Today

Session 61: DB2 for IBM i Services
Starting at 14:00 UK time with Scott Forstie
90 min session

IBM Meetings → Put questions into the Chat box
or Toll Free phone for better audio
- 0800 279 6716 = UK Toll Free
- 020 7660 2093 = UK but you pay for the call
- Then 93505484# Participant Code
- Other countries ibm.biz/globalaccessnumbers

Previous Sessions:
IBM i PDI
Top Tech Tips
LPM/SRR Automation Tool
PowerVC 1.3.1 update
IBM i Mobile Access
Application Runtime Expert
Hands On with AIX7.2
Accessing IBM i Now.Future
PowerVC New Features
Boost IBM i perf with Flash
Simplified Remote Restart
Linux on POWER Field Exp
And more.....

Future Sessions:
8th Feb: HMC V8R860 - recent, important updates and changes
15th Mar: IBM i Predictive Performance Management
19th Apr: IBM i Virtualization Performance
17th May: Relax and Recover (the opensource mksysb for Linux on Power)

Webinar wiki: http://tinyurl.com/PowerSystemsTechnicalWebinars
Youtube Channel: http://tinyurl.com/IBMPowerVUGYoutubeChannel
Twitter: Scott Forstie @Forstie_IBMi
Jyoti Dodhia @JyotiDodhia
Therese Eaton @tetweetings
IBM Power Systems technical webinar series

© 2017 IBM Corporation

Automating the role of the Database Engineer (DBE)

Scott Forstie
DB2 for i Business Architect
forstie@us.ibm.com
@Forstie_IBMi

© 2017 IBM Corporation
Database Engineer (DBE)

Challenge:
• DB2 for i is easy to use and also easy to neglect
• You should have a Database Engineer (DBE) on staff

Benefit:
• Pain can be detected and avoided
• Performance can be optimized
• Avoid solving performance issues with cores and upgrades

How:
• Job description: http://db2fori.blogspot.com/2012/11/db2-for-i-database-engineer-description.html
• How to become one: http://db2fori.blogspot.com/2012/12/how-to-become-ibm-i-database-engineer.html
• Also...have this person introduced to me

Skillset

“SQL is not magic”

“SQL uses too many resources”

“SQL is magic”
DBA/DBE – Topic areas

What can we hope to automate?

- Performance oriented indexes
- Database health checking
- Regular capture of database query detail
- Enforcement of business guidelines
- Maintaining a central hub for DBE meta-data
- Find the highest use SQL statements & programs
- Find the lowest use indexes
- And more…

Will the results be perfect, complete now and forever?

No… that’s another reason why you need a DBE

DB2 Web Query and IBM i Services

- DB2 Web Query can jump-start your consumption of IBM i Services
- Built-in reports, dashboards and stored procedures
- Enables mobile, graphical, modern system administration
- If you’d like some help with this, contact qu2@us.ibm.com

Follow DB2 Web Query:

- @mckdmoly
- db2webqueryi@blogspot.com
- ibm.co/db2wqwiki

No need to start from scratch
DBA/DBE – Setup

The DBE needs to establish processes and procedures for the database detail which is captured and archived.

Topics:
1. **Organization**
   - Why? Statistical and definitional detail should be easily consumed
   - Suggestion: Create and use a standard set of libraries
2. **Cadence**
   - Why? By establishing a cadence for collection of detail, the DBE will be well positioned to recognize and understand anomalies
   - Suggestion: Start with weekly collections and revise as needed
3. **Security**
   - Why? DBE detail most likely contains sensitive information and should be governed
   - Suggestion: Remove *PUBLIC access to the DBE libraries, leverage the function usage ID for Database administration, and leverage a group profile for DBE membership
4. **And more…**

Also: optionally dedicate a library to generated DDL source
**DBA/DBE – Setup**

**How?**

--- Establish DBE authorization
CL: CRTUSRPRF USRPRF(DBEGROUP) PASSWORD(*NONE) SPCHAUS(*ALLOBJ);
CL: CHGFCNUSG FCNID(QIBM_DB_SQLADM) USER(DBEGROUP) USAGE(*ALLOWED);
CL: CHGUSRPRF USRPRF(FRANKDBA) GRPPRF(DBEGROUP);

--- Establish Database artifact libraries for the DBE
CL: CRTLIB DBESTUDY;
...

--- Establish group management and authorization
CL: CHGOBJOWN OBJ(DBESTUDY) OBJTYPE(*LIB) NEWOWN(DBEGROUP);
...

--- Establish database artifact libraries for the DBE
CL: CRTLIB DBESTUDY;
...

--- Establish group management and authorization
CL: CHGOBJOWN OBJ(DBESTUDY) OBJTYPE(*LIB) NEWOWN(DBEGROUP);
...
**SYSTOOLS.ACT_ON_INDEX_ADVICE – Procedure**

- Input parameters are criteria for finding actionable advice

1. IN P_LIBRARY CHAR(10)
2. IN P_FILE CHAR(10)
3. IN P_TIMES ADVISED BIGINT
4. IN P_MTI USED BIGINT
5. IN P_AVERAGE_QUERY_ESTIMATE INTEGER

-- If a Maintained Temporary Index (MTI) has been used
-- more than 1000 times, for any table create a permanent index

```
CALL SYSTOOLS.ACT_ON_INDEX_ADVICE('TOYSTORE',
                                  NULL,
                                  NULL,
                                  1000,
                                  NULL)
```

**SYSTOOLS.REMOVE_INDEXES – Procedure**

- Input parameters are criteria for detecting performance indexes that should be removed due to lack of use
- Only indexes created by the ACT_ON_INDEX_ADVICE or HARVEST_INDEX_ADVICE are examined

1. IN P_LIBRARY CHAR(10)
2. IN P_TIMES_USED BIGINT
3. IN P_INDEX_AGE VARCHAR(100)

-- Find indexes created by ACT_ON_INDEX_ADVICE
-- that are at least 7 days old.
-- For any index which has been used less than 500 times
-- by the Query engine, drop the index

```
CALL SYSTOOLS.REMOVE_INDEXES('TOYSTORE', 500, ' 7 days ')
```
SYSTOOLS.HARVEST_INDEX_ADVICE – Procedure

• Input parameters are criteria for finding actionable advice

1. IN P_LIBRARY CHAR(10)
2. IN P_FILE CHAR(10)
3. IN P_TIMES ADVISED BIGINT
4. IN P_MTI_USED BIGINT
5. IN P_AVERAGE_QUERY_ESTIMATE INTEGER
6. IN T_LIBRARY CHAR(10)
7. IN T_FILE CHAR(10)

-- If a Maintained Temporary Index (MTI) has been used
-- more than 1000 times, build CREATE INDEX statements
-- and place them in the GENSOURCE/IDXSRC file

CALL SYSTOOLS.HARVEST_INDEX_ADVICE(‘TOYSTORE’,
                          NULL,
                          NULL,
                          1000,
                          NULL,
                          GENSOURCE,
                          IDXSRC)

Automation Example

Create the SQL source:
CRTSRCPF QGPL/QSQLSRC
STRSEU SRCFILE(QGPL/QSQLSRC) SRCMBR(INDEXMAINT)

Add these lines:
CALL SYSTOOLS.ACT_ON_INDEX_ADVICE(NULL, NULL, NULL, 1000, NULL);
CALL SYSTOOLS.REMOVE_INDEXES(NULL, 500, ‘7 DAYS’);

Retrieve and modify the system startup program source code:
RTVCLSRC PGM(QSYS/QSTRUP) SRCFILE(QGPL/QCLSRC) SRCMBR(QSTRUP) TYPE(CLPI)
OPTION(2)

Immediately after the DONE: label, add the following three lines:
SBMJOB SCDDATE(*SAT) SCDTIME(040000) +
CMD(RUNSQLSTM SRCFILE(QGPL/QCLSRC) SRCMBR(INDEXMAINT) +
COMMIT(*NONE) NAMING(*SQL) OUTPUT(*PRINT))

Build and replace the system startup program:
CRTCLPGM PGM(QSYS/QSTRUP) SRCFILE(QGPL/QCLSRC)
REPLACE(*YES)
QSYS2.RESET_TABLE_INDEX_STATISTICS – Procedure

- Zeroes the QUERY_USE_COUNT and QUERY_STATISTICS_COUNT usage statistics for indexes over a specified table(s), without needing an exclusive lock
- The CHGOBJD command includes USECOUNT(*RESET), but requires an exclusive lock
- LAST_QUERY_USE, LAST_STATISTICS_USE, LAST_USE_DATE and NUMBER_DAYS_USED are not affected
- The same wild card characters ( _ and % ) allowed in the SQL LIKE predicate are supported.

```
-- Description: Reset indexes over the EMPLOYEE table
CALL QSYS2.RESET_TABLE_INDEX_STATISTICS('TOYSTORE', 'EMPLOYEE');

-- Description: Reset any table like TOYSTORE/5%
CALL QSYS2.RESET_TABLE_INDEX_STATISTICS('TOYSTORE','5%');
```
Archiving index statistics

- `QSYS2.RESET_TABLE_INDEX_STATISTICS()` processed index detail can be archived

```sql
-- Description: Review indexes that were reset
SELECT INDEX_NAME, INDEX_TYPE, LAST_STATISTICS_USE,
LAST_QUERY_USE, QUERY_USE_COUNT, QUERY_STATISTICS_COUNT
FROM SESSION.SQL_INDEXES_RESET;
```

<table>
<thead>
<tr>
<th>INDEX_NAME</th>
<th>INDEX_TYPE</th>
<th>LAST_STATISTICS_USE</th>
<th>LAST_QUERY_USE</th>
<th>QUERY_USE_COUNT</th>
<th>QUERY_STATISTICS_COUNT</th>
</tr>
</thead>
<tbody>
<tr>
<td>EMPLOYEE_900000000000000000000000</td>
<td>PRIMARY KEY</td>
<td>2016-04-26 06:39:40.0000000</td>
<td>0</td>
<td>0</td>
<td></td>
</tr>
<tr>
<td>TEST_EMPLOYEE_900000000000000000000000</td>
<td>PRIMARY KEY</td>
<td>2016-04-26 15:57:01.0000000</td>
<td>0</td>
<td>0</td>
<td></td>
</tr>
<tr>
<td>EMPLOYEE_900000000000000000000000</td>
<td>INDEX</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td>EMPLOYEE_900000000000000000000000</td>
<td>INDEX</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td>EMPLOYEE_900000000000000000000000</td>
<td>INDEX</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td>EMPLOYEE_900000000000000000000000</td>
<td>INDEX</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td>EMPLOYEE_900000000000000000000000</td>
<td>INDEX</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td>EMPLOYEE_900000000000000000000000</td>
<td>INDEX</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td>EMPLOYEE_900000000000000000000000</td>
<td>INDEX</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td>EMPLOYEE_900000000000000000000000</td>
<td>INDEX</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td>EMPLOYEE_900000000000000000000000</td>
<td>INDEX</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td>EMPLOYEE_900000000000000000000000</td>
<td>INDEX</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td>EMPLOYEE_900000000000000000000000</td>
<td>INDEX</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
</tr>
</tbody>
</table>

IBM® DB2® for i Services

- Health Center Procedures
- Performance Services
- Utility Procedures
- Plan Cache Procedures
- Application Services

Daily Plan Cache Snapshot

- A daily capture of the most expensive queries enables several DBE tasks
- Most expensive is defined as the longest elapsed time

```
-- Procedure: Daily Plan Cache snapper
-- Purpose: This procedure captures the 100 most expensive queries within a plan cache snapshot
-- Naming formula: "SNP<Julian date>"
-- The new snapshot is imported into the Navigator control table.
-- The procedure deletes the snapshot that is 60 days old, to prevent an endless accumulation of snapshots.
```

```
CREATE OR REPLACE PROCEDURE SNAPSHOTS.DAILY_PC()
BEGIN
    DECLARE OLDEST_SNAP_NAME CHAR(10);
    DECLARE SNAP_COMMENT VARCHAR(100);
    CALL QSYS2.IMPORT_PC_SNAPSHOT('SNAPSHOTS', SNAP_NAME, 'Top 100 Queries - ' || SNAP_COMMENT);
    CALL QSYS2.REMOVE_PC_SNAPSHOT('SNAPSHOTS', OLDEST_SNAP_NAME);
END;
```

Purpose: This procedure captures the 100 most expensive queries and imports them into the Navigator control table. It also deletes the snapshot that is 60 days old to prevent an endless accumulation of snapshots.
SYSTOOLS.CHECK_SYSROUTINE() procedure

- High Availability (HA) and Disaster Recovery (DR) work better when the database catalogs are identical, across Production, HA, and DR.
- DB2 for i Catalogs are not replicated objects
- Given the complex nature of keeping SQL and external procedure/function database catalog entries in sync, DB2 for i has provided a catalog assessment utility

```
-- Search for procedure and function differences
CALL SYSTOOLS.CHECK_SYSROUTINE(
  <target-database-name>,
  <schema-to-compare>,
  <optional-result-set-parameter>)
```
SYSTOOLS.CHECK_SYSCST() procedure

- Similar to the COMPARE_SYSROUTINE() procedure, but for Constraints
- Expectation is to receive an empty result set
- Not satisfied with the procedure? Extract the source and modify.

```
-- Search for constraint differences
CALL SYSTOOLS.CHECK_SYSCST (<target-database-name>,
                               <schema-to-compare>,
                               <optional-result-set-parameter>)
```

<table>
<thead>
<tr>
<th>SERVER_NAME</th>
<th>CONSTRAINT_SCHEMA</th>
<th>CONSTRAINT_NAME</th>
<th>CONSTRAINT_TYPE</th>
<th>CONSTRAINT_STATE</th>
<th>ENABL</th>
</tr>
</thead>
<tbody>
<tr>
<td>IPU18</td>
<td>CORP_DB</td>
<td></td>
<td>PRIMARY</td>
<td>ESTABLISHED</td>
<td>YES</td>
</tr>
<tr>
<td>IPU18</td>
<td>CORP_DB</td>
<td></td>
<td>CHECK</td>
<td>ESTABLISHED</td>
<td>NO</td>
</tr>
<tr>
<td>X1423</td>
<td>CORP_DB</td>
<td></td>
<td>CHECK</td>
<td>ESTABLISHED</td>
<td>YES</td>
</tr>
</tbody>
</table>

IBM© DB2© for i Catalogs

- **Catalogs**: SYSCATALOGS, INFORMATION_SCHEMA_CATALOG_NAME
- **Schemas**: SYSCOLAUTH, SYSCOLCONS
- **Tables Views Indexes**: SYSTABLES, SYSCOLINDEXES
- **Constraints**: SYSCONTROLS, SYSCONSTRAINTS
- **Privileges**: SQL_PRIVILEGES, SQL_TABLE_PRIVILEGES
- **Routines**: SQL_DEPENDENCIES, SQL_FUNCTIONS, SQL_PROCEDURES
- **Statistics**: SYSCOLINDEXES, SYSCOLINDEXSTAT

IBM Power Systems technical webinar series

© 2017 IBM Corporation
-- Capture database file detail
CREATE OR REPLACE TABLE DBESTUDY.STAR100G_TABLE_RUNTIME_DETAILS
  (TABLE_SCHEMA, TABLE_NAME, TABLE_PARTITION, PARTITION_TYPE, NUMBER_DELETED_ROWS, NUMBER_ROWS, DATA_SIZE, OVERFLOW, VARIABLE_LENGTH_SIZE, MAINTAINED_TEMPORARY_INDEX_SIZE, OPEN_OPERATIONS, CLOSE_OPERATIONS, INSERT_OPERATIONS, UPDATE_OPERATIONS, DELETE_OPERATIONS, PHYSICAL_READS, SEQUENTIAL_READS, RANDOM_READS, KEEP_IN_MEMORY, MEDIA_PREFERENCE, CAPTURE_TIME)
AS (SELECT TABLE_SCHEMA, TABLE_NAME, TABLE_PARTITION, PARTITION_TYPE, NUMBER_DELETED_ROWS, NUMBER_ROWS, DATA_SIZE, OVERFLOW, VARIABLE_LENGTH_SIZE, MAINTAINED_TEMPORARY_INDEX_SIZE, OPEN_OPERATIONS, CLOSE_OPERATIONS, INSERT_OPERATIONS, UPDATE_OPERATIONS, DELETE_OPERATIONS, PHYSICAL_READS, SEQUENTIAL_READS, RANDOM_READS, VARCHAR(CASE KEEP_IN_MEMORY WHEN '1' THEN 'YES' ELSE 'NO' END, DEFAULT, 37), VARCHAR(CASE MEDIA_PREFERENCE WHEN 255 THEN 'SSD' ELSE 'ANY' END, DEFAULT, 37), CURRENT_TIMESTAMP FROM QSYS2.SYSPARTITIONSTAT
WHERE TABLE_SCHEMA = 'PRODLIB') WITH DATA ON REPLACE DELETE ROWS;

-- Identify candidates for physical file reorganization
SELECT TABLE_SCHEMA, TABLE_NAME, NUMBER_ROWS AS VALID_ROWS, NUMBER_DELETED_ROWS AS DELETED_ROWS, DATA_SIZE AS DATA_SPACE_SIZE_IN_BYTES, DEC(DEC(NUMBER_DELETED_ROWS, 19, 2) * 100, 19, 2) AS DELETED ROW PERCENTAGE FROM DBESTUDY.STAR100G_TABLE_RUNTIME_DETAILS A WHERE NUMBER_DELETED_ROWS > 100000 ORDER BY DELETED_ROW_PERCENTAGE DESC;
QSYS2.SYSPARTITIONSTAT - View

-- Review variable length column efficiency

SELECT *
FROM DBESTUDY.STAR100G_TABLE_RUNTIME_DETAILS
ORDER BY VARIABLE_LENGTH_SIZE DESC;

Capture database index detail

-- Capture database index detail

CREATE OR REPLACE TABLE DBESTUDY.STAR100G_INDEX_RUNTIME_DETAILS
(INDEX_SCHEMA, INDEX_NAME, INDEX_MEMBER, INDEX_TYPE, TABLE_SCHEMA, TABLE_NAME, TABLE_PARTITION, PARTITION_TYPE, LAST_QUERY_USE, LAST_STATISTICS_USE, INDEX_VALID, INDEX_SIZE, ESTIMATED_BUILD_TIME, LAST_BUILD_TIME, LAST_BUILD_DEGREE, SPARSE, DERIVED_KEY, PARTITIONED, ACCPTH_TYPE, INDEX_HELD, PHYSICAL_READS, SEQUENTIAL_READS, RANDOM_READS, KEEP_IN_MEMORY, MEDIA_PREFERENCE, CAPTURE_TIME) AS
(SELECT INDEX_SCHEMA, INDEX_NAME, INDEX_MEMBER, INDEX_TYPE, TABLE_SCHEMA, TABLE_NAME, TABLE_PARTITION, PARTITION_TYPE, LAST_QUERY_USE, LAST_STATISTICS_USE, INDEX_VALID, INDEX_SIZE, ESTIMATED_BUILD_TIME, LAST_BUILD_TIME, LAST_BUILD_DEGREE, SPARSE, DERIVED_KEY, PARTITIONED, ACCPTH_TYPE, INDEX_HELD, PHYSICAL_READS, SEQUENTIAL_READS, RANDOM_READS, KEEP_IN_MEMORY, MEDIA_PREFERENCE, CURRENT_TIMESTAMP)
FROM QSYS2.SYSPARTITIONINDEXSTAT
WHERE TABLE_SCHEMA = 'STAR100G'
WITH DATA ON REPLACE DELETE ROWS;
QSYS2.SYSPARTITIONINDEXSTAT - View

**Identify candidates for SSD**

```sql
SELECT *
FROM DBESTUDY.STAR100G_TABLE_RUNTIME_DETAILS
ORDER BY VARIABLE_LENGTH_SIZE DESC;
```

QSYS2.SYSPARTITIONINDEXDISK - View

- Another good friend to the DBE
- Study storage use

**Contrast Solid State Drives (SSD) vs Spinning Disk**

```sql
CREATE OR REPLACE TABLE DBESTUDY.STAR100G_INDEX_STORAGE

(Index_Name, Index_Member, Index_Type, SSD_Space, NonSSD_Space, Collection_time)

AS (SELECT INDEX_NAME, INDEX_MEMBER, INDEX_TYPE, SUM(CASE UNIT_TYPE WHEN 1 THEN UNIT_SPACE_USED ELSE 0 END) AS SSD_SPACE, SUM(CASE UNIT_TYPE WHEN 0 THEN UNIT_SPACE_USED ELSE 0 END) AS NONSSD_SPACE, CURRENT_TIMESTAMP)

FROM QSYS2.SYSPARTITIONINDEXDISK

WHERE INDEX_SCHEMA = 'STAR100G'

GROUP BY INDEX_NAME, INDEX_MEMBER, INDEX_TYPE, TABLE_SCHEMA, TABLE_NAME, TABLE_PARTITION

WITH DATA ON REPLACE DELETE ROWS;
```
Alert when a DB file is growing very large

CL: ALCOBJ OBJ((QSYS2/SYSLIMTBL *FILE *EXCL)) CONFLICT(*RQSRLS)
CL: DLCOBJ OBJ((QSYS2/SYSLIMTBL *FILE *EXCL));

CREATE OR REPLACE TRIGGER MYLIB.SYSTEM_LIMITS_LARGE_FILE
AFTER INSERT ON QSYS2.SYSLIMTBL
REFERENCING NEW AS N FOR EACH ROW MODE DB2ROW
SET OPTION USRPRF=*OWNER, DYNUSRPRF=*OWNER
BEGIN ATOMIC
DECLARE V_CMDSTMT VARCHAR(200);
DECLARE ERROR INTEGER;
DECLARE EXIT HANDLER FOR SQLEXCEPTION SET ERROR = 1;
/* If a table is nearing the maximum size, alert the operator */
IF (N.LIMIT_ID = 15000 AND N.CURRENT_VALUE > 3000000000) THEN
SET V_CMDSTMT = 'SNDMSG MSG('Table: '||CONCAT N.SYSTEM_SCHEMA_NAME CONCAT '/'||CONCAT N.SYSTEM_OBJECT_NAME CONCAT ' ) IS GETTING VERY LARGE - ROW COUNT = '||CONCAT CURRENT_VALUE CONCAT ' ) TOUSR(*SYSOPR) MSGTYPE(*INFO) ';
CALL QSYS2.QCMDEXC(V_CMDSTMT);
END IF;
END;
Find the largest IFS Stream Files

SELECT LASTCHG, JOB_NAME, ASP_NUMBER, IFS_PATH_NAME, USER_NAME, CURRENT_VALUE FROM QSYS2.SYSLIMITS WHERE LIMIT_ID = 18409 ORDER BY CURRENT_VALUE DESC;

DB2 for IBM i Lab Services

- Facilitated workshops covering current state, requirements, future state, possible solutions, implementation best practices, and formulation of a strategic roadmap:
  - RCAC
  - Temporal Tables

- Customized consulting workshops
  - Advanced SQL and Data-centric Programming
  - SQL Performance Best Practices, Monitoring and Tuning
  - Remote performance assessments

- Consulting on any DB2 for i topic
- For more information, contact mcain@us.ibm.com
QSYS2.HEALTH_DATABASE_OVERVIEW – procedure

- Service used by System i Navigator for Database -> Health Center -> Overview
- The QSYS2.Health_Database_Overview() procedure returns counts of all the different types of DB2 for i objects within the target schema or schemas. The counts are broken down by object type and subtype.
- ‘%’ is used to wildcard the schema name
- A single row result set is returned for all matching schema names

```sql
-- Retrieve the overview for the entire database
CALL QSYS2.Health_Database_Overview(1, '%', NULL, NULL, NULL);
```

QSYS2.HEALTH_DESIGN_LIMITS – procedure

- The QSYS2.Health_Design_Limits() procedure returns detailed counts of design limits over a set of objects within one or more schemas. Design limits correspond to architectural constructs.
- The Database Health Center Design limits include:
  - MAXIMUM NUMBER OF MEMBERS
  - MAXIMUM NUMBER OF RECORD FORMATS
  - MAXIMUM JOURNAL RECEIVER SIZE
  - TOTAL SQL STATEMENTS
  - TOTAL ACTIVE SQL STATEMENTS
  - MAXIMUM SQL PACKAGE SIZE
  - MAXIMUM LARGE SQL PACKAGE SIZE
  - MAXIMUM SQL PROGRAM ASSOCIATED SPACE SIZE

```sql
-- Retrieve the overview for the entire database
CALL QSYS2.Health_Design_Limits(1, 0, 'PRODLIB1', '%', 20, NULL, NULL, NULL);
```
The QSYS2.Health_Size_Limits () procedure returns detailed size information for database objects within one or more schemas. Size limits help you understand trends towards reaching a database limit.

The Database Health Center Design limits include:

- MAXIMUM NUMBER OF ALL ROWS
- MAXIMUM NUMBER OF VALID ROWS
- MAXIMUM ROW LENGTH
- MAXIMUM ROW LENGTH WITH LOBS
- MAXIMUM NUMBER OF PARTITIONS
- MAXIMUM NUMBER OF REFERENCED TABLES
- MAXIMUM LENGTH OF CHECK CONSTRAINT
- MAXIMUM *MAX4GB INDEX SIZE
- MAXIMUM *MAX1TB INDEX SIZE
- MAXIMUM NUMBER OF INDEX ENTRIES
- MAXIMUM KEY COLUMNS
- MAXIMUM KEY LENGTH
- MAXIMUM NUMBER OF PARTITIONING KEYS
- MAXIMUM NUMBER OF FUNCTION PARAMETERS
- MAXIMUM NUMBER OF PROCEDURE PARAMETERS

```
-- Retrieve the size limits for TOYSTORE/S* objects
CALL QSYS2.Health_Size_Limits(1, 0, TOYSTORE', 'S%', 5, NULL, NULL, NULL);
```

The QSYS2.Health_Activity () procedure returns summary counts of database and SQL operations over a set of objects within one or more schemas.

The Database Health Center Activity counts include:

- INSERT OPERATIONS
- UPDATE OPERATIONS
- DELETE OPERATIONS
- LOGICAL READS
- PHYSICAL READS
- CLEAR OPERATIONS
- INDEX Builds/Rebuilds
- DATA SPACE REORGANIZE OPERATIONS
- DATA SPACE COPY OPERATIONS
- FULL OPENS
- FULL CLOSES
- DAYS USED
- INDEX QUERY USE
- INDEX QUERY STATISTICS USE
- RANDOM READS
- INDEX LOGICAL READS
- INDEX RANDOM READS
- SQL STATEMENT COMPRESSION COUNT
- SQL STATEMENT CONTENTION COUNT
- RANDOM READS
- SEQUENTIAL READS
- RANDOM READS

```
-- Retrieve the size limits for TOYSTORE/S* objects
CALL QSYS2.Health_Activity(1, 0, TOYSTORE', '%', 10, NULL, NULL, NULL);
```
The QSYS2.Health_Environmental_Limits() procedure returns detail on the top 10 jobs on the system, for different SQL or application limits. The jobs do not have to be in existence. The top 10 information is maintained within DB2® for i and gets reset when the machine is IPLed, the IASP is varied ON, or when the QSYS2.Reset_Environmental_Limits() procedure is called.

The Database Health Center Environmental limits include:
- MAXIMUM NUMBER OF LOB AND XML LOCATORS PER JOB
- MAXIMUM NUMBER OF LOB AND XML LOCATORS PER SERVER JOB
- MAXIMUM NUMBER OF ACTIVATION GROUPS
- MAXIMUM NUMBER OF DESCRIPTORS
- MAXIMUM NUMBER OF CLI HANDLES
- MAXIMUM NUMBER OF SQL OPEN CURSORS
- MAXIMUM NUMBER OF SQL PSEUDO OPEN CURSORS
- MAXIMUM LENGTH OF SQL STATEMENT

```
-- Retrieve the size limits for TOYSTORE/S* objects
CALL QSYS2.Health_Environmental_Limits(1, 0, NULL, NULL);
```

The QSYS2.Reset_Environmental_Limits() procedure clears out the environment limit cache for the database. If IASPs are being used, this procedure clears the environment limit cache for the IASP within which it is called.

Consider calling this procedure if you are iterating through applications changes based upon Health_Environmental_Limits data.

```
-- Reset the top 10
CALL QSYS2.Reset_Environmental_Limits;
```
Automating performance index creation and removal
Real example of automating performance indexes... for Infor clients

“Scott:
I have developed a tool to automate (to a certain extent) the build of indexes and have beta tested the process at a single customer. The front end seen below interfaces with SYSTOOLS capabilities on the backend to run scheduled jobs.”

Special notices

This document was developed for IBM offerings in the United States as of the date of publication. IBM may not make these offerings available in other countries, and the information is subject to change without notice. Contact your local IBM business contact for information on the IBM offerings available in your area.

Information in this document concerning non-IBM products was obtained from the suppliers of these products or other public sources. Questions on the capabilities of non-IBM products should be addressed to the suppliers of those products.

IBM may have patents or pending patent applications covering subject matter in this document. The furnishing of this document does not give you any license to these patents. Send license inquiries, in writing, to IBM Director of Licensing, IBM Corporation, New Castle Drive, Armonk, NY 10504-1785 USA.

All statements regarding IBM future direction and intent are subject to change or withdrawal without notice, and represent goals and objectives only. The information contained in this document has not been submitted to any formal IBM test and is provided “AS IS” with no warranties or guarantees either expressed or implied. All examples cited or described in this document are presented as illustrations of the manner in which some IBM products can be used and the results that may be achieved. Actual environmental costs and performance characteristics will vary depending on individual client configurations and conditions.

IBM Global Financing offerings are provided through IBM Credit Corporation in the United States and other IBM subsidiaries and divisions worldwide to qualified commercial and government clients. Rates are based on a client’s credit rating, financing terms, offering type, equipment type and options, and may vary by country. Other restrictions may apply. Rates and offerings are subject to change, extension or withdrawal without notice.

IBM is not responsible for printing errors in this document that result in pricing or information inaccuracies. All prices shown are IBM’s United States suggested list prices and are subject to change without notice; reseller prices may vary. IBM hardware products are manufactured from new parts, or new and serviceable used parts. Regardless, our warranty terms apply. Any performance data contained in this document was determined in a controlled environment. Actual results may vary significantly and are dependent on many factors including system hardware configuration and software design and configuration. Some measurements quoted in this document may have been made on development-level systems. There is no guarantee these measurements will be the same on generally available systems. Some measurements quoted in this document may have been estimated through extrapolation. Users of this document should verify the applicable data for their specific environment.