. 事件监视信息

事件类型	逻辑数据分组	监视开关
事件日志头	event_log_header	始终收集

用法 此元素允许您将要分析的数据与系统目录表中的特定事件监视器相关联。这是可在 SYSCAT.EVENTMONITORS 目录表的 NAME 列中找到的相同名称，该名称是在 CREATE EVENT MONITOR 和 SET EVENT MONITOR 语句上指定的。

event_time -“事件时间”

发生事件的日期和时间。

表 622. 事件监视信息

事件类型	逻辑数据分组	监视开关
表空间	event_tablespace	-
表	event_table	-

用法 可使用此元素来帮助相关事件按时间排序。

event_timestamp -“事件时间戳记”监视元素

数据库管理器生成此事件记录的时间。

表 623. 事件监视信息

事件类型	逻辑数据分组	监视元素收集级别
变更历史记录	changesummary dbdbmcfg ddlstmtexec evmonstart regvar txncompletion utillocation utilphase utilstart utilstop	始终收集
锁定	• event_lock • event_lock_participants • event_lock_participants _activities • event_lock_activity_values	始终收集

表 623. 事件监视信息 (续)

事件类型	逻辑数据分组	监视元素收集级别
程序包高速缓存	• event_pkgcache • event_pkgcache_metrics • event_pkgcache_stmt_args	始终收集
工作单元	• event_uow	始终收集

event_type -“事件类型”监视元素

要报告的事件的事件类型。仅由变更历史记录、锁定和程序包高速缓存事件监视器报告。

许多事件监视器仅捕获一种类型的事件。例如，工作单元事件监视器记录工作单元完成事件，该事件在工作单元完成时捕获其相关数据。其他事件监视器（例如，数据库、表、缓冲池或表空间事件监视器）中的每一个在数据库取消激活后捕获一种类型的事件。

但是，某些事件监视器会生成不同类型的事件。这些事件监视器报告 **event_type** 监视元素中的事件类型。表 624 显示返回此监视元素的事件监视器。第 757 页的表 625 显示此监视元素可具有的可能值。

表 624. 事件监视信息

事件监视器	逻辑数据分组	监视元素收集级别
变更历史记录	changesummary dbdbmcfg ddlstmtexec evmonstart regvar txncompletion utillocation utilphase utilstart utilstop	始终收集
锁定	• event_lock • event_lock_participants • event_lock_participants _activities • event_lock_activity_values	始终收集
工作单元	不适用。有关更多信息，请参阅 第 757 页的表 625中此事件监视器的用法说明。	始终收集
程序包高速缓存	• pkgcache	始终收集

*表 625. **event_type** 的可能值*

事件监视器	**event_type** 元素的可能值	使用说明
变更历史记录	• DBCFG • DBCFGVALUES • DBMCFG • DBMCFGVALUES • REGVAR • REGVARVALUES • DDLSTMTEXEC • TXNCOMPLETION • EVMONSTART • UTILSTART • UTILSTOP • UTILSTARTPROC • UTILSTOPPROC • UTILPHASESTART • UTILPHASESTOP	**event_type** 监视元素作为列包括在此事件监视器的输出中。
锁定	• LOCKTIMEOUT • LOCKWAIT • DEADLOCK	**event_type** 监视元素作为列包括在此事件监视器的输出中，写至常规或无格式事件 (UE) 表时。它还包括在例程 EVMON_FORMAT_UE_TO_TABLES 或 EVMON_FORMAT_UE_TO_XML 创建的输出中。
程序包高速缓存	• PKGCACHEBASE • PKGCACHEDETAILED	**event_type** 监视元素作为列包括在此事件监视器的输出中，写至无格式事件 (UE) 表（而不是常规表）时。它还包括在例程 EVMON_FORMAT_UE_TO_TABLES 或 EVMON_FORMAT_UE_TO_XML 创建的输出中。
工作单元	• UOW	**event_type** 监视元素作为列包括在此事件监视器的输出中，仅当写至无格式事件 (UE) 表时。而且，它仅包括在此事件监视器创建的 UE 表中；它未包括在例程 EVMON_FORMAT_UE_TO_TABLES 或 EVMON_FORMAT_UE_TO_XML 创建的输出中。

evmon_activates -“事件监视器激活数”

激活事件监视器的次数。

元素标识

evmon_activates

元素类型

计数器

并非所有事件监视器都会报告此监视元素。有关更多详细信息，请参阅第 758 页的表 626。

表 626. 事件监视信息

事件类型	逻辑数据分组	监视开关
数据库	event_db	始终收集
表	event_table	始终收集
表空间	event_tablespace	始终收集
缓冲池	event_bufferpool	始终收集
死锁	event_deadlock	始终收集
死锁	event_dlconn	始终收集
带有详细信息的死锁	event_detailed_dlconn	始终收集
语句	event_stmt_history	始终收集
连接数*	-	
事务数*	-	
统计信息*	-	
阈值违例数*	-	

* 此事件监视器仅更新目录表 SYSCAT.EVENTMONITORS 中的 evmon_activates 列。此监视元素未包括在写至事件监视器输出表的任何逻辑数据组中。

用法 使用此元素来关联表 626中列示的事件监视器返回的信息。此元素仅适用于“写至表”事件监视器；对于写至文件或管道的事件监视器，不会保留此元素。

该事件监视器激活时，表 626中的所有事件监视器都会更新 SYSCAT.EVENTMONITORS 目录表的 evmon_activates 列。系统会记录此更改，所以 DATABASE CONFIGURATION 会显示:

```
All committed transactions have been written to disk = NO
```

如果事件监视器是使用 AUTOSTART 选项创建的，并且第一个用户连接至数据库后立即断开连接从而使得数据库被释放，那么会生成日志文件。

除非另行说明，否则这些事件监视器还包括 evmon_activates 的值作为事件数据写至的表中的列。

此表中未包括的事件监视器不会报告 evmon_activates 监视元素。

evmon_wait_time -“事件监视器等待时间”监视元素

代理程序等待事件监视器记录变为可用的时间。

如果代理程序尝试写入事件监视器记录并被阻止直到快速写程序记录变为可用，那么会发生等待。快速写程序用于将大量事件监视器数据并行写至表、文件或管道。

时间以毫秒计。

表 627. 表函数监视信息

表函数	监视元素收集级别
MON_FORMAT_XML_METRICS_BY_ROW - 获取所有度量值的基于已格式化行的输出	不适用；报告 XML 文档中作为格式化函数的输入提供的所有元素
MON_FORMAT_XML_TIMES_BY_ROW - - 获取基于已格式化行的组合层次结构等待时间和处理时间	不适用；报告 XML 文档中作为格式化函数的输入提供的所有元素

表 627. 表函数监视信息 (续)

表函数	监视元素收集级别
MON_FORMAT_XML_WAIT_TIMES_BY _ROW - 获取等待次数的已格式化的基于行的输出	不适用; 报告 XML 文档中作为格式化函数的输入提供的所有元素
MON_GET_ACTIVITY_DETAILS 表函数 - 获取完整的活动详细信息	ACTIVITY METRICS BASE
MON_GET_CONNECTION 表函数 - 获取连接度量值	REQUEST METRICS BASE
MON_GET_CONNECTION_DETAILS 表函数 - 获取详细的连接度量值	REQUEST METRICS BASE
MON_GET_PKG_CACHE_STMT 表函数 - 获取程序包高速缓存中的 SQL 语句活动度量值	ACTIVITY METRICS BASE
MON_GET_PKG_CACHE_STMT_DETAILS 表函数 - 获取程序包高速缓存条目的详细度量值（在 XML 文档 DETAILS 中报告）	ACTIVITY METRICS BASE
MON_GET_SERVICE_SUBCLASS 表函数 - 获取服务子类度量值	REQUEST METRICS BASE
MON_GET_SERVICE_SUBCLASS_DETAILS 表函数 - 获取详细的服务子类度量值	REQUEST METRICS BASE
MON_GET_UNIT_OF_WORK 表函数 - 获取工作单元度量值	REQUEST METRICS BASE
MON_GET_UNIT_OF_WORK_DETAILS 表函数 - 获取详细的工作单元度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE
MON_GET_WORKLOAD 表函数 - 获取工作负载度量值	REQUEST METRICS BASE
MON_GET_WORKLOAD_DETAILS 表函数 - 获取详细的工作负载度量值	REQUEST METRICS BASE

表 628. 事件监视信息

事件类型	事件监视器输出类型	逻辑数据分组	监视开关
工作单元	TABLE、FILE 和 PIPE	uow_metrics	REQUEST METRICS BASE
工作单元	UE TABLE, 由 EVMON_FORMAT_UE 处理 _TO_XML 函数	在 system_metrics 文档中报告 始终收集	REQUEST METRICS BASE
工作单元	UE TABLE, 由 EVMON_FORMAT_UE 处理 _TO_TABLES 函数	在 UOW_EVENT 表的度量值文档中和 UOW_METRICS 表中报告 始终收集	REQUEST METRICS BASE
程序包高速缓存	TABLE、FILE 和 PIPE	pkgcache_metrics	ACTIVITY METRICS BASE

表 628. 事件监视信息 (续)

事件类型	事件监视器输出类型	逻辑数据分组	监视开关
程序包高速缓存	UE TABLE，由 EVMON_FORMAT_UE 处理 _TO_XML 函数	在 activity_metrics 文档中报告始终收集	ACTIVITY METRICS BASE
程序包高速缓存	UE TABLE，由 EVMON_FORMAT_UE 处理 _TO_TABLES 函数	在 PKGCACHE_EVENT 表的度量值文档中和 PKGCACHE_METRICS 表中报告始终收集	ACTIVITY METRICS BASE
活动	TABLE、FILE 和 PIPE	event_activitymetrics	ACTIVITY METRICS BASE
活动	TABLE、FILE 和 PIPE	event_activity（在 XML 文档 DETAILS 中报告）	ACTIVITY METRICS BASE
统计信息	TABLE、FILE 和 PIPE	event_scstats（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE
统计信息	TABLE、FILE 和 PIPE	event_wlstats（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE

evmon_waits_total -“事件监视器总等待次数”监视元素

代理程序等待事件监视器记录变为可用的次数。

如果代理程序尝试写入事件监视器记录并被阻止直到快速写程序记录变为可用，那么会发生等待。快速写程序用于将大量事件监视器数据并行写至表、文件或管道。

表 629. 表函数监视信息

表函数	监视元素收集级别
MON_FORMAT_XML_METRICS_BY_ROW - 获取所有度量值的基于已格式化行的输出	不适用；报告 XML 文档中作为格式化函数的输入提供的所有元素
MON_GET_ACTIVITY_DETAILS 表函数 - 获取完整的活动详细信息	ACTIVITY METRICS BASE
MON_GET_CONNECTION 表函数 - 获取连接度量值	REQUEST METRICS BASE
MON_GET_CONNECTION_DETAILS 表函数 - 获取详细的连接度量值	REQUEST METRICS BASE
MON_GET_PKG_CACHE_STMT 表函数 - 获取程序包高速缓存中的 SQL 语句活动度量值	ACTIVITY METRICS BASE
MON_GET_PKG_CACHE_STMT_DETAILS 表函数 - 获取程序包高速缓存条目的详细度量值（在 XML 文档 DETAILS 中报告）	ACTIVITY METRICS BASE
MON_GET_SERVICE_SUBCLASS 表函数 - 获取服务子类度量值	REQUEST METRICS BASE
MON_GET_SERVICE_SUBCLASS_DETAILS 表函数 - 获取详细的服务子类度量值	REQUEST METRICS BASE
MON_GET_UNIT_OF_WORK 表函数 - 获取工作单元度量值	REQUEST METRICS BASE

表 629. 表函数监视信息 (续)

表函数	监视元素收集级别
MON_GET_UNIT_OF_WORK_DETAILS 表函数 - 获取详细的工作单元度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE
MON_GET_WORKLOAD 表函数 - 获取工作负载度量值	REQUEST METRICS BASE
MON_GET_WORKLOAD_DETAILS 表函数 - 获取详细的工作负载度量值	REQUEST METRICS BASE

表 630. 事件监视信息

事件类型	事件监视器输出类型	逻辑数据分组	监视开关
工作单元	TABLE	uow_metrics	REQUEST METRICS BASE
工作单元	UE TABLE, 由 EVMON_FORMAT_UE 处理 _TO_XML 函数	在 system_metrics 文档中报告 始终收集	REQUEST METRICS BASE
工作单元	UE TABLE, 由 EVMON_FORMAT_UE 处理 _TO_TABLES 函数	在 UOW_EVENT 表的度量值文档中和 UOW_METRICS 表中报告始终收集	REQUEST METRICS BASE
程序包高速缓存	TABLE	pkgcache_metrics	ACTIVITY METRICS BASE
程序包高速缓存	UE TABLE, 由 EVMON_FORMAT_UE 处理 _TO_XML 函数	在 activity_metrics 文档中报告始终收集	ACTIVITY METRICS BASE
程序包高速缓存	UE TABLE, 由 EVMON_FORMAT_UE 处理 _TO_TABLES 函数	在 PKGCACHE_EVENT 表的度量值文档中和 PKGCACHE_METRICS 表中报告始终收集	ACTIVITY METRICS BASE
活动	TABLE	event_activitymetrics	ACTIVITY METRICS BASE
活动	TABLE	event_activity（在 DETAILS_XML 文档中报告）	ACTIVITY METRICS BASE
统计信息	TABLE	event_scstats（在 DETAILS_XML 文档中报告）	REQUEST METRICS BASE
统计信息	TABLE	event_wlstats（在 DETAILS_XML 文档中报告）	REQUEST METRICS BASE

executable_id -“可执行文件标识”监视元素

在数据服务器上生成的加密二进制标记，用于唯一地标识已执行的 SQL 语句节。对于非 SQL 活动，将返回长度为 0 的字符串值。

表 631. 表函数监视信息

表函数	监视元素收集级别
MON_GET_ACTIVITY_DETAILS 表函数 - 获取完整的活动详细信息（在 XML 文档 DETAILS 中报告）	ACTIVITY METRICS BASE
MON_GET_INDEX_USAGE_LIST 表函数 - 返回索引用法列表中的信息	ACTIVITY METRICS BASE
MON_GET_PKG_CACHE_STMT 表函数 - 获取程序包高速缓存中的 SQL 语句活动度量值	ACTIVITY METRICS BASE
MON_GET_PKG_CACHE_STMT_DETAILS 表函数 - 获取程序包高速缓存条目的详细度量值	ACTIVITY METRICS BASE
MON_GET_TABLE_USAGE_LIST 表函数 - 从表用法列表返回信息	ACTIVITY METRICS BASE
WLM_GET_SERVICE_CLASS_AGENTS 表函数 - 列示正在服务类中运行的代理程序	ACTIVITY METRICS BASE
WLM_GET_WORKLOAD_OCCURRENCE_ACTIVITIES 表函数 - 返回活动列表	ACTIVITY METRICS BASE

表 632. 事件监视信息

事件类型	逻辑数据分组	监视开关
活动	event_activitystmt	-
程序包高速缓存	-	COLLECT BASE DATA

用法

使用此监视元素作为不同监视接口的输入，以获取关于该节的数据。MON_GET_PKG_CACHE_STMT 表函数（用于获取程序包高速缓存中的 SQL 语句活动度量值）接受可执行文件标识作为输入。

executable_list_size -“可执行列表大小”监视元素

特定工作单元的可执行标识列表中的条目数。

表 633. 事件监视信息

事件类型	逻辑数据分组	监视开关
工作单元	uow	

executable_list_truncated -“截断可执行列表”监视元素

指示可执行列表是否已截断。可能的值为 YES 或 NO。如果处理期间没有足够可用内存来存储整个可执行列表，那么该列表可能会被截断。

表 634. 事件监视信息

事件类型	逻辑数据分组	监视开关
工作单元	uow	

evmon_flushes -“事件监视器清空数”

发出 FLUSH EVENT MONITOR SQL 语句的次数。

元素标识

evmon_flushes

元素类型

信息

表 635. 事件监视信息

事件类型	逻辑数据分组	监视开关
数据库	event_db	始终收集
表	event_table	始终收集
表空间	event_tablespace	始终收集
缓冲池	event_bufferpool	始终收集

用法 应用程序连接至数据库之后，数据库管理器每次成功处理 FLUSH EVENT MONITOR SQL 请求时，此标识就会递增。此元素有助于唯一标识数据库、表、表空间和缓冲池数据。

executable_id -“可执行文件标识”监视元素

在数据服务器上生成的加密二进制标记，用于唯一地标识已执行的 SQL 语句节。对于非 SQL 活动，将返回长度为 0 的字符串值。

表 636. 表函数监视信息

表函数	监视元素收集级别
MON_GET_ACTIVITY_DETAILS 表函数 - 获取完整的活动详细信息（在 XML 文档 DETAILS 中报告）	ACTIVITY METRICS BASE
MON_GET_INDEX_USAGE_LIST 表函数 - 返回索引用法列表中的信息	ACTIVITY METRICS BASE
MON_GET_PKG_CACHE_STMT 表函数 - 获取程序包高速缓存中的 SQL 语句活动度量值	ACTIVITY METRICS BASE
MON_GET_PKG_CACHE_STMT_DETAILS 表函数 - 获取程序包高速缓存条目的详细度量值	ACTIVITY METRICS BASE
MON_GET_TABLE_USAGE_LIST 表函数 - 从表用法列表返回信息	ACTIVITY METRICS BASE

表 636. 表函数监视信息 (续)

表函数	监视元素收集级别
WLM_GET_SERVICE_CLASS_AGENTS 表函数 - 列示正在服务类中运行的代理程序	ACTIVITY METRICS BASE
WLM_GET_WORKLOAD_OCCURRENCE_ACTIVITIES 表函数 - 返回活动列表	ACTIVITY METRICS BASE

表 637. 事件监视信息

事件类型	逻辑数据分组	监视开关
活动	event_activitystmt	-
程序包高速缓存	-	COLLECT BASE DATA

用法

使用此监视元素作为不同监视接口的输入，以获取关于该节的数据。MON_GET_PKG_CACHE_STMT 表函数（用于获取程序包高速缓存中的 SQL 语句活动度量值）接受可执行文件标识作为输入。

execution_id -“用户登录标识”

用户在登录至操作系统时指定的标识。此标识与 auth_id 不同，auth_id 是用户在连接至数据库时指定的。

表 638. 快照监视信息

快照级别	逻辑数据分组	监视开关
应用程序	appl_info	基本
应用程序	appl	基本
DCS 应用程序	dcs_appl_info	基本

表 639. 事件监视信息

事件类型	逻辑数据分组	监视开关
连接	event_connheader	-

用法 可使用此元素来确定个人的操作系统用户标识，这些人正在运行正在监视的应用程序。

failed_sql_stmts -“失败的语句操作”

尝试但失败了的 SQL 语句数。

表 640. 快照监视信息

快照级别	逻辑数据分组	监视开关
数据库	dbase	基本
数据库	dbase_remote	基本
应用程序	appl	基本
应用程序	appl_remote	基本

表 640. 快照监视信息 (续)

快照级别	逻辑数据分组	监视开关
DCS 数据库	dcs_dbase	基本
DCS 应用程序	dcs_appl	基本

可将快照监视的计数器重置。

表 641. 事件监视信息

事件类型	逻辑数据分组	监视开关
数据库	event_db	始终收集
连接	event_conn	始终收集

用法 可使用此元素来计算在数据库或应用程序级别成功执行的 SQL 语句总数:

```
    dynamic_sql_stmts
+ static_sql_stmts
- failed_sql_stmts
= 监视期间的吞吐量
```

此计数包括接收到负值 SQLCODE 的所有 SQL 语句数。

此元素还可帮助您确定性能低下的原因，这是因为失败的语句意味着数据库管理器浪费了时间并因而导致数据库的吞吐量很低。

fcm_congested_sends -“FCM 拥塞发送数”监视元素

向远程成员发送但因为拥塞而延迟的操作的数目。

通过网络连接发送的数据因为远程成员未接收到先前发送的数据而阻塞时，会出现拥塞状态。拥塞可能是接收方处理的结果延迟问题或导致包丢失和重新传输数据的网络问题。

表 642. 表函数监视信息

表函数	监视元素收集级别
MON_GET_FCM_CONNECTION_LIST - 获取有关所有 FCM 连接的详细信息	ACTIVITY METRICS BASE

fcm_congestion_time -“FCM 拥塞时间”监视元素

与远程成员的 FCM 网络连接处于拥塞状态的持续时间。计量单位为毫秒。

通过网络连接发送的数据因为远程成员未接收到先前发送的数据而阻塞时，会出现拥塞状态。拥塞可能是接收方处理的结果延迟问题或导致包丢失和重新传输数据的网络问题。

表 643. 表函数监视信息

表函数	监视元素收集级别
MON_GET_FCM_CONNECTION_LIST - 获取有关所有 FCM 连接的详细信息	ACTIVITY METRICS BASE

fcm_num_congestion_timeouts -“FCM 拥塞超时数”监视元素

FCM 网络连接上的拥塞状态在给定时间段内未自己解决的次数。如果发送方因为远程成员暂时不可访问而断开连接，那么发生了超时。

通过网络连接发送的数据因为远程成员未接收到先前发送的数据而阻塞时，会出现拥塞状态。拥塞可能是接收方处理的结果延迟问题或导致包丢失和重新传输数据的网络问题。

表 644. 表函数监视信息

表函数	监视元素收集级别
MON_GET_FCM_CONNECTION_LIST - 获取有关所有 FCM 连接的详细信息	ACTIVITY METRICS BASE

fcm_num_conn_lost -“FCM 连接断开次数”监视元素

与远程成员的 FCM 网络连接意外断开的次数。

表 645. 表函数监视信息

表函数	监视元素收集级别
MON_GET_FCM_CONNECTION_LIST - 获取有关所有 FCM 连接的详细信息	ACTIVITY METRICS BASE

fcm_num_conn_timeouts -“FCM 连接超时次数”监视元素

尝试与远程成员建立 FCM 网络连接时的超时次数。

表 646. 表函数监视信息

表函数	监视元素收集级别
MON_GET_FCM_CONNECTION_LIST - 获取有关所有 FCM 连接的详细信息	ACTIVITY METRICS BASE

fcm_message_recv_volume -“接收 FCM 消息量”监视元素

为 FCM 通信层所分发的内部请求（例如 RPC）接收的数据量。此值以字节计。

表 647. 表函数监视信息

表函数	监视元素收集级别
MON_FORMAT_XML_METRICS_BY_ROW - 获取所有度量值的基于已格式化行的输出	不适用; 报告 XML 文档中作为格式化函数的输入提供的所有元素
MON_GET_ACTIVITY_DETAILS 表函数 - 获取完整的活动详细信息（在 XML 文档 DETAILS 中报告）	ACTIVITY METRICS BASE
MON_GET_CONNECTION 表函数 - 获取连接度量值	REQUEST METRICS BASE

表 647. 表函数监视信息 (续)

表函数	监视元素收集级别
MON_GET_CONNECTION_DETAILS 表函数 - 获取详细的连接度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE
MON_GET_FCM_CONNECTION_LIST - 获取有关所有 FCM 连接的详细信息	ACTIVITY METRICS BASE
MON_GET_PKG_CACHE_STMT 表函数 - 获取程序包高速缓存中的 SQL 语句活动度量值	ACTIVITY METRICS BASE
MON_GET_PKG_CACHE_STMT_DETAILS 表函数 - 获取程序包高速缓存条目的详细度量值	ACTIVITY METRICS BASE
MON_GET_SERVICE_SUBCLASS 表函数 - 获取服务子类度量值	REQUEST METRICS BASE
MON_GET_SERVICE_SUBCLASS_DETAILS 表函数 - 获取详细的服务子类度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE
MON_GET_UNIT_OF_WORK 表函数 - 获取工作单元度量值	REQUEST METRICS BASE
MON_GET_UNIT_OF_WORK 表函数 - 获取详细的工作单元度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE
MON_GET_WORKLOAD 表函数 - 获取工作负载度量值	REQUEST METRICS BASE
MON_GET_WORKLOAD_DETAILS 表函数 - 获取详细的工作负载度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE

表 648. 事件监视信息

事件类型	逻辑数据分组	监视开关
活动	event_activity（在 details_xml 文档中报告）	ACTIVITY METRICS BASE
活动	event_activitymetrics	ACTIVITY METRICS BASE
统计信息	event_scstats（在 details_xml 文档中报告）	REQUEST METRICS BASE
统计信息	event_wlstats（在 details_xml 文档中报告）	REQUEST METRICS BASE
工作单元	在 system_metrics 文档中报告。	REQUEST METRICS BASE
程序包高速缓存	在 activity_metrics 文档中报告。	ACTIVITY METRICS BASE

用法

使用此元素来确定在通过 FCM 子系统发送的数据量中，用于请求或应答消息流量（而不是实际的表数据）的数据量。

fcm_message_recv_wait_time -“接收 FCM 消息等待时间”监视元素

代理程序在等待 FCM 应答消息时耗用的时间（该消息包含先前发送的 FCM 请求消息的结果）。此值既反映使用 FCM 在分区之间发送响应所需的时间，也反映子代理程序处理请求消息所需的时间。此值以毫秒计。

表 649. 表函数监视信息

表函数	监视元素收集级别
MON_FORMAT_XML_METRICS_BY_ROW - 获取所有度量值的基于已格式化行的输出	不适用; 报告 XML 文档中作为格式化函数的输入提供的所有元素
MON_FORMAT_XML_TIMES_BY_ROW - - 获取基于已格式化行的组合层次结构等待时间和处理时间	不适用; 报告 XML 文档中作为格式化函数的输入提供的所有元素
MON_FORMAT_XML_WAIT_TIMES_BY _ROW - 获取等待时间的已格式化的基于行的输出	不适用; 报告 XML 文档中作为格式化函数的输入提供的所有元素
MON_GET_ACTIVITY_DETAILS 表函数 - 获取完整的活动详细信息（在 XML 文档 DETAILS 中报告）	ACTIVITY METRICS BASE
MON_GET_CONNECTION 表函数 - 获取连接度量值	REQUEST METRICS BASE
MON_GET_CONNECTION_DETAILS 表函数 - 获取详细的连接度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE
MON_GET_PKG_CACHE_STMT 表函数 - 获取程序包高速缓存中的 SQL 语句活动度量值	ACTIVITY METRICS BASE
MON_GET_PKG_CACHE_STMT_DETAILS 表函数 - 获取程序包高速缓存条目的详细度量值	ACTIVITY METRICS BASE
MON_GET_SERVICE_SUBCLASS 表函数 - 获取服务子类度量值	REQUEST METRICS BASE
MON_GET_SERVICE_SUBCLASS_DETAILS 表函数 - 获取详细的服务子类度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE
MON_GET_UNIT_OF_WORK 表函数 - 获取工作单元度量值	REQUEST METRICS BASE
MON_GET_UNIT_OF_WORK 表函数 - 获取详细的工作单元度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE
MON_GET_WORKLOAD 表函数 - 获取工作负载度量值	REQUEST METRICS BASE
MON_GET_WORKLOAD_DETAILS 表函数 - 获取详细的工作负载度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE

表 650. 事件监视信息

事件类型	逻辑数据分组	监视开关
活动	event_activity（在 details_xml 文档中报告）	ACTIVITY METRICS BASE
活动	event_activitymetrics	ACTIVITY METRICS BASE
统计信息	event_scstats（在 details_xml 文档中报告）	REQUEST METRICS BASE
统计信息	event_wlstats（在 details_xml 文档中报告）	REQUEST METRICS BASE
工作单元	在 system_metrics 文档中报告。	REQUEST METRICS BASE
程序包高速缓存	在 activity_metrics 文档中报告。	ACTIVITY METRICS BASE

用法

此元素可用于确定在多分区实例的给定分区中，等待在其他分区中处理请求时耗用的时间。

fcm_message_recvs_total -“接收 FCM 消息总数”监视元素

作为 FCM 应答消息的组成部分接收的缓冲区的总数（该消息包含先前发送的 FCM 请求消息的结果）。

表 651. 表函数监视信息

表函数	监视元素收集级别
MON_FORMAT_XML_METRICS_BY_ROW - 获取所有度量值的基于已格式化行的输出	不适用; 报告 XML 文档中作为格式化函数的输入提供的所有元素
MON_GET_ACTIVITY_DETAILS 表函数 - 获取完整的活动详细信息（在 XML 文档 DETAILS 中报告）	ACTIVITY METRICS BASE
MON_GET_CONNECTION 表函数 - 获取连接度量值	REQUEST METRICS BASE
MON_GET_CONNECTION_DETAILS 表函数 - 获取详细的连接度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE
MON_GET_PKG_CACHE_STMT 表函数 - 获取程序包高速缓存中的 SQL 语句活动度量值	ACTIVITY METRICS BASE
MON_GET_PKG_CACHE_STMT_DETAILS 表函数 - 获取程序包高速缓存条目的详细度量值	ACTIVITY METRICS BASE
MON_GET_SERVICE_SUBCLASS 表函数 - 获取服务子类度量值	REQUEST METRICS BASE
MON_GET_SERVICE_SUBCLASS_DETAILS 表函数 - 获取详细的服务子类度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE
MON_GET_UNIT_OF_WORK 表函数 - 获取工作单元度量值	REQUEST METRICS BASE

表 651. 表函数监视信息 (续)

表函数	监视元素收集级别
MON_GET_UNIT_OF_WORK 表函数 - 获取详细的工作单元度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE
MON_GET_WORKLOAD 表函数 - 获取工作负载度量值	REQUEST METRICS BASE
MON_GET_WORKLOAD_DETAILS 表函数 - 获取详细的工作负载度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE

表 652. 事件监视信息

事件类型	逻辑数据分组	监视开关
活动	event_activity（在 details_xml 文档中报告）	ACTIVITY METRICS BASE
活动	event_activitymetrics	ACTIVITY METRICS BASE
统计信息	event_scstats（在 details_xml 文档中报告）	REQUEST METRICS BASE
统计信息	event_wlstats（在 details_xml 文档中报告）	REQUEST METRICS BASE
程序包高速缓存	在 activity_metrics 文档中报告。	ACTIVITY METRICS BASE
工作单元	在 system_metrics 文档中报告。	REQUEST METRICS BASE

用法

此元素可用于确定所接收的每条 FCM 消息的平均数据量以及等待接收单一 FCM 消息时的平均耗用时间。

fcm_message_send_volume -“发送 FCM 消息量”监视元素

通过内部 FCM 请求发送的数据量。此值以字节计。

表 653. 表函数监视信息

表函数	监视元素收集级别
MON_FORMAT_XML_METRICS_BY_ROW - 获取所有度量值的基于已格式化行的输出	不适用; 报告 XML 文档中作为格式化函数的输入提供的所有元素
MON_GET_ACTIVITY_DETAILS 表函数 - 获取完整的活动详细信息（在 XML 文档 DETAILS 中报告）	ACTIVITY METRICS BASE
MON_GET_CONNECTION 表函数 - 获取连接度量值	REQUEST METRICS BASE
MON_GET_CONNECTION_DETAILS 表函数 - 获取详细的连接度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE

表 653. 表函数监视信息 (续)

表函数	监视元素收集级别
MON_GET_FCM_CONNECTION_LIST - 获取有关所有 FCM 连接的详细信息	ACTIVITY METRICS BASE
MON_GET_PKG_CACHE_STMT 表函数 - 获取程序包高速缓存中的 SQL 语句活动度量值	ACTIVITY METRICS BASE
MON_GET_PKG_CACHE_STMT_DETAILS 表函数 - 获取程序包高速缓存条目的详细度量值	ACTIVITY METRICS BASE
MON_GET_SERVICE_SUBCLASS 表函数 - 获取服务子类度量值	REQUEST METRICS BASE
MON_GET_SERVICE_SUBCLASS_DETAILS 表函数 - 获取详细的服务子类度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE
MON_GET_UNIT_OF_WORK 表函数 - 获取工作单元度量值	REQUEST METRICS BASE
MON_GET_UNIT_OF_WORK 表函数 - 获取详细的工作单元度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE
MON_GET_WORKLOAD 表函数 - 获取工作负载度量值	REQUEST METRICS BASE
MON_GET_WORKLOAD_DETAILS 表函数 - 获取详细的工作负载度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE

表 654. 事件监视信息

事件类型	逻辑数据分组	监视开关
活动	event_activity（在 details_xml 文档中报告）	ACTIVITY METRICS BASE
活动	event_activitymetrics	ACTIVITY METRICS BASE
统计信息	event_scstats（在 details_xml 文档中报告）	REQUEST METRICS BASE
统计信息	event_wlstats（在 details_xml 文档中报告）	REQUEST METRICS BASE
程序包高速缓存	在 activity_metrics 文档中报告。	ACTIVITY METRICS BASE
工作单元	在 system_metrics 文档中报告。	REQUEST METRICS BASE

用法

使用此元素来确定在通过 FCM 子系统发送的数据量中，用于发送请求和应答消息流量（而不是实际的表数据）的数据量。

fcm_message_send_wait_time -“发送 FCM 消息等待时间”监视元素

发送 FCM 消息时被阻塞的时间。此值以毫秒计。此监视元素反映在数据库系统中分发内部请求期间将 FCM 缓冲区从 FCM 通道中清仓时被阻塞的时间。

表 655. 表函数监视信息

表函数	监视元素收集级别
MON_FORMAT_XML_METRICS_BY_ROW - 获取所有度量值的基于已格式化行的输出	不适用; 报告 XML 文档中作为格式化函数的输入提供的所有元素
MON_FORMAT_XML_TIMES_BY_ROW - - 获取基于已格式化行的组合层次结构等待时间和处理时间	不适用; 报告 XML 文档中作为格式化函数的输入提供的所有元素
MON_FORMAT_XML_WAIT_TIMES_BY _ROW - 获取等待时间的已格式化的基于行的输出	不适用; 报告 XML 文档中作为格式化函数的输入提供的所有元素
MON_GET_ACTIVITY_DETAILS 表函数 - 获取完整的活动详细信息（在 XML 文档 DETAILS 中报告）	ACTIVITY METRICS BASE
MON_GET_CONNECTION 表函数 - 获取连接度量值	REQUEST METRICS BASE
MON_GET_CONNECTION_DETAILS 表函数 - 获取详细的连接度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE
MON_GET_PKG_CACHE_STMT 表函数 - 获取程序包高速缓存中的 SQL 语句活动度量值	ACTIVITY METRICS BASE
MON_GET_PKG_CACHE_STMT_DETAILS 表函数 - 获取程序包高速缓存条目的详细度量值	ACTIVITY METRICS BASE
MON_GET_SERVICE_SUBCLASS 表函数 - 获取服务子类度量值	REQUEST METRICS BASE
MON_GET_SERVICE_SUBCLASS_DETAILS 表函数 - 获取详细的服务子类度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE
MON_GET_UNIT_OF_WORK 表函数 - 获取工作单元度量值	REQUEST METRICS BASE
MON_GET_UNIT_OF_WORK 表函数 - 获取详细的工作单元度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE
MON_GET_WORKLOAD 表函数 - 获取工作负载度量值	REQUEST METRICS BASE
MON_GET_WORKLOAD_DETAILS 表函数 - 获取详细的工作负载度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE

表 656. 事件监视信息

事件类型	逻辑数据分组	监视开关
活动	event_activity（在 details_xml 文档中报告）	ACTIVITY METRICS BASE

表 656. 事件监视信息 (续)

事件类型	逻辑数据分组	监视开关
活动	event_activitymetrics	ACTIVITY METRICS BASE
统计信息	event_scstats（在 details_xml 文档中报告）	REQUEST METRICS BASE
统计信息	event_wlstats（在 details_xml 文档中报告）	REQUEST METRICS BASE
程序包高速缓存	在 activity_metrics 文档中报告。	ACTIVITY METRICS BASE
工作单元	在 system_metrics 文档中报告。	REQUEST METRICS BASE

用法

使用此元素来确定代理程序等待通过 FCM 子系统发送 FCM 请求消息时耗用的时间。根据 FCM 守护程序的繁忙程度不同，代理程序在尝试发送消息时可能需要等待。

fcm_message_sends_total -“发送 FCM 消息总数”监视元素

使用 FCM 通信机制作为内部请求的组成部分分发的缓冲区的总数。

表 657. 表函数监视信息

表函数	监视元素收集级别
MON_FORMAT_XML_METRICS_BY_ROW - 获取所有度量值的基于已格式化行的输出	不适用; 报告 XML 文档中作为格式化函数的输入提供的所有元素
MON_GET_ACTIVITY_DETAILS 表函数 - 获取完整的活动详细信息（在 XML 文档 DETAILS 中报告）	ACTIVITY METRICS BASE
MON_GET_CONNECTION 表函数 - 获取连接度量值	REQUEST METRICS BASE
MON_GET_CONNECTION_DETAILS 表函数 - 获取详细的连接度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE
MON_GET_PKG_CACHE_STMT 表函数 - 获取程序包高速缓存中的 SQL 语句活动度量值	ACTIVITY METRICS BASE
MON_GET_PKG_CACHE_STMT_DETAILS 表函数 - 获取程序包高速缓存条目的详细度量值	ACTIVITY METRICS BASE
MON_GET_SERVICE_SUBCLASS 表函数 - 获取服务子类度量值	REQUEST METRICS BASE
MON_GET_SERVICE_SUBCLASS_DETAILS 表函数 - 获取详细的服务子类度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE
MON_GET_UNIT_OF_WORK 表函数 - 获取工作单元度量值	REQUEST METRICS BASE
MON_GET_UNIT_OF_WORK 表函数 - 获取详细的工作单元度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE

表 657. 表函数监视信息 (续)

表函数	监视元素收集级别
MON_GET_WORKLOAD 表函数 - 获取工作负载度量值	REQUEST METRICS BASE
MON_GET_WORKLOAD_DETAILS 表函数 - 获取详细的工作负载度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE

表 658. 事件监视信息

事件类型	逻辑数据分组	监视开关
活动	event_activity（在 details_xml 文档中报告）	ACTIVITY METRICS BASE
活动	event_activitymetrics	ACTIVITY METRICS BASE
统计信息	event_scstats（在 details_xml 文档中报告）	REQUEST METRICS BASE
统计信息	event_wlstats（在 details_xml 文档中报告）	REQUEST METRICS BASE
工作单元	在 system_metrics 文档中报告。	REQUEST METRICS BASE
程序包高速缓存	在 activity_metrics 文档中报告。	ACTIVITY METRICS BASE

用法

使用此元素来确定为每条 FCM 请求消息发送的平均数据量以及处理每条 FCM 消息时的平均等待时间。

fcm_recv_volume -“FCM 接收量”监视元素

通过 FCM 通信层接收的数据总量。此值以字节计。

表 659. 表函数监视信息

表函数	监视元素收集级别
MON_FORMAT_XML_METRICS_BY_ROW - 获取所有度量值的基于已格式化行的输出	不适用; 报告 XML 文档中作为格式化函数的输入提供的所有元素
MON_GET_ACTIVITY_DETAILS 表函数 - 获取完整的活动详细信息（在 XML 文档 DETAILS 中报告）	ACTIVITY METRICS BASE
MON_GET_CONNECTION 表函数 - 获取连接度量值	REQUEST METRICS BASE
MON_GET_CONNECTION_DETAILS 表函数 - 获取详细的连接度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE
MON_GET_FCM_CONNECTION_LIST - 获取有关所有 FCM 连接的详细信息	ACTIVITY METRICS BASE
MON_GET_PKG_CACHE_STMT 表函数 - 获取程序包高速缓存中的 SQL 语句活动度量值	ACTIVITY METRICS BASE

表 659. 表函数监视信息 (续)

表函数	监视元素收集级别
MON_GET_PKG_CACHE_STMT_DETAILS - 获取程序包高速缓存条目的详细度量值	ACTIVITY METRICS BASE
MON_GET_SERVICE_SUBCLASS 表函数 - 获取服务子类度量值	REQUEST METRICS BASE
MON_GET_SERVICE_SUBCLASS_DETAILS 表函数 - 获取详细的服务子类度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE
MON_GET_UNIT_OF_WORK 表函数 - 获取工作单元度量值	REQUEST METRICS BASE
MON_GET_UNIT_OF_WORK 表函数 - 获取详细的工作单元度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE
MON_GET_WORKLOAD 表函数 - 获取工作负载度量值	REQUEST METRICS BASE
MON_GET_WORKLOAD_DETAILS 表函数 - 获取详细的工作负载度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE

表 660. 事件监视信息

事件类型	逻辑数据分组	监视开关
活动	event_activity（在 details_xml 文档中报告）	ACTIVITY METRICS BASE
活动	event_activitymetrics	ACTIVITY METRICS BASE
统计信息	event_scstats（在 details_xml 文档中报告）	REQUEST METRICS BASE
统计信息	event_wlstats（在 details_xml 文档中报告）	REQUEST METRICS BASE
工作单元	在 system_metrics 文档中报告。	REQUEST METRICS BASE
程序包高速缓存	在 activity_metrics 文档中报告。	ACTIVITY METRICS BASE

用法

指示在此分区中使用 FCM 接收的数据总量，其中包括消息流量和表队列数据量。

fcm_recv_wait_time -“FCM 接收等待时间”监视元素

等待通过 FCM 接收数据时的耗用时间总计。此值以毫秒计。

表 661. 表函数监视信息

表函数	监视元素收集级别
MON_FORMAT_XML_METRICS_BY_ROW - 获取所有度量值的基于已格式化行的输出	不适用；报告 XML 文档中作为格式化函数的输入提供的所有元素

表 661. 表函数监视信息 (续)

表函数	监视元素收集级别
MON_FORMAT_XML_TIMES_BY_ROW - - 获取基于已格式化行的组合层次结构等待时间和处理时间	不适用; 报告 XML 文档中作为格式化函数的输入提供的所有元素
MON_FORMAT_XML_WAIT_TIMES_BY _ROW - 获取等待时间的已格式化的基于行的输出	不适用; 报告 XML 文档中作为格式化函数的输入提供的所有元素
MON_GET_ACTIVITY_DETAILS 表函数 - 获取完整的活动详细信息（在 XML 文档 DETAILS 中报告）	ACTIVITY METRICS BASE
MON_GET_CONNECTION 表函数 - 获取连接度量值	REQUEST METRICS BASE
MON_GET_CONNECTION_DETAILS 表函数 - 获取详细的连接度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE
MON_GET_PKG_CACHE_STMT 表函数 - 获取程序包高速缓存中的 SQL 语句活动度量值	ACTIVITY METRICS BASE
MON_GET_PKG_CACHE_STMT_DETAILS 表函数 - 获取程序包高速缓存条目的详细度量值	ACTIVITY METRICS BASE
MON_GET_SERVICE_SUBCLASS 表函数 - 获取服务子类度量值	REQUEST METRICS BASE
MON_GET_SERVICE_SUBCLASS_DETAILS 表函数 - 获取详细的服务子类度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE
MON_GET_UNIT_OF_WORK 表函数 - 获取工作单元度量值	REQUEST METRICS BASE
MON_GET_UNIT_OF_WORK 表函数 - 获取详细的工作单元度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE
MON_GET_WORKLOAD 表函数 - 获取工作负载度量值	REQUEST METRICS BASE
MON_GET_WORKLOAD_DETAILS 表函数 - 获取详细的工作负载度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE

表 662. 事件监视信息

事件类型	逻辑数据分组	监视开关
活动	event_activity（在 details_xml 文档中报告）	ACTIVITY METRICS BASE
活动	event_activitymetrics	ACTIVITY METRICS BASE
统计信息	event_scstats（在 details_xml 文档中报告）	REQUEST METRICS BASE
统计信息	event_wlstats（在 details_xml 文档中报告）	REQUEST METRICS BASE
工作单元	在 system_metrics 文档中报告。	REQUEST METRICS BASE

表 662. 事件监视信息 (续)

事件类型	逻辑数据分组	监视开关
程序包高速缓存	在 activity_metrics 文档中报告。	ACTIVITY METRICS BASE

用法

使用此元素来确定在此数据库分区中等待通过 FCM 接收数据时的耗用时间总计。这既包括请求消息应答数据也包括表队列数据。

fcm_recvs_total -“FCM 接收总计”监视元素

使用 FCM 通信机制为内部请求接收的缓冲区的总数。fcm_recvs_total 监视元素值是 fcm_message_recvs_total 监视元素值与 fcm_tq_recvs_total 监视元素值之和。

表 663. 表函数监视信息

表函数	监视元素收集级别
MON_FORMAT_XML_METRICS_BY_ROW - 获取所有度量值的基于已格式化行的输出	不适用; 报告 XML 文档中作为格式化函数的输入提供的所有元素
MON_GET_ACTIVITY_DETAILS 表函数 - 获取完整的活动详细信息（在 XML 文档 DETAILS 中报告）	ACTIVITY METRICS BASE
MON_GET_CONNECTION 表函数 - 获取连接度量值	REQUEST METRICS BASE
MON_GET_CONNECTION_DETAILS 表函数 - 获取详细的连接度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE
MON_GET_PKG_CACHE_STMT 表函数 - 获取程序包高速缓存中的 SQL 语句活动度量值	ACTIVITY METRICS BASE
MON_GET_PKG_CACHE_STMT_DETAILS 表函数 - 获取程序包高速缓存条目的详细度量值	ACTIVITY METRICS BASE
MON_GET_SERVICE_SUBCLASS 表函数 - 获取服务子类度量值	REQUEST METRICS BASE
MON_GET_SERVICE_SUBCLASS_DETAILS 表函数 - 获取详细的服务子类度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE
MON_GET_UNIT_OF_WORK 表函数 - 获取工作单元度量值	REQUEST METRICS BASE
MON_GET_UNIT_OF_WORK 表函数 - 获取详细的工作单元度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE
MON_GET_WORKLOAD 表函数 - 获取工作负载度量值	REQUEST METRICS BASE
MON_GET_WORKLOAD_DETAILS 表函数 - 获取详细的工作负载度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE

表 664. 事件监视信息

事件类型	逻辑数据分组	监视开关
活动	event_activity（在 details_xml 文档中报告）	ACTIVITY METRICS BASE
活动	event_activitymetrics	ACTIVITY METRICS BASE
统计信息	event_scstats（在 details_xml 文档中报告）	REQUEST METRICS BASE
统计信息	event_wlstats（在 details_xml 文档中报告）	REQUEST METRICS BASE
工作单元	在 system_metrics 文档中报告。	REQUEST METRICS BASE
程序包高速缓存	在 activity_metrics 文档中报告。	ACTIVITY METRICS BASE

用法

通过将此元素与 **fcm_recv_wait_time** 监视元素配合使用，可以确定每个 FCM 接收操作的平均等待时间以及 FCM 接收操作所返回的平均数据量。

fcm_send_volume -“FCM 发送量”监视元素

通过 FCM 通信层分发的数据总量。此值以字节计。

表 665. 表函数监视信息

表函数	监视元素收集级别
MON_FORMAT_XML_METRICS_BY_ROW - 获取所有度量值的基于已格式化行的输出	不适用；报告 XML 文档中作为格式化函数的输入提供的所有元素
MON_GET_ACTIVITY_DETAILS 表函数 - 获取完整的活动详细信息（在 XML 文档 DETAILS 中报告）	ACTIVITY METRICS BASE
MON_GET_CONNECTION 表函数 - 获取连接度量值	REQUEST METRICS BASE
MON_GET_CONNECTION_DETAILS 表函数 - 获取详细的连接度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE
MON_GET_FCM_CONNECTION_LIST - 获取有关所有 FCM 连接的详细信息	ACTIVITY METRICS BASE
MON_GET_PKG_CACHE_STMT 表函数 - 获取程序包高速缓存中的 SQL 语句活动度量值	ACTIVITY METRICS BASE
MON_GET_PKG_CACHE_STMT_DETAILS 表函数 - 获取程序包高速缓存条目的详细度量值	ACTIVITY METRICS BASE
MON_GET_SERVICE_SUBCLASS 表函数 - 获取服务子类度量值	REQUEST METRICS BASE
MON_GET_SERVICE_SUBCLASS_DETAILS 表函数 - 获取详细的服务子类度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE
MON_GET_UNIT_OF_WORK 表函数 - 获取工作单元度量值	REQUEST METRICS BASE

表 665. 表函数监视信息 (续)

表函数	监视元素收集级别
MON_GET_UNIT_OF_WORK 表函数 - 获取详细的工作单元度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE
MON_GET_WORKLOAD 表函数 - 获取工作负载度量值	REQUEST METRICS BASE
MON_GET_WORKLOAD_DETAILS 表函数 - 获取详细的工作负载度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE

表 666. 事件监视信息

事件类型	逻辑数据分组	监视开关
活动	event_activitymetrics	ACTIVITY METRICS BASE
程序包高速缓存	在 activity_metrics 文档中报告。	ACTIVITY METRICS BASE

用法

使用此监视元素来确定使用 FCM 发送的数据总量，其中包括消息流量和表队列数据量。

fcm_send_wait_time -“FCM 发送等待时间”监视元素

执行 FCM 发送操作时被阻塞的时间。这包括等待内部请求的缓冲区被清仓时耗用的时间以及通过表队列发送数据时等待窗口计数确认时耗用的时间。此值以毫秒计。

表 667. 表函数监视信息

表函数	监视元素收集级别
MON_FORMAT_XML_METRICS_BY_ROW - 获取所有度量值的基于已格式化行的输出	不适用; 报告 XML 文档中作为格式化函数的输入提供的所有元素
MON_FORMAT_XML_TIMES_BY_ROW - - 获取基于已格式化行的组合层次结构等待时间和处理时间	不适用; 报告 XML 文档中作为格式化函数的输入提供的所有元素
MON_FORMAT_XML_WAIT_TIMES_BY _ROW - 获取等待时间的已格式化的基于行的输出	不适用; 报告 XML 文档中作为格式化函数的输入提供的所有元素
MON_GET_ACTIVITY_DETAILS 表函数 - 获取完整的活动详细信息（在 XML 文档 DETAILS 中报告）	ACTIVITY METRICS BASE
MON_GET_CONNECTION 表函数 - 获取连接度量值	REQUEST METRICS BASE
MON_GET_CONNECTION_DETAILS 表函数 - 获取详细的连接度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE
MON_GET_PKG_CACHE_STMT 表函数 - 获取程序包高速缓存中的 SQL 语句活动度量值	ACTIVITY METRICS BASE

表 667. 表函数监视信息 (续)

表函数	监视元素收集级别
MON_GET_PKG_CACHE_STMT_DETAILS 表函数 - 获取程序包高速缓存条目的详细度量值	ACTIVITY METRICS BASE
MON_GET_SERVICE_SUBCLASS 表函数 - 获取服务子类度量值	REQUEST METRICS BASE
MON_GET_SERVICE_SUBCLASS_DETAILS 表函数 - 获取详细的服务子类度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE
MON_GET_UNIT_OF_WORK 表函数 - 获取工作单元度量值	REQUEST METRICS BASE
MON_GET_UNIT_OF_WORK 表函数 - 获取详细的工作单元度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE
MON_GET_WORKLOAD 表函数 - 获取工作负载度量值	REQUEST METRICS BASE
MON_GET_WORKLOAD_DETAILS 表函数 - 获取详细的工作负载度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE

表 668. 事件监视信息

事件类型	逻辑数据分组	监视开关
活动	event_activity（在 details_xml 文档中报告）	ACTIVITY METRICS BASE
活动	event_activitymetrics	ACTIVITY METRICS BASE
统计信息	event_scstats（在 details_xml 文档中报告）	REQUEST METRICS BASE
统计信息	event_wlstats（在 details_xml 文档中报告）	REQUEST METRICS BASE
工作单元	在 system_metrics 文档中报告。	REQUEST METRICS BASE
程序包高速缓存	在 activity_metrics 文档中报告。	ACTIVITY METRICS BASE

用法

使用此元素来确定等待通过 FCM 发送数据时的耗用时间总计。这既包括发送请求消息时耗用的时间，也包括发送表队列数据时耗用的时间。

fcm_sends_total -“FCM 发送总计”监视元素

使用内部 FCM 通信层发送的缓冲区的总数。

表 669. 表函数监视信息

表函数	监视元素收集级别
MON_FORMAT_XML_METRICS_BY_ROW - 获取所有度量值的基于已格式化行的输出	不适用；报告 XML 文档中作为格式化函数的输入提供的所有元素

表 669. 表函数监视信息 (续)

表函数	监视元素收集级别
MON_GET_ACTIVITY_DETAILS 表函数 - 获取完整的活动详细信息（在 XML 文档 DETAILS 中报告）	ACTIVITY METRICS BASE
MON_GET_CONNECTION 表函数 - 获取连接度量值	REQUEST METRICS BASE
MON_GET_CONNECTION_DETAILS 表函数 - 获取详细的连接度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE
MON_GET_PKG_CACHE_STMT 表函数 - 获取程序包高速缓存中的 SQL 语句活动度量值	ACTIVITY METRICS BASE
MON_GET_PKG_CACHE_STMT_DETAILS 表函数 - 获取程序包高速缓存条目的详细度量值	ACTIVITY METRICS BASE
MON_GET_SERVICE_SUBCLASS 表函数 - 获取服务子类度量值	REQUEST METRICS BASE
MON_GET_SERVICE_SUBCLASS_DETAILS 表函数 - 获取详细的服务子类度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE
MON_GET_UNIT_OF_WORK 表函数 - 获取工作单元度量值	REQUEST METRICS BASE
MON_GET_UNIT_OF_WORK 表函数 - 获取详细的工作单元度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE
MON_GET_WORKLOAD 表函数 - 获取工作负载度量值	REQUEST METRICS BASE
MON_GET_WORKLOAD_DETAILS 表函数 - 获取详细的工作负载度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE

表 670. 事件监视信息

事件类型	逻辑数据分组	监视开关
活动	event_activity（在 details_xml 文档中报告）	ACTIVITY METRICS BASE
活动	event_activitymetrics	ACTIVITY METRICS BASE
统计信息	event_scstats（在 details_xml 文档中报告）	REQUEST METRICS BASE
统计信息	event_wlstats（在 details_xml 文档中报告）	REQUEST METRICS BASE
工作单元	在 system_metrics 文档中报告。	REQUEST METRICS BASE
程序包高速缓存	在 activity_metrics 文档中报告。	ACTIVITY METRICS BASE

用法

使用此元素来确定每个 FCM 接收操作的平均等待时间以及 FCM 接收操作所返回的平均数据量。

fcm_tq_recv_volume -“FCM 表队列接收量”监视元素

FCM 通信层通过表队列接收的数据量。此值以字节计。

表 671. 表函数监视信息

表函数	监视元素收集级别
MON_FORMAT_XML_METRICS_BY_ROW - 获取所有度量值的基于已格式化行的输出	不适用; 报告 XML 文档中作为格式化函数的输入提供的所有元素
MON_GET_ACTIVITY_DETAILS 表函数 - 获取完整的活动详细信息（在 XML 文档 DETAILS 中报告）	ACTIVITY METRICS BASE
MON_GET_CONNECTION 表函数 - 获取连接度量值	REQUEST METRICS BASE
MON_GET_CONNECTION_DETAILS 表函数 - 获取详细的连接度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE
MON_GET_FCM_CONNECTION_LIST - 获取有关所有 FCM 连接的详细信息	ACTIVITY METRICS BASE
MON_GET_PKG_CACHE_STMT 表函数 - 获取程序包高速缓存中的 SQL 语句活动度量值	ACTIVITY METRICS BASE
MON_GET_PKG_CACHE_STMT_DETAILS 表函数 - 获取程序包高速缓存条目的详细度量值	ACTIVITY METRICS BASE
MON_GET_SERVICE_SUBCLASS 表函数 - 获取服务子类度量值	REQUEST METRICS BASE
MON_GET_SERVICE_SUBCLASS_DETAILS 表函数 - 获取详细的服务子类度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE
MON_GET_UNIT_OF_WORK 表函数 - 获取工作单元度量值	REQUEST METRICS BASE
MON_GET_UNIT_OF_WORK 表函数 - 获取详细的工作单元度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE
MON_GET_WORKLOAD 表函数 - 获取工作负载度量值	REQUEST METRICS BASE
MON_GET_WORKLOAD_DETAILS 表函数 - 获取详细的工作负载度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE

表 672. 事件监视信息

事件类型	逻辑数据分组	监视开关
活动	event_activity（在 details_xml 文档中报告）	ACTIVITY METRICS BASE

表 672. 事件监视信息 (续)

事件类型	逻辑数据分组	监视开关
活动	event_activitymetrics	ACTIVITY METRICS BASE
统计信息	event_scstats（在 details_xml 文档中报告）	REQUEST METRICS BASE
统计信息	event_wlstats（在 details_xml 文档中报告）	REQUEST METRICS BASE
工作单元	在 system_metrics 文档中报告。	REQUEST METRICS BASE
程序包高速缓存	在 activity_metrics 文档中报告。	ACTIVITY METRICS BASE

用法

使用此监视元素来确定通过表队列接收的数据总量。

fcm_tq_recv_wait_time -“FCM 表队列接收等待时间”监视元素

等待从表队列中接收下一个缓冲区时耗用的时间。此值以毫秒计。

表 673. 表函数监视信息

表函数	监视元素收集级别
MON_FORMAT_XML_METRICS_BY_ROW - 获取所有度量值的基于已格式化行的输出	不适用; 报告 XML 文档中作为格式化函数的输入提供的所有元素
MON_FORMAT_XML_TIMES_BY_ROW - - 获取基于已格式化行的组合层次结构等待时间和处理时间	不适用; 报告 XML 文档中作为格式化函数的输入提供的所有元素
MON_FORMAT_XML_WAIT_TIMES_BY _ROW - 获取等待时间的已格式化的基于行的输出	不适用; 报告 XML 文档中作为格式化函数的输入提供的所有元素
MON_GET_ACTIVITY_DETAILS 表函数 - 获取完整的活动详细信息（在 XML 文档 DETAILS 中报告）	ACTIVITY METRICS BASE
MON_GET_CONNECTION 表函数 - 获取连接度量值	REQUEST METRICS BASE
MON_GET_CONNECTION_DETAILS 表函数 - 获取详细的连接度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE
MON_GET_PKG_CACHE_STMT 表函数 - 获取程序包高速缓存中的 SQL 语句活动度量值	ACTIVITY METRICS BASE
MON_GET_PKG_CACHE_STMT_DETAILS 表函数 - 获取程序包高速缓存条目的详细度量值	ACTIVITY METRICS BASE
MON_GET_SERVICE_SUBCLASS 表函数 - 获取服务子类度量值	REQUEST METRICS BASE
MON_GET_SERVICE_SUBCLASS_DETAILS 表函数 - 获取详细的服务子类度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE

表 673. 表函数监视信息 (续)

表函数	监视元素收集级别
MON_GET_UNIT_OF_WORK 表函数 - 获取工作单元度量值	REQUEST METRICS BASE
MON_GET_UNIT_OF_WORK 表函数 - 获取详细的工作单元度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE
MON_GET_WORKLOAD 表函数 - 获取工作负载度量值	REQUEST METRICS BASE
MON_GET_WORKLOAD_DETAILS 表函数 - 获取详细的工作负载度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE

表 674. 事件监视信息

事件类型	逻辑数据分组	监视开关
活动	event_activity（在 details_xml 文档中报告）	ACTIVITY METRICS BASE
活动	event_activitymetrics	ACTIVITY METRICS BASE
统计信息	event_scstats（在 details_xml 文档中报告）	REQUEST METRICS BASE
统计信息	event_wlstats（在 details_xml 文档中报告）	REQUEST METRICS BASE
工作单元	在 system_metrics 文档中报告。	REQUEST METRICS BASE
程序包高速缓存	在 activity_metrics 文档中报告。	ACTIVITY METRICS BASE

用法

使用此元素来确定代理程序等待通过表队列接收数据时耗用的时间。

fcm_tq_recvs_total -“FCM 表队列接收总量”监视元素

使用内部 FCM 通信机制从表队列接收的缓冲区的总数。

表 675. 表函数监视信息

表函数	监视元素收集级别
MON_FORMAT_XML_METRICS_BY_ROW - 获取所有度量值的基于已格式化行的输出	不适用; 报告 XML 文档中作为格式化函数的输入提供的所有元素
MON_GET_ACTIVITY_DETAILS 表函数 - 获取完整的活动详细信息（在 XML 文档 DETAILS 中报告）	ACTIVITY METRICS BASE
MON_GET_CONNECTION 表函数 - 获取连接度量值	REQUEST METRICS BASE
MON_GET_CONNECTION_DETAILS 表函数 - 获取详细的连接度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE

表 675. 表函数监视信息 (续)

表函数	监视元素收集级别
MON_GET_PKG_CACHE_STMT 表函数 - 获取程序包高速缓存中的 SQL 语句活动度量值	ACTIVITY METRICS BASE
MON_GET_PKG_CACHE_STMT_DETAILS 表函数 - 获取程序包高速缓存条目的详细度量值	ACTIVITY METRICS BASE
MON_GET_SERVICE_SUBCLASS 表函数 - 获取服务子类度量值	REQUEST METRICS BASE
MON_GET_SERVICE_SUBCLASS_DETAILS 表函数 - 获取详细的服务子类度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE
MON_GET_UNIT_OF_WORK 表函数 - 获取工作单元度量值	REQUEST METRICS BASE
MON_GET_UNIT_OF_WORK 表函数 - 获取详细的工作单元度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE
MON_GET_WORKLOAD 表函数 - 获取工作负载度量值	REQUEST METRICS BASE
MON_GET_WORKLOAD_DETAILS 表函数 - 获取详细的工作负载度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE

表 676. 事件监视信息

事件类型	逻辑数据分组	监视开关
活动	event_activity（在 details_xml 文档中报告）	ACTIVITY METRICS BASE
活动	event_activitymetrics	ACTIVITY METRICS BASE
统计信息	event_scstats（在 details_xml 文档中报告）	REQUEST METRICS BASE
统计信息	event_wlstats（在 details_xml 文档中报告）	REQUEST METRICS BASE
工作单元	在 system_metrics 文档中报告。	REQUEST METRICS BASE
程序包高速缓存	在 activity_metrics 文档中报告。	ACTIVITY METRICS BASE

用法

通过将此元素与 **fcm_tq_recv_volume** 和 **fcm_tq_recv_wait_time** 配合使用，可以确定接收每个表队列缓冲区时的平均等待时间和接收数据量。

fcm_tq_send_volume -“FCM 表队列发送量”监视元素

FCM 通信层通过表队列发送的数据量。此值以字节计。

表 677. 表函数监视信息

表函数	监视元素收集级别
MON_FORMAT_XML_METRICS_BY_ROW - 获取所有度量值的基于已格式化行的输出	不适用; 报告 XML 文档中作为格式化函数的输入提供的所有元素
MON_GET_ACTIVITY_DETAILS 表函数 - 获取完整的活动详细信息（在 XML 文档 DETAILS 中报告）	ACTIVITY METRICS BASE
MON_GET_CONNECTION 表函数 - 获取连接度量值	REQUEST METRICS BASE
MON_GET_CONNECTION_DETAILS 表函数 - 获取详细的连接度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE
MON_GET_FCM_CONNECTION_LIST - 获取有关所有 FCM 连接的详细信息	ACTIVITY METRICS BASE
MON_GET_PKG_CACHE_STMT 表函数 - 获取程序包高速缓存中的 SQL 语句活动度量值	ACTIVITY METRICS BASE
MON_GET_PKG_CACHE_STMT_DETAILS 表函数 - 获取程序包高速缓存条目的详细度量值	ACTIVITY METRICS BASE
MON_GET_SERVICE_SUBCLASS 表函数 - 获取服务子类度量值	REQUEST METRICS BASE
MON_GET_SERVICE_SUBCLASS_DETAILS 表函数 - 获取详细的服务子类度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE
MON_GET_UNIT_OF_WORK 表函数 - 获取工作单元度量值	REQUEST METRICS BASE
MON_GET_UNIT_OF_WORK 表函数 - 获取详细的工作单元度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE
MON_GET_WORKLOAD 表函数 - 获取工作负载度量值	REQUEST METRICS BASE
MON_GET_WORKLOAD_DETAILS 表函数 - 获取详细的工作负载度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE

表 678. 事件监视信息

事件类型	逻辑数据分组	监视开关
活动	event_activity（在 details_xml 文档中报告）	ACTIVITY METRICS BASE
活动	event_activitymetrics	ACTIVITY METRICS BASE
统计信息	event_scstats（在 details_xml 文档中报告）	REQUEST METRICS BASE
统计信息	event_wlstats（在 details_xml 文档中报告）	REQUEST METRICS BASE

表 678. 事件监视信息 (续)

事件类型	逻辑数据分组	监视开关
工作单元	在 system_metrics 文档中报告。	REQUEST METRICS BASE
程序包高速缓存	在 activity_metrics 文档中报告。	ACTIVITY METRICS BASE

用法

使用此监视元素来确定使用 FCM 通过表队列缓冲区发送的数据总量。

fcm_tq_send_wait_time -“FCM 表队列发送等待时间”监视元素

等待通过表队列发送下一个缓冲区时耗用的时间。此元素反映等待来自表队列接收端的窗口计数确认时耗用的时间。此值以毫秒计。

表 679. 表函数监视信息

表函数	监视元素收集级别
MON_FORMAT_XML_METRICS_BY_ROW - 获取所有度量值的基于已格式化行的输出	不适用; 报告 XML 文档中作为格式化函数的输入提供的所有元素
MON_FORMAT_XML_TIMES_BY_ROW - - 获取基于已格式化行的组合层次结构等待时间和处理时间	不适用; 报告 XML 文档中作为格式化函数的输入提供的所有元素
MON_FORMAT_XML_WAIT_TIMES_BY _ROW - 获取等待时间的已格式化的基于行的输出	不适用; 报告 XML 文档中作为格式化函数的输入提供的所有元素
MON_GET_ACTIVITY_DETAILS 表函数 - 获取完整的活动详细信息（在 XML 文档 DETAILS 中报告）	ACTIVITY METRICS BASE
MON_GET_CONNECTION 表函数 - 获取连接度量值	REQUEST METRICS BASE
MON_GET_CONNECTION_DETAILS 表函数 - 获取详细的连接度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE
MON_GET_PKG_CACHE_STMT 表函数 - 获取程序包高速缓存中的 SQL 语句活动度量值	ACTIVITY METRICS BASE
MON_GET_PKG_CACHE_STMT_DETAILS 表函数 - 获取程序包高速缓存条目的详细度量值	ACTIVITY METRICS BASE
MON_GET_SERVICE_SUBCLASS 表函数 - 获取服务子类度量值	REQUEST METRICS BASE
MON_GET_SERVICE_SUBCLASS_DETAILS 表函数 - 获取详细的服务子类度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE
MON_GET_UNIT_OF_WORK 表函数 - 获取工作单元度量值	REQUEST METRICS BASE
MON_GET_UNIT_OF_WORK 表函数 - 获取详细的工作单元度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE

表 679. 表函数监视信息 (续)

表函数	监视元素收集级别
MON_GET_WORKLOAD 表函数 - 获取工作负载度量值	REQUEST METRICS BASE
MON_GET_WORKLOAD_DETAILS 表函数 - 获取详细的工作负载度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE

表 680. 事件监视信息

事件类型	逻辑数据分组	监视开关
活动	event_activity（在 details_xml 文档中报告）	ACTIVITY METRICS BASE
活动	event_activitymetrics	ACTIVITY METRICS BASE
统计信息	event_scstats（在 details_xml 文档中报告）	REQUEST METRICS BASE
统计信息	event_wlstats（在 details_xml 文档中报告）	REQUEST METRICS BASE
工作单元	在 system_metrics 文档中报告。	REQUEST METRICS BASE
程序包高速缓存	在 activity_metrics 文档中报告。	ACTIVITY METRICS BASE

用法

使用此监视元素来确定等待使用 FCM 通过表队列发送数据缓冲区时耗用的时间。

fcm_tq_sends_total -“FCM 表队列发送总次数”监视元素

使用内部 FCM 通信机制发送的缓冲区（包含表队列数据）的总数。

表 681. 表函数监视信息

表函数	监视元素收集级别
MON_FORMAT_XML_METRICS_BY_ROW - 获取所有度量值的基于已格式化行的输出	不适用; 报告 XML 文档中作为格式化函数的输入提供的所有元素
MON_GET_ACTIVITY_DETAILS 表函数 - 获取完整的活动详细信息（在 XML 文档 DETAILS 中报告）	ACTIVITY METRICS BASE
MON_GET_CONNECTION 表函数 - 获取连接度量值	REQUEST METRICS BASE
MON_GET_CONNECTION_DETAILS 表函数 - 获取详细的连接度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE
MON_GET_PKG_CACHE_STMT 表函数 - 获取程序包高速缓存中的 SQL 语句活动度量值	ACTIVITY METRICS BASE
MON_GET_PKG_CACHE_STMT_DETAILS 表函数 - 获取程序包高速缓存条目的详细度量值	ACTIVITY METRICS BASE

表 681. 表函数监视信息 (续)

表函数	监视元素收集级别
MON_GET_SERVICE_SUBCLASS 表函数 - 获取服务子类度量值	REQUEST METRICS BASE
MON_GET_SERVICE_SUBCLASS_DETAILS 表函数 - 获取详细的服务子类度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE
MON_GET_UNIT_OF_WORK 表函数 - 获取工作单元度量值	REQUEST METRICS BASE
MON_GET_UNIT_OF_WORK 表函数 - 获取详细的工作单元度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE
MON_GET_WORKLOAD 表函数 - 获取工作负载度量值	REQUEST METRICS BASE
MON_GET_WORKLOAD_DETAILS 表函数 - 获取详细的工作负载度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE

表 682. 事件监视信息

事件类型	逻辑数据分组	监视开关
活动	event_activity（在 details_xml 文档中报告）	ACTIVITY METRICS BASE
活动	event_activitymetrics	ACTIVITY METRICS BASE
统计信息	event_scstats（在 details_xml 文档中报告）	REQUEST METRICS BASE
统计信息	event_wlstats（在 details_xml 文档中报告）	REQUEST METRICS BASE
工作单元	在 system_metrics 文档中报告。	REQUEST METRICS BASE
程序包高速缓存	在 activity_metrics 文档中报告。	ACTIVITY METRICS BASE

用法

通过将此元素与 **fcm_tq_send_volume** 和 **fcm_tq_send_wait_time** 监视元素配合使用，可以确定使用表队列发送的缓冲区的平均数据量和平均等待时间。

fetch_count -“成功的访存数”

成功的物理访存数或尝试的物理访存数，取决于快照监视级别。

- 对于 stmt 和 dynsql 快照监视级别和语句事件类型：对特定游标执行的成功访存数。
- 对于 dcs_stmt 快照监视级别：在语句执行期间尝试的物理访存数（不管应用程序访存的行数是多少）。在这种情况下，**fetch_count** 表示处理语句时服务器需要将应答数据发送回网关的次数。

表 683. 快照监视信息

快照级别	逻辑数据分组	监视开关
应用程序	stmt	语句
DCS 语句	dcs_stmt	语句
动态 SQL	dynsql	语句

对于动态 SQL 快照监视，可以重置此计数器。

表 684. 事件监视信息

事件类型	逻辑数据分组	监视开关
语句	event_stmt	始终收集

用法

可使用此元素来了解数据库管理器内的当前活动级别。

由于性能原因，语句事件监视器不会对每个 FETCH 语句生成语句事件记录。仅当 FETCH 返回非零 SQLCODE 时，才生成记录事件。

files_closed -“关闭数据库文件数”监视元素

关闭的数据库文件的总数。

表 685. 表函数监视信息

表函数	监视元素收集命令和级别
MON_GET_BUFFERPOOL 表函数 - 获取缓冲池度量值	DATA OBJECT METRICS BASE
MON_GET_TABLESPACE 表函数 - 获取表空间度量值	DATA OBJECT METRICS BASE

表 686. 快照监视信息

快照级别	逻辑数据分组	监视开关
数据库	dbase	缓冲池
表空间	tablespace	缓冲池
缓冲池	bufferpool	缓冲池

可将快照监视的计数器重置。

表 687. 事件监视信息

事件类型	逻辑数据分组	监视开关
数据库	event_db	始终收集
表空间	event_tablespace	始终收集

用法

数据库管理器打开文件以便读入缓冲池和写出缓冲池。任何时间应用程序打开的最大数据库文件数都由 **maxfilop** 配置参数控制。如果达到最大文件数，那么在打开新文件之前必须先关闭某个文件。注意，实际的打开文件数可能不等于关闭文件数。

您可以使用此元素来帮助确定最佳的 **maxfilop** 配置参数值。

first_active_log -“第一个活动日志文件编号”

第一个活动日志文件的文件号。

元素标识

first_active_log

元素类型

信息

表 688. 表函数监视信息

表函数	监视元素收集级别
MON_GET_TRANSACTION_LOG 表函数 - 获取日志信息	ACTIVITY METRICS BASE

表 689. 快照监视信息

快照级别	逻辑数据分组	监视开关
数据库	detail_log	基本

表 690. 事件监视信息

事件类型	逻辑数据分组	监视开关
数据库	event_db	始终收集

用法 将此元素与 *last_active_log* 和 *current_active_log* 元素一起使用来确定活动日志文件的范围。知道活动日志文件的范围可帮助您确定日志文件所需的磁盘空间。

您也可以使用此元素来确定哪些日志文件包含数据，以帮助您标识分割镜像支持所需的日志文件。

first_overflow_time -“第一次事件溢出时间”

此溢出记录的第一次溢出的日期和时间。

表 691. 事件监视信息

事件类型	逻辑数据分组	监视开关
溢出记录	event_overflow	-

用法 将此元素与 *last_over_flow time* 配合使用来计算生成溢出记录所耗用的时间。

fs_caching -“文件系统高速缓存”监视元素

指示特定表空间是否使用文件系统高速缓存。如果 **fs_caching** 为 0，那么表明已启用文件系统高速缓存。如果 **fs_caching** 为 1，那么表明已将文件系统高速缓存禁用。

表 692. 表函数监视信息

表函数	监视元素收集命令和级别
MON_GET_TABLESPACE 表函数 - 获取表空间度量值	ACTIVITY METRICS BASE

表 693. 快照监视信息

快照级别	逻辑数据分组	监视开关
表空间	tablespace	基本

表 694. 事件监视信息

事件类型	逻辑数据分组	监视开关
表空间	event_tablespace	-

fs_id -“唯一文件系统标识号”监视元素

此元素指示操作系统为存储器路径或容器所指向的文件系统提供的唯一标识号。

表 695. 表函数监视信息

表函数	监视元素收集级别
ADMIN_GET_STORAGE_PATHS 表函数 - 获取存储器组的存储器路径信息	ACTIVITY METRICS BASE
MON_GET_CONTAINER 表函数 - 获取表空间容器度量值	ACTIVITY METRICS BASE

表 696. 快照监视信息

快照级别	逻辑数据分组	监视开关
数据库	db_sto_path_info	缓冲池

用法

通过将此元素与下列元素配合使用，可以收集数据库的空间利用率数据:

- **db_storage_path**
- **sto_path_free_sz**
- **fs_used_size**
- **fs_total_size**

fs_total_size -“文件系统总大小”监视元素

此元素指示存储器路径或容器所指向的文件系统的容量（以字节为单位）。

表 697. 表函数监视信息

表函数	监视元素收集级别
ADMIN_GET_STORAGE_PATHS 表函数 - 获取存储器组的存储器路径信息	ACTIVITY METRICS BASE
MON_GET_CONTAINER 表函数 - 获取表空间容器度量值	ACTIVITY METRICS BASE

表 698. 快照监视信息

快照级别	逻辑数据分组	监视开关
数据库	db_sto_path_info	缓冲池

用法

可以将此元素与下列元素配合使用以收集数据库空间利用率数据:

- **db_storage_path**
- **sto_path_free_sz**
- **fs_used_size**
- **fs_id**

fs_used_size -“文件系统中的已用空间量”监视元素

此元素指示存储器路径或容器所指向的文件系统中的已用空间量（以字节为单位）。

表 699. 表函数监视信息

表函数	监视元素收集级别
ADMIN_GET_STORAGE_PATHS 表函数 - 获取存储器组的存储器路径信息	ACTIVITY METRICS BASE
MON_GET_CONTAINER 表函数 - 获取表空间容器度量值	ACTIVITY METRICS BASE

表 700. 快照监视信息

快照级别	逻辑数据分组	监视开关
数据库	db_sto_path_info	缓冲池

用法

可以将此元素与下列元素配合使用以收集数据库空间利用率数据:

- **db_storage_path**
- **sto_path_free_sz**
- **fs_total_size**
- **fs_id**

global_transaction_id -“全局事务标识”监视元素

事件发生时正使用的 XA 事务标识。

表 701. 事件监视信息

事件类型	逻辑数据分组	监视开关
变更历史记录	ddlstmtexec txncompletion	始终收集
工作单元	uow	始终收集

用法

对于变更历史记录事件监视器，这是事件发生时正使用的全局事务标识。这是作为事务日志一部分的 SQLP_GXID 结构中的数据字段。

gw_comm_error_time -“通信错误时间”

DCS 应用程序尝试连接至主机数据库时或处理 SQL 语句时发生最新通信错误（SQL30081）的日期和时间。

表 702. 快照监视信息

快照级别	逻辑数据分组	监视开关
DCS 数据库	dcs_dbase	时间戳记

用法

将此元素与通信错误及管理通知日志中记录的通信错误一起使用来确定问题。

gw_comm_errors -“通信错误”

DCS 应用程序尝试连接至主机数据库时或处理 SQL 语句时发生通信错误（SQL30081）的次数。

表 703. 快照监视信息

快照级别	逻辑数据分组	监视开关
DCS 数据库	dcs_dbase	基本

可将快照监视的计数器重置。

用法 通过监视一段时间内发生通信错误的次数，可评估 DB2 Connect 网关与特定主机数据库的连接是否存在问题。可设置您希望的正常错误阈值，这样每当错误数超过此阈值时都会进行通信错误调查。

将此元素与管理通知日志中记录的通信错误一起使用来确定问题。

gw_con_time -“DB2 Connect 网关首次启动的连接”

从 DB2 Connect 网关首次启动与主机数据库的连接的日期和时间。

表 704. 快照监视信息

快照级别	逻辑数据分组	监视开关
DCS 数据库	dcs_dbase	时间戳记
DCS 应用程序	dcs_appl	时间戳记

用法 此元素用于 DCS 应用程序的问题确定。

gw_connections_top -“与主机数据库的最大并行连接数”

自首次数据库连接后由 DB2 Connect 网关处理的与主机数据库的并行连接的最大数目。

元素标识

gw_connections_top

元素类型

水位标记

表 705. 快照监视信息

快照级别	逻辑数据分组	监视开关
DCS 数据库	dcs_dbase	基本

用法 此元素帮助您了解 DB2 Connect 网关上的活动级别及系统资源的关联使用。

gw_cons_wait_client -“等待客户机发送请求的连接数”

由 DB2 Connect 网关处理并且等待客户机发送请求的与主机数据库的当前连接的数目。

元素标识

gw_cons_wait_client

元素类型

标尺

表 706. 快照监视信息

快照级别	逻辑数据分组	监视开关
数据库管理器	db2	基本
DCS 数据库	dcs_dbase	基本

用法 此值可能经常更改。在延长时间段内应按一定时间间隔对其进行采样以了解真实的网关使用情况。

gw_cons_wait_host -“等待主机应答的连接数”

由 DB2 Connect 网关处理并且等待主机应答的与主机数据库的当前连接的数目。

元素标识

gw_cons_wait_host

元素类型

标尺

表 707. 快照监视信息

快照级别	逻辑数据分组	监视开关
数据库管理器	db2	基本
DCS 数据库	dcs_dbase	基本

用法 此值可能经常更改。在延长时间段内应按一定时间间隔对其进行采样以了解真实的网关使用情况。

gw_cur_cons -“DB2 Connect 的当前连接数”

由 DB2 Connect 网关处理的与主机数据库的当前连接的数目。

表 708. 快照监视信息

快照级别	逻辑数据分组	监视开关
数据库管理器	db2	基本
DCS 数据库	dcs_dbase	基本

用法 此元素帮助您了解 DB2 Connect 网关上的活动级别及系统资源的关联使用。

gw_db_alias -“网关上的数据库别名”

DB2 Connect 网关上用来连接至主机数据库的别名。

元素标识

gw_db_alias

元素类型

信息

表 709. 快照监视信息

快照级别	逻辑数据分组	监视开关
DCS 应用程序	dcs_appl_info	基本

用法 此元素用于 DCS 应用程序的问题确定。

gw_exec_time -“DB2 Connect 网关处理所耗用的时间”

DB2 Connect 网关处理应用程序请求（自建立连接后）或处理单个语句所花的时间（以秒和微秒计）。

表 710. 快照监视信息

快照级别	逻辑数据分组	监视开关
DCS 应用程序	dcs_appl	语句，时间戳记
DCS 语句	dcs_stmt	语句，时间戳记

可将快照监视的计数器重置。

用法 使用此元素来确定整体处理时间的哪一部分用于 DB2 Connect 网关处理。

gw_total_cons -“对 DB2 Connect 尝试连接的总数”

自最后一次 db2start 命令或最后一次重置后尝试从 DB2 Connect 网关进行连接的总次数。

表 711. 快照监视信息

快照级别	逻辑数据分组	监视开关
数据库管理器	db2	基本
DCS 数据库	dcs_dbase	基本

可将快照监视的计数器重置。

用法 此元素帮助您了解 DB2 Connect 网关上的活动级别及系统资源的关联使用。

hadr_connect_status -“HADR 连接状态”监视元素

数据库的高可用性灾难恢复 (HADR) 连接状态。

表 712. 表函数监视信息

表函数	监视元素收集级别
MON_GET_HADR 表函数 - 返回高可用性灾难恢复 (HADR) 监视信息	ACTIVITY METRICS BASE

表 713. 快照监视信息

快照级别	逻辑数据分组	监视开关
数据库	hadr	基本

用法

使用此元素来确定数据库的 HADR 连接状态。

此元素的数据类型是整型。

如果数据库具有 HADR 主角色或备用角色，那么此元素的值是下列其中一个常量:

SQLM_HADR_CONN_CONNECTED
数据库连接至其伙伴节点。

SQLM_HADR_CONN_DISCONNECTED
数据库未连接至其伙伴节点。

SQLM_HADR_CONN_CONGESTED
数据库连接至其伙伴节点，但连接拥塞。当主数据库与备用数据库之间的 TCP/IP 套接字连接仍然活动但一端不能将信息发送至另一端时连接拥塞。例如，接收端未从套接字连接接收，导致 TCP/IP 发送空间变满。网络连接拥塞的原因包括下列几项:

- 网络被太多资源共享，或者网络相对于主 HADR 节点的事务量而言不够快。

- 备用 HADR 节点所在的服务器处理能力不足，无法以必要的速率检索通信子系统中的信息。

要点: V10.1 中已经不推荐使用此监视元素，在以后的发行版中可能会将其除去。有关更多信息，请参阅 《*DB2 V10.1 新增内容*》中的『不推荐使用 HADR 的某些监视接口』。

如果数据库的 HADR 角色是标准的，那么应该忽略此元素。使用 **hadr_role** 监视元素来确定数据库的 HADR 角色。

hadr_connect_time -"HADR 连接时间"监视元素

此监视元素可返回下列值中的一个: 高可用性灾难恢复 (HADR) 连接时间、HADR 拥塞时间或 HADR 断开连接时间。

表 714. 快照监视信息

快照级别	逻辑数据分组	监视开关
数据库	hadr	基本

用法

使用此元素来确定当前 HADR 连接状态开始的时间。

如果数据库为 HADR 主数据库或备用数据库，那么此元素的含义取决于 **hadr_connect_status** 元素的值:

- 如果 **hadr_connect_status** 元素的值为 SQLM_HADR_CONN_CONNECTED，那么此元素显示连接时间。
- 如果 **hadr_connect_status** 元素的值为 SQLM_HADR_CONN_CONGESTED，那么此元素显示拥塞开始的时间。
- 如果 **hadr_connect_status** 元素的值为 SQLM_HADR_CONN_DISCONNECTED，那么此元素显示断开连接时间。

如果自 HADR 引擎可拆离单元 (EDU) 启动后没有任何连接，那么会将连接状态报告为"已断开连接"并且 HADR EDU 启动时间将用于断开连接时间。因为 HADR 连接和断开连接事件相对少见，所以即使 DFT_MON_TIMESTAMP 开关为 OFF，也会收集并报告该时间。

要点: V10.1 中已经不推荐使用此监视元素，在以后的发行版中可能会将其除去。有关更多信息，请参阅 《*DB2 V10.1 新增内容*》中的『不推荐使用 HADR 的某些监视接口』。

如果数据库的 HADR 角色是标准的，那么应该忽略此元素。使用 **hadr_role** 监视元素来确定数据库的 HADR 角色。

hadr_heartbeat -"HADR 脉动信号"监视元素

在高可用性灾难恢复 HADR 连接上连续丢失的脉动信号数。

数据库再次接收到脉动信号时，此数字将重置为零。如果数据库具有 HADR 主角色或备用角色，那么此元素指示 HADR 连接的运行状况。

表 715. 快照监视信息

快照级别	逻辑数据分组	监视开关
数据库	hadr	基本

对于快照监视来说，无法重置此计数器。

用法说明:

使用此元素来确定 HADR 连接的运行状况。

脉动信号是以固定时间间隔从其他 HADR 数据库发送的消息。如果此元素的值为零，那么表明未丢失脉动信号，并且连接的运行状况正常。此值越大，连接的运行状况就越差。

在断开连接方式下，因为丢失的脉动信号不适用，所以它始终显示为 0。

脉动信号间隔派生自配置参数（例如，**hadr_timeout** 和 **hadr_peer_window**），最大设置为 30 秒。

此元素的数据类型是整型。

要点: V10.1 中已经不推荐使用此监视元素，在以后的发行版中可能会将其除去。有关更多信息，请参阅 《*DB2 V10.1 新增内容*》中的『不推荐使用 HADR 的某些监视接口』。

如果数据库的 HADR 角色是标准数据库，请忽略此元素。使用 **hadr_role** 监视元素来确定数据库的 HADR 角色。

hadr_local_host -“HADR 本地主机”监视元素

本地高可用性灾难恢复 (HADR) 主机名。该值显示为主机名字符串或 IP 地址字符串，如“1.2.3.4”。

表 716. 快照监视信息

快照级别	逻辑数据分组	监视开关
数据库	hadr	基本

用法

使用此元素来确定有效 HADR 本地主机名称。HADR 数据库配置参数是静态的。停止并重新启动数据库之后，对参数所作的更改才会生效。此监视元素报告 HADR 系统实际使用的值，而不是数据库配置文件中的值。

对此元素的更改将在数据库激活时生效，或者，如果数据库已联机，那么在 HADR 在主数据库上停止然后重新启动后生效。

注: 使用的任何名称都必须解析为一个 IP 地址。尝试启动 HADR 时，解析为多个地址的名称将导致错误。

要点: V10.1 中已经不推荐使用此监视元素，在以后的发行版中可能会将其除去。有关更多信息，请参阅 《*DB2 V10.1 新增内容*》中的『不推荐使用 HADR 的某些监视接口』。

如果数据库的 HADR 角色是标准的，那么应该忽略此元素。使用 **hadr_role** 监视元素来确定数据库的 HADR 角色。

hadr_local_service -"HADR 本地服务"监视元素

本地 HADR TCP 服务。此值将显示为服务名称字符串或端口号字符串。

表 717. 快照监视信息

快照级别	逻辑数据分组	监视开关
数据库	hadr	基本

用法

使用此元素来确定有效 HADR 本地服务名称。

对此元素的更改将在数据库激活时生效，或者，如果数据库已联机，那么在 HADR 在主数据库上停止然后重新启动后生效。

要点: V10.1 中已经不推荐使用此监视元素，在以后的发行版中可能会将其除去。有关更多信息，请参阅 《*DB2 V10.1 新增内容*》中的『不推荐使用 HADR 的某些监视接口』。

如果数据库的 HADR 角色是标准的，那么应该忽略此元素。使用 **hadr_role** 监视元素来确定数据库的 HADR 角色。

hadr_log_gap -"HADR 日志间隔"

此元素显示主日志序号（LSN）与备用日志 LSN 之间正在运行的平均间隔。该间隔是以字节数度量的。

表 718. 表函数监视信息

表函数	监视元素收集级别
MON_GET_HADR 表函数 - 返回高可用性灾难恢复 (HADR) 监视信息	ACTIVITY METRICS BASE

表 719. 快照监视信息

快照级别	逻辑数据分组	监视开关
数据库	hadr	基本

用法

使用此元素来确定主 HADR 数据库日志与备用 HADR 数据库日志之间的间隔。

当日志文件被截断时，下一个日志文件中的 LSN 将会开始，就好像上一个文件未被截断一样。此 LSN 洞口未包含任何日志数据。这样的洞口可能导致日志间隔不反映主 HADR 数据库日志与备用 HADR 数据库日志之间的实际日志差别。

要点: V10.1 中已经不推荐使用此监视元素，在以后的发行版中可能会将其除去。有关更多信息，请参阅 《*DB2 V10.1 新增内容*》中的『不推荐使用 HADR 的某些监视接口』。

如果数据库的 HADR 角色是标准的，那么应该忽略此元素。使用 **hadr_role** 监视元素来确定数据库的 HADR 角色。

hadr_peer_window -“HADR 对等窗口”监视元素

HADR_PEER_WINDOW 数据库配置参数的值。

表 720. 快照监视信息

快照级别	逻辑数据分组	监视开关
数据库	hadr	基本

用法

使用此元素来确定 HADR_PEER_WINDOW 数据库配置参数的值。

hadr_peer_window_end -“HADR 对等时间结束”监视元素

只要主数据库处于活动状态，在高可用性灾难恢复 (HADR) 主数据库承诺将时间点停留在对等或断开对等状态之前的时间点。

表 721. 快照监视信息

快照级别	逻辑数据分组	监视开关
数据库	hadr	基本

用法

使用此元素来确定在主数据库承诺将时间点停留在对等或断开对等状态之前的时间点。

主数据库报告的值可能与备用数据库报告的值不同。发生此情况的原因是：主数据库在发送脉动信号消息时对值进行了更新，而新值仅在备用数据库接收并处理消息之后才显示在备用数据库上。

如果数据库脱离对等或断开对等状态，那么此监视元素的值不会被重置。将保留并返回最后知道的值。如果数据库从未达到对等状态，那么将返回零值。

对等时间结束时间由主数据库设置，然后发送至备用数据库。因此，对等时间结束值将基于主数据库时钟。当您将对等时间结束时间与主数据库停机时间进行比较时，如果两个时钟未同步良好，那么您可能需要添加偏移量以将时间戳记转换为主数据库时钟。

要点: V10.1 中已经不推荐使用此监视元素，在以后的发行版中可能会将其除去。有关更多信息，请参阅 《*DB2 V10.1 新增内容*》中的『不推荐使用 HADR 的某些监视接口』。

hadr_primary_log_file -"HADR 主日志文件"监视元素

当前日志文件在主 HADR 数据库上的名称。

表 722. 快照监视信息

快照级别	逻辑数据分组	监视开关
数据库	hadr	基本

用法

使用此元素来确定主 HADR 数据库上的当前日志文件。

要点: V10.1 中已经不推荐使用此监视元素，在以后的发行版中可能会将其除去。有关更多信息，请参阅 《*DB2 V10.1 新增内容*》中的『不推荐使用 HADR 的某些监视接口』。

如果数据库的 HADR 角色是标准的，那么应该忽略此元素。使用 **hadr_role** 监视元素来确定数据库的 HADR 角色。

hadr_primary_log_lsn -"HADR 主日志 LSN"监视元素

主 HADR 数据库的当前日志位置。日志序号（LSN）是数据库的日志流中的字节位移。

表 723. 快照监视信息

快照级别	逻辑数据分组	监视开关
数据库	hadr	基本

用法

使用此元素来确定主 HADR 数据库上的当前日志位置。

要点: V10.1 中已经不推荐使用此监视元素，在以后的发行版中可能会将其除去。有关更多信息，请参阅 《*DB2 V10.1 新增内容*》中的『不推荐使用 HADR 的某些监视接口』。

如果数据库的 HADR 角色是标准的，那么应该忽略此元素。使用 **hadr_role** 监视元素来确定数据库的 HADR 角色。

hadr_primary_log_page -“HADR 主日志页”监视元素

当前日志文件中的页号，指示当前日志在主 HADR 数据库上的位置。页号与日志文件相关。例如，页零是文件的开头。

表 724. 快照监视信息

快照级别	逻辑数据分组	监视开关
数据库	hadr	基本

用法

使用此元素来确定主 HADR 数据库上的当前日志页。

要点: V10.1 中已经不推荐使用此监视元素，在以后的发行版中可能会将其除去。有关更多信息，请参阅 《*DB2 V10.1 新增内容*》中的『不推荐使用 HADR 的某些监视接口』。

如果数据库的 HADR 角色是标准的，那么应该忽略此元素。使用 **hadr_role** 监视元素来确定数据库的 HADR 角色。

hadr_remote_host -“HADR 远程主机”监视元素

远程高可用性灾难恢复 (HADR) 主机名。该值显示为主机名字符串或 IP 地址字符串，如“1.2.3.4”。

表 725. 快照监视信息

快照级别	逻辑数据分组	监视开关
数据库	hadr	基本

用法

使用此元素来确定有效 HADR 远程主机名称。

对此元素的更改将在数据库激活时生效，或者，如果数据库已联机，那么在 HADR 在主数据库上停止然后重新启动后生效。

注: 使用的任何名称都必须解析为一个 IP 地址。尝试启动 HADR 时，解析为多个地址的名称将导致错误。

要点: V10.1 中已经不推荐使用此监视元素，在以后的发行版中可能会将其除去。有关更多信息，请参阅 《*DB2 V10.1 新增内容*》中的『不推荐使用 HADR 的某些监视接口』。

如果数据库的 HADR 角色是标准的，那么应该忽略此元素。使用 *hadr_role* 监视元素来确定数据库的 HADR 角色。

hadr_remote_instance -“HADR 远程实例”监视元素

远程 HADR 实例名。

表 726. 快照监视信息

快照级别	逻辑数据分组	监视开关
数据库	hadr	基本

用法

使用此元素来确定有效 HADR 远程实例名。

对此元素的更改将在数据库激活时生效，或者，如果数据库已联机，那么在 HADR 在主数据库上停止然后重新启动后生效。

要点：V10.1 中已经不推荐使用此监视元素，在以后的发行版中可能会将其除去。有关更多信息，请参阅 《*DB2 V10.1 新增内容*》中的『不推荐使用 HADR 的某些监视接口』。

如果数据库的 HADR 角色是标准的，那么应该忽略此元素。使用 **hadr_role** 监视元素来确定数据库的 HADR 角色。

hadr_remote_service -“HADR 远程服务”监视元素

远程 HADR TCP 服务。此值将显示为服务名称字符串或端口号字符串。

表 727. 快照监视信息

快照级别	逻辑数据分组	监视开关
数据库	hadr	基本

用法

使用此元素来确定有效 HADR 远程服务名称。

对此元素的更改将在数据库激活时生效，或者，如果数据库已联机，那么在 HADR 在主数据库上停止然后重新启动后生效。

要点：V10.1 中已经不推荐使用此监视元素，在以后的发行版中可能会将其除去。有关更多信息，请参阅 《*DB2 V10.1 新增内容*》中的『不推荐使用 HADR 的某些监视接口』。

如果数据库的 HADR 角色是标准的，那么应该忽略此元素。使用 **hadr_role** 监视元素来确定数据库的 HADR 角色。

hadr_role -“HADR 角色”

数据库的高可用性灾难恢复 (HADR) 角色。

表 728. 表函数监视信息

表函数	监视元素收集级别
MON_GET_HADR 表函数 - 返回高可用性灾难恢复 (HADR) 监视信息	ACTIVITY METRICS BASE

表 729. 快照监视信息

快照级别	逻辑数据分组	监视开关
数据库	hadr	基本

用法

使用此元素来确定数据库的 HADR 角色。

此元素的数据类型是整型。

此元素的值是下列其中一个常量:

SQLM_HADR_ROLE_STANDARD
数据库不是 HADR 数据库。

SQLM_HADR_ROLE_PRIMARY
数据库是主 HADR 数据库。

SQLM_HADR_ROLE_STANDBY
数据库是备用 HADR 数据库。

要点: V10.1 中已经不推荐使用此监视元素，在以后的发行版中可能会将其除去。有关更多信息，请参阅 《*DB2 V10.1 新增内容*》中的『不推荐使用 HADR 的某些监视接口』。

hadr_standby_log_file -“HADR 备用日志文件”监视元素

当前日志文件在备用 HADR 数据库上的名称。

表 730. 快照监视信息

快照级别	逻辑数据分组	监视开关
数据库	hadr	基本

用法

使用此元素来确定备用 HADR 数据库上的当前日志文件。

要点: V10.1 中已经不推荐使用此监视元素，在以后的发行版中可能会将其除去。有关更多信息，请参阅 《*DB2 V10.1 新增内容*》中的『不推荐使用 HADR 的某些监视接口』。

如果数据库的 HADR 角色是标准的，那么应该忽略此元素。使用 **hadr_role** 监视元素来确定数据库的 HADR 角色。

hadr_standby_log_lsn -“HADR 备用日志 LSN”监视元素

备用 HADR 数据库的当前日志位置。日志序号（LSN）是数据库的日志流中的字节位移。

表 731. 快照监视信息

快照级别	逻辑数据分组	监视开关
数据库	hadr	基本

用法

使用此元素来确定备用 HADR 数据库上的当前日志位置。

要点：V10.1 中已经不推荐使用此监视元素，在以后的发行版中可能会将其除去。有关更多信息，请参阅 《*DB2 V10.1 新增内容*》中的『不推荐使用 HADR 的某些监视接口』。

如果数据库的 HADR 角色是标准的，那么应该忽略此元素。使用 **hadr_role** 监视元素来确定数据库的 HADR 角色。

hadr_standby_log_page -“HADR 备用日志页”监视元素

当前日志文件中的页号，指示当前日志在备用 HADR 数据库上的位置。页号与日志文件相关。例如，页零是文件的开头。

表 732. 快照监视信息

快照级别	逻辑数据分组	监视开关
数据库	hadr	基本

用法

使用此元素来确定备用 HADR 数据库上的当前日志页。

要点：V10.1 中已经不推荐使用此监视元素，在以后的发行版中可能会将其除去。有关更多信息，请参阅 《*DB2 V10.1 新增内容*》中的『不推荐使用 HADR 的某些监视接口』。

如果数据库的 HADR 角色是标准的，那么应该忽略此元素。使用 **hadr_role** 监视元素来确定数据库的 HADR 角色。

hadr_state -“HADR 状态”监视元素

数据库的高可用性灾难恢复 (HADR) 状态。

表 733. 表函数监视信息

表函数	监视元素收集级别
MON_GET_HADR 表函数 - 返回高可用性灾难恢复 (HADR) 监视信息	ACTIVITY METRICS BASE

表 734. 快照监视信息

快照级别	逻辑数据分组	监视开关
数据库	hadr	基本

用法

使用此元素来确定数据库的 HADR 状态。

此元素的数据类型是整型。如果数据库具有 HADR 主角色或备用角色，那么此元素的值是下列其中一个常量:

SQLM_HADR_STATE_DISCONNECTED
数据库未连接至它的伙伴数据库。

SQLM_HADR_STATE_LOC_CATCHUP
数据库正在进行本地同步复制。

SQLM_HADR_STATE_REM_CATCH_PEND
数据库正在等待连接至它的伙伴数据库以执行远程同步复制。

SQLM_HADR_STATE_REM_CATCHUP
数据库正在进行远程同步复制。

SQLM_HADR_STATE_PEER
在主数据库与备用数据库之间已建立连接，并且它们处于对等状态。

SQLM_HADR_STATE_DISCONN_PEER
主数据库与备用数据库处于断开对等状态。

如果数据库的 HADR 角色是标准的，那么应该忽略此元素。使用 **hadr_role** 监视元素来确定数据库的 HADR 角色。

要点: V10.1 中已经不推荐使用此监视元素，在以后的发行版中可能会将其除去。有关更多信息，请参阅 《*DB2 V10.1 新增内容*》中的『不推荐使用 HADR 的某些监视接口』。

hadr_syncmode -“HADR 同步方式”监视元素

数据库的高可用性灾难恢复 (HADR) 同步方式。

表 735. 表函数监视信息

表函数	监视元素收集级别
MON_GET_HADR 表函数 - 返回高可用性灾难恢复 (HADR) 监视信息	ACTIVITY METRICS BASE

表 736. 快照监视信息

快照级别	逻辑数据分组	监视开关
数据库	hadr	基本

用法

使用此元素来确定数据库的 HADR 同步方式。

此元素的数据类型是整型。

对此元素的更改将在数据库激活时生效，或者，如果数据库已联机，那么在 HADR 在主数据库上停止然后重新启动后生效。

如果数据库具有 HADR 主角色或备用角色，那么此元素的值是下列其中一个常量:

SQLM_HADR_SYNCMODE_SYNC
SYNC 方式。

SQLM_HADR_SYNCMODE_NEARSYNC
NEARSYNC 方式。

SQLM_HADR_SYNCMODE_ASYNC
ASYNC 方式。

SQLM_HADR_SYNCMODE_SUPERASYNC
SUPERASYNC 方式。

要点: V10.1 中已经不推荐使用此监视元素，在以后的发行版中可能会将其除去。有关更多信息，请参阅 《*DB2 V10.1 新增内容*》中的『不推荐使用 HADR 的某些监视接口』。

如果数据库的 HADR 角色是标准的，那么应该忽略此元素。使用 **hadr_role** 监视元素来确定数据库的 HADR 角色。

hadr_timeout -"HADR 超时"监视元素

没有与其伙伴进行任何通信的秒数，HADR 数据库服务器在此时间之后将认为它们之间的连接出现故障。

表 737. 表函数监视信息

表函数	监视元素收集级别
MON_GET_HADR 表函数 - 返回高可用性灾难恢复 (HADR) 监视信息	ACTIVITY METRICS BASE

表 738. 快照监视信息

快照级别	逻辑数据分组	监视开关
数据库	hadr	基本

用法

使用此元素来确定有效 HADR 超时值。

对此元素的更改将在数据库激活时生效，或者，如果数据库已联机，那么在 HADR 在主数据库上停止然后重新启动后生效。

要点: V10.1 中已经不推荐使用此监视元素，在以后的发行版中可能会将其除去。有关更多信息，请参阅 《*DB2 V10.1 新增内容*》中的『不推荐使用 HADR 的某些监视接口』。

如果数据库的 HADR 角色是标准的，那么应该忽略此元素。使用 **hadr_role** 监视元素来确定数据库的 HADR 角色。

hash_join_overflows -“散列连接溢出数”

散列连接数据超过可用排序堆空间的次数。

元素标识

hash_join_overflows

元素类型

计数器

表 739. 快照监视信息

快照级别	逻辑数据分组	监视开关
数据库	dbase	基本
应用程序	appl	基本

可将快照监视的计数器重置。

表 740. 事件监视信息

事件类型	逻辑数据分组	监视开关
数据库	event_db	始终收集
连接	event_conn	始终收集

用法 在数据库级别，如果 hash_join_small_overflows 的值大于此 hash_join_overflows 的 10%，那么应该考虑增加排序堆大小。应用程序级别的值可用于评估各个应用程序的散列连接性能。

hash_join_small_overflows -“散列连接小溢出数”

散列连接数据超过可用排序堆空间的部分小于 10% 的次数。

元素标识

hash_join_small_overflows

元素类型

计数器

表 741. 快照监视信息

快照级别	逻辑数据分组	监视开关
数据库	dbase	基本
应用程序	appl	基本

可将快照监视的计数器重置。

表 742. 事件监视信息

事件类型	逻辑数据分组	监视开关
数据库	event_db	始终收集
连接	event_conn	始终收集

用法 如果此值和 hash_join_overflows 很高，那么应考虑增加排序堆阈值。如果此值超过 hash_join_overflows 的 10%，那么应考虑增加排序堆大小。

histogram_type -“直方图类型”监视元素

直方图的类型，字符串格式。

有 7 个直方图类型。

CoordActQueueTime

在协调程序成员上测量的非嵌套活动排队（例如，在阈值队列中）的耗用时间直方图。

CoordActExecTime

非嵌套活动在协调程序成员执行时所耗的时间的直方图。执行时间不包括初始化或排队所花的时间。对于游标来说，执行时间只包括打开、访存和关闭请求所花的时间。在服务子类之间重新映射活动时，只有在该活动所在的服务子类执行完成的情况下，才会更新执行时间直方图。

CoordActLifetime

从非嵌套活动被数据库管理器标识到该活动执行完成的耗用时间直方图（在协调程序成员上测量）。 在服务子类之间重新映射活动时，只有在该活动所在的服务子类执行完成的情况下，才会更新生存期直方图。

CoordActInterArrivalTime

非嵌套协调程序活动的到达之间的时间间隔的直方图。将对活动进入系统时所借助于的服务子类计算到达之间时间平均值。在服务子类之间重新映射活动时，活动重新映射到的服务子类的到达之间时间直方图不受影响。

CoordActEstCost

非嵌套 DML 活动估算成本直方图。活动的估计成本只计入该活动从中进入系统的服务子类。

ReqExecTime

请求执行次数的直方图，它包括对协调程序成员的请求和对协调程序与非协调程序成员的子请求（如 RPC 请求或 SMP 子代理程序请求）。所包括的请求可能与活动相关联，也可能不会与活动相关联: 例如，PREPARE 和 OPEN 请求均包括在此直方图中，但当 OPEN 请求始终与游标活动相关联时，PREPARE

请求不是任何活动的一部分。重新映射所涉及的服务子类的执行时间直方图对该服务子类中的不完整请求所耗用的执行时间进行计数。

UowLifetime

从数据库管理器标识工作单元到此工作单元完成执行（已落实或已回滚）的耗用时间（以毫秒计）的直方图。

表 743. 事件监视信息

事件类型	逻辑数据分组	监视开关
统计信息	event_histogrambin	始终收集

用法

使用此元素来标识直方图的类型。可以有几个直方图属于同一统计信息记录，但每种类型只能有一个。

hld_application_handle -“挂起锁定的应用程序的标识”监视元素

挂起锁定的应用程序的系统范围内的唯一标识。如果挂起此锁定的应用程序未知或者找不到，那么将返回值 NULL。

表 744. 表函数监视信息

表函数	监视元素收集级别
MON_GET_APPL_LOCKWAIT 表函数 - 获取有关应用程序正在等待的锁定的信息	ACTIVITY METRICS BASE

hld_member - 挂起锁定的应用程序的数据库成员

应用程序挂起的锁定所在的数据库成员。

表 745. 表函数监视信息

表函数	监视元素收集级别
MON_GET_APPL_LOCKWAIT 表函数 - 获取有关应用程序正在等待的锁定的信息	ACTIVITY METRICS BASE

用法

如果要挂起的锁定在远程成员上，那么 **hld_member** 的值为 -2。要确定所挂起的锁定所在的成员，请使用 MON_GET_LOCKS 表函数并将 **lock_name** 指定为搜索参数。

host_ccsid -“主机编码字符集标识”

此项是主机数据库的编码字符集标识（CCSID）。

元素标识

host_ccsid

元素类型

信息

表 746. 快照监视信息

快照级别	逻辑数据分组	监视开关
DCS 应用程序	dcs_appl_info	基本

用法 此元素用于 DCS 应用程序的问题确定。

host_db_name -“主机数据库名称”

对其收集信息或应用程序连接至的主机数据库的真实名称。这是创建数据库时给定的名称。

表 747. 快照监视信息

快照级别	逻辑数据分组	监视开关
DCS 数据库	dcs_dbase	基本
DCS 应用程序	dcs_appl_info	基本

用法 此元素用于 DCS 应用程序的问题确定。

hostname -“主机名”监视元素

数据库成员所在机器的主机名。

表 748. 表函数监视信息

表函数	监视元素收集级别
DB2_CLUSTER_HOST_STATE 管理视图和 DB2_GET_CLUSTER_HOST_STATE 表函数 - 获取有关主机的信息	ACTIVITY METRICS BASE
MON_GET_CF_CMD 表函数 - 获取集群高速缓存工具命令处理时间	ACTIVITY METRICS BASE
MON_GET_CF_WAIT_TIME 表函数 - 获取集群高速缓存工具命令等待时间	ACTIVITY METRICS BASE
MON_GET_FCM - 获取 FCM 度量值	ACTIVITY METRICS BASE
MON_GET_SERVERLIST 表函数 - 获取成员优先级详细信息	ACTIVITY METRICS BASE
MON_GET_SERVICE_SUBCLASS - 获取服务子类度量值	ACTIVITY METRICS BASE
MON_GET_WORKLOAD - 获取工作负载度量值	ACTIVITY METRICS BASE
MON_SAMPLE_SERVICE_CLASS_METRICS - 获取样本服务类度量值	ACTIVITY METRICS BASE
MON_SAMPLE_WORKLOAD_METRICS - 获取样本工作负载度量值	ACTIVITY METRICS BASE

host_name -“主机名”监视元素

集群高速缓存设施进程所在的主机的名称。

表 749. 表函数监视信息

表函数	监视元素收集级别
DB_MEMBERS 表函数	ACTIVITY METRICS BASE
ENV_GET_NETWORK_RESOURCES 表函数 - 返回网络适配器信息	ACTIVITY METRICS BASE
ENV_GET_SYSTEM_RESOURCES 表函数 - 返回系统信息	ACTIVITY METRICS BASE
MON_GET_CF 表函数 - 获取 CF 度量值	ACTIVITY METRICS BASE
MON_GET_MEMORY_POOL 表函数 - 获取内存池信息	ACTIVITY METRICS BASE
MON_GET_MEMORY_SET 表函数 - 获取内存集合信息	ACTIVITY METRICS BASE

host_prdid -“主机产品/版本标识”

正在服务器上运行的产品和版本。

表 750. 快照监视信息

快照级别	逻辑数据分组	监视开关
DCS 应用程序	dcs_appl_info	基本

用法 用于标识 DRDA 主机数据库产品及其代码版本号。其格式为 PPPVVRRM，其中:

- PPP 标识主机 DRDA 产品
  - ARI 用于 DB2 Server for VSE & VM
  - DSN 用于 DB2 z/OS 版
  - QSQ 用于 DB2 i 版
  - SQL 用于其他 DB2 产品。
- VV 标识两位版本号（版本只有一位时则高位为 0）
- RR 标识两位发行版号（发行版只有一位时高位为 0）
- M 标识 1 个字符的修改级别（0-9 或 A-Z）

host_response_time -“主机响应时间”

在 DCS 语句级别，此项是语句从 DB2 Connect 网关发送至主机以进行处理的时间与从主机接收到结果的时间之间所耗用的时间。在 DCS 数据库和 DCS 应用程序级别，此项是对特定应用程序或数据库执行的所有语句所耗用的时间。在数据传输级别，此项是使用这么多数据传输的所有语句的主机响应时间的总和。

表 751. 快照监视信息

快照级别	逻辑数据分组	监视开关
DCS 数据库	dcs_dbase	语句

表 751. 快照监视信息 (续)

快照级别	逻辑数据分组	监视开关
DCS 应用程序	dcs_appl	语句，时间戳记
DCS 语句	dcs_stmt	语句，时间戳记
数据传输	stmt_transmissions	语句，时间戳记

对于语句级别的快照监视，不能重置此计数器。可在其他级别重置此计数器。

用法

将此元素与发送的出站字节数和接收的出站字节配合使用来计算出站响应时间（传输速率）：

```
(发送的出站字节数 + 接收的出站字节数) / 主机响应时间
```

此元素由两个子元素组成，它们报告耗用时间的秒数和微秒（一秒的百万分之一）数。这些子元素的名称可通过将"_s"和"_ms"添加至此监视元素的名称派生而成。要检索此监视元素耗用的总时间，必须将这两个子元素的值加在一起。例如，如果"_s"子元素值为 3，"_ms"子元素值为 20，那么此监视元素耗用的总时间为 3.00002 秒。

id -"集群高速缓存设施标识"监视元素

集群高速缓存设施标识，如 `db2nodes.cfg` 文件中的定义。

表 752. 表函数监视信息

表函数	监视元素收集级别
DB2_MEMBER 管理视图、DB2_CF 管理视图和 DB2_GET_INSTANCE_INFO 表函数	ACTIVITY METRICS BASE
MON_GET_CF 表函数 - 获取 CF 度量值	ACTIVITY METRICS BASE
MON_GET_CF_CMD 表函数 - 获取集群高速缓存工具命令处理时间	ACTIVITY METRICS BASE
MON_GET_CF_WAIT_TIME 表函数 - 获取集群高速缓存工具命令等待时间	ACTIVITY METRICS BASE

idle_agents -"空闲代理程序数"

代理程序池中当前未分配给应用程序而"空闲"的代理程序数。

表 753. 快照监视信息

快照级别	逻辑数据分组	监视开关
数据库管理器	db2	基本

用法 可使用此元素来帮助设置 *num_poolagents* 配置参数。如果具有可用来处理代理程序请求的空闲代理程序，就可以改进性能。

iid -“索引标识”监视元素

索引的标识。

表 754. 表函数监视信息

表函数	监视元素收集级别
ADMIN_GET_INDEX_COMPRESS_INFO 表函数 - 返回压缩索引信息	ACTIVITY METRICS BASE
ADMIN_GET_INDEX_INFO 表函数 - 返回索引信息	ACTIVITY METRICS BASE
MON_GET_INDEX 表函数 - 获取索引度量值	ACTIVITY METRICS BASE
MON_GET_PAGE_ACCESS_INFO 表函数 - 获取缓冲池页等待信息	ACTIVITY METRICS BASE

inbound_bytes_received -“接收的入站字节数”

DB2 Connect 网关从客户机接收的字节数，通信协议使用情况（如 TCP/IP）除外。

表 755. 快照监视信息

快照级别	逻辑数据分组	监视开关
DCS 应用程序	dcs_appl	基本
DCS 语句	dcs_stmt	语句

对于应用程序级别的快照监视，可重置此计数器。不能在其他级别重置此计数器。

用法 使用此元素来度量从客户机至 DB2 Connect 网关的吞吐量。

inbound_bytes_sent -“发送的入站字节数”

DB2 Connect 网关发送到客户机的字节数，通信协议使用情况（如 TCP/IP）除外。

表 756. 快照监视信息

快照级别	逻辑数据分组	监视开关
DCS 应用程序	dcs_appl	基本
DCS 语句	dcs_stmt	语句

对于应用程序级别的快照监视，可重置此计数器。不能在其他级别重置此计数器。

用法 使用此元素来度量从 DB2 Connect 网关至客户机的吞吐量。

inbound_comm_address -“入站通信地址”

此项是客户机的通信地址。例如，它可能是 TCP/IP 的 IP 地址和端口号。

表 757. 快照监视信息

快照级别	逻辑数据分组	监视开关
应用程序	appl_info	基本
DCS 应用程序	dcs_appl_info	基本

用法 此元素用于 DCS 应用程序的问题确定。

include_col_updates -“更新包括列次数”监视元素

更新包括列次数。

表 758. 表函数监视信息

表函数	监视元素收集级别
MON_GET_INDEX 表函数 - 获取索引度量值	ACTIVITY METRICS BASE

incremental_bind -“增量绑定”监视元素

指示在执行时程序包是否以增量方式绑定。可能的值为 YES 或 NO。

表 759. 事件监视信息

事件类型	逻辑数据分组	监视开关
锁定	lock_participant_activities	

index_jump_scans -“索引跳跃扫描数”监视元素

跳跃扫描数。跳跃扫描是这样一种索引扫描，其中索引开始关键字与停止关键字之间存在间隔并且索引的未产生结果的部分被跳过。

表 760. 表函数监视信息

表函数	监视元素收集级别
MON_GET_INDEX 表函数 - 获取索引度量值	ACTIVITY METRICS BASE

用法

可使用此监视元素来验证跳跃扫描按期望速率发生。如果此监视元素的值未按预期增加，请检查以下情况:

- 对于在您期望发生跳跃扫描的位置发出的查询，请验证目标表是否有适当组合索引以及查询谓词是否引入了索引间隔。如果没有索引间隔，那么 DB2 优化器不会创建带有跳跃扫描的方案。
- 跳跃扫描不会扫描下列类型的索引:
  - 范围集群表索引
  - 扩展索引（例如，空间索引）
  - XML 索引
  - 文本索引（对于文本搜索）

注: 创建存取方案时，DB2 优化器会执行成本分析以确定是否应执行跳跃扫描。优化器有时会发现不使用跳跃扫描更高效。

index_name -“索引名”监视元素

索引的名称。

表 761. 表函数监视信息

表函数	监视元素收集级别
ADMIN_GET_INDEX_COMPRESS_INFO 表函数 - 返回压缩索引信息	ACTIVITY METRICS BASE
ADMIN_GET_INDEX_INFO 表函数 - 返回索引信息	ACTIVITY METRICS BASE
MON_GET_INDEX_USAGE_LIST 表函数 - 返回索引用法列表中的信息	ACTIVITY METRICS BASE

index_schema -“索引模式”监视元素

索引模式的名称。

表 762. 表函数监视信息

表函数	监视元素收集级别
ADMIN_GET_INDEX_COMPRESS_INFO 表函数 - 返回压缩索引信息	ACTIVITY METRICS BASE
ADMIN_GET_INDEX_INFO 表函数 - 返回索引信息	ACTIVITY METRICS BASE
MON_GET_INDEX_USAGE_LIST 表函数 - 返回索引用法列表中的信息	ACTIVITY METRICS BASE

index_object_pages -“索引对象页数”

对表定义的所有索引消耗的磁盘页数。

表 763. 快照监视信息

快照级别	逻辑数据分组	监视开关
表	表	基本

表 764. 事件监视信息

事件类型	逻辑数据分组	监视开关
表	event_table	始终收集

用法 此元素提供了一种机制，可用来查看对特定表定义的索引消耗的实际空间量。此元素可与表事件监视器配合使用以跟踪一段时间内索引的增长率。不会对分区表返回此元素。

index_object_l_pages -“索引数据逻辑页数”监视元素

与此表相关联的所有索引在磁盘上使用的逻辑页数。对于分区表，返回的值为 NULL。

表 765. 表函数监视信息

表函数	监视元素收集级别
MON_GET_TABLE 表函数 - 获取表度量值	ACTIVITY METRICS BASE

用法

- 此值可能小于实际上为该对象分配的空间量。使用 TRUNCATE 语句的 REUSE STORAGE 选项时，可能会发生这种情况。此选项导致为该表分配的存储空间继续被分配，尽管该存储空间被视为空。此外，此监视元素的值可能小于逻辑上为该对象分配的空间量，因为逻辑上分配的空间总量包括少量附加元数据。

  要检索对象的逻辑大小或物理大小的准确度量，请使用 ADMIN_GET_TAB_INFO_V97 函数。此函数提供的有关对象大小的信息比您可（通过将针对此监视元素报告的页数乘以页大小）获取的对象大小更准确。

index_only_scans -“纯索引扫描次数”监视元素

纯索引扫描次数。如果仅通过访问索引即可获得扫描结果，那么将执行纯索引扫描。

表 766. 表函数监视信息

表函数	监视元素收集级别
MON_GET_INDEX 表函数 - 获取索引度量值	ACTIVITY METRICS BASE

index_scans -“索引扫描次数”监视元素

索引扫描次数。

表 767. 表函数监视信息

表函数	监视元素收集级别
MON_GET_INDEX 表函数 - 获取索引度量值	ACTIVITY METRICS BASE

index_tbsp_id -“索引表空间标识”监视元素

一个表空间的标识，此表空间用于存放对此表创建的索引。

表 768. 表函数监视信息

表函数	监视元素收集命令和级别
MON_GET_TABLE 表函数 - 获取表度量值	ACTIVITY METRICS BASE

用法

此元素的值与视图 SYSCAT.TABLESPACES 的列 TBSPACEID 中的值相匹配。

input_db_alias -“输入数据库别名”

调用快照函数时提供的数据库别名。

表 769. 快照监视信息

快照级别	逻辑数据分组	监视开关
数据库	dbase	基本
应用程序	appl_id_info	基本
表空间	tablespace_list	缓冲池
缓冲池	bufferpool	缓冲池
表	table_list	表
锁定	db_lock_list	基本

用法 此元素可用来标识监视器数据适用的特定数据库。除非请求与特定数据库相关的监视器信息，否则它将包含空白。

此字段的值可能不同于 *client_db_alias* 监视元素的值，原因是数据库可能具有许多不同别名。不同应用程序和用户可使用不同别名来连接至同一数据库。

insert_sql_stmts -“插入数”

此元素包含自联合服务器实例启动或数据库监视计数器最后一次重置以后（取较晚者），联合服务器代表任何应用程序对此数据源发出 INSERT 语句的总次数。

表 770. 快照监视信息

快照级别	逻辑数据分组	监视开关
数据库	dbase_remote	基本
应用程序	appl_remote	基本

可将快照监视的计数器重置。

用法

使用此元素来确定联合服务器或应用程序对此数据源执行的数据库活动的级别。

还可使用此元素并借助以下公式来确定联合服务器或应用程序对此数据源执行的写入活动的百分比:

```
write_activity =
         (INSERT 语句数 + UPDATE 语句数 + DELETE 语句数 ) /
         (SELECT 语句数 + INSERT 语句数 + UPDATE 语句数 +
         DELETE 语句数)
```

insert_time -“插入响应时间”

此元素包含此数据源响应 INSERT 所花的总时间（以毫秒计），这些 INSERT 来自联合服务器启动后或数据库监视计数器上一次重置后（取较晚者）在此联合服务器实例上运行的所有应用程序或单个应用程序。

响应时间是以联合服务器将 INSERT 语句提交给数据源的时间与数据源响应联合服务器以指示已处理 INSERT 的时间之差量度的。

表 771. 快照监视信息

快照级别	逻辑数据分组	监视开关
数据库	dbase_remote	时间戳记
应用程序	appl_remote	时间戳记

可将快照监视的计数器重置。

用法

使用此元素来确定等待处理对此数据源执行 INSERT 语句所花的实际时间。此信息对于容量规划和容量调整非常有用。

insert_timestamp -“插入时间戳记”监视元素

将语句或部分插入到高速缓存中的时间。对于动态 SQL 快照，这表示将语句输入到高速缓存中的时间。对于 MON_GET_PKG_CACHE_STMT、MON_GET_PKG_CACHE_STMT_DETAILS 和程序包高速缓存事件监视器，值的粒度更细，并且表示将此语句的单个部分插入到高速缓存中的时间。

表 772. 表函数监视信息

表函数	监视元素收集级别
MON_GET_PKG_CACHE_STMT 表函数 - 获取程序包高速缓存中的 SQL 语句活动度量值	ACTIVITY METRICS BASE
MON_GET_PKG_CACHE_STMT_DETAILS 表函数 - 获取程序包高速缓存条目的详细度量值	ACTIVITY METRICS BASE

表 773. 快照监视信息

快照级别	逻辑数据分组	监视开关
动态 SQL	dynsql	基本

表 774. 事件监视信息

事件类型	逻辑数据分组	监视开关
程序包高速缓存	-	COLLECT BASE DATA

用法

此元素指定语句插入到高速缓存中的时间。它可用来估计高速缓存中语句的生存期。

int_auto_rebinds -“内部自动重新绑定次数”

尝试的自动重新绑定（或重新编译）数。

表 775. 快照监视信息

快照级别	逻辑数据分组	监视开关
数据库	dbase	基本

表 775. 快照监视信息 (续)

快照级别	逻辑数据分组	监视开关
应用程序	appl	基本

可将快照监视的计数器重置。

表 776. 事件监视信息

事件类型	逻辑数据分组	监视开关
数据库	event_db	始终收集
连接	event_conn	始终收集

用法 自动重新绑定是程序包无效时系统执行的内部绑定。当数据库管理器需要从程序包执行 SQL 语句时，将首次执行重新绑定。例如，当您执行以下操作时程序包将变得无效:

- 删除计划所依赖的对象，如表、视图或索引
- 添加或删除外键
- 撤销计划所依赖的对象特权。

可使用此元素来确定应用程序或数据库级别的数据库活动的级别。因为 int_auto_rebinds 对性能有严重影响，所以应尽可能少使用它们。

还可使用此元素并借助以下公式来确定重新绑定活动的百分比:

```
int_auto_rebinds / 语句总数
```

此信息对于分析应用程序活动和吞吐量非常有用。

int_commits -“内部落实数”监视元素

数据库管理器在内部启动的落实总数。

表 777. 表函数监视信息

表函数	监视元素收集级别
MON_FORMAT_XML_METRICS_BY_ROW - 获取所有度量值的基于已格式化行的输出	不适用; 报告 XML 文档中作为格式化函数的输入提供的所有元素
MON_GET_CONNECTION 表函数 - 获取连接度量值	REQUEST METRICS BASE
MON_GET_CONNECTION_DETAILS 表函数 - 获取详细的连接度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE
MON_GET_SERVICE_SUBCLASS 表函数 - 获取服务子类度量值	REQUEST METRICS BASE
MON_GET_SERVICE_SUBCLASS_DETAILS 表函数 - 获取详细的服务子类度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE
MON_GET_UNIT_OF_WORK 表函数 - 获取工作单元度量值	REQUEST METRICS BASE

表 777. 表函数监视信息 (续)

表函数	监视元素收集级别
MON_GET_UNIT_OF_WORK 表函数 - 获取详细的工作单元度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE
MON_GET_WORKLOAD 表函数 - 获取工作负载度量值	REQUEST METRICS BASE
MON_GET_WORKLOAD_DETAILS 表函数 - 获取详细的工作负载度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE

表 778. 快照监视信息

快照级别	逻辑数据分组	监视开关
数据库	dbase	基本
应用程序	appl	基本

可将快照监视的计数器重置。

表 779. 事件监视信息

事件类型	逻辑数据分组	监视开关
数据库	event_db	始终收集
连接	event_conn	始终收集
统计信息	event_scstats（在 details_xml 文档中报告）	REQUEST METRICS BASE
统计信息	event_wlstats（在 details_xml 文档中报告）	REQUEST METRICS BASE
工作单元	在 system_metrics 文档中报告。	REQUEST METRICS BASE

用法

在发生下列任一事件期间可能发生内部落实:

- 重组
- 导入
- 绑定或预编译
- 应用程序在未执行显式 SQL COMMIT 语句的情况下结束（在 UNIX 上）。

此值（不包括显式 SQL COMMIT 语句）表示自出现下列情况后发生的内部落实数:

- 与数据库的连接（对于数据库级别信息，这是第一次连接的时间）
- 数据库监视器计数器的最后一次重置。

可使用此元素并通过下列表达式计算总和来计算工作单元总数:

```
    commit_sql_stmts
+ int_commits
+ rollback_sql_stmts
+ int_rollbacks
```

注: 计算的工作单元数将仅包括发生以下情况之后出现的工作单元:

- 与数据库的连接（对于数据库级别信息，这是第一次连接的时间）
- 数据库监视器计数器的最后一次重置。

此计算可在应用程序或数据库级别完成。

int_deadlock_rollbacks -“死锁导致的内部回滚数”

数据库管理器因为死锁而启动的强制回滚总数。为解决死锁，将对数据库管理器选择的应用程序中的当前工作单元执行回滚。

元素标识

int_deadlock_rollbacks

元素类型

计数器

表 780. 快照监视信息

快照级别	逻辑数据分组	监视开关
数据库	dbase	基本
应用程序	appl	基本

可将快照监视的计数器重置。

表 781. 事件监视信息

事件类型	逻辑数据分组	监视开关
连接	event_conn	始终收集

用法 此元素显示已中断并且可用作并行性问题的指示符的死锁数。这很重要，原因是 int_deadlock_rollbacks 会降低数据库的吞吐量。

此值包括在 int_rollbacks 给定的值中。

int_node_splits -“中间节点分割次数”监视元素

在插入操作期间分割中间索引节点的次数。

表 782. 表函数监视信息

表函数	监视元素收集级别
MON_GET_INDEX 表函数 - 获取索引度量值	ACTIVITY METRICS BASE

int_rollbacks -“内部回滚数”监视元素

数据库管理器在内部启动的回滚总数。

表 783. 表函数监视信息

表函数	监视元素收集级别
MON_FORMAT_XML_METRICS_BY_ROW - 获取所有度量值的基于已格式化行的输出	不适用; 报告 XML 文档中作为格式化函数的输入提供的所有元素

表 783. 表函数监视信息 (续)

表函数	监视元素收集级别
MON_GET_CONNECTION 表函数 - 获取连接度量值	REQUEST METRICS BASE
MON_GET_CONNECTION_DETAILS 表函数 - 获取详细的连接度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE
MON_GET_SERVICE_SUBCLASS 表函数 - 获取服务子类度量值	REQUEST METRICS BASE
MON_GET_SERVICE_SUBCLASS_DETAILS 表函数 - 获取详细的服务子类度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE
MON_GET_UNIT_OF_WORK 表函数 - 获取工作单元度量值	REQUEST METRICS BASE
MON_GET_UNIT_OF_WORK 表函数 - 获取详细的工作单元度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE
MON_GET_WORKLOAD 表函数 - 获取工作负载度量值	REQUEST METRICS BASE
MON_GET_WORKLOAD_DETAILS 表函数 - 获取详细的工作负载度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE

表 784. 快照监视信息

快照级别	逻辑数据分组	监视开关
数据库	dbase	基本
应用程序	appl	基本

可将快照监视的计数器重置。

表 785. 事件监视信息

事件类型	逻辑数据分组	监视开关
数据库	event_db	始终收集
连接	event_conn	始终收集
统计信息	event_scstats（在 details_xml 文档中报告）	REQUEST METRICS BASE
统计信息	event_wlstats（在 details_xml 文档中报告）	REQUEST METRICS BASE
工作单元	在 system_metrics 文档中报告。	REQUEST METRICS BASE

用法

当不能成功完成下列任何一项任务时，将进行内部回滚:

- 重组
- 导入

- 绑定或预编译
- 应用程序因为出现死锁或锁定超时而结束
- 应用程序在未执行显式落实或回滚语句的情况下结束（在 Windows 上）。

此值表示自发生下列情况之后出现的内部回滚数:

- 与数据库的连接（对于数据库级别信息，这是第一次连接的时间）
- 数据库监视器计数器的最后一次重置。

虽然此值不包括显式 SQL ROLLBACK 语句，但是包括 **int_deadlock_rollbacks** 监视元素中的计数。

可使用此元素并通过计算下列各项的总和来计算工作单元总数:

```
    commit_sql_stmts
+ int_commits
+ rollback_sql_stmts
+ int_rollbacks
```

注: 计算的工作单元数将包括发生以下情况之后出现的工作单元:

- 与数据库的连接（对于数据库级别信息，这是第一次连接的时间）
- 数据库监视器计数器的最后一次重置。

此计算可在应用程序或数据库级别完成。

int_rows_deleted -“删除的内部行数”

此项是因为内部活动而从数据库中删除的行数。

表 786. 快照监视信息

快照级别	逻辑数据分组	监视开关
数据库	dbase	基本
应用程序	appl	基本
应用程序	stmt	基本
动态 SQL	dynsql	语句

可将快照监视的计数器重置。

表 787. 事件监视信息

事件类型	逻辑数据分组	监视开关
数据库	event_db	始终收集
连接	event_conn	始终收集
语句	event_stmt	始终收集

用法 此元素可帮助了解您不熟悉的数据库管理器的内部活动。如果此活动很多，那么您可能想要评估表设计以确定对数据库定义的引用约束或触发器是否必要。

内部删除活动可能是由下列原因引起的:

- 强制实施 ON CASCADE DELETE 引用约束的级联删除
- 被触发的触发器。

int_rows_inserted -“插入的内部行数”

因为触发器导致的内部活动而插入到数据库中的行数。

表 788. 快照监视信息

快照级别	逻辑数据分组	监视开关
数据库	dbase	基本
应用程序	appl	基本
应用程序	stmt	基本
动态 SQL	dynsql	语句

可将快照监视的计数器重置。

表 789. 事件监视信息

事件类型	逻辑数据分组	监视开关
数据库	event_db	始终收集
连接	event_conn	始终收集
语句	event_stmt	始终收集

用法 此元素可帮助您了解数据库管理器的内部活动。如果此活动很多，那么您可能想要评估设计以确定是否可以更改它以降低此活动。

int_rows_updated -“更新的内部行数”

此项是因为内部活动而从数据库中更新的行数。

表 790. 快照监视信息

快照级别	逻辑数据分组	监视开关
数据库	dbase	基本
应用程序	appl	基本
应用程序	stmt	基本
动态 SQL	dynsql	语句

可将快照监视的计数器重置。

表 791. 事件监视信息

事件类型	逻辑数据分组	监视开关
数据库	event_db	始终收集
连接	event_conn	始终收集
语句	event_stmt	始终收集

用法 此元素可帮助了解您不熟悉的数据库管理器的内部活动。如果此活动很多，那么您可能想要评估表设计以确定对数据库定义的引用约束是否必要。

内部更新活动可能是由下列原因引起的:

- 强制使用 ON DELETE SET NULL 规则定义的引用约束的 *set null* 行更新
- 被触发的触发器。

intra_parallel_state -“分区内并行性的当前状态”监视元素

在语句级别、活动级别、事务级别或工作负载级别报告的分区内并行性的当前状态。可能的值为“YES”和“NO”，“YES”指示启用分区内并行性以便语句可与子代理程序一起运行，“NO”指示禁用分区内并行性。

表 792. 表函数监视信息

表函数	监视元素收集级别
MON_GET_ACTIVITY_DETAILS 表函数 - 获取完整的活动详细信息（在 XML 文档 DETAILS 中报告）	ACTIVITY METRICS BASE
MON_GET_CONNECTION 表函数 - 获取连接度量值	ACTIVITY METRICS BASE
MON_GET_CONNECTION_DETAILS 表函数 - 获取详细的连接度量值（在 XML 文档 DETAILS 中报告）	ACTIVITY METRICS BASE
MON_GET_UNIT_OF_WORK 表函数 - 获取工作单元度量值	ACTIVITY METRICS BASE
MON_GET_UNIT_OF_WORK_DETAILS 表函数 - 获取详细的工作单元度量值	ACTIVITY METRICS BASE

表 793. 事件监视信息

事件类型	逻辑数据分组	监视开关
活动	-	始终收集
锁定		
工作单元		

invocation_id -“调用标识”监视元素

此标识用于将例程的一次调用与工作单元中位于同一嵌套级别的其他调用区分开。它在特定嵌套级别的工作单元中是唯一的。

invocation_id 监视元素是 **stmt_invocation_id** 监视元素的别名。

表 794. 表函数监视信息

表函数	监视元素收集级别
MON_GET_ACTIVITY_DETAILS 表函数 - 获取完整的活动详细信息（在 XML 文档 DETAILS 中报告）	ACTIVITY METRICS BASE

表 795. 事件监视信息

事件类型	逻辑数据分组	监视开关
活动	event_activitystmt	-
锁定	-	-
带有详细信息的死锁的历史记录值[1]	event_stmt_history	-

表 795. 事件监视信息 (续)

事件类型	逻辑数据分组	监视开关
带有详细信息的死锁的历史记录 [1]	event_stmt_history	-
工作单元	在程序包列表中报告。	-

1 建议不要使用此选项。建议不要再使用此选项，将来的发行版可能会将其除去。请使用 CREATE EVENT MONITOR FOR LOCKING 语句来监视与锁定相关的事件，例如锁定超时、锁定等待和死锁。

用法

可以使用此元素来唯一地标识在其中执行了特定 SQL 语句的调用。还可将此元素与其他语句历史记录条目一起使用来了解导致死锁的 SQL 语句的顺序。

ipc_recv_volume -“进程间通信接收量”监视元素

数据服务器通过 IPC 从客户机接收的数据量。此值以字节计。

表 796. 表函数监视信息

表函数	监视元素收集级别
MON_FORMAT_XML_METRICS_BY_ROW - 获取所有度量值的基于已格式化行的输出	不适用; 报告 XML 文档中作为格式化函数的输入提供的所有元素
MON_GET_CONNECTION 表函数 - 获取连接度量值	REQUEST METRICS BASE
MON_GET_CONNECTION_DETAILS 表函数 - 获取详细的连接度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE
MON_GET_SERVICE_SUBCLASS 表函数 - 获取服务子类度量值	REQUEST METRICS BASE
MON_GET_SERVICE_SUBCLASS_DETAILS 表函数 - 获取详细的服务子类度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE
MON_GET_UNIT_OF_WORK 表函数 - 获取工作单元度量值	REQUEST METRICS BASE
MON_GET_UNIT_OF_WORK 表函数 - 获取详细的工作单元度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE
MON_GET_WORKLOAD 表函数 - 获取工作负载度量值	REQUEST METRICS BASE
MON_GET_WORKLOAD_DETAILS 表函数 - 获取详细的工作负载度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE

表 797. 事件监视信息

事件类型	逻辑数据分组	监视开关
统计信息	event_scstats（在 details_xml 文档中报告）	REQUEST METRICS BASE
统计信息	event_wlstats（在 details_xml 文档中报告）	REQUEST METRICS BASE
工作单元	在 system_metrics 文档中报告。	-

ipc_recv_wait_time -“进程间通信接收等待时间”监视元素

代理程序使用 IPC 通信协议接收传入客户机请求时耗用的时间。此值以毫秒计。

表 798. 表函数监视信息

表函数	监视元素收集级别
MON_FORMAT_XML_METRICS_BY_ROW - 获取所有度量值的基于已格式化行的输出	不适用; 报告 XML 文档中作为格式化函数的输入提供的所有元素
MON_FORMAT_XML_TIMES_BY_ROW - - 获取基于已格式化行的组合层次结构等待时间和处理时间	不适用; 报告 XML 文档中作为格式化函数的输入提供的所有元素
MON_FORMAT_XML_WAIT_TIMES_BY _ROW - 获取等待时间的已格式化的基于行的输出	不适用; 报告 XML 文档中作为格式化函数的输入提供的所有元素
MON_GET_CONNECTION 表函数 - 获取连接度量值	REQUEST METRICS BASE
MON_GET_CONNECTION_DETAILS 表函数 - 获取详细的连接度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE
MON_GET_SERVICE_SUBCLASS 表函数 - 获取服务子类度量值	REQUEST METRICS BASE
MON_GET_SERVICE_SUBCLASS_DETAILS 表函数 - 获取详细的服务子类度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE
MON_GET_UNIT_OF_WORK 表函数 - 获取工作单元度量值	REQUEST METRICS BASE
MON_GET_UNIT_OF_WORK 表函数 - 获取详细的工作单元度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE
MON_GET_WORKLOAD 表函数 - 获取工作负载度量值	REQUEST METRICS BASE
MON_GET_WORKLOAD_DETAILS 表函数 - 获取详细的工作负载度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE

表 799. 事件监视信息

事件类型	逻辑数据分组	监视开关
统计信息	event_scstats（在 details_xml 文档中报告）	REQUEST METRICS BASE
统计信息	event_wlstats（在 details_xml 文档中报告）	REQUEST METRICS BASE
工作单元	在 system_metrics 文档中报告。	-

ipc_recvs_total -“进程间通信接收总次数”监视元素

数据库服务器使用 IPC 从客户机应用程序接收数据的次数。

表 800. 表函数监视信息

表函数	监视元素收集级别
MON_FORMAT_XML_METRICS_BY_ROW - 获取所有度量值的基于已格式化行的输出	不适用; 报告 XML 文档中作为格式化函数的输入提供的所有元素
MON_GET_CONNECTION 表函数 - 获取连接度量值	REQUEST METRICS BASE
MON_GET_CONNECTION_DETAILS 表函数 - 获取详细的连接度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE
MON_GET_SERVICE_SUBCLASS 表函数 - 获取服务子类度量值	REQUEST METRICS BASE
MON_GET_SERVICE_SUBCLASS_DETAILS 表函数 - 获取详细的服务子类度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE
MON_GET_UNIT_OF_WORK 表函数 - 获取工作单元度量值	REQUEST METRICS BASE
MON_GET_UNIT_OF_WORK 表函数 - 获取详细的工作单元度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE
MON_GET_WORKLOAD 表函数 - 获取工作负载度量值	REQUEST METRICS BASE
MON_GET_WORKLOAD_DETAILS 表函数 - 获取详细的工作负载度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE

表 801. 事件监视信息

事件类型	逻辑数据分组	监视开关
统计信息	event_scstats（在 details_xml 文档中报告）	REQUEST METRICS BASE
统计信息	event_wlstats（在 details_xml 文档中报告）	REQUEST METRICS BASE
工作单元	在 system_metrics 文档中报告。	-

ipc_send_volume -“进程间通信发送量”监视元素

数据服务器通过 IPC 协议发送到客户机的数据量。此值以字节计。

表 802. 表函数监视信息

表函数	监视元素收集级别
MON_FORMAT_XML_METRICS_BY_ROW - 获取所有度量值的基于已格式化行的输出	不适用; 报告 XML 文档中作为格式化函数的输入提供的所有元素
MON_GET_CONNECTION 表函数 - 获取连接度量值	REQUEST METRICS BASE
MON_GET_CONNECTION_DETAILS 表函数 - 获取详细的连接度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE
MON_GET_SERVICE_SUBCLASS 表函数 - 获取服务子类度量值	REQUEST METRICS BASE
MON_GET_SERVICE_SUBCLASS_DETAILS 表函数 - 获取详细的服务子类度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE
MON_GET_UNIT_OF_WORK 表函数 - 获取工作单元度量值	REQUEST METRICS BASE
MON_GET_UNIT_OF_WORK 表函数 - 获取详细的工作单元度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE
MON_GET_WORKLOAD 表函数 - 获取工作负载度量值	REQUEST METRICS BASE
MON_GET_WORKLOAD_DETAILS 表函数 - 获取详细的工作负载度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE

表 803. 事件监视信息

事件类型	逻辑数据分组	监视开关
统计信息	event_scstats（在 details_xml 文档中报告）	REQUEST METRICS BASE
统计信息	event_wlstats（在 details_xml 文档中报告）	REQUEST METRICS BASE
工作单元	在 system_metrics 文档中报告。	-

ipc_send_wait_time -“进程间通信发送等待时间”监视元素

通过 IPC 向客户机发送数据时被阻塞的时间。此值以毫秒计。

表 804. 表函数监视信息

表函数	监视元素收集级别
MON_FORMAT_XML_METRICS_BY_ROW - 获取所有度量值的基于已格式化行的输出	不适用; 报告 XML 文档中作为格式化函数的输入提供的所有元素

表 804. 表函数监视信息 (续)

表函数	监视元素收集级别
MON_FORMAT_XML_TIMES_BY_ROW - - 获取基于已格式化行的组合层次结构等待时间和处理时间	不适用; 报告 XML 文档中作为格式化函数的输入提供的所有元素
MON_FORMAT_XML_WAIT_TIMES_BY _ROW - 获取等待时间的已格式化的基于行的输出	不适用; 报告 XML 文档中作为格式化函数的输入提供的所有元素
MON_GET_CONNECTION 表函数 - 获取连接度量值	REQUEST METRICS BASE
MON_GET_CONNECTION_DETAILS 表函数 - 获取详细的连接度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE
MON_GET_SERVICE_SUBCLASS 表函数 - 获取服务子类度量值	REQUEST METRICS BASE
MON_GET_SERVICE_SUBCLASS_DETAILS 表函数 - 获取详细的服务子类度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE
MON_GET_UNIT_OF_WORK 表函数 - 获取工作单元度量值	REQUEST METRICS BASE
MON_GET_UNIT_OF_WORK 表函数 - 获取详细的工作单元度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE
MON_GET_WORKLOAD 表函数 - 获取工作负载度量值	REQUEST METRICS BASE
MON_GET_WORKLOAD_DETAILS 表函数 - 获取详细的工作负载度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE

表 805. 事件监视信息

事件类型	逻辑数据分组	监视开关
统计信息	event_scstats（在 details_xml 文档中报告）	REQUEST METRICS BASE
统计信息	event_wlstats（在 details_xml 文档中报告）	REQUEST METRICS BASE
工作单元	在 system_metrics 文档中报告。	-

ipc_sends_total -“进程间通信发送总次数”监视元素

数据库服务器使用 IPC 向客户机应用程序发送数据的次数。

表 806. 表函数监视信息

表函数	监视元素收集级别
MON_FORMAT_XML_METRICS_BY_ROW - 获取所有度量值的基于已格式化行的输出	不适用; 报告 XML 文档中作为格式化函数的输入提供的所有元素

表 806. 表函数监视信息 (续)

表函数	监视元素收集级别
MON_GET_CONNECTION 表函数 - 获取连接度量值	REQUEST METRICS BASE
MON_GET_CONNECTION_DETAILS 表函数 - 获取详细的连接度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE
MON_GET_SERVICE_SUBCLASS 表函数 - 获取服务子类度量值	REQUEST METRICS BASE
MON_GET_SERVICE_SUBCLASS_DETAILS 表函数 - 获取详细的服务子类度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE
MON_GET_UNIT_OF_WORK 表函数 - 获取工作单元度量值	REQUEST METRICS BASE
MON_GET_UNIT_OF_WORK 表函数 - 获取详细的工作单元度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE
MON_GET_WORKLOAD 表函数 - 获取工作负载度量值	REQUEST METRICS BASE
MON_GET_WORKLOAD_DETAILS 表函数 - 获取详细的工作负载度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE

表 807. 事件监视信息

事件类型	逻辑数据分组	监视开关
统计信息	event_scstats（在 details_xml 文档中报告）	REQUEST METRICS BASE
统计信息	event_wlstats（在 details_xml 文档中报告）	REQUEST METRICS BASE
工作单元	在 system_metrics 文档中报告。	-

is_system_appl -“是系统应用程序”监视元素

指示应用程序是否是系统应用程序。

表 808. 快照监视信息

快照级别	逻辑数据分组	监视开关
应用程序	appl_info	基本

用法

is_system_appl 监视元素指示应用程序是否是内部系统应用程序。可能的值包括

0 用户应用程序

1 系统应用程序

key_updates -“更新键次数”监视元素

更新键次数。

表 809. 表函数监视信息

表函数	监视元素收集级别
MON_GET_INDEX 表函数 - 获取索引度量值	ACTIVITY METRICS BASE

last_active_log -“最后一个活动日志文件编号”

最后一个活动日志文件的文件号。

元素标识

last_active_log

元素类型

信息

表 810. 表函数监视信息

表函数	监视元素收集级别
MON_GET_TRANSACTION_LOG 表函数 - 获取日志信息	ACTIVITY METRICS BASE

表 811. 快照监视信息

快照级别	逻辑数据分组	监视开关
数据库	detail_log	基本

表 812. 事件监视信息

事件类型	逻辑数据分组	监视开关
数据库	event_db	始终收集

用法 将此元素与 *first_active_log* 和 *current_active_log* 元素一起使用来确定活动日志文件的范围。知道活动日志文件的范围可帮助您确定日志文件所需的磁盘空间。

您也可以使用此元素来确定哪些日志文件包含数据，以帮助您标识分割镜像支持所需的日志文件。

last_backup -“上次备份时间戳记”

完成上次数据库备份的日期和时间。

元素标识

last_backup

元素类型

时间戳记

表 813. 快照监视信息

快照级别	逻辑数据分组	监视开关
数据库	dbase	时间戳记

用法 可使用此元素来帮助您标识最近没有备份的数据库，或者标识最新的数据库备份文件。如果数据库从未进行备份，那么此时间戳记将初始化为零。

last_executable_id -“上一个可执行文件标识”监视元素

应用程序最近完成的语句的可执行文件标识。

表 814. 表函数监视信息

表函数	监视元素收集级别
MON_GET_CONNECTION 表函数 - 获取连接度量值	ACTIVITY METRICS BASE
MON_GET_CONNECTION_DETAILS 表函数 - 获取详细的连接度量值（在 XML 文档 DETAILS 中报告）	ACTIVITY METRICS BASE
MON_GET_UNIT_OF_WORK 表函数 - 获取工作单元度量值	ACTIVITY METRICS BASE
MON_GET_UNIT_OF_WORK 表函数 - 获取详细的工作单元度量值（在 XML 文档 DETAILS 中报告）	ACTIVITY METRICS BASE

last_extent -“移动的最后一个扩展数据块”监视元素

表空间重新平衡程序过程所移动的最后一个扩展数据块的数字标识。

表 815. 表函数监视信息

表函数	监视元素收集级别
MON_GET_EXTENT_MOVEMENT_STATUS - 获取扩展数据块移动进度状态度量值	ACTIVITY METRICS BASE

last_metrics_update -“最近一次更新度量的时间戳记”监视元素

用于反映上一次更新此高速缓存条目的时间度量值的时间戳记。

表 816. 表函数监视信息

表函数	监视元素收集级别
MON_GET_PKG_CACHE_STMT 表函数 - 获取程序包高速缓存中的 SQL 语句活动度量值	ACTIVITY METRICS BASE
MON_GET_PKG_CACHE_STMT_DETAILS 表函数 - 获取程序包高速缓存条目的详细度量值	ACTIVITY METRICS BASE

表 817. 事件监视信息

事件类型	逻辑数据分组	监视开关
程序包高速缓存	-	COLLECT BASE DATA

last_overflow_time -“最后一次事件溢出时间”

此溢出记录的最后一次溢出的日期和时间。

表 818. 事件监视信息

事件类型	逻辑数据分组	监视开关
溢出记录	event_overflow	-

用法 将此元素与 *first_overflow_time* 配合使用来计算生成溢出记录所耗用的时间。

last_reference_time -“上次引用时间”监视元素

此活动上次被某个请求访问时的时间。

表 819. 表函数监视信息

表函数	监视元素收集级别
MON_GET_ACTIVITY_DETAILS 表函数 - 获取完整的活动详细信息（在 XML 文档 DETAILS 中报告）	ACTIVITY METRICS BASE

用法

last_request_type -“上一个请求类型”监视元素

应用程序完成的上一个请求的类型。

表 820. 表函数监视信息

表函数	监视元素收集级别
MON_GET_CONNECTION 表函数 - 获取连接度量值	ACTIVITY METRICS BASE
MON_GET_CONNECTION_DETAILS 表函数 - 获取详细的连接度量值（在 XML 文档 DETAILS 中报告）	ACTIVITY METRICS BASE
MON_GET_UNIT_OF_WORK 表函数 - 获取工作单元度量值	ACTIVITY METRICS BASE
MON_GET_UNIT_OF_WORK 表函数 - 获取详细的工作单元度量值（在 XML 文档 DETAILS 中报告）	ACTIVITY METRICS BASE

用法

仅对于应用程序的协调程序成员才会报告此监视元素。

它可以为下列值。

- CLOSE
- COMMIT
- COMPILE
- DESCRIBE
- EXCSQLSET
- EXECIMMD
- EXECUTE
- FETCH
- INTERNAL *number*，其中 *number* 是内部常量的值
- OPEN
- PREPARE
- REBIND
- REDISTRIBUTE
- REORG
- ROLLBACK
- RUNSTATS

last_reset -“最后重置时间戳记”

指示监视器计数器对发出 GET SNAPSHOT 的应用程序重置的日期和时间。

元素标识

last_reset

元素类型

时间戳记

表 821. 表函数监视信息

表函数	监视元素收集级别
WLM_GET_QUEUE_STATS 表函数 - 返回阈值队列统计信息	ACTIVITY METRICS BASE
WLM_GET_SERVICE_SUBCLASS_STATS 表函数 - 返回服务子类的统计信息	ACTIVITY METRICS BASE
WLM_GET_SERVICE_SUPERCLASS_STATS 表函数 - 返回服务超类的统计信息	ACTIVITY METRICS BASE
WLM_GET_WORK_ACTION_SET_STATS 表函数 - 返回工作操作集统计信息	ACTIVITY METRICS BASE
WLM_GET_WORKLOAD_STATS 表函数 - 返回工作负载统计信息	ACTIVITY METRICS BASE

表 822. 快照监视信息

快照级别	逻辑数据分组	监视开关
数据库管理器	db2	时间戳记
数据库	dbase	时间戳记
应用程序	appl	时间戳记
表空间	tablespace_list	缓冲池，时间戳记
表	table_list	时间戳记
DCS 数据库	dcs_dbase	时间戳记
DCS 应用程序	dcs_appl	时间戳记

用法 可使用此元素来帮助定义数据库系统监视器返回的信息的作用域。

如果计数器从未重置，那么此元素将为零。

仅当重置所有活动数据库时，数据库管理器计数器才会重置。

last_updated -“最近一次更新时间戳记”监视元素

用于指示最近一次更新此条目的时间戳记。

表 823. 表函数监视信息

表函数	监视元素收集级别
MON_GET_INDEX_USAGE_LIST 表函数 - 返回索引用法列表中的信息	ACTIVITY METRICS BASE
MON_GET_TABLE_USAGE_LIST 表函数 - 从表用法列表返回信息	ACTIVITY METRICS BASE

last_wlm_reset -“最后一次重置时间”监视元素

此元素为本地时间戳记形式，显示创建此类型的最后一个统计信息事件记录的时间。

表 824. 事件监视信息

事件类型	逻辑数据分组	监视开关
统计信息	event_scstats	-
统计信息	event_wlstats	-
统计信息	event_wcstats	-
统计信息	event_qstats	-

用法

使用 **wlm_last_reset** 和 **statistics_timestamp** 监视元素来确定收集事件监视器统计信息记录中的统计信息的时间段。收集时间间隔从 **wlm_last_reset** 时间开始，并且在 **statistics_timestamp** 结束。

lob_object_pages -“LOB 对象页数”

LOB 数据消耗的磁盘页数。

表 825. 快照监视信息

快照级别	逻辑数据分组	监视开关
表	表	基本

表 826. 事件监视信息

事件类型	逻辑数据分组	监视开关
表	event_table	始终收集

用法 此元素提供了一种机制，可用来查看特定表中的 LOB 数据消耗的实际空间量。此元素可与表事件监视器配合使用以跟踪一段时间内 LOB 数据的增长率。

lob_object_l_pages -“LOB 数据逻辑页数”监视元素

与此表相关联的 LOB 在磁盘上使用的逻辑页数。

表 827. 表函数监视信息

表函数	监视元素收集级别
MON_GET_TABLE 表函数 - 获取表度量值	ACTIVITY METRICS BASE

用法

- 此值可能小于实际上为该对象分配的空间量。使用 TRUNCATE 语句的 REUSE STORAGE 选项时，可能会发生这种情况。此选项导致为该表分配的存储空间继续被分配，尽管该存储空间被视为空。此外，此监视元素的值可能小于逻辑上为该对象分配的空间量，因为逻辑上分配的空间总量包括少量附加元数据。

  要检索对象的逻辑大小或物理大小的准确度量，请使用 ADMIN_GET_TAB_INFO_V97 函数。此函数提供的有关对象大小的信息比您可（通过将针对此监视元素报告的页数乘以页大小）获取的对象大小更准确。

local_cons -“本地连接数”

当前连接至正在监视的数据库管理器实例中的数据库的本地应用程序数。

表 828. 快照监视信息

快照级别	逻辑数据分组	监视开关
数据库管理器	db2	基本

用法

此数目可帮助您确定数据库管理器中进行的并行处理的级别。此值经常更改，所以可能需要在很长的时间段内按特定时间间隔对其进行采样，以了解实际的系统使用情况。

此数目仅包括从与数据库管理器相同的实例启动的应用程序。这些应用程序将进行连接，但它们可能执行也可能不执行数据库中的工作单元。

与 rem_cons_in 监视元素一起使用时，此元素可帮助您调整 **max_connections** 配置参数的设置。

local_cons_in_exec -“数据库管理器中正在执行的本地连接数”

当前连接至正在监视的数据库管理器实例中的数据库并且正在处理工作单元的本地应用程序数。

表 829. 快照监视信息

快照级别	逻辑数据分组	监视开关
数据库管理器	db2	基本

用法

此数目可帮助您确定数据库管理器中进行的并行处理的级别。此值经常更改，所以可能需要在很长的时间段内按特定时间间隔对其进行采样，以了解实际的系统使用情况。此数目仅包括从与数据库管理器相同的实例启动的应用程序。

与 **rem_cons_in_exec** 监视元素使用使用时，此元素可帮助您调整 **max_coordagents** 配置参数的设置。

以下建议仅适用于非集中器配置。如果启用了集中器，DB2 会将大量客户机连接多路复用到较小的协调代理程序池。在这种情况下，通常可以使 **rem_cons_in_exec** 与 **local_cons_in_exec** 之和接近 **max_coordagents** 值。

- 如果 **max_coordagents** 设置为 AUTOMATIC，那么不要作任何调整。
- 如果 **max_coordagents** 未设置为 AUTOMATIC，并且 **rem_cons_in_exec** 与 **local_cons_in_exec** 之和接近 **max_coordagents**，那么请增加 **max_coordagents** 的值。

local_start_time -“本地开始时间”监视元素

此活动开始作用于成员的时间。这是本地时间。如果此活动已进入系统，但由于正在队列中排队而尚未开始执行，那么此字段是空字符串。

表 830. 表函数监视信息

表函数	监视元素收集级别
MON_GET_ACTIVITY_DETAILS 表函数 - 获取完整的活动详细信息（在 XML 文档 DETAILS 中报告）	ACTIVITY METRICS BASE
WLM_GET_WORKLOAD_OCCURRENCE_ACTIVITIES 表函数 - 返回活动列表	ACTIVITY METRICS BASE

用法

local_transaction_id -“本地事务标识”监视元素

事件发生时正使用的本地事务标识。

表 831. 事件监视信息

事件类型	逻辑数据分组	监视开关
变更历史记录	ddlstmtexec txncompletion	始终收集
工作单元	uow	

用法

对于变更历史记录事件监视器，这是事件发生时正使用的本地事务标识。这是作为事务日志一部分的 SQLU_TID 结构。

location -“位置”

标识与该事件相关联的位置。

表 832. 事件监视信息

事件类型	逻辑数据分组	监视开关
变更历史记录	UTILLOCATION	始终收集

用法

对于变更历史记录事件监视器，位置取决于 UTILITY_TYPE，例如，装入输入文件或备份目标路径名。

location_type -“位置类型”

此位置的用途描述。

表 833. 事件监视信息

事件类型	逻辑数据分组	监视开关
变更历史记录	UTILLOCATION	始终收集

用法

如果 `utility_type` 元素为 LOAD，那么为下列其中一个:

C 复制目标

D 输入数据

L LOB 路径

X XML 路径

如果 `utility_type` 元素为 BACKUP，那么为下列其中一个:

B 备份目标位置

如果 utility_type 元素为 RESTORE，那么为下列其中一个:

S 恢复源位置

如果 utility_type 元素为 ROLLFORWARD，那么为下列其中一个:

O 作为 ROLLFORWARD DATABASE 命令的一部分捕获的备用溢出日志路径。请注意，如果使用缺省溢出日志路径，那么不会捕获任何位置记录。

否则为空白字符。

lock_attributes -“锁定属性”监视元素

当前正在挂起锁定的应用程序的锁定属性。

表 834. 表函数监视信息

表函数	监视元素收集级别
MON_GET_APPL_LOCKWAIT 表函数 - 获取有关应用程序正在等待的锁定的信息	ACTIVITY METRICS BASE
MON_GET_LOCKS 表函数 - 列示当前所连接的数据库中的所有锁定	ACTIVITY METRICS BASE

表 835. 快照监视信息

快照级别	逻辑数据分组	监视开关
锁定	lock	基本
锁定	lock_wait	基本

表 836. 事件监视信息

事件类型	逻辑数据分组	监视开关
锁定	-	始终收集
死锁 [1]	lock	始终收集
死锁 [1]	event_dlconn	始终收集

1 建议不要使用此事件监视器。建议不要再使用此选项，将来的发行版可能会将其除去。请使用 CREATE EVENT MONITOR FOR LOCKING 语句来监视与锁定相关的事件，例如锁定超时、锁定等待和死锁。

用法

下表列示所有可能的锁定属性设置。每个锁定属性设置都以 sqlmon.h 中定义的位标志值为基础。

表函数中的锁定属性值	API 常量	描述
0000000000000001	SQLM_LOCKATTR_WAIT_FOR_AVAIL	等待可用性。
0000000000000002	SQLM_LOCKATTR_ESCALATED	由升级获取。
0000000000000004	SQLM_LOCKATTR_RR_IN_BLOCK	块中的 RR 锁定。
0000000000000008	SQLM_LOCKATTR_INSERT	插入锁定。

表函数中的锁定属性

值	API 常量	描述
0000000000000010	SQLM_LOCKATTR_RR	由 RR 扫描锁定。
0000000000000020	SQLM_LOCKATTR_UPDATE_DELETE	更新/删除行锁定。
0000000000000040	SQLM_LOCKATTR_ALLOW_NEW	允许新锁定请求。
0000000000000080	SQLM_LOCKATTR_NEW_REQUEST	新锁定请求者。
0000000000000200	SQLM_LOCKATTR_INDOUBT	由不确定事务挂起的锁定。
0000000000000400	SQLM_LOCKATTR_LOW_PRIORITY	由低优先级应用程序挂起的锁定。

已返回、但是未列示在上面显示的表中的位将保留给内部使用。

lock_count -“锁定计数”监视元素

正在挂起的锁定上的锁定数目。

表 837. 表函数监视信息

表函数	监视元素收集级别
MON_GET_APPL_LOCKWAIT 表函数 - 获取有关应用程序正等待的锁定的信息	ACTIVITY METRICS BASE

表 838. 快照监视信息

快照级别	逻辑数据分组	监视开关
锁定	lock	基本

表 839. 事件监视信息

事件类型	逻辑数据分组	监视开关
锁定	-	始终收集
死锁 [1]	lock	始终收集
死锁 [1]	event_dlconn	始终收集

1 建议不要使用此事件监视器。建议不要再使用此选项，将来的发行版可能会将其除去。请使用 CREATE EVENT MONITOR FOR LOCKING 语句来监视与锁定相关的事件，例如锁定超时、锁定等待和死锁。

用法

此值的范围在 1 到 255 之间。获得新锁定时，此值递增；释放锁定时，此值递减。

lock_count 监视元素的值为 255 时，指示正在挂起事务持续时间锁定。此时，获得或释放锁定时，**lock_count** 不再递增或递减。在下列任一可能方式下，**lock_count** 监视元素的值设置为 255:

1. **lock_count** 监视元素值因为获取新锁定而递增 255 次。
2. 显式获得事务持续时间锁定。例如使用 LOCK TABLE 语句或 INSERT 语句。

lock_current_mode -“转换前的原始锁定方式”监视元素

在执行锁定转换操作期间，在完成转换之前，等待获取锁定的应用程序拥有的锁定方式。

表 840. 表函数监视信息

表函数	监视元素收集级别
MON_GET_APPL_LOCKWAIT 表函数 - 获取有关应用程序正在等待的锁定的信息	ACTIVITY METRICS BASE
MON_GET_LOCKS 表函数 - 列示当前所连接的数据库中的所有锁定	ACTIVITY METRICS BASE

表 841. 快照监视信息

快照级别	逻辑数据分组	监视开关
锁定	lock	基本
锁定	lock_wait	基本

表 842. 事件监视信息

事件类型	逻辑数据分组	监视开关
锁定	-	始终收集
死锁 [1]	lock	始终收集
死锁 [1]	event_dlconn	始终收集

1 建议不要使用此事件监视器。建议不要再使用此选项，将来的发行版可能会将其除去。请使用 CREATE EVENT MONITOR FOR LOCKING 语句来监视与锁定相关的事件，例如锁定超时、锁定等待和死锁。

用法

以下方案描述锁定转换示例。在更新或删除操作期间，可以等待针对目标行的 X 锁定。如果事务要挂起针对行的 S 或 V 锁定，那么需要转换。此时将对 **lock_current_mode** 元素指定值 S 或 V，而锁定等待转换为 X 锁定。

下表列示了可能采用的锁定方式。

方式	锁定类型	API 常量
	没有锁定	SQLM_LNON
IS	意向共享锁定	SQLM_LOIS
IX	意向互斥锁定	SQLM_LOIX
S	共享锁定	SQLM_LOOS
SIX	与意向互斥锁定共享	SQLM_LSIX
X	互斥锁定	SQLM_LOOX
IN	Intent None	SQLM_LOIN
Z	超级互斥锁定	SQLM_LOOZ
U	更新锁定	SQLM_LOOU
NS	扫描共享锁定	SQLM_LONS

方式	锁定类型	API 常量
NW	下一键弱互斥锁定	SQLM_LONW

lock_escalation -“锁定升级”监视元素

指示正在等待获取此锁定的应用程序是否是锁定升级请求的结果。可能值为 Y（Yes）和 N（No）。

表 843. 表函数监视信息

表函数	监视元素收集命令和级别
MON_GET_APPL_LOCKWAIT 表函数 - 获取有关应用程序正在等待的锁定的信息	ACTIVITY METRICS BASE

表 844. 快照监视信息

快照级别	逻辑数据分组	监视开关
锁定	lock	锁定
锁定	lock_wait	锁定

表 845. 事件监视信息

事件类型	逻辑数据分组	监视开关
锁定	-	始终收集
死锁 [1]	lock	始终收集
死锁 [1]	event_dlconn	始终收集
带有详细信息的死锁 [1]	event_detailed_dlconn	始终收集

1 建议不要使用此事件监视器。建议不要再使用此选项，将来的发行版可能会将其除去。请使用 CREATE EVENT MONITOR FOR LOCKING 语句来监视与锁定相关的事件，例如锁定超时、锁定等待和死锁。

用法

使用此元素来更好地理解死锁的原因。如果遇到涉及执行锁定升级的应用程序的死锁，那么您可能想要增加锁定内存量或更改任一应用程序可请求的锁定百分比。

lock_escals -“锁定升级次数”监视元素

锁定已从若干行锁定升级至表锁定的次数。

表 846. 表函数监视信息

表函数	监视元素收集级别
MON_FORMAT_XML_METRICS_BY_ROW - 获取所有度量值的基于已格式化行的输出	不适用；报告 XML 文档中作为格式化函数的输入提供的所有元素
MON_GET_ACTIVITY_DETAILS 表函数 - 获取完整的活动详细信息（在 XML 文档 DETAILS 中报告）	ACTIVITY METRICS BASE

表 846. 表函数监视信息 (续)

表函数	监视元素收集级别
MON_GET_CONNECTION 表函数 - 获取连接度量值	REQUEST METRICS BASE
MON_GET_CONNECTION_DETAILS 表函数 - 获取详细的连接度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE
MON_GET_PKG_CACHE_STMT 表函数 - 获取程序包高速缓存中的 SQL 语句活动度量值	ACTIVITY METRICS BASE
MON_GET_PKG_CACHE_STMT_DETAILS 表函数 - 获取程序包高速缓存条目的详细度量值	ACTIVITY METRICS BASE
MON_GET_SERVICE_SUBCLASS 表函数 - 获取服务子类度量值	REQUEST METRICS BASE
MON_GET_SERVICE_SUBCLASS_DETAILS 表函数 - 获取详细的服务子类度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE
MON_GET_TABLE 表函数 - 获取表度量值	DATA OBJECT METRICS EXTENDED
MON_GET_TABLE_USAGE_LIST 表函数 - 从表用法列表返回信息	DATA OBJECT METRICS EXTENDED
MON_GET_UNIT_OF_WORK 表函数 - 获取工作单元度量值	REQUEST METRICS BASE
MON_GET_UNIT_OF_WORK 表函数 - 获取详细的工作单元度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE
MON_GET_WORKLOAD 表函数 - 获取工作负载度量值	REQUEST METRICS BASE
MON_GET_WORKLOAD_DETAILS 表函数 - 获取详细的工作负载度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE

表 847. 快照监视信息

快照级别	逻辑数据分组	监视开关
数据库	dbase	基本
应用程序	appl	基本

可将快照监视的计数器重置。

表 848. 事件监视信息

事件类型	逻辑数据分组	监视开关
活动	event_activity（在 details_xml 文档中报告）	ACTIVITY METRICS BASE
活动	event_activitymetrics	ACTIVITY METRICS BASE
统计信息	event_scstats（在 details_xml 文档中报告）	REQUEST METRICS BASE
统计信息	event_wlstats（在 details_xml 文档中报告）	REQUEST METRICS BASE

表 848. 事件监视信息 (续)

事件类型	逻辑数据分组	监视开关
工作单元	在 system_metrics 文档中报告。	REQUEST METRICS BASE
数据库	event_db	始终收集
连接	event_conn	始终收集
事务	event_xact	始终收集
程序包高速缓存	在 activity_metrics 文档中报告。	ACTIVITY METRICS BASE

用法

当应用程序挂起的锁定总数达到可供应用程序使用的最大锁定列表空间量，或者所有应用程序消耗的锁定列表空间达到总锁定列表空间时，锁定将会升级。可用锁定列表空间量由 **maxlocks** 和 **locklist** 配置参数确定。

当应用程序达到允许的最大锁定数并且没有其他要升级的锁定时，它将使用锁定列表中为其他应用程序分配的空间。当整个锁定列表已满时，将发生错误。

此数据项包括所有锁定升级的计数（包括互斥锁定升级和 DB2 pureScale 环境中的升级）。为了确定仅 DB2 pureScale 环境中的锁定升级，请使用 **lock_escals_global** 监视元素。

以下几种原因可能会导致产生过量锁定升级:

- 锁定列表大小（**locklist**）对于并行应用程序数目而言可能太小
- 可供每个应用程序使用的锁定列表百分比（**maxlocks**）可能太小
- 一个或多个应用程序使用的锁定数可能过量。
- 在 DB2 pureScale 环境中，全局锁定列表大小 (**cf_lock_sz**) 可能太小。

要解决这些问题，可以:

- 增加 **locklist** 配置参数值。
- 增加 **maxlocks** 配置参数值。
- 确定执行大量锁定的应用程序，或者使用以下一个公式并将值与 **maxlocks** 进行比较来确定过多占用锁定列表的应用程序:
  - 在 64 位系统上，`(((locks held * 64) / (locklist * 4096)) * 100)`
  - 在 32 位系统上，`(((locks held * 48 / (locklist * 4096)) * 100)`

  这些应用程序还可能因为在锁定列表中使用过量资源而导致其他应用程序中发生锁定升级。这些应用程序可能需要使用表锁定来代替行锁定，尽管表锁定可能会导致 **lock_waits** 和 **lock_wait_time** 监视元素值增大。

lock_escals_global -“全局锁定升级数”监视元素

因为全局锁定内存使用达到 **cf_lock_sz** 数据库配置参数中指定的限制的锁定升级数。

表 849. 表函数监视信息

表函数	监视元素收集级别
MON_FORMAT_XML_METRICS_BY_ROW - 获取所有度量值的基于已格式化行的输出	不适用; 报告 XML 文档中作为格式化函数的输入提供的所有元素
MON_GET_ACTIVITY_DETAILS 表函数 - 获取完整的活动详细信息（在 XML 文档 DETAILS 中报告）	ACTIVITY METRICS BASE
MON_GET_CONNECTION 表函数 - 获取连接度量值	REQUEST METRICS BASE
MON_GET_CONNECTION_DETAILS 表函数 - 获取详细的连接度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE
MON_GET_PKG_CACHE_STMT 表函数 - 获取程序包高速缓存中的 SQL 语句活动度量值	ACTIVITY METRICS BASE
MON_GET_SERVICE_SUBCLASS 表函数 - 获取服务子类度量值	REQUEST METRICS BASE
MON_GET_SERVICE_SUBCLASS_DETAILS 表函数 - 获取详细的服务子类度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE
MON_GET_TABLE 表函数 - 获取表度量值	DATA OBJECT METRICS EXTENDED
MON_GET_TABLE_USAGE_LIST 表函数 - 从表用法列表返回信息	DATA OBJECT METRICS EXTENDED
MON_GET_UNIT_OF_WORK 表函数 - 获取工作单元度量值	REQUEST METRICS BASE
MON_GET_UNIT_OF_WORK 表函数 - 获取详细的工作单元度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE
MON_GET_WORKLOAD 表函数 - 获取工作负载度量值	REQUEST METRICS BASE
MON_GET_WORKLOAD_DETAILS 表函数 - 获取详细的工作负载度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE

表 850. 事件监视信息

事件类型	逻辑数据分组	监视开关
活动	event_activity（在 details_xml 文档中报告）	ACTIVITY METRICS BASE
活动	event_activitymetrics	ACTIVITY METRICS BASE
统计信息	event_scstats（在 details_xml 文档中报告）	REQUEST METRICS BASE
统计信息	event_wlstats（在 details_xml 文档中报告）	REQUEST METRICS BASE
工作单元	-	始终收集

表 850. 事件监视信息 (续)

事件类型	逻辑数据分组	监视开关
程序包高速缓存	-	始终收集
锁定	-	始终收集

用法

将此监视元素与 **lock_escals_maxlocks** 和 **lock_escals_locklist** 监视元素配合使用以确定导致数据库上发生升级的锁定空间配置参数。

lock_escals_locklist -“locklist 锁定升级数”监视元素

因为局部锁定内存使用达到 **locklist** 数据库配置参数中指定的限制的锁定升级数。

表 851. 表函数监视信息

表函数	监视元素收集级别
MON_FORMAT_XML_METRICS_BY_ROW - 获取所有度量值的基于已格式化行的输出	不适用; 报告 XML 文档中作为格式化函数的输入提供的所有元素
MON_GET_ACTIVITY_DETAILS 表函数 - 获取完整的活动详细信息（在 XML 文档 DETAILS 中报告）	ACTIVITY METRICS BASE
MON_GET_CONNECTION 表函数 - 获取连接度量值	REQUEST METRICS BASE
MON_GET_CONNECTION_DETAILS 表函数 - 获取详细的连接度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE
MON_GET_PKG_CACHE_STMT 表函数 - 获取程序包高速缓存中的 SQL 语句活动度量值	ACTIVITY METRICS BASE
MON_GET_SERVICE_SUBCLASS 表函数 - 获取服务子类度量值	REQUEST METRICS BASE
MON_GET_SERVICE_SUBCLASS_DETAILS 表函数 - 获取详细的服务子类度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE
MON_GET_UNIT_OF_WORK 表函数 - 获取工作单元度量值	REQUEST METRICS BASE
MON_GET_UNIT_OF_WORK 表函数 - 获取详细的工作单元度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE
MON_GET_WORKLOAD 表函数 - 获取工作负载度量值	REQUEST METRICS BASE
MON_GET_WORKLOAD_DETAILS 表函数 - 获取详细的工作负载度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE

表 852. 事件监视信息

事件类型	逻辑数据分组	监视开关
活动	event_activity（在 details_xml 文档中报告）	ACTIVITY METRICS BASE
活动	event_activitymetrics	ACTIVITY METRICS BASE
统计信息	event_scstats（在 details_xml 文档中报告）	REQUEST METRICS BASE
统计信息	event_wlstats（在 details_xml 文档中报告）	REQUEST METRICS BASE
工作单元	-	始终收集
程序包高速缓存	-	始终收集
锁定	-	始终收集

用法

将此监视元素与 **lock_escals_maxlocks** 和 **lock_escals_global** 监视元素配合使用以确定导致数据库上发生升级的锁定空间配置参数。

lock_escals_maxlocks -“maxlocks 锁定升级数”监视元素

因为局部锁定内存使用达到 **maxlocks** 数据库配置参数中指定的限制的锁定升级数。

表 853. 表函数监视信息

表函数	监视元素收集级别
MON_FORMAT_XML_METRICS_BY_ROW - 获取所有度量值的基于已格式化行的输出	不适用; 报告 XML 文档中作为格式化函数的输入提供的所有元素
MON_GET_ACTIVITY_DETAILS 表函数 - 获取完整的活动详细信息（在 XML 文档 DETAILS 中报告）	ACTIVITY METRICS BASE
MON_GET_CONNECTION 表函数 - 获取连接度量值	REQUEST METRICS BASE
MON_GET_CONNECTION_DETAILS 表函数 - 获取详细的连接度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE
MON_GET_PKG_CACHE_STMT 表函数 - 获取程序包高速缓存中的 SQL 语句活动度量值	ACTIVITY METRICS BASE
MON_GET_SERVICE_SUBCLASS 表函数 - 获取服务子类度量值	REQUEST METRICS BASE
MON_GET_SERVICE_SUBCLASS_DETAILS 表函数 - 获取详细的服务子类度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE
MON_GET_UNIT_OF_WORK 表函数 - 获取工作单元度量值	REQUEST METRICS BASE
MON_GET_UNIT_OF_WORK 表函数 - 获取详细的工作单元度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE

表 853. 表函数监视信息 (续)

表函数	监视元素收集级别
MON_GET_WORKLOAD 表函数 - 获取工作负载度量值	REQUEST METRICS BASE
MON_GET_WORKLOAD_DETAILS 表函数 - 获取详细的工作负载度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE

表 854. 事件监视信息

事件类型	逻辑数据分组	监视开关
活动	event_activity（在 details_xml 文档中报告）	ACTIVITY METRICS BASE
活动	event_activitymetrics	ACTIVITY METRICS BASE
统计信息	event_scstats（在 details_xml 文档中报告）	REQUEST METRICS BASE
统计信息	event_wlstats（在 details_xml 文档中报告）	REQUEST METRICS BASE
工作单元	-	始终收集
程序包高速缓存	-	始终收集
锁定	-	始终收集

用法

将此监视元素与 **lock_escals_locklist** 和 **lock_escals_global** 监视元素配合使用以确定导致数据库上发生升级的锁定空间配置参数。

lock_hold_count -“锁定挂起计数”监视元素

挂起锁定的次数。使用 WITH HOLD 子句和一些 DB2 实用程序注册的游标放置在锁定上的挂起数。落实事务时，不会释放带有挂起的锁定。

表 855. 快照监视信息

快照级别	逻辑数据分组	监视开关
锁定	lock	基本

表 856. 事件监视信息

事件类型	逻辑数据分组	监视开关
锁定	-	始终收集
死锁 [1]	lock	始终收集
死锁 [1]	event_dlconn	始终收集

1 建议不要使用此事件监视器。建议不要再使用此选项，将来的发行版可能会将其除去。请使用 CREATE EVENT MONITOR FOR LOCKING 语句来监视与锁定相关的事件，例如锁定超时、锁定等待和死锁。

lock_list_in_use -“正在使用的锁定列表内存总量”监视元素

正在使用的锁定列表内存总量（以字节计）。

表 857. 快照监视信息

快照级别	逻辑数据分组	监视开关
数据库	dbase	基本

用法

此元素可以与 **locklist** 配置参数配合使用以计算锁定列表利用率。如果锁定列表利用率很高，那么您可能要考虑增大该参数。

注: 在计算利用率时，注意下面这一点十分重要: **locklist** 配置参数是以 4KB/页为单位进行分配的，而此监视元素以字节为单位来提供结果。

lock_mode -“锁定方式”监视元素

正在挂起的锁定的类型。 如果不知道方式，那么此监视元素的值为 NULL。

表 858. 表函数监视信息

表函数	监视元素收集命令和级别
MON_GET_APPL_LOCKWAIT 表函数 - 获取有关应用程序正在等待的锁定的信息	ACTIVITY METRICS BASE
MON_GET_LOCKS 表函数 - 列示当前所连接的数据库中的所有锁定	ACTIVITY METRICS BASE

表 859. 快照监视信息

快照级别	逻辑数据分组	监视开关
应用程序	appl	锁定
锁定	lock	锁定
锁定	lock_wait	锁定

表 860. 事件监视信息

事件类型	逻辑数据分组	监视开关
锁定	-	-
死锁 [1]	lock	-
死锁 [1]	event_dlconn	-
带有详细信息的死锁 [1]	event_detailed_dlconn	-

1 建议不要使用此事件监视器。建议不要再使用此选项，将来的发行版可能会将其除去。请使用 CREATE EVENT MONITOR FOR LOCKING 语句来监视与锁定相关的事件，例如锁定超时、锁定等待和死锁。

用法

此方式可帮助您确定资源争用的源头。

根据要检查的监视器信息的类型，此元素指示下列其中一种情况:

- 另一应用程序针对此应用程序等待锁定的对象挂起的锁定类型（对于应用程序监视级别和死锁监视级别）。
- 此应用程序针对对象挂起的锁定类型（对于对象锁定级别）。

此字段的可能值包括:

方式	锁定类型	API 常量
	没有锁定	SQLM_LNON
IS	意向共享锁定	SQLM_LOIS
IX	意向互斥锁定	SQLM_LOIX
S	共享锁定	SQLM_LOOS
SIX	与意向互斥锁定共享	SQLM_LSIX
X	互斥锁定	SQLM_LOOX
IN	Intent None	SQLM_LOIN
Z	超级互斥锁定	SQLM_LOOZ
U	更新锁定	SQLM_LOOU
NS	扫描共享锁定	SQLM_LONS
NW	下一键弱互斥锁定	SQLM_LONW

lock_mode_requested -“请求的锁定方式”监视元素

等待获取锁定的应用程序请求的锁定方式。

表 861. 表函数监视信息

表函数	监视元素收集命令和级别
MON_GET_APPL_LOCKWAIT 表函数 - 获取有关应用程序正在等待的锁定的信息	ACTIVITY METRICS BASE

表 862. 快照监视信息

快照级别	逻辑数据分组	监视开关
锁定	lock_wait	锁定

表 863. 事件监视信息

事件类型	逻辑数据分组	监视开关
锁定	-	始终收集
死锁 [1]	event_dlconn	始终收集
带有详细信息的死锁 [1]	event_detailed_dlconn	始终收集

1 建议不要使用此事件监视器。建议不要再使用此选项，将来的发行版可能会将

其除去。请使用 CREATE EVENT MONITOR FOR LOCKING 语句来监视与锁定相关的事件，例如锁定超时、锁定等待和死锁。

用法

应用程序请求的锁定方式。此值可帮助您确定资源争用的源头。

下表列示了可能采用的锁定方式。

方式	锁定类型	API 常量
	没有锁定	SQLM_LNON
IS	意向共享锁定	SQLM_LOIS
IX	意向互斥锁定	SQLM_LOIX
S	共享锁定	SQLM_LOOS
SIX	与意向互斥锁定共享	SQLM_LSIX
X	互斥锁定	SQLM_LOOX
IN	Intent None	SQLM_LOIN
Z	超级互斥锁定	SQLM_LOOZ
U	更新锁定	SQLM_LOOU
NS	扫描共享锁定	SQLM_LONS
NW	下一键弱互斥锁定	SQLM_LONW

lock_name -“锁定名称”监视元素

内部二进制锁定名称。此元素将充当锁定的唯一标识。

表 864. 表函数监视信息

表函数	监视元素收集命令和级别
MON_GET_APPL_LOCKWAIT 表函数 - 获取有关应用程序正在等待的锁定的信息	ACTIVITY METRICS BASE
MON_GET_LOCKS 表函数 - 列示当前所连接的数据库中的所有锁定	ACTIVITY METRICS BASE

表 865. 快照监视信息

快照级别	逻辑数据分组	监视开关
锁定	lock	基本
锁定	lock_wait	lock_wait

表 866. 事件监视信息

事件类型	逻辑数据分组	监视开关
锁定	-	-
死锁 [1]	lock	-
死锁 [1]	event_dlconn	-

1 建议不要使用此事件监视器。建议不要再使用此选项，将来的发行版可能会将

其除去。请使用 CREATE EVENT MONITOR FOR LOCKING 语句来监视与锁定相关的事件，例如锁定超时、锁定等待和死锁。

用法

可以通过使用 MON_FORMAT_LOCK_NAME 例程获取有关锁定的更多详细信息，从而调整内部名称的格式。例如，如果这是表锁定，那么可以获得该锁定引用的表和表空间。

lock_node -“锁定节点”

锁定涉及的节点。

表 867. 快照监视信息

快照级别	逻辑数据分组	监视开关
应用程序	subsection	语句
死锁	event_dlconn	语句
带有详细信息的死锁	event_detailed_dlconn	语句

用法 它可以用于故障诊断。

lock_object_name -“锁定对象名称”

此元素仅供参考。它是应用程序对其挂起锁定的对象的名称（对于对象锁定级别信息），或者应用程序等待对其获取锁定的对象的名称（对于应用程序级别和死锁级别信息）。

注: 不推荐使用此监视元素。使用此监视元素不会生成错误。但是不会返回有效值。建议不要再使用此监视元素，将来的发行版中可能会将其除去。

表 868. 快照监视信息

快照级别	逻辑数据分组	监视开关
应用程序	appl	锁定
锁定	appl_lock_list	锁定
锁定	lock	基本

表 869. 事件监视信息

事件类型	逻辑数据分组	监视开关
死锁	lock	始终收集
死锁	event_dlconn	始终收集
带有详细信息的死锁	event_detailed_dlconn	始终收集

用法 对于表级别锁定，该对象名是 SMS 和 DMS 表空间的文件标识（FID）。对于行级别锁定，该对象名是行标识（RID）。对于表空间锁定，该对象名留为空白。对于缓冲池锁定，该对象名是缓冲池的名称。

要确定挂起锁定的表，请使用 *table_name* 和 *table_schema* 而不是文件标识，原因是文件标识可能不是唯一的。

要确定挂起锁定的表空间，请使用 *tablespace_name*。

lock_object_type -“等待的锁定对象类型”监视元素

应用程序对其挂起锁定的对象的类型（对于对象锁定级别信息），或者应用程序等待对其获取锁定的对象的类型（对于应用程序级别和死锁级别信息）。

表 870. 表函数监视信息

表函数	监视元素收集命令和级别
MON_FORMAT_LOCK_NAME 表函数 - 设置内部锁定名称的格式并返回详细信息	ACTIVITY METRICS BASE
MON_GET_APPL_LOCKWAIT 表函数 - 获取有关应用程序正在等待的锁定的信息	ACTIVITY METRICS BASE
MON_GET_LOCKS 表函数 - 列示当前所连接的数据库中的所有锁定	ACTIVITY METRICS BASE

表 871. 快照监视信息

快照级别	逻辑数据分组	监视开关
应用程序	appl	锁定
锁定	appl_lock_list	锁定
锁定	lock	基本
锁定	lock_wait	锁定

表 872. 事件监视信息

事件类型	逻辑数据分组	监视开关
锁定	-	-
死锁 [1]	lock	-
死锁 [1]	event_dlconn	-
带有详细信息的死锁 [1]	event_detailed_dlconn	-

1 建议不要使用此事件监视器。建议不要再使用此选项，将来的发行版可能会将其除去。请使用 CREATE EVENT MONITOR FOR LOCKING 语句来监视与锁定相关的事件，例如锁定超时、锁定等待和死锁。

用法

此元素可帮助您确定资源争用的源头。

对于快照监视和死锁 [1]事件监视器，对象类型标识是在 `sqlmon.h` 中定义的。对象可以是下列其中一种类型:

- 表空间（sqlmon.h 中的 SQLM_TABLESPACE_LOCK）
- 表
- 缓冲池
- 块
- 记录（或行）
- 数据分区（sqlmon.h 中的 SQLM_TABLE_PART_LOCK）

- 内部（数据库管理器内部挂起的另一类型的锁定）
- 自动调整大小
- 自动存储器。

对于表 1 中的锁定事件监视器和监视表函数，**lock_object_type** 监视元素的可能值是在表 4 中定义的。

表 873. lock_object_type 监视元素的可能值

可能值	描述
TABLE	表锁定
ROW	行锁定
TABLESPACE	表空间锁定
EOT	表锁定结束
KEYVALUE	键值锁定
SYSBOOT	Sysboot 锁定
PLAN	方案锁定
VARIATION	变体锁定
SEQUENCE	序列锁定
BUFFERPOOL	缓冲池锁定
LOB	LOB/Long 区域锁定
CATALOG	目录高速缓存锁定
ONLINE_BACKUP	联机备份锁定
OBJECT_TABLE	对象表锁定
ALTER_TABLE	表改变锁定
DMS_SEQUENCE	DMS 序列锁定
REORG	现场重组锁定
MDC_BLOCK	MDC 块锁定
TABLE_PARTITION	表分区锁定
AUTORESIZE	自动调整大小锁定
AUTOSTORAGE	自动存储器锁定
XMLPATH	XML 路径锁定
EXTENT_MOVEMENT	扩展数据块移动锁定
WORKLOAD	工作负载权限锁定
FED_SERVER	联合服务器锁定
FED_USER	联合用户映射锁定
CHUNK	大块锁定
LOAD_PRE_PART	装入表预分区锁定
LOAD_PART	装入表分区锁定
LOAD_TS	装入表空间锁定
LONG_FIELD_ESC	长字段升级锁定
LONG_FIELD_SPACE	长字段伙伴空间锁定

lock_release_flags -“锁定释放标志”监视元素

锁定释放标志。

表 874. 表函数监视信息

表函数	监视元素收集级别
MON_GET_LOCKS 表函数 - 列示当前所连接的数据库中的所有锁定	ACTIVITY METRICS BASE

表 875. 快照监视信息

快照级别	逻辑数据分组	监视开关
锁定	lock	基本
锁定	lock_wait	基本

表 876. 事件监视信息

事件类型	逻辑数据分组	监视开关
锁定	-	始终收集
死锁 [1]	lock	始终收集
死锁 [1]	event_dlconn	始终收集

1 建议不要使用此事件监视器。建议不要再使用此选项，将来的发行版可能会将其除去。请使用 CREATE EVENT MONITOR FOR LOCKING 语句来监视与锁定相关的事件，例如锁定超时、锁定等待和死锁。

用法

下表列示所有可能的释放标志设置。每个释放标志都以 `sqlmon.h` 中定义的位标志值为基础。

API 常量	描述
SQLM_LOCKRELFLAGS_SQLCOMPILER	SQL 编译器进行的锁定。
SQLM_LOCKRELFLAGS_UNTRACKED	非唯一的未记录锁定。

注: 所有非指定位将用于应用程序游标。

lock_status -“锁定状态”监视元素

指示锁定的内部状态。

表 877. 表函数监视信息

表函数	监视元素收集命令和级别
MON_GET_APPL_LOCKWAIT 表函数 - 获取有关应用程序正在等待的锁定的信息	ACTIVITY METRICS BASE
MON_GET_LOCKS 表函数 - 列示当前所连接的数据库中的所有锁定	ACTIVITY METRICS BASE

表 878. 快照监视信息

快照级别	逻辑数据分组	监视开关
锁定	lock	基本

表 879. 事件监视信息

事件类型	逻辑数据分组	监视开关
锁定	-	始终收集
死锁 [1]	lock	始终收集

1 建议不要使用此事件监视器。建议不要再使用此选项，将来的发行版可能会将其除去。请使用 CREATE EVENT MONITOR FOR LOCKING 语句来监视与锁定相关的事件，例如锁定超时、锁定等待和死锁。

用法

此元素可帮助说明应用程序等待获取对某个对象的锁定时所发生的情况。虽然可能已经具有对所需对象的锁定，但应用程序必须等待以获取对同一对象的另一类型的锁定。

锁定可以是下列其中一种状态:

G 已授予状态: 应用程序的锁定处于 **lock_mode** 监视元素所指定状态的锁定。

C 正在转换状态: 应用程序正在尝试将挂起的锁定更改为另一类型；例如，从共享锁定更改为互斥锁定。

W 正在等待状态。

注: API 用户应参考包含数据库系统监视器常量定义的 `sqlmon.h` 头文件。

lock_timeout_val -“锁定超时值”监视元素

指示应用程序发出 SET CURRENT LOCK TIMEOUT 语句时的超时值（以秒计）。如果未执行语句，那么将显示数据库级别锁定超时。

表 880. 快照监视信息

快照级别	逻辑数据分组	监视开关
应用程序	appl	基本
应用程序	agent	基本

表 881. 事件监视信息

事件类型	逻辑数据分组	监视开关
锁定	-	-

用法

可使用 SET CURRENT LOCK TIMEOUT 语句来指定应用程序代理程序等待表或索引锁定的最长持续时间。

如果应用程序等待锁定的时间过长，那么可以检查 **lock_timeout_val** 监视元素值以了解该值在应用程序中是否设置得过高。如果符合应用程序逻辑，那么可以修改应用程序以降低锁定超时值，从而允许应用程序超时。可使用 SET CURRENT LOCK TIMEOUT 语句来完成此修改。

如果应用程序经常超时，那么请检查锁定超时值是否设置得过低，并在适当时增大该值。

lock_timeouts -“锁定超时次数”监视元素

要锁定对象的请求由于发生超时而未获得锁定的次数。

表 882. 表函数监视信息

表函数	监视元素收集级别
MON_FORMAT_XML_METRICS_BY_ROW - 获取所有度量值的基于已格式化行的输出	不适用; 报告 XML 文档中作为格式化函数的输入提供的所有元素
MON_GET_ACTIVITY_DETAILS 表函数 - 获取完整的活动详细信息（在 XML 文档 DETAILS 中报告）	ACTIVITY METRICS BASE
MON_GET_CONNECTION 表函数 - 获取连接度量值	REQUEST METRICS BASE
MON_GET_CONNECTION_DETAILS 表函数 - 获取详细的连接度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE
MON_GET_PKG_CACHE_STMT 表函数 - 获取程序包高速缓存中的 SQL 语句活动度量值	ACTIVITY METRICS BASE
MON_GET_PKG_CACHE_STMT_DETAILS 表函数 - 获取程序包高速缓存条目的详细度量值	ACTIVITY METRICS BASE
MON_GET_SERVICE_SUBCLASS 表函数 - 获取服务子类度量值	REQUEST METRICS BASE
MON_GET_SERVICE_SUBCLASS_DETAILS 表函数 - 获取详细的服务子类度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE
MON_GET_UNIT_OF_WORK 表函数 - 获取工作单元度量值	REQUEST METRICS BASE
MON_GET_UNIT_OF_WORK 表函数 - 获取详细的工作单元度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE
MON_GET_WORKLOAD 表函数 - 获取工作负载度量值	REQUEST METRICS BASE
MON_GET_WORKLOAD_DETAILS 表函数 - 获取详细的工作负载度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE

表 883. 快照监视信息

快照级别	逻辑数据分组	监视开关
数据库	dbase	基本

表 883. 快照监视信息 (续)

快照级别	逻辑数据分组	监视开关
应用程序	appl	基本

可将快照监视的计数器重置。

表 884. 事件监视信息

事件类型	逻辑数据分组	监视开关
活动	event_activity（在 details_xml 文档中报告）	ACTIVITY METRICS BASE
活动	event_activitymetrics	ACTIVITY METRICS BASE
统计信息	event_scstats（在 details_xml 文档中报告）	REQUEST METRICS BASE
统计信息	event_wlstats（在 details_xml 文档中报告）	REQUEST METRICS BASE
工作单元	在 system_metrics 文档中报告。	REQUEST METRICS BASE
数据库	event_db	始终收集
连接	event_conn	始终收集
程序包高速缓存	在 activity_metrics 文档中报告。	ACTIVITY METRICS BASE

用法

此元素可帮助您调整 **locktimeout** 数据库配置参数的设置。如果锁定超时次数与正常操作水平相比显得过多，那么可能表明应用程序挂起锁定的时间过长。在这种情况下，此元素可能指示应该分析其他某些锁定和死锁监视元素，以确定是否存在应用程序问题。

如果 **locktimeout** 数据库配置参数设置得过大，那么锁定超时情况将会过少。在这种情况下，应用程序在获取锁定前的等待时间将会过长。

lock_timeouts_global -“锁定超时全局”监视元素

锁定超时数，持有此锁定的应用程序在远程成员上。

表 885. 表函数监视信息

表函数	监视元素收集级别
MON_FORMAT_XML_METRICS_BY_ROW - 获取所有度量值的基于已格式化行的输出	不适用；报告 XML 文档中作为格式化函数的输入提供的所有元素
MON_GET_ACTIVITY_DETAILS 表函数 - 获取完整的活动详细信息（在 XML 文档 DETAILS 中报告）	ACTIVITY METRICS BASE
MON_GET_CONNECTION 表函数 - 获取连接度量值	REQUEST METRICS BASE
MON_GET_CONNECTION_DETAILS 表函数 - 获取详细的连接度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE

表 885. 表函数监视信息 (续)

表函数	监视元素收集级别
MON_GET_PKG_CACHE_STMT 表函数 - 获取程序包高速缓存中的 SQL 语句活动度量值	ACTIVITY METRICS BASE
MON_GET_SERVICE_SUBCLASS 表函数 - 获取服务子类度量值	REQUEST METRICS BASE
MON_GET_SERVICE_SUBCLASS_DETAILS 表函数 - 获取详细的服务子类度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE
MON_GET_UNIT_OF_WORK 表函数 - 获取工作单元度量值	REQUEST METRICS BASE
MON_GET_UNIT_OF_WORK 表函数 - 获取详细的工作单元度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE
MON_GET_WORKLOAD 表函数 - 获取工作负载度量值	REQUEST METRICS BASE
MON_GET_WORKLOAD_DETAILS 表函数 - 获取详细的工作负载度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE

表 886. 事件监视信息

事件类型	逻辑数据分组	监视开关
活动	event_activity（在 details_xml 文档中报告）	ACTIVITY METRICS BASE
活动	event_activitymetrics	ACTIVITY METRICS BASE
统计信息	event_scstats（在 details_xml 文档中报告）	REQUEST METRICS BASE
统计信息	event_wlstats（在 details_xml 文档中报告）	REQUEST METRICS BASE
工作单元	-	始终收集
程序包高速缓存	-	始终收集
锁定	-	始终收集

用法

将此元素与 **lock_timeouts** 监视元素一起使用。**lock_timeouts_global** 监视元素表示等待获取在另一成员上持有的锁定期间出现锁定超时的次数。要确定等待获取在同一成员上持有的锁定期间发生的锁定超时次数，请使用以下公式:

```
lock_timeouts - lock_timeouts_global
```

在 DB2 pureScale 环境外部，此值始终为零。

lock_wait_end_time -“锁定等待结束时间戳记”监视元素

应用程序停止等待获取对当前锁定的对象的锁定的日期和时间。

表 887. 事件监视信息

事件类型	逻辑数据分组	监视开关
锁定	lock_participants	

lock_wait_start_time -“锁定等待开始时间戳记”监视元素

此应用程序开始等待获取锁定的日期和时间，该锁定针对当前被另一应用程序锁定的对象。

表 888. 表函数监视信息

表函数	监视元素收集级别
MON_GET_APPL_LOCKWAIT 表函数 - 获取有关应用程序正在等待的锁定的信息	ACTIVITY METRICS BASE

表 889. 快照监视信息

快照级别	逻辑数据分组	监视开关
应用程序	appl	锁定，时间戳记
锁定	lock_wait	锁定，时间戳记

表 890. 事件监视信息

事件类型	逻辑数据分组	监视开关
锁定	-	始终收集
死锁 [1]	event_dlconn	时间戳记
带有详细信息的死锁 [1]	event_detailed_dlconn	时间戳记

1 建议不要使用此事件监视器。建议不要再使用此选项，将来的发行版可能会将其除去。请使用 CREATE EVENT MONITOR FOR LOCKING 语句来监视与锁定相关的事件，例如锁定超时、锁定等待和死锁。

用法

此元素可帮助您确定资源争用的严重性。

lock_wait_time -“等待锁定时间”监视元素

等待锁定的耗用时间总计。此值以毫秒计。

表 891. 表函数监视信息

表函数	监视元素收集级别
MON_FORMAT_XML_METRICS_BY_ROW - 获取所有度量值的基于已格式化行的输出	不适用；报告 XML 文档中作为格式化函数的输入提供的所有元素

表 891. 表函数监视信息 (续)

表函数	监视元素收集级别
MON_FORMAT_XML_TIMES_BY_ROW - - 获取基于已格式化行的组合层次结构等待时间和处理时间	不适用; 报告 XML 文档中作为格式化函数的输入提供的所有元素
MON_FORMAT_XML_WAIT_TIMES_BY _ROW - 获取等待时间的已格式化的基于行的输出	不适用; 报告 XML 文档中作为格式化函数的输入提供的所有元素
MON_GET_ACTIVITY_DETAILS 表函数 - 获取完整的活动详细信息（在 XML 文档 DETAILS 中报告）	ACTIVITY METRICS BASE
MON_GET_CONNECTION 表函数 - 获取连接度量值	REQUEST METRICS BASE
MON_GET_CONNECTION_DETAILS 表函数 - 获取详细的连接度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE
MON_GET_PKG_CACHE_STMT 表函数 - 获取程序包高速缓存中的 SQL 语句活动度量值	ACTIVITY METRICS BASE
MON_GET_PKG_CACHE_STMT_DETAILS 表函数 - 获取程序包高速缓存条目的详细度量值	ACTIVITY METRICS BASE
MON_GET_SERVICE_SUBCLASS 表函数 - 获取服务子类度量值	REQUEST METRICS BASE
MON_GET_SERVICE_SUBCLASS_DETAILS 表函数 - 获取详细的服务子类度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE
MON_GET_TABLE 表函数 - 获取表度量值	DATA OBJECT METRICS EXTENDED
MON_GET_TABLE_USAGE_LIST 表函数 - 从表用法列表返回信息	DATA OBJECT METRICS EXTENDED
MON_GET_UNIT_OF_WORK 表函数 - 获取工作单元度量值	REQUEST METRICS BASE
MON_GET_UNIT_OF_WORK 表函数 - 获取详细的工作单元度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE
MON_GET_WORKLOAD 表函数 - 获取工作负载度量值	REQUEST METRICS BASE
MON_GET_WORKLOAD_DETAILS 表函数 - 获取详细的工作负载度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE

表 892. 快照监视信息

快照级别	逻辑数据分组	监视开关
数据库	dbase	锁定
应用程序	appl	锁定
锁定	appl_lock_list	appl_lock_list

可将快照监视的计数器重置。

表 893. 事件监视信息

事件类型	逻辑数据分组	监视开关
活动	event_activity（在 details_xml 文档中报告）	ACTIVITY METRICS BASE
活动	event_activitymetrics	ACTIVITY METRICS BASE
统计信息	event_scstats（在 details_xml 文档中报告）	REQUEST METRICS BASE
统计信息	event_wlstats（在 details_xml 文档中报告）	REQUEST METRICS BASE
工作单元	在 system_metrics 文档中报告。	REQUEST METRICS BASE
数据库	event_db	始终收集
连接	event_conn	始终收集
事务	event_xact	始终收集
程序包高速缓存	在 activity_metrics 文档中报告。	ACTIVITY METRICS BASE

用法

在数据库级别，此项表示所有应用程序在此数据库中等待锁定的总次数。 此耗用时间度量可能包括活动期间获取锁定所耗的时间以及其他处理（例如，编译）期间获取锁定所耗的时间。

在应用程序连接和事务级别，此项表示此连接或事务等待授予锁定的耗用时间总计。

此元素的值不包括当前仍处于锁定等待状态的代理程序的锁定等待时间。它仅包括已完成锁定等待的代理程序的锁定等待时间。

此元素可与 **lock_waits** 监视元素一起使用以计算锁定的平均等待时间。此计算可在数据库级别或应用程序连接级别执行。

使用提供耗用时间的监视元素时，应考虑:

- 耗用时间受系统负载影响，所以运行的进程越多，此耗用时间值越高。
- 要在数据库级别计算此元素，数据库系统监视器会将应用程序级别时间求和。这会导致对数据库级别的耗用时间双倍计数，原因是多个应用程序进程可能同时运行。

  为提供有意义的数据，可计算锁定的平均等待时间，如上所述。

lock_wait_time_global -“锁定等待时间全局”监视元素

在全局锁定等待上所耗的时间。时间的度量单位为毫秒。

表 894. 表函数监视信息

表函数	监视元素收集级别
MON_FORMAT_XML_METRICS_BY_ROW - 获取所有度量值的基于已格式化行的输出	不适用；报告 XML 文档中作为格式化函数的输入提供的所有元素
MON_FORMAT_XML_TIMES_BY_ROW - - 获取基于已格式化行的组合层次结构等待时间和处理时间	不适用；报告 XML 文档中作为格式化函数的输入提供的所有元素

表 894. 表函数监视信息 (续)

表函数	监视元素收集级别
MON_FORMAT_XML_WAIT_TIMES_BY_ROW - 获取等待时间的已格式化的基于行的输出	不适用; 报告 XML 文档中作为格式化函数的输入提供的所有元素
MON_GET_ACTIVITY_DETAILS 表函数 - 获取完整的活动详细信息（在 XML 文档 DETAILS 中报告）	ACTIVITY METRICS BASE
MON_GET_CONNECTION 表函数 - 获取连接度量值	REQUEST METRICS BASE
MON_GET_CONNECTION_DETAILS 表函数 - 获取详细的连接度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE
MON_GET_PKG_CACHE_STMT 表函数 - 获取程序包高速缓存中的 SQL 语句活动度量值	ACTIVITY METRICS BASE
MON_GET_SERVICE_SUBCLASS 表函数 - 获取服务子类度量值	REQUEST METRICS BASE
MON_GET_SERVICE_SUBCLASS_DETAILS 表函数 - 获取详细的服务子类度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE
MON_GET_TABLE 表函数 - 获取表度量值	DATA OBJECT METRICS EXTENDED
MON_GET_TABLE_USAGE_LIST 表函数 - 从表用法列表返回信息	DATA OBJECT METRICS EXTENDED
MON_GET_UNIT_OF_WORK 表函数 - 获取工作单元度量值	REQUEST METRICS BASE
MON_GET_UNIT_OF_WORK 表函数 - 获取详细的工作单元度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE
MON_GET_WORKLOAD 表函数 - 获取工作负载度量值	REQUEST METRICS BASE
MON_GET_WORKLOAD_DETAILS 表函数 - 获取详细的工作负载度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE

表 895. 事件监视信息

事件类型	逻辑数据分组	监视开关
活动	event_activity（在 details_xml 文档中报告）	ACTIVITY METRICS BASE
活动	event_activitymetrics	ACTIVITY METRICS BASE
统计信息	event_scstats（在 details_xml 文档中报告）	REQUEST METRICS BASE
统计信息	event_wlstats（在 details_xml 文档中报告）	REQUEST METRICS BASE
工作单元	-	始终收集
程序包高速缓存	-	始终收集
锁定	-	始终收集

用法

将此监视元素与 **lock_wait_time** 监视元素一起使用，后者表示等待锁定所耗的所有时间。**lock_wait_time_global** 监视元素表示等待由不同成员上的冲突应用程序持有的锁定所耗的时间。要确定等待由同一成员上的冲突应用程序持有的锁定所耗的总时间，请使用以下公式:

```
lock_wait_time - lock_wait_time_global
```

在 DB2 pureScale 环境外部，此值始终为零。

lock_wait_time_global_top -“最长全局锁定等待时间”监视元素

在另一成员上挂起的锁定的最长锁定等待。此值以毫秒计。

表 896. 事件监视信息

事件类型	逻辑数据分组	监视开关
统计信息	event_wlstats	始终收集

lock_wait_time_top -“最长锁定等待时间”监视元素

工作负载中任何请求的锁定等待时间的高水位标记。单位为毫秒。系统始终对工作负载收集 lock_wait_time_top 高水位标记。仅当启用了请求度量值时，请求才会影响此高水位标记。

表 897. 事件监视信息

事件类型	逻辑数据分组	监视开关
统计信息	event_wlstats	始终收集

用法

使用此元素来确定在收集时间间隔内，分区中某个工作负载的任何请求的最大锁定等待时间。

lock_wait_val -“锁定等待值”监视元素

生成 mon_lockwait 的事件之前等待锁定时耗用的时间（以毫秒计）。

表 898. 事件监视信息

事件类型	逻辑数据分组	监视开关
锁定	lock_participants	

lock_waits -“等待锁定次数”监视元素

应用程序或连接等待锁定的总次数。

表 899. 表函数监视信息

表函数	监视元素收集级别
MON_FORMAT_XML_METRICS_BY_ROW - 获取所有度量值的基于已格式化行的输出	不适用; 报告 XML 文档中作为格式化函数的输入提供的所有元素
MON_GET_ACTIVITY_DETAILS 表函数 - 获取完整的活动详细信息（在 XML 文档 DETAILS 中报告）	ACTIVITY METRICS BASE
MON_GET_CONNECTION 表函数 - 获取连接度量值	REQUEST METRICS BASE
MON_GET_CONNECTION_DETAILS 表函数 - 获取详细的连接度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE
MON_GET_PKG_CACHE_STMT 表函数 - 获取程序包高速缓存中的 SQL 语句活动度量值	ACTIVITY METRICS BASE
MON_GET_PKG_CACHE_STMT_DETAILS 表函数 - 获取程序包高速缓存条目的详细度量值	ACTIVITY METRICS BASE
MON_GET_SERVICE_SUBCLASS 表函数 - 获取服务子类度量值	REQUEST METRICS BASE
MON_GET_SERVICE_SUBCLASS_DETAILS 表函数 - 获取详细的服务子类度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE
MON_GET_TABLE 表函数 - 获取表度量值	DATA OBJECT METRICS EXTENDED
MON_GET_TABLE_USAGE_LIST 表函数 - 从表用法列表返回信息	DATA OBJECT METRICS EXTENDED
MON_GET_UNIT_OF_WORK 表函数 - 获取工作单元度量值	REQUEST METRICS BASE
MON_GET_UNIT_OF_WORK 表函数 - 获取详细的工作单元度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE
MON_GET_WORKLOAD 表函数 - 获取工作负载度量值	REQUEST METRICS BASE
MON_GET_WORKLOAD_DETAILS 表函数 - 获取详细的工作负载度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE

表 900. 快照监视信息

快照级别	逻辑数据分组	监视开关
数据库	dbase	基本
应用程序	appl	基本

可将快照监视的计数器重置。

表 901. 事件监视信息

事件类型	逻辑数据分组	监视开关
活动	event_activity（在 details_xml 文档中报告）	ACTIVITY METRICS BASE
活动	event_activitymetrics	ACTIVITY METRICS BASE
统计信息	event_scstats（在 details_xml 文档中报告）	REQUEST METRICS BASE
统计信息	event_wlstats（在 details_xml 文档中报告）	REQUEST METRICS BASE
工作单元	在 system_metrics 文档中报告。	REQUEST METRICS BASE
数据库	event_db	始终收集
连接	event_conn	始终收集
程序包高速缓存	在 activity_metrics 文档中报告。	ACTIVITY METRICS BASE

用法

在数据库级别，此项表示应用程序在数据库中等待锁定的总次数。

在应用程序连接级别，此项表示此连接请求锁定但必须等待（因为另一连接已经挂起针对该数据的锁定）的总次数。

此元素可与 **lock_wait_time** 配合使用以在数据库级别计算锁定的平均等待时间。此计算可在数据库级别或应用程序连接级别完成。

如果平均锁定等待时间很长，那么应查找挂起许多锁定或具有锁定升级的应用程序，并把重点放在调整应用程序以改进并行性上（如果适当）。如果平均锁定等待时间很长是升级导致的，那么 **locklist** 和/或 **maxlocks** 配置参数的值可能太低了。

lock_waits_global -“锁定等待全局”监视元素

因为持有针对远程成员的锁定的应用程序而导致的锁定等待数。

表 902. 表函数监视信息

表函数	监视元素收集级别
MON_FORMAT_XML_METRICS_BY_ROW - 获取所有度量值的基于已格式化行的输出	不适用；报告 XML 文档中作为格式化函数的输入提供的所有元素
MON_GET_ACTIVITY_DETAILS 表函数 - 获取完整的活动详细信息（在 XML 文档 DETAILS 中报告）	ACTIVITY METRICS BASE
MON_GET_CONNECTION 表函数 - 获取连接度量值	REQUEST METRICS BASE
MON_GET_CONNECTION_DETAILS 表函数 - 获取详细的连接度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE
MON_GET_PKG_CACHE_STMT 表函数 - 获取程序包高速缓存中的 SQL 语句活动度量值	ACTIVITY METRICS BASE

表 902. 表函数监视信息 (续)

表函数	监视元素收集级别
MON_GET_SERVICE_SUBCLASS 表函数 - 获取服务子类度量值	REQUEST METRICS BASE
MON_GET_SERVICE_SUBCLASS_DETAILS 表函数 - 获取详细的服务子类度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE
MON_GET_TABLE 表函数 - 获取表度量值	DATA OBJECT METRICS EXTENDED
MON_GET_TABLE_USAGE_LIST 表函数 - 从表用法列表返回信息	DATA OBJECT METRICS EXTENDED
MON_GET_UNIT_OF_WORK 表函数 - 获取工作单元度量值	REQUEST METRICS BASE
MON_GET_UNIT_OF_WORK 表函数 - 获取详细的工作单元度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE
MON_GET_WORKLOAD 表函数 - 获取工作负载度量值	REQUEST METRICS BASE
MON_GET_WORKLOAD_DETAILS 表函数 - 获取详细的工作负载度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE

表 903. 事件监视信息

事件类型	逻辑数据分组	监视开关
活动	event_activity（在 details_xml 文档中报告）	ACTIVITY METRICS BASE
活动	event_activitymetrics	ACTIVITY METRICS BASE
统计信息	event_scstats（在 details_xml 文档中报告）	REQUEST METRICS BASE
统计信息	event_wlstats（在 details_xml 文档中报告）	REQUEST METRICS BASE
工作单元	-	始终收集
程序包高速缓存	-	始终收集
锁定	-	始终收集

用法

将此监视元素与 **lock_waits** 监视元素配合使用，后者报告因为由所有成员上的冲突应用程序持有的锁定而导致的锁定等待总数。**lock_waits_global** 监视元素指示由不同成员上的冲突应用程序持有的锁定等待的次数。要确定由等待应用程序所在成员上的冲突应用程序持有的锁定等待数，请使用以下公式:

```
lock_waits - lock_waits_global
```

在 DB2 pureScale 环境外部，此值始终为零。

locks_held -“挂起的锁定数”监视元素

当前挂起的锁定数目。

表 904. 表函数监视信息

表函数	监视元素收集级别
MON_GET_CONNECTION 表函数 - 获取连接度量值	ACTIVITY METRICS BASE
MON_GET_CONNECTION_DETAILS 表函数 - 获取详细的连接度量值（在 XML 文档 DETAILS 中报告）	ACTIVITY METRICS BASE
MON_GET_UNIT_OF_WORK 表函数 - 获取工作单元度量值	ACTIVITY METRICS BASE
MON_GET_UNIT_OF_WORK 表函数 - 获取详细的工作单元度量值（在 XML 文档 DETAILS 中报告）	ACTIVITY METRICS BASE

表 905. 快照监视信息

快照级别	逻辑数据分组	监视开关
数据库	dbase	基本
应用程序	appl	基本
锁定	db_lock_list	基本
锁定	appl_lock_list	基本

表 906. 事件监视信息

事件类型	逻辑数据分组	监视开关
带有详细信息的死锁	event_detailed_dlconn	始终收集

用法

如果监视器信息为数据库级别，那么此项表示数据库中的所有应用程序当前挂起的总锁定数。

如果监视器信息为应用程序级别，那么此项表示应用程序的所有代理程序当前挂起的总锁定数。

locks_held_top -“挂起的最大锁定数”监视元素

此事务期间挂起的最大锁定数。

表 907. 事件监视信息

事件类型	逻辑数据分组	监视开关
事务	event_xact	始终收集

用法

可使用此元素来确定应用程序是否达到 **maxlocks** 配置参数定义的最大可用锁定数。此参数指示发生锁定升级之前每个应用程序可以使用的锁定列表百分比。锁定升级可能导致连接至数据库的应用程序之间的并行度下降。

因为 **maxlocks** 参数被指定为百分比并且此元素为计数器，所以可以借助以下一个公式进行计算以将此元素提供的计数与应用程序可挂起的总锁定数进行比较:

- 在 64 位系统上，`(locklist * 4096 / 64 ) * (maxlocks / 100)`
- 在 32 位系统上，`(locklist * 4096 / 48 ) * (maxlocks / 100)`

如果您有大量锁定，那么可能需要在应用程序中执行更多落实操作以便可以释放一些锁定。

locks_in_list -“报告的锁定数”

事件监视器要报告的特定应用程序挂起的锁定数。

表 908. 事件监视信息

事件类型	逻辑数据分组	监视开关
带有详细信息的死锁	event_detailed_dlconn	始终收集

locks_waiting -“当前正在等待锁定的代理程序数”监视元素

指示正在等待锁定的代理程序数。

表 909. 快照监视信息

快照级别	逻辑数据分组	监视开关
数据库	dbase	基本
应用程序	appl	基本
锁定	db_lock_list	基本

用法

与 **appls_cur_cons** 一起使用时，此元素指示正在等待锁定的应用程序所占的百分比。如果此数目很大，那么应用程序可能存在并行性问题，您应当指定长时间挂起锁定或互斥锁定的应用程序。

log_buffer_wait_time -“日志缓冲区等待时间”监视元素

代理程序等待日志缓冲区空间时耗用的时间。此值以毫秒计。

表 910. 表函数监视信息

表函数	监视元素收集级别
MON_FORMAT_XML_METRICS_BY_ROW - 获取所有度量值的基于已格式化行的输出	不适用；报告 XML 文档中作为格式化函数的输入提供的所有元素

表 910. 表函数监视信息 (续)

表函数	监视元素收集级别
MON_FORMAT_XML_TIMES_BY_ROW - - 获取基于已格式化行的组合层次结构等待时间和处理时间	不适用; 报告 XML 文档中作为格式化函数的输入提供的所有元素
MON_FORMAT_XML_WAIT_TIMES_BY _ROW - 获取等待时间的已格式化的基于行的输出	不适用; 报告 XML 文档中作为格式化函数的输入提供的所有元素
MON_GET_ACTIVITY_DETAILS 表函数 - 获取完整的活动详细信息（在 XML 文档 DETAILS 中报告）	ACTIVITY METRICS BASE
MON_GET_CONNECTION 表函数 - 获取连接度量值	REQUEST METRICS BASE
MON_GET_CONNECTION_DETAILS 表函数 - 获取详细的连接度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE
MON_GET_PKG_CACHE_STMT 表函数 - 获取程序包高速缓存中的 SQL 语句活动度量值	ACTIVITY METRICS BASE
MON_GET_PKG_CACHE_STMT_DETAILS 表函数 - 获取程序包高速缓存条目的详细度量值	ACTIVITY METRICS BASE
MON_GET_SERVICE_SUBCLASS 表函数 - 获取服务子类度量值	REQUEST METRICS BASE
MON_GET_SERVICE_SUBCLASS_DETAILS 表函数 - 获取详细的服务子类度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE
MON_GET_UNIT_OF_WORK 表函数 - 获取工作单元度量值	REQUEST METRICS BASE
MON_GET_UNIT_OF_WORK 表函数 - 获取详细的工作单元度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE
MON_GET_WORKLOAD 表函数 - 获取工作负载度量值	REQUEST METRICS BASE
MON_GET_WORKLOAD_DETAILS 表函数 - 获取详细的工作负载度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE

表 911. 事件监视信息

事件类型	逻辑数据分组	监视开关
活动	event_activity（在 details_xml 文档中报告）	ACTIVITY METRICS BASE
活动	event_activitymetrics	ACTIVITY METRICS BASE
统计信息	event_scstats（在 details_xml 文档中报告）	REQUEST METRICS BASE
统计信息	event_wlstats（在 details_xml 文档中报告）	REQUEST METRICS BASE
工作单元	在 system_metrics 文档中报告。	REQUEST METRICS BASE

表 911. 事件监视信息 (续)

事件类型	逻辑数据分组	监视开关
程序包高速缓存	在 activity_metrics 文档中报告。	ACTIVITY METRICS BASE

log_disk_wait_time -“日志磁盘等待时间”监视元素

代理程序等待日志记录被清仓到磁盘时耗用的时间。此值以毫秒计。

表 912. 表函数监视信息

表函数	监视元素收集级别
MON_FORMAT_XML_METRICS_BY_ROW - 获取所有度量值的基于已格式化行的输出	不适用; 报告 XML 文档中作为格式化函数的输入提供的所有元素
MON_FORMAT_XML_TIMES_BY_ROW - - 获取基于已格式化行的组合层次结构等待时间和处理时间	不适用; 报告 XML 文档中作为格式化函数的输入提供的所有元素
MON_FORMAT_XML_WAIT_TIMES_BY _ROW - 获取等待时间的已格式化的基于行的输出	不适用; 报告 XML 文档中作为格式化函数的输入提供的所有元素
MON_GET_ACTIVITY_DETAILS 表函数 - 获取完整的活动详细信息（在 XML 文档 DETAILS 中报告）	ACTIVITY METRICS BASE
MON_GET_CONNECTION 表函数 - 获取连接度量值	REQUEST METRICS BASE
MON_GET_CONNECTION_DETAILS 表函数 - 获取详细的连接度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE
MON_GET_PKG_CACHE_STMT 表函数 - 获取程序包高速缓存中的 SQL 语句活动度量值	ACTIVITY METRICS BASE
MON_GET_PKG_CACHE_STMT_DETAILS 表函数 - 获取程序包高速缓存条目的详细度量值	ACTIVITY METRICS BASE
MON_GET_SERVICE_SUBCLASS 表函数 - 获取服务子类度量值	REQUEST METRICS BASE
MON_GET_SERVICE_SUBCLASS_DETAILS 表函数 - 获取详细的服务子类度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE
MON_GET_UNIT_OF_WORK 表函数 - 获取工作单元度量值	REQUEST METRICS BASE
MON_GET_UNIT_OF_WORK 表函数 - 获取详细的工作单元度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE
MON_GET_WORKLOAD 表函数 - 获取工作负载度量值	REQUEST METRICS BASE
MON_GET_WORKLOAD_DETAILS 表函数 - 获取详细的工作负载度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE

表 913. 事件监视信息

事件类型	逻辑数据分组	监视开关
活动	event_activity（在 details_xml 文档中报告）	ACTIVITY METRICS BASE
活动	event_activitymetrics	ACTIVITY METRICS BASE
统计信息	event_scstats（在 details_xml 文档中报告）	REQUEST METRICS BASE
统计信息	event_wlstats（在 details_xml 文档中报告）	REQUEST METRICS BASE
工作单元	在 system_metrics 文档中报告。	REQUEST METRICS BASE
程序包高速缓存	在 activity_metrics 文档中报告。	ACTIVITY METRICS BASE

log_disk_waits_total -“日志磁盘等待总次数”监视元素

代理程序被迫等待日志数据被写入磁盘的次数。

表 914. 表函数监视信息

表函数	监视元素收集级别
MON_FORMAT_XML_METRICS_BY_ROW - 获取所有度量值的基于已格式化行的输出	不适用; 报告 XML 文档中作为格式化函数的输入提供的所有元素
MON_GET_ACTIVITY_DETAILS 表函数 - 获取完整的活动详细信息（在 XML 文档 DETAILS 中报告）	ACTIVITY METRICS BASE
MON_GET_CONNECTION 表函数 - 获取连接度量值	REQUEST METRICS BASE
MON_GET_CONNECTION_DETAILS 表函数 - 获取详细的连接度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE
MON_GET_PKG_CACHE_STMT 表函数 - 获取程序包高速缓存中的 SQL 语句活动度量值	ACTIVITY METRICS BASE
MON_GET_PKG_CACHE_STMT_DETAILS 表函数 - 获取程序包高速缓存条目的详细度量值	ACTIVITY METRICS BASE
MON_GET_SERVICE_SUBCLASS 表函数 - 获取服务子类度量值	REQUEST METRICS BASE
MON_GET_SERVICE_SUBCLASS_DETAILS 表函数 - 获取详细的服务子类度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE
MON_GET_UNIT_OF_WORK 表函数 - 获取工作单元度量值	REQUEST METRICS BASE
MON_GET_UNIT_OF_WORK 表函数 - 获取详细的工作单元度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE
MON_GET_WORKLOAD 表函数 - 获取工作负载度量值	REQUEST METRICS BASE

表 914. 表函数监视信息 (续)

表函数	监视元素收集级别
MON_GET_WORKLOAD_DETAILS 表函数 - 获取详细的工作负载度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE

表 915. 事件监视信息

事件类型	逻辑数据分组	监视开关
活动	event_activity（在 details_xml 文档中报告）	ACTIVITY METRICS BASE
活动	event_activitymetrics	ACTIVITY METRICS BASE
统计信息	event_scstats（在 details_xml 文档中报告）	REQUEST METRICS BASE
统计信息	event_wlstats（在 details_xml 文档中报告）	REQUEST METRICS BASE
工作单元	在 system_metrics 文档中报告。	REQUEST METRICS BASE
程序包高速缓存	在 activity_metrics 文档中报告。	ACTIVITY METRICS BASE

log_held_by_dirty_pages -“脏页占用的日志空间量”

对应数据库中的最旧脏页与活动日志顶部之间的差的日志量（以字节计）。

元素标识

log_held_by_dirty_pages

元素类型

水位标记

表 916. 表函数监视信息

表函数	监视元素收集级别
MON_GET_TRANSACTION_LOG 表函数 - 获取日志信息	ACTIVITY METRICS BASE

表 917. 快照监视信息

快照级别	逻辑数据分组	监视开关
数据库	dbase	基本

表 918. 事件监视信息

事件类型	逻辑数据分组	监视开关
数据库	event_db	始终收集

用法 获取快照时，将根据该快照的时间上的条件计算此值。

使用此元素来评估对缓冲池中的旧页进行页清除的效果。

缓冲池中的旧页清除由 *softmax* 数据库配置参数控制。如果页清除有效果，那么 *log_held_by_dirty_pages* 应小于或大致等于:

```
(softmax / 100)  *  logfilsiz * 4096
```

如果不是这样，那么增加页清除程序的数目（*num_iocleaners*）配置参数。

如果符合条件并且要求脏页占用的日志少一些，那么降低 *softmax* 配置参数。

log_read_time -“日志读取时间”

记录器从磁盘读取日志数据的耗用时间总计。对于写至表的事件监视器，此元素的值将通过使用 BIGINT 数据类型以微秒为单位给定。

表 919. 表函数监视信息

表函数	监视元素收集级别
MON_GET_TRANSACTION_LOG 表函数 - 获取日志信息	ACTIVITY METRICS BASE

表 920. 快照监视信息

快照级别	逻辑数据分组	监视开关
数据库	dbase	基本

可将快照监视的计数器重置。

表 921. 事件监视信息

事件类型	逻辑数据分组	监视开关
数据库	event_db	始终收集

用法 将此元素与 *log_reads*、*num_log_read_io* 和 *num_log_data_found_in_buffer* 元素一起使用来确定以下情况是否属实:

- 当前磁盘对于记录操作而言是够用的。
- 日志缓冲区大小够用。

log_reads -“读取的日志页数”

记录器从磁盘读取的日志页数。

元素标识

log_reads

元素类型

计数器

表 922. 表函数监视信息

表函数	监视元素收集级别
MON_GET_TRANSACTION_LOG 表函数 - 获取日志信息	ACTIVITY METRICS BASE

表 923. 快照监视信息

快照级别	逻辑数据分组	监视开关
数据库	dbase	基本

可将快照监视的计数器重置。

表 924. 事件监视信息

事件类型	逻辑数据分组	监视开关
数据库	event_db	始终收集

用法 可将此元素与操作系统监视器配合使用来确定设备上可归因于数据库活动的 I/O 量。

log_to_redo_for_recovery -“要为恢复重做的日志量”

必须为崩溃恢复重做的日志量（以字节计）。

元素标识

log_to_redo_for_recovery

元素类型

水位标记

表 925. 表函数监视信息

表函数	监视元素收集级别
MON_GET_TRANSACTION_LOG 表函数 - 获取日志信息	ACTIVITY METRICS BASE

表 926. 快照监视信息

快照级别	逻辑数据分组	监视开关
数据库	dbase	基本

表 927. 事件监视信息

事件类型	逻辑数据分组	监视开关
数据库	event_db	始终收集

用法 获取快照时，将根据该快照的时间上的条件计算此值。如果值较大，那么指示系统崩溃后的恢复时间较长。如果值看起来过大，那么检查 *log_held_by_dirty_pages* 监视元素以了解是否需要调整页清除。还应检查是否存在任何长时间运行并且需要终止的事务。

log_write_time -“日志写入时间”

记录器将日志数据写至磁盘的耗用时间总计。对于写至表的事件监视器，此元素的值将通过使用 BIGINT 数据类型以微秒为单位给定。

表 928. 表函数监视信息

表函数	监视元素收集级别
MON_GET_TRANSACTION_LOG 表函数 - 获取日志信息	ACTIVITY METRICS BASE

表 929. 快照监视信息

快照级别	逻辑数据分组	监视开关
数据库	dbase	基本

可将快照监视的计数器重置。

表 930. 事件监视信息

事件类型	逻辑数据分组	监视开关
数据库	event_db	始终收集

用法 将此元素与 *log_writes* 和 *num_log_write_io* 元素一起使用来确定当前磁盘对于记录操作而言是否够用。

log_writes -“写入的日志页数”

记录器写至磁盘的日志页数。

元素标识

log_writes

元素类型

计数器

表 931. 表函数监视信息

表函数	监视元素收集级别
MON_GET_TRANSACTION_LOG 表函数 - 获取日志信息	ACTIVITY METRICS BASE

表 932. 快照监视信息

快照级别	逻辑数据分组	监视开关
数据库	dbase	基本

可将快照监视的计数器重置。

表 933. 事件监视信息

事件类型	逻辑数据分组	监视开关
数据库	event_db	始终收集

用法 可将此元素与操作系统监视器配合使用来确定设备上可归因于数据库活动的 I/O 量。

注: 当日志页写至磁盘时，最后一页可能未满。在这种情况下，部分日志页保留在日志缓冲区中，而其他日志记录将写至页。因此记录器可能会多次将日志页写至磁盘。不应使用此元素来量度 DB2 生成的页数。

long_object_pages -“长对象页数”

表中的长数据消耗的磁盘页数。

表 934. 快照监视信息

快照级别	逻辑数据分组	监视开关
表	表	基本

表 935. 事件监视信息

事件类型	逻辑数据分组	监视开关
表	event_table	始终收集

用法 此元素提供了一种机制，可用来查看特定表中的长数据消耗的实际空间量。此元素可与表事件监视器配合使用以跟踪一段时间内长数据的增长率。

long_object_l_pages -“长对象数据逻辑页数”监视元素

此表中包含的长数据在磁盘上使用的逻辑页数。

表 936. 表函数监视信息

表函数	监视元素收集级别
MON_GET_TABLE 表函数 - 获取表度量值	ACTIVITY METRICS BASE

用法

- 此值可能小于实际上为该对象分配的空间量。使用 TRUNCATE 语句的 REUSE STORAGE 选项时，可能会发生这种情况。此选项导致为该表分配的存储空间继续被分配，尽管该存储空间被视为空。此外，此监视元素的值可能小于逻辑上为该对象分配的空间量，因为逻辑上分配的空间总量包括少量附加元数据。

  要检索对象的逻辑大小或物理大小的准确度量，请使用 ADMIN_GET_TAB_INFO_V97 函数。此函数提供的有关对象大小的信息比您可（通过将针对此监视元素报告的页数乘以页大小）获取的对象大小更准确。

long_tbsp_id -“长表空间标识”监视元素

一个表空间的标识，此表空间用于存放此表的长数据（LONG 或 LOB 类型的列）。

表 937. 表函数监视信息

表函数	监视元素收集命令和级别
MON_GET_TABLE 表函数 - 获取表度量值	ACTIVITY METRICS BASE

用法

此元素的值与视图 SYSCAT.TABLESPACES 的列 TBSPACEID 中的值相匹配。

machine_identification -“主机硬件标识”监视元素

用于描述处理器体系结构的字符串。例如，“x86 64 bit”。对此标识返回的值由正在主机上运行的操作系统确定。

表 938. 表函数监视信息

表函数	监视元素收集级别
ENV_GET_SYSTEM_RESOURCES 表函数 - 返回系统信息	ACTIVITY METRICS BASE

max_agent_overflows -“最大代理程序溢出次数”

达到“最大代理程序数”（**maxagents**）配置参数后接收到创建新代理程序的请求的次数。

注: 从 DB2 V9.5 开始，不推荐使用 **max_agent_overflows** 监视元素。使用此监视元素不会生成错误。但是不会返回有效值。建议不要再使用此监视元素，将来的发行版中可能会将其除去。

表 939. 快照监视信息

快照级别	逻辑数据分组	监视开关
数据库管理器	db2	基本

用法

如果达到 **maxagents** 配置参数后仍然接收到代理程序创建请求，那么可能指示此节点的工作负载太高。

max_coord_stmt_exec_time -“最长协调程序语句执行时间”监视元素

运行一次语句的最长协调程序执行时间（以毫秒计）。如果该语句未执行，那么此值在非协调程序节点上将为零。

表 940. 表函数监视信息

表函数	监视元素收集级别
MON_GET_PKG_CACHE_STMT 表函数 - 获取程序包高速缓存中的 SQL 语句活动度量值	ACTIVITY METRICS BASE
MON_GET_PKG_CACHE_STMT_DETAILS 表函数 - 获取程序包高速缓存条目的详细度量值（在 XML 文档 DETAILS 中报告）	ACTIVITY METRICS BASE

表 941. 事件监视信息

事件类型	逻辑数据分组	监视开关
程序包高速缓存	pkgcache	ACTIVITY METRICS BASE

max_coord_stmt_exec_time_args -“协调程序语句执行时间参数的最大数目”监视元素

一个 XML 文档，它呈示提供给以下语句的输入参数，此语句是在协调程序成员上运行一次 (**max_coord_stmt_exec_time**) 消耗最长执行时间的语句。如果该语句尚未运行或没有输入参数，那么此列为 NULL。

此文档包含名为 **max_coord_stmt_exec_time_args** 的父元素，该元素由一个或多个名为 **max_coord_stmt_exec_time_arg** 的元素组成。有关 XML 文档的结构的示例，请参阅第 883 页的图 19。

表 942. 表函数监视信息

表函数	监视元素收集级别
MON_GET_PKG_CACHE_STMT 表函数 - 获取程序包高速缓存中的 SQL 语句活动度量值	ACTIVITY METRICS BASE
MON_GET_PKG_CACHE_STMT_DETAILS 表函数 - 获取程序包高速缓存条目的详细度量值（在 XML 文档 DETAILS 中报告）	ACTIVITY METRICS BASE

表 943. 事件监视信息

事件类型	逻辑数据分组	监视开关
程序包高速缓存	pkgcache_stmt_args	ACTIVITY METRICS BASE

用法

可使用 XMLPARSE 标量函数来查看此文档的内容。例如:

```
SELECT XMLPARSE(DOCUMENT MAX_COORD_STMT_EXEC_STMT_ARGS)
  FROM TABLE(MON_GET_PKG_CACHE_STMT(NULL, NULL, NULL, -2));
```

第 883 页的图 19显示先前语句返回的 XML 文档内容的示例。

```
<?xml version="1.0" encoding="ISO-8859-1" ?>
 <max_coord_stmt_exec_time_args
  xmlns="http://www.ibm.com/xmlns/prod/db2/mon" release="10010000">
   <max_coord_stmt_exec_time_arg>
      <stmt_value_index>1</stmt_value_index>
      <stmt_value_isreopt id="0">no</stmt_value_isreopt>
      <stmt_value_isnull id="0">no</stmt_value_isnull>
      <stmt_value_type>INTEGER</stmt_value_type>
      <stmt_value_data>5</stmt_value_data>
   </max_coord_stmt_exec_time_arg>
   <max_coord_stmt_exec_time_arg>
      <stmt_value_index>2</stmt_value_index>
      <stmt_value_isreopt id="0">no</stmt_value_isreopt>
       <stmt_value_isnull id="0">no</stmt_value_isnull>
       <stmt_value_type>VARCHAR</stmt_value_type>
       <stmt_value_data>78</stmt_value_data>
   </max_coord_stmt_exec_time_arg>
   <max_coord_stmt_exec_time_arg>
      <stmt_value_index>3</stmt_value_index>
      <stmt_value_isreopt id="0">no</stmt_value_isreopt>
       <stmt_value_isnull id="0">no</stmt_value_isnull>
      <stmt_value_type>VARCHAR</stmt_value_type>
      <stmt_value_data>john</stmt_value_data>
   </max_coord_stmt_exec_time_arg>
   <max_coord_stmt_exec_time_arg>
      <stmt_value_index>4</stmt_value_index>
      <stmt_value_isreopt id="0">no</stmt_value_isreopt>
       <stmt_value_isnull id="0">no</stmt_value_isnull>
      <stmt_value_type>VARCHAR</stmt_value_type>
      <stmt_value_data>x</stmt_value_data>
   </max_coord_stmt_exec_time_arg>
   <max_coord_stmt_exec_time_arg>
     <stmt_value_index>5</stmt_value_index>
     <stmt_value_isreopt id="0">no</stmt_value_isreopt>
     <stmt_value_isnull id="0">no</stmt_value_isnull>
     <stmt_value_type>DATE</stmt_value_type>
     <stmt_value_data>2001-02-12</stmt_value_data>

⋮

   <max_coord_stmt_exec_time_arg>
      <stmt_value_index>15</stmt_value_index>
      <stmt_value_isreopt id="0">no</stmt_value_isreopt>
      <stmt_value_isnull id="0">no</stmt_value_isnull>
      <stmt_value_type>DECIMAL</stmt_value_type>
      <stmt_value_data>+0002000.55</stmt_value_data>
    </max_coord_stmt_exec_time_arg>
  </max_coord_stmt_exec_time_args>
```

图 19. **max_coord_stmt_exec_time_args** *的样本内容.* 在此示例中，此文档显示已将 15 个参数传递至该语句。

以下数据类型的条目记录在此 XML 文档中，但是，STMT_VALUE_DATA 元素中不会捕获这些类型的参数的实际值:

- BLOB
- CLOB
- REF
- BOOLEAN
- 结构化数据类型
- DATALINK
- LONG VARGRAPHIC

- LONG VARCHAR
- XML 类型
- DBCLOB
- ARRAY 类型
- ROW 类型
- ROWID
- CURSOR 变量

在语句中，最先出现的输入自变量在记录时从 1 开始，后续每个输入自变量依次加 1。可记录的输入参数的数目仅受包含该 XML 文档的 BLOB 的大小的上限所限制。实际上，捕获的输入自变量数目可能导致达到此限制。

可在路径 sqllib/misc/DB2EvmonPkgCache.xsd 中找到包含此元素的 XML 文档的模式。

max_coord_stmt_exec_timestamp -“最大协调程序语句执行时间戳记”监视元素

生成 max_coord_stmt_exec_time 值的语句开始执行的时间。

表 944. 表函数监视信息

表函数	监视元素收集级别
MON_GET_PKG_CACHE_STMT 表函数 - 获取程序包高速缓存中的 SQL 语句活动度量值	ACTIVITY METRICS BASE
MON_GET_PKG_CACHE_STMT_DETAILS 表函数 - 获取程序包高速缓存条目的详细度量值（在 XML 文档 DETAILS 中报告）	ACTIVITY METRICS BASE

表 945. 事件监视信息

事件类型	逻辑数据分组	监视开关
程序包高速缓存	pkgcache	ACTIVITY METRICS BASE

max_data_received_1024 -“接收的出站字节数在 513 到 1024 字节之间的语句数”

此元素表示接收的出站字节数在 513 到 1024 字节之间（包括 513 字节和 1024 字节）的语句或链的数目。

元素标识

max_data_received_1024

元素类型

计数器

表 946. 快照监视信息

快照级别	逻辑数据分组	监视开关
DCS 数据库	dcs_dbase	语句
DCS 应用程序	dcs_appl	语句
数据传输	stmt_transmissions	语句

可将快照监视的计数器重置。

用法 使用此元素来更好地了解数据库或应用程序级别的数据库活动和网络流量。

max_data_received_128 -“接收的出站字节数在 1 到 128 字节之间的语句数”

此元素表示接收的出站字节数在 1 到 128 字节之间（包括 1 字节和 128 字节）的语句或链的数目。

元素标识

max_data_received_128

元素类型

计数器

表 947. 快照监视信息

快照级别	逻辑数据分组	监视开关
DCS 数据库	dcs_dbase	语句
DCS 应用程序	dcs_appl	语句
数据传输	stmt_transmissions	语句

可将快照监视的计数器重置。

用法 使用此元素来更好地了解数据库或应用程序级别的数据库活动和网络流量。

max_data_received_16384 -“接收的出站字节数在 8193 到 16384 字节之间的语句数”

此元素表示接收的出站字节数在 8193 到 16384 字节之间（包括 8193 字节和 16384 字节）的语句或链的数目。

元素标识

max_data_received_16384

元素类型

计数器

表 948. 快照监视信息

快照级别	逻辑数据分组	监视开关
DCS 数据库	dcs_dbase	语句
DCS 应用程序	dcs_appl	语句
数据传输	stmt_transmissions	语句

可将快照监视的计数器重置。

用法 使用此元素来更好地了解数据库或应用程序级别的数据库活动和网络流量。

max_data_received_2048 -“接收的出站字节数在 1025 到 2048 字节之间的语句数”

此元素表示接收的出站字节数在 1025 到 2048 字节之间（包括 1025 字节和 2048 字节）的语句或链的数目。

元素标识

max_data_received_2048

元素类型

计数器

表 949. 快照监视信息

快照级别	逻辑数据分组	监视开关
DCS 数据库	dcs_dbase	语句
DCS 应用程序	dcs_appl	语句
数据传输	stmt_transmissions	语句

可将快照监视的计数器重置。

用法 使用此元素来更好地了解数据库或应用程序级别的数据库活动和网络流量。

max_data_received_256 -“接收的出站字节数在 129 到 256 字节之间的语句数”

此元素表示接收的出站字节数在 129 到 256 字节之间（包括 129 字节和 256 字节）的语句或链的数目。

元素标识

max_data_received_256

元素类型

计数器

表 950. 快照监视信息

快照级别	逻辑数据分组	监视开关
DCS 数据库	dcs_dbase	语句
DCS 应用程序	dcs_appl	语句
数据传输	stmt_transmissions	语句

可将快照监视的计数器重置。

用法 使用此元素来更好地了解数据库或应用程序级别的数据库活动和网络流量。

max_data_received_31999 -“接收的出站字节数在 16385 到 31999 字节之间的语句数”监视元素

此元素表示接收的出站字节数在 16385 到 31999 字节之间（包括 16385 字节和 31999 字节）的语句或链的数目。

表 951. 快照监视信息

快照级别	逻辑数据分组	监视开关
DCS 数据库	dcs_dbase	语句
DCS 应用程序	dcs_appl	语句
数据传输	stmt_transmissions	语句

可将快照监视的计数器重置。

用法 使用此元素来更好地了解数据库或应用程序级别的数据库活动和网络流量。

max_data_received_4096 -“接收的出站字节数在 2049 到 4096 字节之间的语句数”

此元素表示接收的出站字节数在 2049 到 4096 字节之间（包括 2049 字节和 4096 字节）的语句或链的数目。

元素标识

max_data_received_4096

元素类型

计数器

表 952. 快照监视信息

快照级别	逻辑数据分组	监视开关
DCS 数据库	dcs_dbase	语句
DCS 应用程序	dcs_appl	语句
数据传输	stmt_transmissions	语句

可将快照监视的计数器重置。

用法 使用此元素来更好地了解数据库或应用程序级别的数据库活动和网络流量。

max_data_received_512 -“接收的出站字节数在 257 到 512 字节之间的语句数”

此元素表示接收的出站字节数在 257 到 512 字节之间（包括 257 字节和 512 字节）的语句或链的数目。

元素标识

max_data_received_512

元素类型

计数器

表 953. 快照监视信息

快照级别	逻辑数据分组	监视开关
DCS 数据库	dcs_dbase	语句
DCS 应用程序	dcs_appl	语句
数据传输	stmt_transmissions	语句

可将快照监视的计数器重置。

用法 使用此元素来更好地了解数据库或应用程序级别的数据库活动和网络流量。

max_data_received_64000 -“接收的出站字节数在 32000 到 64000 字节之间的语句数”监视元素

此元素表示接收的出站字节数在 32000 到 64000 字节之间（包括 32000 字节和 64000 字节）的语句或链的数目。

表 954. 快照监视信息

快照级别	逻辑数据分组	监视开关
DCS 数据库	dcs_dbase	语句
DCS 应用程序	dcs_appl	语句
数据传输	stmt_transmissions	语句

可将快照监视的计数器重置。

用法 使用此元素来更好地了解数据库或应用程序级别的数据库活动和网络流量。

max_data_received_8192 -“接收的出站字节数在 4097 到 8192 字节之间的语句”

此元素表示接收的出站字节数在 4097 到 8192 字节之间（包括 4097 字节和 8192 字节）的语句或链的数目。

元素标识

max_data_received_8192

元素类型

计数器

表 955. 快照监视信息

快照级别	逻辑数据分组	监视开关
DCS 数据库	dcs_dbase	语句
DCS 应用程序	dcs_appl	语句
数据传输	stmt_transmissions	语句

可将快照监视的计数器重置。

用法 使用此元素来更好地了解数据库或应用程序级别的数据库活动和网络流量。

max_data_received_gt64000 -“接收的出站字节数高于 64000 的语句数”

此元素表示接收的出站字节数高于 64000 的语句或链的数目。

元素标识

max_data_received_gt64000

元素类型

计数器

表 956. 快照监视信息

快照级别	逻辑数据分组	监视开关
DCS 数据库	dcs_dbase	语句
DCS 应用程序	dcs_appl	语句
数据传输	stmt_transmissions	语句

可将快照监视的计数器重置。

用法 使用此元素来更好地了解数据库或应用程序级别的数据库活动和网络流量。

max_data_sent_1024 -“发送的出站字节数在 513 到 1024 字节之间的语句数”

此元素表示发送的出站字节数在 513 到 1024 字节之间（包括 513 字节和 1024 字节）的语句或链的数目。

元素标识

max_data_sent_1024

元素类型

计数器

表 957. 快照监视信息

快照级别	逻辑数据分组	监视开关
DCS 数据库	dcs_dbase	语句
DCS 应用程序	dcs_appl	语句
数据传输	stmt_transmissions	语句

可将快照监视的计数器重置。

用法 使用此元素来更好地了解数据库或应用程序级别的数据库活动和网络流量。

max_data_sent_128 -“发送的出站字节数在 1 到 128 字节之间的语句数”

此元素表示发送的出站字节数在 1 到 128 字节之间（包括 1 字节和 128 字节）的语句或链的数目。

元素标识

max_data_sent_128

元素类型

计数器

表 958. 快照监视信息

快照级别	逻辑数据分组	监视开关
DCS 数据库	dcs_dbase	语句
DCS 应用程序	dcs_appl	语句
数据传输	stmt_transmissions	语句

可将快照监视的计数器重置。

用法 使用此元素来更好地了解数据库或应用程序级别的数据库活动和网络流量。

max_data_sent_16384 -“发送的出站字节数在 8193 到 16384 字节之间的语句数”

此元素表示发送的出站字节数在 8193 到 16384 字节之间（包括 8193 字节和 16384 字节）的语句或链的数目。

元素标识

max_data_sent_16384

元素类型

计数器

表 959. 快照监视信息

快照级别	逻辑数据分组	监视开关
DCS 数据库	dcs_dbase	语句
DCS 应用程序	dcs_appl	语句
数据传输	stmt_transmissions	语句

可将快照监视的计数器重置。

用法 使用此元素来更好地了解数据库或应用程序级别的数据库活动和网络流量。

max_data_sent_2048 -“发送的出站字节数在 1025 到 2048 字节之间的语句数”

此元素表示发送的出站字节数在 1025 到 2048 字节之间（包括 1025 字节和 2048 字节）的语句或链的数目。

元素标识

max_data_sent_2048

元素类型

计数器

表 960. 快照监视信息

快照级别	逻辑数据分组	监视开关
DCS 数据库	dcs_dbase	语句
DCS 应用程序	dcs_appl	语句
数据传输	stmt_transmissions	语句

可将快照监视的计数器重置。

用法 使用此元素来更好地了解数据库或应用程序级别的数据库活动和网络流量。

max_data_sent_256 -“发送的出站字节数在 129 到 256 字节之间的语句数”

此元素表示发送的出站字节数在 129 到 256 字节之间（包括 129 字节和 256 字节）的语句或链的数目。

元素标识

max_data_sent_256

元素类型

计数器

表 961. 快照监视信息

快照级别	逻辑数据分组	监视开关
DCS 数据库	dcs_dbase	语句
DCS 应用程序	dcs_appl	语句
数据传输	stmt_transmissions	语句

可将快照监视的计数器重置。

用法 使用此元素来更好地了解数据库或应用程序级别的数据库活动和网络流量。

max_data_sent_31999 -“发送的出站字节数在 16385 到 31999 字节之间的语句数”

此元素表示发送的出站字节数在 16385 到 31999 字节之间（包括 16385 字节和 31999 字节）的语句或链的数目。

元素标识

max_data_sent_31999

元素类型

计数器

表 962. 快照监视信息

快照级别	逻辑数据分组	监视开关
DCS 数据库	dcs_dbase	语句
DCS 应用程序	dcs_appl	语句
数据传输	stmt_transmissions	语句

可将快照监视的计数器重置。

用法 使用此元素来更好地了解数据库或应用程序级别的数据库活动和网络流量。

max_data_sent_4096 -“发送的出站字节数在 2049 到 4096 字节之间的语句数”

此元素表示发送的出站字节数在 2049 到 4096 字节之间（包括 2049 字节和 4096 字节）的语句或链的数目。

元素标识

max_data_sent_4096

元素类型

计数器

表 963. 快照监视信息

快照级别	逻辑数据分组	监视开关
DCS 数据库	dcs_dbase	语句
DCS 应用程序	dcs_appl	语句
数据传输	stmt_transmissions	语句

可将快照监视的计数器重置。

用法 使用此元素来更好地了解数据库或应用程序级别的数据库活动和网络流量。

max_data_sent_512 -“发送的出站字节数在 257 到 512 字节之间的语句数”

此元素表示发送的出站字节数在 257 到 512 字节之间（包括 257 字节和 512 字节）的语句或链的数目。

元素标识

max_data_sent_512

元素类型

计数器

表 964. 快照监视信息

快照级别	逻辑数据分组	监视开关
DCS 数据库	dcs_dbase	语句
DCS 应用程序	dcs_appl	语句
数据传输	stmt_transmissions	语句

可将快照监视的计数器重置。

用法 使用此元素来更好地了解数据库或应用程序级别的数据库活动和网络流量。

max_data_sent_64000 -“发送的出站字节数在 32000 到 64000 字节之间的语句数”

此元素表示发送的出站字节数在 32000 到 64000 字节之间（包括 32000 字节和 64000 字节）的语句或链的数目。

元素标识

max_data_sent_64000

元素类型

计数器

表 965. 快照监视信息

快照级别	逻辑数据分组	监视开关
DCS 数据库	dcs_dbase	语句
DCS 应用程序	dcs_appl	语句
数据传输	stmt_transmissions	语句

可将快照监视的计数器重置。

用法 使用此元素来更好地了解数据库或应用程序级别的数据库活动和网络流量。

max_data_sent_8192 -“发送的出站字节数在 4097 到 8192 字节之间的语句数”

此元素表示发送的出站字节数在 4097 到 8192 字节之间（包括 4097 字节和 8192 字节）的语句或链的数目。

元素标识

max_data_sent_8192

元素类型

计数器

表 966. 快照监视信息

快照级别	逻辑数据分组	监视开关
DCS 数据库	dcs_dbase	语句
DCS 应用程序	dcs_appl	语句
数据传输	stmt_transmissions	语句

可将快照监视的计数器重置。

用法 使用此元素来更好地了解数据库或应用程序级别的数据库活动和网络流量。

max_data_sent_gt64000 -“发送的出站字节数高于 64000 的语句数”

此元素表示发送的出站字节数高于 64000 的语句或链的数目。

元素标识

max_data_sent_gt64000

元素类型

计数器

表 967. 快照监视信息

快照级别	逻辑数据分组	监视开关
DCS 数据库	dcs_dbase	语句

表 967. 快照监视信息 (续)

快照级别	逻辑数据分组	监视开关
DCS 应用程序	dcs_appl	语句
数据传输	stmt_transmissions	语句

可将快照监视的计数器重置。

用法 使用此元素来更好地了解数据库或应用程序级别的数据库活动和网络流量。

max_network_time_100_ms -“网络时间在 16 到 100 毫秒之间的语句数”

此元素表示网络时间大于 16 毫秒但小于或等于 100 毫秒的语句或链的数目。（网络时间是主机响应时间与语句或链所耗用的执行时间之差。）

表 968. 快照监视信息

快照级别	逻辑数据分组	监视开关
DCS 数据库	dcs_dbase	语句
DCS 应用程序	dcs_appl	语句
数据传输	stmt_transmissions	语句

可将快照监视的计数器重置。

用法 使用此元素来更好地了解数据库或应用程序级别的数据库活动和网络流量。

max_network_time_16_ms -“网络时间在 4 到 16 毫秒之间的语句数”

此元素表示网络时间大于 4 毫秒但小于或等于 16 毫秒的语句或链的数目。（网络时间是主机响应时间与语句或链所耗用的执行时间之差。）

元素标识

max_network_time_16_ms

元素类型

计数器

表 969. 快照监视信息

快照级别	逻辑数据分组	监视开关
DCS 数据库	dcs_dbase	语句
DCS 应用程序	dcs_appl	语句
数据传输	stmt_transmissions	语句

可将快照监视的计数器重置。

用法 使用此元素来更好地了解数据库或应用程序级别的数据库活动和网络流量。

max_network_time_1_ms -“网络时间最多为 1 毫秒的语句数”

此元素表示网络时间小于或等于 1 毫秒的语句或链的数目。（网络时间是主机响应时间与语句或链所耗用的执行时间之差。）

表 970. 快照监视信息

快照级别	逻辑数据分组	监视开关
DCS 数据库	dcs_dbase	语句
DCS 应用程序	dcs_appl	语句
数据传输	stmt_transmissions	语句

可将快照监视的计数器重置。

用法 使用此元素来更好地了解数据库或应用程序级别的数据库活动和网络流量。

max_network_time_4_ms -“网络时间在 1 到 4 毫秒之间的语句数”

此元素表示网络时间大于 1 毫秒但小于或等于 4 毫秒的语句或链的数目。（网络时间是主机响应时间与语句或链所耗用的执行时间之差。）

元素标识

max_network_time_4_ms

元素类型

计数器

表 971. 快照监视信息

快照级别	逻辑数据分组	监视开关
DCS 数据库	dcs_dbase	语句
DCS 应用程序	dcs_appl	语句
数据传输	stmt_transmissions	语句

可将快照监视的计数器重置。

用法 使用此元素来更好地了解数据库或应用程序级别的数据库活动和网络流量。

max_network_time_500_ms -“网络时间在 100 到 500 毫秒之间的语句数”

此元素表示网络时间大于 100 毫秒但小于或等于 500 毫秒的语句或链的数目。（网络时间是主机响应时间与语句或链所耗用的执行时间之差。）

表 972. 快照监视信息

快照级别	逻辑数据分组	监视开关
DCS 数据库	dcs_dbase	语句
DCS 应用程序	dcs_appl	语句
数据传输	stmt_transmissions	语句

可将快照监视的计数器重置。

用法 使用此元素来更好地了解数据库或应用程序级别的数据库活动和网络流量。

max_network_time_gt500_ms -“网络时间大于 500 毫秒的语句数”

此元素表示网络时间大于 500 毫秒的语句或链的数目。（网络时间是主机响应时间与语句或链所耗用的执行时间之差。）

表 973. 快照监视信息

快照级别	逻辑数据分组	监视开关
DCS 数据库	dcs_dbase	语句
DCS 应用程序	dcs_appl	语句
数据传输	stmt_transmissions	语句

可将快照监视的计数器重置。

用法 使用此元素来更好地了解数据库或应用程序级别的数据库活动和网络流量。

member -“数据库成员”监视元素

数据库成员的数字标识，此结果记录的数据接收自该成员。

表 974. 表函数监视信息

表函数	监视元素收集级别
ADMIN_GET_MEM_USAGE 表函数 - 获取实例的总内存消耗	ACTIVITY METRICS BASE
AUDIT_ARCHIVE 过程和表函数 - 归档审计日志文件	ACTIVITY METRICS BASE
DBCFG 管理视图和 DB_GET_CFG 表函数 - 检索数据库配置参数信息	ACTIVITY METRICS BASE
ENV_GET_REG_VARIABLES 表函数 - 检索正使用的 DB2 注册表设置	ACTIVITY METRICS BASE
ENV_GET_DB2_SYSTEM_RESOURCES 表函数 - 返回 DB2(r) 系统信息	ACTIVITY METRICS BASE
ENV_GET_SYSTEM_RESOURCES 表函数 - 返回系统信息	ACTIVITY METRICS BASE
ENV_GET_NETWORK_RESOURCES 表函数 - 返回网络适配器信息	ACTIVITY METRICS BASE
MON_GET_ACTIVITY_DETAILS 表函数 - 获取完整的活动详细信息	ACTIVITY METRICS BASE
MON_GET_AUTO_MAINT_QUEUE 表函数 - 获取有关自动维护作业的信息	ACTIVITY METRICS BASE
MON_GET_AUTO_RUNSTATS_QUEUE 表函数 - 检索有关为求值而排队的对象的信息	ACTIVITY METRICS BASE
MON_GET_BUFFERPOOL 表函数 - 获取缓冲池度量值	ACTIVITY METRICS BASE
MON_GET_CF_WAIT_TIME 表函数 - 获取集群高速缓存工具命令等待时间	ACTIVITY METRICS BASE

表 974. 表函数监视信息 (续)

表函数	监视元素收集级别
MON_GET_CONNECTION 表函数 - 获取连接度量值	ACTIVITY METRICS BASE
MON_GET_CONNECTION_DETAILS 表函数 - 获取详细的连接度量值	ACTIVITY METRICS BASE
MON_GET_CONTAINER 表函数 - 获取表空间容器度量值	ACTIVITY METRICS BASE
MON_GET_EXTENDED_LATCH_WAIT 表函数 - 返回锁存器的信息	ACTIVITY METRICS BASE
MON_GET_EXTENT_MOVEMENT_STATUS - 获取扩展数据块移动进度状态度量值	ACTIVITY METRICS BASE
MON_GET_FCM - 获取 FCM 度量值	ACTIVITY METRICS BASE
MON_GET_FCM_CONNECTION_LIST - 获取有关所有 FCM 连接的详细信息	ACTIVITY METRICS BASE
MON_GET_GROUP_BUFFERPOOL 表函数 - 获取组缓冲池度量值	ACTIVITY METRICS BASE
MON_GET_INDEX 表函数 - 获取索引度量值	ACTIVITY METRICS BASE
MON_GET_INDEX_USAGE_LIST 表函数 - 返回索引用法列表中的信息	ACTIVITY METRICS BASE
MON_GET_LOCKS 表函数 - 列示当前所连接的数据库中的所有锁定	ACTIVITY METRICS BASE
MON_GET_MEMORY_POOL 表函数 - 获取内存池信息	ACTIVITY METRICS BASE
MON_GET_MEMORY_SET 表函数 - 获取内存集合信息	ACTIVITY METRICS BASE
MON_GET_PAGE_ACCESS_INFO 表函数 - 获取缓冲池页等待信息	ACTIVITY METRICS BASE
MON_GET_PKG_CACHE_STMT 表函数 - 获取程序包高速缓存中的 SQL 语句活动度量值	ACTIVITY METRICS BASE
MON_GET_PKG_CACHE_STMT_DETAILS 表函数 - 获取程序包高速缓存条目的详细度量值	ACTIVITY METRICS BASE
MON_GET_REBALANCE_STATUS 表函数 - 获取表空间的重新平衡进度	ACTIVITY METRICS BASE
MON_GET_RTS_RQST 表函数 - 检索有关实时统计请求的信息	ACTIVITY METRICS BASE
MON_GET_SERVERLIST 表函数 - 获取成员优先级详细信息	ACTIVITY METRICS BASE
MON_GET_SERVICE_SUBCLASS 表函数 - 获取服务子类度量值	ACTIVITY METRICS BASE
MON_GET_SERVICE_SUBCLASS_DETAILS 表函数 - 获取详细的服务子类度量值	ACTIVITY METRICS BASE
MON_GET_TABLE 表函数 - 获取表度量值	ACTIVITY METRICS BASE
MON_GET_TABLE_USAGE_LIST 表函数 - 从表用法列表返回信息	ACTIVITY METRICS BASE

表 974. 表函数监视信息 (续)

表函数	监视元素收集级别
MON_GET_TABLESPACE 表函数 - 获取表空间度量值	ACTIVITY METRICS BASE
MON_GET_TRANSACTION_LOG 表函数 - 获取日志信息	ACTIVITY METRICS BASE
MON_GET_UNIT_OF_WORK 表函数 - 获取工作单元度量值	ACTIVITY METRICS BASE
MON_GET_UNIT_OF_WORK_DETAILS 表函数 - 获取详细的工作单元度量值	ACTIVITY METRICS BASE
MON_GET_USAGE_LIST_STATUS 表函数 - 返回用法列表的状态	ACTIVITY METRICS BASE
MON_GET_WORKLOAD 表函数 - 获取工作负载度量值	ACTIVITY METRICS BASE
MON_GET_WORKLOAD_DETAILS 表函数 - 获取详细的工作负载度量值	ACTIVITY METRICS BASE
MON_SAMPLE_SERVICE_CLASS_METRICS - 获取样本服务类度量值	ACTIVITY METRICS BASE
MON_SAMPLE_WORKLOAD_METRICS - 获取样本	ACTIVITY METRICS BASE
PD_GET_DIAG_HIST 表函数 - 从给定设施返回记录	ACTIVITY METRICS BASE
PDLOGMSGS_LAST24HOURS 管理视图和 PD_GET_LOG_MSGS 表函数 - 检索问题确定消息	ACTIVITY METRICS BASE
WLM_GET_QUEUE_STATS 表函数 - 返回阈值队列统计信息	ACTIVITY METRICS BASE
WLM_GET_SERVICE_CLASS_AGENTS 表函数 - 列示正在服务类中运行的代理程序	ACTIVITY METRICS BASE
WLM_GET_SERVICE_CLASS_WORKLOAD_OCCURRENCES 表函数 - 列示工作负载出现次数	ACTIVITY METRICS BASE
WLM_GET_SERVICE_SUBCLASS_STATS 表函数 - 返回服务子类的统计信息	ACTIVITY METRICS BASE
WLM_GET_SERVICE_SUPERCLASS_STATS 表函数 - 返回服务超类的统计信息	ACTIVITY METRICS BASE
WLM_GET_WORK_ACTION_SET_STATS 表函数 - 返回工作操作集统计信息	ACTIVITY METRICS BASE
WLM_GET_WORKLOAD_OCCURRENCE_ACTIVITIES 表函数 - 返回活动列表	ACTIVITY METRICS BASE
WLM_GET_WORKLOAD_STATS 表函数 - 返回工作负载统计信息	ACTIVITY METRICS BASE

表 975. 事件监视信息

事件类型	逻辑数据分组	监视开关
活动	event_activity（在 details_xml 文档中报告）	ACTIVITY METRICS BASE
统计信息	event_scstats（在 details_xml 文档中报告）	REQUEST METRICS BASE
统计信息	event_wlstats（在 details_xml 文档中报告）	REQUEST METRICS BASE
工作单元	在 system_metrics 文档中报告。	REQUEST METRICS BASE
程序包高速缓存	-	COLLECT BASE DATA
锁定	-	始终收集
变更历史记录	changesummary dbdbmcfg regvar ddlstmtexec txncompletion evmonstart utilstart utillocation utilstop	始终收集

用法

DB2 成员是在单一主机上运行 DB2 服务器软件的数据库管理器实例。DB2 成员接受并处理来自与其相连接的应用程序的数据库请求。

memory_free -“可用物理内存量”监视元素

此主机上未分配至正在运行的进程的物理内存总量，以 MB 计。

表 976. 表函数监视信息

表函数	监视元素收集级别
ENV_GET_SYSTEM_RESOURCES 表函数 - 返回系统信息	ACTIVITY METRICS BASE

memory_pool_used_hwm -“内存池高水位标记”监视元素

自创建此池以来分配到此池的最大内存量（以 KB 计）。

表 977. 表函数监视信息

表函数	监视元素收集级别
MON_GET_MEMORY_POOL 表函数 - 获取内存集合信息	ACTIVITY METRICS BASE

memory_pool_id -“内存池标识”监视元素

内存池标识

表 978. 表函数监视信息

表函数	监视元素收集级别
MON_GET_MEMORY_POOL 表函数 - 获取内存池信息	始终收集

memory_pool_type -“内存池名称”监视元素

内存池的名称。

表 979. 表函数监视信息

表函数	监视元素收集级别
MON_GET_MEMORY_POOL 表函数 - 获取内存池信息	ACTIVITY METRICS BASE

用法

使用 memory_pool_type 元素来标识内存池的类型。此监视元素返回的可能值列示在表 980 中。

表 980. 对 memory_pool_type 返回的可能值.

内存池名称*	描述	其他信息
APM	代理程序池管理 (APM) 堆	内部内存池
APPL_SHARED	应用程序共享堆	内部内存池
APPLICATION	应用程序堆	请参阅 applheapsz -“应用程序堆大小”配置参数。
APS	APS 堆	内部内存池
BSU_CF	基本服务实用程序 (BSU) CF 堆	内部内存池
BSU	基本服务实用程序 (BSU) 堆	内部内存池
BP	缓冲池堆	请参阅 CREATE BUFFERPOOL 语句。
CAT_CACHE	目录高速缓存堆	请参阅 catalogcache_sz -“目录高速缓存大小”配置参数。
DATABASE_CF	Database CF 堆	内部内存池
DATABASE	数据库堆	请参阅 dbheap -“数据库堆”配置参数。
调试	调试堆	内部内存池
DROP_INDEX	删除索引堆	内部内存池
EDU	引擎可分派单元 (EDU) 堆	内部内存池
FCMBP	快速通信管理器 (FCM) 缓冲区堆	请参阅 fcm_num_buffers -“FCM 缓冲区数”配置参数。
FCM_CHANNEL	FCM 通道堆	请参阅 fcm_num_channels -“FCM 通道数”配置参数

表 980. 对 memory_pool_type 返回的可能值 (续).

内存池名称*	描述	其他信息
FCM_CONTROL	FCM 控制堆	内部内存池
FCM_LOCAL	FCM 本地堆	内部内存池
FCM_SESSION	FCM 会话堆	内部内存池
FEDERATED	联合堆	内部内存池
KERNEL _CONTROL	内核控制块堆	内部内存池
KERNEL	内核堆	内部内存池
LOCK_MGR	锁管理器堆	请参阅 locklist -“锁定列表的最大存储量”配置参数。
MISC	其他堆	请参阅 DB2_FMP_COMM_HEAPSZ 注册表变量。
MONITOR	监视器堆	请参阅 mon_heap_sz -“数据库系统监视器堆大小”配置参数。
OPTPROF _PARSER	OptProf XML 解析器堆	内部内存池
OSS_TRACKER	OSS 资源跟踪堆	内部内存池
PERSISTENT _PRIVATE	持久专用堆	内部内存池
PACKAGE _CACHE	程序包高速缓存堆	请参阅 pckcachesz -“程序包高速缓存大小”配置参数。
PRIVATE	专用	内部内存池
RESYNC	再同步堆	内部内存池
SORT	专用排序堆	请参阅 sortheap -“排序堆大小”配置参数。
SHARED_SORT	共享排序堆	请参阅 sheapthres_shr -“共享排序的排序堆阈值”配置参数。
SQL_COMPILER	SQL 编译器堆	内部内存池
STATEMENT	语句堆	请参阅 stmtheap -“语句堆大小”配置参数。
STATISTICS	统计信息堆	请参阅 stat_heap_sz -“统计信息堆大小”配置参数。
USER_DATA	用户数据堆	内部内存池
实用程序	实用程序堆	请参阅 util_heap_sz -“实用程序堆大小”配置参数。
XMLCACHE	XML 高速缓存堆	内部内存池
XMLPARSER	XML 解析器堆	内部内存池

* 由 **db2pd** 返回的这些池的名称可能是缩写格式。这些池的名称以及由 **db2pd** 使用的任何缩写都在 sqlopoolinfo.h 中进行定义。

memory_pool_used -“正在使用的内存池量”监视元素

此内存池正在使用的已落实内存量（以 KB 计）。

表 981. 表函数监视信息

表函数	监视元素收集级别
MON_GET_MEMORY_POOL 表函数 - 获取内存池信息	ACTIVITY METRICS BASE

memory_set_committed -“当前已落实的内存”监视元素

当前已落实到此内存集合的内存量（以 KB 计）。

表 982. 表函数监视信息

表函数	监视元素收集级别
MON_GET_MEMORY_SET 表函数 - 获取内存集合信息	ACTIVITY METRICS BASE

用法

已落实内存是系统上的 RAM 和/或调页空间备份的内存。

memory_set_id -“内存集合标识”监视元素

映射至特定内存集合类型的数字标识。

表 983. 表函数监视信息

表函数	监视元素收集级别
MON_GET_MEMORY_SET 表函数 - 获取内存集合信息	ACTIVITY METRICS BASE

memory_set_size -“内存集合大小”监视元素

最大内存落实限制（以 KB 计）。

此值表示某个内存集合的配置设置或自动管理的那些内存集合的内部计算值。

表 984. 表函数监视信息

表函数	监视元素收集级别
MON_GET_MEMORY_SET 表函数 - 获取内存集合信息	ACTIVITY METRICS BASE

memory_set_type -“内存集合类型”监视元素

内存集合的类型。

表 985. 表函数监视信息

表函数	监视元素收集级别
MON_GET_MEMORY_POOL 表函数 - 获取内存池信息	ACTIVITY METRICS BASE
MON_GET_MEMORY_SET 表函数 - 获取内存集合信息	ACTIVITY METRICS BASE

用法

表 986 中描述了此监视元素返回的可能值:

*表 986. **memory_set_type** 的可能值*

内存集合类型	描述	作用域
DBMS	数据库管理器内存集合	实例
FMP	受防护方式进程内存集合	实例
PRIVATE	专用内存集合	实例
DATABASE	数据库内存集合	数据库
APPLICATION	应用程序内存集合	数据库
FCM	快速通信管理器 (FCM) 内存集合	实例和主机

memory_set_used -“此集合正在使用的内存”监视元素

此集合中已分配到内存池的内存量（以 KB 计）。

表 987. 表函数监视信息

表函数	监视元素收集级别
MON_GET_MEMORY_SET 表函数 - 获取内存集合信息	ACTIVITY METRICS BASE

用法

此监视元素表示的所有内存都是已落实内存；此监视元素的返回值包括在 MEMORY_SET_COMMITTED 中。将高速缓存已落实但未在使用中的任何其他内存以提高性能。

memory_set_used_hwm -“内存集合高水位标记”监视元素

自创建此内存集合以来从此集合中分配到内存池的最大内存量（以 KB 计）。

表 988. 表函数监视信息

表函数	监视元素收集级别
MON_GET_MEMORY_SET 表函数 - 获取内存集合信息	ACTIVITY METRICS BASE

memory_swap_free -“总可用交换空间”监视元素

此主机上未使用的交换空间总量，以 MB 计。

表 989. 表函数监视信息

表函数	监视元素收集级别
ENV_GET_SYSTEM_RESOURCES 表函数 - 返回系统信息	ACTIVITY METRICS BASE

memory_swap_total -“总交换空间”监视元素

此主机上的交换空间总量，以 MB 计。

表 990. 表函数监视信息

表函数	监视元素收集级别
ENV_GET_SYSTEM_RESOURCES 表函数 - 返回系统信息	ACTIVITY METRICS BASE

memory_total -“总物理内存”监视元素

此主机上的物理内存总量，以 MB 计。

表 991. 表函数监视信息

表函数	监视元素收集级别
ENV_GET_SYSTEM_RESOURCES 表函数 - 返回系统信息	ACTIVITY METRICS BASE

message -“控制表消息”

MESSAGE_TIME 列中的时间戳记特征。此元素仅供写至表事件监视器在 CONTROL 表中使用。

表 992. 事件监视信息

事件类型	逻辑数据分组	监视开关
-	-	始终收集

用法

以下是可能的值:

DROPPED RECORDS: *n*
因为无法为其分配 MONHEAP 而被删除的活动记录的数目。

FIRST_CONNECT
数据库激活后第一次连接至数据库的时间。

EVMON_START
EVMONNAME 列中列示的事件监视器启动的时间。

OVERFLOWS: *n*
说明因为缓冲区溢出而删除了 *n* 个记录。

LAST DROPPED RECORD
上次删除活动记录的时间。

message_time -“时间戳记控制表消息”

对应于 MESSAGE 列中描述的事件的时间戳记。此元素仅供写至表事件监视器在 CONTROL 表中使用。

表 993. 事件监视信息

事件类型	逻辑数据分组	监视开关
-	-	始终收集

mon_interval_id -“监视时间间隔标识”监视元素

完成特定事务后 MON_INTERVAL_ID 数据库全局变量的值。

表 994. 表函数监视信息

表函数	监视元素收集级别
MON_GET_INDEX_USAGE_LIST 表函数 - 返回索引用法列表中的信息	ACTIVITY METRICS BASE
MON_GET_TABLE_USAGE_LIST 表函数 - 从表用法列表返回信息	ACTIVITY METRICS BASE

表 995. 事件监视信息

事件类型	逻辑数据分组	监视开关
统计信息	event_qstats	始终收集
统计信息	event_scstats	始终收集
统计信息	event_histogrambin	始终收集
统计信息	event_wcstats	始终收集
统计信息	event_wlstats	始终收集
活动	event_activity	始终收集
工作单元	uow	始终收集

nesting_level -“嵌套级别”监视元素

此元素显示运行语句时生效的嵌套或递归的级别；每个嵌套级别对应一个存储过程或用户定义的函数（UDF）的嵌套或递归调用。

nesting_level 监视元素是 **stmt_nest_level** 监视元素的别名。

表 996. 表函数监视信息

表函数	监视元素收集级别
MON_GET_ACTIVITY_DETAILS 表函数 - 获取完整的活动详细信息（在 XML 文档 DETAILS 中报告）	ACTIVITY METRICS BASE

表 997. 事件监视信息

事件类型	逻辑数据分组	监视开关
锁定	-	-
带有详细信息的死锁的历史记录值[1]	event_stmt_history	-
带有详细信息的死锁的历史记录 [1]	event_stmt_history	-
活动	event_activitystmt	-
工作单元	在程序包列表中报告。	-

1 建议不要使用此事件监视器。建议不要再使用此选项，将来的发行版可能会将其除去。请使用 CREATE EVENT MONITOR FOR LOCKING 语句来监视与锁定相关的事件，例如锁定超时、锁定等待和死锁。

用法

通过将此元素与 **stmt_invocation_id** 监视元素配合使用，可以唯一地标识在其中已经执行了特定 SQL 语句的调用。还可将此元素与其他语句历史记录条目一起使用来了解导致死锁的 SQL 语句的顺序。

network_time_bottom -“语句的最短网络时间”

此元素表示对此 DCS 数据库或在此 DCS 应用程序中执行语句的最短网络时间，或者表示执行使用这么多数据传输的语句的最短网络时间。（网络时间是主机响应时间与语句所耗用的执行时间之差。）

元素标识

network_time_bottom

元素类型

水位标记

表 998. 快照监视信息

快照级别	逻辑数据分组	监视开关
DCS 数据库	dcs_dbase	语句，时间戳记
DCS 应用程序	dcs_appl	语句，时间戳记

表 998. 快照监视信息 (续)

快照级别	逻辑数据分组	监视开关
数据传输	stmt_transmissions	语句，时间戳记

可将快照监视的计数器重置。

用法

使用此元素来更好地了解数据库或应用程序级别的数据库活动和网络流量。

此元素由两个子元素组成，它们报告耗用时间的秒数和微秒（一秒的百万分之一）数。这些子元素的名称可通过将“_s”和“_ms”添加至此监视元素的名称派生而成。要检索此监视元素耗用的总时间，必须将这两个子元素的值加在一起。例如，如果“_s”子元素值为 3，“_ms”子元素值为 20，那么此监视元素耗用的总时间为 3.00002 秒。

network_time_top -“语句的最长网络时间”

此元素表示对此 DCS 数据库或在此 DCS 应用程序中执行语句的最长网络时间，或者表示执行使用这么多数据传输的语句的最长网络时间。（网络时间是主机响应时间与语句所耗用的执行时间之差。）

元素标识

network_time_top

元素类型

水位标记

表 999. 快照监视信息

快照级别	逻辑数据分组	监视开关
DCS 数据库	dcs_dbase	语句，时间戳记
DCS 应用程序	dcs_appl	语句，时间戳记
数据传输	stmt_transmissions	语句，时间戳记

可将快照监视的计数器重置。

用法

使用此元素来更好地了解数据库或应用程序级别的数据库活动和网络流量。注意，时间戳记开关设置为 OFF 时不收集此元素。

此元素由两个子元素组成，它们报告耗用时间的秒数和微秒（一秒的百万分之一）数。这些子元素的名称可通过将“_s”和“_ms”添加至此监视元素的名称派生而成。要检索此监视元素耗用的总时间，必须将这两个子元素的值加在一起。例如，如果“_s”子元素值为 3，“_ms”子元素值为 20，那么此监视元素耗用的总时间为 3.00002 秒。

nleaf -“叶子页数”监视元素

大致的叶子页数。

表 1000. 表函数监视信息

表函数	监视元素收集级别
MON_GET_INDEX 表函数 - 获取索引度量值	ACTIVITY METRICS BASE

nlevels -“索引层数”监视元素

索引层数。这是近似值。

表 1001. 表函数监视信息

表函数	监视元素收集级别
MON_GET_INDEX 表函数 - 获取索引度量值	ACTIVITY METRICS BASE

no_change_updates -“无更改行的更新数”监视元素

导致未更改行的任何列值的行更新数。如果 UPDATE 语句的 SET 子句中包括 LOB、XML 或 LONG 列，那么该语句影响的每行都被认为会更改并且不参与此计数器。

表 1002. 表函数监视信息

表函数	监视元素收集级别
MON_GET_TABLE 表函数 - 获取表度量值	ACTIVITY METRICS BASE

node_number -“节点号”

在 *db2nodes.cfg* 文件中指定给节点的编号。

表 1003. 快照监视信息

快照级别	逻辑数据分组	监视开关
数据库管理器	collected	基本
数据库管理器	memory_pool	基本
数据库管理器	fcm	基本
数据库管理器	fcm_node	基本
数据库管理器	utility_info	基本
数据库	detail_log	基本
缓冲池	bufferpool_nodeinfo	缓冲池
表空间	rollforward	基本
锁定	lock	基本
锁定	lock_wait	基本
数据库	db_sto_path_info	缓冲池

表 1004. 事件监视信息

事件类型	逻辑数据分组	监视开关
连接	event_connheader	始终收集
死锁	lock	始终收集
溢出记录	event_overflow	始终收集
数据库	event_dbmemuse	始终收集
连接	event_connmemuse	始终收集

用法 此值标识当前节点号，在监视多个节点时可使用节点号。

nonboundary_leaf_node_splits -“非边界叶节点分割次数”监视元素

插入操作期间分割非边界叶节点的次数。

表 1005. 表函数监视信息

表函数	监视元素收集级别
MON_GET_INDEX 表函数 - 获取索引度量值	ACTIVITY METRICS BASE

num_agents -“正在处理语句的代理程序数”

当前执行语句或子节的并行代理程序数。

元素标识

num_agents

元素类型

标尺

表 1006. 快照监视信息

快照级别	逻辑数据分组	监视开关
应用程序	stmt	语句
应用程序	subsection	语句

用法 指示查询并行度的指示符。此项对于通过获取连续快照来跟踪查询执行进度非常有用。

num_assoc_agents -“关联代理程序数”

在应用程序级别，此项是与应用程序相关联的子代理程序数。在数据库级别，此项是用于所有应用程序的子代理程序数。

元素标识

num_assoc_agents

元素类型

标尺

表 1007. 快照监视信息

快照级别	逻辑数据分组	监视开关
数据库	dbase	基本
应用程序	appl_info	基本

用法 可使用此元素来帮助您评估代理程序配置参数的设置。

num_compilations -“语句编译次数”

特定 SQL 语句的不同编译的次数。

表 1008. 快照监视信息

快照级别	逻辑数据分组	监视开关
动态 SQL	dynsql	基本

用法 对于某些对不同模式发出的 SQL 语句（例如，**SELECT t1 FROM test**），尽管它们引用不同的存取方案，但却表现为在 DB2 高速缓存中的同一个语句中。将此值与 num_executions 配合使用，以确定不佳编译环境是否会导致动态 SQL 快照统计信息的结果出现偏差。

num_coord_exec -“协调代理程序执行的次数”监视元素

协调代理程序执行此部分的次数。

表 1009. 表函数监视信息

表函数	监视元素收集级别
MON_GET_PKG_CACHE_STMT 表函数 - 获取程序包高速缓存中的 SQL 语句活动度量值	ACTIVITY METRICS BASE
MON_GET_PKG_CACHE_STMT_DETAILS 表函数 - 获取程序包高速缓存条目的详细度量值	ACTIVITY METRICS BASE

表 1010. 事件监视信息

事件类型	逻辑数据分组	监视开关
程序包高速缓存	-	COLLECT BASE DATA

num_coord_exec_with_metrics -“协调代理程序执行的次数以及度量”监视元素

协调代理程序执行此部分的次数以及所捕获的监视度量。

表 1011. 表函数监视信息

表函数	监视元素收集级别
MON_GET_PKG_CACHE_STMT 表函数 - 获取程序包高速缓存中的 SQL 语句活动度量值	ACTIVITY METRICS BASE
MON_GET_PKG_CACHE_STMT_DETAILS 表函数 - 获取程序包高速缓存条目的详细度量值	ACTIVITY METRICS BASE

表 1012. 事件监视信息

事件类型	逻辑数据分组	监视开关
程序包高速缓存	-	COLLECT BASE DATA

num_db_storage_paths -“自动存储器路径数”

与数据库相关联的自动存储器路径的数目。

表 1013. 快照监视信息

快照级别	逻辑数据分组	监视开关
数据库	dbase	基本

用法说明

可将此元素与 db_storage_path 监视元素配合使用来标识与此数据库相关联的存储器路径。

如果使用存储器组，那么此元素仅显示缺省数据库存储器组中的自动存储器路径数。

num_executions -“语句执行次数”监视元素

已执行 SQL 语句的次数。

表 1014. 表函数监视信息

表函数	监视元素收集级别
MON_GET_PKG_CACHE_STMT 表函数 - 获取程序包高速缓存中的 SQL 语句活动度量值	ACTIVITY METRICS BASE
MON_GET_PKG_CACHE_STMT_DETAILS 表函数 - 获取程序包高速缓存条目的详细度量值	ACTIVITY METRICS BASE

表 1015. 快照监视信息

快照级别	逻辑数据分组	监视开关
动态 SQL	dynsql	基本

可将快照监视的计数器重置。

表 1016. 事件监视信息

事件类型	逻辑数据分组	监视开关
程序包高速缓存	-	COLLECT BASE DATA

用法

可使用此元素来标识系统中执行得最频繁的 SQL 语句。

在程序包高速缓存级别，使用此选项来计算对每个语句报告的活动度量值的平均值。例如，通过使用以下公式，可以计算在程序包高速缓存级别报告的语句执行的平均 CPU 使用量:

```
total_cpu_time / num_exec_with_metrics
```

在计算平均值时，请使用 **num_exec_with_metrics** 监视元素来代替 **num_executions** 监视元素，这是因为 **num_executions** 监视元素对语句的所有执行进行计数，而不考虑该语句的执行是否与报告的活动度量值相关。

num_exec_with_metrics -“在收集度量值情况下的执行次数”监视元素

在收集度量值情况下执行此 SQL 语句节的次数。此元素可用于计算程序包高速缓存中各个语句的监视元素的每次执行值。

表 1017. 表函数监视信息

表函数	监视元素收集级别
MON_GET_PKG_CACHE_STMT 表函数 - 获取程序包高速缓存中的 SQL 语句活动度量值	ACTIVITY METRICS BASE
MON_GET_PKG_CACHE_STMT_DETAILS 表函数 - 获取程序包高速缓存条目的详细度量值	ACTIVITY METRICS BASE

表 1018. 事件监视信息

事件类型	逻辑数据分组	监视开关
程序包高速缓存	-	COLLECT BASE DATA

num_extents_left -“尚未处理的扩展数据块数”监视元素

在此表重新平衡过程中尚未移动的扩展数据块数。

表 1019. 表函数监视信息

表函数	监视元素收集级别
MON_GET_EXTENT_MOVEMENT_STATUS - 获取扩展数据块移动进度状态度量值	ACTIVITY METRICS BASE

num_extents_moved -“移动的扩展数据块数”监视元素

在此扩展数据块移动操作期间迄今为止移动的扩展数据块数。

表 1020. 表函数监视信息

表函数	监视元素收集级别
MON_GET_EXTENT_MOVEMENT_STATUS - 获取扩展数据块移动进度状态度量值	ACTIVITY METRICS BASE

num_gw_conn_switches -“连接交换次数”

代理程序池中的代理程序准备好进行连接但被重新分配以与另一 DRDA 数据库配合使用的次数。

表 1021. 快照监视信息

快照级别	逻辑数据分组	监视开关
数据库管理器	db2	基本

用法

对于大多数用户而言，**num_poolagents** 配置参数的缺省设置能确保最优性能。此配置参数的缺省设置自动管理代理程序分组并避免重新分配代理程序。

要减小此监视元素的值，请调整 **num_poolagents** 配置参数的值。

num_indoubt_trans -“不确定事务数”

数据库中的不确定事务的数目。

表 1022. 表函数监视信息

表函数	监视元素收集级别
MON_GET_TRANSACTION_LOG 表函数 - 获取日志信息	ACTIVITY METRICS BASE

表 1023. 快照监视信息

快照级别	逻辑数据分组	监视开关
数据库	dbase	基本

用法 不确定事务具有未落实事务的日志空间，这可能导致日志变满。日志变满时，就不能完成更多事务。此问题的解决办法涉及以启发方式解决不确定事务的手动过程。此监视元素提供当前未完成并且必须以启发方式解决的不确定事务的数目。

num_log_buffer_full -“日志缓冲区变满而导致代理程序等待的次数”监视元素

在将日志记录复制至日志缓冲区时，代理程序必须等待日志数据写入磁盘的次数。每个代理程序每一次发生这种情况时，此值都会递增。例如，如果两个代理程序尝试在缓冲区变满时复制日志数据，那么此值将加上 2。

表 1024. 表函数监视信息

表函数	监视元素收集级别
MON_FORMAT_XML_METRICS_BY_ROW - 获取所有度量值的基于已格式化行的输出	不适用；报告 XML 文档中作为格式化函数的输入提供的所有元素
MON_GET_ACTIVITY_DETAILS 表函数 - 获取完整的活动详细信息（在 XML 文档 DETAILS 中报告）	ACTIVITY METRICS BASE

表 1024. 表函数监视信息 (续)

表函数	监视元素收集级别
MON_GET_CONNECTION 表函数 - 获取连接度量值	REQUEST METRICS BASE
MON_GET_CONNECTION_DETAILS 表函数 - 获取详细的连接度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE
MON_GET_PKG_CACHE_STMT 表函数 - 获取程序包高速缓存中的 SQL 语句活动度量值	ACTIVITY METRICS BASE
MON_GET_PKG_CACHE_STMT_DETAILS 表函数 - 获取程序包高速缓存条目的详细度量值	ACTIVITY METRICS BASE
MON_GET_SERVICE_SUBCLASS 表函数 - 获取服务子类度量值	REQUEST METRICS BASE
MON_GET_SERVICE_SUBCLASS_DETAILS 表函数 - 获取详细的服务子类度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE
MON_GET_TRANSACTION_LOG 表函数 - 获取日志信息	ACTIVITY METRICS BASE
MON_GET_UNIT_OF_WORK 表函数 - 获取工作单元度量值	REQUEST METRICS BASE
MON_GET_UNIT_OF_WORK 表函数 - 获取详细的工作单元度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE
MON_GET_WORKLOAD 表函数 - 获取工作负载度量值	REQUEST METRICS BASE
MON_GET_WORKLOAD_DETAILS 表函数 - 获取详细的工作负载度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE

表 1025. 快照监视信息

快照级别	逻辑数据分组	监视开关
数据库	dbase	基本

可将快照监视的计数器重置。

表 1026. 事件监视信息

事件类型	逻辑数据分组	监视开关
活动	event_activity（在 details_xml 文档中报告）	ACTIVITY METRICS BASE
活动	event_activitymetrics	ACTIVITY METRICS BASE
统计信息	event_scstats（在 details_xml 文档中报告）	REQUEST METRICS BASE
统计信息	event_wlstats（在 details_xml 文档中报告）	REQUEST METRICS BASE
工作单元	在 system_metrics 文档中报告。	REQUEST METRICS BASE

表 1026. 事件监视信息 (续)

事件类型	逻辑数据分组	监视开关
程序包高速缓存	在 activity_metrics 文档中报告。	ACTIVITY METRICS BASE

用法

使用此元素来确定是否需要增大 **logbufsz** 数据库配置参数。

num_log_data_found_in_buffer -“在缓冲区中找到日志数据的次数”

代理程序从缓冲区读取日志数据的次数。从缓冲区读取日志数据比从磁盘读取数据要好一些，这是因为从磁盘读取时速度较慢。

表 1027. 表函数监视信息

表函数	监视元素收集级别
MON_GET_TRANSACTION_LOG 表函数 - 获取日志信息	ACTIVITY METRICS BASE

表 1028. 快照监视信息

快照级别	逻辑数据分组	监视开关
数据库	dbase	基本

可将快照监视的计数器重置。

表 1029. 事件监视信息

事件类型	逻辑数据分组	监视开关
数据库	event_db	始终收集

用法 将此元素与 *num_log_read_io* 元素一起使用来确定是否需要增加 LOGBUFSZ 数据库配置参数的大小。

num_log_part_page_io -“部分日志页写入数”

记录器为了将部分日志数据写至磁盘而发出的 I/O 请求数。

表 1030. 表函数监视信息

表函数	监视元素收集级别
MON_GET_TRANSACTION_LOG 表函数 - 获取日志信息	ACTIVITY METRICS BASE

表 1031. 快照监视信息

快照级别	逻辑数据分组	监视开关
数据库	dbase	基本

可将快照监视的计数器重置。

表 1032. 事件监视信息

事件类型	逻辑数据分组	监视开关
数据库	event_db	始终收集

用法 将此元素与 *log_writes*、*log_write_time* 和 *num_log_write_io* 元素一起使用来确定当前磁盘对于记录操作而言是否够用。

num_log_read_io -“日志读取数”

记录器为了从磁盘读取日志数据而发出的 I/O 请求数。

表 1033. 表函数监视信息

表函数	监视元素收集级别
MON_GET_TRANSACTION_LOG 表函数 - 获取日志信息	ACTIVITY METRICS BASE

表 1034. 快照监视信息

快照级别	逻辑数据分组	监视开关
数据库	dbase	基本

可将快照监视的计数器重置。

表 1035. 事件监视信息

事件类型	逻辑数据分组	监视开关
数据库	event_db	始终收集

用法 将此元素与 *log_reads* 和 *log_read_time* 元素一起使用来确定当前磁盘对于记录操作而言是否够用。

num_log_write_io -“日志写入次数”

记录器为了将日志数据写至磁盘而发出的 I/O 请求数。

表 1036. 表函数监视信息

表函数	监视元素收集级别
MON_GET_TRANSACTION_LOG 表函数 - 获取日志信息	ACTIVITY METRICS BASE

表 1037. 快照监视信息

快照级别	逻辑数据分组	监视开关
数据库	dbase	基本

可将快照监视的计数器重置。

表 1038. 事件监视信息

事件类型	逻辑数据分组	监视开关
数据库	event_db	始终收集

用法 将此元素与 *log_writes* 和 *log_write_time* 元素一起使用来确定当前磁盘对于记录操作而言是否够用。

num_lw_thresh_exceeded -“超过锁定等待阈值的次数”监视元素

此监视元素将报告超过锁定等待阈值（使用 **mon_lw_thresh** 配置参数设置此阈值）并且锁定事件监视器捕获到锁定等待事件的次数。如果未生成锁定等待事件，那么此监视元素不会增大。

表 1039. 表函数监视信息

表函数	监视元素收集级别
MON_FORMAT_XML_METRICS_BY_ROW - 获取所有度量值的基于已格式化行的输出	不适用; 报告 XML 文档中作为格式化函数的输入提供的所有元素
MON_GET_ACTIVITY_DETAILS 表函数 - 获取完整的活动详细信息（在 XML 文档 DETAILS 中报告）	ACTIVITY METRICS BASE
MON_GET_CONNECTION 表函数 - 获取连接度量值	REQUEST METRICS BASE
MON_GET_CONNECTION_DETAILS 表函数 - 获取详细的连接度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE
MON_GET_PKG_CACHE_STMT 表函数 - 获取程序包高速缓存中的 SQL 语句活动度量值	ACTIVITY METRICS BASE
MON_GET_PKG_CACHE_STMT_DETAILS 表函数 - 获取程序包高速缓存条目的详细度量值	ACTIVITY METRICS BASE
MON_GET_SERVICE_SUBCLASS 表函数 - 获取服务子类度量值	REQUEST METRICS BASE
MON_GET_SERVICE_SUBCLASS_DETAILS 表函数 - 获取详细的服务子类度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE
MON_GET_UNIT_OF_WORK 表函数 - 获取工作单元度量值	REQUEST METRICS BASE
MON_GET_UNIT_OF_WORK 表函数 - 获取详细的工作单元度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE
MON_GET_WORKLOAD 表函数 - 获取工作负载度量值	REQUEST METRICS BASE
MON_GET_WORKLOAD_DETAILS 表函数 - 获取详细的工作负载度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE

表 1040. 事件监视信息

事件类型	逻辑数据分组	监视开关
活动	event_activity（在 details_xml 文档中报告）	ACTIVITY METRICS BASE
活动	event_activitymetrics	ACTIVITY METRICS BASE

表 1040. 事件监视信息 (续)

事件类型	逻辑数据分组	监视开关
统计信息	event_scstats（在 details_xml 文档中报告）	REQUEST METRICS BASE
统计信息	event_wlstats（在 details_xml 文档中报告）	REQUEST METRICS BASE
程序包高速缓存	在 activity_metrics 文档中报告。	ACTIVITY METRICS BASE
工作单元	在 system_metrics 文档中报告。	REQUEST METRICS BASE

num_nodes_in_db2_instance -“分区中的节点数”

获取快照的实例上的节点数。

表 1041. 快照监视信息

快照级别	逻辑数据分组	监视开关
数据库管理器	db2	基本

表 1042. 事件监视信息

事件类型	逻辑数据分组	监视开关
事件日志头	event_log_header	始终收集

用法 使用此元素来确定实例的节点数。对于非分区系统数据库，此值将为 1。

num_page_dict_built - 已创建或重新创建的页级别压缩字典数

自最近一次激活数据库以来为表创建或重新创建的页级别压缩字典数。

表 1043. 表函数监视信息

表函数	监视元素收集级别
MON_GET_TABLE 表函数 - 获取表度量值	ACTIVITY METRICS BASE

num_ref_with_metrics -“带有度量值的引用数”监视元素

部分引用数据库对象的总次数。数据对象的用法列表必须已创建并处于活动状态；此外，必须对该部分启用对象度量值的收集。

表 1044. 表函数监视信息

表函数	监视元素收集级别
MON_GET_INDEX_USAGE_LIST 表函数 - 返回索引用法列表中的信息	ACTIVITY METRICS BASE
MON_GET_TABLE_USAGE_LIST 表函数 - 从表用法列表返回信息	ACTIVITY METRICS BASE

num_references -“引用数”监视元素

自此对象添加至列表后，此片段引用此对象的次数。

表 1045. 表函数监视信息

表函数	监视元素收集级别
MON_GET_INDEX_USAGE_LIST 表函数 - 返回索引用法列表中的信息	ACTIVITY METRICS BASE
MON_GET_TABLE_USAGE_LIST 表函数 - 从表用法列表返回信息	ACTIVITY METRICS BASE

num_remaps -“重新映射次数”监视元素

此活动被重新映射的次数。如果 num_remaps 大于零，那么此活动记录的 service_class_id 是此活动上次重新映射到的服务类的标识。

表 1046. 事件监视信息

事件类型	逻辑数据分组	监视开关
活动	event_activity	始终收集

用法

使用此信息来验证此活动是否已被映射期望的次数。

num_tbsps -“表空间数”监视元素

与已记录事件相关联的表空间的数目。

表 1047. 事件监视信息

事件类型	逻辑数据分组	监视开关
变更历史记录	utilstart	始终收集

num_threshold_violations -“阈值违例次数”监视元素

自最后一次激活此数据库后，在其中发生的阈值违例次数。

此监视元素是 第 1223 页的『 thresh_violations -“阈值违例次数”监视元素 』 监视元素的别名，后者由某些监视 (MON_*) 表函数返回。

表 1048. 快照监视信息

快照级别	逻辑数据分组	监视开关
数据库	dbase	基本

可将快照监视的计数器重置。

表 1049. 事件监视信息

事件类型	逻辑数据分组	监视开关
数据库	event_db	始终收集

用法

此元素可用来帮助确定阈值对此特定应用程序是否有效，或者阈值违例是否过多。

num_transmissions -“传输次数”

DB2 Connect 网关与用于处理此 DCS 语句的主机之间的数据传输次数。（一次数据传输包括一次发送或一次接收）。

注:

这是旧的监视元素，DB2 UDB V8.1.2 或更高版本中已不再使用此元素。如果在使用 DB2 UDB V8.1.2 或更高版本，那么参阅 **num_transmissions_group** 监视元素。

元素标识

num_transmissions

元素类型

计数器

-->

表 1050. 快照监视信息

快照级别	逻辑数据分组	监视开关
DCS 语句	dcs_stmt	语句

用法 使用此元素来更好地了解特定语句执行时间很长的原因。例如，返回的结果集很大的查询可能需要许多次数据传输才能完成。

num_transmissions_group -“传输组数目”

DB2 Connect 网关与用于处理此 DCS 语句的主机之间的数据传输范围。（一次数据传输包括一次发送或一次接收）。

表 1051. 快照监视信息

快照级别	逻辑数据分组	监视开关
DCS 语句	dcs_stmt	语句

用法 使用此元素来更好地了解特定语句执行时间很长的原因。例如，返回的结果集很大的查询可能需要许多次数据传输才能完成。

下面描述了表示传输范围的常量，它们是在 sqlmon.h 中定义的。

API 常量	描述
SQLM_DCS_TRANS_GROUP_2	2 次传输
SQLM_DCS_TRANS_GROUP_3TO7	3 到 7 次传输
SQLM_DCS_TRANS_GROUP_8TO15	8 到 15 次传输
SQLM_DCS_TRANS_GROUP_16TO64	16 到 64 次传输
SQLM_DCS_TRANS_GROUP_GT64	大于 64 次传输

number_in_bin -“条形中的数目”监视元素

此元素为直方图条形内的活动数或请求数。

表 1052. 事件监视信息

事件类型	逻辑数据分组	监视开关
统计信息	event_histogrambin	-

用法

使用此元素来表示直方图中条形的高度。

object_data_gbp_indep_pages_found_in_lbp -“本地缓冲池中发现的独立于组缓冲池的数据页数”监视元素

代理程序在本地缓冲池 (LBP) 中发现的独立于组缓冲池 (GBP) 的数据页数。

表 1053. 表函数监视信息

表函数	监视元素收集级别
MON_GET_TABLE 表函数 - 获取表度量值	DATA OBJECT METRICS EXTENDED
MON_GET_TABLE_USAGE_LIST 表函数 - 从表用法列表返回信息	DATA OBJECT METRICS EXTENDED

表 1054. 事件监视信息

事件类型	逻辑数据分组	监视开关
活动	event_activitymetrics	ACTIVITY METRICS BASE
程序包高速缓存	pkgcache_metrics	ACTIVITY METRICS BASE
工作单元	uow_metrics	REQUEST METRICS BASE

object_data_gbp_invalid_pages -“表的无效 GBP 数据页数”监视元素

针对表从组缓冲池 (GBP) 请求数据页的次数。请求此页是因为本地缓冲池 (LBP) 中的此页版本无效。在 DB2 pureScale 环境外部，此值为空。

表 1055. 表函数监视信息

表函数	监视元素收集级别
MON_GET_TABLE 表函数 - 获取表度量值	DATA OBJECT METRICS EXTENDED
MON_GET_TABLE_USAGE_LIST 表函数 - 从表用法列表返回信息	DATA OBJECT METRICS EXTENDED

用法

要确定在 LBP 中发现所请求数据页的频率，请使用以下公式（该公式使用监视元素的值）：

```
(object_data_lbp_pages_found + object_xda_lbp_pages_found)
 / (object_data_l_reads + object_xda_l_reads)
```

要确定在 GBP 中发现所请求数据页的频率，请使用以下公式（该公式也使用监视元素的值）：

```
(object_data_GBP_l_reads + object_xda_l_reads - object_data_GBP_p_reads
 - object_xda_p_reads) /
(object_data_GBP_l_reads + object_xda_l_reads)
```

LBP 和 GBP 命中率是集群高速缓存设施的整体性能的重要因子。使用这些公式可帮助您确定 LBP 或 GBP 是否会限制数据库吞吐量。

object_data_gbp_l_reads -“表的 GBP 数据逻辑读取数”监视元素

针对表从 GBP 请求依赖于组缓冲池 (GBP) 的数据页的次数。请求此页是因为本地缓冲池 (LBP) 中没有此页的有效版本。

表 1056. 表函数监视信息

表函数	监视元素收集级别
MON_GET_TABLE 表函数 - 获取表度量值	DATA OBJECT METRICS EXTENDED
MON_GET_TABLE_USAGE_LIST 表函数 - 从表用法列表返回信息	DATA OBJECT METRICS EXTENDED

用法

要确定在 LBP 中发现所请求数据页的频率，请使用以下公式（该公式使用监视元素的值）：

```
(object_data_lbp_pages_found + object_xda_lbp_pages_found)
 / (object_data_l_reads + object_xda_l_reads)
```

要确定在 GBP 中发现所请求数据页的频率，请使用以下公式（该公式也使用监视元素的值）：

```
(object_data_GBP_l_reads + object_xda_l_reads - object_data_GBP_p_reads
 - object_xda_p_reads) /
(object_data_GBP_l_reads + object_xda_l_reads)
```

LBP 和 GBP 命中率是集群高速缓存设施的整体性能的重要因子。使用这些公式可帮助您确定 LBP 或 GBP 是否会限制数据库吞吐量。

object_data_gbp_p_reads -“表的 GBP 数据物理读取数”监视元素

针对表将依赖于组缓冲池 (GBP) 的数据页从磁盘读取到本地缓冲池 (LBP) 中的次数。将此页从磁盘读取到 LBP 中是因为该页不在 GBP 中。

表 1057. 表函数监视信息

表函数	监视元素收集级别
MON_GET_TABLE 表函数 - 获取表度量值	DATA OBJECT METRICS EXTENDED
MON_GET_TABLE_USAGE_LIST 表函数 - 从表用法列表返回信息	DATA OBJECT METRICS EXTENDED

用法

要确定在 LBP 中发现所请求数据页的频率，请使用以下公式（该公式使用监视元素的值）：

```
(object_data_lbp_pages_found + object_xda_lbp_pages_found)
 / (object_data_l_reads + object_xda_l_reads)
```

要确定在 GBP 中发现所请求数据页的频率，请使用以下公式（该公式也使用监视元素的值）：

```
(object_data_GBP_l_reads + object_xda_l_reads - object_data_GBP_p_reads
 - object_xda_p_reads) /
(object_data_GBP_l_reads + object_xda_l_reads)
```

LBP 和 GBP 命中率是集群高速缓存设施的整体性能的重要因子。使用这些公式可帮助您确定 LBP 或 GBP 是否会限制数据库吞吐量。

object_data_lbp_pages_found -“发现表的 LBP 数据页数”监视元素

本地缓冲池 (LBP) 中某个表的某个数据页的出现次数。

表 1058. 表函数监视信息

表函数	监视元素收集级别
MON_GET_TABLE 表函数 - 获取表度量值	DATA OBJECT METRICS EXTENDED
MON_GET_TABLE_USAGE_LIST 表函数 - 从表用法列表返回信息	DATA OBJECT METRICS EXTENDED

用法

要确定在 LBP 中发现所请求数据页的频率，请使用以下公式（该公式使用监视元素的值）：

```
(object_data_lbp_pages_found + object_xda_lbp_pages_found)
 / (object_data_l_reads + object_xda_l_reads)
```

要确定在 GBP 中发现所请求数据页的频率，请使用以下公式（该公式也使用监视元素的值）：

```
(object_data_GBP_l_reads + object_xda_l_reads - object_data_GBP_p_reads
 - object_xda_p_reads) /
(object_data_GBP_l_reads + object_xda_l_reads)
```

LBP 和 GBP 命中率是集群高速缓存设施的整体性能的重要因子。使用这些公式可帮助您确定 LBP 或 GBP 是否会限制数据库吞吐量。

object_data_l_reads -“表的缓冲池数据逻辑读取数”监视元素

逻辑上从缓冲池读取的某个表的数据页数。

表 1059. 表函数监视信息

表函数	监视元素收集级别
MON_GET_TABLE 表函数 - 获取表度量值	DATA OBJECT METRICS EXTENDED
MON_GET_TABLE_USAGE_LIST 表函数 - 从表用法列表返回信息	DATA OBJECT METRICS EXTENDED

用法

此监视元素跟踪访问以下数据的次数:

- 数据库管理器 需要处理该页时，数据位于缓冲池中
- 必须先将该数据读取到缓冲池中，数据库管理器 才能处理该页

通过使用以下公式来计算数据页命中率（该公式使用监视元素的值）:

```
(1 - (object_data_p_reads + object_xda_p_reads) /
(object_data_l_reads + object_xda_l_reads))
```

如果命中率很低，那么提高缓冲池页数可以改进性能。

object_data_p_reads -“表的缓冲池物理数据读取数”监视元素

物理上读取的某个表的数据页数。

表 1060. 表函数监视信息

表函数	监视元素收集级别
MON_GET_TABLE 表函数 - 获取表度量值	DATA OBJECT METRICS EXTENDED
MON_GET_TABLE_USAGE_LIST 表函数 - 从表用法列表返回信息	DATA OBJECT METRICS EXTENDED

用法

通过使用以下公式来计算数据页命中率（该公式使用监视元素的值）:

```
(1 - (object_data_p_reads + object_xda_p_reads) /
(object_data_l_reads + object_xda_l_reads))
```

如果命中率很低，那么提高缓冲池页数可以改进性能。

object_index_gbp_indep_pages_found_in_lbp -“本地缓冲池中发现的独立于组缓冲池的索引页数”监视元素

代理程序在本地缓冲池 (LBP) 中发现的独立于组缓冲池 (GBP) 的索引页数。

表 1061. 表函数监视信息

表函数	监视元素收集级别
MON_GET_INDEX 表函数 - 获取索引度量值	DATA OBJECT METRICS EXTENDED
MON_GET_INDEX_USAGE_LIST 表函数 - 返回索引用法列表中的信息	DATA OBJECT METRICS EXTENDED

表 1062. 事件监视信息

事件类型	逻辑数据分组	监视开关
活动	event_activitymetrics	ACTIVITY METRICS BASE
程序包高速缓存	pkgcache_metrics	ACTIVITY METRICS BASE
工作单元	uow_metrics	REQUEST METRICS BASE

object_index_gbp_invalid_pages -“索引的无效 GBP 索引页数”监视元素

针对索引从组缓冲池 (GBP) 请求索引页的次数。请求此页是因为本地缓冲池 (LBP) 中的此页版本无效。

表 1063. 表函数监视信息

表函数	监视元素收集级别
MON_GET_INDEX 表函数 - 获取索引度量值	DATA OBJECT METRICS EXTENDED
MON_GET_INDEX_USAGE_LIST 表函数 - 返回索引用法列表中的信息	DATA OBJECT METRICS EXTENDED

用法

要确定在 LBP 中发现所请求索引页的频率，请使用以下公式（该公式使用监视元素的值）：

```
object_index_lbp_pages_found / object_index_l_reads
```

要确定在 GBP 中发现所请求索引页的次数，请使用以下公式（该公式也使用监视元素的值）：

```
(object_index_gbp_l_reads - object_index_gbp_p_reads) / object_index_gbp_l_reads
```

LBP 和 GBP 命中率是集群高速缓存设施的整体性能的重要因子。使用这些公式可帮助您确定 LBP 或 GBP 是否会限制数据库吞吐量。

object_index_gbp_l_reads -“索引的 GBP 索引逻辑读取数”监视元素

针对索引从 GBP 请求依赖于组缓冲池 (GBP) 的索引页的次数。请求此页是因为本地缓冲池 (LBP) 中没有此页的有效版本。

表 1064. 表函数监视信息

表函数	监视元素收集级别
MON_GET_INDEX 表函数 - 获取索引度量值	DATA OBJECT METRICS EXTENDED
MON_GET_INDEX_USAGE_LIST 表函数 - 返回索引用法列表中的信息	DATA OBJECT METRICS EXTENDED

用法

要确定在 LBP 中发现所请求索引页的频率，请使用以下公式（该公式使用监视元素的值）：

```
object_index_lbp_pages_found / object_index_l_reads
```

要确定在 GBP 中发现所请求索引页的次数，请使用以下公式（该公式也使用监视元素的值）：

```
(object_index_gbp_l_reads - object_index_gbp_p_reads) / object_index_gbp_l_reads
```

LBP 和 GBP 命中率是集群高速缓存设施的整体性能的重要因子。使用这些公式可帮助您确定 LBP 或 GBP 是否会限制数据库吞吐量。

object_index_gbp_p_reads -“索引的 GBP 索引物理读取数”监视元素

针对索引将依赖于组缓冲池 (GBP) 的索引页从磁盘读取到本地缓冲池 (LBP) 中的次数。将此页从磁盘读取到 LBP 中是因为该页不在 GBP 中。

表 1065. 表函数监视信息

表函数	监视元素收集级别
MON_GET_INDEX 表函数 - 获取索引度量值	DATA OBJECT METRICS EXTENDED
MON_GET_INDEX_USAGE_LIST 表函数 - 返回索引用法列表中的信息	DATA OBJECT METRICS EXTENDED

用法

要确定在 LBP 中发现所请求索引页的频率，请使用以下公式（该公式使用监视元素的值）：

```
object_index_lbp_pages_found / object_index_l_reads
```

要确定在 GBP 中发现所请求索引页的次数，请使用以下公式（该公式也使用监视元素的值）：

```
(object_index_gbp_l_reads - object_index_gbp_p_reads) / object_index_gbp_l_reads
```

LBP 和 GBP 命中率是集群高速缓存设施的整体性能的重要因子。使用这些公式可帮助您确定 LBP 或 GBP 是否会限制数据库吞吐量。

object_index_lbp_pages_found -“发现索引的 LBP 索引页数”监视元素

本地缓冲池 (LBP) 中某个索引的某个索引页的出现次数。

表 1066. 表函数监视信息

表函数	监视元素收集级别
MON_GET_INDEX 表函数 - 获取索引度量值	DATA OBJECT METRICS EXTENDED
MON_GET_INDEX_USAGE_LIST 表函数 - 返回索引用法列表中的信息	DATA OBJECT METRICS EXTENDED

用法

要确定在 LBP 中发现所请求索引页的频率，请使用以下公式（该公式使用监视元素的值）：

```
object_index_lbp_pages_found / object_index_l_reads
```

要确定在 GBP 中发现所请求索引页的次数，请使用以下公式（该公式也使用监视元素的值）：

```
(object_index_gbp_l_reads - object_index_gbp_p_reads) / object_index_gbp_l_reads
```

LBP 和 GBP 命中率是集群高速缓存设施的整体性能的重要因子。使用这些公式可帮助您确定 LBP 或 GBP 是否会限制数据库吞吐量。

object_index_l_reads -“索引的缓冲池索引逻辑读取数”监视元素

逻辑上从缓冲池读取的某个索引的索引页数。

表 1067. 表函数监视信息

表函数	监视元素收集级别
MON_GET_INDEX 表函数 - 获取索引度量值	DATA OBJECT METRICS EXTENDED
MON_GET_INDEX_USAGE_LIST 表函数 - 返回索引用法列表中的信息	DATA OBJECT METRICS EXTENDED

用法

此监视元素跟踪访问以下页的次数:

- 数据库管理器 需要处理这些页时，索引页位于缓冲池中
- 必须先将索引页读取到缓冲池中，数据库管理器 才能处理这些页

通过使用以下公式来计算索引页命中率（该公式使用监视元素的值）:

```
(1 - (object_index_p_reads / object_index_l_reads))
```

如果命中率很低，那么提高缓冲池页数可以改进性能。

object_index_p_reads - 索引的缓冲池索引物理读取数

物理上读取的某个索引的数据页数。

表 1068. 表函数监视信息

表函数	监视元素收集级别
MON_GET_INDEX 表函数 - 获取索引度量值	DATA OBJECT METRICS EXTENDED
MON_GET_INDEX_USAGE_LIST 表函数 - 返回索引用法列表中的信息	DATA OBJECT METRICS EXTENDED

用法

通过使用以下公式来计算索引页命中率（该公式使用监视元素的值）:

```
(1 - (object_index_p_reads / object_index_l_reads))
```

如果命中率很低，那么提高缓冲池页数可以改进性能。

object_name -“对象名”监视元素

表名或索引名（取决于 **objtype** 监视元素的值）。

表 1069. 表函数监视信息

表函数	监视元素收集级别
MON_GET_AUTO_MAINT_QUEUE 表函数 - 获取有关自动维护作业的信息	ACTIVITY METRICS BASE
MON_GET_AUTO_RUNSTATS_QUEUE 表函数 - 检索有关为求值而排队的对象的信息	ACTIVITY METRICS BASE

表 1069. 表函数监视信息 (续)

表函数	监视元素收集级别
MON_GET_RTS_RQST 表函数 - 检索有关实时统计请求的信息	ACTIVITY METRICS BASE
MON_GET_USAGE_LIST_STATUS 表函数 - 返回用法列表的状态	ACTIVITY METRICS BASE

表 1070. 事件监视信息

事件类型	逻辑数据分组	监视开关
变更历史记录	utilphase utilstart	始终收集

用法

对于变更历史记录事件监视器，如果 `object_type` 元素为 INDEX、PARTIONGROUP 或 TABLE，那么这是索引、分区组或表的名称。

object_requested -“所请求对象”监视元素

请求者尝试从所有者处获取的锁定的类型。值可以是 LOCK（对于数据库锁定）或 TICKET（对于 WLM 凭单）。

表 1071. 事件监视信息

事件类型	逻辑数据分组	监视开关
锁定	lock_participants	

object_schema -“对象模式”监视元素

表模式或索引模式（取决于 **objtype** 监视元素的值）。

表 1072. 表函数监视信息

表函数	监视元素收集级别
MON_GET_AUTO_MAINT_QUEUE 表函数 - 获取有关自动维护作业的信息	ACTIVITY METRICS BASE
MON_GET_AUTO_RUNSTATS_QUEUE 表函数 - 检索有关为求值而排队的对象的信息	ACTIVITY METRICS BASE
MON_GET_RTS_RQST 表函数 - 检索有关实时统计请求的信息	ACTIVITY METRICS BASE
MON_GET_USAGE_LIST_STATUS 表函数 - 返回用法列表的状态	ACTIVITY METRICS BASE

表 1073. 事件监视信息

事件类型	逻辑数据分组	监视开关
变更历史记录	utilphase utilstart	始终收集

用法

对于变更历史记录事件监视器，如果 object_type 元素为 INDEX 或 TABLE，那么这是索引或表的模式，否则为空字符串。

object_xda_gbp_indep_pages_found_in_lbp -“本地缓冲池中发现的独立于组缓冲池的 XDA 页数”监视元素

代理程序在本地缓冲池 (LBP) 中发现的独立于组缓冲池 (GBP) 的 XML 存储器对象 (XDA) 数据页数。

表 1074. 表函数监视信息

表函数	监视元素收集级别
MON_GET_TABLE 表函数 - 获取表度量值	DATA OBJECT METRICS EXTENDED
MON_GET_TABLE_USAGE_LIST 表函数 - 从表用法列表返回信息	DATA OBJECT METRICS EXTENDED

表 1075. 事件监视信息

事件类型	逻辑数据分组	监视开关
活动	event_activitymetrics	ACTIVITY METRICS BASE
程序包高速缓存	pkgcache_metrics	ACTIVITY METRICS BASE
工作单元	uow_metrics	REQUEST METRICS BASE

object_xda_gbp_invalid_pages -“表的无效 GBP XDA 数据页数”监视元素

针对表从组缓冲池 (GBP) 请求 XML 存储器对象 (XDA) 的数据页的次数。请求此页是因为本地缓冲池 (LBP) 中的此页版本无效。

表 1076. 表函数监视信息

表函数	监视元素收集级别
MON_GET_TABLE 表函数 - 获取表度量值	DATA OBJECT METRICS EXTENDED
MON_GET_TABLE_USAGE_LIST 表函数 - 从表用法列表返回信息	DATA OBJECT METRICS EXTENDED

用法

要确定在 LBP 中发现所请求 XDA 页的频率，请使用以下公式（该公式使用监视元素的值）：

```
object_xda_lbp_pages_found / object_xda_l_reads
```

要确定在 GBP 中发现所请求 XDA 页的次数，请使用以下公式（该公式也使用监视元素的值）：

```
(object_xda_gbp_l_reads - object_xda_gbp_p_reads) / object_xda_gbp_l_reads
```

LBP 和 GBP 命中率是集群高速缓存设施的整体性能的重要因子。使用这些公式可帮助您确定 LBP 或 GBP 是否会限制数据库吞吐量。

object_xda_gbp_l_reads -“表的 GBP XDA 数据逻辑读取请求数”监视元素

针对表从 GBP 请求 XML 存储器对象 (XDA) 的独立于组缓冲池 (GBP) 的数据页的次数。请求此页是因为本地缓冲池 (LBP) 中没有此页的有效版本。

表 1077. 表函数监视信息

表函数	监视元素收集级别
MON_GET_TABLE 表函数 - 获取表度量值	DATA OBJECT METRICS EXTENDED
MON_GET_TABLE_USAGE_LIST 表函数 - 从表用法列表返回信息	DATA OBJECT METRICS EXTENDED

用法

要确定在 LBP 中发现所请求 XDA 页的频率，请使用以下公式（该公式使用监视元素的值）：

```
object_xda_lbp_pages_found / object_xda_l_reads
```

要确定在 GBP 中发现所请求 XDA 页的次数，请使用以下公式（该公式也使用监视元素的值）：

```
(object_xda_gbp_l_reads - object_xda_gbp_p_reads) / object_xda_gbp_l_reads
```

LBP 和 GBP 命中率是集群高速缓存设施的整体性能的重要因子。使用这些公式可帮助您确定 LBP 或 GBP 是否会限制数据库吞吐量。

object_xda_gbp_p_reads -“表的 GBP XDA 数据物理读取请求数”监视元素

针对表将 XML 存储器对象 (XDA) 的依赖于组缓冲池 (GBP) 的数据页从磁盘读取到本地缓冲池 (LBP) 中的次数。将此页从磁盘读取到 LBP 中是因为该页不在 GBP 中。

表 1078. 表函数监视信息

表函数	监视元素收集级别
MON_GET_TABLE 表函数 - 获取表度量值	DATA OBJECT METRICS EXTENDED
MON_GET_TABLE_USAGE_LIST 表函数 - 从表用法列表返回信息	DATA OBJECT METRICS EXTENDED

用法

要确定在 LBP 中发现所请求 XDA 页的频率，请使用以下公式（该公式使用监视元素的值）：

```
object_xda_lbp_pages_found / object_xda_l_reads
```

要确定在 GBP 中发现所请求 XDA 页的次数，请使用以下公式（该公式也使用监视元素的值）：

```
(object_xda_gbp_l_reads - object_xda_gbp_p_reads) / object_xda_gbp_l_reads
```

LBP 和 GBP 命中率是集群高速缓存设施的整体性能的重要因子。使用这些公式可帮助您确定 LBP 或 GBP 是否会限制数据库吞吐量。

object_xda_lbp_pages_found -“发现表的 LBP XDA 数据页数”监视元素

本地缓冲池 (LBP) 中某个表的某个 XML 存储器对象 (XDA) 数据页的出现次数。

表 1079. 表函数监视信息

表函数	监视元素收集级别
MON_GET_TABLE 表函数 - 获取表度量值	DATA OBJECT METRICS EXTENDED
MON_GET_TABLE_USAGE_LIST 表函数 - 从表用法列表返回信息	DATA OBJECT METRICS EXTENDED

用法

要确定在 LBP 中发现所请求数据页的频率，请使用以下公式（该公式使用监视元素的值）：

```
(object_data_lbp_pages_found + object_xda_lbp_pages_found)
 / (object_data_l_reads + object_xda_l_reads)
```

要确定在 GBP 中发现所请求数据页的频率，请使用以下公式（该公式也使用监视元素的值）：

```
(object_data_GBP_l_reads + object_xda_l_reads - object_data_GBP_p_reads
 - object_xda_p_reads) /
(object_data_GBP_l_reads + object_xda_l_reads)
```

LBP 和 GBP 命中率是集群高速缓存设施的整体性能的重要因子。使用这些公式可帮助您确定 LBP 或 GBP 是否会限制数据库吞吐量。

object_xda_l_reads -“表的缓冲池 XDA 数据逻辑读取数”监视元素

逻辑上从缓冲池读取的某个表的 XML 存储器对象 (XDA) 数据页数。

表 1080. 表函数监视信息

表函数	监视元素收集级别
MON_GET_TABLE 表函数 - 获取表度量值	DATA OBJECT METRICS EXTENDED
MON_GET_TABLE_USAGE_LIST 表函数 - 从表用法列表返回信息	DATA OBJECT METRICS EXTENDED

用法

此监视元素跟踪访问以下数据的次数:

- 数据库管理器 需要处理该页时，该数据位于缓冲池中
- 必须先将该数据读取到缓冲池中，数据库管理器 才能处理该页

通过使用以下公式来计算数据页命中率（该公式使用监视元素的值）：

```
(1 - (object_data_p_reads + object_xda_p_reads) /
(object_data_l_reads + object_xda_l_reads))
```

如果命中率很低，那么提高缓冲池页数可以改进性能。

object_xda_p_reads -“表的缓冲池 XDA 数据物理读取数”监视元素

物理上读取的某个表的 XML 存储器对象 (XDA) 数据页数。

表 1081. 表函数监视信息

表函数	监视元素收集级别
MON_GET_TABLE 表函数 - 获取表度量值	DATA OBJECT METRICS EXTENDED
MON_GET_TABLE_USAGE_LIST 表函数 - 从表用法列表返回信息	DATA OBJECT METRICS EXTENDED

用法

通过使用以下公式来计算数据页命中率（该公式使用监视元素的值）：

```
(1 - (object_data_p_reads + object_xda_p_reads) /
(object_data_l_reads + object_xda_l_reads))
```

如果命中率很低，那么提高缓冲池页数可以改进性能。

objtype -“对象类型”监视元素

正报告其监视数据的对象的类型。此监视元素是 object_type 监视元素的别名。

表 1082. 表函数监视信息

表函数	监视元素收集级别
ADMIN_GET_TAB_COMPRESS_INFO 表函数 - 估算压缩节省量	ACTIVITY METRICS BASE
ADMIN_GET_TAB_DICTIONARY_INFO 表函数 - 报告现有表字典的属性	ACTIVITY METRICS BASE
MON_GET_AUTO_MAINT_QUEUE 表函数 - 获取有关自动维护作业的信息	ACTIVITY METRICS BASE
MON_GET_AUTO_RUNSTATS_QUEUE 表函数 - 检索有关为求值而排队的对象的信息	ACTIVITY METRICS BASE
MON_GET_PAGE_ACCESS_INFO 表函数 - 获取缓冲池页等待信息	ACTIVITY METRICS BASE
MON_GET_RTS_RQST 表函数 - 检索有关实时统计请求的信息	ACTIVITY METRICS BASE
MON_GET_USAGE_LIST_STATUS 表函数 - 返回用法列表的状态	ACTIVITY METRICS BASE
PD_GET_DIAG_HIST 表函数 - 从给定设施返回记录	ACTIVITY METRICS BASE

表 1083. 事件监视信息

事件类型	逻辑数据分组	监视开关
变更历史记录	utilphase utilstart	始终收集

使用说明

- ADMIN_GET_TAB_COMPRESS_INFO 表函数或 ADMIN_GET_TAB_DICTIONARY_INFO 表函数返回此元素时，object_type 监视元素的返回值可以是“XML”或“DATA”。
- MON_GET_RTS_RQST 表函数返回此元素时，object_type 监视元素的返回值为“TABLE”。
- MON_GET_AUTO_MAINT_QUEUE 表函数返回此元素时，object_type 监视元素的返回值可以是“DATABASE”、“TABLE”、“NICKNAME”或“VIEW”。
- MON_GET_AUTO_RUNSTATS_QUEUE 表函数返回此元素时，object_type 监视元素的返回值可以是“TABLE”、“NICKNAME”或“VIEW”。
- 变更历史记录事件监视器返回此元素时，object_type 监视元素的返回值可以是“DATABASE”、“INDEX”、“PARTITIONGROUP”、“TABLE”或“TABLESPACE”。

olap_func_overflows -“OLAP 函数溢出次数”监视元素

OLAP 函数数据超过可用排序堆空间的次数。

表 1084. 快照监视信息

快照级别	逻辑数据分组	监视开关
数据库	dbase	基本
应用程序	appl	基本

可将快照监视的计数器重置。

表 1085. 事件监视信息

事件类型	逻辑数据分组	监视开关
数据库	event_db	始终收集
连接	event_conn	始终收集

用法

在数据库级别，将此元素与 total_olap_funcs 配合使用以计算溢出至磁盘的 OLAP 函数的百分比。如果此百分比很高，并且使用 OLAP 函数的应用程序的性能需要提高，那么您应该考虑增加排序堆大小。

在应用程序级别，使用此元素来评估各个应用程序的 OLAP 函数性能。

open_cursors -“打开的游标数”

当前对应用程序打开的游标数。

表 1086. 快照监视信息

快照级别	逻辑数据分组	监视开关
DCS 应用程序	dcs_appl	语句

用法 使用此元素来评估分配的内存量。DB2 客户机、DB2 Connect 或目标数据库上的数据库代理分配的内存量与当前打开的游标数有关。了解此信息可帮助您进

行容量规划。例如，每个执行分块的打开游标的缓冲区大小为 RQRIOBLK。如果启用了 *deferred_prepare*，那么将分配两个缓冲区。

此元素不包括之前的关闭操作关闭的游标。之前的关闭操作是在主机数据库将最后一个记录返回至客户机时发生的。游标在主机和网关上关闭，但在客户机上仍然是打开的。可使用 DB2 调用级接口来设置之前的关闭游标。

open_loc_curs -“打开的本地游标数”

当前对此应用程序打开的本地游标数，包括 *open_loc_curs_blk* 计数的那些游标数。

元素标识

open_loc_curs

元素类型

标尺

表 1087. 快照监视信息

快照级别	逻辑数据分组	监视开关
应用程序	appl	基本

用法 可将此元素与 *open_loc_curs_blk* 一起使用来计算作为分块游标的本地游标的百分比。如果该百分比很低，那么可通过改进应用程序中的行分块来改进性能。

有关远程应用程序使用的游标，请参阅 *open_rem_curs*。

open_loc_curs_blk -“打开的本地分块游标数”

当前对此应用程序打开的本地分块游标数。

元素标识

open_loc_curs_blk

元素类型

标尺

表 1088. 快照监视信息

快照级别	逻辑数据分组	监视开关
应用程序	appl	基本

用法 可将此元素与 *open_loc_curs* 一起使用来计算作为分块游标的本地游标的百分比。如果该百分比很低，那么可通过改进应用程序中的行分块来改进性能:

- 检查记录分块的预编译选项以了解模糊游标的处理方式
- 重新定义游标以允许进行分块（例如，如果可能对游标指定 FOR FETCH ONLY）。

rej_curs_blk 和 acc_curs_blk 提供了附加信息，这些信息可帮助您调整配置参数以改进应用程序中的行分块。

有关远程应用程序使用的分块游标，请参阅 *open_rem_curs_blk*。

open_rem_curs -“打开的远程游标数”

当前对此应用程序打开的远程游标数，包括 *open_rem_curs_blk* 计数的那些游标数。

元素标识

open_rem_curs

元素类型

标尺

表 1089. 快照监视信息

快照级别	逻辑数据分组	监视开关
应用程序	appl	基本

用法 可将此元素与 *open_rem_curs_blk* 配合使用以计算作为分块游标的远程游标的百分比。如果该百分比很低，那么可通过改进应用程序中的行分块来改进性能。有关更多信息，请参阅 *open_rem_curs_blk*。

有关应用程序用于连接至本地数据库的打开的游标数，请参阅 *open_loc_curs*。

open_rem_curs_blk -“打开的远程分块游标数”

当前对此应用程序打开的远程分块游标数。

元素标识

open_rem_curs_blk

元素类型

标尺

表 1090. 快照监视信息

快照级别	逻辑数据分组	监视开关
应用程序	appl	基本

用法 可将此元素与 *open_rem_curs* 配合使用以计算作为分块游标的远程游标的百分比。如果该百分比很低，那么可通过改进应用程序中的行分块来改进性能:

- 检查记录分块的预编译选项以了解模糊游标的处理方式
- 重新定义游标以允许进行分块（例如，如果可能对游标指定 FOR FETCH ONLY）。

rej_curs_blk 和 *acc_curs_blk* 提供了附加信息，这些信息可帮助您调整配置参数以改进应用程序中的行分块。

有关应用程序用来连接至本地数据库的打开的分块游标的数目，请参阅 *open_loc_curs_blk*。

os_level -“操作系统级别”监视元素

在此主机上运行的操作系统的修改级别。仅对 Linux 系统报告。

表 1091. 表函数监视信息

表函数	监视元素收集级别
ENV_GET_SYSTEM_RESOURCES 表函数 - 返回系统信息	ACTIVITY METRICS BASE

os_name -“操作系统名称”监视元素

在此主机上运行的操作系统的名称。

表 1092. 表函数监视信息

表函数	监视元素收集级别
ENV_GET_SYSTEM_RESOURCES 表函数 - 返回系统信息	ACTIVITY METRICS BASE

os_release -“操作系统发行版”监视元素

在此主机上运行的操作系统的发行版。

表 1093. 表函数监视信息

表函数	监视元素收集级别
ENV_GET_SYSTEM_RESOURCES 表函数 - 返回系统信息	ACTIVITY METRICS BASE

os_version -“操作系统版本”监视元素

在此主机上运行的操作系统的版本。

表 1094. 表函数监视信息

表函数	监视元素收集级别
ENV_GET_SYSTEM_RESOURCES 表函数 - 返回系统信息	ACTIVITY METRICS BASE

outbound_appl_id -“出站应用程序标识”

此标识是在应用程序连接至 DRDA 主机数据库时生成的。它用来将 DB2 Connect 网关连接至主机，而 **appl_id** 监视元素用来将客户机连接至 DB2 Connect 网关。

元素标识

outbound_appl_id

元素类型

信息

表 1095. 快照监视信息

快照级别	逻辑数据分组	监视开关
DCS 应用程序	dcs_appl_info	基本

用法

可以将此元素与 **appl_id** 一起使用，以使应用程序信息的客户机部分与服务器部分相关联。

此标识在网络上是唯一的。

当网关集中器处于打开状态，或者 DCS 应用程序不在逻辑工作单元中时，此元素将是空白的。

格式 `Network.LU Name.Application instance`

示例 `CAIBMTOR.OSFDBM0.930131194520`

outbound_bytes_received -“接收的出站字节数”

DB2 Connect 网关从主机接收的字节数，通信协议使用情况（如 TCP/IP）除外。对于数据传输级别：处理使用此数目的数据传输的所有语句期间 DB2 Connect 网关从主机接收的字节数。

表 1096. 快照监视信息

快照级别	逻辑数据分组	监视开关
DCS 数据库	dcs_dbase	基本
DCS 应用程序	dcs_appl	基本
DCS 语句	dcs_stmt	语句
数据传输	stmt_transmissions	语句

对于语句级别的快照监视，不能重置此计数器。可在其他级别重置此计数器。

用法

使用此元素来度量从主机数据库至 DB2 Connect 网关的吞吐量。

outbound_bytes_received_bottom -“接收的最小出站字节数”

在此 DCS 数据库或在此 DCS 应用程序中处理使用此数目的数据传输的所有语句或链期间，DB2 Connect 网关对每个语句或链从主机接收的最少字节数。

表 1097. 快照监视信息

快照级别	逻辑数据分组	监视开关
数据传输	stmt_transmissions	语句

用法 将此元素与“接收的出站字节数”一起使用，将后者作为说明主机数据库至 DB2 Connect 网关的吞吐量的另一参数。

outbound_bytes_received_top -“接收的最大出站字节数”

在此 DCS 数据库或在此 DCS 应用程序中处理使用此数目的数据传输的所有语句或链期间，DB2 Connect 网关对每个语句或链从主机接收的最多字节数。

表 1098. 快照监视信息

快照级别	逻辑数据分组	监视开关
数据传输	stmt_transmissions	语句

用法 将此元素与“接收的出站字节数”一起使用，将后者作为说明主机数据库至 DB2 Connect 网关的吞吐量的另一参数。

outbound_bytes_sent -“发送的出站字节数”

DB2 Connect 网关发送到主机的字节数，通信协议使用情况（如 TCP/IP）除外。对于数据传输级别: 处理使用此数目的数据传输的所有语句期间 DB2 Connect 网关发送至主机的字节数。

表 1099. 快照监视信息

快照级别	逻辑数据分组	监视开关
DCS 数据库	dcs_dbase	基本
DCS 应用程序	dcs_appl	基本
DCS 语句	dcs_stmt	语句
数据传输	stmt_transmissions	语句

对于语句级别的快照监视，不能重置此计数器。可在其他级别重置此计数器。

用法 使用此元素来度量从 DB2 Connect 网关至主机数据库的吞吐量。

outbound_bytes_sent_bottom -“发送的最小出站字节数”

在此 DCS 数据库或在此 DCS 应用程序中处理使用此数目的数据传输的所有语句或链期间，DB2 Connect 网关对每个语句或链发送至主机的最少字节数。

表 1100. 快照监视信息

快照级别	逻辑数据分组	监视开关
数据传输	stmt_transmissions	语句

用法 将此元素与“发送的出站字节数”一起使用，将后者作为说明 DB2 Connect 网关至主机数据库的吞吐量的另一参数。

outbound_bytes_sent_top -“发送的最大出站字节数”

在此 DCS 数据库或在此 DCS 应用程序中处理使用此数目的数据传输的所有语句或链期间，DB2 Connect 网关对每个语句或链发送至主机的最多字节数。

表 1101. 快照监视信息

快照级别	逻辑数据分组	监视开关
数据传输	stmt_transmissions	语句

用法 将此元素与“发送的出站字节数”一起使用，将后者作为说明 DB2 Connect 网关至主机数据库的吞吐量的另一参数。

outbound_comm_address -“出站通信地址”

此项是目标数据库的通信地址。例如，它可能是 TCP/IP 的 IP 地址和端口号。

元素标识

outbound_comm_address

元素类型

信息

表 1102. 快照监视信息

快照级别	逻辑数据分组	监视开关
应用程序	appl_info	基本
DCS 应用程序	dcs_appl_info	基本

用法 此元素用于 DCS 应用程序的问题确定。

outbound_comm_protocol -“出站通信协议”

DB2 Connect 网关与主机之间使用的通信协议。

表 1103. 快照监视信息

快照级别	逻辑数据分组	监视开关
DCS 应用程序	dcs_appl_info	基本

用法

此元素用于 DCS 应用程序的问题确定。有效值为:

- SQLM_PROT_TCPIP

outbound_sequence_no -“出站序号”

当网关集中器处于打开状态，或者 DCS 应用程序不在逻辑工作单元中时，此元素将是空白的。

表 1104. 快照监视信息

快照级别	逻辑数据分组	监视开关
DCS 应用程序	dcs_appl_info	基本

overflow_accesses -“访问溢出记录次数”监视元素

对此表的溢出行的访问（读和写）次数。

表 1105. 表函数监视信息

表函数	监视元素收集级别
MON_GET_TABLE 表函数 - 获取表度量值	ACTIVITY METRICS BASE
MON_GET_TABLE_USAGE_LIST 表函数 - 从表用法列表返回信息	DATA OBJECT METRICS EXTENDED

表 1106. 快照监视信息

快照级别	逻辑数据分组	监视开关
表	表	基本

可将快照监视的计数器重置。

表 1107. 事件监视信息

事件类型	逻辑数据分组	监视开关
表	event_table	始终收集

用法

溢出行表明已发生数据分段。如果此数目较高，那么您可以使用 **REORG** 实用程序来重组该表（这将消除分段情况），从而提高表的性能。

如果某行已更新并且无法再装入到一开始写入的数据页中，那么该行将会溢出。这种情况通常是因为更新 VARCHAR 或 ALTER TABLE 语句造成的。

overflow_creates -“创建溢出行数”监视元素

对此表创建的溢出行的数目。

表 1108. 表函数监视信息

表函数	监视元素收集级别
MON_GET_TABLE 表函数 - 获取表度量值	ACTIVITY METRICS BASE
MON_GET_TABLE_USAGE_LIST 表函数 - 从表用法列表返回信息	DATA OBJECT METRICS EXTENDED

用法

package_id -“程序包标识”监视元素

程序包的唯一标识。

表 1109. 事件监视信息

事件类型	逻辑数据分组	监视开关
工作单元	在程序包列表中报告。	始终收集

用法

此元素的值与 SYSCAT.PACKAGES 视图的 PKGID 列中的值相匹配。

package_elapsed_time -“程序包耗用时间”监视元素

执行程序包中的各个部分所耗用的时间。此值以毫秒计。

表 1110. 事件监视信息

事件类型	逻辑数据分组	监视开关
工作单元	在程序包列表中报告。	始终收集

package_list_count -“程序包列表计数”监视元素

在特定工作单元的程序包列表中提供的条目数

表 1111. 事件监视信息

事件类型	逻辑数据分组	监视开关
工作单元	-	始终收集

package_list_exceeded -“超过了程序包列表的容量”监视元素

指示在工作单元中使用的程序包数目是否超过了程序包列表的容量。可能的值为 YES 和 NO。

表 1112. 事件监视信息

事件类型	逻辑数据分组	监视开关
工作单元	-	始终收集

package_list_size -“程序包列表大小”监视元素

程序包列表中包括的程序包标识计数。

表 1113. 事件监视信息

事件类型	逻辑数据分组	监视开关
工作单元	uow	

package_name -“程序包名”监视元素

包含 SQL 语句的程序包的名称。

表 1114. 表函数监视信息

表函数	监视元素收集级别
MON_GET_ACTIVITY_DETAILS 表函数 - 获取完整的活动详细信息（在 XML 文档 DETAILS 中报告）	ACTIVITY METRICS BASE

表 1114. 表函数监视信息 (续)

表函数	监视元素收集级别
MON_GET_PKG_CACHE_STMT 表函数 - 获取程序包高速缓存中的 SQL 语句活动度量值	ACTIVITY METRICS BASE
MON_GET_PKG_CACHE_STMT_DETAILS 表函数 - 获取程序包高速缓存条目的详细度量值	ACTIVITY METRICS BASE

表 1115. 快照监视信息

快照级别	逻辑数据分组	监视开关
应用程序	stmt	语句
DCS 语句	dcs_stmt	语句

表 1116. 事件监视信息

事件类型	逻辑数据分组	监视开关
锁定	-	-
带有详细信息的死锁 [1]	event_detailed_dlconn	-
语句	event_stmt	-
活动	event_activitystmt	-
程序包高速缓存	-	COLLECT BASE DATA

1 建议不要使用此事件监视器。建议不要再使用此选项，将来的发行版可能会将其除去。请使用 CREATE EVENT MONITOR FOR LOCKING 语句来监视与锁定相关的事件，例如锁定超时、锁定等待和死锁。

用法

可使用此元素来帮助标识正在执行的应用程序和 SQL 语句。

package_schema -“程序包模式”监视元素

与 SQL 语句相关联的程序包的模式名。

表 1117. 表函数监视信息

表函数	监视元素收集级别
MON_GET_ACTIVITY_DETAILS 表函数 - 获取完整的活动详细信息（在 XML 文档 DETAILS 中报告）	ACTIVITY METRICS BASE
MON_GET_PKG_CACHE_STMT 表函数 - 获取程序包高速缓存中的 SQL 语句活动度量值	ACTIVITY METRICS BASE
MON_GET_PKG_CACHE_STMT_DETAILS 表函数 - 获取程序包高速缓存条目的详细度量值	ACTIVITY METRICS BASE

表 1118. 事件监视信息

事件类型	逻辑数据分组	监视开关
锁定	-	-
程序包高速缓存	-	COLLECT BASE DATA

package_version_id -“程序包版本”监视元素

对于给定程序包名称和创建者，可以有（从 DB2 V8 开始）多个版本。程序包版本指定当前正在执行的 SQL 语句所在程序包的版本标识。程序包版本由嵌入式 SQL 程序在预编译（PREP）时使用 VERSION 关键字确定。如果在预编译时未指定，那么程序包版本的值为”（空字符串）。

表 1119. 表函数监视信息

表函数	监视元素收集级别
MON_GET_ACTIVITY_DETAILS 表函数 - 获取完整的活动详细信息（在 XML 文档 DETAILS 中报告）	ACTIVITY METRICS BASE
MON_GET_PKG_CACHE_STMT 表函数 - 获取程序包高速缓存中的 SQL 语句活动度量值	ACTIVITY METRICS BASE
MON_GET_PKG_CACHE_STMT_DETAILS 表函数 - 获取程序包高速缓存条目的详细度量值	ACTIVITY METRICS BASE

表 1120. 快照监视信息

快照级别	逻辑数据分组	监视开关
应用程序	stmt	语句

表 1121. 事件监视信息

事件类型	逻辑数据分组	监视开关
锁定	-	始终收集
语句	event_stmt	始终收集
活动	event_activitystmt	始终收集
程序包高速缓存	-	COLLECT BASE DATA

用法

使用此元素来帮助确定程序包以及当前正在执行的 SQL 语句。

packet_receive_errors -“包接收错误数”监视元素

网络适配器启动后接收包时发生的错误数。

表 1122. 表函数监视信息

表函数	监视元素收集级别
ENV_GET_NETWORK_RESOURCES 表函数 - 返回网络适配器信息	ACTIVITY METRICS BASE

packets_received -“所接收包数”监视元素

网络适配器启动后接收的包数。

表 1123. 表函数监视信息

表函数	监视元素收集级别
ENV_GET_NETWORK_RESOURCES 表函数 - 返回网络适配器信息	ACTIVITY METRICS BASE

packet_send_errors -“包发送错误数”监视元素

网络适配器启动后发送包时发生的错误数。

表 1124. 表函数监视信息

表函数	监视元素收集级别
ENV_GET_NETWORK_RESOURCES 表函数 - 返回网络适配器信息	ACTIVITY METRICS BASE

packets_sent -“所发送包数”监视元素

网络适配器启动后发送的包数。

表 1125. 表函数监视信息

表函数	监视元素收集级别
ENV_GET_NETWORK_RESOURCES 表函数 - 返回网络适配器信息	ACTIVITY METRICS BASE

page_allocations -“分配页数”监视元素

已分配给索引的页数。

表 1126. 表函数监视信息

表函数	监视元素收集级别
MON_GET_INDEX 表函数 - 获取索引度量值	ACTIVITY METRICS BASE

page_reorgs -“页重组”监视元素

对表执行的页重组数。

表 1127. 表函数监视信息

表函数	监视元素收集级别
MON_GET_TABLE 表函数 - 获取表度量值	DATA OBJECT METRICS EXTENDED

表 1128. 快照监视信息

快照级别	逻辑数据分组	监视开关
表	表	基本

可将快照监视的计数器重置。

表 1129. 事件监视信息

事件类型	逻辑数据分组	监视开关
表	event_table	始终收集

用法

尽管页具有足够空间，但可能会因为下列情况而导致页形成碎片:

- 插入新行时
- 更新现有行并且更新导致记录大小增加时

页形成碎片时可能需要重组。重组会将所有碎片空间移至连续区域，可在该区域中写入新记录。这种页重组可能需要几千个指令。它还会生成操作的日志记录。

如果页重组过多，那么可能导致插入性能无法达到最优。可使用 REORG TABLE 实用程序来重组表并消除碎片。还可对 ALTER TABLE 语句使用 APPEND 参数来指示将所有插入追加在表的结尾以避免进行页重组。

如果行更新导致行长度增加，那么页可能有足够的空间来容纳新行，但可能需要页重组来对该空间进行碎片整理。如果页没有足够的空间来容纳增大的新行，那么将创建导致在读取期间产生 *overflow_accesses* 的溢出记录。可通过使用固定长度列而不是可变长度列来避免出现这两种情况。

page_reclaims_x -“互斥存取页回收数”监视元素

DB2 pureScale实例中的另一成员回收的与此对象相关的页的次数（在计划释放此页之前），其中回收此页的成员需要互斥存取。

表 1130. 表函数监视信息

表函数	监视元素收集级别
MON_GET_PAGE_ACCESS_INFO 表函数 - 获取缓冲池页等待信息	ACTIVITY METRICS BASE

用法

与内部保留的对象空间映射相关的页回收数是单独计算的。

page_reclaims_s -“共享访问页回收数”监视元素

DB2 pureScale实例中的另一成员回收的与此对象相关的页的次数（在计划释放此页之前），其中回收此页的成员需要共享存取。

表 1131. 表函数监视信息

表函数	监视元素收集级别
MON_GET_PAGE_ACCESS_INFO 表函数 - 获取缓冲池页等待信息	ACTIVITY METRICS BASE

用法

与内部保留的对象空间映射相关的页回收数是单独计算的。

page_reclaims_initiated_x -“互斥存取导致的页回收数”监视元素

以互斥方式访问页从而导致从另一成员回收此页的次数。与内部保留的对象空间映射相关的页回收数是单独计算的。

表 1132. 表函数监视信息

表函数	监视元素收集级别
MON_GET_PAGE_ACCESS_INFO 表函数 - 获取缓冲池页等待信息	ACTIVITY METRICS BASE

page_reclaims_initiated_s -“共享存取导致的页回收数”监视元素

以共享方式访问页从而导致从另一成员回收此页的次数。与内部保留的对象空间映射相关的页回收数是单独计算的。

表 1133. 表函数监视信息

表函数	监视元素收集级别
MON_GET_PAGE_ACCESS_INFO 表函数 - 获取缓冲池页等待信息	ACTIVITY METRICS BASE

pages_from_block_ios -“块 I/O 读取总页数”监视元素

块 I/O 读取到缓冲池的块区域中的总页数。

表 1134. 表函数监视信息

表函数	监视元素收集命令和级别
MON_GET_BUFFERPOOL 表函数 - 获取缓冲池度量值	DATA OBJECT METRICS BASE
MON_GET_CONTAINER 表函数 - 获取表空间容器度量值	DATA OBJECT METRICS BASE
MON_GET_TABLESPACE 表函数 - 获取表空间度量值	DATA OBJECT METRICS BASE

表 1135. 快照监视信息

快照级别	逻辑数据分组	监视开关
缓冲池	bufferpool	缓冲池

用法

如果已启用基于块的缓冲池，那么此元素将报告块 I/O 读取的总页数。否则，此元素将返回 0。

要计算每个基于块的 I/O 按顺序预取的平均页数，请将 **pages_from_block_ios** 监视元素的值除以 **block_ios** 监视元素的值。如果此值比您在 CREATE BUFFERPOOL 或 ALTER BUFFERPOOL 语句中为基于块的缓冲池定义的 BLOCKSIZE 选项小很多，那么表明未充分利用基于块的 I/O。出现这种情况的一个可能原因是，正在按顺序预取的表空间的扩展数据块大小与基于块的缓冲池的块大小不匹配。

pages_from_vectored_ios -“向量 I/O 读取总页数”监视元素

向量 I/O 读取到缓冲池的页区域中的总页数。

表 1136. 表函数监视信息

表函数	监视元素收集命令和级别
MON_GET_BUFFERPOOL 表函数 - 获取缓冲池度量值	DATA OBJECT METRICS BASE
MON_GET_CONTAINER 表函数 - 获取表空间容器度量值	DATA OBJECT METRICS BASE
MON_GET_TABLESPACE 表函数 - 获取表空间度量值	DATA OBJECT METRICS BASE

表 1137. 快照监视信息

快照级别	逻辑数据分组	监视开关
缓冲池	bufferpool	缓冲池

pages_merged -“合并页数”监视元素

已合并的索引页数。

表 1138. 表函数监视信息

表函数	监视元素收集级别
MON_GET_INDEX 表函数 - 获取索引度量值	ACTIVITY METRICS BASE

pages_read -“读取页数”监视元素

从常规表空间和大型表空间的物理表空间容器读取的页数（包括数据页、索引页和 XML 页）。

表 1139. 表函数监视信息

表函数	监视元素收集级别
MON_GET_CONTAINER 表函数 - 获取表空间容器度量值	DATA OBJECT METRICS BASE

用法

pages_written -“写入页数”监视元素

以物理方式写入表空间容器的页数（包括数据页、索引页和 XML 页）。

表 1140. 表函数监视信息

表函数	监视元素收集级别
MON_GET_CONTAINER 表函数 - 获取表空间容器度量值	DATA OBJECT METRICS BASE

用法

parent_activity_id -“父活动标识”监视元素

此活动的父活动在父活动的工作单元中的唯一标识。如果没有父活动，那么此监视元素的值为 0。

表 1141. 表函数监视信息

表函数	监视元素收集命令和级别
MON_GET_ACTIVITY_DETAILS 表函数 - 获取完整的活动详细信息（在 XML 文档 DETAILS 中报告）	ACTIVITY METRICS BASE
WLM_GET_SERVICE_CLASS_AGENTS 表函数 - 列示正在服务类中运行的代理程序	ACTIVITY METRICS BASE
WLM_GET_WORKLOAD_OCCURRENCE_ACTIVITIES 表函数 - 返回活动列表	ACTIVITY METRICS BASE

表 1142. 事件监视信息

事件类型	逻辑数据分组	监视开关
活动	event_activity	始终收集

用法

将此元素与 **parent_uow_id** 元素和 **appl_id** 元素配合使用，以唯一地标识此活动记录中描述的活动的父活动。

parent_uow_id -“父工作单元标识”监视元素

应用程序句柄内的唯一工作单元标识。此活动的父活动所来源于的工作单元的标识。如果没有父活动，那么该值为 0。

表 1143. 表函数监视信息

表函数	监视元素收集命令和级别
MON_GET_ACTIVITY_DETAILS 表函数 - 获取完整的活动详细信息（在 XML 文档 DETAILS 中报告）	ACTIVITY METRICS BASE
WLM_GET_SERVICE_CLASS_AGENTS 表函数 - 列示正在服务类中运行的代理程序	ACTIVITY METRICS BASE

表 1143. 表函数监视信息 (续)

表函数	监视元素收集命令和级别
WLM_GET_WORKLOAD_OCCURRENCE_ACTIVITIES 表函数 - 返回活动列表	ACTIVITY METRICS BASE

表 1144. 事件监视信息

事件类型	逻辑数据分组	监视开关
活动	event_activity	始终收集

用法

将此元素与 **parent_activity_id** 元素和 **appl_id** 元素配合使用，以唯一地标识此活动记录中描述的活动的父活动。

partial_record -“部分记录”监视元素

指示事件监视器记录只是部分记录。

表 1145. 事件监视信息

事件类型	逻辑数据分组	监视开关
数据库	event_db	-
表	event_table	-
表空间	event_tablespace	-
缓冲池	event_bufferpool	-
连接	event_conn	-
语句	event_stmt	-
语句	event_subsection	-
事务	event_xact	-
活动	event_activity	-

用法

在数据库释放之前，大多数事件监视器不会输出结果。可使用 FLUSH EVENT MONITOR <monitorName> 语句来强制将监视器值输出至事件监视器输出写程序。这允许您在不需要停止并重新启动事件监视器的情况下强制它将记录输出至写程序。此元素指示事件监视器记录是不是清空操作的结果并且因此成为部分记录。

清空事件监视器不会导致值重置。这意味着触发事件监视器时仍会生成完整的事件监视器记录。

在 event_activity 逻辑数据分组时，**partial_record** 监视元素可能的值包括:

0 活动记录通常在活动结束时生成。

1 活动记录在调用 WLM_CAPTURE_ACTIVITY_IN_PROGRESS 存储过程后生成。

2 由于没有足够的存储器来创建记录，所以缺少此活动的信息。event_activity、event_activitystmt 或 event_activityvals 记录可能缺少信息。

participant_no -“死锁参与者”

唯一标识死锁参与者的序号。

元素标识

participant_no

元素类型

信息

表 1146. 事件监视信息

事件类型	逻辑数据分组	监视开关
死锁	event_dlconn	始终收集
带有详细信息的死锁	event_detailed_dlconn	始终收集

用法 在监视应用程序中使用此元素，以将死锁连接事件记录与死锁事件记录相关联。

participant_no_holding_lk -“对应用程序所需对象挂起锁定的参与者”

对某个对象挂起锁定的应用程序的参与者编号，此应用程序正在等待对该对象获取锁定。

元素标识

participant_no_holding_lk

元素类型

信息

表 1147. 事件监视信息

事件类型	逻辑数据分组	监视开关
死锁	event_dlconn	始终收集
带有详细信息的死锁	event_detailed_dlconn	始终收集

用法 此元素可帮助您确定存在资源争用的应用程序。

participant_type -“参与者类型”监视元素

可成为请求者或所有者的锁定参与者的类型。

表 1148. 事件监视信息

事件类型	逻辑数据分组	监视开关
锁定	lock_participants	

partition_key -“分区键”监视元素

事件监视器表的分区键。此值将被选中，以便每个事件监视器进程将数据插入到运行该进程的数据库分区中；即，插入操作将在运行事件监视器进程的数据库分区本地执行。

表 1149. 事件监视信息

事件类型	逻辑数据分组	监视开关
阈值违例	event_thresholdviolations	
阈值违例	control	
统计信息	event_qstats	
统计信息	event_scstats	
统计信息	event_histogrambin	
统计信息	event_wcstats	
统计信息	event_wcstats	
统计信息	control	
锁定	lock	
锁定	lock_participants	
锁定	lock_participant_activities	
锁定	lock_activity_values	
锁定	control	
程序包高速缓存	pkgcache_metrics	
程序包高速缓存	pkgcache_stmt_args	
程序包高速缓存	control	
工作单元	uow	
工作单元	uow_metrics	
工作单元	uow_package_list	
工作单元	uow_executable_list	
工作单元	control	
活动	event_activity	
活动	event_activitystmt	
活动	event_activityvals	
活动	event_activitymetrics	
活动	control	

partition_number -“分区号”

此元素仅供分区数据库环境或 DB2 pureScale 环境中的写至表事件监视器在目标 SQL 表中使用。此值指示插入了事件监视器数据的成员的编号。

表 1150. 表函数监视信息

表函数	监视元素收集级别
DB_MEMBERS 表函数	ACTIVITY METRICS BASE

表 1151. 事件监视信息

事件类型	逻辑数据分组	监视开关
活动	event_activitymetrics	ACTIVITY METRICS BASE
-	-	始终收集

passthru_time -“传递时间”

此元素包含此数据源响应 PASSTHRU 语句所花的总时间（以毫秒计），这些 PASSTHRU 语句来自联合服务器启动后或数据库监视计数器上一次重置后（取较晚者）在此联合服务器实例上运行的所有应用程序或单个应用程序。响应时间是以联合服务器将 PASSTHRU 语句提交给数据源的时间与数据源响应以指示已处理 PASSTHRU 语句的时间之差量度的。

表 1152. 快照监视信息

快照级别	逻辑数据分组	监视开关
数据库	dbase_remote	时间戳记
应用程序	appl_remote	时间戳记

可将快照监视的计数器重置。

用法

使用此元素来确定在此数据源上以通过方式处理语句所花的实际时间。

passthrus -“传递数”

此元素包含自联合服务器实例启动或数据库监视计数器最后一次重置以后（取较晚者），联合服务器代表任何应用程序直接传递到此数据源的 SQL 语句总数。

表 1153. 快照监视信息

快照级别	逻辑数据分组	监视开关
数据库	dbase_remote	基本
应用程序	appl_remote	基本

可将快照监视的计数器重置。

用法 使用此元素来确定可由联合服务器在本地处理的 SQL 语句的百分比以及需要以通过方式处理的 SQL 语句的百分比。如果此值很高，那么应确定原因和调查方法以便更好地使用本机支持。

past_activities_wrapped -“合并过去活动列表”监视元素

指示活动列表是否已合并。对任何一个应用程序所保留的先前活动数的缺省限制是 250。可使用 *DB2_MAX_INACT_STMTS* 注册表变量覆盖此缺省值。用户可能希望选择另一个限制值，以便增加或减少用于不活动语句信息的系统监视器堆空间量。

表 1154. 事件监视信息

事件类型	逻辑数据分组	监视开关
锁定	lock_participants	

phase_start_event_id -“阶段开始事件标识”

对应 UTILPHASESTART 的 EVENT_ID。

表 1155. 事件监视信息

事件类型	逻辑数据分组	监视开关
变更历史记录	UTILPHASE	始终收集

用法

对于变更历史记录事件监视器:

- 如果事件是停止实用程序阶段或处理阶段 (UTILPHASESTOP)，那么这是实用程序阶段的对应开始 (UTILPHASESTART) 的 EVENT_ID，否则为 -1。

与 PHASE_START_EVENT_TIMESTAMP 和成员元素配合使用以使阶段停止记录与对应开始记录相关联。

phase_start_event_timestamp -“阶段开始事件时间戳记”

对应 UTILPHASESTART 的时间

表 1156. 事件监视信息

事件类型	逻辑数据分组	监视开关
变更历史记录	UTILPHASE	始终收集

用法

对于变更历史记录事件监视器:

- 如果事件是停止实用程序阶段或处理阶段 (UTILPHASESTOP)，那么这是实用程序阶段的对应开始 (UTILPHASESTART) 的时间，否则为空。

与 PHASE_START_EVENT_ID 和成员元素配合使用以使阶段停止记录与对应开始记录相关联。

piped_sorts_accepted -“接受的管道排序数”

接受的管道排序数。

表 1157. 快照监视信息

快照级别	逻辑数据分组	监视开关
数据库管理器	db2	基本

可将快照监视的计数器重置。

用法 系统上每个活动排序分配内存可能导致排序占用过多可用系统内存。

如果接受的管道排序数低于请求的管道排序数，可通过调整下列配置参数中的一个或全部来改进排序性能:

- sortheap
- sheapthres

如果拒绝管道排序，那么可以考虑降低排序堆或提高排序堆阈值。您应该了解其中每个选项的可能含义。如果提高排序堆阈值，那么可能会保留更多内存以便分配给排序使用。这可能导致内存至磁盘的页面调度。如果降低排序堆，那么可能需要进行其他的合并阶段，这可能会导致排序速度下降。

piped_sorts_requested -“请求的管道排序数”

请求的管道排序数。

表 1158. 快照监视信息

快照级别	逻辑数据分组	监视开关
数据库管理器	db2	基本

可将快照监视的计数器重置。

用法 系统上每个活动排序分配内存可能导致排序占用过多可用系统内存。

排序列表堆（*sortheap*）和排序堆阈值（*sheapthres*）配置参数帮助控制用于排序操作的内存量。这些参数还将用于确定排序是否以管道方式进行。

因为管道排序可以降低磁盘 I/O，允许更多管道排序可以改进排序操作的性能并且可能改进整个系统的性能。如果对排序分配排序堆时将超过排序堆阈值，那么不接受管道排序。如果遇到拒绝管道排序的情况，请参阅 *piped_sorts_accepted* 以获取更多信息。

SQL EXPLAIN 输出将显示优化器是否请求管道排序。

pkg_cache_inserts -“程序包高速缓存插入数”监视元素

请求段不可用并且必须装入到程序包高速缓存中的总次数。此计数包括系统执行的所有隐式编译。

表 1159. 表函数监视信息

表函数	监视元素收集级别
MON_FORMAT_XML_METRICS_BY_ROW - 获取所有度量值的基于已格式化行的输出	不适用; 报告 XML 文档中作为格式化函数的输入提供的所有元素
MON_GET_CONNECTION 表函数 - 获取连接度量值	REQUEST METRICS BASE
MON_GET_CONNECTION_DETAILS 表函数 - 获取详细的连接度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE
MON_GET_SERVICE_SUBCLASS 表函数 - 获取服务子类度量值	REQUEST METRICS BASE
MON_GET_SERVICE_SUBCLASS_DETAILS 表函数 - 获取详细的服务子类度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE
MON_GET_UNIT_OF_WORK 表函数 - 获取工作单元度量值	REQUEST METRICS BASE
MON_GET_UNIT_OF_WORK 表函数 - 获取详细的工作单元度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE
MON_GET_WORKLOAD 表函数 - 获取工作负载度量值	REQUEST METRICS BASE
MON_GET_WORKLOAD_DETAILS 表函数 - 获取详细的工作负载度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE

表 1160. 快照监视信息

快照级别	逻辑数据分组	监视开关
数据库	dbase	基本
应用程序	appl	基本

可将快照监视的计数器重置。

表 1161. 事件监视信息

事件类型	逻辑数据分组	监视开关
数据库	event_db	始终收集
连接	event_conn	始终收集
统计信息	event_scstats（在 details_xml 文档中报告）	REQUEST METRICS BASE
统计信息	event_wlstats（在 details_xml 文档中报告）	REQUEST METRICS BASE
工作单元	在 system_metrics 文档中报告。	REQUEST METRICS BASE

用法

通过与 **pkg_cache_lookups** 监视元素一起使用，可以使用此监视元素并通过以下公式来计算程序包高速缓存命中率:

```
1 - (程序包高速缓存插入数/程序包高速缓存查询数)
```

pkg_cache_lookups -“程序包高速缓存查询数”监视元素

应用程序在程序包高速缓存中查找某个段或程序包的次数。在数据库级别，这指示自数据库启动或监视器数据重置后的整体引用数。此计数器包括该段已装入到高速缓存中的情况和该段必须装入到高速缓存中的情况。在代理程序要与不同应用程序相关联的集中器环境中，如果新的代理程序在本地存储器中没有必需的段或程序包可用，那么可能需要附加程序包高速缓存查询。

表 1162. 表函数监视信息

表函数	监视元素收集级别
MON_FORMAT_XML_METRICS_BY_ROW - 获取所有度量值的基于已格式化行的输出	不适用; 报告 XML 文档中作为格式化函数的输入提供的所有元素
MON_GET_CONNECTION 表函数 - 获取连接度量值	REQUEST METRICS BASE
MON_GET_CONNECTION_DETAILS 表函数 - 获取详细的连接度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE
MON_GET_SERVICE_SUBCLASS 表函数 - 获取服务子类度量值	REQUEST METRICS BASE
MON_GET_SERVICE_SUBCLASS_DETAILS 表函数 - 获取详细的服务子类度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE
MON_GET_UNIT_OF_WORK 表函数 - 获取工作单元度量值	REQUEST METRICS BASE
MON_GET_UNIT_OF_WORK 表函数 - 获取详细的工作单元度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE
MON_GET_WORKLOAD 表函数 - 获取工作负载度量值	REQUEST METRICS BASE
MON_GET_WORKLOAD_DETAILS 表函数 - 获取详细的工作负载度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE

表 1163. 快照监视信息

快照级别	逻辑数据分组	监视开关
数据库	dbase	基本
应用程序	appl	基本

可将快照监视的计数器重置。

表 1164. 事件监视信息

事件类型	逻辑数据分组	监视开关
数据库	event_db	始终收集
连接	event_conn	始终收集
统计信息	event_scstats（在 details_xml 文档中报告）	REQUEST METRICS BASE
统计信息	event_wlstats（在 details_xml 文档中报告）	REQUEST METRICS BASE
工作单元	在 system_metrics 文档中报告。	REQUEST METRICS BASE

用法

要计算程序包高速缓存命中率，请使用以下公式:

```
1 - (程序包高速缓存插入数/程序包高速缓存查询数)
```

程序包高速缓存命中率显示是否在高效使用程序包高速缓存。如果命中率很高（超过 0.8），那么表示高速缓存确实起到作用。如果命中率较低，可能表示应该增大程序包高速缓存。

您需要使用程序包高速缓存大小进行试验，以找出 **pckcachesz** 配置参数的最优数字。例如，如果降低高速缓存的大小时 **pkg_cache_inserts** 元素中的值没有增加，那么可以使用更小的程序包高速缓存大小。降低程序包高速缓存大小将释放系统资源以供其他任务使用。如果增加程序包高速缓存大小使得 **pkg_cache_inserts** 数降低，那么可以通过这样做来改进整体系统性能。在满工作负载条件下能够最好地完成此实验。

可将此元素与 **ddl_sql_stmts** 配合使用以确定执行 DDL 语句是否会影响程序包高速缓存的性能。执行 DDL 语句时，动态 SQL 语句的各段可能会变得无效。下次使用时系统会对无效段进行隐式编译。执行 DDL 语句可能会使许多段变得无效，并且准备这些段时要求的额外处理时间会严重影响性能。在此情况下，程序包高速缓存命中率会影响无效段的隐式重新编译。但它不会影响将新段插入到高速缓存中，所以增加程序包高速缓存大小不会改进整体性能。您可能会发现在满负载环境中工作之前独自为应用程序调整高速缓存会更加清晰明了。

在决定采取什么操作之前，需要确定 DDL 语句在程序包高速缓存命中率的值中所起的作用。如果 DDL 语句很少出现，那么可通过增加高速缓存大小来改进性能。如果 DDL 语句频繁出现，那么可能需要通过限制 DDL 语句的使用（将 DDL 语句的可能使用时间限制为特定时间段）来改进性能。

static_sql_stmts 和 **dynamic_sql_stmts** 计数可用来帮助提供有关要高速缓存的段的数量和类型的信息。

注: 您可能想要在数据库级别使用此信息来计算每个应用程序的平均程序包高速缓存命中率。您应该在应用程序级别查看此信息以找出给定应用程序的精确程序包高速缓存命中率。为了满足很少执行的应用程序的高速缓存要求而增加程序包高速缓存大小是不值得的。

pkg_cache_num_overflows -“程序包高速缓存溢出数”

程序包高速缓存溢出其分配内存边界的次数。

表 1165. 快照监视信息

快照级别	逻辑数据分组	监视开关
数据库	dbase	基本

可将快照监视的计数器重置。

表 1166. 事件监视信息

事件类型	逻辑数据分组	监视开关
数据库	event_db	始终收集

用法

将此元素与 **pkg_cache_size_top** 监视元素配合使用来确定是否需要增加程序包高速缓存的大小以避免溢出。

pkg_cache_size_top -“程序包高速缓存高水位标记”

程序包高速缓存达到的最大大小。

注: 从 DB2 V9.5 开始，不推荐使用 **pkg_cache_size_top** 监视元素。使用此监视元素不会生成错误。但是不会返回有效值。建议不要再使用此监视元素，将来的发行版中可能会将其除去。

表 1167. 快照监视信息

快照级别	逻辑数据分组	监视开关
数据库	dbase	基本

表 1168. 事件监视信息

事件类型	逻辑数据分组	监视开关
数据库	event_db	始终收集

用法

如果程序包高速缓存溢出，那么表示此元素包含溢出期间程序包高速缓存达到的最大大小。

检查 **pkg_cache_num_overflows** 监视元素以确定是否发生此类情况。

可通过以下公式来确定工作负载需要的程序包高速缓存最小大小:

```
最大程序包高速缓存大小 / 4096
```

通过将结果四舍五入为整数，指示为避免溢出程序包高速缓存最少需要的页数（每页 4K 字节）。

pool_async_data_gbp_indep_pages_found_in_lbp -“本地缓冲池中由异步 EDU 发现的独立于组缓冲池的数据页数”监视元素

本地缓冲池中由异步 EDU 发现的独立于组缓冲池 (GBP) 的数据页数。

表 1169. 表函数监视信息

表函数	监视元素收集级别
MON_GET_BUFFERPOOL 表函数 - 获取缓冲池度量值	DATA OBJECT METRICS BASE
MON_GET_TABLESPACE 表函数 - 获取表空间度量值	DATA OBJECT METRICS BASE

表 1170. 事件监视信息

事件类型	逻辑数据分组	监视开关
活动	event_activitymetrics	ACTIVITY METRICS BASE
程序包高速缓存	pkgcache_metrics	ACTIVITY METRICS BASE
工作单元	uow_metrics	REQUEST METRICS BASE

pool_async_data_gbp_invalid_pages -“异步组缓冲池无效数据页数”监视元素

预取程序尝试从组缓冲池读取数据页（因为该页在本地缓冲池中无效）的次数。

表 1171. 表函数监视信息

表函数	监视元素收集级别
MON_GET_BUFFERPOOL 表函数 - 获取缓冲池度量值	DATA OBJECT METRICS BASE
MON_GET_TABLESPACE 表函数 - 获取表空间度量值	DATA OBJECT METRICS BASE

用法

可按如下所示计算预取程序（或异步）数据页命中率:

```
GBP = ( pool_async_data_gbp_l_reads - pool_async_data_gbp_p_reads )
 / pool_async_data_gbp_l_reads
```

缓冲池命中速率是 IBM DB2 pureScale Feature 的整体性能中的重要因素。使用此公式可帮助您确定组缓冲池会否成为数据库吞吐量的限制因素。

pool_async_data_gbp_l_reads -“异步组缓冲池数据逻辑读取数”监视元素

预取程序尝试从组缓冲池读取依赖于组缓冲池 (GBP) 的数据页（因为该页在本地缓冲池中无效或不存在）的次数。

表 1172. 表函数监视信息

表函数	监视元素收集级别
MON_GET_BUFFERPOOL 表函数 - 获取缓冲池度量值	DATA OBJECT METRICS BASE
MON_GET_TABLESPACE 表函数 - 获取表空间度量值	DATA OBJECT METRICS BASE

用法

可按如下所示计算预取程序（或异步）数据页命中率:

```
GBP = ( pool_async_data_gbp_l_reads - pool_async_data_gbp_p_reads )
 / pool_async_data_gbp_l_reads
```

缓冲池命中速率是 IBM DB2 pureScale Feature 的整体性能中的重要因素。使用此公式可帮助您确定组缓冲池会否成为数据库吞吐量的限制因素。

pool_async_data_gbp_p_reads -“异步组缓冲池数据物理读取数”监视元素

预取程序尝试将磁盘中依赖于组缓冲池 (GBP) 的数据页读取到本地缓冲池中（因为在 GBP 中找不到该页）的次数。

表 1173. 表函数监视信息

表函数	监视元素收集级别
MON_GET_BUFFERPOOL 表函数 - 获取缓冲池度量值	DATA OBJECT METRICS BASE
MON_GET_TABLESPACE 表函数 - 获取表空间度量值	DATA OBJECT METRICS BASE

用法

可按如下所示计算预取程序（或异步）数据页命中率:

```
GBP = ( pool_async_data_gbp_l_reads - pool_async_data_gbp_p_reads )
 / pool_async_data_gbp_l_reads
```

缓冲池命中速率是 IBM DB2 pureScale Feature 的整体性能中的重要因素。使用此公式可帮助您确定组缓冲池会否成为数据库吞吐量的限制因素。

pool_async_data_lbp_pages_found -“发现的异步本地缓冲池数据页数”监视元素

预取程序尝试访问数据页时此数据页出现在本地缓冲池中的次数。

表 1174. 表函数监视信息

表函数	监视元素收集级别
MON_GET_BUFFERPOOL 表函数 - 获取缓冲池度量值	DATA OBJECT METRICS BASE
MON_GET_TABLESPACE 表函数 - 获取表空间度量值	DATA OBJECT METRICS BASE

用法

可按如下所示计算预取程序（或异步）数据页命中率:

```
GBP = ( pool_async_data_gbp_l_reads - pool_async_data_gbp_p_reads )
 / pool_async_data_gbp_l_reads
```

缓冲池命中速率是 IBM DB2 pureScale Feature 的整体性能中的重要因素。使用此公式可帮助您确定组缓冲池会否成为数据库吞吐量的限制因素。

pool_async_data_read_reqs -“缓冲池异步读请求数”监视元素

预取程序对操作系统发出的异步读请求的数目。这些请求通常是涉及多页的大型块 I/O。

表 1175. 表函数监视信息

表函数	监视元素收集命令和级别
MON_GET_BUFFERPOOL 表函数 - 获取缓冲池度量值	DATA OBJECT METRICS BASE
MON_GET_TABLESPACE 表函数 - 获取表空间度量值	DATA OBJECT METRICS BASE

表 1176. 快照监视信息

快照级别	逻辑数据分组	监视开关
数据库	dbase	缓冲池
表空间	tablespace	缓冲池
缓冲池	bufferpool	缓冲池

可将快照监视的计数器重置。

表 1177. 事件监视信息

事件类型	逻辑数据分组	监视开关
数据库	event_db	始终收集
表空间	event_tablespace	始终收集

用法

要计算每个读请求的平均数据页数，请使用下列公式:

```
pool_async_data_reads / pool_async_data_read_reqs
```

此平均值可以帮助您确定预取程序所使用的平均读 I/O 大小。此数据还可以帮助您了解所测量的工作负载的大型块 I/O 要求。

预取程序读 I/O 最大大小是所涉及表空间的 CREATE TABLESPACE 语句中 EXTENTSIZE 选项指定的值，但在某些情况下可能更小:

- 扩展数据块的某些页已经在缓冲池中
- 超出操作系统能力
- EXTENTSIZE 选项值非常大，执行大型 I/O 将导致整体性能下降

pool_async_data_reads -“缓冲池异步数据读次数”监视元素

指示异步引擎可派遣单元 (EDU) 从所有类型的表空间的物理表空间容器中读取的数据页数。

表 1178. 表函数监视信息

表函数	监视元素收集级别
MON_GET_BUFFERPOOL 表函数 - 获取缓冲池度量值	DATA OBJECT METRICS BASE
MON_GET_TABLESPACE 表函数 - 获取表空间度量值	DATA OBJECT METRICS BASE

表 1179. 快照监视信息

快照级别	逻辑数据分组	监视开关
数据库	dbase	缓冲池
表空间	tablespace	缓冲池
缓冲池	bufferpool	缓冲池

可将快照监视的计数器重置。

表 1180. 事件监视信息

事件类型	逻辑数据分组	监视开关
数据库	event_db	始终收集
表空间	event_tablespace	始终收集

用法

可将此元素与 **pool_data_p_reads** 配合使用，以计算同步执行的物理读取数（即数据库管理器代理程序执行的物理数据页读取数）。请使用以下公式:

```
1-((pool_data_p_reads+pool_index_p_reads)-(pool_async_data_reads+pool_async_index_reads))
/(pool_data_l_reads+pool_index_l_reads)
```

通过比较异步读取数与同步读取数的比率，可了解预取程序的执行情况。在调整 **num_ioservers** 配置参数时，此元素会非常有用。

异步读取是由数据库管理器预取程序执行的。

pool_async_data_writes -“缓冲池异步数据写次数”监视元素

异步页清除程序或预取程序将缓冲池数据页物理写至磁盘的次数。预取程序可将脏页写至磁盘以便为要预取的页腾出空间。

表 1181. 表函数监视信息

表函数	监视元素收集级别
MON_GET_BUFFERPOOL 表函数 - 获取缓冲池度量值	DATA OBJECT METRICS BASE
MON_GET_TABLESPACE 表函数 - 获取表空间度量值	DATA OBJECT METRICS BASE

表 1182. 快照监视信息

快照级别	逻辑数据分组	监视开关
数据库	dbase	缓冲池
表空间	tablespace	缓冲池
缓冲池	bufferpool	缓冲池

可将快照监视的计数器重置。

表 1183. 事件监视信息

事件类型	逻辑数据分组	监视开关
数据库	event_db	始终收集
表空间	event_tablespace	始终收集

用法 通过将此元素与 **pool_data_writes** 监视元素配合使用，可以计算以同步方式执行的物理写请求（即，由数据库管理器代理程序执行的物理数据页写操作）的数目。请使用以下公式:

```
pool_data_writes - pool_async_data_writes
```

通过比较异步写入数与同步写入数的比率，可了解缓冲池页清除程序的执行情况。在调整 **num_iocleaners** 配置参数时，此比率会非常有用。

pool_async_index_gbp_indep_pages_found_in_lbp -“本地缓冲池中由异步 EDU 发现的独立于组缓冲池的索引页数”监视元素

本地缓冲池中由异步 EDU 发现的独立于组缓冲池 (GBP) 的索引页数。

表 1184. 表函数监视信息

表函数	监视元素收集级别
MON_GET_BUFFERPOOL 表函数 - 获取缓冲池度量值	DATA OBJECT METRICS BASE

表 1184. 表函数监视信息 (续)

表函数	监视元素收集级别
MON_GET_TABLESPACE 表函数 - 获取表空间度量值	DATA OBJECT METRICS BASE

表 1185. 事件监视信息

事件类型	逻辑数据分组	监视开关
活动	event_activitymetrics	ACTIVITY METRICS BASE
程序包高速缓存	pkgcache_metrics	ACTIVITY METRICS BASE
工作单元	uow_metrics	REQUEST METRICS BASE

pool_async_index_gbp_invalid_pages -“异步组缓冲池无效索引页数”监视元素

预取程序尝试从组缓冲池读取索引页（因为该页在本地缓冲池中无效）的次数。

表 1186. 表函数监视信息

表函数	监视元素收集级别
MON_GET_BUFFERPOOL 表函数 - 获取缓冲池度量值	DATA OBJECT METRICS BASE
MON_GET_TABLESPACE 表函数 - 获取表空间度量值	DATA OBJECT METRICS BASE

用法

可按如下所示计算预取程序（或异步）索引页命中率:

```
GBP = ( pool_async_index_gbp_l_reads - pool_async_index_gbp_p_reads )
 / pool_async_index_gbp_l_reads
```

缓冲池命中速率是 DB2 pureScale实例的整体性能中的重要因素。使用以上公式可帮助您确定组缓冲池会否成为数据库吞吐量的限制因素。

pool_async_index_gbp_l_reads -“异步组缓冲池索引逻辑读取数”监视元素

预取程序尝试从组缓冲池读取依赖于组缓冲池 (GBP) 的索引页（因为该页在本地缓冲池中无效或不存在）的次数。

表 1187. 表函数监视信息

表函数	监视元素收集级别
MON_GET_BUFFERPOOL 表函数 - 获取缓冲池度量值	DATA OBJECT METRICS BASE
MON_GET_TABLESPACE 表函数 - 获取表空间度量值	DATA OBJECT METRICS BASE

用法

可按如下所示计算预取程序（或异步）索引页命中率:

```
GBP = ( pool_async_index_gbp_l_reads - pool_async_index_gbp_p_reads )
 / pool_async_index_gbp_l_reads
```

缓冲池命中速率是 DB2 pureScale实例的整体性能中的重要因素。使用以上公式可帮助您确定组缓冲池会否成为数据库吞吐量的限制因素。

pool_async_index_gbp_p_reads -“异步组缓冲池索引物理读取数”监视元素

预取程序尝试将磁盘中依赖于组缓冲池 (GBP) 的索引页读取到本地缓冲池中（因为在 GBP 中找不到该页）的次数。

表 1188. 表函数监视信息

表函数	监视元素收集级别
MON_GET_BUFFERPOOL 表函数 - 获取缓冲池度量值	DATA OBJECT METRICS BASE
MON_GET_TABLESPACE 表函数 - 获取表空间度量值	DATA OBJECT METRICS BASE

用法

可按如下所示计算预取程序（或异步）索引页命中率:

```
GBP = ( pool_async_index_gbp_l_reads - pool_async_index_gbp_p_reads )
 / pool_async_index_gbp_l_reads
```

缓冲池命中速率是 DB2 pureScale实例的整体性能中的重要因素。使用以上公式可帮助您确定组缓冲池会否成为数据库吞吐量的限制因素。

pool_async_index_lbp_pages_found -“发现的异步本地缓冲池索引页数”监视元素

预取程序尝试访问索引页时此索引页出现在本地缓冲池中的次数。

表 1189. 表函数监视信息

表函数	监视元素收集级别
MON_GET_BUFFERPOOL 表函数 - 获取缓冲池度量值	DATA OBJECT METRICS BASE
MON_GET_TABLESPACE 表函数 - 获取表空间度量值	DATA OBJECT METRICS BASE

用法

可按如下所示计算预取程序（或异步）索引页命中率:

```
GBP = ( pool_async_index_gbp_l_reads - pool_async_index_gbp_p_reads )
 / pool_async_index_gbp_l_reads
```

缓冲池命中速率是 DB2 pureScale实例的整体性能中的重要因素。使用以上公式可帮助您确定组缓冲池会否成为数据库吞吐量的限制因素。

pool_async_index_read_reqs -“缓冲池异步索引读请求数”监视元素

针对索引页的异步读请求的数目。

表 1190. 表函数监视信息

表函数	监视元素收集命令和级别
MON_GET_BUFFERPOOL 表函数 - 获取缓冲池度量值	DATA OBJECT METRICS BASE
MON_GET_TABLESPACE 表函数 - 获取表空间度量值	DATA OBJECT METRICS BASE

表 1191. 快照监视信息

快照级别	逻辑数据分组	监视开关
数据库	dbase	缓冲池
表空间	tablespace	缓冲池
缓冲池	bufferpool	缓冲池

可将快照监视的计数器重置。

表 1192. 事件监视信息

事件类型	逻辑数据分组	监视开关
数据库	event_db	始终收集
表空间	event_tablespace	始终收集

用法 要计算每个异步请求读取的索引页数，请使用以下公式:

```
pool_async_index_reads / pool_async_index_read_reqs
```

此平均数可帮助您确定每次与预取程序交互时对索引页进行的异步 I/O 量。

pool_async_index_reads -“缓冲池异步索引读取数”监视元素

指示异步引擎可派遣单元 (EDU) 从所有类型的表空间的物理表空间容器中读取的索引页数。

表 1193. 表函数监视信息

表函数	监视元素收集级别
MON_GET_BUFFERPOOL 表函数 - 获取缓冲池度量值	DATA OBJECT METRICS BASE
MON_GET_TABLESPACE 表函数 - 获取表空间度量值	DATA OBJECT METRICS BASE

表 1194. 快照监视信息

快照级别	逻辑数据分组	监视开关
数据库	dbase	缓冲池
表空间	tablespace	缓冲池
缓冲池	bufferpool	缓冲池

可将快照监视的计数器重置。

表 1195. 事件监视信息

事件类型	逻辑数据分组	监视开关
数据库	event_db	始终收集
表空间	event_tablespace	始终收集

用法

通过将此元素与 **pool_index_p_reads** 监视元素配合使用，可以计算以同步方式执行的物理读操作（即，由数据库管理器代理程序执行的物理索引页读操作）的数目。请使用以下公式:

```
1 - ((pool_data_p_reads + pool_index_p_reads) - (pool_async_data_reads + pool_async_index_reads))
 / (pool_data_l_reads + pool_index_l_reads)
```

通过比较异步读取数与同步读取数的比率，可了解预取程序的执行情况。在调整 **num_ioservers** 配置参数时，此元素会非常有用。

异步读取是由数据库管理器预取程序执行的。

pool_async_index_writes -“缓冲池异步索引写次数”监视元素

异步页清除程序或预取程序将缓冲池索引页物理写至磁盘的次数。预取程序可将脏页写至磁盘以便为要预取的页腾出空间。

表 1196. 表函数监视信息

表函数	监视元素收集命令和级别
MON_GET_BUFFERPOOL 表函数 - 获取缓冲池度量值	DATA OBJECT METRICS BASE
MON_GET_TABLESPACE 表函数 - 获取表空间度量值	DATA OBJECT METRICS BASE

表 1197. 快照监视信息

快照级别	逻辑数据分组	监视开关
数据库	dbase	缓冲池
表空间	tablespace	缓冲池
缓冲池	bufferpool	缓冲池

可将快照监视的计数器重置。

表 1198. 事件监视信息

事件类型	逻辑数据分组	监视开关
数据库	event_db	始终收集
表空间	event_tablespace	始终收集

用法

通过将此元素与 **pool_index_writes** 监视元素配合使用，可以计算以同步方式执行的物理索引写请求（即，由数据库管理器代理程序执行的物理索引页写操作）的数目。请使用以下公式:

```
pool_index_writes - pool_async_index_writes
```

通过比较异步写入数与同步写入数的比率，可了解缓冲池页清除程序的执行情况。在调整 **num_iocleaners** 配置参数时，此比率会非常有用。

pool_async_read_time -“缓冲池异步读取时间”

指示从表空间容器（物理）中由异步引擎可派遣单元 (EDU) 对所有类型的表空间读取数据和索引页所花费的总时间。此值以毫秒计。

元素标识

pool_async_read_time

元素类型

计数器

表 1199. 表函数监视信息

表函数	监视元素收集级别
MON_GET_BUFFERPOOL 表函数 - 获取缓冲池度量值	DATA OBJECT METRICS BASE
MON_GET_TABLESPACE 表函数 - 获取表空间度量值	DATA OBJECT METRICS BASE

表 1200. 快照监视信息

快照级别	逻辑数据分组	监视开关
数据库	dbase	缓冲池
表空间	tablespace	缓冲池
缓冲池	bufferpool	缓冲池

可将快照监视的计数器重置。

表 1201. 事件监视信息

事件类型	逻辑数据分组	监视开关
数据库	event_db	始终收集
表空间	event_tablespace	始终收集

用法 可使用此元素并借助以下公式来计算同步读取所耗用的时间:

```
pool_read_time - pool_async_read_time
```

还可使用此元素并借助以下公式来计算平均异步读取时间:

```
pool_async_read_time / pool_async_data_reads
```

这些计算可用来了解要执行的 I/O 处理。

pool_async_write_time -“缓冲池异步写入时间”监视元素

数据库管理器页清除程序将数据或索引页从缓冲池写至磁盘的耗用时间总计。 此值以毫秒计。

表 1202. 表函数监视信息

表函数	监视元素收集级别
MON_GET_BUFFERPOOL 表函数 - 获取缓冲池度量值	DATA OBJECT METRICS BASE
MON_GET_TABLESPACE 表函数 - 获取表空间度量值	DATA OBJECT METRICS BASE

表 1203. 快照监视信息

快照级别	逻辑数据分组	监视开关
数据库	dbase	缓冲池
表空间	tablespace	缓冲池
缓冲池	bufferpool	缓冲池

可将快照监视的计数器重置。

表 1204. 事件监视信息

事件类型	逻辑数据分组	监视开关
数据库	event_db	始终收集
表空间	event_tablespace	始终收集

用法

要计算同步写入页所耗用的时间，请使用以下公式:

```
pool_write_time - pool_async_write_time
```

还可使用此元素并借助以下公式来计算平均异步写入时间:

```
pool_async_write_time
   / (pool_async_data_writes
       + pool_async_index_writes)
```

这些计算可用来了解要执行的 I/O 处理。

chayzchpool_async_xda_gbp_indep_pages_found_in_lbp -“本地缓冲池中由异步 EDU 发现的独立于组缓冲池的 XML 存储器对象 (XDA) 页数”监视元素

本地缓冲池中由异步 EDU 发现的独立于组缓冲池 (GBP) 的 XML 存储器对象 (XDA) 页数。

表 1205. 表函数监视信息

表函数	监视元素收集级别
MON_GET_BUFFERPOOL 表函数 - 获取缓冲池度量值	DATA OBJECT METRICS BASE
MON_GET_TABLESPACE 表函数 - 获取表空间度量值	DATA OBJECT METRICS BASE

表 1206. 事件监视信息

事件类型	逻辑数据分组	监视开关
活动	event_activitymetrics	ACTIVITY METRICS BASE
程序包高速缓存	pkgcache_metrics	ACTIVITY METRICS BASE
工作单元	uow_metrics	REQUEST METRICS BASE

pool_async_xda_gbp_invalid_pages -“异步组缓冲池无效 XDA 数据页数”监视元素

预取程序向组缓冲池发出针对 XML 存储器对象 (XDA) 的数据页的请求（因为该页在本地缓冲池中标记为无效）的次数。

表 1207. 表函数监视信息

表函数	监视元素收集级别
MON_GET_BUFFERPOOL 表函数 - 获取缓冲池度量值	DATA OBJECT METRICS BASE
MON_GET_TABLESPACE 表函数 - 获取表空间度量值	DATA OBJECT METRICS BASE

用法

可按如下所示计算预取程序（或异步）XDA 页命中率:

```
GBP = ( pool_async_xda_gbp_l_reads - pool_async_xda_gbp_p_reads )
 / pool_async_xda_gbp_l_reads
```

缓冲池命中速率是 DB2 pureScale实例的整体性能中的重要因素。使用以上公式可帮助您确定组缓冲池会否成为数据库吞吐量的限制因素。

pool_async_xda_gbp_l_reads -“组缓冲池 XDA 数据异步逻辑读取请求数”监视元素

预取程序尝试从组缓冲池读取 XML 存储器对象 (XDA) 的依赖于 GBP 的数据页（因为该页在本地缓冲池中无效或不存在）的次数。

表 1208. 表函数监视信息

表函数	监视元素收集级别
MON_GET_BUFFERPOOL 表函数 - 获取缓冲池度量值	DATA OBJECT METRICS BASE
MON_GET_TABLESPACE 表函数 - 获取表空间度量值	DATA OBJECT METRICS BASE

用法

可按如下所示计算预取程序（或异步）XDA 页命中率:

```
GBP = ( pool_async_xda_gbp_l_reads - pool_async_xda_gbp_p_reads )
 / pool_async_xda_gbp_l_reads
```

缓冲池命中速率是 DB2 pureScale实例的整体性能中的重要因素。使用以上公式可帮助您确定组缓冲池会否成为数据库吞吐量的限制因素。

pool_async_xda_gbp_p_reads -“组缓冲池 XDA 数据异步物理读取请求数”监视元素

预取程序将磁盘中 XML 存储器对象 (XDA) 的依赖于 GBP 的数据页读取到本地缓冲池中（因为在 GBP 中找不到该页）的次数。

表 1209. 表函数监视信息

表函数	监视元素收集级别
MON_GET_BUFFERPOOL 表函数 - 获取缓冲池度量值	DATA OBJECT METRICS BASE
MON_GET_TABLESPACE 表函数 - 获取表空间度量值	DATA OBJECT METRICS BASE

用法

可按如下所示计算预取程序（或异步）XDA 页命中率:

```
GBP = ( pool_async_xda_gbp_l_reads - pool_async_xda_gbp_p_reads )
 / pool_async_xda_gbp_l_reads
```

缓冲池命中速率是 DB2 pureScale实例的整体性能中的重要因素。使用以上公式可帮助您确定组缓冲池会否成为数据库吞吐量的限制因素。

pool_async_xda_lbp_pages_found -“发现的异步本地缓冲池 XDA 数据页数”监视元素

预取程序向本地缓冲池请求并在其中发现 XML 存储器对象 (XDA) 的数据页的次数。

表 1210. 表函数监视信息

表函数	监视元素收集级别
MON_GET_BUFFERPOOL 表函数 - 获取缓冲池度量值	DATA OBJECT METRICS BASE
MON_GET_TABLESPACE 表函数 - 获取表空间度量值	DATA OBJECT METRICS BASE

用法

可按如下所示计算预取程序（或异步）XDA 页命中率:

```
GBP = ( pool_async_xda_gbp_l_reads - pool_async_xda_gbp_p_reads )
 / pool_async_xda_gbp_l_reads
```

缓冲池命中速率是 DB2 pureScale实例的整体性能中的重要因素。使用以上公式可帮助您确定组缓冲池会否成为数据库吞吐量的限制因素。

pool_async_xda_read_reqs -“缓冲池异步 XDA 读请求数”监视元素

针对 XML 存储器对象（XDA）数据的异步读取请求数。

表 1211. 表函数监视信息

表函数	监视元素收集命令和级别
MON_GET_BUFFERPOOL 表函数 - 获取缓冲池度量值	DATA OBJECT METRICS BASE
MON_GET_TABLESPACE 表函数 - 获取表空间度量值	DATA OBJECT METRICS BASE

表 1212. 快照监视信息

快照级别	逻辑数据分组	监视开关
数据库	dbase	缓冲池
表空间	tablespace	缓冲池
缓冲池	bufferpool	缓冲池

可将快照监视的计数器重置。

表 1213. 事件监视信息

事件类型	逻辑数据分组	监视开关
数据库	event_db	始终收集
表空间	event_tablespace	始终收集

用法 要计算每个异步请求读取的平均 XML 存储器对象数据页数，请使用以下公式:

```
pool_async_xda_reads / pool_async_xda_read_reqs
```

此平均数可帮助您确定每次与预取程序交互时完成的异步 I/O 量。

pool_async_xda_reads -“缓冲池异步 XDA 数据读取数”监视元素

指示异步引擎可派遣单元 (EDU) 从所有类型的表空间的物理表空间容器中读取的 XML 存储器对象（XDA）数据页数。

表 1214. 表函数监视信息

表函数	监视元素收集命令和级别
MON_GET_BUFFERPOOL 表函数 - 获取缓冲池度量值	DATA OBJECT METRICS BASE
MON_GET_TABLESPACE 表函数 - 获取表空间度量值	DATA OBJECT METRICS BASE

表 1215. 快照监视信息

快照级别	逻辑数据分组	监视开关
数据库	dbase	缓冲池
表空间	tablespace	缓冲池
缓冲池	bufferpool	缓冲池

可将快照监视的计数器重置。

表 1216. 事件监视信息

事件类型	逻辑数据分组	监视开关
数据库	event_db	始终收集
表空间	event_tablespace	始终收集

用法

通过使用 **pool_async_xda_reads** 和 **pool_xda_p_reads** 监视元素，可以计算以同步方式对 XML 存储器对象数据页执行的物理读操作（即，数据库管理器代理程序对 XML 数据执行的物理数据页读操作）的数目。请使用以下公式:

```
pool_xda_p_reads - pool_async_xda_reads
```

通过比较异步读取数与同步读取数的比率，可了解预取程序的执行情况。在调整 **num_ioservers** 配置参数时，此元素会非常有用。

异步读取是由数据库管理器预取程序执行的。

pool_async_xda_writes -“缓冲池异步 XDA 数据写次数”监视元素

异步页清除程序或预取程序将 XML 存储器对象（XDA）的缓冲池数据页物理写至磁盘的次数。预取程序可将脏页写至磁盘以便为要预取的页腾出空间。

表 1217. 表函数监视信息

表函数	监视元素收集命令和级别
MON_GET_BUFFERPOOL 表函数 - 获取缓冲池度量值	DATA OBJECT METRICS BASE
MON_GET_TABLESPACE 表函数 - 获取表空间度量值	DATA OBJECT METRICS BASE

表 1218. 快照监视信息

快照级别	逻辑数据分组	监视开关
数据库	dbase	缓冲池
表空间	tablespace	缓冲池
缓冲池	bufferpool	缓冲池

可将快照监视的计数器重置。

表 1219. 事件监视信息

事件类型	逻辑数据分组	监视开关
数据库	event_db	始终收集
表空间	event_tablespace	始终收集

用法 通过将此元素与 **pool_xda_writes** 监视元素配合使用，可以计算以同步方式对 XML 存储器对象数据页执行的物理写请求（即，数据库管理器代理程序对 XML 数据执行的物理数据页写操作）的数目。请使用以下公式:

```
pool_xda_writes - pool_async_xda_writes
```

通过比较异步写入数与同步写入数的比率，可了解缓冲池页清除程序的执行情况。在调整 **num_iocleaners** 配置参数时，此比率会非常有用。

pool_config_size -“内存池的已配置大小”

DB2 数据库系统中的内存池的内部配置大小。此值以字节计。

表 1220. 快照监视信息

快照级别	逻辑数据分组	监视开关
数据库管理器	memory_pool	基本
数据库	memory_pool	基本
应用程序	memory_pool	基本

表 1221. 事件监视信息

事件类型	逻辑数据分组	监视开关
数据库	event_dbmemuse	始终收集

表 1221. 事件监视信息 (续)

事件类型	逻辑数据分组	监视开关
连接	event_connmemuse	始终收集

用法 要跟踪系统内存使用情况，请将此值与 **pool_cur_size**、**pool_id** 和 **pool_watermark** 一起使用。

要查看内存池是否接近已满状态，将 **pool_config_size** 与 **pool_cur_size** 进行比较。例如，假定实用程序堆非常小。可以按一定时间间隔获取快照并查看快照输出的实用程序堆部分，以诊断这一特定问题。如果需要，可能会允许 **pool_cur_size** 超过 **pool_config_size** 以避免内存不足故障。如果这种情况不常发生，那么可能不需要进一步的操作。但是，如果 **pool_cur_size** 一直接近或大于 **pool_config_size**，那么可能要考虑增加实用程序堆的大小。

pool_cur_size -“内存池的当前大小”

内存池的当前大小。此值以字节计。

表 1222. 快照监视信息

快照级别	逻辑数据分组	监视开关
数据库管理器	memory_pool	基本
数据库	memory_pool	基本
应用程序	memory_pool	基本

表 1223. 事件监视信息

事件类型	逻辑数据分组	监视开关
数据库	event_dbmemuse	始终收集
连接	event_connmemuse	始终收集

用法 要跟踪系统内存使用情况，请将此值与 *pool_config_size*、*pool_id* 和 *pool_watermark* 一起使用。

要查看内存池是否接近已满状态，将 *pool_config_size* 与 *pool_cur_size* 进行比较。例如，假定实用程序堆非常小。可以按一定时间间隔获取快照并查看快照输出的实用程序堆部分，以诊断这一特定问题。如果 *pool_cur_size* 的值一直接近 *pool_config_size*，那么您可能要考虑增加实用程序堆的大小。

pool_data_gbp_indep_pages_found _in_lbp -“本地缓冲池中发现的独立于组缓冲池的数据页数”监视元素

代理程序在本地缓冲池 (LBP) 中发现的独立于组缓冲池 (GBP) 的数据页数。

表 1224. 表函数监视信息

表函数	监视元素收集级别
MON_FORMAT_XML_METRICS_BY_ROW - 获取所有度量值的基于已格式化行的输出	不适用；报告 XML 文档中作为格式化函数的输入提供的所有元素

表 1224. 表函数监视信息 (续)

表函数	监视元素收集级别
MON_GET_ACTIVITY_DETAILS 表函数 - 获取完整的活动详细信息（在 XML 文档 DETAILS 中报告）	ACTIVITY METRICS BASE
MON_GET_BUFFERPOOL 表函数 - 获取缓冲池度量值	DATA OBJECT METRICS BASE
MON_GET_CONNECTION 表函数 - 获取连接度量值	REQUEST METRICS BASE
MON_GET_CONNECTION_DETAILS 表函数 - 获取详细连接度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE
MON_GET_PKG_CACHE_STMT 表函数 - 获取程序包高速缓存中的 SQL 语句活动度量值	ACTIVITY METRICS BASE
MON_GET_PKG_CACHE_STMT_DETAILS - 获取程序包高速缓存条目的详细度量值（在 XML 文档 DETAILS 中报告）	ACTIVITY METRICS BASE
MON_GET_SERVICE_SUBCLASS 表函数 - 获取服务子类度量值	REQUEST METRICS BASE
MON_GET_SERVICE_SUBCLASS_DETAILS 表函数 - 获取详细的服务子类度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE
MON_GET_TABLESPACE 表函数 - 获取表空间度量值	DATA OBJECT METRICS BASE
MON_GET_UNIT_OF_WORK 表函数 - 获取工作单元度量值	REQUEST METRICS BASE
MON_GET_UNIT_OF_WORK 表函数 - 获取详细的工作单元度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE
MON_GET_WORKLOAD 表函数 - 获取工作负载度量值	REQUEST METRICS BASE
MON_GET_WORKLOAD_DETAILS 表函数 - 获取详细的工作负载度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE

表 1225. 事件监视信息

事件类型	逻辑数据分组	监视开关
活动	event_activitymetrics	ACTIVITY METRICS BASE
程序包高速缓存	pkgcache_metrics	ACTIVITY METRICS BASE
工作单元	uow_metrics	REQUEST METRICS BASE

pool_data_gbp_invalid_pages -“组缓冲池无效数据页数”监视元素

数据页在本地缓冲池中无效并因此改为从组缓冲池中读取的次数。在 DB2 pureScale 环境外部，此值为空。

表 1226. 表函数监视信息

表函数	监视元素收集级别
MON_FORMAT_XML_METRICS_BY_ROW - 获取所有度量值的基于已格式化行的输出	不适用；报告 XML 文档中作为格式化函数的输入提供的所有元素
MON_GET_ACTIVITY_DETAILS 表函数 - 获取完整的活动详细信息（在 XML 文档 DETAILS 中报告）	ACTIVITY METRICS BASE
MON_GET_BUFFERPOOL 表函数 - 获取缓冲池度量值	DATA OBJECT METRICS BASE
MON_GET_CONNECTION 表函数 - 获取连接度量值	REQUEST METRICS BASE
MON_GET_CONNECTION_DETAILS 表函数 - 获取详细的连接度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE
MON_GET_PKG_CACHE_STMT 表函数 - 获取程序包高速缓存中的 SQL 语句活动度量值	ACTIVITY METRICS BASE
MON_GET_SERVICE_SUBCLASS 表函数 - 获取服务子类度量值	REQUEST METRICS BASE
MON_GET_SERVICE_SUBCLASS_DETAILS 表函数 - 获取详细的服务子类度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE
MON_GET_TABLESPACE 表函数 - 获取表空间度量值	DATA OBJECT METRICS BASE
MON_GET_UNIT_OF_WORK 表函数 - 获取工作单元度量值	REQUEST METRICS BASE
MON_GET_UNIT_OF_WORK 表函数 - 获取详细的工作单元度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE
MON_GET_WORKLOAD 表函数 - 获取工作负载度量值	REQUEST METRICS BASE
MON_GET_WORKLOAD_DETAILS 表函数 - 获取详细的工作负载度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE

表 1227. 事件监视信息

事件类型	逻辑数据分组	监视开关
活动	event_activity（在 details_xml 文档中报告）	ACTIVITY METRICS BASE
活动	event_activitymetrics	ACTIVITY METRICS BASE
统计信息	event_scstats（在 details_xml 文档中报告）	REQUEST METRICS BASE

表 1227. 事件监视信息 (续)

事件类型	逻辑数据分组	监视开关
统计信息	event_wlstats（在 details_xml 文档中报告）	REQUEST METRICS BASE
工作单元	-	始终收集
程序包高速缓存	-	始终收集
锁定	-	始终收集

用法

要确定在本地缓冲池中发现所请求页的频率，请使用以下公式:

```
(POOL_DATA_LBP_PAGES_FOUND - POOL_ASYNC_DATA_LBP_PAGES_FOUND) / POOL_DATA_L_READS
```

要确定在组缓冲池中发现所请求页的频率，请使用以下公式

```
(POOL_DATA_GBP_L_READS - POOL_DATA_GBP_P_READS) / POOL_DATA_GBP_L_READS
```

本地缓冲池和组缓冲池命中率都是集群高速缓存设施的整体性能中的重要因素。使用这些公式可帮助您确定本地缓冲池或组缓冲池会否成为数据库吞吐量的限制因素。

pool_data_gbp_l_reads -“组缓冲池数据逻辑读取数”监视元素

尝试从组缓冲池读取依赖于组缓冲池 (GBP) 的数据页（因为该页在本地缓冲池 (LBP) 中无效或不存在）的次数。

表 1228. 表函数监视信息

表函数	监视元素收集级别
MON_FORMAT_XML_METRICS_BY_ROW - 获取所有度量值的基于已格式化行的输出	不适用; 报告 XML 文档中作为格式化函数的输入提供的所有元素
MON_GET_ACTIVITY_DETAILS 表函数 - 获取完整的活动详细信息（在 XML 文档 DETAILS 中报告）	ACTIVITY METRICS BASE
MON_GET_BUFFERPOOL 表函数 - 获取缓冲池度量值	DATA OBJECT METRICS BASE
MON_GET_CONNECTION 表函数 - 获取连接度量值	REQUEST METRICS BASE
MON_GET_CONNECTION_DETAILS 表函数 - 获取详细的连接度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE
MON_GET_PKG_CACHE_STMT 表函数 - 获取程序包高速缓存中的 SQL 语句活动度量值	ACTIVITY METRICS BASE
MON_GET_SERVICE_SUBCLASS 表函数 - 获取服务子类度量值	REQUEST METRICS BASE
MON_GET_SERVICE_SUBCLASS_DETAILS 表函数 - 获取详细的服务子类度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE
MON_GET_TABLESPACE 表函数 - 获取表空间度量值	DATA OBJECT METRICS BASE

表 1228. 表函数监视信息 (续)

表函数	监视元素收集级别
MON_GET_UNIT_OF_WORK 表函数 - 获取工作单元度量值	REQUEST METRICS BASE
MON_GET_UNIT_OF_WORK 表函数 - 获取详细的工作单元度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE
MON_GET_WORKLOAD 表函数 - 获取工作负载度量值	REQUEST METRICS BASE
MON_GET_WORKLOAD_DETAILS 表函数 - 获取详细的工作负载度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE

表 1229. 事件监视信息

事件类型	逻辑数据分组	监视开关
活动	event_activity（在 details_xml 文档中报告）	ACTIVITY METRICS BASE
活动	event_activitymetrics	ACTIVITY METRICS BASE
统计信息	event_scstats（在 details_xml 文档中报告）	REQUEST METRICS BASE
统计信息	event_wlstats（在 details_xml 文档中报告）	REQUEST METRICS BASE
工作单元	-	始终收集
程序包高速缓存	-	始终收集
锁定	-	始终收集

用法

要确定在本地缓冲池中发现所请求页的频率，请使用以下公式:

```
(POOL_DATA_LBP_PAGES_FOUND - POOL_ASYNC_DATA_LBP_PAGES_FOUND) / POOL_DATA_L_READS
```

要确定在组缓冲池中发现所请求页的频率，请使用以下公式

```
(POOL_DATA_GBP_L_READS - POOL_DATA_GBP_P_READS) / POOL_DATA_GBP_L_READS
```

本地缓冲池和组缓冲池命中率都是集群高速缓存设施的整体性能中的重要因素。使用这些公式可帮助您确定本地缓冲池或组缓冲池会否成为数据库吞吐量的限制因素。

pool_data_gbp_p_reads -“组缓冲池数据物理读取数”监视元素

尝试将磁盘中依赖于组缓冲池 (GBP) 的数据页读取到本地缓冲池中（因为在 GBP 中找不到该页）的次数。

表 1230. 表函数监视信息

表函数	监视元素收集级别
MON_FORMAT_XML_METRICS_BY_ROW - 获取所有度量值的基于已格式化行的输出	不适用; 报告 XML 文档中作为格式化函数的输入提供的所有元素

表 1230. 表函数监视信息 (续)

表函数	监视元素收集级别
MON_GET_ACTIVITY_DETAILS 表函数 - 获取完整的活动详细信息（在 XML 文档 DETAILS 中报告）	ACTIVITY METRICS BASE
MON_GET_BUFFERPOOL 表函数 - 获取缓冲池度量值	DATA OBJECT METRICS BASE
MON_GET_CONNECTION 表函数 - 获取连接度量值	REQUEST METRICS BASE
MON_GET_CONNECTION_DETAILS 表函数 - 获取详细的连接度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE
MON_GET_PKG_CACHE_STMT 表函数 - 获取程序包高速缓存中的 SQL 语句活动度量值	ACTIVITY METRICS BASE
MON_GET_SERVICE_SUBCLASS 表函数 - 获取服务子类度量值	REQUEST METRICS BASE
MON_GET_SERVICE_SUBCLASS_DETAILS 表函数 - 获取详细的服务子类度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE
MON_GET_TABLESPACE 表函数 - 获取表空间度量值	DATA OBJECT METRICS BASE
MON_GET_UNIT_OF_WORK 表函数 - 获取工作单元度量值	REQUEST METRICS BASE
MON_GET_UNIT_OF_WORK 表函数 - 获取详细的工作单元度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE
MON_GET_WORKLOAD 表函数 - 获取工作负载度量值	REQUEST METRICS BASE
MON_GET_WORKLOAD_DETAILS 表函数 - 获取详细的工作负载度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE

表 1231. 事件监视信息

事件类型	逻辑数据分组	监视开关
活动	event_activity（在 details_xml 文档中报告）	ACTIVITY METRICS BASE
活动	event_activitymetrics	ACTIVITY METRICS BASE
统计信息	event_scstats（在 details_xml 文档中报告）	REQUEST METRICS BASE
统计信息	event_wlstats（在 details_xml 文档中报告）	REQUEST METRICS BASE
工作单元	-	始终收集
程序包高速缓存	-	始终收集
锁定	-	始终收集

用法

要确定在本地缓冲池中发现所请求页的频率，请使用以下公式:

```
(POOL_DATA_LBP_PAGES_FOUND - POOL_ASYNC_DATA_LBP_PAGES_FOUND) / POOL_DATA_L_READS
```

要确定在组缓冲池中发现所请求页的频率，请使用以下公式

```
(POOL_DATA_GBP_L_READS - POOL_DATA_GBP_P_READS) / POOL_DATA_GBP_L_READS
```

本地缓冲池和组缓冲池命中率都是集群高速缓存设施的整体性能中的重要因素。使用这些公式可帮助您确定本地缓冲池或组缓冲池会否成为数据库吞吐量的限制因素。

pool_data_lbp_pages_found -“本地缓冲池发现的数据页数”监视元素

数据页出现在本地缓冲池中的次数。

表 1232. 表函数监视信息

表函数	监视元素收集级别
MON_FORMAT_XML_METRICS_BY_ROW - 获取所有度量值的基于已格式化行的输出	不适用; 报告 XML 文档中作为格式化函数的输入提供的所有元素
MON_GET_ACTIVITY_DETAILS 表函数 - 获取完整的活动详细信息（在 XML 文档 DETAILS 中报告）	ACTIVITY METRICS BASE
MON_GET_BUFFERPOOL 表函数 - 获取缓冲池度量值	DATA OBJECT METRICS BASE
MON_GET_CONNECTION 表函数 - 获取连接度量值	REQUEST METRICS BASE
MON_GET_CONNECTION_DETAILS 表函数 - 获取详细的连接度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE
MON_GET_PKG_CACHE_STMT 表函数 - 获取程序包高速缓存中的 SQL 语句活动度量值	ACTIVITY METRICS BASE
MON_GET_SERVICE_SUBCLASS 表函数 - 获取服务子类度量值	REQUEST METRICS BASE
MON_GET_SERVICE_SUBCLASS_DETAILS 表函数 - 获取详细的服务子类度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE
MON_GET_TABLESPACE 表函数 - 获取表空间度量值	DATA OBJECT METRICS BASE
MON_GET_UNIT_OF_WORK 表函数 - 获取工作单元度量值	REQUEST METRICS BASE
MON_GET_UNIT_OF_WORK 表函数 - 获取详细的工作单元度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE
MON_GET_WORKLOAD 表函数 - 获取工作负载度量值	REQUEST METRICS BASE
MON_GET_WORKLOAD_DETAILS 表函数 - 获取详细的工作负载度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE

表 1233. 事件监视信息

事件类型	逻辑数据分组	监视开关
活动	event_activity（在 details_xml 文档中报告）	ACTIVITY METRICS BASE
活动	event_activitymetrics	ACTIVITY METRICS BASE
统计信息	event_scstats（在 details_xml 文档中报告）	REQUEST METRICS BASE
统计信息	event_wlstats（在 details_xml 文档中报告）	REQUEST METRICS BASE
工作单元	-	始终收集
程序包高速缓存	-	始终收集
锁定	-	始终收集

用法

要确定在本地缓冲池中发现所请求页的频率，请使用以下公式:

```
(POOL_DATA_LBP_PAGES_FOUND - POOL_ASYNC_DATA_LBP_PAGES_FOUND) / POOL_DATA_L_READS
```

要确定在组缓冲池中发现所请求页的频率，请使用以下公式

```
(POOL_DATA_GBP_L_READS - POOL_DATA_GBP_P_READS) / POOL_DATA_GBP_L_READS
```

本地缓冲池和组缓冲池命中率都是集群高速缓存设施的整体性能中的重要因素。使用这些公式可帮助您确定本地缓冲池或组缓冲池会否成为数据库吞吐量的限制因素。

pool_data_l_reads -“缓冲池数据逻辑读取数”监视元素

从常规表空间和大型表空间的逻辑缓冲池中请求获取的数据页的数目。

表 1234. 表函数监视信息

表函数	监视元素收集级别
MON_FORMAT_XML_METRICS_BY_ROW - 获取所有度量值的基于已格式化行的输出	不适用; 报告 XML 文档中作为格式化函数的输入提供的所有元素
MON_GET_ACTIVITY_DETAILS 表函数 - 获取完整的活动详细信息（在 XML 文档 DETAILS 中报告）	ACTIVITY METRICS BASE
MON_GET_BUFFERPOOL 表函数 - 获取缓冲池度量值	DATA OBJECT METRICS BASE
MON_GET_CONNECTION 表函数 - 获取连接度量值	REQUEST METRICS BASE
MON_GET_CONNECTION_DETAILS 表函数 - 获取详细的连接度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE
MON_GET_PKG_CACHE_STMT 表函数 - 获取程序包高速缓存中的 SQL 语句活动度量值	ACTIVITY METRICS BASE
MON_GET_PKG_CACHE_STMT_DETAILS 表函数 - 获取程序包高速缓存条目的详细度量值	ACTIVITY METRICS BASE

表 1234. 表函数监视信息 (续)

表函数	监视元素收集级别
MON_GET_SERVICE_SUBCLASS 表函数 - 获取服务子类度量值	REQUEST METRICS BASE
MON_GET_SERVICE_SUBCLASS_DETAILS 表函数 - 获取详细的服务子类度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE
MON_GET_TABLESPACE 表函数 - 获取表空间度量值	DATA OBJECT METRICS BASE
MON_GET_UNIT_OF_WORK 表函数 - 获取工作单元度量值	REQUEST METRICS BASE
MON_GET_UNIT_OF_WORK 表函数 - 获取详细的工作单元度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE
MON_GET_WORKLOAD 表函数 - 获取工作负载度量值	REQUEST METRICS BASE
MON_GET_WORKLOAD_DETAILS 表函数 - 获取详细的工作负载度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE

表 1235. 快照监视信息

快照级别	逻辑数据分组	监视开关
数据库	dbase	缓冲池
表空间	tablespace	缓冲池
缓冲池	bufferpool	缓冲池
应用程序	appl	缓冲池
应用程序	stmt	缓冲池
动态 SQL	dynsql	缓冲池，语句

可将快照监视的计数器重置。

表 1236. 事件监视信息

事件类型	逻辑数据分组	监视开关
活动	event_activity（在 details_xml 文档中报告）	ACTIVITY METRICS BASE
活动	event_activitymetrics	ACTIVITY METRICS BASE
统计信息	event_scstats（在 details_xml 文档中报告）	REQUEST METRICS BASE
统计信息	event_wlstats（在 details_xml 文档中报告）	REQUEST METRICS BASE
工作单元	在 system_metrics 文档中报告。	REQUEST METRICS BASE
数据库	event_db	始终收集
表空间	event_tablespace	始终收集
连接	event_conn	始终收集
语句	event_stmt	始终收集

表 1236. 事件监视信息 (续)

事件类型	逻辑数据分组	监视开关
活动	event_activity	缓冲池，语句
程序包高速缓存	在 activity_metrics 文档中报告。	ACTIVITY METRICS BASE

用法

此计数包括数据处于下列情况时对数据的访问:

- 当数据库管理器需要处理页时，数据已经在缓冲池中。
- 应读取到缓冲池中，数据库管理器才能处理页。

可使用 **pool_data_l_reads** 和 **pool_data_p_reads** 监视元素来计算数据页命中率。例如，以下公式返回 DB2 环境中的数据页命中率（在没有 IBM DB2 pureScale Feature 的情况下）:

```
(pool_data_lbp_pages_found
- pool_async_data_lbp_pages_found - pool_temp_data_l_reads) / pool_data_l_reads)
× 100
```

有关更多信息，请参阅第 1379 页的『用于计算缓冲池命中率的公式』 。

增加缓冲池大小一般会改进命中率，但您会达到一个最优状态，而无法继续改进。从理论上说，如果能够分配大到足以存储整个数据库的缓冲池，那么系统启动并运行后你可以得到 100% 的命中率。但在许多情况下这是不现实的。命中率的高低实际上取决于数据的大小以及访问数据的方式。如果数据库很大并且数据访问比较平均，那么命中率将会很低。对于非常大的表，您几乎无能为力。

对于较小并且访问较为频繁的表和索引，要提高其命中率，请将其分配到不同的缓冲池。

pool_data_p_reads -“缓冲池数据物理读取数”监视元素

指示从常规表空间和大型表空间的物理表空间容器中读取的数据页数。

表 1237. 表函数监视信息

表函数	监视元素收集级别
MON_FORMAT_XML_METRICS_BY_ROW - 获取所有度量值的基于已格式化行的输出	不适用; 报告 XML 文档中作为格式化函数的输入提供的所有元素
MON_GET_ACTIVITY_DETAILS 表函数 - 获取完整的活动详细信息（在 XML 文档 DETAILS 中报告）	ACTIVITY METRICS BASE
MON_GET_BUFFERPOOL 表函数 - 获取缓冲池度量值	DATA OBJECT METRICS BASE
MON_GET_CONNECTION 表函数 - 获取连接度量值	REQUEST METRICS BASE
MON_GET_CONNECTION_DETAILS 表函数 - 获取详细的连接度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE

表 1237. 表函数监视信息 (续)

表函数	监视元素收集级别
MON_GET_PKG_CACHE_STMT 表函数 - 获取程序包高速缓存中的 SQL 语句活动度量值	ACTIVITY METRICS BASE
MON_GET_PKG_CACHE_STMT_DETAILS 表函数 - 获取程序包高速缓存条目的详细度量值	ACTIVITY METRICS BASE
MON_GET_SERVICE_SUBCLASS 表函数 - 获取服务子类度量值	REQUEST METRICS BASE
MON_GET_SERVICE_SUBCLASS_DETAILS 表函数 - 获取详细的服务子类度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE
MON_GET_TABLESPACE 表函数 - 获取表空间度量值	DATA OBJECT METRICS BASE
MON_GET_UNIT_OF_WORK 表函数 - 获取工作单元度量值	REQUEST METRICS BASE
MON_GET_UNIT_OF_WORK 表函数 - 获取详细的工作单元度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE
MON_GET_WORKLOAD 表函数 - 获取工作负载度量值	REQUEST METRICS BASE
MON_GET_WORKLOAD_DETAILS 表函数 - 获取详细的工作负载度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE

表 1238. 快照监视信息

快照级别	逻辑数据分组	监视开关
数据库	dbase	缓冲池
表空间	tablespace	缓冲池
缓冲池	bufferpool	缓冲池
应用程序	appl	缓冲池
应用程序	stmt	缓冲池
动态 SQL	dynsql	缓冲池，语句

可将快照监视的计数器重置。

表 1239. 事件监视信息

事件类型	逻辑数据分组	监视开关
活动	event_activity（在 details_xml 文档中报告）	ACTIVITY METRICS BASE
活动	event_activitymetrics	ACTIVITY METRICS BASE
统计信息	event_scstats（在 details_xml 文档中报告）	REQUEST METRICS BASE
统计信息	event_wlstats（在 details_xml 文档中报告）	REQUEST METRICS BASE
工作单元	在 system_metrics 文档中报告。	REQUEST METRICS BASE

表 1239. 事件监视信息 (续)

事件类型	逻辑数据分组	监视开关
数据库	event_db	始终收集
表空间	event_tablespace	始终收集
连接	event_conn	始终收集
语句	event_stmt	始终收集
活动	event_activity	缓冲池，语句
程序包高速缓存	在 activity_metrics 文档中报告。	ACTIVITY METRICS BASE

用法

通过将此元素与 **pool_data_l_reads** 和 **pool_async_data_reads** 监视元素配合使用，可以计算以同步方式执行的物理读操作（即，数据库管理器代理程序执行的物理数据页读操作）的数目。请使用以下公式:

```
1 - ((pool_data_p_reads + pool_index_p_reads) - (pool_async_data_reads + pool_async_index_reads))
 / (pool_data_l_reads + pool_index_l_reads)
```

通过比较异步读取数与同步读取数的比率，可了解预取程序的执行情况。在调整 **num_ioservers** 配置参数时，此信息非常有用。

pool_data_writes -“缓冲池数据写次数”监视元素

以物理方式将缓冲池数据页写入磁盘的次数。

表 1240. 表函数监视信息

表函数	监视元素收集级别
MON_FORMAT_XML_METRICS_BY_ROW - 获取所有度量值的基于已格式化行的输出	不适用; 报告 XML 文档中作为格式化函数的输入提供的所有元素
MON_GET_ACTIVITY_DETAILS 表函数 - 获取完整的活动详细信息（在 XML 文档 DETAILS 中报告）	ACTIVITY METRICS BASE
MON_GET_BUFFERPOOL 表函数 - 获取缓冲池度量值	DATA OBJECT METRICS BASE
MON_GET_CONNECTION 表函数 - 获取连接度量值	REQUEST METRICS BASE
MON_GET_CONNECTION_DETAILS 表函数 - 获取详细的连接度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE
MON_GET_PKG_CACHE_STMT 表函数 - 获取程序包高速缓存中的 SQL 语句活动度量值	ACTIVITY METRICS BASE
MON_GET_PKG_CACHE_STMT_DETAILS 表函数 - 获取程序包高速缓存条目的详细度量值	ACTIVITY METRICS BASE
MON_GET_SERVICE_SUBCLASS 表函数 - 获取服务子类度量值	REQUEST METRICS BASE

表 1240. 表函数监视信息 (续)

表函数	监视元素收集级别
MON_GET_SERVICE_SUBCLASS_DETAILS 表函数 - 获取详细的服务子类度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE
MON_GET_TABLESPACE 表函数 - 获取表空间度量值	DATA OBJECT METRICS BASE
MON_GET_UNIT_OF_WORK 表函数 - 获取工作单元度量值	REQUEST METRICS BASE
MON_GET_UNIT_OF_WORK 表函数 - 获取详细的工作单元度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE
MON_GET_WORKLOAD 表函数 - 获取工作负载度量值	REQUEST METRICS BASE
MON_GET_WORKLOAD_DETAILS 表函数 - 获取详细的工作负载度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE

表 1241. 快照监视信息

快照级别	逻辑数据分组	监视开关
数据库	dbase	缓冲池
表空间	tablespace	缓冲池
缓冲池	bufferpool	缓冲池
应用程序	appl	缓冲池

可将快照监视的计数器重置。

表 1242. 事件监视信息

事件类型	逻辑数据分组	监视开关
活动	event_activity（在 details_xml 文档中报告）	ACTIVITY METRICS BASE
活动	event_activitymetrics	ACTIVITY METRICS BASE
统计信息	event_scstats（在 details_xml 文档中报告）	REQUEST METRICS BASE
统计信息	event_wlstats（在 details_xml 文档中报告）	REQUEST METRICS BASE
工作单元	在 system_metrics 文档中报告。	REQUEST METRICS BASE
数据库	event_db	始终收集
表空间	event_tablespace	始终收集
连接	event_conn	始终收集
程序包高速缓存	在 activity_metrics 文档中报告。	ACTIVITY METRICS BASE

用法

如果缓冲池数据页被写入磁盘的次数在 **pool_data_p_reads** 监视元素值中占较高的百分比，那么可通过增加可供数据库使用的缓冲池页数来提高性能。

缓冲池数据页将因为下列原因写至磁盘:

- 释放缓冲池中的某一页以便可以读取另一页
- 将缓冲池清仓

系统不会总是通过写入页为新页腾出空间。如果该页未被更新，那么只需将其替换。此元素不考虑此替换。

在需要缓冲池空间之前，数据页可由异步页清除程序代理程序写入，这由 **pool_async_data_writes** 监视元素报告。这些异步页写入与同步页写入一起包括在此元素的值中。

计算此百分比时，忽略一开始填充缓冲池所需的物理读取的数目。要确定写入的页数:

1. 运行应用程序以装入缓冲区。
2. 记录此元素的值。
3. 再次运行应用程序。
4. 从此元素的新值中减去步骤 2 中记录的值。

为了避免在应用程序的各次运行之间取消分配缓冲池，应该执行下列其中一项操作:

- 使用 **ACTIVATE DATABASE** 命令来激活数据库。
- 将一个空闲的应用程序连接至数据库。

如果所有应用程序都要更新该数据库，那么提高缓冲池大小对性能可能没有多大影响，原因是大多数缓冲池页包含已更新数据，而这些数据必须写入磁盘。但是，如果已更新页在写出之前可供其他工作单元使用，那么缓冲池可保存读写操作，这将对性能有所改进。

pool_drty_pg_steal_clns -“触发缓冲池牺牲页清除程序次数”监视元素

因为数据库的牺牲缓冲区替换期间需要同步写入而调用页清除程序的次数。

表 1243. 表函数监视信息

表函数	监视元素收集级别
MON_GET_BUFFERPOOL 表函数 - 获取缓冲池度量值	DATA OBJECT METRICS BASE

表 1244. 快照监视信息

快照级别	逻辑数据分组	监视开关
数据库	dbase	缓冲池

可将快照监视的计数器重置。

表 1245. 事件监视信息

事件类型	逻辑数据分组	监视开关
数据库	event_db	始终收集

用法

通过使用以下公式，可以计算通过此元素表示的所有清除程序调用的百分比:

```
  pool_drty_pg_steal_clns
    / (pool_drty_pg_steal_clns
+ pool_drty_pg_thrsh_clns
+ pool_lsn_gap_clns)
```

如果此比率很低，那么可能指示您定义的页清除程序过多。如果 **chngpgs_thresh** 配置参数设置得太小，那么可能会写出将来会成为脏页的页。主动清除使得缓冲池失去了尽可能延迟写入这一用途。

如果此比率很高，那么可能表明未定义足够的页清除程序。页清除程序不足将导致发生故障后的恢复时间延长。

当 DB2_USE_ALTERNATE_PAGE_CLEANING 注册表变量为 OFF 时:

- **pool_drty_pg_steal_clns** 监视元素将插入到监视器流中。
- **pool_drty_pg_steal_clns** 监视元素计算因为数据库的牺牲缓冲区替换期间需要同步写入而调用页清除程序的次数。

当 DB2_USE_ALTERNATE_PAGE_CLEANING 注册表变量为 ON 时:

- **pool_drty_pg_steal_clns** 监视元素将 0 插入到监视器流中。
- 牺牲缓冲区替换期间需要同步写入时不会显式触发页清除程序。要确定为数据库或特定缓冲池配置的页清除程序数是否正确，请参阅 **pool_no_victim_buffer** 监视元素。

注: 虽然会将脏页写入磁盘，但除非需要空间来读入新页，否则不会立即从缓冲池中除去这些页。

pool_drty_pg_thrsh_clns -“触发缓冲池阈值清除程序次数”监视元素

因为缓冲池达到数据库的脏页阈值条件而调用页清除程序的次数。

表 1246. 表函数监视信息

表函数	监视元素收集命令和级别
MON_GET_BUFFERPOOL 表函数 - 获取缓冲池度量值	DATA OBJECT METRICS BASE

表 1247. 快照监视信息

快照级别	逻辑数据分组	监视开关
数据库	dbase	缓冲池

可将快照监视的计数器重置。

表 1248. 事件监视信息

事件类型	逻辑数据分组	监视开关
数据库	event_db	始终收集

用法 该阈值由 **chngpgs_thresh** 配置参数设置。它是应用于缓冲池大小的百分比。如果池中的脏页数超过此值，将触发清除程序。

如果 **chngpgs_thresh** 配置参数值设置得太小，那么可能会过早将页写出，从而导致需要重新读入那些页。如果此值设置得过大，那么可能会累积过多的页，从而要求用户以同步方式将这些页写出。

当 DB2_USE_ALTERNATE_PAGE_CLEANING 注册表变量为 OFF 时:

- **pool_drty_pg_thrsh_clns** 监视元素将插入到监视器流中。
- **pool_drty_pg_thrsh_clns** 监视元素计算因为缓冲池达到数据库的脏页阈值条件而调用页清除程序的次数。

当 DB2_USE_ALTERNATE_PAGE_CLEANING 注册表变量为 ON 时:

- **pool_drty_pg_thrsh_clns** 监视元素将 0 插入到监视器流中。
- 页清除程序总是处于活动状态，以试图确保有足够的可用缓冲区以供牺牲，而不是等待条件值触发。

pool_failed_async_data_reqs -“失败数据预取请求数”监视元素

尝试让数据预取请求排队但失败的次数。一个可能原因是预取队列已满。

表 1249. 表函数监视信息

表函数	监视元素收集级别
MON_FORMAT_XML_METRICS_BY_ROW - 获取所有度量值的基于已格式化行的输出	不适用; 报告 XML 文档中作为格式化函数的输入提供的所有元素
MON_GET_ACTIVITY_DETAILS 表函数 - 获取完整的活动详细信息（在 XML 文档 DETAILS 中报告）	ACTIVITY METRICS BASE
MON_GET_BUFFERPOOL 表函数 - 获取缓冲池度量值	DATA OBJECT METRICS BASE
MON_GET_CONNECTION 表函数 - 获取连接度量值	REQUEST METRICS BASE
MON_GET_CONNECTION_DETAILS 表函数 - 获取详细的连接度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE
MON_GET_PKG_CACHE_STMT 表函数 - 获取程序包高速缓存中的 SQL 语句活动度量值	ACTIVITY METRICS BASE
MON_GET_PKG_CACHE_STMT_DETAILS 表函数 - 获取程序包高速缓存条目的详细度量值（在 XML 文档 DETAILS 中报告）	ACTIVITY METRICS BASE
MON_GET_SERVICE_SUBCLASS 表函数 - 获取服务子类度量值	REQUEST METRICS BASE

表 1249. 表函数监视信息 (续)

表函数	监视元素收集级别
MON_GET_SERVICE_SUBCLASS_DETAILS 表函数 - 获取详细的服务子类度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE
MON_GET_TABLESPACE 表函数 - 获取表空间度量值	DATA OBJECT METRICS BASE
MON_GET_UNIT_OF_WORK 表函数 - 获取工作单元度量值	REQUEST METRICS BASE
MON_GET_UNIT_OF_WORK 表函数 - 获取详细的工作单元度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE
MON_GET_WORKLOAD 表函数 - 获取工作负载度量值	REQUEST METRICS BASE
MON_GET_WORKLOAD_DETAILS 表函数 - 获取详细的工作负载度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE

表 1250. 事件监视信息

事件类型	逻辑数据分组	监视开关
活动	event_activity（在 details_xml 文档中报告） event_activitymetrics	ACTIVITY METRICS BASE
程序包高速缓存	pkgcache（在 metrics.xml 文档中报告） pkgcache_metrics	ACTIVITY METRICS BASE
工作单元	uow（在 metrics.xml 文档中报告） uow_metrics	REQUEST METRICS BASE
统计信息	event_scstats（在 details_xml 文档中报告）	REQUEST METRICS BASE
统计信息	event_wlstats（在 details_xml 文档中报告）	REQUEST METRICS BASE

用法

此元素与其他 **pool_failed_async_..._reqs** 元素一起表示未能添加至预取队列的预取请求数。如果预取队列太小或预取程序运行太慢，那么请求可能无法添加至预取队列。无法将请求添加至预取队列时，数据库代理进程通常会同步执行磁盘 IO，这比预取效率低。可将这些元素与其他预取程序监视元素配合使用来确定在系统上执行的预取的效率。例如，可使用类似如下的公式来了解已成功添加至预取队列的请求的百分比:

```
1 -
 (
 POOL_FAILED_ASYNC_DATA_REQS +
 POOL_FAILED_ASYNC_INDEX_REQS +
 POOL_FAILED_ASYNC_XDA_REQS +
 POOL_FAILED_ASYNC_TEMP_DATA_REQS +
```

```
POOL_FAILED_ASYNC_TEMP_INDEX_REQS +
POOL_FAILED_ASYNC_TEMP_XDA_REQS
)
÷
(
  (
  POOL_FAILED_ASYNC_DATA_REQS +
  POOL_FAILED_ASYNC_INDEX_REQS +
  POOL_FAILED_ASYNC_XDA_REQS +
  POOL_FAILED_ASYNC_TEMP_DATA_REQS +
  POOL_FAILED_ASYNC_TEMP_INDEX_REQS +
  POOL_FAILED_ASYNC_TEMP_XDA_REQS
  )
+
  (
  POOL_QUEUED_ASYNC_DATA_REQS +
  POOL_QUEUED_ASYNC_INDEX_REQS +
  POOL_QUEUED_ASYNC_XDA_REQS +
  POOL_QUEUED_ASYNC_TEMP_DATA_REQS +
  POOL_QUEUED_ASYNC_TEMP_INDEX_REQS +
  POOL_QUEUED_ASYNC_TEMP_XDA_REQS
  )
) × 100
```

此公式计算成功预取请求占所提出请求总数的百分比。失败预取请求是未能添加至预取队列的请求。如果要创建大量请求，或者预取程序运行得太慢（因为配置或调整太差），那么请求可能无法添加至预取队列。如果成功请求的百分比很低，那么这可能指示预取机制出现瓶颈。可能需要通过修改配置参数 **num_ioservers** 的值来配置更多预取程序。预取队列变满的情况还有可能由于代理程序提交太多小请求导致；可使用相关监视元素 **pool_queued_async_..._pages** 和 **pool_queued_async_..._reqs** 来确定平均预取请求大小。

pool_failed_async_index_reqs -“失败的索引预取请求数”监视元素

尝试让索引预取请求排队但失败的次数。一个可能原因是预取队列已满并且无法从空闲列表获取请求。

表 1251. 表函数监视信息

表函数	监视元素收集级别
MON_FORMAT_XML_METRICS_BY_ROW - 获取所有度量值的基于已格式化行的输出	不适用; 报告 XML 文档中作为格式化函数的输入提供的所有元素
MON_GET_ACTIVITY_DETAILS 表函数 - 获取完整的活动详细信息（在 XML 文档 DETAILS 中报告）	ACTIVITY METRICS BASE
MON_GET_BUFFERPOOL 表函数 - 获取缓冲池度量值	DATA OBJECT METRICS BASE
MON_GET_CONNECTION 表函数 - 获取连接度量值	REQUEST METRICS BASE
MON_GET_CONNECTION_DETAILS 表函数 - 获取详细的连接度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE
MON_GET_PKG_CACHE_STMT 表函数 - 获取程序包高速缓存中的 SQL 语句活动度量值	ACTIVITY METRICS BASE

表 1251. 表函数监视信息 (续)

表函数	监视元素收集级别
MON_GET_PKG_CACHE_STMT_DETAILS 表函数 - 获取程序包高速缓存条目的详细度量值（在 XML 文档 DETAILS 中报告）	ACTIVITY METRICS BASE
MON_GET_SERVICE_SUBCLASS 表函数 - 获取服务子类度量值	REQUEST METRICS BASE
MON_GET_SERVICE_SUBCLASS_DETAILS 表函数 - 获取详细的服务子类度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE
MON_GET_TABLESPACE 表函数 - 获取表空间度量值	DATA OBJECT METRICS BASE
MON_GET_UNIT_OF_WORK 表函数 - 获取工作单元度量值	REQUEST METRICS BASE
MON_GET_UNIT_OF_WORK 表函数 - 获取详细的工作单元度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE
MON_GET_WORKLOAD 表函数 - 获取工作负载度量值	REQUEST METRICS BASE
MON_GET_WORKLOAD_DETAILS 表函数 - 获取详细的工作负载度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE

表 1252. 事件监视信息

事件类型	逻辑数据分组	监视开关
活动	event_activity（在 details_xml 文档中报告） event_activitymetrics	ACTIVITY METRICS BASE
程序包高速缓存	pkgcache（在 metrics.xml 文档中报告） pkgcache_metrics	ACTIVITY METRICS BASE
工作单元	uow（在 metrics.xml 文档中报告） uow_metrics	REQUEST METRICS BASE
统计信息	event_scstats（在 details_xml 文档中报告）	REQUEST METRICS BASE
统计信息	event_wlstats（在 details_xml 文档中报告）	REQUEST METRICS BASE

用法

此元素与其他 **pool_failed_async_..._reqs** 元素一起表示未能添加至预取队列的预取请求数。如果预取队列太小或预取程序运行太慢，那么请求可能无法添加至预取队列。无法将请求添加至预取队列时，数据库代理进程通常会同步执行磁盘 IO，这比预取效率低。可将这些元素与其他预取程序监视元素配合使用来确定在系统上执行的预取的效率。例如，可使用类似如下的公式来了解已成功添加至预取队列的请求的百分比:

```
1 -
 (
 POOL_FAILED_ASYNC_DATA_REQS +
 POOL_FAILED_ASYNC_INDEX_REQS +
 POOL_FAILED_ASYNC_XDA_REQS +
 POOL_FAILED_ASYNC_TEMP_DATA_REQS +
 POOL_FAILED_ASYNC_TEMP_INDEX_REQS +
 POOL_FAILED_ASYNC_TEMP_XDA_REQS
 )
 ÷
 (
   (
   POOL_FAILED_ASYNC_DATA_REQS +
   POOL_FAILED_ASYNC_INDEX_REQS +
   POOL_FAILED_ASYNC_XDA_REQS +
   POOL_FAILED_ASYNC_TEMP_DATA_REQS +
   POOL_FAILED_ASYNC_TEMP_INDEX_REQS +
   POOL_FAILED_ASYNC_TEMP_XDA_REQS
   )
 +
   (
   POOL_QUEUED_ASYNC_DATA_REQS +
   POOL_QUEUED_ASYNC_INDEX_REQS +
   POOL_QUEUED_ASYNC_XDA_REQS +
   POOL_QUEUED_ASYNC_TEMP_DATA_REQS +
   POOL_QUEUED_ASYNC_TEMP_INDEX_REQS +
   POOL_QUEUED_ASYNC_TEMP_XDA_REQS
   )
 ) × 100
```

此公式计算成功预取请求占所提出请求总数的百分比。失败预取请求是未能添加至预取队列的请求。如果要创建大量请求，或者预取程序运行得太慢（因为配置或调整太差），那么请求可能无法添加至预取队列。如果成功请求的百分比很低，那么这可能指示预取机制出现瓶颈。可能需要通过修改配置参数 **num_ioservers** 的值来配置更多预取程序。预取队列变满的情况还有可能由于代理程序提交太多小请求导致；可使用相关监视元素 **pool_queued_async_..._pages** 和 **pool_queued_async_..._reqs** 来确定平均预取请求大小。

pool_failed_async_other_reqs -“失败的非预取请求数”监视元素

尝试让非预取请求排队但失败的次数。此元素适用于预取程序完成的非预取工作。请求失败的一个可能原因是预取队列已满。

表 1253. 表函数监视信息

表函数	监视元素收集级别
MON_FORMAT_XML_METRICS_BY_ROW - 获取所有度量值的基于已格式化行的输出	不适用；报告 XML 文档中作为格式化函数的输入提供的所有元素
MON_GET_ACTIVITY_DETAILS 表函数 - 获取完整的活动详细信息（在 XML 文档 DETAILS 中报告）	ACTIVITY METRICS BASE
MON_GET_BUFFERPOOL 表函数 - 获取缓冲池度量值	DATA OBJECT METRICS BASE
MON_GET_CONNECTION 表函数 - 获取连接度量值	REQUEST METRICS BASE

表 1253. 表函数监视信息 (续)

表函数	监视元素收集级别
MON_GET_CONNECTION_DETAILS 表函数 - 获取详细的连接度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE
MON_GET_PKG_CACHE_STMT 表函数 - 获取程序包高速缓存中的 SQL 语句活动度量值	ACTIVITY METRICS BASE
MON_GET_PKG_CACHE_STMT_DETAILS 表函数 - 获取程序包高速缓存条目的详细度量值（在 XML 文档 DETAILS 中报告）	ACTIVITY METRICS BASE
MON_GET_SERVICE_SUBCLASS 表函数 - 获取服务子类度量值	REQUEST METRICS BASE
MON_GET_SERVICE_SUBCLASS_DETAILS 表函数 - 获取详细的服务子类度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE
MON_GET_TABLESPACE 表函数 - 获取表空间度量值	DATA OBJECT METRICS BASE
MON_GET_UNIT_OF_WORK 表函数 - 获取工作单元度量值	REQUEST METRICS BASE
MON_GET_UNIT_OF_WORK 表函数 - 获取详细的工作单元度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE
MON_GET_WORKLOAD 表函数 - 获取工作负载度量值	REQUEST METRICS BASE
MON_GET_WORKLOAD_DETAILS 表函数 - 获取详细的工作负载度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE

表 1254. 事件监视信息

事件类型	逻辑数据分组	监视开关
活动	event_activity（在 details_xml 文档中报告） event_activitymetrics	ACTIVITY METRICS BASE
程序包高速缓存	pkgcache（在 metrics.xml 文档中报告） pkgcache_metrics	ACTIVITY METRICS BASE
工作单元	uow（在 metrics.xml 文档中报告） uow_metrics	REQUEST METRICS BASE
统计信息	event_scstats（在 details_xml 文档中报告）	REQUEST METRICS BASE
统计信息	event_wlstats（在 details_xml 文档中报告）	REQUEST METRICS BASE

用法

此监视元素报告未能添加至预取队列的请求数，这些请求针对与存取方案决定的预取无关的工作。备份实用程序之类的实用程序使用预取程序机制来执行它们的任务，但以不同于存取方案针对 SQL 语句执行的方式执行。请求可能因为预取队列已满而无法添加至该队列。

pool_failed_async_temp_data_reqs -“临时表空间的失败数据预取请求数”监视元素

尝试让临时表空间的数据预取请求排队但失败的次数。一个可能原因是预取队列已满并且无法从空闲列表获取请求。

表 1255. 表函数监视信息

表函数	监视元素收集级别
MON_FORMAT_XML_METRICS_BY_ROW - 获取所有度量值的基于已格式化行的输出	不适用; 报告 XML 文档中作为格式化函数的输入提供的所有元素
MON_GET_ACTIVITY_DETAILS 表函数 - 获取完整的活动详细信息（在 XML 文档 DETAILS 中报告）	ACTIVITY METRICS BASE
MON_GET_BUFFERPOOL 表函数 - 获取缓冲池度量值	DATA OBJECT METRICS BASE
MON_GET_CONNECTION 表函数 - 获取连接度量值	REQUEST METRICS BASE
MON_GET_CONNECTION_DETAILS 表函数 - 获取详细的连接度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE
MON_GET_PKG_CACHE_STMT 表函数 - 获取程序包高速缓存中的 SQL 语句活动度量值	ACTIVITY METRICS BASE
MON_GET_PKG_CACHE_STMT_DETAILS 表函数 - 获取程序包高速缓存条目的详细度量值（在 XML 文档 DETAILS 中报告）	ACTIVITY METRICS BASE
MON_GET_SERVICE_SUBCLASS 表函数 - 获取服务子类度量值	REQUEST METRICS BASE
MON_GET_SERVICE_SUBCLASS_DETAILS 表函数 - 获取详细的服务子类度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE
MON_GET_TABLESPACE 表函数 - 获取表空间度量值	DATA OBJECT METRICS BASE
MON_GET_UNIT_OF_WORK 表函数 - 获取工作单元度量值	REQUEST METRICS BASE
MON_GET_UNIT_OF_WORK 表函数 - 获取详细的工作单元度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE
MON_GET_WORKLOAD 表函数 - 获取工作负载度量值	REQUEST METRICS BASE

表 1255. 表函数监视信息 (续)

表函数	监视元素收集级别
MON_GET_WORKLOAD_DETAILS 表函数 - 获取详细的工作负载度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE

表 1256. 事件监视信息

事件类型	逻辑数据分组	监视开关
活动	event_activity（在 details_xml 文档中报告） event_activitymetrics	ACTIVITY METRICS BASE
程序包高速缓存	pkgcache（在 metrics.xml 文档中报告） pkgcache_metrics	ACTIVITY METRICS BASE
工作单元	uow（在 metrics.xml 文档中报告） uow_metrics	REQUEST METRICS BASE
统计信息	event_scstats（在 details_xml 文档中报告）	REQUEST METRICS BASE
统计信息	event_wlstats（在 details_xml 文档中报告）	REQUEST METRICS BASE

用法

此元素与其他 **pool_failed_async_..._reqs** 元素一起表示未能添加至预取队列的预取请求数。如果预取队列太小或预取程序运行太慢，那么请求可能无法添加至预取队列。无法将请求添加至预取队列时，数据库代理进程通常会同步执行磁盘 IO，这比预取效率低。可将这些元素与其他预取程序监视元素配合使用来确定在系统上执行的预取的效率。例如，可使用类似如下的公式来了解已成功添加至预取队列的请求的百分比:

```
1 -
 (
 POOL_FAILED_ASYNC_DATA_REQS +
 POOL_FAILED_ASYNC_INDEX_REQS +
 POOL_FAILED_ASYNC_XDA_REQS +
 POOL_FAILED_ASYNC_TEMP_DATA_REQS +
 POOL_FAILED_ASYNC_TEMP_INDEX_REQS +
 POOL_FAILED_ASYNC_TEMP_XDA_REQS
 )
 ÷
 (
   (
   POOL_FAILED_ASYNC_DATA_REQS +
   POOL_FAILED_ASYNC_INDEX_REQS +
   POOL_FAILED_ASYNC_XDA_REQS +
   POOL_FAILED_ASYNC_TEMP_DATA_REQS +
   POOL_FAILED_ASYNC_TEMP_INDEX_REQS +
   POOL_FAILED_ASYNC_TEMP_XDA_REQS
   )
 +
   (
   POOL_QUEUED_ASYNC_DATA_REQS +
   POOL_QUEUED_ASYNC_INDEX_REQS +
   POOL_QUEUED_ASYNC_XDA_REQS +
```

```
    POOL_QUEUED_ASYNC_TEMP_DATA_REQS +
    POOL_QUEUED_ASYNC_TEMP_INDEX_REQS +
    POOL_QUEUED_ASYNC_TEMP_XDA_REQS
    )
  ) × 100
```

此公式计算成功预取请求占所提出请求总数的百分比。失败预取请求是未能添加至预取队列的请求。如果要创建大量请求，或者预取程序运行得太慢（因为配置或调整太差），那么请求可能无法添加至预取队列。如果成功请求的百分比很低，那么这可能指示预取机制出现瓶颈。可能需要通过修改配置参数 **num_ioservers** 的值来配置更多预取程序。预取队列变满的情况还有可能由于代理程序提交太多小请求导致；可使用相关监视元素 **pool_queued_async_..._pages** 和 **pool_queued_async_..._reqs** 来确定平均预取请求大小。

pool_failed_async_temp_index_reqs -“临时表空间的失败索引预取请求数”监视元素

尝试让临时表空间的索引预取请求排队但失败的次数。一个可能原因是预取队列已满并且无法从空闲列表获取请求。

表 1257. 表函数监视信息

表函数	监视元素收集级别
MON_FORMAT_XML_METRICS_BY_ROW - 获取所有度量值的基于已格式化行的输出	不适用; 报告 XML 文档中作为格式化函数的输入提供的所有元素
MON_GET_ACTIVITY_DETAILS 表函数 - 获取完整的活动详细信息（在 XML 文档 DETAILS 中报告）	ACTIVITY METRICS BASE
MON_GET_BUFFERPOOL 表函数 - 获取缓冲池度量值	DATA OBJECT METRICS BASE
MON_GET_CONNECTION 表函数 - 获取连接度量值	REQUEST METRICS BASE
MON_GET_CONNECTION_DETAILS 表函数 - 获取详细的连接度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE
MON_GET_PKG_CACHE_STMT 表函数 - 获取程序包高速缓存中的 SQL 语句活动度量值	ACTIVITY METRICS BASE
MON_GET_PKG_CACHE_STMT_DETAILS 表函数 - 获取程序包高速缓存条目的详细度量值（在 XML 文档 DETAILS 中报告）	ACTIVITY METRICS BASE
MON_GET_SERVICE_SUBCLASS 表函数 - 获取服务子类度量值	REQUEST METRICS BASE
MON_GET_SERVICE_SUBCLASS_DETAILS 表函数 - 获取详细的服务子类度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE
MON_GET_TABLESPACE 表函数 - 获取表空间度量值	DATA OBJECT METRICS BASE
MON_GET_UNIT_OF_WORK 表函数 - 获取工作单元度量值	REQUEST METRICS BASE

表 1257. 表函数监视信息 (续)

表函数	监视元素收集级别
MON_GET_UNIT_OF_WORK 表函数 - 获取详细的工作单元度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE
MON_GET_WORKLOAD 表函数 - 获取工作负载度量值	REQUEST METRICS BASE
MON_GET_WORKLOAD_DETAILS 表函数 - 获取详细的工作负载度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE

表 1258. 事件监视信息

事件类型	逻辑数据分组	监视开关
活动	event_activity（在 details_xml 文档中报告） event_activitymetrics	ACTIVITY METRICS BASE
程序包高速缓存	pkgcache（在 metrics.xml 文档中报告） pkgcache_metrics	ACTIVITY METRICS BASE
工作单元	uow（在 metrics.xml 文档中报告） uow_metrics	REQUEST METRICS BASE
统计信息	event_scstats（在 details_xml 文档中报告）	REQUEST METRICS BASE
统计信息	event_wlstats（在 details_xml 文档中报告）	REQUEST METRICS BASE

用法

此元素与其他 **pool_failed_async_..._reqs** 元素一起表示未能添加至预取队列的预取请求数。如果预取队列太小或预取程序运行太慢，那么请求可能无法添加至预取队列。无法将请求添加至预取队列时，数据库代理进程通常会同步执行磁盘 IO，这比预取效率低。可将这些元素与其他预取程序监视元素配合使用来确定在系统上执行的预取的效率。例如，可使用类似如下的公式来了解已成功添加至预取队列的请求的百分比:

```
1 -
 (
 POOL_FAILED_ASYNC_DATA_REQS +
 POOL_FAILED_ASYNC_INDEX_REQS +
 POOL_FAILED_ASYNC_XDA_REQS +
 POOL_FAILED_ASYNC_TEMP_DATA_REQS +
 POOL_FAILED_ASYNC_TEMP_INDEX_REQS +
 POOL_FAILED_ASYNC_TEMP_XDA_REQS
 )
 ÷
 (
   (
   POOL_FAILED_ASYNC_DATA_REQS +
   POOL_FAILED_ASYNC_INDEX_REQS +
   POOL_FAILED_ASYNC_XDA_REQS +
   POOL_FAILED_ASYNC_TEMP_DATA_REQS +
```

```
  POOL_FAILED_ASYNC_TEMP_INDEX_REQS +
  POOL_FAILED_ASYNC_TEMP_XDA_REQS
  )
+
  (
  POOL_QUEUED_ASYNC_DATA_REQS +
  POOL_QUEUED_ASYNC_INDEX_REQS +
  POOL_QUEUED_ASYNC_XDA_REQS +
  POOL_QUEUED_ASYNC_TEMP_DATA_REQS +
  POOL_QUEUED_ASYNC_TEMP_INDEX_REQS +
  POOL_QUEUED_ASYNC_TEMP_XDA_REQS
  )
) × 100
```

此公式计算成功预取请求占所提出请求总数的百分比。失败预取请求是未能添加至预取队列的请求。如果要创建大量请求，或者预取程序运行得太慢（因为配置或调整太差），那么请求可能无法添加至预取队列。如果成功请求的百分比很低，那么这可能指示预取机制出现瓶颈。可能需要通过修改配置参数 **num_ioservers** 的值来配置更多预取程序。预取队列变满的情况还有可能由于代理程序提交太多小请求导致；可使用相关监视元素 **pool_queued_async_..._pages** 和 **pool_queued_async_..._reqs** 来确定平均预取请求大小。

pool_failed_async_temp_xda_reqs -“临时表空间的失败 XDA 预取请求数”监视元素

尝试让临时表空间的 XML 存储器对象 (XDA) 数据预取请求排队但失败的次数。一个可能原因是预取队列已满并且无法从空闲列表获取请求。

表 1259. 表函数监视信息

表函数	监视元素收集级别
MON_FORMAT_XML_METRICS_BY_ROW - 获取所有度量值的基于已格式化行的输出	不适用; 报告 XML 文档中作为格式化函数的输入提供的所有元素
MON_GET_ACTIVITY_DETAILS 表函数 - 获取完整的活动详细信息（在 XML 文档 DETAILS 中报告）	ACTIVITY METRICS BASE
MON_GET_BUFFERPOOL 表函数 - 获取缓冲池度量值	DATA OBJECT METRICS BASE
MON_GET_CONNECTION 表函数 - 获取连接度量值	REQUEST METRICS BASE
MON_GET_CONNECTION_DETAILS 表函数 - 获取详细的连接度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE
MON_GET_PKG_CACHE_STMT 表函数 - 获取程序包高速缓存中的 SQL 语句活动度量值	ACTIVITY METRICS BASE
MON_GET_PKG_CACHE_STMT_DETAILS 表函数 - 获取程序包高速缓存条目的详细度量值（在 XML 文档 DETAILS 中报告）	ACTIVITY METRICS BASE
MON_GET_SERVICE_SUBCLASS 表函数 - 获取服务子类度量值	REQUEST METRICS BASE

表 1259. 表函数监视信息 (续)

表函数	监视元素收集级别
MON_GET_SERVICE_SUBCLASS_DETAILS 表函数 - 获取详细的服务子类度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE
MON_GET_TABLESPACE 表函数 - 获取表空间度量值	DATA OBJECT METRICS BASE
MON_GET_UNIT_OF_WORK 表函数 - 获取工作单元度量值	REQUEST METRICS BASE
MON_GET_UNIT_OF_WORK 表函数 - 获取详细的工作单元度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE
MON_GET_WORKLOAD 表函数 - 获取工作负载度量值	REQUEST METRICS BASE
MON_GET_WORKLOAD_DETAILS 表函数 - 获取详细的工作负载度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE

表 1260. 事件监视信息

事件类型	逻辑数据分组	监视开关
活动	event_activity（在 details_xml 文档中报告） event_activitymetrics	ACTIVITY METRICS BASE
程序包高速缓存	pkgcache（在 metrics.xml 文档中报告） pkgcache_metrics	ACTIVITY METRICS BASE
工作单元	uow（在 metrics.xml 文档中报告） uow_metrics	REQUEST METRICS BASE
统计信息	event_scstats（在 details_xml 文档中报告）	REQUEST METRICS BASE
统计信息	event_wlstats（在 details_xml 文档中报告）	REQUEST METRICS BASE

用法

此元素与其他 **pool_failed_async_..._reqs** 元素一起表示未能添加至预取队列的预取请求数。如果预取队列太小或预取程序运行太慢，那么请求可能无法添加至预取队列。无法将请求添加至预取队列时，数据库代理进程通常会同步执行磁盘 IO，这比预取效率低。可将这些元素与其他预取程序监视元素配合使用来确定在系统上执行的预取的效率。例如，可使用类似如下的公式来了解已成功添加至预取队列的请求的百分比:

```
1 -
 (
 POOL_FAILED_ASYNC_DATA_REQS +
 POOL_FAILED_ASYNC_INDEX_REQS +
 POOL_FAILED_ASYNC_XDA_REQS +
 POOL_FAILED_ASYNC_TEMP_DATA_REQS +
```

```
POOL_FAILED_ASYNC_TEMP_INDEX_REQS +
POOL_FAILED_ASYNC_TEMP_XDA_REQS
)
÷
(
  (
  POOL_FAILED_ASYNC_DATA_REQS +
  POOL_FAILED_ASYNC_INDEX_REQS +
  POOL_FAILED_ASYNC_XDA_REQS +
  POOL_FAILED_ASYNC_TEMP_DATA_REQS +
  POOL_FAILED_ASYNC_TEMP_INDEX_REQS +
  POOL_FAILED_ASYNC_TEMP_XDA_REQS
  )
+
  (
  POOL_QUEUED_ASYNC_DATA_REQS +
  POOL_QUEUED_ASYNC_INDEX_REQS +
  POOL_QUEUED_ASYNC_XDA_REQS +
  POOL_QUEUED_ASYNC_TEMP_DATA_REQS +
  POOL_QUEUED_ASYNC_TEMP_INDEX_REQS +
  POOL_QUEUED_ASYNC_TEMP_XDA_REQS
  )
) × 100
```

此公式计算成功预取请求占所提出请求总数的百分比。失败预取请求是未能添加至预取队列的请求。如果要创建大量请求，或者预取程序运行得太慢（因为配置或调整太差），那么请求可能无法添加至预取队列。如果成功请求的百分比很低，那么这可能指示预取机制出现瓶颈。可能需要通过修改配置参数 **num_ioservers** 的值来配置更多预取程序。预取队列变满的情况还有可能由于代理程序提交太多小请求导致；可使用相关监视元素 **pool_queued_async_..._pages** 和 **pool_queued_async_..._reqs** 来确定平均预取请求大小。

pool_failed_async_xda_reqs -“失败的 XDA 预取请求数”监视元素

尝试让 XML 存储器对象 (XDA) 数据预取请求排队但失败的次数。一个可能原因是预取队列已满并且无法从空闲列表获取请求。

表 1261. 表函数监视信息

表函数	监视元素收集级别
MON_FORMAT_XML_METRICS_BY_ROW - 获取所有度量值的基于已格式化行的输出	不适用; 报告 XML 文档中作为格式化函数的输入提供的所有元素
MON_GET_ACTIVITY_DETAILS 表函数 - 获取完整的活动详细信息（在 XML 文档 DETAILS 中报告）	ACTIVITY METRICS BASE
MON_GET_BUFFERPOOL 表函数 - 获取缓冲池度量值	DATA OBJECT METRICS BASE
MON_GET_CONNECTION 表函数 - 获取连接度量值	REQUEST METRICS BASE
MON_GET_CONNECTION_DETAILS 表函数 - 获取详细的连接度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE
MON_GET_PKG_CACHE_STMT 表函数 - 获取程序包高速缓存中的 SQL 语句活动度量值	ACTIVITY METRICS BASE

表 1261. 表函数监视信息 (续)

表函数	监视元素收集级别
MON_GET_PKG_CACHE_STMT_DETAILS 表函数 - 获取程序包高速缓存条目的详细度量值（在 XML 文档 DETAILS 中报告）	ACTIVITY METRICS BASE
MON_GET_SERVICE_SUBCLASS 表函数 - 获取服务子类度量值	REQUEST METRICS BASE
MON_GET_SERVICE_SUBCLASS_DETAILS 表函数 - 获取详细的服务子类度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE
MON_GET_TABLESPACE 表函数 - 获取表空间度量值	DATA OBJECT METRICS BASE
MON_GET_UNIT_OF_WORK 表函数 - 获取工作单元度量值	REQUEST METRICS BASE
MON_GET_UNIT_OF_WORK 表函数 - 获取详细的工作单元度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE
MON_GET_WORKLOAD 表函数 - 获取工作负载度量值	REQUEST METRICS BASE
MON_GET_WORKLOAD_DETAILS 表函数 - 获取详细的工作负载度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE

表 1262. 事件监视信息

事件类型	逻辑数据分组	监视开关
活动	event_activity（在 details_xml 文档中报告） event_activitymetrics	ACTIVITY METRICS BASE
程序包高速缓存	pkgcache（在 metrics.xml 文档中报告） pkgcache_metrics	ACTIVITY METRICS BASE
工作单元	uow（在 metrics.xml 文档中报告） uow_metrics	REQUEST METRICS BASE
统计信息	event_scstats（在 details_xml 文档中报告）	REQUEST METRICS BASE
统计信息	event_wlstats（在 details_xml 文档中报告）	REQUEST METRICS BASE

用法

此元素与其他 **pool_failed_async_..._reqs** 元素一起表示未能添加至预取队列的预取请求数。如果预取队列太小或预取程序运行太慢，那么请求可能无法添加至预取队列。无法将请求添加至预取队列时，数据库代理进程通常会同步执行磁盘 IO，这比预取效率低。可将这些元素与其他预取程序监视元素配合使用来确定在系统上执行的预取的效率。例如，可使用类似如下的公式来了解已成功添加至预取队列的请求的百分比:

```
1 -
 (
 POOL_FAILED_ASYNC_DATA_REQS +
 POOL_FAILED_ASYNC_INDEX_REQS +
 POOL_FAILED_ASYNC_XDA_REQS +
 POOL_FAILED_ASYNC_TEMP_DATA_REQS +
 POOL_FAILED_ASYNC_TEMP_INDEX_REQS +
 POOL_FAILED_ASYNC_TEMP_XDA_REQS
 )
 ÷
 (
   (
   POOL_FAILED_ASYNC_DATA_REQS +
   POOL_FAILED_ASYNC_INDEX_REQS +
   POOL_FAILED_ASYNC_XDA_REQS +
   POOL_FAILED_ASYNC_TEMP_DATA_REQS +
   POOL_FAILED_ASYNC_TEMP_INDEX_REQS +
   POOL_FAILED_ASYNC_TEMP_XDA_REQS
   )
 +
   (
   POOL_QUEUED_ASYNC_DATA_REQS +
   POOL_QUEUED_ASYNC_INDEX_REQS +
   POOL_QUEUED_ASYNC_XDA_REQS +
   POOL_QUEUED_ASYNC_TEMP_DATA_REQS +
   POOL_QUEUED_ASYNC_TEMP_INDEX_REQS +
   POOL_QUEUED_ASYNC_TEMP_XDA_REQS
   )
 ) × 100
```

此公式计算成功预取请求占所提出请求总数的百分比。失败预取请求是未能添加至预取队列的请求。如果要创建大量请求，或者预取程序运行得太慢（因为配置或调整太差），那么请求可能无法添加至预取队列。如果成功请求的百分比很低，那么这可能指示预取机制出现瓶颈。可能需要通过修改配置参数 **num_ioservers** 的值来配置更多预取程序。预取队列变满的情况还有可能由于代理程序提交太多小请求导致；可使用相关监视元素 **pool_queued_async_..._pages** 和 **pool_queued_async_..._reqs** 来确定平均预取请求大小。

pool_id -“内存池标识”

内存池的类型。

表 1263. 快照监视信息

快照级别	逻辑数据分组	监视开关
数据库管理器	memory_pool	基本
数据库	memory_pool	基本
应用程序	memory_pool	基本

表 1264. 事件监视信息

事件类型	逻辑数据分组	监视开关
数据库	event_dbmemuse	-
连接	event_connmemuse	-

用法

要跟踪系统内存使用情况，请将此值与 **pool_config_size**、**pool_cur_size** 和 **pool_watermark** 配合使用。

使用 **pool_id** 来标识系统监视器输出中讨论的内存池。各种内存池标识都可在 sqlmon.h 中找到。在正常操作情况下，可以使用下列每个内存池的其中一个或多个。

API 常量	描述
SQLM_HEAP_APPLICATION	应用程序堆
SQLM_HEAP_DATABASE	数据库堆
SQLM_HEAP_LOCK_MGR	锁管理器堆
SQLM_HEAP_UTILITY	备份/复原/实用程序堆
SQLM_HEAP_STATISTICS	统计信息堆
SQLM_HEAP_PACKAGE_CACHE	程序包高速缓存堆
SQLM_HEAP_CAT_CACHE	目录高速缓存堆
SQLM_HEAP_MONITOR	数据库监视器堆
SQLM_HEAP_STATEMENT	语句堆
SQLM_HEAP_FCMBP	FCMBP 堆
SQLM_HEAP_IMPORT_POOL	导入池
SQLM_HEAP_OTHER	其他内存
SQLM_HEAP_BP	缓冲池堆
SQLM_HEAP_APPL_SHARED	应用程序共享堆
SQLM_HEAP_SHARED_SORT	排序共享堆

pool_index_gbp_indep_pages_found_in_lbp -“本地缓冲池中发现的独立于组缓冲池的索引页数”监视元素

代理程序在本地缓冲池 (LBP) 中发现的独立于组缓冲池 (GBP) 的索引页数。

表 1265. 表函数监视信息

表函数	监视元素收集级别
MON_FORMAT_XML_METRICS_BY_ROW - 获取所有度量值的基于已格式化行的输出	不适用; 报告 XML 文档中作为格式化函数的输入提供的所有元素
MON_GET_ACTIVITY_DETAILS 表函数 - 获取完整的活动详细信息（在 XML 文档 DETAILS 中报告）	ACTIVITY METRICS BASE
MON_GET_BUFFERPOOL 表函数 - 获取缓冲池度量值	DATA OBJECT METRICS BASE
MON_GET_CONNECTION 表函数 - 获取连接度量值	REQUEST METRICS BASE
MON_GET_CONNECTION_DETAILS 表函数 - 获取详细连接度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE
MON_GET_PKG_CACHE_STMT 表函数 - 获取程序包高速缓存中的 SQL 语句活动度量值	ACTIVITY METRICS BASE

表 1265. 表函数监视信息 (续)

表函数	监视元素收集级别
MON_GET_PKG_CACHE_STMT_DETAILS - 获取程序包高速缓存条目的详细度量值（在 XML 文档 DETAILS 中报告）	ACTIVITY METRICS BASE
MON_GET_SERVICE_SUBCLASS 表函数 - 获取服务子类度量值	REQUEST METRICS BASE
MON_GET_SERVICE_SUBCLASS_DETAILS 表函数 - 获取详细的服务子类度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE
MON_GET_TABLESPACE 表函数 - 获取表空间度量值	DATA OBJECT METRICS BASE
MON_GET_UNIT_OF_WORK 表函数 - 获取工作单元度量值	REQUEST METRICS BASE
MON_GET_UNIT_OF_WORK 表函数 - 获取详细的工作单元度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE
MON_GET_WORKLOAD 表函数 - 获取工作负载度量值	REQUEST METRICS BASE
MON_GET_WORKLOAD_DETAILS 表函数 - 获取详细的工作负载度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE

表 1266. 事件监视信息

事件类型	逻辑数据分组	监视开关
活动	event_activitymetrics	ACTIVITY METRICS BASE
程序包高速缓存	pkgcache_metrics	ACTIVITY METRICS BASE
工作单元	uow_metrics	REQUEST METRICS BASE

pool_index_gbp_invalid_pages -“组缓冲池无效索引页数”监视元素

尝试从组缓冲池读取索引页（因为该页在本地缓冲池中无效）的次数。

表 1267. 表函数监视信息

表函数	监视元素收集级别
MON_FORMAT_XML_METRICS_BY_ROW - 获取所有度量值的基于已格式化行的输出	不适用; 报告 XML 文档中作为格式化函数的输入提供的所有元素
MON_GET_ACTIVITY_DETAILS 表函数 - 获取完整的活动详细信息（在 XML 文档 DETAILS 中报告）	ACTIVITY METRICS BASE
MON_GET_BUFFERPOOL 表函数 - 获取缓冲池度量值	DATA OBJECT METRICS BASE
MON_GET_CONNECTION 表函数 - 获取连接度量值	REQUEST METRICS BASE

表 1267. 表函数监视信息 (续)

表函数	监视元素收集级别
MON_GET_CONNECTION_DETAILS 表函数 - 获取详细的连接度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE
MON_GET_PKG_CACHE_STMT 表函数 - 获取程序包高速缓存中的 SQL 语句活动度量值	ACTIVITY METRICS BASE
MON_GET_SERVICE_SUBCLASS 表函数 - 获取服务子类度量值	REQUEST METRICS BASE
MON_GET_SERVICE_SUBCLASS_DETAILS 表函数 - 获取详细的服务子类度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE
MON_GET_TABLESPACE 表函数 - 获取表空间度量值	DATA OBJECT METRICS BASE
MON_GET_UNIT_OF_WORK 表函数 - 获取工作单元度量值	REQUEST METRICS BASE
MON_GET_UNIT_OF_WORK 表函数 - 获取详细的工作单元度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE
MON_GET_WORKLOAD 表函数 - 获取工作负载度量值	REQUEST METRICS BASE
MON_GET_WORKLOAD_DETAILS 表函数 - 获取详细的工作负载度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE

表 1268. 事件监视信息

事件类型	逻辑数据分组	监视开关
活动	event_activity（在 details_xml 文档中报告）	ACTIVITY METRICS BASE
活动	event_activitymetrics	ACTIVITY METRICS BASE
统计信息	event_scstats（在 details_xml 文档中报告）	REQUEST METRICS BASE
统计信息	event_wlstats（在 details_xml 文档中报告）	REQUEST METRICS BASE
工作单元	-	始终收集
程序包高速缓存	-	始终收集
锁定	-	始终收集

用法

要确定在本地缓冲池中发现所请求索引页的频率，请使用以下公式:

```
(POOL_INDEX_LBP_PAGES_FOUND - POOL_ASYNC_INDEX_LBP_PAGES_FOUND) / POOL_INDEX_L_READS
```

要确定在组缓冲池中发现所请求索引页的次数，请使用以下公式

```
(POOL_INDEX_GBP_L_READS - POOL_INDEX_GBP_P_READS) / POOL_INDEX_GBP_L_READS
```

本地缓冲池和组缓冲池命中率都是集群高速缓存设施的整体性能中的重要因素。使用这些公式可帮助您确定本地缓冲池或组缓冲池会否成为数据库吞吐量的限制因素。

pool_index_gbp_l_reads -“组缓冲池索引逻辑读取数”监视元素

尝试从组缓冲池读取依赖于组缓冲池 (GBP) 的索引页（因为该页在本地缓冲池中无效或不存在）的次数。

表 1269. 表函数监视信息

表函数	监视元素收集级别
MON_FORMAT_XML_METRICS_BY_ROW - 获取所有度量值的基于已格式化行的输出	不适用; 报告 XML 文档中作为格式化函数的输入提供的所有元素
MON_GET_ACTIVITY_DETAILS 表函数 - 获取完整的活动详细信息（在 XML 文档 DETAILS 中报告）	ACTIVITY METRICS BASE
MON_GET_BUFFERPOOL 表函数 - 获取缓冲池度量值	DATA OBJECT METRICS BASE
MON_GET_CONNECTION 表函数 - 获取连接度量值	REQUEST METRICS BASE
MON_GET_CONNECTION_DETAILS 表函数 - 获取详细的连接度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE
MON_GET_PKG_CACHE_STMT 表函数 - 获取程序包高速缓存中的 SQL 语句活动度量值	ACTIVITY METRICS BASE
MON_GET_SERVICE_SUBCLASS 表函数 - 获取服务子类度量值	REQUEST METRICS BASE
MON_GET_SERVICE_SUBCLASS_DETAILS 表函数 - 获取详细的服务子类度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE
MON_GET_TABLESPACE 表函数 - 获取表空间度量值	DATA OBJECT METRICS BASE
MON_GET_UNIT_OF_WORK 表函数 - 获取工作单元度量值	REQUEST METRICS BASE
MON_GET_UNIT_OF_WORK 表函数 - 获取详细的工作单元度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE
MON_GET_WORKLOAD 表函数 - 获取工作负载度量值	REQUEST METRICS BASE
MON_GET_WORKLOAD_DETAILS 表函数 - 获取详细的工作负载度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE

表 1270. 事件监视信息

事件类型	逻辑数据分组	监视开关
活动	event_activity（在 details_xml 文档中报告）	ACTIVITY METRICS BASE
活动	event_activitymetrics	ACTIVITY METRICS BASE

表 1270. 事件监视信息 (续)

事件类型	逻辑数据分组	监视开关
统计信息	event_scstats（在 details_xml 文档中报告）	REQUEST METRICS BASE
统计信息	event_wlstats（在 details_xml 文档中报告）	REQUEST METRICS BASE
工作单元	-	始终收集
程序包高速缓存	-	始终收集
锁定	-	始终收集

用法

要确定在本地缓冲池中发现所请求索引页的频率，请使用以下公式:

```
(POOL_INDEX_LBP_PAGES_FOUND - POOL_ASYNC_INDEX_LBP_PAGES_FOUND) / POOL_INDEX_L_READS
```

要确定在组缓冲池中发现所请求索引页的次数，请使用以下公式

```
(POOL_INDEX_GBP_L_READS - POOL_INDEX_GBP_P_READS) / POOL_INDEX_GBP_L_READS
```

本地缓冲池和组缓冲池命中率都是集群高速缓存设施的整体性能中的重要因素。使用这些公式可帮助您确定本地缓冲池或组缓冲池会否成为数据库吞吐量的限制因素。

pool_index_gbp_p_reads -“组缓冲池索引物理读取数”监视元素

尝试将磁盘中依赖于组缓冲池 (GBP) 的索引页读取到本地缓冲池中（因为在 GBP 中找不到该页）的次数。

表 1271. 表函数监视信息

表函数	监视元素收集级别
MON_FORMAT_XML_METRICS_BY_ROW - 获取所有度量值的基于已格式化行的输出	不适用; 报告 XML 文档中作为格式化函数的输入提供的所有元素
MON_GET_ACTIVITY_DETAILS 表函数 - 获取完整的活动详细信息（在 XML 文档 DETAILS 中报告）	ACTIVITY METRICS BASE
MON_GET_BUFFERPOOL 表函数 - 获取缓冲池度量值	DATA OBJECT METRICS BASE
MON_GET_CONNECTION 表函数 - 获取连接度量值	REQUEST METRICS BASE
MON_GET_CONNECTION_DETAILS 表函数 - 获取详细的连接度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE
MON_GET_PKG_CACHE_STMT 表函数 - 获取程序包高速缓存中的 SQL 语句活动度量值	ACTIVITY METRICS BASE
MON_GET_SERVICE_SUBCLASS 表函数 - 获取服务子类度量值	REQUEST METRICS BASE
MON_GET_SERVICE_SUBCLASS_DETAILS 表函数 - 获取详细的服务子类度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE

表 1271. 表函数监视信息 (续)

表函数	监视元素收集级别
MON_GET_TABLESPACE 表函数 - 获取表空间度量值	DATA OBJECT METRICS BASE
MON_GET_UNIT_OF_WORK 表函数 - 获取工作单元度量值	REQUEST METRICS BASE
MON_GET_UNIT_OF_WORK 表函数 - 获取详细的工作单元度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE
MON_GET_WORKLOAD 表函数 - 获取工作负载度量值	REQUEST METRICS BASE
MON_GET_WORKLOAD_DETAILS 表函数 - 获取详细的工作负载度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE

表 1272. 事件监视信息

事件类型	逻辑数据分组	监视开关
活动	event_activity（在 details_xml 文档中报告）	ACTIVITY METRICS BASE
活动	event_activitymetrics	ACTIVITY METRICS BASE
统计信息	event_scstats（在 details_xml 文档中报告）	REQUEST METRICS BASE
统计信息	event_wlstats（在 details_xml 文档中报告）	REQUEST METRICS BASE
工作单元	-	始终收集
程序包高速缓存	-	始终收集
锁定	-	始终收集

用法

要确定在本地缓冲池中发现所请求索引页的频率，请使用以下公式:

```
(POOL_INDEX_LBP_PAGES_FOUND - POOL_ASYNC_INDEX_LBP_PAGES_FOUND) / POOL_INDEX_L_READS
```

要确定在组缓冲池中发现所请求索引页的次数，请使用以下公式

```
(POOL_INDEX_GBP_L_READS - POOL_INDEX_GBP_P_READS) / POOL_INDEX_GBP_L_READS
```

本地缓冲池和组缓冲池命中率都是集群高速缓存设施的整体性能中的重要因素。使用这些公式可帮助您确定本地缓冲池或组缓冲池会否成为数据库吞吐量的限制因素。

pool_index_lbp_pages_found -“发现的本地缓冲池索引页数”监视元素

索引页出现在本地缓冲池中的次数。

表 1273. 表函数监视信息

表函数	监视元素收集级别
MON_FORMAT_XML_METRICS_BY_ROW - 获取所有度量值的基于已格式化行的输出	不适用；报告 XML 文档中作为格式化函数的输入提供的所有元素

表 1273. 表函数监视信息 (续)

表函数	监视元素收集级别
MON_GET_ACTIVITY_DETAILS 表函数 - 获取完整的活动详细信息（在 XML 文档 DETAILS 中报告）	ACTIVITY METRICS BASE
MON_GET_BUFFERPOOL 表函数 - 获取缓冲池度量值	DATA OBJECT METRICS BASE
MON_GET_CONNECTION 表函数 - 获取连接度量值	REQUEST METRICS BASE
MON_GET_CONNECTION_DETAILS 表函数 - 获取详细的连接度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE
MON_GET_PKG_CACHE_STMT 表函数 - 获取程序包高速缓存中的 SQL 语句活动度量值	ACTIVITY METRICS BASE
MON_GET_SERVICE_SUBCLASS 表函数 - 获取服务子类度量值	REQUEST METRICS BASE
MON_GET_SERVICE_SUBCLASS_DETAILS 表函数 - 获取详细的服务子类度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE
MON_GET_TABLESPACE 表函数 - 获取表空间度量值	DATA OBJECT METRICS BASE
MON_GET_UNIT_OF_WORK 表函数 - 获取工作单元度量值	REQUEST METRICS BASE
MON_GET_UNIT_OF_WORK 表函数 - 获取详细的工作单元度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE
MON_GET_WORKLOAD 表函数 - 获取工作负载度量值	REQUEST METRICS BASE
MON_GET_WORKLOAD_DETAILS 表函数 - 获取详细的工作负载度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE

表 1274. 事件监视信息

事件类型	逻辑数据分组	监视开关
活动	event_activity（在 details_xml 文档中报告）	ACTIVITY METRICS BASE
活动	event_activitymetrics	ACTIVITY METRICS BASE
统计信息	event_scstats（在 details_xml 文档中报告）	REQUEST METRICS BASE
统计信息	event_wlstats（在 details_xml 文档中报告）	REQUEST METRICS BASE
工作单元	-	始终收集
程序包高速缓存	-	始终收集
锁定	-	始终收集

用法

要确定在本地缓冲池中发现所请求索引页的频率，请使用以下公式:

```
(POOL_INDEX_LBP_PAGES_FOUND - POOL_ASYNC_INDEX_LBP_PAGES_FOUND) / POOL_INDEX_L_READS
```

要确定在组缓冲池中发现所请求索引页的次数，请使用以下公式

```
(POOL_INDEX_GBP_L_READS - POOL_INDEX_GBP_P_READS) / POOL_INDEX_GBP_L_READS
```

本地缓冲池和组缓冲池命中率都是集群高速缓存设施的整体性能中的重要因素。使用这些公式可帮助您确定本地缓冲池或组缓冲池会否成为数据库吞吐量的限制因素。

pool_index_l_reads -“缓冲池索引逻辑读取数”监视元素

指示从常规表空间和大型表空间的逻辑缓冲池中请求获取的索引页数。

表 1275. 表函数监视信息

表函数	监视元素收集级别
MON_FORMAT_XML_METRICS_BY_ROW - 获取所有度量值的基于已格式化行的输出	不适用; 报告 XML 文档中作为格式化函数的输入提供的所有元素
MON_GET_ACTIVITY_DETAILS 表函数 - 获取完整的活动详细信息（在 XML 文档 DETAILS 中报告）	ACTIVITY METRICS BASE
MON_GET_BUFFERPOOL 表函数 - 获取缓冲池度量值	DATA OBJECT METRICS BASE
MON_GET_CONNECTION 表函数 - 获取连接度量值	REQUEST METRICS BASE
MON_GET_CONNECTION_DETAILS 表函数 - 获取详细的连接度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE
MON_GET_PKG_CACHE_STMT 表函数 - 获取程序包高速缓存中的 SQL 语句活动度量值	ACTIVITY METRICS BASE
MON_GET_PKG_CACHE_STMT_DETAILS 表函数 - 获取程序包高速缓存条目的详细度量值	ACTIVITY METRICS BASE
MON_GET_SERVICE_SUBCLASS 表函数 - 获取服务子类度量值	REQUEST METRICS BASE
MON_GET_SERVICE_SUBCLASS_DETAILS 表函数 - 获取详细的服务子类度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE
MON_GET_TABLESPACE 表函数 - 获取表空间度量值	DATA OBJECT METRICS BASE
MON_GET_UNIT_OF_WORK 表函数 - 获取工作单元度量值	REQUEST METRICS BASE
MON_GET_UNIT_OF_WORK 表函数 - 获取详细的工作单元度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE
MON_GET_WORKLOAD 表函数 - 获取工作负载度量值	REQUEST METRICS BASE

表 1275. 表函数监视信息 (续)

表函数	监视元素收集级别
MON_GET_WORKLOAD_DETAILS 表函数 - 获取详细的工作负载度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE

表 1276. 快照监视信息

快照级别	逻辑数据分组	监视开关
数据库	dbase	缓冲池
表空间	tablespace	缓冲池
缓冲池	bufferpool	缓冲池
应用程序	appl	缓冲池
应用程序	stmt	缓冲池
动态 SQL	dynsql	缓冲池，语句

可将快照监视的计数器重置。

表 1277. 事件监视信息

事件类型	逻辑数据分组	监视开关
活动	event_activity（在 details_xml 文档中报告）	ACTIVITY METRICS BASE
活动	event_activitymetrics	ACTIVITY METRICS BASE
统计信息	event_scstats（在 details_xml 文档中报告）	REQUEST METRICS BASE
统计信息	event_wlstats（在 details_xml 文档中报告）	REQUEST METRICS BASE
工作单元	在 system_metrics 文档中报告。	REQUEST METRICS BASE
数据库	event_db	始终收集
表空间	event_tablespace	始终收集
连接	event_conn	始终收集
语句	event_stmt	始终收集
活动	event_activity	缓冲池，语句
程序包高速缓存	在 activity_metrics 文档中报告。	ACTIVITY METRICS BASE

用法

此计数包括索引页处于下列情况时对索引页的访问:

- 当数据库管理器需要处理页时索引页已经在缓冲池中
- 应读取到缓冲池中，数据库管理器才能处理页。

可将 **pool_index_l_reads** 与 **pool_index_p_reads** 和 **pool_async_index_reads** 一起使用来计算缓冲池的索引页命中率。例如，以下公式返回 DB2 环境中的索引页命中率（在没有 IBM DB2 pureScale Feature 的情况下）。

```
((pool_index_lbp_pages_found
- pool_async_index_lbp_pages_found - pool_temp_index_l_reads)
/ pool_index_l_reads) × 100
```

有关更多信息，请参阅第 1379 页的『用于计算缓冲池命中率的公式』。

如果命中率很低，那么提高缓冲池页数可以改进性能。

pool_index_p_reads -“缓冲池索引物理读取数”监视元素

指示从常规表空间和大型表空间的物理表空间容器中读取的索引页数。

表 1278. 表函数监视信息

表函数	监视元素收集级别
MON_FORMAT_XML_METRICS_BY_ROW - 获取所有度量值的基于已格式化行的输出	不适用; 报告 XML 文档中作为格式化函数的输入提供的所有元素
MON_GET_ACTIVITY_DETAILS 表函数 - 获取完整的活动详细信息（在 XML 文档 DETAILS 中报告）	ACTIVITY METRICS BASE
MON_GET_BUFFERPOOL 表函数 - 获取缓冲池度量值	DATA OBJECT METRICS BASE
MON_GET_CONNECTION 表函数 - 获取连接度量值	REQUEST METRICS BASE
MON_GET_CONNECTION_DETAILS 表函数 - 获取详细的连接度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE
MON_GET_PKG_CACHE_STMT 表函数 - 获取程序包高速缓存中的 SQL 语句活动度量值	ACTIVITY METRICS BASE
MON_GET_PKG_CACHE_STMT_DETAILS 表函数 - 获取程序包高速缓存条目的详细度量值	ACTIVITY METRICS BASE
MON_GET_SERVICE_SUBCLASS 表函数 - 获取服务子类度量值	REQUEST METRICS BASE
MON_GET_SERVICE_SUBCLASS_DETAILS 表函数 - 获取详细的服务子类度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE
MON_GET_TABLESPACE 表函数 - 获取表空间度量值	DATA OBJECT METRICS BASE
MON_GET_UNIT_OF_WORK 表函数 - 获取工作单元度量值	REQUEST METRICS BASE
MON_GET_UNIT_OF_WORK 表函数 - 获取详细的工作单元度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE
MON_GET_WORKLOAD 表函数 - 获取工作负载度量值	REQUEST METRICS BASE
MON_GET_WORKLOAD_DETAILS 表函数 - 获取详细的工作负载度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE

表 1279. 快照监视信息

快照级别	逻辑数据分组	监视开关
数据库	dbase	缓冲池
表空间	tablespace	缓冲池
缓冲池	bufferpool	缓冲池
应用程序	appl	缓冲池
应用程序	stmt	缓冲池
动态 SQL	dynsql	缓冲池，语句

可将快照监视的计数器重置。

表 1280. 事件监视信息

事件类型	逻辑数据分组	监视开关
活动	event_activity（在 details_xml 文档中报告）	ACTIVITY METRICS BASE
活动	event_activitymetrics	ACTIVITY METRICS BASE
统计信息	event_scstats（在 details_xml 文档中报告）	REQUEST METRICS BASE
统计信息	event_wlstats（在 details_xml 文档中报告）	REQUEST METRICS BASE
工作单元	在 system_metrics 文档中报告。	REQUEST METRICS BASE
数据库	event_db	始终收集
表空间	event_tablespace	始终收集
连接	event_conn	始终收集
语句	event_stmt	始终收集
活动	event_activity	缓冲池，语句
程序包高速缓存	在 activity_metrics 文档中报告。	ACTIVITY METRICS BASE

用法

可将 **pool_index_p_reads** 与 **pool_index_l_reads** 和 **pool_async_index_reads** 一起使用来计算缓冲池的索引页命中率。例如，以下公式返回 DB2 环境中的索引页命中率（在没有 IBM DB2 pureScale Feature 的情况下）。

```
((pool_index_lbp_pages_found
- pool_async_index_lbp_pages_found - pool_temp_index_l_reads)
/ pool_index_l_reads) × 100
```

有关更多信息，请参阅第 1379 页的『用于计算缓冲池命中率的公式』。

pool_index_writes -“缓冲池索引写次数”监视元素

指示缓冲池索引页物理写至磁盘的次数。

表 1281. 表函数监视信息

表函数	监视元素收集级别
MON_FORMAT_XML_METRICS_BY_ROW - 获取所有度量值的基于已格式化行的输出	不适用; 报告 XML 文档中作为格式化函数的输入提供的所有元素
MON_GET_ACTIVITY_DETAILS 表函数 - 获取完整的活动详细信息（在 XML 文档 DETAILS 中报告）	ACTIVITY METRICS BASE
MON_GET_BUFFERPOOL 表函数 - 获取缓冲池度量值	DATA OBJECT METRICS BASE
MON_GET_CONNECTION 表函数 - 获取连接度量值	REQUEST METRICS BASE
MON_GET_CONNECTION_DETAILS 表函数 - 获取详细的连接度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE
MON_GET_PKG_CACHE_STMT 表函数 - 获取程序包高速缓存中的 SQL 语句活动度量值	ACTIVITY METRICS BASE
MON_GET_PKG_CACHE_STMT_DETAILS 表函数 - 获取程序包高速缓存条目的详细度量值	ACTIVITY METRICS BASE
MON_GET_SERVICE_SUBCLASS 表函数 - 获取服务子类度量值	REQUEST METRICS BASE
MON_GET_SERVICE_SUBCLASS_DETAILS 表函数 - 获取详细的服务子类度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE
MON_GET_TABLESPACE 表函数 - 获取表空间度量值	DATA OBJECT METRICS BASE
MON_GET_UNIT_OF_WORK 表函数 - 获取工作单元度量值	REQUEST METRICS BASE
MON_GET_UNIT_OF_WORK 表函数 - 获取详细的工作单元度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE
MON_GET_WORKLOAD 表函数 - 获取工作负载度量值	REQUEST METRICS BASE
MON_GET_WORKLOAD_DETAILS 表函数 - 获取详细的工作负载度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE

表 1282. 快照监视信息

快照级别	逻辑数据分组	监视开关
数据库	dbase	缓冲池
表空间	tablespace	缓冲池
缓冲池	bufferpool	缓冲池
应用程序	appl	缓冲池

可将快照监视的计数器重置。

表 1283. 事件监视信息

事件类型	逻辑数据分组	监视开关
活动	event_activity（在 details_xml 文档中报告）	ACTIVITY METRICS BASE
活动	event_activitymetrics	ACTIVITY METRICS BASE
统计信息	event_scstats（在 details_xml 文档中报告）	REQUEST METRICS BASE
统计信息	event_wlstats（在 details_xml 文档中报告）	REQUEST METRICS BASE
工作单元	在 system_metrics 文档中报告。	REQUEST METRICS BASE
数据库	event_db	始终收集
表空间	event_tablespace	始终收集
连接	event_conn	始终收集
程序包高速缓存	在 activity_metrics 文档中报告。	ACTIVITY METRICS BASE

用法

与数据页一样，缓冲池索引页将因为下列原因写至磁盘:

- 释放缓冲池中的某一页以便可以读取另一页
- 将缓冲池清仓

系统不会总是通过写入页为新页腾出空间。如果该页未被更新，那么只需将其替换。此元素不考虑此替换。

在需要缓冲池空间之前，索引页可由异步页清除程序代理程序写入。这些异步索引页写次数与同步索引页写次数一起包括在此元素的值中（参见 **pool_async_index_writes** 监视元素）。

如果缓冲池索引页被写入磁盘的次数在 **pool_index_p_reads** 监视元素值中占较高的百分比，那么可通过增加可供数据库使用的缓冲池页数来提高性能。

计算此百分比时，忽略一开始填充缓冲池所需的物理读取的数目。要确定写入的页数:

1. 运行应用程序以装入缓冲区。
2. 记录此元素的值。
3. 再次运行应用程序。
4. 从此元素的新值中减去步骤 2 中记录的值。

为了避免在应用程序的各次运行之间取消分配缓冲池，应该执行下列其中一项操作:

- 使用 **ACTIVATE DATABASE** 命令来激活数据库。
- 将一个空闲的应用程序连接至数据库。

如果所有应用程序都要更新该数据库，那么提高缓冲池大小对性能可能没有多大影响，原因是大多数页包含已更新数据，而这些数据必须写入磁盘。

pool_lsn_gap_clns -“触发缓冲池日志空间清除程序次数”监视元素

因为使用的记录空间已达到数据库的预定义条件而调用页清除程序的次数。

表 1284. 表函数监视信息

表函数	监视元素收集命令和级别
MON_GET_BUFFERPOOL 表函数 - 获取缓冲池度量值	DATA OBJECT METRICS BASE

表 1285. 快照监视信息

快照级别	逻辑数据分组	监视开关
数据库	dbase	缓冲池

可将快照监视的计数器重置。

表 1286. 事件监视信息

事件类型	逻辑数据分组	监视开关
数据库	event_db	始终收集

用法

此元素可用来帮助评估您是否具有足够的记录空间以及您是否需要更多或更大的日志文件。

页清除条件将由 **softmax** 配置参数的设置确定。如果缓冲池中最旧的页包含一个更新，此更新由比条件值定义的当前日志位置还要旧的日志记录描述，那么将触发页清除程序。

当 DB2_USE_ALTERNATE_PAGE_CLEANING 注册表变量为 OFF 时:

- **pool_lsn_gap_clns** 监视元素将插入到监视器流中。
- 如果缓冲池中最旧的页包含一个更新，此更新由比条件值定义的当前日志位置还要旧的日志记录描述，那么将触发页清除程序。

当 DB2_USE_ALTERNATE_PAGE_CLEANING 注册表变量为 ON 时:

- **pool_lsn_gap_clns** 监视元素会将 0 插入到监视器流中。
- 页清除程序主动写入页而不是等待条件值触发。

pool_no_victim_buffer -“缓冲池无牺牲缓冲区次数”监视元素

代理程序没有预先选择的可用牺牲缓冲区的次数。

表 1287. 表函数监视信息

表函数	监视元素收集级别
MON_GET_BUFFERPOOL 表函数 - 获取缓冲池度量值	DATA OBJECT METRICS BASE

表 1288. 快照监视信息

快照级别	逻辑数据分组	监视开关
数据库	dbase	缓冲池
表空间	tablespace	缓冲池
缓冲池	bufferpool	缓冲池

可将快照监视的计数器重置。

表 1289. 事件监视信息

事件类型	逻辑数据分组	监视开关
数据库	event_db	始终收集
表空间	event_tablespace	始终收集

用法 此元素可用来帮助评估使用前摄页清除时您是否具有足够的页清除程序用于给定缓冲池。

DB2_USE_ALTERNATE_PAGE_CLEANING 注册表变量为 ON 时，pool_no_victim_buffer 元素计算代理程序找不到预先选择的牺牲缓冲区可供立即使用，并且强制搜索缓冲池以查找适合的牺牲缓冲区的次数。

如果 pool_no_victim_buffer 元素的值相对于缓冲池中的逻辑读取数过高，那么 DB2 数据库系统难以确保有足够的良好牺牲缓冲区可供使用。增加页清除程序数目将增加 DB2 提供预选牺牲缓冲区的能力。

当 DB2_USE_ALTERNATE_PAGE_CLEANING 注册表变量为 OFF 时，pool_no_victim_buffer 元素没有预测值，并且可以安全地忽略。在此配置中，DB2 数据库系统不会尝试确保代理程序预选可供使用的牺牲缓冲区，所以对缓冲池的大多数访问需要代理程序搜索缓冲池以查找牺牲缓冲区。

pool_queued_async_data_pages -“预取请求的数据页数”监视元素

成功请求预取的数据页的数目。

表 1290. 表函数监视信息

表函数	监视元素收集级别
MON_FORMAT_XML_METRICS_BY_ROW - 获取所有度量值的基于已格式化行的输出	不适用; 报告 XML 文档中作为格式化函数的输入提供的所有元素
MON_GET_ACTIVITY_DETAILS 表函数 - 获取完整的活动详细信息（在 XML 文档 DETAILS 中报告）	ACTIVITY METRICS BASE
MON_GET_BUFFERPOOL 表函数 - 获取缓冲池度量值	DATA OBJECT METRICS BASE
MON_GET_CONNECTION 表函数 - 获取连接度量值	REQUEST METRICS BASE
MON_GET_CONNECTION_DETAILS 表函数 - 获取详细的连接度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE
MON_GET_PKG_CACHE_STMT 表函数 - 获取程序包高速缓存中的 SQL 语句活动度量值	ACTIVITY METRICS BASE

表 1290. 表函数监视信息 (续)

表函数	监视元素收集级别
MON_GET_PKG_CACHE_STMT_DETAILS 表函数 - 获取程序包高速缓存条目的详细度量值（在 XML 文档 DETAILS 中报告）	ACTIVITY METRICS BASE
MON_GET_SERVICE_SUBCLASS 表函数 - 获取服务子类度量值	REQUEST METRICS BASE
MON_GET_SERVICE_SUBCLASS_DETAILS 表函数 - 获取详细的服务子类度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE
MON_GET_TABLESPACE 表函数 - 获取表空间度量值	DATA OBJECT METRICS BASE
MON_GET_UNIT_OF_WORK 表函数 - 获取工作单元度量值	REQUEST METRICS BASE
MON_GET_UNIT_OF_WORK 表函数 - 获取详细的工作单元度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE
MON_GET_WORKLOAD 表函数 - 获取工作负载度量值	REQUEST METRICS BASE
MON_GET_WORKLOAD_DETAILS 表函数 - 获取详细的工作负载度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE

表 1291. 事件监视信息

事件类型	逻辑数据分组	监视开关
活动	event_activity（在 details_xml 文档中报告） event_activitymetrics	ACTIVITY METRICS BASE
程序包高速缓存	pkgcache（在 metrics.xml 文档中报告） pkgcache_metrics	ACTIVITY METRICS BASE
工作单元	uow（在 metrics.xml 文档中报告） uow_metrics	REQUEST METRICS BASE
统计信息	event_scstats（在 details_xml 文档中报告）	REQUEST METRICS BASE
统计信息	event_wlstats（在 details_xml 文档中报告）	REQUEST METRICS BASE

用法

此监视元素与其他 **pool_queued_async_..._pages** 元素一起表示预取请求检索到的数据页数。可使用此信息来确定是否在系统上有效地执行了预取请求。例如，可使用类似如下的公式来计算每个预取请求的平均页数:

```
(POOL_QUEUED_ASYNC_DATA_PAGES +
 POOL_QUEUED_ASYNC_INDEX_PAGES +
 POOL_QUEUED_ASYNC_XDA_PAGES +
 POOL_QUEUED_ASYNC_TEMP_DATA_PAGES +
```

```
   POOL_QUEUED_ASYNC_TEMP_INDEX_PAGES +
   POOL_QUEUED_ASYNC_TEMP_XDA_PAGES)
÷
  (POOL_QUEUED_ASYNC_DATA_REQS +
   POOL_QUEUED_ASYNC_INDEX_REQS +
   POOL_QUEUED_ASYNC_XDA_REQS +
   POOL_QUEUED_ASYNC_TEMP_DATA_REQS +
   POOL_QUEUED_ASYNC_TEMP_INDEX_REQS +
   POOL_QUEUED_ASYNC_TEMP_XDA_REQS)
```

如果每个请求的平均页数很低，并且系统上的预取量很大，那么系统要执行的 IO 操作可能比需要执行的 IO 操作多。通常，请求大小基于预取大小，后者至少应与扩展数据块大小相同。所以，平均请求大小较小可能指示预取大小设置得太低，将预取大小增加至扩展数据块大小的倍数可能会改进性能。而且应注意，平均请求大小较小可能意味着预取队列填充得太快，所以还应监视关联 **pool_failed_async_..._reqs** 监视元素

pool_queued_async_data_reqs -“数据预取请求数”监视元素

成功添加至预取队列的数据预取请求的数目。

表 1292. 表函数监视信息

表函数	监视元素收集级别
MON_FORMAT_XML_METRICS_BY_ROW - 获取所有度量值的基于已格式化行的输出	不适用; 报告 XML 文档中作为格式化函数的输入提供的所有元素
MON_GET_ACTIVITY_DETAILS 表函数 - 获取完整的活动详细信息（在 XML 文档 DETAILS 中报告）	ACTIVITY METRICS BASE
MON_GET_BUFFERPOOL 表函数 - 获取缓冲池度量值	DATA OBJECT METRICS BASE
MON_GET_CONNECTION 表函数 - 获取连接度量值	REQUEST METRICS BASE
MON_GET_CONNECTION_DETAILS 表函数 - 获取详细的连接度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE
MON_GET_PKG_CACHE_STMT 表函数 - 获取程序包高速缓存中的 SQL 语句活动度量值	ACTIVITY METRICS BASE
MON_GET_PKG_CACHE_STMT_DETAILS 表函数 - 获取程序包高速缓存条目的详细度量值（在 XML 文档 DETAILS 中报告）	ACTIVITY METRICS BASE
MON_GET_SERVICE_SUBCLASS 表函数 - 获取服务子类度量值	REQUEST METRICS BASE
MON_GET_SERVICE_SUBCLASS_DETAILS 表函数 - 获取详细的服务子类度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE
MON_GET_TABLESPACE 表函数 - 获取表空间度量值	DATA OBJECT METRICS BASE
MON_GET_UNIT_OF_WORK 表函数 - 获取工作单元度量值	REQUEST METRICS BASE

表 1292. 表函数监视信息 (续)

表函数	监视元素收集级别
MON_GET_UNIT_OF_WORK 表函数 - 获取详细的工作单元度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE
MON_GET_WORKLOAD 表函数 - 获取工作负载度量值	REQUEST METRICS BASE
MON_GET_WORKLOAD_DETAILS 表函数 - 获取详细的工作负载度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE

表 1293. 事件监视信息

事件类型	逻辑数据分组	监视开关
活动	event_activity（在 details_xml 文档中报告） event_activitymetrics	ACTIVITY METRICS BASE
程序包高速缓存	pkgcache（在 metrics.xml 文档中报告） pkgcache_metrics	ACTIVITY METRICS BASE
工作单元	uow（在 metrics.xml 文档中报告） uow_metrics	REQUEST METRICS BASE
统计信息	event_scstats（在 details_xml 文档中报告）	REQUEST METRICS BASE
统计信息	event_wlstats（在 details_xml 文档中报告）	REQUEST METRICS BASE

用法

此元素与其他 **pool_queued_async_*_reqs** 元素一起表示已添加至预取队列的预取请求数。可使用此信息来查看数据库管理器执行预取的频率。可将这些元素与其他预取程序监视元素配合使用来确定在系统上执行的预取的效率。例如，可使用类似如下的公式来了解已成功添加至预取队列的请求的百分比:

```
1 -
 (
 POOL_FAILED_ASYNC_DATA_REQS +
 POOL_FAILED_ASYNC_INDEX_REQS +
 POOL_FAILED_ASYNC_XDA_REQS +
 POOL_FAILED_ASYNC_TEMP_DATA_REQS +
 POOL_FAILED_ASYNC_TEMP_INDEX_REQS +
 POOL_FAILED_ASYNC_TEMP_XDA_REQS
 )
 ÷
 (
   (
   POOL_FAILED_ASYNC_DATA_REQS +
   POOL_FAILED_ASYNC_INDEX_REQS +
   POOL_FAILED_ASYNC_XDA_REQS +
   POOL_FAILED_ASYNC_TEMP_DATA_REQS +
   POOL_FAILED_ASYNC_TEMP_INDEX_REQS +
   POOL_FAILED_ASYNC_TEMP_XDA_REQS
   )
```

```
+
  (
  POOL_QUEUED_ASYNC_DATA_REQS +
  POOL_QUEUED_ASYNC_INDEX_REQS +
  POOL_QUEUED_ASYNC_XDA_REQS +
  POOL_QUEUED_ASYNC_TEMP_DATA_REQS +
  POOL_QUEUED_ASYNC_TEMP_INDEX_REQS +
  POOL_QUEUED_ASYNC_TEMP_XDA_REQS
  )
) * 100
```

此公式计算失败预取请求占所提出请求总数的百分比。失败预取请求是未能添加至预取队列的请求。如果该百分比很低，那么可能需要通过修改 **num_ioservers** 配置参数来配置更多预取程序。

pool_queued_async_index_pages -“预取请求的索引页数”监视元素

成功请求预取的索引页数。

表 1294. 表函数监视信息

表函数	监视元素收集级别
MON_FORMAT_XML_METRICS_BY_ROW - 获取所有度量值的基于已格式化行的输出	不适用; 报告 XML 文档中作为格式化函数的输入提供的所有元素
MON_GET_ACTIVITY_DETAILS 表函数 - 获取完整的活动详细信息（在 XML 文档 DETAILS 中报告）	ACTIVITY METRICS BASE
MON_GET_BUFFERPOOL 表函数 - 获取缓冲池度量值	DATA OBJECT METRICS BASE
MON_GET_CONNECTION 表函数 - 获取连接度量值	REQUEST METRICS BASE
MON_GET_CONNECTION_DETAILS 表函数 - 获取详细的连接度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE
MON_GET_PKG_CACHE_STMT 表函数 - 获取程序包高速缓存中的 SQL 语句活动度量值	ACTIVITY METRICS BASE
MON_GET_PKG_CACHE_STMT_DETAILS 表函数 - 获取程序包高速缓存条目的详细度量值（在 XML 文档 DETAILS 中报告）	ACTIVITY METRICS BASE
MON_GET_SERVICE_SUBCLASS 表函数 - 获取服务子类度量值	REQUEST METRICS BASE
MON_GET_SERVICE_SUBCLASS_DETAILS 表函数 - 获取详细的服务子类度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE
MON_GET_TABLESPACE 表函数 - 获取表空间度量值	DATA OBJECT METRICS BASE
MON_GET_UNIT_OF_WORK 表函数 - 获取工作单元度量值	REQUEST METRICS BASE
MON_GET_UNIT_OF_WORK 表函数 - 获取详细的工作单元度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE

表 1294. 表函数监视信息 (续)

表函数	监视元素收集级别
MON_GET_WORKLOAD 表函数 - 获取工作负载度量值	REQUEST METRICS BASE
MON_GET_WORKLOAD_DETAILS 表函数 - 获取详细的工作负载度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE

表 1295. 事件监视信息

事件类型	逻辑数据分组	监视开关
活动	event_activity（在 details_xml 文档中报告） event_activitymetrics	ACTIVITY METRICS BASE
程序包高速缓存	pkgcache（在 metrics.xml 文档中报告） pkgcache_metrics	ACTIVITY METRICS BASE
工作单元	uow（在 metrics.xml 文档中报告） uow_metrics	REQUEST METRICS BASE
统计信息	event_scstats（在 details_xml 文档中报告）	REQUEST METRICS BASE
统计信息	event_wlstats（在 details_xml 文档中报告）	REQUEST METRICS BASE

用法

此监视元素与其他 **pool_queued_async_..._pages** 元素一起表示预取请求检索到的数据页数。可使用此信息来确定是否在系统上有效地执行了预取请求。例如，可使用类似如下的公式来计算每个预取请求的平均页数:

```
  (POOL_QUEUED_ASYNC_DATA_PAGES +
   POOL_QUEUED_ASYNC_INDEX_PAGES +
   POOL_QUEUED_ASYNC_XDA_PAGES +
   POOL_QUEUED_ASYNC_TEMP_DATA_PAGES +
   POOL_QUEUED_ASYNC_TEMP_INDEX_PAGES +
   POOL_QUEUED_ASYNC_TEMP_XDA_PAGES)
÷
  (POOL_QUEUED_ASYNC_DATA_REQS +
   POOL_QUEUED_ASYNC_INDEX_REQS +
   POOL_QUEUED_ASYNC_XDA_REQS +
   POOL_QUEUED_ASYNC_TEMP_DATA_REQS +
   POOL_QUEUED_ASYNC_TEMP_INDEX_REQS +
   POOL_QUEUED_ASYNC_TEMP_XDA_REQS)
```

如果每个请求的平均页数很低，并且系统上的预取量很大，那么系统要执行的 IO 操作可能比需要执行的 IO 操作多。通常，请求大小基于预取大小，后者至少应与扩展数据块大小相同。所以，平均请求大小较小可能指示预取大小设置得太低，将预取大小增加至扩展数据块大小的倍数可能会改进性能。而且应注意，平均请求大小较小可能意味着预取队列填充得太快，所以还应监视关联 **pool_failed_async_..._reqs** 监视元素

pool_queued_async_index_reqs -“索引预取请求数”监视元素

成功添加至预取队列的索引预取请求的数目。

表 1296. 表函数监视信息

表函数	监视元素收集级别
MON_FORMAT_XML_METRICS_BY_ROW - 获取所有度量值的基于已格式化行的输出	不适用; 报告 XML 文档中作为格式化函数的输入提供的所有元素
MON_GET_ACTIVITY_DETAILS 表函数 - 获取完整的活动详细信息（在 XML 文档 DETAILS 中报告）	ACTIVITY METRICS BASE
MON_GET_BUFFERPOOL 表函数 - 获取缓冲池度量值	DATA OBJECT METRICS BASE
MON_GET_CONNECTION 表函数 - 获取连接度量值	REQUEST METRICS BASE
MON_GET_CONNECTION_DETAILS 表函数 - 获取详细的连接度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE
MON_GET_PKG_CACHE_STMT 表函数 - 获取程序包高速缓存中的 SQL 语句活动度量值	ACTIVITY METRICS BASE
MON_GET_PKG_CACHE_STMT_DETAILS 表函数 - 获取程序包高速缓存条目的详细度量值（在 XML 文档 DETAILS 中报告）	ACTIVITY METRICS BASE
MON_GET_SERVICE_SUBCLASS 表函数 - 获取服务子类度量值	REQUEST METRICS BASE
MON_GET_SERVICE_SUBCLASS_DETAILS 表函数 - 获取详细的服务子类度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE
MON_GET_TABLESPACE 表函数 - 获取表空间度量值	DATA OBJECT METRICS BASE
MON_GET_UNIT_OF_WORK 表函数 - 获取工作单元度量值	REQUEST METRICS BASE
MON_GET_UNIT_OF_WORK 表函数 - 获取详细的工作单元度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE
MON_GET_WORKLOAD 表函数 - 获取工作负载度量值	REQUEST METRICS BASE
MON_GET_WORKLOAD_DETAILS 表函数 - 获取详细的工作负载度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE

表 1297. 事件监视信息

事件类型	逻辑数据分组	监视开关
活动	event_activity（在 details_xml 文档中报告） event_activitymetrics	ACTIVITY METRICS BASE

表 1297. 事件监视信息 (续)

事件类型	逻辑数据分组	监视开关
程序包高速缓存	pkgcache（在 metrics.xml 文档中报告） pkgcache_metrics	ACTIVITY METRICS BASE
工作单元	uow（在 metrics.xml 文档中报告） uow_metrics	REQUEST METRICS BASE
统计信息	event_scstats（在 details_xml 文档中报告）	REQUEST METRICS BASE
统计信息	event_wlstats（在 details_xml 文档中报告）	REQUEST METRICS BASE

用法

此元素与其他 **pool_queued_async_*_reqs** 元素一起表示已添加至预取队列的预取请求数。可使用此信息来查看数据库管理器执行预取的频率。可将这些元素与其他预取程序监视元素配合使用来确定在系统上执行的预取的效率。例如，可使用类似如下的公式来了解已成功添加至预取队列的请求的百分比:

```
1 -
 (
 POOL_FAILED_ASYNC_DATA_REQS +
 POOL_FAILED_ASYNC_INDEX_REQS +
 POOL_FAILED_ASYNC_XDA_REQS +
 POOL_FAILED_ASYNC_TEMP_DATA_REQS +
 POOL_FAILED_ASYNC_TEMP_INDEX_REQS +
 POOL_FAILED_ASYNC_TEMP_XDA_REQS
 )
 ÷
 (
   (
   POOL_FAILED_ASYNC_DATA_REQS +
   POOL_FAILED_ASYNC_INDEX_REQS +
   POOL_FAILED_ASYNC_XDA_REQS +
   POOL_FAILED_ASYNC_TEMP_DATA_REQS +
   POOL_FAILED_ASYNC_TEMP_INDEX_REQS +
   POOL_FAILED_ASYNC_TEMP_XDA_REQS
   )
 +
   (
   POOL_QUEUED_ASYNC_DATA_REQS +
   POOL_QUEUED_ASYNC_INDEX_REQS +
   POOL_QUEUED_ASYNC_XDA_REQS +
   POOL_QUEUED_ASYNC_TEMP_DATA_REQS +
   POOL_QUEUED_ASYNC_TEMP_INDEX_REQS +
   POOL_QUEUED_ASYNC_TEMP_XDA_REQS
   )
 ) * 100
```

此公式计算失败预取请求占所提出请求总数的百分比。失败预取请求是未能添加至预取队列的请求。如果该百分比很低，那么可能需要通过修改 **num_ioservers** 配置参数来配置更多预取程序。

pool_queued_async_other_reqs -“预取程序处理的其他请求数”监视元素

成功添加至预取队列的非预取工作请求数。这表示预取程序完成的其他工作。

表 1298. 表函数监视信息

表函数	监视元素收集级别
MON_FORMAT_XML_METRICS_BY_ROW - 获取所有度量值的基于已格式化行的输出	不适用; 报告 XML 文档中作为格式化函数的输入提供的所有元素
MON_GET_ACTIVITY_DETAILS 表函数 - 获取完整的活动详细信息（在 XML 文档 DETAILS 中报告）	ACTIVITY METRICS BASE
MON_GET_BUFFERPOOL 表函数 - 获取缓冲池度量值	DATA OBJECT METRICS BASE
MON_GET_CONNECTION 表函数 - 获取连接度量值	REQUEST METRICS BASE
MON_GET_CONNECTION_DETAILS 表函数 - 获取详细的连接度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE
MON_GET_PKG_CACHE_STMT 表函数 - 获取程序包高速缓存中的 SQL 语句活动度量值	ACTIVITY METRICS BASE
MON_GET_PKG_CACHE_STMT_DETAILS 表函数 - 获取程序包高速缓存条目的详细度量值（在 XML 文档 DETAILS 中报告）	ACTIVITY METRICS BASE
MON_GET_SERVICE_SUBCLASS 表函数 - 获取服务子类度量值	REQUEST METRICS BASE
MON_GET_SERVICE_SUBCLASS_DETAILS 表函数 - 获取详细的服务子类度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE
MON_GET_TABLESPACE 表函数 - 获取表空间度量值	DATA OBJECT METRICS BASE
MON_GET_UNIT_OF_WORK 表函数 - 获取工作单元度量值	REQUEST METRICS BASE
MON_GET_UNIT_OF_WORK 表函数 - 获取详细的工作单元度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE
MON_GET_WORKLOAD 表函数 - 获取工作负载度量值	REQUEST METRICS BASE
MON_GET_WORKLOAD_DETAILS 表函数 - 获取详细的工作负载度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE

表 1299. 事件监视信息

事件类型	逻辑数据分组	监视开关
活动	event_activity（在 details_xml 文档中报告） event_activitymetrics	ACTIVITY METRICS BASE

表 1299. 事件监视信息 (续)

事件类型	逻辑数据分组	监视开关
程序包高速缓存	pkgcache（在 metrics.xml 文档中报告） pkgcache_metrics	ACTIVITY METRICS BASE
工作单元	uow（在 metrics.xml 文档中报告） uow_metrics	REQUEST METRICS BASE
统计信息	event_scstats（在 details_xml 文档中报告）	REQUEST METRICS BASE
统计信息	event_wlstats（在 details_xml 文档中报告）	REQUEST METRICS BASE

用法

此监视元素报告添加至预取队列的请求数，这些请求针对与存取方案决定的预取无关的 IO 工作。备份实用程序之类的实用程序使用预取程序机制来执行它们的任务，但以不同于存取方案针对 SQL 语句执行的方式执行。

pool_queued_async_temp_data_pages -“预取请求的临时表空间数据页数”监视元素

成功请求预取的临时表空间数据页的数目。

表 1300. 表函数监视信息

表函数	监视元素收集级别
MON_FORMAT_XML_METRICS_BY_ROW - 获取所有度量值的基于已格式化行的输出	不适用; 报告 XML 文档中作为格式化函数的输入提供的所有元素
MON_GET_ACTIVITY_DETAILS 表函数 - 获取完整的活动详细信息（在 XML 文档 DETAILS 中报告）	ACTIVITY METRICS BASE
MON_GET_BUFFERPOOL 表函数 - 获取缓冲池度量值	DATA OBJECT METRICS BASE
MON_GET_CONNECTION 表函数 - 获取连接度量值	REQUEST METRICS BASE
MON_GET_CONNECTION_DETAILS 表函数 - 获取详细的连接度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE
MON_GET_PKG_CACHE_STMT 表函数 - 获取程序包高速缓存中的 SQL 语句活动度量值	ACTIVITY METRICS BASE
MON_GET_PKG_CACHE_STMT_DETAILS 表函数 - 获取程序包高速缓存条目的详细度量值（在 XML 文档 DETAILS 中报告）	ACTIVITY METRICS BASE
MON_GET_SERVICE_SUBCLASS 表函数 - 获取服务子类度量值	REQUEST METRICS BASE

表 1300. 表函数监视信息 (续)

表函数	监视元素收集级别
MON_GET_SERVICE_SUBCLASS_DETAILS 表函数 - 获取详细的服务子类度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE
MON_GET_TABLESPACE 表函数 - 获取表空间度量值	DATA OBJECT METRICS BASE
MON_GET_UNIT_OF_WORK 表函数 - 获取工作单元度量值	REQUEST METRICS BASE
MON_GET_UNIT_OF_WORK 表函数 - 获取详细的工作单元度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE
MON_GET_WORKLOAD 表函数 - 获取工作负载度量值	REQUEST METRICS BASE
MON_GET_WORKLOAD_DETAILS 表函数 - 获取详细的工作负载度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE

表 1301. 事件监视信息

事件类型	逻辑数据分组	监视开关
活动	event_activity（在 details_xml 文档中报告） event_activitymetrics	ACTIVITY METRICS BASE
程序包高速缓存	pkgcache（在 metrics.xml 文档中报告） pkgcache_metrics	ACTIVITY METRICS BASE
工作单元	uow（在 metrics.xml 文档中报告） uow_metrics	REQUEST METRICS BASE
统计信息	event_scstats（在 details_xml 文档中报告）	REQUEST METRICS BASE
统计信息	event_wlstats（在 details_xml 文档中报告）	REQUEST METRICS BASE

用法

此监视元素与其他 **pool_queued_async_..._pages** 元素一起表示预取请求检索到的数据页数。可使用此信息来确定是否在系统上有效地执行了预取请求。例如，可使用类似如下的公式来计算每个预取请求的平均页数:

```
  (POOL_QUEUED_ASYNC_DATA_PAGES +
   POOL_QUEUED_ASYNC_INDEX_PAGES +
   POOL_QUEUED_ASYNC_XDA_PAGES +
   POOL_QUEUED_ASYNC_TEMP_DATA_PAGES +
   POOL_QUEUED_ASYNC_TEMP_INDEX_PAGES +
   POOL_QUEUED_ASYNC_TEMP_XDA_PAGES)
÷
  (POOL_QUEUED_ASYNC_DATA_REQS +
   POOL_QUEUED_ASYNC_INDEX_REQS +
```

```
POOL_QUEUED_ASYNC_XDA_REQS +
POOL_QUEUED_ASYNC_TEMP_DATA_REQS +
POOL_QUEUED_ASYNC_TEMP_INDEX_REQS +
POOL_QUEUED_ASYNC_TEMP_XDA_REQS)
```

如果每个请求的平均页数很低，并且系统上的预取量很大，那么系统要执行的 IO 操作可能比需要执行的 IO 操作多。通常，请求大小基于预取大小，后者至少应与扩展数据块大小相同。所以，平均请求大小较小可能指示预取大小设置得太低，将预取大小增加至扩展数据块大小的倍数可能会改进性能。而且应注意，平均请求大小较小可能意味着预取队列填充得太快，所以还应监视关联 **pool_failed_async_..._reqs** 监视元素

pool_queued_async_temp_data_reqs -“临时表空间数据预取请求数”监视元素

成功添加至预取队列的临时表空间数据预取请求的数目。

表 1302. 表函数监视信息

表函数	监视元素收集级别
MON_FORMAT_XML_METRICS_BY_ROW - 获取所有度量值的基于已格式化行的输出	不适用; 报告 XML 文档中作为格式化函数的输入提供的所有元素
MON_GET_ACTIVITY_DETAILS 表函数 - 获取完整的活动详细信息（在 XML 文档 DETAILS 中报告）	ACTIVITY METRICS BASE
MON_GET_BUFFERPOOL 表函数 - 获取缓冲池度量值	DATA OBJECT METRICS BASE
MON_GET_CONNECTION 表函数 - 获取连接度量值	REQUEST METRICS BASE
MON_GET_CONNECTION_DETAILS 表函数 - 获取详细的连接度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE
MON_GET_PKG_CACHE_STMT 表函数 - 获取程序包高速缓存中的 SQL 语句活动度量值	ACTIVITY METRICS BASE
MON_GET_PKG_CACHE_STMT_DETAILS 表函数 - 获取程序包高速缓存条目的详细度量值（在 XML 文档 DETAILS 中报告）	ACTIVITY METRICS BASE
MON_GET_SERVICE_SUBCLASS 表函数 - 获取服务子类度量值	REQUEST METRICS BASE
MON_GET_SERVICE_SUBCLASS_DETAILS 表函数 - 获取详细的服务子类度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE
MON_GET_TABLESPACE 表函数 - 获取表空间度量值	DATA OBJECT METRICS BASE
MON_GET_UNIT_OF_WORK 表函数 - 获取工作单元度量值	REQUEST METRICS BASE
MON_GET_UNIT_OF_WORK 表函数 - 获取详细的工作单元度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE

表 1302. 表函数监视信息 (续)

表函数	监视元素收集级别
MON_GET_WORKLOAD 表函数 - 获取工作负载度量值	REQUEST METRICS BASE
MON_GET_WORKLOAD_DETAILS 表函数 - 获取详细的工作负载度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE

表 1303. 事件监视信息

事件类型	逻辑数据分组	监视开关
活动	event_activity（在 details_xml 文档中报告） event_activitymetrics	ACTIVITY METRICS BASE
程序包高速缓存	pkgcache（在 metrics.xml 文档中报告） pkgcache_metrics	ACTIVITY METRICS BASE
工作单元	uow（在 metrics.xml 文档中报告） uow_metrics	REQUEST METRICS BASE
统计信息	event_scstats（在 details_xml 文档中报告）	REQUEST METRICS BASE
统计信息	event_wlstats（在 details_xml 文档中报告）	REQUEST METRICS BASE

用法

此元素与其他 **pool_queued_async_*_reqs** 元素一起表示已添加至预取队列的预取请求数。可使用此信息来查看数据库管理器执行预取的频率。可将这些元素与其他预取程序监视元素配合使用来确定在系统上执行的预取的效率。例如，可使用类似如下的公式来了解已成功添加至预取队列的请求的百分比:

```
1 -
 (
 POOL_FAILED_ASYNC_DATA_REQS +
 POOL_FAILED_ASYNC_INDEX_REQS +
 POOL_FAILED_ASYNC_XDA_REQS +
 POOL_FAILED_ASYNC_TEMP_DATA_REQS +
 POOL_FAILED_ASYNC_TEMP_INDEX_REQS +
 POOL_FAILED_ASYNC_TEMP_XDA_REQS
 )
 ÷
 (
   (
   POOL_FAILED_ASYNC_DATA_REQS +
   POOL_FAILED_ASYNC_INDEX_REQS +
   POOL_FAILED_ASYNC_XDA_REQS +
   POOL_FAILED_ASYNC_TEMP_DATA_REQS +
   POOL_FAILED_ASYNC_TEMP_INDEX_REQS +
   POOL_FAILED_ASYNC_TEMP_XDA_REQS
   )
 +
   (
   POOL_QUEUED_ASYNC_DATA_REQS +
   POOL_QUEUED_ASYNC_INDEX_REQS +
```

```
    POOL_QUEUED_ASYNC_XDA_REQS +
    POOL_QUEUED_ASYNC_TEMP_DATA_REQS +
    POOL_QUEUED_ASYNC_TEMP_INDEX_REQS +
    POOL_QUEUED_ASYNC_TEMP_XDA_REQS
    )
  ) * 100
```

此公式计算失败预取请求占所提出请求总数的百分比。失败预取请求是未能添加至预取队列的请求。如果该百分比很低，那么可能需要通过修改 **num_ioservers** 配置参数来配置更多预取程序。

pool_queued_async_temp_index_pages -“预取请求的临时表空间索引页数”监视元素

成功请求预取的临时表空间索引页数。

表 1304. 表函数监视信息

表函数	监视元素收集级别
MON_FORMAT_XML_METRICS_BY_ROW - 获取所有度量值的基于已格式化行的输出	不适用; 报告 XML 文档中作为格式化函数的输入提供的所有元素
MON_GET_ACTIVITY_DETAILS 表函数 - 获取完整的活动详细信息（在 XML 文档 DETAILS 中报告）	ACTIVITY METRICS BASE
MON_GET_BUFFERPOOL 表函数 - 获取缓冲池度量值	DATA OBJECT METRICS BASE
MON_GET_CONNECTION 表函数 - 获取连接度量值	REQUEST METRICS BASE
MON_GET_CONNECTION_DETAILS 表函数 - 获取详细的连接度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE
MON_GET_PKG_CACHE_STMT 表函数 - 获取程序包高速缓存中的 SQL 语句活动度量值	ACTIVITY METRICS BASE
MON_GET_PKG_CACHE_STMT_DETAILS 表函数 - 获取程序包高速缓存条目的详细度量值（在 XML 文档 DETAILS 中报告）	ACTIVITY METRICS BASE
MON_GET_SERVICE_SUBCLASS 表函数 - 获取服务子类度量值	REQUEST METRICS BASE
MON_GET_SERVICE_SUBCLASS_DETAILS 表函数 - 获取详细的服务子类度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE
MON_GET_TABLESPACE 表函数 - 获取表空间度量值	DATA OBJECT METRICS BASE
MON_GET_UNIT_OF_WORK 表函数 - 获取工作单元度量值	REQUEST METRICS BASE
MON_GET_UNIT_OF_WORK 表函数 - 获取详细的工作单元度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE
MON_GET_WORKLOAD 表函数 - 获取工作负载度量值	REQUEST METRICS BASE

表 1304. 表函数监视信息 (续)

表函数	监视元素收集级别
MON_GET_WORKLOAD_DETAILS 表函数 - 获取详细的工作负载度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE

表 1305. 事件监视信息

事件类型	逻辑数据分组	监视开关
活动	event_activity（在 details_xml 文档中报告） event_activitymetrics	ACTIVITY METRICS BASE
程序包高速缓存	pkgcache（在 metrics.xml 文档中报告） pkgcache_metrics	ACTIVITY METRICS BASE
工作单元	uow（在 metrics.xml 文档中报告） uow_metrics	REQUEST METRICS BASE
统计信息	event_scstats（在 details_xml 文档中报告）	REQUEST METRICS BASE
统计信息	event_wlstats（在 details_xml 文档中报告）	REQUEST METRICS BASE

用法

此监视元素与其他 **pool_queued_async_..._pages** 元素一起表示预取请求检索到的数据页数。可使用此信息来确定是否在系统上有效地执行了预取请求。例如，可使用类似如下的公式来计算每个预取请求的平均页数:

```
  (POOL_QUEUED_ASYNC_DATA_PAGES +
   POOL_QUEUED_ASYNC_INDEX_PAGES +
   POOL_QUEUED_ASYNC_XDA_PAGES +
   POOL_QUEUED_ASYNC_TEMP_DATA_PAGES +
   POOL_QUEUED_ASYNC_TEMP_INDEX_PAGES +
   POOL_QUEUED_ASYNC_TEMP_XDA_PAGES)
÷
  (POOL_QUEUED_ASYNC_DATA_REQS +
   POOL_QUEUED_ASYNC_INDEX_REQS +
   POOL_QUEUED_ASYNC_XDA_REQS +
   POOL_QUEUED_ASYNC_TEMP_DATA_REQS +
   POOL_QUEUED_ASYNC_TEMP_INDEX_REQS +
   POOL_QUEUED_ASYNC_TEMP_XDA_REQS)
```

如果每个请求的平均页数很低，并且系统上的预取量很大，那么系统要执行的 IO 操作可能比需要执行的 IO 操作多。通常，请求大小基于预取大小，后者至少应与扩展数据块大小相同。所以，平均请求大小较小可能指示预取大小设置得太低，将预取大小增加至扩展数据块大小的倍数可能会改进性能。而且应注意，平均请求大小较小可能意味着预取队列填充得太快，所以还应监视关联 **pool_failed_async_..._reqs** 监视元素

pool_queued_async_temp_index_reqs -“临时表空间索引预取请求数”监视元素

成功添加至预取队列的临时表空间索引预取请求的数目。

表 1306. 表函数监视信息

表函数	监视元素收集级别
MON_FORMAT_XML_METRICS_BY_ROW - 获取所有度量值的基于已格式化行的输出	不适用; 报告 XML 文档中作为格式化函数的输入提供的所有元素
MON_GET_ACTIVITY_DETAILS 表函数 - 获取完整的活动详细信息（在 XML 文档 DETAILS 中报告）	ACTIVITY METRICS BASE
MON_GET_BUFFERPOOL 表函数 - 获取缓冲池度量值	DATA OBJECT METRICS BASE
MON_GET_CONNECTION 表函数 - 获取连接度量值	REQUEST METRICS BASE
MON_GET_CONNECTION_DETAILS 表函数 - 获取详细的连接度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE
MON_GET_PKG_CACHE_STMT 表函数 - 获取程序包高速缓存中的 SQL 语句活动度量值	ACTIVITY METRICS BASE
MON_GET_PKG_CACHE_STMT_DETAILS 表函数 - 获取程序包高速缓存条目的详细度量值（在 XML 文档 DETAILS 中报告）	ACTIVITY METRICS BASE
MON_GET_SERVICE_SUBCLASS 表函数 - 获取服务子类度量值	REQUEST METRICS BASE
MON_GET_SERVICE_SUBCLASS_DETAILS 表函数 - 获取详细的服务子类度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE
MON_GET_TABLESPACE 表函数 - 获取表空间度量值	DATA OBJECT METRICS BASE
MON_GET_UNIT_OF_WORK 表函数 - 获取工作单元度量值	REQUEST METRICS BASE
MON_GET_UNIT_OF_WORK 表函数 - 获取详细的工作单元度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE
MON_GET_WORKLOAD 表函数 - 获取工作负载度量值	REQUEST METRICS BASE
MON_GET_WORKLOAD_DETAILS 表函数 - 获取详细的工作负载度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE

表 1307. 事件监视信息

事件类型	逻辑数据分组	监视开关
活动	event_activity（在 details_xml 文档中报告） event_activitymetrics	ACTIVITY METRICS BASE

表 1307. 事件监视信息 (续)

事件类型	逻辑数据分组	监视开关
程序包高速缓存	pkgcache（在 metrics.xml 文档中报告） pkgcache_metrics	ACTIVITY METRICS BASE
工作单元	uow（在 metrics.xml 文档中报告） uow_metrics	REQUEST METRICS BASE
统计信息	event_scstats（在 details_xml 文档中报告）	REQUEST METRICS BASE
统计信息	event_wlstats（在 details_xml 文档中报告）	REQUEST METRICS BASE

用法

此元素与其他 **pool_queued_async_*_reqs** 元素一起表示已添加至预取队列的预取请求数。可使用此信息来查看数据库管理器执行预取的频率。可将这些元素与其他预取程序监视元素配合使用来确定在系统上执行的预取的效率。例如，可使用类似如下的公式来了解已成功添加至预取队列的请求的百分比:

```
1 -
 (
 POOL_FAILED_ASYNC_DATA_REQS +
 POOL_FAILED_ASYNC_INDEX_REQS +
 POOL_FAILED_ASYNC_XDA_REQS +
 POOL_FAILED_ASYNC_TEMP_DATA_REQS +
 POOL_FAILED_ASYNC_TEMP_INDEX_REQS +
 POOL_FAILED_ASYNC_TEMP_XDA_REQS
 )
 ÷
 (
   (
   POOL_FAILED_ASYNC_DATA_REQS +
   POOL_FAILED_ASYNC_INDEX_REQS +
   POOL_FAILED_ASYNC_XDA_REQS +
   POOL_FAILED_ASYNC_TEMP_DATA_REQS +
   POOL_FAILED_ASYNC_TEMP_INDEX_REQS +
   POOL_FAILED_ASYNC_TEMP_XDA_REQS
   )
 +
   (
   POOL_QUEUED_ASYNC_DATA_REQS +
   POOL_QUEUED_ASYNC_INDEX_REQS +
   POOL_QUEUED_ASYNC_XDA_REQS +
   POOL_QUEUED_ASYNC_TEMP_DATA_REQS +
   POOL_QUEUED_ASYNC_TEMP_INDEX_REQS +
   POOL_QUEUED_ASYNC_TEMP_XDA_REQS
   )
 ) * 100
```

此公式计算失败预取请求占所提出请求总数的百分比。失败预取请求是未能添加至预取队列的请求。如果该百分比很低，那么可能需要通过修改 **num_ioservers** 配置参数来配置更多预取程序。

pool_queued_async_temp_xda_pages -“预取请求的临时表空间 XDA 数据页数”监视元素

成功请求预取的临时表空间 XML 存储器对象 (XDA) 数据页的数目。

表 1308. 表函数监视信息

表函数	监视元素收集级别
MON_FORMAT_XML_METRICS_BY_ROW - 获取所有度量值的基于已格式化行的输出	不适用; 报告 XML 文档中作为格式化函数的输入提供的所有元素
MON_GET_ACTIVITY_DETAILS 表函数 - 获取完整的活动详细信息（在 XML 文档 DETAILS 中报告）	ACTIVITY METRICS BASE
MON_GET_BUFFERPOOL 表函数 - 获取缓冲池度量值	DATA OBJECT METRICS BASE
MON_GET_CONNECTION 表函数 - 获取连接度量值	REQUEST METRICS BASE
MON_GET_CONNECTION_DETAILS 表函数 - 获取详细的连接度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE
MON_GET_PKG_CACHE_STMT 表函数 - 获取程序包高速缓存中的 SQL 语句活动度量值	ACTIVITY METRICS BASE
MON_GET_PKG_CACHE_STMT_DETAILS 表函数 - 获取程序包高速缓存条目的详细度量值（在 XML 文档 DETAILS 中报告）	ACTIVITY METRICS BASE
MON_GET_SERVICE_SUBCLASS 表函数 - 获取服务子类度量值	REQUEST METRICS BASE
MON_GET_SERVICE_SUBCLASS_DETAILS 表函数 - 获取详细的服务子类度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE
MON_GET_TABLESPACE 表函数 - 获取表空间度量值	DATA OBJECT METRICS BASE
MON_GET_UNIT_OF_WORK 表函数 - 获取工作单元度量值	REQUEST METRICS BASE
MON_GET_UNIT_OF_WORK 表函数 - 获取详细的工作单元度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE
MON_GET_WORKLOAD 表函数 - 获取工作负载度量值	REQUEST METRICS BASE
MON_GET_WORKLOAD_DETAILS 表函数 - 获取详细的工作负载度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE

表 1309. 事件监视信息

事件类型	逻辑数据分组	监视开关
活动	event_activity（在 details_xml 文档中报告） event_activitymetrics	ACTIVITY METRICS BASE

表 1309. 事件监视信息 (续)

事件类型	逻辑数据分组	监视开关
程序包高速缓存	pkgcache（在 metrics.xml 文档中报告） pkgcache_metrics	ACTIVITY METRICS BASE
工作单元	uow（在 metrics.xml 文档中报告） uow_metrics	REQUEST METRICS BASE
统计信息	event_scstats（在 details_xml 文档中报告）	REQUEST METRICS BASE
统计信息	event_wlstats（在 details_xml 文档中报告）	REQUEST METRICS BASE

用法

此监视元素与其他 **pool_queued_async_..._pages** 元素一起表示预取请求检索到的数据页数。可使用此信息来确定是否在系统上有效地执行了预取请求。例如，可使用类似如下的公式来计算每个预取请求的平均页数:

```
  (POOL_QUEUED_ASYNC_DATA_PAGES +
   POOL_QUEUED_ASYNC_INDEX_PAGES +
   POOL_QUEUED_ASYNC_XDA_PAGES +
   POOL_QUEUED_ASYNC_TEMP_DATA_PAGES +
   POOL_QUEUED_ASYNC_TEMP_INDEX_PAGES +
   POOL_QUEUED_ASYNC_TEMP_XDA_PAGES)
÷
  (POOL_QUEUED_ASYNC_DATA_REQS +
   POOL_QUEUED_ASYNC_INDEX_REQS +
   POOL_QUEUED_ASYNC_XDA_REQS +
   POOL_QUEUED_ASYNC_TEMP_DATA_REQS +
   POOL_QUEUED_ASYNC_TEMP_INDEX_REQS +
   POOL_QUEUED_ASYNC_TEMP_XDA_REQS)
```

如果每个请求的平均页数很低，并且系统上的预取量很大，那么系统要执行的 IO 操作可能比需要执行的 IO 操作多。通常，请求大小基于预取大小，后者至少应与扩展数据块大小相同。所以，平均请求大小较小可能指示预取大小设置得太低，将预取大小增加至扩展数据块大小的倍数可能会改进性能。而且应注意，平均请求大小较小可能意味着预取队列填充得太快，所以还应监视关联 **pool_failed_async_..._reqs** 监视元素

pool_queued_async_temp_xda_reqs -“临时表空间 XDA 数据预取请求数”监视元素

成功添加至预取队列的临时表空间 XML 存储器对象 (XDA) 数据预取请求的数目。

表 1310. 表函数监视信息

表函数	监视元素收集级别
MON_FORMAT_XML_METRICS_BY_ROW - 获取所有度量值的基于已格式化行的输出	不适用；报告 XML 文档中作为格式化函数的输入提供的所有元素
MON_GET_ACTIVITY_DETAILS 表函数 - 获取完整的活动详细信息（在 XML 文档 DETAILS 中报告）	ACTIVITY METRICS BASE

表 1310. 表函数监视信息 (续)

表函数	监视元素收集级别
MON_GET_BUFFERPOOL 表函数 - 获取缓冲池度量值	DATA OBJECT METRICS BASE
MON_GET_CONNECTION 表函数 - 获取连接度量值	REQUEST METRICS BASE
MON_GET_CONNECTION_DETAILS 表函数 - 获取详细的连接度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE
MON_GET_PKG_CACHE_STMT 表函数 - 获取程序包高速缓存中的 SQL 语句活动度量值	ACTIVITY METRICS BASE
MON_GET_PKG_CACHE_STMT_DETAILS 表函数 - 获取程序包高速缓存条目的详细度量值（在 XML 文档 DETAILS 中报告）	ACTIVITY METRICS BASE
MON_GET_SERVICE_SUBCLASS 表函数 - 获取服务子类度量值	REQUEST METRICS BASE
MON_GET_SERVICE_SUBCLASS_DETAILS 表函数 - 获取详细的服务子类度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE
MON_GET_TABLESPACE 表函数 - 获取表空间度量值	DATA OBJECT METRICS BASE
MON_GET_UNIT_OF_WORK 表函数 - 获取工作单元度量值	REQUEST METRICS BASE
MON_GET_UNIT_OF_WORK 表函数 - 获取详细的工作单元度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE
MON_GET_WORKLOAD 表函数 - 获取工作负载度量值	REQUEST METRICS BASE
MON_GET_WORKLOAD_DETAILS 表函数 - 获取详细的工作负载度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE

表 1311. 事件监视信息

事件类型	逻辑数据分组	监视开关
活动	event_activity（在 details_xml 文档中报告） event_activitymetrics	ACTIVITY METRICS BASE
程序包高速缓存	pkgcache（在 metrics.xml 文档中报告） pkgcache_metrics	ACTIVITY METRICS BASE
工作单元	uow（在 metrics.xml 文档中报告） uow_metrics	REQUEST METRICS BASE
统计信息	event_scstats（在 details_xml 文档中报告）	REQUEST METRICS BASE

表 1311. 事件监视信息 (续)

事件类型	逻辑数据分组	监视开关
统计信息	event_wlstats（在 details_xml 文档中报告）	REQUEST METRICS BASE

用法

此元素与其他 **pool_queued_async_*_reqs** 元素一起表示已添加至预取队列的预取请求数。可使用此信息来查看数据库管理器执行预取的频率。可将这些元素与其他预取程序监视元素配合使用来确定在系统上执行的预取的效率。例如，可使用类似如下的公式来了解已成功添加至预取队列的请求的百分比:

```
1 -
 (
 POOL_FAILED_ASYNC_DATA_REQS +
 POOL_FAILED_ASYNC_INDEX_REQS +
 POOL_FAILED_ASYNC_XDA_REQS +
 POOL_FAILED_ASYNC_TEMP_DATA_REQS +
 POOL_FAILED_ASYNC_TEMP_INDEX_REQS +
 POOL_FAILED_ASYNC_TEMP_XDA_REQS
 )
 ÷
 (
   (
   POOL_FAILED_ASYNC_DATA_REQS +
   POOL_FAILED_ASYNC_INDEX_REQS +
   POOL_FAILED_ASYNC_XDA_REQS +
   POOL_FAILED_ASYNC_TEMP_DATA_REQS +
   POOL_FAILED_ASYNC_TEMP_INDEX_REQS +
   POOL_FAILED_ASYNC_TEMP_XDA_REQS
   )
 +
   (
   POOL_QUEUED_ASYNC_DATA_REQS +
   POOL_QUEUED_ASYNC_INDEX_REQS +
   POOL_QUEUED_ASYNC_XDA_REQS +
   POOL_QUEUED_ASYNC_TEMP_DATA_REQS +
   POOL_QUEUED_ASYNC_TEMP_INDEX_REQS +
   POOL_QUEUED_ASYNC_TEMP_XDA_REQS
   )
 ) * 100
```

此公式计算失败预取请求占所提出请求总数的百分比。失败预取请求是未能添加至预取队列的请求。如果该百分比很低，那么可能需要通过修改 **num_ioservers** 配置参数来配置更多预取程序。

pool_queued_async_xda_pages -“预取请求的 XDA 页数”监视元素

成功请求预取的 XML 存储器对象 (XDA) 数据页的数目。

表 1312. 表函数监视信息

表函数	监视元素收集级别
MON_FORMAT_XML_METRICS_BY_ROW - 获取所有度量值的基于已格式化行的输出	不适用；报告 XML 文档中作为格式化函数的输入提供的所有元素
MON_GET_ACTIVITY_DETAILS 表函数 - 获取完整的活动详细信息（在 XML 文档 DETAILS 中报告）	ACTIVITY METRICS BASE

表 1312. 表函数监视信息 (续)

表函数	监视元素收集级别
MON_GET_BUFFERPOOL 表函数 - 获取缓冲池度量值	DATA OBJECT METRICS BASE
MON_GET_CONNECTION 表函数 - 获取连接度量值	REQUEST METRICS BASE
MON_GET_CONNECTION_DETAILS 表函数 - 获取详细的连接度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE
MON_GET_PKG_CACHE_STMT 表函数 - 获取程序包高速缓存中的 SQL 语句活动度量值	ACTIVITY METRICS BASE
MON_GET_PKG_CACHE_STMT_DETAILS 表函数 - 获取程序包高速缓存条目的详细度量值（在 XML 文档 DETAILS 中报告）	ACTIVITY METRICS BASE
MON_GET_SERVICE_SUBCLASS 表函数 - 获取服务子类度量值	REQUEST METRICS BASE
MON_GET_SERVICE_SUBCLASS_DETAILS 表函数 - 获取详细的服务子类度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE
MON_GET_TABLESPACE 表函数 - 获取表空间度量值	DATA OBJECT METRICS BASE
MON_GET_UNIT_OF_WORK 表函数 - 获取工作单元度量值	REQUEST METRICS BASE
MON_GET_UNIT_OF_WORK 表函数 - 获取详细的工作单元度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE
MON_GET_WORKLOAD 表函数 - 获取工作负载度量值	REQUEST METRICS BASE
MON_GET_WORKLOAD_DETAILS 表函数 - 获取详细的工作负载度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE

表 1313. 事件监视信息

事件类型	逻辑数据分组	监视开关
活动	event_activity（在 details_xml 文档中报告） event_activitymetrics	ACTIVITY METRICS BASE
程序包高速缓存	pkgcache（在 metrics.xml 文档中报告） pkgcache_metrics	ACTIVITY METRICS BASE
工作单元	uow（在 metrics.xml 文档中报告） uow_metrics	REQUEST METRICS BASE
统计信息	event_scstats（在 details_xml 文档中报告）	REQUEST METRICS BASE

表 1313. 事件监视信息 (续)

事件类型	逻辑数据分组	监视开关
统计信息	event_wlstats（在 details_xml 文档中报告）	REQUEST METRICS BASE

用法

此监视元素与其他 **pool_queued_async_..._pages** 元素一起表示预取请求检索到的数据页数。可使用此信息来确定是否在系统上有效地执行了预取请求。例如，可使用类似如下的公式来计算每个预取请求的平均页数:

```
  (POOL_QUEUED_ASYNC_DATA_PAGES +
   POOL_QUEUED_ASYNC_INDEX_PAGES +
   POOL_QUEUED_ASYNC_XDA_PAGES +
   POOL_QUEUED_ASYNC_TEMP_DATA_PAGES +
   POOL_QUEUED_ASYNC_TEMP_INDEX_PAGES +
   POOL_QUEUED_ASYNC_TEMP_XDA_PAGES)
÷
  (POOL_QUEUED_ASYNC_DATA_REQS +
   POOL_QUEUED_ASYNC_INDEX_REQS +
   POOL_QUEUED_ASYNC_XDA_REQS +
   POOL_QUEUED_ASYNC_TEMP_DATA_REQS +
   POOL_QUEUED_ASYNC_TEMP_INDEX_REQS +
   POOL_QUEUED_ASYNC_TEMP_XDA_REQS)
```

如果每个请求的平均页数很低，并且系统上的预取量很大，那么系统要执行的 IO 操作可能比需要执行的 IO 操作多。通常，请求大小基于预取大小，后者至少应与扩展数据块大小相同。所以，平均请求大小较小可能指示预取大小设置得太低，将预取大小增加至扩展数据块大小的倍数可能会改进性能。而且应注意，平均请求大小较小可能意味着预取队列填充得太快，所以还应监视关联 **pool_failed_async_..._reqs** 监视元素

pool_queued_async_xda_reqs -“XDA 预取请求数”监视元素

成功添加至预取队列的 XML 存储器对象 (XDA) 数据预取请求的数目。

表 1314. 表函数监视信息

表函数	监视元素收集级别
MON_FORMAT_XML_METRICS_BY_ROW - 获取所有度量值的基于已格式化行的输出	不适用；报告 XML 文档中作为格式化函数的输入提供的所有元素
MON_GET_ACTIVITY_DETAILS 表函数 - 获取完整的活动详细信息（在 XML 文档 DETAILS 中报告）	ACTIVITY METRICS BASE
MON_GET_BUFFERPOOL 表函数 - 获取缓冲池度量值	DATA OBJECT METRICS BASE
MON_GET_CONNECTION 表函数 - 获取连接度量值	REQUEST METRICS BASE
MON_GET_CONNECTION_DETAILS 表函数 - 获取详细的连接度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE
MON_GET_PKG_CACHE_STMT 表函数 - 获取程序包高速缓存中的 SQL 语句活动度量值	ACTIVITY METRICS BASE

表 1314. 表函数监视信息 (续)

表函数	监视元素收集级别
MON_GET_PKG_CACHE_STMT_DETAILS 表函数 - 获取程序包高速缓存条目的详细度量值（在 XML 文档 DETAILS 中报告）	ACTIVITY METRICS BASE
MON_GET_SERVICE_SUBCLASS 表函数 - 获取服务子类度量值	REQUEST METRICS BASE
MON_GET_SERVICE_SUBCLASS_DETAILS 表函数 - 获取详细的服务子类度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE
MON_GET_TABLESPACE 表函数 - 获取表空间度量值	DATA OBJECT METRICS BASE
MON_GET_UNIT_OF_WORK 表函数 - 获取工作单元度量值	REQUEST METRICS BASE
MON_GET_UNIT_OF_WORK 表函数 - 获取详细的工作单元度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE
MON_GET_WORKLOAD 表函数 - 获取工作负载度量值	REQUEST METRICS BASE
MON_GET_WORKLOAD_DETAILS 表函数 - 获取详细的工作负载度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE

表 1315. 事件监视信息

事件类型	逻辑数据分组	监视开关
活动	event_activity（在 details_xml 文档中报告） event_activitymetrics	ACTIVITY METRICS BASE
程序包高速缓存	pkgcache（在 metrics.xml 文档中报告） pkgcache_metrics	ACTIVITY METRICS BASE
工作单元	uow（在 metrics.xml 文档中报告） uow_metrics	REQUEST METRICS BASE
统计信息	event_scstats（在 details_xml 文档中报告）	REQUEST METRICS BASE
统计信息	event_wlstats（在 details_xml 文档中报告）	REQUEST METRICS BASE

用法

此元素与其他 **pool_queued_async_*_reqs** 元素一起表示已添加至预取队列的预取请求数。可使用此信息来查看数据库管理器执行预取的频率。可将这些元素与其他预取程序监视元素配合使用来确定在系统上执行的预取的效率。例如，可使用类似如下的公式来了解已成功添加至预取队列的请求的百分比:

```
1 -
 (
 POOL_FAILED_ASYNC_DATA_REQS +
 POOL_FAILED_ASYNC_INDEX_REQS +
 POOL_FAILED_ASYNC_XDA_REQS +
 POOL_FAILED_ASYNC_TEMP_DATA_REQS +
 POOL_FAILED_ASYNC_TEMP_INDEX_REQS +
 POOL_FAILED_ASYNC_TEMP_XDA_REQS
 )
 ÷
 (
   (
   POOL_FAILED_ASYNC_DATA_REQS +
   POOL_FAILED_ASYNC_INDEX_REQS +
   POOL_FAILED_ASYNC_XDA_REQS +
   POOL_FAILED_ASYNC_TEMP_DATA_REQS +
   POOL_FAILED_ASYNC_TEMP_INDEX_REQS +
   POOL_FAILED_ASYNC_TEMP_XDA_REQS
   )
 +
   (
   POOL_QUEUED_ASYNC_DATA_REQS +
   POOL_QUEUED_ASYNC_INDEX_REQS +
   POOL_QUEUED_ASYNC_XDA_REQS +
   POOL_QUEUED_ASYNC_TEMP_DATA_REQS +
   POOL_QUEUED_ASYNC_TEMP_INDEX_REQS +
   POOL_QUEUED_ASYNC_TEMP_XDA_REQS
   )
 ) * 100
```

此公式计算失败预取请求占所提出请求总数的百分比。失败预取请求是未能添加至预取队列的请求。如果该百分比很低，那么可能需要通过修改 **num_ioservers** 配置参数来配置更多预取程序。

pool_read_time -“缓冲池物理读时间总计”监视元素

指示从所有类型的表空间的物理表空间容器读取数据和索引页时的耗费时间总计。此值以毫秒计。

表 1316. 表函数监视信息

表函数	监视元素收集级别
MON_FORMAT_XML_METRICS_BY_ROW - 获取所有度量值的基于已格式化行的输出	不适用；报告 XML 文档中作为格式化函数的输入提供的所有元素
MON_FORMAT_XML_TIMES_BY_ROW - - 获取基于已格式化行的组合层次结构等待时间和处理时间	不适用；报告 XML 文档中作为格式化函数的输入提供的所有元素
MON_FORMAT_XML_WAIT_TIMES_BY _ROW - 获取等待时间的已格式化的基于行的输出	不适用；报告 XML 文档中作为格式化函数的输入提供的所有元素
MON_GET_ACTIVITY_DETAILS 表函数 - 获取完整的活动详细信息（在 XML 文档 DETAILS 中报告）	ACTIVITY METRICS BASE
MON_GET_BUFFERPOOL 表函数 - 获取缓冲池度量值	DATA OBJECT METRICS BASE
MON_GET_CONNECTION 表函数 - 获取连接度量值	REQUEST METRICS BASE

表 1316. 表函数监视信息 (续)

表函数	监视元素收集级别
MON_GET_CONNECTION_DETAILS 表函数 - 获取详细的连接度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE
MON_GET_CONTAINER 表函数 - 获取表空间容器度量值	DATA OBJECT METRICS BASE
MON_GET_PKG_CACHE_STMT 表函数 - 获取程序包高速缓存中的 SQL 语句活动度量值	ACTIVITY METRICS BASE
MON_GET_PKG_CACHE_STMT_DETAILS 表函数 - 获取程序包高速缓存条目的详细度量值	ACTIVITY METRICS BASE
MON_GET_SERVICE_SUBCLASS 表函数 - 获取服务子类度量值	REQUEST METRICS BASE
MON_GET_SERVICE_SUBCLASS_DETAILS 表函数 - 获取详细的服务子类度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE
MON_GET_TABLESPACE 表函数 - 获取表空间度量值	DATA OBJECT METRICS BASE
MON_GET_UNIT_OF_WORK 表函数 - 获取工作单元度量值	REQUEST METRICS BASE
MON_GET_UNIT_OF_WORK 表函数 - 获取详细的工作单元度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE
MON_GET_WORKLOAD 表函数 - 获取工作负载度量值	REQUEST METRICS BASE
MON_GET_WORKLOAD_DETAILS 表函数 - 获取详细的工作负载度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE

表 1317. 快照监视信息

快照级别	逻辑数据分组	监视开关
数据库	dbase	缓冲池
表空间	tablespace	缓冲池
缓冲池	bufferpool	缓冲池
应用程序	appl	缓冲池

可将快照监视的计数器重置。

表 1318. 事件监视信息

事件类型	逻辑数据分组	监视开关
活动	event_activity（在 details_xml 文档中报告）	ACTIVITY METRICS BASE
活动	event_activitymetrics	ACTIVITY METRICS BASE
统计信息	event_scstats（在 details_xml 文档中报告）	REQUEST METRICS BASE

表 1318. 事件监视信息 (续)

事件类型	逻辑数据分组	监视开关
统计信息	event_wlstats（在 details_xml 文档中报告）	REQUEST METRICS BASE
数据库	event_db	始终收集
表空间	event_tablespace	始终收集
连接	event_conn	始终收集
程序包高速缓存	在 activity_metrics 文档中报告。	ACTIVITY METRICS BASE

用法

通过将此元素与 **pool_data_p_reads** 和 **pool_index_p_reads** 监视元素配合使用，可以计算平均读页时间。此平均值非常重要，它表示存在 I/O 等待状态，而 I/O 等待状态又表示应该将数据移至另一设备。

在数据库和表空间级别，此元素包括 **pool_async_read_time** 监视元素的值。

pool_secondary_id -“内存池辅助标记”

一个附加标识，用于帮助确定对其返回监视器数据的内存池。

元素标识

pool_secondary_id

元素类型

信息

表 1319. 快照监视信息

快照级别	逻辑数据分组	监视开关
数据库管理器	memory_pool	基本
数据库	memory_pool	基本
应用程序	memory_pool	基本

表 1320. 事件监视信息

事件类型	逻辑数据分组	监视开关
数据库	event_dbmemuse	始终收集
连接	event_connmemuse	始终收集

用法 与 pool_id 一起使用来确定对其返回监视器数据的内存池。pool_secondary_id 的数据仅在必要时才会出现。例如，当指示的 pool_id 是用于确定与监视器数据相关的缓冲池的缓冲池堆时，它就会出现。

创建新数据库后，它将具有缺省缓冲池 IBMDEFAULTBP，其大小将由平台确定。此缓冲池的辅助标识为“1”。除了此缓冲池及您创建的所有缓冲池之外，在缺省情况下还会创建一组系统缓冲池，每个缓冲池对应不同页大小。这些缓冲池的标识会出现在 pool_secondary_id 的快照中:

- 系统 32K 缓冲池

- 系统 16K 缓冲池
- 系统 8K 缓冲池
- 系统 4K 缓冲池

pool_sync_data_gbp_reads -“同步组缓冲池读取数”监视元素

在 DB2 pureScale 环境中，这是从组缓冲池中检索到某个数据页的次数（本来是该数据页应该出现在缓冲池中的次数）。对于 DB2 pureScale 环境外部的环境，此值将为 0。

pool_sync_data_reads -“同步缓冲池数据读取数”监视元素

这是从磁盘中读取到某个数据页的次数（本来是该数据页应该出现在缓冲池中的次数）。

pool_sync_index_gbp_reads -“同步组缓冲池索引读取数”监视元素

在 DB2 pureScale 环境中，这是从组缓冲池中检索到某个索引页的次数（本来是该索引页应该出现在缓冲池中的次数）。对于 DB2 pureScale 环境外部的环境，此值将为 0。

pool_sync_index_reads -“同步缓冲池索引读取数”监视元素

这是从磁盘中读取到某个索引页的次数（本来是该索引页应该出现在缓冲池中的次数）。

pool_sync_xda_gbp_reads -“同步组缓冲池 XDA 数据读取数”监视元素

在 DB2 pureScale 环境上，这是从组缓冲池中检索到某个 XML 页的次数（本来是该 XML 页应该出现在缓冲池中的次数）。对于 DB2 pureScale 环境外部的环境，此值将为 0。

pool_sync_xda_reads -“同步缓冲池 XDA 数据读取数”监视元素

这是从磁盘中读取到某个 XML 页的次数（本来是该 XML 页应该出现在缓冲池中的次数）。

pool_temp_data_l_reads -“缓冲池临时数据逻辑读取数”监视元素

指示向临时表空间的逻辑缓冲池请求的数据页数。

表 1321. 表函数监视信息

表函数	监视元素收集级别
MON_FORMAT_XML_METRICS_BY_ROW - 获取所有度量值的基于已格式化行的输出	不适用; 报告 XML 文档中作为格式化函数的输入提供的所有元素
MON_GET_ACTIVITY_DETAILS 表函数 - 获取完整的活动详细信息（在 XML 文档 DETAILS 中报告）	ACTIVITY METRICS BASE
MON_GET_BUFFERPOOL 表函数 - 获取缓冲池度量值	DATA OBJECT METRICS BASE

表 1321. 表函数监视信息 (续)

表函数	监视元素收集级别
MON_GET_CONNECTION 表函数 - 获取连接度量值	REQUEST METRICS BASE
MON_GET_CONNECTION_DETAILS 表函数 - 获取详细的连接度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE
MON_GET_CONTAINER 表函数 - 获取表空间容器度量值	DATA OBJECT METRICS BASE
MON_GET_PKG_CACHE_STMT 表函数 - 获取程序包高速缓存中的 SQL 语句活动度量值	ACTIVITY METRICS BASE
MON_GET_PKG_CACHE_STMT_DETAILS 表函数 - 获取程序包高速缓存条目的详细度量值	ACTIVITY METRICS BASE
MON_GET_SERVICE_SUBCLASS 表函数 - 获取服务子类度量值	REQUEST METRICS BASE
MON_GET_SERVICE_SUBCLASS_DETAILS 表函数 - 获取详细的服务子类度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE
MON_GET_TABLESPACE 表函数 - 获取表空间度量值	DATA OBJECT METRICS BASE
MON_GET_UNIT_OF_WORK 表函数 - 获取工作单元度量值	REQUEST METRICS BASE
MON_GET_UNIT_OF_WORK 表函数 - 获取详细的工作单元度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE
MON_GET_WORKLOAD 表函数 - 获取工作负载度量值	REQUEST METRICS BASE
MON_GET_WORKLOAD_DETAILS 表函数 - 获取详细的工作负载度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE

表 1322. 快照监视信息

快照级别	逻辑数据分组	监视开关
数据库	dbase	缓冲池
表空间	tablespace	缓冲池
缓冲池	bufferpool	缓冲池
应用程序	appl	缓冲池
应用程序	stmt	缓冲池
动态 SQL	dynsql	缓冲池，语句

可将快照监视的计数器重置。

表 1323. 事件监视信息

事件类型	逻辑数据分组	监视开关
活动	event_activity（在 details_xml 文档中报告）	ACTIVITY METRICS BASE
活动	event_activitymetrics	ACTIVITY METRICS BASE
统计信息	event_scstats（在 details_xml 文档中报告）	REQUEST METRICS BASE
统计信息	event_wlstats（在 details_xml 文档中报告）	REQUEST METRICS BASE
数据库	event_db	始终收集
表空间	event_tablespace	始终收集
连接	event_conn	始终收集
语句	event_stmt	始终收集
活动	event_activity	缓冲池，语句
程序包高速缓存	在 activity_metrics 文档中报告。	ACTIVITY METRICS BASE

用法

通过与 **pool_temp_data_p_reads** 元素配合使用，可计算临时表空间中缓冲池的数据页命中率:

有关更多信息，请参阅第 1379 页的『用于计算缓冲池命中率的公式』。

pool_temp_data_p_reads -“缓冲池临时数据物理读取数”监视元素

指示从临时表空间的物理表空间容器中读取的数据页数。

表 1324. 表函数监视信息

表函数	监视元素收集级别
MON_FORMAT_XML_METRICS_BY_ROW - 获取所有度量值的基于已格式化行的输出	不适用; 报告 XML 文档中作为格式化函数的输入提供的所有元素
MON_GET_ACTIVITY_DETAILS 表函数 - 获取完整的活动详细信息（在 XML 文档 DETAILS 中报告）	ACTIVITY METRICS BASE
MON_GET_BUFFERPOOL 表函数 - 获取缓冲池度量值	DATA OBJECT METRICS BASE
MON_GET_CONNECTION 表函数 - 获取连接度量值	REQUEST METRICS BASE
MON_GET_CONNECTION_DETAILS 表函数 - 获取详细的连接度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE
MON_GET_CONTAINER 表函数 - 获取表空间容器度量值	DATA OBJECT METRICS BASE
MON_GET_PKG_CACHE_STMT 表函数 - 获取程序包高速缓存中的 SQL 语句活动度量值	ACTIVITY METRICS BASE

表 1324. 表函数监视信息 (续)

表函数	监视元素收集级别
MON_GET_PKG_CACHE_STMT_DETAILS 表函数 - 获取程序包高速缓存条目的详细度量值	ACTIVITY METRICS BASE
MON_GET_SERVICE_SUBCLASS 表函数 - 获取服务子类度量值	REQUEST METRICS BASE
MON_GET_SERVICE_SUBCLASS_DETAILS 表函数 - 获取详细的服务子类度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE
MON_GET_TABLESPACE 表函数 - 获取表空间度量值	DATA OBJECT METRICS BASE
MON_GET_UNIT_OF_WORK 表函数 - 获取工作单元度量值	REQUEST METRICS BASE
MON_GET_UNIT_OF_WORK 表函数 - 获取详细的工作单元度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE
MON_GET_WORKLOAD 表函数 - 获取工作负载度量值	REQUEST METRICS BASE
MON_GET_WORKLOAD_DETAILS 表函数 - 获取详细的工作负载度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE

表 1325. 快照监视信息

快照级别	逻辑数据分组	监视开关
数据库	dbase	缓冲池
表空间	tablespace	缓冲池
缓冲池	bufferpool	缓冲池
应用程序	appl	缓冲池
应用程序	stmt	缓冲池
动态 SQL	dynsql	缓冲池，语句

可将快照监视的计数器重置。

表 1326. 事件监视信息

事件类型	逻辑数据分组	监视开关
活动	event_activity（在 details_xml 文档中报告）	ACTIVITY METRICS BASE
活动	event_activitymetrics	ACTIVITY METRICS BASE
统计信息	event_scstats（在 details_xml 文档中报告）	REQUEST METRICS BASE
统计信息	event_wlstats（在 details_xml 文档中报告）	REQUEST METRICS BASE
数据库	event_db	始终收集
表空间	event_tablespace	始终收集
连接	event_conn	始终收集

表 1326. 事件监视信息 (续)

事件类型	逻辑数据分组	监视开关
语句	event_stmt	始终收集
活动	event_activity	缓冲池，语句
程序包高速缓存	在 activity_metrics 文档中报告。	ACTIVITY METRICS BASE
工作单元	在 system_metrics 文档中报告。	REQUEST METRICS BASE

用法

API 和 CLP 快照请求支持在语句级别记录缓冲池信息的功能。

通过与 **pool_temp_data_l_reads** 元素配合使用，可计算临时表空间中缓冲池的数据页命中率：有关更多信息，请参阅第 1379 页的『用于计算缓冲池命中率的公式』

pool_temp_index_l_reads -“缓冲池临时索引逻辑读取数”监视元素

指示向临时表空间的逻辑缓冲池请求的索引页数。

表 1327. 表函数监视信息

表函数	监视元素收集级别
MON_FORMAT_XML_METRICS_BY_ROW - 获取所有度量值的基于已格式化行的输出	不适用；报告 XML 文档中作为格式化函数的输入提供的所有元素
MON_GET_ACTIVITY_DETAILS 表函数 - 获取完整的活动详细信息（在 XML 文档 DETAILS 中报告）	ACTIVITY METRICS BASE
MON_GET_BUFFERPOOL 表函数 - 获取缓冲池度量值	DATA OBJECT METRICS BASE
MON_GET_CONNECTION 表函数 - 获取连接度量值	REQUEST METRICS BASE
MON_GET_CONNECTION_DETAILS 表函数 - 获取详细的连接度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE
MON_GET_CONTAINER 表函数 - 获取表空间容器度量值	DATA OBJECT METRICS BASE
MON_GET_PKG_CACHE_STMT 表函数 - 获取程序包高速缓存中的 SQL 语句活动度量值	ACTIVITY METRICS BASE
MON_GET_PKG_CACHE_STMT_DETAILS 表函数 - 获取程序包高速缓存条目的详细度量值	ACTIVITY METRICS BASE
MON_GET_SERVICE_SUBCLASS 表函数 - 获取服务子类度量值	REQUEST METRICS BASE
MON_GET_SERVICE_SUBCLASS_DETAILS 表函数 - 获取详细的服务子类度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE
MON_GET_TABLESPACE 表函数 - 获取表空间度量值	DATA OBJECT METRICS BASE

表 1327. 表函数监视信息 (续)

表函数	监视元素收集级别
MON_GET_UNIT_OF_WORK 表函数 - 获取工作单元度量值	REQUEST METRICS BASE
MON_GET_UNIT_OF_WORK 表函数 - 获取详细的工作单元度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE
MON_GET_WORKLOAD 表函数 - 获取工作负载度量值	REQUEST METRICS BASE
MON_GET_WORKLOAD_DETAILS 表函数 - 获取详细的工作负载度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE

表 1328. 快照监视信息

快照级别	逻辑数据分组	监视开关
数据库	dbase	缓冲池
表空间	tablespace	缓冲池
缓冲池	bufferpool	缓冲池
应用程序	appl	缓冲池
应用程序	stmt	缓冲池
动态 SQL	dynsql	缓冲池，语句

可将快照监视的计数器重置。

表 1329. 事件监视信息

事件类型	逻辑数据分组	监视开关
活动	event_activity（在 details_xml 文档中报告）	ACTIVITY METRICS BASE
活动	event_activitymetrics	ACTIVITY METRICS BASE
统计信息	event_scstats（在 details_xml 文档中报告）	REQUEST METRICS BASE
统计信息	event_wlstats（在 details_xml 文档中报告）	REQUEST METRICS BASE
数据库	event_db	始终收集
表空间	event_tablespace	始终收集
连接	event_conn	始终收集
语句	event_stmt	始终收集
活动	event_activity	缓冲池，语句
程序包高速缓存	在 activity_metrics 文档中报告。	ACTIVITY METRICS BASE

用法

通过将此元素与 **pool_temp_index_p_reads** 元素配合使用，可以计算临时表空间中缓冲池的索引页命中率: 有关更多信息，请参阅第 1379 页的『用于计算缓冲池命中率的公式』。

pool_temp_index_p_reads -“缓冲池临时索引物理读取数”监视元素

指示从临时表空间的物理表空间容器中读取的索引页数。

表 1330. 表函数监视信息

表函数	监视元素收集级别
MON_FORMAT_XML_METRICS_BY_ROW - 获取所有度量值的基于已格式化行的输出	不适用; 报告 XML 文档中作为格式化函数的输入提供的所有元素
MON_GET_ACTIVITY_DETAILS 表函数 - 获取完整的活动详细信息（在 XML 文档 DETAILS 中报告）	ACTIVITY METRICS BASE
MON_GET_BUFFERPOOL 表函数 - 获取缓冲池度量值	DATA OBJECT METRICS BASE
MON_GET_CONNECTION 表函数 - 获取连接度量值	REQUEST METRICS BASE
MON_GET_CONNECTION_DETAILS 表函数 - 获取详细的连接度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE
MON_GET_CONTAINER 表函数 - 获取表空间容器度量值	DATA OBJECT METRICS BASE
MON_GET_PKG_CACHE_STMT 表函数 - 获取程序包高速缓存中的 SQL 语句活动度量值	ACTIVITY METRICS BASE
MON_GET_PKG_CACHE_STMT_DETAILS 表函数 - 获取程序包高速缓存条目的详细度量值	ACTIVITY METRICS BASE
MON_GET_SERVICE_SUBCLASS 表函数 - 获取服务子类度量值	REQUEST METRICS BASE
MON_GET_SERVICE_SUBCLASS_DETAILS 表函数 - 获取详细的服务子类度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE
MON_GET_TABLESPACE 表函数 - 获取表空间度量值	DATA OBJECT METRICS BASE
MON_GET_UNIT_OF_WORK 表函数 - 获取工作单元度量值	REQUEST METRICS BASE
MON_GET_UNIT_OF_WORK 表函数 - 获取详细的工作单元度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE
MON_GET_WORKLOAD 表函数 - 获取工作负载度量值	REQUEST METRICS BASE
MON_GET_WORKLOAD_DETAILS 表函数 - 获取详细的工作负载度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE

表 1331. 快照监视信息

快照级别	逻辑数据分组	监视开关
数据库	dbase	缓冲池
表空间	tablespace	缓冲池
缓冲池	bufferpool	缓冲池
应用程序	appl	缓冲池
应用程序	stmt	缓冲池
动态 SQL	dynsql	缓冲池，语句

可将快照监视的计数器重置。

表 1332. 事件监视信息

事件类型	逻辑数据分组	监视开关
活动	event_activity（在 details_xml 文档中报告）	ACTIVITY METRICS BASE
活动	event_activitymetrics	ACTIVITY METRICS BASE
统计信息	event_scstats（在 details_xml 文档中报告）	REQUEST METRICS BASE
统计信息	event_wlstats（在 details_xml 文档中报告）	REQUEST METRICS BASE
数据库	event_db	始终收集
表空间	event_tablespace	始终收集
连接	event_conn	始终收集
语句	event_stmt	始终收集
活动	event_activity	缓冲池，语句
程序包高速缓存	在 activity_metrics 文档中报告。	ACTIVITY METRICS BASE
工作单元	在 system_metrics 文档中报告。	REQUEST METRICS BASE

用法

通过将此元素与 **pool_temp_index_l_reads** 元素配合使用，可以计算临时表空间中缓冲池的索引页命中率：有关更多信息，请参阅第 1379 页的『用于计算缓冲池命中率的公式』。

pool_temp_xda_l_reads -“缓冲池临时 XDA 数据逻辑读取数”监视元素

指示向临时表空间的逻辑缓冲池请求的 XML 存储器对象（XDA）数据页数。

表 1333. 表函数监视信息

表函数	监视元素收集级别
MON_FORMAT_XML_METRICS_BY_ROW - 获取所有度量值的基于已格式化行的输出	不适用；报告 XML 文档中作为格式化函数的输入提供的所有元素

表 1333. 表函数监视信息 (续)

表函数	监视元素收集级别
MON_GET_ACTIVITY_DETAILS 表函数 - 获取完整的活动详细信息（在 XML 文档 DETAILS 中报告）	ACTIVITY METRICS BASE
MON_GET_BUFFERPOOL 表函数 - 获取缓冲池度量值	DATA OBJECT METRICS BASE
MON_GET_CONNECTION 表函数 - 获取连接度量值	REQUEST METRICS BASE
MON_GET_CONNECTION_DETAILS 表函数 - 获取详细的连接度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE
MON_GET_CONTAINER 表函数 - 获取表空间容器度量值	DATA OBJECT METRICS BASE
MON_GET_PKG_CACHE_STMT 表函数 - 获取程序包高速缓存中的 SQL 语句活动度量值	ACTIVITY METRICS BASE
MON_GET_SERVICE_SUBCLASS 表函数 - 获取服务子类度量值	REQUEST METRICS BASE
MON_GET_SERVICE_SUBCLASS_DETAILS 表函数 - 获取详细的服务子类度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE
MON_GET_TABLESPACE 表函数 - 获取表空间度量值	DATA OBJECT METRICS BASE
MON_GET_UNIT_OF_WORK 表函数 - 获取工作单元度量值	REQUEST METRICS BASE
MON_GET_UNIT_OF_WORK 表函数 - 获取详细的工作单元度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE
MON_GET_WORKLOAD 表函数 - 获取工作负载度量值	REQUEST METRICS BASE
MON_GET_WORKLOAD_DETAILS 表函数 - 获取详细的工作负载度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE

表 1334. 快照监视信息

快照级别	逻辑数据分组	监视开关
数据库	dbase	缓冲池
表空间	tablespace	缓冲池
缓冲池	bufferpool	缓冲池
应用程序	appl	缓冲池
应用程序	stmt	缓冲池
动态 SQL	dynsql	缓冲池，语句

可将快照监视的计数器重置。

表 1335. 事件监视信息

事件类型	逻辑数据分组	监视开关
活动	event_activity（在 details_xml 文档中报告）	ACTIVITY METRICS BASE
活动	event_activitymetrics	ACTIVITY METRICS BASE
统计信息	event_scstats（在 details_xml 文档中报告）	REQUEST METRICS BASE
统计信息	event_wlstats（在 details_xml 文档中报告）	REQUEST METRICS BASE
数据库	event_db	始终收集
表空间	event_tablespace	始终收集
连接	event_conn	始终收集
语句	event_stmt	始终收集
活动	event_activity	缓冲池，语句
程序包高速缓存	在 activity_metrics 文档中报告。	ACTIVITY METRICS BASE

用法

通过将 **pool_temp_xda_l_reads** 监视元素与 **pool_temp_xda_p_reads**、**pool_temp_data_l_reads** 和 **pool_temp_data_p_reads** 监视元素配合使用并借助以下公式，可以计算临时表空间中缓冲池的数据页命中率:

```
1 - ((pool_temp_data_p_reads + pool_temp_xda_p_reads)
      / (pool_temp_data_l_reads + pool_temp_xda_l_reads))
```

pool_temp_xda_p_reads -“缓冲池临时 XDA 数据物理读取数”监视元素

指示从临时表空间的物理表空间容器中读取的 XML 存储器对象（XDA）数据页数。

表 1336. 表函数监视信息

表函数	监视元素收集级别
MON_FORMAT_XML_METRICS_BY_ROW - 获取所有度量值的基于已格式化行的输出	不适用; 报告 XML 文档中作为格式化函数的输入提供的所有元素
MON_GET_ACTIVITY_DETAILS 表函数 - 获取完整的活动详细信息（在 XML 文档 DETAILS 中报告）	ACTIVITY METRICS BASE
MON_GET_BUFFERPOOL 表函数 - 获取缓冲池度量值	DATA OBJECT METRICS BASE
MON_GET_CONNECTION 表函数 - 获取连接度量值	REQUEST METRICS BASE
MON_GET_CONNECTION_DETAILS 表函数 - 获取详细的连接度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE
MON_GET_CONTAINER 表函数 - 获取表空间容器度量值	DATA OBJECT METRICS BASE

表 1336. 表函数监视信息 (续)

表函数	监视元素收集级别
MON_GET_PKG_CACHE_STMT 表函数 - 获取程序包高速缓存中的 SQL 语句活动度量值	ACTIVITY METRICS BASE
MON_GET_PKG_CACHE_STMT_DETAILS 表函数 - 获取程序包高速缓存条目的详细度量值	ACTIVITY METRICS BASE
MON_GET_SERVICE_SUBCLASS 表函数 - 获取服务子类度量值	REQUEST METRICS BASE
MON_GET_SERVICE_SUBCLASS_DETAILS 表函数 - 获取详细的服务子类度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE
MON_GET_TABLESPACE 表函数 - 获取表空间度量值	DATA OBJECT METRICS BASE
MON_GET_UNIT_OF_WORK 表函数 - 获取工作单元度量值	REQUEST METRICS BASE
MON_GET_UNIT_OF_WORK 表函数 - 获取详细的工作单元度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE
MON_GET_WORKLOAD 表函数 - 获取工作负载度量值	REQUEST METRICS BASE
MON_GET_WORKLOAD_DETAILS 表函数 - 获取详细的工作负载度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE

表 1337. 快照监视信息

快照级别	逻辑数据分组	监视开关
数据库	dbase	缓冲池
表空间	tablespace	缓冲池
缓冲池	bufferpool	缓冲池
应用程序	appl	缓冲池
应用程序	stmt	缓冲池
动态 SQL	dynsql	缓冲池，语句

可将快照监视的计数器重置。

表 1338. 事件监视信息

事件类型	逻辑数据分组	监视开关
活动	event_activity（在 details_xml 文档中报告）	ACTIVITY METRICS BASE
活动	event_activitymetrics	ACTIVITY METRICS BASE
统计信息	event_scstats（在 details_xml 文档中报告）	REQUEST METRICS BASE
统计信息	event_wlstats（在 details_xml 文档中报告）	REQUEST METRICS BASE
数据库	event_db	始终收集

表 1338. 事件监视信息 (续)

事件类型	逻辑数据分组	监视开关
表空间	event_tablespace	始终收集
连接	event_conn	始终收集
语句	event_stmt	始终收集
活动	event_activity	缓冲池，语句
程序包高速缓存	在 activity_metrics 文档中报告。	ACTIVITY METRICS BASE
工作单元	在 system_metrics 文档中报告。	REQUEST METRICS BASE

用法

通过将 **pool_temp_xda_p_reads** 监视元素与 **pool_temp_xda_l_reads**、**pool_temp_data_l_reads** 和 **pool_temp_data_p_reads** 监视元素配合使用并借助以下公式，可以计算临时表空间中缓冲池的数据页命中率:

```
1 - ((pool_temp_data_p_reads + pool_temp_xda_p_reads) /
(pool_temp_data_l_reads + pool_temp_xda_l_reads))
```

pool_watermark -“内存池水位标记”

自创建内存池后内存池的最大大小。此值以字节计。

元素标识

pool_watermark

元素类型

信息

表 1339. 快照监视信息

快照级别	逻辑数据分组	监视开关
数据库管理器	memory_pool	基本
数据库	memory_pool	基本
应用程序	memory_pool	基本

表 1340. 事件监视信息

事件类型	逻辑数据分组	监视开关
数据库	event_dbmemuse	始终收集
连接	event_connmemuse	始终收集

用法 在一直运行的系统上，可将 *pool_watermark* 和 *pool_config_size* 元素一起使用来预测潜在的内存问题。

例如，按一定时间间隔获取快照（如每天），并检查 *pool_watermark* 和 *pool_config_size* 值。如果发现 *pool_watermark* 的值开始逐步接近 *pool_config_size*（预示将来可能出现内存相关问题），那么可能指示应增加内存池的大小。

pool_write_time -“缓冲池物理写时间总计”监视元素

提供以物理方式将缓冲池中的数据或索引页写至磁盘时耗用的总时间。耗用时间以毫秒计。

表 1341. 表函数监视信息

表函数	监视元素收集级别
MON_FORMAT_XML_METRICS_BY_ROW - 获取所有度量值的基于已格式化行的输出	不适用; 报告 XML 文档中作为格式化函数的输入提供的所有元素
MON_FORMAT_XML_TIMES_BY_ROW - - 获取基于已格式化行的组合层次结构等待时间和处理时间	不适用; 报告 XML 文档中作为格式化函数的输入提供的所有元素
MON_FORMAT_XML_WAIT_TIMES_BY _ROW - 获取等待时间的已格式化的基于行的输出	不适用; 报告 XML 文档中作为格式化函数的输入提供的所有元素
MON_GET_ACTIVITY_DETAILS 表函数 - 获取完整的活动详细信息（在 XML 文档 DETAILS 中报告）	ACTIVITY METRICS BASE
MON_GET_BUFFERPOOL 表函数 - 获取缓冲池度量值	DATA OBJECT METRICS BASE
MON_GET_CONNECTION 表函数 - 获取连接度量值	REQUEST METRICS BASE
MON_GET_CONNECTION_DETAILS 表函数 - 获取详细的连接度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE
MON_GET_CONTAINER 表函数 - 获取表空间容器度量值	DATA OBJECT METRICS BASE
MON_GET_PKG_CACHE_STMT 表函数 - 获取程序包高速缓存中的 SQL 语句活动度量值	ACTIVITY METRICS BASE
MON_GET_PKG_CACHE_STMT_DETAILS 表函数 - 获取程序包高速缓存条目的详细度量值	ACTIVITY METRICS BASE
MON_GET_SERVICE_SUBCLASS 表函数 - 获取服务子类度量值	REQUEST METRICS BASE
MON_GET_SERVICE_SUBCLASS_DETAILS 表函数 - 获取详细的服务子类度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE
MON_GET_TABLESPACE 表函数 - 获取表空间度量值	DATA OBJECT METRICS BASE
MON_GET_UNIT_OF_WORK 表函数 - 获取工作单元度量值	REQUEST METRICS BASE
MON_GET_UNIT_OF_WORK 表函数 - 获取详细的工作单元度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE
MON_GET_WORKLOAD 表函数 - 获取工作负载度量值	REQUEST METRICS BASE
MON_GET_WORKLOAD_DETAILS 表函数 - 获取详细的工作负载度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE

表 1342. 快照监视信息

快照级别	逻辑数据分组	监视开关
数据库	dbase	缓冲池
表空间	tablespace	缓冲池
缓冲池	bufferpool	缓冲池
应用程序	appl	缓冲池

可将快照监视的计数器重置。

表 1343. 事件监视信息

事件类型	逻辑数据分组	监视开关
活动	event_activity（在 details_xml 文档中报告）	ACTIVITY METRICS BASE
活动	event_activitymetrics	ACTIVITY METRICS BASE
统计信息	event_scstats（在 details_xml 文档中报告）	REQUEST METRICS BASE
统计信息	event_wlstats（在 details_xml 文档中报告）	REQUEST METRICS BASE
工作单元	在 system_metrics 文档中报告。	REQUEST METRICS BASE
数据库	event_db	始终收集
表空间	event_tablespace	始终收集
连接	event_conn	始终收集
程序包高速缓存	在 activity_metrics 文档中报告。	ACTIVITY METRICS BASE

用法

通过将此元素与 **pool_data_writes** 和 **pool_index_writes** 监视元素配合使用，可以计算平均写页时间。此平均值非常重要，它表示存在 I/O 等待状态，而 I/O 等待状态又表示应该将数据移至另一设备。

在数据库和表空间级别，此元素包括 **pool_async_write_time** 监视元素的值。

pool_xda_gbp_indep_pages_found_in_lbp -“本地缓冲池中发现的独立于组缓冲池的 XDA 页数”监视元素

代理程序在本地缓冲池 (LBP) 中发现的独立于组缓冲池 (GBP) 的 XML 存储器对象 (XDA) 数据页数。

表 1344. 表函数监视信息

表函数	监视元素收集级别
MON_FORMAT_XML_METRICS_BY_ROW - 获取所有度量值的基于已格式化行的输出	不适用；报告 XML 文档中作为格式化函数的输入提供的所有元素

表 1344. 表函数监视信息 (续)

表函数	监视元素收集级别
MON_GET_ACTIVITY_DETAILS 表函数 - 获取完整的活动详细信息（在 XML 文档 DETAILS 中报告）	ACTIVITY METRICS BASE
MON_GET_BUFFERPOOL 表函数 - 获取缓冲池度量值	DATA OBJECT METRICS BASE
MON_GET_CONNECTION 表函数 - 获取连接度量值	REQUEST METRICS BASE
MON_GET_CONNECTION_DETAILS 表函数 - 获取详细连接度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE
MON_GET_PKG_CACHE_STMT 表函数 - 获取程序包高速缓存中的 SQL 语句活动度量值	ACTIVITY METRICS BASE
MON_GET_PKG_CACHE_STMT_DETAILS - 获取程序包高速缓存条目的详细度量值（在 XML 文档 DETAILS 中报告）	ACTIVITY METRICS BASE
MON_GET_SERVICE_SUBCLASS 表函数 - 获取服务子类度量值	REQUEST METRICS BASE
MON_GET_SERVICE_SUBCLASS_DETAILS 表函数 - 获取详细的服务子类度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE
MON_GET_TABLESPACE 表函数 - 获取表空间度量值	DATA OBJECT METRICS BASE
MON_GET_UNIT_OF_WORK 表函数 - 获取工作单元度量值	REQUEST METRICS BASE
MON_GET_UNIT_OF_WORK 表函数 - 获取详细的工作单元度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE
MON_GET_WORKLOAD 表函数 - 获取工作负载度量值	REQUEST METRICS BASE
MON_GET_WORKLOAD_DETAILS 表函数 - 获取详细的工作负载度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE

表 1345. 事件监视信息

事件类型	逻辑数据分组	监视开关
活动	event_activitymetrics	ACTIVITY METRICS BASE
程序包高速缓存	pkgcache_metrics	ACTIVITY METRICS BASE
工作单元	uow_metrics	REQUEST METRICS BASE

pool_xda_gbp_invalid_pages -“组缓冲池无效 XDA 数据页数”监视元素

向组缓冲池发出针对 XML 存储器对象 (XDA) 的数据页的请求（因为该页在本地缓冲池中标记为无效）的次数。

表 1346. 表函数监视信息

表函数	监视元素收集级别
MON_FORMAT_XML_METRICS_BY_ROW - 获取所有度量值的基于已格式化行的输出	不适用; 报告 XML 文档中作为格式化函数的输入提供的所有元素
MON_GET_ACTIVITY_DETAILS 表函数 - 获取完整的活动详细信息（在 XML 文档 DETAILS 中报告）	ACTIVITY METRICS BASE
MON_GET_BUFFERPOOL 表函数 - 获取缓冲池度量值	DATA OBJECT METRICS BASE
MON_GET_CONNECTION 表函数 - 获取连接度量值	REQUEST METRICS BASE
MON_GET_CONNECTION_DETAILS 表函数 - 获取详细的连接度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE
MON_GET_PKG_CACHE_STMT 表函数 - 获取程序包高速缓存中的 SQL 语句活动度量值	ACTIVITY METRICS BASE
MON_GET_PKG_CACHE_STMT_DETAILS 表函数 - 获取程序包高速缓存条目的详细度量值（在 XML 文档 DETAILS 中报告）	ACTIVITY METRICS BASE
MON_GET_SERVICE_SUBCLASS 表函数 - 获取服务子类度量值	REQUEST METRICS BASE
MON_GET_SERVICE_SUBCLASS_DETAILS 表函数 - 获取详细的服务子类度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE
MON_GET_TABLESPACE 表函数 - 获取表空间度量值	DATA OBJECT METRICS BASE
MON_GET_UNIT_OF_WORK 表函数 - 获取工作单元度量值	REQUEST METRICS BASE
MON_GET_UNIT_OF_WORK 表函数 - 获取详细的工作单元度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE
MON_GET_WORKLOAD 表函数 - 获取工作负载度量值	REQUEST METRICS BASE
MON_GET_WORKLOAD_DETAILS 表函数 - 获取详细的工作负载度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE

表 1347. 事件监视信息

事件类型	逻辑数据分组	监视开关
活动	event_activity（在 details_xml 文档中报告）	ACTIVITY METRICS BASE
活动	event_activitymetrics	ACTIVITY METRICS BASE

表 1347. 事件监视信息 (续)

事件类型	逻辑数据分组	监视开关
统计信息	event_scstats（在 details_xml 文档中报告）	REQUEST METRICS BASE
统计信息	event_wlstats（在 details_xml 文档中报告）	REQUEST METRICS BASE
工作单元	在 system_metrics 文档中报告。	REQUEST METRICS BASE
程序包高速缓存	在 activity_metrics 文档中报告。	ACTIVITY METRICS BASE

用法

要确定在本地缓冲池中发现所请求 XDA 页的频率，请使用以下公式:

```
(pool_xda_lbp_pages_found - pool_async_xda_lbp_pages_found) / pool_xda_l_reads
```

要确定在组缓冲池中发现所请求 XDA 页的次数，请使用以下公式

```
(pool_xda_gbp_l_reads - pool_xda_gbp_p_reads) / pool_xda_gbp_l_reads
```

本地缓冲池和组缓冲池命中率都是集群高速缓存设施的整体性能中的重要因素。使用这些公式可帮助您确定本地缓冲池或组缓冲池会否成为数据库吞吐量的限制因素。

pool_xda_gbp_l_reads -“组缓冲池 XDA 数据逻辑读取请求数”监视元素

尝试从组缓冲池读取 XML 存储器对象 (XDA) 的依赖于 GBP 的数据页（因为该页在本地缓冲池中无效或不存在）的次数。

表 1348. 表函数监视信息

表函数	监视元素收集级别
MON_FORMAT_XML_METRICS_BY_ROW - 获取所有度量值的基于已格式化行的输出	不适用; 报告 XML 文档中作为格式化函数的输入提供的所有元素
MON_GET_ACTIVITY_DETAILS 表函数 - 获取完整的活动详细信息（在 XML 文档 DETAILS 中报告）	ACTIVITY METRICS BASE
MON_GET_BUFFERPOOL 表函数 - 获取缓冲池度量值	DATA OBJECT METRICS BASE
MON_GET_CONNECTION 表函数 - 获取连接度量值	REQUEST METRICS BASE
MON_GET_CONNECTION_DETAILS 表函数 - 获取详细的连接度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE
MON_GET_PKG_CACHE_STMT 表函数 - 获取程序包高速缓存中的 SQL 语句活动度量值	ACTIVITY METRICS BASE
MON_GET_PKG_CACHE_STMT_DETAILS 表函数 - 获取程序包高速缓存条目的详细度量值（在 XML 文档 DETAILS 中报告）	ACTIVITY METRICS BASE
MON_GET_SERVICE_SUBCLASS 表函数 - 获取服务子类度量值	REQUEST METRICS BASE

表 1348. 表函数监视信息 (续)

表函数	监视元素收集级别
MON_GET_SERVICE_SUBCLASS_DETAILS 表函数 - 获取详细的服务子类度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE
MON_GET_TABLESPACE 表函数 - 获取表空间度量值	DATA OBJECT METRICS BASE
MON_GET_UNIT_OF_WORK 表函数 - 获取工作单元度量值	REQUEST METRICS BASE
MON_GET_UNIT_OF_WORK 表函数 - 获取详细的工作单元度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE
MON_GET_WORKLOAD 表函数 - 获取工作负载度量值	REQUEST METRICS BASE
MON_GET_WORKLOAD_DETAILS 表函数 - 获取详细的工作负载度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE

表 1349. 事件监视信息

事件类型	逻辑数据分组	监视开关
活动	event_activity（在 details_xml 文档中报告）	ACTIVITY METRICS BASE
活动	event_activitymetrics	ACTIVITY METRICS BASE
统计信息	event_scstats（在 details_xml 文档中报告）	REQUEST METRICS BASE
统计信息	event_wlstats（在 details_xml 文档中报告）	REQUEST METRICS BASE
工作单元	在 system_metrics 文档中报告。	REQUEST METRICS BASE
程序包高速缓存	在 activity_metrics 文档中报告。	ACTIVITY METRICS BASE

用法

要确定在本地缓冲池中发现所请求 XDA 页的频率，请使用以下公式:

```
(pool_xda_lbp_pages_found - pool_async_xda_lbp_pages_found) / pool_xda_l_reads
```

要确定在组缓冲池中发现所请求 XDA 页的次数，请使用以下公式

```
(pool_xda_gbp_l_reads - pool_xda_gbp_p_reads) / pool_xda_gbp_l_reads
```

本地缓冲池和组缓冲池命中率都是集群高速缓存设施的整体性能中的重要因素。使用这些公式可帮助您确定本地缓冲池或组缓冲池会否成为数据库吞吐量的限制因素。

pool_xda_gbp_p_reads -“组缓冲池 XDA 数据物理读取请求数”监视元素

尝试将磁盘中 XML 存储器对象 (XDA) 的依赖于 GBP 的数据页读取到本地缓冲池中（因为在组缓冲池中找不到该页）的次数。

表 1350. 表函数监视信息

表函数	监视元素收集级别
MON_FORMAT_XML_METRICS_BY_ROW - 获取所有度量值的基于已格式化行的输出	不适用; 报告 XML 文档中作为格式化函数的输入提供的所有元素
MON_GET_ACTIVITY_DETAILS 表函数 - 获取完整的活动详细信息（在 XML 文档 DETAILS 中报告）	ACTIVITY METRICS BASE
MON_GET_BUFFERPOOL 表函数 - 获取缓冲池度量值	DATA OBJECT METRICS BASE
MON_GET_CONNECTION 表函数 - 获取连接度量值	REQUEST METRICS BASE
MON_GET_CONNECTION_DETAILS 表函数 - 获取详细的连接度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE
MON_GET_PKG_CACHE_STMT 表函数 - 获取程序包高速缓存中的 SQL 语句活动度量值	ACTIVITY METRICS BASE
MON_GET_PKG_CACHE_STMT_DETAILS 表函数 - 获取程序包高速缓存条目的详细度量值（在 XML 文档 DETAILS 中报告）	ACTIVITY METRICS BASE
MON_GET_SERVICE_SUBCLASS 表函数 - 获取服务子类度量值	REQUEST METRICS BASE
MON_GET_SERVICE_SUBCLASS_DETAILS 表函数 - 获取详细的服务子类度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE
MON_GET_TABLESPACE 表函数 - 获取表空间度量值	DATA OBJECT METRICS BASE
MON_GET_UNIT_OF_WORK 表函数 - 获取工作单元度量值	REQUEST METRICS BASE
MON_GET_UNIT_OF_WORK 表函数 - 获取详细的工作单元度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE
MON_GET_WORKLOAD 表函数 - 获取工作负载度量值	REQUEST METRICS BASE
MON_GET_WORKLOAD_DETAILS 表函数 - 获取详细的工作负载度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE

表 1351. 事件监视信息

事件类型	逻辑数据分组	监视开关
活动	event_activity（在 details_xml 文档中报告）	ACTIVITY METRICS BASE
活动	event_activitymetrics	ACTIVITY METRICS BASE

表 1351. 事件监视信息 (续)

事件类型	逻辑数据分组	监视开关
统计信息	event_scstats（在 details_xml 文档中报告）	REQUEST METRICS BASE
统计信息	event_wlstats（在 details_xml 文档中报告）	REQUEST METRICS BASE
工作单元	在 system_metrics 文档中报告。	REQUEST METRICS BASE
程序包高速缓存	在 activity_metrics 文档中报告。	ACTIVITY METRICS BASE

用法

要确定在本地缓冲池中发现所请求 XDA 页的频率，请使用以下公式:

```
(pool_xda_lbp_pages_found - pool_async_xda_lbp_pages_found) / pool_xda_l_reads
```

要确定在组缓冲池中发现所请求 XDA 页的次数，请使用以下公式

```
(pool_xda_gbp_l_reads - pool_xda_gbp_p_reads) / pool_xda_gbp_l_reads
```

本地缓冲池和组缓冲池命中率都是集群高速缓存设施的整体性能中的重要因素。使用这些公式可帮助您确定本地缓冲池或组缓冲池会否成为数据库吞吐量的限制因素。

pool_xda_l_reads -“缓冲池 XDA 数据逻辑读取数”监视元素

指示向常规表空间和大型表空间的逻辑缓冲池请求的 XML 存储器对象（XDA）数据页数。

表 1352. 表函数监视信息

表函数	监视元素收集级别
MON_FORMAT_XML_METRICS_BY_ROW - 获取所有度量值的基于已格式化行的输出	不适用; 报告 XML 文档中作为格式化函数的输入提供的所有元素
MON_GET_ACTIVITY_DETAILS 表函数 - 获取完整的活动详细信息（在 XML 文档 DETAILS 中报告）	ACTIVITY METRICS BASE
MON_GET_BUFFERPOOL 表函数 - 获取缓冲池度量值	DATA OBJECT METRICS BASE
MON_GET_CONNECTION 表函数 - 获取连接度量值	REQUEST METRICS BASE
MON_GET_CONNECTION_DETAILS 表函数 - 获取详细的连接度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE
MON_GET_CONTAINER 表函数 - 获取表空间容器度量值	DATA OBJECT METRICS BASE
MON_GET_PKG_CACHE_STMT 表函数 - 获取程序包高速缓存中的 SQL 语句活动度量值	ACTIVITY METRICS BASE
MON_GET_SERVICE_SUBCLASS 表函数 - 获取服务子类度量值	REQUEST METRICS BASE

表 1352. 表函数监视信息 (续)

表函数	监视元素收集级别
MON_GET_SERVICE_SUBCLASS_DETAILS 表函数 - 获取详细的服务子类度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE
MON_GET_TABLESPACE 表函数 - 获取表空间度量值	DATA OBJECT METRICS BASE
MON_GET_UNIT_OF_WORK 表函数 - 获取工作单元度量值	REQUEST METRICS BASE
MON_GET_UNIT_OF_WORK 表函数 - 获取详细的工作单元度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE
MON_GET_WORKLOAD 表函数 - 获取工作负载度量值	REQUEST METRICS BASE
MON_GET_WORKLOAD_DETAILS 表函数 - 获取详细的工作负载度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE

表 1353. 快照监视信息

快照级别	逻辑数据分组	监视开关
数据库	dbase	缓冲池
表空间	tablespace	缓冲池
缓冲池	bufferpool	缓冲池
应用程序	appl	缓冲池
应用程序	stmt	缓冲池
动态 SQL	dynsql	缓冲池，语句

可将快照监视的计数器重置。

表 1354. 事件监视信息

事件类型	逻辑数据分组	监视开关
活动	event_activity（在 details_xml 文档中报告）	ACTIVITY METRICS BASE
活动	event_activitymetrics	ACTIVITY METRICS BASE
统计信息	event_scstats（在 details_xml 文档中报告）	REQUEST METRICS BASE
统计信息	event_wlstats（在 details_xml 文档中报告）	REQUEST METRICS BASE
工作单元	在 system_metrics 文档中报告。	始终收集
数据库	event_db	始终收集
表空间	event_tablespace	始终收集
连接	event_conn	始终收集
语句	event_stmt	始终收集
活动	event_activity	缓冲池，语句

表 1354. 事件监视信息 (续)

事件类型	逻辑数据分组	监视开关
程序包高速缓存	在 activity_metrics 文档中报告。	ACTIVITY METRICS BASE

用法

此计数包括数据处于下列情况时对数据的访问:

- 当数据库管理器需要处理页时索引页已经在缓冲池中
- 应读取到缓冲池中，数据库管理器才能处理页。

使用 **pool_xda_l_reads**、**pool_xda_p_reads**、**pool_data_l_reads** 和 **pool_data_p_reads** 监视元素来计算缓冲池的数据页命中率。有关更多信息，请参阅第 1379 页的『用于计算缓冲池命中率的公式』

例如，可按如下所示计算整体缓冲池命中率:

```
     ((pool_data_lbp_pages_found + pool_index_lbp_pages_found
+ pool_xda_lbp_pages_found
-
pool_async_data_lbp_pages_found - pool_async_index_lbp_pages_found
- pool_async_xda_lbp_pages_found)
/ (pool_data_l_reads
+ pool_index_l_reads + pool_xda_l_reads + pool_temp_data_l_reads +
pool_temp_xda_l_reads + pool_temp_index_l_reads)) × 100
```

此计算公式考虑缓冲池高速缓存的所有页（索引和数据）。

增加缓冲池大小一般会改进命中率，但您会达到一个最优状态，而无法继续改进。从理论上说，如果能够分配大到足以存储整个数据库的缓冲池，那么系统启动并运行后你可以得到 100% 的命中率。但在许多情况下这是不现实的。命中率的高低取决于数据的大小以及访问数据的方式。如果数据库很大并且数据访问比较平均，那么命中率将会很低。对于非常大的表，您几乎无能为力。在此类情况下，应该将重点放在较小并且访问较为频繁的表以及索引上。

pool_xda_lbp_pages_found -“发现的本地缓冲池 XDA 数据页数”监视元素

向本地缓冲池请求并在其中发现 XML 存储器对象 (XDA) 的数据页的次数。

表 1355. 表函数监视信息

表函数	监视元素收集级别
MON_FORMAT_XML_METRICS_BY_ROW - 获取所有度量值的基于已格式化行的输出	不适用; 报告 XML 文档中作为格式化函数的输入提供的所有元素
MON_GET_ACTIVITY_DETAILS 表函数 - 获取完整的活动详细信息（在 XML 文档 DETAILS 中报告）	ACTIVITY METRICS BASE
MON_GET_BUFFERPOOL 表函数 - 获取缓冲池度量值	DATA OBJECT METRICS BASE
MON_GET_CONNECTION 表函数 - 获取连接度量值	REQUEST METRICS BASE

表 1355. 表函数监视信息 (续)

表函数	监视元素收集级别
MON_GET_CONNECTION_DETAILS 表函数 - 获取详细的连接度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE
MON_GET_PKG_CACHE_STMT 表函数 - 获取程序包高速缓存中的 SQL 语句活动度量值	ACTIVITY METRICS BASE
MON_GET_PKG_CACHE_STMT_DETAILS 表函数 - 获取程序包高速缓存条目的详细度量值（在 XML 文档 DETAILS 中报告）	ACTIVITY METRICS BASE
MON_GET_SERVICE_SUBCLASS 表函数 - 获取服务子类度量值	REQUEST METRICS BASE
MON_GET_SERVICE_SUBCLASS_DETAILS 表函数 - 获取详细的服务子类度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE
MON_GET_TABLESPACE 表函数 - 获取表空间度量值	DATA OBJECT METRICS BASE
MON_GET_UNIT_OF_WORK 表函数 - 获取工作单元度量值	REQUEST METRICS BASE
MON_GET_UNIT_OF_WORK 表函数 - 获取详细的工作单元度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE
MON_GET_WORKLOAD 表函数 - 获取工作负载度量值	REQUEST METRICS BASE
MON_GET_WORKLOAD_DETAILS 表函数 - 获取详细的工作负载度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE

表 1356. 事件监视信息

事件类型	逻辑数据分组	监视开关
活动	event_activity（在 details_xml 文档中报告）	ACTIVITY METRICS BASE
活动	event_activitymetrics	ACTIVITY METRICS BASE
统计信息	event_scstats（在 details_xml 文档中报告）	REQUEST METRICS BASE
统计信息	event_wlstats（在 details_xml 文档中报告）	REQUEST METRICS BASE
工作单元	在 system_metrics 文档中报告。	REQUEST METRICS BASE
程序包高速缓存	在 activity_metrics 文档中报告。	ACTIVITY METRICS BASE

用法

要确定在本地缓冲池中发现所请求 XDA 页的频率，请使用以下公式:

```
(pool_xda_lbp_pages_found - pool_async_xda_lbp_pages_found) / pool_xda_l_reads
```

要确定在组缓冲池中发现所请求 XDA 页的次数，请使用以下公式

```
(pool_xda_gbp_l_reads - pool_xda_gbp_p_reads) / pool_xda_gbp_l_reads
```

本地缓冲池和组缓冲池命中率都是集群高速缓存设施的整体性能中的重要因素。使用这些公式可帮助您确定本地缓冲池或组缓冲池会否成为数据库吞吐量的限制因素。

pool_xda_p_reads -“缓冲池 XDA 数据物理读取数”监视元素

指示从常规表空间和大型表空间的物理表空间容器中读取的 XML 存储器对象（XDA）数据页数。

表 1357. 表函数监视信息

表函数	监视元素收集级别
MON_FORMAT_XML_METRICS_BY_ROW - 获取所有度量值的基于已格式化行的输出	不适用; 报告 XML 文档中作为格式化函数的输入提供的所有元素
MON_GET_ACTIVITY_DETAILS 表函数 - 获取完整的活动详细信息（在 XML 文档 DETAILS 中报告）	ACTIVITY METRICS BASE
MON_GET_BUFFERPOOL 表函数 - 获取缓冲池度量值	DATA OBJECT METRICS BASE
MON_GET_CONNECTION 表函数 - 获取连接度量值	REQUEST METRICS BASE
MON_GET_CONNECTION_DETAILS 表函数 - 获取详细的连接度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE
MON_GET_PKG_CACHE_STMT 表函数 - 获取程序包高速缓存中的 SQL 语句活动度量值	ACTIVITY METRICS BASE
MON_GET_PKG_CACHE_STMT_DETAILS 表函数 - 获取程序包高速缓存条目的详细度量值	ACTIVITY METRICS BASE
MON_GET_SERVICE_SUBCLASS 表函数 - 获取服务子类度量值	REQUEST METRICS BASE
MON_GET_SERVICE_SUBCLASS_DETAILS 表函数 - 获取详细的服务子类度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE
MON_GET_TABLESPACE 表函数 - 获取表空间度量值	DATA OBJECT METRICS BASE
MON_GET_UNIT_OF_WORK 表函数 - 获取工作单元度量值	REQUEST METRICS BASE
MON_GET_UNIT_OF_WORK 表函数 - 获取详细的工作单元度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE
MON_GET_WORKLOAD 表函数 - 获取工作负载度量值	REQUEST METRICS BASE
MON_GET_WORKLOAD_DETAILS 表函数 - 获取详细的工作负载度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE

表 1358. 快照监视信息

快照级别	逻辑数据分组	监视开关
数据库	dbase	缓冲池
表空间	tablespace	缓冲池
缓冲池	bufferpool	缓冲池
应用程序	appl	缓冲池
应用程序	stmt	缓冲池
动态 SQL	dynsql	缓冲池，语句

可将快照监视的计数器重置。

表 1359. 事件监视信息

事件类型	逻辑数据分组	监视开关
活动	event_activity（在 details_xml 文档中报告）	ACTIVITY METRICS BASE
活动	event_activitymetrics	ACTIVITY METRICS BASE
统计信息	event_scstats（在 details_xml 文档中报告）	REQUEST METRICS BASE
统计信息	event_wlstats（在 details_xml 文档中报告）	REQUEST METRICS BASE
工作单元	在 system_metrics 文档中报告。	REQUEST METRICS BASE
数据库	event_db	始终收集
表空间	event_tablespace	始终收集
连接	event_conn	始终收集
语句	event_stmt	始终收集
活动	event_activity	缓冲池，语句
程序包高速缓存	在 activity_metrics 文档中报告。	ACTIVITY METRICS BASE

用法

通过使用 **pool_async_xda_reads** 和 **pool_xda_p_reads** 监视元素，可以计算以同步方式对 XML 存储器对象数据页执行的物理读操作（即，数据库管理器代理程序对 XML 数据执行的物理数据页读操作）的数目。请使用以下公式:

```
pool_xda_p_reads - pool_async_xda_reads
```

通过比较异步读取数与同步读取数的比率，可了解预取程序的执行情况。在调整 **num_ioservers** 配置参数时，此元素会非常有用。

使用 **pool_xda_l_reads**、**pool_xda_p_reads**、**pool_data_l_reads** 和 **pool_data_p_reads** 监视元素来计算缓冲池的数据页命中率。有关更多信息，请参阅第 1379 页的『用于计算缓冲池命中率的公式』

pool_xda_writes -“缓冲池 XDA 数据写次数”监视元素

指示以物理方式将 XML 存储器对象（XDA）的缓冲池数据页写入磁盘的次数。

表 1360. 表函数监视信息

表函数	监视元素收集级别
MON_FORMAT_XML_METRICS_BY_ROW - 获取所有度量值的基于已格式化行的输出	不适用; 报告 XML 文档中作为格式化函数的输入提供的所有元素
MON_GET_ACTIVITY_DETAILS 表函数 - 获取完整的活动详细信息（在 XML 文档 DETAILS 中报告）	ACTIVITY METRICS BASE
MON_GET_BUFFERPOOL 表函数 - 获取缓冲池度量值	DATA OBJECT METRICS BASE
MON_GET_CONNECTION 表函数 - 获取连接度量值	REQUEST METRICS BASE
MON_GET_CONNECTION_DETAILS 表函数 - 获取详细的连接度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE
MON_GET_PKG_CACHE_STMT 表函数 - 获取程序包高速缓存中的 SQL 语句活动度量值	ACTIVITY METRICS BASE
MON_GET_PKG_CACHE_STMT_DETAILS 表函数 - 获取程序包高速缓存条目的详细度量值	ACTIVITY METRICS BASE
MON_GET_SERVICE_SUBCLASS 表函数 - 获取服务子类度量值	REQUEST METRICS BASE
MON_GET_SERVICE_SUBCLASS_DETAILS 表函数 - 获取详细的服务子类度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE
MON_GET_TABLESPACE 表函数 - 获取表空间度量值	DATA OBJECT METRICS BASE
MON_GET_UNIT_OF_WORK 表函数 - 获取工作单元度量值	REQUEST METRICS BASE
MON_GET_UNIT_OF_WORK 表函数 - 获取详细的工作单元度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE
MON_GET_WORKLOAD 表函数 - 获取工作负载度量值	REQUEST METRICS BASE
MON_GET_WORKLOAD_DETAILS 表函数 - 获取详细的工作负载度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE

表 1361. 快照监视信息

快照级别	逻辑数据分组	监视开关
数据库	dbase	缓冲池
表空间	tablespace	缓冲池
缓冲池	bufferpool	缓冲池
应用程序	appl	缓冲池

可将快照监视的计数器重置。

表 1362. 事件监视信息

事件类型	逻辑数据分组	监视开关
活动	event_activity（在 details_xml 文档中报告）	ACTIVITY METRICS BASE
活动	event_activitymetrics	ACTIVITY METRICS BASE
统计信息	event_scstats（在 details_xml 文档中报告）	REQUEST METRICS BASE
统计信息	event_wlstats（在 details_xml 文档中报告）	REQUEST METRICS BASE
工作单元	在 system_metrics 文档中报告。	REQUEST METRICS BASE
数据库	event_db	始终收集
表空间	event_tablespace	始终收集
连接	event_conn	始终收集
程序包高速缓存	在 activity_metrics 文档中报告。	ACTIVITY METRICS BASE

用法

此监视元素帮助您评估通过增加数据库的可用缓冲池页数能否改进性能。对于包含 XML 数据的数据库，您既应该考虑有关 XML 数据的缓冲池页写入数与缓冲池页读取数比率（使用 **pool_xda_writes** 和 **pool_xda_p_reads** 监视元素），也应该考虑有关关系数据类型的缓冲池页写入数与缓冲池页读取数比率（使用 **pool_data_writes** 和 **pool_data_p_reads** 监视元素）。

使用 **pool_xda_l_reads**、**pool_xda_p_reads**、**pool_data_l_reads** 和 **pool_data_p_reads** 监视元素来计算缓冲池的数据页命中率。有关更多信息，请参阅第 1379 页的『用于计算缓冲池命中率的公式』

port_number -“端口号”监视元素

成员正在针对客户机连接侦听的 TCP/IP 端口。

表 1363. 表函数监视信息

表函数	监视元素收集级别
DB_MEMBERS 表函数	始终收集
MON_GET_SERVERLIST 表函数 - 获取成员优先级详细信息	始终收集

post_shrthreshold_hash_joins -“阈值后散列连接数”

排序内存限量算法已限制的散列连接总数。内存受限散列连接是指获得的内存量少于排序内存管理器所请求内存量的散列连接。

表 1364. 快照监视信息

快照级别	逻辑数据分组	监视开关
数据库	dbase	-

可将快照监视的计数器重置。

表 1365. 事件监视信息

事件类型	逻辑数据分组	监视开关
数据库	event_db	始终收集

当从共享排序堆中分配的内存量接近数据库配置参数 *sheapthres_shr* 设置的限制时，就会对散列连接进行内存限制。这种内存限制将显著减少在配置不当的系统中因超过 *sheapthres_shr* 限制导致的内存溢出次数。此元素报告的数据仅反映正在使用从共享排序堆分配的内存的散列连接。

post_shrthreshold_sorts -“共享阈值后排序数”监视元素

排序内存限量算法已限制的排序总数。内存受限排序是指获得的内存量少于排序内存管理器所请求内存量的排序。

表 1366. 表函数监视信息

表函数	监视元素收集级别
MON_FORMAT_XML_METRICS_BY_ROW - 获取所有度量值的基于已格式化行的输出	不适用; 报告 XML 文档中作为格式化函数的输入提供的所有元素
MON_GET_ACTIVITY_DETAILS 表函数 - 获取完整的活动详细信息（在 XML 文档 DETAILS 中报告）	ACTIVITY METRICS BASE
MON_GET_CONNECTION 表函数 - 获取连接度量值	REQUEST METRICS BASE
MON_GET_CONNECTION_DETAILS 表函数 - 获取详细的连接度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE
MON_GET_PKG_CACHE_STMT 表函数 - 获取程序包高速缓存中的 SQL 语句活动度量值	ACTIVITY METRICS BASE
MON_GET_PKG_CACHE_STMT_DETAILS 表函数 - 获取程序包高速缓存条目的详细度量值	ACTIVITY METRICS BASE
MON_GET_SERVICE_SUBCLASS 表函数 - 获取服务子类度量值	REQUEST METRICS BASE
MON_GET_SERVICE_SUBCLASS_DETAILS 表函数 - 获取详细的服务子类度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE

表 1366. 表函数监视信息 (续)

表函数	监视元素收集级别
MON_GET_UNIT_OF_WORK 表函数 - 获取工作单元度量值	REQUEST METRICS BASE
MON_GET_UNIT_OF_WORK 表函数 - 获取详细的工作单元度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE
MON_GET_WORKLOAD 表函数 - 获取工作负载度量值	REQUEST METRICS BASE
MON_GET_WORKLOAD_DETAILS 表函数 - 获取详细的工作负载度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE

表 1367. 快照监视信息

快照级别	逻辑数据分组	监视开关
数据库	dbase	排序

可将快照监视的计数器重置。

表 1368. 事件监视信息

事件类型	逻辑数据分组	监视开关
活动	event_activity（在 details_xml 文档中报告）	ACTIVITY METRICS BASE
活动	event_activitymetrics	ACTIVITY METRICS BASE
统计信息	event_scstats（在 details_xml 文档中报告）	REQUEST METRICS BASE
统计信息	event_wlstats（在 details_xml 文档中报告）	REQUEST METRICS BASE
工作单元	在 system_metrics 文档中报告。	REQUEST METRICS BASE
数据库	event_db	始终收集
程序包高速缓存	在 activity_metrics 文档中报告。	ACTIVITY METRICS BASE

用法

当为排序分配的内存量接近数据库配置参数 **sheapthres_shr** 设置的限制时，就会对排序进行内存限制。这种内存限制将显著减少在配置不当的系统中因超过 **sheapthres_shr** 限制导致的内存溢出次数。此元素报告的数据仅反映正在使用从共享排序堆分配的内存的排序。

post_threshold_hash_joins -“散列连接阈值”

散列连接堆请求因为并行使用共享或专用排序堆空间而受限的总次数。

元素标识

post_threshold_hash_joins

元素类型

计数器

表 1369. 快照监视信息

快照级别	逻辑数据分组	监视开关
数据库管理器	db2	基本

可将快照监视的计数器重置。

用法 如果此值很大（超过 hash_join_overflows 的 5%），那么应增加排序堆阈值。

post_threshold_olap_funcs -“OLAP 函数阈值”监视元素

超过排序堆阈值后请求排序堆的 OLAP 函数数。

表 1370. 快照监视信息

快照级别	逻辑数据分组	监视开关
数据库管理器	db2	基本

可将快照监视的计数器重置。

用法

排序、散列连接和 OLAP 函数是利用排序堆的操作的示例。在正常条件下，数据库管理器将使用 sortheap 配置参数指定的值来分配排序堆。如果分配给排序堆的内存量超过排序堆阈值（sheapthres 配置参数），那么数据库管理器将使用小于 sortheap 配置参数指定值的值来分配后续排序堆。

在达到排序堆阈值后启动的 OLAP 函数可能无法接收最优内存量来执行。

为提高排序、散列连接、OLAP 函数的性能以及系统整体性能，请修改排序堆阈值和排序堆大小配置参数。

如果此元素的值过高，那么请增加排序堆阈值（sheapthres）。

post_threshold_peas -“部分提前聚集阈值”监视元素

因为超过排序堆阈值而导致部分提前聚集操作接收到的内存量少于所请求内存量的次数。

表 1371. 表函数监视信息

表函数	监视元素收集级别
MON_FORMAT_XML_METRICS_BY_ROW - 获取所有度量值的基于已格式化行的输出	不适用；报告 XML 文档中作为格式化函数的输入提供的所有元素
MON_GET_ACTIVITY_DETAILS 表函数 - 获取完整的活动详细信息	ACTIVITY METRICS BASE
MON_GET_CONNECTION 表函数 - 获取连接度量值	REQUEST METRICS BASE

表 1371. 表函数监视信息 (续)

表函数	监视元素收集级别
MON_GET_CONNECTION_DETAILS 表函数 - 获取详细的连接度量值	REQUEST METRICS BASE
MON_GET_PKG_CACHE_STMT 表函数 - 获取程序包高速缓存中的 SQL 语句活动度量值	ACTIVITY METRICS BASE
MON_GET_PKG_CACHE_STMT_DETAILS - 获取程序包高速缓存条目的详细度量值	ACTIVITY METRICS BASE
MON_GET_SERVICE_SUBCLASS 表函数 - 获取服务子类度量值	REQUEST METRICS BASE
MON_GET_SERVICE_SUBCLASS_DETAILS 表函数 - 获取详细的服务子类度量值	REQUEST METRICS BASE
MON_GET_UNIT_OF_WORK 表函数 - 获取工作单元度量值	REQUEST METRICS BASE
MON_GET_UNIT_OF_WORK_DETAILS 表函数 - 获取详细的工作单元度量值	REQUEST METRICS BASE
MON_GET_WORKLOAD 表函数 - 获取工作负载度量值	REQUEST METRICS BASE
MON_GET_WORKLOAD_DETAILS 表函数 - 获取详细的工作负载度量值	REQUEST METRICS BASE

表 1372. 表函数监视信息

表函数	监视元素收集级别
MON_FORMAT_XML_METRICS_BY_ROW - 获取所有度量值的基于已格式化行的输出	不适用; 报告 XML 文档中作为格式化函数的输入提供的所有元素
MON_GET_ACTIVITY_DETAILS 表函数 - 获取完整的活动详细信息（在 XML 文档 DETAILS 中报告）	ACTIVITY METRICS BASE
MON_GET_CONNECTION 表函数 - 获取连接度量值	REQUEST METRICS BASE
MON_GET_CONNECTION_DETAILS 表函数 - 获取详细的连接度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE
MON_GET_PKG_CACHE_STMT 表函数 - 获取程序包高速缓存中的 SQL 语句活动度量值	ACTIVITY METRICS BASE
MON_GET_PKG_CACHE_STMT_DETAILS 表函数 - 获取程序包高速缓存条目的详细度量值（在 XML 文档 DETAILS 中报告）	ACTIVITY METRICS BASE
MON_GET_SERVICE_SUBCLASS 表函数 - 获取服务子类度量值	REQUEST METRICS BASE
MON_GET_SERVICE_SUBCLASS_DETAILS 表函数 - 获取详细的服务子类度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE
MON_GET_UNIT_OF_WORK 表函数 - 获取工作单元度量值	REQUEST METRICS BASE

表 1372. 表函数监视信息 (续)

表函数	监视元素收集级别
MON_GET_UNIT_OF_WORK 表函数 - 获取详细的工作单元度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE
MON_GET_WORKLOAD 表函数 - 获取工作负载度量值	REQUEST METRICS BASE
MON_GET_WORKLOAD_DETAILS 表函数 - 获取详细的工作负载度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE

表 1373. 事件监视信息

事件类型	逻辑数据分组	监视开关
活动	event_activity（在 details_xml 文档中报告）	ACTIVITY METRICS BASE
统计信息	event_scstats（在 details_xml 文档中报告）	REQUEST METRICS BASE
统计信息	event_wlstats（在 details_xml 文档中报告）	REQUEST METRICS BASE
工作单元	在 system_metrics 文档中报告。	REQUEST METRICS BASE
连接	event_conn	-
语句	event_stmt	-
事务	event_xact	-
程序包高速缓存	在 activity_metrics 文档中报告。	ACTIVITY METRICS BASE

用法

将此元素与 **total_peas** 监视元素配合使用来确定大多数时间部分提前聚集操作是否获取足够的排序堆内存。如果 **post_thresold_peas** 监视元素与 **total_peas** 监视元素的比率很高，那么数据库性能可能欠佳。应考虑增加排序堆大小和/或排序堆阈值。

post_threshold_peds -“部分提前相异数阈值”监视元素

因为超过排序堆阈值而导致部分提前相异操作接收到的内存量少于所请求内存量的次数。

表 1374. 表函数监视信息

表函数	监视元素收集级别
MON_FORMAT_XML_METRICS_BY_ROW - 获取所有度量值的基于已格式化行的输出	不适用；报告 XML 文档中作为格式化函数的输入提供的所有元素
MON_GET_ACTIVITY_DETAILS 表函数 - 获取完整的活动详细信息（在 XML 文档 DETAILS 中报告）	ACTIVITY METRICS BASE
MON_GET_CONNECTION 表函数 - 获取连接度量值	REQUEST METRICS BASE

表 1374. 表函数监视信息 (续)

表函数	监视元素收集级别
MON_GET_CONNECTION_DETAILS 表函数 - 获取详细的连接度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE
MON_GET_PKG_CACHE_STMT 表函数 - 获取程序包高速缓存中的 SQL 语句活动度量值	ACTIVITY METRICS BASE
MON_GET_PKG_CACHE_STMT_DETAILS 表函数 - 获取程序包高速缓存条目的详细度量值（在 XML 文档 DETAILS 中报告）	ACTIVITY METRICS BASE
MON_GET_SERVICE_SUBCLASS 表函数 - 获取服务子类度量值	REQUEST METRICS BASE
MON_GET_SERVICE_SUBCLASS_DETAILS 表函数 - 获取详细的服务子类度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE
MON_GET_UNIT_OF_WORK 表函数 - 获取工作单元度量值	REQUEST METRICS BASE
MON_GET_UNIT_OF_WORK 表函数 - 获取详细的工作单元度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE
MON_GET_WORKLOAD 表函数 - 获取工作负载度量值	REQUEST METRICS BASE
MON_GET_WORKLOAD_DETAILS 表函数 - 获取详细的工作负载度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE

表 1375. 事件监视信息

事件类型	逻辑数据分组	监视开关
活动	event_activity（在 details_xml 文档中报告）	ACTIVITY METRICS BASE
统计信息	event_scstats（在 details_xml 文档中报告）	REQUEST METRICS BASE
统计信息	event_wlstats（在 details_xml 文档中报告）	REQUEST METRICS BASE
工作单元	在 system_metrics 文档中报告。	REQUEST METRICS BASE
连接	event_conn	-
语句	event_stmt	-
事务	event_xact	-
程序包高速缓存	在 activity_metrics 文档中报告。	ACTIVITY METRICS BASE

用法

将此元素与 **total_peds** 监视元素配合使用来确定大多数时间部分提前相异操作是否获取足够的排序堆内存。如果 **post_thresold_peds** 监视元素与 **total_peds** 监视元素的比率很高，那么数据库性能可能欠佳。应考虑增加排序堆大小和/或排序堆阈值。

post_threshold_sorts -“超出阈值后的排序次数”监视元素

超过排序堆阈值后请求堆的排序数。

表 1376. 表函数监视信息

表函数	监视元素收集级别
MON_FORMAT_XML_METRICS_BY_ROW - 获取所有度量值的基于已格式化行的输出	不适用; 报告 XML 文档中作为格式化函数的输入提供的所有元素
MON_GET_ACTIVITY_DETAILS 表函数 - 获取完整的活动详细信息（在 XML 文档 DETAILS 中报告）	ACTIVITY METRICS BASE
MON_GET_CONNECTION 表函数 - 获取连接度量值	REQUEST METRICS BASE
MON_GET_CONNECTION_DETAILS 表函数 - 获取详细的连接度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE
MON_GET_PKG_CACHE_STMT 表函数 - 获取程序包高速缓存中的 SQL 语句活动度量值	ACTIVITY METRICS BASE
MON_GET_PKG_CACHE_STMT_DETAILS 表函数 - 获取程序包高速缓存条目的详细度量值	ACTIVITY METRICS BASE
MON_GET_SERVICE_SUBCLASS 表函数 - 获取服务子类度量值	REQUEST METRICS BASE
MON_GET_SERVICE_SUBCLASS_DETAILS 表函数 - 获取详细的服务子类度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE
MON_GET_UNIT_OF_WORK 表函数 - 获取工作单元度量值	REQUEST METRICS BASE
MON_GET_UNIT_OF_WORK 表函数 - 获取详细的工作单元度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE
MON_GET_WORKLOAD 表函数 - 获取工作负载度量值	REQUEST METRICS BASE
MON_GET_WORKLOAD_DETAILS 表函数 - 获取详细的工作负载度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE

表 1377. 快照监视信息

快照级别	逻辑数据分组	监视开关
数据库管理器	db2	排序

可将快照监视的计数器重置。

表 1378. 事件监视信息

事件类型	逻辑数据分组	监视开关
活动	event_activity（在 details_xml 文档中报告）	ACTIVITY METRICS BASE
活动	event_activitymetrics	ACTIVITY METRICS BASE

表 1378. 事件监视信息 (续)

事件类型	逻辑数据分组	监视开关
统计信息	event_scstats（在 details_xml 文档中报告）	REQUEST METRICS BASE
统计信息	event_wlstats（在 details_xml 文档中报告）	REQUEST METRICS BASE
工作单元	在 system_metrics 文档中报告。	REQUEST METRICS BASE
程序包高速缓存	在 activity_metrics 文档中报告。	ACTIVITY METRICS BASE

用法

在正常条件下，数据库管理器将使用 **sortheap** 配置参数指定的值来分配排序堆。如果分配给排序堆的内存量超过排序堆阈值（**sheapthres** 配置参数），那么数据库管理器将使用小于**sortheap**配置参数指定值的值来分配排序堆。

系统上每个活动排序分配内存可能导致排序占用过多系统可用内存。在达到排序堆阈值后启动的排序可能无法接收最优内存量来执行，但整个系统将因此而获益。通过修改排序堆阈值和排序堆大小配置参数，排序操作性能和整体系统性能可以得到改进。如果此元素的值过高，那么可以:

- 提高排序堆阈值（**sheapthres**）或者
- 通过 SQL 查询更改将应用程序调整为使用数量较少范围较小的排序。

prefetch_wait_time -“等待预取的时间”监视元素

应用程序等待 I/O 服务器（预取程序）完成将页装入到缓冲池中的操作所花的时间。 此值以毫秒计。

表 1379. 表函数监视信息

表函数	监视元素收集级别
MON_FORMAT_XML_METRICS_BY_ROW - 获取所有度量值的基于已格式化行的输出	不适用; 报告 XML 文档中作为格式化函数的输入提供的所有元素
MON_FORMAT_XML_TIMES_BY_ROW - - 获取基于已格式化行的组合层次结构等待时间和处理时间	不适用; 报告 XML 文档中作为格式化函数的输入提供的所有元素
MON_FORMAT_XML_WAIT_TIMES_BY _ROW - 获取基于已格式化行的输出用于等待的次数	不适用; 报告 XML 文档中作为格式化函数的输入提供的所有元素
MON_GET_ACTIVITY_DETAILS 表函数 - 获取完整的活动详细信息（在 XML 文档 DETAILS 中报告）	ACTIVITY METRICS BASE
MON_GET_BUFFERPOOL 表函数 - 获取缓冲池度量值	DATA OBJECT METRICS BASE
MON_GET_CONNECTION 表函数 - 获取连接度量值	REQUEST METRICS BASE

表 1379. 表函数监视信息 (续)

表函数	监视元素收集级别
MON_GET_CONNECTION_DETAILS 表函数 - 获取详细的连接度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE
MON_GET_PKG_CACHE_STMT 表函数 - 获取程序包高速缓存中的 SQL 语句活动度量值	ACTIVITY METRICS BASE
MON_GET_PKG_CACHE_STMT_DETAILS 表函数 - 获取程序包高速缓存条目的详细度量值（在 XML 文档 DETAILS 中报告）	ACTIVITY METRICS BASE
MON_GET_SERVICE_SUBCLASS 表函数 - 获取服务子类度量值	REQUEST METRICS BASE
MON_GET_SERVICE_SUBCLASS_DETAILS 表函数 - 获取详细的服务子类度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE
MON_GET_TABLESPACE 表函数 - 获取表空间度量值	DATA OBJECT METRICS BASE
MON_GET_UNIT_OF_WORK 表函数 - 获取工作单元度量值	REQUEST METRICS BASE
MON_GET_UNIT_OF_WORK 表函数 - 获取详细的工作单元度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE
MON_GET_WORKLOAD 表函数 - 获取工作负载度量值	REQUEST METRICS BASE
MON_GET_WORKLOAD_DETAILS 表函数 - 获取详细的工作负载度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE

表 1380. 快照监视信息

快照级别	逻辑数据分组	监视开关
数据库	dbase	缓冲池
应用程序	appl	缓冲池

表 1381. 事件监视信息

事件类型	逻辑数据分组	监视开关
活动	event_activity（在 details_xml 文档中报告）	ACTIVITY METRICS BASE
程序包高速缓存	pkgcache（在 metrics.xml 文档中报告） pkgcache_metrics	ACTIVITY METRICS BASE
工作单元	uow（在 metrics.xml 文档中报告） uow_metrics	REQUEST METRICS BASE
统计信息	event_scstats（在 details_xml 文档中报告）	REQUEST METRICS BASE

表 1381. 事件监视信息 (续)

事件类型	逻辑数据分组	监视开关
统计信息	event_wlstats（在 details_xml 文档中报告）	REQUEST METRICS BASE
数据库	event_db	Bufferpool
连接	event_db	Bufferpool

用法 可通过更改 I/O 服务器数目和 I/O 服务器大小来使用此元素进行试验。

prefetch_waits -“预取程序等待计数”监视元素

等待 I/O 服务器（预取程序）完成将页装入到缓冲池中的操作的次数。

表 1382. 表函数监视信息

表函数	监视元素收集级别
MON_FORMAT_XML_METRICS_BY_ROW - 获取所有度量值的基于已格式化行的输出	不适用; 报告 XML 文档中作为格式化函数的输入提供的所有元素
MON_GET_ACTIVITY_DETAILS 表函数 - 获取完整的活动详细信息（在 XML 文档 DETAILS 中报告）	ACTIVITY METRICS BASE
MON_GET_BUFFERPOOL 表函数 - 获取缓冲池度量值	DATA OBJECT METRICS BASE
MON_GET_CONNECTION 表函数 - 获取连接度量值	REQUEST METRICS BASE
MON_GET_CONNECTION_DETAILS 表函数 - 获取详细的连接度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE
MON_GET_PKG_CACHE_STMT 表函数 - 获取程序包高速缓存中的 SQL 语句活动度量值	ACTIVITY METRICS BASE
MON_GET_PKG_CACHE_STMT_DETAILS 表函数 - 获取程序包高速缓存条目的详细度量值（在 XML 文档 DETAILS 中报告）	ACTIVITY METRICS BASE
MON_GET_SERVICE_SUBCLASS 表函数 - 获取服务子类度量值	REQUEST METRICS BASE
MON_GET_SERVICE_SUBCLASS_DETAILS 表函数 - 获取详细的服务子类度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE
MON_GET_TABLESPACE 表函数 - 获取表空间度量值	DATA OBJECT METRICS BASE
MON_GET_UNIT_OF_WORK 表函数 - 获取工作单元度量值	REQUEST METRICS BASE
MON_GET_UNIT_OF_WORK 表函数 - 获取详细的工作单元度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE
MON_GET_WORKLOAD 表函数 - 获取工作负载度量值	REQUEST METRICS BASE

表 1382. 表函数监视信息 (续)

表函数	监视元素收集级别
MON_GET_WORKLOAD_DETAILS 表函数 - 获取详细的工作负载度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE

表 1383. 事件监视信息

事件类型	逻辑数据分组	监视开关
活动	event_activity（在 details_xml 文档中报告） event_activitymetrics	ACTIVITY METRICS BASE
程序包高速缓存	pkgcache（在 metrics.xml 文档中报告） pkgcache_metrics	ACTIVITY METRICS BASE
工作单元	uow（在 metrics.xml 文档中报告） uow_metrics	REQUEST METRICS BASE
统计信息	event_scstats（在 details_xml 文档中报告）	REQUEST METRICS BASE
统计信息	event_wlstats（在 details_xml 文档中报告）	REQUEST METRICS BASE

prep_time -“编译时间”监视元素

编译 SQL 语句所需的时间（以毫秒计；要求活动是 SQL 语句；否则，值为 0）。

表 1384. 表函数监视信息

表函数	监视元素收集级别
MON_GET_PKG_CACHE_STMT 表函数 - 获取程序包高速缓存中的 SQL 语句活动度量值	ACTIVITY METRICS BASE
MON_GET_PKG_CACHE_STMT_DETAILS 表函数 - 获取程序包高速缓存条目的详细度量值	ACTIVITY METRICS BASE

表 1385. 事件监视信息

事件类型	逻辑数据分组	监视开关
活动	event_activity	始终收集
程序包高速缓存	-	COLLECT BASE DATA

用法

prep_time 监视元素指示编译 SQL 语句所花的时间；如果此活动是 SQL 语句，那么此监视元素指示第一次将此语句引入 DB2 程序包高速缓存的时间。此编译时间并不是活动生存期的一部分，也不表示在该语句的特定调用期间所花的时间（如果在执行该调用之前已经将该语句高速缓存在程序包高速缓存中）。

prep_time_best -“语句最短编译时间”监视元素

编译特定 SQL 语句所需的最短时间（以毫秒计）。

表 1386. 快照监视信息

快照级别	逻辑数据分组	监视开关
动态 SQL	dynsql	基本

用法

将此值与 **prep_time_worst** 配合使用来标识编译成本高昂的 SQL 语句。

prep_time_worst -“语句最长编译时间”监视元素

编译特定 SQL 语句所需的最长时间（以毫秒计）。

表 1387. 快照监视信息

快照级别	逻辑数据分组	监视开关
动态 SQL	dynsql	基本

用法

将此值与 **prep_time_best** 配合使用来标识编译成本高昂的 SQL 语句。

prev_uow_stop_time -“上一个工作单元完成时间戳记”

这是工作单元的完成时间。

元素标识

prev_uow_stop_time

元素类型

时间戳记

表 1388. 快照监视信息

快照级别	逻辑数据分组	监视开关
应用程序	appl	工作单元，时间戳记
DCS 应用程序	dcs_appl	工作单元，时间戳记

表 1389. 事件监视信息

事件类型	逻辑数据分组	监视开关
事务	event_xact	始终收集

用法 可将此元素与 *uow_stop_time* 配合使用来计算 COMMIT/ROLLBACK 点之间的耗用时间总计，还可将其与 *uow_start_time* 配合使用来计算在工作单元之间的应用程序上耗用的时间。以下其中一个操作的时间:

- 对于当前在工作单元中的应用程序，这是最新工作单元的完成时间。

- 对于当前不在 工作单元 中的应用程序（该应用程序已完成某个 工作单元，但尚未启动新的 工作单元），这是刚刚完成 工作单元 之前完成的最后一个工作单元 的停止时间。刚刚完成的工作单元的时间用 uow_stop_time 指示。
- 对于第一个工作单元中的应用程序，这是数据库连接请求的完成时间。

priority -“优先级值”监视元素

描述要处理工作的成员的相对能力。此值越高，客户机应分配给该成员的工作越多。

表 1390. 表函数监视信息

表函数	监视元素收集级别
MON_GET_SERVERLIST 表函数 - 获取成员优先级详细信息	始终收集

使用说明

- 此监视器元素代表成员的相对装入，也称为权重。例如，如果成员 A 具有优先级值 80，成员 B 具有优先级值 40，那么这表示成员 A 应接收的工作量是给成员 B 的工作量的二倍。
- 此值不代表百分比。
- 此监视器元素的最大值是 100。

priv_workspace_num_overflows -“专用工作空间溢出数”

专用工作空间溢出其分配内存边界的次数。

注: 不推荐使用此监视元素。使用此监视元素不会生成错误。但是不会返回有效值。建议不要再使用此监视元素，将来的发行版中可能会将其除去。

表 1391. 快照监视信息

快照级别	逻辑数据分组	监视开关
数据库	dbase	基本
应用程序	appl	基本

可将快照监视的计数器重置。

表 1392. 事件监视信息

事件类型	逻辑数据分组	监视开关
数据库	event_db	始终收集
连接	event_conn	始终收集

用法 将此元素与 priv_workspace_size_top 配合使用来确定专用工作空间是否需要增加大小以避免溢出。专用工作空间的溢出可能导致性能下降，以及从代理程序专用内存分配的其他堆出现内存不足错误。

在数据库级别，报告的元素将来自报告为具有相同最大专用工作空间大小的元素的专用工作空间。在应用程序级别，此项是为当前应用程序提供服务的每个代理程序的工作空间的溢出数。

priv_workspace_section_inserts -“专用工作空间节插入数”

应用程序在专用工作空间中插入 SQL 节的次数。

注: 不推荐使用此监视元素。使用此监视元素不会生成错误。但是不会返回有效值。建议不要再使用此监视元素，将来的发行版中可能会将其除去。

表 1393. 快照监视信息

快照级别	逻辑数据分组	监视开关
数据库	dbase	基本
应用程序	appl	基本

可将快照监视的计数器重置。

表 1394. 事件监视信息

事件类型	逻辑数据分组	监视开关
数据库	event_db	始终收集
连接	event_conn	始终收集

用法 可执行部分的工作副本存储在专用工作空间中。

此计数器指示副本何时不可用并且必须插入。在数据库级别，此项是针对数据库的所有专用工作空间中的每个应用程序进行的所有插入的累积总数。在应用程序级别，此项是针对此应用程序的专用工作空间中的所有节的所有插入的累积总数。

在代理程序要与不同应用程序相关联的集中器环境中，如果新的代理程序在专用工作空间中没有必需的节可用，那么可能需要附加专用工作空间插入。

priv_workspace_section_lookups -“专用工作空间节查询数”

应用程序在其代理程序的专用工作空间中对 SQL 节进行查询的次数。

注: 不推荐使用此监视元素。使用此监视元素不会生成错误。但是不会返回有效值。建议不要再使用此监视元素，将来的发行版中可能会将其除去。

表 1395. 快照监视信息

快照级别	逻辑数据分组	监视开关
数据库	dbase	基本
应用程序	appl	基本

可将快照监视的计数器重置。

表 1396. 事件监视信息

事件类型	逻辑数据分组	监视开关
数据库	event_db	始终收集
连接	event_conn	始终收集

用法 每个应用程序都可以访问为其工作的代理程序的专用工作空间。

此计数器指示为找到应用程序的特定节而访问专用工作空间的次数。在数据库级别，此项是针对数据库的所有专用工作空间中的每个应用程序进行的所有查询的累积总数。在应用程序级别，此项是针对此应用程序的专用工作空间中的所有节的所有查询的累积总数。

可将此元素与“专用工作空间节插入数”一起使用来调整专用工作空间的大小。专用工作空间的大小由 applheapsz 配置参数控制。

priv_workspace_size_top -“最大专用工作空间大小”

专用工作空间达到的最大大小。

注: 不推荐使用此监视元素。使用此监视元素不会生成错误。但是不会返回有效值。建议不要再使用此监视元素，将来的发行版中可能会将其除去。

表 1397. 快照监视信息

快照级别	逻辑数据分组	监视开关
数据库	dbase	基本
应用程序	appl	基本

表 1398. 事件监视信息

事件类型	逻辑数据分组	监视开关
数据库	event_db	始终收集
连接	event_conn	始终收集

用法 每个代理程序都有专用工作空间，它所服务的应用程序对该工作空间具有访问权。此元素指示专用工作空间中提供服务的任何代理程序所需的最多字节数。在数据库级别，此项是连接至当前数据库的所有代理程序在所有专用工作空间中必需的最多字节数。 在应用程序级别，此项是为当前应用程序提供服务的所有代理程序的专用工作空间中的最大大小。

专用工作空间溢出时，将会临时从代理程序专用内存的其他实体借出内存。这可能导致这些实体出现内存不足错误，也可能导致性能下降。可通过增加 APPLHEAPSZ 来降低溢出的机率。

product_name -“产品名称”

正在运行的 DB2 实例的版本的详细信息。

表 1399. 快照监视信息

快照级别	逻辑数据分组	监视开关
数据库管理器	db2	基本

progress_completed_units -“完成的进度工作单元数”

当前阶段完成的工作单元数。

表 1400. 快照监视信息

快照级别	逻辑数据分组	监视开关
数据库管理器	progress	基本

此元素的值通常会在实用程序操作时递增。此元素总是小于或等于 *progress_total_units*（如果同时定义了这两个元素）。

注:

1. 可能并非所有实用程序都包含此元素。
2. 此元素在 *progress_work_metric* 监视元素中以单元表示。

用法 使用此元素来确定某个阶段完成的工作量。此元素本身可用来监视正在运行的实用程序的活动。实用程序执行时，此元素应不断递增。如果 *progress_completed_units* 在很长一段时间内未能递增，那么实用程序可能已停止。

如果定义了 *progress_total_units*，那么此元素可用来计算完成工作的百分比:

```
percentage complete = progress_completed_units / progress_total_units* 100
```

progress_description -“进度描述”

描述工作阶段。

表 1401. 快照监视信息

快照级别	逻辑数据分组	监视开关
数据库管理器	progress	基本

Load 实用程序的示例值包括:

- DELETE
- LOAD
- REDO

用法 使用此元素来获取对某个阶段的一般描述。

progress_list_attr -“当前进度列表属性”

此元素描述如何解释进度元素列表。

表 1402. 快照监视信息

快照级别	逻辑数据分组	监视开关
数据库管理器	progress list	基本

用法

此元素的值是下列其中一个常量:

- SQLM_ELM_PROGRESS_LIST_ATTR_SERIAL - 列表中的元素将解释为一组串行阶段，这意味着第一次更新元素 *n*+1 的已完成工作之前，已完成工作必须等于元素 *n* 的总工作。此属性用于描述由一组串行阶段组成的任务的进度，其中串行阶段意味着必须彻底完成一个阶段才能开始下一个阶段。
- SQLM_ELM_PROGRESS_LIST_ATTR_CONCURRENT - 进度列表中的任何元素可以随时更新。

使用此元素来确定更新进度列表的各个元素的方式。

progress_list_cur_seq_num -“当前进度列表序号”

如果实用程序包含多个顺序阶段，那么此元素显示当前阶段的编号。

表 1403. 快照监视信息

快照级别	逻辑数据分组	监视开关
数据库管理器	progress_list	基本

用法 使用此元素来确定多阶段实用程序的当前阶段。请参阅『progress_seq_num -“进度序号”』。

progress_seq_num -“进度序号”

阶段号。

注: 仅对由多个执行阶段组成的实用程序显示阶段号。

表 1404. 快照监视信息

快照级别	逻辑数据分组	监视开关
数据库管理器	progress	基本

用法 使用此元素来确定多阶段实用程序内的阶段顺序。该实用程序将按进度序号递增顺序来顺序执行各个阶段。可通过将 *progress_seq_num* 与 *progress_list_current_seq_num* 的值相匹配来找到多阶段实用程序的当前阶段。

progress_start_time -“进度开始时间”

表示阶段开始的时间戳记。

表 1405. 快照监视信息

快照级别	逻辑数据分组	监视开关
数据库管理器	progress	基本

用法 使用此元素来确定阶段开始的时间。如果该阶段尚未开始，那么省略此元素。

progress_total_units -“进度工作单元总数”

为了完成某个阶段而执行的工作总量。

表 1406. 快照监视信息

快照级别	逻辑数据分组	监视开关
数据库管理器	progress	基本

某些实用程序可能无法确定总工作量，所以它们将持续更新此元素。其他实用程序可能无法估计总工作量，所以可能完全省略此元素。

此元素在 *progress_work_metric* 监视元素中以单元表示。

用法 使用此元素来确定某个阶段的总工作量。将此元素与 *progress_completed_units* 配合使用来计算某个阶段完成的工作百分比:

```
percentage complete = progress_completed_units / progress_total_units* 100
```

progress_work_metric -“进度工作度量”

解释 *progress_total_units* 和 *progress_completed_units* 元素的度量。

表 1407. 快照监视信息

快照级别	逻辑数据分组	监视开关
数据库管理器	progress	基本

示例值包括:

- SQLM_WORK_METRIC_BYTES
- SQLM_WORK_METRIC_EXTENTS

注:

1. 可能并非所有实用程序都包含此元素。
2. 此元素的值可在 sqlmon.h 中找到。

用法 使用此元素的值来确定用作报告度量的 *progress_total_units* 和 *progress_completed_units*。

pseudo_deletes -“伪删除数”监视元素

已经标记为“伪删除”的密钥数。

表 1408. 表函数监视信息

表函数	监视元素收集级别
MON_GET_INDEX 表函数 - 获取索引度量值	ACTIVITY METRICS BASE

pseudo_empty_pages -“伪空页数”监视元素

已标识为伪空页的页数。伪空页就是已将所有键伪删除的页。

表 1409. 表函数监视信息

表函数	监视元素收集级别
MON_GET_INDEX 表函数 - 获取索引度量值	ACTIVITY METRICS BASE

用法

注: 此监视元素并不报告伪空页的当前数目。

query_actual_degree -“实际运行时分区内并行度”监视元素

在语句级别、活动级别、事务级别或工作负载级别报告的实际运行时分区内并行度。状态为已启用或已禁用。

表 1410. 表函数监视信息

表函数	监视元素收集级别
MON_GET_ACTIVITY_DETAILS 表函数 - 获取完整的活动详细信息（在 XML 文档 DETAILS 中报告）	ACTIVITY METRICS BASE

表 1411. 事件监视信息

事件类型	逻辑数据分组	监视开关
活动	-	始终收集
锁定		

query_card_estimate -“行查询数估计”

查询将返回的行数估计。

表 1412. 快照监视信息

快照级别	逻辑数据分组	监视开关
应用程序	stmt	语句
DCS 语句	dcs_stmt	语句
活动	event_activity	-

用法 SQL 编译器进行的此估计可与运行时实际结果进行比较。

在监视DB2 Connect时，此元素还会返回有关下列 SQL 语句的信息。

- INSERT、UPDATE 和 DELETE

  指示受影响的行数。

- PREPARE

  估计将返回的行数。仅当 DRDA 服务器为 DB2 Database for Linux, UNIX, and Windows、DB2 VM 和 VSE 版或 DB2 OS/400® 版时才收集此信息。

- FETCH

  设置为访存行数。仅当 DRDA 服务器为 DB2 OS/400 版时，才会收集此信息。

如果不对 DRDA 服务器收集信息，那么该元素设置为零。

query_cost_estimate -“查询估算成本”监视元素

由 SQL 编译器确定的查询估算成本。此值以 timeron 计。

表 1413. 表函数监视信息

表函数	监视元素收集级别
MON_GET_ACTIVITY_DETAILS 表函数 - 获取完整的活动详细信息（在 XML 文档 DETAILS 中报告）	ACTIVITY METRICS BASE
MON_GET_PKG_CACHE_STMT 表函数 - 获取程序包高速缓存中的 SQL 语句活动度量值	ACTIVITY METRICS BASE
MON_GET_PKG_CACHE_STMT_DETAILS 表函数 - 获取程序包高速缓存条目的详细度量值	ACTIVITY METRICS BASE
WLM_GET_WORKLOAD_OCCURRENCE_ACTIVITIES 表函数 - 返回活动列表	ACTIVITY METRICS BASE

表 1414. 快照监视信息

快照级别	逻辑数据分组	监视开关
应用程序	stmt	语句
DCS 语句	dcs_stmt	语句
活动	event_activity	-
程序包高速缓存	-	COLLECT BASE DATA

用法

此监视元素允许使实际运行时值与编译时估计值相关。

在监视DB2 Connect时，此元素还会返回有关下列 SQL 语句的信息。

- PREPARE

  表示预编译 SQL 语句的相对成本。
- FETCH

  包含已检索行的长度。仅当 DRDA 服务器为 DB2 OS/400 版时，才会收集此信息。

如果不对 DRDA 服务器收集信息，那么该元素设置为零。

注: 如果 DRDA 服务器为 DB2 OS/390® 版和 z/OS 版，那么此估计可能高于 2**32 - 1（可通过不带符号的长整型变量表示的最大整数）。在此情况下，监视器对此元素返回的值将为 2**32 - 1。

query_data_tag_list -“估算的查询数据标记列表”监视元素

由编译器估算的将在语句中引用的数据标记值的逗号分隔列表。如果编译器预测该语句将访问其数据表空间定义非零数据标记属性的表，那么该表列中将包含数据标记值。该列表不包含重复值。

表 1415. 表函数监视信息

表函数	监视元素收集命令和级别
MON_GET_ACTIVITY_DETAILS 表函数 - 获取完整的活动详细信息	ACTIVITY METRICS BASE
MON_GET_PKG_CACHE_STMT 表函数 - 获取程序包高速缓存中的 SQL 语句活动度量值	ACTIVITY METRICS BASE
MON_GET_PKG_CACHE_STMT_DETAILS 表函数 - 获取程序包高速缓存条目的详细度量值	ACTIVITY METRICS BASE

表 1416. 事件监视信息

事件类型	逻辑数据分组	监视开关
活动	event_activity	ACTIVITY METRICS BASE
程序包高速缓存	-	COLLECT BASE DATA

用法说明

如果查询访问的数据表空间都未定义数据标记，那么此列表为空。

queue_assignments_total -“队列分配总次数”监视元素

自从上次重置后任何连接或活动被分配到此阈值队列的次数。

表 1417. 表函数监视信息

表函数	监视元素收集级别
WLM_GET_QUEUE_STATS 表函数 - 返回阈值队列统计信息	ACTIVITY METRICS BASE

表 1418. 事件监视信息

事件类型	逻辑数据分组	监视开关
统计信息	event_qstats	-

用法

此元素可用于确定，任何连接或活动在由统计信息收集时间间隔确定的给定时间段内在此特定队列中进行排队的次数。这样可帮助确定队列阈值的有效性。

queue_start_time -“队列开始时间戳记”监视元素

应用程序开始在队列中等待获取阈值凭单的日期和时间。

表 1419. 事件监视信息

事件类型	逻辑数据分组	监视开关
锁定	lock_participants	

queue_size_top -“最大队列大小”监视元素

自最后一次重置后达到的最大队列大小。

表 1420. 表函数监视信息

表函数	监视元素收集级别
WLM_GET_QUEUE_STATS 表函数 - 返回阈值队列统计信息	ACTIVITY METRICS BASE

表 1421. 事件监视信息

事件类型	逻辑数据分组	监视开关
统计信息	event_qstats	-

用法

使用此元素来测量队列阈值的有效性以及检测队列何时变得过长。

queue_time_total -“总队列时间”监视元素

自最后一次重置以后，此队列中的所有连接或活动在队列中所花的总时间。单位为毫秒。

表 1422. 表函数监视信息

表函数	监视元素收集级别
WLM_GET_QUEUE_STATS 表函数 - 返回阈值队列统计信息	ACTIVITY METRICS BASE

表 1423. 事件监视信息

事件类型	逻辑数据分组	监视开关
统计信息	event_qstats	始终收集

此元素用来测量活动队列阈值的有效性以及检测活动队列何时变得过长。

使用说明

在统计信息收集时间间隔结束时，并不会重置 **queue_time_total**。如果所使用的 **queue_time_total** 跨越了多个时间间隔，那么它可能大于 **wlm_collect_int** 与 **queue_size_top** 的乘积。

queued_agents -“已排队阈值代理程序数”监视元素

当前正在阈值队列中进行排队的代理程序的总数。

表 1424. 事件监视信息

事件类型	逻辑数据分组	监视开关
锁定	lock_participants	

quiescer_agent_id -“停顿者代理程序标识”

具有停顿状态的代理程序的代理程序标识。

元素标识

quiescer_agent_id

元素类型

信息

表 1425. 快照监视信息

快照级别	逻辑数据分组	监视开关
表空间	tablespace_quiescer	基本

用法 将此元素与 quiescer_auth_id 配合使用，以确定停顿表空间的人员。

quiescer_auth_id -“停顿者用户授权标识”

具有停顿状态的用户的授权标识。

元素标识

quiescer_auth_id

元素类型

信息

表 1426. 快照监视信息

快照级别	逻辑数据分组	监视开关
表空间	tablespace_quiescer	基本

用法 使用此元素来确定停顿表空间的人员。

quiescer_obj_id -“停顿者对象标识”

导致表空间停顿的对象的对象标识。

元素标识

quiescer_obj_id

元素类型

信息

表 1427. 快照监视信息

快照级别	逻辑数据分组	监视开关
表空间	tablespace_quiescer	基本

用法 将此元素与 quiescer_ts_id 和 quiescer_auth_id 配合使用以确定停顿表空间的人员。此元素的值与视图 SYSCAT.TABLES 的列 TABLEID 中的值相匹配。

quiescer_state -“停顿者状态”

要完成的停顿类型（如“SHARE”、“INTENT TO UPDATE”或“EXCLUSIVE”）。

元素标识

quiescer_state

元素类型

信息

表 1428. 快照监视信息

快照级别	逻辑数据分组	监视开关
表空间	tablespace_quiescer	基本

用法 此元素的值与 sqlutil.h 中的常量 SQLB_QUIESCED_SHARE、SQLB_QUIESCED_UPDATE 或 SQLB_QUIESCED_EXCLUSIVE 的值相匹配。

quiescer_ts_id -“停顿者表空间标识”

导致表空间停顿的对象的表空间标识。

元素标识

quiescer_ts_id

元素类型

信息

表 1429. 快照监视信息

快照级别	逻辑数据分组	监视开关
表空间	tablespace_quiescer	基本

用法 将此元素与 quiescer_obj_id 和 quiescer_auth_id 配合使用，以确定停顿表空间的人员。此元素的值与视图 SYSCAT.TABLES 的列 TBSPACEID 中的值相匹配。

range_adjustment -“范围调整”

此值表示容器数组中范围实际开始的偏移。

表 1430. 快照监视信息

快照级别	逻辑数据分组	监视开关
表空间	tablespace_range	基本

用法 此元素仅适用于 DMS 表空间。

range_container_id -“范围容器”

在范围内唯一定义容器的整数。

表 1431. 快照监视信息

快照级别	逻辑数据分组	监视开关
表空间	tablespace_range	基本

用法 此元素仅适用于 DMS 表空间。

range_end_stripe -“结束分割区”

此值表示范围中的最后一个分割区的编号。

表 1432. 快照监视信息

快照级别	逻辑数据分组	监视开关
表空间	tablespace_range	基本

用法 此元素仅适用于 DMS 表空间。

range_max_extent -“范围中的最大扩展数据块”

此值表示范围映射的最大扩展数据块编号。

表 1433. 快照监视信息

快照级别	逻辑数据分组	监视开关
表空间	tablespace_range	基本

用法 此元素仅适用于 DMS 表空间。

range_max_page_number -“范围中的最大页”

此值表示范围映射的最大页号。

表 1434. 快照监视信息

快照级别	逻辑数据分组	监视开关
表空间	tablespace_range	基本

用法 此元素仅适用于 DMS 表空间。

range_num_containers -“范围中的容器数”

此值表示当前范围中的容器数目。

表 1435. 快照监视信息

快照级别	逻辑数据分组	监视开关
表空间	tablespace_range	基本

用法 此元素仅适用于 DMS 表空间。

range_number -“范围编号”

此值表示表空间映射内的范围的编号。

表 1436. 快照监视信息

快照级别	逻辑数据分组	监视开关
表空间	tablespace_range	基本

用法 此元素仅适用于 DMS 表空间。

range_offset -“范围偏移”

从范围所属的分割集开头的分割区 0 开始的偏移。

表 1437. 快照监视信息

快照级别	逻辑数据分组	监视开关
表空间	tablespace_range	基本

用法 此元素仅适用于 DMS 表空间。

range_start_stripe -“起始分割区”

此值表示范围中的第一个分割区的编号。

表 1438. 快照监视信息

快照级别	逻辑数据分组	监视开关
表空间	tablespace_range	基本

用法 此元素仅适用于 DMS 表空间。

range_stripe_set_number -“分割集编号”

此值表示范围所在的分割集。

表 1439. 快照监视信息

快照级别	逻辑数据分组	监视开关
表空间	tablespace_range	基本

用法 此元素仅适用于 DMS 表空间。

reclaim_wait_time -“回收等待时间”监视元素

在 DB2 pureScale 环境中，此元素表示等待页锁定所耗的时间量（其中锁定请求导致页被回收）。时间的度量单位为毫秒。

表 1440. 表函数监视信息

表函数	监视元素收集级别
MON_FORMAT_XML_METRICS_BY_ROW - 获取所有度量值的基于已格式化行的输出	不适用; 报告 XML 文档中作为格式化函数的输入提供的所有元素
MON_FORMAT_XML_TIMES_BY_ROW - - 获取基于已格式化行的组合层次结构等待时间和处理时间	不适用; 报告 XML 文档中作为格式化函数的输入提供的所有元素
MON_FORMAT_XML_WAIT_TIMES_BY _ROW - 获取等待时间的已格式化的基于行的输出	不适用; 报告 XML 文档中作为格式化函数的输入提供的所有元素
MON_GET_ACTIVITY_DETAILS 表函数 - 获取完整的活动详细信息（在 XML 文档 DETAILS 中报告）	ACTIVITY METRICS BASE
MON_GET_CONNECTION 表函数 - 获取连接度量值	REQUEST METRICS BASE
MON_GET_CONNECTION_DETAILS 表函数 - 获取详细的连接度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE
MON_GET_PAGE_ACCESS_INFO 表函数 - 获取缓冲池页等待信息	ACTIVITY METRICS BASE
MON_GET_PKG_CACHE_STMT 表函数 - 获取程序包高速缓存中的 SQL 语句活动度量值	ACTIVITY METRICS BASE
MON_GET_SERVICE_SUBCLASS 表函数 - 获取服务子类度量值	REQUEST METRICS BASE
MON_GET_SERVICE_SUBCLASS_DETAILS 表函数 - 获取详细的服务子类度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE
MON_GET_UNIT_OF_WORK 表函数 - 获取工作单元度量值	REQUEST METRICS BASE
MON_GET_UNIT_OF_WORK 表函数 - 获取详细的工作单元度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE
MON_GET_WORKLOAD 表函数 - 获取工作负载度量值	ACTIVITY METRICS BASE
MON_GET_WORKLOAD_DETAILS 表函数 - 获取详细的工作负载度量值（在 XML 文档 DETAILS 中报告）	ACTIVITY METRICS BASE

表 1441. 事件监视信息

事件类型	逻辑数据分组	监视开关
活动	event_activity（在 details_xml 文档中报告）	ACTIVITY METRICS BASE

表 1441. 事件监视信息 (续)

事件类型	逻辑数据分组	监视开关
活动	event_activitymetrics	ACTIVITY METRICS BASE
统计信息	event_scstats（在 details_xml 文档中报告）	REQUEST METRICS BASE
统计信息	event_wlstats（在 details_xml 文档中报告）	REQUEST METRICS BASE
工作单元	-	始终收集

用法

等待针对空间映射页的回收所耗的时间是独立计算的，并且在 **spacemappage_reclaim_wait_time** 监视元素中报告。

reclaimable_space_enabled -“已启用可回收空间指示器”监视元素

如果已对表空间启用可回收存储器，那么此监视元素将返回值 1。否则，它返回值 0。

表 1442. 表函数监视信息

表函数	监视元素收集级别
MON_GET_TABLESPACE 表函数 - 获取表空间度量值	ACTIVITY METRICS BASE

regvar_collection_type -“注册表变量收集类型”

指示收集注册表变量值的时间。

表 1443. 事件监视信息

事件类型	逻辑数据分组	监视开关
变更历史记录	REGVAR	始终收集

用法

变更历史记录事件监视器将此值收集为:

I 激活事件监视器时捕获的初始值。

U 已更新值

regvar_level -“注册表变量级别”

指示注册表变量的级别。

表 1444. 事件监视信息

事件类型	逻辑数据分组	监视开关
变更历史记录	REGVAR	始终收集

用法

对于变更历史记录事件监视器，注册表变量的级别为下列其中一种:

E 环境

G 全局

I 实例级别

P 数据库分区

regvar_name -“注册表变量名称”

注册表变量的名称。

表 1445. 事件监视信息

事件类型	逻辑数据分组	监视开关
变更历史记录	REGVAR	始终收集

用法

对于变更历史记录事件监视器，此元素标识作为 REGVAR 事件一部分更新的或作为 REGVARVALUES 事件一部分在事件监视器启动时捕获的注册表变量。这些事件表示以下各项:

REGVAR
更改注册表变量值

REGVARVALUES
在事件监视器启动时捕获注册表变量值

regvar_old_value -“注册表变量旧值”

注册表变量的旧值。

表 1446. 事件监视信息

事件类型	逻辑数据分组	监视开关
变更历史记录	REGVAR	始终收集

用法

对于变更历史记录事件监视器，如果未设置注册表变量值，那么此值为空字符串。

regvar_value -“注册表变量值”

这是注册表变量的值。

表 1447. 事件监视信息

事件类型	逻辑数据分组	监视开关
变更历史记录	REGVAR	始终收集

用法

对于变更历史记录事件监视器，如果未设置此值，那么此值为空字符串。

只有即时注册表变量更新才会生成 REGVAR 事件。

rej_curs_blk -“拒绝的块游标请求数”

在服务器上拒绝请求 I/O 块并且请求转换为非分块 I/O 的次数。

元素标识

rej_curs_blk

元素类型

计数器

表 1448. 快照监视信息

快照级别	逻辑数据分组	监视开关
应用程序	appl	基本

表 1449. 事件监视信息

事件类型	逻辑数据分组	监视开关
连接	event_conn	始终收集

用法 如果有许多游标分块数据，那么通信堆可能会变满。此堆变满时，不会返回错误。而是不会再对分块游标分配 I/O 块。如果游标无法对数据进行分块，那么性能会受到影响。

如果大量游标无法执行数据分块，那么可通过执行以下操作来改进性能:

- 增加 *query_heap* 数据库管理器配置参数的大小。

rem_cons_in -“与数据库管理器的远程连接数”

从远程客户机启动的与正在监视的数据库管理器实例的当前连接数。

表 1450. 快照监视信息

快照级别	逻辑数据分组	监视开关
数据库管理器	db2	基本

用法

显示此实例中从远程客户机至数据库的连接的数目。此值经常更改，所以可能需要在很长的时间段内按特定时间间隔对其进行采样，以了解实际的系统使用情况。此数目不包括从与数据库管理器相同的实例启动的应用程序。

与 local_cons 监视元素一起使用时，这些元素可帮助您调整 **max_coordagents** 和 **max_connections** 配置参数的设置。

rem_cons_in_exec -“数据库管理器中正在执行的远程连接数”

当前连接至数据库，并且正在处理要监视的数据库管理器实例中的工作单元的远程应用程序数。

表 1451. 快照监视信息

快照级别	逻辑数据分组	监视开关
数据库管理器	db2	基本

用法

此数目可帮助您确定数据库管理器上进行的并行处理的级别。此值经常更改，所以可能需要在很长的时间段内按特定时间间隔对其进行采样，以了解实际的系统使用情况。此数目不包括从与数据库管理器相同的实例启动的应用程序。

与 local_cons_in_exec 监视元素一起使用时，此元素可帮助您调整 **max_coordagents** 配置参数的设置。

如果 **max_coordagents** 设置为 AUTOMATIC，那么您不需要作任何调整。如果不是设置为 AUTOMATIC，并且 rem_cons_in_exec 与 local_cons_in_exec 的和接近 **max_coordagents**，那么应该增加 **max_coordagents** 的值。

remote_lock_time -“远程锁定时间”

此元素包含此数据源对远程锁定所花的总时间（以毫秒计），这些远程锁定来自联合服务器启动后或数据库监视计数器上一次重置后（取较晚者）在此联合服务器实例上运行的所有应用程序或单个应用程序。响应时间是以联合服务器将远程锁定提交给数据源的时间与联合服务器在数据源上释放远程锁定的时间之差量度的。

表 1452. 快照监视信息

快照级别	逻辑数据分组	监视开关
数据库	dbase_remote	时间戳记
应用程序	appl_remote	时间戳记

可将快照监视的计数器重置。

用法

使用此元素来确定此数据源对远程锁定所花的实际时间。

remote_locks -“远程锁定”

此元素包含自联合服务器实例启动或数据库监视计数器最后一次重置以后（取较晚者），联合服务器代表任何应用程序在此数据源上调用的远程锁定总数。

表 1453. 快照监视信息

快照级别	逻辑数据分组	监视开关
数据库	dbase_remote	基本
应用程序	appl_remote	基本

可将快照监视的计数器重置。

用法 使用此元素来确定在数据源上进行的远程锁定的数目。

remote_member -“远程成员”监视元素

通过使用快速通信管理器 (FCM) 将数据发送至的数据库成员或从其中接收到数据的数据库成员的数字标识。

表 1454. 表函数监视信息

表函数	监视元素收集级别
MON_GET_FCM_CONNECTION_LIST - 获取有关所有 FCM 连接的详细信息	ACTIVITY METRICS BASE

用法

MON_GET_FCM_CONNECTION_LIST 表函数返回的所有度量值都适用于 **member** 和 **remote_member** 监视元素中所描述成员之间的 FCM 连接。

reopt -“REOPT 绑定选项”监视元素

用于预编译此包的 REOPT 绑定选项。可能的值包括：NONE、ONCE 和 ALWAYS。

表 1455. 事件监视信息

事件类型	逻辑数据分组	监视开关
锁定	lock_participant_activities	

reorg_completion -“重组完成标志”

表重组成功指示器，这包括从多维集群 (MDC) 表或插入时间集群 (ITC) 表中回收扩展数据块。对于分区表来说，此值指示数据分区的完成状态。

表 1456. 快照监视信息

快照级别	逻辑数据分组	监视开关
表	table_reorg	基本

用法 如果表或数据分区重组操作成功，此元素的值将为 0。如果表或数据分区重组操作不成功，那么此元素的值将为 -1。成功和失败值将在 sqlmon.h 中作如下定义：

- 成功：SQLM_REORG_SUCCESS
- 失败：SQLM_REORG_FAIL

如果表重组不成功，那么请参阅历史记录文件以获取任何诊断信息，包括警告和错误。可使用 LIST HISTORY 命令来访问此数据。对于分区表，将对每个数据分区指示完成状态。如果索引重建在分区表上失败，那么将在所有数据分区上更新失败状态。有关进一步的诊断信息，请参阅管理通知日志。

reorg_current_counter -“重组进度”

指示重组完成量的进度单元。此值表示的进度与 reorg_max_counter 的值有关，后者表示要完成的表重组的总量。

表 1457. 快照监视信息

快照级别	逻辑数据分组	监视开关
表	table_reorg	基本

用法

可使用以下公式来确定已完成的表重组的百分比:

```
表重组进度 = reorg_current_counter / reorg_max_counter * 100
```

reorg_end -“表重组结束时间”

表重组（包括为了从多维集群 (MDC) 表或插入时间集群 (ITC) 表中回收扩展数据块而进行的重组）的结束时间。对于分区表来说，此时间指示每个数据分区重组的结束时间。

表 1458. 快照监视信息

快照级别	逻辑数据分组	监视开关
表	table_reorg	基本

reorg_index_id -“用于重组表的索引”

用于重组表的索引。

表 1459. 快照监视信息

快照级别	逻辑数据分组	监视开关
表	table_reorg	基本

reorg_long_tbspc_id -“用来重组长对象的表空间”监视元素

将用来重组任何长对象（LONG VARCHAR 或 LOB 数据）的表空间。对于分区表来说，这是将用来重组每个分区的 LONG VARCHAR 和 LOB 的表空间。

表 1460. 快照监视信息

快照级别	逻辑数据分组	监视开关
表	table_reorg	基本

reorg_max_counter -“重组总量”

此值指示要在重组中完成的总工作量。此值包括重组以从多维集群 (MDC) 表或插入时间集群 (ITC) 表中回收扩展数据块。此值可与 reorg_current_counter 配合使用以确定重组进度，reorg_current_counter 表示完成的工作量。

表 1461. 快照监视信息

快照级别	逻辑数据分组	监视开关
表	table_reorg	基本

reorg_max_phase -“最大重组阶段”

重组处理期间发生的最大重组阶段数。此数目适用于经典（脱机）重组和带 RECLAIM EXTENTS 选项的重组。值的范围为 2 到 4（[SORT], BUILD, REPLACE,[INDEX_RECREATE]）。此值还可能指示执行重组时为了从多维集群 (MDC) 表或插入时间集群 (ITC) 表中回收扩展数据块而完成的工作总量。执行这样的重组时，此值是 3（SCAN、DRAIN 和 RELEASE）。

表 1462. 快照监视信息

快照级别	逻辑数据分组	监视开关
表	table_reorg	基本

reorg_phase -“表重组阶段”监视元素

指示表的重组阶段。对于分区表来说，此元素还将指示每个数据分区的重组阶段。此元素仅适用于脱机表重组。

表 1463. 快照监视信息

快照级别	逻辑数据分组	监视开关
表	table_reorg	基本

用法

对于分区表来说，重组是逐个数据分区进行的。对于传统表重组而言，可能的阶段如下所示（这些阶段与它们在 sqlmon.h 中的相应定义一起列示）：

- 排序：SQLM_REORG_SORT
- 构建：SQLM_REORG_BUILD
- 替换：SQLM_REORG_REPLACE
- 索引重新创建：SQLM_REORG_INDEX_RECREATE
- 字典构建：SQLM_REORG_DICT_SAMPLE

对于分区表而言，在数据分区的“替换”阶段完成后，可以直接进入分区索引（如果有的话）的“索引重建”阶段。仅当每个数据分区上的所有先前阶段成功完成后，**reorg_phase** 元素才会指示“索引重新创建”阶段。

在 XDA 对象压缩期间，XML 数据重组阶段涉及识别表的 XML 存储器对象。XML 字典构建阶段涉及尝试为 XML 存储器对象创建压缩字典。对于 XDA 对象压缩而言，可能的两个阶段如下所示:

- XML 重组: SQLM_REORG_XML_DATA
- XML 字典构建: SQLM_REORG_XML_DICT_SAMPLE

对于分区表，在执行扩展数据块回收操作时，可能的阶段如下所示:

- 扫描: SQLM_REORG_SCAN
- 漏出: SQLM_REORG_DRAIN
- 释放: SQLM_REORG_RELEASE

reorg_phase_start -“重组阶段开始时间”

表重组或回收重组阶段的开始时间。对于分区表来说，此元素还将指示每个数据分区的重组阶段的开始时间。对于非分区索引而言，在索引重建阶段，所有数据分区的数据组将同时进行更新。

表 1464. 快照监视信息

快照级别	逻辑数据分组	监视开关
表	table_reorg	基本

reorg_rows_compressed -“压缩行数”

重组期间在表中压缩的行数。

表 1465. 快照监视信息

快照级别	逻辑数据分组	监视开关
表	table_reorg	基本

用法 重组期间在表中压缩的行数的连续计数。某些记录永远不会被压缩（如果记录长度小于最小记录长度）。

重要的是要注意，这个行数未反映数据压缩效率，它只显示了符合压缩条件的记录的个数。

reorg_rows_rejected_for_compression -“拒绝压缩行数”

重组期间由于记录长度小于或等于最小记录长度而未压缩的行数。

表 1466. 快照监视信息

快照级别	逻辑数据分组	监视开关
表	table_reorg	基本

用法 如果记录长度小于或等于最小记录长度，就不会压缩该记录。已拒绝的行数反映了这些未符合此压缩要求的记录的连续计数。

reorg_start -“表重组开始时间”

表重组（包括为了从多维集群 (MDC) 表或插入时间集群 (ITC) 表中回收扩展数据块而进行的重组）的开始时间。对于分区表来说，此时间指示每个数据分区重组的开始时间。

表 1467. 快照监视信息

快照级别	逻辑数据分组	监视开关
表	table_reorg	基本

reorg_status -“表重组状态”

现场（联机）表重组或数据分区级别重组的状态。此项不适用于传统（脱机）表重组。

表 1468. 快照监视信息

快照级别	逻辑数据分组	监视开关
表	table_reorg	基本

用法 归位表或数据分区重组可能处于下列其中一种状态（状态与它们在 sqlmon.h 中的相应定义列示在一起）：

- 启动/继续: SQLM_REORG_STARTED
- 暂停: SQLM_REORG_PAUSED
- 停止: SQLM_REORG_STOPPED
- 完成: SQLM_REORG_COMPLETED
- 截断: SQLM_REORG_TRUNCATE

用于回收扩展数据块的归位表或数据分区重组可能处于下列其中一种状态:

- 启动: SQLM_REORG_STARTED
- 停止: SQLM_REORG_STOPPED
- 完成: SQLM_REORG_COMPLETED

reorg_tbspc_id -“用来重组表或数据分区的表空间”

用来重组表的表空间。对于分区表来说，这将指示用来重组每个数据分区的表空间。

表 1469. 快照监视信息

快照级别	逻辑数据分组	监视开关
表	table_reorg	基本

reorg_type -“表重组属性”

表重组属性设置。

表 1470. 快照监视信息

快照级别	逻辑数据分组	监视开关
表	table_reorg	基本

用法 可能的属性设置如下所示。每个属性设置都基于 db2ApiDf.h 中定义的位标志值。

- 允许写访问: DB2REORG_ALLOW_WRITE
- 允许读访问: DB2REORG_ALLOW_READ
- 不允许访问: DB2REORG_ALLOW_NONE
- 通过索引扫描重新集群: DB2REORG_INDEXSCAN
- 重组长型字段 LOB 数据: DB2REORG_LONGLOB
- 不截断表: DB2REORG_NOTRUNCATE_ONLINE
- 替换压缩字典: DB2REORG_RESET_DICTIONARY
- 保留压缩字典: DB2REORG_KEEP_DICTIONARY
- 回收扩展数据块: DB2REORG_RECLAIM_EXTS

除了上述属性设置以外，在 GET SNAPSHOT FOR TABLES 命令的 CLP 输出中还列示了下列属性。这些属性设置基于其他属性设置值或表重组监视元素值。

- 重新集群: 如果 reorg_index_id 监视元素值不为零，那么表重组操作具有此属性。
- 重新声明: 如果 reorg_index_id 监视元素值为零，那么表重组操作具有此属性。
- 原位表重组: 如果 reorg_status 监视元素值不为空，那么表示正在使用原位（联机）重组方法。
- 表重组: 如果 reorg_phase 监视元素值不为空，那么表示正在使用传统（脱机）重组方法。
- 通过表扫描重新集群: 如果未设置 DB2REORG_INDEXSCAN 标志，那么表重组操作具有此属性。
- 仅重组数据: 如果未设置 DB2REORG_LONGLOB 标志，那么表重组操作具有此属性。

reorg_xml_regions_compressed -“已压缩的 XML 区域数”监视元素

在表重组过程中压缩的 XML 区域的数目。

表 1471. 快照监视信息

快照级别	逻辑数据分组	监视开关
表	table_reorg	基本

reorg_xml_regions_rejected_for_compression -“拒绝压缩的 XML 区域数”监视元素

在表重组过程中未压缩的 XML 区域的数目。

表 1472. 快照监视信息

快照级别	逻辑数据分组	监视开关
表	table_reorg	基本

req_agent_tid -“正在等待获取锁定的代理程序的线程标识”监视元素

正在等待获取锁定的代理程序或系统实体的线程标识。

表 1473. 表函数监视信息

表函数	监视元素收集级别
MON_GET_APPL_LOCKWAIT 表函数 - 获取有关应用程序正在等待的锁定的信息	ACTIVITY METRICS BASE

req_application_handle -“正在等待获取锁定的应用程序的标识”监视元素

正在等待获取锁定的应用程序的系统范围内的唯一标识。

表 1474. 表函数监视信息

表函数	监视元素收集级别
MON_GET_APPL_LOCKWAIT 表函数 - 获取有关应用程序正在等待的锁定的信息	ACTIVITY METRICS BASE

req_executable_id -“正在等待获取锁定的语句部分的标识”监视元素

在数据服务器上生成的二进制标记，用于唯一地标识正在等待获取锁定的 SQL 语句部分。对于非 SQL 活动，将返回长度为 0 的字符串值。

表 1475. 表函数监视信息

表函数	监视元素收集级别
MON_GET_APPL_LOCKWAIT 表函数 - 获取有关应用程序正在等待的锁定的信息	ACTIVITY METRICS BASE

req_member -“正在等待获取锁定的应用程序的成员”监视元素

正在等待获取此锁定的应用程序所在的数据库成员。

表 1476. 表函数监视信息

表函数	监视元素收集级别
MON_GET_APPL_LOCKWAIT 表函数 - 获取有关应用程序正在等待的锁定的信息	ACTIVITY METRICS BASE

request_exec_time_avg -“平均请求执行时间”监视元素

自最后一次重置以后与此服务子类相关联的请求的执行时间算术平均值。如果内部跟踪的平均值已溢出，那么将返回值 -2。当服务子类的 COLLECT AGGREGATE REQUEST DATA 设置为 NONE 时，此监视元素返回 -1。单位为毫秒。

使用 REMAP ACTIVITY 操作在服务子类之间重新映射活动时，request_exec_time_avg 平均值将对重新映射所涉及的每个子类中的不完整请求进行计数。

表 1477. 表函数监视信息

表函数	监视元素收集级别
WLM_GET_SERVICE_SUBCLASS_STATS 表函数 - 返回服务子类的统计信息	COLLECT AGGREGATE REQUEST DATA

表 1478. 事件监视信息

事件类型	逻辑数据分组	监视开关
统计信息	event_scstats	-

用法

使用此统计信息以快速了解成员在处理此服务子类中的每个请求所花的平均时间量。

另外，还可以使用此平均值来确定用于请求执行时间直方图的直方图模板是否合适。根据请求执行时间直方图来计算平均请求执行时间。将计算出来的平均值与此监视元素进行比较。如果计算出来的平均值偏离了此监视元素报告的真实平均值，那么考虑修改请求执行时间直方图的直方图模板并使用更为适合您的数据的一组 bin 值。

rf_log_num -“正在前滚的日志”监视元素

正在前滚操作中处理的日志。

表 1479. 快照监视信息

快照级别	逻辑数据分组	监视开关
表空间	rollforward	基本

用法

如果正在进行前滚，那么此元素标识前滚涉及的日志。 在 DB2 pureScale 环境中，**rf_log_num** 监视元素标识每个日志流中当前正涉及前滚操作的日志文件。

rf_status -“日志阶段”

恢复的状态。

元素标识

rf_status

元素类型

信息

表 1480. 快照监视信息

快照级别	逻辑数据分组	监视开关
表空间	rollforward	基本

用法 此元素指示恢复的进度。它指示恢复是处于撤销（回滚）阶段还是处于重做（前滚）阶段。

rf_timestamp -“前滚时间戳记”

上次落实的事务的时间戳记。

元素标识

rf_timestamp

元素类型

时间戳记

表 1481. 快照监视信息

快照级别	逻辑数据分组	监视开关
表空间	rollforward	时间戳记

用法 如果正在进行前滚，那么这是前滚恢复操作所处理的上次落实事务的时间戳记。这是前滚操作的进度指示符。

rf_type -“前滚类型“

正在进行的前滚的类型。

元素标识

rf_type

元素类型

信息

表 1482. 快照监视信息

快照级别	逻辑数据分组	监视开关
表空间	rollforward	基本

用法 指示是在数据库级别还是表空间级别进行恢复的指示符。

rollback_sql_stmts -“尝试的回滚语句数”

尝试的 SQL ROLLBACK 语句总数。

元素标识

rollback_sql_stmts

元素类型

计数器

表 1483. 快照监视信息

快照级别	逻辑数据分组	监视开关
数据库	dbase	基本
数据库	dbase_remote	基本
应用程序	appl	基本
应用程序	appl_remote	基本
DCS 数据库	dcs_dbase	基本
DCS 应用程序	dcs_appl	基本

可将快照监视的计数器重置。

表 1484. 事件监视信息

事件类型	逻辑数据分组	监视开关
数据库	event_db	始终收集
连接	event_conn	始终收集

用法 回滚可能是应用程序请求、死锁或错误情况导致的。此元素**仅**对从应用程序发出的回滚语句计数。

在应用程序级别，此元素可帮助您确定应用程序的数据库活动的级别以及与其他应用程序的冲突程度。在数据库级别，它可以帮助您确定数据库中的活动量以及数据库上的应用程序间的冲突程度。

注: 应尝试将回滚次数降至最低，原因是回滚活动越高，数据库的吞吐量越低。

还可使用此元素并通过计算下列表达式的总和来计算 工作单元 的总数:

```
    commit_sql_stmts
+ int_commits
+ rollback_sql_stmts
+ int_rollbacks
```

rolled_back_agent_id -“回滚的代理程序”

发生死锁时回滚的代理程序。

元素标识

rolled_back_agent_id

元素类型

信息

表 1485. 事件监视信息

事件类型	逻辑数据分组	监视开关
死锁	event_deadlock	始终收集

用法 系统管理员可使用此信息来确定未完成更新的应用程序以及应重新启动的应用程序。

rolled_back_appl_id -“回滚的应用程序”

发生死锁时回滚的应用程序标识。

元素标识

rolled_back_appl_id

元素类型

信息

表 1486. 事件监视信息

事件类型	逻辑数据分组	监视开关
死锁	event_deadlock	始终收集

用法 系统管理员可使用此信息来确定未完成更新的应用程序以及应重新启动的应用程序。

rolled_back_participant_no -“回滚的应用程序参与者”监视元素

用于标识已回滚的应用程序的参与者编号。

表 1487. 事件监视信息

事件类型	逻辑数据分组	监视开关
锁定	-	始终收集
死锁 [1]	event_deadlock	始终收集

1 建议不要使用此事件监视器。建议不要再使用此选项，将来的发行版可能会将其除去。请使用 CREATE EVENT MONITOR FOR LOCKING 语句来监视与锁定相关的事件，例如锁定超时、锁定等待和死锁。

用法

系统管理员可使用此信息来确定未完成更新的应用程序以及应重新启动的应用程序。

rolled_back_sequence_no -“回滚的序号”

发生死锁时回滚的应用程序的序号。

元素标识

rolled_back_sequence_no

元素类型

信息

表 1488. 事件监视信息

事件类型	逻辑数据分组	监视开关
死锁	event_deadlock	始终收集

用法 系统管理员可使用此信息来确定未完成更新的应用程序以及应重新启动的应用程序。

root_node_splits -“根节点分割次数”监视元素

在插入操作期间分割索引根节点的次数。

表 1489. 表函数监视信息

表函数	监视元素收集级别
MON_GET_INDEX 表函数 - 获取索引度量值	ACTIVITY METRICS BASE

routine_id -“例程标识”监视元素

唯一的例程标识。如果此活动未包含在例程中，那么此监视元素将返回 0。

表 1490. 表函数监视信息

表函数	监视元素收集级别
MON_GET_ACTIVITY_DETAILS 表函数 - 获取完整的活动详细信息（在 XML 文档 DETAILS 中报告）	ACTIVITY METRICS BASE
MON_GET_PKG_CACHE_STMT 表函数	ACTIVITY METRICS BASE
MON_GET_PKG_CACHE_STMT_DETAILS - 获取程序包高速缓存条目的详细度量值	ACTIVITY METRICS BASE
WLM_GET_SERVICE_CLASS_AGENTS 表函数 - 列示正在服务类中运行的代理程序	ACTIVITY METRICS BASE
WLM_GET_WORKLOAD_OCCURRENCE_ACTIVITIES 表函数 - 返回活动列表	ACTIVITY METRICS BASE

表 1491. 事件监视信息

事件类型	逻辑数据分组	监视开关
活动	event_activitystmt	始终收集
工作单元	uow_package_list	始终收集
程序包高速缓存	pkgcache_metrics	ACTIVITY METRICS BASE

用法

此元素的值与 SYSCAT.ROUTINES 视图的 ROUTINEID 列中的值相匹配。如果此活动包含在您在另一个 SQL PL 例程中声明的例程中，那么此元素的值是外部例程的 ROUTINEID。

rows_deleted -“删除行数”监视元素

这是所尝试的行删除操作的数目。

表 1492. 表函数监视信息

表函数	监视元素收集级别
MON_GET_TABLE 表函数 - 获取表度量值	ACTIVITY METRICS BASE
MON_GET_TABLE_USAGE_LIST 表函数 - 从表用法列表返回信息	DATA OBJECT METRICS EXTENDED

表 1493. 快照监视信息

快照级别	逻辑数据分组	监视开关
数据库	dbase	基本
数据库	dbase_remote	基本
应用程序	appl	基本
应用程序	appl_remote	基本

可将快照监视的计数器重置。

表 1494. 事件监视信息

事件类型	逻辑数据分组	监视开关
数据库	event_db	始终收集
连接	event_conn	始终收集

用法 您可以使用此元素来确定数据库的当前活动级别。

此计数不包括 **int_rows_deleted** 监视元素中记录的尝试次数。

rows_fetched -“访存的行数”监视元素

从表中读取的行数。

此监视元素是 **rows_read** 监视元素的别名。

注: 此监视元素仅报告为其记录了此信息的成员的值。在多成员数据库环境中，这些值可能未反映整个活动的正确总计。

表 1495. 事件监视信息

事件类型	逻辑数据分组	监视开关
活动	event_activity	语句

用法

有关详细信息，请参阅 **rows_read** 监视元素。

rows_inserted -“插入行数”监视元素

尝试插入的行数。

表 1496. 表函数监视信息

表函数	监视元素收集级别
MON_GET_TABLE 表函数 - 获取表度量值	ACTIVITY METRICS BASE
MON_GET_TABLE_USAGE_LIST 表函数 - 从表用法列表返回信息	DATA OBJECT METRICS EXTENDED

表 1497. 快照监视信息

快照级别	逻辑数据分组	监视开关
数据库	dbase	基本
数据库	dbase_remote	基本
应用程序	appl	基本
应用程序	appl_remote	基本

可将快照监视的计数器重置。

表 1498. 事件监视信息

事件类型	逻辑数据分组	监视开关
数据库	event_db	始终收集
连接	event_conn	始终收集

用法 您可以使用此元素来确定数据库的当前活动级别。

在联合系统中，每个 INSERT 语句可插入多行，原因是联合服务器会在适当时将 INSERT FROM SUBSELECT 推送至数据源。

此计数不包括 **int_rows_inserted** 监视元素中记录的尝试次数。

rows_modified -“修改的行数”监视元素

插入、更新或删除的行数。

此监视元素是 **rows_written** 监视元素的别名。

表 1499. 表函数监视信息

表函数	监视元素收集级别
MON_FORMAT_XML_METRICS_BY_ROW - 获取所有度量值的基于已格式化行的输出	不适用；报告 XML 文档中作为格式化函数的输入提供的所有元素
MON_GET_ACTIVITY_DETAILS 表函数 - 获取完整的活动详细信息（在 XML 文档 DETAILS 中报告）	ACTIVITY METRICS BASE
MON_GET_CONNECTION 表函数 - 获取连接度量值	REQUEST METRICS BASE
MON_GET_CONNECTION_DETAILS 表函数 - 获取详细的连接度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE
MON_GET_PKG_CACHE_STMT 表函数 - 获取程序包高速缓存中的 SQL 语句活动度量值	ACTIVITY METRICS BASE
MON_GET_PKG_CACHE_STMT_DETAILS 表函数 - 获取程序包高速缓存条目的详细度量值	ACTIVITY METRICS BASE
MON_GET_SERVICE_SUBCLASS 表函数 - 获取服务子类度量值	REQUEST METRICS BASE
MON_GET_SERVICE_SUBCLASS_DETAILS 表函数 - 获取详细的服务子类度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE

表 1499. 表函数监视信息 (续)

表函数	监视元素收集级别
MON_GET_UNIT_OF_WORK 表函数 - 获取工作单元度量值	REQUEST METRICS BASE
MON_GET_UNIT_OF_WORK 表函数 - 获取详细的工作单元度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE
MON_GET_WORKLOAD 表函数 - 获取工作负载度量值	REQUEST METRICS BASE
MON_GET_WORKLOAD_DETAILS 表函数 - 获取详细的工作负载度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE

表 1500. 事件监视信息

事件类型	逻辑数据分组	监视开关
活动	event_activity（在 details_xml 文档中报告）	ACTIVITY METRICS BASE
活动	event_activitymetrics	ACTIVITY METRICS BASE
统计信息	event_scstats（在 details_xml 文档中报告）	REQUEST METRICS BASE
统计信息	event_wlstats（在 details_xml 文档中报告）	REQUEST METRICS BASE
工作单元	在 system_metrics 文档中报告。	REQUEST METRICS BASE
活动	event_activity	语句
程序包高速缓存	在 activity_metrics 文档中报告。	ACTIVITY METRICS BASE

用法

有关详细信息，请参阅 **rows_written** 监视元素。

rows_read -“读取行数”监视元素

从表中读取的行数。

表 1501. 表函数监视信息

表函数	监视元素收集级别
MON_FORMAT_XML_METRICS_BY_ROW - 获取所有度量值的基于已格式化行的输出	不适用；报告 XML 文档中作为格式化函数的输入提供的所有元素
MON_GET_ACTIVITY_DETAILS 表函数 - 获取完整的活动详细信息（在 XML 文档 DETAILS 中报告）	ACTIVITY METRICS BASE
MON_GET_CONNECTION 表函数 - 获取连接度量值	REQUEST METRICS BASE

表 1501. 表函数监视信息 (续)

表函数	监视元素收集级别
MON_GET_CONNECTION_DETAILS 表函数 - 获取详细的连接度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE
MON_GET_PKG_CACHE_STMT 表函数 - 获取程序包高速缓存中的 SQL 语句活动度量值	ACTIVITY METRICS BASE
MON_GET_PKG_CACHE_STMT_DETAILS 表函数 - 获取程序包高速缓存条目的详细度量值	ACTIVITY METRICS BASE
MON_GET_SERVICE_SUBCLASS 表函数 - 获取服务子类度量值	REQUEST METRICS BASE
MON_GET_SERVICE_SUBCLASS_DETAILS 表函数 - 获取详细的服务子类度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE
MON_GET_TABLE 表函数 - 获取表度量值	ACTIVITY METRICS BASE
MON_GET_TABLE_USAGE_LIST 表函数 - 从表用法列表返回信息	DATA OBJECT METRICS EXTENDED
MON_GET_UNIT_OF_WORK 表函数 - 获取工作单元度量值	REQUEST METRICS BASE
MON_GET_UNIT_OF_WORK 表函数 - 获取详细的工作单元度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE
MON_GET_WORKLOAD 表函数 - 获取工作负载度量值	REQUEST METRICS BASE
MON_GET_WORKLOAD_DETAILS 表函数 - 获取详细的工作负载度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE
WLM_GET_WORKLOAD_OCCURRENCE_ACTIVITIES 表函数 - 返回活动列表	REQUEST METRICS BASE

表 1502. 快照监视信息

快照级别	逻辑数据分组	监视开关
数据库	dbase	基本
表	表	表
应用程序	appl	基本
应用程序	stmt	基本
应用程序	subsection	语句
动态 SQL	dynsql	语句

可将快照监视的计数器重置。

表 1503. 事件监视信息

事件类型	逻辑数据分组	监视开关
活动	event_activity（在 details_xml 文档中报告）	ACTIVITY METRICS BASE

表 1503. 事件监视信息 (续)

事件类型	逻辑数据分组	监视开关
活动	event_activitymetrics	ACTIVITY METRICS BASE
统计信息	event_scstats（在 details_xml 文档中报告）	REQUEST METRICS BASE
统计信息	event_wlstats（在 details_xml 文档中报告）	REQUEST METRICS BASE
工作单元	在 system_metrics 文档中报告。	REQUEST METRICS BASE
连接	event_conn	始终收集
表	event_table	始终收集
语句	event_stmt	始终收集
事务	event_xact	始终收集
程序包高速缓存	在 activity_metrics 文档中报告。	ACTIVITY METRICS BASE

用法

此元素帮助标识使用频率很高并且您可能想要为其创建附加索引的表。为了避免维护非必需的索引，请使用 SQL EXPLAIN 语句来确定程序包是否使用索引。

此计数不是返回至调用应用程序的行数。而是必须读取以返回结果集的行数。例如，以下语句向应用程序返回一行，但读取了许多行以确定平均薪水:

```
SELECT AVG(SALARY) FROM USERID.EMPLOYEE
```

此计数包括 **overflow_accesses** 监视元素中的值。另外，此计数不包括任何索引访问。即，如果存取方案仅使用索引访问方法，并且不会访问该表以查看实际的行，那么 **rows_read** 监视元素的值不会递增。

rows_returned -“返回的行数”监视元素

已选择并返回到应用程序的行数。对于部分活动记录，此元素具有 0 值（例如，当活动仍在执行或当完整活动记录因内存限制而无法写入事件监视器时，如果收集活动，那么就会出现此情况）。

此监视元素是 **fetch_count** 监视元素的别名。

表 1504. 表函数监视信息

表函数	监视元素收集级别
MON_FORMAT_XML_METRICS_BY_ROW - 获取所有度量值的基于已格式化行的输出	不适用; 报告 XML 文档中作为格式化函数的输入提供的所有元素
MON_GET_ACTIVITY_DETAILS 表函数 - 获取完整的活动详细信息（在 XML 文档 DETAILS 中报告）	ACTIVITY METRICS BASE
MON_GET_CONNECTION 表函数 - 获取连接度量值	REQUEST METRICS BASE

表 1504. 表函数监视信息 (续)

表函数	监视元素收集级别
MON_GET_CONNECTION_DETAILS 表函数 - 获取详细的连接度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE
MON_GET_PKG_CACHE_STMT 表函数 - 获取程序包高速缓存中的 SQL 语句活动度量值	ACTIVITY METRICS BASE
MON_GET_PKG_CACHE_STMT_DETAILS 表函数 - 获取程序包高速缓存条目的详细度量值	ACTIVITY METRICS BASE
MON_GET_SERVICE_SUBCLASS 表函数 - 获取服务子类度量值	REQUEST METRICS BASE
MON_GET_SERVICE_SUBCLASS_DETAILS 表函数 - 获取详细的服务子类度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE
MON_GET_UNIT_OF_WORK 表函数 - 获取工作单元度量值	REQUEST METRICS BASE
MON_GET_UNIT_OF_WORK 表函数 - 获取详细的工作单元度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE
MON_GET_WORKLOAD 表函数 - 获取工作负载度量值	REQUEST METRICS BASE
MON_GET_WORKLOAD_DETAILS 表函数 - 获取详细的工作负载度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE
WLM_GET_WORKLOAD_OCCURRENCE_ACTIVITIES 表函数 - 返回活动列表	REQUEST METRICS BASE

表 1505. 事件监视信息

事件类型	逻辑数据分组	监视开关
活动	event_activity（在 details_xml 文档中报告）	ACTIVITY METRICS BASE
活动	event_activitymetrics	ACTIVITY METRICS BASE
统计信息	event_scstats（在 details_xml 文档中报告）	REQUEST METRICS BASE
统计信息	event_wlstats（在 details_xml 文档中报告）	REQUEST METRICS BASE
工作单元	在 system_metrics 文档中报告。	REQUEST METRICS BASE
活动	event_activity	始终收集
程序包高速缓存	在 activity_metrics 文档中报告。	ACTIVITY METRICS BASE

用法

此元素可用来帮助确定返回到应用程序的行数的阈值，或者用来验证该阈值配置是否正确且正常工作。

rows_returned_top -“最高实际返回行数”监视元素

服务类或工作类中所有嵌套级别的 DML 活动实际返回行数的高水位标记。对于服务类，当服务类的 COLLECT AGGREGATE ACTIVITY DATA 设置为 NONE 时，此监视元素返回 -1。对于工作类，如果未对该工作类指定 COLLECT AGGREGATE ACTIVITY DATA 工作操作，那么此监视元素将返回 -1。对于工作负载而言，当工作负载的 COLLECT AGGREGATE ACTIVITY DATA 设置为 NONE 时，此监视元素返回 -1。

对于服务类而言，使用 REMAP ACTIVITY 操作在服务子类之间重新映射活动时，将仅更新完成该活动的服务子类的 rows_returned_top 高水位标记。该活动所映射到但未在其中完成该活动的服务子类的高水位标记不受影响。

表 1506. 事件监视信息

事件类型	逻辑数据分组	监视开关
统计信息	event_scstats	-
统计信息	event_wcstats	-
统计信息	event_wlstats	-

用法

使用此元素可了解在收集时间间隔内，成员对服务类、工作负载 或工作类上达到最高 DML 活动实际返回行数。

rows_selected -“选择的行数”

此项是已选择并且返回至应用程序的行数。

表 1507. 快照监视信息

快照级别	逻辑数据分组	监视开关
数据库	dbase	基本
数据库	dbase_remote	基本
应用程序	appl	基本
应用程序	appl_remote	基本

可将快照监视的计数器重置。

表 1508. 事件监视信息

事件类型	逻辑数据分组	监视开关
数据库	event_db	始终收集
连接	event_conn	始终收集

用法 您可以使用此元素来确定数据库的当前活动级别。

此元素不包括对 COUNT(*) 或连接这样的操作读取的行计数。

对于联合系统，可计算将数据源中的一行返回至联合服务器的平均时间:

```
平均时间 = 返回的行数 / 聚集查询响应时间
```

可使用这些结果来修改 SYSCAT.SERVERS 中的 CPU 速度或通信速度参数。修改这些参数会影响优化器是否将请求发送至数据源。

注: 如果被监视的网关为 DB2 数据库版本 7.2 或更低版本，那么将在 dcs_dbase 和 dcs_appl 快照监视器逻辑数据组中收集此元素。

rows_updated -“更新行数”监视元素

这是尝试更新的行数。

表 1509. 表函数监视信息

表函数	监视元素收集级别
MON_GET_TABLE 表函数 - 获取表度量值	ACTIVITY METRICS BASE
MON_GET_TABLE_USAGE_LIST 表函数 - 从表用法列表返回信息	DATA OBJECT METRICS EXTENDED

表 1510. 快照监视信息

快照级别	逻辑数据分组	监视开关
数据库	dbase	基本
数据库	dbase_remote	基本
应用程序	appl	基本
应用程序	appl_remote	基本

可将快照监视的计数器重置。

表 1511. 事件监视信息

事件类型	逻辑数据分组	监视开关
数据库	event_db	始终收集
连接	event_conn	始终收集

用法 您可以使用此元素来确定数据库的当前活动级别。

此值不包括 **int_rows_updated** 监视元素中记录的更新计数。但是，将对每个更新计算多个更新语句更新的行数。

rows_written -“写入的行数”

此项是表中更改（插入、删除或更新）的行数。

表 1512. 快照监视信息

快照级别	逻辑数据分组	监视开关
表	表	基本
应用程序	appl	基本
应用程序	stmt	基本
应用程序	subsection	语句
动态 SQL	dynsql	语句

可将快照监视的计数器重置。

表 1513. 事件监视信息

事件类型	逻辑数据分组	监视开关
连接	event_conn	始终收集
表	event_table	始终收集
语句	event_stmt	始终收集
事务	event_xact	始终收集

用法 如果表级别信息的值很高，那么指示该表的使用频率太高，您可能使用“运行统计”（RUNSTATS）实用程序来保持用于此表的程序包的效率。

对于应用程序连接和语句，此元素包括临时表中插入、更新和删除的行数。

在应用程序、事务和语句级别，此元素对于分析相对活动级别和标识调整候选对象会非常有用。

rqsts_completed_total -“完成请求总数”监视元素

已执行的请求的总数，其中包括应用程序请求和内部请求。对于服务子类而言，将仅在此请求的完成位置更新此监视元素。如果此请求曾在不同服务子类之间移动，那么将计数两次。

表 1514. 表函数监视信息

表函数	监视元素收集级别
MON_FORMAT_XML_METRICS_BY_ROW - 获取所有度量值的基于已格式化行的输出	不适用; 报告 XML 文档中作为格式化函数的输入提供的所有元素
MON_GET_CONNECTION 表函数 - 获取连接度量值	REQUEST METRICS BASE
MON_GET_CONNECTION_DETAILS 表函数 - 获取详细的连接度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE
MON_GET_SERVICE_SUBCLASS 表函数 - 获取服务子类度量值	REQUEST METRICS BASE
MON_GET_SERVICE_SUBCLASS_DETAILS 表函数 - 获取详细的服务子类度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE
MON_GET_UNIT_OF_WORK 表函数 - 获取工作单元度量值	REQUEST METRICS BASE
MON_GET_UNIT_OF_WORK 表函数 - 获取详细的工作单元度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE
MON_GET_WORKLOAD 表函数 - 获取工作负载度量值	REQUEST METRICS BASE
MON_GET_WORKLOAD_DETAILS 表函数 - 获取详细的工作负载度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE

表 1515. 事件监视信息

事件类型	逻辑数据分组	监视开关
统计信息	event_scstats（在 details_xml 文档中报告）	REQUEST METRICS BASE
统计信息	event_wlstats（在 details_xml 文档中报告）	REQUEST METRICS BASE
工作单元	在 system_metrics 文档中报告。	REQUEST METRICS BASE

savepoint_id -“保存点标识”

工作单元内设置的保存点的标识。

表 1516. 事件监视信息

事件类型	逻辑数据分组	监视开关
变更历史记录	DDLSTMTEXEC TXNCOMPLETION	始终收集

sc_work_action_set_id -“服务类工作操作集标识”监视元素

如果此活动已划分到服务类作用域的某个工作类，那么此监视元素显示与该工作类所属的工作类集相关联的工作操作集的标识。否则，此监视元素将显示值 0。

表 1517. 表函数监视信息

表函数	监视元素收集命令和级别
WLM_GET_ACTIVITY_DETAILS_COMPLETE 表函数 - 获取完整的活动详细信息（在 XML 文档 DETAILS 中报告）	ACTIVITY METRICS BASE

表 1518. 事件监视信息

事件类型	逻辑数据分组	监视开关
活动	event_activity	始终收集

用法

可将此元素与 **sc_work_class_id** 元素配合使用，以唯一地标识活动的服务类工作类（如果有）。

sc_work_class_id -“服务类工作类标识”监视元素

如果此活动已划分到服务类作用域的某个工作类，那么此监视元素将显示分配给此活动的工作类的标识。否则，此监视元素将显示值 0。

表 1519. 表函数监视信息

表函数	监视元素收集命令和级别
WLM_GET_ACTIVITY_DETAILS_COMPLETE 表函数 - 获取完整的活动详细信息（在 XML 文档 DETAILS 中报告）	ACTIVITY METRICS BASE

表 1520. 事件监视信息

事件类型	逻辑数据分组	监视开关
活动	event_activity	始终收集

用法

可将此元素与 **sc_work_action_set_id** 元素配合使用，以唯一地标识活动的服务类工作类（如果有）。

sec_log_used_top -“使用的最大辅助日志空间”

使用的最大辅助日志空间（以字节计）。

表 1521. 表函数监视信息

表函数	监视元素收集级别
MON_GET_TRANSACTION_LOG 表函数 - 获取日志信息	ACTIVITY METRICS BASE

表 1522. 快照监视信息

快照级别	逻辑数据分组	监视开关
数据库	dbase	基本

表 1523. 事件监视信息

事件类型	逻辑数据分组	监视开关
数据库	event_db	始终收集

用法 可将此元素与 *sec_logs_allocated* 和 *tot_log_used_top* 配合使用以显示当前对辅助日志的依赖性。如果此值很高，那么可能需要更大的日志文件或者更多的主日志文件，又或是在应用程序中更频繁地使用 COMMIT 语句。

因此，可能需要调整下列配置参数:

- logfilsiz
- logprimary
- logsecond
- logarchmeth1

如果数据库没有任何辅助日志文件，那么该值将为零。如果未定义辅助日志文件，那么将出现此情况。

注: 虽然数据库系统监视器信息是以字节为单位给定的，但配置参数是以页为单位设置的，每页为 4K 字节。

sec_logs_allocated -“当前分配的辅助日志数”

当前用于数据库的辅助日志文件总数。

表 1524. 表函数监视信息

表函数	监视元素收集级别
MON_GET_TRANSACTION_LOG 表函数 - 获取日志信息	ACTIVITY METRICS BASE

表 1525. 快照监视信息

快照级别	逻辑数据分组	监视开关
数据库	dbase	基本

用法 可将此元素与 *sec_log_used_top* 和 *tot_log_used_top* 一起使用来显示当前对辅助日志的依赖性。如果此值一直很高，那么可能需要更大的日志文件或者更多的主日志文件，又或是在应用程序中更频繁地使用 COMMIT 语句。

因此，可能需要调整下列配置参数:

- logfilsiz
- logprimary
- logsecond
- logarchmeth1

section_actuals -“部分实际值”监视元素

在数据服务器中生成的二进制字符串，它包含已执行的部分的运行时统计信息。如果未启用捕获部分或收集实际值，那么此值是一个长度为 0 的字符串。对于非 SQL 活动（例如，LOAD），此值是一个长度为 0 的字符串。

表 1526. 事件监视信息

事件类型	逻辑数据分组	监视开关
活动	event_activity	始终收集

用法

使用 EXPLAIN_FROM_ACTIVITY 存储过程来执行段说明时，将使用在 **section_actuals** 监视元素或者每个使用 WLM_SET_CONN_ENV 的连接中收集的数据。在执行 EXPLAIN 处理期间使用此数据来填充 EXPLAIN_ACTUALS 说明表，以及表示存取方案中的运算符的运行时统计信息。

注:

- 仅当已经使用 **section_actuals** 数据库配置参数启用了部分实际值（即，将此配置参数设置为 BASE），或者已经使用 WLM_SET_CONN_ENV 存储过程为特定应用程序启用了部分实际值时，部分实际值才可用。有关描述此存储过程的更多信息，请参阅 WLM_SET_CONN_ENV。
- 由 WLM_SET_CONN_ENV 过程为应用程序指定的 **section_actuals** 设置会立即生效。

section_env -“节环境”监视元素

包含 SQL 语句的部分的 BLOB。它是实际部分内容，是查询方案的可执行形式。

表 1527. 事件监视信息

事件类型	逻辑数据分组	监视开关
活动	event_activitystmt	始终收集
程序包高速缓存	-	COLLECT DETAILED DATA

用法

将此元素与部分说明过程配合使用来说明语句和查看该语句的存取方案。

section_number -“节号”监视元素

静态 SQL 语句在程序包中的内部节号。

表 1528. 表函数监视信息

表函数	监视元素收集级别
MON_GET_ACTIVITY_DETAILS 表函数 - 获取完整的活动详细信息（在 XML 文档 DETAILS 中报告）	ACTIVITY METRICS BASE
MON_GET_PKG_CACHE_STMT 表函数 - 获取程序包高速缓存中的 SQL 语句活动度量值	ACTIVITY METRICS BASE
MON_GET_PKG_CACHE_STMT_DETAILS 表函数 - 获取程序包高速缓存条目的详细度量值	ACTIVITY METRICS BASE

表 1529. 快照监视信息

快照级别	逻辑数据分组	监视开关
应用程序	stmt	语句
DCS 语句	dcs_stmt	语句

表 1530. 事件监视信息

事件类型	逻辑数据分组	监视开关
锁定	-	-
带有详细信息的死锁 [1]	event_detailed_dlconn	-
语句	event_stmt	-
活动	event_activitystmt	-
程序包高速缓存	-	COLLECT BASE DATA

1 建议不要使用此事件监视器。建议不要再使用此选项，将来的发行版可能会将其除去。请使用 CREATE EVENT MONITOR FOR LOCKING 语句来监视与锁定相关的事件，例如锁定超时、锁定等待和死锁。

用法

对于静态 SQL 语句，可以将此元素与 **creator**、**package_version_id** 和 **package_name** 监视元素配合使用，以便使用以下样本查询来查询 SYSCAT.STATEMENTS 系统目录表并获取静态 SQL 语句文本:

```
SELECT SEQNO, SUBSTR(TEXT,1,120)
       FROM SYSCAT.STATEMENTS
       WHERE PKGNAME   = 'package_name' AND
             PKGSCHEMA = 'creator'      AND
             VERSION = 'package_version_id' AND
             SECTNO    = section_number
       ORDER BY SEQNO
```

注: 在获取静态语句文本时将产生警告，原因是针对系统目录表执行的此查询可能会导致锁定争用。尽可能在没有其他针对该数据库的活动时才使用此查询。

section_type -“节类型指示器”监视元素

指示 SQL 语句节是动态的还是静态的。

表 1531. 表函数监视信息

表函数	监视元素收集级别
MON_GET_PKG_CACHE_STMT 表函数 - 获取程序包高速缓存中的 SQL 语句活动度量值	ACTIVITY METRICS BASE
MON_GET_PKG_CACHE_STMT_DETAILS 表函数 - 获取程序包高速缓存条目的详细度量值	ACTIVITY METRICS BASE

表 1532. 事件监视信息

事件类型	逻辑数据分组	监视开关
程序包高速缓存	-	COLLECT BASE DATA

用法

此监视元素的可能值包括:

- D: 动态
- S: 静态

select_sql_stmts -“执行的 Select SQL 语句数”

执行的 SQL SELECT 语句数。

表 1533. 快照监视信息

快照级别	逻辑数据分组	监视开关
数据库	dbase	基本
数据库	dbase_remote	基本
表空间	tablespace	基本
应用程序	appl	基本
应用程序	appl_remote	基本

可将快照监视的计数器重置。

表 1534. 事件监视信息

事件类型	逻辑数据分组	监视开关
数据库	event_db	始终收集
连接	event_conn	始终收集

用法 可使用此元素来确定应用程序或数据库级别的数据库活动的级别。

还可使用以下公式来确定 SELECT 语句数与语句总数的比率:

```
    select_sql_stmts
/ ( static_sql_stmts
    + dynamic_sql_stmts )
```

此信息对于分析应用程序活动和吞吐量非常有用。

select_time -“查询响应时间”

此元素包含此数据源响应查询所花的总时间（以毫秒计），这些查询来自联合服务器启动后或数据库监视计数器上一次重置后（取较晚者）在此联合服务器实例上运行的所有应用程序或单个应用程序。

注: 因为存在查询分块，并非联合服务器检索行的所有尝试都能进行通信处理，所以获取下一行的请求可能要通过返回行块来得到处理。因此，聚集查询响应时间并不总是指示数据源上的处理，但它通常指示数据源或客户机上的处理。

表 1535. 快照监视信息

快照级别	逻辑数据分组	监视开关
数据库	dbase_remote	时间戳记
应用程序	appl_remote	时间戳记

可将快照监视的计数器重置。

用法

使用此元素来确定等待此数据源中的数据所花的实际时间。它在容量规划和容量调整 SYSCAT.SERVERS 中的 CPU 速度和通信速率时很有用。修改这些参数会影响优化器是否将请求发送至数据源。

响应时间是以联合服务器请求数据源中的行的时间与该行可供联合服务器使用的时间之差量度的。

sequence_no -“序号”监视元素

每当工作单元结束（即 COMMIT 或 ROLLBACK 终止工作单元）时，此标识就会递增。**appl_id** 与 **sequence_no** 一起唯一地标识一个事务。

表 1536. 快照监视信息

快照级别	逻辑数据分组	监视开关
应用程序	appl_id_info	基本

表 1536. 快照监视信息 (续)

快照级别	逻辑数据分组	监视开关
DCS 应用程序	dcs_appl_info	基本

表 1537. 事件监视信息

事件类型	逻辑数据分组	监视开关
连接	event_conn	-
连接	event_connheader	-
语句	event_stmt	-
事务	event_xact	-
死锁	event_dlconn	-
带有详细信息的死锁	event_detailed_dlconn	-
带有详细信息的死锁的历史纪录	event_detailed_dlconn	-
带有详细信息的死锁的历史纪录	event_stmt_history	-
带有详细信息的死锁的历史记录值	event_detailed_dlconn	-
带有详细信息的死锁的历史记录值	event_stmt_history	-

sequence_no_holding_lk -“挂起锁定的序号”

对某个对象挂起锁定的应用程序的序号，此应用程序正在等待对该对象获取锁定。

元素标识

sequence_no_holding_lk

元素类型

信息

表 1538. 快照监视信息

快照级别	逻辑数据分组	监视开关
应用程序	appl	基本
锁定	appl_lock_list	基本

表 1539. 事件监视信息

事件类型	逻辑数据分组	监视开关
死锁	event_dlconn	始终收集
带有详细信息的死锁	event_detailed_dlconn	始终收集

用法 此标识与 appl_id 配合使用以唯一标识对某个对象挂起锁定的事务，此应用程序正在等待对该对象获取锁定。

server_db2_type -“受监视的（服务器）节点上的数据库管理器类型”

标识要监视的数据库管理器的类型。

表 1540. 快照监视信息

快照级别	逻辑数据分组	监视开关
数据库管理器	collected	基本

用法 它包含数据库管理器的下列配置类型的其中一个:

API 符号常量
命令行处理器输出

sqlf_nt_server
带本地客户机和远程客户机的数据库服务器

sqlf_nt_stand_req
带本地客户机的数据库服务器

API 符号常量是在包含文件 *sqlutil.h* 中定义的。

server_instance_name -“服务器实例名称”

对其获取快照的数据库管理器实例的名称。

元素标识
server_instance_name

元素类型
信息

表 1541. 快照监视信息

快照级别	逻辑数据分组	监视开关
数据库管理器	collected	基本

表 1542. 事件监视信息

事件类型	逻辑数据分组	监视开关
事件日志头	event_log_header	始终收集

用法 如果同一系统上存在多个数据库管理器实例，那么此数据项将用于唯一标识对其发出快照调用的实例。如果要将监视器输出保存在文件或数据库中以便以后进行分析，并且需要区分来自不同数据库管理器实例的数据，那么此信息将会非常有用。

server_platform -“服务器操作系统”

运行数据库服务器的操作系统。

元素标识
server_platform

元素类型
信息

表 1543. 快照监视信息

快照级别	逻辑数据分组	监视开关
数据库	dbase	基本

表 1544. 事件监视信息

事件类型	逻辑数据分组	监视开关
数据库	event_db	始终收集

用法 此元素可用于远程应用程序的问题确定。可在头文件 *sqlmon.h* 中找到此字段的值。

server_prdid -“服务器产品/版本标识”

正在服务器上运行的产品和版本。

表 1545. 快照监视信息

快照级别	逻辑数据分组	监视开关
数据库管理器	collected	基本

表 1546. 事件监视信息

事件类型	逻辑数据分组	监视开关
事件日志头	event_log_header	-

用法 其格式为 PPPVVRRM，其中:

PPP 为 SQL

VV 标识两位版本号（版本只有一位时高位为 0）

RR 标识两位发行版号（发行版只有一位时高位为 0）

M 标识 1 个字符的修改级别（0-9 或 A-Z）

server_version -“服务器版本”

返回该信息的服务器的版本。

表 1547. 快照监视信息

快照级别	逻辑数据分组	监视开关
数据库管理器	collected	基本

用法

此字段标识收集数据库系统监视器信息的数据库服务器的级别。这允许应用程序根据返回数据的服务器级别来解释数据。有效值为:

SQLM_DBMON_VERSION1
数据由 DB2 版本 1 返回

SQLM_DBMON_VERSION2
数据由 DB2 版本 2 返回

SQLM_DBMON_VERSION5
数据由 DB2 通用数据库版本 5 返回

SQLM_DBMON_VERSION5_2
数据由 DB2 通用数据库版本 5.2 返回

SQLM_DBMON_VERSION6
数据由 DB2 通用数据库 V6 返回

SQLM_DBMON_VERSION7
数据由 DB2 通用数据库 V7 返回

SQLM_DBMON_VERSION8
数据由 DB2 通用数据库 V8 返回

SQLM_DBMON_VERSION9
数据由 DB2 Database for Linux, UNIX, and Windows V9 返回

SQLM_DBMON_VERSION9_5
数据由 DB2 Database for Linux, UNIX, and Windows V9.5 返回

service_class_id -"服务类标识"监视元素

服务子类的唯一标识。对于工作单元而言，此标识代表发出此工作单元的连接的相关联工作负载的服务子类标识。

表 1548. 表函数监视信息

表函数	监视元素收集级别
MON_GET_ACTIVITY_DETAILS 表函数 - 获取完整的活动详细信息（在 XML 文档 DETAILS 中报告）	ACTIVITY METRICS BASE
MON_GET_SERVICE_SUBCLASS 表函数 - 获取服务子类度量值	ACTIVITY METRICS BASE
MON_GET_SERVICE_SUBCLASS_DETAILS 表函数 - 获取详细的服务子类度量值	ACTIVITY METRICS BASE
MON_GET_UNIT_OF_WORK 表函数 - 获取工作单元度量值	ACTIVITY METRICS BASE
MON_GET_UNIT_OF_WORK_DETAILS 表函数 - 获取详细的工作单元度量值	ACTIVITY METRICS BASE
MON_SAMPLE_SERVICE_CLASS_METRICS - 获取样本服务类度量值	ACTIVITY METRICS BASE
WLM_GET_WORKLOAD_OCCURRENCE _ACTIVITIES 表函数 - 返回活动列表	ACTIVITY METRICS BASE

表 1549. 事件监视信息

事件类型	逻辑数据分组	监视开关
活动	event_activity（在 details_xml 文档中报告）	ACTIVITY METRICS BASE
统计信息	event_scstats（在 details_xml 文档中报告）	REQUEST METRICS BASE
锁定	-	始终收集
工作单元	-	始终收集

表 1549. 事件监视信息 (续)

事件类型	逻辑数据分组	监视开关
统计信息	event_histogrambin	始终收集
统计信息	event_scstats	始终收集

用法

此元素的值与视图 SYSCAT.SERVICECLASSES 的列 SERVICECLASSID 中的某个值匹配。使用此元素来查找服务子类名或者对来自不同来源的服务子类的信息进行链接。例如，将服务类统计信息与直方图条形记录相连接。

满足以下条件时，此元素的值为 0:

- 在 event_histogrambin 逻辑数据组中报告了该元素。
- 针对服务类以外的对象收集了直方图数据。

service_level -“服务级别”

这是 DB2 实例的当前校正服务级别。

表 1550. 快照监视信息

快照级别	逻辑数据分组	监视开关
数据库管理器	db2	基本

service_subclass_name -“服务子类名”监视元素

服务子类的名称。

表 1551. 表函数监视信息

表函数	监视元素收集级别
MON_GET_SERVICE_SUBCLASS 表函数 - 获取服务子类度量值	ACTIVITY METRICS BASE
MON_GET_SERVICE_SUBCLASS_DETAILS 表函数 - 获取详细的服务子类度量值（在 XML 文档 DETAILS 中报告）	ACTIVITY METRICS BASE
MON_GET_UNIT_OF_WORK 表函数 - 获取工作单元度量值	ACTIVITY METRICS BASE
MON_GET_UNIT_OF_WORK 表函数 - 获取详细的工作单元度量值（在 XML 文档 DETAILS 中报告）	ACTIVITY METRICS BASE
MON_SAMPLE_SERVICE_CLASS_METRICS - 获取样本服务类度量值	ACTIVITY METRICS BASE
WLM_GET_QUEUE_STATS 表函数 - 返回阈值队列统计信息	ACTIVITY METRICS BASE
WLM_GET_SERVICE_CLASS_AGENTS 表函数 - 列示正在服务类中运行的代理程序	ACTIVITY METRICS BASE

表 1551. 表函数监视信息 (续)

表函数	监视元素收集级别
WLM_GET_SERVICE_CLASS_WORKLOAD_OCCURRENCES 表函数 - 列示工作负载出现次数	ACTIVITY METRICS BASE
WLM_GET_SERVICE_SUBCLASS_STATS 表函数 - 返回服务子类的统计信息	ACTIVITY METRICS BASE

表 1552. 事件监视信息

事件类型	逻辑数据分组	监视开关
统计信息	event_scstats（在 details_xml 文档中报告）	REQUEST METRICS BASE
锁定	-	始终收集
工作单元	-	始终收集
活动	event_activity	始终收集
统计信息	event_scstats	始终收集
统计信息	event_qstats	始终收集

用法

将此元素与其他活动元素配合使用来分析活动的行为，或者与其他统计信息元素配合使用来分析服务类或阈值队列。

service_superclass_name -“服务超类名”监视元素

服务超类的名称。

表 1553. 表函数监视信息

表函数	监视元素收集级别
MON_GET_SERVICE_SUBCLASS 表函数 - 获取服务子类度量值	ACTIVITY METRICS BASE
MON_GET_SERVICE_SUBCLASS_DETAILS 表函数 - 获取详细的服务子类度量值（在 XML 文档 DETAILS 中报告）	ACTIVITY METRICS BASE
MON_GET_UNIT_OF_WORK 表函数 - 获取工作单元度量值	ACTIVITY METRICS BASE
MON_GET_UNIT_OF_WORK 表函数 - 获取详细的工作单元度量值（在 XML 文档 DETAILS 中报告）	ACTIVITY METRICS BASE
MON_SAMPLE_SERVICE_CLASS_METRICS - 获取样本服务类度量值	ACTIVITY METRICS BASE
WLM_GET_QUEUE_STATS 表函数 - 返回阈值队列统计信息	ACTIVITY METRICS BASE
WLM_GET_SERVICE_CLASS_AGENTS 表函数 - 列示正在服务类中运行的代理程序	ACTIVITY METRICS BASE

表 1553. 表函数监视信息 (续)

表函数	监视元素收集级别
WLM_GET_SERVICE_CLASS_WORKLOAD_OCCURRENCES 表函数 - 列示工作负载出现次数	ACTIVITY METRICS BASE
WLM_GET_SERVICE_SUBCLASS_STATS 表函数 - 返回服务子类的统计信息	ACTIVITY METRICS BASE
WLM_GET_SERVICE_SUPERCLASS_STATS 表函数 - 返回服务超类的统计信息	ACTIVITY METRICS BASE

表 1554. 事件监视信息

事件类型	逻辑数据分组	监视开关
统计信息	event_scstats（在 details_xml 文档中报告）	REQUEST METRICS BASE
工作单元	-	始终收集
活动	event_activity	始终收集
统计信息	event_scstats	始终收集
统计信息	event_qstats	始终收集

用法

将此元素与其他活动元素配合使用来分析活动的行为，或者与其他统计信息元素配合使用来分析服务类或阈值队列。

session_auth_id -“会话授权标识”监视元素

此应用程序将使用的会话的当前授权标识。为监视工作负载管理活动，此监视元素描述将活动插入到系统中时使用的会话授权标识。

表 1555. 表函数监视信息

表函数	监视元素收集级别
MON_GET_CONNECTION 表函数 - 获取连接度量值	ACTIVITY METRICS BASE
MON_GET_CONNECTION_DETAILS 表函数 - 获取详细的连接度量值（在 XML 文档 DETAILS 中报告）	ACTIVITY METRICS BASE
MON_GET_UNIT_OF_WORK 表函数 - 获取工作单元度量值	ACTIVITY METRICS BASE
MON_GET_UNIT_OF_WORK 表函数 - 获取详细的工作单元度量值（在 XML 文档 DETAILS 中报告）	ACTIVITY METRICS BASE
WLM_GET_SERVICE_CLASS_AGENTS 表函数 - 列示正在服务类中运行的代理程序	ACTIVITY METRICS BASE
WLM_GET_SERVICE_CLASS_WORKLOAD_OCCURRENCES 表函数 - 列示工作负载出现次数	ACTIVITY METRICS BASE

表 1556. 快照监视信息

快照级别	逻辑数据分组	监视开关
应用程序	appl_info	基本
锁定	appl_lock_list	基本

表 1557. 事件监视信息

事件类型	逻辑数据分组	监视元素收集级别
工作单元	-	始终收集
活动	event_activity	始终收集
阈值违例	event_activity	始终收集
变更历史记录	changesummary	始终收集

用法

可使用此元素来确定要用于预编译 SQL 语句和/或执行 SQL 语句的授权标识。此监视元素不报告在执行存储过程内设置的任何会话授权标识值。

shr_workspace_num_overflows -“共享工作空间溢出数”

共享工作空间溢出其分配内存边界的次数。

注: 不推荐使用此监视元素。使用此监视元素不会生成错误。但是不会返回有效值。建议不要再使用此监视元素，将来的发行版中可能会将其除去。

表 1558. 快照监视信息

快照级别	逻辑数据分组	监视开关
数据库	dbase	基本
应用程序	appl	基本

可将快照监视的计数器重置。

表 1559. 事件监视信息

事件类型	逻辑数据分组	监视开关
数据库	event_db	始终收集
连接	event_conn	始终收集

用法 将此元素与 shr_workspace_size_top 配合使用来确定是否需要增加共享工作空间的大小以避免溢出。共享工作空间的溢出可能导致性能下降，以及从应用程序共享内存分配的其他堆出现内存不足错误。

在数据库级别，报告的元素将来自报告具有最大共享工作空间大小的元素的共享工作空间。在应用程序级别，此项是当前应用程序使用的工作空间的溢出数。

shr_workspace_section_inserts -“共享工作空间节插入数”

应用程序将 SQL 节插入到共享工作空间中的次数。

注: 不推荐使用此监视元素。使用此监视元素不会生成错误。但是不会返回有效值。建议不要再使用此监视元素，将来的发行版中可能会将其除去。

表 1560. 快照监视信息

快照级别	逻辑数据分组	监视开关
数据库	dbase	基本
应用程序	appl	基本

可将快照监视的计数器重置。

表 1561. 事件监视信息

事件类型	逻辑数据分组	监视开关
数据库	event_db	始终收集
连接	event_conn	始终收集

用法 可执行段的工作副本存储在共享工作空间中。此计数器指示副本何时不可用并且必须插入。

在数据库级别，此项是针对数据库的所有共享工作空间中的每个应用程序进行的所有插入的累积总数。在应用程序级别，此项是针对此应用程序的共享工作空间中的所有段的所有插入的累积总数。

shr_workspace_section_lookups -“共享工作空间节查询数”

应用程序在共享工作空间中对 SQL 节进行查询的次数。

注: 不推荐使用此监视元素。使用此监视元素不会生成错误。但是不会返回有效值。建议不要再使用此监视元素，将来的发行版中可能会将其除去。

表 1562. 快照监视信息

快照级别	逻辑数据分组	监视开关
数据库	dbase	基本
应用程序	appl	基本

可将快照监视的计数器重置。

表 1563. 事件监视信息

事件类型	逻辑数据分组	监视开关
数据库	event_db	始终收集
连接	event_conn	始终收集

用法 每个应用程序都可以访问共享工作空间，该工作空间中保留了可执行段的工作副本。

此计数器指示为找到应用程序的特定段而访问共享工作空间的次数。在数据库级别，此项是针对数据库的所有共享工作空间中的每个应用程序进行的所有查询的累积总数。在应用程序级别，此项是针对此应用程序的共享工作空间中的所有段的所有查询的累积总数。

可将此元素与“共享工作空间节插入数”一起使用来调整共享工作空间的大小。共享工作空间的大小由 app_ctl_heap_sz 配置参数控制。

shr_workspace_size_top -“最大共享工作空间大小”

共享工作空间达到的最大大小。

注: 不推荐使用此监视元素。使用此监视元素不会生成错误。但是不会返回有效值。建议不要再使用此监视元素，将来的发行版中可能会将其除去。

表 1564. 快照监视信息

快照级别	逻辑数据分组	监视开关
数据库	dbase	基本
应用程序	appl	基本

表 1565. 事件监视信息

事件类型	逻辑数据分组	监视开关
数据库	event_db	始终收集
连接	event_conn	始终收集

用法 此元素指示对激活后的数据库运行工作负载所需的共享工作空间最多字节数。在数据库级别，此项是所有共享工作空间达到的最大大小。 在应用程序级别，此项是当前应用程序使用的共享工作空间的最大大小。

如果共享工作空间溢出，那么表示此元素包含溢出期间共享工作空间达到的最大大小。检查共享工作空间溢出数以确定是否存在此情况。

工作空间溢出时，将会临时从应用程序共享内存的其他实体借出内存。这可能导致这些实体出现内存不足错误，也可能导致性能下降。可通过增加 APP_CTL_HEAP_SZ 来降低溢出的机率。

skipped_prefetch_data_p_reads -“跳过的预取数据物理读取数”监视元素

I/O 服务器（预取程序）跳过的数据页数（因为已将这些页装入到缓冲池中）。

表 1566. 表函数监视信息

表函数	监视元素收集级别
MON_GET_BUFFERPOOL 表函数 - 获取缓冲池度量值	DATA OBJECT METRICS BASE
MON_GET_TABLESPACE 表函数 - 获取表空间度量值	DATA OBJECT METRICS BASE

用法

此监视元素与其他 **skipped_prefetch_*_p_reads** 元素一起说明已安排预取程序检索某页但因为该页已在缓冲池中而未预取该页的次数。已在缓冲池中的页在该处的可能原因包括:

- 该页是新页，并且尚未在磁盘上创建该页。

- 另一代理程序可能需要同一页，因此另一预取请求已将该页装入到缓冲池中。在此情况及上一情况下，所跳过预取请求的增加可能并非问题，因为所生成的额外预取请求是冗余的。
- 在预取程序之前直接从磁盘中检索这些页的代理程序能够完成预取操作。如果系统未配置足够的预取程序或者存在另一类型的预取瓶颈，那么系统可能会强制代理程序直接从磁盘读取页。. 例如，在 OLTP 系统（其中工作负载的大部分本质上通常是事务性的）中，可能会通过将配置参数 **num_ioservers** 设置为 1 来配置最低的预取程序数。但是，如果执行使用预取的操作（例如，表扫描），那么单个预取程序可能无法跟上，所以代理程序直接请求这些页。此行为可能导致性能下降，因为应用程序等待本应由预取程序执行的 IO。在此情况下，请考虑调整配置参数 **num_ioservers** 以增加预取程序数。其他潜在原因包括预取大小过大（可能导致预取时间比正常情况长）或未设置 **db2_parallel_io** 注册表变量（这可能会将并行预取限制在表空间容器内）。

skipped_prefetch_*_p_reads 元素说明所有已跳过读取请求数，不管跳过读取的原因如何都是如此。要查看因为同一工作单元中在预取程序之前执行读取的代理程序能够检索该页而跳过的请求数，请检查 **skipped_prefetch_uow_*_p_reads** 监视元素。

skipped_prefetch_index_p_reads -“跳过的预取索引物理读取数”监视元素

I/O 服务器（预取程序）跳过的索引页数（因为已将这些页装入到缓冲池中）。

表 1567. 表函数监视信息

表函数	监视元素收集级别
MON_GET_BUFFERPOOL 表函数 - 获取缓冲池度量值	DATA OBJECT METRICS BASE
MON_GET_TABLESPACE 表函数 - 获取表空间度量值	DATA OBJECT METRICS BASE

用法

此监视元素与其他 **skipped_prefetch_*_p_reads** 元素一起说明已安排预取程序检索某页但因为该页已在缓冲池中而未预取该页的次数。已在缓冲池中的页在该处的可能原因包括:

- 该页是新页，并且尚未在磁盘上创建该页。
- 另一代理程序可能需要同一页，因此另一预取请求已将该页装入到缓冲池中。在此情况及上一情况下，所跳过预取请求的增加可能并非问题，因为所生成的额外预取请求是冗余的。
- 在预取程序之前直接从磁盘中检索这些页的代理程序能够完成预取操作。如果系统未配置足够的预取程序或者存在另一类型的预取瓶颈，那么系统可能会强制代理程序直接从磁盘读取页。. 例如，在 OLTP 系统（其中工作负载的大部分本质上通常是事务性的）中，可能会通过将配置参数 **num_ioservers** 设置为 1 来配置最低的预取程序数。但是，如果执行使用预取的操作（例如，表扫描），那么单个预取程序可能无法跟上，所以代理程序直接请求这些页。此行为可能导致性能下降，因为应用程序等待本应由预取程序执行的 IO。在此情况下，请考虑调整配置参数 **num_ioservers** 以增加预取程序数。其他潜在原因包括预取大小过大（可能导致预取时间比正常情况长）或未设置 **db2_parallel_io** 注册表变量（这可能会将并行预取限制在表空间容器内）。

skipped_prefetch_*_p_reads 元素说明所有已跳过读取请求数，不管跳过读取的原因如何都是如此。要查看因为同一工作单元中在预取程序之前执行读取的代理程序能够检索该页而跳过的请求数，请检查 **skipped_prefetch_uow_*_p_reads** 监视元素。

skipped_prefetch_temp_data_p_reads -“跳过的预取临时数据物理读取数”监视元素

对于临时表空间，I/O 服务器（预取程序）跳过的数据页数（因为已将这些页装入到缓冲池中）。

表 1568. 表函数监视信息

表函数	监视元素收集级别
MON_GET_BUFFERPOOL 表函数 - 获取缓冲池度量值	DATA OBJECT METRICS BASE
MON_GET_TABLESPACE 表函数 - 获取表空间度量值	DATA OBJECT METRICS BASE

用法

此监视元素与其他 **skipped_prefetch_*_p_reads** 元素一起说明已安排预取程序检索某页但因为该页已在缓冲池中而未预取该页的次数。已在缓冲池中的页在该处的可能原因包括:

- 该页是新页，并且尚未在磁盘上创建该页。
- 另一代理程序可能需要同一页，因此另一预取请求已将该页装入到缓冲池中。在此情况及上一情况下，所跳过预取请求的增加可能并非问题，因为所生成的额外预取请求是冗余的。
- 在预取程序之前直接从磁盘中检索这些页的代理程序能够完成预取操作。如果系统未配置足够的预取程序或者存在另一类型的预取瓶颈，那么系统可能会强制代理程序直接从磁盘读取页。. 例如，在 OLTP 系统（其中工作负载的大部分本质上通常是事务性的）中，可能会通过将配置参数 **num_ioservers** 设置为 1 来配置最低的预取程序数。但是，如果执行使用预取的操作（例如，表扫描），那么单个预取程序可能无法跟上，所以代理程序直接请求这些页。此行为可能导致性能下降，因为应用程序等待本应由预取程序执行的 IO。在此情况下，请考虑调整配置参数 **num_ioservers** 以增加预取程序数。其他潜在原因包括预取大小过大（可能导致预取时间比正常情况长）或未设置 **db2_parallel_io** 注册表变量（这可能会将并行预取限制在表空间容器内）。

skipped_prefetch_*_p_reads 元素说明所有已跳过读取请求数，不管跳过读取的原因如何都是如此。要查看因为同一工作单元中在预取程序之前执行读取的代理程序能够检索该页而跳过的请求数，请检查 **skipped_prefetch_uow_*_p_reads** 监视元素。

skipped_prefetch_temp_index_p_reads -“跳过的预取临时索引物理读取数”监视元素

对于临时表空间，I/O 服务器（预取程序）跳过的索引页数（因为已将这些页装入到缓冲池中）。

表 1569. 表函数监视信息

表函数	监视元素收集级别
MON_GET_BUFFERPOOL 表函数 - 获取缓冲池度量值	DATA OBJECT METRICS BASE
MON_GET_TABLESPACE 表函数 - 获取表空间度量值	DATA OBJECT METRICS BASE

用法

此监视元素与其他 **skipped_prefetch_*_p_reads** 元素一起说明已安排预取程序检索某页但因为该页已在缓冲池中而未预取该页的次数。已在缓冲池中的页在该处的可能原因包括:

- 该页是新页，并且尚未在磁盘上创建该页。
- 另一代理程序可能需要同一页，因此另一预取请求已将该页装入到缓冲池中。在此情况及上一情况下，所跳过预取请求的增加可能并非问题，因为所生成的额外预取请求是冗余的。
- 在预取程序之前直接从磁盘中检索这些页的代理程序能够完成预取操作。如果系统未配置足够的预取程序或者存在另一类型的预取瓶颈，那么系统可能会强制代理程序直接从磁盘读取页。. 例如，在 OLTP 系统（其中工作负载的大部分本质上通常是事务性的）中，可能会通过将配置参数 **num_ioservers** 设置为 1 来配置最低的预取程序数。但是，如果执行使用预取的操作（例如，表扫描），那么单个预取程序可能无法跟上，所以代理程序直接请求这些页。此行为可能导致性能下降，因为应用程序等待本应由预取程序执行的 IO。在此情况下，请考虑调整配置参数 **num_ioservers** 以增加预取程序数。其他潜在原因包括预取大小过大（可能导致预取时间比正常情况长）或未设置 **db2_parallel_io** 注册表变量（这可能会将并行预取限制在表空间容器内）。

skipped_prefetch_*_p_reads 元素说明所有已跳过读取请求数，不管跳过读取的原因如何都是如此。要查看因为同一工作单元中在预取程序之前执行读取的代理程序能够检索该页而跳过的请求数，请检查 **skipped_prefetch_uow_*_p_reads** 监视元素。

skipped_prefetch_temp_xda_p_reads -“跳过的预取临时 XDA 数据物理读取数”监视元素

对于临时表空间，I/O 服务器（预取程序）跳过的 XML 存储器对象 (XDA) 数据页数（因为已将这些页装入到缓冲池中）。

表 1570. 表函数监视信息

表函数	监视元素收集级别
MON_GET_BUFFERPOOL 表函数 - 获取缓冲池度量值	DATA OBJECT METRICS BASE

表 1570. 表函数监视信息 (续)

表函数	监视元素收集级别
MON_GET_TABLESPACE 表函数 - 获取表空间度量值	DATA OBJECT METRICS BASE

用法

此监视元素与其他 **skipped_prefetch_*_p_reads** 元素一起说明已安排预取程序检索某页但因为该页已在缓冲池中而未预取该页的次数。已在缓冲池中的页在该处的可能原因包括:

- 该页是新页，并且尚未在磁盘上创建该页。
- 另一代理程序可能需要同一页，因此另一预取请求已将该页装入到缓冲池中。在此情况及上一情况下，所跳过预取请求的增加可能并非问题，因为所生成的额外预取请求是冗余的。
- 在预取程序之前直接从磁盘中检索这些页的代理程序能够完成预取操作。如果系统未配置足够的预取程序或者存在另一类型的预取瓶颈，那么系统可能会强制代理程序直接从磁盘读取页。. 例如，在 OLTP 系统（其中工作负载的大部分本质上通常是事务性的）中，可能会通过将配置参数 **num_ioservers** 设置为 1 来配置最低的预取程序数。但是，如果执行使用预取的操作（例如，表扫描），那么单个预取程序可能无法跟上，所以代理程序直接请求这些页。此行为可能导致性能下降，因为应用程序等待本应由预取程序执行的 IO。在此情况下，请考虑调整配置参数 **num_ioservers** 以增加预取程序数。其他潜在原因包括预取大小过大（可能导致预取时间比正常情况长）或未设置 **db2_parallel_io** 注册表变量（这可能会将并行预取限制在表空间容器内）。

skipped_prefetch_*_p_reads 元素说明所有已跳过读取请求数，不管跳过读取的原因如何都是如此。要查看因为同一工作单元中在预取程序之前执行读取的代理程序能够检索该页而跳过的请求数，请检查 **skipped_prefetch_uow_*_p_reads** 监视元素。

skipped_prefetch_uow_data_p_reads -“跳过的预取工作单元数据物理读取数”监视元素

I/O 服务器（预取程序）跳过的数据页数（因为同一工作单元中的代理程序已将这些页装入到缓冲池中）。

表 1571. 表函数监视信息

表函数	监视元素收集级别
MON_GET_BUFFERPOOL 表函数 - 获取缓冲池度量值	DATA OBJECT METRICS BASE
MON_GET_TABLESPACE 表函数 - 获取表空间度量值	DATA OBJECT METRICS BASE

用法

此监视元素与其他 **skipped_prefetch_uow_*_p_reads** 元素一起说明预取请求中由同一工作单元（导致创建该预取请求）中的代理程序直接读取的页数。如果系统未配置足够的预取程序或者存在另一类型的预取瓶颈，那么系统可能会强制代理程序直接从磁

盘读取页。. 例如，在 OLTP 系统（其中工作负载的大部分本质上通常是事务性的）中，可能会通过将配置参数 **num_ioservers** 设置为 1 来配置最低的预取程序数。但是，如果执行使用预取的操作（例如，表扫描），那么单个预取程序可能无法跟上，所以代理程序直接请求这些页。此行为可能导致性能下降，因为应用程序等待本应由预取程序执行的 IO。在此情况下，请考虑调整配置参数 **num_ioservers** 以增加预取程序数。其他潜在原因包括预取大小过大（可能导致预取时间比正常情况长）或未设置 **db2_parallel_io** 注册表变量（这可能会将并行预取限制在表空间容器内）。

skipped_prefetch_uow_index_p_reads -“跳过的预取工作单元索引物理读取数”监视元素

I/O 服务器（预取程序）跳过的索引页数（因为同一工作单元中的代理程序已将这些页装入到缓冲池中）。

表 1572. 表函数监视信息

表函数	监视元素收集级别
MON_GET_BUFFERPOOL 表函数 - 获取缓冲池度量值	DATA OBJECT METRICS BASE
MON_GET_TABLESPACE 表函数 - 获取表空间度量值	DATA OBJECT METRICS BASE

用法

此监视元素与其他 **skipped_prefetch_uow_*_p_reads** 元素一起说明预取请求中由同一工作单元（导致创建该预取请求）中的代理程序直接读取的页数。如果系统未配置足够的预取程序或者存在另一类型的预取瓶颈，那么系统可能会强制代理程序直接从磁盘读取页。. 例如，在 OLTP 系统（其中工作负载的大部分本质上通常是事务性的）中，可能会通过将配置参数 **num_ioservers** 设置为 1 来配置最低的预取程序数。但是，如果执行使用预取的操作（例如，表扫描），那么单个预取程序可能无法跟上，所以代理程序直接请求这些页。此行为可能导致性能下降，因为应用程序等待本应由预取程序执行的 IO。在此情况下，请考虑调整配置参数 **num_ioservers** 以增加预取程序数。其他潜在原因包括预取大小过大（可能导致预取时间比正常情况长）或未设置 **db2_parallel_io** 注册表变量（这可能会将并行预取限制在表空间容器内）。

skipped_prefetch_uow_temp_data_p_reads -“跳过的预取工作单元临时数据物理读取数”监视元素

I/O 服务器（预取程序）跳过的临时表空间数据页数（因为同一工作单元中的代理程序已将这些页装入到缓冲池中）。

表 1573. 表函数监视信息

表函数	监视元素收集级别
MON_GET_BUFFERPOOL 表函数 - 获取缓冲池度量值	DATA OBJECT METRICS BASE
MON_GET_TABLESPACE 表函数 - 获取表空间度量值	DATA OBJECT METRICS BASE

用法

此监视元素与其他 **skipped_prefetch_uow_*_p_reads** 元素一起说明预取请求中由同一工作单元（导致创建该预取请求）中的代理程序直接读取的页数。如果系统未配置足够的预取程序或者存在另一类型的预取瓶颈，那么系统可能会强制代理程序直接从磁盘读取页。. 例如，在 OLTP 系统（其中工作负载的大部分本质上通常是事务性的）中，可能会通过将配置参数 **num_ioservers** 设置为 1 来配置最低的预取程序数。但是，如果执行使用预取的操作（例如，表扫描），那么单个预取程序可能无法跟上，所以代理程序直接请求这些页。此行为可能导致性能下降，因为应用程序等待本应由预取程序执行的 IO。在此情况下，请考虑调整配置参数 **num_ioservers** 以增加预取程序数。其他潜在原因包括预取大小过大（可能导致预取时间比正常情况长）或未设置 **db2_parallel_io** 注册表变量（这可能会将并行预取限制在表空间容器内）。

skipped_prefetch_uow_temp_index_p_reads -“跳过的预取工作单元临时索引物理读取数”监视元素

对于临时表空间，I/O 服务器（预取程序）跳过的索引页数（因为同步事务已将这些页装入到缓冲池中）。

表 1574. 表函数监视信息

表函数	监视元素收集级别
MON_GET_BUFFERPOOL 表函数 - 获取缓冲池度量值	DATA OBJECT METRICS BASE
MON_GET_TABLESPACE 表函数 - 获取表空间度量值	DATA OBJECT METRICS BASE

skipped_prefetch_uow_temp_xda_p_reads -“跳过的预取工作单元临时 XDA 数据物理读取数”监视元素

对于临时表空间，I/O 服务器（预取程序）跳过的 XML 存储器对象 (XDA) 数据页数（因为同步事务已将这些页装入到缓冲池中）。

表 1575. 表函数监视信息

表函数	监视元素收集级别
MON_GET_BUFFERPOOL 表函数 - 获取缓冲池度量值	DATA OBJECT METRICS BASE
MON_GET_TABLESPACE 表函数 - 获取表空间度量值	DATA OBJECT METRICS BASE

skipped_prefetch_uow_xda_p_reads -“跳过的预取工作单元 XDA 数据物理读取数”监视元素

I/O 服务器（预取程序）跳过的 XML 存储器对象 (XDA) 数据页数（因为同一工作单元中的代理程序已将这些页装入到缓冲池中）。

表 1576. 表函数监视信息

表函数	监视元素收集级别
MON_GET_BUFFERPOOL 表函数 - 获取缓冲池度量值	DATA OBJECT METRICS BASE
MON_GET_TABLESPACE 表函数 - 获取表空间度量值	DATA OBJECT METRICS BASE

用法

此监视元素与其他 **skipped_prefetch_uow_*_p_reads** 元素一起说明预取请求中由同一工作单元（导致创建该预取请求）中的代理程序直接读取的页数。如果系统未配置足够的预取程序或者存在另一类型的预取瓶颈，那么系统可能会强制代理程序直接从磁盘读取页。. 例如，在 OLTP 系统（其中工作负载的大部分本质上通常是事务性的）中，可能会通过将配置参数 **num_ioservers** 设置为 1 来配置最低的预取程序数。但是，如果执行使用预取的操作（例如，表扫描），那么单个预取程序可能无法跟上，所以代理程序直接请求这些页。此行为可能导致性能下降，因为应用程序等待本应由预取程序执行的 IO。在此情况下，请考虑调整配置参数 **num_ioservers** 以增加预取程序数。其他潜在原因包括预取大小过大（可能导致预取时间比正常情况长）或未设置 **db2_parallel_io** 注册表变量（这可能会将并行预取限制在表空间容器内）。

skipped_prefetch_xda_p_reads -“跳过的预取 XDA 物理读取数”监视元素

I/O 服务器（预取程序）跳过的 XML 存储器对象 (XDA) 数据页数（因为已将这些页装入到缓冲池中）。

表 1577. 表函数监视信息

表函数	监视元素收集级别
MON_GET_BUFFERPOOL 表函数 - 获取缓冲池度量值	DATA OBJECT METRICS BASE
MON_GET_TABLESPACE 表函数 - 获取表空间度量值	DATA OBJECT METRICS BASE

用法

此监视元素与其他 **skipped_prefetch_*_p_reads** 元素一起说明已安排预取程序检索某页但因为该页已在缓冲池中而未预取该页的次数。已在缓冲池中的页在该处的可能原因包括:

- 该页是新页，并且尚未在磁盘上创建该页。
- 另一代理程序可能需要同一页，因此另一预取请求已将该页装入到缓冲池中。在此情况及上一情况下，所跳过预取请求的增加可能并非问题，因为所生成的额外预取请求是冗余的。

- 在预取程序之前直接从磁盘中检索这些页的代理程序能够完成预取操作。如果系统未配置足够的预取程序或者存在另一类型的预取瓶颈，那么系统可能会强制代理程序直接从磁盘读取页。. 例如，在 OLTP 系统（其中工作负载的大部分本质上通常是事务性的）中，可能会通过将配置参数 **num_ioservers** 设置为 1 来配置最低的预取程序数。但是，如果执行使用预取的操作（例如，表扫描），那么单个预取程序可能无法跟上，所以代理程序直接请求这些页。此行为可能导致性能下降，因为应用程序等待本应由预取程序执行的 IO。在此情况下，请考虑调整配置参数 **num_ioservers** 以增加预取程序数。其他潜在原因包括预取大小过大（可能导致预取时间比正常情况长）或未设置 **db2_parallel_io** 注册表变量（这可能会将并行预取限制在表空间容器内）。

skipped_prefetch_*_p_reads 元素说明所有已跳过读取请求数，不管跳过读取的原因如何都是如此。要查看因为同一工作单元中在预取程序之前执行读取的代理程序能够检索该页而跳过的请求数，请检查 **skipped_prefetch_uow_*_p_reads** 监视元素。

smallest_log_avail_node -“带有最少可用日志空间的节点”

此元素指示带有最少可用日志空间量（以字节计）的节点并且仅对全局快照返回。

元素标识

smallest_log_avail_node

元素类型

信息

表 1578. 快照监视信息

快照级别	逻辑数据分组	监视开关
数据库	dbase	基本

用法 将此元素与 appl_id_oldest_xact 一起使用以确保数据库有足够的日志空间可用。在全局快照中，appl_id_oldest_xact、total_log_used 和 total_log_available 对应于此节点上的值。

sort_heap_allocated -“分配的总排序堆”

排序堆空间的总分配页数，用于获取快照时处于所选级别的所有排序。

表 1579. 快照监视信息

快照级别	逻辑数据分组	监视开关
数据库管理器	db2	基本
数据库	dbase	基本

用法 为每个排序分配的内存量可以是可用排序堆大小的一部分或全部。排序堆大小是按 *sortheap* 数据库配置参数中定义的可用于每个排序的内存量。

单个应用程序可使多个排序同时处于活动状态。例如，在某些情况下带有子查询的 SELECT 语句可能导致并行排序。

可在两个级别收集信息:

- 在数据库管理器级别，它表示为数据库管理器的所有活动数据库中的所有排序分配的排序堆空间的总和

- 在数据库级别，它表示为数据库中的所有排序分配的排序堆空间的总和。

常规内存估计不包括排序堆空间。如果出现过多排序，除了需要用于运行数据库管理器的基本内存之外，还应额外添加用于排序堆的内存。通常排序堆越大，排序越有效。适当使用索引可以降低必需的排序量。

可使用在数据库管理器级别返回的信息来帮助您调整 *sheapthres* 配置参数。如果元素值大于或等于 *sheapthres*，那么表示排序未达到 *sortheap* 参数定义的排序堆已满的状态。

sort_heap_top -“排序专用堆高水位标记”

数据库管理器范围的专用排序内存高水位标记（以 4 KB 页计）。

表 1580. 快照监视信息

快照级别	逻辑数据分组	监视开关
数据库管理器	db2	基本

用法 此元素可用来确定是否已将 SHEAPTHRES 配置参数设置为最佳的值。例如，如果此水位标记接近或超过 SHEAPTHRES，那么可能应该增大 SHEAPTHRES。这是因为，超过 SHEAPTHRES 后，专用排序操作可使用的内存就会减少，这会对系统性能产生不利影响。

sort_overflows -“排序溢出数”监视元素

用完排序堆并且可能需要磁盘空间以供临时存储器使用的排序总数。

表 1581. 表函数监视信息

表函数	监视元素收集级别
MON_FORMAT_XML_METRICS_BY_ROW - 获取所有度量值的基于已格式化行的输出	不适用；报告 XML 文档中作为格式化函数的输入提供的所有元素
MON_GET_ACTIVITY_DETAILS 表函数 - 获取完整的活动详细信息（在 XML 文档 DETAILS 中报告）	ACTIVITY METRICS BASE
MON_GET_CONNECTION 表函数 - 获取连接度量值	REQUEST METRICS BASE
MON_GET_CONNECTION_DETAILS 表函数 - 获取详细的连接度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE
MON_GET_PKG_CACHE_STMT 表函数 - 获取程序包高速缓存中的 SQL 语句活动度量值	ACTIVITY METRICS BASE
MON_GET_PKG_CACHE_STMT_DETAILS 表函数 - 获取程序包高速缓存条目的详细度量值	ACTIVITY METRICS BASE
MON_GET_SERVICE_SUBCLASS 表函数 - 获取服务子类度量值	REQUEST METRICS BASE
MON_GET_SERVICE_SUBCLASS_DETAILS 表函数 - 获取详细的服务子类度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE

表 1581. 表函数监视信息 (续)

表函数	监视元素收集级别
MON_GET_UNIT_OF_WORK 表函数 - 获取工作单元度量值	REQUEST METRICS BASE
MON_GET_UNIT_OF_WORK 表函数 - 获取详细的工作单元度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE
MON_GET_WORKLOAD 表函数 - 获取工作负载度量值	REQUEST METRICS BASE
MON_GET_WORKLOAD_DETAILS 表函数 - 获取详细的工作负载度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE

表 1582. 快照监视信息

快照级别	逻辑数据分组	监视开关
数据库	dbase	基本
应用程序	appl	基本
应用程序	stmt	基本
动态 SQL	dynsql	基本

可将快照监视的计数器重置。

表 1583. 事件监视信息

事件类型	逻辑数据分组	监视开关
活动	event_activity（在 details_xml 文档中报告）	ACTIVITY METRICS BASE
活动	event_activitymetrics	ACTIVITY METRICS BASE
统计信息	event_scstats（在 details_xml 文档中报告）	REQUEST METRICS BASE
统计信息	event_wlstats（在 details_xml 文档中报告）	REQUEST METRICS BASE
工作单元	在 system_metrics 文档中报告。	REQUEST METRICS BASE
数据库	event_db	始终收集
连接	event_conn	始终收集
语句	event_stmt	始终收集
活动	event_activity	语句，排序
程序包高速缓存	在 activity_metrics 文档中报告。	ACTIVITY METRICS BASE

用法

在数据库或应用程序级别，将此元素与 **total_sorts** 一起使用来计算必须溢出至磁盘的排序的百分比。如果此百分比很高，那么您可能想要通过增加 **sortheap** 的值来调整数据库配置。

在语句级别，使用此元素来标识需要大量排序的语句。如果另外进行调整以降低所需排序量，这些语句将从中受益。

当排序溢出时，因为排序需要合并阶段并且可能需要更多 I/O（如果数据需要写至磁盘），所以需要额外的处理时间。

此元素提供有关一个语句、一个应用程序或访问一个数据库的所有应用程序的信息。

sort_shrheap_allocated -“对当前分配的共享堆进行排序”

数据库中分配的共享排序内存总量。

表 1584. 快照监视信息

快照级别	逻辑数据分组	监视开关
数据库	dbase	基本

用法 此元素可用来评估共享排序内存的阈值。如果此值经常大幅高于或低于当前共享排序内存阈值，那么可能应该调整阈值。

注: 如果 SHEAPTHRES_SHR 数据库配置参数为 0，那么“共享排序内存阈值”由 SHEAPTHRES 数据库管理器配置参数确定。否则将由 SHEAPTHRES_SHR 的值确定。

sort_shrheap_top -“排序共享堆高水位标记”

数据库范围的共享排序内存高水位标记（以 4 KB 页计）。

表 1585. 快照监视信息

快照级别	逻辑数据分组	监视开关
数据库	dbase	基本

用法 此元素可用来评估是否已将 SHEAPTHRES（或 SHEAPTHRES_SHR）设置为最佳的值。例如，如果这个高水位标记持续地远小于共享排序内存阈值，那么有可能需要减小这个阈值，从而释放内存以供其他数据库功能使用。相反，如果这个高水位标记开始接近共享排序内存阈值，那么可能表示需要增大这个阈值。由于共享排序内存阈值是硬限制，所以这一点很重要。排序内存总量达到这个阈值后，就无法启动更多的共享排序操作。

此元素和专用排序内存高水位标记还可以帮助用户确定是否需要相互独立地设置共享排序阈值和专用排序阈值。通常，如果 SHEAPTHRES_SHR 数据库配置选项值为 0，那么共享排序内存阈值由 SHEAPTHRES 数据库管理器配置选项值确定。但是，如果专用排序内存高水位标记与共享排序内存高水位标记相差很大，那么可能表明用户需要重设 SHEAPTHRES，并将 SHEAPTHRES_SHR 设置为基于共享排序内存高水位标记的更合适的值。

注: 此元素将报告排序内存控制器已授权的排序预留请求的高水位标记。已授权的请求并非始终会导致类似级别的内存分配，这是因为这些请求只允许排序堆的使用者在处理 SQL 请求期间分配必要的内存，不能超过已授权的内存量。此元素的值与共享排序内存池的高水位标记 (pool_watermark) 之间存在差异是正常的。

source_service_class_id -“源服务类标识”监视元素

生成此元素所属的阈值违例记录时从中重新映射此活动的服务子类的标识。对于除 REMAP ACTIVITY 以外的任何阈值操作，此元素的值为零。

表 1586. 事件监视信息

事件类型	逻辑数据分组	监视开关
阈值违例	event_thresholdviolations	-

用法

使用此元素来跟踪活动在它重新映射到的各个服务类之间的路径。此元素还可用于计算从给定服务子类向外映射的活动数的聚集值。

sp_rows_selected -“存储过程返回的行数”

此元素包含自启动联合服务器实例或最后一次重置数据库监视计数器以后，因对此应用程序执行存储过程操作而导致从数据源发送至联合服务器的行数。

表 1587. 快照监视信息

快照级别	逻辑数据分组	监视开关
数据库	dbase_remote	基本
应用程序	appl_remote	基本

可将快照监视的计数器重置。

用法 此元素有若干用途。可使用它并借助以下公式来计算对于每个存储过程而言从数据源发送至联合服务器的平均行数:

```
每个存储过程的行数
= 返回的行数
/ 调用的存储过程数
```

还可以计算对于此应用程序而言行从数据源返回至联合服务器的平均时间:

```
平均时间 = 聚集存储过程响应时间 / 返回的行数
```

sql_chains -“尝试的 SQL 链数”

表示语句处理期间在 DB2 Connect 网关与主机之间采用 *n* 个数据传输的 SQL 语句数。范围 *n* 是由 *num_transmissions_group* 元素指定的。

表 1588. 快照监视信息

快照级别	逻辑数据分组	监视开关
数据传输	stmt_transmissions	基本

可将快照监视的计数器重置。

例如，如果链接已打开，并且 PREP 和 OPEN 语句链接在一起，该链一共占用了两个传输，那么 *sql_chains* 报告为“1”，而 *sql_stmts* 报告为“2”。

如果链接已关闭，那么 *sql_chains* 计数等于 *sql_stmts* 计数。

用法 使用此元素来获取有关处理期间使用 2、3、4（等等）个数据传输的语句数的统计信息。（要处理语句，至少需要 2 个数据传输：发送和接收。）这些统计信息能够让您更好地了解数据库或应用程序级别的数据库或应用程序活动和网络流量。

注： *sql_stmts* 监视元素表示尝试将 SQL 语句发送至服务器的次数。在传输级别，同一游标内的所有语句在计数时都被当作单个 SQL 语句。

sql_req_id -“SQL 语句的请求标识”

SQL 语句中某个操作的请求标识。

表 1589. 事件监视信息

事件类型	逻辑数据分组	监视开关
语句	event_stmt	-

用法 第一个应用程序连接至数据库之后，数据库管理器每次成功处理 SQL 操作时，此标识就会递增。它的值在数据库中是唯一的，可以唯一地标识语句操作。

sql_reqs_since_commit -“上次落实后的 SQL 请求数”

自上次落实后提交的 SQL 请求数。

表 1590. 快照监视信息

快照级别	逻辑数据分组	监视开关
应用程序	appl	基本

用法 可使用此元素来监视事务的进度。

sql_stmts -“尝试的 SQL 语句数”

对于数据传输快照，此元素表示语句处理期间在 DB2 Connect 网关与主机之间采用 *n* 个数据传输的 SQL 语句数。范围 *n* 是由 *num_transmissions_group* 元素指定的。

表 1591. 快照监视信息

快照级别	逻辑数据分组	监视开关
DCS 数据库	dcs_dbase	基本
DCS 应用程序	dcs_appl	基本
数据传输	stmt_transmissions	基本

可将快照监视的计数器重置。

对于 DCS DATABASE 快照，此语句计数为自激活数据库后处理的语句数。

对于 DCS APPLICATION 快照，此语句计数为自此应用程序建立与数据库的连接后处理的语句数。

用法 使用此元素来度量数据库或应用程序级别的数据库活动。要计算给定时间段的 SQL 语句吞吐量，可按两次快照之间所耗用的时间来分割此元素。

对于数据传输级别：使用此元素来获取有关处理期间使用 2、3、4（等等）个数据传输的语句数的统计信息。（要处理语句，至少需要 2 个数据传输：发送和接收。）这些统计信息能够让您更好地了解数据库或应用程序级别的数据库或应用程序活动和网络流量。

注:

1. *sql_stmts* 监视元素表示尝试将 SQL 语句发送至服务器的次数:
   - 在应用程序级别和数据库级别，游标内的每个 SQL 语句是分开计数的。
   - 在传输级别，同一游标内的所有语句在计数时都被当作单个 SQL 语句。

sqlca -“SQL 通信区（SQLCA）”

在语句完成时返回至应用程序的 SQLCA 数据结构。

表 1592. 事件监视信息

事件类型	逻辑数据分组	监视开关
语句	event_stmt	-
活动	event_activity	-

用法

SQLCA 数据结构可用来确定语句是否成功完成。有关 SQLCA 内容的信息，请参阅 *SQL Reference Volume 1* 中的『SQLCA（SQL 通信区）』或 *Administrative API Reference* 中的『SQLCA 数据结构』。

sqlrowsread_threshold_id -“读取 SQL 行数阈值标识”监视元素

应用于此活动的 SQLROWSREAD 阈值的标识。

表 1593. 表函数监视信息

表函数	监视元素收集级别
MON_GET_ACTIVITY_DETAILS 表函数 - 获取完整的活动详细信息（在 XML 文档 DETAILS 中报告）	ACTIVITY METRICS BASE

用法

使用此元素来确定应用于此活动的 SQLROWSREAD 阈值（如果有的话）。

sqlrowsread_threshold_value -“读取 SQL 行数阈值”监视元素

应用于此活动的 SQLROWSREAD 阈值的上限。

表 1594. 表函数监视信息

表函数	监视元素收集级别
MON_GET_ACTIVITY_DETAILS 表函数 - 获取完整的活动详细信息（在 XML 文档 DETAILS 中报告）	ACTIVITY METRICS BASE

用法

使用此元素来确定应用于此活动的 SQLROWSREAD 阈值（如果有的话）。

sqlrowsread_threshold_violated -“违反读取 SQL 行数阈值”监视元素

此监视元素返回“Yes”表明此活动已违反 SQLROWSREAD 阈值。“No”表明此活动尚未违反该阈值。

表 1595. 表函数监视信息

表函数	监视元素收集级别
MON_GET_ACTIVITY_DETAILS 表函数 - 获取完整的活动详细信息（在 XML 文档 DETAILS 中报告）	ACTIVITY METRICS BASE

用法

使用此元素来确定此活动是否已违反应用于此活动的 SQLROWSREAD 阈值。

sqlrowsreadinsc_threshold_id -“服务类中读取 SQL 行数阈值标识”监视元素

应用于此活动的 SQLROWSREADINSC 阈值的标识。

表 1596. 表函数监视信息

表函数	监视元素收集级别
MON_GET_ACTIVITY_DETAILS 表函数 - 获取完整的活动详细信息（在 XML 文档 DETAILS 中报告）	ACTIVITY METRICS BASE

用法

使用此元素来确定应用于此活动的 SQLROWSREADINSC 阈值（如果有的话）。

sqlrowsreadinsc_threshold_value -“服务类中读取 SQL 行数阈值”监视元素

应用于此活动的 SQLROWSREADINSC 阈值的上限。

表 1597. 表函数监视信息

表函数	监视元素收集级别
MON_GET_ACTIVITY_DETAILS 表函数 - 获取完整的活动详细信息（在 XML 文档 DETAILS 中报告）	ACTIVITY METRICS BASE

用法

使用此元素来确定应用于此活动的 SQLROWSREADINSC 阈值（如果有的话）。

sqlrowsreadinsc_threshold_violated -“违反服务类中读取 SQL 行数阈值”监视元素

此监视元素返回“Yes”表明此活动已违反 SQLROWSREADINSC 阈值。“No”表明此活动尚未违反该阈值。

表 1598. 表函数监视信息

表函数	监视元素收集级别
MON_GET_ACTIVITY_DETAILS 表函数 - 获取完整的活动详细信息（在 XML 文档 DETAILS 中报告）	ACTIVITY METRICS BASE

用法

使用此元素来确定此活动是否已违反应用于此活动的 SQLROWSREADINSC 阈值。

sqlrowsreturned_threshold_id -“返回所读取 SQL 行数阈值标识”监视元素

应用于此活动的 SQLROWSRETURNED 阈值的标识。

表 1599. 表函数监视信息

表函数	监视元素收集级别
MON_GET_ACTIVITY_DETAILS 表函数 - 获取完整的活动详细信息（在 XML 文档 DETAILS 中报告）	ACTIVITY METRICS BASE

用法

使用此元素来确定应用于此活动的 SQLROWSRETURNED 阈值（如果有的话）。

sqlrowsreturned_threshold_value -“返回所读取 SQL 行数阈值”监视元素

应用于此活动的 SQLROWSRETURNED 阈值的上限。

表 1600. 表函数监视信息

表函数	监视元素收集级别
MON_GET_ACTIVITY_DETAILS 表函数 - 获取完整的活动详细信息（在 XML 文档 DETAILS 中报告）	ACTIVITY METRICS BASE

用法

使用此元素来确定应用于此活动的 SQLROWSRETURNED 阈值（如果有的话）。

sqlrowsreturned_threshold_violated -“违反返回所读取 SQL 行数阈值”监视元素

此监视元素返回“Yes”表明此活动已违反 SQLROWSRETURNED 阈值。“No”表明此活动尚未违反该阈值。

表 1601. 表函数监视信息

表函数	监视元素收集级别
MON_GET_ACTIVITY_DETAILS 表函数 - 获取完整的活动详细信息（在 XML 文档 DETAILS 中报告）	ACTIVITY METRICS BASE

用法

使用此元素来确定此活动是否已违反应用于此活动的 SQLROWSRETURNED 阈值。

sqltempspace_threshold_id -“SQL 临时空间阈值标识”监视元素

应用于此活动的 SQLTEMPSPACE 阈值的标识。

表 1602. 表函数监视信息

表函数	监视元素收集命令和级别
MON_GET_ACTIVITY_DETAILS 表函数 - 获取完整的活动详细信息（在 XML 文档 DETAILS 中报告）	ACTIVITY METRICS BASE

用法

使用此元素来确定应用于此活动的 SQLTEMPSPACE 阈值（如果有的话）。

sqltempspace_threshold_value -“SQL 临时空间阈值”监视元素

应用于此活动的 SQLTEMPSPACE 阈值的上限。

表 1603. 表函数监视信息

表函数	监视元素收集级别
MON_GET_ACTIVITY_DETAILS 表函数 - 获取完整的活动详细信息（在 XML 文档 DETAILS 中报告）	ACTIVITY METRICS BASE

用法

使用此元素来确定应用于此活动的 SQLTEMPSPACE 阈值（如果有的话）。

sqltempspace_threshold_violated -“违反 SQL 临时空间阈值”监视元素

此监视元素返回“Yes”表明此活动已违反 SQLTEMPSPACE 阈值。“No”表明此活动尚未违反该阈值。

表 1604. 表函数监视信息

表函数	监视元素收集级别
MON_GET_ACTIVITY_DETAILS 表函数 - 获取完整的活动详细信息（在 XML 文档 DETAILS 中报告）	ACTIVITY METRICS BASE

用法

使用此元素来确定此活动是否已违反应用于此活动的 SQLTEMPSPACE 阈值。

spacemappage_page_reclaims_x -“互斥存取导致的空间映射页回收数”监视元素

与空间映射页相关的页被 DB2 pureScale实例中的另一成员回收的次数（在计划释放此页之前）。回收此页的成员需要对空间映射页具有互斥存取权。

表 1605. 表函数监视信息

表函数	监视元素收集级别
MON_GET_PAGE_ACCESS_INFO 表函数 - 获取缓冲池页等待信息	ACTIVITY METRICS BASE

用法

仅对与对象相关的表空间（即，对可回收存储器启用的表空间）报告此值。使用 **reclaimable_space_enabled** 监视元素来确定是否对可回收存储器启用了此表空间。

因为扩展数据块映射页 (EMP) 是元数据，所以 EMP 包括在此监视元素的值中。

数据空间映射页包含用户数据，因此它们除了包括在 **spacemappage_page_reclaims_x** 监视元素的值中以外，还包括在 **page_reclaims_x** 监视元素的值中。索引空间映射页未包含用户数据，因此它们仅包括在 **spacemappage_page_reclaims_x** 监视元素的值中。

spacemappage_page_reclaims_s -“共享存取的空间映射页回收数”监视元素

与空间映射页相关的页被 DB2 pureScale实例中的另一成员回收的次数（在计划释放此页之前）。回收此页的成员需要对空间映射页具有共享存取权。

表 1606. 表函数监视信息

表函数	监视元素收集级别
MON_GET_PAGE_ACCESS_INFO 表函数 - 获取缓冲池页等待信息	ACTIVITY METRICS BASE

用法

仅对与对象相关的表空间（即，对可回收存储器启用的表空间）报告此值。使用 **reclaimable_space_enabled** 监视元素来确定是否对可回收存储器启用了此表空间。

因为扩展数据块映射页 (EMP) 是元数据，所以 EMP 包括在此监视元素的值中。

数据空间映射页包含用户数据，因此它们除了包括在 **spacemappage_page_reclaims_s** 监视元素的值中以外，还包括在 **page_reclaims_s** 监视元素的值中。索引空间页未包含用户数据，因此它们仅包括在 **spacemappage_page_reclaims_s** 监视元素的值中。

spacemappage_page_reclaims_initiated_x -“互斥存取导致的空间映射页回收数”监视元素

以空间映射页的互斥方式访问页从而导致从另一成员回收此页的次数。

表 1607. 表函数监视信息

表函数	监视元素收集级别
MON_GET_PAGE_ACCESS_INFO 表函数 - 获取缓冲池页等待信息	ACTIVITY METRICS BASE

用法

仅对与对象相关的表空间（即，对可回收存储器启用的表空间）报告此值。使用 **reclaimable_space_enabled** 监视元素来确定是否对可回收存储器启用了此表空间。

因为扩展数据块映射页 (EMP) 是元数据，所以 EMP 包括在此监视元素的值中。

数据空间映射页包含用户数据，因此它们除了包括在 **spacemappage_page_reclaims_initiated_x** 监视元素的值中以外，还包括在 **page_reclaims_initiated_x** 监视元素的值中。索引空间页未包含用户数据，因此它们仅包括在 **spacemappage_page_reclaims_initiated_x** 监视元素的值中。

spacemappage_page_reclaims_initiated_s -“共享存取导致的空间映射页回收数”监视元素

以空间映射页的共享方式访问页从而导致从另一成员回收此页的次数。

表 1608. 表函数监视信息

表函数	监视元素收集级别
MON_GET_PAGE_ACCESS_INFO 表函数 - 获取缓冲池页等待信息	ACTIVITY METRICS BASE

用法

仅对与对象相关的表空间（即，对可回收存储器启用的表空间）报告此值。使用 **reclaimable_space_enabled** 监视元素来确定是否对可回收存储器启用了此表空间。

因为扩展数据块映射页 (EMP) 是元数据，所以 EMP 包括在此监视元素的值中。

数据空间映射页包含用户数据，因此它们除了包括在 **spacemappage_page_reclaims_initiated_s** 监视元素的值中以外，还包括在 **page_reclaims_initiated_s** 监视元素的值中。索引空间页未包含用户数据，因此它们仅包括在 **spacemappage_page_reclaims_initiated_s** 监视元素的值中。

spacemappage_reclaim_wait_time -“空间映射页回收等待时间”监视元素

在 DB2 pureScale 环境中，此元素表示等待页锁定所耗的时间量（其中锁定请求导致从另一成员回收），这些页锁定针对与内部保留的对象空间管理相关的页。时间的度量单位为毫秒。

表 1609. 表函数监视信息

表函数	监视元素收集级别
MON_FORMAT_XML_METRICS_BY_ROW - 获取所有度量值的基于已格式化行的输出	不适用; 报告 XML 文档中作为格式化函数的输入提供的所有元素
MON_FORMAT_XML_TIMES_BY_ROW - - 获取基于已格式化行的组合层次结构等待时间和处理时间	不适用; 报告 XML 文档中作为格式化函数的输入提供的所有元素
MON_FORMAT_XML_WAIT_TIMES_BY _ROW - 获取等待时间的已格式化的基于行的输出	不适用; 报告 XML 文档中作为格式化函数的输入提供的所有元素
MON_GET_ACTIVITY_DETAILS 表函数 - 获取完整的活动详细信息（在 XML 文档 DETAILS 中报告）	ACTIVITY METRICS BASE
MON_GET_CONNECTION 表函数 - 获取连接度量值	REQUEST METRICS BASE
MON_GET_CONNECTION_DETAILS 表函数 - 获取详细的连接度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE
MON_GET_PAGE_ACCESS_INFO 表函数 - 获取缓冲池页等待信息	REQUEST METRICS BASE
MON_GET_PKG_CACHE_STMT 表函数 - 获取程序包高速缓存中的 SQL 语句活动度量值	ACTIVITY METRICS BASE
MON_GET_SERVICE_SUBCLASS 表函数 - 获取服务子类度量值	REQUEST METRICS BASE
MON_GET_SERVICE_SUBCLASS_DETAILS 表函数 - 获取详细的服务子类度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE
MON_GET_UNIT_OF_WORK 表函数 - 获取工作单元度量值	REQUEST METRICS BASE
MON_GET_UNIT_OF_WORK 表函数 - 获取详细的工作单元度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE
MON_GET_WORKLOAD 表函数 - 获取工作负载度量值	REQUEST METRICS BASE
MON_GET_WORKLOAD_DETAILS 表函数 - 获取详细的工作负载度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE

表 1610. 事件监视信息

事件类型	逻辑数据分组	监视开关
活动	event_activity（在 details_xml 文档中报告）	ACTIVITY METRICS BASE
活动	event_activitymetrics	ACTIVITY METRICS BASE
统计信息	event_scstats（在 details_xml 文档中报告）	REQUEST METRICS BASE
统计信息	event_wlstats（在 details_xml 文档中报告）	REQUEST METRICS BASE
工作单元	-	始终收集

ss_exec_time -“子节执行耗用时间”

执行子节所花的时间（以秒计）。

表 1611. 快照监视信息

快照级别	逻辑数据分组	监视开关
应用程序	subsection	语句

表 1612. 事件监视信息

事件类型	逻辑数据分组	监视开关
语句	event_subsection	-

用法 允许您跟踪子节进度。

ss_node_number -“子节节点号”

执行子节的节点。

表 1613. 快照监视信息

快照级别	逻辑数据分组	监视开关
应用程序	subsection	语句

表 1614. 事件监视信息

事件类型	逻辑数据分组	监视开关
语句	event_subsection	-

用法 用于使每个子节与执行该子节的数据库分区相关联。

ss_number -“子节号”监视元素

标识与返回的信息相关联的子节。

表 1615. 表函数监视信息

表函数	监视元素收集级别
MON_GET_APPL_LOCKWAIT 表函数 - 获取有关应用程序正在等待的锁定的信息	ACTIVITY METRICS BASE

表 1616. 快照监视信息

快照级别	逻辑数据分组	监视开关
应用程序	subsection	语句

表 1617. 事件监视信息

事件类型	逻辑数据分组	监视开关
语句	event_subsection	始终收集

用法

此编号与存取方案中可使用 **db2expln** 获取的子节号相关联。

ss_status -“子节状态”监视元素

正在执行的子节的当前状态。

表 1618. 快照监视信息

快照级别	逻辑数据分组	监视开关
应用程序	subsection	语句

用法

当前状态值可能是:

- 正在执行（sqlmon.h 中的 SQLM_SSEXEC）
- 等待锁定
- 在表队列上等待接收数据
- 在表队列上等待发送数据

ss_sys_cpu_time -“子节使用的系统 CPU 时间”

当前执行的语句子节使用的总系统 CPU 时间（以秒和微秒计）。对于写至表的事件监视器，此元素的值将通过使用 BIGINT 数据类型以微秒为单位给定。

元素标识

ss_sys_cpu_time

元素类型

时间

表 1619. 快照监视信息

快照级别	逻辑数据分组	监视开关
应用程序	subsection	时间戳记

表 1620. 事件监视信息

事件类型	逻辑数据分组	监视开关
语句	event_subsection	时间戳记

用法 此元素与其他相关 CPU 时间元素一起使用可帮助您了解应用程序内的活动级别，还可以帮助您标识可能因为调整而受益的应用程序。

系统 CPU 表示花费在系统调用上的时间。用户 CPU 表示花费在执行数据库管理器代码上的时间。

ss_usr_cpu_time -“子节使用的用户 CPU 时间”

当前执行的语句子节使用的总用户 CPU 时间（以秒和微秒计）。对于写至表的事件监视器，此元素的值将通过使用 BIGINT 数据类型以微秒为单位给定。

元素标识

ss_usr_cpu_time

元素类型

时间

表 1621. 快照监视信息

快照级别	逻辑数据分组	监视开关
应用程序	subsection	时间戳记

表 1622. 事件监视信息

事件类型	逻辑数据分组	监视开关
语句	event_subsection	时间戳记

用法 此元素与其他相关 CPU 时间元素一起使用可帮助您了解应用程序内的活动级别，还可以帮助您标识可能因为调整而受益的应用程序。

系统 CPU 表示花费在系统调用上的时间。用户 CPU 表示花费在执行数据库管理器代码上的时间。

ssl_port_number -“SSL 端口号”监视元素

成员正在针对客户机连接侦听的 SSL TCP/IP 端口。

表 1623. 表函数监视信息

表函数	监视元素收集级别
MON_GET_SERVERLIST 表函数 - 获取成员优先级详细信息	始终收集

start_event_id -“开始事件标识”

对应 UTILSTART 或 UTILSTARTPROC 事件的唯一标识。

表 1624. 事件监视信息

事件类型	逻辑数据分组	监视开关
变更历史记录	UTILSTOP	始终收集

用法

对于变更历史记录事件监视器，这是实用程序事件的对应开始（UTILSTART 或 UTILSTARTPROC）的唯一标识。将此元素与 START_EVENT_TIMESTAMP 和成员元素配合使用以使停止记录与对应开始记录相关联。

start_event_timestamp -“开始事件时间戳记”

对应 UTILSTART 或 UTILSTARTPROC 事件的时间。

表 1625. 事件监视信息

事件类型	逻辑数据分组	监视开关
变更历史记录	UTILSTOP	始终收集

用法

对于变更历史记录事件监视器，将其与 START_EVENT_ID 和成员元素配合使用以使停止记录与对应开始记录相关联。

start_time -“事件启动时间”

启动工作单元、启动语句或检测死锁的日期和时间。此元素在 event_start API 结构中指示事件监视器的启动时间。

表 1626. 事件监视信息

事件类型	逻辑数据分组	监视开关
数据库	event_start	时间戳记
语句	event_stmt	时间戳记
死锁	event_deadlock	时间戳记
死锁	event_dlconn	时间戳记
带有详细信息的死锁	event_detailed_dlconn	时间戳记

用法 可使用此元素来将死锁连接记录与死锁事件记录相关联，并与 *stop_time* 一起使用来计算执行语句或事务所耗用的时间。

注: 时间戳记开关设置为 OFF 时，此元素显示为“0”。

static_sql_stmts -“尝试的静态 SQL 语句数”

尝试的静态 SQL 语句数。

表 1627. 快照监视信息

快照级别	逻辑数据分组	监视开关
数据库	dbase	基本
应用程序	appl	基本

可将快照监视的计数器重置。

表 1628. 事件监视信息

事件类型	逻辑数据分组	监视开关
数据库	event_db	始终收集
连接	event_conn	始终收集

用法 可使用此元素来计算在数据库或应用程序级别成功执行的 SQL 语句总数:

```
    dynamic_sql_stmts
+ static_sql_stmts
- failed_sql_stmts
= 监视期间的吞吐量
```

statistics_timestamp -“统计信息时间戳记”监视元素

生成此统计信息记录的时间。

表 1629. 事件监视信息

事件类型	逻辑数据分组	监视开关
统计信息	event_scstats	-
统计信息	event_wlstats	-
统计信息	event_wcstats	-
统计信息	event_qstats	-
统计信息	event_histogrambin	-

用法

使用此元素来确定生成此统计信息记录的时间。

将此元素与 **last_wlm_reset** 元素配合使用以确定生成此统计信息记录中的统计信息的时间间隔。

此监视元素还可用来将对同一收集时间间隔生成的所有统计信息记录分组。

stats_cache_size -“统计信息高速缓存大小”监视元素

统计信息高速缓存的当前大小（以字节计），此高速缓存在目录分区中用于高速缓存实时统计信息收集生成的统计信息。

注: 由于统计信息高速缓存驻留在目录分区中，所以只有在目录分区捕获的快照才报告统计信息高速缓存大小。而在其他分区捕获的快照将报告零值。当捕获全局快照时，所有数据库分区报告的值将汇总合计。

表 1630. 表函数监视信息

表函数	监视元素收集级别
SNAP_GET_DB_V97 表函数 - 从 dbase 逻辑组检索快照信息	
SNAPDB 管理视图和 SNAP_GET_DB 表函数 - 从 dbase 逻辑组检索快照信息	

表 1631. 快照监视信息

快照级别	逻辑数据分组	监视开关
数据库	dbase	-

表 1632. 事件监视信息

事件类型	逻辑数据分组	监视开关
数据库	event_db	始终收集

用法

使用此元素来确定当前统计信息高速缓存的大小。此值经常更改。为了评估系统使用情况，请长期在特定时间间隔捕获快照。使用此元素来调整 **catalogcache_sz** 配置参数的值。

stats_fabricate_time -“生成统计信息的活动所花的总时间”监视元素

实时统计信息收集在生成统计信息所花的总时间（以毫秒计）。生成统计信息是在查询编译期间生成统计信息所需的统计信息收集活动。如果在数据库级别收集监视元素，那么此监视元素表示在数据库上运行的所有应用程序的实时统计信息收集活动所花的总时间。如果是在语句级别收集，那么它表示语句的最新实时统计信息收集所花的时间。所有数据库分区报告的时间将汇总合计。

表 1633. 表函数监视信息

表函数	监视元素收集级别
SNAP_GET_DB_V97 表函数 - 从 dbase 逻辑组检索快照信息	
SNAPDB 管理视图和 SNAP_GET_DB 表函数 - 从 dbase 逻辑组检索快照信息	

表 1634. 快照监视信息

快照级别	逻辑数据分组	监视开关
数据库	dbase	语句
动态 SQL	dynsql	语句

对于快照监视来说，此元素可以重置。

表 1635. 事件监视信息

事件类型	逻辑数据分组	监视开关
数据库	event_db	始终收集
语句	event_stmt	始终收集

用法

将此元素与 **stats_fabrications** 配合使用，以评估数据库级别的实时统计信息收集对性能的影响。对于动态 SQL 快照监视器，可以将此元素与 **total_exec_time** 和 **num_executions** 配合使用以评估生成统计信息的影响。对于语句事件监视器，可以将此元素与 **stmt_start** 和 **stmt_stop** 配合使用，以进一步评估实时统计信息收集的影响。

stats_fabrications -“生成统计信息的次数”监视元素

实时统计信息在查询编译期间为所有数据库应用程序执行的生成统计信息的总次数。与通过扫描表中或索引中存储的数据来获取统计信息不同，统计信息是根据索引和数据管理器维护的元数据来产生的。所有数据库分区报告的值将汇总合计。

表 1636. 表函数监视信息

表函数	监视元素收集级别
SNAP_GET_DB_V97 表函数 - 从 dbase 逻辑组检索快照信息	
SNAPDB 管理视图和 SNAP_GET_DB 表函数 - 从 dbase 逻辑组检索快照信息	

表 1637. 快照监视信息

快照级别	逻辑数据分组	监视开关
数据库	dbase	语句

可将快照监视的计数器重置。

表 1638. 事件监视信息

事件类型	逻辑数据分组	监视开关
数据库	event_db	始终收集

用法

使用此元素来确定数据库中生成统计信息的频率。此值经常更改。为更好地了解系统使用情况，请长期在特定时间间隔捕获快照。与 **stats_fabricate_time** 配合使用时，此元素可帮助您评估生成统计信息的影响。

status_change_time -“应用程序状态更改时间”

应用程序进入当前状态的日期和时间。

元素标识

status_change_time

元素类型

时间戳记

表 1639. 快照监视信息

快照级别	逻辑数据分组	监视开关
应用程序	appl_id_info	工作单元，时间戳记
锁定	appl_lock_list	工作单元，时间戳记
DCS 应用程序	dcs_appl_info	工作单元，时间戳记

用法 此元素允许您确定应用程序进入当前状态后所经历的时间。如果应用程序已处于同一状态很长一段时间，那么可能指示存在问题。

stmt_elapsed_time -“最新语句耗用时间”

最新完成的语句耗用的执行时间。

表 1640. 快照监视信息

快照级别	逻辑数据分组	监视开关
应用程序	stmt	语句，时间戳记
DCS 语句	dcs_stmt	语句，时间戳记

用法

将此元素用作完成语句所花时间的指示符。

此元素由两个子元素组成，它们报告耗用时间的秒数和微秒（一秒的百万分之一）数。这些子元素的名称可通过将“_s”和“_ms”添加至此监视元素的名称派生而成。要检索此监视元素耗用的总时间，必须将这两个子元素的值加在一起。例如，如果“_s”子元素值为 3，“_ms”子元素值为 20，那么此监视元素耗用的总时间为 3.00002 秒。

stmt_exec_time -“语句执行时间”监视元素

此成员上的所有代理程序执行此语句所花的总时间。此值以毫秒计。

表 1641. 表函数监视信息

表函数	监视元素收集级别
MON_FORMAT_XML_COMPONENT _TIMES_BY_ROW - 获取基于已格式化行的组件次数	不适用；报告 XML 文档中作为格式化函数的输入提供的所有元素
MON_FORMAT_XML_METRICS_BY_ROW - 获取所有度量值的基于已格式化行的输出	不适用；报告 XML 文档中作为格式化函数的输入提供的所有元素
MON_FORMAT_XML_TIMES_BY_ROW - - 获取基于已格式化行的组合层次结构等待时间和处理时间	不适用；报告 XML 文档中作为格式化函数的输入提供的所有元素
MON_GET_ACTIVITY_DETAILS 表函数 - 获取完整的活动详细信息（在 XML 文档 DETAILS 中报告）	ACTIVITY METRICS BASE

表 1641. 表函数监视信息 (续)

表函数	监视元素收集级别
MON_GET_PKG_CACHE_STMT 表函数 - 获取程序包高速缓存中的 SQL 语句活动度量值	ACTIVITY METRICS BASE
MON_GET_PKG_CACHE_STMT_DETAILS 表函数 - 获取程序包高速缓存条目的详细度量值	ACTIVITY METRICS BASE

表 1642. 事件监视信息

事件类型	逻辑数据分组	监视开关
活动	event_activity（在 details_xml 文档中报告）	ACTIVITY METRICS BASE
活动	event_activitymetrics	ACTIVITY METRICS BASE
程序包高速缓存	在 activity_metrics 文档中报告。	ACTIVITY METRICS BASE

stmt_first_use_time -“第一次使用语句时的时间戳记”监视元素

此元素显示第一次处理语句条目的时间。对于游标操作，**stmt_first_use_time** 将显示打开游标的时间。在应用程序协调节点，此值反映应用程序请求；在非协调程序节点，此值反映从原始节点接收请求的时间。

表 1643. 事件监视信息

事件类型	逻辑数据分组	监视开关
锁定	-	-
带有详细信息的死锁的历史记录值[1]	event_stmt_history	时间戳记
带有详细信息的死锁的历史记录 [1]	event_stmt_history	时间戳记
活动	event_activitystmt	时间戳记

1 建议不要使用此事件监视器。建议不要再使用此选项，将来的发行版可能会将其除去。请使用 CREATE EVENT MONITOR FOR LOCKING 语句来监视与锁定相关的事件，例如锁定超时、锁定等待和死锁。

用法

将此元素与其他语句历史记录条目一起使用来了解导致死锁的 SQL 语句的顺序。

stmt_history_id -“语句历史记录标识”

此数字元素显示相对于其他语句历史记录元素而言，该语句在 sequence_no 元素指示的工作单元中的运行位置。在工作单元中最早运行的语句的值最低。如果同一语句在同一工作单元中运行两次，那么该语句在两个不同位置出现时将显示两个不同的 stmt_history_id 值。

表 1644. 事件监视信息

事件类型	逻辑数据分组	监视开关
带有详细信息的死锁的历史记录值	event_stmt_history	-
带有详细信息的死锁的历史记录值	event_data_value	-
带有详细信息的死锁的历史纪录	event_stmt_history	-

用法 可使用此信息来了解导致死锁的 SQL 语句的顺序。

inact_stmthist_sz -“语句历史记录列表大小”

当带有历史记录的详细信息死锁事件监视器运行时，此元素将报告数据库监视器堆（MON_HEAP_SZ）中用于记录语句历史记录列表项的字节数。

表 1645. 快照监视信息

快照级别	逻辑数据分组	监视开关
应用程序	appl	-
数据库	db	-

用法 可在调整数据库监视器堆时使用此元素。

stmt_invocation_id -“语句调用标识”监视元素

此标识用于将例程的一次调用与工作单元中位于同一嵌套级别的其他调用区分开。它在特定嵌套级别的工作单元中是唯一的。

表 1646. 表函数监视信息

表函数	监视元素收集级别
MON_GET_ACTIVITY_DETAILS 表函数 - 获取完整的活动详细信息（在 XML 文档 DETAILS 中报告）	ACTIVITY METRICS BASE

表 1647. 事件监视信息

事件类型	逻辑数据分组	监视开关
活动	event_activitystmt	-
锁定	-	-
带有详细信息的死锁的历史记录值[1]	event_stmt_history	-
带有详细信息的死锁的历史记录 [1]	event_stmt_history	-
工作单元	在程序包列表中报告。	-

1 建议不要使用此选项。建议不要再使用此选项，将来的发行版可能会将其除去。请使用 CREATE EVENT MONITOR FOR LOCKING 语句来监视与锁定相关的事件，例如锁定超时、锁定等待和死锁。

用法

可以使用此元素来唯一地标识在其中执行了特定 SQL 语句的调用。还可将此元素与其他语句历史记录条目一起使用来了解导致死锁的 SQL 语句的顺序。

stmt_isolation -“语句隔离”

此元素显示运行语句时对该语句生效的隔离值。

表 1648. 事件监视信息

事件类型	逻辑数据分组	监视开关
带有详细信息的死锁的历史记录值	event_stmt_history	-
带有详细信息的死锁的历史纪录	event_stmt_history	-
活动	event_activitystmt	-

可能的隔离级别值包括:

- SQLM_ISOLATION_LEVEL_NONE 0（未指定隔离级别）
- SQLM_ISOLATION_LEVEL_UR 1（未落实的读）
- SQLM_ISOLATION_LEVEL_CS 2（游标稳定性）
- SQLM_ISOLATION_LEVEL_RS 3（读稳定性）
- SQLM_ISOLATION_LEVEL_RR 4（可重复读）

用法 可将此元素与其他语句历史记录条目一起使用来了解导致死锁的原因和特定 SQL 语句的执行行为。

stmt_last_use_time -“上一次使用语句时的时间戳记”监视元素

此元素显示上一次处理语句条目的时间。对于游标操作，**stmt_last_use_time** 显示游标上操作可能为打开、访存或关闭的上一次操作的时间。在应用程序协调节点，此值反映应用程序请求；在非协调程序节点，此值反映从原始节点接收请求的时间。

表 1649. 事件监视信息

事件类型	逻辑数据分组	监视开关
锁定	-	-
带有详细信息的死锁的历史记录值[1]	event_stmt_history	时间戳记
带有详细信息的死锁的历史记录 [1]	event_stmt_history	时间戳记
活动	event_activitystmt	时间戳记

1 建议不要使用此事件监视器。建议不要再使用此选项，将来的发行版可能会将其除去。请使用 CREATE EVENT MONITOR FOR LOCKING 语句来监视与锁定相关的事件，例如锁定超时、锁定等待和死锁。

用法

将此元素与其他语句历史记录条目一起使用来了解导致死锁的 SQL 语句的顺序。

stmt_lock_timeout -“语句锁定超时”监视元素

此元素显示运行语句时对该语句生效的锁定超时值。

表 1650. 事件监视信息

事件类型	逻辑数据分组	监视开关
锁定	-	-
带有详细信息的死锁的历史记录值[1]	event_stmt_history	-
带有详细信息的死锁的历史记录 [1]	event_stmt_history	-
活动	event_activitystmt	-

1 建议不要使用此事件监视器。建议不要再使用此选项，将来的发行版可能会将其除去。请使用 CREATE EVENT MONITOR FOR LOCKING 语句来监视与锁定相关的事件，例如锁定超时、锁定等待和死锁。

用法

可将此元素与其他语句历史记录条目一起使用来了解导致死锁的原因和特定 SQL 语句的执行行为。

stmt_nest_level -“语句嵌套级别”监视元素

此元素显示运行语句时生效的嵌套或递归的级别；每个嵌套级别对应一个存储过程或用户定义的函数（UDF）的嵌套或递归调用。

表 1651. 表函数监视信息

表函数	监视元素收集级别
MON_GET_ACTIVITY_DETAILS 表函数 - 获取完整的活动详细信息（在 XML 文档 DETAILS 中报告）	ACTIVITY METRICS BASE

表 1652. 事件监视信息

事件类型	逻辑数据分组	监视开关
锁定	-	-
带有详细信息的死锁的历史记录值[1]	event_stmt_history	-
带有详细信息的死锁的历史记录 [1]	event_stmt_history	-
活动	event_activitystmt	-
工作单元	在程序包列表中报告。	-

1 建议不要使用此事件监视器。建议不要再使用此选项，将来的发行版可能会将

其除去。请使用 CREATE EVENT MONITOR FOR LOCKING 语句来监视与锁定相关的事件，例如锁定超时、锁定等待和死锁。

用法

通过将此元素与 **stmt_invocation_id** 监视元素配合使用，可以唯一地标识在其中已经执行了特定 SQL 语句的调用。还可将此元素与其他语句历史记录条目一起使用来了解导致死锁的 SQL 语句的顺序。

stmt_node_number -“语句节点”

执行语句的节点。

表 1653. 快照监视信息

快照级别	逻辑数据分组	监视开关
应用程序	stmt	语句

用法 用于使每个语句与执行该语句的节点相关联。

stmt_operation/operation -“语句操作”监视元素

当前处理或最新处理的语句操作（如果当前未运行任何操作）。

表 1654. 表函数监视信息

表函数	监视元素收集级别
SNAPSHOT_STATEMENT 表函数	ACTIVITY METRICS BASE
SNAPSTMT 管理视图和 SNAP_GET_STMT 表函数 - 检索语句快照信息	ACTIVITY METRICS BASE

表 1655. 快照监视信息

快照级别	逻辑数据分组	监视开关
应用程序	stmt	语句
DCS 语句	dcs_stmt	语句

表 1656. 事件监视信息

事件类型	逻辑数据分组	监视开关
锁定	-	始终收集
带有详细信息的死锁 [1]	event_detailed_dlconn	始终收集
语句	event_stmt	始终收集

1 建议不要使用此事件监视器。建议不要再使用此选项，将来的发行版可能会将其除去。请使用 CREATE EVENT MONITOR FOR LOCKING 语句来监视与锁定相关的事件，例如锁定超时、锁定等待和死锁。

用法

可使用此元素来确定正在执行或最近完成的操作。

这可以是下列其中一个值:

对于 SQL 操作:

- SELECT
- PREPARE
- EXECUTE
- EXECUTE IMMEDIATE
- OPEN
- FETCH
- CLOSE
- DESCRIBE
- STATIC COMMIT
- STATIC ROLLBACK
- FREE LOCATOR
- PREP_COMMIT
- CALL
- PREP_OPEN
- PREP_EXEC
- COMPILE
- DROP PACKAGE

对于非 SQL 操作:

- RUN STATISTICS
- REORG
- REBIND
- REDISTRIBUTE
- GET TABLE AUTHORIZATION
- GET ADMINISTRATIVE AUTHORIZATION

注: API 用户应参考包含数据库系统监视器常量定义的 `sqlmon.h` 头文件。

stmt_pkgcache_id -“语句程序包高速缓存标识”监视元素

此元素显示动态 SQL 语句的内部程序包高速缓存标识。

表 1657. 表函数监视信息

表函数	监视元素收集级别
MON_GET_ACTIVITY_DETAILS 表函数 - 获取完整的活动详细信息（在 XML 文档 DETAILS 中报告）	ACTIVITY METRICS BASE

表 1657. 表函数监视信息 (续)

表函数	监视元素收集级别
MON_GET_PKG_CACHE_STMT 表函数 - 获取程序包高速缓存中的 SQL 语句活动度量值	ACTIVITY METRICS BASE
MON_GET_PKG_CACHE_STMT_DETAILS 表函数 - 获取程序包高速缓存条目的详细度量值	ACTIVITY METRICS BASE

表 1658. 快照监视信息

快照级别	逻辑数据分组	监视开关
动态 SQL	dynsql	基本

表 1659. 事件监视信息

事件类型	逻辑数据分组	监视开关
锁定	-	始终收集
带有详细信息的死锁的历史记录值[1]	event_stmt_history	始终收集
带有详细信息的死锁的历史记录 [1]	event_stmt_history	始终收集
活动	event_activitystmt	始终收集
程序包高速缓存	-	COLLECT BASE DATA

1 建议不要使用此事件监视器。建议不要再使用此选项，将来的发行版可能会将其除去。请使用 CREATE EVENT MONITOR FOR LOCKING 语句来监视与锁定相关的事件，例如锁定超时、锁定等待和死锁。

用法

在多分区环境中，对于高速缓存的语句，每个分区都有一个唯一的语句标识。给定语句在各分区中的标识可能不同。

在全局动态 SQL 快照中，只返回第一个语句标识。

stmt_query_id -“语句查询标识”监视元素

此元素显示对用作游标的任何 SQL 语句指定的内部查询标识。

表 1660. 事件监视信息

事件类型	逻辑数据分组	监视开关
锁定	-	-
带有详细信息的死锁的历史记录值[1]	event_stmt_history	-
带有详细信息的死锁的历史记录 [1]	event_stmt_history	-
活动	event_activitystmt	-

用法

通过将此元素与 **stmt_nest_level** 监视元素配合使用，可以唯一地标识特定 SQL 语句的调用。还可将此元素与其他语句历史记录条目一起使用来了解导致死锁的原因。

stmt_sorts -“语句排序数”

为处理 stmt_operation 而对一组数据排序的总次数。

表 1661. 快照监视信息

快照级别	逻辑数据分组	监视开关
应用程序	appl	语句
应用程序	stmt	语句
动态 SQL	dynsql	语句

用法 可使用此元素来帮助标识对索引的需求，原因是索引可以降低数据排序的需求。通过使用上述表中的相关元素，您可以标识此元素为其提供排序信息的 SQL 语句，然后通过查看要排序的列（如 ORDER BY 和 GROUP BY 子句和连接列中使用的列）分析此语句以确定索引候选项。有关检查索引是否用于优化排序性能的信息，请参阅*管理指南*中的**说明**。

此计数包括数据库管理器为执行语句而在内部生成的临时表排序。排序数与 SQL 语句的第一个 FETCH 操作相关联。当语句的操作是第一个 FETCH 时，将返回此信息。您应注意到对于分块游标而言，打开游标时将执行若干次访存。在这种情况下，很难使用快照监视器来获取排序数，原因是 DB2 在内部发出第一个 FETCH 时需要获取快照。

如果要确定使用分块游标时执行的排序数，那么更可靠的方法是使用对语句声明的事件监视器。CLOSE 游标的语句事件中的 total_sorts 计数器包含执行定义了游标的语句时执行的排序总数。

stmt_source_id -“语句源标识”

此元素显示对运行的 SQL 语句的源指定的内部标识。

表 1662. 事件监视信息

事件类型	逻辑数据分组	监视开关
锁定	-	-
带有详细信息的死锁的历史记录值[1]	event_stmt_history	-
带有详细信息的死锁的历史记录 [1]	event_stmt_history	-
活动	event_activitystmt	-

1 建议不要使用此事件监视器。建议不要再使用此选项，将来的发行版可能会将其除去。请使用 CREATE EVENT MONITOR FOR LOCKING 语句来监视与锁定相关的事件，例如锁定超时、锁定等待和死锁。

用法

通过将此元素与 **appl_id** 配合使用，可以唯一地标识运行特定 SQL 语句的请求的来源。还可将此元素与其他语句历史记录条目一起使用来了解导致死锁的原因。

stmt_start -“语句操作开始时间戳记”

stmt_operation 开始执行的日期和时间。

元素标识

stmt_start

元素类型

时间戳记

表 1663. 快照监视信息

快照级别	逻辑数据分组	监视开关
应用程序	stmt	语句，时间戳记
DCS 语句	dcs_stmt	语句，时间戳记

用法 可将此元素与 stmt_stop 一起使用来计算执行语句操作时耗用的时间。

stmt_stop -“语句操作停止时间戳记”

stmt_operation 停止执行的日期和时间。

元素标识

stmt_stop

元素类型

时间戳记

表 1664. 快照监视信息

快照级别	逻辑数据分组	监视开关
应用程序	stmt	语句，时间戳记
DCS 语句	dcs_stmt	语句，时间戳记

用法 可将此元素与 stmt_start 一起使用来计算执行语句操作时耗用的时间。

stmt_sys_cpu_time -“语句使用的系统 CPU 时间”

当前执行的语句使用的总系统 CPU 时间（以秒和微秒计）。

元素标识

stmt_sys_cpu_time

元素类型

时间

表 1665. 快照监视信息

快照级别	逻辑数据分组	监视开关
应用程序	appl	语句，时间戳记

表 1665. 快照监视信息 (续)

快照级别	逻辑数据分组	监视开关
应用程序	stmt	语句，时间戳记

用法 此元素与其他相关 CPU 时间元素一起使用可帮助您了解应用程序内的活动级别，还可以帮助您标识可能因为调整而受益的应用程序。

此计数器包括花费在 SQL 和非 SQL 语句上的时间，同时包括花费在应用程序执行的所有不受防护的用户定义的函数（UDF）或存储过程上的时间。

系统 CPU 表示花费在系统调用上的时间。用户 CPU 表示花费在执行数据库管理器代码上的时间。

注: 如果此信息对您的操作系统不可用，那么此元素将设置为 0。

stmt_text -“SQL 语句文本”监视元素

SQL 语句的文本。

表 1666. 表函数监视信息

表函数	监视元素收集级别
MON_GET_ACTIVITY_DETAILS 表函数 - 获取完整的活动详细信息（在 XML 文档 DETAILS 中报告）	ACTIVITY METRICS BASE
MON_GET_PKG_CACHE_STMT 表函数 - 获取程序包高速缓存中的 SQL 语句活动度量值	ACTIVITY METRICS BASE
MON_GET_PKG_CACHE_STMT_DETAILS 表函数 - 获取程序包高速缓存条目的详细度量值	ACTIVITY METRICS BASE

表 1667. 快照监视信息

快照级别	逻辑数据分组	监视开关
应用程序	stmt	语句
动态 SQL	dynsql	基本
DCS 语句	dcs_stmt	语句

表 1668. 事件监视信息

事件类型	逻辑数据分组	监视开关
锁定	-	始终收集
带有详细信息的死锁 [1]	event_detailed_dlconn	始终收集
带有详细信息的死锁的历史记录 [1]	event_stmt_history	始终收集
语句	event_stmt	始终收集
活动	event_activitystmt	始终收集
程序包高速缓存	-	COLLECT BASE DATA
变更历史记录	ddlstmtexec	始终收集

1 建议不要使用此事件监视器。建议不要再使用此选项，将来的发行版可能会将

其除去。请使用 CREATE EVENT MONITOR FOR LOCKING 语句来监视与锁定相关的事件，例如锁定超时、锁定等待和死锁。

用法

对于应用程序快照，此语句文本帮助您标识获取快照时执行的应用程序或最新处理的应用程序（如果获取快照时正好没有在处理的语句）。

此元素返回的信息是从 SQL 语句高速缓存中获取的，并且在高速缓存溢出时可能不可用。如果要捕获语句的 SQL 文本，那么唯一保险的方法是对语句使用事件监视器。

对于动态 SQL 语句，此元素标识与程序包相关联的 SQL 文本。

对于语句事件监视器而言，将仅对动态语句返回此元素。如果语句事件监视器记录在语句事件监视器的 BUFFERSIZE 选项所指定大小的缓冲区中放不下，那么 **stmt_text** 监视器的值可能会被截断以使该记录能够放得下。

对于 EVENT_STMT_HISTORY 事件监视器而言，将仅对动态语句返回此元素。对于其余事件监视器而言，仅当 **stmt_text** 包含在 SQL 语句高速缓存中时，才会对动态和静态语句返回此元素。

有关如何查询系统目录表以获取因为性能注意事项而未提供的静态 SQL 语句文本的信息，请参阅 **section_number** 监视元素。

stmt_type -“语句类型”监视元素

所处理语句的类型。

表 1669. 快照监视信息

快照级别	逻辑数据分组	监视开关
应用程序	stmt	语句

表 1670. 事件监视信息

事件类型	逻辑数据分组	监视开关
锁定	-	始终收集
带有详细信息的死锁 [1]	event_detailed_dlconn	始终收集
语句	event_stmt	始终收集
活动	event_activitystmt	始终收集

1 建议不要使用此事件监视器。建议不要再使用此选项，将来的发行版可能会将其除去。请使用 CREATE EVENT MONITOR FOR LOCKING 语句来监视与锁定相关的事件，例如锁定超时、锁定等待和死锁。

用法

可使用此元素来确定正在执行的语句的类型。它可以是下列语句中的一个:

- 静态 SQL 语句
- 动态 SQL 语句
- SQL 语句之外的操作；如绑定或预编译操作。

对于快照监视器，此元素描述当前正在处理或最新处理的语句。

注: API 用户应参考包含数据库系统监视器常量定义的 sqlmon.h 头文件。

stmt_type_id -“语句类型标识”监视元素

语句类型标识。

表 1671. 表函数监视信息

表函数	监视元素收集级别
MON_GET_PKG_CACHE_STMT 表函数 - 获取程序包高速缓存中的 SQL 语句活动度量值	ACTIVITY METRICS BASE
MON_GET_PKG_CACHE_STMT_DETAILS 表函数 - 获取程序包高速缓存条目的详细度量值	ACTIVITY METRICS BASE

表 1672. 事件监视信息

事件类型	逻辑数据分组	监视开关
程序包高速缓存	-	COLLECT BASE DATA

用法

stmt_type_id 监视元素具有以下可能的值:

- Statement not prepared
- DDL, (not Set Constraints)
- DDL, Set Constraints
- DML, Select
- DML, Insert/Update/Delete
- Authorization
- DML, Select (blockable)
- DML, Lock Table
- DML, Commit/Rollback
- Set environment
- DDL, Savepoint
- DDL, (declared user temp)
- Passthru support
- CALL
- Free locator
- DML, Select with IUD
- DML, Select with IUD (blockable)
- Top-level SET, no SQL
- Top-level SET, reads SQL
- DDL, (issues internal commit)
- Top-level SET, modifies SQL

- Unknown

stmt_unicode -“语句 Unicode 标志”监视元素

SQL 语句 Unicode 标志。可能的值: Yes 或 No。

表 1673. 事件监视信息

事件类型	逻辑数据分组	监视开关
锁定	lock_participant_activities	

stmt_usr_cpu_time -“语句使用的用户 CPU 时间”

当前执行的语句使用的总*用户* CPU 时间（以秒和微秒计）。

元素标识

stmt_usr_cpu_time

元素类型

时间

表 1674. 快照监视信息

快照级别	逻辑数据分组	监视开关
应用程序	appl	语句，时间戳记
应用程序	stmt	语句，时间戳记

用法 此元素与其他相关 CPU 时间元素一起使用可帮助您了解应用程序内的活动级别，还可以帮助您标识可能因为调整而受益的应用程序。

此计数器包括花费在 SQL 和非 SQL 语句上的时间，同时包括花费在应用程序执行的所有不受防护的用户定义的函数（UDF）或存储过程上的时间。

系统 CPU 表示花费在系统调用上的时间。用户 CPU 表示花费在执行数据库管理器代码上的时间。

注: 如果此信息对您的操作系统不可用，那么此元素将设置为 0。

stmt_value_data -“值数据”

此元素包含 SQL 语句的数据值的字符串表示。LOB、LONG 和结构化类型参数显示为空字符串。日期、时间和时间戳记字段记录为 ISO 格式。

表 1675. 表函数监视信息

表函数	监视元素收集级别
MON_GET_PKG_CACHE_STMT_DETAILS - 获取程序包高速缓存条目的详细度量值	ACTIVITY METRICS BASE

表 1676. 事件监视信息

事件类型	逻辑数据分组	监视开关
锁定	-	-

表 1676. 事件监视信息 (续)

事件类型	逻辑数据分组	监视开关
带有详细信息的死锁的历史记录值[1]	stmt_value_data	-
活动	event_activityvals	-

1 建议不要使用此事件监视器。建议不要再使用此选项，将来的发行版可能会将其除去。请使用 CREATE EVENT MONITOR FOR LOCKING 语句来监视与锁定相关的事件，例如锁定超时、锁定等待和死锁。

用法

可将此元素与其他语句历史记录条目一起使用来了解导致死锁的原因。

stmt_value_index -“值索引”

此元素表示 SQL 语句中使用的输入参数标记或主变量的位置。

表 1677. 表函数监视信息

表函数	监视元素收集级别
MON_GET_PKG_CACHE_STMT_DETAILS - 获取程序包高速缓存条目的详细度量值	ACTIVITY METRICS BASE

表 1678. 事件监视信息

事件类型	逻辑数据分组	监视开关
锁定	-	-
带有详细信息的死锁的历史记录值[1]	stmt_value_data	-
活动	event_activityvals	-

1 建议不要使用此事件监视器。建议不要再使用此选项，将来的发行版可能会将其除去。请使用 CREATE EVENT MONITOR FOR LOCKING 语句来监视与锁定相关的事件，例如锁定超时、锁定等待和死锁。

用法

可将此元素与其他语句历史记录条目一起使用来了解导致死锁的原因。

stmt_value_isnull -“包含空值”监视元素

此元素指出与 SQL 语句相关联的数据值是否为 NULL 值；是否使用了扩展指示符来指定缺省值；或者是否未指定此语句值。

可能的值包括:

- 如果值不是 NULL，那么此监视元素的值为 0 或“no”
- 如果值为 NULL，那么此监视元素的值为 1 或“yes”
- 如果为此语句值指定了扩展指示符值“default”(-5)，那么此监视元素的值为 2 或“default”

- 如果为此语句值指定了扩展指示符值“unassigned”(-7)，那么此监视元素的值为 3 或“unassigned”

表 1679. 表函数监视信息

表函数	监视元素收集级别
MON_GET_PKG_CACHE_STMT_DETAILS - 获取程序包高速缓存条目的详细度量值	ACTIVITY METRICS BASE

表 1680. 事件监视信息

事件类型	逻辑数据分组	监视开关
锁定	-	-
带有详细信息的死锁的历史记录值[1]	stmt_value_isnull	-
活动	event_activityvals	-

1 建议不要使用此事件监视器。建议不要再使用此选项，将来的发行版可能会将其除去。请使用 CREATE EVENT MONITOR FOR LOCKING 语句来监视与锁定相关的事件，例如锁定超时、锁定等待和死锁。

用法

可将此元素与其他语句历史记录条目一起使用来了解导致死锁的原因。

stmt_value_isreopt -“用于语句重新优化的变量”监视元素

此元素显示是否在语句重新优化期间使用了提供的值。如果语句重新优化（例如，因为设置 REOPT 绑定选项），或者该值在此重新优化期间用作 SQL 编译器的输入，那么返回值“True”。

表 1681. 表函数监视信息

表函数	监视元素收集级别
MON_GET_PKG_CACHE_STMT_DETAILS - 获取程序包高速缓存条目的详细度量值	ACTIVITY METRICS BASE

表 1682. 事件监视信息

事件类型	逻辑数据分组	监视开关
锁定	-	-
带有详细信息的死锁的历史记录值[1]	event_data_value	-
活动	event_activityvals	-

1 建议不要使用此事件监视器。建议不要再使用此选项，将来的发行版可能会将其除去。请使用 CREATE EVENT MONITOR FOR LOCKING 语句来监视与锁定相关的事件，例如锁定超时、锁定等待和死锁。

用法

可以将此元素与提供的编译环境一起使用，以允许就 SQL 编译器对 SQL 语句的处理进行完整的分析。

stmt_value_type -“值类型”监视元素

此元素包含与 SQL 语句相关联的数据值类型的字符串表示。

表 1683. 表函数监视信息

表函数	监视元素收集级别
MON_GET_PKG_CACHE_STMT_DETAILS - 获取程序包高速缓存条目的详细度量值	ACTIVITY METRICS BASE

表 1684. 事件监视信息

事件类型	逻辑数据分组	监视开关
锁定	-	-
带有详细信息的死锁的历史记录值[1]	stmt_value_type	-
活动	event_activityvals	-

1 建议不要使用此事件监视器。建议不要再使用此选项，将来的发行版可能会将其除去。请使用 CREATE EVENT MONITOR FOR LOCKING 语句来监视与锁定相关的事件，例如锁定超时、锁定等待和死锁。

用法

可将此元素与其他语句历史记录条目一起使用来了解导致死锁的原因。

sto_path_free_sz -“自动存储器路径可用空间量”监视元素

此元素显示存储器路径指向的文件系统上的可用空间量（以字节为单位）。如果多个存储器路径指向同一个文件系统，那么不会在它们之间划分可用大小。

表 1685. 表函数监视信息

表函数	监视元素收集级别
ADMIN_GET_STORAGE_PATHS 表函数 - 获取存储器组的存储器路径信息	ACTIVITY METRICS BASE

表 1686. 快照监视信息

快照级别	逻辑数据分组	监视开关
数据库	db_sto_path_info	缓冲池

用法

可以将此元素与下列元素配合使用以收集每个节点的数据库空间利用率数据:

- **db_storage_path**
- **fs_used_size**

- **fs_total_size**
- **fs_id**

stop_time -“事件停止时间”

语句停止执行的日期和时间。

表 1687. 事件监视信息

事件类型	逻辑数据分组	监视开关
语句	event_stmt	时间戳记

用法 可将此元素与 *start_time* 配合使用来计算执行语句所耗用的时间。

对于 FETCH 语句事件，此项是上一次成功的访存操作的时间。

注: 时间戳记开关设置为 OFF 时，此元素显示为“0”。

storage_group_id -“存储器组标识”

一个整数，它唯一表示当前数据库使用的存储器组。

表 1688. 表函数监视信息

表函数	监视元素收集级别
ADMIN_GET_STORAGE_PATHS 表函数 - 获取存储器组的存储器路径信息	ACTIVITY METRICS BASE
MON_GET_TABLESPACE 表函数 - 获取表空间度量值	ACTIVITY METRICS BASE

使用说明

- 如果使用 ADMIN_GET_STORAGE_PATHS 表函数，那么存储器组标识会指示存储器路径定义至的存储器组。
- 如果使用 MON_GET_TABLESPACES 表函数，那么存储器组标识指示在其中定义该表空间的存储器组。

storage_group_name -“存储器组名称”

存储器组的名称。

表 1689. 表函数监视信息

表函数	监视元素收集级别
ADMIN_GET_STORAGE_PATHS 表函数 - 获取存储器组的存储器路径信息	ACTIVITY METRICS BASE
MON_GET_TABLESPACE 表函数 - 获取表空间度量值	ACTIVITY METRICS BASE

使用说明

- 如果使用 ADMIN_GET_STORAGE_PATHS 表函数，那么此监视元素会指示存储器路径定义至的存储器组。

- 如果使用 MON_GET_TABLESPACES 表函数，那么此监视元素指示在其中定义该表空间的存储器组。

stored_proc_time -“存储过程时间”

此元素包含此数据源响应存储过程语句所花的总时间（以毫秒计），这些存储过程语句来自联合服务器启动后或数据库监视计数器上一次重置后在此联合服务器实例上运行的所有应用程序或单个应用程序。

表 1690. 快照监视信息

快照级别	逻辑数据分组	监视开关
数据库	dbase_remote	时间戳记
应用程序	appl_remote	时间戳记

可将快照监视的计数器重置。

响应时间是以联合服务器将存储过程提交给数据源的时间与数据源响应以指示已处理存储过程的时间之差量度的。

用法 使用此元素来确定在此数据源上处理存储过程所花的实际时间。

stored_procs -“存储过程”

此元素包含自联合服务器实例启动或数据库监视计数器最后一次重置后，联合服务器代表任何应用程序在此数据源上调用的存储过程总数。

表 1691. 快照监视信息

快照级别	逻辑数据分组	监视开关
数据库	dbase_remote	基本
应用程序	appl_remote	基本

可将快照监视的计数器重置。

用法 使用此元素来确定联合数据库或应用程序在本地对联合数据库调用的存储过程数。

swap_pages_in -“从磁盘换入的页数”监视元素

系统启动后从磁盘换入的页数。仅对 AIX 和 Linux 系统报告。

表 1692. 表函数监视信息

表函数	监视元素收集级别
ENV_GET_SYSTEM_RESOURCES 表函数 - 返回系统信息	ACTIVITY METRICS BASE

swap_pages_out -“换出至磁盘的页数”监视元素

系统启动后换出至磁盘的页数。仅对 AIX 和 Linux 系统报告。

表 1693. 表函数监视信息

表函数	监视元素收集级别
ENV_GET_SYSTEM_RESOURCES 表函数 - 返回系统信息	ACTIVITY METRICS BASE

swap_page_size -“交换页大小”监视元素

用于交换空间的页大小，以字节计。仅对 AIX 和 Linux 系统报告。

表 1694. 表函数监视信息

表函数	监视元素收集级别
ENV_GET_SYSTEM_RESOURCES 表函数 - 返回系统信息	ACTIVITY METRICS BASE

sync_runstats -“同步 RUNSTATS 活动总数”监视元素

实时统计信息收集对数据库中的所有应用程序触发的同步 RUNSTATS 总数。此值同时包括成功和不成功的同步 RUNSTATS 命令。所有数据库分区报告的值将汇总合计。

表 1695. 表函数监视信息

表函数	监视元素收集级别
SNAP_GET_DB_V97 表函数 - 从 dbase 逻辑组检索快照信息	
SNAPDB 管理视图和 SNAP_GET_DB 表函数 - 从 dbase 逻辑组检索快照信息	

表 1696. 快照监视信息

快照级别	逻辑数据分组	监视开关
数据库	dbase	语句

可将快照监视的计数器重置。

表 1697. 事件监视信息

事件类型	逻辑数据分组	监视开关
数据库	event_db	始终收集

用法

使用此监视元素来确定实时统计信息收集已在数据库中触发的同步 RUNSTATS 活动数。此值经常更改。为更好地了解系统使用情况，请长期在特定时间间隔捕获快照。与 **sync_runstats_time** 配合使用时，此元素可帮助您评估实时统计信息收集触发的同步 RUNSTATS 活动对性能的影响。

sync_runstats_time -“同步 RUNSTATS 活动所花的总时间”监视元素

实时统计信息收集触发的同步 RUNSTATS 活动所花的总时间（以毫秒计）。在查询编译期间发生同步 RUNSTATS 活动。在数据库级别，此监视元素表示由实时统计信息收集触发的在数据库上运行的所有应用程序的同步 RUNSTATS 活动所花的总时间。在语句级别，它表示由实时统计信息收集触发的特定语句的最新同步 RUNSTATS 活动所花的时间。所有数据库分区报告的值将汇总合计。

表 1698. 表函数监视信息

表函数	监视元素收集级别
SNAP_GET_DB_V97 表函数 - 从 dbase 逻辑组检索快照信息	
SNAPDB 管理视图和 SNAP_GET_DB 表函数 - 从 dbase 逻辑组检索快照信息	

表 1699. 快照监视信息

快照级别	逻辑数据分组	监视开关
数据库	dbase	语句
动态 SQL	dynsql	语句

对于快照监视来说，此元素可以重置。

表 1700. 事件监视信息

事件类型	逻辑数据分组	监视开关
数据库	event_db	始终收集
语句	event_stmt	始终收集

用法

将此元素与 **sync_runstats** 配合使用，以评估在数据库级别由实时统计信息收集触发的同步 RUNSTATS 活动对性能的影响。

对于动态 SQL 快照监视器，将此元素与 **total_exec_time** 和 **num_executions** 配合使用以评估同步 RUNSTATS 对查询性能的影响。

对于语句事件监视器，将此元素与 **stmt_start** 和 **stmt_stop** 配合使用以进一步评估实时统计信息收集的影响。

system_auth_id -“系统授权标识”监视元素

连接的系统授权标识。

表 1701. 表函数监视信息

表函数	监视元素收集级别
MON_GET_CONNECTION 表函数 - 获取连接度量值	ACTIVITY METRICS BASE

表 1701. 表函数监视信息 (续)

表函数	监视元素收集级别
MON_GET_CONNECTION_DETAILS 表函数 - 获取详细的连接度量值（在 XML 文档 DETAILS 中报告）	ACTIVITY METRICS BASE
WLM_GET_SERVICE_CLASS_WORKLOAD_OCCURRENCES 表函数 - 列示工作负载出现次数	ACTIVITY METRICS BASE

表 1702. 事件监视信息

事件类型	逻辑数据分组	监视元素收集级别
变更历史记录	changesummary	始终收集

system_cpu_time -“系统 CPU 时间”监视元素

数据库管理器代理进程、工作单元或语句使用的总系统 CPU 时间（以秒和微秒计）。对于写至表的事件监视器，此元素的值将通过使用 BIGINT 数据类型以微秒为单位给定。

当监视开关或时间戳记开关未打开时，将不收集此元素。在这种情况下，监视元素改为显示 -1。

表 1703. 事件监视信息

事件类型	逻辑数据分组	监视开关
连接	event_conn	始终收集
事务	event_xact	始终收集
语句	event_stmt	始终收集
活动	event_activity	始终收集

用法

此元素与其他相关 CPU 时间元素一起使用可帮助您了解应用程序内的活动级别，还可以帮助您标识可能因为其他调整而受益的应用程序。

注: 如果此信息对您的操作系统不可用，那么此元素将设置为 0。

注: 由于 DB2 系统收集统计信息时所使用的详细程度不同，**total_exec_time** 监视元素的值可能与 **system_cpu_time** 和 **user_cpu_time** 监视元素的值的总和不相等。在此情况下，**system_cpu_time** 与 **user_cpu_time** 监视元素的值之和更准确地反映了实际总执行时间。

tab_file_id -“表文件标识”监视元素

表的文件标识（FID）。

表 1704. 表函数监视信息

表函数	监视元素收集命令和级别
MON_GET_TABLE_METRICS 表函数 - 获取表度量值	ACTIVITY METRICS BASE

用法

tab_type -“表类型”监视元素

此接口返回基于 sqlmon.h 中的定义的文本标识，并且为 USER_TABLE、TEMP_TABLE 或 CATALOG_TABLE 的其中之一。

表 1705. 表函数监视信息

表函数	监视元素收集级别
MON_GET_TABLE_METRICS 表函数 - 获取表度量值	ACTIVITY METRICS BASE

用法

table_file_id -“表文件标识”监视元素

表的文件标识（FID）。

表 1706. 表函数监视信息

表函数	监视元素收集级别
ADMINTEMPTABLES 管理视图和 ADMIN_GET_TEMP_TABLES 表函数 - 检索临时表的信息	ACTIVITY METRICS BASE
MON_GET_APPL_LOCKWAIT 表函数 - 获取有关应用程序正在等待的锁定的信息	ACTIVITY METRICS BASE
MON_GET_LOCKS 表函数 - 列示当前所连接的数据库中的所有锁定	ACTIVITY METRICS BASE
MON_GET_TABLE 表函数 - 获取表度量值	ACTIVITY METRICS BASE

表 1707. 快照监视信息

快照级别	逻辑数据分组	监视开关
应用程序	appl	锁定
表	表	基本
锁定	appl_lock_list	锁定
锁定	lock	锁定

表 1708. 事件监视信息

事件类型	逻辑数据分组	监视开关
死锁	lock	始终收集

用法

对于快照监视来说，此元素仅供参考。返回它是为了获取与数据库系统监视器的先前版本的兼容性，并且它可能未唯一地标识该表。请使用 **table_name** 和 **table_schema** 监视元素来标识表。

在 MON_GET_LOCKS 和 MON_GET_APPL_LOCKWAIT 表函数中，此元素表示锁定将引用的表的文件标识（FID）。

table_name -“表名”监视元素

表的名称。

表 1709. 表函数监视信息

表函数	监视元素收集级别
ADMIN_GET_INDEX_COMPRESS_INFO 表函数 - 返回压缩索引信息	ACTIVITY METRICS BASE
ADMIN_GET_INDEX_INFO 表函数 - 返回索引信息	ACTIVITY METRICS BASE
ADMIN_GET_TAB_COMPRESS_INFO 表函数 - 估算压缩节省量	ACTIVITY METRICS BASE
ADMIN_GET_TAB_DICTIONARY_INFO 表函数 - 报告现有表字典的属性	ACTIVITY METRICS BASE
ADMINTABINFO 管理视图和 ADMIN_GET_TAB_INFO 表函数 - 检索表大小和状态信息	ACTIVITY METRICS BASE
ADMINTEMPCOLUMNS 管理视图和 ADMIN_GET_TEMP_COLUMNS 表函数 - 检索临时表的列信息	ACTIVITY METRICS BASE
ADMINTEMPTABLES 管理视图和 ADMIN_GET_TEMP_TABLES 表函数 - 检索临时表的信息	ACTIVITY METRICS BASE
MON_FORMAT_LOCK_NAME 表函数 - 设置内部锁定名称的格式并返回详细信息	ACTIVITY METRICS BASE
MON_GET_INDEX 表函数 - 获取索引度量值	ACTIVITY METRICS BASE
MON_GET_PAGE_ACCESS_INFO 表函数 - 获取缓冲池页等待信息	ACTIVITY METRICS BASE
MON_GET_TABLE 表函数 - 获取表度量值	ACTIVITY METRICS BASE
MON_GET_TABLE_USAGE_LIST 表函数 - 从表用法列表返回信息	ACTIVITY METRICS BASE

表 1710. 快照监视信息

快照级别	逻辑数据分组	监视开关
表	表	基本
应用程序	appl	锁定
锁定	appl_lock_list	锁定
锁定	锁定	锁定
锁定	lock_wait	锁定

表 1711. 事件监视信息

事件类型	逻辑数据分组	监视开关
锁定	-	始终收集
表	event_table	始终收集
死锁 [1]	锁定	始终收集
死锁 [1]	event_dlconn	始终收集
带有详细信息的死锁 [1]	event_detailed_dlconn	始终收集

1 建议不要使用此事件监视器。建议不要再使用此选项，将来的发行版可能会将其除去。请使用 CREATE EVENT MONITOR FOR LOCKING 语句来监视与锁定相关的事件，例如锁定超时、锁定等待和死锁。

用法

通过将此元素与 **table_schema** 配合使用，可以确定资源争用的根源。

在应用程序级别、应用程序锁定级别和死锁监视级别，此项是应用程序等待锁定的表，原因是它目前被另一应用程序锁定。对于快照监视而言，仅当“锁定”监视器组信息设置为 ON 并且 **lock_object_type** 表明应用程序正在等待获取表锁定时，此项才有效。

对于对象锁定级别的快照监视，将对表级别和行级别锁定返回此项。在此级别报告的表就是此应用程序对其持有这些锁定的表。

对于表级别的快照和事件监视，此项是对其收集信息的表。对于临时表，**table_name** 的格式为 『TEMP (*n*, *m*)』，其中:

- *n* 是表空间标识
- *m* 是 **table_file_id** 元素

table_scans -“表扫描次数”监视元素

对此表执行扫描的次数。

表 1712. 表函数监视信息

表函数	监视元素收集级别
MON_GET_TABLE 表函数 - 获取表度量值	ACTIVITY METRICS BASE

用法

table_schema -“表模式名”监视元素

表的模式。

表 1713. 表函数监视信息

表函数	监视元素收集级别
ADMIN_GET_INDEX_COMPRESS_INFO 表函数 - 返回压缩索引信息	ACTIVITY METRICS BASE
ADMIN_GET_INDEX_INFO 表函数 - 返回索引信息	ACTIVITY METRICS BASE
ADMIN_GET_TAB_COMPRESS_INFO 表函数 - 估算压缩节省量	ACTIVITY METRICS BASE
ADMIN_GET_TAB_DICTIONARY_INFO 表函数 - 报告现有表字典的属性	ACTIVITY METRICS BASE
ADMINTABINFO 管理视图和 ADMIN_GET_TAB_INFO 表函数 - 检索表大小和状态信息	ACTIVITY METRICS BASE
ADMINTEMPCOLUMNS 管理视图和 ADMIN_GET_TEMP_COLUMNS 表函数 - 检索临时表的列信息	ACTIVITY METRICS BASE
ADMINTEMPTABLES 管理视图和 ADMIN_GET_TEMP_TABLES 表函数 - 检索临时表的信息	ACTIVITY METRICS BASE
MON_FORMAT_LOCK_NAME 表函数 - 设置内部锁定名称的格式并返回详细信息	ACTIVITY METRICS BASE
MON_GET_INDEX 表函数 - 获取索引度量值	ACTIVITY METRICS BASE
MON_GET_PAGE_ACCESS_INFO 表函数 - 获取缓冲池页等待信息	ACTIVITY METRICS BASE
MON_GET_TABLE 表函数 - 获取表度量值	ACTIVITY METRICS BASE
MON_GET_TABLE_USAGE_LIST 表函数 - 从表用法列表返回信息	ACTIVITY METRICS BASE

表 1714. 快照监视信息

快照级别	逻辑数据分组	监视开关
表	表	基本
应用程序	appl	锁定
锁定	appl_lock_list	锁定
锁定	锁定	锁定
锁定	lock_wait	锁定

表 1715. 事件监视信息

事件类型	逻辑数据分组	监视开关
锁定	-	始终收集
表	event_table	始终收集
死锁 [1]	锁定	始终收集

表 1715. 事件监视信息 (续)

事件类型	逻辑数据分组	监视开关
死锁 [1]	event_dlconn	始终收集
带有详细信息的死锁 [1]	event_detailed_dlconn	始终收集

1 建议不要使用此事件监视器。建议不要再使用此选项，将来的发行版可能会将其除去。请使用 CREATE EVENT MONITOR FOR LOCKING 语句来监视与锁定相关的事件，例如锁定超时、锁定等待和死锁。

用法

通过将此元素与 **table_name** 配合使用，可以确定资源争用的根源。

对于应用程序级别、应用程序锁定级别和死锁监视级别，此项是应用程序等待锁定的表的模式，原因是它目前被另一应用程序锁定。仅当 **lock_object_type** 指示应用程序正在等待获取表锁定时，才设置此元素。对于应用程序级别和应用程序锁定级别的快照监视而言，仅当“锁定”监视器组信息设置为 ON 时，此项才有效。

对于对象锁定级别的快照监视，将对表级别和行级别锁定返回此项。在此级别报告的表就是此应用程序对其持有这些锁定的表。

对于表级别的快照和事件监视，此元素标识对其收集信息的表的模式。对于临时表，**table_schema** 的格式为 『*<agent_id><auth_id>*』，其中:

- *agent_id* 是创建临时表的应用程序的应用程序句柄
- *auth_id* 是应用程序用来连接至数据库的授权标识

table_type -“表类型”监视元素

与返回的信息相对应的表的类型。

表 1716. 表函数监视信息

表函数	监视元素收集级别
MON_GET_TABLE 表函数 - 获取表度量值	ACTIVITY METRICS BASE

表 1717. 快照监视信息

快照级别	逻辑数据分组	监视开关
表	表	基本

表 1718. 事件监视信息

事件类型	逻辑数据分组	监视开关
表	event_table	始终收集

用法

使用此元素来帮助确定与返回的信息相对应的表。如果该表是用户表或系统目录表，那么可使用 **table_name** 和 **table_schema** 来标识表。

表的类型可以是下列值中的一个。可能的值是基于 sqlmon.h 文件中的定义的文本字符串。

USER_TABLE
用户表。

TEMP_TABLE
临时表。将返回有关临时表的信息，即使这些表在使用后未保留在数据库中。您会发现有关此类型的表的信息仍然有用。

CATALOG_TABLE
系统目录表。

tablespace_auto_resize_enabled -“允许自动调整表空间大小”监视元素

此元素描述是否允许自动调整表空间的大小。值为 1 表示“是”，值为 0 表示“否”。

表 1719. 表函数监视信息

表函数	监视元素收集级别
MON_GET_TABLESPACE 表函数 - 获取表空间度量值	ACTIVITY METRICS BASE

表 1720. 快照监视信息

快照级别	逻辑数据分组	监视开关
表空间	tablespace	基本

用法

此元素仅适用于 DMS 表空间和非临时自动存储器表空间。如果此元素设置为 1，那么能够自动调整大小。有关表空间的增长率和最大大小的信息，请参阅下列监视元素。

- **tablespace_max_size**
- **tablespace_increase_size**
- **tablespace_increase_size_percent**

tablespace_content_type -“表空间内容类型”监视元素

表空间中的内容类型。

表 1721. 表函数监视信息

表函数	监视元素收集级别
MON_GET_TABLESPACE 表函数 - 获取表空间度量值	ACTIVITY METRICS BASE

表 1722. 快照监视信息

快照级别	逻辑数据分组	监视开关
表空间	tablespace	基本

用法

表空间中的内容类型（在 sqlmon.h 中定义）可以是下列值中的一个:

- 所有类型的永久数据。
  - 常规表空间: SQLM_TABLESPACE_CONTENT_ANY
  - 大型表空间: SQLM_TABLESPACE_CONTENT_LARGE
- 系统临时数据: SQLM_TABLESPACE_CONTENT_SYSTEMP
- 用户临时数据: SQLM_TABLESPACE_CONTENT_USRTEMP

tablespace_cur_pool_id -“当前使用的缓冲池”监视元素

表空间当前使用的缓冲池的标识。

表 1723. 表函数监视信息

表函数	监视元素收集命令和级别
MON_GET_TABLESPACE 表函数 - 获取表空间度量值	ACTIVITY METRICS BASE

表 1724. 快照监视信息

快照级别	逻辑数据分组	监视开关
表空间	tablespace	基本

用法 每个缓冲池由唯一的整数标识。此元素的值与视图 SYSCAT.BUFFERPOOLS 的列 BUFFERPOOLID 中的值相匹配。

tablespace_current_size -“当前表空间大小”

此元素显示表空间的当前大小（以字节计）。

表 1725. 快照监视信息

快照级别	逻辑数据分组	监视开关
表空间	tablespace_nodeinfo	基本

用法 对于 DMS 和自动存储器表空间，此元素表示所有表空间容器的总大小（以字节计）。此值等于表空间的总页数（tablespace_total_pages）乘以表空间的页大小（tablespace_page_size）的积。此元素不适用于 SMS 表空间或临时自动存储器表空间。

在为自动存储器表空间创建表空间时，当前大小可能与初始大小不匹配。当前大小的值应在页大小乘以扩展数据块大小乘以创建时初始大小的存储器路径数的积的范围内 （通常大于该积，但有时小于该积）。它将总是小于或等于 tablespace_max_size（如果设置了）。这是因为容器只能以完整的扩展数据块大小增长，并且必须以集合的方式增长。

tablespace_extent_size -“表空间扩展数据块大小”监视元素

表空间使用的扩展数据块大小。

表 1726. 表函数监视信息

表函数	监视元素收集命令和级别
MON_GET_TABLESPACE 表函数 - 获取表空间度量值	ACTIVITY METRICS BASE

表 1727. 快照监视信息

快照级别	逻辑数据分组	监视开关
表空间	tablespace	基本

tablespace_free_pages -“表空间中的空闲页数”监视元素

表空间中当前空闲的总页数。

表 1728. 表函数监视信息

表函数	监视元素收集级别
MON_GET_TABLESPACE 表函数 - 获取表空间度量值	ACTIVITY METRICS BASE

表 1729. 快照监视信息

快照级别	逻辑数据分组	监视开关
表空间	tablespace_nodeinfo	基本

用法

此元素仅适用于 DMS 表空间。

tablespace_id -“表空间标识”监视元素

唯一表示当前数据库使用的表空间的整数。

表 1730. 表函数监视信息

表函数	监视元素收集级别
ADMINTEMPTABLES 管理视图和 ADMIN_GET_TEMP_TABLES 表函数 - 检索临时表的信息	ACTIVITY METRICS BASE
MON_GET_APPL_LOCKWAIT 表函数 - 获取有关应用程序正在等待的锁定的信息	ACTIVITY METRICS BASE
MON_GET_CONTAINER 表函数 - 获取表空间容器度量值	ACTIVITY METRICS BASE
MON_GET_EXTENT_MOVEMENT_STATUS - 获取扩展数据块移动进度状态度量值	ACTIVITY METRICS BASE
MON_GET_LOCKS 表函数 - 列示当前所连接的数据库中的所有锁定	ACTIVITY METRICS BASE

表 1730. 表函数监视信息 (续)

表函数	监视元素收集级别
MON_GET_REBALANCE_STATUS 表函数 - 获取表空间的重新平衡进度	ACTIVITY METRICS BASE
MON_GET_TABLE 表函数 - 获取表度量值	ACTIVITY METRICS BASE
MON_GET_TABLESPACE 表函数 - 获取表空间度量值	ACTIVITY METRICS BASE

表 1731. 快照监视信息

快照级别	逻辑数据分组	监视开关
表空间	tablespace	基本
表	表	基本

表 1732. 事件监视信息

事件类型	逻辑数据分组	监视开关
表	event_table	始终收集

用法

此元素的值与视图 SYSCAT.TABLESPACES 的列 TBSPACEID 中的值相匹配。

tablespace_increase_size -“增加字节大小”

此元素显示表空间已满并且需要更多空间时自动调整大小的表空间将增加的大小（以字节计）。

表 1733. 快照监视信息

快照级别	逻辑数据分组	监视开关
表空间	tablespace_nodeinfo	基本

用法 此项表示将添加表空间的空间量，当该表空间变满并且请求更多空间，同时又未达到最大表空间大小时，该表空间可自动调整大小。如果此元素的值为 -1（或者在快照输出中为 『AUTOMATIC』），那么 DB2 将自动确定需要添加空间时的值。此元素仅适用于能够自动调整大小的表空间。

tablespace_increase_size_percent -“增加大小（以百分比计）”监视元素

此元素显示表空间已满并且需要更多空间时自动调整大小的表空间将增加的量。实际字节数是根据表空间当时的大小来调整大小时确定的。

表 1734. 快照监视信息

快照级别	逻辑数据分组	监视开关
表空间	tablespace_nodeinfo	基本

用法 此项表示将添加表空间的空间量，当该表空间变满并且请求更多空间，同时又

未达到最大表空间大小时，该表空间可自动调整大小。增长率以调整表空间大小时当前表空间大小（tablespace_current_size）所占的百分比为基础。此元素仅适用于能够自动调整大小的表空间。

tablespace_initial_size -“初始表空间大小”

自动存储器表空间的初始大小（以字节计）。

表 1735. 快照监视信息

快照级别	逻辑数据分组	监视开关
表空间	tablespace_nodeinfo	基本

用法 对于非临时自动存储器表空间，此监视元素表示创建表空间时的初始大小（以字节计）。

tablespace_last_resize_failed -“上一次调整大小尝试失败”

此元素描述上一次尝试自动增加表空间大小是否失败。值为 1 意味着肯定；0 意味着否定。

表 1736. 快照监视信息

快照级别	逻辑数据分组	监视开关
表空间	tablespace_nodeinfo	基本

用法 对于自动存储器表空间，此元素可能显示任何数据库存储器路径上都没有留下空间。对于非自动存储器表空间，失败意味着因为文件系统已满而未能扩展其中一个容器。失败的另一个原因是已达到表空间的最大大小。此元素仅适用于能够自动调整大小的表空间。

tablespace_last_resize_time -“上次成功调整大小的时间”

此元素显示用来表示上次成功增加表空间的大小的时间戳记。

表 1737. 快照监视信息

快照级别	逻辑数据分组	监视开关
表空间	tablespace_nodeinfo	基本

用法 对于可自动调整大小的表空间，此元素表示上次表空间变满并且请求更多空间，同时又未达到最大表空间大小时，自动对该表空间添加空间的时间。此元素仅适用于能够自动调整大小的表空间。

tablespace_max_size -“最大表空间大小”

此元素显示表空间可自动调整大小或增长至的最大大小（以字节计）。

表 1738. 表函数监视信息

表函数	监视元素收集级别
MON_GET_TABLESPACE 表函数 - 获取表空间度量值	ACTIVITY METRICS BASE

表 1739. 快照监视信息

快照级别	逻辑数据分组	监视开关
表空间	tablespace_nodeinfo	基本

用法 此项表示可自动调整大小的表空间可自动增长至的最大大小（以字节计）。如果此值等于 tablespace_current_size 元素，那么没有空间来容纳表空间的增长。如果此元素的值为 -1，那么最大大小被认为是『无限』并且表空间可自动调整大小直到文件系统已满或达到表空间的体系结构大小限制。（此限制将在*SQL Reference*中的 SQL 限制附录中描述）。此元素仅适用于能够自动调整大小的表空间。

tablespace_min_recovery_time -“前滚的最短恢复时间”监视元素

显示可前滚表空间的最早时间点的时间戳记。 此时间戳记反映本地时间。

表 1740. 表函数监视信息

表函数	监视元素收集级别
MON_GET_TABLESPACE 表函数 - 获取表空间度量值	ACTIVITY METRICS BASE

表 1741. 快照监视信息

快照级别	逻辑数据分组	监视开关
表空间	tablespace_nodeinfo	基本

用法

只有不为零时才显示。

tablespace_name -“表空间名称”监视元素

表空间的名称。

表 1742. 表函数监视信息

表函数	监视元素收集级别
MON_FORMAT_LOCK_NAME 表函数 - 设置内部锁定名称的格式并返回详细信息	ACTIVITY METRICS BASE
MON_GET_CONTAINER 表函数 - 获取表空间容器度量值	ACTIVITY METRICS BASE
MON_GET_EXTENT_MOVEMENT_STATUS - 获取扩展数据块移动进度状态度量值	ACTIVITY METRICS BASE
MON_GET_REBALANCE_STATUS - 获取表空间的重新平衡进度	ACTIVITY METRICS BASE
MON_GET_TABLESPACE 表函数 - 获取表空间度量值	ACTIVITY METRICS BASE

表 1743. 快照监视信息

快照级别	逻辑数据分组	监视开关
表空间	tablespace	基本
锁定	appl_lock_list	基本
锁定	lock	锁定
锁定	lock_wait	锁定

表 1744. 事件监视信息

事件类型	逻辑数据分组	监视开关
锁定	-	-
死锁 [1]	lock	-
死锁 [1]	event_dlconn	-
带有详细信息的死锁 [1]	event_detailed_dlconn	-
表空间	tablespace_list	-

1 建议不要使用此事件监视器。建议不要再使用此选项，将来的发行版可能会将其除去。请使用 CREATE EVENT MONITOR FOR LOCKING 语句来监视与锁定相关的事件，例如锁定超时、锁定等待和死锁。

用法

此元素可帮助您确定资源争用的源头。

它相当于数据库目录表 SYSCAT.TABLESPACES 中的 TBSPACE 列。在应用程序级别、应用程序锁定级别和死锁监视级别，此项是应用程序等待锁定的表空间的名称。另一个应用程序当前持有针对此表空间的锁定。

在锁定级别，此项是应用程序当前对其持有锁定的表空间的名称。

在表空间级别（缓冲池监视器组设置为“ON”时），此项是与所返回信息相对应的表空间的名称。

对于针对分区表持有的表锁定，将不会返回此元素。

tablespace_next_pool_id -“下次启动时使用的缓冲池”监视元素

表空间在下一次数据库启动时使用的缓冲池的标识。

表 1745. 表函数监视信息

表函数	监视元素收集命令和级别
MON_GET_TABLESPACE 表函数 - 获取表空间度量值	ACTIVITY METRICS BASE

表 1746. 快照监视信息

快照级别	逻辑数据分组	监视开关
表空间	tablespace	基本

用法　每个缓冲池由唯一的整数标识。此元素的值与视图 SYSCAT.BUFFERPOOLS 的列 BUFFERPOOLID 中的值相匹配。

tablespace_num_containers -“表空间中的容器数目”

表空间中的容器总数。

表 1747. 快照监视信息

快照级别	逻辑数据分组	监视开关
表空间	tablespace_nodeinfo	基本

tablespace_num_quiescers -“停顿者数目”

停顿表空间的用户数目（可以在范围 0 到 5 之间）。

表 1748. 快照监视信息

快照级别	逻辑数据分组	监视开关
表空间	tablespace_nodeinfo	基本

用法　此值表示停顿表空间的代理程序数（以“SHARE”、“UPDATE”或“EXCLUSIVE”方式）。对于每个停顿者，将在 tablespace_quiescer 逻辑数据组中返回以下信息:

- 停顿者的用户授权标识
- 停顿者的代理程序标识
- 因为停顿而导致此表空间停顿的对象的表空间标识
- 因为停顿而导致此表空间停顿的对象的对象标识
- 停顿状态

tablespace_num_ranges -“表空间映射中的范围数”

表空间映射中的范围（条目）数。此项的范围在 1 到 100 之间（通常小于 12）。仅 DMS 表空间存在表空间映射。

表 1749. 快照监视信息

快照级别	逻辑数据分组	监视开关
表空间	tablespace_nodeinfo	基本

tablespace_page_size -“表空间页大小”监视元素

表空间使用的页大小（以字节计）。

表 1750. 表函数监视信息

表函数	监视元素收集命令和级别
MON_GET_TABLESPACE 表函数 - 获取表空间度量值	ACTIVITY METRICS BASE

表 1751. 快照监视信息

快照级别	逻辑数据分组	监视开关
表空间	tablespace	基本

tablespace_page_top -“表空间高水位标记”监视元素

表空间中包含高水位标记的页。

表 1752. 表函数监视信息

表函数	监视元素收集级别
MON_GET_TABLESPACE 表函数 - 获取表空间度量值	ACTIVITY METRICS BASE

表 1753. 快照监视信息

快照级别	逻辑数据分组	监视开关
表空间	tablespace_nodeinfo	基本

用法

对于 DMS，此元素表示上一次分配的表空间扩展数据块之后第一个可用扩展数据块的页号。注意，这并非实际意义上的“高水位标记”，而是“当前水位标记”，这是因为该值可能下降。对于 SMS，此项不适用。

tablespace_paths_dropped -“表空间正在使用已删除的路径”监视元素

指示表空间正在使用已删除的存储器路径。

表 1754. 表函数监视信息

表函数	监视元素收集级别
MON_GET_TABLESPACE 表函数 - 获取表空间度量值	ACTIVITY METRICS BASE

表 1755. 快照监视信息

快照级别	逻辑数据分组	监视开关
表空间	tablespace_nodeinfo	基本

用法

对于正在使用自动存储器的表空间，请使用此监视元素来确定是否有任何表空间容器驻留在已被删除的存储器路径中。以物理方式从数据库中删除存储器路径之前，所有表空间都必须停止使用这些路径。要停止使用已删除的存储器路径，请删除该表空间，或者使用 ALTER TABLESPACE 语句的 REBALANCE 子句对该表空间进行重新平衡。

tablespace_pending_free_pages -“表空间中的暂挂可用页数”监视元素

如果已落实或回滚所有暂挂事务，并且已经为对象请求了新的空间，那么在表空间中将变得可用的页的数目。

表 1756. 表函数监视信息

表函数	监视元素收集级别
MON_GET_TABLESPACE 表函数 - 获取表空间度量值	ACTIVITY METRICS BASE

表 1757. 快照监视信息

快照级别	逻辑数据分组	监视开关
表空间	tablespace_nodeinfo	基本

用法

此元素仅适用于 DMS 表空间。

tablespace_prefetch_size -“表空间预取大小”监视元素

预取程序一次从磁盘获取的最大页数。

表 1758. 表函数监视信息

表函数	监视元素收集级别
MON_GET_TABLESPACE 表函数 - 获取表空间度量值	ACTIVITY METRICS BASE

表 1759. 快照监视信息

快照级别	逻辑数据分组	监视开关
表空间	tablespace	基本
表空间	tablespace_nodeinfo	基本

用法

- 对于表函数监视，此元素始终报告表空间预取大小的实际值。
- 对于快照监视，如果启用了自动预取大小，那么此元素在*表空间*逻辑数据分组中将显示值“-1”，而在 *tablespace_nodeinfo* 逻辑数据分组中显示实际值。
- 对于快照监视，如果未启用自动预取大小，那么此元素将在*表空间*逻辑数据分组中显示实际值，并且该元素不会出现在 *tablespace_nodeinfo* 逻辑数据分组中。

tablespace_rebalancer_extents_processed -“重新平衡程序已经处理的扩展数据块数”

自重新平衡程序启动或重新启动后（选择最近的时间）重新平衡程序已经移动的扩展数据块数。

表 1760. 表函数监视信息

表函数	监视元素收集级别
MON_GET_REBALANCE_STATUS 表函数 - 获取表空间的重新平衡进度	ACTIVITY METRICS BASE

表 1761. 快照监视信息

快照级别	逻辑数据分组	监视开关
表空间	tablespace_nodeinfo	基本

用法 此项可用作重新平衡程序的完成级别的指示符。可通过记录此元素在一段时间内的更改来监视重新平衡进度。可使用 tablespace_state 和 rebalance_mode 来检查重新平衡是否完成。此元素仅适用于 DMS 表空间。

tablespace_rebalancer_extents_remaining -“重新平衡程序要处理的扩展数据块总数”

要移动的扩展数据块数目。此值是在重新平衡程序启动时间或重新启动时间计算的（选择最近的时间）。

表 1762. 表函数监视信息

表函数	监视元素收集级别
MON_GET_REBALANCE_STATUS 表函数 - 获取表空间的重新平衡进度	ACTIVITY METRICS BASE

表 1763. 快照监视信息

快照级别	逻辑数据分组	监视开关
表空间	tablespace_nodeinfo	基本

用法 此元素可用作重新平衡程序的完成级别的指示符。可通过记录此元素在一段时间内的更改来监视重新平衡进度。可使用 tablespace_state 来检查重新平衡是否完成。此元素仅适用于 DMS 表空间。

tablespace_rebalancer_last_extent_moved -“重新平衡程序移动的最后一个扩展数据块”

重新平衡程序移动的最后一个扩展数据块。

表 1764. 表函数监视信息

表函数	监视元素收集级别
MON_GET_REBALANCE_STATUS 表函数 - 获取表空间的重新平衡进度	ACTIVITY METRICS BASE

表 1765. 快照监视信息

快照级别	逻辑数据分组	监视开关
表空间	tablespace_nodeinfo	基本

用法 此项可用作重新平衡程序的完成级别的指示符。可通过记录此元素在一段时间内的更改来监视重新平衡进度。可使用 tablespace_state 和 rebalance_mode 来检查重新平衡是否完成。此元素仅适用于 DMS 表空间。

tablespace_rebalancer_mode -"重新平衡程序方式"监视元素

指示当前重新平衡过程是正在从表空间中除去空间还是正在对表空间添加空间。

表 1766. 表函数监视信息

表函数	监视元素收集级别
MON_GET_REBALANCE_STATUS 表函数 - 获取表空间的重新平衡进度	ACTIVITY METRICS BASE
MON_GET_TABLESPACE 表函数 - 获取表空间度量值	ACTIVITY METRICS BASE

表 1767. 快照监视信息

快照级别	逻辑数据分组	监视开关
表空间	tablespace_nodeinfo	基本

用法

在添加新容器或者增大现有容器的大小时，将执行*正向重新平衡*。在正向重新平衡操作中，数据移动以表空间中的第一个扩展数据块开始并以高水位标记扩展数据块结束。

在除去容器或者减小其大小并且需要从正在释放的空间中移出数据时，将执行*反向重新平衡*。在反向重新平衡操作中，数据移动以高水位标记扩展数据块开始按反向顺序处理整个表空间，并以表空间中的第一个扩展数据块结束。

*双程重新平衡*是指先执行正向重新平衡，然后再执行反向重新平衡。如果在重新平衡操作期间既添加容器也删除容器，那么将执行双程重新平衡。

对于 DMS 非自动存储器表空间而言，此监视元素指示正在对该表空间执行的重新平衡的类型。对于 DMS 非自动表空间而言，只能执行单程正向重新平衡或单程反向重新平衡。

对于自动存储器表空间而言，此监视元素指示当前重新平衡过程正在对该表空间执行的操作。通常，启动重新平衡操作时，只需要执行单程正向重新平衡或单程反向重新平衡。但是，在某些情况下，有必要对自动存储器表空间执行双程重新平衡。

可能的 **tablespace_rebalancer_mode** 值由 sqlmon.h 文件定义。监视快照时会返回以下值:

SQLM_TABLESPACE_NO_REBAL
未执行重新平衡。

SQLM_TABLESPACE_FWD_REBAL
正在执行正向重新平衡。

SQLM_TABLESPACE_REV_REBAL
正在执行反向重新平衡。

SQLM_TABLESPACE_FWD_REBAL_OF_2PASS
正在执行双程重新平衡操作的正向重新平衡阶段。

SQLM_TABLESPACE_REV_REBAL_OF_2PASS
正在执行双程重新平衡操作的反向重新平衡阶段。

如果使用 MON_GET_TABLESPACE 或 MON_GET_REBALANCE_STATUS 表函数，那么会返回以下 rebalancer_mode 值:

- NO_REBAL
- FWD_REBAL
- REV_REBAL
- FWD_REBAL_OF_2PASS
- REV_REBAL_OF_2PASS

tablespace_rebalancer_priority -“当前重新平衡程序优先级”

在数据库中运行的重新平衡程序的优先级。

表 1768. 表函数监视信息

表函数	监视元素收集级别
MON_GET_REBALANCE_STATUS 表函数 - 获取表空间的重新平衡进度	ACTIVITY METRICS BASE

表 1769. 快照监视信息

快照级别	逻辑数据分组	监视开关
表空间	tablespace_nodeinfo	基本

用法 此元素仅适用于 DMS 表空间。

tablespace_rebalancer_restart_time -“重新平衡程序重新启动时间”

表示重新平衡程序在暂停或暂挂后何时重新启动的时间戳记。

表 1770. 表函数监视信息

表函数	监视元素收集级别
MON_GET_REBALANCE_STATUS 表函数 - 获取表空间的重新平衡进度	ACTIVITY METRICS BASE

表 1771. 快照监视信息

快照级别	逻辑数据分组	监视开关
表空间	tablespace_nodeinfo	基本

用法 此项可用作重新平衡程序的完成级别的指示符。它将说明重新平衡程序重新启动的时间，并且允许派生重新平衡程序的速度和直到完成所耗用的时间。此元素仅适用于 DMS 表空间。

tablespace_rebalancer_source_storage_group_id -“重新平衡程序源存储器组标识”

重新平衡程序将一个存储器组中的表空间移至另一个存储器组时的源存储器组标识。否则，它为 -1。

表 1772. 表函数监视信息

表函数	监视元素收集级别
MON_GET_REBALANCE_STATUS 表函数 - 获取表空间的重新平衡进度	ACTIVITY METRICS BASE

tablespace_rebalancer_source_storage_group_name -“重新平衡程序源存储器组名称”

重新平衡程序将一个存储器组中的表空间移至另一个存储器组时的源存储器组名称。否则，它为 NULL。

表 1773. 表函数监视信息

表函数	监视元素收集级别
MON_GET_REBALANCE_STATUS 表函数 - 获取表空间的重新平衡进度	ACTIVITY METRICS BASE

tablespace_rebalancer_start_time -“重新平衡程序启动时间”

表示重新平衡程序最初启动时间的时间戳记。

表 1774. 表函数监视信息

表函数	监视元素收集级别
MON_GET_REBALANCE_STATUS 表函数 - 获取表空间的重新平衡进度	ACTIVITY METRICS BASE

表 1775. 快照监视信息

快照级别	逻辑数据分组	监视开关
表空间	tablespace_nodeinfo	基本

用法 此项用于指示重新平衡程序最初启动的时间。它可以用来派生度量值，以测量运行重新平衡程序的速度和完成重新平衡的估计时间。此元素仅适用于 DMS 表空间。

tablespace_rebalancer_status -“重新平衡程序状态”监视元素

指示重新平衡操作的当前状态。

表 1776. 表函数监视信息

表函数	监视元素收集级别
MON_GET_REBALANCE_STATUS 表函数 - 获取表空间的重新平衡进度	ACTIVITY METRICS BASE

用法说明

重新平衡操作的当前状态为下列其中一项:

- ACTIVE - 重新平衡操作处于活动状态。
- SUSPENDED - 重新平衡操作已被使用 ALTER TABLESPACE 语句的用户显式暂挂。
- PAUSED - 重新平衡操作因为联机备份已隐式暂停。重新平衡将在此备份完成时继续。

如果重新平衡操作已显式暂挂并隐式暂停，那么状态将报告为 SUSPENDED。

tablespace_rebalancer_target_storage_group_id -“重新平衡程序目标存储器组标识”

重新平衡程序将一个存储器组中的表空间移至另一个存储器组时的目标存储器组标识。否则，它为 -1。

表 1777. 表函数监视信息

表函数	监视元素收集级别
MON_GET_REBALANCE_STATUS 表函数 - 获取表空间的重新平衡进度	ACTIVITY METRICS BASE

此监视元素连接至 target_storage_group_name 监视元素、source_storage_group_id 监视元素和 source_storage_group_name 监视元素。可使用这些元素来了解重新平衡操作是否正将表空间从一个存储器组移至另一个存储器组，并了解该表空间正移出的存储器组（源）和移至的存储器组（目标）。

tablespace_rebalancer_target_storage_group_name -“重新平衡程序目标存储器组名”

重新平衡程序将一个存储器组中的表空间移至另一个存储器组时的目标存储器组名称。否则，它为 NULL。

表 1778. 表函数监视信息

表函数	监视元素收集级别
MON_GET_REBALANCE_STATUS 表函数 - 获取表空间的重新平衡进度	ACTIVITY METRICS BASE

tablespace_state -“表空间状态”监视元素

此元素描述表空间的当前状态。

表 1779. 表函数监视信息

表函数	监视元素收集级别
MON_GET_TABLESPACE 表函数 - 获取表空间度量值	ACTIVITY METRICS BASE

表 1780. 快照监视信息

快照级别	逻辑数据分组	监视开关
表空间	tablespace_nodeinfo	基本

用法

在管理视图和表函数中，此监视元素将根据 `sqlutil.h` 中的定义来返回文本标识，并且是用“+”号分隔的下列值的组合:

- BACKUP_IN_PROGRESS
- BACKUP_PENDING
- DELETE_PENDING
- DISABLE_PENDING
- DROP_PENDING
- LOAD_IN_PROGRESS
- LOAD_PENDING
- MOVE_IN_PROGRESS
- NORMAL
- OFFLINE
- PSTAT_CREATION
- PSTAT_DELETION
- QUIESCED_EXCLUSIVE
- QUIESCED_SHARE
- QUIESCED_UPDATE
- REBAL_IN_PROGRESS
- REDIST_IN_PROGRESS
- REORG_IN_PROGRESS
- RESTORE_IN_PROGRESS
- RESTORE_PENDING
- ROLLFORWARD_IN_PROGRESS
- ROLLFORWARD_PENDING
- STORDEF_ALLOWED
- STORDEF_CHANGED
- STORDEF_FINAL_VERSION

- STORDEF_PENDING
- SUSPEND_WRITE

此元素包含指示当前表空间状态的十六进制值。表空间的外部可视状态由特定状态值的十六进制和组成。例如，如果状态为“停顿：EXCLUSIVE”和“装入暂挂”，那么值为 0x0004 + 0x0008，即 0x000c。使用 **db2tbst** 命令来获取与给定十六进制值相关联的表空间状态。

表 1781. sqlutil.h 中列示的位定义

十六进制值	十进制值	State
0x0	0	正常（请参阅 sqlutil.h 中的定义 SQLB_NORMAL）
0x1	1	停顿：SHARE
0x2	2	停顿：UPDATE
0x4	4	停顿：EXCLUSIVE
0x8	8	装入暂挂
0x10	16	删除暂挂
0x20	32	备份暂挂
0x40	64	正在前滚
0x80	128	前滚暂挂
0x100	256	复原暂挂
0x100	256	恢复暂挂（未使用）
0x200	512	禁用暂挂
0x400	1024	正在重组
0x800	2048	正在备份
0x1000	4096	必须定义存储器
0x2000	8192	正在复原
0x4000	16384	脱机并且不可访问
0x8000	32768	删除暂挂
0x10000	65536	不允许写入
0x20000	131072	正在装入
0x40000	262144	正在重新分发
0x80000	524288	正在移动
0x2000000	33554432	可以定义存储器
0x4000000	67108864	存储器定义处于“最终”状态
0x8000000	134217728	在前滚之前已更改存储器定义
0x10000000	268435456	DMS 重新平衡程序处于活动状态
0x20000000	536870912	正在进行 TBS 删除
0x40000000	1073741824	正在进行 TBS 创建

注： DB2 LOAD 不会将表空间状态设置为装入暂挂或删除暂挂。

tablespace_state_change_object_id -“状态更改对象标识”

导致表空间状态设置为“装入暂挂”或“删除暂挂”的对象。

元素标识

tablespace_state_change_object_id

元素类型

信息

表 1782. 快照监视信息

快照级别	逻辑数据分组	监视开关
表空间	tablespace_nodeinfo	基本

用法 仅当表空间状态为“装入暂挂”或“删除暂挂”时，此元素才有意义。如果此元素的值非零，那么此元素的值与视图 SYSCAT.TABLES 的列 TABLEID 中的值相匹配。

注: DB2 LOAD 不会将表空间状态设置为装入暂挂或删除暂挂。

tablespace_state_change_ts_id -“状态更改表空间标识”

如果表空间状态为“装入暂挂”或“删除暂挂”，那么此项显示导致设置表空间状态的对象的表空间标识。

元素标识

tablespace_state_change_ts_id

元素类型

信息

表 1783. 快照监视信息

快照级别	逻辑数据分组	监视开关
表空间	tablespace_nodeinfo	基本

用法 仅当表空间状态为“装入暂挂”或“删除暂挂”时，此元素才有意义。如果此元素的值非零，那么此元素的值与视图 SYSCAT.TABLES 的列 TABLESPACEID 中的值相匹配。

注: DB2 LOAD 不会将表空间状态设置为装入暂挂或删除暂挂。

tablespace_total_pages -“表空间中的总页数”监视元素

表空间中的总页数。

表 1784. 表函数监视信息

表函数	监视元素收集级别
MON_GET_TABLESPACE 表函数 - 获取表空间度量值	ACTIVITY METRICS BASE

表 1785. 快照监视信息

快照级别	逻辑数据分组	监视开关
表空间	tablespace_nodeinfo	基本（DMS 表空间）
		缓冲池（SMS 表空间）

用法

表空间占用的总操作系统空间。对于 DMS，此项是容器大小的总和。对于 SMS，此项是用于此表空间中存储的表的所有文件空间的总和（并且仅在缓冲池开关设置为 ON 时收集此项）。

tablespace_type -“表空间类型”监视元素

表空间的类型。

表 1786. 表函数监视信息

表函数	监视元素收集命令和级别
MON_GET_TABLESPACE 表函数 - 获取表空间度量值	ACTIVITY METRICS BASE

表 1787. 快照监视信息

快照级别	逻辑数据分组	监视开关
表空间	tablespace	基本

用法

此元素显示此表空间是数据库管理的表空间 (DMS) 还是系统管理的表空间 (SMS)。

tablespace_type（它是在 sqlmon.h 中定义的）的值如下所示:

- 对于 DMS: SQLM_TABLESPACE_TYP_DMS
- 对于 SMS: SQLM_TABLESPACE_TYP_SMS

tablespace_usable_pages -“表空间中的可用页数”监视元素

表空间中的总页数减去开销页数。

表 1788. 表函数监视信息

表函数	监视元素收集级别
MON_GET_TABLESPACE 表函数 - 获取表空间度量值	ACTIVITY METRICS BASE

表 1789. 快照监视信息

快照级别	逻辑数据分组	监视开关
表空间	tablespace_nodeinfo	基本（DMS 表空间）
		缓冲池（SMS 表空间）

用法

此元素仅适用于 DMS 表空间。对于 SMS 表空间，此元素的值与 **tablespace_total_pages** 的值相同。

在表空间重新平衡期间，可用页数将包括新添加的容器的页数，但在重新平衡完成之前，这些新页可能不会反映在空闲页数中。如果未进行表空间重新平衡，那么已使用页数加上空闲页数再加上暂挂空闲页数，将等于可用页数。

tablespace_used_pages -“表空间中的已使用页数”监视元素

当前在表空间中使用的（非空闲）总页数。

表 1790. 表函数监视信息

表函数	监视元素收集级别
MON_GET_TABLESPACE 表函数 - 获取表空间度量值	ACTIVITY METRICS BASE

表 1791. 快照监视信息

快照级别	逻辑数据分组	监视开关
表空间	tablespace_nodeinfo	基本（DMS 表空间） 缓冲池（SMS 表空间）

用法

这是用于 DMS 表空间的总页数。对于 SMS 表空间，它等于 **tablespace_total_pages** 监视元素的值。

tablespace_using_auto_storage -“已对表空间启用自动存储器”监视元素

此元素描述表空间是否被创建为自动存储器表空间。值为 1 表示“是”，值为 0 表示“否”。

表 1792. 表函数监视信息

表函数	监视元素收集命令和级别
MON_GET_TABLESPACE 表函数 - 获取表空间度量值	ACTIVITY METRICS BASE

表 1793. 快照监视信息

快照级别	逻辑数据分组	监视开关
表空间	tablespace	基本

用法

可使用此元素来确定是否使用自动存储器创建指定表空间（即使用 MANAGED BY AUTOMATIC STORAGE 子句创建），而不是使用显式提供的容器创建。表空间的某些容器可以在与数据库相关联的某些或所有存储器路径中。

target_cf_gbp_size -“目标集群高速缓存设施组缓冲池大小”监视元素

执行动态调整大小操作期间，此监视元素显示组缓冲池内存目标值（以大小为 4 KB 的页计）。目标值与所配置值相匹配时，调整大小操作完成。

表 1794. 表函数监视信息

表函数	监视元素收集级别
MON_GET_CF 表函数 - 获取 CF 度量值	ACTIVITY METRICS BASE

target_cf_lock_size -“目标集群高速缓存设施锁定大小”监视元素

执行动态调整大小操作期间，此监视元素显示全局锁定内存目标值（以大小为 4 KB 的页计）。目标值与所配置值相匹配时，调整大小操作完成。

表 1795. 表函数监视信息

表函数	监视元素收集级别
MON_GET_CF 表函数 - 获取 CF 度量值	ACTIVITY METRICS BASE

target_cf_sca_size -“目标集群高速缓存设施共享通信区大小”监视元素

执行动态调整大小操作期间，此监视元素显示共享通信区内存目标值（以大小为 4 KB 的页计）。目标值与所配置值相匹配时，调整大小操作完成。

表 1796. 表函数监视信息

表函数	监视元素收集级别
MON_GET_CF 表函数 - 获取 CF 度量值	ACTIVITY METRICS BASE

tbsp_datatag -“表空间数据标记”

此元素标识表空间的有效数据标记值。有效数据标记是对表空间显式指定的数据标记值或从表空间存储器组继承的数据标记值。有效用户指定范围是 1 到 0; 0 指示未指定任何数据标记。

表 1797. 表函数监视信息

表函数	监视元素收集级别
MON_GET_TABLESPACE 表函数 - 获取表空间度量值	ACTIVITY METRICS BASE

用法说明

数据标记用于在 WLM 配置内标识可引用的数据及对其分组。WLM 配置确定标记的效果，它可能影响用户工作的处理优先级。

tbsp_last_consec_page -“最后一个连续对象表页”监视元素

表空间的最后连续元数据页的对象相对页号。此值仅对 DMS 表空间有效。它本来为 0。

表 1798. 表函数监视信息

表函数	监视元素收集级别
MON_GET_TABLESPACE 表函数 - 获取表空间度量值	ACTIVITY METRICS BASE

tbsp_max_page_top -“最大表空间页号高水位标记”监视元素

上次激活数据库之后 DMS 表空间的最大分配页号。

表 1799. 表函数监视信息

表函数	监视元素收集级别
MON_GET_TABLESPACE 表函数 - 获取表空间度量值	ACTIVITY METRICS BASE

用法

每当 **tablespace_page_top** 监视元素的值递增时，此值都将更改。

tbsp_names -“表空间名称”

此元素列示实用程序作用于的表空间名称。

表 1800. 事件监视信息

事件类型	逻辑数据分组	监视开关
变更历史记录	UTILSTART	始终收集

用法

对于变更历史记录事件监视器，如果 object_type 元素为 DATABASE 或 TABLESPACE，那么这是实用程序作用于的表空间名称的逗号定界列表。

tbsp_trackmod_state -表空间 trackmod 状态监视元素

表空间在上次或下次备份时的修改状态。

表 1801. 表函数监视信息

表函数	监视元素收集级别
MON_GET_TABLESPACE 表函数 - 获取表空间度量值	始终

用法

可以使用此监视元素来确定表空间的修改状态。表空间的状态可以是下列其中一种状态:

CLEAN
自上次备份以来，表空间中未发生修改。如果此时执行增量备份或差异备份，那么不会备份此表空间中的数据页面。

DIRTY 表空间包含下次备份时需要读取的数据。

ININCREMENTAL
表空间包含已复制到增量备份中的修改。此状态是相对于完全备份的 DIRTY 状态，所以将来进行增量备份需要包括此池中的某些页面。此状态也是 CLEAN 状态，所以将来进行差异备份不需要包括此池中的任何页面。

READFULL
这是最新的表空间修改状态更改，如果完全备份读取脏表空间，而该备份操作可能尚未成功完成或者正在进行中，那么就会导致该更改。

READINCREMENTAL
这是最新的表空间修改状态更改，如果增量备份读取脏表空间，而该增量备份操作可能尚未成功完成或者正在进行中，那么就会导致该更改。

UNAVAILABLE
trackmod 配置参数已设置为 No。因此，不会提供表空间修改状态信息。

tcpip_recv_volume -“TCP/IP 接收量”监视元素

数据服务器通过 TCP/IP 从客户机接收的数据量。此值以字节计。

表 1802. 表函数监视信息

表函数	监视元素收集级别
MON_FORMAT_XML_METRICS_BY_ROW - 获取所有度量值的基于已格式化行的输出	不适用; 报告 XML 文档中作为格式化函数的输入提供的所有元素
MON_GET_CONNECTION 表函数 - 获取连接度量值	REQUEST METRICS BASE
MON_GET_CONNECTION_DETAILS 表函数 - 获取详细的连接度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE
MON_GET_SERVICE_SUBCLASS 表函数 - 获取服务子类度量值	REQUEST METRICS BASE
MON_GET_SERVICE_SUBCLASS_DETAILS 表函数 - 获取详细的服务子类度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE
MON_GET_UNIT_OF_WORK 表函数 - 获取工作单元度量值	REQUEST METRICS BASE
MON_GET_UNIT_OF_WORK 表函数 - 获取详细的工作单元度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE
MON_GET_WORKLOAD 表函数 - 获取工作负载度量值	REQUEST METRICS BASE

表 1802. 表函数监视信息 (续)

表函数	监视元素收集级别
MON_GET_WORKLOAD_DETAILS 表函数 - 获取详细的工作负载度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE

表 1803. 事件监视信息

事件类型	逻辑数据分组	监视开关
统计信息	event_scstats（在 details_xml 文档中报告）	REQUEST METRICS BASE
统计信息	event_wlstats（在 details_xml 文档中报告）	REQUEST METRICS BASE
工作单元	在 system_metrics 文档中报告。	-

tcpip_recv_wait_time -“TCP/IP 接收等待时间”监视元素

通过 TCP/IP 等待传入客户机请求时耗用的时间（不包括空闲时间）。此值以毫秒计。

表 1804. 表函数监视信息

表函数	监视元素收集级别
MON_FORMAT_XML_METRICS_BY_ROW - 获取所有度量值的基于已格式化行的输出	不适用; 报告 XML 文档中作为格式化函数的输入提供的所有元素
MON_FORMAT_XML_TIMES_BY_ROW - - 获取基于已格式化行的组合层次结构等待时间和处理时间	不适用; 报告 XML 文档中作为格式化函数的输入提供的所有元素
MON_FORMAT_XML_WAIT_TIMES_BY_ROW - 获取等待时间的已格式化的基于行的输出	不适用; 报告 XML 文档中作为格式化函数的输入提供的所有元素
MON_GET_CONNECTION 表函数 - 获取连接度量值	REQUEST METRICS BASE
MON_GET_CONNECTION_DETAILS 表函数 - 获取详细的连接度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE
MON_GET_SERVICE_SUBCLASS 表函数 - 获取服务子类度量值	REQUEST METRICS BASE
MON_GET_SERVICE_SUBCLASS_DETAILS 表函数 - 获取详细的服务子类度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE
MON_GET_UNIT_OF_WORK 表函数 - 获取工作单元度量值	REQUEST METRICS BASE
MON_GET_UNIT_OF_WORK 表函数 - 获取详细的工作单元度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE
MON_GET_WORKLOAD 表函数 - 获取工作负载度量值	REQUEST METRICS BASE

表 1804. 表函数监视信息 (续)

表函数	监视元素收集级别
MON_GET_WORKLOAD_DETAILS 表函数 - 获取详细的工作负载度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE

表 1805. 事件监视信息

事件类型	逻辑数据分组	监视开关
统计信息	event_scstats（在 details_xml 文档中报告）	REQUEST METRICS BASE
统计信息	event_wlstats（在 details_xml 文档中报告）	REQUEST METRICS BASE
工作单元	在 system_metrics 文档中报告。	-

tcpip_recvs_total -“TCP/IP 接收总次数”监视元素

数据库服务器通过 TCP/IP 从客户机应用程序接收数据的次数。

表 1806. 表函数监视信息

表函数	监视元素收集级别
MON_FORMAT_XML_METRICS_BY_ROW - 获取所有度量值的基于已格式化行的输出	不适用; 报告 XML 文档中作为格式化函数的输入提供的所有元素
MON_GET_CONNECTION 表函数 - 获取连接度量值	REQUEST METRICS BASE
MON_GET_CONNECTION_DETAILS 表函数 - 获取详细的连接度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE
MON_GET_SERVICE_SUBCLASS 表函数 - 获取服务子类度量值	REQUEST METRICS BASE
MON_GET_SERVICE_SUBCLASS_DETAILS 表函数 - 获取详细的服务子类度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE
MON_GET_UNIT_OF_WORK 表函数 - 获取工作单元度量值	REQUEST METRICS BASE
MON_GET_UNIT_OF_WORK 表函数 - 获取详细的工作单元度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE
MON_GET_WORKLOAD 表函数 - 获取工作负载度量值	REQUEST METRICS BASE
MON_GET_WORKLOAD_DETAILS 表函数 - 获取详细的工作负载度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE

表 1807. 事件监视信息

事件类型	逻辑数据分组	监视开关
统计信息	event_scstats（在 details_xml 文档中报告）	REQUEST METRICS BASE
统计信息	event_wlstats（在 details_xml 文档中报告）	REQUEST METRICS BASE
工作单元	在 system_metrics 文档中报告。	-

tcpip_send_volume -“TCP/IP 发送量”监视元素

数据服务器发送到客户机的数据量。此值以字节计。

表 1808. 表函数监视信息

表函数	监视元素收集级别
MON_FORMAT_XML_METRICS_BY_ROW - 获取所有度量值的基于已格式化行的输出	不适用; 报告 XML 文档中作为格式化函数的输入提供的所有元素
MON_GET_CONNECTION 表函数 - 获取连接度量值	REQUEST METRICS BASE
MON_GET_CONNECTION_DETAILS 表函数 - 获取详细的连接度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE
MON_GET_SERVICE_SUBCLASS 表函数 - 获取服务子类度量值	REQUEST METRICS BASE
MON_GET_SERVICE_SUBCLASS_DETAILS 表函数 - 获取详细的服务子类度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE
MON_GET_UNIT_OF_WORK 表函数 - 获取工作单元度量值	REQUEST METRICS BASE
MON_GET_UNIT_OF_WORK 表函数 - 获取详细的工作单元度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE
MON_GET_WORKLOAD 表函数 - 获取工作负载度量值	REQUEST METRICS BASE
MON_GET_WORKLOAD_DETAILS 表函数 - 获取详细的工作负载度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE

表 1809. 事件监视信息

事件类型	逻辑数据分组	监视开关
统计信息	event_scstats（在 details_xml 文档中报告）	REQUEST METRICS BASE
统计信息	event_wlstats（在 details_xml 文档中报告）	REQUEST METRICS BASE
工作单元	在 system_metrics 文档中报告。	-

tcpip_send_wait_time -“TCP/IP 发送等待时间”监视元素

通过 TCP/IP 向客户机发送数据时被阻塞的时间。此值以毫秒计。

表 1810. 表函数监视信息

表函数	监视元素收集级别
MON_FORMAT_XML_METRICS_BY_ROW - 获取所有度量值的基于已格式化行的输出	不适用; 报告 XML 文档中作为格式化函数的输入提供的所有元素
MON_FORMAT_XML_TIMES_BY_ROW - - 获取基于已格式化行的组合层次结构等待时间和处理时间	不适用; 报告 XML 文档中作为格式化函数的输入提供的所有元素
MON_FORMAT_XML_WAIT_TIMES_BY _ROW - 获取等待时间的已格式化的基于行的输出	不适用; 报告 XML 文档中作为格式化函数的输入提供的所有元素
MON_GET_CONNECTION 表函数 - 获取连接度量值	REQUEST METRICS BASE
MON_GET_CONNECTION_DETAILS 表函数 - 获取详细的连接度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE
MON_GET_SERVICE_SUBCLASS 表函数 - 获取服务子类度量值	REQUEST METRICS BASE
MON_GET_SERVICE_SUBCLASS_DETAILS 表函数 - 获取详细的服务子类度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE
MON_GET_UNIT_OF_WORK 表函数 - 获取工作单元度量值	REQUEST METRICS BASE
MON_GET_UNIT_OF_WORK 表函数 - 获取详细的工作单元度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE
MON_GET_WORKLOAD 表函数 - 获取工作负载度量值	REQUEST METRICS BASE
MON_GET_WORKLOAD_DETAILS 表函数 - 获取详细的工作负载度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE

表 1811. 事件监视信息

事件类型	逻辑数据分组	监视开关
统计信息	event_scstats（在 details_xml 文档中报告）	REQUEST METRICS BASE
统计信息	event_wlstats（在 details_xml 文档中报告）	REQUEST METRICS BASE
工作单元	在 system_metrics 文档中报告。	-

tcpip_sends_total -“TCP/IP 发送总次数”监视元素

通过 TCP/IP 从数据库服务器向客户机应用程序发送数据的次数。

表 1812. 表函数监视信息

表函数	监视元素收集级别
MON_FORMAT_XML_METRICS_BY_ROW - 获取所有度量值的基于已格式化行的输出	不适用; 报告 XML 文档中作为格式化函数的输入提供的所有元素
MON_GET_CONNECTION 表函数 - 获取连接度量值	REQUEST METRICS BASE
MON_GET_CONNECTION_DETAILS 表函数 - 获取详细的连接度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE
MON_GET_SERVICE_SUBCLASS 表函数 - 获取服务子类度量值	REQUEST METRICS BASE
MON_GET_SERVICE_SUBCLASS_DETAILS 表函数 - 获取详细的服务子类度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE
MON_GET_UNIT_OF_WORK 表函数 - 获取工作单元度量值	REQUEST METRICS BASE
MON_GET_UNIT_OF_WORK 表函数 - 获取详细的工作单元度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE
MON_GET_WORKLOAD 表函数 - 获取工作负载度量值	REQUEST METRICS BASE
MON_GET_WORKLOAD_DETAILS 表函数 - 获取详细的工作负载度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE

表 1813. 事件监视信息

事件类型	逻辑数据分组	监视开关
统计信息	event_scstats（在 details_xml 文档中报告）	REQUEST METRICS BASE
统计信息	event_wlstats（在 details_xml 文档中报告）	REQUEST METRICS BASE
工作单元	在 system_metrics 文档中报告。	始终收集

temp_tablespace_top -“最大临时表空间”监视元素

服务类或工作类中所有嵌套级别的 DML 活动使用的临时表空间的高水位标记（以 KB 计）。对于服务类，当服务类的 COLLECT AGGREGATE ACTIVITY DATA 设置为 NONE 时，此监视元素返回 -1。对于工作类，如果未对该工作类指定 COLLECT AGGREGATE ACTIVITY DATA 工作操作，那么此监视元素将返回 -1。 对于工作负载而言，当工作负载的 COLLECT AGGREGATE ACTIVITY DATA 设置为 NONE 时，此监视元素返回 -1。

对于服务类而言，使用 REMAP ACTIVITY 操作在服务子类之间重新映射活动时，将仅更改完成该活动的服务子类的 temp_tablespace_top 高水位标记。该活动所映射到但未在其中完成该活动的服务子类的高水位标记不受影响。

表 1814. 事件监视信息

事件类型	逻辑数据分组	监视开关
统计信息	event_scstats	-
统计信息	event_wcstats	-
统计信息	event_wlstats	-

用法

使用此元素可确定在收集时间间隔内，成员对服务类、工作负载或工作类上达到的最高 DML 活动系统临时表空间使用量。

此元素仅由对其应用了临时表空间阈值的活动更新。如果未对活动应用临时表空间阈值，那么将返回 0。

territory_code -“数据库地域代码”

对其收集监视器数据的数据库的地域代码。此监视元素先前称为 country_code。

表 1815. 快照监视信息

快照级别	逻辑数据分组	监视开关
应用程序	appl_info	基本
应用程序	appl	基本

表 1816. 事件监视信息

事件类型	逻辑数据分组	监视开关
事件日志头	event_log_header	始终收集
连接	event_connheader	始终收集

用法 地域代码信息记录在数据库配置文件中。

对于 DRDA AS 连接，此元素将设置为 0。

thresh_violations -“阈值违例次数”监视元素

违反阈值的次数。

此监视元素是 第 919 页的『 num_threshold_violations -“阈值违例次数”监视元素 』 监视元素的别名，后者由快照监视例程和数据库事件监视器返回。

表 1817. 表函数监视信息

表函数	监视元素收集级别
MON_FORMAT_XML_METRICS_BY_ROW - 获取所有度量值的基于已格式化行的输出	不适用；报告 XML 文档中作为格式化函数的输入提供的所有元素

表 1817. 表函数监视信息 (续)

表函数	监视元素收集级别
MON_GET_ACTIVITY_DETAILS 表函数 - 获取完整的活动详细信息（在 XML 文档 DETAILS 中报告）	ACTIVITY METRICS BASE
MON_GET_CONNECTION 表函数 - 获取连接度量值	REQUEST METRICS BASE
MON_GET_CONNECTION_DETAILS 表函数 - 获取详细的连接度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE
MON_GET_PKG_CACHE_STMT 表函数 - 获取程序包高速缓存中的 SQL 语句活动度量值	ACTIVITY METRICS BASE
MON_GET_PKG_CACHE_STMT_DETAILS 表函数 - 获取程序包高速缓存条目的详细度量值	ACTIVITY METRICS BASE
MON_GET_SERVICE_SUBCLASS 表函数 - 获取服务子类度量值	REQUEST METRICS BASE
MON_GET_SERVICE_SUBCLASS_DETAILS 表函数 - 获取详细的服务子类度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE
MON_GET_UNIT_OF_WORK 表函数 - 获取工作单元度量值	REQUEST METRICS BASE
MON_GET_UNIT_OF_WORK 表函数 - 获取详细的工作单元度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE
MON_GET_WORKLOAD 表函数 - 获取工作负载度量值	REQUEST METRICS BASE
MON_GET_WORKLOAD_DETAILS 表函数 - 获取详细的工作负载度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE

表 1818. 事件监视信息

事件类型	逻辑数据分组	监视开关
活动	event_activity（在 details_xml 文档中报告）	ACTIVITY METRICS BASE
活动	event_activitymetrics	ACTIVITY METRICS BASE
统计信息	event_scstats（在 details_xml 文档中报告）	REQUEST METRICS BASE
统计信息	event_wlstats（在 details_xml 文档中报告）	REQUEST METRICS BASE
程序包高速缓存	在 activity_metrics 文档中报告。	ACTIVITY METRICS BASE
工作单元	在 system_metrics 文档中报告。	REQUEST METRICS BASE

用法

使用此元素来快速确定是否已经违反了任何 WLM 阈值。如果已经违反了阈值，那么可以使用阈值违例事件监视器（如果已经创建了并且处于活动状态）来获取有关阈值违例的详细信息。

例如，获取有关违反了哪个阈值的详细信息。

threshold_action -“阈值操作”监视元素

此阈值违例记录适用的阈值操作。可能的值包括 Stop、Continue 和 Remap。

表 1819. 事件监视信息

事件类型	逻辑数据分组	监视开关
阈值违例	event_thresholdviolations	-

用法

使用此元素来确定违反阈值的活动在发生违例时被停止、被允许继续执行还是被重新映射到另一个服务子类。如果该活动被停止，那么提交活动的应用程序将接收到 SQL4712N 错误。如果该活动被重新映射到另一个服务子类，那么为该成员上的活动工作的代理程序将移至该阈值的目标服务子类。

threshold_domain -“阈值域”监视元素

负责此队列的阈值的域。

可能的值包括

- 数据库
- 工作操作集
- 服务超类
- 服务子类
- 工作负载

表 1820. 表函数监视信息

表函数	监视元素收集级别
WLM_GET_QUEUE_STATS 表函数 - 返回阈值队列统计信息	ACTIVITY METRICS BASE

表 1821. 事件监视信息

事件类型	逻辑数据分组	监视开关
统计信息	event_qstats	-

用法

此元素可用来区分谓词相同但域不同的阈值的队列统计信息。

threshold_maxvalue -“阈值最大值”监视元素

对于非队列阈值，此监视元素表示被超过而导致此阈值违例的值。对于队列阈值，此监视元素表示导致队列的并行级别。导致队列阈值违例的并行级别是 **threshold_maxvalue** 和 **threshold_queuesize** 监视元素之和。

表 1822. 事件监视信息

事件类型	逻辑数据分组	监视开关
阈值违例	event_thresholdviolations	始终收集

用法

对于活动阈值，此元素提供阈值违例时域值最大值的历史记录。如果阈值的最大值在违例后更改了，且旧值在 SYSCAT.THRESHOLDS 视图中不再可用，那么此元素将很有用。对于 DATATAGINSC IN 和 DATATAGINSC NOT IN 阈值，此元素包含违反阈值的数据标记的值。

threshold_name -“阈值名称”监视元素

负责此队列的阈值的唯一名称。

表 1823. 表函数监视信息

表函数	监视元素收集级别
WLM_GET_QUEUE_STATS 表函数 - 返回阈值队列统计信息	ACTIVITY METRICS BASE

表 1824. 事件监视信息

事件类型	逻辑数据分组	监视开关
统计信息	event_qstats	-

用法

使用此元素来唯一地标识此记录所表示的统计信息的队列阈值。

threshold_predicate -“阈值谓词”监视元素

标识已违反的阈值类型或对其收集统计信息的阈值类型。

表 1825. 表函数监视信息

表函数	监视元素收集级别
WLM_GET_QUEUE_STATS 表函数 - 返回阈值队列统计信息	ACTIVITY METRICS BASE

表 1826. 事件监视信息

事件类型	逻辑数据分组	监视开关
阈值违例	event_thresholdviolations	始终收集
统计信息	event_qstats	始终收集

用法

将此监视元素与其他统计信息或阈值违例监视元素配合使用来分析阈值违例。

在 event_thresholdviolations 逻辑组中报告时此监视元素的有效值包括:

AggSQLTempSpace

SQLTempSpace

SQLRowsReturned

ActivityTotalTime

EstimatedSQLCost

TotalMemberConnections

ConnectionIdleTime

ConcurrentWorkloadOccurrences

ConcurrentWorkloadActivities

ConcurrentDBCoordActivities

TotalSCMemberConnections

SQLRowsRead

SQLRowsReadInSC

CPUTime

CPUTimeInSC

UowTotalTime

DataTagInSC

DataTagNotInSC

在 event_qstats 逻辑组中报告时此监视元素的有效值包括:

TotalMemberConnections

ConcurrentDBCoordActivities

TotalSCMemberConnections

threshold_queuesize -“阈值队列大小”监视元素

队列阈值的队列大小。尝试超过此大小将导致阈值违例。对于非队列阈值，此值为 0。

表 1827. 事件监视信息

事件类型	逻辑数据分组	监视开关
阈值违例	event_thresholdviolations	-

用法

使用此元素来确定违反阈值时此阈值的队列中的活动或连接数。

thresholdid -“阈值标识”监视元素

标识阈值违例记录适用的阈值或为其收集队列统计信息的阈值。

表 1828. 表函数监视信息

表函数	监视元素收集级别
WLM_GET_QUEUE_STATS 表函数 - 返回阈值队列统计信息	ACTIVITY METRICS BASE

表 1829. 事件监视信息

事件类型	逻辑数据分组	监视开关
阈值违例	event_thresholdviolations	-
统计信息	event_qstats	-

用法

将此监视元素与其他活动历史监视元素配合使用来分析阈值队列或分析违反阈值的活动。

time_completed -“完成时间”监视元素

此活动记录描述的活动完成执行的时间。此元素为本地时间戳记。

表 1830. 事件监视信息

事件类型	逻辑数据分组	监视开关
活动	event_activity	始终收集

用法

将此元素与其他活动历史元素配合使用来分析活动的行为。

如果由于内存局限性而无法将完整活动记录写入表事件监视器，那么此字段的值将为“0000-00-00-00.00.00.000000”。如果活动在执行时被捕获，那么此字段表示收集活动的时间。

time_created -“创建时间”监视元素

用户提交此活动记录描述的活动的时间。此元素为本地时间戳记。

表 1831. 事件监视信息

事件类型	逻辑数据分组	监视开关
活动	event_activity	-

用法

将此元素与其他活动历史元素配合使用来分析活动的行为。

time_of_violation -“违例时间”监视元素

发此此阈值违例记录描述的阈值违例的时间。此元素为本地时间戳记。

表 1832. 事件监视信息

事件类型	逻辑数据分组	监视开关
阈值违例	event_thresholdviolations	-

用法

将此元素与其他阈值违例监视元素配合使用来分析阈值违例。

time_stamp -“快照时间”

收集数据库系统监视器信息的日期和时间。

表 1833. 快照监视信息

快照级别	逻辑数据分组	监视开关
数据库管理器	collected	基本

用法 如果要将结果保存在文件或数据库中以便将来进行分析，那么可使用此元素来帮助将相关数据按时间顺序排列。

time_started -“开始时间”监视元素

此活动记录描述的活动开始执行的时间。此元素为本地时间戳记。

表 1834. 事件监视信息

事件类型	逻辑数据分组	监视开关
活动	event_activity	始终收集

用法

将此元素与其他活动历史元素配合使用来分析活动的行为。

如果拒绝了该活动，那么 **act_exec_time** 监视元素的值为 0。在这种情况下，**time_started** 监视元素的值等于 **time_completed** 监视元素的值。

time_zone_disp -“时区偏移”

表示本地时区与格林威治标准时间（GMT）之间的偏移的秒数。

表 1835. 快照监视信息

快照级别	逻辑数据分组	监视开关
数据库管理器	collected	基本

用法 数据库系统监视器报告的所有时间为 GMT，而此偏移计算本地时间。

top -“最大直方图类别数”监视元素

直方图类别范围的包含顶端。此监视元素的值也是下一直方图类别的范围的不包含底端。

表 1836. 事件监视信息

事件类型	逻辑数据分组	监视开关
统计信息	event_histogrambin	-

用法

将此元素与相应的 **bottom** 元素配合使用来确定直方图内的类别范围。

tot_log_used_top -“使用的最大总日志空间”

使用的最大总日志空间（以字节计）。

表 1837. 表函数监视信息

表函数	监视元素收集级别
MON_GET_TRANSACTION_LOG 表函数 - 获取日志信息	ACTIVITY METRICS BASE

表 1838. 快照监视信息

快照级别	逻辑数据分组	监视开关
数据库	dbase	基本

表 1839. 事件监视信息

事件类型	逻辑数据分组	监视开关
数据库	event_db	始终收集

用法 可使用此元素来帮助评估分配的主日志空间量。将此元素的值与分配的主日志空间量进行比较可帮助您评估配置参数设置。可借助以下公式计算主日志空间分配:

```
logprimary x logfilsiz x 4096（请参阅以下注释）
```

可将此元素与 *sec_log_used_top* 和 *sec_logs_allocated* 一起使用来显示当前对辅助日志的依赖性。

此值同时包括在主日志文件和辅助日志文件中使用的空间。

可能需要调整下列配置参数:

- logfilsiz
- logprimary
- logsecond

注: 虽然数据库系统监视器信息是以字节为单位给定的，但配置参数是以页为单位设置的，每页为 4K 字节。

total_act_time -“活动时间总计”监视元素

执行活动时的耗用时间总计。此值以毫秒计。

表 1840. 表函数监视信息

表函数	监视元素收集级别
MON_FORMAT_XML_METRICS_BY_ROW - 获取所有度量值的基于已格式化行的输出	不适用; 报告 XML 文档中作为格式化函数的输入提供的所有元素
MON_GET_ACTIVITY_DETAILS 表函数 - 获取完整的活动详细信息（在 XML 文档 DETAILS 中报告）	ACTIVITY METRICS BASE
MON_GET_CONNECTION 表函数 - 获取连接度量值	REQUEST METRICS BASE
MON_GET_CONNECTION_DETAILS 表函数 - 获取详细的连接度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE
MON_GET_PKG_CACHE_STMT 表函数 - 获取程序包高速缓存中的 SQL 语句活动度量值	ACTIVITY METRICS BASE
MON_GET_PKG_CACHE_STMT_DETAILS 表函数 - 获取程序包高速缓存条目的详细度量值	ACTIVITY METRICS BASE
MON_GET_SERVICE_SUBCLASS 表函数 - 获取服务子类度量值	REQUEST METRICS BASE
MON_GET_SERVICE_SUBCLASS_DETAILS 表函数 - 获取详细的服务子类度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE
MON_GET_UNIT_OF_WORK 表函数 - 获取工作单元度量值	REQUEST METRICS BASE
MON_GET_UNIT_OF_WORK 表函数 - 获取详细的工作单元度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE
MON_GET_WORKLOAD 表函数 - 获取工作负载度量值	REQUEST METRICS BASE
MON_GET_WORKLOAD_DETAILS 表函数 - 获取详细的工作负载度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE

表 1841. 事件监视信息

事件类型	逻辑数据分组	监视开关
活动	event_activity（在 details_xml 文档中报告）	ACTIVITY METRICS BASE
活动	event_activitymetrics	ACTIVITY METRICS BASE
程序包高速缓存	在 activity_metrics 文档中报告。	ACTIVITY METRICS BASE
工作单元	在 system_metrics 文档中报告。	REQUEST METRICS BASE

用法

通过将此监视元素与 **total_act_wait_time** 监视元素配合使用，可以确定数据服务器处理此活动时耗用的时间所占的百分比。

```
(total_act_time - total_act_wait_time)/(total_act_time) =
 数据服务器主动处理活动的时间所占的百分比
```

total_act_wait_time -“活动等待时间总计”监视元素

处理活动期间，在 DB2 数据库服务器中进行等待时的耗用时间总计。此值以毫秒计。

表 1842. 表函数监视信息

表函数	监视元素收集级别
MON_FORMAT_XML_METRICS_BY_ROW - 获取所有度量值的基于已格式化行的输出	不适用; 报告 XML 文档中作为格式化函数的输入提供的所有元素
MON_FORMAT_XML_TIMES_BY_ROW - - 获取基于已格式化行的组合层次结构等待时间和处理时间	不适用; 报告 XML 文档中作为格式化函数的输入提供的所有元素
MON_FORMAT_XML_WAIT_TIMES_BY _ROW - 获取等待次数的已格式化的基于行的输出	不适用; 报告 XML 文档中作为格式化函数的输入提供的所有元素
MON_GET_ACTIVITY_DETAILS 表函数 - 获取完整的活动详细信息（在 XML 文档 DETAILS 中报告）	ACTIVITY METRICS BASE
MON_GET_CONNECTION 表函数 - 获取连接度量值	REQUEST METRICS BASE
MON_GET_CONNECTION_DETAILS 表函数 - 获取详细的连接度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE
MON_GET_PKG_CACHE_STMT 表函数 - 获取程序包高速缓存中的 SQL 语句活动度量值	ACTIVITY METRICS BASE
MON_GET_PKG_CACHE_STMT_DETAILS 表函数 - 获取程序包高速缓存条目的详细度量值	ACTIVITY METRICS BASE
MON_GET_SERVICE_SUBCLASS 表函数 - 获取服务子类度量值	REQUEST METRICS BASE
MON_GET_SERVICE_SUBCLASS_DETAILS 表函数 - 获取详细的服务子类度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE
MON_GET_UNIT_OF_WORK 表函数 - 获取工作单元度量值	REQUEST METRICS BASE
MON_GET_UNIT_OF_WORK 表函数 - 获取详细的工作单元度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE
MON_GET_WORKLOAD 表函数 - 获取工作负载度量值	REQUEST METRICS BASE
MON_GET_WORKLOAD_DETAILS 表函数 - 获取详细的工作负载度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE

表 1843. 事件监视信息

事件类型	逻辑数据分组	监视开关
活动	event_activity（在 details_xml 文档中报告）	ACTIVITY METRICS BASE
活动	event_activitymetrics	ACTIVITY METRICS BASE
程序包高速缓存	在 activity_metrics 文档中报告。	ACTIVITY METRICS BASE
工作单元	在 system_metrics 文档中报告。	REQUEST METRICS BASE

用法

通过将此监视元素与 **total_act_time** 监视元素配合使用，可以确定数据服务器处理此活动时耗用的时间所占的百分比。

```
(total_act_time - total_act_wait_time)/(total_act_time) =
 数据服务器主动处理活动的时间所占的百分比
```

total_app_commits -“应用程序落实次数总计”监视元素

客户机应用程序发出的 commit 语句总数。

表 1844. 表函数监视信息

表函数	监视元素收集级别
MON_FORMAT_XML_METRICS_BY_ROW - 获取所有度量值的基于已格式化行的输出	不适用；报告 XML 文档中作为格式化函数的输入提供的所有元素
MON_GET_CONNECTION 表函数 - 获取连接度量值	REQUEST METRICS BASE
MON_GET_CONNECTION_DETAILS 表函数 - 获取详细的连接度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE
MON_GET_SERVICE_SUBCLASS 表函数 - 获取服务子类度量值	REQUEST METRICS BASE
MON_GET_SERVICE_SUBCLASS_DETAILS 表函数 - 获取详细的服务子类度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE
MON_GET_UNIT_OF_WORK 表函数 - 获取工作单元度量值	REQUEST METRICS BASE
MON_GET_UNIT_OF_WORK 表函数 - 获取详细的工作单元度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE
MON_GET_WORKLOAD 表函数 - 获取工作负载度量值	REQUEST METRICS BASE
MON_GET_WORKLOAD_DETAILS 表函数 - 获取详细的工作负载度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE

表 1845. 事件监视信息

事件类型	逻辑数据分组	监视开关
统计信息	event_scstats（在 details_xml 文档中报告）	REQUEST METRICS BASE
统计信息	event_wlstats（在 details_xml 文档中报告）	REQUEST METRICS BASE
工作单元	在 system_metrics 文档中报告。	REQUEST METRICS BASE

total_app_rollbacks -“应用程序回滚次数总计”监视元素

客户机应用程序发出的 rollback 语句总数。

表 1846. 表函数监视信息

表函数	监视元素收集级别
MON_FORMAT_XML_METRICS_BY_ROW - 获取所有度量值的基于已格式化行的输出	不适用; 报告 XML 文档中作为格式化函数的输入提供的所有元素
MON_GET_CONNECTION 表函数 - 获取连接度量值	REQUEST METRICS BASE
MON_GET_CONNECTION_DETAILS 表函数 - 获取详细的连接度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE
MON_GET_SERVICE_SUBCLASS 表函数 - 获取服务子类度量值	REQUEST METRICS BASE
MON_GET_SERVICE_SUBCLASS_DETAILS 表函数 - 获取详细的服务子类度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE
MON_GET_UNIT_OF_WORK 表函数 - 获取工作单元度量值	REQUEST METRICS BASE
MON_GET_UNIT_OF_WORK 表函数 - 获取详细的工作单元度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE
MON_GET_WORKLOAD 表函数 - 获取工作负载度量值	REQUEST METRICS BASE
MON_GET_WORKLOAD_DETAILS 表函数 - 获取详细的工作负载度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE

表 1847. 事件监视信息

事件类型	逻辑数据分组	监视开关
统计信息	event_scstats（在 details_xml 文档中报告）	REQUEST METRICS BASE
统计信息	event_wlstats（在 details_xml 文档中报告）	REQUEST METRICS BASE
工作单元	在 system_metrics 文档中报告。	REQUEST METRICS BASE

total_app_rqst_time -“应用程序请求时间总计”监视元素

处理应用程序请求时的耗用时间总计；这是服务器上的协调代理程序执行应用程序请求时的耗用时间总计。此值以毫秒计。

表 1848. 表函数监视信息

表函数	监视元素收集级别
MON_FORMAT_XML_METRICS_BY_ROW - 获取所有度量值的基于已格式化行的输出	不适用；报告 XML 文档中作为格式化函数的输入提供的所有元素
MON_GET_CONNECTION 表函数 - 获取连接度量值	REQUEST METRICS BASE
MON_GET_CONNECTION_DETAILS 表函数 - 获取详细的连接度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE
MON_GET_SERVICE_SUBCLASS 表函数 - 获取服务子类度量值	REQUEST METRICS BASE
MON_GET_SERVICE_SUBCLASS_DETAILS 表函数 - 获取详细的服务子类度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE
MON_GET_UNIT_OF_WORK 表函数 - 获取工作单元度量值	REQUEST METRICS BASE
MON_GET_UNIT_OF_WORK 表函数 - 获取详细的工作单元度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE
MON_GET_WORKLOAD 表函数 - 获取工作负载度量值	REQUEST METRICS BASE
MON_GET_WORKLOAD_DETAILS 表函数 - 获取详细的工作负载度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE

用法

使用此监视元素来确定应用程序请求在 DB2 数据服务器中耗用的时间。此值可用于帮助确定数据服务器是否是所检测到的性能问题的根源。

例如，如果用户报告应用程序遇到问题，其返回时间长达 20 分钟，而您确定应用程序请求时间总计为 1 分钟，并且当前未通过该连接处理应用程序请求，那么性能问题可能并非由 DB2 数据服务器所致。

total_app_section_executions -“应用程序执行部分执行的总次数”监视元素

应用程序执行部分执行的次数。

表 1849. 表函数监视信息

表函数	监视元素收集级别
MON_FORMAT_XML_METRICS_BY_ROW - 获取所有度量值的基于已格式化行的输出	不适用；报告 XML 文档中作为格式化函数的输入提供的所有元素

表 1849. 表函数监视信息 (续)

表函数	监视元素收集级别
MON_GET_ACTIVITY_DETAILS 表函数 - 获取完整的活动详细信息（在 XML 文档 DETAILS 中报告）	ACTIVITY METRICS BASE
MON_GET_CONNECTION 表函数 - 获取连接度量值	REQUEST METRICS BASE
MON_GET_CONNECTION_DETAILS 表函数 - 获取详细的连接度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE
MON_GET_PKG_CACHE_STMT 表函数 - 获取程序包高速缓存中的 SQL 语句活动度量值	ACTIVITY METRICS BASE
MON_GET_PKG_CACHE_STMT_DETAILS 表函数 - 获取程序包高速缓存条目的详细度量值	ACTIVITY METRICS BASE
MON_GET_SERVICE_SUBCLASS 表函数 - 获取服务子类度量值	REQUEST METRICS BASE
MON_GET_SERVICE_SUBCLASS_DETAILS 表函数 - 获取详细的服务子类度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE
MON_GET_UNIT_OF_WORK 表函数 - 获取工作单元度量值	REQUEST METRICS BASE
MON_GET_UNIT_OF_WORK 表函数 - 获取详细的工作单元度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE
MON_GET_WORKLOAD 表函数 - 获取工作负载度量值	REQUEST METRICS BASE
MON_GET_WORKLOAD_DETAILS 表函数 - 获取详细的工作负载度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE

表 1850. 事件监视信息

事件类型	逻辑数据分组	监视开关
活动	event_activity（在 details_xml 文档中报告）	ACTIVITY METRICS BASE
活动	event_activitymetrics	ACTIVITY METRICS BASE
统计信息	event_scstats（在 details_xml 文档中报告）	REQUEST METRICS BASE
统计信息	event_wlstats（在 details_xml 文档中报告）	REQUEST METRICS BASE
程序包高速缓存	在 activity_metrics 文档中报告。	ACTIVITY METRICS BASE
工作单元	在 system_metrics 文档中报告。	REQUEST METRICS BASE

total_buffers_rcvd -“接收到的 FCM 缓冲区总数”

对于快照监视器，此监视元素报告由发出 GET SNAPSHOT 命令的节点从 **node_number** 监视元素所标识的节点中接收到的 FCM 缓冲区总数。对于表函数监视器，此监视元素报告从远程数据库成员接收到的 FCM 缓冲区总数。

表 1851. 表函数监视信息

表函数	监视元素收集级别
MON_GET_FCM_CONNECTION_LIST - 获取有关所有 FCM 连接的详细信息	ACTIVITY METRICS BASE

表 1852. 快照监视信息

快照级别	逻辑数据分组	监视开关
数据库管理器	fcm_node	基本

用法

使用此元素来度量当前成员与远程成员之间的通信量级别。如果从此成员接收到的 FCM 缓冲区总数很高，那么可考虑重新分发数据库或者移动表，以降低成员之间的通信量。

total_buffers_sent -“发送的 FCM 缓冲区总数”

对于快照监视器，此监视元素报告已经从发出 GET SNAPSHOT 命令的节点发送至 **node_number** 监视元素所标识的节点的 FCM 缓冲区总数。对于表函数监视器，此监视元素报告从当前数据库成员发送至远程数据库成员的 FCM 缓冲区总数。

表 1853. 表函数监视信息

表函数	监视元素收集级别
MON_GET_FCM_CONNECTION_LIST - 获取有关所有 FCM 连接的详细信息	ACTIVITY METRICS BASE

表 1854. 快照监视信息

快照级别	逻辑数据分组	监视开关
数据库管理器	fcm_node	基本

用法

使用此元素来度量当前成员与远程成员之间的通信量级别。如果发送至此成员的 FCM 缓冲区总数很高，那么可考虑重新分发数据库或者移动表，以降低成员之间的通信量。

total_bytes_received -“所接收字节数”监视元素

网络适配器启动后接收的字节总数。

表 1855. 表函数监视信息

表函数	监视元素收集级别
ENV_GET_NETWORK_RESOURCES 表函数 - 返回网络适配器信息	ACTIVITY METRICS BASE

total_bytes_sent -“所发送字节数”监视元素

网络适配器启动后发送的字节总数。

表 1856. 表函数监视信息

表函数	监视元素收集级别
ENV_GET_NETWORK_RESOURCES 表函数 - 返回网络适配器信息	ACTIVITY METRICS BASE

total_commit_proc_time -“落实处理时间总计”监视元素

在数据库服务器上执行落实处理所花的处理（无等待）时间总计。此值以毫秒计。

表 1857. 表函数监视信息

表函数	监视元素收集级别
MON_FORMAT_XML_METRICS_BY_ROW - 获取所有度量值的基于已格式化行的输出	不适用; 报告 XML 文档中作为格式化函数的输入提供的所有元素
MON_FORMAT_XML_TIMES_BY_ROW - - 获取基于已格式化行的组合层次结构等待时间和处理时间	不适用; 报告 XML 文档中作为格式化函数的输入提供的所有元素
MON_GET_CONNECTION 表函数 - 获取连接度量值	REQUEST METRICS BASE
MON_GET_CONNECTION_DETAILS 表函数 - 获取详细的连接度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE
MON_GET_SERVICE_SUBCLASS 表函数 - 获取服务子类度量值	REQUEST METRICS BASE
MON_GET_SERVICE_SUBCLASS_DETAILS 表函数 - 获取详细的服务子类度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE
MON_GET_UNIT_OF_WORK 表函数 - 获取工作单元度量值	REQUEST METRICS BASE
MON_GET_UNIT_OF_WORK 表函数 - 获取详细的工作单元度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE
MON_GET_WORKLOAD 表函数 - 获取工作负载度量值	REQUEST METRICS BASE

表 1857. 表函数监视信息 (续)

表函数	监视元素收集级别
MON_GET_WORKLOAD_DETAILS 表函数 - 获取详细的工作负载度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE

表 1858. 事件监视信息

事件类型	逻辑数据分组	监视开关
统计信息	event_scstats（在 details_xml 文档中报告）	REQUEST METRICS BASE
统计信息	event_wlstats（在 details_xml 文档中报告）	REQUEST METRICS BASE
工作单元	在 system_metrics 文档中报告。	REQUEST METRICS BASE

total_commit_time -“落实时间总计”监视元素

在数据库服务器上执行落实处理所花的总时间。此值以毫秒计。

表 1859. 表函数监视信息

表函数	监视元素收集级别
MON_FORMAT_XML_COMPONENT _TIMES_BY_ROW - 获取基于已格式化行的组件次数	不适用；报告 XML 文档中作为格式化函数的输入提供的所有元素
MON_FORMAT_XML_METRICS_BY_ROW - 获取所有度量值的基于已格式化行的输出	不适用；报告 XML 文档中作为格式化函数的输入提供的所有元素
MON_GET_CONNECTION 表函数 - 获取连接度量值	REQUEST METRICS BASE
MON_GET_CONNECTION_DETAILS 表函数 - 获取详细的连接度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE
MON_GET_SERVICE_SUBCLASS 表函数 - 获取服务子类度量值	REQUEST METRICS BASE
MON_GET_SERVICE_SUBCLASS_DETAILS 表函数 - 获取详细的服务子类度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE
MON_GET_UNIT_OF_WORK 表函数 - 获取工作单元度量值	REQUEST METRICS BASE
MON_GET_UNIT_OF_WORK 表函数 - 获取详细的工作单元度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE
MON_GET_WORKLOAD 表函数 - 获取工作负载度量值	REQUEST METRICS BASE
MON_GET_WORKLOAD_DETAILS 表函数 - 获取详细的工作负载度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE

表 1860. 事件监视信息

事件类型	逻辑数据分组	监视开关
统计信息	event_scstats（在 details_xml 文档中报告）	REQUEST METRICS BASE
统计信息	event_wlstats（在 details_xml 文档中报告）	REQUEST METRICS BASE
工作单元	在 system_metrics 文档中报告。	REQUEST METRICS BASE

total_compilations -“编译次数总计”监视元素

数据库服务器上的显式编译的总数。显式编译是由用户请求（例如，绑定、重新绑定、准备或立即执行）直接启动的编译。

表 1861. 表函数监视信息

表函数	监视元素收集级别
MON_FORMAT_XML_METRICS_BY_ROW - 获取所有度量值的基于已格式化行的输出	不适用; 报告 XML 文档中作为格式化函数的输入提供的所有元素
MON_GET_CONNECTION 表函数 - 获取连接度量值	REQUEST METRICS BASE
MON_GET_CONNECTION_DETAILS 表函数 - 获取详细的连接度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE
MON_GET_SERVICE_SUBCLASS 表函数 - 获取服务子类度量值	REQUEST METRICS BASE
MON_GET_SERVICE_SUBCLASS_DETAILS 表函数 - 获取详细的服务子类度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE
MON_GET_UNIT_OF_WORK 表函数 - 获取工作单元度量值	REQUEST METRICS BASE
MON_GET_UNIT_OF_WORK 表函数 - 获取详细的工作单元度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE
MON_GET_WORKLOAD 表函数 - 获取工作负载度量值	REQUEST METRICS BASE
MON_GET_WORKLOAD_DETAILS 表函数 - 获取详细的工作负载度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE

表 1862. 事件监视信息

事件类型	逻辑数据分组	监视开关
统计信息	event_scstats（在 details_xml 文档中报告）	REQUEST METRICS BASE
统计信息	event_wlstats（在 details_xml 文档中报告）	REQUEST METRICS BASE
工作单元	在 system_metrics 文档中报告。	REQUEST METRICS BASE

total_compile_proc_time -“编译处理时间总计”监视元素

在数据库服务器上执行显式编译所花的处理（无等待）时间总计。显式编译是由用户请求（例如，绑定、重新绑定、准备或立即执行）直接启动的编译。此值以毫秒计。

表 1863. 表函数监视信息

表函数	监视元素收集级别
MON_FORMAT_XML_METRICS_BY_ROW - 获取所有度量值的基于已格式化行的输出	不适用; 报告 XML 文档中作为格式化函数的输入提供的所有元素
MON_FORMAT_XML_TIMES_BY_ROW - - 获取基于已格式化行的组合层次结构等待时间和处理时间	不适用; 报告 XML 文档中作为格式化函数的输入提供的所有元素
MON_GET_CONNECTION 表函数 - 获取连接度量值	REQUEST METRICS BASE
MON_GET_CONNECTION_DETAILS 表函数 - 获取详细的连接度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE
MON_GET_SERVICE_SUBCLASS 表函数 - 获取服务子类度量值	REQUEST METRICS BASE
MON_GET_SERVICE_SUBCLASS_DETAILS 表函数 - 获取详细的服务子类度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE
MON_GET_UNIT_OF_WORK 表函数 - 获取工作单元度量值	REQUEST METRICS BASE
MON_GET_UNIT_OF_WORK 表函数 - 获取详细的工作单元度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE
MON_GET_WORKLOAD 表函数 - 获取工作负载度量值	REQUEST METRICS BASE
MON_GET_WORKLOAD_DETAILS 表函数 - 获取详细的工作负载度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE

表 1864. 事件监视信息

事件类型	逻辑数据分组	监视开关
统计信息	event_scstats（在 details_xml 文档中报告）	REQUEST METRICS BASE
统计信息	event_wlstats（在 details_xml 文档中报告）	REQUEST METRICS BASE
工作单元	在 system_metrics 文档中报告。	REQUEST METRICS BASE

total_compile_time -“编译时间总计”监视元素

在数据库服务器上执行显式编译所花的总时间。显式编译是由用户请求（例如，绑定、重新绑定、准备或立即执行）直接启动的编译。此值以毫秒计。

表 1865. 表函数监视信息

表函数	监视元素收集级别
MON_FORMAT_XML_COMPONENT _TIMES_BY_ROW - 获取基于已格式化行的组件次数	不适用; 报告 XML 文档中作为格式化函数的输入提供的所有元素
MON_FORMAT_XML_METRICS_BY_ROW - 获取所有度量值的基于已格式化行的输出	不适用; 报告 XML 文档中作为格式化函数的输入提供的所有元素
MON_GET_CONNECTION 表函数 - 获取连接度量值	REQUEST METRICS BASE
MON_GET_CONNECTION_DETAILS 表函数 - 获取详细的连接度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE
MON_GET_SERVICE_SUBCLASS 表函数 - 获取服务子类度量值	REQUEST METRICS BASE
MON_GET_SERVICE_SUBCLASS_DETAILS 表函数 - 获取详细的服务子类度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE
MON_GET_UNIT_OF_WORK 表函数 - 获取工作单元度量值	REQUEST METRICS BASE
MON_GET_UNIT_OF_WORK 表函数 - 获取详细的工作单元度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE
MON_GET_WORKLOAD 表函数 - 获取工作负载度量值	REQUEST METRICS BASE
MON_GET_WORKLOAD_DETAILS 表函数 - 获取详细的工作负载度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE

表 1866. 事件监视信息

事件类型	逻辑数据分组	监视开关
统计信息	event_scstats（在 details_xml 文档中报告）	REQUEST METRICS BASE
统计信息	event_wlstats（在 details_xml 文档中报告）	REQUEST METRICS BASE
工作单元	在 system_metrics 文档中报告。	REQUEST METRICS BASE

total_cons -“数据库激活以后的连接数”

指示第一次连接、激活或上一次重置（协调代理程序）后与数据库的连接的数目。

表 1867. 快照监视信息

快照级别	逻辑数据分组	监视开关
数据库	dbase	基本
数据库	dbase_remote	基本

可将快照监视的计数器重置。

表 1868. 事件监视信息

事件类型	逻辑数据分组	监视开关
数据库	event_db	始终收集

用法 可将此元素与 db_conn_time 和 db2start_time 监视元素配合使用以计算应用程序与数据库进行连接的频率。

如果连接频率很低，那么您可能想要在连接任何其他应用程序之前使用 ACTIVATE DATABASE 命令显式激活数据库，这是因为第一个与数据库的连接存在额外处理时间（例如，初始缓冲池分配）。这将导致后续连接以较高的频率进行处理。

注: 重置此元素时，其值将设置为当前连接的应用程序数而不是设置为零。

total_connect_authentication_proc_time -“连接认证处理时间总计”监视元素

执行连接或交换机用户认证时所花的处理（非等待）时间（以毫秒计）。

表 1869. 表函数监视信息

表函数	监视元素收集级别
MON_FORMAT_XML_METRICS_BY_ROW - 获取所有度量值的基于已格式化行的输出	不适用; 报告 XML 文档中作为格式化函数的输入提供的所有元素
MON_FORMAT_XML_TIMES_BY_ROW - - 获取基于已格式化行的组合层次结构等待时间和处理时间	不适用; 报告 XML 文档中作为格式化函数的输入提供的所有元素
MON_GET_CONNECTION 表函数 - 获取连接度量值	REQUEST METRICS BASE
MON_GET_CONNECTION_DETAILS 表函数 - 获取详细的连接度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE
MON_GET_SERVICE_SUBCLASS 表函数 - 获取服务子类度量值	REQUEST METRICS BASE
MON_GET_SERVICE_SUBCLASS_DETAILS 表函数 - 获取详细的服务子类度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE

表 1869. 表函数监视信息 (续)

表函数	监视元素收集级别
MON_GET_UNIT_OF_WORK 表函数 - 获取工作单元度量值	REQUEST METRICS BASE
MON_GET_UNIT_OF_WORK 表函数 - 获取详细的工作单元度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE
MON_GET_WORKLOAD 表函数 - 获取工作负载度量值	REQUEST METRICS BASE
MON_GET_WORKLOAD_DETAILS 表函数 - 获取详细的工作负载度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE

表 1870. 事件监视信息

事件类型	逻辑数据分组	监视开关
统计信息	event_scstats（在 details_xml 文档中报告）	REQUEST METRICS BASE
统计信息	event_wlstats（在 details_xml 文档中报告）	REQUEST METRICS BASE
工作单元	在 system_metrics 文档中报告。	REQUEST METRICS BASE

total_connect_authentications -“执行的连接或交换机用户认证数”监视元素

执行的连接或交换机用户认证数。

表 1871. 表函数监视信息

表函数	监视元素收集级别
MON_FORMAT_XML_METRICS_BY_ROW - 获取所有度量值的基于已格式化行的输出	不适用; 报告 XML 文档中作为格式化函数的输入提供的所有元素
MON_GET_CONNECTION 表函数 - 获取连接度量值	REQUEST METRICS BASE
MON_GET_CONNECTION_DETAILS 表函数 - 获取详细的连接度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE
MON_GET_SERVICE_SUBCLASS 表函数 - 获取服务子类度量值	REQUEST METRICS BASE
MON_GET_SERVICE_SUBCLASS_DETAILS 表函数 - 获取详细的服务子类度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE
MON_GET_UNIT_OF_WORK 表函数 - 获取工作单元度量值	REQUEST METRICS BASE
MON_GET_UNIT_OF_WORK 表函数 - 获取详细的工作单元度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE
MON_GET_WORKLOAD 表函数 - 获取工作负载度量值	REQUEST METRICS BASE

表 1871. 表函数监视信息 (续)

表函数	监视元素收集级别
MON_GET_WORKLOAD_DETAILS 表函数 - 获取详细的工作负载度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE

表 1872. 事件监视信息

事件类型	逻辑数据分组	监视开关
统计信息	event_scstats（在 details_xml 文档中报告）	REQUEST METRICS BASE
统计信息	event_wlstats（在 details_xml 文档中报告）	REQUEST METRICS BASE
工作单元	uow（在 metrics.xml 文档中报告） uow_metrics	REQUEST METRICS BASE

total_connect_authentication_time -“连接或交换机用户认证请求时间总计”监视元素

执行连接或交换机用户认证时所花的时间（以毫秒计）。

表 1873. 表函数监视信息

表函数	监视元素收集级别
MON_FORMAT_XML_COMPONENT_TIMES_BY_ROW - 获取基于已格式化行的组件次数	不适用; 报告 XML 文档中作为格式化函数的输入提供的所有元素
MON_FORMAT_XML_METRICS_BY_ROW - 获取所有度量值的基于已格式化行的输出	不适用; 报告 XML 文档中作为格式化函数的输入提供的所有元素
MON_GET_CONNECTION 表函数 - 获取连接度量值	REQUEST METRICS BASE
MON_GET_CONNECTION_DETAILS 表函数 - 获取详细的连接度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE
MON_GET_SERVICE_SUBCLASS 表函数 - 获取服务子类度量值	REQUEST METRICS BASE
MON_GET_SERVICE_SUBCLASS_DETAILS 表函数 - 获取详细的服务子类度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE
MON_GET_UNIT_OF_WORK 表函数 - 获取工作单元度量值	REQUEST METRICS BASE
MON_GET_UNIT_OF_WORK 表函数 - 获取详细的工作单元度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE
MON_GET_WORKLOAD 表函数 - 获取工作负载度量值	REQUEST METRICS BASE

表 1873. 表函数监视信息 (续)

表函数	监视元素收集级别
MON_GET_WORKLOAD_DETAILS 表函数 - 获取详细的工作负载度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE

表 1874. 事件监视信息

事件类型	逻辑数据分组	监视开关
统计信息	event_scstats（在 details_xml 文档中报告）	REQUEST METRICS BASE
统计信息	event_wlstats（在 details_xml 文档中报告）	REQUEST METRICS BASE
工作单元	uow（在 metrics.xml 文档中报告） uow_metrics	REQUEST METRICS BASE

total_connect_request_proc_time -“连接或交换机用户请求处理时间总计”监视元素

处理连接或交换机用户请求时所花的处理（非等待）时间（以毫秒计）。

表 1875. 表函数监视信息

表函数	监视元素收集级别
MON_FORMAT_XML_METRICS_BY_ROW - 获取所有度量值的基于已格式化行的输出	不适用; 报告 XML 文档中作为格式化函数的输入提供的所有元素
MON_FORMAT_XML_TIMES_BY_ROW - - 获取基于已格式化行的组合层次结构等待时间和处理时间	不适用; 报告 XML 文档中作为格式化函数的输入提供的所有元素
MON_GET_CONNECTION 表函数 - 获取连接度量值	REQUEST METRICS BASE
MON_GET_CONNECTION_DETAILS 表函数 - 获取详细的连接度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE
MON_GET_SERVICE_SUBCLASS 表函数 - 获取服务子类度量值	REQUEST METRICS BASE
MON_GET_SERVICE_SUBCLASS_DETAILS 表函数 - 获取详细的服务子类度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE
MON_GET_UNIT_OF_WORK 表函数 - 获取工作单元度量值	REQUEST METRICS BASE
MON_GET_UNIT_OF_WORK 表函数 - 获取详细的工作单元度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE
MON_GET_WORKLOAD 表函数 - 获取工作负载度量值	REQUEST METRICS BASE

表 1875. 表函数监视信息 (续)

表函数	监视元素收集级别
MON_GET_WORKLOAD_DETAILS 表函数 - 获取详细的工作负载度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE

表 1876. 事件监视信息

事件类型	逻辑数据分组	监视开关
统计信息	event_scstats（在 details_xml 文档中报告）	REQUEST METRICS BASE
统计信息	event_wlstats（在 details_xml 文档中报告）	REQUEST METRICS BASE
工作单元	uow（在 metrics.xml 文档中报告） uow_metrics	REQUEST METRICS BASE

total_connect_requests -“连接或交换机用户请求数”监视元素

连接或交换机用户请求的总数。

表 1877. 表函数监视信息

表函数	监视元素收集级别
MON_FORMAT_XML_METRICS_BY_ROW - 获取所有度量值的基于已格式化行的输出	不适用; 报告 XML 文档中作为格式化函数的输入提供的所有元素
MON_GET_CONNECTION 表函数 - 获取连接度量值	REQUEST METRICS BASE
MON_GET_CONNECTION_DETAILS 表函数 - 获取详细的连接度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE
MON_GET_SERVICE_SUBCLASS 表函数 - 获取服务子类度量值	REQUEST METRICS BASE
MON_GET_SERVICE_SUBCLASS_DETAILS 表函数 - 获取详细的服务子类度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE
MON_GET_UNIT_OF_WORK 表函数 - 获取工作单元度量值	REQUEST METRICS BASE
MON_GET_UNIT_OF_WORK 表函数 - 获取详细的工作单元度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE
MON_GET_WORKLOAD 表函数 - 获取工作负载度量值	REQUEST METRICS BASE
MON_GET_WORKLOAD_DETAILS 表函数 - 获取详细的工作负载度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE

表 1878. 事件监视信息

事件类型	逻辑数据分组	监视开关
统计信息	event_scstats（在 details_xml 文档中报告）	REQUEST METRICS BASE
统计信息	event_wlstats（在 details_xml 文档中报告）	REQUEST METRICS BASE
工作单元	uow（在 metrics.xml 文档中报告） uow_metrics	REQUEST METRICS BASE

total_connect_request_time -“连接或交换机用户请求时间总计”监视元素

执行连接或交换机用户请求时所花的时间（以毫秒计）。

表 1879. 表函数监视信息

表函数	监视元素收集级别
MON_FORMAT_XML_COMPONENT _TIMES_BY_ROW - 获取基于已格式化行的组件次数	不适用; 报告 XML 文档中作为格式化函数的输入提供的所有元素
MON_FORMAT_XML_METRICS_BY_ROW - 获取所有度量值的基于已格式化行的输出	不适用; 报告 XML 文档中作为格式化函数的输入提供的所有元素
MON_GET_CONNECTION 表函数 - 获取连接度量值	REQUEST METRICS BASE
MON_GET_CONNECTION_DETAILS 表函数 - 获取详细的连接度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE
MON_GET_SERVICE_SUBCLASS 表函数 - 获取服务子类度量值	REQUEST METRICS BASE
MON_GET_SERVICE_SUBCLASS_DETAILS 表函数 - 获取详细的服务子类度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE
MON_GET_UNIT_OF_WORK 表函数 - 获取工作单元度量值	REQUEST METRICS BASE
MON_GET_UNIT_OF_WORK 表函数 - 获取详细的工作单元度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE
MON_GET_WORKLOAD 表函数 - 获取工作负载度量值	REQUEST METRICS BASE
MON_GET_WORKLOAD_DETAILS 表函数 - 获取详细的工作负载度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE

表 1880. 事件监视信息

事件类型	逻辑数据分组	监视开关
统计信息	event_scstats（在 details_xml 文档中报告）	REQUEST METRICS BASE

表 1880. 事件监视信息 (续)

事件类型	逻辑数据分组	监视开关
统计信息	event_wlstats（在 details_xml 文档中报告）	REQUEST METRICS BASE
工作单元	uow（在 metrics.xml 文档中报告） uow_metrics	REQUEST METRICS BASE

total_cpu_time -“CPU 时间总计”监视元素

在 DB2 中耗用的 CPU 时间总计。此值代表用户 CPU 时间与系统 CPU 时间的总计。此值以微秒计。

如果 WLM_GET_SERVICE_SUBCLASS_STATS 或 WLM_GET_WORKLOAD_STATS 表函数返回此监视元素，那么此监视元素表示自上次重置统计信息以来的 CPU 时间总计。如果由 MON_SAMPLE_SERVICE_CLASS_METRICS 或 MON_SAMPLE_WORKLOAD_METRICS 表函数返回此监视元素，那么此监视元素表示自执行该函数以来的 CPU 时间总计。

表 1881. 表函数监视信息

表函数	监视元素收集级别
MON_FORMAT_XML_METRICS_BY_ROW - 获取所有度量值的基于已格式化行的输出	不适用; 报告 XML 文档中作为格式化函数的输入提供的所有元素
MON_GET_ACTIVITY_DETAILS 表函数 - 获取完整的活动详细信息（在 XML 文档 DETAILS 中报告）	ACTIVITY METRICS BASE
MON_GET_CONNECTION 表函数 - 获取连接度量值	REQUEST METRICS BASE
MON_GET_CONNECTION_DETAILS 表函数 - 获取详细的连接度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE
MON_GET_PKG_CACHE_STMT 表函数 - 获取程序包高速缓存中的 SQL 语句活动度量值	ACTIVITY METRICS BASE
MON_GET_PKG_CACHE_STMT_DETAILS 表函数 - 获取程序包高速缓存条目的详细度量值	ACTIVITY METRICS BASE
MON_GET_SERVICE_SUBCLASS 表函数 - 获取服务子类度量值	REQUEST METRICS BASE
MON_GET_SERVICE_SUBCLASS_DETAILS 表函数 - 获取详细的服务子类度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE
MON_GET_UNIT_OF_WORK 表函数 - 获取工作单元度量值	REQUEST METRICS BASE
MON_GET_UNIT_OF_WORK 表函数 - 获取详细的工作单元度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE
MON_GET_WORKLOAD 表函数 - 获取工作负载度量值	REQUEST METRICS BASE

表 1881. 表函数监视信息 (续)

表函数	监视元素收集级别
MON_GET_WORKLOAD_DETAILS 表函数 - 获取详细的工作负载度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE
MON_SAMPLE_SERVICE_CLASS_METRICS - 获取样本服务类度量值	REQUEST METRICS BASE
MON_SAMPLE_WORKLOAD_METRICS - 获取样本工作负载度量值	REQUEST METRICS BASE
WLM_GET_SERVICE_SUBCLASS_STATS 表函数 - 返回服务子类的统计信息	REQUEST METRICS BASE
WLM_GET_WORKLOAD_OCCURRENCE_ACTIVITIES 表函数 - 返回活动列表	REQUEST METRICS BASE
WLM_GET_WORKLOAD_STATS 表函数 - 返回工作负载统计信息	REQUEST METRICS BASE

表 1882. 事件监视信息

事件类型	逻辑数据分组	监视开关
活动	event_activity（在 details_xml 文档中报告）	ACTIVITY METRICS BASE
活动	event_activitymetrics	ACTIVITY METRICS BASE
统计信息	event_scstats（在 details_xml 文档中报告）	REQUEST METRICS BASE
统计信息	event_wlstats（在 details_xml 文档中报告）	REQUEST METRICS BASE
工作单元	在 system_metrics 文档中报告。	REQUEST METRICS BASE
程序包高速缓存	在 activity_metrics 文档中报告。	ACTIVITY METRICS BASE

用法

total_cpu_time 可与 **total_disp_run_queue_time** 监视元素一起使用，以计算 CPU 资源的争用量度量值（以 0 到 1 的标度来度量），数字越低，意味着 CPU 资源的争用量越高。此度量值（称为 CPU 速率）是通过将服务类中的工作可访问 CPU 的时间量除以访问 CPU 或等待访问 CPU 时所耗的时间总量来计算的。CPU 速率给出服务类中执行的工作的执行效率相对于可达到的执行效率（这类工作永远不需要等待 CPU 时）的度量值。公式如下所示:

```
CPU velocity = total_cpu_time / (total_cpu_time + total_disp_run_queue_time)
```

total_disp_run_queue_time -“分派器运行队列时间总计”监视元素

在此服务类中运行的请求等待访问 CPU 所耗的时间总计。此值以微秒计。

表 1883. 表函数监视信息

表函数	监视元素收集级别
MON_FORMAT_XML_METRICS_BY_ROW - 获取所有度量值的基于已格式化行的输出	不适用; 报告 XML 文档中作为格式化函数的输入提供的所有元素
MON_GET_ACTIVITY_DETAILS 表函数 - 获取完整的活动详细信息（在 XML 文档 DETAILS 中报告）	ACTIVITY METRICS BASE
MON_GET_CONNECTION 表函数 - 获取连接度量值	REQUEST METRICS BASE
MON_GET_CONNECTION_DETAILS 表函数 - 获取详细的连接度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE
MON_GET_PKG_CACHE_STMT 表函数 - 获取程序包高速缓存中的 SQL 语句活动度量值	ACTIVITY METRICS BASE
MON_GET_PKG_CACHE_STMT_DETAILS - 获取程序包高速缓存条目的详细度量值	ACTIVITY METRICS BASE
MON_GET_SERVICE_SUBCLASS 表函数 - 获取服务子类度量值	REQUEST METRICS BASE
MON_GET_SERVICE_SUBCLASS_DETAILS 表函数 - 获取详细的服务子类度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE
MON_GET_UNIT_OF_WORK 表函数 - 获取工作单元度量值	REQUEST METRICS BASE
MON_GET_UNIT_OF_WORK 表函数 - 获取详细的工作单元度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE
MON_GET_WORKLOAD 表函数 - 获取工作负载度量值	REQUEST METRICS BASE
MON_GET_WORKLOAD_DETAILS 表函数 - 获取详细的工作负载度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE
MON_SAMPLE_SERVICE_CLASS_METRICS - 获取样本服务类度量值	REQUEST METRICS BASE
MON_SAMPLE_WORKLOAD_METRICS - 获取样本工作负载度量值	REQUEST METRICS BASE
WLM_GET_SERVICE_SUBCLASS_STATS 表函数 - 返回服务子类的统计信息	REQUEST METRICS BASE
WLM_GET_WORKLOAD_STATS 表函数 - 返回工作负载统计信息	REQUEST METRICS BASE

表 1884. 事件监视信息

事件类型	逻辑数据分组	监视开关
活动	event_activity（在 details_xml 文档中报告）	ACTIVITY METRICS BASE
程序包高速缓存	在 activity_metrics 文档中报告	ACTIVITY METRICS BASE
统计信息	event_scstats（在 details_xml 文档中报告）	REQUEST METRICS BASE
统计信息	event_wlstats（在 details_xml 文档中报告）	REQUEST METRICS BASE
工作单元	在 system_metrics 文档中报告	REQUEST METRICS BASE

用法

total_disp_run_queue_time 监视元素可与 **total_cpu_time** 一起使用，以计算 CPU 资源的争用量度量值（以 0 到 1 的标度来度量），数字越低，意味着 CPU 资源的争用量越高。此度量值（称为 CPU 速率）是通过将服务类中的工作可访问 CPU 的时间量除以访问 CPU 或等待访问 CPU 时所耗的时间来计算的。它给出正在执行的工作的执行效率相对于可达到的执行效率（这类工作永远不需要等待 CPU 时）的度量值。公式如下所示:

```
CPU velocity = total_cpu_time / (total_cpu_time + total_disp_run_queue_time)
```

如果由 WLM_GET_SERVICE_SUBCLASS_STATS 或 WLM_GET_WORKLOAD_STATS 函数返回此监视元素，那么此监视元素表示自上次重置统计信息以来的分派器运行队列等待时间总计。

如果由 MON_SAMPLE_SERVICE_CLASS_METRICS 或 MON_SAMPLE_WORKLOAD_METRICS 函数返回此监视元素，那么此监视元素表示自执行该函数以来的分派器运行队列等待时间总计。

total_exec_time -“执行语句所耗用的时间”监视元素

在 SQL 高速缓存中执行特定语句所花的总时间（以秒和微秒计）。

表 1885. 快照监视信息

快照级别	逻辑数据分组	监视开关
动态 SQL	dynsql	语句

可将快照监视的计数器重置。

用法

将此元素与 **num_executions** 监视元素配合使用来确定语句的平均耗用时间并标识调整其 SQL 时受益最多的 SQL 语句。在评估此元素的内容时，必须考虑 **num_compilation** 监视元素。

注: 由于 DB2 系统收集统计信息时所使用的详细程度不同，**total_exec_time** 监视元素的值可能与 **system_cpu_time** 和 **user_cpu_time** 监视元素的值的总和不相等。在此情况下，**system_cpu_time** 与 **user_cpu_time** 监视元素的值之和更准确地反映了实际总执行时间。

total_extended_latch_wait_time -“扩展锁存器等待时间总计”监视元素

花在扩展锁存器等待上的时间（以毫秒计）。

表 1886. 表函数监视信息

表函数	监视元素收集级别
MON_FORMAT_XML_METRICS_BY_ROW - 获取所有度量值的基于已格式化行的输出	不适用; 报告 XML 文档中作为输入提供的任何元素。
MON_FORMAT_XML_TIMES_BY_ROW - - 获取基于已格式化行的组合层次结构等待时间和处理时间	不适用; 报告 XML 文档中作为输入提供的任何元素。
MON_FORMAT_XML_WAIT_TIMES_BY _ROW - 获取等待次数的已格式化的基于行的输出	不适用; 报告 XML 文档中作为输入提供的任何元素。
MON_GET_ACTIVITY_DETAILS 表函数 - 获取完整的活动详细信息（在 XML 文档 DETAILS 中报告）	ACTIVITY METRICS BASE
MON_GET_CONNECTION 表函数 - 获取连接度量值	REQUEST METRICS BASE
MON_GET_CONNECTION_DETAILS 表函数 - 获取详细的连接度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE
MON_GET_EXTENDED_LATCH_WAITS	REQUEST METRICS BASE
MON_GET_PKG_CACHE_STMT 表函数 - 获取程序包高速缓存中的 SQL 语句活动度量值	ACTIVITY METRICS BASE
MON_GET_PKG_CACHE_STMT_DETAILS 表函数 - 获取程序包高速缓存条目的详细度量值（在 XML 文档 DETAILS 中报告）	ACTIVITY METRICS BASE
MON_GET_SERVICE_SUBCLASS 表函数 - 获取服务子类度量值	REQUEST METRICS BASE
MON_GET_SERVICE_SUBCLASS_DETAILS 表函数 - 获取详细的服务子类度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE
MON_GET_UNIT_OF_WORK 表函数 - 获取工作单元度量值	REQUEST METRICS BASE
MON_GET_UNIT_OF_WORK 表函数 - 获取详细的工作单元度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE
MON_GET_WORKLOAD 表函数 - 获取工作负载度量值	REQUEST METRICS BASE
MON_GET_WORKLOAD_DETAILS 表函数 - 获取详细的工作负载度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE

表 1887. 事件监视信息

事件类型	逻辑数据分组	监视开关
活动	event_activity（在 details_xml 文档中报告）	ACTIVITY METRICS BASE
活动	event_activitymetrics	ACTIVITY METRICS BASE
统计信息	event_scstats（在 details_xml 文档中报告）	REQUEST METRICS BASE
统计信息	event_wlstats（在 details_xml 文档中报告）	REQUEST METRICS BASE
工作单元	uow（在 metrics.xml 文档中报告） uow_metrics	REQUEST METRICS BASE
程序包高速缓存	pkgcache（在 metrics.xml 文档中报告） pkgcache_metrics	ACTIVITY METRICS BASE

用法

- 使用以下公式来确定扩展锁存器等待时间占总等待时间的百分比。此公式可用于确定花在等待扩展锁存器上的时间是否太多（相对于总等待时间）。

  ```
  (TOTAL_EXTENDED_LATCH_WAIT_TIME / TOTAL_WAIT_TIME) * 100
  ```

- 使用以下公式来确定扩展锁存器等待的平均时间（以毫秒计）。

  ```
  TOTAL_EXTENDED_LATCH_WAITS / TOTAL_EXTENDED_LATCH_WAITS
  ```

total_extended_latch_waits -“扩展锁存器等待总计”监视元素

扩展锁存器等待计数。

表 1888. 表函数监视信息

表函数	监视元素收集级别
MON_FORMAT_XML_METRICS_BY_ROW - 获取所有度量值的基于已格式化行的输出	不适用; 报告 XML 文档中作为格式化函数的输入提供的所有元素
MON_GET_ACTIVITY_DETAILS 表函数 - 获取完整的活动详细信息（在 XML 文档 DETAILS 中报告）	ACTIVITY METRICS BASE
MON_GET_CONNECTION 表函数 - 获取连接度量值	REQUEST METRICS BASE
MON_GET_CONNECTION_DETAILS 表函数 - 获取详细的连接度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE
MON_GET_PKG_CACHE_STMT 表函数 - 获取程序包高速缓存中的 SQL 语句活动度量值	ACTIVITY METRICS BASE
MON_GET_PKG_CACHE_STMT_DETAILS 表函数 - 获取程序包高速缓存条目的详细度量值（在 XML 文档 DETAILS 中报告）	ACTIVITY METRICS BASE

表 1888. 表函数监视信息 (续)

表函数	监视元素收集级别
MON_GET_SERVICE_SUBCLASS 表函数 - 获取服务子类度量值	REQUEST METRICS BASE
MON_GET_SERVICE_SUBCLASS_DETAILS 表函数 - 获取详细的服务子类度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE
MON_GET_UNIT_OF_WORK 表函数 - 获取工作单元度量值	REQUEST METRICS BASE
MON_GET_UNIT_OF_WORK 表函数 - 获取详细的工作单元度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE
MON_GET_WORKLOAD 表函数 - 获取工作负载度量值	REQUEST METRICS BASE
MON_GET_WORKLOAD_DETAILS 表函数 - 获取详细的工作负载度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE
MON_GET_EXTENDED_LATCH_WAITS	REQUEST METRICS BASE

表 1889. 事件监视信息

事件类型	逻辑数据分组	监视开关
活动	event_activity（在 details_xml 文档中报告）	ACTIVITY METRICS BASE
活动	event_activitymetrics	ACTIVITY METRICS BASE
统计信息	event_scstats（在 details_xml 文档中报告）	REQUEST METRICS BASE
统计信息	event_wlstats（在 details_xml 文档中报告）	REQUEST METRICS BASE
工作单元	uow（在 metrics.xml 文档中报告） uow_metrics	REQUEST METRICS BASE
程序包高速缓存	pkgcache（在 metrics.xml 文档中报告） pkgcache_metrics	ACTIVITY METRICS BASE

用法

使用以下公式来确定扩展锁存器等待的平均时间（以毫秒计）。

```
TOTAL_EXTENDED_LATCH_WAIT_TIME / TOTAL_EXTENDED_LATCH_WAITS
```

total_move_time -“扩展数据块移动时间总计”监视元素

表空间重新平衡过程期间移动的所有扩展数据块的移动时间总计（以毫秒计）。

表 1890. 表函数监视信息

表函数	监视元素收集级别
MON_GET_EXTENT_MOVEMENT_STATUS - 获取扩展数据块移动进度状态度量值	ACTIVITY METRICS BASE

total_hash_joins -“散列连接总数”

已执行的散列连接的总数。

元素标识

total_hash_joins

元素类型

计数器

表 1891. 快照监视信息

快照级别	逻辑数据分组	监视开关
数据库	dbase	基本
应用程序	appl	基本

可将快照监视的计数器重置。

表 1892. 事件监视信息

事件类型	逻辑数据分组	监视开关
数据库	event_db	始终收集
连接	event_conn	始终收集

用法 在数据库或应用程序级别，将此值与 hash_join_overflows 和 hash_join_small_overflows 配合使用，以确定适度增大排序堆是否能够使相当百分比的散列连接受益。

total_hash_loops -“总散列循环数”

散列连接的单个分区大于可用排序堆空间的总次数。

表 1893. 快照监视信息

快照级别	逻辑数据分组	监视开关
数据库	dbase	基本
应用程序	appl	基本

可将快照监视的计数器重置。

表 1894. 事件监视信息

事件类型	逻辑数据分组	监视开关
数据库	event_db	始终收集
连接	event_conn	始终收集

用法 此元素的值指示散列连接的执行没有效率。它可能指示排序堆大小太小或者排序堆阈值太小。将此值与其他散列连接变量一起使用来调整排序堆大小（*sortheap*）和排序堆阈值（*sheapthres*）配置参数。

total_implicit_compilations -“隐式编译总数”监视元素

数据库服务器上的隐式编译的总数。隐式编译是那些不是由用户直接请求的编译。即，它们不是绑定、重新绑定、准备或执行立即请求的结果。例如，当执行一个是使用 VALIDATE RUN 选项绑定的语句时，如果需要在执行时编译该语句，就会进行隐式编译。

表 1895. 表函数监视信息

表函数	监视元素收集级别
MON_FORMAT_XML_METRICS_BY_ROW - 获取所有度量值的基于已格式化行的输出	不适用; 报告 XML 文档中作为格式化函数的输入提供的所有元素
MON_GET_CONNECTION 表函数 - 获取连接度量值	REQUEST METRICS BASE
MON_GET_CONNECTION_DETAILS 表函数 - 获取详细的连接度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE
MON_GET_SERVICE_SUBCLASS 表函数 - 获取服务子类度量值	REQUEST METRICS BASE
MON_GET_SERVICE_SUBCLASS_DETAILS 表函数 - 获取详细的服务子类度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE
MON_GET_UNIT_OF_WORK 表函数 - 获取工作单元度量值	REQUEST METRICS BASE
MON_GET_UNIT_OF_WORK 表函数 - 获取详细的工作单元度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE
MON_GET_WORKLOAD 表函数 - 获取工作负载度量值	REQUEST METRICS BASE
MON_GET_WORKLOAD_DETAILS 表函数 - 获取详细的工作负载度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE

表 1896. 事件监视信息

事件类型	逻辑数据分组	监视开关
统计信息	event_scstats（在 details_xml 文档中报告）	REQUEST METRICS BASE
统计信息	event_wlstats（在 details_xml 文档中报告）	REQUEST METRICS BASE

表 1896. 事件监视信息 (续)

事件类型	逻辑数据分组	监视开关
工作单元	在 system_metrics 文档中报告。	REQUEST METRICS BASE

total_implicit_compile_proc_time -“隐式编译处理时间总计”监视元素

在数据库服务器上执行隐式编译所花的处理（无等待）时间的总计。隐式编译是那些不是由用户直接请求的编译。即，它们不是绑定、重新绑定、准备或执行立即请求的结果。例如，当执行一个是使用 VALIDATE RUN 选项绑定的语句时，如果需要在执行时编译该语句，就会进行隐式编译。此值以毫秒计。

表 1897. 表函数监视信息

表函数	监视元素收集级别
MON_FORMAT_XML_METRICS_BY_ROW - 获取所有度量值的基于已格式化行的输出	不适用; 报告 XML 文档中作为格式化函数的输入提供的所有元素
MON_FORMAT_XML_TIMES_BY_ROW - - 获取基于已格式化行的组合层次结构等待时间和处理时间	不适用; 报告 XML 文档中作为格式化函数的输入提供的所有元素
MON_GET_CONNECTION 表函数 - 获取连接度量值	REQUEST METRICS BASE
MON_GET_CONNECTION_DETAILS 表函数 - 获取详细的连接度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE
MON_GET_SERVICE_SUBCLASS 表函数 - 获取服务子类度量值	REQUEST METRICS BASE
MON_GET_SERVICE_SUBCLASS_DETAILS 表函数 - 获取详细的服务子类度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE
MON_GET_UNIT_OF_WORK 表函数 - 获取工作单元度量值	REQUEST METRICS BASE
MON_GET_UNIT_OF_WORK 表函数 - 获取详细的工作单元度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE
MON_GET_WORKLOAD 表函数 - 获取工作负载度量值	REQUEST METRICS BASE
MON_GET_WORKLOAD_DETAILS 表函数 - 获取详细的工作负载度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE

表 1898. 事件监视信息

事件类型	逻辑数据分组	监视开关
统计信息	event_scstats（在 details_xml 文档中报告）	REQUEST METRICS BASE
统计信息	event_wlstats（在 details_xml 文档中报告）	REQUEST METRICS BASE
工作单元	在 system_metrics 文档中报告。	REQUEST METRICS BASE

total_implicit_compile_time -“隐式编译时间总计”监视元素

在数据库服务器上执行隐式编译所花的总时间。隐式编译是那些不是由用户直接请求的编译。即，它们不是绑定、重新绑定、准备或执行立即请求的结果。例如，当执行一个是使用 VALIDATE RUN 选项绑定的语句时，如果需要在执行时编译该语句，就会进行隐式编译。此值以毫秒计。

表 1899. 表函数监视信息

表函数	监视元素收集级别
MON_FORMAT_XML_COMPONENT _TIMES_BY_ROW - 获取基于已格式化行的组件次数	不适用；报告 XML 文档中作为格式化函数的输入提供的所有元素
MON_FORMAT_XML_METRICS_BY_ROW - 获取所有度量值的基于已格式化行的输出	不适用；报告 XML 文档中作为格式化函数的输入提供的所有元素
MON_GET_CONNECTION 表函数 - 获取连接度量值	REQUEST METRICS BASE
MON_GET_CONNECTION_DETAILS 表函数 - 获取详细的连接度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE
MON_GET_SERVICE_SUBCLASS 表函数 - 获取服务子类度量值	REQUEST METRICS BASE
MON_GET_SERVICE_SUBCLASS_DETAILS 表函数 - 获取详细的服务子类度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE
MON_GET_UNIT_OF_WORK 表函数 - 获取工作单元度量值	REQUEST METRICS BASE
MON_GET_UNIT_OF_WORK 表函数 - 获取详细的工作单元度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE
MON_GET_WORKLOAD 表函数 - 获取工作负载度量值	REQUEST METRICS BASE
MON_GET_WORKLOAD_DETAILS 表函数 - 获取详细的工作负载度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE

表 1900. 事件监视信息

事件类型	逻辑数据分组	监视开关
统计信息	event_scstats（在 details_xml 文档中报告）	REQUEST METRICS BASE
统计信息	event_wlstats（在 details_xml 文档中报告）	REQUEST METRICS BASE
工作单元	在 system_metrics 文档中报告。	REQUEST METRICS BASE

total_load_proc_time -“装入处理时间总计”监视元素

在数据库服务器上执行装入处理所花的处理（无等待）时间总计。此值以毫秒计。

表 1901. 表函数监视信息

表函数	监视元素收集级别
MON_FORMAT_XML_METRICS_BY_ROW - 获取所有度量值的基于已格式化行的输出	不适用; 报告 XML 文档中作为格式化函数的输入提供的所有元素
MON_FORMAT_XML_TIMES_BY_ROW - - 获取基于已格式化行的组合层次结构等待时间和处理时间	不适用; 报告 XML 文档中作为格式化函数的输入提供的所有元素
MON_GET_CONNECTION 表函数 - 获取连接度量值	REQUEST METRICS BASE
MON_GET_CONNECTION_DETAILS 表函数 - 获取详细的连接度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE
MON_GET_SERVICE_SUBCLASS 表函数 - 获取服务子类度量值	REQUEST METRICS BASE
MON_GET_SERVICE_SUBCLASS_DETAILS 表函数 - 获取详细的服务子类度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE
MON_GET_UNIT_OF_WORK 表函数 - 获取工作单元度量值	REQUEST METRICS BASE
MON_GET_UNIT_OF_WORK 表函数 - 获取详细的工作单元度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE
MON_GET_WORKLOAD 表函数 - 获取工作负载度量值	REQUEST METRICS BASE
MON_GET_WORKLOAD_DETAILS 表函数 - 获取详细的工作负载度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE

表 1902. 事件监视信息

事件类型	逻辑数据分组	监视开关
统计信息	event_scstats（在 details_xml 文档中报告）	REQUEST METRICS BASE
统计信息	event_wlstats（在 details_xml 文档中报告）	REQUEST METRICS BASE
工作单元	在 system_metrics 文档中报告。	REQUEST METRICS BASE

total_load_time -“装入时间总计”监视元素

在数据库服务器上执行装入所花的总时间。此值以毫秒计。

表 1903. 表函数监视信息

表函数	监视元素收集级别
MON_FORMAT_XML_COMPONENT _TIMES_BY_ROW - 获取基于已格式化行的组件次数	不适用; 报告 XML 文档中作为格式化函数的输入提供的所有元素
MON_FORMAT_XML_METRICS_BY_ROW - 获取所有度量值的基于已格式化行的输出	不适用; 报告 XML 文档中作为格式化函数的输入提供的所有元素
MON_GET_CONNECTION 表函数 - 获取连接度量值	REQUEST METRICS BASE
MON_GET_CONNECTION_DETAILS 表函数 - 获取详细的连接度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE
MON_GET_SERVICE_SUBCLASS 表函数 - 获取服务子类度量值	REQUEST METRICS BASE
MON_GET_SERVICE_SUBCLASS_DETAILS 表函数 - 获取详细的服务子类度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE
MON_GET_UNIT_OF_WORK 表函数 - 获取工作单元度量值	REQUEST METRICS BASE
MON_GET_UNIT_OF_WORK 表函数 - 获取详细的工作单元度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE
MON_GET_WORKLOAD 表函数 - 获取工作负载度量值	REQUEST METRICS BASE
MON_GET_WORKLOAD_DETAILS 表函数 - 获取详细的工作负载度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE

表 1904. 事件监视信息

事件类型	逻辑数据分组	监视开关
统计信息	event_scstats（在 details_xml 文档中报告）	REQUEST METRICS BASE
统计信息	event_wlstats（在 details_xml 文档中报告）	REQUEST METRICS BASE
工作单元	在 system_metrics 文档中报告。	REQUEST METRICS BASE

total_loads -“装入操作总数”监视元素

在数据库服务器上执行的装入操作的总数。

表 1905. 表函数监视信息

表函数	监视元素收集级别
MON_FORMAT_XML_METRICS_BY_ROW - 获取所有度量值的基于已格式化行的输出	不适用; 报告 XML 文档中作为格式化函数的输入提供的所有元素
MON_GET_CONNECTION 表函数 - 获取连接度量值	REQUEST METRICS BASE
MON_GET_CONNECTION_DETAILS 表函数 - 获取详细的连接度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE
MON_GET_SERVICE_SUBCLASS 表函数 - 获取服务子类度量值	REQUEST METRICS BASE
MON_GET_SERVICE_SUBCLASS_DETAILS 表函数 - 获取详细的服务子类度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE
MON_GET_UNIT_OF_WORK 表函数 - 获取工作单元度量值	REQUEST METRICS BASE
MON_GET_UNIT_OF_WORK 表函数 - 获取详细的工作单元度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE
MON_GET_WORKLOAD 表函数 - 获取工作负载度量值	REQUEST METRICS BASE
MON_GET_WORKLOAD_DETAILS 表函数 - 获取详细的工作负载度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE

表 1906. 事件监视信息

事件类型	逻辑数据分组	监视开关
统计信息	event_scstats（在 details_xml 文档中报告）	REQUEST METRICS BASE
统计信息	event_wlstats（在 details_xml 文档中报告）	REQUEST METRICS BASE
工作单元	在 system_metrics 文档中报告。	REQUEST METRICS BASE

total_log_available -“可用的总日志量”

数据库中未被未落实事务使用的活动日志空间量（以字节计）。

表 1907. 表函数监视信息

表函数	监视元素收集级别
MON_GET_TRANSACTION_LOG 表函数 - 获取日志信息	ACTIVITY METRICS BASE

表 1908. 快照监视信息

快照级别	逻辑数据分组	监视开关
数据库	dbase	基本

用法

将此元素与 total_log_used 一起使用来确定是否需要调整下列配置参数以避免用完日志空间:

- logfilsiz
- logprimary
- logsecond

如果 total_log_available 降低至 0，那么将返回 SQL0964N。您可能需要增加以上列出的配置参数，或通过 COMMIT、ROLLBACK 或 FORCE APPLICATION 结束最旧的事务。

如果 logsecond 设置为 -1，那么此元素将包含 SQLM_LOGSPACE_INFINITE。

注: 虽然数据库系统监视器信息是以字节为单位给定的，但配置参数是以页为单位设置的，每页为 4K 字节。

total_log_used -“使用的总日志空间”

当前在数据库中使用的总活动日志空间量（以字节计）。

表 1909. 表函数监视信息

表函数	监视元素收集级别
MON_GET_TRANSACTION_LOG 表函数 - 获取日志信息	ACTIVITY METRICS BASE

表 1910. 快照监视信息

快照级别	逻辑数据分组	监视开关
数据库	dbase	基本

用法 将此元素与 total_log_available 一起使用来确定是否需要调整下列配置参数以避免用完日志空间:

- logfilsiz
- logprimary
- logsecond

注: 虽然数据库系统监视器信息是以字节为单位给定的，但配置参数是以页为单位设置的，每页为 4K 字节。

total_move_time -“扩展数据块移动时间总计”监视元素

表空间重新平衡过程期间移动的所有扩展数据块的移动时间总计（以毫秒计）。

表 1911. 表函数监视信息

表函数	监视元素收集级别
MON_GET_EXTENT_MOVEMENT_STATUS - 获取扩展数据块移动进度状态度量值	ACTIVITY METRICS BASE

total_olap_funcs -“OLAP 函数总数”监视元素

执行的 OLAP 函数总数

表 1912. 快照监视信息

快照级别	逻辑数据分组	监视开关
数据库	dbase	基本
应用程序	appl	基本

可将快照监视的计数器重置。

表 1913. 事件监视信息

事件类型	逻辑数据分组	监视开关
数据库	event_db	始终收集
连接	event_conn	始终收集

用法

在数据库或应用程序级别，将此值与 olap_func_overflows 配合使用，以确定适度增大排序堆是否能够使相当百分比的 OLAP 函数受益。

total_peas -“部分提前聚集总数”监视元素

已执行部分提前聚集操作的总次数。

表 1914. 表函数监视信息

表函数	监视元素收集级别
MON_FORMAT_XML_METRICS_BY_ROW - 获取所有度量值的基于已格式化行的输出	不适用；报告 XML 文档中作为格式化函数的输入提供的所有元素
MON_GET_ACTIVITY_DETAILS 表函数 - 获取完整的活动详细信息	ACTIVITY METRICS BASE
MON_GET_CONNECTION 表函数 - 获取连接度量值	REQUEST METRICS BASE
MON_GET_CONNECTION_DETAILS 表函数 - 获取详细的连接度量值	REQUEST METRICS BASE
MON_GET_PKG_CACHE_STMT 表函数 - 获取程序包高速缓存中的 SQL 语句活动度量值	ACTIVITY METRICS BASE

表 1914. 表函数监视信息 (续)

表函数	监视元素收集级别
MON_GET_PKG_CACHE_STMT_DETAILS - 获取程序包高速缓存条目的详细度量值	ACTIVITY METRICS BASE
MON_GET_SERVICE_SUBCLASS 表函数 - 获取服务子类度量值	REQUEST METRICS BASE
MON_GET_SERVICE_SUBCLASS_DETAILS 表函数 - 获取详细的服务子类度量值	REQUEST METRICS BASE
MON_GET_UNIT_OF_WORK 表函数 - 获取工作单元度量值	REQUEST METRICS BASE
MON_GET_UNIT_OF_WORK_DETAILS 表函数 - 获取详细的工作单元度量值	REQUEST METRICS BASE
MON_GET_WORKLOAD 表函数 - 获取工作负载度量值	REQUEST METRICS BASE
MON_GET_WORKLOAD_DETAILS 表函数 - 获取详细的工作负载度量值	REQUEST METRICS BASE

表 1915. 表函数监视信息

表函数	监视元素收集级别
MON_FORMAT_XML_METRICS_BY_ROW - 获取所有度量值的基于已格式化行的输出	不适用; 报告 XML 文档中作为格式化函数的输入提供的所有元素
MON_GET_ACTIVITY_DETAILS 表函数 - 获取完整的活动详细信息（在 XML 文档 DETAILS 中报告）	ACTIVITY METRICS BASE
MON_GET_CONNECTION 表函数 - 获取连接度量值	REQUEST METRICS BASE
MON_GET_CONNECTION_DETAILS 表函数 - 获取详细的连接度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE
MON_GET_PKG_CACHE_STMT 表函数 - 获取程序包高速缓存中的 SQL 语句活动度量值	ACTIVITY METRICS BASE
MON_GET_PKG_CACHE_STMT_DETAILS 表函数 - 获取程序包高速缓存条目的详细度量值（在 XML 文档 DETAILS 中报告）	ACTIVITY METRICS BASE
MON_GET_SERVICE_SUBCLASS 表函数 - 获取服务子类度量值	REQUEST METRICS BASE
MON_GET_SERVICE_SUBCLASS_DETAILS 表函数 - 获取详细的服务子类度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE
MON_GET_UNIT_OF_WORK 表函数 - 获取工作单元度量值	REQUEST METRICS BASE
MON_GET_UNIT_OF_WORK 表函数 - 获取详细的工作单元度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE
MON_GET_WORKLOAD 表函数 - 获取工作负载度量值	REQUEST METRICS BASE

表 1915. 表函数监视信息 (续)

表函数	监视元素收集级别
MON_GET_WORKLOAD_DETAILS 表函数 - 获取详细的工作负载度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE

表 1916. 事件监视信息

事件类型	逻辑数据分组	监视开关
活动	event_activity（在 details_xml 文档中报告）	ACTIVITY METRICS BASE
统计信息	event_scstats（在 details_xml 文档中报告）	REQUEST METRICS BASE
统计信息	event_wlstats（在 details_xml 文档中报告）	REQUEST METRICS BASE
工作单元	在 system_metrics 文档中报告。	REQUEST METRICS BASE
连接	event_conn	-
语句	event_stmt	-
事务	event_xact	-
程序包高速缓存	在 activity_metrics 文档中报告。	ACTIVITY METRICS BASE

用法

在数据库级别或应用程序级别，将此值与 **post_threshold_peas** 配合使用以确定大量部分提前聚集操作是否受益于排序堆大小或排序堆阈值的增加。如果 **post_threshold_peas** 与 **total_peas** 的比率过高，那么增大排序堆大小和/或排序堆阈值可能会提高数据库或应用程序的性能。

total_peds -“部分提前相异总数”监视元素

已执行部分提前相异操作的总次数。

表 1917. 表函数监视信息

表函数	监视元素收集级别
MON_FORMAT_XML_METRICS_BY_ROW - 获取所有度量值的基于已格式化行的输出	不适用; 报告 XML 文档中作为格式化函数的输入提供的所有元素
MON_GET_ACTIVITY_DETAILS 表函数 - 获取完整的活动详细信息（在 XML 文档 DETAILS 中报告）	ACTIVITY METRICS BASE
MON_GET_CONNECTION 表函数 - 获取连接度量值	REQUEST METRICS BASE
MON_GET_CONNECTION_DETAILS 表函数 - 获取详细的连接度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE
MON_GET_PKG_CACHE_STMT 表函数 - 获取程序包高速缓存中的 SQL 语句活动度量值	ACTIVITY METRICS BASE

表 1917. 表函数监视信息 (续)

表函数	监视元素收集级别
MON_GET_PKG_CACHE_STMT_DETAILS 表函数 - 获取程序包高速缓存条目的详细度量值（在 XML 文档 DETAILS 中报告）	ACTIVITY METRICS BASE
MON_GET_SERVICE_SUBCLASS 表函数 - 获取服务子类度量值	REQUEST METRICS BASE
MON_GET_SERVICE_SUBCLASS_DETAILS 表函数 - 获取详细的服务子类度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE
MON_GET_UNIT_OF_WORK 表函数 - 获取工作单元度量值	REQUEST METRICS BASE
MON_GET_UNIT_OF_WORK 表函数 - 获取详细的工作单元度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE
MON_GET_WORKLOAD 表函数 - 获取工作负载度量值	REQUEST METRICS BASE
MON_GET_WORKLOAD_DETAILS 表函数 - 获取详细的工作负载度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE

表 1918. 事件监视信息

事件类型	逻辑数据分组	监视开关
活动	event_activity（在 details_xml 文档中报告）	ACTIVITY METRICS BASE
统计信息	event_scstats（在 details_xml 文档中报告）	REQUEST METRICS BASE
统计信息	event_wlstats（在 details_xml 文档中报告）	REQUEST METRICS BASE
工作单元	在 system_metrics 文档中报告。	REQUEST METRICS BASE
连接	event_conn	-
语句	event_stmt	-
事务	event_xact	-
程序包高速缓存	在 activity_metrics 文档中报告。	ACTIVITY METRICS BASE

用法

在数据库级别或应用程序级别，将此值与 **disabled_peds** 监视元素和 **post_threshold_peds** 监视元素配合使用以确定大量部分提前相异操作是否受益于排序堆大小或排序堆阈值的增加。如果 **disabled_peds** 监视元素和 **post_threshold_peds** 监视元素与 **total_peds** 监视元素的比率很高，那么增加排序堆大小和/或排序堆阈值可能会改进数据库或应用程序的性能。

total_reorg_proc_time -“重组处理时间总计”监视元素

在数据库服务器上执行重组操作所花的处理（无等待）时间总计。此值以毫秒计。

表 1919. 表函数监视信息

表函数	监视元素收集级别
MON_FORMAT_XML_METRICS_BY_ROW - 获取所有度量值的基于已格式化行的输出	不适用; 报告 XML 文档中作为格式化函数的输入提供的所有元素
MON_FORMAT_XML_TIMES_BY_ROW - - 获取基于已格式化行的组合层次结构等待时间和处理时间	不适用; 报告 XML 文档中作为格式化函数的输入提供的所有元素
MON_GET_CONNECTION 表函数 - 获取连接度量值	REQUEST METRICS BASE
MON_GET_CONNECTION_DETAILS 表函数 - 获取详细的连接度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE
MON_GET_SERVICE_SUBCLASS 表函数 - 获取服务子类度量值	REQUEST METRICS BASE
MON_GET_SERVICE_SUBCLASS_DETAILS 表函数 - 获取详细的服务子类度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE
MON_GET_UNIT_OF_WORK 表函数 - 获取工作单元度量值	REQUEST METRICS BASE
MON_GET_UNIT_OF_WORK 表函数 - 获取详细的工作单元度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE
MON_GET_WORKLOAD 表函数 - 获取工作负载度量值	REQUEST METRICS BASE
MON_GET_WORKLOAD_DETAILS 表函数 - 获取详细的工作负载度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE

表 1920. 事件监视信息

事件类型	逻辑数据分组	监视开关
统计信息	event_scstats（在 details_xml 文档中报告）	REQUEST METRICS BASE
统计信息	event_wlstats（在 details_xml 文档中报告）	REQUEST METRICS BASE
工作单元	在 system_metrics 文档中报告。	REQUEST METRICS BASE

total_reorg_time -“重组时间总计”监视元素

在数据库服务器上执行重组操作所花的总时间。此值以毫秒计。

表 1921. 表函数监视信息

表函数	监视元素收集级别
MON_FORMAT_XML_COMPONENT _TIMES_BY_ROW - 获取基于已格式化行的组件次数	不适用; 报告 XML 文档中作为格式化函数的输入提供的所有元素
MON_FORMAT_XML_METRICS_BY_ROW - 获取所有度量值的基于已格式化行的输出	不适用; 报告 XML 文档中作为格式化函数的输入提供的所有元素
MON_GET_CONNECTION 表函数 - 获取连接度量值	REQUEST METRICS BASE
MON_GET_CONNECTION_DETAILS 表函数 - 获取详细的连接度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE
MON_GET_SERVICE_SUBCLASS 表函数 - 获取服务子类度量值	REQUEST METRICS BASE
MON_GET_SERVICE_SUBCLASS_DETAILS 表函数 - 获取详细的服务子类度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE
MON_GET_UNIT_OF_WORK 表函数 - 获取工作单元度量值	REQUEST METRICS BASE
MON_GET_UNIT_OF_WORK 表函数 - 获取详细的工作单元度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE
MON_GET_WORKLOAD 表函数 - 获取工作负载度量值	REQUEST METRICS BASE
MON_GET_WORKLOAD_DETAILS 表函数 - 获取详细的工作负载度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE

表 1922. 事件监视信息

事件类型	逻辑数据分组	监视开关
统计信息	event_scstats（在 details_xml 文档中报告）	REQUEST METRICS BASE
统计信息	event_wlstats（在 details_xml 文档中报告）	REQUEST METRICS BASE
工作单元	在 system_metrics 文档中报告。	REQUEST METRICS BASE

total_reorgs -“重组操作总数”监视元素

对数据库服务器发出的重组操作总数。

表 1923. 表函数监视信息

表函数	监视元素收集级别
MON_FORMAT_XML_METRICS_BY_ROW - 获取所有度量值的基于已格式化行的输出	不适用; 报告 XML 文档中作为格式化函数的输入提供的所有元素
MON_GET_CONNECTION 表函数 - 获取连接度量值	REQUEST METRICS BASE
MON_GET_CONNECTION_DETAILS 表函数 - 获取详细的连接度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE
MON_GET_SERVICE_SUBCLASS 表函数 - 获取服务子类度量值	REQUEST METRICS BASE
MON_GET_SERVICE_SUBCLASS_DETAILS 表函数 - 获取详细的服务子类度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE
MON_GET_UNIT_OF_WORK 表函数 - 获取工作单元度量值	REQUEST METRICS BASE
MON_GET_UNIT_OF_WORK 表函数 - 获取详细的工作单元度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE
MON_GET_WORKLOAD 表函数 - 获取工作负载度量值	REQUEST METRICS BASE
MON_GET_WORKLOAD_DETAILS 表函数 - 获取详细的工作负载度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE

表 1924. 事件监视信息

事件类型	逻辑数据分组	监视开关
统计信息	event_scstats（在 details_xml 文档中报告）	REQUEST METRICS BASE
统计信息	event_wlstats（在 details_xml 文档中报告）	REQUEST METRICS BASE
工作单元	在 system_metrics 文档中报告。	REQUEST METRICS BASE

total_rollback_proc_time -“回滚处理时间总计”监视元素

在数据库服务器上执行回滚操作所花的处理（无等待）时间总计。此值以毫秒计。

表 1925. 表函数监视信息

表函数	监视元素收集级别
MON_FORMAT_XML_METRICS_BY_ROW - 获取所有度量值的基于已格式化行的输出	不适用; 报告 XML 文档中作为格式化函数的输入提供的所有元素

表 1925. 表函数监视信息 (续)

表函数	监视元素收集级别
MON_FORMAT_XML_TIMES_BY_ROW - - 获取基于已格式化行的组合层次结构等待时间和处理时间	不适用; 报告 XML 文档中作为格式化函数的输入提供的所有元素
MON_GET_CONNECTION 表函数 - 获取连接度量值	REQUEST METRICS BASE
MON_GET_CONNECTION_DETAILS 表函数 - 获取详细的连接度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE
MON_GET_SERVICE_SUBCLASS 表函数 - 获取服务子类度量值	REQUEST METRICS BASE
MON_GET_SERVICE_SUBCLASS_DETAILS 表函数 - 获取详细的服务子类度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE
MON_GET_UNIT_OF_WORK 表函数 - 获取工作单元度量值	REQUEST METRICS BASE
MON_GET_UNIT_OF_WORK 表函数 - 获取详细的工作单元度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE
MON_GET_WORKLOAD 表函数 - 获取工作负载度量值	REQUEST METRICS BASE
MON_GET_WORKLOAD_DETAILS 表函数 - 获取详细的工作负载度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE

表 1926. 事件监视信息

事件类型	逻辑数据分组	监视开关
统计信息	event_scstats（在 details_xml 文档中报告）	REQUEST METRICS BASE
统计信息	event_wlstats（在 details_xml 文档中报告）	REQUEST METRICS BASE
工作单元	在 system_metrics 文档中报告。	REQUEST METRICS BASE

total_rollback_time -“回滚时间总计”监视元素

在数据库服务器上执行回滚操作所花的总时间。此值以毫秒计。

表 1927. 表函数监视信息

表函数	监视元素收集级别
MON_FORMAT_XML_COMPONENT _TIMES_BY_ROW - 获取基于已格式化行的组件次数	不适用; 报告 XML 文档中作为格式化函数的输入提供的所有元素
MON_FORMAT_XML_METRICS_BY_ROW - 获取所有度量值的基于已格式化行的输出	不适用; 报告 XML 文档中作为格式化函数的输入提供的所有元素

表 1927. 表函数监视信息 (续)

表函数	监视元素收集级别
MON_GET_CONNECTION 表函数 - 获取连接度量值	REQUEST METRICS BASE
MON_GET_CONNECTION_DETAILS 表函数 - 获取详细的连接度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE
MON_GET_SERVICE_SUBCLASS 表函数 - 获取服务子类度量值	REQUEST METRICS BASE
MON_GET_SERVICE_SUBCLASS_DETAILS 表函数 - 获取详细的服务子类度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE
MON_GET_UNIT_OF_WORK 表函数 - 获取工作单元度量值	REQUEST METRICS BASE
MON_GET_UNIT_OF_WORK 表函数 - 获取详细的工作单元度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE
MON_GET_WORKLOAD 表函数 - 获取工作负载度量值	REQUEST METRICS BASE
MON_GET_WORKLOAD_DETAILS 表函数 - 获取详细的工作负载度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE

表 1928. 事件监视信息

事件类型	逻辑数据分组	监视开关
统计信息	event_scstats（在 details_xml 文档中报告）	REQUEST METRICS BASE
统计信息	event_wlstats（在 details_xml 文档中报告）	REQUEST METRICS BASE
工作单元	在 system_metrics 文档中报告。	REQUEST METRICS BASE

total_routine_invocations -“例程调用总计”监视元素

调用例程的总次数。

表 1929. 表函数监视信息

表函数	监视元素收集级别
MON_FORMAT_XML_METRICS_BY_ROW - 获取所有度量值的基于已格式化行的输出	不适用; 报告 XML 文档中作为格式化函数的输入提供的所有元素
MON_GET_ACTIVITY_DETAILS 表函数 - 获取完整的活动详细信息（在 XML 文档 DETAILS 中报告）	ACTIVITY METRICS BASE
MON_GET_CONNECTION 表函数 - 获取连接度量值	REQUEST METRICS BASE

表 1929. 表函数监视信息 (续)

表函数	监视元素收集级别
MON_GET_CONNECTION_DETAILS 表函数 - 获取详细的连接度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE
MON_GET_PKG_CACHE_STMT 表函数 - 获取程序包高速缓存中的 SQL 语句活动度量值	ACTIVITY METRICS BASE
MON_GET_PKG_CACHE_STMT_DETAILS 表函数 - 获取程序包高速缓存条目的详细度量值	ACTIVITY METRICS BASE
MON_GET_SERVICE_SUBCLASS 表函数 - 获取服务子类度量值	REQUEST METRICS BASE
MON_GET_SERVICE_SUBCLASS_DETAILS 表函数 - 获取详细的服务子类度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE
MON_GET_UNIT_OF_WORK 表函数 - 获取工作单元度量值	REQUEST METRICS BASE
MON_GET_UNIT_OF_WORK 表函数 - 获取详细的工作单元度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE
MON_GET_WORKLOAD 表函数 - 获取工作负载度量值	REQUEST METRICS BASE
MON_GET_WORKLOAD_DETAILS 表函数 - 获取详细的工作负载度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE

表 1930. 事件监视信息

事件类型	逻辑数据分组	监视开关
活动	event_activity（在 details_xml 文档中报告）	ACTIVITY METRICS BASE
活动	event_activitymetrics	ACTIVITY METRICS BASE
统计信息	event_scstats（在 details_xml 文档中报告）	REQUEST METRICS BASE
统计信息	event_wlstats（在 details_xml 文档中报告）	REQUEST METRICS BASE
程序包高速缓存	在 activity_metrics 文档中报告。	ACTIVITY METRICS BASE
工作单元	在 system_metrics 文档中报告。	REQUEST METRICS BASE

total_routine_non_sect_proc_time -“非部分处理时间”监视元素

此语句用于执行例程中的非部分执行的处理时间总计。此值同时包括执行例程中的用户代码所花的时间以及执行非部分操作（例如，落实或回滚）所花的时间。处理时间不包括等待时间。此值以毫秒计。

表 1931. 表函数监视信息

表函数	监视元素收集级别
MON_FORMAT_XML_METRICS_BY_ROW - 获取所有度量值的基于已格式化行的输出	不适用; 报告 XML 文档中作为格式化函数的输入提供的所有元素
MON_FORMAT_XML_TIMES_BY_ROW - - 获取基于已格式化行的组合层次结构等待时间和处理时间	不适用; 报告 XML 文档中作为格式化函数的输入提供的所有元素
MON_GET_ACTIVITY_DETAILS 表函数 - 获取完整的活动详细信息（在 XML 文档 DETAILS 中报告）	ACTIVITY METRICS BASE
MON_GET_PKG_CACHE_STMT 表函数 - 获取程序包高速缓存中的 SQL 语句活动度量值	ACTIVITY METRICS BASE
MON_GET_PKG_CACHE_STMT_DETAILS 表函数 - 获取程序包高速缓存条目的详细度量值	ACTIVITY METRICS BASE

表 1932. 事件监视信息

事件类型	逻辑数据分组	监视开关
活动	event_activity（在 details_xml 文档中报告）	ACTIVITY METRICS BASE
活动	event_activitymetrics	ACTIVITY METRICS BASE
程序包高速缓存	在 activity_metrics 文档中报告。	ACTIVITY METRICS BASE

total_routine_non_sect_time -“非部分例程执行时间”监视元素

此语句用于执行例程中的非部分执行的时间总计。此值同时包括执行例程中的用户代码所花的时间以及执行非部分操作（例如，落实或回滚）所花的时间。此值以毫秒计。

表 1933. 表函数监视信息

表函数	监视元素收集级别
MON_FORMAT_XML_COMPONENT _TIMES_BY_ROW - 获取基于已格式化行的组件次数	不适用; 报告 XML 文档中作为格式化函数的输入提供的所有元素
MON_FORMAT_XML_METRICS_BY_ROW - 获取所有度量值的基于已格式化行的输出	不适用; 报告 XML 文档中作为格式化函数的输入提供的所有元素
MON_GET_ACTIVITY_DETAILS 表函数 - 获取完整的活动详细信息（在 XML 文档 DETAILS 中报告）	ACTIVITY METRICS BASE
MON_GET_PKG_CACHE_STMT 表函数 - 获取程序包高速缓存中的 SQL 语句活动度量值	ACTIVITY METRICS BASE

表 1933. 表函数监视信息 (续)

表函数	监视元素收集级别
MON_GET_PKG_CACHE_STMT_DETAILS 表函数 - 获取程序包高速缓存条目的详细度量值	ACTIVITY METRICS BASE

表 1934. 事件监视信息

事件类型	逻辑数据分组	监视开关
活动	event_activity（在 details_xml 文档中报告）	ACTIVITY METRICS BASE
活动	event_activitymetrics	ACTIVITY METRICS BASE
程序包高速缓存	在 activity_metrics 文档中报告。	ACTIVITY METRICS BASE

total_routine_time -“例程时间总计”监视元素

执行例程所耗用的时间总计。此值以毫秒计。

表 1935. 表函数监视信息

表函数	监视元素收集级别
MON_FORMAT_XML_COMPONENT _TIMES_BY_ROW - 获取基于已格式化行的组件次数	不适用; 报告 XML 文档中作为格式化函数的输入提供的所有元素
MON_FORMAT_XML_METRICS_BY_ROW - 获取所有度量值的基于已格式化行的输出	不适用; 报告 XML 文档中作为格式化函数的输入提供的所有元素
MON_GET_ACTIVITY_DETAILS 表函数 - 获取完整的活动详细信息（在 XML 文档 DETAILS 中报告）	ACTIVITY METRICS BASE
MON_GET_CONNECTION 表函数 - 获取连接度量值	REQUEST METRICS BASE
MON_GET_CONNECTION_DETAILS 表函数 - 获取详细的连接度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE
MON_GET_PKG_CACHE_STMT 表函数 - 获取程序包高速缓存中的 SQL 语句活动度量值	ACTIVITY METRICS BASE
MON_GET_PKG_CACHE_STMT_DETAILS 表函数 - 获取程序包高速缓存条目的详细度量值	ACTIVITY METRICS BASE
MON_GET_SERVICE_SUBCLASS 表函数 - 获取服务子类度量值	REQUEST METRICS BASE
MON_GET_SERVICE_SUBCLASS_DETAILS 表函数 - 获取详细的服务子类度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE
MON_GET_UNIT_OF_WORK 表函数 - 获取工作单元度量值	REQUEST METRICS BASE
MON_GET_UNIT_OF_WORK 表函数 - 获取详细的工作单元度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE

表 1935. 表函数监视信息 (续)

表函数	监视元素收集级别
MON_GET_WORKLOAD 表函数 - 获取工作负载度量值	REQUEST METRICS BASE
MON_GET_WORKLOAD_DETAILS 表函数 - 获取详细的工作负载度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE

表 1936. 事件监视信息

事件类型	逻辑数据分组	监视开关
活动	event_activity（在 details_xml 文档中报告）	ACTIVITY METRICS BASE
活动	event_activitymetrics	ACTIVITY METRICS BASE
统计信息	event_scstats（在 details_xml 文档中报告）	REQUEST METRICS BASE
统计信息	event_wlstats（在 details_xml 文档中报告）	REQUEST METRICS BASE
程序包高速缓存	在 activity_metrics 文档中报告。	ACTIVITY METRICS BASE
工作单元	在 system_metrics 文档中报告。	REQUEST METRICS BASE

用法

如果收集级别设置为 BASE，那么 **total_routine_time** 监视元素的值不包含执行使用 NO SQL 子句定义的函数所耗用的时间。

如果收集级别设置为 EXTENDED，那么 **total_routine_time** 监视元素的值包含所有例程中所耗用的时间。

total_routine_user_code_proc_time -“例程用户代码处理时间总计”监视元素

除了已知 DB2 时间之外，在例程中执行所耗用的处理时间总计（通常为例程中的用户代码）。此值以毫秒计。

表 1937. 表函数监视信息

表函数	监视元素收集级别
MON_FORMAT_XML_METRICS_BY_ROW - 获取所有度量值的基于已格式化行的输出	不适用；报告 XML 文档中作为格式化函数的输入提供的所有元素
MON_FORMAT_XML_TIMES_BY_ROW - - 获取基于已格式化行的组合层次结构等待时间和处理时间	不适用；报告 XML 文档中作为格式化函数的输入提供的所有元素
MON_GET_ACTIVITY_DETAILS 表函数 - 获取完整的活动详细信息（在 XML 文档 DETAILS 中报告）	ACTIVITY METRICS BASE
MON_GET_CONNECTION 表函数 - 获取连接度量值	REQUEST METRICS BASE

表 1937. 表函数监视信息 (续)

表函数	监视元素收集级别
MON_GET_CONNECTION_DETAILS 表函数 - 获取详细的连接度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE
MON_GET_PKG_CACHE_STMT 表函数 - 获取程序包高速缓存中的 SQL 语句活动度量值	ACTIVITY METRICS BASE
MON_GET_PKG_CACHE_STMT_DETAILS 表函数 - 获取程序包高速缓存条目的详细度量值	ACTIVITY METRICS BASE
MON_GET_SERVICE_SUBCLASS 表函数 - 获取服务子类度量值	REQUEST METRICS BASE
MON_GET_SERVICE_SUBCLASS_DETAILS 表函数 - 获取详细的服务子类度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE
MON_GET_UNIT_OF_WORK 表函数 - 获取工作单元度量值	REQUEST METRICS BASE
MON_GET_UNIT_OF_WORK 表函数 - 获取详细的工作单元度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE
MON_GET_WORKLOAD 表函数 - 获取工作负载度量值	REQUEST METRICS BASE
MON_GET_WORKLOAD_DETAILS 表函数 - 获取详细的工作负载度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE

表 1938. 事件监视信息

事件类型	逻辑数据分组	监视开关
活动	event_activity（在 details_xml 文档中报告）	ACTIVITY METRICS BASE
活动	event_activitymetrics	ACTIVITY METRICS BASE
统计信息	event_scstats（在 details_xml 文档中报告）	REQUEST METRICS BASE
统计信息	event_wlstats（在 details_xml 文档中报告）	REQUEST METRICS BASE
程序包高速缓存	在 activity_metrics 文档中报告。	ACTIVITY METRICS BASE
工作单元	在 system_metrics 文档中报告。	REQUEST METRICS BASE

用法

如果收集级别设置为 BASE，那么此监视元素不包含执行使用 NO SQL 子句定义的函数所耗用的任何处理时间。此时间而是包含在 **total_section_proc_time** 监视元素的值中。

如果收集级别设置为 EXTENDED，那么此监视元素的值包含执行所有例程所耗用的处理时间。

total_routine_user_code_time -“例程用户代码时间总计”监视元素

除了已知 DB2 时间之外，在例程中执行所耗用的时间总计（通常为例程中的用户代码）。此值以毫秒计。

表 1939. 表函数监视信息

表函数	监视元素收集级别
MON_FORMAT_XML_COMPONENT _TIMES_BY_ROW - 获取基于已格式化行的组件次数	不适用; 报告 XML 文档中作为格式化函数的输入提供的所有元素
MON_FORMAT_XML_METRICS_BY_ROW - 获取所有度量值的基于已格式化行的输出	不适用; 报告 XML 文档中作为格式化函数的输入提供的所有元素
MON_GET_ACTIVITY_DETAILS 表函数 - 获取完整的活动详细信息（在 XML 文档 DETAILS 中报告）	ACTIVITY METRICS BASE
MON_GET_CONNECTION 表函数 - 获取连接度量值	REQUEST METRICS BASE
MON_GET_CONNECTION_DETAILS 表函数 - 获取详细的连接度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE
MON_GET_PKG_CACHE_STMT 表函数 - 获取程序包高速缓存中的 SQL 语句活动度量值	ACTIVITY METRICS BASE
MON_GET_PKG_CACHE_STMT_DETAILS 表函数 - 获取程序包高速缓存条目的详细度量值	ACTIVITY METRICS BASE
MON_GET_SERVICE_SUBCLASS 表函数 - 获取服务子类度量值	REQUEST METRICS BASE
MON_GET_SERVICE_SUBCLASS_DETAILS 表函数 - 获取详细的服务子类度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE
MON_GET_UNIT_OF_WORK 表函数 - 获取工作单元度量值	REQUEST METRICS BASE
MON_GET_UNIT_OF_WORK 表函数 - 获取详细的工作单元度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE
MON_GET_WORKLOAD 表函数 - 获取工作负载度量值	REQUEST METRICS BASE
MON_GET_WORKLOAD_DETAILS 表函数 - 获取详细的工作负载度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE

表 1940. 事件监视信息

事件类型	逻辑数据分组	监视开关
活动	event_activity（在 details_xml 文档中报告）	ACTIVITY METRICS BASE
活动	event_activitymetrics	ACTIVITY METRICS BASE
统计信息	event_scstats（在 details_xml 文档中报告）	REQUEST METRICS BASE

表 1940. 事件监视信息 (续)

事件类型	逻辑数据分组	监视开关
统计信息	event_wlstats（在 details_xml 文档中报告）	REQUEST METRICS BASE
程序包高速缓存	在 activity_metrics 文档中报告。	ACTIVITY METRICS BASE
工作单元	在 system_metrics 文档中报告。	REQUEST METRICS BASE

用法

如果收集级别设置为 BASE，那么此监视元素的值不包含执行使用 NO SQL 子句定义的函数所耗用的时间。此时间而是包含在 **total_section_time** 监视元素的值中。

如果收集级别设置为 EXTENDED，那么此监视元素的值包含执行所有例程所耗用的时间。

total_rqst_mapped_in -“映入请求总数”监视元素

通过重新映射阈值或工作操作集映射到此服务子类中的请求的总数。

表 1941. 表函数监视信息

表函数	监视元素收集级别
MON_GET_SERVICE_SUBCLASS 表函数 - 获取服务子类度量值	REQUEST METRICS BASE
MON_GET_SERVICE_SUBCLASS_DETAILS 表函数 - 获取详细的服务子类度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE

表 1942. 事件监视信息

事件类型	逻辑数据分组	监视开关
统计信息	event_scstats（在 details_xml 文档中报告）	REQUEST METRICS BASE

total_rqst_mapped_out -“映出请求总数”监视元素

通过重新映射阈值或工作操作集从此服务子类中映射出的请求的总数。

表 1943. 表函数监视信息

表函数	监视元素收集级别
MON_GET_SERVICE_SUBCLASS 表函数 - 获取服务子类度量值	REQUEST METRICS BASE
MON_GET_SERVICE_SUBCLASS_DETAILS 表函数 - 获取详细的服务子类度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE

表 1944. 事件监视信息

事件类型	逻辑数据分组	监视开关
统计信息	event_scstats（在 details_xml 文档中报告）	REQUEST METRICS BASE

total_rqst_time -“请求时间总计”监视元素

处理请求时的耗用时间总计。此值以毫秒计。

表 1945. 表函数监视信息

表函数	监视元素收集级别
MON_FORMAT_XML_COMPONENT _TIMES_BY_ROW - 获取基于已格式化行的组件次数	不适用; 报告 XML 文档中作为格式化函数的输入提供的所有元素
MON_FORMAT_XML_METRICS_BY_ROW - 获取所有度量值的基于已格式化行的输出	不适用; 报告 XML 文档中作为格式化函数的输入提供的所有元素
MON_FORMAT_XML_TIMES_BY_ROW - - 获取基于已格式化行的组合层次结构等待时间和处理时间	不适用; 报告 XML 文档中作为格式化函数的输入提供的所有元素
MON_GET_CONNECTION 表函数 - 获取连接度量值	REQUEST METRICS BASE
MON_GET_CONNECTION_DETAILS 表函数 - 获取详细的连接度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE
MON_GET_SERVICE_SUBCLASS 表函数 - 获取服务子类度量值	REQUEST METRICS BASE
MON_GET_SERVICE_SUBCLASS_DETAILS 表函数 - 获取详细的服务子类度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE
MON_GET_UNIT_OF_WORK 表函数 - 获取工作单元度量值	REQUEST METRICS BASE
MON_GET_UNIT_OF_WORK 表函数 - 获取详细的工作单元度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE
MON_GET_WORKLOAD 表函数 - 获取工作负载度量值	REQUEST METRICS BASE
MON_GET_WORKLOAD_DETAILS 表函数 - 获取详细的工作负载度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE

表 1946. 事件监视信息

事件类型	逻辑数据分组	监视开关
统计信息	event_scstats（在 details_xml 文档中报告）	REQUEST METRICS BASE
统计信息	event_wlstats（在 details_xml 文档中报告）	REQUEST METRICS BASE

表 1946. 事件监视信息 (续)

事件类型	逻辑数据分组	监视开关
工作单元	在 system_metrics 文档中报告。	REQUEST METRICS BASE

total_runstats -“运行时统计信息总计”监视元素

在数据库服务器上执行的 runstats 操作的总数。

表 1947. 表函数监视信息

表函数	监视元素收集级别
MON_FORMAT_XML_METRICS_BY_ROW - 获取所有度量值的基于已格式化行的输出	不适用; 报告 XML 文档中作为格式化函数的输入提供的所有元素
MON_GET_CONNECTION 表函数 - 获取连接度量值	REQUEST METRICS BASE
MON_GET_CONNECTION_DETAILS 表函数 - 获取详细的连接度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE
MON_GET_SERVICE_SUBCLASS 表函数 - 获取服务子类度量值	REQUEST METRICS BASE
MON_GET_SERVICE_SUBCLASS_DETAILS 表函数 - 获取详细的服务子类度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE
MON_GET_UNIT_OF_WORK 表函数 - 获取工作单元度量值	REQUEST METRICS BASE
MON_GET_UNIT_OF_WORK 表函数 - 获取详细的工作单元度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE
MON_GET_WORKLOAD 表函数 - 获取工作负载度量值	REQUEST METRICS BASE
MON_GET_WORKLOAD_DETAILS 表函数 - 获取详细的工作负载度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE

表 1948. 事件监视信息

事件类型	逻辑数据分组	监视开关
统计信息	event_scstats（在 details_xml 文档中报告）	REQUEST METRICS BASE
统计信息	event_wlstats（在 details_xml 文档中报告）	REQUEST METRICS BASE
工作单元	在 system_metrics 文档中报告。	REQUEST METRICS BASE

total_runstats_proc_time -“运行时统计信息处理时间总计”监视元素

在数据库服务器上执行 runstats 操作所花的处理（无等待）时间总计。此值以毫秒计。runstats 实用程序调节速度所花的时间都不会计入 runstats 处理时间。

表 1949. 表函数监视信息

表函数	监视元素收集级别
MON_FORMAT_XML_METRICS_BY_ROW - 获取所有度量值的基于已格式化行的输出	不适用; 报告 XML 文档中作为格式化函数的输入提供的所有元素
MON_FORMAT_XML_TIMES_BY_ROW - - 获取基于已格式化行的组合层次结构等待时间和处理时间	不适用; 报告 XML 文档中作为格式化函数的输入提供的所有元素
MON_GET_CONNECTION 表函数 - 获取连接度量值	REQUEST METRICS BASE
MON_GET_CONNECTION_DETAILS 表函数 - 获取详细的连接度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE
MON_GET_SERVICE_SUBCLASS 表函数 - 获取服务子类度量值	REQUEST METRICS BASE
MON_GET_SERVICE_SUBCLASS_DETAILS 表函数 - 获取详细的服务子类度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE
MON_GET_UNIT_OF_WORK 表函数 - 获取工作单元度量值	REQUEST METRICS BASE
MON_GET_UNIT_OF_WORK 表函数 - 获取详细的工作单元度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE
MON_GET_WORKLOAD 表函数 - 获取工作负载度量值	REQUEST METRICS BASE
MON_GET_WORKLOAD_DETAILS 表函数 - 获取详细的工作负载度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE

表 1950. 事件监视信息

事件类型	逻辑数据分组	监视开关
统计信息	event_scstats（在 details_xml 文档中报告）	REQUEST METRICS BASE
统计信息	event_wlstats（在 details_xml 文档中报告）	REQUEST METRICS BASE
工作单元	在 system_metrics 文档中报告。	REQUEST METRICS BASE

total_runstats_time -“运行时统计信息时间总计”监视元素

在数据库服务器上执行 runstats 操作所花的总时间。此值以毫秒计。

表 1951. 表函数监视信息

表函数	监视元素收集级别
MON_FORMAT_XML_COMPONENT _TIMES_BY_ROW - 获取基于已格式化行的组件次数	不适用; 报告 XML 文档中作为格式化函数的输入提供的所有元素
MON_FORMAT_XML_METRICS_BY_ROW - 获取所有度量值的基于已格式化行的输出	不适用; 报告 XML 文档中作为格式化函数的输入提供的所有元素
MON_GET_CONNECTION 表函数 - 获取连接度量值	REQUEST METRICS BASE
MON_GET_CONNECTION_DETAILS 表函数 - 获取详细的连接度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE
MON_GET_SERVICE_SUBCLASS 表函数 - 获取服务子类度量值	REQUEST METRICS BASE
MON_GET_SERVICE_SUBCLASS_DETAILS 表函数 - 获取详细的服务子类度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE
MON_GET_UNIT_OF_WORK 表函数 - 获取工作单元度量值	REQUEST METRICS BASE
MON_GET_UNIT_OF_WORK 表函数 - 获取详细的工作单元度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE
MON_GET_WORKLOAD 表函数 - 获取工作负载度量值	REQUEST METRICS BASE
MON_GET_WORKLOAD_DETAILS 表函数 - 获取详细的工作负载度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE

表 1952. 事件监视信息

事件类型	逻辑数据分组	监视开关
统计信息	event_scstats（在 details_xml 文档中报告）	REQUEST METRICS BASE
统计信息	event_wlstats（在 details_xml 文档中报告）	REQUEST METRICS BASE
工作单元	在 system_metrics 文档中报告。	REQUEST METRICS BASE

total_sec_cons -“辅助连接数”

子代理程序与节点上的数据库建立的连接的数目。

元素标识

total_sec_cons

元素类型

计数器

表 1953. 快照监视信息

快照级别	逻辑数据分组	监视开关
数据库	dbase	基本

用法 可将此元素与 total_cons、db_conn_time 和 db2start_time 监视元素一起使用来计算应用程序与数据库建立连接的频率。

total_section_proc_time -“部分处理时间总计”监视元素

代理程序用于执行部分执行的处理时间总计。处理时间不包括等待时间。此值以毫秒计。

表 1954. 表函数监视信息

表函数	监视元素收集级别
MON_FORMAT_XML_METRICS_BY_ROW - 获取所有度量值的基于已格式化行的输出	不适用; 报告 XML 文档中作为格式化函数的输入提供的所有元素
MON_FORMAT_XML_TIMES_BY_ROW - - 获取基于已格式化行的组合层次结构等待时间和处理时间	不适用; 报告 XML 文档中作为格式化函数的输入提供的所有元素
MON_GET_ACTIVITY_DETAILS 表函数 - 获取完整的活动详细信息（在 XML 文档 DETAILS 中报告）	ACTIVITY METRICS BASE
MON_GET_CONNECTION 表函数 - 获取连接度量值	REQUEST METRICS BASE
MON_GET_CONNECTION_DETAILS 表函数 - 获取详细的连接度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE
MON_GET_PKG_CACHE_STMT 表函数 - 获取程序包高速缓存中的 SQL 语句活动度量值	ACTIVITY METRICS BASE
MON_GET_PKG_CACHE_STMT_DETAILS 表函数 - 获取程序包高速缓存条目的详细度量值	ACTIVITY METRICS BASE
MON_GET_SERVICE_SUBCLASS 表函数 - 获取服务子类度量值	REQUEST METRICS BASE
MON_GET_SERVICE_SUBCLASS_DETAILS 表函数 - 获取详细的服务子类度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE
MON_GET_UNIT_OF_WORK 表函数 - 获取工作单元度量值	REQUEST METRICS BASE
MON_GET_UNIT_OF_WORK 表函数 - 获取详细的工作单元度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE
MON_GET_WORKLOAD 表函数 - 获取工作负载度量值	REQUEST METRICS BASE

表 1954. 表函数监视信息 (续)

表函数	监视元素收集级别
MON_GET_WORKLOAD_DETAILS 表函数 - 获取详细的工作负载度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE

表 1955. 事件监视信息

事件类型	逻辑数据分组	监视开关
活动	event_activity（在 details_xml 文档中报告）	ACTIVITY METRICS BASE
活动	event_activitymetrics	ACTIVITY METRICS BASE
统计信息	event_scstats（在 details_xml 文档中报告）	REQUEST METRICS BASE
统计信息	event_wlstats（在 details_xml 文档中报告）	REQUEST METRICS BASE
程序包高速缓存	在 activity_metrics 文档中报告。	ACTIVITY METRICS BASE
工作单元	在 system_metrics 文档中报告。	REQUEST METRICS BASE

用法

如果收集级别设置为 BASE，那么 **total_section_proc_time** 监视元素的值包含执行使用 NO SQL 子句定义的函数所耗用的处理时间。

如果收集级别设置为 EXTENDED，那么 **total_section_proc_time** 监视元素的值中不包含执行这些函数所耗用的处理时间。这些处理时间包含在 **total_routine_user_code_proc_time** 监视元素的值中。

total_section_sort_proc_time -“节排序处理时间总计”监视元素

在执行节期间执行排序时耗用的非等待时间总计（执行节是指执行已编译的查询方案，此方案由客户机应用程序发出的 SQL 语句生成）。此值以毫秒计。

表 1956. 表函数监视信息

表函数	监视元素收集级别
MON_FORMAT_XML_METRICS_BY_ROW - 获取所有度量值的基于已格式化行的输出	不适用；报告 XML 文档中作为格式化函数的输入提供的所有元素
MON_FORMAT_XML_TIMES_BY_ROW - - 获取基于已格式化行的组合层次结构等待时间和处理时间	不适用；报告 XML 文档中作为格式化函数的输入提供的所有元素
MON_GET_ACTIVITY_DETAILS 表函数 - 获取完整的活动详细信息（在 XML 文档 DETAILS 中报告）	ACTIVITY METRICS BASE
MON_GET_CONNECTION 表函数 - 获取连接度量值	REQUEST METRICS BASE

表 1956. 表函数监视信息 (续)

表函数	监视元素收集级别
MON_GET_CONNECTION_DETAILS 表函数 - 获取详细的连接度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE
MON_GET_PKG_CACHE_STMT 表函数 - 获取程序包高速缓存中的 SQL 语句活动度量值	ACTIVITY METRICS BASE
MON_GET_PKG_CACHE_STMT_DETAILS 表函数 - 获取程序包高速缓存条目的详细度量值	ACTIVITY METRICS BASE
MON_GET_SERVICE_SUBCLASS 表函数 - 获取服务子类度量值	REQUEST METRICS BASE
MON_GET_SERVICE_SUBCLASS_DETAILS 表函数 - 获取详细的服务子类度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE
MON_GET_UNIT_OF_WORK 表函数 - 获取工作单元度量值	REQUEST METRICS BASE
MON_GET_UNIT_OF_WORK 表函数 - 获取详细的工作单元度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE
MON_GET_WORKLOAD 表函数 - 获取工作负载度量值	REQUEST METRICS BASE
MON_GET_WORKLOAD_DETAILS 表函数 - 获取详细的工作负载度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE

表 1957. 事件监视信息

事件类型	逻辑数据分组	监视开关
活动	event_activity（在 details_xml 文档中报告）	ACTIVITY METRICS BASE
活动	event_activitymetrics	ACTIVITY METRICS BASE
统计信息	event_scstats（在 details_xml 文档中报告）	REQUEST METRICS BASE
统计信息	event_wlstats（在 details_xml 文档中报告）	REQUEST METRICS BASE
程序包高速缓存	在 activity_metrics 文档中报告。	ACTIVITY METRICS BASE
工作单元	在 system_metrics 文档中报告。	REQUEST METRICS BASE

用法

在系统级别，通过将此元素与 **total_section_sorts** 监视元素配合使用，可以计算在执行部分期间的平均排序处理时间（不包括等待时间），此时间可以指示就性能而言排序是否存在问题。

在活动级别，使用此元素来确定耗用大量时间进行排序的语句。如果另外进行调整以降低排序时间，这些语句将从中受益。

total_section_sort_time -“节排序时间总计”监视元素

在执行节期间执行排序时的耗用时间总计（执行节是指执行已编译的查询方案，此方案由客户机应用程序发出的 SQL 语句生成）。此值以毫秒计。

表 1958. 表函数监视信息

表函数	监视元素收集级别
MON_FORMAT_XML_COMPONENT _TIMES_BY_ROW - 获取基于已格式化行的组件次数	不适用; 报告 XML 文档中作为格式化函数的输入提供的所有元素
MON_FORMAT_XML_METRICS_BY_ROW - 获取所有度量值的基于已格式化行的输出	不适用; 报告 XML 文档中作为格式化函数的输入提供的所有元素
MON_GET_ACTIVITY_DETAILS 表函数 - 获取完整的活动详细信息（在 XML 文档 DETAILS 中报告）	ACTIVITY METRICS BASE
MON_GET_CONNECTION 表函数 - 获取连接度量值	REQUEST METRICS BASE
MON_GET_CONNECTION_DETAILS 表函数 - 获取详细的连接度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE
MON_GET_PKG_CACHE_STMT 表函数 - 获取程序包高速缓存中的 SQL 语句活动度量值	ACTIVITY METRICS BASE
MON_GET_PKG_CACHE_STMT_DETAILS 表函数 - 获取程序包高速缓存条目的详细度量值	ACTIVITY METRICS BASE
MON_GET_SERVICE_SUBCLASS 表函数 - 获取服务子类度量值	REQUEST METRICS BASE
MON_GET_SERVICE_SUBCLASS_DETAILS 表函数 - 获取详细的服务子类度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE
MON_GET_UNIT_OF_WORK 表函数 - 获取工作单元度量值	REQUEST METRICS BASE
MON_GET_UNIT_OF_WORK 表函数 - 获取详细的工作单元度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE
MON_GET_WORKLOAD 表函数 - 获取工作负载度量值	REQUEST METRICS BASE
MON_GET_WORKLOAD_DETAILS 表函数 - 获取详细的工作负载度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE

表 1959. 事件监视信息

事件类型	逻辑数据分组	监视开关
活动	event_activity（在 details_xml 文档中报告）	ACTIVITY METRICS BASE
活动	event_activitymetrics	ACTIVITY METRICS BASE
统计信息	event_scstats（在 details_xml 文档中报告）	REQUEST METRICS BASE

表 1959. 事件监视信息 (续)

事件类型	逻辑数据分组	监视开关
统计信息	event_wlstats（在 details_xml 文档中报告）	REQUEST METRICS BASE
程序包高速缓存	在 activity_metrics 文档中报告。	ACTIVITY METRICS BASE
工作单元	在 system_metrics 文档中报告。	REQUEST METRICS BASE

用法

在系统级别，通过将此元素与 **total_section_sorts** 监视元素配合使用，可以计算执行节期间的平均排序时间，此时间可以指示就语句性能而言排序是否存在问题。

total_section_sort_time 元素既包含等待时间也包含处理时间。如果 (total_section_sort_time - total_section_sort_proc_time) 的值较大，表明执行排序操作时耗用了大量时间进行等待。例如，如果排序频繁溢出到磁盘，那么 **total_section_sort_time** 监视元素的值将由于 I/O 等待而增大。此时间将不会包括在 **total_section_sort_proc_time** 监视元素值中，而只是计作主动处理排序所耗用的时间。在这种情况下，您可以考虑调整排序内存以提高性能。

在活动级别，使用此元素来确定耗用大量时间进行排序的语句。如果另外进行调整以降低排序时间，这些语句将从中受益。

total_section_sorts -“节排序总次数”监视元素

执行节期间执行的排序的总次数（执行节是指执行已编译的查询方案，此方案由客户机应用程序发出的 SQL 语句生成）。

表 1960. 表函数监视信息

表函数	监视元素收集级别
MON_FORMAT_XML_METRICS_BY_ROW - 获取所有度量值的基于已格式化行的输出	不适用; 报告 XML 文档中作为格式化函数的输入提供的所有元素
MON_GET_ACTIVITY_DETAILS 表函数 - 获取完整的活动详细信息（在 XML 文档 DETAILS 中报告）	ACTIVITY METRICS BASE
MON_GET_CONNECTION 表函数 - 获取连接度量值	REQUEST METRICS BASE
MON_GET_CONNECTION_DETAILS 表函数 - 获取详细的连接度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE
MON_GET_PKG_CACHE_STMT 表函数 - 获取程序包高速缓存中的 SQL 语句活动度量值	ACTIVITY METRICS BASE
MON_GET_PKG_CACHE_STMT_DETAILS 表函数 - 获取程序包高速缓存条目的详细度量值	ACTIVITY METRICS BASE
MON_GET_SERVICE_SUBCLASS 表函数 - 获取服务子类度量值	REQUEST METRICS BASE

表 1960. 表函数监视信息 (续)

表函数	监视元素收集级别
MON_GET_SERVICE_SUBCLASS_DETAILS 表函数 - 获取详细的服务子类度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE
MON_GET_UNIT_OF_WORK 表函数 - 获取工作单元度量值	REQUEST METRICS BASE
MON_GET_UNIT_OF_WORK 表函数 - 获取详细的工作单元度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE
MON_GET_WORKLOAD 表函数 - 获取工作负载度量值	REQUEST METRICS BASE
MON_GET_WORKLOAD_DETAILS 表函数 - 获取详细的工作负载度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE

表 1961. 事件监视信息

事件类型	逻辑数据分组	监视开关
活动	event_activity（在 details_xml 文档中报告）	ACTIVITY METRICS BASE
活动	event_activitymetrics	ACTIVITY METRICS BASE
统计信息	event_scstats（在 details_xml 文档中报告）	REQUEST METRICS BASE
统计信息	event_wlstats（在 details_xml 文档中报告）	REQUEST METRICS BASE
程序包高速缓存	在 activity_metrics 文档中报告。	ACTIVITY METRICS BASE
工作单元	在 system_metrics 文档中报告。	REQUEST METRICS BASE

用法

通过将此元素与 **total_section_sort_time** 监视元素配合使用，可以计算在执行部分期间执行排序所耗用的平均时间。

在活动和程序包高速缓存级别，使用此元素来标识执行大量排序的语句。如果另外进行调整以降低排序数目，这些语句将从中受益。还可使用 EXPLAIN 语句来标识语句执行的排序数。

total_section_time -“部分时间总计”监视元素

代理程序用于执行部分执行的时间总计。此值以毫秒计。

表 1962. 表函数监视信息

表函数	监视元素收集级别
MON_FORMAT_XML_COMPONENT_TIMES_BY_ROW - 获取基于已格式化行的组件次数	不适用；报告 XML 文档中作为格式化函数的输入提供的所有元素

表 1962. 表函数监视信息 (续)

表函数	监视元素收集级别
MON_FORMAT_XML_METRICS_BY_ROW - 获取所有度量值的基于已格式化行的输出	不适用; 报告 XML 文档中作为格式化函数的输入提供的所有元素
MON_GET_ACTIVITY_DETAILS 表函数 - 获取完整的活动详细信息（在 XML 文档 DETAILS 中报告）	ACTIVITY METRICS BASE
MON_GET_CONNECTION 表函数 - 获取连接度量值	REQUEST METRICS BASE
MON_GET_CONNECTION_DETAILS 表函数 - 获取详细的连接度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE
MON_GET_PKG_CACHE_STMT 表函数 - 获取程序包高速缓存中的 SQL 语句活动度量值	ACTIVITY METRICS BASE
MON_GET_PKG_CACHE_STMT_DETAILS 表函数 - 获取程序包高速缓存条目的详细度量值	ACTIVITY METRICS BASE
MON_GET_SERVICE_SUBCLASS 表函数 - 获取服务子类度量值	REQUEST METRICS BASE
MON_GET_SERVICE_SUBCLASS_DETAILS 表函数 - 获取详细的服务子类度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE
MON_GET_UNIT_OF_WORK 表函数 - 获取工作单元度量值	REQUEST METRICS BASE
MON_GET_UNIT_OF_WORK 表函数 - 获取详细的工作单元度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE
MON_GET_WORKLOAD 表函数 - 获取工作负载度量值	REQUEST METRICS BASE
MON_GET_WORKLOAD_DETAILS 表函数 - 获取详细的工作负载度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE

表 1963. 事件监视信息

事件类型	逻辑数据分组	监视开关
活动	event_activity（在 details_xml 文档中报告）	ACTIVITY METRICS BASE
活动	event_activitymetrics	ACTIVITY METRICS BASE
统计信息	event_scstats（在 details_xml 文档中报告）	REQUEST METRICS BASE
统计信息	event_wlstats（在 details_xml 文档中报告）	REQUEST METRICS BASE
程序包高速缓存	在 activity_metrics 文档中报告。	ACTIVITY METRICS BASE
工作单元	在 system_metrics 文档中报告。	REQUEST METRICS BASE

用法

如果收集级别设置为 BASE，那么 **total_section_time** 监视元素的值包含执行使用 NO SQL 子句定义的函数所耗用的时间。

如果收集级别设置为 EXTENDED，那么 **total_section_time** 监视元素的值中不包含执行这些函数所耗用的时间。这些时间而是包含在 **total_routine_user_code_time** 监视元素的值中。

total_sort_time -“排序时间总计”监视元素

已执行的所有排序的耗用时间总计。此值以毫秒计。

表 1964. 快照监视信息

快照级别	逻辑数据分组	监视开关
数据库	dbase	排序
应用程序	appl	排序
应用程序	stmt	排序
动态 SQL	dynsql	排序

可将快照监视的计数器重置。

表 1965. 事件监视信息

事件类型	逻辑数据分组	监视开关
数据库	event_db	始终收集
连接	event_conn	始终收集
语句	event_stmt	始终收集
活动	event_activity	语句，排序

用法

在数据库或应用程序级别，将此元素与 **total_sorts** 配合使用来计算平均排序时间，它可以指示就性能而言排序是否存在问题。

在语句级别，使用此元素来标识花费大量时间进行排序的语句。如果另外进行调整以降低排序时间，这些语句将从中受益。

此计数还包括相关操作期间创建的临时表的排序时间。它提供有关一个语句、一个应用程序或访问一个数据库的所有应用程序的信息。

使用提供耗用时间的监视元素时，应考虑:

1. 耗用时间受系统负载影响，所以运行的进程越多，此耗用时间值越高。
2. 要在数据库级别计算此监视元素，数据库系统监视器会将应用程序级别时间求和。这会导致对数据库级别的耗用时间双倍计数，原因是多个应用程序进程可能同时运行。

   要在数据库级别提供有意义的数据，应将数据标准化以将其降至较低级别。例如:

```
total_sort_time / total_sorts
```

提供有关每个排序的平均耗用时间的信息。

total_sorts -“排序总数”监视元素

已执行的排序总数。

表 1966. 表函数监视信息

表函数	监视元素收集级别
MON_FORMAT_XML_METRICS_BY_ROW - 获取所有度量值的基于已格式化行的输出	不适用; 报告 XML 文档中作为格式化函数的输入提供的所有元素
MON_GET_ACTIVITY_DETAILS 表函数 - 获取完整的活动详细信息（在 XML 文档 DETAILS 中报告）	ACTIVITY METRICS BASE
MON_GET_CONNECTION 表函数 - 获取连接度量值	REQUEST METRICS BASE
MON_GET_CONNECTION_DETAILS 表函数 - 获取详细的连接度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE
MON_GET_PKG_CACHE_STMT 表函数 - 获取程序包高速缓存中的 SQL 语句活动度量值	ACTIVITY METRICS BASE
MON_GET_PKG_CACHE_STMT_DETAILS 表函数 - 获取程序包高速缓存条目的详细度量值	ACTIVITY METRICS BASE
MON_GET_SERVICE_SUBCLASS 表函数 - 获取服务子类度量值	REQUEST METRICS BASE
MON_GET_SERVICE_SUBCLASS_DETAILS 表函数 - 获取详细的服务子类度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE
MON_GET_UNIT_OF_WORK 表函数 - 获取工作单元度量值	REQUEST METRICS BASE
MON_GET_UNIT_OF_WORK 表函数 - 获取详细的工作单元度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE
MON_GET_WORKLOAD 表函数 - 获取工作负载度量值	REQUEST METRICS BASE
MON_GET_WORKLOAD_DETAILS 表函数 - 获取详细的工作负载度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE

表 1967. 快照监视信息

快照级别	逻辑数据分组	监视开关
数据库	dbase	基本
应用程序	appl	基本

可将快照监视的计数器重置。

表 1968. 事件监视信息

事件类型	逻辑数据分组	监视开关
活动	event_activity（在 details_xml 文档中报告）	ACTIVITY METRICS BASE
活动	event_activitymetrics	ACTIVITY METRICS BASE
统计信息	event_scstats（在 details_xml 文档中报告）	REQUEST METRICS BASE
统计信息	event_wlstats（在 details_xml 文档中报告）	REQUEST METRICS BASE
工作单元	在 system_metrics 文档中报告。	REQUEST METRICS BASE
数据库	event_db	始终收集
连接	event_conn	始终收集
语句	event_stmt	始终收集
活动	event_activity	语句，排序
程序包高速缓存	在 activity_metrics 文档中报告。	ACTIVITY METRICS BASE

用法

在数据库或应用程序级别，将此值与 **sort_overflows** 配合使用来计算需要更多堆空间的排序的百分比。还可将其与 **total_sort_time** 配合使用来计算平均排序时间。

如果排序溢出数相对排序总数较小，那么除非此缓冲区大小实际上增加了，否则提高排序堆大小对性能没什么影响。

在语句级别，使用此元素来标识执行大量排序的语句。如果另外进行调整以降低排序数目，这些语句将从中受益。还可使用 SQL EXPLAIN 语句来标识语句执行的排序数。

total_stats_fabrication_proc_time -“统计信息生成处理时间总计”监视元素

实时统计信息收集在生成统计信息所花的非等待时间总计（以毫秒计）。生成统计信息是在查询编译期间生成统计信息所需的统计信息收集活动。

表 1969. 表函数监视信息

表函数	监视元素收集级别
MON_FORMAT_XML_METRICS_BY_ROW - 获取所有度量值的基于已格式化行的输出	不适用；报告 XML 文档中作为格式化函数的输入提供的所有元素
MON_FORMAT_XML_TIMES_BY_ROW - - 获取基于已格式化行的组合层次结构等待时间和处理时间	不适用；报告 XML 文档中作为格式化函数的输入提供的所有元素
MON_GET_CONNECTION 表函数 - 获取连接度量值	REQUEST METRICS BASE
MON_GET_CONNECTION_DETAILS 表函数 - 获取详细的连接度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE
MON_GET_SERVICE_SUBCLASS 表函数 - 获取服务子类度量值	REQUEST METRICS BASE

表 1969. 表函数监视信息 (续)

表函数	监视元素收集级别
MON_GET_SERVICE_SUBCLASS_DETAILS 表函数 - 获取详细的服务子类度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE
MON_GET_UNIT_OF_WORK 表函数 - 获取工作单元度量值	REQUEST METRICS BASE
MON_GET_UNIT_OF_WORK 表函数 - 获取详细的工作单元度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE
MON_GET_WORKLOAD 表函数 - 获取工作负载度量值	REQUEST METRICS BASE
MON_GET_WORKLOAD_DETAILS 表函数 - 获取详细的工作负载度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE

表 1970. 事件监视信息

事件类型	逻辑数据分组	监视开关
统计信息	event_scstats（在 details_xml 文档中报告）	REQUEST METRICS BASE
统计信息	event_wlstats（在 details_xml 文档中报告）	REQUEST METRICS BASE
工作单元	在 system_metrics 文档中报告。	REQUEST METRICS BASE

total_stats_fabrication_time -“统计信息生成时间总计”监视元素

实时统计信息收集在生成统计信息所花的总时间（以毫秒计）。生成统计信息是在查询编译期间生成统计信息所需的统计信息收集活动。

表 1971. 表函数监视信息

表函数	监视元素收集级别
MON_FORMAT_XML_COMPONENT _TIMES_BY_ROW - 获取基于已格式化行的组件次数	不适用; 报告 XML 文档中作为格式化函数的输入提供的所有元素
MON_FORMAT_XML_METRICS_BY_ROW - 获取所有度量值的基于已格式化行的输出	不适用; 报告 XML 文档中作为格式化函数的输入提供的所有元素
MON_GET_CONNECTION 表函数 - 获取连接度量值	REQUEST METRICS BASE
MON_GET_CONNECTION_DETAILS 表函数 - 获取详细的连接度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE
MON_GET_PKG_CACHE_STMT 表函数 - 获取程序包高速缓存中的 SQL 语句活动度量值	ACTIVITY METRICS BASE
MON_GET_PKG_CACHE_STMT_DETAILS 表函数 - 获取程序包高速缓存条目的详细度量值（在 XML 文档 DETAILS 中报告）	ACTIVITY METRICS BASE

表 1971. 表函数监视信息 (续)

表函数	监视元素收集级别
MON_GET_SERVICE_SUBCLASS 表函数 - 获取服务子类度量值	REQUEST METRICS BASE
MON_GET_SERVICE_SUBCLASS_DETAILS 表函数 - 获取详细的服务子类度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE
MON_GET_UNIT_OF_WORK 表函数 - 获取工作单元度量值	REQUEST METRICS BASE
MON_GET_UNIT_OF_WORK 表函数 - 获取详细的工作单元度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE
MON_GET_WORKLOAD 表函数 - 获取工作负载度量值	REQUEST METRICS BASE
MON_GET_WORKLOAD_DETAILS 表函数 - 获取详细的工作负载度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE

表 1972. 事件监视信息

事件类型	逻辑数据分组	监视开关
活动	event_activity	始终收集
统计信息	event_scstats（在 details_xml 文档中报告）	REQUEST METRICS BASE
统计信息	event_wlstats（在 details_xml 文档中报告）	REQUEST METRICS BASE
工作单元	uow_metrics	REQUEST METRICS BASE
程序包高速缓存	pkgcache	始终收集

total_stats_fabrications -“统计信息生成总计”监视元素

实时统计信息收集执行的统计信息生成总数。生成统计信息是在查询编译期间生成统计信息所需的统计信息收集活动。

表 1973. 表函数监视信息

表函数	监视元素收集级别
MON_FORMAT_XML_METRICS_BY_ROW - 获取所有度量值的基于已格式化行的输出	不适用; 报告 XML 文档中作为格式化函数的输入提供的所有元素
MON_GET_CONNECTION 表函数 - 获取连接度量值	REQUEST METRICS BASE
MON_GET_CONNECTION_DETAILS 表函数 - 获取详细的连接度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE
MON_GET_PKG_CACHE_STMT 表函数 - 获取程序包高速缓存中的 SQL 语句活动度量值	ACTIVITY METRICS BASE

表 1973. 表函数监视信息 (续)

表函数	监视元素收集级别
MON_GET_PKG_CACHE_STMT_DETAILS 表函数 - 获取程序包高速缓存条目的详细度量值（在 XML 文档 DETAILS 中报告）	ACTIVITY METRICS BASE
MON_GET_SERVICE_SUBCLASS 表函数 - 获取服务子类度量值	REQUEST METRICS BASE
MON_GET_SERVICE_SUBCLASS_DETAILS 表函数 - 获取详细的服务子类度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE
MON_GET_UNIT_OF_WORK 表函数 - 获取工作单元度量值	REQUEST METRICS BASE
MON_GET_UNIT_OF_WORK 表函数 - 获取详细的工作单元度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE
MON_GET_WORKLOAD 表函数 - 获取工作负载度量值	REQUEST METRICS BASE
MON_GET_WORKLOAD_DETAILS 表函数 - 获取详细的工作负载度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE

表 1974. 事件监视信息

事件类型	逻辑数据分组	监视开关
活动	event_activity	ACTIVITY METRICS BASE
统计信息	event_scstats（在 details_xml 文档中报告）	REQUEST METRICS BASE
统计信息	event_wlstats（在 details_xml 文档中报告）	REQUEST METRICS BASE
工作单元	在 system_metrics 文档中报告。	REQUEST METRICS BASE
程序包高速缓存	在 activity_metrics 文档中报告。	始终收集

total_sync_runstats_time -“同步 RUNSTATS 时间总计”监视元素

实时统计信息收集触发的同步 RUNSTATS 活动所花的总时间（以毫秒计）。在查询编译期间发生同步 RUNSTATS 活动。

表 1975. 表函数监视信息

表函数	监视元素收集级别
MON_FORMAT_XML_COMPONENT _TIMES_BY_ROW - 获取基于已格式化行的组件次数	不适用; 报告 XML 文档中作为格式化函数的输入提供的所有元素
MON_FORMAT_XML_METRICS_BY_ROW - 获取所有度量值的基于已格式化行的输出	不适用; 报告 XML 文档中作为格式化函数的输入提供的所有元素
MON_GET_CONNECTION 表函数 - 获取连接度量值	REQUEST METRICS BASE

表 1975. 表函数监视信息 (续)

表函数	监视元素收集级别
MON_GET_CONNECTION_DETAILS 表函数 - 获取详细的连接度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE
MON_GET_PKG_CACHE_STMT 表函数 - 获取程序包高速缓存中的 SQL 语句活动度量值	ACTIVITY METRICS BASE
MON_GET_PKG_CACHE_STMT_DETAILS 表函数 - 获取程序包高速缓存条目的详细度量值（在 XML 文档 DETAILS 中报告）	ACTIVITY METRICS BASE
MON_GET_SERVICE_SUBCLASS 表函数 - 获取服务子类度量值	REQUEST METRICS BASE
MON_GET_SERVICE_SUBCLASS_DETAILS 表函数 - 获取详细的服务子类度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE
MON_GET_UNIT_OF_WORK 表函数 - 获取工作单元度量值	REQUEST METRICS BASE
MON_GET_UNIT_OF_WORK 表函数 - 获取详细的工作单元度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE
MON_GET_WORKLOAD 表函数 - 获取工作负载度量值	REQUEST METRICS BASE
MON_GET_WORKLOAD_DETAILS 表函数 - 获取详细的工作负载度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE

表 1976. 事件监视信息

事件类型	逻辑数据分组	监视开关
活动	event_activity	ACTIVITY METRICS BASE
统计信息	event_scstats（在 details_xml 文档中报告）	REQUEST METRICS BASE
统计信息	event_wlstats（在 details_xml 文档中报告）	REQUEST METRICS BASE
工作单元	在 system_metrics 文档中报告。	REQUEST METRICS BASE
程序包高速缓存	在 activity_metrics 文档中报告。	始终收集

total_sync_runstats_proc_time -“同步 RUNSTATS 处理时间总计”监视元素

实时统计信息收集触发的同步 RUNSTATS 活动所花的非等待时间（以毫秒计）。在查询编译期间发生同步 RUNSTATS 活动。

表 1977. 表函数监视信息

表函数	监视元素收集级别
MON_FORMAT_XML_METRICS_BY_ROW - 获取所有度量值的基于已格式化行的输出	不适用; 报告 XML 文档中作为格式化函数的输入提供的所有元素
MON_FORMAT_XML_TIMES_BY_ROW - - 获取基于已格式化行的组合层次结构等待时间和处理时间	不适用; 报告 XML 文档中作为格式化函数的输入提供的所有元素
MON_GET_CONNECTION 表函数 - 获取连接度量值	REQUEST METRICS BASE
MON_GET_CONNECTION_DETAILS 表函数 - 获取详细的连接度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE
MON_GET_SERVICE_SUBCLASS 表函数 - 获取服务子类度量值	REQUEST METRICS BASE
MON_GET_SERVICE_SUBCLASS_DETAILS 表函数 - 获取详细的服务子类度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE
MON_GET_UNIT_OF_WORK 表函数 - 获取工作单元度量值	REQUEST METRICS BASE
MON_GET_UNIT_OF_WORK 表函数 - 获取详细的工作单元度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE
MON_GET_WORKLOAD 表函数 - 获取工作负载度量值	REQUEST METRICS BASE
MON_GET_WORKLOAD_DETAILS 表函数 - 获取详细的工作负载度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE

表 1978. 事件监视信息

事件类型	逻辑数据分组	监视开关
统计信息	event_scstats（在 details_xml 文档中报告）	REQUEST METRICS BASE
统计信息	event_wlstats（在 details_xml 文档中报告）	REQUEST METRICS BASE
工作单元	在 system_metrics 文档中报告。	REQUEST METRICS BASE

total_sync_runstats -“同步 RUNSTATS 活动总数”监视元素

实时统计信息收集触发同步 RUNSTATS 活动的总数。在查询编译期间发生同步 RUNSTATS 活动。

表 1979. 表函数监视信息

表函数	监视元素收集级别
MON_FORMAT_XML_METRICS_BY_ROW - 获取所有度量值的基于已格式化行的输出	不适用; 报告 XML 文档中作为格式化函数的输入提供的所有元素
MON_GET_ACTIVITY_DETAILS 表函数 - 获取完整的活动详细信息	ACTIVITY METRICS BASE
MON_GET_CONNECTION 表函数 - 获取连接度量值	REQUEST METRICS BASE
MON_GET_CONNECTION_DETAILS 表函数 - 获取详细的连接度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE
MON_GET_PKG_CACHE_STMT 表函数 - 获取程序包高速缓存中的 SQL 语句活动度量值	ACTIVITY METRICS BASE
MON_GET_PKG_CACHE_STMT_DETAILS 表函数 - 获取程序包高速缓存条目的详细度量值（在 XML 文档 DETAILS 中报告）	ACTIVITY METRICS BASE
MON_GET_SERVICE_SUBCLASS 表函数 - 获取服务子类度量值	REQUEST METRICS BASE
MON_GET_SERVICE_SUBCLASS_DETAILS 表函数 - 获取详细的服务子类度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE
MON_GET_UNIT_OF_WORK 表函数 - 获取工作单元度量值	REQUEST METRICS BASE
MON_GET_UNIT_OF_WORK 表函数 - 获取详细的工作单元度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE
MON_GET_WORKLOAD 表函数 - 获取工作负载度量值	REQUEST METRICS BASE
MON_GET_WORKLOAD_DETAILS 表函数 - 获取详细的工作负载度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE

表 1980. 事件监视信息

事件类型	逻辑数据分组	监视开关
活动	event_activity	ACTIVITY METRICS BASE
统计信息	event_scstats（在 details_xml 文档中报告）	REQUEST METRICS BASE
统计信息	event_wlstats（在 details_xml 文档中报告）	REQUEST METRICS BASE
工作单元	在 system_metrics 文档中报告。	REQUEST METRICS BASE
程序包高速缓存	在 activity_metrics 文档中报告。	始终收集

total_sys_cpu_time -“语句的系统 CPU 时间总计”监视元素

SQL 语句的总系统 CPU 时间。

表 1981. 快照监视信息

快照级别	逻辑数据分组	监视开关
动态 SQL	dynsql	语句

可将快照监视的计数器重置。

用法

将此元素与执行语句所耗用的时间和语句的总用户 CPU 配合使用以找出执行成本最高的语句。

此元素由两个子元素组成，它们报告耗用时间的秒数和微秒（一秒的百万分之一）数。这些子元素的名称可通过将“_s”和“_ms”添加至此监视元素的名称派生而成。要检索此监视元素耗用的总时间，必须将这两个子元素的值加在一起。例如，如果“_s”子元素值为 3，“_ms”子元素值为 20，那么此监视元素耗用的总时间为 3.00002 秒。

total_sorts -“排序总数”监视元素

已执行的排序总数。

表 1982. 表函数监视信息

表函数	监视元素收集级别
MON_FORMAT_XML_METRICS_BY_ROW - 获取所有度量值的基于已格式化行的输出	不适用；报告 XML 文档中作为格式化函数的输入提供的所有元素
MON_GET_ACTIVITY_DETAILS 表函数 - 获取完整的活动详细信息（在 XML 文档 DETAILS 中报告）	ACTIVITY METRICS BASE
MON_GET_CONNECTION 表函数 - 获取连接度量值	REQUEST METRICS BASE
MON_GET_CONNECTION_DETAILS 表函数 - 获取详细的连接度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE
MON_GET_PKG_CACHE_STMT 表函数 - 获取程序包高速缓存中的 SQL 语句活动度量值	ACTIVITY METRICS BASE
MON_GET_PKG_CACHE_STMT_DETAILS 表函数 - 获取程序包高速缓存条目的详细度量值	ACTIVITY METRICS BASE
MON_GET_SERVICE_SUBCLASS 表函数 - 获取服务子类度量值	REQUEST METRICS BASE
MON_GET_SERVICE_SUBCLASS_DETAILS 表函数 - 获取详细的服务子类度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE

表 1982. 表函数监视信息 (续)

表函数	监视元素收集级别
MON_GET_UNIT_OF_WORK 表函数 - 获取工作单元度量值	REQUEST METRICS BASE
MON_GET_UNIT_OF_WORK 表函数 - 获取详细的工作单元度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE
MON_GET_WORKLOAD 表函数 - 获取工作负载度量值	REQUEST METRICS BASE
MON_GET_WORKLOAD_DETAILS 表函数 - 获取详细的工作负载度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE

表 1983. 快照监视信息

快照级别	逻辑数据分组	监视开关
数据库	dbase	基本
应用程序	appl	基本

可将快照监视的计数器重置。

表 1984. 事件监视信息

事件类型	逻辑数据分组	监视开关
活动	event_activity（在 details_xml 文档中报告）	ACTIVITY METRICS BASE
活动	event_activitymetrics	ACTIVITY METRICS BASE
统计信息	event_scstats（在 details_xml 文档中报告）	REQUEST METRICS BASE
统计信息	event_wlstats（在 details_xml 文档中报告）	REQUEST METRICS BASE
工作单元	在 system_metrics 文档中报告。	REQUEST METRICS BASE
数据库	event_db	始终收集
连接	event_conn	始终收集
语句	event_stmt	始终收集
活动	event_activity	语句，排序
程序包高速缓存	在 activity_metrics 文档中报告。	ACTIVITY METRICS BASE

用法

在数据库或应用程序级别，将此值与 **sort_overflows** 配合使用来计算需要更多堆空间的排序的百分比。还可将其与 **total_sort_time** 配合使用来计算平均排序时间。

如果排序溢出数相对排序总数较小，那么除非此缓冲区大小实际上增加了，否则提高排序堆大小对性能没什么影响。

在语句级别，使用此元素来标识执行大量排序的语句。如果另外进行调整以降低排序数目，这些语句将从中受益。还可使用 SQL EXPLAIN 语句来标识语句执行的排序数。

total_usr_cpu_time -“语句的用户 CPU 时间总计”监视元素

SQL 语句的总用户 CPU 时间。

表 1985. 快照监视信息

快照级别	逻辑数据分组	监视开关
动态 SQL	dynsql	语句

可将快照监视的计数器重置。

用法

将此元素与执行语句所耗用的时间配合使用以找出运行时间最长的语句。

此元素由两个子元素组成，它们报告耗用时间的秒数和微秒（一秒的百万分之一）数。这些子元素的名称可通过将“_s”和“_ms”添加至此监视元素的名称派生而成。要检索此监视元素耗用的总时间，必须将这两个子元素的值加在一起。例如，如果“_s”子元素值为 3，“_ms”子元素值为 20，那么此监视元素耗用的总时间为 3.00002 秒。

total_wait_time -“等待时间总计”监视元素

在 DB2 数据库服务器中进行等待时的耗用时间总计。此值以毫秒计。

表 1986. 表函数监视信息

表函数	监视元素收集级别
MON_FORMAT_XML_METRICS_BY_ROW - 获取所有度量值的基于已格式化行的输出	不适用; 报告 XML 文档中作为格式化函数的输入提供的所有元素
MON_FORMAT_XML_TIMES_BY_ROW - - 获取基于已格式化行的组合层次结构等待时间和处理时间	不适用; 报告 XML 文档中作为格式化函数的输入提供的所有元素
MON_FORMAT_XML_WAIT_TIMES_BY _ROW - 获取等待次数的已格式化的基于行的输出	不适用; 报告 XML 文档中作为格式化函数的输入提供的所有元素
MON_GET_CONNECTION 表函数 - 获取连接度量值	REQUEST METRICS BASE
MON_GET_CONNECTION_DETAILS 表函数 - 获取详细的连接度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE
MON_GET_SERVICE_SUBCLASS 表函数 - 获取服务子类度量值	REQUEST METRICS BASE
MON_GET_SERVICE_SUBCLASS_DETAILS 表函数 - 获取详细的服务子类度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE
MON_GET_UNIT_OF_WORK 表函数 - 获取工作单元度量值	REQUEST METRICS BASE

表 1986. 表函数监视信息 (续)

表函数	监视元素收集级别
MON_GET_UNIT_OF_WORK 表函数 - 获取详细的工作单元度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE
MON_GET_WORKLOAD 表函数 - 获取工作负载度量值	REQUEST METRICS BASE
MON_GET_WORKLOAD_DETAILS 表函数 - 获取详细的工作负载度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE

表 1987. 事件监视信息

事件类型	逻辑数据分组	监视开关
统计信息	event_scstats（在 details_xml 文档中报告）	REQUEST METRICS BASE
统计信息	event_wlstats（在 details_xml 文档中报告）	REQUEST METRICS BASE
工作单元	在 system_metrics 文档中报告。	REQUEST METRICS BASE

用法

要了解数据库服务器主动处理请求时耗用的时间所占的百分比，请使用以下比率:

```
(total_rqst_time - total_wait_time) / total_rqst_time
```

client_idle_wait_time 监视元素的值未包括在 **total_wait_time** 监视元素的值中。**total_wait_time** 元素仅代表数据库服务器处理请求期间进行等待时耗用的时间。

tpmon_acc_str -“TP 监视器客户机记帐字符串”监视元素

如果在此连接中发出了 sqleseti API，那么此项是为了用于记录和诊断而传递至目标数据库的数据。 此连接、工作单元或活动的 CLIENT_ACCTNG 专用寄存器的当前值。

此监视元素与 **client_acctng** 监视元素同义。**client_acctng** 监视元素用于监视写入 DB2 V9.7 所引入的无格式表的表函数和事件监视器。**tpmon_acc_str** 监视元素用于写表、文件和管道的快照监视器和事件监视器。

表 1988. 快照监视信息

快照级别	逻辑数据分组	监视开关
应用程序	appl_info	基本
DCS 应用程序	dcs_appl	基本

表 1989. 事件监视信息

事件类型	逻辑数据分组	监视开关
活动	event_activity	-
死锁	event_dlconn	-
事务	event_xact	-

用法

此元素用于问题确定和记帐目的。

tpmon_client_app -“TP 监视器客户机应用程序名称”监视元素

如果在此连接中发出 sqleseti API，那么此项标识执行事务时出现的服务器事务程序问题。 此连接、工作单元或活动的 CLIENT_APPLNAME 专用寄存器的当前值。

此监视元素与 **client_applname** 监视元素同义。**client_applname** 监视元素用于监视写入 DB2 V9.7 所引入的无格式表的表函数和事件监视器。**tpmon_client_app** 监视元素用于写表、文件和管道的快照监视器和事件监视器。

表 1990. 快照监视信息

快照级别	逻辑数据分组	监视开关
应用程序	appl_info	基本
DCS 应用程序	dcs_appl	基本

表 1991. 事件监视信息

事件类型	逻辑数据分组	监视开关
活动	event_activity	-
死锁	event_dlconn	-
事务	event_xact	-

用法

此元素用于问题确定和记帐目的。

tpmon_client_userid -“TP 监视器客户机用户标识”监视元素

如果使用 sqleseti API，那么此项是由事务管理器生成的并且提供给服务器的客户机用户标识。 此连接、工作单元或活动的 CLIENT_USERID 专用寄存器的当前值。

此监视元素与 **client_userid** 监视元素同义。**client_userid** 监视元素用于监视写入 DB2 V9.7 所引入的无格式表的表函数和事件监视器。**tpmon_client_userid** 监视元素用于写表、文件和管道的快照监视器和事件监视器。

表 1992. 快照监视信息

快照级别	逻辑数据分组	监视开关
应用程序	appl_info	基本
DCS 应用程序	dcs_appl	基本

表 1993. 事件监视信息

事件类型	逻辑数据分组	监视开关
活动	event_activity	-
死锁	event_dlconn	-
事务	event_xact	-

用法

通过在应用程序服务器或事务处理监视器环境中使用此元素，可以标识为其执行事务的最终用户。

tpmon_client_wkstn -“TP 监视器客户机工作站名称”监视元素

如果在此连接中发出了 sqleseti API，那么此项标识客户机的系统或工作站（如 CICS EITERMID）。此连接、工作单元或活动的 CLIENT_WRKSTNNAME 专用寄存器的当前值。

此监视元素与 **client_wrkstnname** 监视元素同义。**client_wrkstnname** 监视元素用于监视写入 DB2 V9.7 所引入的无格式表的表函数和事件监视器。**tpmon_client_wkstn** 监视元素用于写表、文件和管道的快照监视器和事件监视器。

表 1994. 快照监视信息

快照级别	逻辑数据分组	监视开关
应用程序	appl_info	基本
DCS 应用程序	dcs_appl	基本

表 1995. 事件监视信息

事件类型	逻辑数据分组	监视开关
活动	event_activity	-
死锁	event_dlconn	-
事务	event_xact	-

用法

使用此元素并借助节点标识、终端标识或类似标识来标识用户的机器。

tq_cur_send_spills -“当前溢出的表队列缓冲区数”监视元素

当前驻留在临时表中的表队列缓冲区数。

表 1996. 快照监视信息

快照级别	逻辑数据分组	监视开关
应用程序	subsection	语句

用法 写至表队列的代理程序可能正将行发送至若干阅读器。当向其发送行的代理程序不接受行而另一代理程序需要行以进行处理时，写代理程序会将缓冲区溢出至临时表。溢出至临时表允许写程序和其他阅读器同时继续进行处理。

当读取代理程序准备接受更多行时，已溢出的行将发送至该代理程序。

如果此数字很高，并且查询失败（其 sqlcode 为 -968），同时 db2diad.log 中的消息指示 TEMP 表空间中的临时空间已用完，那么可能是表队列溢出造成的。这可能指示另一节点存在问题（如锁定）。应通过在所有分区上获取此查询的快照来进行调查。

还可能出现下列情况：因为数据的分区方式而使得许多缓冲区需要溢出以供该查询使用。在这类情况下，您需要为临时表空间添加更多磁盘。

tq_id_waiting_on -“在表队列的节点上等待”监视元素

等待发送或接收数据的表队列的标识。

表 1997. 快照监视信息

快照级别	逻辑数据分组	监视开关
应用程序	subsection	语句

用法

它可以用于故障诊断。

tq_max_send_spills -“最大表队列缓冲区溢出数”

溢出至临时表的最大表队列缓冲区数。

元素标识

tq_max_send_spills

元素类型

水位标记

表 1998. 快照监视信息

快照级别	逻辑数据分组	监视开关
应用程序	subsection	语句

表 1999. 事件监视信息

事件类型	逻辑数据分组	监视开关
语句	event_subsection	始终收集

用法 指示已写至临时表的最大表队列缓冲区数。

tq_node_waited_for -“在表队列上等待节点”

如果子节状态 ss_status 为*等待接收*或*等待发送*并且 tq_wait_for_any 为 FALSE，那么此项是此代理程序正在等待的节点的编号。

元素标识

tq_node_waited_for

元素类型

信息

表 2000. 快照监视信息

快照级别	逻辑数据分组	监视开关
应用程序	subsection	语句

用法 它可以用于故障诊断。您可能想要在子节正在等待的节点上获取应用程序快照。例如，应用程序在该节点上可能处于锁定等待状态。

tq_rows_read -“从表队列读取的行数”

从表队列读取的总行数。

元素标识

tq_rows_read

元素类型

计数器

表 2001. 快照监视信息

快照级别	逻辑数据分组	监视开关
应用程序	subsection	语句

表 2002. 事件监视信息

事件类型	逻辑数据分组	监视开关
语句	event_subsection	始终收集

用法 如果监视未指示此数字正在增加，那么表示处理未在进行。

如果此数字在节点间有很大差别，那么表示一些节点可能使用过度而另一些节点使用得不多。

如果此数字很大，那么表示节点间传递的数据很多，建议进行优化以改进存取方案。

tq_rows_written -“写至表队列的行数”

写至表队列的总行数。

元素标识

tq_rows_written

元素类型

计数器

表 2003. 快照监视信息

快照级别	逻辑数据分组	监视开关
应用程序	subsection	语句

表 2004. 事件监视信息

事件类型	逻辑数据分组	监视开关
语句	event_subsection	始终收集

用法 如果监视未指示此数字正在增加，那么表示处理未在进行。

如果此数字在节点间有很大差别，那么表示一些节点可能使用过度而另一些节点使用得不多。

如果此数字很大，那么表示节点间传递的数据很多，建议进行优化以改进存取方案。

tq_sort_heap_rejections -“表队列排序堆拒绝数”监视元素

表队列请求排序堆内存并因为超过排序堆阈值而被拒绝的次数。

表 2005. 表函数监视信息

表函数	监视元素收集级别
MON_FORMAT_XML_METRICS_BY_ROW - 获取所有度量值的基于已格式化行的输出	不适用; 报告 XML 文档中作为格式化函数的输入提供的所有元素
MON_GET_ACTIVITY_DETAILS 表函数 - 获取完整的活动详细信息	ACTIVITY METRICS BASE
MON_GET_CONNECTION 表函数 - 获取连接度量值	REQUEST METRICS BASE
MON_GET_CONNECTION_DETAILS 表函数 - 获取详细的连接度量值	REQUEST METRICS BASE
MON_GET_PKG_CACHE_STMT 表函数 - 获取程序包高速缓存中的 SQL 语句活动度量值	ACTIVITY METRICS BASE
MON_GET_PKG_CACHE_STMT_DETAILS - 获取程序包高速缓存条目的详细度量值	ACTIVITY METRICS BASE
MON_GET_SERVICE_SUBCLASS 表函数 - 获取服务子类度量值	REQUEST METRICS BASE
MON_GET_SERVICE_SUBCLASS_DETAILS 表函数 - 获取详细的服务子类度量值	REQUEST METRICS BASE
MON_GET_UNIT_OF_WORK 表函数 - 获取工作单元度量值	REQUEST METRICS BASE
MON_GET_UNIT_OF_WORK_DETAILS 表函数 - 获取详细的工作单元度量值	REQUEST METRICS BASE
MON_GET_WORKLOAD 表函数 - 获取工作负载度量值	REQUEST METRICS BASE
MON_GET_WORKLOAD_DETAILS 表函数 - 获取详细的工作负载度量值	REQUEST METRICS BASE

表 2006. 表函数监视信息

表函数	监视元素收集级别
MON_FORMAT_XML_METRICS_BY_ROW - 获取所有度量值的基于已格式化行的输出	不适用; 报告 XML 文档中作为格式化函数的输入提供的所有元素
MON_GET_ACTIVITY_DETAILS 表函数 - 获取完整的活动详细信息（在 XML 文档 DETAILS 中报告）	ACTIVITY METRICS BASE
MON_GET_CONNECTION 表函数 - 获取连接度量值	REQUEST METRICS BASE
MON_GET_CONNECTION_DETAILS 表函数 - 获取详细的连接度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE

表 2006. 表函数监视信息 (续)

表函数	监视元素收集级别
MON_GET_PKG_CACHE_STMT 表函数 - 获取程序包高速缓存中的 SQL 语句活动度量值	ACTIVITY METRICS BASE
MON_GET_PKG_CACHE_STMT_DETAILS 表函数 - 获取程序包高速缓存条目的详细度量值（在 XML 文档 DETAILS 中报告）	ACTIVITY METRICS BASE
MON_GET_SERVICE_SUBCLASS 表函数 - 获取服务子类度量值	REQUEST METRICS BASE
MON_GET_SERVICE_SUBCLASS_DETAILS 表函数 - 获取详细的服务子类度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE
MON_GET_UNIT_OF_WORK 表函数 - 获取工作单元度量值	REQUEST METRICS BASE
MON_GET_UNIT_OF_WORK 表函数 - 获取详细的工作单元度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE
MON_GET_WORKLOAD 表函数 - 获取工作负载度量值	REQUEST METRICS BASE
MON_GET_WORKLOAD_DETAILS 表函数 - 获取详细的工作负载度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE

表 2007. 事件监视信息

事件类型	逻辑数据分组	监视开关
活动	event_activity（在 details_xml 文档中报告）	ACTIVITY METRICS BASE
统计信息	event_scstats（在 details_xml 文档中报告）	REQUEST METRICS BASE
统计信息	event_wlstats（在 details_xml 文档中报告）	REQUEST METRICS BASE
工作单元	在 system_metrics 文档中报告。	REQUEST METRICS BASE
连接	event_conn	-
语句	event_stmt	-
事务	event_xact	-
程序包高速缓存	在 activity_metrics 文档中报告。	ACTIVITY METRICS BASE

用法

将此元素与 **tq_sort_heap_requests** 监视元素配合使用来确定大多数时间表队列是否获取足够的排序堆内存。如果 **tq_sort_heap_rejections** 监视元素与 **tq_sort_heap_requests** 监视元素的比率很高，那么数据库性能可能欠佳。考虑增加排序堆大小。

tq_sort_heap_requests -“表队列排序堆请求数”监视元素

表队列请求排序堆内存以存储数据的次数。

表 2008. 表函数监视信息

表函数	监视元素收集级别
MON_FORMAT_XML_METRICS_BY_ROW - 获取所有度量值的基于已格式化行的输出	不适用; 报告 XML 文档中作为格式化函数的输入提供的所有元素
MON_GET_ACTIVITY_DETAILS 表函数 - 获取完整的活动详细信息（在 XML 文档 DETAILS 中报告）	ACTIVITY METRICS BASE
MON_GET_CONNECTION 表函数 - 获取连接度量值	REQUEST METRICS BASE
MON_GET_CONNECTION_DETAILS 表函数 - 获取详细的连接度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE
MON_GET_PKG_CACHE_STMT 表函数 - 获取程序包高速缓存中的 SQL 语句活动度量值	ACTIVITY METRICS BASE
MON_GET_PKG_CACHE_STMT_DETAILS 表函数 - 获取程序包高速缓存条目的详细度量值（在 XML 文档 DETAILS 中报告）	ACTIVITY METRICS BASE
MON_GET_SERVICE_SUBCLASS 表函数 - 获取服务子类度量值	REQUEST METRICS BASE
MON_GET_SERVICE_SUBCLASS_DETAILS 表函数 - 获取详细的服务子类度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE
MON_GET_UNIT_OF_WORK 表函数 - 获取工作单元度量值	REQUEST METRICS BASE
MON_GET_UNIT_OF_WORK 表函数 - 获取详细的工作单元度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE
MON_GET_WORKLOAD 表函数 - 获取工作负载度量值	REQUEST METRICS BASE
MON_GET_WORKLOAD_DETAILS 表函数 - 获取详细的工作负载度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE

表 2009. 事件监视信息

事件类型	逻辑数据分组	监视开关
活动	event_activity（在 details_xml 文档中报告）	ACTIVITY METRICS BASE
统计信息	event_scstats（在 details_xml 文档中报告）	REQUEST METRICS BASE
统计信息	event_wlstats（在 details_xml 文档中报告）	REQUEST METRICS BASE
工作单元	在 system_metrics 文档中报告。	REQUEST METRICS BASE
连接	event_conn	-

表 2009. 事件监视信息 (续)

事件类型	逻辑数据分组	监视开关
语句	event_stmt	-
事务	event_xact	-
程序包高速缓存	在 activity_metrics 文档中报告。	ACTIVITY METRICS BASE

用法

将此元素与 **tq_sort_heap_rejections** 监视元素配合使用来确定大多数时间表队列是否获取足够的排序堆内存。如果 **tq_sort_heap_rejections** 监视元素与 **tq_sort_heap_requests** 监视元素的比率很高，那么数据库性能可能欠佳。考虑增加排序堆大小。

tq_tot_send_spills -“溢出表队列缓冲区总数”监视元素

溢出至临时表的表队列缓冲区的总数。

表 2010. 表函数监视信息

表函数	监视元素收集级别
MON_FORMAT_XML_METRICS_BY_ROW - 获取所有度量值的基于已格式化行的输出	不适用; 报告 XML 文档中作为格式化函数的输入提供的所有元素
MON_GET_ACTIVITY_DETAILS 表函数 - 获取完整的活动详细信息	ACTIVITY METRICS BASE
MON_GET_CONNECTION 表函数 - 获取连接度量值	REQUEST METRICS BASE
MON_GET_CONNECTION_DETAILS 表函数 - 获取详细的连接度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE
MON_GET_PKG_CACHE_STMT 表函数 - 获取程序包高速缓存中的 SQL 语句活动度量值	ACTIVITY METRICS BASE
MON_GET_PKG_CACHE_STMT_DETAILS 表函数 - 获取程序包高速缓存条目的详细度量值	ACTIVITY METRICS BASE
MON_GET_SERVICE_SUBCLASS 表函数 - 获取服务子类度量值	REQUEST METRICS BASE
MON_GET_SERVICE_SUBCLASS_DETAILS 表函数 - 获取详细的服务子类度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE
MON_GET_UNIT_OF_WORK 表函数 - 获取工作单元度量值	REQUEST METRICS BASE
MON_GET_UNIT_OF_WORK 表函数 - 获取详细的工作单元度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE
MON_GET_WORKLOAD 表函数 - 获取工作负载度量值	REQUEST METRICS BASE

表 2010. 表函数监视信息 (续)

表函数	监视元素收集级别
MON_GET_WORKLOAD_DETAILS 表函数 - 获取详细的工作负载度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE

表 2011. 快照监视信息

快照级别	逻辑数据分组	监视开关
应用程序	subsection	语句

表 2012. 事件监视信息

事件类型	逻辑数据分组	监视开关
活动	event_activity（在 details_xml 文档中报告）	ACTIVITY METRICS BASE
活动	event_activitymetrics	ACTIVITY METRICS BASE
统计信息	event_scstats（在 details_xml 文档中报告）	REQUEST METRICS BASE
统计信息	event_wlstats（在 details_xml 文档中报告）	REQUEST METRICS BASE
工作单元	在 system_metrics 文档中报告。	REQUEST METRICS BASE
语句	event_subsection	始终收集
程序包高速缓存	在 activity_metrics 文档中报告。	ACTIVITY METRICS BASE

用法

指示已写入临时表的表队列缓冲区的总数。有关更多信息，请参阅 **tq_cur_send_spills** 监视元素。

tq_wait_for_any -“在表队列上等待发送任何节点”

此标志用于指示子节已阻塞，原因是它正在等待从任何节点接收行。

元素标识

tq_wait_for_any

元素类型

信息

表 2013. 快照监视信息

快照级别	逻辑数据分组	监视开关
应用程序	subsection	语句

用法 如果 ss_status 指示*在表队列上等待接收数据*并且此标志为 TRUE，那么表示子节正等待从任何节点接收行。这通常指示 SQL 语句未处理至可将数据传递至等待代理程序的点。例如，写代理程序可能正在执行排序并且在排序完成之前不

会写入行。从 db2expln 输出来确定与表队列相关联的子节号，代理程序正在等待接收来自该表队列的行。然后可通过在执行子节的每个节点上获取快照来检查该子节的状态。

ts_name -“正在前滚的表空间”监视元素

目前已前滚的表空间的名称。

元素标识

ts_name

元素类型

信息

表 2014. 快照监视信息

快照级别	逻辑数据分组	监视开关
表空间	rollforward	基本

用法 如果正在进行前滚，那么此元素标识前滚涉及的表空间。

txn_completion_status -“事务完成状态”

此元素指示事务状态。

表 2015. 事件监视信息

事件类型	逻辑数据分组	监视开关
变更历史记录	TXNCOMPLETION	始终收集

用法

对于变更历史记录事件监视器，事务的状态为下列其中一种:

C 落实

R 回滚

S 回滚至保存点

uid_sql_stmts -“执行的 Update/Insert/Delete SQL 语句数”

执行的 SQL UPDATE、INSERT 和 DELETE 语句数。

表 2016. 快照监视信息

快照级别	逻辑数据分组	监视开关
数据库	dbase	基本
应用程序	appl	基本

可将快照监视的计数器重置。

表 2017. 事件监视信息

事件类型	逻辑数据分组	监视开关
数据库	event_db	始终收集
连接	event_conn	始终收集

用法 可使用此元素来确定应用程序或数据库级别的数据库活动的级别。

还可使用以下公式来确定 UPDATE、INSERT 和 DELETE 语句数与语句总数的比率:

```
    uid_sql_stmts
/ (static_sql_stmts + dynamic_sql_stmts )
```

此信息对于分析应用程序活动和吞吐量非常有用。

unread_prefetch_pages -“未读取的预取页数”监视元素

指示预取读入但从未使用的页数。

表 2018. 表函数监视信息

表函数	监视元素收集命令和级别
MON_GET_BUFFERPOOL 表函数 - 获取缓冲池度量值	DATA OBJECT METRICS BASE
MON_GET_TABLESPACE 表函数 - 获取表空间度量值	DATA OBJECT METRICS BASE

表 2019. 快照监视信息

快照级别	逻辑数据分组	监视开关
数据库	dbase	缓冲池
表空间	tablespace	缓冲池
缓冲池	bufferpool	缓冲池
应用程序	appl	缓冲池

可将快照监视的计数器重置。

表 2020. 事件监视信息

事件类型	逻辑数据分组	监视开关
数据库	event_db	始终收集
表空间	event_tablespace	始终收集
连接	event_conn	始终收集

用法

如果此数目很高，那么预取程序会将不会使用到的页读入缓冲池，从而导致执行不必要的 I/O。

uow_comp_status -“工作单元完成状态”

工作单元的状态以及停止方式。

元素标识

uow_comp_status

元素类型

信息

表 2021. 快照监视信息

快照级别	逻辑数据分组	监视开关
应用程序	appl	工作单元
DCS 应用程序	dcs_appl	基本

表 2022. 事件监视信息

事件类型	逻辑数据分组	监视开关
事务	event_xact	始终收集

用法 可使用此元素来确定工作单元是否因为死锁或异常终止而结束。它可能已经:

- 因为落实语句而落实
- 因为回滚语句而回滚
- 因为死锁而回滚
- 因为异常终止而回滚
- 在正常应用程序终止时落实。
- 因为对正在运行的工作单元执行 FLUSH EVENT MONITOR 命令而处于未知状态。

注: API 用户应参考包含数据库系统监视器常量的定义的头文件（*sqlmon.h*）。

uow_completed_total -“完成的工作单元总数”监视元素

通过落实或回滚完成的工作单元总数。

表 2023. 表函数监视信息

表函数	监视元素收集级别
MON_SAMPLE_SERVICE_CLASS_METRICS - 获取样本服务类度量值	REQUEST METRICS BASE
MON_SAMPLE_WORKLOAD_METRICS - 获取样本工作负载度量值	REQUEST METRICS BASE
WLM_GET_SERVICE_SUBCLASS_STATS 表函数 - 返回服务子类的统计信息	REQUEST METRICS BASE
WLM_GET_WORKLOAD_STATS 表函数 - 返回工作负载统计信息	REQUEST METRICS BASE

表 2024. 事件监视信息

事件类型	逻辑数据分组	监视开关
统计信息	event_scstats（在 details_xml 文档中报告）	始终收集
统计信息	event_wlstats（在 details_xml 文档中报告）	始终收集

用法

如果由 WLM_GET_SERVICE_SUBCLASS_STATS 或 WLM_GET_WORKLOAD_STATS 函数返回此监视元素，那么此监视元素表示自上次重置统计信息以来完成的工作单元总数。

如果由 MON_SAMPLE_SERVICE_CLASS_METRICS 或 MON_SAMPLE_WORKLOAD_METRICS 函数返回此监视元素，那么此监视元素表示自执行该函数以来完成的工作单元总数。

uow_elapsed_time -“最新工作单元耗用时间”

最新完成的工作单元耗用的执行时间。

元素标识

uow_elapsed_time

元素类型

时间

表 2025. 快照监视信息

快照级别	逻辑数据分组	监视开关
应用程序	appl	工作单元，时间戳记
DCS 应用程序	dcs_appl	工作单元，时间戳记

用法

将此元素用作完成工作单元所花时间的指示符。

此元素由两个子元素组成，它们报告耗用时间的秒数和微秒（一秒的百万分之一）数。这些子元素的名称可通过将“_s”和“_ms”添加至此监视元素的名称派生而成。要检索此监视元素耗用的总时间，必须将这两个子元素的值加在一起。例如，如果“_s”子元素值为 3，“_ms”子元素值为 20，那么此监视元素耗用的总时间为 3.00002 秒。

uow_id -“工作单元标识”监视元素

工作单元标识。工作单元标识在应用程序句柄内是唯一的。

表 2026. 表函数监视信息

表函数	监视元素收集级别
MON_GET_ACTIVITY_DETAILS 表函数 - 获取完整的活动详细信息	ACTIVITY METRICS BASE

表 2026. 表函数监视信息 (续)

表函数	监视元素收集级别
MON_GET_UNIT_OF_WORK 表函数 - 获取工作单元度量值	ACTIVITY METRICS BASE
MON_GET_UNIT_OF_WORK_DETAILS 表函数 - 获取详细的工作单元度量值	ACTIVITY METRICS BASE
WLM_GET_SERVICE_CLASS_AGENTS 表函数 - 列示正在服务类中运行的代理程序	ACTIVITY METRICS BASE
WLM_GET_SERVICE_CLASS_WORKLOAD_OCCURRENCES 表函数 - 列示工作负载出现次数	ACTIVITY METRICS BASE
WLM_GET_WORKLOAD_OCCURRENCE_ACTIVITIES 表函数 - 返回活动列表	ACTIVITY METRICS BASE

表 2027. 事件监视信息

事件类型	逻辑数据分组	监视开关
锁定	-	始终收集
工作单元	-	始终收集
活动	event_activity	始终收集
活动	event_activitystmt	始终收集
活动	event_activityvals	始终收集
活动	event_activitymetrics	ACTIVITY METRICS BASE
阈值违例	event_thresholdviolations	始终收集
变更历史记录	ddlstmtexec txncompletion	始终收集

用法

将此元素与其他活动历史元素配合使用来分析活动的行为。

还可以将此元素与 **activity_id** 和 **appl_id** 监视元素配合使用来唯一地标识某项活动。

uow_lifetime_avg -“工作单元平均生存期”监视元素

工作单元的平均生存期。此监视元素以毫秒计。

表 2028. 表函数监视信息

表函数	监视元素收集级别
MON_SAMPLE_SERVICE_CLASS_METRICS - 获取样本服务类度量值	REQUEST METRICS BASE
MON_SAMPLE_WORKLOAD_METRICS - 获取样本工作负载度量值	REQUEST METRICS BASE
WLM_GET_SERVICE_SUBCLASS_STATS 表函数 - 返回服务子类的统计信息	REQUEST METRICS BASE
WLM_GET_WORKLOAD_STATS 表函数 - 返回工作负载统计信息	REQUEST METRICS BASE

表 2029. 事件监视信息

事件类型	逻辑数据分组	监视开关
统计信息	event_scstats（在 details_xml 文档中报告）	始终收集
统计信息	event_wlstats（在 details_xml 文档中报告）	始终收集

用法

如果由 WLM_GET_SERVICE_SUBCLASS_STATS 或 WLM_GET_WORKLOAD_STATS 函数返回此监视元素，那么此监视元素表示自上次重置统计信息以来的工作单元平均生存期。

如果由 MON_SAMPLE_SERVICE_CLASS_METRICS 或 MON_SAMPLE_WORKLOAD_METRICS 函数返回此监视元素，那么此监视元素表示自执行该函数以来的工作单元平均生存期。

uow_lock_wait_time -“工作单元等待锁定的总时间”监视元素

此工作单元等待锁定时的耗用时间总计。此值以毫秒计。

元素标识

uow_lock_wait_time

元素类型

计数器

表 2030. 快照监视信息

快照级别	逻辑数据分组	监视开关
应用程序	appl	工作单元

用法 此元素可帮助您确定资源争用问题的严重性。

uow_log_space_used -“使用的工作单元日志空间”监视元素

被监视应用程序的当前工作单元中使用的日志空间量（以字节计）。

表 2031. 表函数监视信息

表函数	监视元素收集级别
MON_GET_UNIT_OF_WORK 表函数 - 获取工作单元度量值	ACTIVITY METRICS BASE
MON_GET_UNIT_OF_WORK 表函数 - 获取详细的工作单元度量值（在 XML 文档 DETAILS 中报告）	ACTIVITY METRICS BASE

表 2032. 快照监视信息

快照级别	逻辑数据分组	监视开关
应用程序	appl	工作单元

表 2033. 事件监视信息

事件类型	逻辑数据分组	监视开关
事务	event_xact	始终收集
工作单元	-	始终收集

用法

可使用此元素来了解工作单元级别的记录要求。

uow_start_time -“工作单元开始时间戳记”监视元素

工作单元第一次需要数据库资源的日期和时间。

表 2034. 表函数监视信息

表函数	监视元素收集级别
MON_GET_UNIT_OF_WORK 表函数 - 获取工作单元度量值	ACTIVITY METRICS BASE
MON_GET_UNIT_OF_WORK 表函数 - 获取详细的工作单元度量值（在 XML 文档 DETAILS 中报告）	ACTIVITY METRICS BASE

表 2035. 快照监视信息

快照级别	逻辑数据分组	监视开关
应用程序	appl	工作单元，时间戳记
DCS 应用程序	dcs_appl	工作单元，时间戳记

表 2036. 事件监视信息

事件类型	逻辑数据分组	监视开关
工作单元	-	-
事务	event_xact	-

用法

此资源要求出现在该工作单元的第一个 SQL 语句执行中:

- 对于第一个工作单元，这是 **conn_complete_time** 之后的第一个数据库请求（SQL 语句执行）的时间。
- 对于后续工作单元，这是上一个 COMMIT 或 ROLLBACK 之后第一个数据库请求（SQL 语句执行）的时间。

注: *SQL Reference* 将工作单元的边界定义为 COMMIT 或 ROLLBACK 点。

数据库系统监视器排除 COMMIT/ROLLBACK 与工作单元定义中的下一个 SQL 语句之间所花的时间。此量度方法反映数据库管理器在处理数据库请求时所花的时间，并且将此时间与该工作单元的第一个 SQL 语句之间的应用程序逻辑所花的时间隔开。工作单元耗用时间包括在工作单元内的 SQL 语句之间运行应用程序逻辑所花的时间。

可将此元素与 **uow_stop_time** 监视元素配合使用来计算工作单元的耗用时间总计，并与 **prev_uow_stop_time** 监视元素配合使用来计算在工作单元之间的应用程序上耗用的时间。

可以使用 **uow_stop_time** 和 **prev_uow_stop_time** 监视元素来计算工作单元的 *SQL Reference* 定义的耗用时间。

uow_status -“工作单元状态”

工作单元的状态。

元素标识

uow_status

元素类型

信息

表 2037. 事件监视信息

事件类型	逻辑数据分组	监视开关
事务	event_xact	始终收集

用法 可使用此元素来确定工作单元的状态。API 用户应参考包含数据库系统监视器常量定义的 sqlmon.h 头文件。

uow_stop_time -“工作单元停止时间戳记”监视元素

最新工作单元的完成日期和时间，该工作单元将在落实或回滚数据库更改时完成。

表 2038. 快照监视信息

快照级别	逻辑数据分组	监视开关
应用程序	appl	工作单元，时间戳记
DCS 应用程序	dcs_appl	工作单元，时间戳记

表 2039. 事件监视信息

事件类型	逻辑数据分组	监视开关
工作单元	-	-

用法

通过将此元素与 **prev_uow_stop_time** 监视元素配合使用，可以计算 COMMIT/ROLLBACK 点之间的耗用时间总计，通过将此元素与 **uow_start_time** 监视元素配合使用，可以计算最后一个工作单元的耗用时间。

时间戳记内容的设置将为如下所示:

- 如果应用程序已完成一个工作单元并且尚未启动新的工作单元（由 **uow_start_time** 监视元素定义），那么此元素将报告有效的非零时间戳记。
- 如果应用程序当前在执行工作单元，那么此元素将报告零。
- 当应用程序第一次连接至数据库时，此元素将设置为 **conn_complete_time** 监视元素的值。

当新的工作单元启动时，此元素的内容将移至 **prev_uow_stop_time** 监视元素。

uow_throughput -“工作单元吞吐量”监视元素

工作单元完成速率，以每秒工作单元数计。

表 2040. 表函数监视信息

表函数	监视元素收集级别
MON_SAMPLE_SERVICE_CLASS_METRICS - 获取样本服务类度量值	REQUEST METRICS BASE
MON_SAMPLE_WORKLOAD_METRICS - 获取样本工作负载度量值	REQUEST METRICS BASE
WLM_GET_SERVICE_SUBCLASS_STATS 表函数 - 返回服务子类的统计信息	REQUEST METRICS BASE
WLM_GET_WORKLOAD_STATS 表函数 - 返回工作负载统计信息	REQUEST METRICS BASE

表 2041. 事件监视信息

事件类型	逻辑数据分组	监视开关
统计信息	event_scstats（在 details_xml 文档中报告）	始终收集
统计信息	event_wlstats（在 details_xml 文档中报告）	始终收集

用法

如果由 WLM_GET_SERVICE_SUBCLASS_STATS 或 WLM_GET_WORKLOAD_STATS 函数返回此监视元素，那么此监视元素表示自上次重置统计信息以来的工作单元吞吐量。

如果由 MON_SAMPLE_SERVICE_CLASS_METRICS 或 MON_SAMPLE_WORKLOAD_METRICS 函数返回此监视元素，那么此监视元素表示自执行该函数以来的工作单元吞吐量。

uow_total_time_top -“最长 UOW 时间总计”监视元素

工作单元生存期的高水位标记（以毫秒计）。

表 2042. 表函数监视信息

表函数	监视元素收集级别
WLM_GET_SERVICE_SUBCLASS_STATS 表函数 - 返回服务子类的统计信息	REQUEST METRICS BASE
WLM_GET_WORKLOAD_STATS 表函数 - 返回工作负载统计信息	REQUEST METRICS BASE

表 2043. 事件监视信息

事件类型	逻辑数据分组	监视开关
统计信息	event_wlstats	始终收集
统计信息	event_scstats	始终收集

用法

此元素可用来帮助确定 UOWTOTALTIME 阈值是否有效，还可以帮助确定如何配置这种阈值。

对于服务类，当服务类的 COLLECT AGGREGATE ACTIVITY DATA 设置为 NONE 时，此监视元素返回 -1。

对于工作负载而言，如果工作负载的 COLLECT AGGREGATE ACTIVITY DATA 设置为 NONE，那么此监视元素将返回 -1。

对于服务类，将计算由工作负载指定的服务类的此高水位标记采用的度量。工作操作集用来更改活动的服务类的任何映射都不会影响此高水位标记。

update_sql_stmts -“更新数”

此元素包含自联合服务器实例启动或数据库监视计数器最后一次重置以后，联合服务器代表任何应用程序对此数据源发出 UPDATE 语句的总次数。

表 2044. 快照监视信息

快照级别	逻辑数据分组	监视开关
数据库	dbase_remote	基本
应用程序	appl_remote	基本

可将快照监视的计数器重置。

用法 使用此元素来确定联合服务器或应用程序对此数据源执行的数据库活动的级别。

还可使用此元素并借助以下公式来确定联合服务器或应用程序对此数据源执行的写入活动的百分比:

```
write_activity =
        (INSERT 语句数 + UPDATE 语句数 + DELETE 语句数 ) /
        (SELECT 语句数 + INSERT 语句数 + UPDATE 语句数 +
        DELETE 语句数)
```

update_time -“更新响应时间”

此元素包含此数据源响应 UPDATE 所花的总时间（以毫秒计），这些 UPDATE 来自联合服务器启动后或数据库监视计数器上一次重置后在此联合服务器实例上运行的所有应用程序或单个应用程序。

表 2045. 快照监视信息

快照级别	逻辑数据分组	监视开关
数据库	dbase_remote	时间戳记

表 2045. 快照监视信息 (续)

快照级别	逻辑数据分组	监视开关
应用程序	appl_remote	时间戳记

可将快照监视的计数器重置。

响应时间是以联合服务器将 UPDATE 语句提交给数据源的时间与数据源响应联合服务器以指示已处理 UPDATE 的时间之差量度的。

用法 使用此元素来确定等待处理对此数据源执行 UPDATE 语句所花的实际时间。此信息对于容量规划和容量调整非常有用。

usage_list_last_state_change -“最近一次状态更改”监视元素

用于指示最近一次更改 **usage_list_state** 监视元素值的时间的时间戳记。

表 2046. 表函数监视信息

表函数	监视元素收集级别
MON_GET_USAGE_LIST_STATUS 表函数 - 返回用法列表的状态	ACTIVITY METRICS BASE

usage_list_last_updated -“用法列表最近一次更新时间”监视元素

用于指示最近一次更新特定部分的时间的时间戳记。此部分由 **executable_id** 和 **mon_interval_id** 监视元素的值表示。

表 2047. 表函数监视信息

表函数	监视元素收集级别
MON_GET_INDEX_USAGE_LIST 表函数 - 返回索引用法列表中的信息	ACTIVITY METRICS BASE
MON_GET_TABLE_USAGE_LIST 表函数 - 从表用法列表返回信息	ACTIVITY METRICS BASE

usage_list_mem_size -“用法列表内存大小”监视元素

为特定用法列表分配的内存总量（以千字节计）。

表 2048. 表函数监视信息

表函数	监视元素收集级别
MON_GET_USAGE_LIST_STATUS 表函数 - 返回用法列表的状态	ACTIVITY METRICS BASE

usage_list_name -“用法列表名称”监视元素

用法列表名称。

表 2049. 表函数监视信息

表函数	监视元素收集级别
MON_GET_INDEX_USAGE_LIST 表函数 - 返回索引用法列表中的信息	ACTIVITY METRICS BASE
MON_GET_TABLE_USAGE_LIST 表函数 - 从表用法列表返回信息	ACTIVITY METRICS BASE
MON_GET_USAGE_LIST_STATUS 表函数 - 返回用法列表的状态	ACTIVITY METRICS BASE

usage_list_num_references -“引用数”监视元素

自特定部分添加至用法列表后该部分引用特定对象的总次数。

表 2050. 表函数监视信息

表函数	监视元素收集级别
MON_GET_INDEX_USAGE_LIST 表函数 - 返回索引用法列表中的信息	ACTIVITY METRICS BASE
MON_GET_TABLE_USAGE_LIST 表函数 - 从表用法列表返回信息	ACTIVITY METRICS BASE

usage_list_num_ref_with_metrics -“带有度量值的引用数”监视元素

自特定部分添加至用法列表并已收集统计信息后该部分引用特定对象的总次数。

表 2051. 表函数监视信息

表函数	监视元素收集级别
MON_GET_INDEX_USAGE_LIST 表函数 - 返回索引用法列表中的信息	ACTIVITY METRICS BASE
MON_GET_TABLE_USAGE_LIST 表函数 - 从表用法列表返回信息	ACTIVITY METRICS BASE

usage_list_schema -“用法列表模式”监视元素

用法列表模式的名称。

表 2052. 表函数监视信息

表函数	监视元素收集级别
MON_GET_INDEX_USAGE_LIST 表函数 - 返回索引用法列表中的信息	ACTIVITY METRICS BASE
MON_GET_TABLE_USAGE_LIST 表函数 - 从表用法列表返回信息	ACTIVITY METRICS BASE
MON_GET_USAGE_LIST_STATUS 表函数 - 返回用法列表的状态	ACTIVITY METRICS BASE

usage_list_size -“用法列表大小”监视元素

特定用法列表可容纳的最大条目数。

表 2053. 表函数监视信息

表函数	监视元素收集级别
MON_GET_USAGE_LIST_STATUS 表函数 - 返回用法列表的状态	ACTIVITY METRICS BASE

usage_list_state -“用法列表状态”监视元素

特定用法列表的状态。

可能的值如下所示:

A 处于活动状态。

F 未能激活。

I 处于不活动状态。

P 激活暂挂。

表 2054. 表函数监视信息

表函数	监视元素收集级别
MON_GET_USAGE_LIST_STATUS 表函数 - 返回用法列表的状态	ACTIVITY METRICS BASE

usage_list_used_entries -“用法列表已使用条目数”监视元素

用法列表中当前已使用的条目数。如果用法列表处于不活动状态，那么此监视元素表示此用法列表上次激活以进行监视时此用法列表中的条目数。

表 2055. 表函数监视信息

表函数	监视元素收集级别
MON_GET_USAGE_LIST_STATUS 表函数 - 返回用法列表的状态	ACTIVITY METRICS BASE

usage_list_wrapped -“用法列表合并指示符”监视元素

说明特定用法列表是否已合并的指示符。用法列表变满时，缺省行为是合并条目，这意味着最旧的条目被最新的条目替换。

可能的值为 Y 和 N。

表 2056. 表函数监视信息

表函数	监视元素收集级别
MON_GET_USAGE_LIST_STATUS 表函数 - 返回用法列表的状态	ACTIVITY METRICS BASE

user_cpu_time -“用户 CPU 时间”监视元素

数据库管理器代理进程、工作单元或语句使用的总*用户* CPU 时间（以秒和微秒计）。对于写至表的事件监视器，此元素的值将通过使用 BIGINT 数据类型以微秒为单位给定。

当监视开关或时间戳记开关未打开时，将不收集此元素，并改为写入 -1。

表 2057. 事件监视信息

事件类型	逻辑数据分组	监视开关
连接	event_conn	始终收集
事务	event_xact	始终收集
语句	event_stmt	始终收集
活动	event_activity	始终收集

用法

此元素与其他相关 CPU 时间元素一起使用可帮助您了解应用程序内的活动级别，还可以帮助您标识可能因为调整而受益的应用程序。

注: 如果此信息对您的操作系统不可用，那么此元素将设置为 0。

注: 由于 DB2 系统收集统计信息时所使用的详细程度不同，**total_exec_time** 监视元素的值可能与 **system_cpu_time** 和 **user_cpu_time** 监视元素的值的总和不相等。在此情况下，**system_cpu_time** 与 **user_cpu_time** 监视元素的值之和更准确地反映了实际总执行时间。

utility_dbname -“实用程序操作的数据库”

实用程序操作的数据库。

表 2058. 快照监视信息

快照级别	逻辑数据分组	监视开关
数据库管理器	utility_info	基本

utility_description -“实用程序描述”

对实用程序执行的任务的简短描述。例如，重新平衡调用可能包含“Tablespace ID: 2”，表示此重新平衡程序正在处理标识为 2 的表空间。此字段的格式取决于实用程序的类，并且在发行版之间可能更改。

表 2059. 快照监视信息

快照级别	逻辑数据分组	监视开关
数据库管理器	utility_info	基本

utility_detail -“实用程序详细信息”

此元素包含实用程序正在执行的工作的简短描述。

表 2060. 事件监视信息

事件类型	逻辑数据分组	监视开关
变更历史记录	UTILSTART	始终收集

用法

实用程序正执行的工作的简短描述，包括对该实用程序指定的一些选项。例如，针对 REORG 调用的记录将包括部分重构的命令字符串，包括该实用程序使用的某些不同选项，例如，访问方式。此字段的格式取决于实用程序的类型，并且在不同发行版中可能变化。

utility_id -“实用程序标识”

对应于实用程序调用的唯一标识。

表 2061. 表函数监视信息

表函数	监视元素收集命令和级别
MON_GET_ACTIVITY_DETAILS 表函数 - 获取完整的活动详细信息（在 XML 文档 DETAILS 中报告）	ACTIVITY METRICS BASE
WLM_GET_WORKLOAD_OCCURRENCE_ACTIVITIES 表函数 - 返回活动列表	ACTIVITY METRICS BASE

表 2062. 快照监视信息

快照级别	逻辑数据分组	监视开关
数据库管理器	utility_info	基本

utility_invocation_id -“实用程序调用标识”

对应于实用程序调用的唯一标识。

表 2063. 事件监视信息

事件类型	逻辑数据分组	监视元素收集级别
变更历史记录	changesummary utillocation utilphase utilstart utilstop	始终收集

用法

utility_invocation_id 实用程序是二进制标记，用于唯一标识实用程序的给定调用。**utility_invocation_id** 在执行该实用程序的每个成员上相同。**utility_invocation_id** 在数据库取消激活、重新激活和成员关闭期间保持其唯一性，以允许快速标识对应实用程序的给定调用的所有事件监视器记录。

utility_invoker_type -“实用程序调用者类型”

此元素描述实用程序是如何被调用的。

表 2064. 快照监视信息

快照级别	逻辑数据分组	监视开关
数据库管理器	utility_info	基本

表 2065. 事件监视信息

事件类型	逻辑数据分组	监视开关
变更历史记录	utilstart	始终收集

用法

使用此元素来确定实用程序是如何被调用的。例如，可以使用此元素来确定实用程序是 DB2 自动调用的还是用户调用的。此元素的值（列示如下）是在 sqlmon.h 中定义的。

API 常量	实用程序
SQLM_UTILITY_INVOKER_USER	实用程序是用户调用的
SQLM_UTILITY_INVOKER_AUTO	实用程序是 DB2 自动调用的

对于变更事件历史记录监视器，此元素指示如何调用实用程序:

USER 实用程序由用户调用

AUTO 实用程序是 DB2 自动调用的

utility_operation_type -“实用程序操作类型”

指示实用程序操作的类型。

表 2066. 事件监视信息

事件类型	逻辑数据分组	监视开关
变更历史记录	UTILSTART	始终收集

用法

对于变更历史记录事件监视器，此元素包含有关启动的实用程序事件 (UTILITY_TYPE) 的详细信息。

如果 UTILITY_TYPE 为 BACKUP，那么为下列其中一个:

D 变化量

I 增量

F 完全

如果 UTILITY_TYPE 为 LOAD，那么为下列其中一个:

I 插入

R 替换

S 重新启动

T 终止

如果 UTILITY_TYPE 为 MOVETABLE，那么为下列其中一个:

A 取消

C 复制

I 初始化

L 清除

M 移动

R 重演

S 交换

V 验证

如果 UTILITY_TYPE 为 REDISTRIBUTE，那么为下列其中一个:

A 中止

C 继续

D 缺省值

T 目标映射

如果 UTILITY_TYPE 为 REORG，那么为下列其中一个:

A 重组所有表索引

I 索引重组

N 原地表重组

R 重组表回收扩展数据块

T 典型表重组

如果 UTILITY_TYPE 为 RESTORE，那么为下列其中一个:

A 增量自动

B 增量中止

F 完全

M 增量手动

如果 UTILITY_TYPE 为 ROLLFORWARD，那么为下列其中一个:

E 日志结束

P 时间点

如果 UTILITY_TYPE 为 RUNSTATS，那么为下列其中一个:

A 表的所有索引

I 索引

T 表

utility_phase_detail -“实用程序阶段详细信息”

此元素已保留以供将来使用。

表 2067. 事件监视信息

事件类型	逻辑数据分组	监视开关
变更历史记录	UTILPHASE	始终收集

utility_phase_type -“实用程序阶段类型”

标识实用程序阶段类型。

表 2068. 事件监视信息

事件类型	逻辑数据分组	监视开关
变更历史记录	UTILPHASE	始终收集

用法

对于变更历史记录事件监视器，如果 `utility_type` 元素为 BACKUP，那么阶段类型为:

BACKUPTS
备份表空间

utility_priority -“实用程序优先级”

实用程序优先级指定调速实用程序相对其调速层而言的相对重要性。优先级 0 指示实用程序以非调速方式执行。非零优先级的范围必须在 1 到 100 之间，100 表示最高优先级，1 表示最低优先级。

表 2069. 快照监视信息

快照级别	逻辑数据分组	监视开关
数据库管理器	utility_info	基本

表 2070. 事件监视信息

事件类型	逻辑数据分组	监视开关
变更历史记录	utilstart	始终收集

utility_start_time -“实用程序启动时间”

初次调用当前实用程序的日期和时间。

表 2071. 快照监视信息

快照级别	逻辑数据分组	监视开关
数据库管理器	utility_info	基本

utility_start_type -“实用程序启动类型”

此元素指示实用程序如何启动。

表 2072. 事件监视信息

事件类型	逻辑数据分组	监视开关
变更历史记录	UTILSTART	始终收集

用法

对于变更历史记录事件监视器，实用程序将以下列其中一种方式启动:

- RESUME
- START

utility_state -“实用程序状态”

此元素描述实用程序的状态。

元素标识

utility_state

元素类型

信息

表 2073. 快照监视信息

快照级别	逻辑数据分组	监视开关
数据库管理器	utility_info	基本

用法 使用此元素来确定活动实用程序的状态。此字段的值（列示如下）是在 sqlmon.h 中定义的。

API 常量	描述
SQLM_UTILITY_STATE_EXECUTE	实用程序正在执行
SQLM_UTILITY_STATE_WAIT	实用程序正在等待事件发生，然后才会继续执行
SQLM_UTILITY_STATE_ERROR	实用程序遇到了错误

utility_stop_type -“实用程序停止类型”

此元素指示如何停止实用程序。

表 2074. 事件监视信息

事件类型	逻辑数据分组	监视开关
变更历史记录	UTILSTOP	始终收集

用法

对于变更历史记录事件监视器，实用程序将以下列其中一种方式停止:

- PAUSE
- STOP

valid -“节有效性指示器”监视元素

指示动态 SQL 语句节是否有效。对于静态 SQL 语句，此监视元素的值始终为 Y。

表 2075. 表函数监视信息

表函数	监视元素收集级别
MON_GET_PKG_CACHE_STMT 表函数 - 获取程序包高速缓存中的 SQL 语句活动度量值	ACTIVITY METRICS BASE
MON_GET_PKG_CACHE_STMT_DETAILS 表函数 - 获取程序包高速缓存条目的详细度量值	ACTIVITY METRICS BASE

用法

此监视元素的有效值是 Y 和 N。有效节在下次被使用时将由系统以隐式方式准备。

utility_type -“实用程序类型”

实用程序的类。

表 2076. 快照监视信息

快照级别	逻辑数据分组	监视开关
数据库管理器	utility_info	基本

表 2077. 事件监视信息

事件类型	逻辑数据分组	监视元素收集级别
变更历史记录	changesummary utillocation utilphase utilstart utilstop	始终收集

用法

此元素的值可以是在 sqlmon.h 中定义的名称以“SQLM_UTILITY_”开头的任何常量。

对于变更历史记录事件监视器，实用程序类型是下列其中的一个:

- BACKUP
- LOAD
- MOVETABLE
- REDISTRIBUTE
- REORG
- RESTORE
- ROLLFORWARD
- RUNSTATS

valid -“节有效性指示器”监视元素

指示动态 SQL 语句节是否有效。对于静态 SQL 语句，此监视元素的值始终为 Y。

表 2078. 表函数监视信息

表函数	监视元素收集级别
MON_GET_PKG_CACHE_STMT 表函数 - 获取程序包高速缓存中的 SQL 语句活动度量值	ACTIVITY METRICS BASE
MON_GET_PKG_CACHE_STMT_DETAILS 表函数 - 获取程序包高速缓存条目的详细度量值	ACTIVITY METRICS BASE

用法

此监视元素的有效值是 Y 和 N。有效节在下次被使用时将由系统以隐式方式准备。

vectored_ios -“向量 I/O 请求数”监视元素

向量 I/O 请求的数目。更具体地说，就是 DB2 在缓冲池的页区域中执行页预取的次数。

表 2079. 表函数监视信息

表函数	监视元素收集级别
MON_GET_BUFFERPOOL 表函数 - 获取缓冲池度量值	DATA OBJECT METRICS BASE
MON_GET_CONTAINER 表函数 - 获取表空间容器度量值	DATA OBJECT METRICS BASE
MON_GET_TABLESPACE 表函数 - 获取表空间度量值	DATA OBJECT METRICS BASE

表 2080. 快照监视信息

快照级别	逻辑数据分组	监视开关
缓冲池	bufferpool	缓冲池

用法

使用此元素来确定执行向量 I/O 的频率。仅在预取期间才监视向量 I/O 请求的数目。

version -“监视器数据版本”

产生事件监视器数据流的数据库管理器版本。

表 2081. 事件监视信息

事件类型	逻辑数据分组	监视开关
事件日志头	event_log_header	-

用法

事件监视器使用的数据结构在数据库管理器发行版之间可能更改。因此，监视器应用程序应检查数据流版本以确定它们能否处理要接收的数据。

对于此发行版，此元素设置为 API 常量 SQLM_DBMON_VERSION9_5。

virtual_mem_free -“可用虚拟内存”监视元素

此主机上可用但未分配至任何进程的虚拟内存量，以 MB 计。

表 2082. 表函数监视信息

表函数	监视元素收集级别
ENV_GET_SYSTEM_RESOURCES 表函数 - 返回系统信息	ACTIVITY METRICS BASE

virtual_mem_reserved -“保留虚拟内存”监视元素

正在运行的进程保留的虚拟内存量，以 MB 计。

表 2083. 表函数监视信息

表函数	监视元素收集级别
ENV_GET_SYSTEM_RESOURCES 表函数 - 返回系统信息	ACTIVITY METRICS BASE

virtual_mem_total -“总虚拟内存”监视元素

此主机上可用的虚拟内存总量，以 MB 计。

表 2084. 表函数监视信息

表函数	监视元素收集级别
ENV_GET_SYSTEM_RESOURCES 表函数 - 返回系统信息	ACTIVITY METRICS BASE

wl_work_action_set_id -“工作负载工作操作集标识”监视元素

如果此活动已划分到工作负载作用域的某个工作类，那么此监视元素显示与该工作类所属的工作类集相关联的工作操作集的标识。否则，此监视元素将显示值 0。

表 2085. 表函数监视信息

表函数	监视元素收集级别
MON_GET_ACTIVITY_DETAILS 表函数 - 获取完整的活动详细信息（在 XML 文档 DETAILS 中报告）	ACTIVITY METRICS BASE

表 2086. 事件监视信息

事件类型	逻辑数据分组	监视开关
活动	event_activity	始终收集

用法

可将此监视元素与 **wl_work_class_id** 监视元素配合使用，以唯一地标识活动的工作负载工作类（如果有）。

wl_work_class_id -“工作负载工作类标识”监视元素

如果此活动已划分到工作负载作用域的某个工作类，那么此监视元素将显示该工作类的标识。否则，此监视元素将显示值 0。

表 2087. 表函数监视信息

表函数	监视元素收集级别
MON_GET_ACTIVITY_DETAILS 表函数 - 获取完整的活动详细信息（在 XML 文档 DETAILS 中报告）	ACTIVITY METRICS BASE

表 2088. 事件监视信息

事件类型	逻辑数据分组	监视开关
活动	event_activity	始终收集

用法

可将此监视元素与 **wl_work_action_set_id** 监视元素配合使用，以唯一地标识活动的工作负载工作类（如果有）。

wlm_queue_assignments_total -“工作负载管理器队列分配总次数”监视元素

WLM 阈值对活动进行排队的次数。

表 2089. 表函数监视信息

表函数	监视元素收集级别
MON_FORMAT_XML_METRICS_BY_ROW - 获取所有度量值的基于已格式化行的输出	不适用; 报告 XML 文档中作为格式化函数的输入提供的所有元素
MON_GET_ACTIVITY_DETAILS 表函数 - 获取完整的活动详细信息（在 XML 文档 DETAILS 中报告）	ACTIVITY METRICS BASE
MON_GET_CONNECTION 表函数 - 获取连接度量值	REQUEST METRICS BASE
MON_GET_CONNECTION_DETAILS 表函数 - 获取详细的连接度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE
MON_GET_PKG_CACHE_STMT 表函数 - 获取程序包高速缓存中的 SQL 语句活动度量值	ACTIVITY METRICS BASE
MON_GET_PKG_CACHE_STMT_DETAILS 表函数 - 获取程序包高速缓存条目的详细度量值	ACTIVITY METRICS BASE
MON_GET_SERVICE_SUBCLASS 表函数 - 获取服务子类度量值	REQUEST METRICS BASE
MON_GET_SERVICE_SUBCLASS_DETAILS 表函数 - 获取详细的服务子类度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE

表 2089. 表函数监视信息 (续)

表函数	监视元素收集级别
MON_GET_UNIT_OF_WORK 表函数 - 获取工作单元度量值	REQUEST METRICS BASE
MON_GET_UNIT_OF_WORK 表函数 - 获取详细的工作单元度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE
MON_GET_WORKLOAD 表函数 - 获取工作负载度量值	REQUEST METRICS BASE
MON_GET_WORKLOAD_DETAILS 表函数 - 获取详细的工作负载度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE

表 2090. 事件监视信息

事件类型	逻辑数据分组	监视开关
活动	event_activity（在 details_xml 文档中报告）	ACTIVITY METRICS BASE
活动	event_activitymetrics	ACTIVITY METRICS BASE
统计信息	event_scstats（在 details_xml 文档中报告）	REQUEST METRICS BASE
统计信息	event_wlstats（在 details_xml 文档中报告）	REQUEST METRICS BASE
程序包高速缓存	在 activity_metrics 文档中报告。	ACTIVITY METRICS BASE
工作单元	在 system_metrics 文档中报告。	REQUEST METRICS BASE

wlm_queue_time_total -“工作负载管理器队列时间总计”监视元素

等待 WLM 队列阈值时耗用的时间。此值以毫秒计。

表 2091. 表函数监视信息

表函数	监视元素收集级别
MON_FORMAT_XML_METRICS_BY_ROW - 获取所有度量值的基于已格式化行的输出	不适用; 报告 XML 文档中作为格式化函数的输入提供的所有元素
MON_FORMAT_XML_TIMES_BY_ROW - - 获取基于已格式化行的组合层次结构等待时间和处理时间	不适用; 报告 XML 文档中作为格式化函数的输入提供的所有元素
MON_FORMAT_XML_WAIT_TIMES_BY_ROW - 获取等待时间的已格式化的基于行的输出	不适用; 报告 XML 文档中作为格式化函数的输入提供的所有元素
MON_GET_ACTIVITY_DETAILS 表函数 - 获取完整的活动详细信息（在 XML 文档 DETAILS 中报告）	ACTIVITY METRICS BASE
MON_GET_CONNECTION 表函数 - 获取连接度量值	REQUEST METRICS BASE

表 2091. 表函数监视信息 (续)

表函数	监视元素收集级别
MON_GET_CONNECTION_DETAILS 表函数 - 获取详细的连接度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE
MON_GET_PKG_CACHE_STMT 表函数 - 获取程序包高速缓存中的 SQL 语句活动度量值	ACTIVITY METRICS BASE
MON_GET_PKG_CACHE_STMT_DETAILS 表函数 - 获取程序包高速缓存条目的详细度量值	ACTIVITY METRICS BASE
MON_GET_SERVICE_SUBCLASS 表函数 - 获取服务子类度量值	REQUEST METRICS BASE
MON_GET_SERVICE_SUBCLASS_DETAILS 表函数 - 获取详细的服务子类度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE
MON_GET_UNIT_OF_WORK 表函数 - 获取工作单元度量值	REQUEST METRICS BASE
MON_GET_UNIT_OF_WORK 表函数 - 获取详细的工作单元度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE
MON_GET_WORKLOAD 表函数 - 获取工作负载度量值	REQUEST METRICS BASE
MON_GET_WORKLOAD_DETAILS 表函数 - 获取详细的工作负载度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE

表 2092. 事件监视信息

事件类型	逻辑数据分组	监视开关
活动	event_activity（在 details_xml 文档中报告）	ACTIVITY METRICS BASE
活动	event_activitymetrics	ACTIVITY METRICS BASE
统计信息	event_scstats（在 details_xml 文档中报告）	REQUEST METRICS BASE
统计信息	event_wlstats（在 details_xml 文档中报告）	REQUEST METRICS BASE
程序包高速缓存	在 activity_metrics 文档中报告。	ACTIVITY METRICS BASE
工作单元	在 system_metrics 文档中报告。	REQUEST METRICS BASE

wlo_completed_total -“完成的工作负载项总数”监视元素

自最后一次重置后完成的工作负载项数。

表 2093. 表函数监视信息

表函数	监视元素收集级别
WLM_GET_WORKLOAD_STATS 表函数 - 返回工作负载统计信息	ACTIVITY METRICS BASE

表 2094. 事件监视信息

事件类型	逻辑数据分组	监视开关
统计信息	event_wlstats	-

用法

使用此元素来确定向该系统添加工作的给定工作负载项个数。

work_action_set_id -“工作操作集标识”监视元素

此统计信息记录适用的工作操作集的标识。

表 2095. 事件监视信息

事件类型	逻辑数据分组	监视开关
统计信息	event_histogrambin	始终收集
统计信息	event_wcstats	始终收集

用法

将此元素与其他活动历史元素配合使用来分析活动的行为，或者与其他统计信息元素配合使用来分析工作类。

满足以下条件时，此元素的值为 0:

- 在 event_histogrambin 逻辑数据组中报告了该元素。
- 针对工作类以外的对象收集了直方图数据。

work_action_set_name -“工作操作集名称”监视元素

与作为此事件的一部分显示的统计信息相关联的工作操作集的名称。

表 2096. 表函数监视信息

表函数	监视元素收集级别
WLM_GET_QUEUE_STATS 表函数 - 返回阈值队列统计信息	ACTIVITY METRICS BASE
WLM_GET_WORK_ACTION_SET_STATS 表函数 - 返回工作操作集统计信息	ACTIVITY METRICS BASE

表 2097. 事件监视信息

事件类型	逻辑数据分组	监视开关
统计信息	event_qstats	-
统计信息	event_wcstats	-

用法

将此元素与 **work_class_name** 元素配合使用，以唯一地标识此记录中显示了其统计信息的工作类或唯一地标识属于此记录中显示了其统计信息的阈值队列的域的工作类。

work_class_id -“工作类标识”监视元素

此统计信息记录适用的工作类的标识。

表 2098. 事件监视信息

事件类型	逻辑数据分组	监视开关
统计信息	event_wcstats	始终收集
统计信息	event_histogrambin	始终收集

用法

将此元素与其他统计信息元素配合使用来分析工作类。

满足以下条件时，此元素的值为 0:

- 在 event_histogrambin 逻辑数据组中报告了该元素。
- 针对工作类以外的对象收集了直方图数据。

work_class_name -“工作类名”监视元素

与作为此事件的一部分显示的统计信息相关联的工作类的名称。

表 2099. 表函数监视信息

表函数	监视元素收集级别
WLM_GET_QUEUE_STATS 表函数 - 返回阈值队列统计信息	ACTIVITY METRICS BASE
WLM_GET_WORK_ACTION_SET_STATS 表函数 - 返回工作操作集统计信息	ACTIVITY METRICS BASE

表 2100. 事件监视信息

事件类型	逻辑数据分组	监视开关
统计信息	event_qstats	-
统计信息	event_wcstats	-

用法

将此元素与 **work_action_set_name** 元素配合使用，以唯一地标识此记录中显示了其统计信息的工作类或唯一地标识属于此记录中显示了其统计信息的阈值队列的域的工作类。

workload_id -“工作负载标识”监视元素

一个用来唯一地标识工作负载的整数。

表 2101. 表函数监视信息

表函数	监视元素收集级别
MON_GET_WORKLOAD 表函数 - 获取工作负载度量值	ACTIVITY METRICS BASE
MON_GET_WORKLOAD_DETAILS 表函数 - 获取详细的工作负载度量值	ACTIVITY METRICS BASE
MON_SAMPLE_WORKLOAD_METRICS - 获取样本	ACTIVITY METRICS BASE

表 2102. 快照监视信息

快照级别	逻辑数据分组	监视开关
应用程序	appl_info	基本

表 2103. 事件监视信息

事件类型	逻辑数据分组	监视开关
锁定	-	始终收集
工作单元	-	始终收集
活动	event_activity（在 details_xml 文档中报告）	ACTIVITY METRICS BASE
统计信息	event_scstats（在 details_xml 文档中报告）	REQUEST METRICS BASE
统计信息	event_wlstats（在 details_xml 文档中报告）	REQUEST METRICS BASE
工作单元	在 system_metrics 文档中报告。	始终收集
统计信息	event_wlstats	始终收集
统计信息	event_histogrambin	始终收集
活动	event_activity	始终收集

用法

使用此标识来唯一地指定此活动、应用程序、直方图条形或工作负载统计信息记录所属的工作负载。

满足以下条件时，此元素的值为 0:

- 在 event_histogrambin 逻辑数据组中报告了该元素。
- 针对工作负载以外的对象收集了直方图数据。

workload_name -“工作负载名称”监视元素

工作负载的名称。

表 2104. 表函数监视信息

表函数	监视元素收集级别
MON_FORMAT_LOCK_NAME 表函数 - 设置内部锁定名称的格式并返回详细信息	ACTIVITY METRICS BASE
MON_GET_UNIT_OF_WORK 表函数 - 获取工作单元度量值	ACTIVITY METRICS BASE
MON_GET_UNIT_OF_WORK_DETAILS 表函数 - 获取详细的工作单元度量值	ACTIVITY METRICS BASE
MON_GET_WORKLOAD 表函数 - 获取工作负载度量值	ACTIVITY METRICS BASE
MON_GET_WORKLOAD_DETAILS 表函数 - 获取详细的工作负载度量值	ACTIVITY METRICS BASE
MON_SAMPLE_WORKLOAD_METRICS - 获取样本	ACTIVITY METRICS BASE
WLM_GET_QUEUE_STATS 表函数 - 返回阈值队列统计信息	ACTIVITY METRICS BASE
WLM_GET_SERVICE_CLASS_AGENTS 表函数 - 列示正在服务类中运行的代理程序	ACTIVITY METRICS BASE
WLM_GET_SERVICE_CLASS_WORKLOAD_OCCURRENCES 表函数 - 列示工作负载出现次数	ACTIVITY METRICS BASE
WLM_GET_WORKLOAD_STATS 表函数 - 返回工作负载统计信息	ACTIVITY METRICS BASE

表 2105. 事件监视信息

事件类型	逻辑数据分组	监视开关
锁定	-	始终收集
工作单元	-	始终收集
活动	event_activity（在 details_xml 文档中报告）	ACTIVITY METRICS BASE
统计信息	event_scstats（在 details_xml 文档中报告）	REQUEST METRICS BASE
统计信息	event_wlstats（在 details_xml 文档中报告）	REQUEST METRICS BASE
工作单元	在 system_metrics 文档中报告。	始终收集
统计信息	event_wlstats	始终收集

用法

在统计信息事件监视器和工作负载表函数中，工作负载名称用于标识正在为其收集并报告统计信息和度量值的工作负载。在工作单元事件监视器和工作单元表函数中，工作负载名称用于标识与工作单元相关联的工作负载。

请使用工作负载名称来标识适用于您感兴趣的特定工作负载的工作单元或信息集。

workload_occurrence_id -“工作负载项标识”监视元素

此活动所属的工作负载项的标识。

表 2106. 表函数监视信息

表函数	监视元素收集级别
MON_GET_UNIT_OF_WORK 表函数 - 获取工作单元度量值	ACTIVITY METRICS BASE
MON_GET_UNIT_OF_WORK_DETAILS 表函数 - 获取详细的工作单元度量值	ACTIVITY METRICS BASE
WLM_GET_SERVICE_CLASS_AGENTS 表函数 - 列示正在服务类中运行的代理程序	ACTIVITY METRICS BASE
WLM_GET_SERVICE_CLASS_WORKLOAD_OCCURRENCES 表函数 - 列示工作负载出现次数	ACTIVITY METRICS BASE

表 2107. 事件监视信息

事件类型	逻辑数据分组	监视开关
工作单元	-	始终收集
活动	event_activity	始终收集

用法

使用它来标识已提交活动的工作负载项。

workload_occurrence_state -“工作负载实例状态”监视元素

工作负载实例的状态。

表 2108. 表函数监视信息

表函数	监视元素收集级别
MON_GET_UNIT_OF_WORK 表函数 - 获取工作单元度量值	ACTIVITY METRICS BASE
MON_GET_UNIT_OF_WORK 表函数 - 获取详细的工作单元度量值（在 XML 文档 DETAILS 中报告）	ACTIVITY METRICS BASE
WLM_GET_SERVICE_CLASS_WORKLOAD_OCCURRENCES 表函数 - 列示工作负载出现次数	ACTIVITY METRICS BASE

用法

可能的值包括:

DECOUPLED

未对此工作负载实例指定协调代理程序（集中器情况）。

DISCONNECTPEND
此工作负载实例正在与数据库断开连接。

FORCED
此工作负载实例已被强制终止并除去。

INTERRUPTED
此工作负载实例已被中断。

QUEUED
工作负载管理队列阈值已对此工作负载实例协调代理程序进行排队。在分区数据库环境中，此状态可能表明协调代理程序使 RPC 的另一成员获取阈值凭单并且尚未接收到响应。

TRANSIENT
尚未将此工作负载实例映射到服务超类。

UOWEXEC
此工作负载实例正在处理请求。

UOWWAIT
此工作负载实例正在等待来自客户机的请求。

x_lock_escals -“互斥锁定升级数”监视元素

锁定从若干行锁定升级至一个互斥表锁定的次数，或者针对行的互斥锁定导致表锁定成为互斥锁定的次数。

表 2109. 快照监视信息

快照级别	逻辑数据分组	监视开关
数据库	dbase	基本
应用程序	appl	基本

可将快照监视的计数器重置。

表 2110. 事件监视信息

事件类型	逻辑数据分组	监视开关
数据库	event_db	始终收集
连接	event_conn	始终收集
事务	event_xact	始终收集

用法

其他应用程序不能访问互斥锁定挂起的数据；因此需要跟踪互斥锁定，原因是它们会影响数据的并行性。

当应用程序挂起的锁定总数达到可供应用程序使用的最大锁定列表空间量时，锁定将会升级。可用锁定列表空间量由 **locklist** 和 **maxlocks** 配置参数确定。

当应用程序达到允许的最大锁定数并且没有其他要升级的锁定时，它将使用锁定列表中为其他应用程序分配的空间。当整个锁定列表已满时，将发生错误。

有关可能会导致出现过量互斥锁定升级的原因及其解决方案，请参阅 **lock_escals** 监视元素。

当共享锁定已足够时，应用程序可能会使用互斥锁定。尽管共享锁定可能不会降低锁定升级总数，但共享锁定升级可能比互斥锁定升级更好一些。

xda_object_pages -“XDA 对象页数”

XML 存储器对象（XDA）数据消耗的磁盘页数。

元素标识

xda_object_pages

元素类型

信息

表 2111. 快照监视信息

快照级别	逻辑数据分组	监视开关
表	表	基本

表 2112. 事件监视信息

事件类型	逻辑数据分组	监视开关
表	event_table	始终收集

用法 此元素提供了一种机制，可用来查看特定表中的 XML 存储器对象（XDA）数据消耗的实际空间量。此元素可与表事件监视器配合使用以跟踪一段时间内 XML 存储器对象数据的增长率。

xda_object_l_pages -“XML 存储器对象 (XDA) 数据逻辑页数”监视元素

XML 存储器对象 (XDA) 数据在磁盘上使用的逻辑页数。

表 2113. 表函数监视信息

表函数	监视元素收集级别
MON_GET_TABLE 表函数 - 获取表度量值	ACTIVITY METRICS BASE

用法

- 此值可能小于实际上为该对象分配的空间量。使用 TRUNCATE 语句的 REUSE STORAGE 选项时，可能会发生这种情况。此选项导致为该表分配的存储空间继续被分配，尽管该存储空间被视为空。此外，此监视元素的值可能小于逻辑上为该对象分配的空间量，因为逻辑上分配的空间总量包括少量附加元数据。

  要检索对象的逻辑大小或物理大小的准确度量，请使用 ADMIN_GET_TAB_INFO_V97 函数。此函数提供的有关对象大小的信息比您可（通过将针对此监视元素报告的页数乘以页大小）获取的对象大小更准确。

xid -“事务标识”

事务管理器在两阶段落实事务中生成的唯一事务标识（在所有数据库上）。

表 2114. 快照监视信息

快照级别	逻辑数据分组	监视开关
DCS 应用程序	dcs_appl	工作单元

用法 此标识可用来将事务管理器生成的事务与对多个数据库执行的事务相关联。它还可用来帮助诊断事务管理器问题，方法是将涉及两阶段落实协议的数据库事务与事务管理器生成的事务关联在一起。

xmlid -“XML 标识”监视元素

唯一文档标识。此标识是按如下所示派生的：
<event_header>_<event_id>_<event_type>_<event_timestamp>_<partition>。

表 2115. 表函数监视信息

表函数	监视元素收集级别
EVMON_FORMAT_UE_TO_TABLES 过程 - 将 XML 文档移至关系表	不适用；报告 XML 文档中作为格式化函数的输入提供的所有元素
EVMON_FORMAT_UE_TO_XML 表函数 - 将无格式事件转换为 XML	不适用；报告 XML 文档中作为格式化函数的输入提供的所有元素

表 2116. 事件监视信息

事件类型	逻辑数据分组	监视开关
锁定		
程序包高速缓存		

xquery_stmts -“尝试的 XQuery 语句数”

已为应用程序或数据库执行的 XQuery 语句。

元素标识

xquery_stmts

元素类型

计数器

表 2117. 快照监视信息

快照级别	逻辑数据分组	监视开关
数据库	dbase	基本
应用程序	appl	基本

可将快照监视的计数器重置。

表 2118. 事件监视信息

事件类型	逻辑数据分组	监视开关
数据库	event_db	始终收集
连接	event_conn	始终收集

用法 可使用此元素来测量本地 XQuery 语言请求的活动。这不包括嵌入式 XQuery 语言请求，例如，xmlquery、xmltable 或 xmlexist。

第 3 部分 在 DB2 pureScale 环境中进行监视

IBM DB2 pureScale Feature 提供了功能强大的高可用数据库处理环境。DB2 pureScale 实例中的一个或多个主机系统的操作可能产生的问题通常可在不中断数据访问的情况下解决。可惜的是，DB2 pureScale 环境中的这一高可用性特征会掩盖可能导致低于最佳性能的问题。监视 DB2 pureScale 环境的特定方面可帮助您识别和解决这类问题。

例如，可能存在导致成员或集群高速缓存设施（又称为 CF）重复故障转移至另一主机的硬件问题。但是，因为恢复在大多数情况下是自动进行的，所以您可能永远不会注意到此问题。如果此问题未被检测到也未被更正，那么您将无法知道 DB2 pureScale 环境可能达到的最佳性能。

因此，建议进行某种级别的不间断的操作性监视 DB2 pureScale 实例。这有助于您回答如下问题:

- 我的 DB2 pureScale实例的所有组件都在运行吗？
- 如果成员或 CF 出现故障，那么它能否成功重新启动？
- 我的 CF 是否正在其首选主要主机上运行？或者它是否已故障转移至另一主机？
- 另一 CF 是否处于准备好在主 CF 失败时接管的状态？

要查看 DB2 pureScale 环境的整体状态，最好从检查实例中的主机、成员和 CF 的操作状态开始。通过检查针对其中每个实体报告的状态和警报信息，可整体了解 DB2 pureScale实例的工作情况。

您不但可以查看 DB2 pureScale实例的整体状态，还可使用 DB2 监视基础结构来检查监视元素，这些监视元素提供有关 DB2 pureScale实例的特定方面的信息。此信息可帮助您更好地了解可解决哪些配置问题和应用程序设计问题来提高整体系统性能。

第 12 章 DB2 pureScale实例的状态监视

查看 DB2 pureScale实例中的组件整体状态可让您了解它的工作效率。可使用表函数和管理视图来查看此信息。还有一些 DB2 命令行处理器 (CLP) 命令和系统命令可用来显示实例的运行状态。

从这些表函数、视图和命令返回的信息可让您深入了解 DB2 pureScale实例的状态。例如，可使用这些界面来查找类似如下的问题的答案:

- 组成 DB2 pureScale实例的主机是否处于活动状态？
- 哪些主机正充当集群高速缓存设施或成员服务器？
- 是否有主机正同时充当成员和集群高速缓存设施？
- 是否有主机正运行多个成员？
- 是否有成员正在其首选原始主机上运行？
- 对于具有多个集群高速缓存设施的配置，哪一个在充当主服务器？另一个非主集群高速缓存设施的当前状态如何，是 PEER 还是 CATCHUP？
- 组成实例的成员的当前*状态*如何？例如，是否使用 **db2stop** 命令停止了任何成员？或者，是否有任何成员当前在等待故障恢复至其原始主机？
- 是否有任何集群高速缓存设施或成员服务器指示需要调查的活动*警报*？

其中许多这些界面返回的信息中包括*状态*和*警报*信息。主机、成员或集群高速缓存设施的状态反映所涉及的对象的当前运行能力。例如，成员的状态可以是 STARTED、STOPPED、RESTARTING、WAITING_FOR_FAILBACK、ERROR 或 UNKNOWN。

警报用于指示某个或某些对象需要进一步调查。如果针对实例中的一个或多个集群高速缓存设施或成员产生了警报，那么可能存在需要调查的问题。

用于检索 DB2 pureScale实例的状态信息的界面

要查看 DB2 pureScale实例中的各种组件的整体状态，可从若干管理视图、表函数和 CLP 命令中进行选择。此外，可在系统命令提示符或 shell 中使用一些命令，没有任何服务器组件在运行时，这些命令特别有用。

可通过以下方式来查看有关 DB2 pureScale实例的状态和警报信息:

- 『表函数和管理视图』
- 第 1352 页的『CLP 命令』
- 第 1353 页的『系统提示符 (shell) 命令』.

表函数和管理视图

表函数提供灵活的界面来检索有关系统的信息。大多数表函数接受用于缩小所返回信息范围的参数。例如，您可能想要查看有关特定主机的信息。

管理视图使您能够快速轻松地访问系统信息。与表函数不同，您不能向管理视图传递参数来缩小查询范围。一般来说，会返回系统中与此视图有关的所有对象（例如，主机、集群高速缓存设施（又称为 CF）或成员）的信息。但是，始终可使用 SQL 来过滤管理视图的输出。

以下表函数及其对应管理视图返回有关 DB2 pureScale实例的整体状态的信息:

表 2119. 用于显示 DB2 pureScale实例中的组件的状态信息的表函数和管理视图

界面	描述
DB2_GET_CLUSTER_HOST_STATE 表函数 DB2_CLUSTER_HOST_STATE 管理视图	这些界面提供有关组成 DB2 pureScale实例的主机的基本信息。它们返回主机及关联状态信息的列表。
DB2_GET_INSTANCE_INFO 表函数 DB2_MEMBER 管理视图 DB2_CF 管理视图	这些界面提供有关 DB2 pureScale实例的更多详细信息。它们返回实例中有关每个主机充当的角色（集群高速缓存设施或成员）的信息，不管每个集群高速缓存设施或成员是否在其原始主机上运行以及每个主机的连接信息。
DB2_INSTANCE_ALERTS 管理视图	此界面提供有关 DB2 pureScale 实例中的警报的信息。

注: 可在 DB2 pureScale实例和其他DB2实例中使用先前界面中的每一个。它们对每个界面返回的结果可能会不同。例如，DB2_MEMBER 管理视图可用于这两种类型的实例；但是，在 DB2 pureScale 环境外部的实例中，所返回信息中未包含任何状态或警报信息。必须记住，应在正查询实例类型的上下文中解释从函数和视图返回的结果。有关详细信息，请参阅每个特定表函数或管理视图的参考主题。

CLP 命令

可在 DB2 命令行处理器 (CLP) 中使用以下命令:

表 2120. 用于显示 DB2 pureScale实例中的组件的状态信息的 CLP 命令

CLP 命令	描述
LIST INSTANCE	此命令返回有关成员、主机和集群高速缓存设施的状态的信息。
LIST INSTANCE SHOW DETAIL	此命令是 LIST INSTANCE 命令的扩展，它会返回添加的信息，包括成员、主机和集群高速缓存设施的分区号和连接信息。

系统提示符 (shell) 命令

可通过系统提示符或 shell 提示符使用以下命令:

表 2121. 用于显示 DB2 pureScale实例中的组件的信息的系统提示符 (shell) 命令

命令	描述
db2instance -list	此命令返回有关 DB2 pureScale实例中的成员、主机和集群高速缓存设施的状态信息。即使没有当前数据库连接或者实例已停止，仍可使用此命令。在后一情况下，**db2instance -list** 命令会使用集群管理器来报告有关 DB2 pureScale实例中的主机的信息。有一些选项可用于将输出限制为仅成员或仅集群高速缓存设施。
db2cluster -list *options*	此命令可用于查看有关 DB2 pureScale实例的信息。对于此命令，有一些其他选项可供选择；指定 **-list** 选项时，还必须指定其他选项以指定要包括在命令输出中的内容。

成员和集群高速缓存设施状态和警报的值

可用于查询 DB2 pureScale 环境中的组件状态的许多表函数、管理视图和命令返回*状态*和*警报*信息。主机、成员或集群高速缓存设施（又称为 CF）的*状态*反映其运行状态。主机、成员或 CF 的*警报*指示存在可能需要调查或干预的问题。

主机、成员和集群高速缓存设施的状态

状态信息由可用于查询 DB2 pureScale 环境的组件状态的许多表函数、管理视图和命令返回。表 2122中显示了每个组件的状态的可能值:

表 2122. 主机、成员和集群高速缓存设施的可能状态

组件	可能的状态	描述
主机	ACTIVE	主机可供使用。这意味着主机系统正在运行，并且可响应操作系统或联网命令，例如，TCP/IP **ping** 命令。
	INACTIVE	主机不可用。这意味着主机系统未在运行，不可用或者未响应系统命令。处于此状态的原因是主机电源断电或者连接或联网问题。

表 2122. 主机、成员和集群高速缓存设施的可能状态 (续)

组件	可能的状态	描述
集群高速缓存设施 (CF)	STOPPED	管理员已在正常关闭时使用 **db2stop** 命令手动停止了 CF。
	RESTARTING	通过执行 **db2start** 命令或在 CF 故障后，CF 处于启动过程。
	BECOMING_ PRIMARY	CF 启动后，如果没有其他 CF 在实例中已承担主 CF 的角色，那么它会尝试承担此角色。
	PRIMARY	CF 作为主 CF 正常运行。
	CATCHUP (*n*%)	备份 CF 一开始已启动，它未包含主 CF 中的任何信息。在 CATCHUP 状态期间，备份 CF 处于从主 CF 获取所有相关信息的副本的过程中。此信息允许它在主 CF 失败的情况下承担主 CF 的角色。*n*% 指示备份 CF 处于从主 CF 复制信息的过程中。当此复制过程完成时，备份 CF 会转为 PEER 状态。 **注**: 使用 **db2instance -list** 命令来查看非主 CF 的状态时，它将处于 **CATCHUP** 状态直到建立与数据库的连接。进行第一个连接后，从主 CF 复制数据的过程开始。
	PEER	备份 CF 准备在主 CF 发生故障时接管主 CF 负责的工作。备份 CF 处于 PEER 状态时，双工会继续。
	ERROR	DB2 集群服务未能自动重新启动 CF。CF 反映 ERROR 状态时，ALERT 字段始终设置为 YES，以指示需要管理员进行干预或调查。DB2 集群服务不再尝试在 CF 处于 ERROR 状态后将其重新启动，除非已清除警报。 如果无法建立与 CF 的连接来查询其状态，那么也可能出现 ERROR 状态。在此情况下，ALERT 字段未设置为 YES，因为此问题可能是暂时的。
成员	STARTED	成员已在实例中启动并且正以正常方式运行。所有数据库处于一致状态，并且成员准备接受或已经接受与数据库的连接。如果成员失败并再次启动，那么可能过程模型已启动，但数据库的崩溃恢复尚未完成。通过任何成员使用 **LIST UTILITIES SHOW DETAIL** 命令来监视恢复进度。
	STOPPED	管理员已在正常关闭时使用 **db2stop** 命令手动停止了成员。
	RESTARTING	成员处于启动或重新启动的过程中。如果当前主机与原始主机相同，那么会进行本地成员重新启动。如果当前主机与原始主机不同，那么成员会故障转移至当前主机，并以轻量级方式重新启动。
	WAITING_FOR_ FAILBACK	此成员的过程模型已在当前主机上以轻量级方式成功重新启动。此成员正在等待其原始主机变为可用，原始主机可用时它在原始主机上进行故障恢复。通过任何活动成员将 **LIST UTILITIES** 命令与 SHOW DETAIL 选项配合使用来监视恢复进度，并了解所有数据库的崩溃恢复是否完成。此成员不接受任何新连接，也不处理任何事务。不确定事务可能仍然存在。
	ERROR	DB2 集群服务未能在其原始主机或 DB2 pureScale实例中的任何其他主机上自动重新启动成员。成员反映 ERROR 状态时，ALERT 字段始终设置为 YES，以指示需要管理员进行干预或调查。DB2 集群服务不再尝试在成员处于 ERROR 状态后将其重新启动，除非已清除警报。

主机、成员和集群高速缓存设施的警报

除了返回状态信息外，查询 DB2 pureScale 环境中的组件状态的命令也会返回警报信息。所有组件的警报的可能值为 YES 或 NO。一般来讲，警报值 NO 指示一切以正常方式运行。警报值 YES 指示存在可能需要手动干预的问题。在某些情况下，报警条件是暂

时的，警报字段可能会清除自身，例如，在重新引导主机时。在其他情况下，警报字段会保留设置直到管理员解决此问题并使用带有 -clear -alert 选项的 **db2cluster** 命令来重设警报字段。

状态信息的解释

查询主机、成员或集群高速缓存设施以获取状态信息时，系统会提供状态和警报信息，以告诉您有关 DB2 pureScale 环境中的各种组件的状态。出现问题时，通常需要检查状态和警报以了解系统中所进行的操作。

主机、成员或集群高速缓存设施（又称为 CF）的*状态*反映其运行状态。一切对象正常操作时，针对主机状态、成员和集群高速缓存设施（又称为 CF）报告的值可让您大致了解系统的状态。例如，成员上的 RESTARTING 或 WAITING_FOR_FAILBACK 状态本身不指示存在问题。可能会有一些有效原因使得成员故障转移至新主机或在其原始主机上重新启动，例如，主机因为维护而脱机时。如果成员频繁重复进行故障转移，那么可能存在需要进一步调查的问题。

主机、成员或 CF 的*警报*指示存在可能需要调查或干预的问题。查看给定系统组件的状态的上下文中的警报可找到有关问题来源的其他信息。下面的部分概述您可能遇到的有关主机、成员或集群高速缓存设施的状态和警报信息的各种组合以及如何解释不同的状态和警报组合。

切记：由此信息中报告的界面返回的状态和警报信息的完整性取决于以下因素：

- 运行表函数、管理视图或命令的实例（例如，DB2 pureScale实例或其他 DB2 实例）的类型
- 是否在该实例中使用了受支持的集群管理器。所有 DB2 pureScale Feature 部署都使用集群管理器。

请参阅第 1358 页的『针对数据共享和非 DB2 pureScale 环境的报告中的差别』以了解详细信息。

主机状态

可使用若干不同界面查看有关 DB2 pureScale 环境中的主机的信息。DB2_CLUSTER_HOST_STATE 管理视图就是一个这样的界面。例如，考虑以下 SQL 查询：

```
SELECT varchar(HOSTNAME,10) AS HOST,
       varchar(STATE,8) AS STATE,
       varchar(INSTANCE_STOPPED,7) AS STOPPED,
       ALERT
FROM SYSIBMADM.DB2_CLUSTER_HOST_STATE
```

运行先前 SQL 语句的输出将如下所示：

```
HOST       STATE    STOPPED ALERT
---------- -------- ------- --------
HOSTD      ACTIVE   NO      NO
HOSTB      ACTIVE   NO      NO
HOSTA      ACTIVE   YES     NO
HOSTC      ACTIVE   NO      NO

  4 record(s) selected.
```

（在先前示例中，STOPPED 列对应于管理视图返回的 INSTANCE_STOPPED 列。）

根据任意给定时间的情况，状态、instance_stopped 和警报列可具有不同值。表 2123 中概述了可能的值。

表 2123. DB2 pureScale 实例中的主机系统上可能的状态、instance_stopped 和警报的组合

STATE	INSTANCE_ STOPPED	ALERT	描述
ACTIVE	NO	NO	主机处于活动状态并且正以正常方式运行。
		YES	主机处于活动状态（即，它响应系统命令），但可能存在阻止它参与 DB2 pureScale 实例的问题。例如，可能存在文件系统问题或网络通信问题，或者 DB2 pureScale Feature 执行故障转移时需要的空闲进程可能未在运行。
	YES	NO	主机处于活动状态。管理员已使用 **db2stop instance on** *hostname* 命令在此主机上显式停止了此实例。
		YES	主机处于活动状态，但是，此主机存在未清除的警报。管理员已显式停止此实例。
INACTIVE	NO	NO	不适用。INSTANCE_STOPPED 和 ALERT 设置为 NO 时，主机不能为 INACTIVE 状态。
		YES	主机未在响应系统命令。管理员未显式停止此实例，但存在警报。此状态信息组合指示主机异常关闭。例如，可能由于主机上的电源故障而导致这类关闭。
	YES	NO	管理员停止实例时的正常状态。主机因为安装软件更新而处于脱机状态时，可能会产生这样的状态信息组合。
		YES	主机未在响应系统命令。主机存在未清除的警报，但管理员已显式停止此实例（即，系统未异常关闭）。

提示: 可使用 DB2_INSTANCE_ALERTS 管理视图来查看有关警报的详细信息。

成员状态

可使用若干不同界面来查看成员状态和警报。DB2_MEMBER 管理视图就是一个这样的界面。DB2_MEMBER 管理视图显示 DB2 pureScale 实例中的成员的状态信息。后跟说明如何使用此管理视图来检索成员状态的示例:

```
SELECT ID,
       varchar(STATE,21) AS STATE,
       varchar(HOME_HOST,10) AS HOME_HOST,
       varchar(CURRENT_HOST,10) AS CUR_HOST,
       ALERT
FROM SYSIBMADM.DB2_MEMBER
```

根据任意给定时间的情况，状态和警报列可具有不同值。表 2124中概述了可能的值。

表 2124. DB2 pureScale 实例中的成员的可能状态和警报组合

STATE	ALERT	描述
STARTED	NO	成员已在实例中启动并且正以正常方式运行。
	YES	成员已在实例中启动。但是，在某一时刻，尝试故障转移至另一主机但未成功。自未成功的故障转移尝试以来，成员已能够成功故障转移至另一主机，或者它已故障恢复至其原始主机。如果此成员在其原始主机上运行，那么它以正常方式运行；如果它在访客主机上运行，那么它以轻量级方式运行。在任一情况下，都应调查警报以确定所发生的情况。

表 2124. DB2 pureScale 实例中的成员的可能状态和警报组合 (续)

STATE	ALERT	描述
STOPPED	NO	管理员已使用 **db2stop** 命令停止了成员。
	YES	管理员已使用 **db2stop** 命令停止了成员，但是，尚未清除警报字段。
RESTARTING	NO	成员正在启动。
	YES	成员正在启动。但是，在某一时刻，尝试在原始主机上启动该成员或故障转移至另一主机但未成功。尚未清除警报字段。
WAITING_FOR_ FAILBACK	NO	成员正以轻量级方式在访客主机上运行，并且正在等待故障恢复至原始主机。您可能想要检查原始主机的状态以了解是否有任何因素阻止成员故障恢复至原始主机（例如，失败的网络适配器）。
	YES	尝试在原始主机上重新启动成员可能已失败、自动故障恢复已禁用或崩溃恢复可能已失败。您需要先解决此问题并手动清除警报，成员才能自动故障恢复至其原始主机。如果自动故障恢复被禁用，请手动清除警报并使用 **db2cluster** 命令来启用自动故障恢复。
ERROR	YES	DB2 集群服务无法在任何主机上启动成员。您需要先解决此问题并手动清除警报，然后尝试重新启动此实例。

提示: 可使用 DB2_INSTANCE_ALERTS 管理视图来查看有关警报的详细信息。

集群高速缓存设施状态

DB2_GET_INSTANCE_INFO 表函数允许您检索 DB2 pureScale实例中的成员的状态信息。表函数的优点之一是您可将参数传递至表函数以缩小所返回结果的范围。例如，要检索有关 DB2 pureScale 实例中 CF 的信息，可构造如下查询:

```
SELECT ID,
       varchar(STATE,17) AS STATE,
       varchar(HOME_HOST,10) AS HOME_HOST,
       varchar(CURRENT_HOST,10) AS CUR_HOST,
       ALERT
FROM TABLE(DB2_GET_INSTANCE_INFO(NULL,'','','CF',NULL))
```

根据任意给定时间的情况，状态和警报列可具有不同值。表 2125中概述了可能的值。

表 2125. DB2 pureScale 实例中的集群高速缓存设施的可能状态和警报组合

STATE	ALERT	描述
STOPPED	NO	已使用 **db2stop** 命令手动停止集群高速缓存设施（又称为 CF）。
	YES	CF 尝试变为主 CF，但未成功。管理员已使用 **db2stop** 命令在实例中手动停止了集群高速缓存设施。
RESTARTING	NO	通过执行 **db2start** 命令或在主 CF 发生故障后，CF 将重新启动。
	YES	CF 正在重新启动，但是，因为 CF 先前尝试承担主角色但失败而产生了暂挂警报，必须手动清除此警报。
BECOMING_ PRIMARY	NO	如果此实例中还没有任何其他主 CF 在运行，那么 CF 将承担主 CF 的角色。
	YES	不适用。CF 无法尝试承担设置了警报条件的主角色。
PRIMARY	NO	CF 已承担主 CF 的角色，并且正以正常方式运行。
	YES	不适用。CF 无法充当设置了警报条件的主 CF。

表 2125. DB2 pureScale 实例中的集群高速缓存设施的可能状态和警报组合 (续)

STATE	ALERT	描述
CATCHUP(*n*%)	NO	此非主 CF 正从主 CF 复制以 PEER 方式运行时所需的信息。 **注**: 使用 **db2instance -list** 命令来查看非主 CF 的状态时，它将处于 **CATCHUP** 状态直到建立与数据库的连接。进行第一个连接后，从主 CF 复制数据的过程开始。
	YES	此非主 CF 正从主 CF 复制以 PEER 方式运行时所需的信息。因为此 CF 先前尝试承担主角色但失败而产生了暂挂警报，所以必须手动清除此警报。
PEER	NO	如果当前主 CF 失败，那么此非主 CF 准备采用主 CF 的角色。
	YES	如果当前主 CF 失败，那么此非主 CF 准备采用主 CF 的角色。因为此 CF 先前尝试承担主角色但失败而产生了暂挂警报，所以必须手动清除此警报。
ERROR	YES	CF 未能在实例中的任何主机上启动。您需要先解决此问题并手动清除警报，然后尝试重新启动此实例。

提示: 可使用 DB2_INSTANCE_ALERTS 管理视图来查看有关警报的详细信息。

针对数据共享和非 DB2 pureScale 环境的报告中的差别

报告主机、成员和集群高速缓存设施的状态数据的各种表函数、管理视图和命令都可在 DB2 pureScale 实例外部使用。但是，这些界面返回的结果可能与您在 DB2 pureScale 实例中见到的结果不同。

在将集群文件系统与受支持集群管理器 (CM) 配合使用的配置（此配置有时称为“集成高可用性”或“集成 HA”）中，针对其中大部分状态报告界面返回的结果与您在 DB2 pureScale 实例中见到的结果很像。当您使用 DB2_GET_CLUSTER_HOST_STATE 表函数或 DB2_CLUSTER_HOST_STATE 管理视图在实例中检索有关主机的信息时是一个例外。在具有集成 HA 的 DB2 pureScale 实例外部，这些界面都不会返回 INSTANCE_STOPPED 列。例如，使用 DB2_CLUSTER_HOST_STATE 管理视图的查询的结果与 图 20 中显示的结果很像

```
HOSTNAME STATE  INSTANCE_STOPPED ALERT
-------- ------ ---------------- -----
HOSTA    ACTIVE -                NO
HOSTB    ACTIVE -                NO
HOSTC    ACTIVE -                NO
HOSTD    ACTIVE -                NO
```

图 20. 具有集群管理器的 DB2 pureScale 实例外部的 DB2_CLUSTER_HOST_STATE 管理视图返回的结果。

专门报告集群高速缓存设施的状态的任何界面是另一个例外。在 DB2 pureScale 环境外部，没有任何集群高速缓存设施，所以没有任何要报告的状态。例如，DB2_CF 管理视图返回的结果类似于非 DB2 pureScale 环境环境中的以下内容:

```
ID     CURRENT_HOST        STATE      ALERT
------ ------------------- ---------- -----

     0 record(s) selected.
```

图 21. DB2 pureScale 实例外部的 DB2_CF 管理视图返回的结果。

在没有 CM 的实例中使用状态报告界面时，根本不会返回任何状态或警报信息。例如，使用 DB2_CLUSTER__HOST_STATE 管理视图的查询的结果与 第 1359 页的图 22 中

显示的结果很像

```
HOSTNAME STATE  INSTANCE_STOPPED ALERT
-------- ------ ---------------- -----
HOSTA    -      -                -
HOSTB    -      -                -
HOSTC    -      -                -
HOSTD    -      -                -

     4 record(s) selected.
```

图 22. 不带集群管理器的 DB2 pureScale 实例外部的 DB2_CLUSTER_HOST_STATE 管理视图返回的结果。

示例: 查看主机、成员和集群高速缓存设施的状态

下面的主题包括使用各种界面来查看 DB2 pureScale实例组件状态的示例。

查看 DB2 pureScale实例中的主机的状态信息

可检索显示 DB2 pureScale实例中的主机的整体状态的基本信息。此信息告诉您主机是否处于活动状态、实例是否正在该主机上运行以及是否有任何需要调查的警报。

关于此任务

可从查看 DB2 pureScale实例中的主机状态着手来整体了解实例的状态。

检索此状态的一个方法是使用 DB2_CLUSTER_HOST_STATE 管理视图。此视图将返回实例中所有主机的状态信息。还可使用以下界面来检索有关主机状态的信息:

- DB2_GET_CLUSTER_HOST_STATE 表函数。如果您要查询特定主机的状态，那么此方法很有用，因为此表函数接受主机标识作为参数。
- **LIST INSTANCE** 命令。
- **db2instance** 命令，带 **-list** 参数

过程

要查看 DB2 pureScale实例中的主机的状态，请执行以下操作:

1. 使用 DB2_CLUSTER_HOST_STATE 管理视图或 DB2_GET_CLUSTER_HOST_STATE 表函数来制定 SQL 语句。 以下示例使用管理视图:

   ```
   SELECT varchar(HOSTNAME,10) AS HOST,
          varchar(STATE,8) AS STATE,
          varchar(INSTANCE_STOPPED,7) AS STOPPED,
          ALERT
   FROM SYSIBMADM.DB2_CLUSTER_HOST_STATE
   ```

2. 运行此查询。

结果

运行先前 SQL 语句的输出将如下所示:

```
HOST       STATE    STOPPED ALERT
---------- -------- ------- --------
HOSTD      ACTIVE   NO      NO
HOSTB      ACTIVE   NO      NO
```

```
HOSTA      ACTIVE   YES     NO
HOSTC      ACTIVE   NO      NO

  4 record(s) selected.
```

在此示例中，此实例包含 4 个主机。三个主机处于活动状态（只意味着这些系统已打开电源），并且能够响应操作系统命令。 一个主机已停止，意味着此实例已在该主机上由管理员显式停止。没有需要调查的警报。

示例

使用 DB2_GET_CLUSTER_HOST_STATE 表函数来检索主机状态

DB2_GET_CLUSTER_HOST_STATE 表函数还允许您检索有关 DB2 pureScale 实例中的主机的状态信息。表函数的优点之一是您可将参数传递至表函数以缩小所返回结果的范围。例如，要检索 DB2 pureScale实例中的主机 HOSTD 的信息，请构造如下所示的查询:

```
SELECT varchar(HOSTNAME,10) as HOST,
       varchar(STATE,10) AS STATE,
       ALERT
FROM TABLE(DB2_GET_CLUSTER_HOST_STATE('HOSTD'))
```

结果:

```
HOST       STATE      ALERT
---------- ---------- --------
HOSTD      ACTIVE     NO

  1 record(s) selected.
```

查看 DB2 pureScale实例中的成员和集群高速缓存设施的状态信息

可查看 DB2 pureScale实例中的成员和集群高速缓存设施（又称为 CF）的运行状态的详细信息，例如，CF 充当的角色（例如，主项或对等项）以及成员是否已故障恢复至另一主机。

关于此任务

此任务中提供的示例显示如何使用 **db2instance** 系统命令来检索 DB2 pureScale实例中的成员和集群高速缓存设施的状态的信息。使用系统命令的好处在于不需要数据库连接。但是，必须从实例中充当成员（并非 CF）的主机来运行此命令。

过程

要使用 **db2instance** 命令来检索有关 DB2 pureScale实例中的成员和 CF 的状态信息，请在实例中的某个成员的系统提示符处输入此命令并指定 `-list` 选项:

```
db2instance -list
```

db2instance 命令返回类似以下查询的信息（已稍微压缩输出以供展示）:

```
ID        TYPE          STATE       HOME_HOST      CURRENT_HOST     ALERT   PARTITION_NUMBER      LOGICAL_PORT    NETNAME
--        ----          -----       ---------      ------------     -----   ----------------      ------------    ---------
0       MEMBER        STARTED          HOSTA             HOSTA        NO                  0                 0  OSTA-ib0
1       MEMBER        STARTED          HOSTB             HOSTB        NO                  0                 0  HOSTB-ib0
2       MEMBER        STARTED          HOSTC             HOSTC        NO                  0                 0  HOSTC-ib0
128     CF            PRIMARY          HOSTD             HOSTD        NO                  -                 0  HOSTD-ib0
129     CF            PEER             HOSTE             HOSTE        NO                  -                 0  HOSTE-ib0

HOSTNAME                 STATE              INSTANCE_STOPPED        ALERT
--------                 -----              ----------------        -----
   HOSTA                ACTIVE                           NO           NO
   HOSTC                ACTIVE                           NO           NO
   HOSTD                ACTIVE                           NO           NO
   HOSTE                ACTIVE                           NO           NO
   HOSTB                ACTIVE                           NO           NO
```

结果

返回的结果取决于 DB2 pureScale实例的结构。在此示例中，此报告显示了下列内容:

- HOSTA 至 HOSTC 的主机配置为成员
- 每个成员已启动，并且正在自己的原始主机上运行
- 有 2 个集群高速缓存设施 CF 正在主机 HOSTD 和 HOSTE 上运行
- 主 CF 在 HOSTD 上运行；另一个 CF 在 HOSTE 上以对等方式运行，指示它准备在主 CF 发生故障时接管主 CF 的职责。

示例

还可使用以下界面来检索成员和集群高速缓存设施的状态信息:

- DB2_MEMBER 或 DB2_CF 管理视图
- DB2_GET_INSTANCE_INFO 表函数
- **LIST INSTANCE** 命令行处理器 (CLP) 命令
- **db2cluster** 系统命令。

以下示例说明如何使用其中某些界面。

示例 1: 使用 DB2_MEMBER 管理视图来检索状态信息

DB2_MEMBER 管理视图显示 DB2 pureScale实例中的成员的状态信息。后跟说明如何使用此管理视图来检索成员状态的示例:

```
SELECT ID,
       varchar(STATE,21) AS STATE,
       varchar(HOME_HOST,10) AS HOME_HOST,
       varchar(CURRENT_HOST,10) AS CUR_HOST,
       ALERT
FROM SYSIBMADM.DB2_MEMBER
```

结果:

```
ID     STATE                HOME_HOST  CUR_HOST   ALERT
------ -------------------- ---------- ---------- --------
     0 WAITING_FOR_FAILBACK       HOSTA      HOSTB NO
     1 STARTED                    HOSTB      HOSTB NO
     2 STARTED                    HOSTC      HOSTC NO

  3 record(s) selected.
```

在此示例中，成员 0 在其原始主机上出现故障并已故障转移至 HOSTB。成员 0 正等待故障恢复至其原始主机 HOSTA。

示例 2: 使用 DB2_GET_INSTANCE_INFO 表函数来检索 CF 的状态信息

DB2_GET_INSTANCE_INFO 表函数允许您检索 DB2 pureScale实例中的成员的状态信息。表函数的优点之一是您可将参数传递至表函数以缩小所返回结果的范围。例如，要检索有关 DB2 pureScale 实例中 CF 的信息，可构造如下查询:

```
SELECT ID,
       varchar(STATE,17) AS STATE,
       varchar(HOME_HOST,10) AS HOME_HOST,
       varchar(CURRENT_HOST,10) AS CUR_HOST,
       ALERT
FROM TABLE(DB2_GET_INSTANCE_INFO(NULL,'','','CF',NULL))
```

结果:

```
ID     STATE             HOME_HOST  CUR_HOST   ALERT
------ ----------------- ---------- ---------- --------
   128 RESTARTING            HOSTD      HOSTD NO
   129 BECOMING_PRIMARY      HOSTE      HOSTE NO

  1 record(s) selected.
```

在此示例中，主机标识为 128 的 CF 失败。主机标识为 129 的 CF 处于以主 CF 身份接管的过程中。

*示例 3: 调查使用 **db2instance -list** 命令报告的警报。*

在此示例中，运行 **db2instance -list** 命令的结果如下所示:

```
$ db2instance -list
ID        TYPE            STATE        HOME_HOST             CURRENT_HOST          ALERT   PARTITION_NUMBER       LOGICAL_PORT    NETNAME
--        ----            -----        ---------             ------------          -----   ----------------       ------------    -------
0       MEMBER          STARTED            HostA                   HostA             NO                 0                  0         -
1       MEMBER          STARTED            HostB                   HostB             NO                 0                  1         -
2       MEMBER          STARTED            HostC                   HostC             NO                 0                  2         -
128     CF              ERROR              HostD                   HostD            YES                 -                  0         -
129     CF              ERROR              HostE                   HostE            YES                 -                  0         -

HOSTNAME                  STATE               INSTANCE_STOPPED       ALERT
--------                  -----               ----------------       -----
   HostA                 ACTIVE                             NO          NO
   HostB                 ACTIVE                             NO          NO
   HostC                 ACTIVE                             NO          NO
   HostD                 ACTIVE                             NO         YES
   HostE                 ACTIVE                             NO         YES

数据共享实例中的成员、CF 或主机当前存在警报。有关此警报、此警报的影响以及如何清除此警报的更多信息，请运行以下命令: "db2cluster -cm -list -alert"
```

在此示例中，实例中的集群高速缓存设施都有警报。而且，CF 的状态显示为 ERROR。按报告结尾处的消息的建议，可将 **db2cluster** 命令与 **-cm -list -alert** 选项配合使用以查看有关这些警报的更多信息:

```
$db2cluster -cm -list -alert

Alert: CF '128' failed to start the PRIMARY role on host 'HostD'. Check the cadiag*.log for failures related to CF '128' for
more information.

Action: This alert must be cleared manually with the command: 'db2cluster -cm -clear -alert'.

Impact: CF '128' on host 'HostD' will be unavailable to service requests from DB2 members until the alert is cleared.
```

示例 4: 成员警报，某个成员未能在 DB2 pureScale实例中启动

在此示例中，运行 **db2instance -list** 命令的结果如下所示:

```
$ db2instance -list
ID        TYPE            STATE        HOME_HOST              CURRENT_HOST           ALERT   PARTITION_NUMBER       LOGICAL_PORT    NETNAME
--        ----            -----        ---------              ------------           -----   ----------------       ------------    -------
0       MEMBER          ERROR            HostA                  HostA              YES               0                  0         -
1       MEMBER          STARTED          HostB                  HostB              NO                0                  1         -
2       MEMBER          STARTED          HostC                  HostC              NO                0                  2         -
128     CF              PRIMARY          HostD                  HostD              NO                -                  0         -
129     CF                PEER           HostE                  HostE              NO                -                  0         -

HOSTNAME                  STATE              INSTANCE_STOPPED       ALERT
--------                  -----              ----------------       -----
   HostA                ACTIVE                           NO          NO
   HostB                ACTIVE                           NO          NO
   HostC                ACTIVE                           NO          NO
   HostD                ACTIVE                           NO          NO
   HostE                ACTIVE                           NO          NO

数据共享实例中的成员、CF 或主机当前存在警报。有关此警报、此警报的影响以及如何清除此警报的更多信息，请运行以下命令："db2cluster -cm -list -alert"
```

在此示例中，某个成员未能在 DB2 pureScale实例中启动。运行带有 **-cm -list -alert** 选项的 **db2cluster** 命令会建议要执行的操作并指出此故障在 DB2 pureScale实例中造成的影响。

```
$db2cluster -cm -list -alert

Alert: DB2 member '0' failed to start on its home host 'HostA'. The cluster manager will attempt to restart the DB2 member in restart
light mode on another host. Check the db2diag.log for messages concerning failures on host 'HostA' for member '0'."

Action: This alert must be cleared manually with the command: 'db2cluster -cm -clear -alert'.

Impact: DB2 member '%0' will not be able to service requests until this alert has been cleared and the DB2 member returns to its home host.
```

示例 5: CF 错误，辅助 CF 未能完成 CATCHUP 阶段

在此示例中，运行 **db2instance -list** 命令的结果如下所示:

```
$ db2instance -list
ID        TYPE            STATE        HOME_HOST              CURRENT_HOST           ALERT   PARTITION_NUMBER       LOGICAL_PORT    NETNAME
--        ----            -----        ---------              ------------           -----   ----------------       ------------    -------
0       MEMBER          STARTED          HostA                  HostA              NO                0                  0         -
1       MEMBER          STARTED          HostB                  HostB              NO                0                  1         -
2       MEMBER          STARTED          HostC                  HostC              NO                0                  2         -
128     CF              PRIMARY          HostD                  HostD              NO                -                  0         -
129     CF                ERROR          HostE                  HostE              YES               -                  0         -

HOSTNAME                  STATE              INSTANCE_STOPPED       ALERT
--------                  -----              ----------------       -----
   HostA                ACTIVE                           NO          NO
   HostB                ACTIVE                           NO          NO
   HostC                ACTIVE                           NO          NO
   HostD                ACTIVE                           NO          NO
   HostE                ACTIVE                           NO          NO

数据共享实例中的成员、CF 或主机当前存在警报。有关此警报、此警报的影响以及如何清除此警报的更多信息，请运行以下命令："db2cluster -cm -list -alert"
```

在此示例中，辅助 CF 未能完成 CATCHUP 阶段。运行带有 **-cm -list -alert** 选项的 **db2cluster** 命令会建议要执行的操作并指出此故障在 DB2 pureScale实例中造成的影响。

```
$db2cluster -cm -list -alert

Alert: CF '129' failed to complete CATCHUP on host 'HostE'. Check the db2diag.log for failure messages pertaining to CATCHUP on CF '129'.

Action: Contact IBM support to determine the reason for the failure. To re-attempt CATCHUP, restart the failed CF with the commands:
'db2stop 129; db2start 129'. This alert will clear itself when the CF is restarted.

Impact: CF '129' on host 'HostE' will not be available until it can undergo CATCHUP successfully.
```

示例 6: 主机警报，主机"HostA"断开了网络连接。

在此示例中，运行 **db2instance -list** 命令的结果如下所示:

```
$ db2instance -list
ID        TYPE                STATE         HOME_HOST              CURRENT_HOST           ALERT   PARTITION_NUMBER       LOGICAL_PORT    NETNAME
--        ----                -----         ---------              ------------           -----   ----------------       ------------    -------
0       MEMBER  WAITING_FOR_FAILBACK            HostA                  HostA              NO                 0                  0         -
1       MEMBER              STARTED             HostB                  HostB              NO                 0                  1         -
2       MEMBER              STARTED             HostC                  HostC              NO                 0                  2         -
128     CF                  PRIMARY             HostD                  HostD              NO                 -                  0         -
129     CF                    PEER              HostE                  HostE              NO                 -                  0         -

HOSTNAME                 STATE              INSTANCE_STOPPED       ALERT
--------                 -----              ----------------       -----
   HostA               INACTIVE                            NO         YES
   HostB                 ACTIVE                            NO          NO
   HostC                 ACTIVE                            NO          NO
   HostD                 ACTIVE                            NO          NO
   HostE                 ACTIVE                            NO          NO

数据共享实例中的成员、CF 或主机当前存在警报。有关此警报、此警报的影响以及如何清除此警报的更多信息，请运行以下命令: "db2cluster -cm -list -alert"。
```

在此示例中，主机“HostA”断开了网络连接。运行带有 **-cm -list -alert** 选项的 **db2cluster** 命令会建议要执行的操作并指出此故障在 DB2 pureScale实例中造成的影响。

```
$db2cluster -cm -list -alert

Alert: Host 'HostA' is INACTIVE. Ensure the host is powered on and connected to the network.

Action: This alert will clear itself when the host is ACTIVE.

Impact: While the host is INACTIVE, the DB2 members on this host will be in restart light mode on other hosts and will be in the
WAITING_FOR_FAILBACK state. Any CF defined on the host will not be able to start, and the host will not be available as a target
for restart light.
```

检查成员的重新启动状态

如果知道成员已失败（可能因为断电或更正后的其他硬件问题），那么您可能想要知道它是否已成功重新启动。

关于此任务

可使用 DB2_MEMBER 管理视图来检查 DB2 pureScale实例中的所有成员的运行状态。还可使用 DB2_GET_INSTANCE_INFO 表函数，此表函数提供用于查询特定主机的选项。

用于检查成员重新启动状态的进程正好显示在第 1360 页的『查看 DB2 pureScale实例中的成员和集群高速缓存设施的状态信息』的示例 1 中。具体地说，制定使用 DB2_MEMBER 管理视图（或 DB2_GET_INSTANCE_INFO 表函数）的 SQL 查询来检索以下列的值:

- ID
- HOME_HOST
- CURRENT_HOST
- STATE
- ALERT

过程

1. 使用您喜欢的界面来制定 SQL 查询。 此示例使用 DB2_MEMBER 管理视图:

```
SELECT ID,
       varchar(STATE,21) AS STATE,
       varchar(HOME_HOST,10) AS HOME_HOST,
       varchar(CURRENT_HOST,10) AS CUR_HOST,
       ALERT
FROM SYSIBMADM.DB2_MEMBER
```

2. 运行此查询。 所返回结果将类似以下所示:

```
ID     STATE                HOME_HOST  CUR_HOST   ALERT
------ -------------------- ---------- ---------- --------
     0 STARTED              HOSTA      HOSTA      NO
```

```
  1 STARTED               HOSTB      HOSTB      NO
  2 STARTED               HOSTC      HOSTC      NO

3 record(s) selected.
```

在上述示例中，所有 成员 都在自己的主机上运行，并且没有任何警报。

结果

查看成员的重新启动状态时，请检查:

- STATE 列的值是 RESTARTING 还是 STARTED。前者指示成员处于重新启动过程，后者指示它已成功重新启动。如果状态为 RESTARTING，请在几分钟后再检查状态以了解状态是否已更改为 STARTED。
- CUR_HOST 的值与 HOME_HOST 的值相同。这指示成员正在其原始主机上运行。
- 您所关心的成员的警报列中没有 YES 值。

如果 CUR_HOST 不同于该 HOME_HOST、状态未超越 RESTARTING 或仍然为 WAITING_FOR_FAILBACK 或者警报列中存在 YES 值，那么可能存在需要进一步调查的问题。

示例

示例 1: 处于重新启动过程的失败成员

在此示例中，成员 0 处于在其原始主机 HOSTA 上重新启动的过程。

```
ID     STATE                HOME_HOST  CUR_HOST   ALERT
------ -------------------- ---------- ---------- --------
     0 RESTARTING           HOSTA      HOSTA      NO
     1 STARTED              HOSTB      HOSTB      NO
     2 STARTED              HOSTC      HOSTC      NO

  3 record(s) selected.
```

要了解重新启动最终是否成功，请在几秒钟后再次运行此查询。

示例 2: 无法重新启动的失败成员

在此示例中，成员 0 等待故障恢复至其原始主机。当前它正以轻量级方式在 HOSTB 上运行。

```
ID     STATE                HOME_HOST  CUR_HOST   ALERT
------ -------------------- ---------- ---------- --------
     0 WAITING_FOR_FAILBACK HOSTA      HOSTB      NO
     1 STARTED              HOSTB      HOSTB      NO
     2 STARTED              HOSTC      HOSTC      NO

  3 record(s) selected.
```

在此情况下，您可能需要检查 HOSTA 的主机状态以了解是否存在问题。使用 DB2_CLUSTER_HOST_STATE 管理视图可能会返回以下结果:

```
HOST       STATE    STOPPED ALERT
---------- -------- ------- --------
HOSTD      ACTIVE   NO      NO
HOSTB      ACTIVE   NO      NO
HOSTA      INACTIVE NO      YES
HOSTC      ACTIVE   NO      NO

  4 record(s) selected.
```

此报告显示 HOSTA 上存在警报，并且此主机处于不活动状态。但是，未使用 **db2stop** 命令停止此实例。可能会进行进一步调查（例如，此主机断电时）。解决此主机的问题后，请再次检查重新启动状态以了解成员能否重新启动。

查看警报的详细信息

如果某个成员或集群高速缓存设施报告警报，那么可使用 DB2_INSTANCE_ALERTS 管理视图来查看有关此警报的更多信息。另外，可将 **db2cluster** 命令与 **-cm -list -alert** 参数配合使用。

关于此任务

此任务假定您已确定 DB2 pureScale 实例中的其中的一个主机上发生了警报。例如，**LIST INSTANCE** 命令可能已显示针对某个成员的警报:

```
ID     TYPE    STATE   HOME_HOST CURRENT_HOST ALERT
------ ------- ------- --------- ------------ -----
     0 MEMBER  STARTED hostA     hostA        YES
     1 MEMBER  STARTED hostB     hostB        NO
     2 MEMBER  STARTED hostC     hostC        NO
     3 MEMBER  STARTED hostD     hostD        NO
   128 CF      PRIMARY              hostE     -            NO
   129 CF      PEER                 hostF     -            NO

HOSTNAME                  STATE               INSTANCE_STOPPED        ALERT
--------- ------ ---------------- -----
hostA    ACTIVE NO               NO
hostB         ACTIVE     NO               NO
hostC         ACTIVE     NO               NO
hostD         ACTIVE     NO               NO
hostE     ACTIVE NO              NO
hostF    ACTIVE   NO               NO
```

在此情况下，成员 0 将显示警报。

过程

要查找有关实例中的警报的更多信息，请执行以下操作:

1. 制定一个使用 DB2_INSTANCE_ALERTS 管理视图的 SQL 语句:

   ```
   SELECT * FROM SYSIBMADM.DB2_INSTANCE_ALERTS
   ```

2. 运行此 SQL 语句。

结果

根据成员 0 上的问题的特性，DB2_INSTANCE_ALERTS 管理视图返回的信息有所不同。例如，您可能会接收到类似如下的消息:

```
MESSAGE
-------------------------------------------------------------------------------------------------------------
未能在主机"hostA"上对 DB2 成员"0"执行轻量级重新启动。请检查 db2diag.log 以获取有关 DB2 成员"0"的指
示主机上的轻量级重新启动或数据库崩溃恢复故障的消息。

ALERT_ACTION
-------------------------------------------------------------------------------------
必须使用以下命令手动清除此警报: "db2cluster -clear -alert -member 0"

IMPACT
-------------------------------------------------------------------------------------------------------
直到清除此警报，DB2 成员"0"才能在主机"hostC"上进行轻量级重新启动。
```

示例

*示例 1: 使用 **db2cluster** 命令来查看警报信息*

在此示例中，**db2instance -list** 命令返回以下信息:

```
$ db2instance -list
ID        TYPE            STATE          HOME_HOST              CURRENT_HOST           ALERT   PARTITION_NUMBER       LOGICAL_PORT    NETNAME
--        ----            -----          ---------              ------------           -----   ----------------       ------------    -------
0       MEMBER          ERROR              HostA                  HostA                YES                0                  0         -
1       MEMBER          STARTED            HostB                  HostB                 NO                0                  1         -
2       MEMBER          STARTED            HostC                  HostC                 NO                0                  2         -
128     CF              PRIMARY            HostD                  HostD                 NO                -                  0         -
129     CF              PEER               HostE                  HostE                 NO                -                  0         -

HOSTNAME                  STATE              INSTANCE_STOPPED       ALERT
--------                  -----              ----------------       -----
   HostA                ACTIVE                           NO         YES
   HostB                ACTIVE                           NO          NO
   HostC                ACTIVE                           NO          NO
   HostD                ACTIVE                           NO          NO
   HostE                ACTIVE                           NO          NO

数据共享实例中的成员、CF 或主机当前存在警报。有关此警报、此警报的影响以及如何清除此警报的更多信息，请运行以下命令: "db2cluster -cm -list -alert"
```

在 第 1356 页的表 2124 中包含的信息中，您可见到 **db2instance -list** 的输出正显示成员 0 无法在任何主机上启动。（如果它已在访客主机上启动，那么状态将为 STARTED，并且当前主机将显示它正在其上运行的主机的名称。）

使用 **db2cluster -cm -list -alert** 命令会显示以下消息:

```
$ db2cluster -cm -list -alert

Alert: DB2 member '0' failed to start on its home host 'Host A'. The cluster manager will
attempt to restart the DB2 member
in restart light mode on another host. Check the db2diag.log for messages concerning failures
on hosts 'HostA' for member '0'.

Action: This alert must be cleared manually with the command: 'db2cluster -cm -clear -alert'.

Impact: DB2 member '0' will not be able to service requests until this alert has been cleared
and the DB2 member returns to its home host.
```

下一步做什么

执行 DB2_INSTANCE_ALERTS 管理视图或 **db2cluster** 命令返回的信息中指定的调查或操作。

第 13 章 在 DB2 pureScale 环境中监视事件、实时数据库和系统

除了查看 DB2 pureScale实例的组件的整体状态外，还可使用 DB2 监视基础结构来检查集群高速缓存设施和成员的操作的特定方面。可使用监视表函数和管理视图来显示此信息。还可使用所选事件监视器以在发生事件时将其捕获。

DB2 V9.7 为 DB2 产品的监视基础结构引入了若干增强功能。其中一个增强功能是一组表函数，它们允许您访问数以百计的*内存中*监视元素，可使用这些监视元素来查询特定时间点的数据库环境状态。其他增强功能包括已改进事件监视器，它们用于捕获锁定、工作单元和活动（发生时）之类的对象的信息。

DB2 pureScale Feature使用监视元素扩展了内置到 DB2 数据库中的监视功能，可使用这些监视元素在 DB2 pureScale实例中查看用于描述集群高速缓存设施（又称为 CF）和成员的操作的特定方面的数据。但是，在 DB2 pureScale实例和其他 DB2 实例中进行监视有一些不同之处需要注意，包括:

- 『监视 CF（除了 DB2 成员之外）的能力』
- 『DB2 pureScale实例中的监视元素的报告方式』
- 报告监视元素时发生组件故障的影响。

监视 CF（除了 DB2 成员之外）的能力

CF，（它们充当的角色与 DB2 pureScale 环境中的成员相比有所不同）引入了其他监视需求。例如，在 DB2 pureScale实例以外的 DB2 实例中，您可能会想要监视缓冲池命中率，命中率表示在内存中发现的页数与必须从磁盘读取的页数之比。一般来讲，缓冲池命中率越高，反映性能越好。性能较高的原因是将所需页导入到内存中时所涉及的 I/0 较少。在 DB2 pureScale 环境中，从磁盘读取所有物理页的操作由成员执行，但仅在它们先咨询 CF 以了解组缓冲池是否有任何具有它们可使用的有效页的其他成员的记录后执行。因此，虽然您可能习惯仅调整 DB2 pureScale 环境以外的 DB2 环境中的本地缓冲池，但监视 CF 中的组缓冲池的缓冲池命中率在 DB2 pureScale 环境中也很重要。可在本地缓冲池或组缓冲池 (GBP) 中找到页的次数越多，必须从磁盘读取页的次数就越少。

除了 GBP 之外，全局锁定管理器 (GLM) 是 CF 的您可监视的另一个组件。GLM 管理 DB2 pureScale实例中的所有成员中的对象的锁定。DB2 pureScale Feature 添加了您可用于监视成员间锁定的监视元素。

DB2 pureScale实例中的监视元素的报告方式

一般来说，DB2 pureScale实例中的监视机制与其他 DB2 实例中的监视机制相似。例如，MON_GET_TABLESPACE 表函数（返回有关数据库中的表空间的信息）的工作方式在 DB2 pureScale实例和其他 DB2 实例中类似。在 DB2 pureScale实例中，某些监视元素的作用域被限制为所有成员中的特定成员，其他监视元素的作用域则为全局。例如，**direct_reads** 或 **pool_data_l_reads** 之类的监视元素中的数据特定于成员执行的读取活动。通过比较，**tbsp_total_pages** 之类的监视元素（表示表空间的物理属性）在所有成员中相同，因为所有成员共享同一表空间。例如，考虑以下查询:

```
SELECT VARCHAR(TBSP_NAME, 30) AS TBSP_NAME,
       MEMBER, POOL_DATA_L_READS,
       TBSP_TOTAL_PAGES
FROM TABLE(MON_GET_TABLESPACE('USERSPACE1',-2))
```

此查询的结果类似以下示例:

```
TBSP_NAME                      MEMBER POOL_DATA_L_READS    TBSP_TOTAL_PAGES
------------------------------ ------ -------------------- --------------------
USERSPACE1                          1                    0                 4096
USERSPACE1                          2                    0                 4096
USERSPACE1                          3                    0                 4096
USERSPACE1                          0                   36                 4096

  4 record(s) selected.
```

在此示例中，从每个成员的本地缓冲池中进行逻辑读取的次数不同，因为每个成员以独立于其他成员的方式执行读取；但是，表空间的总页数在所有成员中相同，因为所有成员在同一实例 USERSPACE1 中工作。

报告监视元素时发生组件故障的影响

如果 DB2 pureScale 环境中的主机、成员或 CF 失败，那么除非整个 DB2 pureScale实例关闭，否则您仍可从该实例中检索监视元素。但是，失败的组件不会生成统计信息。如果您正在运行 第 1369 页的『DB2 pureScale实例中的监视元素的报告方式』 中显示的第一个示例之类的查询（逐个显示每个成员中的数据），那么这一点特别明显。即使您使用一个聚集成员间信息的查询，您也可能会未注意到成员中缺少数据。

要记住的另一点是，如果正在进行监视元素数据收集时成员失败，那么数据收集过程会暂停，直到检测到失败成员的通信问题或经历 TCP/IP 超时时间段。在此情况下，仍会报告此数据，但是，不会有来自失败成员的信息。

最后，请记住如果成员失败，那么监视元素中累积的所有统计信息会重置为 0。

集群高速缓存设施内存和 CPU 使用情况监视概述

集群高速缓存设施的运行有效性的基本指示器是内存和 CPU 以一致方式使用至其最大已配置能力的程度。

内存使用情况

集群高速缓存设施（又称为 CF）将不同内存堆用于以下用途:

组缓冲池内存

组缓冲池内存被用于 DB2 pureScale实例的组缓冲池。如果以一致方式将此类型的内存用至最大已配置能力，那么可能会对性能有负面影响。但是，内存可能被用至极限的事实本身并不指示性能可能受到影响。请检查组缓冲池的命中速率以确保性能是否降低。低命中速率与高组缓冲池内存使用情况一起出现可能指示需要增加此类型的内存。此类型的内存由 **cf_gbp_sz** 配置参数配置。

锁定内存

锁定内存用于管理 DB2 pureScale实例中的页锁定。如果 CF 上的锁定没有足够的内存可用，那么可能会发生以下其中一种或两种情况:

- 锁定升级可能发生，这会降低所涉及对象的并行性
- 锁定请求可能被拒绝，从而导致返回 SQL0912 消息。

此类型的内存由 **cf_lock_sz** 配置参数配置。

共享通信区 (SCA) 内存

SCA 内存包含表、索引、表空间和目录的数据库范围信息。每个数据库在 CF 中都有自己的 SCA 内存。此内存是第一次在任何 DB2 成员上激活数据库期间分配的，直到数据库被废弃或 CF 停止时才会释放。如果使用表分区，那么使 CF 与成员间的表分区数据同步时所需的信息也存储在 SCA 内存中。

如果此类型的内存被用至极限，那么表可能会无法装入，并且会返回错误。此类型的内存由 **cf_sca_sz** 配置参数配置。

整体 CF 内存

整体 CF 内存是对 CF 可用的总物理内存量。它由 **cf_mem_size** 配置参数设置。组缓冲池、锁定和共享通信区的内存都分配自此内存池。因此，为这些特定类型的内存分配的总内存量不能超过使用 **cf_mem_size** 配置参数配置的内存量。

缺省情况下，系统会自动配置其中每个类型的内存。DB2 pureScale Feature提供了一些监视元素，可使用这些监视元素来检查当前正由系统使用的每个类型的内存的量。还有一些相关元素，可使用这些元素来确定每个类型的内存的最大大小以及是否正在执行调整内存大小操作。

除了报告特定类型的 CF 内存的使用情况的监视元素，还可使用 ENV_CF_SYS_RESOURCES 管理视图来检查对 CF 可用的物理存储和虚拟存储总量。

CPU 负载

CF 上的 CPU 负载指示其处理器的负担情况。如果发现正在运行 CF 的主机上的处理器大部分时间工作至极限，那么可能指示正在运行 CF 的主机功能不够强大。您可能需要添加处理器，或升级至功能更强大的系统。

可使用 ENV_CF_SYS_RESOURCES 管理视图来查看在 DB2 pureScale实例中充当 CF 的主机的整体 CPU 负载。

注: 对 CPU 负载报告的值反映 CF 执行的实际处理的 CPU 总使用量，以及 CF 中的进程以外的主机进程的 CPU 总使用量。

用于查看集群高速缓存设施内存使用情况的监视元素

IBM DB2 pureScale Feature 提供了若干监视元素来报告集群高速缓存设施内存使用情况。

监视元素

以下监视元素提供有关如何分配和使用各种 CF 内存堆的信息:

- 第 670 页的『configured_cf_gbp_size -“已配置的集群高速缓存设施组缓冲池大小”监视元素』
- 第 709 页的『current_cf_gbp_size -“当前集群高速缓存设施组缓冲池大小”监视元素』
- 第 1215 页的『target_cf_gbp_size -“目标集群高速缓存设施组缓冲池大小”监视元素』
- 第 670 页的『configured_cf_lock_size -“已配置的集群高速缓存设施锁定大小”监视元素』
- 第 709 页的『current_cf_lock_size -“当前集群高速缓存设施锁定大小”监视元素』

- 第 1215 页的『target_cf_lock_size -“目标集群高速缓存设施锁定大小”监视元素』
- 第 671 页的『configured_cf_mem_size -“已配置的集群高速缓存设施内存大小”监视元素』
- 第 710 页的『current_cf_mem_size -“当前集群高速缓存设施内存大小”监视元素』
- 第 671 页的『configured_cf_sca_size -“已配置的集群高速缓存设施共享通信区大小”监视元素』
- 第 710 页的『current_cf_sca_size -“当前集群高速缓存设施共享通信区大小”监视元素』
- 第 1215 页的『target_cf_sca_size -“目标集群高速缓存设施共享通信区大小”监视元素』

提示: 在所有情况下，针对其中每个监视元素报告的值以大小为 4K 的页数表示。所以，例如，如果您查询了 **current_gbp_size** 监视元素，并且它返回值 350，那么当前用于 GBP 的实际内存量将为 350 × 4096 字节，即 1,433,600 字节。

对于这些类型的内存中的大多数，可查询三个表示内存配置方式的不同方面的监视元素。

当前 当前内存大小（例如，current_cf_gbp_size 或 current_cf_mem_size）表示系统当前使用的该类型的内存量。

已配置 已配置内存大小（例如，configured_cf_sca_size 和 configured_cf_mem_size）表示数据库当前作为最大值配置的该类型的总内存量。当前内存的值决不能超过已配置内存的值。

目标 目标内存大小（例如，target_cf_sca_size）表示该类型的内存的新的已配置最大值。通常，目标大小与已配置大小相同。但是，如果目标大小和已配置大小不同，那么意味着该特定类型的内存正接受其已配置大小的联机更改。分配内存的过程随时间变化发生。在此调整大小过程期间的任一时间点，已配置内存表示可在该特定时间点使用的该类型的最大内存量。最后，已配置内存变为与目标内存相同。

请参阅每个监视元素的参考主题，以了解可用于检查与该监视元素相关联的数据的监视界面。

从集群高速缓存设施内存使用情况监视元素中检索信息

可使用 MON_GET_CF 表函数来检索各种报告 DB2 pureScale实例中集群高速缓存设施的内存使用情况的监视元素。

关于此任务

了解集群高速缓存设施（又称为 CF）中的内存的使用程度可帮助您确定是否调整内存分配。例如，如果集群高速缓存设施正使用的内存量接近最大已分配缓冲池内存量，那么此组缓冲池的命中速率可能低于能够达到的命中速率。或者，如果锁定内存被用至极限，那么您可能会注意到锁定升级数高于预期。

过程

要检索有关 集群高速缓存设施 中内存使用情况的信息，请执行以下操作:

1. 确定要检索的监视元素。 例如，如果要查看锁定内存使用情况，那么可从下列监视元素选择一个或多个:
   - current_cf_lock_size

- configured_cf_lock_size
- target_cf_lock_size

2. 使用 MON_GET_CF 表函数来制定查询。 通过使用步骤 第 1372 页的1 中的示例，此语句将类似以下示例:

```
SELECT SUBSTR(HOST_NAME,1,10) AS HOST,
       ID as HOSTID,
       CURRENT_CF_LOCK_SIZE,
       CONFIGURED_CF_LOCK_SIZE,
       TARGET_CF_LOCK_SIZE
  FROM TABLE( MON_GET_CF( NULL ) )
```

3. 运行此查询。 继续完成此示例，先前查询中的输出将类似以下示例:

```
HOST       HOSTID CURRENT_CF_LOCK_SIZE CONFIGURED_CF_LOCK_SIZE
---------- ------ -------------------- ----------------------
HOSTA         128               133852                 564224
HOSTB         129               133852                 564224

TARGET_CF_LOCK_SIZE
-------------------
             564224
             564224


  2 record(s) selected.
```

注: 两个集群高速缓存设施上的锁定内存通常相同，因为一个集群高速缓存设施会将其信息和配置复制到另一个上。还可使用第 1360 页的『查看 DB2 pureScale实例中的成员和集群高速缓存设施的状态信息』中描述的其中一个界面来检查两个集群高速缓存设施中的哪一个是主项。**LIST INSTANCE** 和 **db2cluster** 命令是两个这样的界面的示例。

结果

先前示例显示所使用的当前锁定内存量为 170,258 个 4k 块 或 697,376,768 字节。可用的最大锁定内存量为 564,224 个 4k 块或 2,311,061,504 字节。

示例

示例 1: 检索组缓冲池内存使用情况数据。

以下查询显示有关系统上所有集群高速缓存设施的组缓冲池内存大小的信息:

```
SELECT SUBSTR(HOST_NAME,1,20) AS HOST,
       ID as HOSTID,
       CURRENT_CF_GBP_SIZE,
       CONFIGURED_CF_GBP_SIZE,
       TARGET_CF_GBP_SIZE
  FROM TABLE( MON_GET_CF(NULL) )
```

以下输出是先前查询所返回内容的示例:

```
HOST                HOSTID CURRENT_CF_GBP_SIZE CONFIGURED_CF_GBP_SIZE TARGET_CF_GBP_SIZE
------------------- ------ ------------------- ---------------------- ------------------
HOSTA                  128              367611                 500224             500224
HOSTB                  129              367611                 500224             500224

  2 record(s) selected.
```

在此示例中，当前 GBP 大小为 367,611 个 4k 页或 1,505,734,656 字节。分配给 GBP 的内存为 500,224 个 4k 页或 2,048,917,504 字节。

示例 2: 检索特定主机的共享通信区 (SCA) 内存使用情况数据。

以下查询显示有关标识为 128 的集群高速缓存设施的 SCA 内存大小的信息:

```
SELECT SUBSTR(HOST_NAME,1,8)AS HOST,
       ID as HOSTID,
       CURRENT_CF_SCA_SIZE,
       CONFIGURED_CF_SCA_SIZE,
       TARGET_CF_SCA_SIZE
FROM TABLE(MON_GET_CF(128))
```

以下输出是先前查询所返回内容的示例:

```
HOST     HOSTID CURRENT_CF_SCA_SIZE CONFIGURED_CF_SCA_SIZE
-------- ------ ------------------- ---------------------
HOSTA       128                  43                 16128

TARGET_CF_SCA_SIZE
------------------
             23280

  1 record(s) selected.
```

在此示例中，当前正使用的 SCA 内存为 43 个 4k 页。检索这些监视元素时，最大 SCA 内存大小为 16,128 个 4k 页。但是，所配置大小和目标大小不同，这意味着 SCA 内存的大小正从先前最大所配置大小增加至 23,280 页。

下一步做什么

在所有先前示例中，请记住针对内存使用情况返回的值仅提供内存使用情况的整体概况。这些值本身不一定会传达足够的信息来告知有关更改内存配置的决策。例如，组缓冲池 (GBP) 的大小本身不会告诉您它对于 DB2 pureScale实例而言是否足够大。在此情况下，考虑使用报告缓冲池活动的监视元素来计算缓冲池命中速率。命中速率可告诉您是否必须调整缓冲池的大小。对于锁定内存，检查所发生的锁定升级数可让您了解是否分配了足够的锁定内存。高速率锁定升级指示可能需要增加锁定内存。

查看集群高速缓存设施处理器负载

可使用 ENV_CF_SYS_RESOURCES 管理视图查看 DB2 pureScale实例中的主集群高速缓存设施上的整体 CPU 负载。

开始之前

必须连接至正在 DB2 pureScale实例中运行的数据库。

关于此任务

ENV_CF_SYS_RESOURCES 管理视图返回 DB2 pureScale实例中的所有集群高速缓存设施（又称为 CF）的信息。在配置了多个 CF 的实例中，一个充当主项，其他充当以 PEER 方式运行的备份 CF，ENV_CF_SYS_RESOURCES 管理视图返回所有 CF 的信息。

注: 对 CPU 负载报告的值反映 CF 执行的实际处理的 CPU 总使用量，以及 CF 中的进程以外的主机进程的 CPU 总使用量。

过程

要确定 DB2 pureScale实例中的 CF 上的 CPU 负载，请执行以下操作:

1. 制定一个使用 ENV_CF_SYS_RESOURCES 管理视图的 SQL 语句。 例如:

```
SELECT  VARCHAR(NAME,20) AS HOST_ATTRIBUTE,
        VARCHAR(VALUE,25) AS VALUE,
        VARCHAR(UNIT,8) AS UNIT
FROM SYSIBMADM.ENV_CF_SYS_RESOURCES
```

2. 运行此语句。 以上查询应返回以下输出:

```
HOST_ATTRIBUTE       VALUE                     UNIT
-------------------- ------------------------- --------
HOST_NAME            HOSTA                       -
MEMORY_TOTAL         24108                     MB
MEMORY_FREE          3504                      MB
MEMORY_SWAP_TOTAL    4102                      MB
MEMORY_SWAP_FREE     4063                      MB
VIRTUAL_MEM_TOTAL    28211                     MB
VIRTUAL_MEM_FREE     7568                      MB
CPU_USAGE_TOTAL      96                        PERCENT
HOST_NAME            HOSTB                     -
MEMORY_TOTAL         24108                     MB
MEMORY_FREE          3342                      MB
MEMORY_SWAP_TOTAL    4102                      MB
MEMORY_SWAP_FREE     4063                      MB
VIRTUAL_MEM_TOTAL    28211                     MB
VIRTUAL_MEM_FREE     7406                      MB
CPU_USAGE_TOTAL      97                        PERCENT

  16 record(s) selected.
```

在此输出中，包含 HOSTA 和 HOSTB 的结果，这指示有两个主机配置为充当 CF。

3. 要确定哪个主机在充当主 CF，可使用 DB2_CF 管理视图:

```
SELECT VARCHAR(CURRENT_HOST,12) AS HOST,
       ID,
       STATE
FROM SYSIBMADM.DB2_CF
```

以上查询返回以下输出:

```
HOST         ID     STATE
------------ ------ --------------------------------
HOSTA           128 PRIMARY
HOSTB           129 PEER

  2 record(s) selected.
```

在此情况下，主要主机为 HOSTA，并且基于步骤 2 中使用的命令的输出，可假定主 CF 上的 CPU 负载为 96%。

下一步做什么

如果发现 CPU 使用率以最大容量在运行，那么向集群高速缓存设施添加处理器或将其升级可能会改进系统吞吐量。

注: 对于有多个逻辑处理器的主机，使用率数字可能超过 100%。例如，包含 8 个处理器的主机的处理器使用率可能达到 800%。

DB2 pureScale 环境中的缓冲池监视

通过检查可在组缓冲池或本地缓冲池中发现成员请求的数据页的次数（相对于需要从磁盘读取这些页的次数），您可了解何处存在与 I/O 有关的性能问题。一般来说，较大的缓冲池会增加在内存中发现必需页的可能性。

查看和比较与缓冲池活动有关的监视元素可帮助您了解集群高速缓存设施中的组缓冲池 (GBP) 以及每个成员的本地缓冲池 (LBP) 在减少系统中的磁盘 I/O 量的程度。

用于查看DB2 pureScale缓冲池活动的监视元素

IBM DB2 pureScale Feature 使用若干监视元素来报告 DB2 pureScale实例中的缓冲池活动。

用于组缓冲池的监视元素

以下监视元素提供有关主 CF 中的组缓冲池 (GBP) 的信息:

- 第 978 页的『pool_data_gbp_l_reads -“组缓冲池数据逻辑读取数”监视元素』
- 第 979 页的『pool_data_gbp_p_reads -“组缓冲池数据物理读取数”监视元素』
- 第 977 页的『pool_data_gbp_invalid_pages -“组缓冲池无效数据页数”监视元素』
- 第 1008 页的『pool_index_gbp_l_reads -“组缓冲池索引逻辑读取数”监视元素』
- 第 1009 页的『pool_index_gbp_p_reads -“组缓冲池索引物理读取数”监视元素』
- 第 1006 页的『pool_index_gbp_invalid_pages -“组缓冲池无效索引页数”监视元素』
- 第 1062 页的『pool_xda_gbp_l_reads -“组缓冲池 XDA 数据逻辑读取请求数”监视元素』
- 第 1064 页的『pool_xda_gbp_p_reads -“组缓冲池 XDA 数据物理读取请求数”监视元素』
- 第 1061 页的『pool_xda_gbp_invalid_pages -“组缓冲池无效 XDA 数据页数”监视元素』
- 第 959 页的『pool_async_data_gbp_indep_pages_found_in_lbp -“本地缓冲池中由异步 EDU 发现的独立于组缓冲池的数据页数”监视元素』
- 第 960 页的『pool_async_data_gbp_l_reads -“异步组缓冲池数据逻辑读取数”监视元素』
- 第 960 页的『pool_async_data_gbp_p_reads -“异步组缓冲池数据物理读取数”监视元素』
- 第 959 页的『pool_async_data_gbp_invalid_pages -“异步组缓冲池无效数据页数”监视元素』
- 第 963 页的『pool_async_index_gbp_indep_pages_found_in_lbp -“本地缓冲池中由异步 EDU 发现的独立于组缓冲池的索引页数”监视元素』
- 第 964 页的『pool_async_index_gbp_l_reads -“异步组缓冲池索引逻辑读取数”监视元素』
- 第 965 页的『pool_async_index_gbp_p_reads -“异步组缓冲池索引物理读取数”监视元素』
- 第 964 页的『pool_async_index_gbp_invalid_pages -“异步组缓冲池无效索引页数”监视元素』
- 第 970 页的『chayzchpool_async_xda_gbp_indep_pages_found_in_lbp -“本地缓冲池中由异步 EDU 发现的独立于组缓冲池的 XML 存储器对象 (XDA) 页数”监视元素』
- 第 971 页的『pool_async_xda_gbp_l_reads -“组缓冲池 XDA 数据异步逻辑读取请求数”监视元素』
- 第 971 页的『pool_async_xda_gbp_p_reads -“组缓冲池 XDA 数据异步物理读取请求数”监视元素』
- 第 970 页的『pool_async_xda_gbp_invalid_pages -“异步组缓冲池无效 XDA 数据页数”监视元素』

注: 这些监视元素分别报告每个成员的缓冲池的数据; 不执行任何聚集。如果要聚集本地缓冲池的缓冲池使用情况信息（例如，计算所有本地缓冲池间的平均命中速率），请使用 SUM 聚集函数。

请参阅每个监视元素的参考主题，以了解可用于检查与该监视元素相关联的数据的监视界面。

用于本地缓冲池的监视元素

以下监视元素提供有关 DB2 pureScale实例中每个成员本地的缓冲池的信息:

- 第 975 页的『pool_data_gbp_indep_pages_found_in_lbp -“本地缓冲池中发现的独立于组缓冲池的数据页数”监视元素』
- 第 981 页的『pool_data_lbp_pages_found -“本地缓冲池发现的数据页数”监视元素』
- 第 1005 页的『pool_index_gbp_indep_pages_found_in_lbp -“本地缓冲池中发现的独立于组缓冲池的索引页数”监视元素』
- 第 1010 页的『pool_index_lbp_pages_found -“发现的本地缓冲池索引页数”监视元素』
- 第 1059 页的『pool_xda_gbp_indep_pages_found_in_lbp -“本地缓冲池中发现的独立于组缓冲池的 XDA 页数”监视元素』
- 第 1067 页的『pool_xda_lbp_pages_found -“发现的本地缓冲池 XDA 数据页数”监视元素』
- 第 961 页的『pool_async_data_lbp_pages_found -“发现的异步本地缓冲池数据页数”监视元素』
- 第 965 页的『pool_async_index_lbp_pages_found -“发现的异步本地缓冲池索引页数”监视元素』
- 第 972 页的『pool_async_xda_lbp_pages_found -“发现的异步本地缓冲池 XDA 数据页数”监视元素』

对下面三个监视元素报告的值表示从磁盘至给定成员的本地缓冲池的总读取数:

- 第 984 页的『pool_data_p_reads -“缓冲池数据物理读取数”监视元素』
- 第 1014 页的『pool_index_p_reads -“缓冲池索引物理读取数”监视元素』
- 第 1069 页的『pool_xda_p_reads -“缓冲池 XDA 数据物理读取数”监视元素』

最后三个监视元素与 DB2 pureScale环境*以外*的环境中使用的监视元素相同。这些元素是相同的，因为 DB2 pureScale 环境中的成员的本地缓冲池等价于 DB2 pureScale 环境以外的环境中的数据库的缓冲池。

要点:

- 最后三个监视元素报告的值反映本地缓冲池从依赖于 GBP 的页（即，成员向 GBP 请求的页）的磁盘读取的次数，这些页不在 GBP 中。它们还包括从临时页之类与 GBP 相关的页（即，成员本地的页以及成员对其没有 GBP 依赖关系）的磁盘读取。
- 因为 DB2 pureScale 环境中的 LBP 与 GBP 之间的关系，计算缓冲池命中率的公式与在 DB2 pureScale 环境外部使用的公式不同。有关更多信息，请参阅第 1379 页的『用于计算缓冲池命中率的公式』。

请参阅每个监视元素的参考主题，以了解可用于检查与该监视元素相关联的数据的监视界面。

DB2 pureScale 环境中的缓冲池命中速率和命中率

度量在内存中找到成员所需页（相对于在磁盘上找到所需页）的程度的一种方法是计算缓冲池*命中率*。缓冲池命中率指示数据库管理器在缓冲池中找到所请求页的次数

（又称为*命中速率*）与必须从磁盘读取此页的次数之比。在 DB2 pureScale 环境中，访问整体性能时，本地缓冲池和组缓冲池命中速率和命中率都是重要因素。

本地缓冲池 (LBP) 命中率反映可在本地缓冲池中找到成员所需并处于有效状态的页的程度。如果成员的 LBP 中的页在自装入到 LBP 中之后未被另一成员更改，那么此页被认为处于有效状态。如果另一成员已更改此页（更改可能会在此页被释放至磁盘之前发生），那么此页被认为*无效*。如果具有无效页的成员需要此页才能执行事务，那么成员必须访问 CF 才能请求此页的新的有效版本。

低 LBP 命中率指示未在本地找到这些页，必须向 CF 请求这些页。

但是，在 DB2 pureScale 环境中，查看 LBP 命中率只能了解缓冲池状况的一个方面。还需要考虑组缓冲池 (GBP) 在检索页时充当的角色以及 GBP 本身的命中率。如果某个成员在其 LBP 中找不到某页的有效副本，那么它会向 CF 请求在 GBP 中搜索此页的有效副本。GBP 执行下列其中一个操作:

- 如果 GBP 包含此页的有效副本，那么它会向发出此请求的成员提供此页。
- 否则，GBP 会告诉请求成员它必须从磁盘读取此页。

使用 LBP 的另一注意事项是独立于 GBP 的页的概念。独立于 GBP 的页是仅曾经通过成员的 LBP 访问但从未存在于 GBP 中的页。一些页可能是独立于 GBP 的，因为只有本地成员访问使用这些页的操作或这些页源自的对象。

组缓冲池命中率反映在组缓冲池中找到成员所需但没有其有效本地副本的页（相对于必须从磁盘读取）的程度。GBP 的低命中率指示 GBP 中提供的实例中成员所需的页数相对较少。增加 GBP 的大小可改进命中速率和整体性能。因此，计算成员的本地缓冲池 (LBP) 中的数据页的命中率时，需要考虑成员尝试从 LBP 读取页的次数相对于尝试读取但在 LBP 中找不到有效页的次数。有关如何使用 LBP 和 GBP 监视元素来计算 GBP 命中速率的详细信息，请参阅第 1379 页的『用于计算缓冲池命中率的公式』。

提示: 命中率可能会根据许多因素（例如，数据库中的数据特性、对其运行的查询以及硬件和软件配置）而变化。一般来讲，缓冲池命中率越高，反映查询性能越好。如果发现命中率好像很低，或者随时间变化不断下降，那么增加缓冲池大小会有帮助。要增加组缓冲池的大小，请在 CF 上调整 **cf_gbp_sz** 配置参数。要调整本地缓冲池，请在具有需要校正的缓冲池的成员上运行 **ALTER BUFFERPOOL** 语句。

缓冲池监视元素报告

在 DB2 pureScale 环境中，与其他 DB2 环境一样，每个 成员 会报告它自己的本地缓冲池。不会在成员中聚集任何数据。必须考虑您所关心的成员或成员并对数据进行相应解释。在某些情况下，您可能想要计算特定成员的命中率。在其他情况下，您可能想要一起查看所有成员的数据，以整体了解 DB2 pureScale 环境的命中速率和命中率。

例如，如果通过将 **pool_data_gbp_p_reads** 监视元素与 MON_GET_BUFFERPOOL 表函数配合使用来提交查询，以返回表示将数据页从磁盘读取到本地缓冲池中（因为在 GBP 中找不到此数据页）的次数的数据，并且您未指定要返回的成员，那么您会见到类似如下的结果:

```
MEMBER BP_NAME             POOL_DATA_GBP_P_READS
------ ------------------- ---------------------
     0 IBMDEFAULTBP                          408
     0 IBMSYSTEMBP4K                           0
     0 IBMSYSTEMBP8K                           0
     0 IBMSYSTEMBP16K                          0
     0 IBMSYSTEMBP32K                          0
     1 IBMDEFAULTBP                          108
     1 IBMSYSTEMBP4K                           0
     1 IBMSYSTEMBP8K                           0
     1 IBMSYSTEMBP16K                          0
     1 IBMSYSTEMBP32K                          0
     2 IBMDEFAULTBP                          112
     2 IBMSYSTEMBP4K                           0
     2 IBMSYSTEMBP8K                           0
     2 IBMSYSTEMBP16K                          0
     2 IBMSYSTEMBP32K                          0

  15 record(s) selected.
```

要点: 在以上示例中，可见到针对临时缓冲池报告的数据全部显示为零。这并非巧合；在 DB2 pureScale实例中，临时对象和表空间在它们的关联成员的本地。它们在 CF 上不使用 GBP。

如果您关心所有成员的结果，那么可使用 SUM 聚集函数将所有成员的数目加到一起:

```
SELECT  VARCHAR(BP_NAME,15) AS BP_NAME,
        SUM(POOL_DATA_GBP_P_READS) AS TOTAL_P_READS
FROM TABLE(MON_GET_BUFFERPOOL('', -2))
GROUP BY BP_NAME
```

以上查询返回类似以下输出的结果:

```
BP_NAME         TOTAL_P_READS
--------------- --------------------
IBMDEFAULTBP                     310
IBMSYSTEMBP16K                     0
IBMSYSTEMBP32K                     0
IBMSYSTEMBP4K                      0
IBMSYSTEMBP8K                      0

  5 record(s) selected.
```

用于计算缓冲池命中率的公式

缓冲池命中率反映在内存中查找查询所需数据（相对于必须从外部存储器读取）的程度。可使用基于缓冲池监视元素的公式来计算命中速率和命中率。

本地缓冲池

表 2126. 用于计算本地缓冲池命中率的公式. 显示的公式以百分比形式表示命中率。

页类型	用于计算缓冲池命中率的公式
数据页	(pool_data_lbp_pages_found - pool_async_data_lbp_pages_found - pool_temp_data_l_reads) / pool_data_l_reads) × 100
索引页	((pool_index_lbp_pages_found - pool_async_index_lbp_pages_found - pool_temp_index_l_reads) / pool_index_l_reads) × 100
临时数据页	((pool_temp_data_l_reads - pool_temp_data_p_reads) / pool_temp_data_l_reads) × 100
临时索引页	((pool_temp_index_l_reads - pool_temp_index_p_reads) / pool_temp_index_l_reads) × 100
XML 存储器对象 (XDA) 页	((pool_xda_gbp_l_reads - pool_xda_gbp_p_reads) / pool_xda_gbp_l_reads) × 100

表 2126. 用于计算本地缓冲池命中率的公式 (续). 显示的公式以百分比形式表示命中率。

页类型	用于计算缓冲池命中率的公式
整体命中率	((pool_data_lbp_pages_found + pool_index_lbp_pages_found + pool_xda_lbp_pages_found - pool_async_data_lbp_pages_found - pool_async_index_lbp_pages_found - pool_async_xda_lbp_pages_found) / (pool_data_l_reads + pool_index_l_reads + pool_xda_l_reads + pool_temp_data_l_reads + pool_temp_xda_l_reads + pool_temp_index_l_reads)) × 100

本地缓冲池（DB2 pureScale 环境）

在 DB2 pureScale 环境中计算 LBP 命中率时，还必须考虑独立于 GBP 的数据页命中率。成员请求数据页时，会先检查 LBP。如果 LBP 中发现该页，那么 **pool_data_lbp_pages_found** 监视元素会递增。如果 LBP 中发现该页，那么直接从磁盘读取，并且 **pool_data_p_reads** 监视元素会递增。但是，如果在本地缓冲池中发现依赖于 GBP 的页，那么 **pool_data_lbp_pages_found** 监视计数器也会递增。新监视元素 **pool_data_gbp_indep_pages_found_in_lbp** 是区分这些情况下的页访问计数的唯一方法。

表 2127. 独立于 GBP (LBP) 的命中率的公式. 显示的公式以百分比形式表示命中率。

页类型	用于计算缓冲池命中率的公式
数据页	(pool_data_gbp_indep_pages_found_in_lbp / (pool_data_gbp_indep_pages_found_in_lbp + pool_data_p_reads + pool_temp_data_p_reads - pool_data_gbp_p_reads) × 100
索引页	(pool_index_gbp_indep_pages_found_in_lbp / (pool_index_gbp_indep_pages_found_in_lbp + pool_index_p_reads + pool_temp_index_p_reads - pool_index_gbp_p_reads) × 100
XML 存储器对象 (XDA) 页	(pool_xda_gbp_indep_pages_found_in_lbp / (pool_xda_gbp_indep_pages_found_in_lbp + pool_xda_p_reads + pool_temp_xda_p_reads - pool_xda_gbp_p_reads) × 100

对于独立于 GBP 的页，调整 LBP 大小将影响命中率。对于独立于 GBP 的某些类型的操作，例如，临时页访问或 NOT LOGGED INITIALLY 操作，那么您要监视独立于 GBP 的页命中率。

或者，要计算独立于 GBP 的页代理程序的本地缓冲池命中率，可使用以下公式。

表 2128. 依赖于 GBP (LBP) 的命中率的公式. 显示的公式以百分比形式表示命中率。

页类型	用于计算缓冲池命中率的公式
数据页	((pool_data_lbp_pages_found - pool_data_gbp_indep_pages_found_in_lbp) / (pool_data_l_reads - pool_data_gbp_indep_pages_found_in_lbp - (pool_data_p_reads - pool_data_gbp_p_reads))) × 100
索引页	((pool_index_lbp_pages_found - pool_index_gbp_indep_pages_found_in_lbp) / (pool_index_l_reads - pool_index_gbp_indep_pages_found_in_lbp - (pool_index_p_reads - pool_index_gbp_p_reads))) × 100
XML 存储器对象 (XDA) 页	((pool_xda_lbp_pages_found - pool_xda_gbp_indep_pages_found_in_lbp) / (pool_xda_l_reads - pool_xda_gbp_indep_pages_found_in_lbp - (pool_xda_p_reads - pool_xda_gbp_p_reads))) × 100

您需要比较 LBP 命中率与 GBP 命中率以决定如何调整这两个缓冲池或在调整 GBP 和 LBP 后验证结果。

组缓冲池（DB2 pureScale 环境）

用于计算 DB2 pureScale环境中的组缓冲池命中率的公式与其他 DB2 环境中使用的命中率公式不同。不同是因为集群高速缓存设施中的组缓冲池使用每个成员中的本地缓

冲池来检索数据页。以下基于缓冲池监视元素的公式可用于计算本地缓冲池和组缓冲池的数据页、索引页和 XML 存储器对象页的命中率。

表 2129. 用于计算组缓冲池 (GBP) 命中率的公式. 显示的公式以百分比形式表示命中率。

页类型	用于计算缓冲池命中率的公式
数据页	((pool_data_gbp_l_reads - pool_data_gbp_p_reads) / pool_data_gbp_l_reads) × 100
索引页	((pool_index_gbp_l_reads - pool_index_gbp_p_reads) / pool_index_gbp_l_reads) × 100
XML 存储器对象 (XDA) 页	((pool_xda_gbp_l_reads - pool_xda_gbp_p_reads) / pool_xda_gbp_l_reads) × 100
整体命中率	((pool_data_gbp_l_reads + pool_index_gbp_l_reads + pool_xda_gbp_l_reads - pool_data_gbp_p_reads - pool_index_gbp_p_reads - pool_xda_gbp_p_reads) / (pool_data_gbp_l_reads + pool_index_gbp_l_reads + pool_xda_gbp_l_reads)) x 100

除先前用于计算缓冲池命中率的公式之外，还可使用以下公式来显示在 GBP 中发现预取页的次数的百分比:

数据页的预取

((pool_async_data_gbp_l_reads - pool_async_data_gbp_p_reads) / pool_async_data_gbp_l_reads) × 100

索引页的预取

((pool_async_index_gbp_l_reads - pool_async_index_gbp_p_reads) / pool_async_index_gbp_l_reads) × 100

XML 存储器对象 (XDA) 页的预取

((pool_async_xda_gbp_l_reads - pool_async_xda_gbp_p_reads) / pool_async_xda_gbp_l_reads) × 100

计算 DB2 pureScale 环境中的缓冲池命中率

计算 DB2 pureScale实例的缓冲池命中率可帮助您了解是否有机会调整缓冲池以提高 I/O 效率。

开始之前

确定您所关心的比率。如果要查看实例中所有成员中的比率，请考虑制定 SQL 以使用 SUM 聚集函数来聚集成员中的数据。如果您只想查看特定成员的数据，那么可在 MON_GET_BUFFERPOOL 表函数中指定要查看其数据的成员。

过程

要计算缓冲池命中率，请执行以下步骤:

1. 检索必需监视元素的信息。 此示例使用 MON_GET_BUFFERPOOL 表函数来检索监视元素，这些监视元素包含计算 GBP、**pool_data_gbp_l_reads** 和 **pool_data_gbp_p_reads** 的数据页的命中率时所需的值。

```
SELECT varchar(bp_name,20) AS bp_name,
       pool_data_gbp_l_reads,
       pool_data_gbp_p_reads,
       member
FROM TABLE(MON_GET_BUFFERPOOL('',-2))
```

以上查询返回类似以下示例的数据:

```
BP_NAME             POOL_DATA_GBP_L_READS POOL_DATA_GBP_P_READS MEMBER
------------------- --------------------- --------------------- ------
IBMDEFAULTBP                      1814911                456990      1
```

```
IBMSYSTEMBP4K                          0                   0      1
IBMSYSTEMBP8K                          0                   0      1
IBMSYSTEMBP16K                         0                   0      1
IBMSYSTEMBP32K                         0                   0      1
IBMDEFAULTBP                     1807959              455287      3
IBMSYSTEMBP4K                          0                   0      3
IBMSYSTEMBP8K                          0                   0      3
IBMSYSTEMBP16K                         0                   0      3
IBMSYSTEMBP32K                         0                   0      3
IBMDEFAULTBP                     1813932              455225      2
IBMSYSTEMBP4K                          0                   0      2
IBMSYSTEMBP8K                          0                   0      2
IBMSYSTEMBP16K                         0                   0      2
IBMSYSTEMBP32K                         0                   0      2
IBMDEFAULTBP                     1113396              278845      0
IBMSYSTEMBP4K                          0                   0      0
IBMSYSTEMBP8K                          0                   0      0
IBMSYSTEMBP16K                         0                   0      0
IBMSYSTEMBP32K                         0                   0      0

  20 record(s) selected.
```

要点: 在以上示例中，可见到针对临时缓冲池报告的数据全部显示为零。这并非巧合；在 DB2 pureScale实例中，临时对象和表空间在它们的关联成员的本地。它们在 CF 上不使用 GBP。

2. 使用针对监视元素返回的值来计算命中率。 用于计算 GBP 的命中率（表示为百分比）的公式为

   ((pool_data_gbp_l_reads - pool_data_gbp_p_reads) ÷ pool_data_gbp_l_reads) × 100

   所以，使用对步骤 第 1381 页的 1 中的监视元素返回的数据:

(((1,814,911+1,807,959 + 1,813,932+1,113,396) - (456,990+455,287 + 455,225+278,845)) ÷ (1,814,911+1,807,959 + 1,813,932+1,113,396)) × 100
= ((6,550,198 - 1,646,347) ÷ 6,550,198) × 100
= 74.9%

   在此示例中，GBP 的命中率为 74.9%

   注: 查询输出中显示的值仅供展示。

示例

示例 1: 查找所有成员中的整体命中速率

此示例与上一个过程中显示的示例类似，只是它使用聚集函数来提供所有成员中的整体命中速率。

```
SELECT VARCHAR(BP_NAME,20) AS BP,
       SUM(POOL_DATA_GBP_L_READS) AS POOL_DATA_GBP_L_READS,
       SUM(POOL_DATA_GBP_P_READS) AS POOL_DATA_GBP_L_READS
FROM TABLE(MON_GET_BUFFERPOOL('',-2))
GROUP BY BP_NAME
```

结果:

```
BP                   POOL_DATA_GBP_L_READS POOL_DATA_GBP_L_READS
-------------------- --------------------- ---------------------
IBMDEFAULTBP                       6550198               1646347
IBMSYSTEMBP16K                           0                     0
IBMSYSTEMBP32K                           0                     0
IBMSYSTEMBP4K                            0                     0
IBMSYSTEMBP8K                            0                     0

  5 record(s) selected.
```

示例 2: 确定所有数据、索引和 XML 存储器对象 (XDA) 页的 GBP 命中率

此示例使用 **MON_GET_BUFFERPOOL** 表函数来检索必需监视元素中包含的数据并针对每个成员计算命中率。要计算所有数据、索引和 XDA 页的 GBP 命中率，请使用以下公式:

((pool_data_gbp_l_reads + pool_index_gbp_l_reads+pool_xda_gbp_l_reads) - (pool_data_gbp_p_reads + pool_index_gbp_p_reads+pool_xda_gbp_p_reads)) ÷ (pool_data_gbp_l_reads + pool_index_gbp_l_reads+pool_xda_gbp_l_reads) × 100

```
WITH BPMETRICS AS (
     SELECT BP_NAME,
            POOL_DATA_GBP_L_READS +
              POOL_INDEX_GBP_L_READS +
              POOL_XDA_GBP_L_READS
            AS LOGICAL_READS,
            POOL_DATA_GBP_P_READS +
              POOL_INDEX_GBP_P_READS +
              POOL_XDA_GBP_P_READS
            AS PHYSICAL_READS,
            MEMBER
     FROM TABLE(MON_GET_BUFFERPOOL('',-2)) AS METRICS)
     SELECT VARCHAR(BP_NAME,20) AS BP_NAME,
            LOGICAL_READS,
            PHYSICAL_READS,
     CASE WHEN LOGICAL_READS > 0
          THEN DEC(((
            FLOAT(LOGICAL_READS) - FLOAT(PHYSICAL_READS)) /
            FLOAT(LOGICAL_READS))
            * 100,5,2)
         ELSE NULL END AS HIT_RATIO,
         MEMBER
     FROM BPMETRICS
```

结果:

```
BP_NAME              LOGICAL_READS        PHYSICAL_READS       HIT_RATIO MEMBER
-------------------- -------------------- -------------------- --------- ------
IBMDEFAULTBP                      5730213               617628     89.22      1
IBMSYSTEMBP4K                           0                    0         -      1
IBMSYSTEMBP8K                           0                    0         -      1
IBMSYSTEMBP16K                          0                    0         -      1
IBMSYSTEMBP32K                          0                    0         -      1
IBMDEFAULTBP                      5724845               615395     89.25      3
IBMSYSTEMBP4K                           0                    0         -      3
IBMSYSTEMBP8K                           0                    0         -      3
IBMSYSTEMBP16K                          0                    0         -      3
IBMSYSTEMBP32K                          0                    0         -      3
IBMDEFAULTBP                      5731714               615814     89.25      2
IBMSYSTEMBP4K                           0                    0         -      2
IBMSYSTEMBP8K                           0                    0         -      2
IBMSYSTEMBP16K                          0                    0         -      2
IBMSYSTEMBP32K                          0                    0         -      2
IBMDEFAULTBP                      5024809               409159     91.85      0
IBMSYSTEMBP4K                           0                    0         -      0
IBMSYSTEMBP8K                           0                    0         -      0
IBMSYSTEMBP16K                          0                    0         -      0
IBMSYSTEMBP32K                          0                    0         -      0

  20 record(s) selected.
```

示例 3: 使用 SUM 聚集函数来计算整体命中率

还可使用 SUM 聚集函数来计算所有成员中的整体命中率，如下所示:

```
WITH BPMETRICS AS (
     SELECT SUM(POOL_DATA_GBP_L_READS) +
       SUM(POOL_INDEX_GBP_L_READS) +
       SUM(POOL_XDA_GBP_L_READS)
     AS LOGICAL_READS,
     SUM(POOL_DATA_GBP_P_READS) +
       SUM(POOL_INDEX_GBP_P_READS) +
       SUM(POOL_XDA_GBP_P_READS)
     AS PHYSICAL_READS
     FROM TABLE(MON_GET_BUFFERPOOL('',-2)) AS METRICS)
     SELECT LOGICAL_READS,
            PHYSICAL_READS,
     CASE WHEN LOGICAL_READS > 0
     THEN DEC(((FLOAT(LOGICAL_READS) - FLOAT(PHYSICAL_READS)) /
```

```
        FLOAT(LOGICAL_READS))
        * 100,5,2)
      ELSE NULL END AS HIT_RATIO
FROM BPMETRICS
```

结果:

```
LOGICAL_READS        PHYSICAL_READS       HIT_RATIO
-------------------- -------------------- ---------
           22211581              2255996     89.84

  1 record(s) selected.
```

下一步做什么

如果命中率看起来很低，或者它们随时间变化下降，那么您可能要在成员和/或 CF 上增加缓冲池的大小。如果见到低于 DB2 pureScale实例整体的 LBP 的预期命中率，请分别查看每个成员的命中速率，因为每个成员上的缓冲池可能会有不同大小。如果一个成员上的 LBP 较小，那么可能会对实例的平均命中速率产生不良影响。

提示: 命中率可能会根据许多因素（例如，数据库中的数据特性、对其运行的查询以及硬件和软件配置）而变化。一般来讲，缓冲池命中率越高，反映查询性能越好。如果发现命中率好像很低，或者随时间变化不断下降，那么增加缓冲池大小会有帮助。要增加组缓冲池的大小，请在 CF 上调整 **cf_gbp_sz** 配置参数。要调整本地缓冲池，请在具有需要校正的缓冲池的成员上运行 **ALTER BUFFERPOOL** 语句。

DB2 pureScale 环境中的锁定监视概述

与传统 DB2 环境一样，DB2 pureScale 环境中的锁定管理对维护数据完整性和高并行级别极为重要。DB2 pureScale 环境中的成员中的锁定由集群高速缓存设施的全局锁定管理器 (GLM) 组件管理。监视 DB2 pureScale 环境中的锁定涉及查看可能在 成员中持有的锁定以及成员之间的锁定等待。

在 DB2 pureScale 环境中，不同成员使用相同数据的事实导致可能出现另一类型的数据争用: 两个成员想要更新同一对象。成员需要针对对象的锁定时，成员内的局部锁定管理器 (LLM) 组件与全局锁定管理器 (GLM) 配合工作: 如果 LLM 尚未持有针对所涉及的对象的锁定，那么 LLM 会向 GLM 请求锁定。这样一来，GLM 会调解向不同成员发出的针对锁定的请求。

在 DB2 pureScale实例的全局级别查看时，**locks_held** 或 **lock_wait_time** 之类的监视元素会报告有关实例中的*所有*锁定（成员内部及之间的锁定）的数据。专门为 DB2 pureScale Feature 添加的监视元素可用于仅检查成员之间的锁定等待。

成员之间的锁定请求

在 DB2 pureScale 环境中，一个成员上的应用程序可能请求针对当前被另一成员锁定的对象的锁定。DB2 pureScale Feature 引入了专门报告有关成员间锁定的信息的监视元素。

在传统 DB2 环境中通常会这样，DB2 pureScale 环境中的成员*内*的处理可能导致对象锁定（一个应用程序尝试执行的操作与另一个应用程序正执行的操作不兼容时）。例如，两个应用程序同时尝试更新同一数据行时可能会发生这种情况。通过使用锁定事件监视器来查看与锁定相关的信息，可监视成员内此类型的锁定的发生程度。在 DB2 pureScale 环境中，成员之*间*还会发生锁定等待（一个成员请求针对另一成员上的应用程序当前锁定的对象的锁定）。所以，在 DB2 pureScale 环境中，除了检查各成员上的锁定，您可能还要查看跨成员锁定信息。

成员之间的锁定等待

以下监视元素仅报告应用程序等待另一成员持有的锁定的时间部分:

- lock_wait_time_global
- lock_wait_time_global_top
- lock_waits_global
- lock_timeouts_global

从 DB2 pureScale实例的透视图整体查看这些监视元素时，这些监视元素报告的时间作为整体 **lock_wait_time** 和 **lock_wait_time_top** 监视元素的部分包括。同样，在实例级别整体查看这些元素的，锁定等待数和超时数作为 **lock_waits** 和 **lock_timeouts** 监视元素的一部分包括。

以下场景说明这些跨成员监视元素或全局锁定等待监视元素与报告成员内的锁定的监视元素之间的关系。

1. 成员 1 上的应用程序 1 持有针对行的共享 (S) 锁定。
2. 成员 2 上的应用程序 2 请求针对同一行的互斥 (X) 锁定。应用程序 2 被强制等待，因为该行当前被应用程序 1 锁定。
3. 2 ms 后，成员 2 上的应用程序 3 请求针对同一行的互斥 (X) 锁定。应用程序 3 也被强制等待。
4. 8 ms 后，应用程序 1 释放其锁定，应用程序 2 获取其锁定。
5. 5 ms 后，应用程序 2 释放其锁定，应用程序 3 获取其锁定。

等待锁定所耗的总时间正如 **lock_wait_time** 所报告为 23 ms; 应用程序 2 必须等待的总时间为 10 ms，应用程序 3 必须等待的总时间为 13 ms。但是，等待成员*之间*的锁定所耗的时间量 **lock_wait_time_global** 只有 10 ms，因为此等待时间只是涉及一个成员（此成员等待一个成员持有的锁定）的整体锁定等待时间的部分。

同样，**lock_waits_global** 报告的持有锁定数为 1。应用程序 2 等待应用程序 1 被视为一个成员等待另一个成员。即使应用程序 3 被成员 1 上的应用程序持有（在它等待的时间的一部分），那么此锁定等待不会被视为成员之间的锁定等待，因为它从成员 2 上的本地锁定管理器获取其锁定。

报告持有锁定的应用程序

锁定事件监视器显示有关持有锁定的应用程序的信息。一般来讲，不管应用程序正在哪个成员上运行，锁定事件监视器都会报告应用程序信息。但是，极少情况下可能无法确定哪个应用程序在远程成员上运行时正持有锁定。请考虑以下示例:

```
ALTER WORKLOAD finance COLLECT LOCK WAIT DATA WITH HISTORY AND VALUES
   FOR LOCKS WAITING MORE THAN 5 SECONDS
```

此语句导致持有超过 5 秒的针对“财务”工作负载的锁定被记录到锁定事件监视器中。现在考虑以下场景。

- 应用程序 1 在成员 1 上持有了锁定。
- 5 秒后，系统尝试将有关该锁定的信息写至锁定事件监视器，包括应用程序和成员信息。但是，如果锁定被应用程序 1 释放然后针对同一对象的锁定立即被授予另一成员上的另一应用程序，那么持有此锁定的应用程序可能被记录为“不可收集锁定持有者”。

还要注意的是，如果持有锁定的成员失败，那么锁定事件监视器会无法报告有关在该成员上运行的哪些应用程序可能已持有针对所需数据块的锁定的信息。

最后，使用建议不要使用的监视功能期间报告持有锁定的应用程序时有一些限制。有关更多信息，请参阅第 1393 页的『使用快照监视器来监视锁定』。

用于查看成员之间的锁定的监视元素

IBM DB2 pureScale Feature 添加了若干新的监视元素，可使用这些元素来监视 DB2 pureScale 环境中的锁定。这些监视元素专门报告有关 DB2 pureScale实例中的成员之间的锁定等待的信息。

与锁定相关的监视元素

以下监视元素可用于获取有关成员之间的锁定等待以及锁定升级的信息:

- 第 869 页的『lock_waits_global -“锁定等待全局”监视元素』
- 第 865 页的『lock_wait_time_global -“锁定等待时间全局”监视元素』
- 第 867 页的『lock_wait_time_global_top -“最长全局锁定等待时间”监视元素』
- 第 861 页的『lock_timeouts_global -“锁定超时全局”监视元素』
- 第 850 页的『lock_escals_maxlocks -“maxlocks 锁定升级数”监视元素』
- 第 849 页的『lock_escals_locklist -“locklist 锁定升级数”监视元素』
- 第 848 页的『lock_escals_global -“全局锁定升级数”监视元素』

以下元素虽然未直接与锁定相关，但可用于获取有关 DB2 pureScale实例中的语句、工作单元或工作负载等待集群高速缓存设施的程度的信息:

- 第 651 页的『cf_waits -“集群高速缓存设施等待次数”监视元素』
- 第 652 页的『cf_wait_time -“集群高速缓存设施等待时间”监视元素』

cf_wait_time 元素显示语句、工作单元或工作负载必须等待集群高速缓存设施处理请求的时间长度。将此元素与 cf_waits (cf_wait_time ÷ cf_waits) 配合使用来确定每个请求的平均等待时间。一般来讲，要由集群高速缓存设施处理的请求的平均等待时间大约为几毫秒。如果发现它们实际上较长（即，一个数量级或更多），那么可能存在 InfiniBand 设置问题。或者，集群高速缓存设施可能会被它无法及时处理的请求所淹没。

请参阅每个监视元素的参考主题，以了解可用于检查与该监视元素相关联的数据的监视界面。

还可使用锁定事件监视器支持的其他监视元素来查看有关成员*内*的锁定的信息。

页回收

DB2 pureScale实例中的不同成员可能需要访问另一成员已在使用的数据页。如果一个成员请求某个进程并且此进程被授予另一成员正在使用的页面，那么此进程被称为*页回收*。

如果不同成员需要访问同一数据页，那么集群高速缓存设施会管理哪些成员在何时访问此页。在某些情况下，集群高速缓存设施（又称为 CF）可能允许一个成员在另一个成员未使用完此页的情况下从此成员*回收*此页。以下示例说明页回收如何发生:

假定有两个 成员（M1 和 M2）计划更新同一数据页上的两个不同行。

1. M1 对数据页内的行 R1 执行更新。它被授予对包含该数据行的页的互斥存取权。
2. M2 需要对同一页的互斥存取权才能更新行 R2。它将此请求传递至 CF。系统处理此请求时，M2 等待。
3. CF 发现成员 M1 已具有对该页的互斥存取权。它向 M1 发出回收此页的请求。同时 M2 等待。
4. M1 通过将此页写回至 GBP 并释放此页来处理回收请求。（M1 保留了它可能具有的任何行或表锁定）。
5. CF 授予 M2 对该页的访问权。M2 从 GBP 读取此页以执行需要对此页执行的任何操作。

要记住的是，正如步骤 4 中所述，成员对行或表的任何锁定（用于更新）会保留下来，直到工作单元完成，即使另一成员在该工作单元结束前回收此页并开始使用此页也是如此。这样一来，不同成员可处理同一数据页而不会损害锁定完整性。如果两个成员需要对同一数据行的不兼容行锁定，那么与单个成员上的锁定管理一样，一个成员必须完成其处理，才允许第二个成员继续。

用于回收页的监视元素

IBM DB2 pureScale Feature添加了若干新监视元素，可使用这些监视元素来监视 DB2 pureScale实例中发生页回收的程度。

与页回收有关的监视元素

以下监视元素可用于获取有关 DB2 pureScale实例中发生页回收的程度的信息:

- 第 945 页的『page_reclaims_x -“互斥存取页回收数”监视元素』
- 第 946 页的『page_reclaims_initiated_s -“共享存取导致的页回收数”监视元素』
- 第 1158 页的『spacemappage_page_reclaims_x -“互斥存取导致的空间映射页回收数”监视元素』
- 第 1158 页的『spacemappage_page_reclaims_s -“共享存取的空间映射页回收数”监视元素』
- 第 946 页的『page_reclaims_initiated_x -“互斥存取导致的页回收数”监视元素』
- 第 946 页的『page_reclaims_initiated_s -“共享存取导致的页回收数”监视元素』
- 第 1159 页的『spacemappage_page_reclaims_initiated_x -“互斥存取导致的空间映射页回收数”监视元素』
- 第 1159 页的『spacemappage_page_reclaims_initiated_s -“共享存取导致的空间映射页回收数”监视元素』
- 第 1099 页的『reclaim_wait_time -“回收等待时间”监视元素』
- 第 1160 页的『spacemappage_reclaim_wait_time -“空间映射页回收等待时间”监视元素』

请参阅每个监视元素的参考主题，以了解可用于检查与该监视元素相关联的数据的监视界面。

监视成员之间的页回收

检查哪些存在特定应用程序或语句在消耗时间（除等待锁定所耗的时间外）时，DB2 pureScale 环境中的应用程序或语句可能需要等待正被另一成员使用的页变为可用。可使用页回收监视元素来查看此类型的等待可能影响系统上的吞吐量的程度。

关于此任务

要查看页回收统计信息，请使用 MON_GET_PAGE_ACCESS_INFO 表函数。此表函数返回有关成员请求当前正被其他成员使用的页的程度以及成员在其他成员的请求下释放这些页的程度的对象级别信息。还可检索所涉及的等待时间。

过程

1. 确定您要查看其结果的页的类型。 下面的示例使用 page_reclaims_x 和 page_reclaims_s 监视元素来检索有关针对所有数据页和索引页执行页回收的次数。
2. 制定一个使用 MON_GET_PAGE_ACCESS_INFO 表函数的 SQL 语句。 例如，要检索有关针对所有成员回收的数据页和索引页的信息，可构造类似如下的语句:

```
SELECT MEMBER,
       VARCHAR(TABNAME,30) AS TABLE,
       VARCHAR(OBJTYPE,8) AS OBJTYPE,
       PAGE_RECLAIMS_X,
       PAGE_RECLAIMS_S
FROM TABLE(MON_GET_PAGE_ACCESS_INFO('DTW','', -2))
WHERE PAGE_RECLAIMS_X !=0 OR  PAGE_RECLAIMS_S !=0
ORDER BY MEMBER ASC, PAGE_RECLAIMS_X ASC
```

3. 运行此查询。 在此情况下，返回的结果应类似以下示例:

```
MEMBER TABLE                          OBJTYPE  PAGE_RECLAIMS_X      PAGE_RECLAIMS_S
------ ------------------------------ -------- -------------------- --------------------
     0 CUSTOMER                       TABLE                     196                    0
     0 STOCK_1_250                    TABLE                     213                    0
     0 STOCK_1251_1500                TABLE                     237                    0
     0 STOCK_251_500                  TABLE                     239                    0
     0 STOCK_501_750                  TABLE                     245                    0
     0 STOCK_1751_2000                TABLE                     253                    0
     0 STOCK_2001_2250                TABLE                     254                    0
     0 STOCK_751_1000                 TABLE                     259                    0
     0 STOCK_1501_1750                TABLE                     269                    0
     0 STOCK_2251_2500                TABLE                     274                    0
     0 STOCK_251_500                  INDEX                     276                 2934
     0 STOCK_1001_1250                TABLE                     280                    0
     0 STOCK_1501_1750                INDEX                     284                 3070
     0 STOCK_501_750                  INDEX                     294                 3029
     0 STOCK_1_250                    INDEX                     296                 2916
     0 STOCK_751_1000                 INDEX                     301                 3056
     1 STOCK_1001_1250                TABLE                     247                    0
     1 STOCK_501_750                  TABLE                     255                    0
     1 STOCK_751_1000                 TABLE                     257                    0
     1 STOCK_1501_1750                TABLE                     257                    0
     1 STOCK_251_500                  INDEX                     287                 2921
     1 STOCK_1_250                    INDEX                     292                 2916
     1 STOCK_751_1000                 INDEX                     316                 3190
     1 STOCK_501_750                  INDEX                     319                 2956
     1 ORDERS                         INDEX                   42434                 1416
     1 ORDER_LINE                     INDEX                  116107                 3731
     2 CUSTOMER                       TABLE                     180                    0
     2 STOCK_2001_2250                TABLE                     221                    0
     .                                  .                       .                    .
     .                                  .                       .                    .
     .                                  .                       .                    .
     2 STOCK_1501_1750                TABLE                     240                    0
     2 STOCK_2251_2500                TABLE                     247                    0
     2 STOCK_1251_1500                TABLE                     268                    0
     2 STOCK_251_500                  INDEX                     276                 2976
     2 STOCK_1_250                    INDEX                     284                 2846
     2 STOCK_501_750                  TABLE                     285                    0
     2 STOCK_501_750                  INDEX                     293                 3143
     2 DISTRICT                       TABLE                   18402                    0
     2 ORDERS                         INDEX                   41581                 1474
```

```
2 ORDER_LINE                    INDEX                114442                3815
3 CUSTOMER                      TABLE                   159                   0
3 STOCK_251_500                 TABLE                   226                   0
3 ORDERS                        INDEX                 42192                1340
3 ORDER_LINE                    INDEX                115459                3871

112 record(s) selected.
```

注: 查询中的部分冗长输出已排除，以垂直省略号表示。

结果

在以上示例中，可见到数据页和索引页的信息是分别返回的。而且，指定了模式以便不导出与 SYSIBM 模式相关联的对象中的数据。

示例

示例 1: 检索页回收等待时间

以下 SQL 检索总回收页数以及所有成员中的总等待时间。

```
SELECT      SUM(PAGE_RECLAIMS_X+PAGE_RECLAIMS_S+SPACEMAPPAGE_PAGE_RECLAIMS_X
       +SPACEMAPPAGE_PAGE_RECLAIMS_S)AS PAGE_RECLAIMS,
     SUM(RECLAIM_WAIT_TIME) AS RECLAIM_WAIT_TIME
FROM TABLE(MON_GET_PAGE_ACCESS_INFO('','', -2))
```

此查询的结果应类似以下示例:

```
PAGE_RECLAIMS        RECLAIM_WAIT_TIME
-------------------- --------------------
                 156                   91

  1 record(s) selected.
```

（等待时间以毫秒为单位）

示例 2: 显示与最高回收页数相关联的 10 个表

此示例说明如何查看页回收涉及的表对象。

```
SELECT SUBSTR(TABSCHEMA,1,8) AS TABSCHEMA,
       SUBSTR(TABNAME,1,20) AS TABNAME,
       RECLAIM_WAIT_TIME,
       MEMBER,
       SUBSTR(OBJTYPE,1,10) AS OBJTYPE
FROM TABLE(MON_GET_PAGE_ACCESS_INFO(NULL,NULL,-2))
WHERE RECLAIM_WAIT_TIME > 0
 ORDER BY RECLAIM_WAIT_TIME DESC
 FETCH FIRST 10 ROWS ONLY
```

结果:

```
TABSCHEMA TABNAME              RECLAIM_WAIT_TIME    MEMBER OBJTYPE
--------- -------------------- -------------------- ------ ----------
DTW       ORDER_LINE                        1307192      1 INDEX
DTW       ORDER_LINE                        1250134      2 INDEX
DTW       ORDER_LINE                        1249452      0 INDEX
DTW       ORDER_LINE                        1159741      3 INDEX
DTW       DISTRICT                           827598      0 TABLE
DTW       DISTRICT                           785354      2 TABLE
DTW       DISTRICT                           767148      1 TABLE
DTW       DISTRICT                           687608      3 TABLE
DTW       ORDERS                             556538      0 INDEX
DTW       ORDERS                             539858      2 INDEX

  10 record(s) selected.
```

（等待时间以毫秒为单位）

示例 3: 显示导致最高回收页数的 10 个语句

此查询是基于先前示例的变体；在此情况下，此查询返回与最高回收页数相关联的 10 个语句:

```
SELECT SUBSTR(STMT_TEXT,1,50) AS STMT_TEXT,
       RECLAIM_WAIT_TIME
FROM TABLE(MON_GET_PKG_CACHE_STMT('D',NULL,NULL,-2))TABLE
       WHERE RECLAIM_WAIT_TIME > 0
ORDER BY RECLAIM_WAIT_TIME DESC
FETCH FIRST 10 ROWS ONLY
```

结果:

```
STMT_TEXT                                          RECLAIM_WAIT_TIME
-------------------------------------------------- --------------------
Select S_QUANTITY, S_DIST_01, S_DIST_02, S_DIST_03                796668
Select S_QUANTITY, S_DIST_01, S_DIST_02, S_DIST_03                785863
Select S_QUANTITY, S_DIST_01, S_DIST_02, S_DIST_03                746521
Select S_QUANTITY, S_DIST_01, S_DIST_02, S_DIST_03                623461
Insert into ORDER_LINE values (?, ?, ?, ?, ?, ?, ?                610602
Insert into ORDER_LINE values (?, ?, ?, ?, ?, ?, ?                522899
Insert into ORDER_LINE values (?, ?, ?, ?, ?, ?, ?                518076
Update DISTRICT set D_NEXT_O_ID = ? where D_W_ID =                 419022
Update DISTRICT set D_NEXT_O_ID = ? where D_W_ID =                 406028
Update DISTRICT set D_NEXT_O_ID = ? where D_W_ID =                 406006

  10 record(s) selected.
```

（等待时间以毫秒为单位）

示例 4: 显示导致最高回收页数的 10 个语句以及每个语句的每次执行的平均等待时间

在先前示例中，等待时间以每个语句的整体值表示。此查询不会考虑给定语句可能多次运行的事实。此示例说明如何检查导致最高回收页数的 10 个语句中每个语句的每次执行的平均等待时间:

```
SELECT SUBSTR(STMT_TEXT,1,75) AS STMT_TEXT,
       NUM_EXECUTIONS,
       RECLAIM_WAIT_TIME,
       DEC(FLOAT(RECLAIM_WAIT_TIME)/FLOAT(NUM_EXECUTIONS),10,8)
         AS AVG_WAIT_PEREXEC
FROM TABLE(MON_GET_PKG_CACHE_STMT('D',NULL,NULL,-2))TABLE
WHERE RECLAIM_WAIT_TIME > 0
ORDER BY AVG_WAIT_PEREXEC DESC
FETCH FIRST 10 ROWS ONLY
```

结果:

```
STMT_TEXT                                                                    NUM_EXECUTIONS       RECLAIM_WAIT_TIME    AVG_WAIT_PEREXEC
---------------------------------------------------------------------------- -------------------- -------------------- ----------------
Update DISTRICT set D_NEXT_O_ID = ? where D_W_ID = ? and D_ID = ?                        157173               419497       2.66901439
Update DISTRICT set D_NEXT_O_ID = ? where D_W_ID = ? and D_ID = ?                        155752               397870       2.55450973
Update DISTRICT set D_NEXT_O_ID = ? where D_W_ID = ? and D_ID = ?                        155352               385613       2.48218883
Update DISTRICT set D_NEXT_O_ID = ? where D_W_ID = ? and D_ID = ?                        155151               347847       2.24199006
Insert into ORDERS values (?, ?, ?, ?, ?, ?, ?, ?)                                      157173               259076       1.64834927
Insert into ORDERS values (?, ?, ?, ?, ?, ?, ?, ?)                                      155752               253548       1.62789562
Insert into ORDERS values (?, ?, ?, ?, ?, ?, ?, ?)                                      155352               232300       1.49531386
Insert into ORDERS values (?, ?, ?, ?, ?, ?, ?, ?)                                      155151               219607       1.41544044
Delete from NEW_ORDER where NO_W_ID = ? and NO_D_ID = ? and NO_O_ID = ?                  152968               106525       0.69638747
Delete from NEW_ORDER where NO_W_ID = ? and NO_D_ID = ? and NO_O_ID = ?                  152591               101367       0.66430523

  10 record(s) selected.
```

（等待时间以毫秒为单位）

此查询的略微不同版本显示每个语句执行时所耗的时间:

```
SELECT SUBSTR(STMT_TEXT,1,75) AS STMT_TEXT,
       NUM_EXECUTIONS,
       RECLAIM_WAIT_TIME,
       DEC(FLOAT(RECLAIM_WAIT_TIME)/FLOAT(NUM_EXECUTIONS),10,8) AS AVG_EXEC_TIME
FROM TABLE(MON_GET_PKG_CACHE_STMT('D',NULL,NULL,-2))TABLE
WHERE RECLAIM_WAIT_TIME > 0
ORDER BY RECLAIM_WAIT_TIME DESC
FETCH FIRST 10 ROWS ONLY
```

结果:

```
STMT_TEXT                                                                        NUM_EXECUTIONS       RECLAIM_WAIT_TIME    AVG_EXEC_TIME
-------------------------------------------------------------------------------- -------------------- -------------------- -------------
Select S_QUANTITY, S_DIST_01, S_DIST_02, S_DIST_03, S_DIST_04, S_DIST_05, S             1555470               755544    0.48573357
Select S_QUANTITY, S_DIST_01, S_DIST_02, S_DIST_03, S_DIST_04, S_DIST_05, S             1554405               754231    0.48522167
Select S_QUANTITY, S_DIST_01, S_DIST_02, S_DIST_03, S_DIST_04, S_DIST_05, S             1570256               741047    0.47192750
Select S_QUANTITY, S_DIST_01, S_DIST_02, S_DIST_03, S_DIST_04, S_DIST_05, S             1550835               707148    0.45597887
Insert into ORDER_LINE values (?, ?, ?, ?, ?, ?, ?, ?, ?, ?)                           1554392               508568    0.32718130
Insert into ORDER_LINE values (?, ?, ?, ?, ?, ?, ?, ?, ?, ?)                           1555454               497197    0.31964751
Insert into ORDER_LINE values (?, ?, ?, ?, ?, ?, ?, ?, ?, ?)                           1570245               493692    0.31440444
Insert into ORDER_LINE values (?, ?, ?, ?, ?, ?, ?, ?, ?, ?)                           1550813               465049    0.29987432
Update DISTRICT set D_NEXT_O_ID = ? where D_W_ID = ? and D_ID = ?                        157145               419283    2.66812816
Update DISTRICT set D_NEXT_O_ID = ? where D_W_ID = ? and D_ID = ?                        155719               397364    2.55180164

  10 record(s) selected.
```

（等待时间以毫秒为单位）

第 14 章 在 DB2 pureScale 环境中使用建议不要使用的监视功能部件

IBM DB2 pureScale Feature使用一组丰富的监视元素来扩展 DB2 监视基础结构，这些监视元素可用于检索有关 DB2 pureScale实例的信息。但是，使用建议不要使用的监视界面来检索和解释监视数据时，需要注意一些局限性。

本主题描述使用下列任何不推荐功能时要注意的局限性:

- 『使用快照监视器来监视锁定』
- 第 1394 页的『死锁事件监视器』
- 第 1397 页的『LIST TABLESPACES 和 LIST TABLESPACE CONTAINERS 命令』

使用快照监视器来监视锁定

如果使用快照监视命令、函数或视图来检查有关成员之间的锁定的信息，那么仅当持有锁定的应用程序在您获取快照的成员上运行时，才会显示有关该应用程序的详细信息。否则，持有锁定的应用程序的标识被报告为 REMOTE APPLICATION，其他信息（例如，应用程序标识和锁定方式）被省略。因此，考虑获取全局快照以便返回所有成员中的信息。

例如，图 23 显示 **GET SNAPSHOT FOR APPLICATION** 命令的输出，其中持有此锁定的应用程序在运行此命令的同一成员上:

```
ID of agent holding lock                = 73
Application ID holding lock             = *N0.user1.080616184956
Database partition lock wait occurred on = 0
Lock name                               = 0x02000400000000000000000054
Lock attributes                         = 0x00000000
Release flags                           = 0x00000000
Lock object type                        = Table
Lock mode                               = Exclusive Lock (X)
Lock mode requested                     = Share Lock (S)
Name of tablespace holding lock         = USERSPACE1
Schema of table holding lock            = USER1
Name of table holding lock              = T1
Data Partition Id of table holding lock  = 0
Lock wait start timestamp               = 06/16/2009 14:50:26.744694
```

图 23. GET SNAPSHOT FOR APPLICATION 命令的输出 - 在持有锁定的成员上运行的命令.
在此示例中，持有此锁定的应用程序正在运行 **GET SNAPSHOT** 命令的同一成员上运行:

但是，如果此锁定由远程成员上的应用程序持有，那么同一报告应类似 第 1394 页的图 24 中所示:

```
Application ID holding lock             = REMOTE APPLICATION
Database partition lock wait occurred on = 0
Lock name                               = 0x020004000000000000000054
Lock attributes                         = 0x00000000
Release flags                           = 0x00000000
Lock object type                        = Table
Lock mode requested                     = Share Lock (S)
Name of tablespace holding lock         = USERSPACE1
Schema of table holding lock            = USER1
Name of table holding lock              = T1
Data Partition Id of table holding lock  = 0
Lock wait start timestamp               = 06/16/2009 14:50:26.744694
```

图 24. GET SNAPSHOT FOR APPLICATION 命令的输出 - 持有锁定的成员以外的成员上运行的命令. 在此示例中，持有锁定的应用程序在正运行 **GET SNAPSHOT** 命令的另一成员上运行。**持有锁定的代理程序的标识**和**锁定方式**行被省略。而且，**持有锁定的应用程序标识**显示为 REMOTE APPLICATION。

如果获取全局快照，那么会返回所有成员的数据。可通过检查快照输出中的锁定名称来确定持有锁定的位置。如果粗略浏览每个成员的报告，那么可迅速找到持有所提到锁定的应用程序。

死锁事件监视器

死锁发生时，死锁检测器会生成事件监视器跟踪死锁时使用的信息。**CREATE EVENT MONITOR ... FOR DEADLOCKS** 命令（不推荐使用）可能不会显示 DB2 pureScale 环境中的成员之间的死锁的某些详细信息。对于死锁事件报告，有关受此死锁影响的应用程序的详细信息可能不会显示在 **dbevmon** 工具的输出中，如图 25和图 26中所示:

```
3) Deadlock Event ...
  Deadlock ID:   1
  Deadlock node: 0
  Number of applications deadlocked: 2
  Deadlock detection time: 06/17/2009 14:46:22.543136
  Rolled back Appl participant no: 2
```

*图 25. 样本 **db2evmon** 输出和 DB2 pureScale 实例*

```
3) Deadlock Event ...
  Deadlock ID:   1
  Deadlock node: 0
  Number of applications deadlocked: 2
  Deadlock detection time: 06/17/2009 14:46:22.543136
  Rolled back Appl participant no: 2
  Rolled back Appl Id: *N0.finance.081217170042
  Rolled back Appl seq number: : 0001
  Rolled back Appl handle: 66
```

*图 26. 样本 **db2evmon** 输出和所有其他类型的 DB2 实例*

（是否显示应用程序详细信息取决于若干因素，所有因素都不受用户控制）。但是，可通过使针对死锁标识所报告的数据和回滚应用程序参与者编号与 **dbevmon** 输出的“Deadlocked Connection”部分找到的信息相关联来确定死锁涉及的应用程序。

同样，输出的“Deadlocked Connection”部分未包含持有所请求锁定的应用程序标识、序列号或锁定方式。但是，会显示死锁标识、成员标识、锁定名称、锁定时间戳记和持有此锁定的应用程序的参与者编号。（成员标识在 **dbevmon** 工具中的输出中显示为

“Deadlock node”。）可使用此信息来关联导致争用的应用程序。来自 **db2evmon** 工具的以下样本输出说明了此行为。用于关联死锁涉及的应用程序的信息加上了下划线:

```
5) Deadlock Event ...
 Deadlock ID:   1
 Deadlock node: 0
 Number of applications deadlocked: 2
 Deadlock detection time: 12/17/2008 12:01:12.735436
 Rolled back Appl participant no: 2
 Rolled back Appl Id: *N0.finance.081217170042
 Rolled back Appl seq number: : 0001
 Rolled back Appl handle: 66

6) Connection Header Event ...
 Appl Handle: 66
 Appl Id: *N0.finance.081217170042
 Appl Seq number: 00001
 DRDA AS Correlation Token: *N0.finance.081217170042
 Program Name    : db2bp
 Authorization Id: FINANCE
 Execution Id    : finance
 Codepage Id: 1208
 Territory code: 1
 Client Process Id: 7201
 Client Database Alias: A
 Client Product Id: SQL09070
 Client Platform: Unknown
 Client Communication Protocol: Local
 Client Network Name: so2.torolab.ibm.com
 Connect timestamp: 12/17/2008 12:00:42.176747

 7) Deadlocked Connection ...
 Deadlock ID:   1
 Deadlock Node: 0
 Participant no.: 2
 Participant no. holding the lock: 1
 Appl Id: *N0.finance.081217170042
 Appl Seq number: 00001
 Appl Id of connection holding the lock: REMOTE_APPLICATION
 Lock wait start time: 12/17/2008 12:01:01.607230
 Lock Name       : 0x02000500040000010000000052
 Lock Attributes : 0x00000000
 Release Flags   : 0x00000000
 Lock Count      : 0
 Hold Count      : 0
 Current Mode    : none
 Deadlock detection time: 12/17/2008 12:01:17.730069
 Table of lock waited on      : T2
 Schema of lock waited on     : FINANCE
 Data partition id for table  : 0
 Tablespace of lock waited on : USERSPACE1
 Type of lock: Row
 Mode application requested on lock: NS  - Share (CS/RS)
 Node lock occured on: 2
 Lock object name: 16777220
 Application Handle: 66
 Deadlocked Statement:
   Type     : Dynamic
   Operation: Fetch
   Section  : 201
   Creator  : NULLID
   Package  : SQLC2G17
   Cursor   : SQLCUR201
   Cursor was blocking: FALSE
   Text     : select * from t2

 List of Locks:
...

     Database partition          : 0
     Lock Name                   : 0x020004000100FFFFFF81000000000052
     Lock Attributes             : 0x00000008
     Release Flags               : 0x40000000
     Lock Count      : 1
     Hold Count                  : 0
     Lock Object Name            : 8454145
     Object Type                 : Row
     Tablespace Name : USERSPACE1
     Table Schema                : FINANCE
```

```
     Table Name                 : T1
     Data partition id          : 0
     Mode                       : X   - Exclusive

     Database partition         : 0
     Lock Name                  : 0x020005000000000000000000054
     Lock Attributes            : 0x00000000
     Release Flags              : 0x00000001
     Lock Count      : 1
     Hold Count                 : 0
     Lock Object Name           : 5
     Object Type                : Table
     Tablespace Name : USERSPACE1
     Table Schema               : FINANCE
     Table Name                 : T2
     Data partition id          : 0
     Mode                       : IS  - Intent Share
...

 Locks Held:    6
 Locks in List: 6
 Locks Displayed: 6

8) Connection Header Event ...
 Appl Handle: 131137
 Appl Id: *N2.finance.081217170053
 Appl Seq number: 00001
 DRDA AS Correlation Token: *N2.finance.081217170053
 Program Name    : db2bp
 Authorization Id: finance
 Execution Id    : finance
 Codepage Id: 1208
 Territory code: 1
 Client Process Id: 7260
 Client Database Alias: A
 Client Product Id: SQL09070
 Client Platform: Unknown
 Client Communication Protocol: Local
 Client Network Name: so2.torolab.ibm.com
 Connect timestamp: 12/17/2008 12:00:43.542242
 9) Deadlocked Connection ...
 Deadlock ID:   1
 Deadlock Node: 0
 Participant no.: 1
 Participant no. holding the lock: 2
 Appl Id: *N2.finance.081217170053
 Appl Seq number: 00001
 Appl Id of connection holding the lock: REMOTE_APPLICATION
 Lock wait start time: 12/17/2008 12:00:57.844388
 Lock Name       : 0x020004000100FFFFFF81000000000052
 Lock Attributes : 0x00000000
 Release Flags   : 0x00000000
 Lock Count      : 0
 Hold Count      : 0
 Current Mode    : none
 Deadlock detection time: 12/17/2008 12:01:17.744611
 Table of lock waited on      : T1
 Schema of lock waited on     : FINANCE
 Data partition id for table  : 0
 Tablespace of lock waited on : USERSPACE1
 Type of lock: Row
 Mode application requested on lock: NS  - Share (CS/RS)
 Node lock occured on: 0
 Lock object name: 8454145
 Application Handle: 131137
 Deadlocked Statement:
   Type     : Dynamic
   Operation: Fetch
   Section  : 201
   Creator  : NULLID
   Package  : SQLC2G17
   Cursor   : SQLCUR201
   Cursor was blocking: FALSE
   Text     : select * from t1
 List of Locks:
...
     Database partition         : 2
     Lock Name                  : 0x020005000400000010000000052
     Lock Attributes            : 0x00000008
     Release Flags              : 0x40000000
```

```
    Lock Count      : 1
    Hold Count                  : 0
    Lock Object Name            : 16777220
    Object Type                 : Row
    Tablespace Name : USERSPACE1
    Table Schema                : FINANCE
    Table Name                  : T2
    Data partition id           : 0
    Mode                        : X   - Exclusive

    Database partition          : 2
    Lock Name                   : 0x020005000000000000000000054
    Lock Attributes             : 0x00000000
    Release Flags               : 0x40000000
    Lock Count      : 1
    Hold Count                  : 0
    Lock Object Name            : 5
    Object Type                 : Table
    Tablespace Name : USERSPACE1
    Table Schema                : FINANCE
    Table Name                  : T2
    Data partition id           : 0
    Mode                        : IX  - Intent Exclusive

    Database partition          : 2
    Lock Name                   : 0x020004000000000000000000054
    Lock Attributes             : 0x00000000
    Release Flags               : 0x00000001
    Lock Count      : 1
    Hold Count                  : 0
    Lock Object Name            : 4
    Object Type                 : Table
    Tablespace Name : USERSPACE1
    Table Schema                : FINANCE
    Table Name                  : T1
    Data partition id           : 0
    Mode                        : IS  - Intent Share

Locks Held:    6
Locks in List: 6
Locks Displayed: 6
```

LIST TABLESPACES 和 LIST TABLESPACE CONTAINERS 命令

LIST TABLESPACES 和 **LIST TABLESPACE CONTAINERS** 命令在 DB2 V9.7 中都不推荐使用。这些命令仅报告运行它们的成员已知的信息。它们不会从实例中的其他成员检索信息。所以，可能未准确报告某些数据（例如，表空间中的已使用页数）。改为使用 MON_GET_TABLESPACE 和 MON_GET_CONTAINER 表函数。

第 15 章 新的和已更改的监视元素

cf_wait_time -“集群高速缓存设施等待时间”监视元素

与集群高速缓存设施通信所耗的时间量。此时间未包括可能因为授予锁定或执行页回收之类的操作而请求的或因为通信而发生的任何处理所耗的时间。时间以毫秒计。

表 2130. 表函数监视信息

表函数	监视元素收集级别
MON_FORMAT_XML_METRICS_BY_ROW - 获取所有度量值的基于已格式化行的输出	不适用; 报告 XML 文档中作为格式化函数的输入提供的所有元素
MON_FORMAT_XML_TIMES_BY_ROW - - 获取基于已格式化行的组合层次结构等待时间和处理时间	不适用; 报告 XML 文档中作为格式化函数的输入提供的所有元素
MON_FORMAT_XML_WAIT_TIMES_BY _ROW - 获取等待次数的已格式化的基于行的输出	不适用; 报告 XML 文档中作为格式化函数的输入提供的所有元素
MON_GET_CONNECTION 表函数 - 获取连接度量值	REQUEST METRICS BASE
MON_GET_PKG_CACHE_STMT 表函数 - 获取程序包高速缓存中的 SQL 语句活动度量值	ACTIVITY METRICS BASE
MON_GET_SERVICE_SUBCLASS 表函数 - 获取服务子类度量值	REQUEST METRICS BASE
MON_GET_UNIT_OF_WORK 表函数 - 获取工作单元度量值	REQUEST METRICS BASE
MON_GET_WORKLOAD 表函数 - 获取工作负载度量值	REQUEST METRICS BASE

用法

此值是 DB2 与集群高速缓存设施通信时消耗在等待上的时间量的指示符。

cf_waits -“集群高速缓存设施等待次数”监视元素

DB2 数据库系统与集群高速缓存设施通信时等待的次数。

表 2131. 表函数监视信息

表函数	监视元素收集级别
MON_FORMAT_XML_METRICS_BY_ROW - 获取所有度量值的基于已格式化行的输出	不适用; 报告 XML 文档中作为格式化函数的输入提供的所有元素
MON_GET_CONNECTION 表函数 - 获取连接度量值	REQUEST METRICS BASE
MON_GET_PKG_CACHE_STMT 表函数 - 获取程序包高速缓存中的 SQL 语句活动度量值	ACTIVITY METRICS BASE

表 2131. 表函数监视信息 (续)

表函数	监视元素收集级别
MON_GET_SERVICE_SUBCLASS 表函数 - 获取服务子类度量值	REQUEST METRICS BASE
MON_GET_UNIT_OF_WORK 表函数 - 获取工作单元度量值	REQUEST METRICS BASE
MON_GET_WORKLOAD 表函数 - 获取工作负载度量值	REQUEST METRICS BASE

表 2132. 事件监视信息

事件类型	逻辑数据分组	监视开关
活动	event_activity（在 details_xml 文档中报告）	ACTIVITY METRICS BASE
统计信息	event_scstats（在 details_xml 文档中报告）	REQUEST METRICS BASE
统计信息	event_wlstats（在 details_xml 文档中报告）	REQUEST METRICS BASE
工作单元	-	始终收集
程序包高速缓存	-	始终收集
锁定	-	始终收集

configured_cf_gbp_size -“已配置的集群高速缓存设施组缓冲池大小”监视元素

使用 **cf_gbp_sz** 配置参数指定的已分配并保留的组缓冲池内存（以大小为 4 KB 的页计）。

表 2133. 表函数监视信息

表函数	监视元素收集级别
MON_GET_CF 表函数 - 获取 CF 度量值	ACTIVITY METRICS BASE

configured_cf_lock_size -“已配置的集群高速缓存设施锁定大小”监视元素

已配置的全局锁定内存（以大小为 4 KB 的页计）。此值是使用 **cf_lock_sz** 配置参数指定的。

表 2134. 表函数监视信息

表函数	监视元素收集级别
MON_GET_CF 表函数 - 获取 CF 度量值	ACTIVITY METRICS BASE

configured_cf_mem_size -“已配置的集群高速缓存设施内存大小”监视元素

为集群高速缓存设施配置的总内存大小（以大小为 4 KB 的页计）。此值是使用 **cf_mem_sz** 配置参数指定的。

表 2135. 表函数监视信息

表函数	监视元素收集级别
MON_GET_CF 表函数 - 获取 CF 度量值	ACTIVITY METRICS BASE

configured_cf_sca_size -“已配置的集群高速缓存设施共享通信区大小”监视元素

当前分配并保留的共享通信区内存（以大小为 4 KB 的页计）。此值是使用 **cf_sca_sz** 配置参数指定的。

表 2136. 表函数监视信息

表函数	监视元素收集级别
MON_GET_CF 表函数 - 获取 CF 度量值	ACTIVITY METRICS BASE

current_cf_gbp_size -“当前集群高速缓存设施组缓冲池大小”监视元素

当前正在集群高速缓存设施中使用的组缓冲池内存（以大小为 4 KB 的页计）。

表 2137. 表函数监视信息

表函数	监视元素收集级别
MON_GET_CF 表函数 - 获取 CF 度量值	ACTIVITY METRICS BASE

current_cf_lock_size -“当前集群高速缓存设施锁定大小”监视元素

当前正在使用的全局锁定内存（以大小为 4 KB 的页计）。

表 2138. 表函数监视信息

表函数	监视元素收集级别
MON_GET_CF 表函数 - 获取 CF 度量值	ACTIVITY METRICS BASE

current_cf_mem_size -“当前集群高速缓存设施内存大小”监视元素

当前正在使用的内存总计（以大小为 4 KB 的页计）。

表 2139. 表函数监视信息

表函数	监视元素收集级别
MON_GET_CF 表函数 - 获取 CF 度量值	ACTIVITY METRICS BASE

current_cf_sca_size -“当前集群高速缓存设施共享通信区大小”监视元素

当前正在使用的共享通信区内存（以大小为 4 KB 的页计）。

表 2140. 表函数监视信息

表函数	监视元素收集级别
MON_GET_CF 表函数 - 获取 CF 度量值	ACTIVITY METRICS BASE

db_name - 数据库名称监视元素

对其收集信息或应用程序连接至的数据库的真实名称。这是创建时给定的数据库名称。

表 2141. 表函数监视信息

表函数	监视元素收集级别
MON_GET_AUTO_MAINT_QUEUE 表函数 - 获取有关自动维护作业的信息	ACTIVITY METRICS BASE
MON_GET_CF 表函数 - 获取 CF 度量值	ACTIVITY METRICS BASE
MON_GET_MEMORY_POOL 表函数 - 获取内存池信息	ACTIVITY METRICS BASE
MON_GET_MEMORY_SET 表函数 - 获取内存集合信息	ACTIVITY METRICS BASE
MON_SAMPLE_SERVICE_CLASS_METRICS - 获取样本服务类度量值	ACTIVITY METRICS BASE
MON_SAMPLE_WORKLOAD_METRICS - 获取样本工作负载度量值	ACTIVITY METRICS BASE

表 2142. 快照监视信息

快照级别	逻辑数据分组	监视开关
数据库	dbase	基本
数据库	dbase_remote	基本
应用程序	appl_id_info	基本
应用程序	appl_remote	基本
表空间	tablespace_list	缓冲池
缓冲池	bufferpool	缓冲池
表	table_list	表
锁定	db_lock_list	基本
动态 SQL	dynsql_list	基本
DCS 数据库	dcs_dbase	基本
DCS 应用程序	dcs_appl_info	基本

表 2143. 事件监视信息

事件类型	逻辑数据分组	监视开关
数据库	event_dbheader	始终收集

用法

可使用此元素来标识应用数据的特定数据库。

对于未使用 DB2 Connect 连接至主机或 System i 数据库服务器的应用程序，可将此元素与 **db_path** 监视元素配合使用来唯一标识该数据库，并协助使监视器提供的信息的不同级别相关。

dbpartitionnum -“数据库分区号”监视元素

在分区数据库环境中，此监视元素是数据库成员的数字标识。对于 DB2 Enterprise Server Edition 和在 DB2 pureScale环境中，此值为 0。

表 2144. 表函数监视信息

表函数	监视元素收集级别
ADMIN_GET_INDEX_COMPRESS_INFO 表函数 - 返回压缩索引信息	ACTIVITY METRICS BASE
ADMIN_GET_INDEX_INFO 表函数 - 返回索引信息	ACTIVITY METRICS BASE
ADMIN_GET_MSGS 表函数 - 检索由通过 ADMIN_CMD 过程执行的数据移动实用程序生成的消息	ACTIVITY METRICS BASE
ADMIN_GET_STORAGE_PATHS 表函数 - 获取存储器组的存储器路径信息	ACTIVITY METRICS BASE
ADMIN_GET_TAB_COMPRESS_INFO 表函数 - 估算压缩节省量	ACTIVITY METRICS BASE
ADMIN_GET_TAB_DICTIONARY_INFO 表函数 - 报告现有表字典的属性	ACTIVITY METRICS BASE
ADMINTABINFO 管理视图和 ADMIN_GET_TAB_INFO 表函数 - 检索表大小和状态信息	ACTIVITY METRICS BASE
AUDIT_ARCHIVE 过程和表函数 - 归档审计日志文件	ACTIVITY METRICS BASE
DBCFG 管理视图和 DB_GET_CFG 表函数 - 检索数据库配置参数信息	ACTIVITY METRICS BASE
DBPATHS 管理视图和 ADMIN_LIST_DB_PATHS 表函数 - 检索数据库路径	ACTIVITY METRICS BASE
MON_GET_CONTAINER 表函数 - 获取表空间容器度量值	ACTIVITY METRICS BASE
MON_GET_REBALANCE_STATUS 表函数 - 获取表空间的重新平衡进度	ACTIVITY METRICS BASE
MON_GET_TABLE 表函数 - 获取表度量值	ACTIVITY METRICS BASE
MON_GET_TABLESPACE 表函数 - 获取表空间度量值	ACTIVITY METRICS BASE
PD_GET_DIAG_HIST 表函数 - 从给定设施返回记录	ACTIVITY METRICS BASE

表 2144. 表函数监视信息 (续)

表函数	监视元素收集级别
PDLOGMSGS_LAST24HOURS 管理视图和 PD_GET_LOG_MSGS 表函数 - 检索问题确定消息	ACTIVITY METRICS BASE
WLM_GET_QUEUE_STATS 表函数 - 返回阈值队列统计信息	ACTIVITY METRICS BASE
WLM_GET_SERVICE_CLASS_AGENTS 表函数 - 列示正在服务类中运行的代理程序	ACTIVITY METRICS BASE
WLM_GET_SERVICE_CLASS_WORKLOAD_OCCURRENCES 表函数 - 列示工作负载出现次数	ACTIVITY METRICS BASE
WLM_GET_SERVICE_SUBCLASS_STATS 表函数 - 返回服务子类的统计信息	ACTIVITY METRICS BASE
WLM_GET_SERVICE_SUPERCLASS_STATS 表函数 - 返回服务超类的统计信息	ACTIVITY METRICS BASE
WLM_GET_WORK_ACTION_SET_STATS 表函数 - 返回工作操作集统计信息	ACTIVITY METRICS BASE
WLM_GET_WORKLOAD_OCCURRENCE_ACTIVITIES 表函数 - 返回活动列表	ACTIVITY METRICS BASE
WLM_GET_WORKLOAD_STATS 表函数 - 返回工作负载统计信息	ACTIVITY METRICS BASE

用法

在 DB2 pureScale环境中，多个成员在单个分区上运行。在这类配置中运行时，系统中所有成员上的存储器物理属性（例如，表空间中的可用页数）是重复的。每个成员报告系统的总准确大小。在多分区配置中，每个分区中的值必须由用户关联才能了解系统的整体值。

dbpartitionnum 监视元素与 **data_partition_id** 监视元素不同，后者用于标识通过根据值对表中数据进行细分所创建的数据分区。

host_name -“主机名”监视元素

集群高速缓存设施进程所在的主机的名称。

表 2145. 表函数监视信息

表函数	监视元素收集级别
DB_MEMBERS 表函数	ACTIVITY METRICS BASE
ENV_GET_NETWORK_RESOURCES 表函数 - 返回网络适配器信息	ACTIVITY METRICS BASE
ENV_GET_SYSTEM_RESOURCES 表函数 - 返回系统信息	ACTIVITY METRICS BASE
MON_GET_CF 表函数 - 获取 CF 度量值	ACTIVITY METRICS BASE
MON_GET_MEMORY_POOL 表函数 - 获取内存池信息	ACTIVITY METRICS BASE

表 2145. 表函数监视信息 (续)

表函数	监视元素收集级别
MON_GET_MEMORY_SET 表函数 - 获取内存集合信息	ACTIVITY METRICS BASE

id -“集群高速缓存设施标识”监视元素

集群高速缓存设施标识，如 db2nodes.cfg 文件中的定义。

表 2146. 表函数监视信息

表函数	监视元素收集级别
DB2_MEMBER 管理视图、DB2_CF 管理视图和 DB2_GET_INSTANCE_INFO 表函数	ACTIVITY METRICS BASE
MON_GET_CF 表函数 - 获取 CF 度量值	ACTIVITY METRICS BASE
MON_GET_CF_CMD 表函数 - 获取集群高速缓存工具命令处理时间	ACTIVITY METRICS BASE
MON_GET_CF_WAIT_TIME 表函数 - 获取集群高速缓存工具命令等待时间	ACTIVITY METRICS BASE

lock_escals -“锁定升级次数”监视元素

锁定已从若干行锁定升级至表锁定的次数。

表 2147. 表函数监视信息

表函数	监视元素收集级别
MON_FORMAT_XML_METRICS_BY_ROW - 获取所有度量值的基于已格式化行的输出	不适用; 报告 XML 文档中作为格式化函数的输入提供的所有元素
MON_GET_ACTIVITY_DETAILS 表函数 - 获取完整的活动详细信息（在 XML 文档 DETAILS 中报告）	ACTIVITY METRICS BASE
MON_GET_CONNECTION 表函数 - 获取连接度量值	REQUEST METRICS BASE
MON_GET_CONNECTION_DETAILS 表函数 - 获取详细的连接度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE
MON_GET_PKG_CACHE_STMT 表函数 - 获取程序包高速缓存中的 SQL 语句活动度量值	ACTIVITY METRICS BASE
MON_GET_PKG_CACHE_STMT_DETAILS 表函数 - 获取程序包高速缓存条目的详细度量值	ACTIVITY METRICS BASE
MON_GET_SERVICE_SUBCLASS 表函数 - 获取服务子类度量值	REQUEST METRICS BASE
MON_GET_SERVICE_SUBCLASS_DETAILS 表函数 - 获取详细的服务子类度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE
MON_GET_TABLE 表函数 - 获取表度量值	DATA OBJECT METRICS EXTENDED

表 2147. 表函数监视信息 (续)

表函数	监视元素收集级别
MON_GET_TABLE_USAGE_LIST 表函数 - 从表用法列表返回信息	DATA OBJECT METRICS EXTENDED
MON_GET_UNIT_OF_WORK 表函数 - 获取工作单元度量值	REQUEST METRICS BASE
MON_GET_UNIT_OF_WORK 表函数 - 获取详细的工作单元度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE
MON_GET_WORKLOAD 表函数 - 获取工作负载度量值	REQUEST METRICS BASE
MON_GET_WORKLOAD_DETAILS 表函数 - 获取详细的工作负载度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE

表 2148. 快照监视信息

快照级别	逻辑数据分组	监视开关
数据库	dbase	基本
应用程序	appl	基本

可将快照监视的计数器重置。

表 2149. 事件监视信息

事件类型	逻辑数据分组	监视开关
活动	event_activity（在 details_xml 文档中报告）	ACTIVITY METRICS BASE
活动	event_activitymetrics	ACTIVITY METRICS BASE
统计信息	event_scstats（在 details_xml 文档中报告）	REQUEST METRICS BASE
统计信息	event_wlstats（在 details_xml 文档中报告）	REQUEST METRICS BASE
工作单元	在 system_metrics 文档中报告。	REQUEST METRICS BASE
数据库	event_db	始终收集
连接	event_conn	始终收集
事务	event_xact	始终收集
程序包高速缓存	在 activity_metrics 文档中报告。	ACTIVITY METRICS BASE

用法

当应用程序挂起的锁定总数达到可供应用程序使用的最大锁定列表空间量，或者所有应用程序消耗的锁定列表空间达到总锁定列表空间时，锁定将会升级。可用锁定列表空间量由 **maxlocks** 和 **locklist** 配置参数确定。

当应用程序达到允许的最大锁定数并且没有其他要升级的锁定时，它将使用锁定列表中为其他应用程序分配的空间。当整个锁定列表已满时，将发生错误。

此数据项包括所有锁定升级的计数（包括互斥锁定升级和 DB2 pureScale 环境中的升级）。为了确定仅 DB2 pureScale 环境中的锁定升级，请使用 **lock_escals_global** 监视元素。

以下几种原因可能会导致产生过量锁定升级:

- 锁定列表大小（**locklist**）对于并行应用程序数目而言可能太小
- 可供每个应用程序使用的锁定列表百分比（**maxlocks**）可能太小
- 一个或多个应用程序使用的锁定数可能过量。
- 在 DB2 pureScale 环境中，全局锁定列表大小 (**cf_lock_sz**) 可能太小。

要解决这些问题，可以:

- 增加 **locklist** 配置参数值。
- 增加 **maxlocks** 配置参数值。
- 确定执行大量锁定的应用程序，或者使用以下一个公式并将值与 **maxlocks** 进行比较来确定过多占用锁定列表的应用程序:
  - 在 64 位系统上，`(((locks held * 64) / (locklist * 4096)) * 100)`
  - 在 32 位系统上，`(((locks held * 48 / (locklist * 4096)) * 100)`

  这些应用程序还可能因为在锁定列表中使用过量资源而导致其他应用程序中发生锁定升级。这些应用程序可能需要使用表锁定来代替行锁定，尽管表锁定可能会导致 **lock_waits** 和 **lock_wait_time** 监视元素值增大。

lock_escals_global -“全局锁定升级数”监视元素

因为全局锁定内存使用达到 **cf_lock_sz** 数据库配置参数中指定的限制的锁定升级数。

表 2150. 表函数监视信息

表函数	监视元素收集级别
MON_FORMAT_XML_METRICS_BY_ROW - 获取所有度量值的基于已格式化行的输出	不适用；报告 XML 文档中作为格式化函数的输入提供的所有元素
MON_GET_ACTIVITY_DETAILS 表函数 - 获取完整的活动详细信息（在 XML 文档 DETAILS 中报告）	ACTIVITY METRICS BASE
MON_GET_CONNECTION 表函数 - 获取连接度量值	REQUEST METRICS BASE
MON_GET_CONNECTION_DETAILS 表函数 - 获取详细的连接度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE
MON_GET_PKG_CACHE_STMT 表函数 - 获取程序包高速缓存中的 SQL 语句活动度量值	ACTIVITY METRICS BASE
MON_GET_SERVICE_SUBCLASS 表函数 - 获取服务子类度量值	REQUEST METRICS BASE
MON_GET_SERVICE_SUBCLASS_DETAILS 表函数 - 获取详细的服务子类度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE
MON_GET_TABLE 表函数 - 获取表度量值	DATA OBJECT METRICS EXTENDED

表 2150. 表函数监视信息 (续)

表函数	监视元素收集级别
MON_GET_TABLE_USAGE_LIST 表函数 - 从表用法列表返回信息	DATA OBJECT METRICS EXTENDED
MON_GET_UNIT_OF_WORK 表函数 - 获取工作单元度量值	REQUEST METRICS BASE
MON_GET_UNIT_OF_WORK 表函数 - 获取详细的工作单元度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE
MON_GET_WORKLOAD 表函数 - 获取工作负载度量值	REQUEST METRICS BASE
MON_GET_WORKLOAD_DETAILS 表函数 - 获取详细的工作负载度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE

表 2151. 事件监视信息

事件类型	逻辑数据分组	监视开关
活动	event_activity（在 details_xml 文档中报告）	ACTIVITY METRICS BASE
活动	event_activitymetrics	ACTIVITY METRICS BASE
统计信息	event_scstats（在 details_xml 文档中报告）	REQUEST METRICS BASE
统计信息	event_wlstats（在 details_xml 文档中报告）	REQUEST METRICS BASE
工作单元	-	始终收集
程序包高速缓存	-	始终收集
锁定	-	始终收集

用法

将此监视元素与 **lock_escals_maxlocks** 和 **lock_escals_locklist** 监视元素配合使用以确定导致数据库上发生升级的锁定空间配置参数。

lock_escals_locklist -“locklist 锁定升级数”监视元素

因为局部锁定内存使用达到 **locklist** 数据库配置参数中指定的限制的锁定升级数。

表 2152. 表函数监视信息

表函数	监视元素收集级别
MON_FORMAT_XML_METRICS_BY_ROW - 获取所有度量值的基于已格式化行的输出	不适用; 报告 XML 文档中作为格式化函数的输入提供的所有元素
MON_GET_ACTIVITY_DETAILS 表函数 - 获取完整的活动详细信息（在 XML 文档 DETAILS 中报告）	ACTIVITY METRICS BASE
MON_GET_CONNECTION 表函数 - 获取连接度量值	REQUEST METRICS BASE

表 2152. 表函数监视信息 (续)

表函数	监视元素收集级别
MON_GET_CONNECTION_DETAILS 表函数 - 获取详细的连接度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE
MON_GET_PKG_CACHE_STMT 表函数 - 获取程序包高速缓存中的 SQL 语句活动度量值	ACTIVITY METRICS BASE
MON_GET_SERVICE_SUBCLASS 表函数 - 获取服务子类度量值	REQUEST METRICS BASE
MON_GET_SERVICE_SUBCLASS_DETAILS 表函数 - 获取详细的服务子类度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE
MON_GET_UNIT_OF_WORK 表函数 - 获取工作单元度量值	REQUEST METRICS BASE
MON_GET_UNIT_OF_WORK 表函数 - 获取详细的工作单元度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE
MON_GET_WORKLOAD 表函数 - 获取工作负载度量值	REQUEST METRICS BASE
MON_GET_WORKLOAD_DETAILS 表函数 - 获取详细的工作负载度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE

表 2153. 事件监视信息

事件类型	逻辑数据分组	监视开关
活动	event_activity（在 details_xml 文档中报告）	ACTIVITY METRICS BASE
活动	event_activitymetrics	ACTIVITY METRICS BASE
统计信息	event_scstats（在 details_xml 文档中报告）	REQUEST METRICS BASE
统计信息	event_wlstats（在 details_xml 文档中报告）	REQUEST METRICS BASE
工作单元	-	始终收集
程序包高速缓存	-	始终收集
锁定	-	始终收集

用法

将此监视元素与 **lock_escals_maxlocks** 和 **lock_escals_global** 监视元素配合使用以确定导致数据库上发生升级的锁定空间配置参数。

lock_escals_maxlocks -“maxlocks 锁定升级数”监视元素

因为局部锁定内存使用达到 **maxlocks** 数据库配置参数中指定的限制的锁定升级数。

表 2154. 表函数监视信息

表函数	监视元素收集级别
MON_FORMAT_XML_METRICS_BY_ROW - 获取所有度量值的基于已格式化行的输出	不适用; 报告 XML 文档中作为格式化函数的输入提供的所有元素
MON_GET_ACTIVITY_DETAILS 表函数 - 获取完整的活动详细信息（在 XML 文档 DETAILS 中报告）	ACTIVITY METRICS BASE
MON_GET_CONNECTION 表函数 - 获取连接度量值	REQUEST METRICS BASE
MON_GET_CONNECTION_DETAILS 表函数 - 获取详细的连接度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE
MON_GET_PKG_CACHE_STMT 表函数 - 获取程序包高速缓存中的 SQL 语句活动度量值	ACTIVITY METRICS BASE
MON_GET_SERVICE_SUBCLASS 表函数 - 获取服务子类度量值	REQUEST METRICS BASE
MON_GET_SERVICE_SUBCLASS_DETAILS 表函数 - 获取详细的服务子类度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE
MON_GET_UNIT_OF_WORK 表函数 - 获取工作单元度量值	REQUEST METRICS BASE
MON_GET_UNIT_OF_WORK 表函数 - 获取详细的工作单元度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE
MON_GET_WORKLOAD 表函数 - 获取工作负载度量值	REQUEST METRICS BASE
MON_GET_WORKLOAD_DETAILS 表函数 - 获取详细的工作负载度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE

表 2155. 事件监视信息

事件类型	逻辑数据分组	监视开关
活动	event_activity（在 details_xml 文档中报告）	ACTIVITY METRICS BASE
活动	event_activitymetrics	ACTIVITY METRICS BASE
统计信息	event_scstats（在 details_xml 文档中报告）	REQUEST METRICS BASE
统计信息	event_wlstats（在 details_xml 文档中报告）	REQUEST METRICS BASE
工作单元	-	始终收集
程序包高速缓存	-	始终收集
锁定	-	始终收集

用法

将此监视元素与 **lock_escals_locklist** 和 **lock_escals_global** 监视元素配合使用以确定导致数据库上发生升级的锁定空间配置参数。

lock_timeouts_global -“锁定超时全局”监视元素

锁定超时数，持有此锁定的应用程序在远程成员上。

表 2156. 表函数监视信息

表函数	监视元素收集级别
MON_FORMAT_XML_METRICS_BY_ROW - 获取所有度量值的基于已格式化行的输出	不适用；报告 XML 文档中作为格式化函数的输入提供的所有元素
MON_GET_ACTIVITY_DETAILS 表函数 - 获取完整的活动详细信息（在 XML 文档 DETAILS 中报告）	ACTIVITY METRICS BASE
MON_GET_CONNECTION 表函数 - 获取连接度量值	REQUEST METRICS BASE
MON_GET_CONNECTION_DETAILS 表函数 - 获取详细的连接度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE
MON_GET_PKG_CACHE_STMT 表函数 - 获取程序包高速缓存中的 SQL 语句活动度量值	ACTIVITY METRICS BASE
MON_GET_SERVICE_SUBCLASS 表函数 - 获取服务子类度量值	REQUEST METRICS BASE
MON_GET_SERVICE_SUBCLASS_DETAILS 表函数 - 获取详细的服务子类度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE
MON_GET_UNIT_OF_WORK 表函数 - 获取工作单元度量值	REQUEST METRICS BASE
MON_GET_UNIT_OF_WORK 表函数 - 获取详细的工作单元度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE
MON_GET_WORKLOAD 表函数 - 获取工作负载度量值	REQUEST METRICS BASE
MON_GET_WORKLOAD_DETAILS 表函数 - 获取详细的工作负载度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE

表 2157. 事件监视信息

事件类型	逻辑数据分组	监视开关
活动	event_activity（在 details_xml 文档中报告）	ACTIVITY METRICS BASE
活动	event_activitymetrics	ACTIVITY METRICS BASE
统计信息	event_scstats（在 details_xml 文档中报告）	REQUEST METRICS BASE

表 2157. 事件监视信息 (续)

事件类型	逻辑数据分组	监视开关
统计信息	event_wlstats（在 details_xml 文档中报告）	REQUEST METRICS BASE
工作单元	-	始终收集
程序包高速缓存	-	始终收集
锁定	-	始终收集

用法

将此元素与 **lock_timeouts** 监视元素一起使用。**lock_timeouts_global** 监视元素表示等待获取在另一成员上持有的锁定期间出现锁定超时的次数。要确定等待获取在同一成员上持有的锁定期间发生的锁定超时次数，请使用以下公式:

```
lock_timeouts - lock_timeouts_global
```

在 DB2 pureScale 环境外部，此值始终为零。

lock_wait_time_global -“锁定等待时间全局”监视元素

在全局锁定等待上所耗的时间。时间的度量单位为毫秒。

表 2158. 表函数监视信息

表函数	监视元素收集级别
MON_FORMAT_XML_METRICS_BY_ROW - 获取所有度量值的基于已格式化行的输出	不适用; 报告 XML 文档中作为格式化函数的输入提供的所有元素
MON_FORMAT_XML_TIMES_BY_ROW - - 获取基于已格式化行的组合层次结构等待时间和处理时间	不适用; 报告 XML 文档中作为格式化函数的输入提供的所有元素
MON_FORMAT_XML_WAIT_TIMES_BY _ROW - 获取等待时间的已格式化的基于行的输出	不适用; 报告 XML 文档中作为格式化函数的输入提供的所有元素
MON_GET_ACTIVITY_DETAILS 表函数 - 获取完整的活动详细信息（在 XML 文档 DETAILS 中报告）	ACTIVITY METRICS BASE
MON_GET_CONNECTION 表函数 - 获取连接度量值	REQUEST METRICS BASE
MON_GET_CONNECTION_DETAILS 表函数 - 获取详细的连接度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE
MON_GET_PKG_CACHE_STMT 表函数 - 获取程序包高速缓存中的 SQL 语句活动度量值	ACTIVITY METRICS BASE
MON_GET_SERVICE_SUBCLASS 表函数 - 获取服务子类度量值	REQUEST METRICS BASE
MON_GET_SERVICE_SUBCLASS_DETAILS 表函数 - 获取详细的服务子类度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE
MON_GET_TABLE 表函数 - 获取表度量值	DATA OBJECT METRICS EXTENDED

表 2158. 表函数监视信息 (续)

表函数	监视元素收集级别
MON_GET_TABLE_USAGE_LIST 表函数 - 从表用法列表返回信息	DATA OBJECT METRICS EXTENDED
MON_GET_UNIT_OF_WORK 表函数 - 获取工作单元度量值	REQUEST METRICS BASE
MON_GET_UNIT_OF_WORK 表函数 - 获取详细的工作单元度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE
MON_GET_WORKLOAD 表函数 - 获取工作负载度量值	REQUEST METRICS BASE
MON_GET_WORKLOAD_DETAILS 表函数 - 获取详细的工作负载度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE

表 2159. 事件监视信息

事件类型	逻辑数据分组	监视开关
活动	event_activity（在 details_xml 文档中报告）	ACTIVITY METRICS BASE
活动	event_activitymetrics	ACTIVITY METRICS BASE
统计信息	event_scstats（在 details_xml 文档中报告）	REQUEST METRICS BASE
统计信息	event_wlstats（在 details_xml 文档中报告）	REQUEST METRICS BASE
工作单元	-	始终收集
程序包高速缓存	-	始终收集
锁定	-	始终收集

用法

将此监视元素与 **lock_wait_time** 监视元素一起使用，后者表示等待锁定所耗的所有时间。**lock_wait_time_global** 监视元素表示等待由不同成员上的冲突应用程序持有的锁定所耗的时间。要确定等待由同一成员上的冲突应用程序持有的锁定所耗的总时间，请使用以下公式:

```
lock_wait_time - lock_wait_time_global
```

在 DB2 pureScale 环境外部，此值始终为零。

lock_wait_time_global_top -“最长全局锁定等待时间”监视元素

在另一成员上挂起的锁定的最长锁定等待。此值以毫秒计。

表 2160. 事件监视信息

事件类型	逻辑数据分组	监视开关
统计信息	event_wlstats	始终收集

lock_waits_global -“锁定等待全局”监视元素

因为持有针对远程成员的锁定的应用程序而导致的锁定等待数。

表 2161. 表函数监视信息

表函数	监视元素收集级别
MON_FORMAT_XML_METRICS_BY_ROW - 获取所有度量值的基于已格式化行的输出	不适用; 报告 XML 文档中作为格式化函数的输入提供的所有元素
MON_GET_ACTIVITY_DETAILS 表函数 - 获取完整的活动详细信息（在 XML 文档 DETAILS 中报告）	ACTIVITY METRICS BASE
MON_GET_CONNECTION 表函数 - 获取连接度量值	REQUEST METRICS BASE
MON_GET_CONNECTION_DETAILS 表函数 - 获取详细的连接度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE
MON_GET_PKG_CACHE_STMT 表函数 - 获取程序包高速缓存中的 SQL 语句活动度量值	ACTIVITY METRICS BASE
MON_GET_SERVICE_SUBCLASS 表函数 - 获取服务子类度量值	REQUEST METRICS BASE
MON_GET_SERVICE_SUBCLASS_DETAILS 表函数 - 获取详细的服务子类度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE
MON_GET_TABLE 表函数 - 获取表度量值	DATA OBJECT METRICS EXTENDED
MON_GET_TABLE_USAGE_LIST 表函数 - 从表用法列表返回信息	DATA OBJECT METRICS EXTENDED
MON_GET_UNIT_OF_WORK 表函数 - 获取工作单元度量值	REQUEST METRICS BASE
MON_GET_UNIT_OF_WORK 表函数 - 获取详细的工作单元度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE
MON_GET_WORKLOAD 表函数 - 获取工作负载度量值	REQUEST METRICS BASE
MON_GET_WORKLOAD_DETAILS 表函数 - 获取详细的工作负载度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE

表 2162. 事件监视信息

事件类型	逻辑数据分组	监视开关
活动	event_activity（在 details_xml 文档中报告）	ACTIVITY METRICS BASE
活动	event_activitymetrics	ACTIVITY METRICS BASE
统计信息	event_scstats（在 details_xml 文档中报告）	REQUEST METRICS BASE
统计信息	event_wlstats（在 details_xml 文档中报告）	REQUEST METRICS BASE
工作单元	-	始终收集

表 2162. 事件监视信息 (续)

事件类型	逻辑数据分组	监视开关
程序包高速缓存	-	始终收集
锁定	-	始终收集

用法

将此监视元素与 **lock_waits** 监视元素配合使用，后者报告因为由所有成员上的冲突应用程序持有的锁定而导致的锁定等待总数。**lock_waits_global** 监视元素指示由不同成员上的冲突应用程序持有的锁定等待的次数。要确定由等待应用程序所在成员上的冲突应用程序持有的锁定等待数，请使用以下公式:

```
lock_waits - lock_waits_global
```

在 DB2 pureScale 环境外部，此值始终为零。

member -“数据库成员”监视元素

数据库成员的数字标识，此结果记录的数据接收自该成员。

表 2163. 表函数监视信息

表函数	监视元素收集级别
ADMIN_GET_MEM_USAGE 表函数 - 获取实例的总内存消耗	ACTIVITY METRICS BASE
AUDIT_ARCHIVE 过程和表函数 - 归档审计日志文件	ACTIVITY METRICS BASE
DBCFG 管理视图和 DB_GET_CFG 表函数 - 检索数据库配置参数信息	ACTIVITY METRICS BASE
ENV_GET_REG_VARIABLES 表函数 - 检索正使用的 DB2 注册表设置	ACTIVITY METRICS BASE
ENV_GET_DB2_SYSTEM_RESOURCES 表函数 - 返回 DB2(r) 系统信息	ACTIVITY METRICS BASE
ENV_GET_SYSTEM_RESOURCES 表函数 - 返回系统信息	ACTIVITY METRICS BASE
ENV_GET_NETWORK_RESOURCES 表函数 - 返回网络适配器信息	ACTIVITY METRICS BASE
MON_GET_ACTIVITY_DETAILS 表函数 - 获取完整的活动详细信息	ACTIVITY METRICS BASE
MON_GET_AUTO_MAINT_QUEUE 表函数 - 获取有关自动维护作业的信息	ACTIVITY METRICS BASE
MON_GET_AUTO_RUNSTATS_QUEUE 表函数 - 检索有关为求值而排队的对象的信息	ACTIVITY METRICS BASE
MON_GET_BUFFERPOOL 表函数 - 获取缓冲池度量值	ACTIVITY METRICS BASE
MON_GET_CF_WAIT_TIME 表函数 - 获取集群高速缓存工具命令等待时间	ACTIVITY METRICS BASE
MON_GET_CONNECTION 表函数 - 获取连接度量值	ACTIVITY METRICS BASE

表 2163. 表函数监视信息 (续)

表函数	监视元素收集级别
MON_GET_CONNECTION_DETAILS 表函数 - 获取详细的连接度量值	ACTIVITY METRICS BASE
MON_GET_CONTAINER 表函数 - 获取表空间容器度量值	ACTIVITY METRICS BASE
MON_GET_EXTENDED_LATCH_WAIT 表函数 - 返回锁存器的信息	ACTIVITY METRICS BASE
MON_GET_EXTENT_MOVEMENT_STATUS - 获取扩展数据块移动进度状态度量值	ACTIVITY METRICS BASE
MON_GET_FCM - 获取 FCM 度量值	ACTIVITY METRICS BASE
MON_GET_FCM_CONNECTION_LIST - 获取有关所有 FCM 连接的详细信息	ACTIVITY METRICS BASE
MON_GET_GROUP_BUFFERPOOL 表函数 - 获取组缓冲池度量值	ACTIVITY METRICS BASE
MON_GET_INDEX 表函数 - 获取索引度量值	ACTIVITY METRICS BASE
MON_GET_INDEX_USAGE_LIST 表函数 - 返回索引用法列表中的信息	ACTIVITY METRICS BASE
MON_GET_LOCKS 表函数 - 列示当前所连接的数据库中的所有锁定	ACTIVITY METRICS BASE
MON_GET_MEMORY_POOL 表函数 - 获取内存池信息	ACTIVITY METRICS BASE
MON_GET_MEMORY_SET 表函数 - 获取内存集合信息	ACTIVITY METRICS BASE
MON_GET_PAGE_ACCESS_INFO 表函数 - 获取缓冲池页等待信息	ACTIVITY METRICS BASE
MON_GET_PKG_CACHE_STMT 表函数 - 获取程序包高速缓存中的 SQL 语句活动度量值	ACTIVITY METRICS BASE
MON_GET_PKG_CACHE_STMT_DETAILS 表函数 - 获取程序包高速缓存条目的详细度量值	ACTIVITY METRICS BASE
MON_GET_REBALANCE_STATUS 表函数 - 获取表空间的重新平衡进度	ACTIVITY METRICS BASE
MON_GET_RTS_RQST 表函数 - 检索有关实时统计请求的信息	ACTIVITY METRICS BASE
MON_GET_SERVERLIST 表函数 - 获取成员优先级详细信息	ACTIVITY METRICS BASE
MON_GET_SERVICE_SUBCLASS 表函数 - 获取服务子类度量值	ACTIVITY METRICS BASE
MON_GET_SERVICE_SUBCLASS_DETAILS 表函数 - 获取详细的服务子类度量值	ACTIVITY METRICS BASE
MON_GET_TABLE 表函数 - 获取表度量值	ACTIVITY METRICS BASE
MON_GET_TABLE_USAGE_LIST 表函数 - 从表用法列表返回信息	ACTIVITY METRICS BASE
MON_GET_TABLESPACE 表函数 - 获取表空间度量值	ACTIVITY METRICS BASE

表 2163. 表函数监视信息 (续)

表函数	监视元素收集级别
MON_GET_TRANSACTION_LOG 表函数 - 获取日志信息	ACTIVITY METRICS BASE
MON_GET_UNIT_OF_WORK 表函数 - 获取工作单元度量值	ACTIVITY METRICS BASE
MON_GET_UNIT_OF_WORK_DETAILS 表函数 - 获取详细的工作单元度量值	ACTIVITY METRICS BASE
MON_GET_USAGE_LIST_STATUS 表函数 - 返回用法列表的状态	ACTIVITY METRICS BASE
MON_GET_WORKLOAD 表函数 - 获取工作负载度量值	ACTIVITY METRICS BASE
MON_GET_WORKLOAD_DETAILS 表函数 - 获取详细的工作负载度量值	ACTIVITY METRICS BASE
MON_SAMPLE_SERVICE_CLASS_METRICS - 获取样本服务类度量值	ACTIVITY METRICS BASE
MON_SAMPLE_WORKLOAD_METRICS - 获取样本	ACTIVITY METRICS BASE
PD_GET_DIAG_HIST 表函数 - 从给定设施返回记录	ACTIVITY METRICS BASE
PDLOGMSGS_LAST24HOURS 管理视图和 PD_GET_LOG_MSGS 表函数 - 检索问题确定消息	ACTIVITY METRICS BASE
WLM_GET_QUEUE_STATS 表函数 - 返回阈值队列统计信息	ACTIVITY METRICS BASE
WLM_GET_SERVICE_CLASS_AGENTS 表函数 - 列示正在服务类中运行的代理程序	ACTIVITY METRICS BASE
WLM_GET_SERVICE_CLASS_WORKLOAD_OCCURRENCES 表函数 - 列示工作负载出现次数	ACTIVITY METRICS BASE
WLM_GET_SERVICE_SUBCLASS_STATS 表函数 - 返回服务子类的统计信息	ACTIVITY METRICS BASE
WLM_GET_SERVICE_SUPERCLASS_STATS 表函数 - 返回服务超类的统计信息	ACTIVITY METRICS BASE
WLM_GET_WORK_ACTION_SET_STATS 表函数 - 返回工作操作集统计信息	ACTIVITY METRICS BASE
WLM_GET_WORKLOAD_OCCURRENCE_ACTIVITIES 表函数 - 返回活动列表	ACTIVITY METRICS BASE
WLM_GET_WORKLOAD_STATS 表函数 - 返回工作负载统计信息	ACTIVITY METRICS BASE

表 2164. 事件监视信息

事件类型	逻辑数据分组	监视开关
活动	event_activity（在 details_xml 文档中报告）	ACTIVITY METRICS BASE

表 2164. 事件监视信息 (续)

事件类型	逻辑数据分组	监视开关
统计信息	event_scstats（在 details_xml 文档中报告）	REQUEST METRICS BASE
统计信息	event_wlstats（在 details_xml 文档中报告）	REQUEST METRICS BASE
工作单元	在 system_metrics 文档中报告。	REQUEST METRICS BASE
程序包高速缓存	-	COLLECT BASE DATA
锁定	-	始终收集
变更历史记录	changesummary dbdbmcfg regvar ddlstmtexec txncompletion evmonstart utilstart utillocation utilstop	始终收集

用法

DB2 成员是在单一主机上运行 DB2 服务器软件的数据库管理器实例。DB2 成员接受并处理来自与其相连接的应用程序的数据库请求。

objtype -“对象类型”监视元素

正报告其监视数据的对象的类型。此监视元素是 object_type 监视元素的别名。

表 2165. 表函数监视信息

表函数	监视元素收集级别
ADMIN_GET_TAB_COMPRESS_INFO 表函数 - 估算压缩节省量	ACTIVITY METRICS BASE
ADMIN_GET_TAB_DICTIONARY_INFO 表函数 - 报告现有表字典的属性	ACTIVITY METRICS BASE
MON_GET_AUTO_MAINT_QUEUE 表函数 - 获取有关自动维护作业的信息	ACTIVITY METRICS BASE
MON_GET_AUTO_RUNSTATS_QUEUE 表函数 - 检索有关为求值而排队的对象的信息	ACTIVITY METRICS BASE
MON_GET_PAGE_ACCESS_INFO 表函数 - 获取缓冲池页等待信息	ACTIVITY METRICS BASE
MON_GET_RTS_RQST 表函数 - 检索有关实时统计请求的信息	ACTIVITY METRICS BASE
MON_GET_USAGE_LIST_STATUS 表函数 - 返回用法列表的状态	ACTIVITY METRICS BASE
PD_GET_DIAG_HIST 表函数 - 从给定设施返回记录	ACTIVITY METRICS BASE

表 2166. 事件监视信息

事件类型	逻辑数据分组	监视开关
变更历史记录	utilphase utilstart	始终收集

使用说明

- ADMIN_GET_TAB_COMPRESS_INFO 表函数或 ADMIN_GET_TAB_DICTIONARY_INFO 表函数返回此元素时，object_type 监视元素的返回值可以是“XML”或“DATA”。
- MON_GET_RTS_RQST 表函数返回此元素时，object_type 监视元素的返回值为“TABLE”。
- MON_GET_AUTO_MAINT_QUEUE 表函数返回此元素时，object_type 监视元素的返回值可以是“DATABASE”、“TABLE”、“NICKNAME”或“VIEW”。
- MON_GET_AUTO_RUNSTATS_QUEUE 表函数返回此元素时，object_type 监视元素的返回值可以是“TABLE”、“NICKNAME”或“VIEW”。
- 变更历史记录事件监视器返回此元素时，object_type 监视元素的返回值可以是“DATABASE”、“INDEX”、“PARTITIONGROUP”、“TABLE”或“TABLESPACE”。

page_reclaims_initiated_s -“共享存取导致的页回收数”监视元素

以共享方式访问页从而导致从另一成员回收此页的次数。与内部保留的对象空间映射相关的页回收数是单独计算的。

表 2167. 表函数监视信息

表函数	监视元素收集级别
MON_GET_PAGE_ACCESS_INFO 表函数 - 获取缓冲池页等待信息	ACTIVITY METRICS BASE

page_reclaims_initiated_x -“互斥存取导致的页回收数”监视元素

以互斥方式访问页从而导致从另一成员回收此页的次数。与内部保留的对象空间映射相关的页回收数是单独计算的。

表 2168. 表函数监视信息

表函数	监视元素收集级别
MON_GET_PAGE_ACCESS_INFO 表函数 - 获取缓冲池页等待信息	ACTIVITY METRICS BASE

page_reclaims_s -“共享访问页回收数”监视元素

DB2 pureScale实例中的另一成员回收的与此对象相关的页的次数（在计划释放此页之前），其中回收此页的成员需要共享存取。

表 2169. 表函数监视信息

表函数	监视元素收集级别
MON_GET_PAGE_ACCESS_INFO 表函数 - 获取缓冲池页等待信息	ACTIVITY METRICS BASE

用法

与内部保留的对象空间映射相关的页回收数是单独计算的。

page_reclaims_x -“互斥存取页回收数”监视元素

DB2 pureScale实例中的另一成员回收的与此对象相关的页的次数（在计划释放此页之前），其中回收此页的成员需要互斥存取。

表 2170. 表函数监视信息

表函数	监视元素收集级别
MON_GET_PAGE_ACCESS_INFO 表函数 - 获取缓冲池页等待信息	ACTIVITY METRICS BASE

用法

与内部保留的对象空间映射相关的页回收数是单独计算的。

pool_async_data_gbp_invalid_pages -“异步组缓冲池无效数据页数”监视元素

预取程序尝试从组缓冲池读取数据页（因为该页在本地缓冲池中无效）的次数。

表 2171. 表函数监视信息

表函数	监视元素收集级别
MON_GET_BUFFERPOOL 表函数 - 获取缓冲池度量值	DATA OBJECT METRICS BASE
MON_GET_TABLESPACE 表函数 - 获取表空间度量值	DATA OBJECT METRICS BASE

用法

可按如下所示计算预取程序（或异步）数据页命中率:

```
GBP = ( pool_async_data_gbp_l_reads - pool_async_data_gbp_p_reads )
 / pool_async_data_gbp_l_reads
```

缓冲池命中速率是 IBM DB2 pureScale Feature 的整体性能中的重要因素。使用此公式可帮助您确定组缓冲池会否成为数据库吞吐量的限制因素。

pool_async_data_gbp_l_reads -“异步组缓冲池数据逻辑读取数”监视元素

预取程序尝试从组缓冲池读取依赖于组缓冲池 (GBP) 的数据页（因为该页在本地缓冲池中无效或不存在）的次数。

表 2172. 表函数监视信息

表函数	监视元素收集级别
MON_GET_BUFFERPOOL 表函数 - 获取缓冲池度量值	DATA OBJECT METRICS BASE
MON_GET_TABLESPACE 表函数 - 获取表空间度量值	DATA OBJECT METRICS BASE

用法

可按如下所示计算预取程序（或异步）数据页命中率:

```
GBP = ( pool_async_data_gbp_l_reads - pool_async_data_gbp_p_reads )
 / pool_async_data_gbp_l_reads
```

缓冲池命中速率是 IBM DB2 pureScale Feature 的整体性能中的重要因素。使用此公式可帮助您确定组缓冲池会否成为数据库吞吐量的限制因素。

pool_async_data_gbp_p_reads -“异步组缓冲池数据物理读取数”监视元素

预取程序尝试将磁盘中依赖于组缓冲池 (GBP) 的数据页读取到本地缓冲池中（因为在 GBP 中找不到该页）的次数。

表 2173. 表函数监视信息

表函数	监视元素收集级别
MON_GET_BUFFERPOOL 表函数 - 获取缓冲池度量值	DATA OBJECT METRICS BASE
MON_GET_TABLESPACE 表函数 - 获取表空间度量值	DATA OBJECT METRICS BASE

用法

可按如下所示计算预取程序（或异步）数据页命中率:

```
GBP = ( pool_async_data_gbp_l_reads - pool_async_data_gbp_p_reads )
 / pool_async_data_gbp_l_reads
```

缓冲池命中速率是 IBM DB2 pureScale Feature 的整体性能中的重要因素。使用此公式可帮助您确定组缓冲池会否成为数据库吞吐量的限制因素。

pool_async_data_lbp_pages_found -“发现的异步本地缓冲池数据页数”监视元素

预取程序尝试访问数据页时此数据页出现在本地缓冲池中的次数。

表 2174. 表函数监视信息

表函数	监视元素收集级别
MON_GET_BUFFERPOOL 表函数 - 获取缓冲池度量值	DATA OBJECT METRICS BASE
MON_GET_TABLESPACE 表函数 - 获取表空间度量值	DATA OBJECT METRICS BASE

用法

可按如下所示计算预取程序（或异步）数据页命中率:

```
GBP = ( pool_async_data_gbp_l_reads - pool_async_data_gbp_p_reads )
 / pool_async_data_gbp_l_reads
```

缓冲池命中速率是 IBM DB2 pureScale Feature 的整体性能中的重要因素。使用此公式可帮助您确定组缓冲池会否成为数据库吞吐量的限制因素。

pool_async_index_gbp_invalid_pages -“异步组缓冲池无效索引页数”监视元素

预取程序尝试从组缓冲池读取索引页（因为该页在本地缓冲池中无效）的次数。

表 2175. 表函数监视信息

表函数	监视元素收集级别
MON_GET_BUFFERPOOL 表函数 - 获取缓冲池度量值	DATA OBJECT METRICS BASE
MON_GET_TABLESPACE 表函数 - 获取表空间度量值	DATA OBJECT METRICS BASE

用法

可按如下所示计算预取程序（或异步）索引页命中率:

```
GBP = ( pool_async_index_gbp_l_reads - pool_async_index_gbp_p_reads )
 / pool_async_index_gbp_l_reads
```

缓冲池命中速率是 DB2 pureScale实例的整体性能中的重要因素。使用以上公式可帮助您确定组缓冲池会否成为数据库吞吐量的限制因素。

pool_async_index_gbp_l_reads -“异步组缓冲池索引逻辑读取数”监视元素

预取程序尝试从组缓冲池读取依赖于组缓冲池 (GBP) 的索引页（因为该页在本地缓冲池中无效或不存在）的次数。

表 2176. 表函数监视信息

表函数	监视元素收集级别
MON_GET_BUFFERPOOL 表函数 - 获取缓冲池度量值	DATA OBJECT METRICS BASE
MON_GET_TABLESPACE 表函数 - 获取表空间度量值	DATA OBJECT METRICS BASE

用法

可按如下所示计算预取程序（或异步）索引页命中率:

```
GBP = ( pool_async_index_gbp_l_reads - pool_async_index_gbp_p_reads )
 / pool_async_index_gbp_l_reads
```

缓冲池命中速率是 DB2 pureScale实例的整体性能中的重要因素。使用以上公式可帮助您确定组缓冲池会否成为数据库吞吐量的限制因素。

pool_async_index_gbp_p_reads -“异步组缓冲池索引物理读取数”监视元素

预取程序尝试将磁盘中依赖于组缓冲池 (GBP) 的索引页读取到本地缓冲池中（因为在 GBP 中找不到该页）的次数。

表 2177. 表函数监视信息

表函数	监视元素收集级别
MON_GET_BUFFERPOOL 表函数 - 获取缓冲池度量值	DATA OBJECT METRICS BASE
MON_GET_TABLESPACE 表函数 - 获取表空间度量值	DATA OBJECT METRICS BASE

用法

可按如下所示计算预取程序（或异步）索引页命中率:

```
GBP = ( pool_async_index_gbp_l_reads - pool_async_index_gbp_p_reads )
 / pool_async_index_gbp_l_reads
```

缓冲池命中速率是 DB2 pureScale实例的整体性能中的重要因素。使用以上公式可帮助您确定组缓冲池会否成为数据库吞吐量的限制因素。

pool_async_index_lbp_pages_found -“发现的异步本地缓冲池索引页数”监视元素

预取程序尝试访问索引页时此索引页出现在本地缓冲池中的次数。

表 2178. 表函数监视信息

表函数	监视元素收集级别
MON_GET_BUFFERPOOL 表函数 - 获取缓冲池度量值	DATA OBJECT METRICS BASE
MON_GET_TABLESPACE 表函数 - 获取表空间度量值	DATA OBJECT METRICS BASE

用法

可按如下所示计算预取程序（或异步）索引页命中率:

```
GBP = ( pool_async_index_gbp_l_reads - pool_async_index_gbp_p_reads )
 / pool_async_index_gbp_l_reads
```

缓冲池命中速率是 DB2 pureScale实例的整体性能中的重要因素。使用以上公式可帮助您确定组缓冲池会否成为数据库吞吐量的限制因素。

pool_async_xda_gbp_invalid_pages -“异步组缓冲池无效 XDA 数据页数”监视元素

预取程序向组缓冲池发出针对 XML 存储器对象 (XDA) 的数据页的请求（因为该页在本地缓冲池中标记为无效）的次数。

表 2179. 表函数监视信息

表函数	监视元素收集级别
MON_GET_BUFFERPOOL 表函数 - 获取缓冲池度量值	DATA OBJECT METRICS BASE
MON_GET_TABLESPACE 表函数 - 获取表空间度量值	DATA OBJECT METRICS BASE

用法

可按如下所示计算预取程序（或异步）XDA 页命中率:

```
GBP = ( pool_async_xda_gbp_l_reads - pool_async_xda_gbp_p_reads )
 / pool_async_xda_gbp_l_reads
```

缓冲池命中速率是 DB2 pureScale实例的整体性能中的重要因素。使用以上公式可帮助您确定组缓冲池会否成为数据库吞吐量的限制因素。

pool_async_xda_gbp_l_reads -“组缓冲池 XDA 数据异步逻辑读取请求数”监视元素

预取程序尝试从组缓冲池读取 XML 存储器对象 (XDA) 的依赖于 GBP 的数据页（因为该页在本地缓冲池中无效或不存在）的次数。

表 2180. 表函数监视信息

表函数	监视元素收集级别
MON_GET_BUFFERPOOL 表函数 - 获取缓冲池度量值	DATA OBJECT METRICS BASE
MON_GET_TABLESPACE 表函数 - 获取表空间度量值	DATA OBJECT METRICS BASE

用法

可按如下所示计算预取程序（或异步）XDA 页命中率:

```
GBP = ( pool_async_xda_gbp_l_reads - pool_async_xda_gbp_p_reads )
 / pool_async_xda_gbp_l_reads
```

缓冲池命中速率是 DB2 pureScale实例的整体性能中的重要因素。使用以上公式可帮助您确定组缓冲池会否成为数据库吞吐量的限制因素。

pool_async_xda_gbp_p_reads -“组缓冲池 XDA 数据异步物理读取请求数”监视元素

预取程序将磁盘中 XML 存储器对象 (XDA) 的依赖于 GBP 的数据页读取到本地缓冲池中（因为在 GBP 中找不到该页）的次数。

表 2181. 表函数监视信息

表函数	监视元素收集级别
MON_GET_BUFFERPOOL 表函数 - 获取缓冲池度量值	DATA OBJECT METRICS BASE
MON_GET_TABLESPACE 表函数 - 获取表空间度量值	DATA OBJECT METRICS BASE

用法

可按如下所示计算预取程序（或异步）XDA 页命中率:

```
GBP = ( pool_async_xda_gbp_l_reads - pool_async_xda_gbp_p_reads )
 / pool_async_xda_gbp_l_reads
```

缓冲池命中速率是 DB2 pureScale实例的整体性能中的重要因素。使用以上公式可帮助您确定组缓冲池会否成为数据库吞吐量的限制因素。

pool_async_xda_lbp_pages_found -“发现的异步本地缓冲池 XDA 数据页数”监视元素

预取程序向本地缓冲池请求并在其中发现 XML 存储器对象 (XDA) 的数据页的次数。

表 2182. 表函数监视信息

表函数	监视元素收集级别
MON_GET_BUFFERPOOL 表函数 - 获取缓冲池度量值	DATA OBJECT METRICS BASE
MON_GET_TABLESPACE 表函数 - 获取表空间度量值	DATA OBJECT METRICS BASE

用法

可按如下所示计算预取程序（或异步）XDA 页命中率:

```
GBP = ( pool_async_xda_gbp_l_reads - pool_async_xda_gbp_p_reads )
 / pool_async_xda_gbp_l_reads
```

缓冲池命中速率是 DB2 pureScale实例的整体性能中的重要因素。使用以上公式可帮助您确定组缓冲池会否成为数据库吞吐量的限制因素。

pool_data_gbp_invalid_pages -“组缓冲池无效数据页数”监视元素

数据页在本地缓冲池中无效并因此改为从组缓冲池中读取的次数。在 DB2 pureScale 环境外部，此值为空。

表 2183. 表函数监视信息

表函数	监视元素收集级别
MON_FORMAT_XML_METRICS_BY_ROW - 获取所有度量值的基于已格式化行的输出	不适用; 报告 XML 文档中作为格式化函数的输入提供的所有元素
MON_GET_ACTIVITY_DETAILS 表函数 - 获取完整的活动详细信息（在 XML 文档 DETAILS 中报告）	ACTIVITY METRICS BASE
MON_GET_BUFFERPOOL 表函数 - 获取缓冲池度量值	DATA OBJECT METRICS BASE
MON_GET_CONNECTION 表函数 - 获取连接度量值	REQUEST METRICS BASE
MON_GET_CONNECTION_DETAILS 表函数 - 获取详细的连接度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE
MON_GET_PKG_CACHE_STMT 表函数 - 获取程序包高速缓存中的 SQL 语句活动度量值	ACTIVITY METRICS BASE
MON_GET_SERVICE_SUBCLASS 表函数 - 获取服务子类度量值	REQUEST METRICS BASE
MON_GET_SERVICE_SUBCLASS_DETAILS 表函数 - 获取详细的服务子类度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE

表 2183. 表函数监视信息 (续)

表函数	监视元素收集级别
MON_GET_TABLESPACE 表函数 - 获取表空间度量值	DATA OBJECT METRICS BASE
MON_GET_UNIT_OF_WORK 表函数 - 获取工作单元度量值	REQUEST METRICS BASE
MON_GET_UNIT_OF_WORK 表函数 - 获取详细的工作单元度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE
MON_GET_WORKLOAD 表函数 - 获取工作负载度量值	REQUEST METRICS BASE
MON_GET_WORKLOAD_DETAILS 表函数 - 获取详细的工作负载度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE

表 2184. 事件监视信息

事件类型	逻辑数据分组	监视开关
活动	event_activity（在 details_xml 文档中报告）	ACTIVITY METRICS BASE
活动	event_activitymetrics	ACTIVITY METRICS BASE
统计信息	event_scstats（在 details_xml 文档中报告）	REQUEST METRICS BASE
统计信息	event_wlstats（在 details_xml 文档中报告）	REQUEST METRICS BASE
工作单元	-	始终收集
程序包高速缓存	-	始终收集
锁定	-	始终收集

用法

要确定在本地缓冲池中发现所请求页的频率，请使用以下公式:

```
(POOL_DATA_LBP_PAGES_FOUND - POOL_ASYNC_DATA_LBP_PAGES_FOUND) / POOL_DATA_L_READS
```

要确定在组缓冲池中发现所请求页的频率，请使用以下公式

```
(POOL_DATA_GBP_L_READS - POOL_DATA_GBP_P_READS) / POOL_DATA_GBP_L_READS
```

本地缓冲池和组缓冲池命中率都是集群高速缓存设施的整体性能中的重要因素。使用这些公式可帮助您确定本地缓冲池或组缓冲池会否成为数据库吞吐量的限制因素。

pool_data_gbp_l_reads -“组缓冲池数据逻辑读取数”监视元素

尝试从组缓冲池读取依赖于组缓冲池 (GBP) 的数据页（因为该页在本地缓冲池 (LBP) 中无效或不存在）的次数。

表 2185. 表函数监视信息

表函数	监视元素收集级别
MON_FORMAT_XML_METRICS_BY_ROW - 获取所有度量值的基于已格式化行的输出	不适用; 报告 XML 文档中作为格式化函数的输入提供的所有元素
MON_GET_ACTIVITY_DETAILS 表函数 - 获取完整的活动详细信息（在 XML 文档 DETAILS 中报告）	ACTIVITY METRICS BASE
MON_GET_BUFFERPOOL 表函数 - 获取缓冲池度量值	DATA OBJECT METRICS BASE
MON_GET_CONNECTION 表函数 - 获取连接度量值	REQUEST METRICS BASE
MON_GET_CONNECTION_DETAILS 表函数 - 获取详细的连接度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE
MON_GET_PKG_CACHE_STMT 表函数 - 获取程序包高速缓存中的 SQL 语句活动度量值	ACTIVITY METRICS BASE
MON_GET_SERVICE_SUBCLASS 表函数 - 获取服务子类度量值	REQUEST METRICS BASE
MON_GET_SERVICE_SUBCLASS_DETAILS 表函数 - 获取详细的服务子类度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE
MON_GET_TABLESPACE 表函数 - 获取表空间度量值	DATA OBJECT METRICS BASE
MON_GET_UNIT_OF_WORK 表函数 - 获取工作单元度量值	REQUEST METRICS BASE
MON_GET_UNIT_OF_WORK 表函数 - 获取详细的工作单元度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE
MON_GET_WORKLOAD 表函数 - 获取工作负载度量值	REQUEST METRICS BASE
MON_GET_WORKLOAD_DETAILS 表函数 - 获取详细的工作负载度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE

表 2186. 事件监视信息

事件类型	逻辑数据分组	监视开关
活动	event_activity（在 details_xml 文档中报告）	ACTIVITY METRICS BASE
活动	event_activitymetrics	ACTIVITY METRICS BASE
统计信息	event_scstats（在 details_xml 文档中报告）	REQUEST METRICS BASE

表 2186. 事件监视信息 (续)

事件类型	逻辑数据分组	监视开关
统计信息	event_wlstats（在 details_xml 文档中报告）	REQUEST METRICS BASE
工作单元	-	始终收集
程序包高速缓存	-	始终收集
锁定	-	始终收集

用法

要确定在本地缓冲池中发现所请求页的频率，请使用以下公式:

```
(POOL_DATA_LBP_PAGES_FOUND - POOL_ASYNC_DATA_LBP_PAGES_FOUND) / POOL_DATA_L_READS
```

要确定在组缓冲池中发现所请求页的频率，请使用以下公式

```
(POOL_DATA_GBP_L_READS - POOL_DATA_GBP_P_READS) / POOL_DATA_GBP_L_READS
```

本地缓冲池和组缓冲池命中率都是集群高速缓存设施的整体性能中的重要因素。使用这些公式可帮助您确定本地缓冲池或组缓冲池会否成为数据库吞吐量的限制因素。

pool_data_gbp_p_reads -“组缓冲池数据物理读取数”监视元素

尝试将磁盘中依赖于组缓冲池 (GBP) 的数据页读取到本地缓冲池中（因为在 GBP 中找不到该页）的次数。

表 2187. 表函数监视信息

表函数	监视元素收集级别
MON_FORMAT_XML_METRICS_BY_ROW - 获取所有度量值的基于已格式化行的输出	不适用; 报告 XML 文档中作为格式化函数的输入提供的所有元素
MON_GET_ACTIVITY_DETAILS 表函数 - 获取完整的活动详细信息（在 XML 文档 DETAILS 中报告）	ACTIVITY METRICS BASE
MON_GET_BUFFERPOOL 表函数 - 获取缓冲池度量值	DATA OBJECT METRICS BASE
MON_GET_CONNECTION 表函数 - 获取连接度量值	REQUEST METRICS BASE
MON_GET_CONNECTION_DETAILS 表函数 - 获取详细的连接度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE
MON_GET_PKG_CACHE_STMT 表函数 - 获取程序包高速缓存中的 SQL 语句活动度量值	ACTIVITY METRICS BASE
MON_GET_SERVICE_SUBCLASS 表函数 - 获取服务子类度量值	REQUEST METRICS BASE
MON_GET_SERVICE_SUBCLASS_DETAILS 表函数 - 获取详细的服务子类度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE
MON_GET_TABLESPACE 表函数 - 获取表空间度量值	DATA OBJECT METRICS BASE

表 2187. 表函数监视信息 (续)

表函数	监视元素收集级别
MON_GET_UNIT_OF_WORK 表函数 - 获取工作单元度量值	REQUEST METRICS BASE
MON_GET_UNIT_OF_WORK 表函数 - 获取详细的工作单元度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE
MON_GET_WORKLOAD 表函数 - 获取工作负载度量值	REQUEST METRICS BASE
MON_GET_WORKLOAD_DETAILS 表函数 - 获取详细的工作负载度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE

表 2188. 事件监视信息

事件类型	逻辑数据分组	监视开关
活动	event_activity（在 details_xml 文档中报告）	ACTIVITY METRICS BASE
活动	event_activitymetrics	ACTIVITY METRICS BASE
统计信息	event_scstats（在 details_xml 文档中报告）	REQUEST METRICS BASE
统计信息	event_wlstats（在 details_xml 文档中报告）	REQUEST METRICS BASE
工作单元	-	始终收集
程序包高速缓存	-	始终收集
锁定	-	始终收集

用法

要确定在本地缓冲池中发现所请求页的频率，请使用以下公式:

```
(POOL_DATA_LBP_PAGES_FOUND - POOL_ASYNC_DATA_LBP_PAGES_FOUND) / POOL_DATA_L_READS
```

要确定在组缓冲池中发现所请求页的频率，请使用以下公式

```
(POOL_DATA_GBP_L_READS - POOL_DATA_GBP_P_READS) / POOL_DATA_GBP_L_READS
```

本地缓冲池和组缓冲池命中率都是集群高速缓存设施的整体性能中的重要因素。使用这些公式可帮助您确定本地缓冲池或组缓冲池会否成为数据库吞吐量的限制因素。

pool_data_lbp_pages_found -“本地缓冲池发现的数据页数”监视元素

数据页出现在本地缓冲池中的次数。

表 2189. 表函数监视信息

表函数	监视元素收集级别
MON_FORMAT_XML_METRICS_BY_ROW - 获取所有度量值的基于已格式化行的输出	不适用; 报告 XML 文档中作为格式化函数的输入提供的所有元素

表 2189. 表函数监视信息 (续)

表函数	监视元素收集级别
MON_GET_ACTIVITY_DETAILS 表函数 - 获取完整的活动详细信息（在 XML 文档 DETAILS 中报告）	ACTIVITY METRICS BASE
MON_GET_BUFFERPOOL 表函数 - 获取缓冲池度量值	DATA OBJECT METRICS BASE
MON_GET_CONNECTION 表函数 - 获取连接度量值	REQUEST METRICS BASE
MON_GET_CONNECTION_DETAILS 表函数 - 获取详细的连接度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE
MON_GET_PKG_CACHE_STMT 表函数 - 获取程序包高速缓存中的 SQL 语句活动度量值	ACTIVITY METRICS BASE
MON_GET_SERVICE_SUBCLASS 表函数 - 获取服务子类度量值	REQUEST METRICS BASE
MON_GET_SERVICE_SUBCLASS_DETAILS 表函数 - 获取详细的服务子类度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE
MON_GET_TABLESPACE 表函数 - 获取表空间度量值	DATA OBJECT METRICS BASE
MON_GET_UNIT_OF_WORK 表函数 - 获取工作单元度量值	REQUEST METRICS BASE
MON_GET_UNIT_OF_WORK 表函数 - 获取详细的工作单元度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE
MON_GET_WORKLOAD 表函数 - 获取工作负载度量值	REQUEST METRICS BASE
MON_GET_WORKLOAD_DETAILS 表函数 - 获取详细的工作负载度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE

表 2190. 事件监视信息

事件类型	逻辑数据分组	监视开关
活动	event_activity（在 details_xml 文档中报告）	ACTIVITY METRICS BASE
活动	event_activitymetrics	ACTIVITY METRICS BASE
统计信息	event_scstats（在 details_xml 文档中报告）	REQUEST METRICS BASE
统计信息	event_wlstats（在 details_xml 文档中报告）	REQUEST METRICS BASE
工作单元	-	始终收集
程序包高速缓存	-	始终收集
锁定	-	始终收集

用法

要确定在本地缓冲池中发现所请求页的频率，请使用以下公式:

```
(POOL_DATA_LBP_PAGES_FOUND - POOL_ASYNC_DATA_LBP_PAGES_FOUND) / POOL_DATA_L_READS
```

要确定在组缓冲池中发现所请求页的频率，请使用以下公式

```
(POOL_DATA_GBP_L_READS - POOL_DATA_GBP_P_READS) / POOL_DATA_GBP_L_READS
```

本地缓冲池和组缓冲池命中率都是集群高速缓存设施的整体性能中的重要因素。使用这些公式可帮助您确定本地缓冲池或组缓冲池会否成为数据库吞吐量的限制因素。

pool_index_gbp_invalid_pages -“组缓冲池无效索引页数”监视元素

尝试从组缓冲池读取索引页（因为该页在本地缓冲池中无效）的次数。

表 2191. 表函数监视信息

表函数	监视元素收集级别
MON_FORMAT_XML_METRICS_BY_ROW - 获取所有度量值的基于已格式化行的输出	不适用; 报告 XML 文档中作为格式化函数的输入提供的所有元素
MON_GET_ACTIVITY_DETAILS 表函数 - 获取完整的活动详细信息（在 XML 文档 DETAILS 中报告）	ACTIVITY METRICS BASE
MON_GET_BUFFERPOOL 表函数 - 获取缓冲池度量值	DATA OBJECT METRICS BASE
MON_GET_CONNECTION 表函数 - 获取连接度量值	REQUEST METRICS BASE
MON_GET_CONNECTION_DETAILS 表函数 - 获取详细的连接度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE
MON_GET_PKG_CACHE_STMT 表函数 - 获取程序包高速缓存中的 SQL 语句活动度量值	ACTIVITY METRICS BASE
MON_GET_SERVICE_SUBCLASS 表函数 - 获取服务子类度量值	REQUEST METRICS BASE
MON_GET_SERVICE_SUBCLASS_DETAILS 表函数 - 获取详细的服务子类度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE
MON_GET_TABLESPACE 表函数 - 获取表空间度量值	DATA OBJECT METRICS BASE
MON_GET_UNIT_OF_WORK 表函数 - 获取工作单元度量值	REQUEST METRICS BASE
MON_GET_UNIT_OF_WORK 表函数 - 获取详细的工作单元度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE
MON_GET_WORKLOAD 表函数 - 获取工作负载度量值	REQUEST METRICS BASE
MON_GET_WORKLOAD_DETAILS 表函数 - 获取详细的工作负载度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE

表 2192. 事件监视信息

事件类型	逻辑数据分组	监视开关
活动	event_activity（在 details_xml 文档中报告）	ACTIVITY METRICS BASE
活动	event_activitymetrics	ACTIVITY METRICS BASE
统计信息	event_scstats（在 details_xml 文档中报告）	REQUEST METRICS BASE
统计信息	event_wlstats（在 details_xml 文档中报告）	REQUEST METRICS BASE
工作单元	-	始终收集
程序包高速缓存	-	始终收集
锁定	-	始终收集

用法

要确定在本地缓冲池中发现所请求索引页的频率，请使用以下公式:

```
(POOL_INDEX_LBP_PAGES_FOUND - POOL_ASYNC_INDEX_LBP_PAGES_FOUND) / POOL_INDEX_L_READS
```

要确定在组缓冲池中发现所请求索引页的次数，请使用以下公式

```
(POOL_INDEX_GBP_L_READS - POOL_INDEX_GBP_P_READS) / POOL_INDEX_GBP_L_READS
```

本地缓冲池和组缓冲池命中率都是集群高速缓存设施的整体性能中的重要因素。使用这些公式可帮助您确定本地缓冲池或组缓冲池会否成为数据库吞吐量的限制因素。

pool_index_gbp_l_reads -“组缓冲池索引逻辑读取数”监视元素

尝试从组缓冲池读取依赖于组缓冲池 (GBP) 的索引页（因为该页在本地缓冲池中无效或不存在）的次数。

表 2193. 表函数监视信息

表函数	监视元素收集级别
MON_FORMAT_XML_METRICS_BY_ROW - 获取所有度量值的基于已格式化行的输出	不适用; 报告 XML 文档中作为格式化函数的输入提供的所有元素
MON_GET_ACTIVITY_DETAILS 表函数 - 获取完整的活动详细信息（在 XML 文档 DETAILS 中报告）	ACTIVITY METRICS BASE
MON_GET_BUFFERPOOL 表函数 - 获取缓冲池度量值	DATA OBJECT METRICS BASE
MON_GET_CONNECTION 表函数 - 获取连接度量值	REQUEST METRICS BASE
MON_GET_CONNECTION_DETAILS 表函数 - 获取详细的连接度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE
MON_GET_PKG_CACHE_STMT 表函数 - 获取程序包高速缓存中的 SQL 语句活动度量值	ACTIVITY METRICS BASE
MON_GET_SERVICE_SUBCLASS 表函数 - 获取服务子类度量值	REQUEST METRICS BASE

表 2193. 表函数监视信息 (续)

表函数	监视元素收集级别
MON_GET_SERVICE_SUBCLASS_DETAILS 表函数 - 获取详细的服务子类度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE
MON_GET_TABLESPACE 表函数 - 获取表空间度量值	DATA OBJECT METRICS BASE
MON_GET_UNIT_OF_WORK 表函数 - 获取工作单元度量值	REQUEST METRICS BASE
MON_GET_UNIT_OF_WORK 表函数 - 获取详细的工作单元度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE
MON_GET_WORKLOAD 表函数 - 获取工作负载度量值	REQUEST METRICS BASE
MON_GET_WORKLOAD_DETAILS 表函数 - 获取详细的工作负载度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE

表 2194. 事件监视信息

事件类型	逻辑数据分组	监视开关
活动	event_activity（在 details_xml 文档中报告）	ACTIVITY METRICS BASE
活动	event_activitymetrics	ACTIVITY METRICS BASE
统计信息	event_scstats（在 details_xml 文档中报告）	REQUEST METRICS BASE
统计信息	event_wlstats（在 details_xml 文档中报告）	REQUEST METRICS BASE
工作单元	-	始终收集
程序包高速缓存	-	始终收集
锁定	-	始终收集

用法

要确定在本地缓冲池中发现所请求索引页的频率，请使用以下公式:

```
(POOL_INDEX_LBP_PAGES_FOUND - POOL_ASYNC_INDEX_LBP_PAGES_FOUND) / POOL_INDEX_L_READS
```

要确定在组缓冲池中发现所请求索引页的次数，请使用以下公式

```
(POOL_INDEX_GBP_L_READS - POOL_INDEX_GBP_P_READS) / POOL_INDEX_GBP_L_READS
```

本地缓冲池和组缓冲池命中率都是集群高速缓存设施的整体性能中的重要因素。使用这些公式可帮助您确定本地缓冲池或组缓冲池会否成为数据库吞吐量的限制因素。

pool_index_gbp_p_reads -“组缓冲池索引物理读取数”监视元素

尝试将磁盘中依赖于组缓冲池 (GBP) 的索引页读取到本地缓冲池中（因为在 GBP 中找不到该页）的次数。

表 2195. 表函数监视信息

表函数	监视元素收集级别
MON_FORMAT_XML_METRICS_BY_ROW - 获取所有度量值的基于已格式化行的输出	不适用; 报告 XML 文档中作为格式化函数的输入提供的所有元素
MON_GET_ACTIVITY_DETAILS 表函数 - 获取完整的活动详细信息（在 XML 文档 DETAILS 中报告）	ACTIVITY METRICS BASE
MON_GET_BUFFERPOOL 表函数 - 获取缓冲池度量值	DATA OBJECT METRICS BASE
MON_GET_CONNECTION 表函数 - 获取连接度量值	REQUEST METRICS BASE
MON_GET_CONNECTION_DETAILS 表函数 - 获取详细的连接度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE
MON_GET_PKG_CACHE_STMT 表函数 - 获取程序包高速缓存中的 SQL 语句活动度量值	ACTIVITY METRICS BASE
MON_GET_SERVICE_SUBCLASS 表函数 - 获取服务子类度量值	REQUEST METRICS BASE
MON_GET_SERVICE_SUBCLASS_DETAILS 表函数 - 获取详细的服务子类度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE
MON_GET_TABLESPACE 表函数 - 获取表空间度量值	DATA OBJECT METRICS BASE
MON_GET_UNIT_OF_WORK 表函数 - 获取工作单元度量值	REQUEST METRICS BASE
MON_GET_UNIT_OF_WORK 表函数 - 获取详细的工作单元度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE
MON_GET_WORKLOAD 表函数 - 获取工作负载度量值	REQUEST METRICS BASE
MON_GET_WORKLOAD_DETAILS 表函数 - 获取详细的工作负载度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE

表 2196. 事件监视信息

事件类型	逻辑数据分组	监视开关
活动	event_activity（在 details_xml 文档中报告）	ACTIVITY METRICS BASE
活动	event_activitymetrics	ACTIVITY METRICS BASE
统计信息	event_scstats（在 details_xml 文档中报告）	REQUEST METRICS BASE

表 2196. 事件监视信息 (续)

事件类型	逻辑数据分组	监视开关
统计信息	event_wlstats（在 details_xml 文档中报告）	REQUEST METRICS BASE
工作单元	-	始终收集
程序包高速缓存	-	始终收集
锁定	-	始终收集

用法

要确定在本地缓冲池中发现所请求索引页的频率，请使用以下公式:

```
(POOL_INDEX_LBP_PAGES_FOUND - POOL_ASYNC_INDEX_LBP_PAGES_FOUND) / POOL_INDEX_L_READS
```

要确定在组缓冲池中发现所请求索引页的次数，请使用以下公式

```
(POOL_INDEX_GBP_L_READS - POOL_INDEX_GBP_P_READS) / POOL_INDEX_GBP_L_READS
```

本地缓冲池和组缓冲池命中率都是集群高速缓存设施的整体性能中的重要因素。使用这些公式可帮助您确定本地缓冲池或组缓冲池会否成为数据库吞吐量的限制因素。

pool_index_lbp_pages_found -“发现的本地缓冲池索引页数”监视元素

索引页出现在本地缓冲池中的次数。

表 2197. 表函数监视信息

表函数	监视元素收集级别
MON_FORMAT_XML_METRICS_BY_ROW - 获取所有度量值的基于已格式化行的输出	不适用; 报告 XML 文档中作为格式化函数的输入提供的所有元素
MON_GET_ACTIVITY_DETAILS 表函数 - 获取完整的活动详细信息（在 XML 文档 DETAILS 中报告）	ACTIVITY METRICS BASE
MON_GET_BUFFERPOOL 表函数 - 获取缓冲池度量值	DATA OBJECT METRICS BASE
MON_GET_CONNECTION 表函数 - 获取连接度量值	REQUEST METRICS BASE
MON_GET_CONNECTION_DETAILS 表函数 - 获取详细的连接度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE
MON_GET_PKG_CACHE_STMT 表函数 - 获取程序包高速缓存中的 SQL 语句活动度量值	ACTIVITY METRICS BASE
MON_GET_SERVICE_SUBCLASS 表函数 - 获取服务子类度量值	REQUEST METRICS BASE
MON_GET_SERVICE_SUBCLASS_DETAILS 表函数 - 获取详细的服务子类度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE
MON_GET_TABLESPACE 表函数 - 获取表空间度量值	DATA OBJECT METRICS BASE

表 2197. 表函数监视信息 (续)

表函数	监视元素收集级别
MON_GET_UNIT_OF_WORK 表函数 - 获取工作单元度量值	REQUEST METRICS BASE
MON_GET_UNIT_OF_WORK 表函数 - 获取详细的工作单元度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE
MON_GET_WORKLOAD 表函数 - 获取工作负载度量值	REQUEST METRICS BASE
MON_GET_WORKLOAD_DETAILS 表函数 - 获取详细的工作负载度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE

表 2198. 事件监视信息

事件类型	逻辑数据分组	监视开关
活动	event_activity（在 details_xml 文档中报告）	ACTIVITY METRICS BASE
活动	event_activitymetrics	ACTIVITY METRICS BASE
统计信息	event_scstats（在 details_xml 文档中报告）	REQUEST METRICS BASE
统计信息	event_wlstats（在 details_xml 文档中报告）	REQUEST METRICS BASE
工作单元	-	始终收集
程序包高速缓存	-	始终收集
锁定	-	始终收集

用法

要确定在本地缓冲池中发现所请求索引页的频率，请使用以下公式:

```
(POOL_INDEX_LBP_PAGES_FOUND - POOL_ASYNC_INDEX_LBP_PAGES_FOUND) / POOL_INDEX_L_READS
```

要确定在组缓冲池中发现所请求索引页的次数，请使用以下公式

```
(POOL_INDEX_GBP_L_READS - POOL_INDEX_GBP_P_READS) / POOL_INDEX_GBP_L_READS
```

本地缓冲池和组缓冲池命中率都是集群高速缓存设施的整体性能中的重要因素。使用这些公式可帮助您确定本地缓冲池或组缓冲池会否成为数据库吞吐量的限制因素。

pool_xda_gbp_invalid_pages -“组缓冲池无效 XDA 数据页数”监视元素

向组缓冲池发出针对 XML 存储器对象 (XDA) 的数据页的请求（因为该页在本地缓冲池中标记为无效）的次数。

表 2199. 表函数监视信息

表函数	监视元素收集级别
MON_FORMAT_XML_METRICS_BY_ROW - 获取所有度量值的基于已格式化行的输出	不适用; 报告 XML 文档中作为格式化函数的输入提供的所有元素

表 2199. 表函数监视信息 (续)

表函数	监视元素收集级别
MON_GET_ACTIVITY_DETAILS 表函数 - 获取完整的活动详细信息（在 XML 文档 DETAILS 中报告）	ACTIVITY METRICS BASE
MON_GET_BUFFERPOOL 表函数 - 获取缓冲池度量值	DATA OBJECT METRICS BASE
MON_GET_CONNECTION 表函数 - 获取连接度量值	REQUEST METRICS BASE
MON_GET_CONNECTION_DETAILS 表函数 - 获取详细的连接度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE
MON_GET_PKG_CACHE_STMT 表函数 - 获取程序包高速缓存中的 SQL 语句活动度量值	ACTIVITY METRICS BASE
MON_GET_PKG_CACHE_STMT_DETAILS 表函数 - 获取程序包高速缓存条目的详细度量值（在 XML 文档 DETAILS 中报告）	ACTIVITY METRICS BASE
MON_GET_SERVICE_SUBCLASS 表函数 - 获取服务子类度量值	REQUEST METRICS BASE
MON_GET_SERVICE_SUBCLASS_DETAILS 表函数 - 获取详细的服务子类度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE
MON_GET_TABLESPACE 表函数 - 获取表空间度量值	DATA OBJECT METRICS BASE
MON_GET_UNIT_OF_WORK 表函数 - 获取工作单元度量值	REQUEST METRICS BASE
MON_GET_UNIT_OF_WORK 表函数 - 获取详细的工作单元度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE
MON_GET_WORKLOAD 表函数 - 获取工作负载度量值	REQUEST METRICS BASE
MON_GET_WORKLOAD_DETAILS 表函数 - 获取详细的工作负载度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE

表 2200. 事件监视信息

事件类型	逻辑数据分组	监视开关
活动	event_activity（在 details_xml 文档中报告）	ACTIVITY METRICS BASE
活动	event_activitymetrics	ACTIVITY METRICS BASE
统计信息	event_scstats（在 details_xml 文档中报告）	REQUEST METRICS BASE
统计信息	event_wlstats（在 details_xml 文档中报告）	REQUEST METRICS BASE
工作单元	在 system_metrics 文档中报告。	REQUEST METRICS BASE

表 2200. 事件监视信息 (续)

事件类型	逻辑数据分组	监视开关
程序包高速缓存	在 activity_metrics 文档中报告。	ACTIVITY METRICS BASE

用法

要确定在本地缓冲池中发现所请求 XDA 页的频率，请使用以下公式:

```
(pool_xda_lbp_pages_found - pool_async_xda_lbp_pages_found) / pool_xda_l_reads
```

要确定在组缓冲池中发现所请求 XDA 页的次数，请使用以下公式

```
(pool_xda_gbp_l_reads - pool_xda_gbp_p_reads) / pool_xda_gbp_l_reads
```

本地缓冲池和组缓冲池命中率都是集群高速缓存设施的整体性能中的重要因素。使用这些公式可帮助您确定本地缓冲池或组缓冲池会否成为数据库吞吐量的限制因素。

pool_xda_gbp_l_reads -“组缓冲池 XDA 数据逻辑读取请求数”监视元素

尝试从组缓冲池读取 XML 存储器对象 (XDA) 的依赖于 GBP 的数据页（因为该页在本地缓冲池中无效或不存在）的次数。

表 2201. 表函数监视信息

表函数	监视元素收集级别
MON_FORMAT_XML_METRICS_BY_ROW - 获取所有度量值的基于已格式化行的输出	不适用; 报告 XML 文档中作为格式化函数的输入提供的所有元素
MON_GET_ACTIVITY_DETAILS 表函数 - 获取完整的活动详细信息（在 XML 文档 DETAILS 中报告）	ACTIVITY METRICS BASE
MON_GET_BUFFERPOOL 表函数 - 获取缓冲池度量值	DATA OBJECT METRICS BASE
MON_GET_CONNECTION 表函数 - 获取连接度量值	REQUEST METRICS BASE
MON_GET_CONNECTION_DETAILS 表函数 - 获取详细的连接度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE
MON_GET_PKG_CACHE_STMT 表函数 - 获取程序包高速缓存中的 SQL 语句活动度量值	ACTIVITY METRICS BASE
MON_GET_PKG_CACHE_STMT_DETAILS 表函数 - 获取程序包高速缓存条目的详细度量值（在 XML 文档 DETAILS 中报告）	ACTIVITY METRICS BASE
MON_GET_SERVICE_SUBCLASS 表函数 - 获取服务子类度量值	REQUEST METRICS BASE
MON_GET_SERVICE_SUBCLASS_DETAILS 表函数 - 获取详细的服务子类度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE
MON_GET_TABLESPACE 表函数 - 获取表空间度量值	DATA OBJECT METRICS BASE

表 2201. 表函数监视信息 (续)

表函数	监视元素收集级别
MON_GET_UNIT_OF_WORK 表函数 - 获取工作单元度量值	REQUEST METRICS BASE
MON_GET_UNIT_OF_WORK 表函数 - 获取详细的工作单元度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE
MON_GET_WORKLOAD 表函数 - 获取工作负载度量值	REQUEST METRICS BASE
MON_GET_WORKLOAD_DETAILS 表函数 - 获取详细的工作负载度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE

表 2202. 事件监视信息

事件类型	逻辑数据分组	监视开关
活动	event_activity（在 details_xml 文档中报告）	ACTIVITY METRICS BASE
活动	event_activitymetrics	ACTIVITY METRICS BASE
统计信息	event_scstats（在 details_xml 文档中报告）	REQUEST METRICS BASE
统计信息	event_wlstats（在 details_xml 文档中报告）	REQUEST METRICS BASE
工作单元	在 system_metrics 文档中报告。	REQUEST METRICS BASE
程序包高速缓存	在 activity_metrics 文档中报告。	ACTIVITY METRICS BASE

用法

要确定在本地缓冲池中发现所请求 XDA 页的频率，请使用以下公式:

```
(pool_xda_lbp_pages_found - pool_async_xda_lbp_pages_found) / pool_xda_l_reads
```

要确定在组缓冲池中发现所请求 XDA 页的次数，请使用以下公式

```
(pool_xda_gbp_l_reads - pool_xda_gbp_p_reads) / pool_xda_gbp_l_reads
```

本地缓冲池和组缓冲池命中率都是集群高速缓存设施的整体性能中的重要因素。使用这些公式可帮助您确定本地缓冲池或组缓冲池会否成为数据库吞吐量的限制因素。

pool_xda_gbp_p_reads -“组缓冲池 XDA 数据物理读取请求数”监视元素

尝试将磁盘中 XML 存储器对象 (XDA) 的依赖于 GBP 的数据页读取到本地缓冲池中（因为在组缓冲池中找不到该页）的次数。

表 2203. 表函数监视信息

表函数	监视元素收集级别
MON_FORMAT_XML_METRICS_BY_ROW - 获取所有度量值的基于已格式化行的输出	不适用; 报告 XML 文档中作为格式化函数的输入提供的所有元素

表 2203. 表函数监视信息 (续)

表函数	监视元素收集级别
MON_GET_ACTIVITY_DETAILS 表函数 - 获取完整的活动详细信息（在 XML 文档 DETAILS 中报告）	ACTIVITY METRICS BASE
MON_GET_BUFFERPOOL 表函数 - 获取缓冲池度量值	DATA OBJECT METRICS BASE
MON_GET_CONNECTION 表函数 - 获取连接度量值	REQUEST METRICS BASE
MON_GET_CONNECTION_DETAILS 表函数 - 获取详细的连接度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE
MON_GET_PKG_CACHE_STMT 表函数 - 获取程序包高速缓存中的 SQL 语句活动度量值	ACTIVITY METRICS BASE
MON_GET_PKG_CACHE_STMT_DETAILS 表函数 - 获取程序包高速缓存条目的详细度量值（在 XML 文档 DETAILS 中报告）	ACTIVITY METRICS BASE
MON_GET_SERVICE_SUBCLASS 表函数 - 获取服务子类度量值	REQUEST METRICS BASE
MON_GET_SERVICE_SUBCLASS_DETAILS 表函数 - 获取详细的服务子类度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE
MON_GET_TABLESPACE 表函数 - 获取表空间度量值	DATA OBJECT METRICS BASE
MON_GET_UNIT_OF_WORK 表函数 - 获取工作单元度量值	REQUEST METRICS BASE
MON_GET_UNIT_OF_WORK 表函数 - 获取详细的工作单元度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE
MON_GET_WORKLOAD 表函数 - 获取工作负载度量值	REQUEST METRICS BASE
MON_GET_WORKLOAD_DETAILS 表函数 - 获取详细的工作负载度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE

表 2204. 事件监视信息

事件类型	逻辑数据分组	监视开关
活动	event_activity（在 details_xml 文档中报告）	ACTIVITY METRICS BASE
活动	event_activitymetrics	ACTIVITY METRICS BASE
统计信息	event_scstats（在 details_xml 文档中报告）	REQUEST METRICS BASE
统计信息	event_wlstats（在 details_xml 文档中报告）	REQUEST METRICS BASE
工作单元	在 system_metrics 文档中报告。	REQUEST METRICS BASE

表 2204. 事件监视信息 (续)

事件类型	逻辑数据分组	监视开关
程序包高速缓存	在 activity_metrics 文档中报告。	ACTIVITY METRICS BASE

用法

要确定在本地缓冲池中发现所请求 XDA 页的频率，请使用以下公式:

```
(pool_xda_lbp_pages_found - pool_async_xda_lbp_pages_found) / pool_xda_l_reads
```

要确定在组缓冲池中发现所请求 XDA 页的次数，请使用以下公式

```
(pool_xda_gbp_l_reads - pool_xda_gbp_p_reads) / pool_xda_gbp_l_reads
```

本地缓冲池和组缓冲池命中率都是集群高速缓存设施的整体性能中的重要因素。使用这些公式可帮助您确定本地缓冲池或组缓冲池会否成为数据库吞吐量的限制因素。

pool_xda_lbp_pages_found -“发现的本地缓冲池 XDA 数据页数”监视元素

向本地缓冲池请求并在其中发现 XML 存储器对象 (XDA) 的数据页的次数。

表 2205. 表函数监视信息

表函数	监视元素收集级别
MON_FORMAT_XML_METRICS_BY_ROW - 获取所有度量值的基于已格式化行的输出	不适用; 报告 XML 文档中作为格式化函数的输入提供的所有元素
MON_GET_ACTIVITY_DETAILS 表函数 - 获取完整的活动详细信息（在 XML 文档 DETAILS 中报告）	ACTIVITY METRICS BASE
MON_GET_BUFFERPOOL 表函数 - 获取缓冲池度量值	DATA OBJECT METRICS BASE
MON_GET_CONNECTION 表函数 - 获取连接度量值	REQUEST METRICS BASE
MON_GET_CONNECTION_DETAILS 表函数 - 获取详细的连接度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE
MON_GET_PKG_CACHE_STMT 表函数 - 获取程序包高速缓存中的 SQL 语句活动度量值	ACTIVITY METRICS BASE
MON_GET_PKG_CACHE_STMT_DETAILS 表函数 - 获取程序包高速缓存条目的详细度量值（在 XML 文档 DETAILS 中报告）	ACTIVITY METRICS BASE
MON_GET_SERVICE_SUBCLASS 表函数 - 获取服务子类度量值	REQUEST METRICS BASE
MON_GET_SERVICE_SUBCLASS_DETAILS 表函数 - 获取详细的服务子类度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE
MON_GET_TABLESPACE 表函数 - 获取表空间度量值	DATA OBJECT METRICS BASE

表 2205. 表函数监视信息 (续)

表函数	监视元素收集级别
MON_GET_UNIT_OF_WORK 表函数 - 获取工作单元度量值	REQUEST METRICS BASE
MON_GET_UNIT_OF_WORK 表函数 - 获取详细的工作单元度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE
MON_GET_WORKLOAD 表函数 - 获取工作负载度量值	REQUEST METRICS BASE
MON_GET_WORKLOAD_DETAILS 表函数 - 获取详细的工作负载度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE

表 2206. 事件监视信息

事件类型	逻辑数据分组	监视开关
活动	event_activity（在 details_xml 文档中报告）	ACTIVITY METRICS BASE
活动	event_activitymetrics	ACTIVITY METRICS BASE
统计信息	event_scstats（在 details_xml 文档中报告）	REQUEST METRICS BASE
统计信息	event_wlstats（在 details_xml 文档中报告）	REQUEST METRICS BASE
工作单元	在 system_metrics 文档中报告。	REQUEST METRICS BASE
程序包高速缓存	在 activity_metrics 文档中报告。	ACTIVITY METRICS BASE

用法

要确定在本地缓冲池中发现所请求 XDA 页的频率，请使用以下公式:

```
(pool_xda_lbp_pages_found - pool_async_xda_lbp_pages_found) / pool_xda_l_reads
```

要确定在组缓冲池中发现所请求 XDA 页的次数，请使用以下公式

```
(pool_xda_gbp_l_reads - pool_xda_gbp_p_reads) / pool_xda_gbp_l_reads
```

本地缓冲池和组缓冲池命中率都是集群高速缓存设施的整体性能中的重要因素。使用这些公式可帮助您确定本地缓冲池或组缓冲池会否成为数据库吞吐量的限制因素。

reclaim_wait_time -“回收等待时间”监视元素

在 DB2 pureScale 环境中，此元素表示等待页锁定所耗的时间量（其中锁定请求导致页被回收）。时间的度量单位为毫秒。

表 2207. 表函数监视信息

表函数	监视元素收集级别
MON_FORMAT_XML_METRICS_BY_ROW - 获取所有度量值的基于已格式化行的输出	不适用；报告 XML 文档中作为格式化函数的输入提供的所有元素

表 2207. 表函数监视信息 (续)

表函数	监视元素收集级别
MON_FORMAT_XML_TIMES_BY_ROW - - 获取基于已格式化行的组合层次结构等待时间和处理时间	不适用; 报告 XML 文档中作为格式化函数的输入提供的所有元素
MON_FORMAT_XML_WAIT_TIMES_BY _ROW - 获取等待时间的已格式化的基于行的输出	不适用; 报告 XML 文档中作为格式化函数的输入提供的所有元素
MON_GET_ACTIVITY_DETAILS 表函数 - 获取完整的活动详细信息（在 XML 文档 DETAILS 中报告）	ACTIVITY METRICS BASE
MON_GET_CONNECTION 表函数 - 获取连接度量值	REQUEST METRICS BASE
MON_GET_CONNECTION_DETAILS 表函数 - 获取详细的连接度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE
MON_GET_PAGE_ACCESS_INFO 表函数 - 获取缓冲池页等待信息	ACTIVITY METRICS BASE
MON_GET_PKG_CACHE_STMT 表函数 - 获取程序包高速缓存中的 SQL 语句活动度量值	ACTIVITY METRICS BASE
MON_GET_SERVICE_SUBCLASS 表函数 - 获取服务子类度量值	REQUEST METRICS BASE
MON_GET_SERVICE_SUBCLASS_DETAILS 表函数 - 获取详细的服务子类度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE
MON_GET_UNIT_OF_WORK 表函数 - 获取工作单元度量值	REQUEST METRICS BASE
MON_GET_UNIT_OF_WORK 表函数 - 获取详细的工作单元度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE
MON_GET_WORKLOAD 表函数 - 获取工作负载度量值	ACTIVITY METRICS BASE
MON_GET_WORKLOAD_DETAILS 表函数 - 获取详细的工作负载度量值（在 XML 文档 DETAILS 中报告）	ACTIVITY METRICS BASE

表 2208. 事件监视信息

事件类型	逻辑数据分组	监视开关
活动	event_activity（在 details_xml 文档中报告）	ACTIVITY METRICS BASE
活动	event_activitymetrics	ACTIVITY METRICS BASE
统计信息	event_scstats（在 details_xml 文档中报告）	REQUEST METRICS BASE
统计信息	event_wlstats（在 details_xml 文档中报告）	REQUEST METRICS BASE
工作单元	-	始终收集

用法

等待针对空间映射页的回收所耗的时间是独立计算的，并且在 **spacemappage_reclaim_wait_time** 监视元素中报告。

spacemappage_page_reclaims_initiated_s -“共享存取导致的空间映射页回收数”监视元素

以空间映射页的共享方式访问页从而导致从另一成员回收此页的次数。

表 2209. 表函数监视信息

表函数	监视元素收集级别
MON_GET_PAGE_ACCESS_INFO 表函数 - 获取缓冲池页等待信息	ACTIVITY METRICS BASE

用法

仅对与对象相关的表空间（即，对可回收存储器启用的表空间）报告此值。使用 **reclaimable_space_enabled** 监视元素来确定是否对可回收存储器启用了此表空间。

因为扩展数据块映射页 (EMP) 是元数据，所以 EMP 包括在此监视元素的值中。

数据空间映射页包含用户数据，因此它们除了包括在 **spacemappage_page_reclaims_initiated_s** 监视元素的值中以外，还包括在 **page_reclaims_initiated_s** 监视元素的值中。索引空间页未包含用户数据，因此它们仅包括在 **spacemappage_page_reclaims_initiated_s** 监视元素的值中。

spacemappage_page_reclaims_initiated_x -“互斥存取导致的空间映射页回收数”监视元素

以空间映射页的互斥方式访问页从而导致从另一成员回收此页的次数。

表 2210. 表函数监视信息

表函数	监视元素收集级别
MON_GET_PAGE_ACCESS_INFO 表函数 - 获取缓冲池页等待信息	ACTIVITY METRICS BASE

用法

仅对与对象相关的表空间（即，对可回收存储器启用的表空间）报告此值。使用 **reclaimable_space_enabled** 监视元素来确定是否对可回收存储器启用了此表空间。

因为扩展数据块映射页 (EMP) 是元数据，所以 EMP 包括在此监视元素的值中。

数据空间映射页包含用户数据，因此它们除了包括在 **spacemappage_page_reclaims_initiated_x** 监视元素的值中以外，还包括在 **page_reclaims_initiated_x** 监视元素的值中。索引空间页未包含用户数据，因此它们仅包括在 **spacemappage_page_reclaims_initiated_x** 监视元素的值中。

spacemappage_page_reclaims_s -“共享存取的空间映射页回收数”监视元素

与空间映射页相关的页被 DB2 pureScale实例中的另一成员回收的次数（在计划释放此页之前）。回收此页的成员需要对空间映射页具有共享存取权。

表 2211. 表函数监视信息

表函数	监视元素收集级别
MON_GET_PAGE_ACCESS_INFO 表函数 - 获取缓冲池页等待信息	ACTIVITY METRICS BASE

用法

仅对与对象相关的表空间（即，对可回收存储器启用的表空间）报告此值。使用 **reclaimable_space_enabled** 监视元素来确定是否对可回收存储器启用了此表空间。

因为扩展数据块映射页 (EMP) 是元数据，所以 EMP 包括在此监视元素的值中。

数据空间映射页包含用户数据，因此它们除了包括在 **spacemappage_page_reclaims_s** 监视元素的值中以外，还包括在 **page_reclaims_s** 监视元素的值中。索引空间页未包含用户数据，因此它们仅包括在 **spacemappage_page_reclaims_s** 监视元素的值中。

spacemappage_page_reclaims_x -“互斥存取导致的空间映射页回收数”监视元素

与空间映射页相关的页被 DB2 pureScale实例中的另一成员回收的次数（在计划释放此页之前）。回收此页的成员需要对空间映射页具有互斥存取权。

表 2212. 表函数监视信息

表函数	监视元素收集级别
MON_GET_PAGE_ACCESS_INFO 表函数 - 获取缓冲池页等待信息	ACTIVITY METRICS BASE

用法

仅对与对象相关的表空间（即，对可回收存储器启用的表空间）报告此值。使用 **reclaimable_space_enabled** 监视元素来确定是否对可回收存储器启用了此表空间。

因为扩展数据块映射页 (EMP) 是元数据，所以 EMP 包括在此监视元素的值中。

数据空间映射页包含用户数据，因此它们除了包括在 **spacemappage_page_reclaims_x** 监视元素的值中以外，还包括在 **page_reclaims_x** 监视元素的值中。索引空间映射页未包含用户数据，因此它们仅包括在 **spacemappage_page_reclaims_x** 监视元素的值中。

spacemappage_reclaim_wait_time -“空间映射页回收等待时间”监视元素

在 DB2 pureScale 环境中，此元素表示等待页锁定所耗的时间量（其中锁定请求导致从另一成员回收），这些页锁定针对与内部保留的对象空间管理相关的页。时间的度量单位为毫秒。

表 2213. 表函数监视信息

表函数	监视元素收集级别
MON_FORMAT_XML_METRICS_BY_ROW - 获取所有度量值的基于已格式化行的输出	不适用; 报告 XML 文档中作为格式化函数的输入提供的所有元素
MON_FORMAT_XML_TIMES_BY_ROW - - 获取基于已格式化行的组合层次结构等待时间和处理时间	不适用; 报告 XML 文档中作为格式化函数的输入提供的所有元素
MON_FORMAT_XML_WAIT_TIMES_BY _ROW - 获取等待时间的已格式化的基于行的输出	不适用; 报告 XML 文档中作为格式化函数的输入提供的所有元素
MON_GET_ACTIVITY_DETAILS 表函数 - 获取完整的活动详细信息（在 XML 文档 DETAILS 中报告）	ACTIVITY METRICS BASE
MON_GET_CONNECTION 表函数 - 获取连接度量值	REQUEST METRICS BASE
MON_GET_CONNECTION_DETAILS 表函数 - 获取详细的连接度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE
MON_GET_PAGE_ACCESS_INFO 表函数 - 获取缓冲池页等待信息	REQUEST METRICS BASE
MON_GET_PKG_CACHE_STMT 表函数 - 获取程序包高速缓存中的 SQL 语句活动度量值	ACTIVITY METRICS BASE
MON_GET_SERVICE_SUBCLASS 表函数 - 获取服务子类度量值	REQUEST METRICS BASE
MON_GET_SERVICE_SUBCLASS_DETAILS 表函数 - 获取详细的服务子类度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE
MON_GET_UNIT_OF_WORK 表函数 - 获取工作单元度量值	REQUEST METRICS BASE
MON_GET_UNIT_OF_WORK 表函数 - 获取详细的工作单元度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE
MON_GET_WORKLOAD 表函数 - 获取工作负载度量值	REQUEST METRICS BASE
MON_GET_WORKLOAD_DETAILS 表函数 - 获取详细的工作负载度量值（在 XML 文档 DETAILS 中报告）	REQUEST METRICS BASE

表 2214. 事件监视信息

事件类型	逻辑数据分组	监视开关
活动	event_activity（在 details_xml 文档中报告）	ACTIVITY METRICS BASE
活动	event_activitymetrics	ACTIVITY METRICS BASE
统计信息	event_scstats（在 details_xml 文档中报告）	REQUEST METRICS BASE
统计信息	event_wlstats（在 details_xml 文档中报告）	REQUEST METRICS BASE
工作单元	-	始终收集

table_name -“表名”监视元素

表的名称。

表 2215. 表函数监视信息

表函数	监视元素收集级别
ADMIN_GET_INDEX_COMPRESS_INFO 表函数 - 返回压缩索引信息	ACTIVITY METRICS BASE
ADMIN_GET_INDEX_INFO 表函数 - 返回索引信息	ACTIVITY METRICS BASE
ADMIN_GET_TAB_COMPRESS_INFO 表函数 - 估算压缩节省量	ACTIVITY METRICS BASE
ADMIN_GET_TAB_DICTIONARY_INFO 表函数 - 报告现有表字典的属性	ACTIVITY METRICS BASE
ADMINTABINFO 管理视图和 ADMIN_GET_TAB_INFO 表函数 - 检索表大小和状态信息	ACTIVITY METRICS BASE
ADMINTEMPCOLUMNS 管理视图和 ADMIN_GET_TEMP_COLUMNS 表函数 - 检索临时表的列信息	ACTIVITY METRICS BASE
ADMINTEMPTABLES 管理视图和 ADMIN_GET_TEMP_TABLES 表函数 - 检索临时表的信息	ACTIVITY METRICS BASE
MON_FORMAT_LOCK_NAME 表函数 - 设置内部锁定名称的格式并返回详细信息	ACTIVITY METRICS BASE
MON_GET_INDEX 表函数 - 获取索引度量值	ACTIVITY METRICS BASE
MON_GET_PAGE_ACCESS_INFO 表函数 - 获取缓冲池页等待信息	ACTIVITY METRICS BASE
MON_GET_TABLE 表函数 - 获取表度量值	ACTIVITY METRICS BASE
MON_GET_TABLE_USAGE_LIST 表函数 - 从表用法列表返回信息	ACTIVITY METRICS BASE

表 2216. 快照监视信息

快照级别	逻辑数据分组	监视开关
表	表	基本
应用程序	appl	锁定
锁定	appl_lock_list	锁定
锁定	锁定	锁定
锁定	lock_wait	锁定

表 2217. 事件监视信息

事件类型	逻辑数据分组	监视开关
锁定	-	始终收集
表	event_table	始终收集
死锁 [1]	锁定	始终收集
死锁 [1]	event_dlconn	始终收集
带有详细信息的死锁 [1]	event_detailed_dlconn	始终收集

1 建议不要使用此事件监视器。建议不要再使用此选项，将来的发行版可能会将其除去。请使用 CREATE EVENT MONITOR FOR LOCKING 语句来监视与锁定相关的事件，例如锁定超时、锁定等待和死锁。

用法

通过将此元素与 **table_schema** 配合使用，可以确定资源争用的根源。

在应用程序级别、应用程序锁定级别和死锁监视级别，此项是应用程序等待锁定的表，原因是它目前被另一应用程序锁定。对于快照监视而言，仅当“锁定”监视器组信息设置为 ON 并且 **lock_object_type** 表明应用程序正在等待获取表锁定时，此项才有效。

对于对象锁定级别的快照监视，将对表级别和行级别锁定返回此项。在此级别报告的表就是此应用程序对其持有这些锁定的表。

对于表级别的快照和事件监视，此项是对其收集信息的表。对于临时表，**table_name** 的格式为 『TEMP (*n*, *m*)』，其中:

- *n* 是表空间标识
- *m* 是 **table_file_id** 元素

table_schema -“表模式名”监视元素

表的模式。

表 2218. 表函数监视信息

表函数	监视元素收集级别
ADMIN_GET_INDEX_COMPRESS_INFO 表函数 - 返回压缩索引信息	ACTIVITY METRICS BASE
ADMIN_GET_INDEX_INFO 表函数 - 返回索引信息	ACTIVITY METRICS BASE

表 2218. 表函数监视信息 (续)

表函数	监视元素收集级别
ADMIN_GET_TAB_COMPRESS_INFO 表函数 - 估算压缩节省量	ACTIVITY METRICS BASE
ADMIN_GET_TAB_DICTIONARY_INFO 表函数 - 报告现有表字典的属性	ACTIVITY METRICS BASE
ADMINTABINFO 管理视图和 ADMIN_GET_TAB_INFO 表函数 - 检索表大小和状态信息	ACTIVITY METRICS BASE
ADMINTEMPCOLUMNS 管理视图和 ADMIN_GET_TEMP_COLUMNS 表函数 - 检索临时表的列信息	ACTIVITY METRICS BASE
ADMINTEMPTABLES 管理视图和 ADMIN_GET_TEMP_TABLES 表函数 - 检索临时表的信息	ACTIVITY METRICS BASE
MON_FORMAT_LOCK_NAME 表函数 - 设置内部锁定名称的格式并返回详细信息	ACTIVITY METRICS BASE
MON_GET_INDEX 表函数 - 获取索引度量值	ACTIVITY METRICS BASE
MON_GET_PAGE_ACCESS_INFO 表函数 - 获取缓冲池页等待信息	ACTIVITY METRICS BASE
MON_GET_TABLE 表函数 - 获取表度量值	ACTIVITY METRICS BASE
MON_GET_TABLE_USAGE_LIST 表函数 - 从表用法列表返回信息	ACTIVITY METRICS BASE

表 2219. 快照监视信息

快照级别	逻辑数据分组	监视开关
表	表	基本
应用程序	appl	锁定
锁定	appl_lock_list	锁定
锁定	锁定	锁定
锁定	lock_wait	锁定

表 2220. 事件监视信息

事件类型	逻辑数据分组	监视开关
锁定	-	始终收集
表	event_table	始终收集
死锁 [1]	锁定	始终收集
死锁 [1]	event_dlconn	始终收集
带有详细信息的死锁 [1]	event_detailed_dlconn	始终收集

1 建议不要使用此事件监视器。建议不要再使用此选项，将来的发行版可能会将其除去。请使用 CREATE EVENT MONITOR FOR LOCKING 语句来监视与锁定相关的事件，例如锁定超时、锁定等待和死锁。

用法

通过将此元素与 **table_name** 配合使用，可以确定资源争用的根源。

对于应用程序级别、应用程序锁定级别和死锁监视级别，此项是应用程序等待锁定的表的模式，原因是它目前被另一应用程序锁定。仅当 **lock_object_type** 指示应用程序正在等待获取表锁定时，才设置此元素。对于应用程序级别和应用程序锁定级别的快照监视而言，仅当“锁定”监视器组信息设置为 ON 时，此项才有效。

对于对象锁定级别的快照监视，将对表级别和行级别锁定返回此项。在此级别报告的表就是此应用程序对其持有这些锁定的表。

对于表级别的快照和事件监视，此元素标识对其收集信息的表的模式。对于临时表，**table_schema** 的格式为 『*<agent_id><auth_id>*』，其中:

- *agent_id* 是创建临时表的应用程序的应用程序句柄
- *auth_id* 是应用程序用来连接至数据库的授权标识

tablespace_min_recovery_time -“前滚的最短恢复时间”监视元素

显示可前滚表空间的最早时间点的时间戳记。 此时间戳记反映本地时间。

表 2221. 表函数监视信息

表函数	监视元素收集级别
MON_GET_TABLESPACE 表函数 - 获取表空间度量值	ACTIVITY METRICS BASE

表 2222. 快照监视信息

快照级别	逻辑数据分组	监视开关
表空间	tablespace_nodeinfo	基本

用法

只有不为零时才显示。

target_cf_gbp_size -“目标集群高速缓存设施组缓冲池大小”监视元素

执行动态调整大小操作期间，此监视元素显示组缓冲池内存目标值（以大小为 4 KB 的页计）。目标值与所配置值相匹配时，调整大小操作完成。

表 2223. 表函数监视信息

表函数	监视元素收集级别
MON_GET_CF 表函数 - 获取 CF 度量值	ACTIVITY METRICS BASE

target_cf_lock_size -“目标集群高速缓存设施锁定大小”监视元素

执行动态调整大小操作期间，此监视元素显示全局锁定内存目标值（以大小为 4 KB 的页计）。目标值与所配置值相匹配时，调整大小操作完成。

表 2224. 表函数监视信息

表函数	监视元素收集级别
MON_GET_CF 表函数 - 获取 CF 度量值	ACTIVITY METRICS BASE

target_cf_sca_size -“目标集群高速缓存设施共享通信区大小”监视元素

执行动态调整大小操作期间，此监视元素显示共享通信区内存目标值（以大小为 4 KB 的页计）。目标值与所配置值相匹配时，调整大小操作完成。

表 2225. 表函数监视信息

表函数	监视元素收集级别
MON_GET_CF 表函数 - 获取 CF 度量值	ACTIVITY METRICS BASE

第 4 部分 附录

附录 A. DB2 技术信息概述

DB2 技术信息以多种可以通过多种方法访问的格式提供。

您可以通过下列工具和方法获得 DB2 技术信息:

- DB2 信息中心
  - 主题（任务、概念和参考主题）
  - 样本程序
  - 教程
- DB2 书籍
  - PDF 文件（可下载）
  - PDF 文件（在 DB2 PDF DVD 中）
  - 印刷版书籍
- 命令行帮助
  - 命令帮助
  - 消息帮助

注: DB2 信息中心主题的更新频率比 PDF 书籍或硬拷贝书籍的更新频率高。要获取最新信息，请安装可用的文档更新或者参阅 ibm.com 上的 DB2 信息中心。

您可以在线访问 ibm.com 上的其他 DB2 技术信息，例如技术说明、白皮书和 IBM Redbooks® 出版物。请访问以下网址处的 DB2 信息管理软件资料库站点: http://www.ibm.com/software/data/sw-library/。

文档反馈

我们非常重视您对 DB2 文档的反馈。如果您想就如何改善 DB2 文档提出建议，请向 db2docs@ca.ibm.com 发送电子邮件。DB2 文档小组将阅读您的所有反馈，但无法直接给您答复。请尽可能提供具体的示例，这样我们才能更好地了解您所关心的问题。如果您要提供有关具体主题或帮助文件的反馈，请加上标题和 URL。

请不要使用以上电子邮件地址与 DB2 客户支持机构联系。如果您遇到文档无法解决的 DB2 技术问题，请与您当地的 IBM 服务中心联系以获得帮助。

硬拷贝或 PDF 格式的 DB2 技术库

下列各表描述 IBM 出版物中心（网址为 www.ibm.com/e-business/linkweb/publications/servlet/pbi.wss）所提供的 DB2 资料库。可从 www.ibm.com/support/docview.wss?rs=71&uid=swg2700947 下载 PDF 格式的 DB2 V10.1 手册的英文版本和翻译版本。

尽管这些表标识书籍有印刷版，但可能未在您所在国家或地区提供。

每次更新手册时，表单号都会递增。确保您正在阅读下面列示的手册的最新版本。

注: *DB2 信息中心*的更新频率比 PDF 或硬拷贝书籍的更新频率高。

表 2226. DB2 技术信息

书名	书号	是否提供印刷版	最近一次更新时间
Administrative API Reference	SC27-3864-00	是	2012 年 4 月
Administrative Routines and Views	SC27-3865-00	否	2012 年 4 月
Call Level Interface Guide and Reference Volume 1	SC27-3866-00	是	2012 年 4 月
Call Level Interface Guide and Reference Volume 2	SC27-3867-00	是	2012 年 4 月
Command Reference	SC27-3868-00	是	2012 年 4 月
数据库管理概念和配置参考	S151-1758-00	是	2012 年 4 月
Data Movement Utilities Guide and Reference	S151-1756-00	是	2012 年 4 月
数据库监视指南和参考	S151-1759-00	是	2012 年 4 月
数据恢复及高可用性指南与参考	S151-1755-00	是	2012 年 4 月
数据库安全性指南	S151-1753-01	是	2012 年 4 月
DB2 Workload Management Guide and Reference	SC27-3891-00	是	2012 年 4 月
开发 ADO.NET 和 OLE DB 应用程序	S151-1765-00	是	2012 年 4 月
开发嵌入式 SQL 应用程序	S151-1763-00	是	2012 年 4 月
Developing Java Applications	SC27-3875-00	是	2012 年 4 月
Developing Perl, PHP, Python, and Ruby on Rails Applications	SC27-3876-00	否	2012 年 4 月
开发用户定义的例程（SQL 和外部例程）	S151-1761-00	是	2012 年 4 月
数据库应用程序开发入门	G151-1764-00	是	2012 年 4 月
Linux 和 Windows 上的 DB2 安装和管理入门	G151-1769-00	是	2012 年 4 月
全球化指南	S151-1757-00	是	2012 年 4 月
安装 DB2 服务器	G151-1768-00	是	2012 年 4 月
安装 IBM Data Server Client	G151-1751-00	否	2012 年 4 月
消息参考第 1 卷	S151-1767-00	否	2012 年 4 月
消息参考第 2 卷	S151-1766-00	否	2012 年 4 月
Net Search Extender 管理和用户指南	S151-1078-00	否	2012 年 4 月

表 2226. DB2 技术信息 (续)

书名	书号	是否提供印刷版	最近一次更新时间
分区和集群指南	S151-1754-00	是	2012 年 4 月
pureXML 指南	S151-1775-00	是	2012 年 4 月
Spatial Extender User's Guide and Reference	SC27-3894-00	否	2012 年 4 月
《SQL 过程语言: 应用程序启用和支持》	S151-1762-00	是	2012 年 4 月
SQL Reference Volume 1	SC27-3885-00	是	2012 年 4 月
SQL Reference Volume 2	SC27-3886-00	是	2012 年 4 月
Text Search Guide	SC27-3888-00	是	2012 年 4 月
故障诊断和调整数据库性能	S151-1760-00	是	2012 年 4 月
升级到 DB2 V10.1	S151-1770-00	是	2012 年 4 月
DB2 V10.1 新增内容	S151-1752-00	是	2012 年 4 月
XQuery 参考	S151-1774-00	否	2012 年 4 月

表 2227. 特定于 DB2 Connect 的技术信息

书名	书号	是否提供印刷版	最近一次更新时间
DB2 Connect 安装和配置 DB2 Connect Personal Edition	S151-1773-00	是	2012 年 4 月
DB2 Connect 安装和配置 DB2 Connect 服务器	S151-1772-00	是	2012 年 4 月
DB2 Connect 用户指南	S151-1771-00	是	2012 年 4 月

从命令行处理器显示 SQL 状态帮助

DB2 产品针对可能充当 SQL 语句结果的条件返回 SQLSTATE 值。SQLSTATE 帮助说明 SQL 状态和 SQL 状态类代码的含义。

过程

要启动 SQL 状态帮助，请打开命令行处理器并输入:

```
? sqlstate or ? class code
```

其中，*sqlstate* 表示有效的 5 位 SQL 状态，*class code* 表示该 SQL 状态的前 2 位。例如，`? 08003` 显示 08003 SQL 状态的帮助，而 `? 08` 显示 08 类代码的帮助。

访问不同版本的 DB2 信息中心

您可以在 ibm.com® 上的不同信息中心中找到其他版本 DB2 产品的文档。

关于此任务

对于 DB2 V10.1 主题，*DB2 信息中心* URL 是 http://publib.boulder.ibm.com/infocenter/db2luw/v10r1。

对于 DB2 V9.8 主题，*DB2 信息中心* URL 是 http://publib.boulder.ibm.com/infocenter/db2luw/v9r8/。

对于 DB2 V9.7 主题，*DB2 信息中心* URL 是 http://publib.boulder.ibm.com/infocenter/db2luw/v9r7/。

对于 DB2 V9.5 主题，*DB2 信息中心* URL 是 http://publib.boulder.ibm.com/infocenter/db2luw/v9r5。

对于 DB2 V9.1 主题，*DB2 信息中心* URL 是 http://publib.boulder.ibm.com/infocenter/db2luw/v9/。

对于 DB2 V8 主题，请转至 *DB2 信息中心* URL: http://publib.boulder.ibm.com/infocenter/db2luw/v8/。

更新安装在计算机或内部网服务器上的 DB2 信息中心

安装在本地的 DB2 信息中心必须定期进行更新。

开始之前

必须已安装 DB2 V10.1 信息中心。有关详细信息，请参阅安装 DB2 服务器中的“使用 DB2 安装向导来安装 DB2 信息中心”主题。所有适用于安装信息中心的先决条件和限制同样适用于更新信息中心。

关于此任务

可以自动或手动更新现有的 DB2 信息中心:

- 自动更新将更新现有的信息中心功能部件和语言。自动更新的一个优点是，与手动更新相比，信息中心的不可用时间较短。另外，自动更新可设置为作为定期运行的其他批处理作业的一部分运行。
- 可以使用手动更新方法来更新现有的信息中心功能部件和语言。自动更新可以缩短更新过程中的停机时间，但如果您想添加功能部件或语言，那么必须执行手动过程。例如，如果本地信息中心最初安装的是英语和法语版，而现在还要安装德语版；那么手动更新将安装德语版，并更新现有信息中心的功能和语言。但是，手动更新要求您手动停止、更新和重新启动信息中心。在整个更新过程期间信息中心不可用。在自动更新过程中，信息中心仅在更新完成后停止工作以重新启动信息中心。

此主题详细说明了自动更新的过程。有关手动更新的指示信息，请参阅“手动更新安装在您的计算机或内部网服务器上的 DB2 信息中心”主题。

过程

要自动更新安装在计算机或内部网服务器上的 DB2 信息中心:

1. 在 Linux 操作系统上，
   a. 浏览至信息中心的安装位置。缺省情况下，DB2 信息中心安装在 `/opt/ibm/db2ic/V10.1` 目录中。
   b. 从安装目录浏览至 `doc/bin` 目录。
   c. 运行 `update-ic` 脚本:

update-ic

2. 在 Windows 操作系统上，
   a. 打开命令窗口。
   b. 浏览至信息中心的安装位置。缺省情况下，DB2 信息中心安装在 <Program Files>\IBM\DB2 Information Center\V10.1 目录中，其中 <Program Files> 表示 Program Files 目录的位置。
   c. 从安装目录浏览至 doc\bin 目录。
   d. 运行 update-ic.bat 文件:

      update-ic.bat

结果

DB2 信息中心将自动重新启动。如果更新可用，那么信息中心会显示新的以及更新后的主题。如果信息中心更新不可用，那么会在日志中添加消息。日志文件位于 doc\eclipse\configuration 目录中。日志文件名称是随机生成的编号。例如，1239053440785.log。

手动更新安装在计算机或内部网服务器上的 DB2 信息中心

如果您已在本地安装 DB2 信息中心，那么可从 IBM 获取文档更新并进行安装。

关于此任务

手动更新安装在本地的 *DB2 信息中心*要求您:

1. 停止计算机上的 *DB2 信息中心*，然后以独立方式重新启动信息中心。如果以独立方式运行信息中心，那么网络上的其他用户将无法访问信息中心，因而您可以应用更新。DB2 信息中心的工作站版本总是以独立方式运行。
2. 使用“更新”功能部件来查看可用的更新。如果有您必须安装的更新，那么请使用“更新”功能部件来获取并安装这些更新。

   注: 如果您的环境要求在一台未连接至因特网的机器上安装 *DB2 信息中心*更新，请使用一台已连接至因特网并已安装 *DB2 信息中心*的机器将更新站点镜像至本地文件系统。如果网络中有许多用户将安装文档更新，那么可以通过在本地也为更新站点制作镜像并为更新站点创建代理来缩短每个人执行更新所需要的时间。
   如果提供了更新包，请使用“更新”功能部件来获取这些更新包。但是，只有在单机方式下才能使用“更新”功能部件。
3. 停止独立信息中心，然后在计算机上重新启动 *DB2 信息中心*。

注: 在 Windows 2008、Windows Vista 和更高版本上，稍后列示在此部分的命令必须作为管理员运行。要打开具有全面管理员特权的命令提示符或图形工具，请右键单击快捷方式，然后选择**以管理员身份运行**。

过程

要更新安装在您的计算机或内部网服务器上的 *DB2 信息中心*:

1. 停止 *DB2 信息中心*。
   - 在 Windows 上，单击**开始 > 控制面板 > 管理工具 > 服务**。右键单击 **DB2 信息中心**服务，并选择**停止**。

- 在 Linux 上，输入以下命令:

  ```
  /etc/init.d/db2icdv10 stop
  ```

2. 以独立方式启动信息中心。
   - 在 Windows 上:
     a. 打开命令窗口。
     b. 浏览至信息中心的安装位置。缺省情况下，*DB2 信息中心*安装在 *Program_Files*\IBM\DB2 Information Center\V10.1 目录中，其中 *Program Files* 表示 Program Files 目录的位置。
     c. 从安装目录浏览至 doc\bin 目录。
     d. 运行 help_start.bat 文件:

        ```
        help_start.bat
        ```

   - 在 Linux 上:
     a. 浏览至信息中心的安装位置。缺省情况下，*DB2 信息中心*安装在 /opt/ibm/db2ic/V10.1 目录中。
     b. 从安装目录浏览至 doc/bin 目录。
     c. 运行 help_start 脚本:

        ```
        help_start
        ```

   系统缺省 Web 浏览器将打开以显示独立信息中心。
3. 单击**更新**按钮（  ）。（必须在浏览器中启用 JavaScript。）在信息中心的右边面板上，单击**查找更新**。将显示现有文档的更新列表。
4. 要启动安装过程，请检查您要安装的选项，然后单击**安装更新**。
5. 在安装进程完成后，请单击**完成**。
6. 要停止独立信息中心，请执行下列操作:
   - 在 Windows 上，浏览至安装目录中的 doc\bin 目录并运行 help_end.bat 文件:

     ```
     help_end.bat
     ```

     注: help_end 批处理文件包含安全地停止使用 help_start 批处理文件启动的进程所需的命令。不要使用 Ctrl-C 或任何其他方法来停止 help_start.bat。
   - 在 Linux 上，浏览至安装目录中的 doc/bin 目录并运行 help_end 脚本:

     ```
     help_end
     ```

     注: help_end 脚本包含安全地停止使用 help_start 脚本启动的进程所需的命令。不要使用任何其他方法来停止 help_start 脚本。
7. 重新启动 *DB2 信息中心*。
   - 在 Windows 上，单击**开始 > 控制面板 > 管理工具 > 服务**。右键单击 **DB2 信息中心**服务，并选择**启动**。
   - 在 Linux 上，输入以下命令:

     ```
     /etc/init.d/db2icdv10 start
     ```

结果

更新后的 *DB2 信息中心*将显示新的以及更新后的主题。

DB2 教程

DB2 教程帮助您了解 DB2 数据库产品的各个方面。这些课程提供了逐步指示信息。

开始之前

您可以在信息中心中查看 XHTML 版的教程: http://publib.boulder.ibm.com/infocenter/db2luw/v10r1/。

某些课程使用了样本数据或代码。有关其特定任务的任何先决条件的描述，请参阅教程。

DB2 教程

要查看教程，请单击标题。

*pureXML 指南*中的『**pureXML**』

设置 DB2 数据库以存储 XML 数据以及对本机 XML 数据存储器执行基本操作。

DB2 故障诊断信息

我们提供了各种各样的故障诊断和问题确定信息来帮助您使用 DB2 数据库产品。

DB2 文档

您可以在*故障诊断和调整数据库性能*或者 *DB2 信息中心*的"数据库基础"部分中找到故障诊断信息，这些信息包含以下内容:

- 有关如何使用 DB2 诊断工具和实用程序来隔离和确定问题的信息。
- 一些最常见问题的解决方案。
- 旨在帮助您解决 DB2 数据库产品使用过程中可能会遇到的其他问题的建议。

IBM 支持门户网站

如果您遇到问题并且希望得到帮助以查找可能的原因和解决方案，请访问 IBM 支持门户网站。这个技术支持站点提供了指向最新 DB2 出版物、技术说明、授权程序分析报告（APAR 或错误修订）、修订包和其他资源的链接。可搜索此知识库并查找问题的可能解决方案。

访问 IBM 支持门户网站: http://www.ibm.com/support/entry/portal/Overview/Software/Information_Management/DB2_for_Linux,_UNIX_and_Windows

信息中心条款和条件

如果符合以下条款和条件，那么授予您使用这些出版物的许可权。

适用性: 用户需要遵循 IBM Web 站点的使用条款及以下条款和条件。

个人使用: 只要保留所有的专有权声明，您就可以为个人、非商业使用复制这些出版物。未经 IBM 明确同意，您不可以分发、展示或制作这些出版物或其中任何部分的演绎作品。

商业使用: 只要保留所有的专有权声明，您就可以仅在企业内复制、分发和展示这些出版物。未经 IBM 明确同意，您不可以制作这些出版物的演绎作品，或者在您的企业外部复制、分发或展示这些出版物或其中的任何部分。

权利：除非本许可权中明确授予，否则不得授予对这些出版物或其中包含的任何信息、数据、软件或其他知识产权的任何许可权、许可证或权利，无论是明示的还是暗含的。

IBM 保留根据自身的判断，认为对出版物的使用损害了 IBM 的权益（由 IBM 自身确定）或未正确遵循以上指示信息时，撤回此处所授予权限的权利。

只有您完全遵循所有适用的法律和法规，包括所有的美国出口法律和法规，您才可以下载、出口或再出口该信息。

IBM 对这些出版物的内容不作任何保证。这些出版物“按现状”提供，不附有任何种类的（无论是明示的还是暗含的）保证，包括但不限于暗含的关于适销和适用于某种特定用途的保证。

附录 B. 声明

本信息是为在美国提供的产品和服务编写的。有关非 IBM 产品的信息是基于首次出版此文档时的可获信息且会随时更新。

IBM 可能在其他国家或地区不提供本文档中讨论的产品、服务或功能特性。有关您当前所在区域的产品和服务的信息，请向您当地的 IBM 代表咨询。任何对 IBM 产品、程序或服务的引用并非意在明示或暗示只能使用 IBM 的产品、程序或服务。只要不侵犯 IBM 的知识产权，任何同等功能的产品、程序或服务，都可以代替 IBM 产品、程序或服务。但是，评估和验证任何非 IBM 产品、程序或服务，则由用户自行负责。

IBM 公司可能已拥有或正在申请与本文档内容有关的各项专利。提供本文档并未授予用户使用这些专利的任何许可。您可以用书面方式将许可查询寄往:

IBM Director of Licensing
IBM Corporation
North Castle Drive
Armonk, NY 10504-1785
U.S.A.

有关双字节字符集 (DBCS) 信息的许可查询，请与您所在国家或地区的 IBM 知识产权部门联系，或用书面方式将查询寄往:

Intellectual Property Licensing
Legal and Intellectual Property Law
IBM Japan, Ltd.
1623-14, Shimotsuruma, Yamato-shi
Kanagawa 242-8502 Japan

本条款不适用英国或任何这样的条款与当地法律不一致的国家或地区: International Business Machines Corporation“按现状”提供本出版物，不附有任何种类的（无论是明示的还是暗含的）保证，包括但不限于暗含的有关非侵权、适销和适用于某种特定用途的保证。某些国家或地区在某些交易中不允许免除明示或暗含的保证。因此本条款可能不适用于您。

本信息中可能包含技术方面不够准确的地方或印刷错误。此处的信息将定期更改；这些更改将编入本资料的新版本中。IBM 可以随时对本资料中描述的产品和/或程序进行改进和/或更改，而不另行通知。

本信息中对非 IBM Web 站点的任何引用都只是为了方便起见才提供的，不以任何方式充当对那些 Web 站点的保证。那些 Web 站点中的资料不是此 IBM 产品资料的一部分，使用那些 Web 站点带来的风险将由您自行承担。

IBM 可以按它认为适当的任何方式使用或分发您所提供的任何信息而无须对您承担任何责任。

本程序的被许可方如果要了解有关程序的信息以达到如下目的: (i) 允许在独立创建的程序和其他程序（包括本程序）之间进行信息交换，以及 (ii) 允许对已经交换的信息进行相互使用，请与下列地址联系:

IBM Canada Limited
U59/3600
3600 Steeles Avenue East
Markham, Ontario L3R 9Z7
CANADA

只要遵守适当的条款和条件，包括某些情形下的一定数量的付费，都可获得这方面的信息。

本资料中描述的许可程序及其所有可用的许可资料均由 IBM 依据 IBM 客户协议、IBM 国际软件许可协议或任何同等协议中的条款提供。

此处包含的任何性能数据都是在受控环境中测得的。因此，在其他操作环境中获得的数据可能会有明显的不同。有些测量可能是在开发级的系统上进行的，因此不保证与一般可用系统上进行的测量结果相同。此外，有些测量是通过推算而估计的，实际结果可能会有差异。本文档的用户应当验证其特定环境的适用数据。

涉及非 IBM 产品的信息可从这些产品的供应商、其出版说明或其他可公开获得的资料中获取。IBM 没有对这些产品进行测试，也无法确认其性能的精确性、兼容性或任何其他关于非 IBM 产品的声明。有关非 IBM 产品性能的问题应当向这些产品的供应商提出。

所有关于 IBM 未来方向或意向的声明都可随时更改或收回，而不另行通知，它们仅仅表示了目标和意愿而已。

本信息可能包含在日常业务操作中使用的数据和报告的示例。为了尽可能完整地说明这些示例，示例中可能会包括个人、公司、品牌和产品的名称。所有这些名称都是虚构的，与实际商业企业所用的名称和地址的任何雷同纯属巧合。

版权许可:

本信息包括源语言形式的样本应用程序，这些样本说明不同操作平台上的编程方法。如果是为按照在编写样本程序的操作平台上的应用程序编程接口 (API) 进行应用程序的开发、使用、经销或分发，您可以任何形式对这些样本程序进行复制、修改、分发，而无须向 IBM 付费。这些示例并未在所有条件下作全面测试。因此，IBM 不能担保或暗示这些程序的可靠性、可维护性或功能。此样本程序“按现状”提供，且不附有任何种类的保证。对于使用此样本程序所引起的任何损坏，IBM 将不承担责任。

凡这些样本程序的每份拷贝或其任何部分或任何衍生产品，都必须包括如下版权声明:

©（*贵公司的名称*）（*年份*）. 此部分代码是根据 IBM 公司的样本程序衍生出来的。© Copyright IBM Corp. *（输入年份）*. All rights reserved.

商标

IBM, the IBM logo, and ibm.com are trademarks or registered trademarks of International Business Machines Corp., registered in many jurisdictions worldwide. Other prod-

uct and service names might be trademarks of IBM or other companies. A current list of IBM trademarks is available on the web at 『Copyright and trademark information』 at www.ibm.com/legal/copytrade.shtml.

The following terms are trademarks or registered trademarks of other companies

- Linux is a registered trademark of Linus Torvalds in the United States, other countries, or both.
- Java and all Java-based trademarks and logos are trademarks or registered trademarks of Oracle, its affiliates, or both.
- UNIX is a registered trademark of The Open Group in the United States and other countries.
- Intel, Intel logo, Intel Inside, Intel Inside logo, Celeron, Intel SpeedStep, Itanium, and Pentium are trademarks or registered trademarks of Intel Corporation or its subsidiaries in the United States and other countries.
- Microsoft, Windows, Windows NT, and the Windows logo are trademarks of Microsoft Corporation in the United States, other countries, or both.

Other company, product, or service names may be trademarks or service marks of others.

索引

[B]

[C]

[D]

[F]

[G]

[H]

[J]

[K]

[L]

[M]

[N]

[P]

[Q]

[R]

[S]

[T]

[W]

[X]

[Y]

[Z]

A

C

D

E

F

G

H

I

J

L

M

N

O

P

R

S

T

U

V

W

X

[特别字符]

Printed in China

S151-1759-00

Spine information:

IBM DB2 10.1 for Linux, UNIX, and Windows

数据库监视指南和参考