IBM solidDB
Version 7.0

IBM solidDB Universal Cache User Guide

SC27-3847-05

Note
Before using this information and the product it supports, read the information in "Notices" on page 187.

First edition, fifth revision

This edition applies to V7.0 Fix Pack 8 of IBM solidDB (product number 5724-V17) and to all subsequent releases and modifications until otherwise indicated in new editions.

Contents

Figures

Tables

Summary of changes

Changes for revision 05

- Section System requirements updated: support for Solaris 11 introduced in Fix Pack 6.

Changes for revision 04

- Added information about troubleshooting bidirectional subscriptions when using DB2® as the backend. See section Troubleshooting.

Changes for revision 03

- Information about supported Informix® editions added in section Supported backend data servers for Universal Cache.

  The following Informix editions are supported:

  – Informix Developer Edition
  – Informix Ultimate Edition
  – Informix Ultimate Warehouse Edition
- Section System requirements updated: support for Windows 8 and Windows Server 2012 introduced in Fix Pack 4.

Changes for revision 02

- New parameter for controlling throttling (**LogReader.UseThrottling**) added in section Log Reader parameters.
- Information about configuring the InfoSphere® CDC for solidDB® instances when using operating-system-based external authentication mechanism for solidDB added in section Configuring InfoSphere CDC.

Changes for revision 01

- New parameters added in section System parameters for InfoSphere CDC for solidDB:
  – **convert_not_nullable_column**
  – **differential_refresh_commit_after_max_operations**
  – **events_max_retain**
  – **fastload_refresh_commit_after_max_operation**
  – **global_max_batch_size**
  – **implicit_transformation_warning**
  – **jdbc_refresh_commit_after_max_operations**
  – **mirror_auto_restart_interval_minutes**
  – **mirror_global_disk_quota_gb**
  – **mirror_interim_commit_threshold**
  – **staging_store_disk_quota_gb**
  – **staging_store_can_run_independently**
  – **userexit_max_lob_size_kb**
- Syntax descriptions updated in section Commands for InfoSphere CDC. The following new commands have been added:
  – **dmclearstagingstore**

  - **dmdisablecontinuouscapture**
  - **dmenablecontinuouscapture**
  - **dmgetstagingstorestatus**
- Section System requirements updated: support for System z® Linux introduced in Fix Pack 1.

About this manual

IBM® solidDB Universal Cache is a solution for speeding up traditional disk-based SQL data servers by way of one or more solidDB in-memory database instances caching the data traffic between the applications and the data servers. IBM InfoSphere Change Data Capture technology is used to implement the data replication between the solidDB and data server instances.

This guide provides an overview of the solidDB Universal Cache as well as instructions for installing and configuring the solidDB Universal Cache. Guidelines for handling failure and troubleshooting scenarios are also included. The chapter CDC for solidDB contains detailed instructions for how to install and configure the InfoSphere CDC for solidDB. This section is needed when configuring the solidDB Universal Cache; it provides comparable information to the *InfoSphere Change Data Capture, End-User Documentation* user manual for your backend data server.

This manual assumes that the reader has general database management system (DBMS) knowledge and familiarity with SQL and solidDB.

Typographic conventions

solidDB documentation uses the following typographic conventions:

Table 1. Typographic conventions

Format	Used for
Database table	This font is used for all ordinary text.
NOT NULL	Uppercase letters on this font indicate SQL keywords and macro names.
`solid.ini`	These fonts indicate file names and path expressions.
`SET SYNC MASTER YES;` `COMMIT WORK;`	This font is used for program code and program output. Example SQL statements also use this font.
`run.sh`	This font is used for sample command lines.
TRIG_COUNT()	This font is used for function names.
java.sql.Connection	This font is used for interface names.
`LockHashSize`	This font is used for parameter names, function arguments, and Windows registry entries.
argument	Words emphasized like this indicate information that the user or the application must provide.
Administrator Guide	This style is used for references to other documents, or chapters in the same document. New terms and emphasized issues are also written like this.

Table 1. Typographic conventions (continued)

Format	Used for
File path presentation	Unless otherwise indicated, file paths are presented in the UNIX format. The slash (/) character represents the installation root directory.
Operating systems	If documentation contains differences between operating systems, the UNIX format is mentioned first. The Microsoft Windows format is mentioned in parentheses after the UNIX format. Other operating systems are separately mentioned. There may also be different chapters for different operating systems.

Syntax notation conventions

solidDB documentation uses the following syntax notation conventions:

Table 2. Syntax notation conventions

Format	Used for
`INSERT INTO` *`table_name`*	Syntax descriptions are on this font. Replaceable sections are on *this* font.
`solid.ini`	This font indicates file names and path expressions.
[ ]	Square brackets indicate optional items; if in bold text, brackets must be included in the syntax.
|	A vertical bar separates two mutually exclusive choices in a syntax line.
{ }	Curly brackets delimit a set of mutually exclusive choices in a syntax line; if in bold text, braces must be included in the syntax.
...	An ellipsis indicates that arguments can be repeated several times.
. . .	A column of three dots indicates continuation of previous lines of code.

1 Universal Cache installation and configuration overview

The installation and configuration procedure depends on the type of topology you want to use, the software and hardware platforms you want to install on, and the backend data server you want to cache data from.

About this task

Installation

In most cases, the complete installation includes the following components:

- Cache
  - solidDB server
  - InfoSphere CDC for solidDB replication engine
- Database
  - Backend data server - prerequisite
  - InfoSphere CDC for backend replication engine
- InfoSphere CDC Access Server
- InfoSphere CDC Management Console
- Drivers
  - solidDB ODBC Driver or solidDB JDBC Driver
  - Backend ODBC driver for SQL passthrough

You can install these components in several different configurations, as described in 1.2, “Installation topologies for Universal Cache,” on page 22.

Setting up caching

Typically you would already have a working installation of the backend data server that contains the data you want to cache into an in-memory solidDB database. Setting up caching then includes defining the connection between the cache database and the backend database, defining which tables to cache, populating the cache database, and finally activating the cache.

Preparing applications for use with Universal Cache

From the application perspective, the backend database connection needs to be replaced or supplemented with a connection to the Universal Cache environment. The SQL passthrough functionality enables applications to access data both in the frontend and backend data servers with a single connection. To use SQL passthrough, you need to install and configure a backend compatible ODBC driver on the solidDB cache node.

Procedure

The installation and configuration of a Universal Cache system includes the following high-level steps:

1. Select an installation topology and select the servers on which to install the different components.
2. Download and extract the installation files to the appropriate computers.
3. Install the Universal Cache components and complete the initial configuration steps.

- The Universal Cache components are installed using the installation programs of each component.
- The solidDB drivers are installed as part of the solidDB server installation. If your application is located on a different computer than the solidDB server, you need to install the drivers on the computer where the application is located.
- You need to perform the following configuration steps:

a. Create a solidDB database.
b. Configure an InfoSphere CDC for solidDB instance.
c. Configure an InfoSphere CDC for backend instance.

4. Optional: Install and configure backend ODBC driver for SQL passthrough.
5. Set up caching of data between your backend data server and solidDB using InfoSphere CDC Management Console. During the setup, you can populate the cache tables with data from the backend tables, or vice versa.
6. Prepare your application for use with Universal Cache.
7. Activate Universal Cache by starting replication between the cache and the backend database.

1.1 Prerequisites

Before installing and configuring Universal Cache, ensure that you have access to the required tooling and development environments and that you meet all of the software, hardware, and operating system prerequisites.

1.1.1 System requirements for Universal Cache

The solidDB product family supports more than 30 different platforms, each understood as a combination of hardware type and operating system. Typically all commonly used platforms are supported. Support for legacy platforms might be available upon request.

IBM solidDB supported platforms

The following table shows an overview of the supported platforms for the components included in the IBM solidDB 7.0 product offering.

More detailed information about the platform support for each component is available through the Software product compatibility reports portal on ibm.com® (see direct links after the table).

Table 3. IBM solidDB supported platforms

Operating system		Hardware	solidDB server 7.0	InfoSphere CDC 6.5											ODBC
				InfoSphere CDC solidDB 7.0	MC	AS	DB2	DB2 z/OS®	DB2 iSeries®	IDS	OR	OT	MS SQL	Sybase	
AIX®	AIX 7.1 AIX 6.1	64-bit systems with POWER5, POWER6®, or POWER7®	X	X		X	X			X	X			X	X
HP-UX	HP-UX 11i v3	Itanium-based HP Integrity Series systems	X	X		X				X	X			X	X

Table 3. IBM solidDB supported platforms (continued)

Operating system		Hardware	solidDB server 7.0	InfoSphere CDC 6.5											ODBC
Linux	Red Hat Enterprise Linux (RHEL) 6, 5 SUSE Linux Enterprise Server (SLES) 11, 10	32-bit and 64-bit systems based on Intel or AMD processors that are capable of running the supported Linux operating systems (x86 and x64 systems)	X	X		X	X			X	X			X	X
	Red Hat Enterprise Linux (RHEL) 5 System z SUSE Linux Enterprise Server (SLES) 10 System z	System z	X[1]				X					X			X
Solaris	Solaris 11	64-bit systems with UltraSPARC processors	X[3]												
		64-bit systems with x86 processors	X[3]												
	Solaris 10	64-bit systems with UltraSPARC processors	X	X		X	X			X	X			X	X
		64-bit systems with x86 processors	X												X
Windows	Windows Server 2012 (Standard Server, Enterprise Server, and Datacenter Editions) Windows 8 (Professional, Enterprise, and Ultimate editions)	32-bit and 64-bit systems based on Intel or AMD processors that are capable of running the supported Windows operating systems (x86 and x64 systems)	X[2]												
	Windows Server 2008 R2, 2008 (Standard Server, Enterprise Server, and Datacenter Editions) Windows 7 (Professional, Enterprise, and Ultimate editions) Windows Vista (Business, Enterprise, and Ultimate editions)		X	X	X	X	X			X			X	X	X
IBM i	i5/OS™ 7.1 i5/OS 6.1 i5/OS 5.4 i5/OS 5.3	POWER® System with i5 processors							X						X
z/OS	z/OS V1.11 z/OS V1.10	System z						X							X

Table 3. IBM solidDB supported platforms (continued)

Operating system	Hardware	solidDB server 7.0	InfoSphere CDC 6.5	ODBC
MC = InfoSphere Change Data Capture Management Console 6.5				
AS = InfoSphere Change Data Capture Access Server 6.5				
DB2 = InfoSphere Change Data Capture DB2 Linux, UNIX, and Windows 6.5				
DB2 z/OS = InfoSphere Change Data Capture DB2 z/OS 6.5				
DB2 iSeries = InfoSphere Change Data Capture DB2 iSeries 6.1				
IDS = InfoSphere Change Data Capture Informix 6.5				
OR = InfoSphere Change Data Capture Oracle Redo 6.5				
OT = InfoSphere Change Data Capture Oracle Trigger 6.5				
MS SQL = InfoSphere Change Data Capture Microsoft SQL Server 6.5				
Sybase = InfoSphere Change Data Capture Sybase 6.5				
ODBC = IBM Data Server Driver for ODBC and CLI 9.7				
[1] Support for System z introduced in V7.0 Fix Pack 1				
[2] Support for Windows 8 and Windows Server 2012 introduced in V7.0 Fix Pack 4				
[3] Support for Solaris 11 introduced in V7.0 Fix Pack 6				

Software product compatibility reports on ibm.com

The Software product compatibility reports portal on ibm.com provides various tools for generating reports on the hardware and software support level of IBM products. Use the following links to view reports specific to IBM solidDB 7.0.

- Operating systems for IBM solidDB 7.0
- IBM solidDB 7.0 on AIX
- IBM solidDB 7.0 on HP-UX
- IBM solidDB 7.0 on Linux
- IBM solidDB 7.0 on Solaris
- IBM solidDB 7.0 on Windows

Related concepts:

"solidDB installation requirements" on page 5

"InfoSphere CDC for solidDB system requirements" on page 7

Supported backend data servers for Universal Cache

The Universal Cache capability supports a number of IBM and other data servers as the backend data server.

IBM DB2 for Linux, UNIX, and Windows

- DB2 V9.8
- DB2 V9.7
- DB2 V9.5
- DB2 V9.1

IBM DB2 for iSeries

- DB2 for i/OS V6R1
- DB2 for i/OS V5R4

IBM DB2 for z/OS

- DB2 for z/OS V10
- DB2 for z/OS V9
- DB2 for z/OS V8

IBM Informix

- Informix V11.70
- Informix V11.50.3

The following Informix editions are supported:

- Informix Developer Edition
- Informix Ultimate Edition
- Informix Ultimate Warehouse Edition

For more information, see Informix product editions.

Oracle Database

- Oracle Database 11g
- Oracle Database 10g
- Oracle Database 9g

Microsoft SQL Server

- Microsoft SQL Server 2008
- Microsoft SQL Server 2005
- Microsoft SQL Server 2000

Sybase Adaptive Server Enterprise (ASE)

- Sybase ASE V15
- Sybase ASE V12.5.4

solidDB installation requirements

Before you install solidDB server, ensure that the system you choose meets the following software and disk and memory requirements.

- About 48 MB of disk space, including the space for separately installed documentation – the number varies considerably, depending on the platform
- At least 40 MB of RAM in the default configuration
- Adequate disk space for your database – an empty database typically requires about 16 MB of disk space
- If you use in-memory tables, additional memory to store those tables
- If you use InfoSphere CDC technology (or, the solidDB log reader is enabled), enough disk space to accommodate transaction log files preserved for replication recovery (catchup) – by default, the required log retention space is 10 GB
- Java™ Runtime Environment (JRE) or Java Development Kit (JDK), version 1.4.2 or newer, is required for
  - solidDB installation program

    Note: On Linux systems, the installation program does not support GNU Compiler for Java (GCJ).
  - Shared memory access (SMA) and linked library access (LLA) with Java

User process resource limits (`ulimits`) considerations in Linux and UNIX environments

In Linux and UNIX environments, you might need to modify the settings for the user process resource limits (**`ulimits`**) of your system. For details, see *OS user limit requirements (Linux and UNIX)*.

Security-enhanced Linux considerations

On Red Hat Enterprise Linux (RHEL) operating systems, if Security-enhanced Linux (SELinux) is enabled and in enforcing mode, the installer might fail because of SELinux restrictions.

To determine whether SELinux is installed and in enforcing mode, complete one of the following actions:

- Check the `/etc/sysconfig/selinux` file.
- Run the **`sestatus`** command.
- Check the `/var/log/messages` file for SELinux notices.

To disable SELinux, complete one of the following actions:

- Set SELinux in permissive mode and run the **`setenforce 0`** command as a superuser.
- Modify `/etc/sysconfig/selinux` and restart the computer.

If the solidDB server installs successfully on an RHEL system, all solidDB processes will run in the unconfined domain. To assign the processes to their own domains, so that also confined users can run them, you must modify the policy modules.

InfoSphere CDC for solidDB system requirements

Disk space requirements

Table 4. Disk space requirements

Disk space
InfoSphere CDC source system: • **100 GB**—Default value for the **Staging Store Disk Quota** for each instance of InfoSphere CDC. Use the InfoSphere CDC configuration tool to configure disk space for this quota. • **5 GB**—For installation files, data queues, and log files. • **Global disk quota**—Disk space is required on your source system for this quota which is used to store in-scope change data that has not been committed in your database. The amount of disk space required is determined by your replication environment and the workload of your source database. Use the **mirror_global_disk_quota_gb** system parameter to configure the amount of disk space used by this quota. InfoSphere CDC target system: • **1 GB**—The minimum amount of disk space allowed for the **Staging Store Disk Quota** for each instance of InfoSphere CDC. The minimum value for this quota is sufficient for all instances created on your target system. Use the InfoSphere CDC configuration tool to configure the disk space for this quota. • **5 GB**—For installation files, data queues, and log files. • **Global disk quota**—Disk space is required on your target system for this quota which is used to store LOB data received from your InfoSphere CDC source system. The amount of disk space required is determined by your replication environment and the amount of LOB data you are replicating. To improve performance, InfoSphere CDC will only persist LOB data to disk if RAM is not available on your target system. Use the **mirror_global_disk_quota_gb** system parameter to configure the amount of disk space used by this quota.

InfoSphere CDC may require additional disk space in the following situations:

- You are running large batch transactions in the database on your source system.
- You are configuring multiple subscriptions and one of your subscriptions is latent. In this type of scenario, InfoSphere CDC on your source system may persist transaction queues to disk if RAM is not available.
- You are replicating large LOB data types.
- You are replicating "wide" tables that have hundreds of columns.
- You are performing regular back ups of your metadata with the **dmbackupmd** command-line utility.

RAM requirements

Table 5. RAM requirements

RAM
Each instance of InfoSphere CDC requires memory for the Java Virtual Machine (JVM). The following default values for memory are assigned: • **1024 MB of RAM**—Default value for each 64-bit instance of InfoSphere CDC. • **512 MB of RAM**—Default value for each 32-bit instance of InfoSphere CDC. Use the InfoSphere CDC configuration tool to configure the memory for each instance of InfoSphere CDC. **Note:** InfoSphere CDC is predominantly a Java-based application. However, some portions of it are written in C. These portions of InfoSphere CDC are not subject to the memory limits specified for the JVM.

Although InfoSphere CDC memory requirements will fluctuate, you must work with your system administrator to ensure the allocated memory for each instance of the product is available at all times. This may involve deployment planning since other applications with memory requirements may be installed on the same server with InfoSphere CDC. Using values other than the defaults or allocating more RAM than is physically available on your server should only be undertaken after considering the impacts on product performance.

InfoSphere CDC source deployments may require additional RAM in the following scenarios:

- You are replicating large LOB data types with your InfoSphere CDC source deployment. These data types are sent to target while being retrieved from the source database. The target waits until all LOBs (for each record) are received before applying a row. LOBs are stored in memory as long as there is adequate RAM, otherwise they are written to disk on the target.
- You are replicating "wide" tables with hundreds of columns.
- You are performing large batch transactions in your source database rather than online transaction processing (OLTP).

Port requirements

InfoSphere CDC requires that you allocate a set of ports for communications with other components in the replication environment. The ports must be accessible through firewalls, although you do not require access to the internet.

Table 6. Port requirements

Protocol	Default port	Purpose
TCP	11101	Accepts connections from: • Management Console • Other installations of InfoSphere CDC as a source of replication • Command line utilities

1.1.2 Component and installation package information

The Universal Cache setups include both solidDB and InfoSphere CDC components. To install Universal Cache, you need the installation packages shown in the Universal Cache column in the table below. You need to install each Universal Cache component separately.

Table 7. solidDB V7.0 installation packages

Component	solidDB	solidDB with InfoSphere CDC replication	solidDB with Universal Cache
IBM solidDB 7.0	X	X	
IBM InfoSphere Change Data Capture solidDB 7.0		X	
IBM InfoSphere Change Data Capture Access Server 6.5		X	
IBM InfoSphere Change Data Capture Management Console 6.5		X	

Table 7. solidDB V7.0 installation packages (continued)

Component	solidDB	solidDB with InfoSphere CDC replication	solidDB with Universal Cache
IBM InfoSphere Change Data Capture *backend data server* 6.5 One of the following: • IBM InfoSphere Change Data Capture DB2 Linux, UNIX, and Windows 6.5 • IBM InfoSphere Change Data Capture Informix 6.5 • IBM InfoSphere Change Data Capture Microsoft SQL Server 6.5 • IBM InfoSphere Change Data Capture Oracle Trigger 6.5 • IBM InfoSphere Change Data Capture Oracle Redo 6.5 • IBM InfoSphere Change Data Capture Sybase 6.5 • IBM InfoSphere Change Data Capture DB2 z/OS 6.5 • IBM InfoSphere Change Data Capture DB2 iSeries 6.1			X
IBM Data Server Driver for ODBC and CLI 9.7 **Note:** Needed only in Universal Cache configurations with SQL passthrough when the backend data server is an IBM data server.			X
IBM solidDB 7.0 License Certificate	X	X	X
IBM solidDB 7.0 Documentation	X	X	X
IBM InfoSphere Change Data Capture Documentation 6.5		X	X

solidDB server package

The solidDB server package contains a complete set of the server software, including the JDBC and ODBC drivers and various utility programs.

The solidDB server package is delivered with an evaluation license certificate file, `solideval.lic`. With the evaluation license, you can evaluate solidDB for 90 days. For acquiring a permanent license, contact IBM Corporation.

Table 8. solidDB installation images

Component name	Installation package
IBM solidDB 7.0	**Linux and UNIX** solidDB-7.0-<platform>.bin **Windows** solidDB-7.0-<platform>.exe

Directory structure:

The default installation of solidDB 7.0 creates a directory called `solidDB7.0`.

The files and subdirectories in the `solidDB7.0` installation directory are explained in the following table.

Table 9. `solidDB7.0` directory structure

Location	Explanation
Root directory	The root directory contains, for example: • a script that is used to facilitate running samples in the database evaluation phase • the evaluation license file • the `welcome.html` file for accessing the package documentation
`bin`	solidDB binary files and dynamic library files

`bin/C`
 	
`bin/N`
	Auxiliary libraries for IBM Global Security Kit (GSKit)

`doc_html`, `doc_txt`	Package documentation in HTML and text format
`eval_kit/standalone`	Working directory for an evaluation version of the solidDB server. This directory contains a sample `solid.ini` configuration file and an evaluation license file (`solideval.lic`).
`eval_kit/cdc`	Working directory for an evaluation version of the solidDB server for use with Universal Cache or InfoSphere CDC replication. This directory contains a sample `solid.ini` configuration file and an evaluation license file (`solideval.lic`).
`include`	C program headers
`jdbc`	solidDB JDBC Driver Data store helper archive for use with WebSphere® (`SolidDataStoreHelper.jar`) solidDB dialect for Hibernate (`SolidSQLDialect.jar`)
`lib`	Static linkable library files
`lib32`	32-bit static linkable library files – 64-bit AIX and Solaris packages only The 32-bit libraries can be installed on 64-bit systems. The 64-bit libraries cannot be installed on 32-bit systems.
`licence`	License and notices files
`manuals`	The English versions of the manuals in PDF format can be downloaded to this folder and then accessed through the **Manuals** link on the **Welcome** page
`procedures`	SQL scripts for creating and running stored procedures for data aging and refresh
`properties`	Metadata for IBM Tivoli® Usage and Accounting Manager
`samples`	Samples that can be used in the database evaluation phase and future application development

Library file names:

The solidDB server provides many files as linkable libraries.

Most of the library files fall into one of the following categories:

- ODBC drivers
- Shared memory access and linked library access files
- Communication library files
- SA (Server API) library file

All platforms do not have every file. For example, some communication library files are available on Windows environments only.

Some library files are static, that is, they are linked to the client application executable program when you do a compile-and-link operation. Other library files are dynamic: these files are stored separately from your executable program and are loaded into memory when your program runs. For many libraries, the solidDB server provides both a static and a dynamic version on some or all platforms.

Library files are found in the following two directories:

- `bin`
- `lib`

As a rule, the `bin` directory contains dynamic libraries (in addition to executable files), while the `lib` directory contains static libraries. On Windows environments, the `lib` directory also contains the import libraries.

Additionally, on Windows environments, the ODBC and communication `.dll` libraries are copied to the `C:\Windows\system32` directory.

If you use the 32-bit installation program to install the solidDB server on a 64-bit environment, the `.dll` library files are copied to the `C:\Windows\SysWOW64` directory.

The exact library file names depend on the platform. See the following tables for examples on Windows and Linux environments:

Table 10. Example: solidDB library files in Windows 32-bit package

File name	Description
bin\	
sacw3270.dll	ODBC library - ASCII
snpw3270.dll	NamedPipes communication protocol link library
socw3270.dll	ODBC library - Unicode
sosw3270.dll	ODBC Driver Manager setup library
ssaw3270.dll	solidDB SA API library
ssolidac70.dll	Linked library access (LLA) dynamic library
stcw3270.dll	TCP/IP communication protocol link library
lib\	

Table 10. Example: solidDB library files in Windows 32-bit package (continued)

File name	Description
solidctrlstub.lib	solidDB Control API (SSC) stub library.
	This static library is used if you want to write code that can be run either locally with the linked library access library, or remotely without the linked library access.
solidimpac.lib	Linked library access (LLA) import library
solidimpodbca.lib	ODBC import library - ASCII
solidimpodbcu.lib	ODBC import library - Unicode
solidimpsa.lib	solidDB SA API import library

Table 11. Example: solidDB library files in Linux 32-bit package

File name	Description
bin\	
sacl2x70.so	ODBC shared library - ASCII
socl2x70.so	ODBC shared library - Unicode
ssal2x70.so	solidDB SA API library
ssolidac70.so	Linked library access (LLA) shared library
ssolidsma70.so	Shared memory access (SMA) shared library
lib\	
solidctrlstub.a	solidDB Control API (SSC) stub library.
	This static library is used if you want to write code that can be run either locally with the linked library access library, or remotely without the linked library access.
solidac.a	Linked library access (LLA) static library
solidodbca.a	ODBC static library - ASCII
solidodbcu.a	ODBC static library - Unicode
solidsa.a	solidDB SA API static library
libssolidac70.so	Symbolic link for shared LLA library
libssolidsma70.so	Symbolic link for shared SMA library
libsacl2x70.so	Symbolic link for shared ODBC library - ASCII
libsocl2x70.so	Symbolic link for shared ODBC library - Unicode
libssal2x70.so	Symbolic link for shared solidDB SA API library
libsolidodbca.a	Symbolic link for static ODBC library - ASCII
libsolidodbcu.a	Symbolic link for static ODBC library - Unicode
libsolidsa.a	Symbolic link for static solidDB SA API library
libsolidac.a	Symbolic link for static LLA library

For a list of the library file names on your installation of solidDB server, see the SDK Notes in the solidDB package, accessible through the **Welcome** page in your solidDB installation directory.

Dynamic library file naming conventions

Dynamic library files use the following naming convention:

```
sLLpppVV.eee
```

where

- `LL` = purpose of the library
  - `ac`: ODBC library - ASCII
  - `np`: NamedPipes communication protocol link library
  - `oc`: ODBC library - Unicode
  - `os`: ODBC Driver Manager setup (for Windows only)
  - `sa`: solidDB SA API library
  - `solidac`: Linked library access (LLA) dynamic library
  - `solidsma`: Shared memory access (SMA) dynamic library
  - `tc`: TCP/IP communication protocol link library
- `ppp` = platform
  - `a5x64`: AIX, 64-bit
  - `hia64`: HP-UX 11 64-bit (IA64)
  - `l2x`: Linux for x86
  - `l2x64`: Linux for x86, 64-bit
  - `lzx64`: Linux for System z, 64-bit
  - `s0x64`: Solaris 10 (SPARC, 64-bit)
  - `s0xi64`: Solaris 10 (ix86, 64-bit)
  - `w32`: Windows 32-bit (x86)
  - `w64`: Windows 64-bit (x86)
- `VV` = first two digits of the solidDB version, for example 70 for version 7.0, 63 for version 6.3
- `eee` = platform-specific file name extension:
  - `*.dll` Dynamic Link Library for Windows
  - `*.so` (Shared Object) for AIX, HP-UX, Linux, and Solaris

ODBC, JDBC, and proprietary programming interfaces:

The solidDB server provides ODBC, JDBC and proprietary interfaces for clients.

For more details, see the *IBM solidDB Programmer Guide*.

solidDB JDBC Driver 2.0

Table 12. solidDB JDBC Driver 2.0 key information

Compatibility	JDBC 2.0, with selected features of JDBC 2.0 Optional Package
Driver location	`<solidDB installation directory>/jdbc/SolidDriver2.0.jar`

JDBC URL format	`jdbc:solid://<hostname>:<port>/<username>/<password>[?<property-name>=<value>]...`
	
	For example:
	`"jdbc:solid://localhost:1964/dba/dba"`

Driver class name	solid.jdbc.SolidDriver

Standard compliance

The solidDB JDBC 2.0 Driver supports the JDBC 2.0 specification. Additionally, Connection Pooling, JNDI Data Sources, and Rowsets of the JDBC 2.0 Optional Package (known before as Standard Extension) are supported too.

Non-standard features include support for IBM WebSphere and timeout control extensions.

The following features of the Optional Package are currently supported by the solidDB JDBC 2.0 driver:

- Connection pooling (class solid.jdbc.ConnectionPoolDataSource)
- Connected RowSet (class solid.jdbc.rowset.SolidJDBCRowSet)
- Implemented JDBC data sources:
  - solid.jdbc.DataSource (implements javax.sqlDataSource)
  - solid.jdbc.SolidConnectionPoolDataSource (implements javax.sql.ConnectionPoolDataSource)
- JTA (Java Transaction API) – XA interface for Java (implements javax.transaction.xa.XAResource and javax.transaction.xa.Xid)

Full documentation for the solidDB JDBC Driver is included in the *IBM solidDB Programmer Guide*.

solidDB JDBC Driver extensions

The solidDB JDBC Driver supports the following non-standard extensions. For more information, see the *IBM solidDB Programmer Guide*.

JDBC URL format

You can set the connection property values in the URL string.

Connection timeout

Connection timeout refers to the response timeout of any JDBC call that invokes data transmission over a connection socket. If the response message is not received within the time that is specified, an I/O exception is thrown. The JDBC standard (2.0/3.0) does not support setting of the connection timeout. The solidDB product has two ways for doing that: one using a non-standard driver manager extension method and the other using the property mechanisms. The time unit in either case is 1 ms.

Login timeout

The timeout fires at the connect time. The setting is implemented with a connection property. The property overrides the login timeout for JDBC specified by other means (like login timeout parameter in Driver Manager).

Connection idle timeout

If the connection is inactive for the amount of time specified with the idle timeout property, the server closes the connection. The connection idle timeout property overrides the server parameter setting for the session.

Statement cache

You can set the size of the statement cache for a connection.

Transparent Connectivity Support

solidDB JDBC driver fully supports solidDB Transparent Connectivity (TC) including transparent failover and load balancing. See the *IBM solidDB High Availability User Guide* for more information about usage of Transparent Connectivity.

Shared memory access (SMA) connection property
The SMA connection property defines that the driver connects to a SMA server with a local connection, bypassing network protocols.

SQL passthrough connection properties
The SQL passthrough connection property defines the default passthrough mode for the connection.

Catalog and schema name connection properties
You can set the catalog and schema names for the connection.

WebSphere support
To support WebSphere, adata source adapter SolidDataStoreHelper is provided in a separate file `SolidDataStoreHelper.jar`, in the `'jdbc'` directory of thesolidDB package.

solidDB ODBC Driver 3.5.x

solidDB provides two ODBC drivers, one for Unicode and one for ASCII character sets. For more information about these drivers, see the *IBM solidDB Programmer Guide*.

The following functions are not supported:

- SQLBrowseConnect
- SQLSetScrollOptions
- SQLParamOptions
- SQLNativeSql
- SQLMoreResults

ODBC extensions
solidDB ODBC driver incorporates several extensions for, for example, timeout controls, statement cache behavior, and support for Transparent Connectivity. For more information, see the *IBM solidDB Programmer Guide*.

Proprietary interfaces

The solidDB Application Programming Interface (SA API) and solidDB Server Control API (SSC API) allow, for example, C programs to directly call functions inside the database server. These proprietary interfaces are provided with the solidDB shared memory access (SMA) and linked library access (LLA) libraries.

System tools and utilities:

The solidDB server package includes console tools for data management and administration, and command-line utilities for data export and import.

The tools and utilities are available in the `'bin'` directory in the solidDB server installation directory.

Console tools

solidDB SQL Editor (`solsql`)
solidDB SQL Editor (**`solsql`**) is a console tool that you can use to issue SQL statements and solidDB ADMIN COMMANDs at the command prompt. You can also execute script files that contain the SQL statements.

solidDB Remote Control (`solcon`)
solidDB Remote Control (**`solcon`**) is a console tool for administration; users

with administrator rights can issue ADMIN COMMANDs at the command prompt or by executing a script file that contains the commands. With **solcon**, the ADMIN COMMANDs can be issued as part of the **solcon** startup command line.

Because only users with administrator rights can access **solcon**, if only **solcon** is deployed at a production site, the administrators cannot accidentally execute SQL statements that could change the data.

Tools for exporting and loading data

solidDB Speed Loader (solloado or solload)
solidDB Speed Loader (**solloado** or **solload**) loads data from an external file into a database.

solidDB Export (solexp)
solidDB Export (**solexp**) exports data from a database into a file. It also creates control files used by solidDB Speed Loader (**solloado** or **solload**) to perform data load operations.

solidDB Data Dictionary (soldd)
solidDB Data Dictionary (**soldd**) exports the data dictionary of a database. It produces an SQL script that contains data definition statements that describe the structure of the database.

InfoSphere CDC packages

The InfoSphere CDC components are delivered as separately deployable packages.

InfoSphere CDC for solidDB:

The InfoSphere CDC for solidDB package contains the software for the replication engine that captures and transfers data changes between solidDB and other databases.

Table 13. InfoSphere CDC for solidDB installation images

Component name	Installation package	Contents

InfoSphere Change Data Capture solidDB	**Linux and UNIX**
	`setup-cdc-<platform>-solid.bin`
	For example:
	`setup-cdc-linux-x86-solid.bin`
	Windows
	`setup-cdc-x86-solid.exe`
		• Software for the configuration tool and the InfoSphere CDC instance for solidDB
		• solidDB JDBC Driver (`SolidDriver2.0.jar` in the `/lib` directory)
		• Tools, utilities, and samples (`/samples` directory)
		– Automation tools, utilities, and samples for scripting most common InfoSphere CDC tasks (`ucutils`, `ucpassthrough`, and `uchsbmonitor` directories)
		– Generic InfoSphere CDC samples for Java user exits and SQL scripts
		• InfoSphere CDC API documentation (`/docs` directory)

InfoSphere CDC for backend:

The InfoSphere CDC for backend package contains the software for the replication engine that captures and transfers data changes between the backend and solidDB databases.

Table 14. InfoSphere CDC for backend installation images

Component name	Installation package	Contents
InfoSphere Change Data Capture *for a backend data server* • IBM InfoSphere Change Data Capture DB2 Linux, UNIX, and Windows 6.5 • IBM InfoSphere Change Data Capture Informix 6.5 • IBM InfoSphere Change Data Capture Microsoft SQL Server 6.5 • IBM InfoSphere Change Data Capture Oracle Trigger 6.5 • IBM InfoSphere Change Data Capture Oracle Redo 6.5 • IBM InfoSphere Change Data Capture Sybase 6.5 • IBM InfoSphere Change Data Capture DB2 z/OS 6.5 • IBM InfoSphere Change Data Capture DB2 iSeries 6.1	**Linux and UNIX:** setup-<platform>-<backend_dataserver>.bin For example: setup-aix-power-udb.bin **Windows:** setup-x86-<backend_dataserver>.exe	• Software for the configuration tool and the InfoSphere CDC instance for *backend data server* • PDF format *InfoSphere Change Data Capture, End-User Documentation* (`/docs` directory) • Sample Java user exits and SQL scripts (`/samples` directory) • InfoSphere CDC API documentation (`/docs` directory)

InfoSphere CDC Access Server:

The InfoSphere CDC Access Server package contains the software for controlling access to the replication environment.

Table 15. InfoSphere CDC for Access Server installation images

Component name	Installation package	Contents

InfoSphere Change Data Capture Access Server	**Linux and UNIX**
	`cdcaccess-<version>-setup.bin`
	For example:
	`cdcaccess-6.5.1618.0-solaris-sparc-setup.bin`
	Windows
	`cdcaccess-<version>-setup.exe`
	For example:
	`cdcaccess-6.5.1618.0-setup.exe`
		• Software for controlling access to your replication environment

InfoSphere CDC Management Console:

The InfoSphere CDC Management Console package contains the software for configuring and monitoring user access and replication subscriptions. Management Console is available only on Windows environments.

Table 16. InfoSphere CDC Management Console installation images

Component name	Installation package	Contents
InfoSphere Change Data Capture Management Console	**Linux and UNIX** Not applicable, Management Console is available only on Windows environments **Windows** `cdcmc-<version>-setup.exe` For example:`cdcmc-6.5.1618.0-setup.exe`	• Software for configuring and monitoring InfoSphere CDC user access and replication subscriptions • PDF format *InfoSphere Change Data Capture Management Console, Administration Guide* (`/documentation` directory) • Online help (accessible through **Help** menu in the Management Console user interface) • IBM Java SDK and Runtime Environment Guides (`/docs` directory)

IBM Data Server Driver for ODBC and CLI package

The IBM Data Server Driver for ODBC and CLI is delivered as a compressed file. It is used with the SQL passthrough feature in Universal Cache if the backend data server is an IBM data server.

- Windows operating systems:
  `ibm_data_server_driver_for_odbc_cli_<platform>.zip`
- Linux and UNIX operating systems:
  `ibm_data_server_driver_for_odbc_cli_<platform>.tar.Z`

There is no installation program for the IBM Data Server Driver for ODBC and CLI. Instead, you must install the driver manually by uncompressing the file.

Documentation packages

Documentation for solidDB is composed of the *IBM solidDB Documentation* package and the *InfoSphere Change Data Capture Documentation* package. Both packages are available as an online information center and in PDF format.

solidDB documentation:

solidDB documentation is available online in the solidDB 7.0 Information Center and in PDF format. Most up-to-date information is always available in the Information Center.

Delivery of solidDB documentation

solidDB 7.0 Information Center

The most up-to-date solidDB documentation is available in the information center format at http://publib.boulder.ibm.com/infocenter/soliddb/v7r0/.

solidDB manuals in PDF format

The PDF manuals are available for download at the following locations:

- Software Support portal for solidDB at ftp://ftp.software.ibm.com/software/data/soliddb/info/7.0/man/.
- IBM Publications Center at http://www.elink.ibmlink.ibm.com/publications/servlet/pbi.wss

In addition, the PDF format manuals are available as the *IBM solidDB Documentation* package. This package is delivered together with the software packages in IBM Passport Advantage®, or in the Quick Start DVD in physical media deliveries.

Tip: If you download the English version PDF files to the `manuals` directory in your solidDB server installation directory, you can access the manuals also through the **Welcome** page of your solidDB software package. For detailed instructions, see section Installing solidDB Documentation package.

InfoSphere CDC documentation:

InfoSphere CDC for solidDB documentation is included in the *IBM solidDB Documentation* package. Documentation for InfoSphere CDC Management Console, InfoSphere CDC Access Server, and InfoSphere CDC engine for the back-end data server is part of the *InfoSphere Change Data Capture Documentation* package.

Delivery and location of documentation for InfoSphere CDC components

The *InfoSphere Change Data Capture Documentation* package is available in information center and PDF format:

- IBM InfoSphere Change Data Capture version 6.5 Information Center
- InfoSphere Change Data Capture 6.5 End-User Documentation in PDF format - IBM Software Support Portal
- Embedded Help accessible through the Management Console **Help** menu
- *InfoSphere Change Data Capture Documentation* installation package (PDF format), available at Passport Advantage

1.1.3 User accounts and database connection data for Universal Cache

When installing and configuring Universal Cache, you need to create or use existing user accounts and database and connection information to enable the different components to communicate with each other. The table in this section summarizes the user accounts and database connection data that are created when setting up Universal Cache.

The default values are given if available.

solidDB

Table 17. User account and network connection data for solidDB

solidDB	Example value (default if available)	Usage
Server connection data (server name and port number)	tcp 1964	• Defined in `solid.ini` configuration file • Needed when creating InfoSphere CDC for solidDB instances
Database login data	Username: soliduser Password: admsolid	• Defined when creating the solidDB database • Needed when creating InfoSphere CDC for solidDB instances

Table 17. User account and network connection data for solidDB (continued)

solidDB	Example value (default if available)	Usage
System catalog name	DBA	• Defined when creating the solidDB database The solidDB syntax for database object hierarchy is the following: `catalog_name.schema_name.database_object` For more details, see section *Managing database objects* in *IBM solidDB SQL Guide*. **Important:** Subscriptions can only include tables that are included in the System catalog.
Schema name	SOLIDUSER	• The default schema name is the username. You can create new schemas using the `CREATE SCHEMA` statement. • Needed when creating InfoSphere CDC for solidDB instances

InfoSphere CDC for solidDB

Table 18. User account and network connection data for InfoSphere CDC for solidDB

InfoSphere CDC for solidDB	Example value (default if available)	Usage
Instance Name	solid-inst	• Defined when creating the InfoSphere CDC instance • Used when administering the instance with `dm` commands.
Server Port	11101 (default)	• Defined when creating the InfoSphere CDC instance • Needed when connecting to the instance from the Management Console / Access Manager
Windows Service user account		• Defined when creating the InfoSphere CDC instance • Needed when administering InfoSphere CDC services (for example, starting the instance)
Database login data	Username: soliduser Password: admsolid Metadata schema: SOLIDUSER	• Specifies the login data to the solidDB database and the schema name that is used for InfoSphere CDC metadata tables
Server connection data	cache-node 1964	• Specifies the connection data to the solidDB server • The host name can be given as the network name or IP address. If the InfoSphere CDC replication engine is located on the same node as the solidDB server, the host name can also be `localhost`. • The port number must be a port that the solidDB server is listening to (defined in the `solid.ini` configuration file)

InfoSphere CDC for backend data server

Table 19. User account and network connection data for InfoSphere CDC for backend data server

InfoSphere CDC for backend data server	Example value (default if available)	Usage
Instance Name	BE-inst	• Defined when creating the InfoSphere CDC instance • Used when administering the instance with `dm` commands.
Server Port	10901 (default depends on the backend data server)	• Defined when creating the InfoSphere CDC instance • Needed when connecting to the instance from the Management Console / Access Manager
Windows Service user account		• Defined when creating the InfoSphere CDC instance • Needed when administering InfoSphere CDC services (for example, starting the instance)
Database login data	Depends on the backend data server	• Specifies the login data and connection settings for your backend database For details, check the section *Before your install*: *Required database, user accounts, and schemas* in the *InfoSphere Change Data Capture, End-User Documentation* for your backend data server.

Access Server

Table 20. User account and network connection data for Access Server

Access Server	Example value (default if available)	Usage
Port Number	10101 (default)	• Defined when installing (Windows) or configuring (Linux and UNIX) the Access Server • Needed when login in to the Management Console
Login data (System Administrator)	Username: admin (default) Password: uc123	• Defined when installing (Windows) or configuring (Linux and UNIX) the Access Server • Specifies the username for the Access Server System Administrator • Needed when login in to the Management Console

Management Console

Table 21. User account and network connection data for Management Console

Management Console	Example value (default if available)	Usage
Login data (System Administrator)	Username: admin (default) Password: uc123	• Defined when installing (Windows) or configuring (Linux and UNIX) the Access Server • Specifies the username for the Access Server System Administrator
Server Name		• Specifies the host name (system name) or full IP address of the workstation running Access Server. • Used my Management Console to connect to the Access Server
Port Number	10101 (default)	• Defined when installing (Windows) or configuring (Linux and UNIX) the Access Server

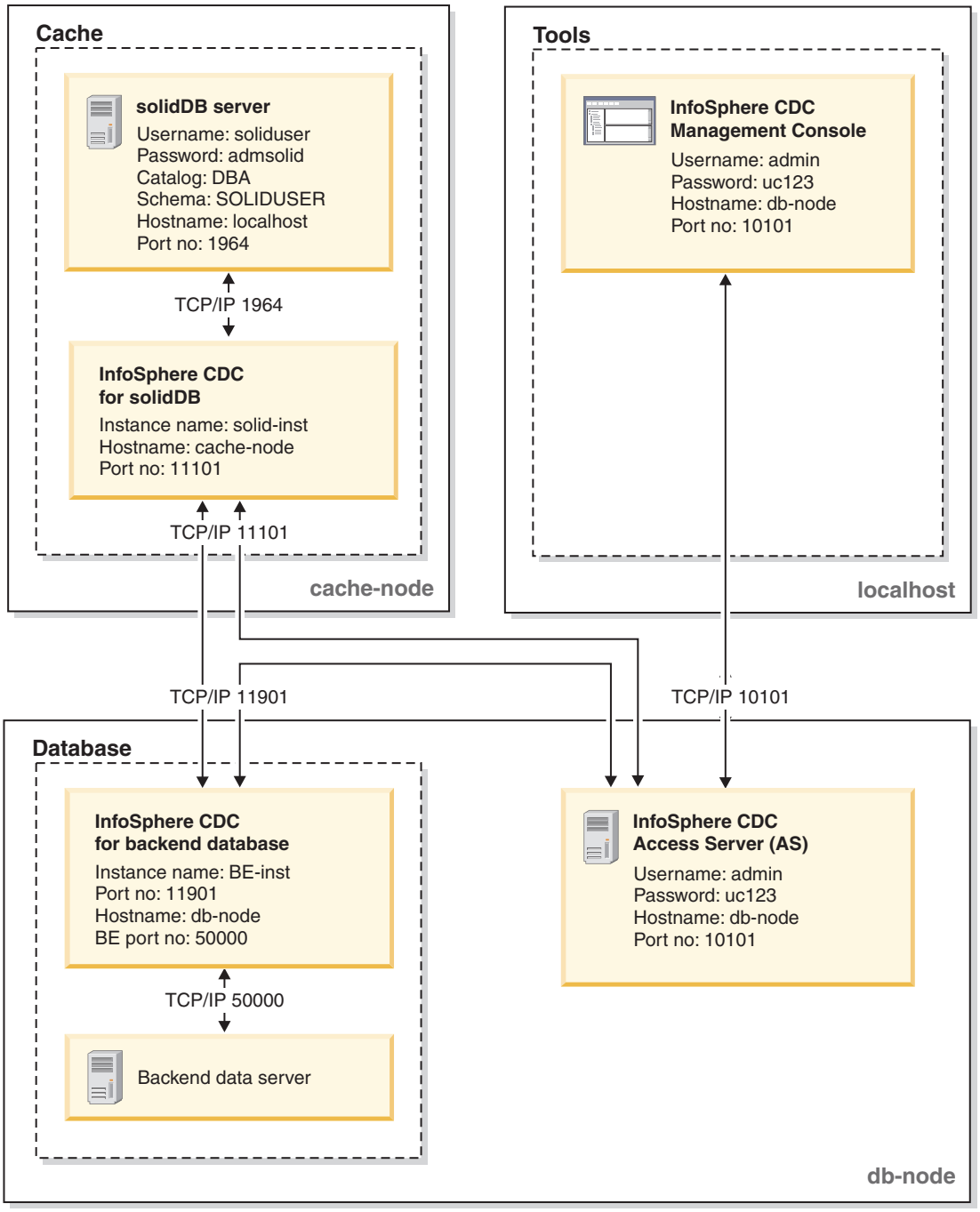

Figure 1. Example: User accounts and database connection data for Universal Cache

1.2 Installation topologies for Universal Cache

You can install the Universal Cache components on the same server for a simple evaluation topology, or on independent servers for a production-level topology.

General principles

- There can be several solidDB cache databases in the Universal Cache deployment, but only one backend data server.

- Typically, the InfoSphere CDC instance is created on each node participating in InfoSphere CDC replication.
- The solidDB server and the InfoSphere CDC for solidDB instance do not need to be located on the same node.

  This is because InfoSphere CDC for solidDB can read and insert data into a solidDB database using both local and remote JDBC connection.
- In configurations using solidDB High Availability (HotStandby), the InfoSphere CDC instance must run on a different node than the solidDB server.

1.2.1 Example: Evaluation topology

In a typical evaluation setup, all the Universal Cache components are installed on a single computer, except for the backend dataserver. Typically you would also already have a working installation of the backend data server that contains the data you want to cache into an in-memory solidDB database.

The evaluation topology is appropriate for exploring the software or for training environments, but not for production environments.

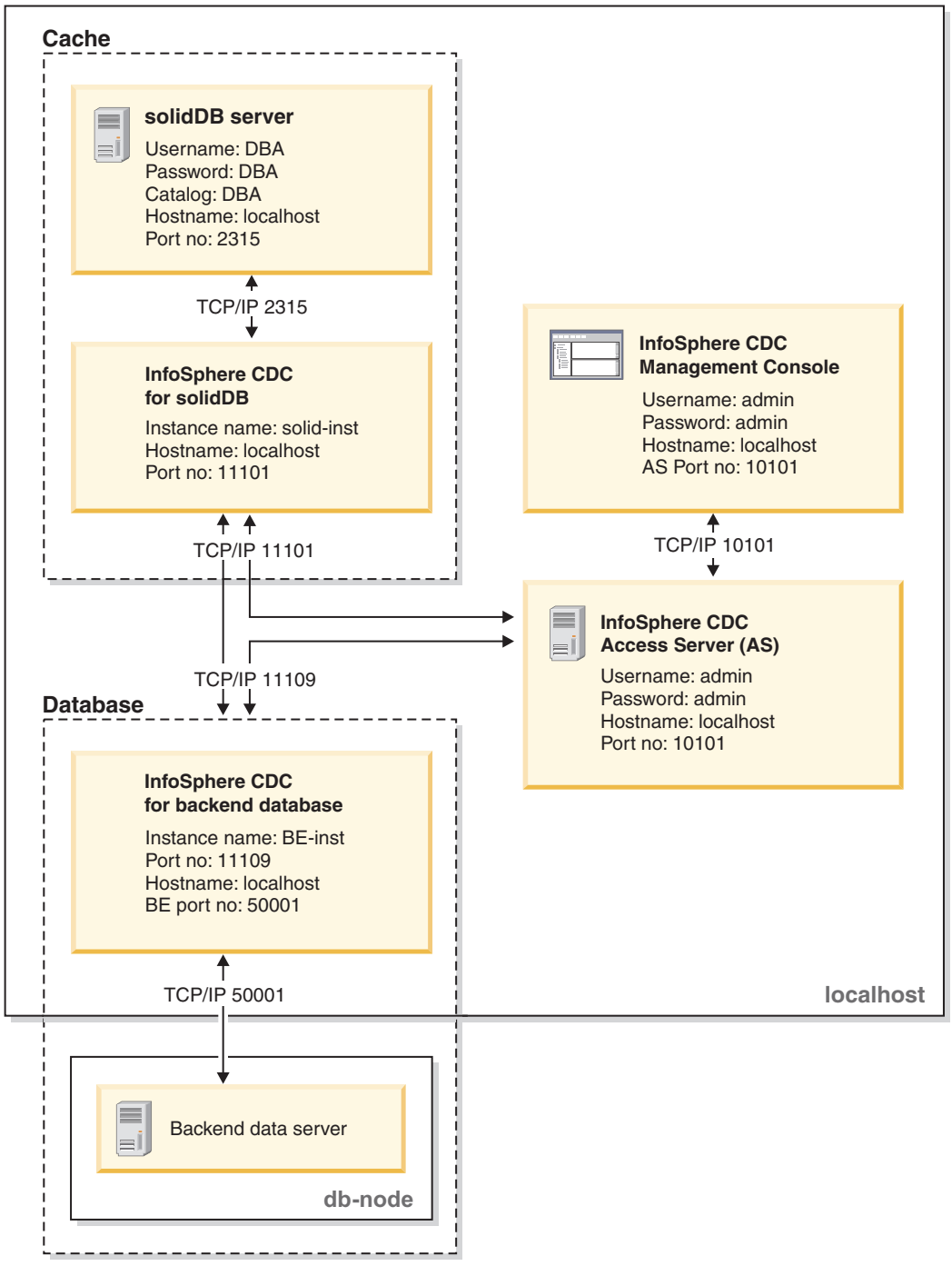

Figure 2. Universal Cache - evaluation topology

1.2.2 Example: Production topology

In a typical production setup, the cache and database components are installed on separate server machines and tooling is located on a management node. Access Server can be located, for example, on the backend database node.

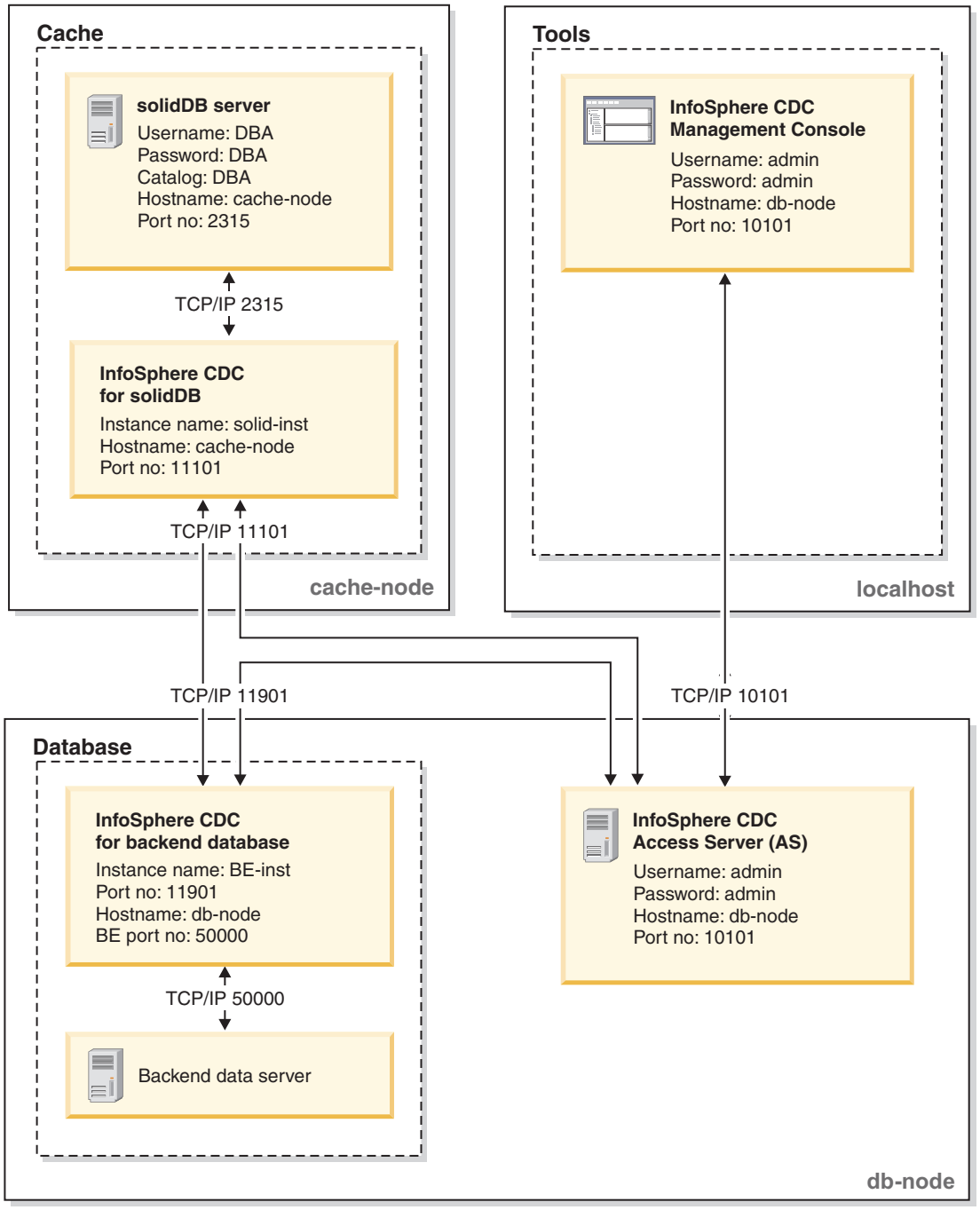

Figure 3. Typical Universal Cache deployment topology - production

1.2.3 Example: Multiple cache databases topology

Multiple solidDB servers can be used, for example, for partitioning backend data over several solidDB cache databases.

Note: In a deployment with multiple cache databases, each solidDB server is autonomous and processes the application requests without accessing data in any of the other solidDB servers.

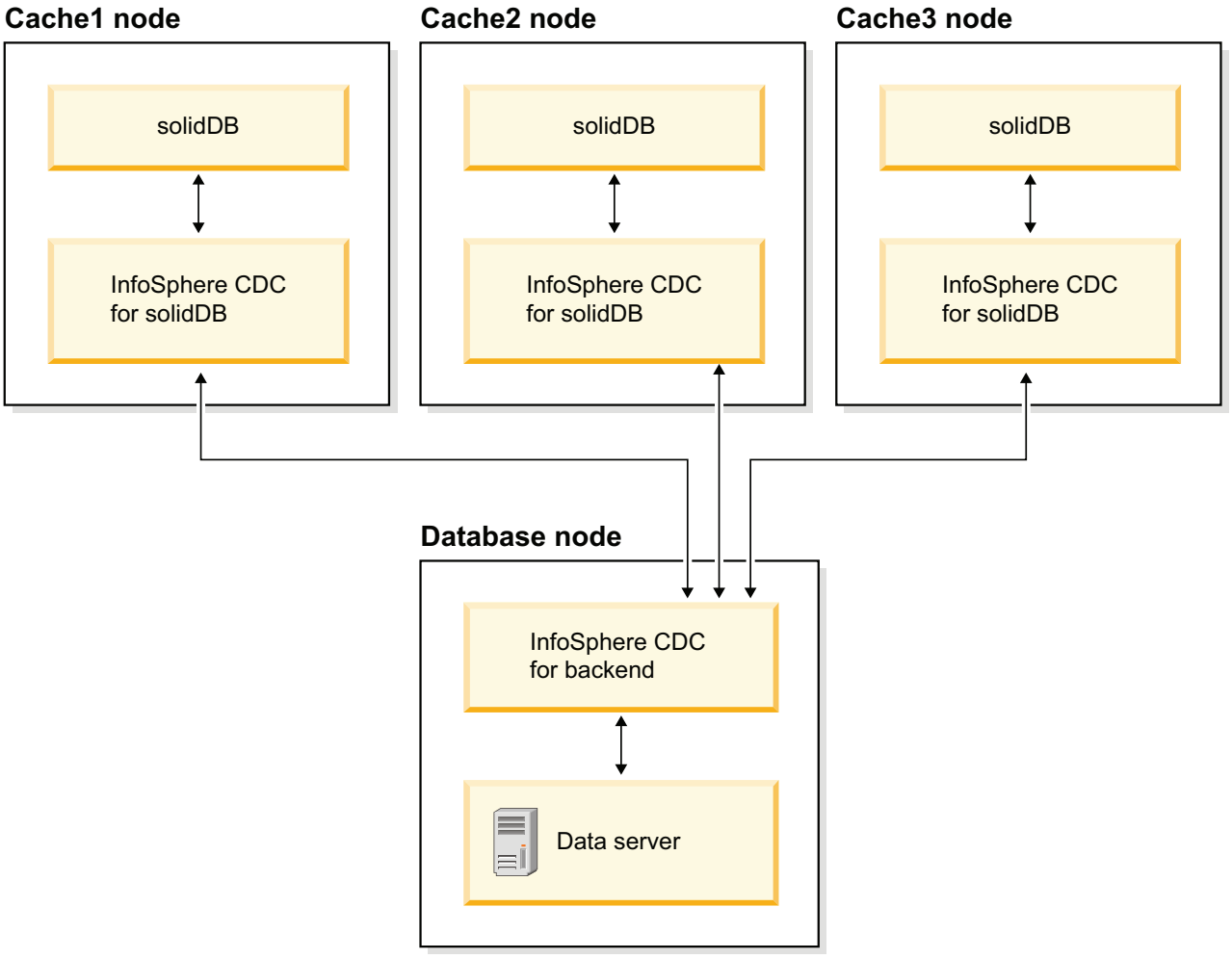

Figure 4. Universal Cache deployment with multiple solidDB servers

Note: The above topology illustration does not include the Access Server or the management tools. The Access Server would typically be located on the Database node and the management tools on a separate management node.

1.2.4 Example: Universal Cache with High Availability topology

In a typical HotStandby setup, all InfoSphere CDC instances will run on the backend database node, and the connection to the HotStandby pair is established remotely. The management tools run on a separate node.

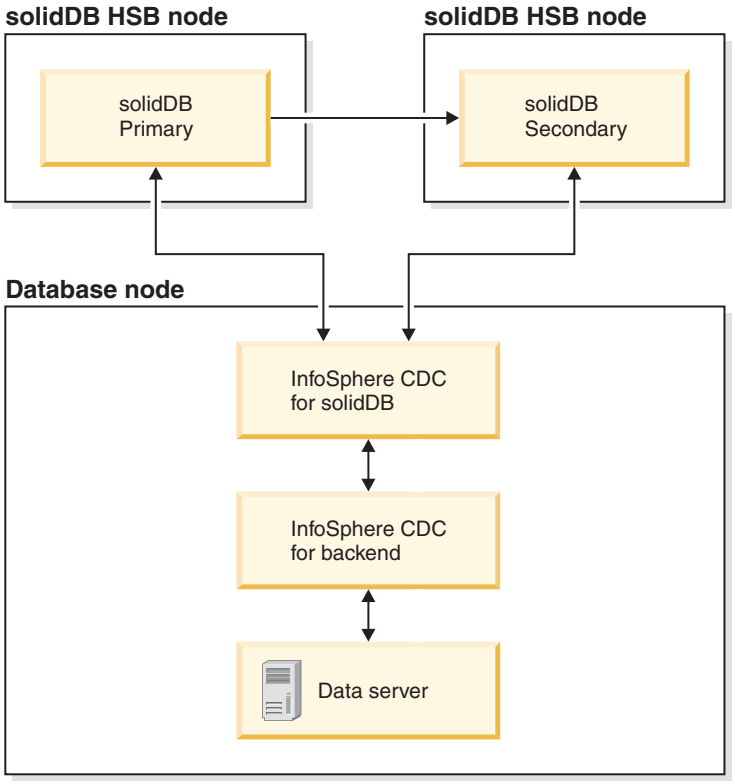

Figure 5. Example: Universal Cache with solidDB High Availability

Note: The above topology illustration does not include the Access Server or the management tools. The Access Server would typically be located on the Database node and the management tools on a separate management node.

1.3 Installing and configuring Universal Cache

You install and configure the Universal Cache components using the individual installation programs. To deploy Universal Cache, you need to install and configure solidDB server, your backend data server, the InfoSphere CDC replication engines for solidDB and backend server, the InfoSphere CDC Access Server, the InfoSphere CDC Management Console, and tooling and drivers as necessary for your environment.

1.3.1 Overview of Universal Cache installation and configuration steps

This section provides a high-level overview of the installation and configuration steps for Universal Cache.

Note:

- The installation and configuration instructions assume your configuration includes only one solidDB server. Repeat the steps for any additional solidDB servers in your configuration.
- Install the components in the order described below; this is to ensure that you meet installation and configuration requirements for each component.

1. **Locate the installation images for the Universal Cache components**.

For a list of the installation images for different platforms, see 1.1.2, "Component and installation package information," on page 8.

2. **Ensure that you have access to all the following documentation packages that are needed when installing Universal Cache.**
   - IBM solidDB 7.0 Information Center or *IBM solidDB 7.0 Documentation* package in PDF format
   - IBM InfoSphere Change Data Capture version 6.5 Information Center or *InfoSphere Change Data Capture Documentation* package in PDF format
3. **Ensure that you have system administrator (or equivalent) access rights to all the nodes where you will install the Universal Cache components.**

   Tip: While setting up Universal Cache, you need to create (or use existing) user accounts, database, and network connection identification data to enable the different components to communicate with each other.

   A summary of the key identification data is available in User accounts and database connection data for Universal Cache.
4. **Install and configure IBM solidDB server.**

   For details, see Installing and configuring solidDB server for Universal Cache.

   Result: You have a working solidDB server installation with an empty database.
5. **Install and configure InfoSphere CDC for IBM solidDB**.

   For details, see 1.3.3, "Installing and configuring InfoSphere CDC for solidDB," on page 31.

   Result: You have a working installation and you have created at least one InfoSphere CDC instance which is connected to your solidDB database.
6. **Install and configure the backend data server.**

   For details, see 1.3.4, "Installing and configuring your backend data server," on page 32.

   Result: You have a working installation of the backend server with a database that contains data that you want to replicate to and from solidDB.
7. **Install and configure InfoSphere CDC for the backend data server.**

   For details, see Installing and configuring InfoSphere CDC for your backend data server.

   Result: You have a working installation of the replication engine and you have created at least one InfoSphere CDC instance that is connected to your backend database.
8. **Install InfoSphere CDC Access Server.**

   For details, see Installing and configuring Access Server.

   Result: You have a working Access Server installation and you have created a system administrator account for logging into the Management Console.
9. **Install InfoSphere CDC Management Console.**

   For details, see Installing and configuring Management Console.

   Result: You have a working Management Console installation and you can login into the InfoSphere CDC Management Console using the system administrator account.
10. **Set up the replication subscriptions.**

    For details, see 2.1, "Setting up caching with Management Console," on page 41.

Result: You have created replication subscriptions between your solidDB and backend data server.

1.3.2 Installing and configuring solidDB server for Universal Cache

Installing solidDB server for Universal Cache

Procedure

1. Install Java Runtime Environment (JRE) or Java Development Kit (JDK), version 1.4.2 or newer, if not already installed.

   JRE or JDK 1.4.2 or newer is needed to run the solidDB installer.

   Note: On Linux systems, GNU Compiler for Java (GCJ) is not supported.
2. On the downloaded installation image or the installation DVD, locate the installation program file for your operating system:
   - `solidDB-7.0-<platform>.exe` (Windows)
   - `solidDB-7.0-<platform>.bin` (Linux and UNIX)
3. Double-click the installation program file. The solidDB installation wizard starts.
4. Follow the instructions on the wizard to complete the installation.

   Note: In Linux and UNIX operating systems, you must be able write to the directory that you are using for the installation. If the installation program cannot create the directory, you are prompted to specify a different directory.
5. Verify your solidDB server installation and familiarize yourself with the basic administration tasks.

   For more information, see the section *Verifying solidDB installation* in the *IBM solidDB Getting Started Guide* , as well as the *IBM solidDB Administrator Guide*.

What to do next

"Configuring solidDB server for Universal Cache capability"

Configuring solidDB server for Universal Cache capability

To be able to use the solidDB server with the InfoSphere CDC technology, you need modify configuration settings so that the InfoSphere CDC for solidDB can connect to and replicate data from your solidDB database.

Before you begin

This section assumes that you are familiar with solidDB administration and have read, for example, the sections *Administering solidDB* and *Configuring solidDB* in the *IBM solidDB Administrator Guide*.

Procedure

1. **Set up your database environment by creating a working directory, your solidDB database, and user accounts.**

   For instructions, see *Creating a new database* in the *IBM solidDB Administrator Guide*.

   Tip:

After you have installed the solidDB server, you can find the following directories in the installation directory:

```
<installation directory>
    bin\
    ..
    eval_kit\
        standalone\
        cdc\
    ..
    samples
    ..
```

You can use the `eval_kit/cdc` directory in the solidDB server installation directory as your working directory; it contains a sample `solid.ini` file for using solidDB with the Universal Cache capability or InfoSphere CDC replication.

2. **Configure the Log Reader by modifying the configuration parameters in the `LogReader` section of the `solid.ini` configuration file.**
   a. Set the **`LogReader.LogReaderEnabled`** parameter to `yes`.

      ```
      [LogReader]
      LogReaderEnabled=yes
      ```

      You must enable the Log Reader to be able to use the solidDB database as a source database in InfoSphere CDC replication. The factory value of the **`LogReader.LogReaderEnabled`** parameter is `no`.

   b. Set the transaction log retention space size with the **`LogReader.MaxLogSize`** parameter.

      ```
      [LogReader]
      MaxLogSize=<MB>
      ```

      The **`LogReader.MaxLogSize`** parameter sets the amount (size) of log files that are available for performing a catchup. The maximum size of log file depends on the available disk space and the expected downtime after which a catchup is needed. The factory value is 10 240 (10 GB).

      If the log reader is enabled, the specified log file retention space is always used fully. Also, the log files can use even more space, if backups are not performed or the parameter **`General.CheckpointDeleteLog`** is set to `no`.

   c. Set the in-memory buffer size for log records with the **`LogReader.MaxSpace`** parameter.

      ```
      [LogReader]
      MaxSpace=<number of log records>
      ```

      The **`MaxSpace`** parameter sets the size (in number of log records) of the in-memory log reader buffer used in throttling. The maximum number of log records depends on the expected size of load bursts. The factory value is 100000 log records.

      The size of a log record is that of the (binary) row size, plus a few bytes of additional metadata overhead. When the buffer fills up, throughput throttling is applied; the operations are blocked until there is room in the log reader buffer.

3. **Modify other performance and database-setup-related configuration parameters as necessary.**
   - **`Logging.DurabilityLevel`**

     By default, the solidDB server durability level is set to relaxed (**`Logging.DurabilityLevel=1`**). Relaxed durability means that most recent transactions can be lost if the server fails unexpectedly.

To prevent loss of data, set the durability level to strict with the following setting in the `solid.ini` file:

```
[Logging]
DurabilityLevel=3
```

Note: Strict durability setting induces a performance penalty when compared to relaxed durability. Relaxed durability can be used without the risk of data loss if solidDB HA (HotStandby) configuration is applied with the 2-Safe replication protocol (default).

- **`General.DefaultStoreIsMemory`**

  By default, the solidDB table storage type is set to M-table (**`General.DefaultStoreIsMemory=yes`**).

- **`Sql.IsolationLevel`**

  By default, the solidDB isolation level is set to Read Committed (**`Sql.IsolationLevel=1`**).

1.3.3 Installing and configuring InfoSphere CDC for solidDB

To install InfoSphere CDC for solidDB, follow the instructions in the installation wizard. After installation, use the InfoSphere CDC configuration tool to configure your InfoSphere CDC for solidDB instances.

Before you begin

Ensure that:

- Your solidDB server is running.
- You have created your solidDB database.
- You know the username and password for your solidDB database.
- You know the network address and port number the solidDB server is listening to.
- You have created a new schema or decided which existing schema you want InfoSphere CDC for solidDB to create the metadata tables in.

Procedure

1. **Install InfoSphere CDC for solidDB.**
   a. On the downloaded installation image or the installation DVD, locate the installation program file for your operating system:
      - `setup-x86-solid.exe` (Windows)
      - `setup-<platform>-solid.bin` (Linux and UNIX)
   b. Double-click the installation program file. The installation wizard starts.
   c. Follow the wizard's instructions to complete the installation.

      Note: In Linux and UNIX operating systems, you must be able write to the directory that you are using for the installation. If the installation program cannot create the directory, you are prompted to specify a different directory.

   At the end of installation, select to launch the InfoSphere CDC configuration tool to configure your InfoSphere CDC for solidDB instances.
2. **Using the configuration tool, create a new instance of InfoSphere CDC for solidDB.**

   For more information about how to create new InfoSphere CDC for solidDB instances, see 10.3, “Configuring InfoSphere CDC,” on page 107.

Note: If your configuration deploys solidDB High Availability, you need to create one InfoSphere CDC instance where you define the host address and port number for the primary and secondary solidDB servers.

What to do next

Continue with 1.3.4, “Installing and configuring your backend data server.”

Related concepts:

10.3, “Configuring InfoSphere CDC,” on page 107

1.3.4 Installing and configuring your backend data server

Install and configure your backend data server according to the instructions provided with your backend data server, noting any special requirements set in the *InfoSphere Change Data Capture, End-User Documentation* for the InfoSphere CDC for your backend data server.

Procedure

1. **Check the installation pre-requisites for your backend data server when using it with InfoSphere CDC.**

   The installation requirements are described in section *Before you install* in the *InfoSphere Change Data Capture, End-User Documentation* for your backend data server.
2. **Install the backend data server according to the instructions provided with the product.**

What to do next

1.3.5, “Installing and configuring InfoSphere CDC for your backend data server”

1.3.5 Installing and configuring InfoSphere CDC for your backend data server

To install InfoSphere CDC for your backend data server, follow the instructions in the installation wizard. After installation, use the InfoSphere CDC configuration tool to configure your InfoSphere CDC instances.

Before you begin

- Check that your backend data server is running.
- You have created your backend database.
- You know the username and password for your backend database.
- You know the network address and port number the backend data server is listening to.
- You have created a new schema or decided which existing schema you want InfoSphere CDC to create the metadata tables in.

Procedure

1. **Check the installation prerequisites.**

   The installation requirements are described in section *Before you install* in the *InfoSphere Change Data Capture, End-User Documentation* for your backend data server.
2. **Install InfoSphere CDC for the backend data server.**

For detailed instructions, see section *Installing InfoSphere CDC* in the *InfoSphere Change Data Capture, End-User Documentation* for the backend data server. At the end of installation, select to launch the InfoSphere CDC configuration tool to configure your InfoSphere CDC instances.

3. **Using the configuration tool, create a new instance of InfoSphere CDC for the backend data server.**

   For detailed instructions, see section *Configuring InfoSphere CDC* in the *InfoSphere Change Data Capture, End-User Documentation* for the backend data server.

   Tip: If you intend to use bidirectional replication and your backend data server is DB2 for Linux, UNIX, and Windows, set the InfoSphere CDC for DB2 system parameter **ddl_awareness** to `false`.

What to do next

1.3.6, “Installing and configuring InfoSphere CDC Access Server”

1.3.6 Installing and configuring InfoSphere CDC Access Server

To install the Access Server, follow the instructions in the installation wizard. After installation, if your network uses a firewall or other security mechanism that requires static ports for communication, you must specify the ports that other computers can use to communicate with Access Server services.

Procedure

1. Install Access Server according to the instructions in the InfoSphere Change Data Capture Access Server and Management Console, Installation Guide.

   Important: The Access Server account is created during the installation. The Access Server account is used for the following operations:

   - Logging in to Management Console
   - Managing users and datastores in Management Console

2. If necessary for your environment, specify the ports that other computers can use to communicate with Access Server services.

   For instructions, see section *After you install Access Server* in the InfoSphere Change Data Capture Access Server and Management Console, Installation Guide.

What to do next

1.3.7, “Installing and configuring InfoSphere CDC Management Console”

1.3.7 Installing and configuring InfoSphere CDC Management Console

To install the Management Console, follow the instructions in the installation wizard. After installation, log in to the Management Console using the system administrator account that you created when installing the Access Server.

Procedure

1. Install Management Console according to the instructions in InfoSphere Change Data Capture Access Server and Management Console, Installation Guide.

2. Log in to the Management Console with the system administrator account that you created when installing the Access Server.

What to do next

- View the Management Console help documentation through the **Help** > **Help Contents** menu path.
- Continue setting up Universal Cache.

1.4 Installing drivers

All solidDB drivers are installed as part of the solidDB server installation. If your application is located on a different computer than the solidDB server, you need to install the drivers on the computer where the application is located. Also, to establish a connection between the cache and backend databases for the purposes of SQL passthrough functionality, you need to install and configure a backend-specific ODBC driver on the solidDB node.

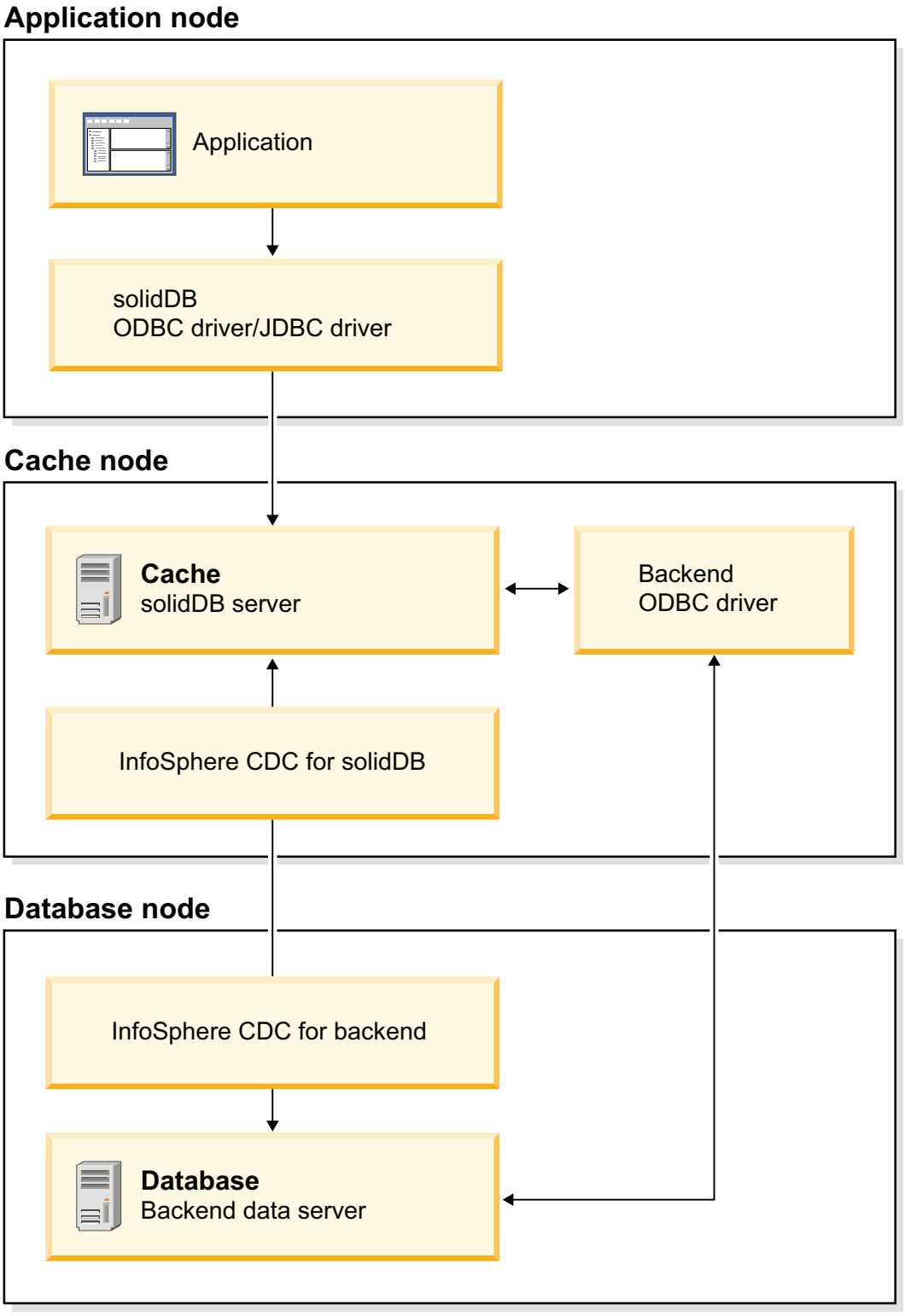

Figure 6. Universal Cache drivers

1.4.1 Installing solidDB JDBC Driver

The solidDB JDBC Driver (`SolidDriver2.0.jar`) is installed during solidDB server installation. Depending on your environment, you might need to set various configuration settings before you use the solidDB JDBC Driver.

Default installation directory

The solidDB JDBC Driver is installed during the solidDB server installation into the `jdbc` directory.

If your application is on a different computer than the solidDB server, you must copy the JDBC driver file to the computer where the application is located.

Tip:

- The `jdbc` directory contains also the solidDB Data Store Helper Class (`SolidDataStoreHelper.jar`) for use with WebSphere.
- The `samples/jdbc` directory in the solidDB installation directory contains Java code samples that use the solidDB JDBC Driver. Instructions for running the sample are available in the `readme.txt` file, which is available in the same directory.

Requirements for Java environment

- Ensure that you have a working Java runtime or development environment that supports JDBC API specification release 2.0.
- Check from your Java environment documentation whether it can use compressed bytecode. The `SolidDriver2.0.jar` contains the solidDB JDBC Driver classes in compressed bytecode format usable by most Java virtual machines. However, some environments (such as Microsoft J++) require uncompressed bytecode. If your environment requires uncompressed bytecode, you must extract the `SolidDriver2.0.jar` file by using a tool that supports long file names.

Setting the CLASSPATH environmental variable

The CLASSPATH environment variable for your environmentmust include the solidDB JDBC Driver `.jar` file installation path.

Windows

The installation adds the default solidDB JDBC Driver installation path to the System CLASSPATH environment variable automatically.

You can check and set the System CLASSPATH environment variable through the Control Panel:

Control Panel > **System** > **Advanced** > **Environment Variables**

Linux and UNIX

Set your CLASSPATH environment variable to include the solidDB JDBC Driver (`SolidDriver2.0.jar`) installation path.

For example, in Bourne shell, use the following command:

```
export CLASSPATH=<solidDB installation directory>/jdbc/SolidDriver2.0.jar:$CLASSPATH
```

If you are using another shell than the Bourne shell, modify this command to make it appropriate for your shell.

1.4.2 Installing solidDB ODBC Driver

The solidDB installation program installs two ODBC Drivers: one for Unicode and one for ASCII. The Unicode version is a superset of the ASCII version; you can use it with either Unicode or ASCII character sets. On Windows environments, you can also use the solidDB installation program to install only the ODBC driver.

Windows

In Windows environments, the solidDB installation program installs the ODBC drivers and the following system Data Source Names (DSN) automatically. You can also add you own user DSNs.

- Windows 32-bit operating systems:
  - IBM solidDB 7.0 32-bit – ANSI
  - IBM solidDB 7.0 32-bit – Unicode
- Windows 64-bit operating systems:
  - IBM solidDB 7.0 64-bit – ANSI
  - IBM solidDB 7.0 64-bit – Unicode

Linux and UNIX

In Linux and UNIX environments, the ODBC driver library files are installed to the following directories:

- `<solidDB installation directory>/bin/`: dynamic library files
  - `sac<platform><version>.sa` or `sac<platform><version>.so` – ANSI
  - `soc<platform><version>.sa` or `soc<platform><version>.so` – Unicode
- `<solidDB installation directory>/lib/`: static library files
  - `solidodbca.sa` or `solidodbca.so` – ANSI
  - `solidodbcu.sa` or `solidodbcu.so` – Unicode

The file extension `.sa` or `.so` depends on the operating system.

Installing ODBC drivers without solidDB installation (Windows)

To install the ODBC drivers without installing solidDB in Windows environments:

1. Start the solidDB installation program.
2. Select **Custom** installation.
3. Select **ODBC** (clear **Server** and **Samples**).
4. Follow the displayed instructions to complete the installation.

Installing ODBC drivers without solidDB installation (Linux and UNIX)

To install the ODBC drivers without installing solidDB in Linux and UNIX environments:

1. Install solidDB using the installation program.
2. Copy the ODBC driver library file to your client node.

1.4.3 Installing and configuring backend ODBC drivers for SQL passthrough

To use the SQL passthrough functionality, you must install and configure the backend ODBC driver on the solidDB frontend node. You can link to the driver directly (using a dynamic driver library) or using a driver manager.

Before you begin

Locate the ODBC driver installation package as well as installation and configuration instructions for your backend data server.

- If your backend data server is an IBM data server, use the *IBM Data Server Driver for ODBC and CLI* provided with the solidDB Universal Cache installation images.
- If your backend data server is not an IBM data server, use the native ODBC driver provided with your backend data server.

Procedure

1. **Install the backend ODBC driver (client) on the solidDB node.**
   - If your backend data server is an IBM data server, proceed as follows.
     a. Copy the compressed file that contains the *IBM Data Server Driver for ODBC and CLI* onto the solidDB node from the installation image.
     b. Uncompress that file into your chosen install directory on the solidDB node.
     c. Optional: remove the compressed file.
     d. If your frontend solidDB data server is running on AIX:
        1) Extract the shared library (`/odbc_cli/clidriver/lib/libdb2.a`) to yield `shr_64.o` on 64-bit operating systems. To avoid confusion, rename the file to `libdb2.so`.

           Issue the following commands:

           ```
           cd odbc_cli/clidriver/lib
           ar -x -X 64 libdb2.a
           mv shr_64.o libdb2.so
           ```

           These steps are necessary on AIX because solidDB will load the driver dynamically.

           Important: When referencing the driver library on AIX systems, remember to use the correct file name (`libdb2.so`).
        2) Set the DB2NOEXITLIST environment variable to ON.

           Issue the following command on the solidDB node:

           ```
           export DB2NOEXITLIST=ON
           ```

           This environment variable ensures that upon shutdown, the driver does not attempt to free resources that have already been freed by solidDB.
   - If your backend data server is not an IBM data server, follow the instructions provided with your backend data server.
2. **Define the connection settings between the ODBC driver and the backend data server.**

   The backend ODBC driver for SQL passthrough is configured in the same way as you would configure a regular remote connection with your backend database. You can link to the driver directly or using a driver manager.

- **Direct linking**

  Depending on your backend data server and operating system, you may need to set environmental variables or other setup parameters to enable direct linking.

  For details, see the examples below or the instructions that came with your backend data server.

- **Driver manager**

  Depending on your backend data server, operating system, and driver manager, you need to configure such settings as data source name, login data, performance options, or connection options.

  For details, see the examples below or the instructions that came with your backend data server.

3. **Define the connection settings between solidDB server and the driver or driver manager by modifying the [Passthrough] section of the solid.ini configuration file.**

   The format of the parameter values depends on whether you link to the driver directly or using a driver manager.

   Direct linking

   - Use **RemoteServerDriverPath** to set the driver path.
   - Use **RemoteServerDSN** to set the driver connect string.

     Note: The exact connect string depends on the driver.

     Example: *IBM Data Server Driver for CLI and ODBC* with DB2 or IDS on Linux operating systems

     ```
     [Passthrough]
     RemoteServerDriverPath=/home/solid/odbc_cli/clidriver/lib/libdb2.so
     RemoteServerDSN="Driver={IBM DB2 ODBC DRIVER};Dat
        abase=my_ids;Hostname=9.212.253.10;Port=9088;protocol=TCPIP;"
     ```

   Driver manager

   - Use **RemoteServerDriverPath** to set the driver manager path.
   - Use **RemoteServerDSN** to set the Data Source Name.

     Example: unixODBC DriverManager with DB2

     ```
     [Passthrough]
     RemoteServerDriverPath=/usr/lib/libodbc.so
     RemoteServerDSN=BE_DB2
     ```

4. **Configure the ODBC driver's code page support according to the solidDB database mode (Unicode or partial Unicode).**

   - **Unicode databases**

     If your solidDB database mode is Unicode (**General.InternalCharEncoding=UTF8**), configure the ODBC driver to expect data from solidDB in UTF-8 encoding.

     The procedure for configuring UTF-8 support depends on the driver. For details, see the instructions that came with your backend data server.

     For example, in DB2 for Linux, UNIX, and Windows environments, the UTF-8 support is configured by setting the environment variable DB2CODEPAGE to 1208. (The value 1208 is the identifier for the UTF-8 code page in DB2 environments.)

   - **Partial Unicode databases**

     – If your solidDB database mode is partial Unicode (**General.InternalCharEncoding=Raw**) and your application and solidDB environments use ASCII or Latin-1 encoding (Western languages), the

backend ODBC driver is likely to handle the character translations correctly without setting any code page support in the ODBC driver explicitly.

For example, if you have installed *IBM Data Server Driver for ODBC and CLI* in a system using ASCII encoding, the installation has set the driver to use the system locale of the installation node automatically.

– If your backend database uses encoding other that ASCII or Latin-1, set the backend ODBC driver to expect data from solidDB in ASCII or Latin-1 encoding.

  The procedure for configuring ASCII or Latin-1 support depends on the driver. For details, see the instructions that came with your backend data server.

Important: The conversion between the application encoding and the solidDB server is handled by the solidDB ODBC Driver or solidDB JDBC Driver.

- In C/ODBC environments, the code page conversions between the application and solidDB are controlled with the server-side parameter **`Srv.ODBCDefaultCharBinding`** or the client-side parameter **`Client.ODBCCharBinding`**.
- In Java/JDBC environments, no settings are needed. The code page conversions are handled by the solidDB JDBC Driver automatically.

For more information about setting the parameters and solidDB Unicode support in general, see Using Unicode in the *IBM solidDB Programmer Guide*.

5. **If you backend data server is DB2, running on a 64-bit system, and you are using the IBM Data Server Driver for CLI and ODBC with direct linking, set the solidDB parameter `Passthrough.Force32bitODBCHandles` to `yes`.**
6. **If your backend data server is DB2 for iSeries or DB2 for z/OS, set the InfoSphere CDC for solidDB system parameter `retrieve_credentials` to `false`.**

Related reference:

2.2.1, “Important InfoSphere CDC system parameter settings for Universal Cache,” on page 46

Depending on your backend data server and database setup, you might need to modify InfoSphere CDC system parameter settings. In addition to the generic parameter settings that apply to any configuration, there are parameters that are specific to Universal Cache use only.

2 Setting up caching

After you have installed all the Universal Cache components, use the InfoSphere CDC Management Console to set up the caching subscriptions. You might need to use a number of InfoSphere CDC parameters and commands in the setup phase too.

2.1 Setting up caching with Management Console

The InfoSphere CDC Management Console is an interactive GUI tool that you can use to configure and monitor replication (caching) *subscriptions* between the cache and backend databases. This section provides a high-level overview of how you can create replication subscriptions for Universal Cache purposes. The steps contain references to more detailed instructions in the *InfoSphere Change Data Capture Management Console, Administration Guide*.

Before you begin

- Check that the tables you intend to replicate exist at least in the backend database. It is also possible to create tables during the replication if the tables do not contain foreign keys.
- Check that your solidDB and backend databases are running.
- Check that your InfoSphere CDC instances for solidDB and backend data servers are running.
- Check that you have sufficient access privileges to your databases.
- Check that you have defined your desired replication principles in accordance with your business rules. For details, see 2.1.2, “Deciding on the replication model,” on page 45.
- If you intend to use bidirectional replication and your backend data server is DB2 for Linux, UNIX, and Windows, set the InfoSphere CDC for DB2 system parameter **`ddl_awareness`** to `false`.

Procedure

1. **Log in to Management Console by connecting to Access Server.**

   For more details, see section *Logging into Management Console (Connecting to Access Server)* in the *InfoSphere Change Data Capture Management Console, Administration Guide*.

   Tip: To be able to work in the Access Manager perspective of the Management Console, you must be a System Administrator that has the privilege to manage datastores and user accounts. The System Administrator account was created during the installation of the Management Console.

2. **Set up datastores for the solidDB and backend databases.**
   a. Add new datastore for the solidDB database.
      1) Click **Access Manager** > **Datastore Management**.
      2) Click **File** > **Access Server** > **New Datastore**.
      3) Type the name of the datastore in the **Name** box.
      4) Type a description in the **Description** box.
      5) In the **Host Name** box, type the host name or the full IP address of the server where you have installed InfoSphere CDC for solidDB.

6) In the **Port** box, type the port number which InfoSphere CDC uses for communication with the other components. For example, InfoSphere CDC for solidDB uses by default port number 11101.
7) Ping the server. If successful, this returns the datastore properties including the type of server where you have installed InfoSphere CDC and the version number of the product.

b. Add new datastore for the backend database.
1) Click **Access Manager** > **Datastore Management**.
2) Click **File** > **Access Server** > **New Datastore**.
3) Type the name of the datastore in the **Name** box.
4) Type a description in the **Description** box.
5) In the **Host Name** box, type the host name or the full IP address of the server where you have installed InfoSphere CDC.
6) In the **Port** box, type the port number which InfoSphere CDC uses for communication with the other components. For example, InfoSphere CDC for Informix uses by default port number 10901.
7) Ping the server. If successful, this returns the datastore properties including the type of server where you have installed InfoSphere CDC and the version number of the product.

c. Assign users to the datastores.

You need to assign the same users to both the solidDB datastore and the backend datastore.
1) Click **Access Manager** > **Datastore Management**.
2) Select a datastore.
3) Right-click and select **Assign User**.
4) Select a user or hold **Ctrl** to select multiple users.
5) Review the connection parameters. Click **OK** to accept the default connection parameters on the datastore or modify the parameters for the selected users.

For detailed instructions, see section *Setting up datastores* in the *InfoSphere Change Data Capture Management Console, Administration Guide*.

3. Optional: **Set system parameters on the solidDB and backend datastores.**

For a summary of Universal Cache specific system parameters, see 2.2.1, "Important InfoSphere CDC system parameter settings for Universal Cache," on page 46.

For detailed instructions on how to set system parameters, see section *Setting system parameters on source and target datastores* in the *InfoSphere Change Data Capture Management Console, Administration Guide*.

4. **Set up subscriptions.**

For detailed instructions on how to set up subscriptions using Management Console, see section *Setting up subscriptions* in the *InfoSphere Change Data Capture Management Console, Administration Guide*.

Tip: As an example, the following steps describe how to create subscriptions for bidirectional replication environment.

a. Create a new backend-to-solidDB subscription.
1) Click **Configuration** > **Subscriptions**.
2) Right-click anywhere in the **Subscriptions** field and select **New Subscription**.

3) Type the name of the new backend-to-solidDB subscription in the **Name** box.
4) Type the description of the new subscription in the **Description** box.
5) Select the backend datastore from the **Source** list.
6) Select the solidDB datastore from the **Target** list.
7) Click **OK**.

b. Create a new solidDB-to-backend subscription.
1) Click **Configuration** > **Subscriptions**.
2) Right-click anywhere in the **Subscriptions** field and select **New Subscription**.
3) Type the name of the new solidDB-to-backend subscription in the **Name** box.
4) Type the description of the new subscription in the **Description** box.
5) Select the solidDB datastore from the **Source** list.
6) Select the backend datastore from the **Target** list.
7) Click **OK**.

5. **Map tables for replication in all subscriptions.** This procedure assumes the backend data server contains the tables you want to cache into the solidDB database.
a. Click **Configuration** > **Subscriptions**.
b. Select the backend-to-solidDB subscription, right-click and select **Map Tables**.
c. Select **Multiple One-to-One Mappings** and click **Next**.
d. Expand the database, schema, or table from the **Source Tables** list to view tables from your database that are available for mapping. Right-click the database user or schema and click **Refresh** if you do not see your table listed.
e. Enable one or more tables to map from the **Source Tables** list.
f. Click **Next**.
g. Click **Create new target tables**.
h. Click **Next**.
i. Specify a target owner for each source owner.
j. Specify how the new target table names relate to their corresponding source table names.
k. Click **Next**.
l. Set the replication method to **Mirror (Change Data Capture)**.
m. Verify the mappings in the **Complete Mappings** dialog, and click **Next**.
n. Review the mapping summary and click **Finish**.

For detailed instructions, see section *Mapping tables* in the *InfoSphere Change Data Capture Management Console, Administration Guide*.

6. Optional: **For each table mapping, set conflict detection and resolution in accordance with your business rules.**
a. Click **Configuration** > **Subscriptions**.
b. Select the subscription.
c. Click the **Table Mappings** view and select the table mapping from the **Source Table** column.
d. Right-click and select **Open Details....**

e. Click the **Conflicts** tab.

f. Select the columns on which you want to detect conflicts.

g. Select the conflict resolution from the **Conflict Resolution Method** list.

h. Click **Save**.

For detailed instructions, see section *Setting conflict detection and resolution* in the *InfoSphere Change Data Capture Management Console, Administration Guide*.

7. Optional: **Set character set conversions for source columns.**

   If your solidDB database mode is Unicode (**`General.InternalCharEncoding=UTF8`**), set the encoding of character data type columns (CHAR, VARCHAR, and so on) to UTF-8.

8. **Start replication on subscriptions.** To start caching, start continuous mirroring on the subscriptions you have created.

   a. Click **Monitoring** > **Subscriptions**.

   b. Right-click on the two subscriptions and select **Start Mirroring**.

   c. Select **Continuous** and click **OK** to start mirroring.

   For detailed instructions, see section *Starting and ending replication on subscriptions* in the *InfoSphere Change Data Capture Management Console, Administration Guide*.

Results

You have set up, for example, bidirectional replication subscriptions between the backend and solidDB databases. As you make changes in either database, InfoSphere CDC replication mechanism takes care of replicating the changes to the other database.

For example, you can use solidDB SQL Editor (**`solsql`**) to issue SQL statements in the solidDB server. The InfoSphere CDC components will then take care of replicating the changes to the backend database.

What to do next

- For general instructions on how to administer the datastores and subscriptions, see the *InfoSphere Change Data Capture Management Console, Administration Guide*.
- For instructions on Universal Cache specific settings and administration tasks, see 2.2, “Universal-Cache-specific settings and tasks for InfoSphere CDC,” on page 46.
- For solidDB-specific instructions on how to optimize and monitor the performance of solidDB Universal Cache, see 4, “Performance tuning and monitoring,” on page 55.

2.1.1 Key concepts for setting up caching with Management Console

Setting up caching with the InfoSphere CDC Management Console requires the implementation of replication subscriptions between the cache and the backend database.

A *subscription* defines the replication direction and various replication rules. The subscriptions also maintain the state of replication, indicating whether or not replication is in progress.

The application and deployment needs dictate the direction of the subscriptions between a *source* and *target* datastore. In the InfoSphere CDC replication solution, a *datastore* is the representation of a database and the related InfoSphere CDC instance.

The cache and backend can act as both source and target data stores in different subscriptions. There can also be several subscriptions between two data stores; multiple subscriptions can be used to partition the data and workload.

Data stores and subscriptions are created and managed with the Management Console or the **dmcreatedatastore** and **dmsubscriptionmanager** command-line tools.

2.1.2 Deciding on the replication model

Before you create subscriptions, define your desired replication principles in accordance with your business rules.

Your replication model depends on the following two aspects:

- Ownership of data

  Does the master copy of the data reside in the backend database, as is typically the case, or does the master copy of the data reside in the cache?
- Read-only or read-write cache

  Do you want changes made to the cache to be reflected in the backend database, or is the cache read-only?

Typically, the backend database represents the master copy of the data and data must be cached in read-only mode. For such setups, only a single subscription is required. The backend datastore should be used as the subscription source and the cache data store (solidDB) should be used as the subscription target. This configuration ensures that any changes made to the backend can be replicated to the cache.

Typical subscription configurations

The following table shows the typical subscription configurations for Universal Cache. The **Procedure** column contains instructions on the type of subscriptions you should create in each case. The **Procedure** column also states the necessary conflict resolution option needed to prevent recursion.

Table 22. Typical subscription configurations

Cache type	Behavior	Procedure
Backend owned, read-only cache	Changes made to backend database are reflected in cache (most typical scenario)	1. Create a single subscription using the backend datastore as source and the cache datastore as target.
Backend owned, read-write cache	Changes made to backend database are reflected in cache; changes made to cache are reflected in backend	1. Create a subscription using the backend data store as source and the cache data store as target. 2. Specify **SOURCE wins** as conflict resolution option. 3. Create another subscription using the cache data store as source and the backend data store as target. 4. Specify **TARGET wins** as conflict resolution option.

Table 22. Typical subscription configurations (continued)

Cache type	Behavior	Procedure
Cache owned, archival	Changes made to cache are archived to backend.	1. Create a single subscription using the cache datastore as source and the backend datastore as target.
Cache owned, read-write cache	Changes made to backend database are reflected in cache; changes made to cache are reflected in backend.	1. Create a subscription using the cache datastore as source and the backend datastore as target. 2. Specify **SOURCE wins** as conflict resolution option. 3. Create another subscription using the backend data store as source and the cache data store as target. 4. Specify **TARGET wins** as conflict resolution option.

2.2 Universal-Cache-specific settings and tasks for InfoSphere CDC

This section provides instructions that are specific to using InfoSphere CDC technology with Universal Cache.

General instructions on administering the InfoSphere CDC instances and replication subscriptions are available in the IBM InfoSphere Change Data Capture version 6.5 Information Center.

2.2.1 Important InfoSphere CDC system parameter settings for Universal Cache

Depending on your backend data server and database setup, you might need to modify InfoSphere CDC system parameter settings. In addition to the generic parameter settings that apply to any configuration, there are parameters that are specific to Universal Cache use only.

Table 23. InfoSphere CDC system parameter settings specific to Universal Cache usage

Component	System parameter	When to modify
InfoSphere CDC for solidDB	`refresh_with_referential_integrity`	If your subscriptions include tables with foreign keys, set the `refresh_with_referential_integrity` system parameter in InfoSphere CDC for solidDB to 'true'. For more information, see 2.2.2, “Enabling use of foreign keys (referential integrity),” on page 47.
	`retrieve_credentials`	If you are using SQL passthrough and your backend data server is DB2 for iSeries or DB2 for z/OS, set the `retrieve_credentials` in InfoSphere CDC for solidDB to 'false'.

	`solid_fast_refresh_on`
	
	`solid_fast_refresh_apply_pipes`
		If you want to enable the fast refresh for subscriptions where solidDB is the source datastore, set the `solid_fast_refresh_on` to 'true', and the `solid_fast_refresh_apply_pipes` to match the number of processors (cores) in your system (default is 2).
		
		For more information, see 2.2.6, “Enabling fast refresh,” on page 49.

Table 23. InfoSphere CDC system parameter settings specific to Universal Cache usage (continued)

Component	System parameter	When to modify
InfoSphere CDC for DB2 Linux, UNIX and Windows	**refresh_allow_fast_loader**	If your subscriptions include tables with foreign keys and your backend data server is DB2 for Linux, UNIX, and Windows, set the **refresh_allow_fast_loader** system parameter in InfoSphere CDC for DB2 to 'false'. For more information, see 2.2.2, "Enabling use of foreign keys (referential integrity)."
	ddl_awareness	If you are using bidirectional replication and your backend data server is DB2 for Linux, UNIX, and Windows, set the **ddl_awareness** system parameter in InfoSphere CDC for DB2 to 'false'.
InfoSphere CDC for Oracle	**refresh_allow_fast_loader** **ts_fast_loader_disable_constraint**	If your subscriptions include tables with foreign keys and your backend data server is Oracle, set the **refresh_allow_fast_loader** and **ts_fast_loader_disable_constraint** system parameters in InfoSphere CDC for Oracle to 'false'. For more information, see 2.2.2, "Enabling use of foreign keys (referential integrity)."

2.2.2 Enabling use of foreign keys (referential integrity)

If you have set up subscriptions with Management Console and the subscriptions include tables with foreign keys, you need to set the InfoSphere CDC for solidDB system parameter **refresh_with_referential_integrity** to `true`. Additionally, if your backend data server is Oracle or DB2 for Linux, UNIX, and Windows, you need to disable the fast loader.

About this task

You can set system parameters

- with the `dmset -I <INSTANCE_NAME> <parameter_name>=<parameter_value>` command, or
- using the Management Console:
  1. In the **Configuration** perspective of the Management Console, select the datastore.
  2. Right-click on the datastore and select **Properties** > **System Parameters**.

If you make changes to a system parameter during active replication, you must stop and restart replication for the changes to take effect.

Procedure

1. Set the InfoSphere CDC for solidDB system parameter **refresh_with_referential_integrity** to `true`

   For example:

   ```
   dmset -I solidDB_1 refresh_with_referential_integrity=true
   ```

2. If your backend data server is DB2 for Linux, UNIX, and Windows, set the **refresh_allow_fast_loader** system parameter in InfoSphere CDC for DB2 to `false`.

   For example:

   ```
   dmset -I DB2_1 refresh_allow_fast_loader=false
   ```

3. If your backend data server is Oracle, set the **refresh_allow_fast_loader** and **ts_fast_loader_disable_constraint** system parameters in InfoSphere CDC for Oracle to `false`.

   For example:

   ```
   dmset -I Oracle_1 refresh_allow_fast_loader=false
   dmset -I Oracle_1 ts_fast_loader_disable_constraint=false
   ```

2.2.3 Dropping and re-creating solidDB source tables

If you need to drop and re-create tables in subscriptions where the solidDB database is the source datastore, you need to reconfigure the table mappings.

Procedure

1. Stop the replication on the subscription where the solidDB server is the source datastore.
2. Remap the source tables.
3. Restart the replication (mirroring) on the subscription.

   For instruction how to map tables and start and stop subscriptions, see *Management Console administration* in the IBM InfoSphere Change Data Capture version 6.5 Information Center.

2.2.4 Starting mirroring in Management Console without synchronizing data

When you start mirroring on a subscription, all tables with a replication method of **Mirror** and a status of **Refresh** are initially refreshed on the subscription. This synchronizes your source and target tables. If you want to start mirroring without a refresh, you can do this by setting manually a capture point from which the mirroring will start. This may be useful, for example, when you know that your frontend and backend databases are synchronized already. If your subscription contains a lot of data, starting mirroring without a refresh can save time.

Procedure

1. Ensure you have ended any active replication on the subscription that contains the source table.
2. Mark the table capture point using Management Console or the `dmmarktablecapturepoint` command.
   - For instruction on how to mark the table capture point using Management Console, see section *Marking a table capture point on a source table* in the *InfoSphere Change Data Capture Management Console, Administration Guide*.
   - For instruction how to use the `dmmarktablecapturepoint` command, see section "dmmarktablecapturepoint - Mark a table capture point on a source table" on page 133.

2.2.5 Using Unicode and partial Unicode databases with Universal Cache

Depending on your solidDB database mode (Unicode or partial Unicode), you might need to specify the encoding of character data type columns (CHAR, VARCHAR, and so on).

About this task

- If your solidDB database mode is *Unicode* (**General.InternalCharEncoding=UTF8**), set the encoding of solidDB character data type columns (CHAR, VARCHAR, and so on) to UTF-8.
- If your solidDB database mode is *partial Unicode* (**General.InternalCharEncoding=Raw**), set the encoding of solidDB character data type columns (CHAR, VARCHAR, and so on) to the encoding used in the application environment.

  Important: By default, the encoding of character data type columns is set to ISOLatin1. If your application uses Latin1 encoding, you do not need to set the encoding explicitly.

Procedure

1. In the Management Console, click **Configuration** > **Subscriptions**.
2. Select the subscription.
3. Click the **Table Mappings** view and select the table mapping.
4. Right-click and select **Edit Mapping Details**.
5. Click the **Translation** tab.
6. Select the character data type (CHAR, VARCHAR, and so on) source column. This enables the **Encoding Conversion** area.
7. Select the character encoding from the **Source** list.
   - Unicode databases: UTF-8
   - Partial Unicode databases: application encoding
8. Select the character encoding to which you want to convert from the **Target** list. For example, your backend data server might be storing the character data types in the UCS-2 big-endian form.
9. Click **Apply**.
10. Repeat the above steps for all subscriptions where solidDB database is the source or target datastore.

Results

When you start replication on the subscription, InfoSphere CDC converts the character encoding in the source column to the encoding you specified and populates the mapped target column with data in the new encoding.

2.2.6 Enabling fast refresh

The fast refresh feature reduces the amount of time needed to replicate a large amount of data from the backend data server to the solidDB server. The fast refresh is enabled by setting the InfoSphere CDC for solidDB system parameter **solid_fast_refresh_on** to 'true'. For further performance improvements, set the **solid_fast_refresh_apply_pipes** system parameter to match the number of processors (cores) in your system.

Before you begin

The fast refresh feature is applicable only in subscriptions where solidDB database is the target datastore.

Most performance gains are achieved with simple setups; factors such as the amount of data per row per table, code page conversions, and column mappings will effect the performance of the fast refresh feature.

Fast refresh does not support the following InfoSphere CDC functionality:

- Conflict detection
- Summarization
- Row consolidation
- Adaptive apply
- User exits

About this task

You can set system parameters

- with the `dmset -I <INSTANCE_NAME> <parameter_name>=<parameter_value>` command, or
- using the Management Console:
  1. In the **Configuration** perspective of the Management Console, select the datastore.
  2. Right-click on the datastore and select **Properties** > **System Parameters**.

If you make changes to a system parameter during active replication, you must stop and restart replication for the changes to take effect.

Procedure

1. Set the InfoSphere CDC for solidDB system parameter **`solid_fast_refresh_on`** to `true` (default is `false`).

   For example:

   ```
   dmset -I solidDB_1 solid_fast_refresh_on=true
   ```

2. Set the InfoSphere CDC for solidDB system parameter **`solid_fast_refresh_apply_pipes`** to match the number of processors (cores) in your system (default is 2).

   For example:

   ```
   dmset -I solidDB_1 solid_fast_refresh_apply_pipes=4
   ```

2.2.7 Using shared memory access (SMA) with Universal Cache

To use SMA with Universal Cache, you need to start a SMA server and enable a local SMA connection between the InfoSphere CDC for solidDB instance and the SMA server.

Before you begin

For SMA connections, the solidDB server and the InfoSphere CDC replication engine must be located on the same node.

Procedure

1. Check that the location of the SMA driver library is included in the LD_LIBRARY_PATH or LIBPATH (Linux and UNIX) or PATH (Windows) environment variable.

For details, see *Configuring your environment for SMA use with Java* in the *IBM solidDB Shared Memory Access and Linked Library Access User Guide*.

2. Create a symbolic link for the SMA driver library (`ssolidsma70`), without the file type extension, in the `<solidDB installation directory>/bin` directory.

   For example in Linux operating systems, use the following command:

   ```
   ln -s ssolidsma70.so ssolidsma70
   ```

3. Start the SMA server by entering the command `solidsma` at the command prompt.
4. Configure the InfoSphere CDC for solidDB instance to use an SMA connection when connecting to the solidDB server.

   Use the InfoSphere CDC for solidDB configuration tool (`dmconfigurets`) to enable the SMA connection.

Table 24. Enabling SMA connections with `dmconfigurets`

Operating system	**To enable SMA connections with `dmconfigurets`**
Linux and UNIX	1. Select the **Single server** configuration type. 2. For the **Enable SMA** option, type `y` and press **Enter**.
Windows	In the **Server** area of the **New instance** or **Edit instance** dialog, select the **Enable SMA** check box.

When you select **Enable SMA**, the solidDB connection property `solid_shared_memory=yes` is added to the connect string.

Related concepts:

10.3, “Configuring InfoSphere CDC,” on page 107

3 Preparing applications for use with Universal Cache

At minimum, you need to make your application to connect to the Universal Cache system using the solidDB JDBC Driver or solidDB ODBC Driver, instead of (or in addition to) connecting to the backend driver.

Additionally, to minimize the changes needed in the application, you can create mappings of backend-specific SQL statements and error messages against the solidDB statements and error messages.

Integrating an existing application to work with Universal Cache

Conceptually the migration to Universal Cache can mean that an existing enterprise data server is simply replaced with a cache database that sits between the backend database and application, making the database appear faster from an application perspective. There are no changes in the database interface layer.

In reality, the conversion from single database system to a cache database system is likely to require changes in the application. For example, the following considerations might require code changes:

- The application must be aware of the properties of two database connections, one to the cache database and the another to the backend database.

  The SQL passthrough functionality can mask the two connections to one ODBC or JDBC connection but will require awareness of the two databases in error processing.
- Queries and transactions combining data from the cache and backend databases are not supported. A combination of backend and cache database is not fully transactional although both individual components are transactional databases.

  However, creating a transactional combination of two or more databases is possible using Distributed Transactions. A Distributed Transaction is a set of database operations where two or more database servers are involved. The database servers provide transactional resources. Additionally, a Transaction Manager is required to create and manage the global transaction that runs on all databases. solidDB supports the standard Java Transaction API (JTA), through providing a set of XA classes. JTA methods enable the Transaction Manager to control solidDB as one of the transactional resources in a global transaction.
- SQL compatibility between the solidDB server and the backend data server might be limited.

  Generally, the applications that have been implemented directly using JDBC or ODBC APIs, or a middleware running on top of those APIs, might require no conversion at all. If no extensions to the SQL Standard are used, the applications are expected to work with minor modifications.

  Because stored procedure languages are not compatible with each other, a rewrite for stored procedures will be required if they are used in the application. This can be automated to some level but a separate project is necessary for stored procedure conversion.

  If you use APIs, access methods, or programming paradigms that are not supported by solidDB (such as embedded SQL), and there is no ODBC or JDBC-based middleware available to act as a gateway, you need to rewrite parts of the application.

Related concepts:

5, “SQL passthrough,” on page 59
The SQL passthrough feature enables applications to access data both in the frontend and backend data servers with a single connection. For example, SQL passthrough can be enabled in such a way that those SQL statements that cannot be executed in the solidDB frontend server are passed over to the backend. The SQL passthrough mode can be set per session or per transaction. By default, SQL passthrough is not enabled.

4 Performance tuning and monitoring

The performance of Universal Cache depends on a number of system and setup considerations. The Management Console monitoring features and the solidDB performance counters provide means for monitoring and analyzing the performance level.

4.1 Factors impacting Universal Cache performance

You can improve Universal Cache performance, for example, by optimizing your system and network setup or introducing parallelism to your configuration setup.

The guidelines given in this chapter complement the tuning information specific to solidDB (see *IBM solidDB Administrator Guide*) and your backend data server.

Optimizing system and network setup

- The amount of memory allocated to each InfoSphere CDC instance should be at least 256 MB.

  The memory allocation for the InfoSphere CDC instance is defined when creating the instance with the Configuration Tool (option **Maximum Memory Allowed**). The minimum allocation is 64 MB; the default is 512 MB for a 32 bit instance and 1024 MB for a 64 bit instance.
- Available processor capacity

  Processing data with InfoSphere CDC is processor intensive; ensure that you have enough processor capacity available in all the nodes included in your Universal Cache setup.
- Network latency and throughput

  Optimizing your network for high throughput and low latency can improve Universal Cache performance.

Improving performance with parallelism

In a typical setup, the InfoSphere CDC replication engine sets the following limitations to the performance:

- At the source end, the performance is limited to about total of 15 000 operations per second for all the subscriptions together.
- At the target end, the performance is limited to about 5 000 operations per second per subscription.

The performance can be improved by using multiple subscriptions or multiple solidDB frontends to partition the data and workload. This is because multiple subscriptions are processed in parallel at both the frontend and the backend.

For example, you can create separate subscriptions for autonomous tables that are not referencing or referenced outside the subscription. You can also place such tables/subscriptions in separate solidDB frontends.

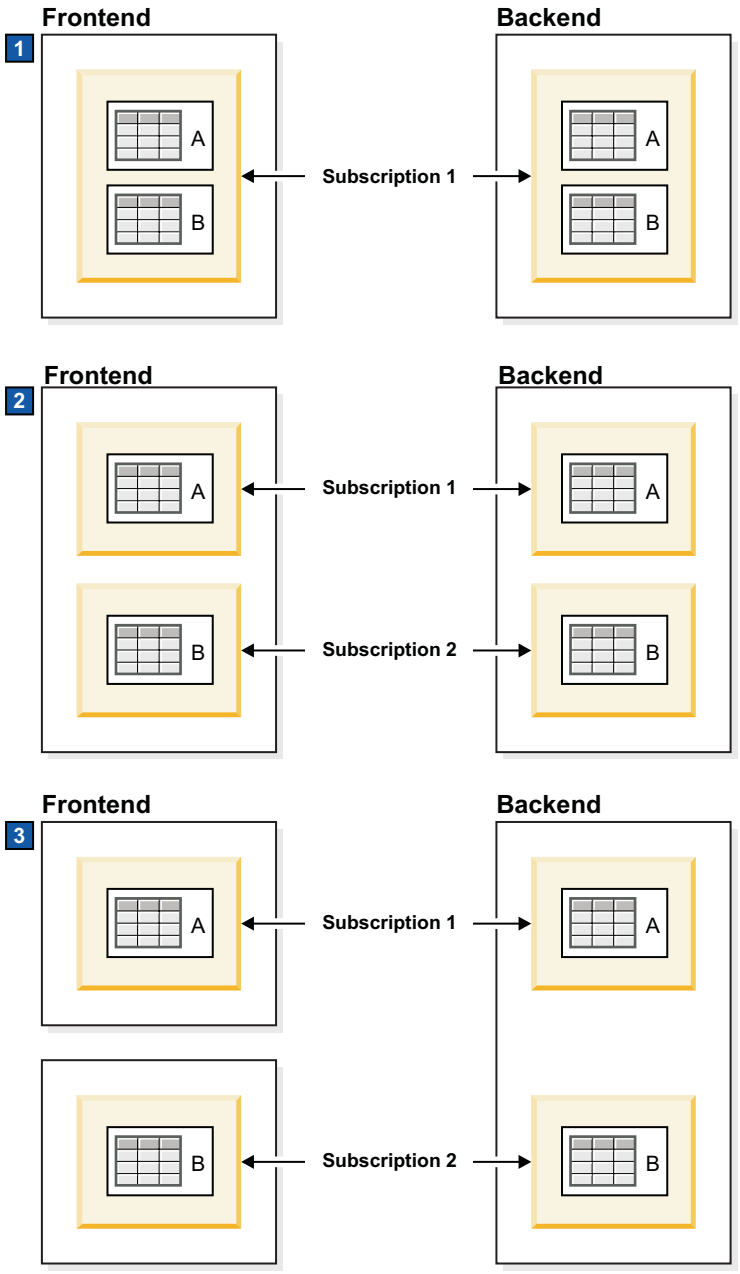

Figure 7. Example: Universal Cache setup with three partitioning models

1. Single solidDB front end with one subscription; no partitioning
2. Single solidDB front end with two subscriptions; partitioning with two parallel subscriptions
3. Two solidDB front ends with one subscription each; partitioning with two front ends

Optimizing log reading behavior

- Throttling

  If the replication cannot accommodate a sustained load in the solidDB server, the processing is throttled down (slowed). From the application standpoint, that results in increased response times. The replication traffic is buffered so that load

bursts can be accommodated. The size of the corresponding in-memory buffer is controlled with the configuration parameter **LogReader.MaxSpace**.

As of V7.0 Fix Pack 3, you can also disable throttling by setting the **LogReader.UseThrottling** parameter to `no`.

- Offline operation and log overflow

  If the replication is stopped or has failed, the solidDB server might continue to process the load, accumulating the data for a later transfer. The limit of the accumulated data is set with the **LogReader.MaxLogSize** configuration parameter. When the amount of accumulated data exceeds the values of the **LogReader.MaxLogSize** parameter, the log overflow occurs, resulting in a situation where replication catchup is no longer possible. If catchup is no longer possible, you need to refresh the subscriptions.

Other considerations

- The target database must be capable of bearing the load generated by the InfoSphere CDC replication engine.
- Any processing of data may introduce bottlenecks, for example:
  - Row level filtering
  - Data transformations and expressions
  - Codepage conversion

Related reference:

Appendix A, "Log Reader parameters," on page 175
The Log Reader parameters appear in the `[LogReader]` section of the client-side `solid.ini` configuration file.

4.2 Monitoring performance

The Management Console **Monitoring** and **Statistics** views can be used for collecting performance statistics on the subscriptions. The solidDB performance counters provide performance data on the solidDB cache database.

Monitoring performance in the Management Console

The Management Console can collect statistics on latency, throughput, and the number and size of the replication operations. The statistics can be viewed in Management Console or saved and exported in .csv format. You can also set latency notifications and thresholds.

For detailed instructions on how to use the monitoring and statistics in Management Console, see Monitoring subscriptions in the IBM InfoSphere Change Data Capture version 6.5 Information Center.

Monitoring solidDB frontend performance

solidDB provides a number of performance counters that are specific to the use of the solidDB server with the InfoSphere CDC technology:

- Counters with the variable name starting with *Logreader*

  For example, *Logreader commits sent* tracks the number of commits sent to the InfoSphere CDC instance per second.

- *TS applied transactions*

  The *TS applied transactions* counter tracks the number of transactions applied into the server by InfoSphere CDC instance when solidDB is a target datastore.

For a detailed list of the solidDB performance counters and how to use them, see Performance counters (perfmon).

For a detailed list of the solidDB performance counters and how to use them, see section *Monitoring solidDB* in the *IBM solidDB Administrator Guide*.

5 SQL passthrough

The SQL passthrough feature enables applications to access data both in the frontend and backend data servers with a single connection. For example, SQL passthrough can be enabled in such a way that those SQL statements that cannot be executed in the solidDB frontend server are passed over to the backend. The SQL passthrough mode can be set per session or per transaction. By default, SQL passthrough is not enabled.

The connection between the frontend and backend is made with a backend compatible ODBC driver which is loaded dynamically in the solidDB server. The solidDB server uses this driver to execute the passthrough statements directly in the backend data server.

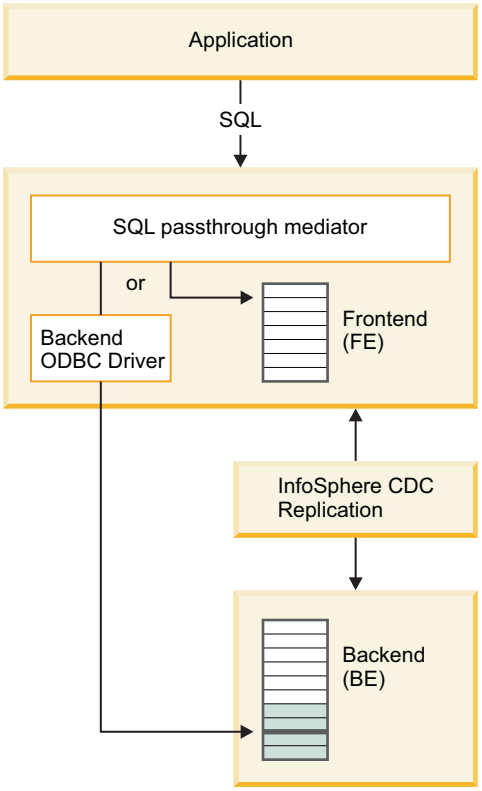

Figure 8. SQL passthrough

5.1 Principles of operation

A layer in the solidDB server called *SQL passthrough mediator* is responsible for handling the passthrough of SQL statements to the backend, according to a chosen passthrough mode. The SQL passthrough mode can be changed dynamically at runtime. The access to the backend server is facilitated with a backend ODBC driver that is linked with the solidDB server. The login data (username and password) for the backend is transferred through the InfoSphere CDC components.

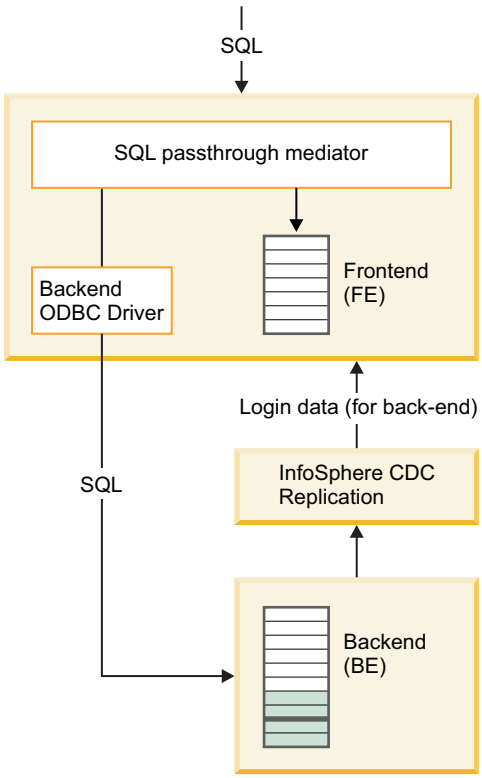

Figure 9. SQL passthrough architecture

Passthrough modes

The passthrough mode defines how read and write statements are be passed to the backend. The passthrough mode is set separately for read statements (SELECT) and write statements (non-read statements including INSERT, UPDATE, DELETE). Three passthrough modes are available:

- FORCE: all read or write statements are passed to the backend.
- NONE (default): read or write statements are never passed to the backend.
- CONDITIONAL: if the statement results in, for example, a missing table error or a syntax error, the statement is passed to the back end.

The conditional passthrough mode is based on a logic utilizing error messages:

- A missing table error or a syntax error invokes passthrough (prepare phase).

  If the passthrough mode is changed after a statement has been prepared, it is re-prepared at a new location if needed.
- Privilege violation errors do not invoke passthrough.
- The errors that occur in the execution phase do not invoke passthrough. For example, if a write statement fails on an integrity constraint violation, it is not passed through.

Additionally, complexity of the SQL statements can be used for defining that long-running statements are always passed through to the backend.

Note: SQL passthrough does not use any information about the data ranges or population of the frontend and backend databases. Specifically, if a query executes

successfully in the frontend but returns no data (or little data), it is not redirected to the backend even though the data might be there.

Transactions and isolation levels

The transaction model for SQL passthrough is designed to preserve consistency of the backend database; the backend transactions can be made to meet the highest isolation levels (REPEATABLE READ or SERIALIZABLE). To preserve consistency of the backend database when using SQL passthrough, the isolation level of the frontend is set to the same (or similar) or higher level than in the backend.

In general, individual transactions execute and commit fully either in the frontend or backend. As local transactions, they preserve the database consistency, given the limitations of the desired isolation levels. However, consecutive transactions can be temporally mutually inconsistent because of the delay induced by the asynchronous replication from the backend to the frontend. For example, a transaction may not see the replicated results of the previous transaction, if the latter executed a passed-through write operation on the backend database.

Only whole statements are passed through, no statement can span both the frontend and back end. This means that distributed queries are not possible.

In certain cases, a transaction reads from either the frontend or the backend database and writes to the other one. Such a transaction may be considered to be composed of two sub-transactions. In such cases, the transaction is committed only if the write subtransaction is committed successfully. If the commit of the write subtransaction fails, the overall transaction fails also.

The execution of transactions depends on the isolation level:

- Execution rules at READ COMMITTED isolation level
  - A transaction can always read from the backend regardless of where it writes to.
  - A transaction can always read from the frontend regardless of where it writes to.
  - A transaction can write only either to the frontend or the back end.
- Execution rules at the REPEATABLE READ (or higher) isolation level
  - A transaction can always read from backend regardless of where it writes to.
  - If a transaction reads from the frontend, it must also write to the frontend.
  - A transaction can only write either to the frontend or backend.

  If a transaction violates any of the above rules, solidDB returns error 13455 at the return of the violating statement.

Accessing data in the backend

The connection between the frontend and backend is made with a backend ODBC driver which is installed on the solidDB node and loaded dynamically in solidDB server. solidDB server uses this driver to execute the passthrough statements directly in the backend data server.

In most cases, the InfoSphere CDC technology is used for transferring backend login data to the frontend. When mirroring or a refresh is started on the first subscription from solidDB server to the backend data server, the InfoSphere CDC

for solidDB instance retrieves the login data from the backend InfoSphere CDC instance and stores it in the solidDB system table SYS_SERVER with the statement `CREATE REMOTE SERVER`. The password stored in the SYS_SERVER table is hidden.

The InfoSphere CDC technology does not transfer the backend login data in the cases described below.

- If the InfoSphere CDC instance for the backend is running under a user ID that automatically has access to the database, the login data does not need to be stored.
- If the backend data server is DB2 for z/OS or DB2 for iSeries, the login data cannot be fetched. To avoid errors, you need to set the InfoSphere CDC for solidDB system parameter **`retrieve_credentials`** to `FALSE`.
- If you have upgraded your InfoSphere CDC for solidDB installation and subscription from V6.3, the 6.3 version of InfoSphere CDC for solidDB has not stored the backend login data in the SYS_SERVER table.

In the latter two cases, use the `CREATE REMOTE SERVER` (or `ALTER REMOTE SERVER`) statement to define the login data manually.

5.2 Considerations for developing applications with SQL passthrough

Access rights

- The passthrough feature is available only for validated users. The validation mechanism is based on the `GRANT PASSTHROUGH` statement.

  In new databases, the administrator has the passthrough validation.
- The user access to a table may be restricted. If there is a statement that is restricted by the privilege restrictions in the frontend database, no corresponding statement is passed over to the backend database, even though there would be no privilege restrictions in the backend database.

SQL statements

- All SQL statements can be passed to the backend, except for the `SET [TRANSACTION] PASSTHROUGH` itself, the `SET [TRANSACTION] ISOLATION LEVEL` statement, and the data aging related statement `SET DELETE CAPTURE`. The following restrictions apply also:
  - In SELECT statements, only forward cursors are supported.
  - UPDATE/DELETE ... WHERE CURRENT OF statements are not supported and may cause unexpected results if used.
  - An error is returned if anything else than next row is fetched.
  - Database-wide metadata queries are always executed in the frontend.

    For example, even if the SQL passthrough mode is set to force, queries such as SELECT * FROM TABLES or JDBC function calls such as getTables return information about the tables in the solidDB database.

    Statement-specific metadata queries (for example, ODBC SQLColAttr()) use the data received from the solidDB database when possible, and otherwise from the backend database.
- If an SQL statement can be executed on both databases, the table definitions (for example, column types) are always taken from the solidDB database. If the solidDB and backend definitions are different, the data between the frontend and backend is converted when possible. The number of columns and column names must match.

- If the solidDB server participates in distributed transactions using the Java Transaction API (JTA) interface, only read statements (SELECT) are passed through.
- You can define that complex SQL statements are always passed through to the backend. Complex queries might be executed more effectively in the backend. The level of the complexity at which a statement is passed through is defined with the following parameters:
  - **Passthrough.ComplexNumTables** – specifies the minimum number of tables in a complex statement. If a statement has less tables than specified with this parameter, the statement is not complex and it is not passed through to the backend.
  - **Passthrough.ComplexNumNonindexedConstr** – specifies the minimum number of non-indexed WHERE clause constraints in a complex statement. If a statement has less non-indexed constraints of the following type, the statement is not complex and it is not passed through to the backend: the WHERE clause constraint does not resolve with index, the index does not exist, or the optimizer chooses different index for constraint.
  - **Passthrough.ComplexNumOrderedRows** – specifies the minimum estimated number of rows which must be sorted in a complex statement. If a statement has less than the estimated number of sortable rows, the statement is not complex and it is not passed through to the backend.

  The factory value for all three parameters is 0 (zero), which means that the given property is not used when estimating if the statement is complex.

Data types and column binding

- SQL passthrough supports all standard SQL standard data types that are supported by the solidDB server. For more details, see Appendix C, "ODBC data type support in SQL passthrough," on page 181.
- At the application-side driver, the column binding is based on standard ODBC binding methods.

Code page support

- The code page support depends on the solidDB database mode:
  - If the solidDB database mode is Unicode, SQL passthrough supports use of different code pages in the frontend and backend without any loss of information.
  - If the solidDB database mode is partial Unicode, only Latin-1 (or ASCII, a subset of Latin-1) code pages are supported.

  If your data uses encoding outside the Latin-1 character set, use the solidDB database in the Unicode mode.
- If your solidDB database mode is Unicode (**General.InternalCharEncoding=UTF8**), you need to set the backend ODBC driver to expect data from solidDB in UTF-8 encoding. This is because in Unicode mode, character data types are stored in solidDB in UTF-8 encoding.

  In Unicode mode environments, the backend ODBC driver handles the conversion between the UTF-8 encoding in the solidDB database and the encoding in the backend. At the application, any binding method available can be used because the conversion between the application and the solidDB frontend encoding is handled by the solidDB ODBC or JDBC driver, as described in section *Using Unicode* in the *IBM solidDB Programmer Guide*.
- If the solidDB database mode is *partial Unicode* (**General.InternalCharEncoding=Raw**) and the application and solidDB

environments use ASCII or Latin-1 encoding (Western languages), the backend ODBC driver is likely to handle the character translations correctly without setting any code page support in the ODBC driver explicitly.

In partial Unicode mode, character data types are stored in the solidDB database in raw (binary) format, without any conversion between the application encoding and the solidDB internal representation — the assumption is that applications are aware of this and handle the conversion as necessary.

Tip: By default, the installation of the IBM Data Server Driver for ODBC and CLI sets the driver to use the system locale of the installation node.

If your backend database uses encoding other than ASCII or Latin-1, you need to set the backend ODBC driver to expect data from the solidDB server in ASCII or Latin-1 encoding.

SQL passthrough support in solidDB tools

- solidDB SQL Editor (**solsql**) is fully supported with SQL passthrough.
- The other solidDB tools are not supported, they can only be used with the frontend.

Error codes

- Errors from the frontend are always native solidDB error codes.
- Errors from the backend are preceded with SQLSTATE, showing the native backend error code and text.
- The backend native error codes can be mapped against solidDB error codes using a mapping file. The mapping file is defined with the **Passthrough.ErrorMapFileName** parameter.

5.3 Configuring and using SQL passthrough

The configuration steps for SQL passthrough include setting solidDB configuration parameters and installing and configuring a backend compatible ODBC driver on the solidDB frontend node. After you have configured SQL passthrough, you can enable and disable it dynamically.

The solidDB server also offers tracing and monitoring functionality for collecting data on the SQL passthrough connection types and statement activities.

5.3.1 Setting up SQL passthrough

The configuration procedure for SQL passthrough depends on the backend and the type of ODBC connection you want to use.

Before you begin

Before enabling SQL passthrough, you should have your Universal Cache environment up and running.

1. Install theUniversal Cache components.
2. Configure the frontend and backend InfoSphere CDC instances.
3. Define at least one subscription with at least one table mapping from the frontend data server to the backend.

Procedure

1. Install and configure the backend ODBC driver for SQL passthrough.

- If your backend data server is an IBM data server, use the *IBM Data Server Driver for ODBC and CLI* provided with the solidDB Universal Cache installation images.
- If your backend data server is not an IBM data server, use the native ODBC driver provided with your backend data server.

2. Configure the default SQL passthrough settings for your system.

   For example, enable the SQL passthrough for your system using the **Passthrough.PassthroughEnabled=yes** parameter and define the default passthrough mode using the **Passthrough.SqlPassthroughRead** and **Passthrough.SqlPassthroughWrite** parameters.
3. Grant SQL passthrough rights to the appropriate users using the GRANT PASSTHROUGH statement.
4. Ensure login data for the backend data server is available.
   a. Connect the solidDB and backend datastores and start replication on a subscription where solidDB database is the source datastore and the backend database is the target datastore.
   b. Check that the solidDB system table SYS_SERVER contains the correct login data.
   - In most cases, when mirroring or a refresh is started on the first subscription from solidDB to the backend data server, the InfoSphere CDC for solidDB instance retrieves the login data from the backend InfoSphere CDC instance and stores it in the solidDB system table SYS_SERVER.
   - If the SYS_SERVER table contains incorrect or no login data, add or modify the login data manually.
5. Start your application.

Related reference:

2.2.1, "Important InfoSphere CDC system parameter settings for Universal Cache," on page 46
Depending on your backend data server and database setup, you might need to modify InfoSphere CDC system parameter settings. In addition to the generic parameter settings that apply to any configuration, there are parameters that are specific to Universal Cache use only.

Installing and configuring backend ODBC drivers for SQL passthrough

To use the SQL passthrough functionality, you must install and configure the backend ODBC driver on the solidDB frontend node. You can link to the driver directly (using a dynamic driver library) or using a driver manager.

Before you begin

Locate the ODBC driver installation package as well as installation and configuration instructions for your backend data server.

- If your backend data server is an IBM data server, use the *IBM Data Server Driver for ODBC and CLI* provided with the solidDB Universal Cache installation images.
- If your backend data server is not an IBM data server, use the native ODBC driver provided with your backend data server.

Procedure

1. **Install the backend ODBC driver (client) on the solidDB node.**
   - If your backend data server is an IBM data server, proceed as follows.

a. Copy the compressed file that contains the *IBM Data Server Driver for ODBC and CLI* onto the solidDB node from the installation image.
b. Uncompress that file into your chosen install directory on the solidDB node.
c. Optional: remove the compressed file.
d. If your frontend solidDB data server is running on AIX:
   1) Extract the shared library (`/odbc_cli/clidriver/lib/libdb2.a`) to yield `shr_64.o` on 64-bit operating systems. To avoid confusion, rename the file to `libdb2.so`.

      Issue the following commands:

      ```
      cd odbc_cli/clidriver/lib
      ar -x -X 64 libdb2.a
      mv shr_64.o libdb2.so
      ```

      These steps are necessary on AIX because solidDB will load the driver dynamically.

      Important: When referencing the driver library on AIX systems, remember to use the correct file name (`libdb2.so`).
   2) Set the DB2NOEXITLIST environment variable to ON.

      Issue the following command on the solidDB node:

      ```
      export DB2NOEXITLIST=ON
      ```

      This environment variable ensures that upon shutdown, the driver does not attempt to free resources that have already been freed by solidDB.

- If your backend data server is not an IBM data server, follow the instructions provided with your backend data server.

2. **Define the connection settings between the ODBC driver and the backend data server.**

   The backend ODBC driver for SQL passthrough is configured in the same way as you would configure a regular remote connection with your backend database. You can link to the driver directly or using a driver manager.

   - **Direct linking**

     Depending on your backend data server and operating system, you may need to set environmental variables or other setup parameters to enable direct linking.

     For details, see the examples below or the instructions that came with your backend data server.
   - **Driver manager**

     Depending on your backend data server, operating system, and driver manager, you need to configure such settings as data source name, login data, performance options, or connection options.

     For details, see the examples below or the instructions that came with your backend data server.

3. **Define the connection settings between solidDB server and the driver or driver manager by modifying the `[Passthrough]` section of the `solid.ini` configuration file.**

   The format of the parameter values depends on whether you link to the driver directly or using a driver manager.

   Direct linking

   - Use **`RemoteServerDriverPath`** to set the driver path.

- Use **RemoteServerDSN** to set the driver connect string.

  Note: The exact connect string depends on the driver.

  Example: *IBM Data Server Driver for CLI and ODBC* with DB2 or IDS on Linux operating systems

  ```
  [Passthrough]
  RemoteServerDriverPath=/home/solid/odbc_cli/clidriver/lib/libdb2.so
  RemoteServerDSN="Driver={IBM DB2 ODBC DRIVER};Dat
     abase=my_ids;Hostname=9.212.253.10;Port=9088;protocol=TCPIP;"
  ```

  Driver manager

- Use **RemoteServerDriverPath** to set the driver manager path.
- Use **RemoteServerDSN** to set the Data Source Name.

  Example: unixODBC DriverManager with DB2

  ```
  [Passthrough]
  RemoteServerDriverPath=/usr/lib/libodbc.so
  RemoteServerDSN=BE_DB2
  ```

4. **Configure the ODBC driver's code page support according to the solidDB database mode (Unicode or partial Unicode).**

   - **Unicode databases**

     If your solidDB database mode is Unicode (**General.InternalCharEncoding=UTF8**), configure the ODBC driver to expect data from solidDB in UTF-8 encoding.

     The procedure for configuring UTF-8 support depends on the driver. For details, see the instructions that came with your backend data server.

     For example, in DB2 for Linux, UNIX, and Windows environments, the UTF-8 support is configured by setting the environment variable DB2CODEPAGE to 1208. (The value 1208 is the identifier for the UTF-8 code page in DB2 environments.)

   - **Partial Unicode databases**

     – If your solidDB database mode is partial Unicode (**General.InternalCharEncoding=Raw**) and your application and solidDB environments use ASCII or Latin-1 encoding (Western languages), the backend ODBC driver is likely to handle the character translations correctly without setting any code page support in the ODBC driver explicitly.

       For example, if you have installed *IBM Data Server Driver for ODBC and CLI* in a system using ASCII encoding, the installation has set the driver to use the system locale of the installation node automatically.

     – If your backend database uses encoding other that ASCII or Latin-1, set the backend ODBC driver to expect data from solidDB in ASCII or Latin-1 encoding.

       The procedure for configuring ASCII or Latin-1 support depends on the driver. For details, see the instructions that came with your backend data server.

   Important: The conversion between the application encoding and the solidDB server is handled by the solidDB ODBC Driver or solidDB JDBC Driver.

   - In C/ODBC environments, the code page conversions between the application and solidDB are controlled with the server-side parameter **Srv.ODBCDefaultCharBinding** or the client-side parameter **Client.ODBCCharBinding**.

- In Java/JDBC environments, no settings are needed. The code page conversions are handled by the solidDB JDBC Driver automatically.

For more information about setting the parameters and solidDB Unicode support in general, see Using Unicode in the *IBM solidDB Programmer Guide*.

5. **If you backend data server is DB2, running on a 64-bit system, and you are using the IBM Data Server Driver for CLI and ODBC with direct linking, set the solidDB parameter `Passthrough.Force32bitODBCHandles` to `yes`.**
6. **If your backend data server is DB2 for iSeries or DB2 for z/OS, set the InfoSphere CDC for solidDB system parameter `retrieve_credentials` to `false`.**

Related reference:

2.2.1, "Important InfoSphere CDC system parameter settings for Universal Cache," on page 46
Depending on your backend data server and database setup, you might need to modify InfoSphere CDC system parameter settings. In addition to the generic parameter settings that apply to any configuration, there are parameters that are specific to Universal Cache use only.

Example: Installing and configuring IBM Data Server Driver for Informix using direct linking:

This example shows how to install and configure the *IBM Data Server Driver for CLI and ODBC* by linking to the dynamic driver library when the backend data server is IBM Informix Dynamic Server (IDS), V11.50 in Windows 32-bit operating systems.

1. Locate the installation package for **IBM Data Server Driver for CLI and ODBC** (`ibm_data_server_driver_for_odbc_cli_win32_v97.zip`) that contains the driver and copy it to an installation directory of your choice, for example, `C:\solid`.
2. Unzip `ibm_data_server_driver_for_odbc_cli_win32_v97.zip`.

   The ODBC driver library file `db2cli.dll` is located in the `clidriver\bin` directory.
3. Ensure that your IDS backend data server is listening to **drtlitcp** or **drsoctcp** protocol (DRDA® connection).

   For example, to use the **drtlitcp** protocol:

   a. Configure a new server alias in the SQLHOSTS file.

      For example:

      ```
      demo_on drtlitcp idshost 9088
      ```

   b. Verify that the ONCONFIG file lists the DRDA connection as one of the server aliases.

   For more details, see section *Configuring Dynamic Server for Connections to IBM Data Server Clients* (http://publib.boulder.ibm.com/infocenter/idshelp/v115/index.jsp?topic=/com.ibm.admin.doc/ids_admin_0207.htm) in the IDS v11.50 Information Center.
4. In the solidDB configuration file (`solid.ini`), define the driver path and the driver connect string for the IDS backend data server.

   For example:

   ```
   [Passthrough]
   RemoteServerDriverPath=C:\solid\clidriver\bin\db2cli.dll
   RemoteServerDSN="Driver={IBM DB2 ODBC DRIVER};Datab
      ase=my_ids;Hostname=9.252.253.10;Port=9088;protocol=TCPIP;"
   ```

Important: The connect string must be given in double quotation marks, without any spaces between the first equal sign and the double quotation mark.

Example: Installing and configuring IBM Data Server Driver for DB2 using direct linking and UTF-8 support:

This example shows how to install and configure the *IBM Data Server Driver for CLI and ODBC* by linking to the dynamic driver library when the backend data server is DB2 V9.7 in Linux 32-bit operating systems. Additionally, the driver is configured to expect data from solidDB in UTF-8 encoding.

1. Locate the installation package for **IBM Data Server Driver for CLI and ODBC** (`ibm_data_server_driver_for_odbc_cli_32_linuxia32_v97.tar.gz`) that contains the driver and copy it to an installation directory of your choice, for example, `$HOME/solid`.
2. Uncompress `ibm_data_server_driver_for_odbc_cli_32_linuxia32_v97.tar.gz`.

   ```
   cd $HOME/solid
   uncompress ibm_data_server_driver_for_odbc_cli_32_linuxia32_v97.tar.gz
   tar -xvf ibm_data_server_driver_for_odbc_cli_32_linuxia32_v97.tar.gz
   ```

   The ODBC driver library file `db2cli.a` is located in the `clidriver/bin` directory.
3. Set the DB2NOEXITLIST environment variable to ON.

   ```
   export DB2NOEXITLIST=ON
   ```
4. In the solidDB configuration file (`solid.ini`), define the driver path and the driver connect string for the IDS backend data server.

   For example:

   ```
   [Passthrough]
   RemoteServerDriverPath=C:\solid\clidriver\bin\db2cli.dll
   RemoteServerDSN="Driver={IBM DB2 ODBC DRIVER};Datab
      ase=my_db2;Hostname=9.252.253.10;Port=9088;protocol=TCPIP;"
   ```

   Important: The connect string must be given in double quotation marks, without any spaces between the first equal sign and the double quotation mark.
5. Configure the driver to expect data from solidDB in UTF-8 encoding.
   a. Ensure that solidDB database is a Unicode database (**`General.InternalCharEncoding=UTF8`**).
   b. In C/ODBC environments, ensure that the solidDB ODBC Driver is configured to handle code page conversions between the application and solidDB for character data type columns.

      For example, if the application uses UTF-8 encoding for character data types, the solidDB ODBC Driver can be configured to expect character data types in UTF-8 encoding with the following parameter setting:

      ```
      [Srv]
       ODBCDefaultCharBinding=utf8
      ```

      For more information about the **`Srv.ODBCDefaultCharBinding`** parameter and solidDB Unicode support in general, see Using Unicode in the *IBM solidDB Programmer Guide*.
   c. Set a DB2-specific environment variable DB2CODEPAGE to 1208.

      The value 1208 is the identifier for the UTF-8 code page in DB2 environments.

Example: Installing and configuring IBM Data Server Driver for DB2 using unixODBC DriverManager:

This example shows how to install and configure the *IBM Data Server Driver for CLI and ODBC* using unixODBC DriverManager when the backend data server is DB2 V9.7 in Linux 32-bit operating systems.

1. If not already installed, install the unixODBC DriverManager on the solidDB node.

   The unixODBC DriverManager is available for download at http://www.unixodbc.org/.

   Typically the unixODBC DriverManager installation path is `/usr/lib/libodbc.so`.

2. Locate the installation package for **IBM Data Server Driver for ODBC and CLI for Linux 32-bit operating systems, V9.7** (`ibm_data_server_driver_for_odbc_cli_32_linuxia32_v97.tar.gz`) that contains the driver and copy it to an installation directory of your choice, for example, `$HOME/solid`.

3. Uncompress `ibm_data_server_driver_for_odbc_cli_32_linuxia32_v97.tar.gz`.

   For example:

   ```
   cd $HOME/solid/odbc_cli
   uncompress ibm_data_server_driver_for_odbc_cli_32_linuxia32_v97.tar.gz
   tar -xvf ibm_data_server_driver_for_odbc_cli_32_linuxia32_v97.tar
   ```

4. Set the DB2NOEXITLIST environment variable to ON.

   ```
   export DB2NOEXITLIST=ON
   ```

5. In the unixODBC `/etc/odbcinst.ini` configuration file, define the driver path and name for the DB2 driver.

   For example:

   ```
   [DB2drv]
   Description  = DB2 ODBC Driver
   Driver  = /home/solid/odbc_cli/clidriver/lib/libdb2.so
   FileUsage  = 1
   DontDLClose = 1
   ```

   Important: Provide an absolute path when specifying the `Driver` path. Do not use a relative path or an environment variable.

6. In the unixODBC `/etc/odbc.ini` configuration file, define the data source.

   For example:

   ```
   [BE_DB2]
   Description  = DB2 backend database @ myhost
   Driver  = DB2drv
   ```

   Important: The driver name (for example, `[DB2drv]`) must be the name defined in the `odbcinst.ini` file.

7. In the DB2 driver `/home/solid/odbc_cli/clidriver/cfg/db2cli.ini` configuration file, define the DB2 data source parameters.

   For example:

   ```
   [BE_DB2]
   Database=mydb
   Protocol=TCPIP
   Hostname=myhost
   Port=50000
   AutoCommit=0
   ```

8. In the solidDB configuration file (`solid.ini`), define the unixODBC DriverManager path and the Data Source Name for the backend data server.

For example:

```
[Passthrough]
RemoteServerDriverPath=/usr/lib/libodbc.so
RemoteServerDSN=BE_DB2
```

Configuring default SQL passthrough settings for your system

The default SQL passthrough behavior is configured using configuration parameters in the `Passthrough` section of the `solid.ini` file.

Before you begin

If not already installed, install the backend specific ODBC driver on the solidDB frontend node. See 1.4.3, "Installing and configuring backend ODBC drivers for SQL passthrough," on page 38 for details.

Procedure

1. Enable SQL passthrough by setting the **`Passthrough.PassthroughEnabled`** parameter to `yes` (default is `no`).

   Additionally, use the **`Passthrough.IgnoreOnDisabled`** parameter to set how the passthrough statements are handled if the passthrough is disabled **`PassthroughEnabled=no`**. If the value is 'yes' (default), all the statements related to passthrough (SET PASSTHROUGH ...) are ignored. If the value is `no`, an error is returned on any effort to execute those statements.
2. Set the default SQL passthrough mode.
   - Use **`Passthrough.SqlPassthroughRead`** parameter to set how read statements are passed from the solidDB server to the backend.
   - Use **`Passthrough.SqlPassthroughWrite`** parameter to set how write statements are passed from the solidDB server to the backend.

   For both, the values `None` (default), `Conditional`, and `Force` are available.

   Tip: You can override the default SQL passthrough mode using the `SET PASSTHROUGH` or `SET TRANSACTION PASSTHROUGH` commands or ODBC/JDBC connection settings. See 5.3.2, "Setting and modifying SQL passthrough mode," on page 73 for details.
3. Optional: Define the name and location of a file that maps native backend error codes to solidDB error codes.
   a. Create the mapping file.

      The entries in the mapping file have the following format:

      ```
      <backend_error> <solidDB error>  ; rest of the line is comment
      ```

      For example:

      ```
      ; this file maps DB2 native errors to solidDB native errors
      -207 13015 ; column not found
      -407 13110 ; NULL not allowed for non NULL column
      ; end of errormappings
      ```

      For more examples on the mapping files, see the `samples/sqlpassthrough` directory in the solidDB installation directory.

   b. Define the mapping file name and location with the **`Passthrough.ErrorMapFileName`** parameter.

      For example:

      ```
      [Passthrough]
      ErrorMapFileName=myfiles/db2tosoliderrors.txt
      ```

      If **`ErrorMapFileName`** is not defined or the error is not mapped, the native backend error codes are mapped to solidDB error 13456 (Passthrough

backend error: SQLState=<value>, NativeError=<backend error identifier>, MessageText=<backend error description>).

4. Optional: Define the complexity level of SQL statement at which the statement is always passed through to the back end.
   - **Passthrough.ComplexNumTables** – specifies the minimum number of tables in a complex statement. If a statement has less tables than specified with this parameter, the statement is not complex and it is not passed through to the backend.
   - **Passthrough.ComplexNumNonindexedConstr** – specifies the minimum number of non-indexed WHERE clause constraints in a complex statement. If a statement has less non-indexed constraints of the following type, the statement is not complex and it is not passed through to the backend: the WHERE clause constraint does not resolve with index, the index does not exist, or the optimizer chooses different index for constraint.
   - **Passthrough.ComplexNumOrderedRows** – specifies the minimum estimated number of rows which must be sorted in a complex statement. If a statement has less than the estimated number of sortable rows, the statement is not complex and it is not passed through to the backend.

Example

Windows 32-bit environment with driver manager (DB2):

```
[Passthrough]
RemoteServerDriverPath = C:\WINDOWS\system32\odbc32.DLL

RemoteServerDSN = BE_DB2

PassthroughEnabled = yes

IgnoreOnDisabled = no

SqlPassthroughRead = Conditional

SqlPassthroughWrite = Conditional
```

Setting login data for the backend manually

You can set the login data for the backend manually using SQL statements.

About this task

In most cases, the InfoSphere CDC technology is used for transferring backend login data to the frontend. When mirroring or a refresh is started on the first subscription from solidDB server to the backend data server, the InfoSphere CDC for solidDB instance retrieves the login data from the backend InfoSphere CDC instance and stores it in the solidDB system table SYS_SERVER with the statement `CREATE REMOTE SERVER`. The password stored in the SYS_SERVER table is hidden.

The InfoSphere CDC technology does not transfer the backend login data in the cases described below.

- If the InfoSphere CDC instance for the backend is running under a user ID that automatically has access to the database, the login data does not need to be stored.
- If the backend data server is DB2 for z/OS or DB2 for iSeries, the login data cannot be fetched. To avoid errors, you need to set the InfoSphere CDC for solidDB system parameter **retrieve_credentials** to `FALSE`.

- If you have upgraded your InfoSphere CDC for solidDB installation and subscription from V6.3, the 6.3 version of InfoSphere CDC for solidDB has not stored the backend login data in the SYS_SERVER table.

In the latter two cases, use the CREATE REMOTE SERVER (or ALTER REMOTE SERVER) statement to define the login data manually.

Procedure

- **Creating login data**

  ```
  CREATE [OR REPLACE] REMOTE SERVER [USERNAME <username> PASSWORD <password>]
  ```

  By default, username and password are stored in uppercase letters. To retain case sensitivity, enter the username and password in single quotation marks.

  For example:

  ```
  CREATE REMOTE SERVER USERNAME 'AdMin' PASSWORD 'PwD123'
  ```

- **Deleting login data**

  ```
  DROP REMOTE SERVER
  ```

- **Modifying login data**

  ```
  ALTER REMOTE SERVER SET USERNAME | PASSWORD <value>
  ```

5.3.2 Setting and modifying SQL passthrough mode

The default SQL passthrough mode is set with the **SqlPassthroughRead** and **SqlPassthroughWrite** parameters. The parameter settings can be overridden per session or per transaction by using the SET PASSTHROUGH and SET TRANSACTION PASSTHROUGH command. Alternatively, the passthrough mode can also be defined per connection with the ODBC connection attributes or JDBC connection properties.

There are three SQL passthrough modes (levels):

- NONE: SQL passthrough is not used; no commands are passed from the frontend to the backend
- CONDITIONAL: SQL passthrough is activated by a missing table or a syntax error.
- FORCE: all statements are passed from the front end to the backend

For the SET TRANSACTION PASSTHROUGH and SET PASSTHROUGH statements, there is also a fourth option, DEFAULT, which returns the passthrough mode to the current session default.

The precedence hierarchy is, from high precedence to low:

1. SET TRANSACTION PASSTHROUGH: transaction-level settings
2. SET PASSTHROUGH: session-level settings
3. ODBC connection attributes and JDBC connection properties
4. Parameter settings specified by the value in solid.ini configuration file
5. solidDB factory value for the parameter; the factory value for **SqlPassthroughRead** and **SqlPassthroughWrite** is 'NONE'.

Setting transaction-level passthrough mode with the SET TRANSACTION PASSTHROUGH command

The SET TRANSACTION PASSTHROUGH command has effect in the beginning of a transaction, and it affects the transaction until commit or abort. If the statement is issued in the middle of a transaction, an error is returned.

```
SET TRANSACTION PASSTHROUGH {READ  <passthrough level> [WRITE <passthrough level>]}
| {WRITE <passthrough level> | [READ  <passthrough level>]}
| <passthrough level>
```

where

```
passthrough level ::=  NONE | CONDITIONAL | FORCE | DEFAULT
```

Setting session-level passthrough mode with the SET PASSTHROUGH command

The SET PASSTHROUGH statement takes effect immediately, starting from the next SQL statement, until it is reverted by a similar statement or SET TRANSACTION PASSTHROUGH.

The syntax for the SET PASSTHROUGH command is:

```
SET PASSTHROUGH {READ  <passthrough level> [WRITE <passthrough level>]}
| {WRITE <passthrough level> | [READ  <passthrough level>]}
| <passthrough level>
```

where

```
passthrough level ::=  NONE | CONDITIONAL | FORCE | DEFAULT
```

Setting connection level settings for ODBC or JDBC

ODBC

The SQL passthrough mode can be set with the following connection attributes:

- SQL_ATTR_PASSTHROUGH_READ; values: NONE", "CONDITIONAL", "FORCE"
- SQL_ATTR_PASSTHROUGH_WRITE; values: NONE", "CONDITIONAL", "FORCE"

JDBC

The SQL passthrough mode can be set with the following connection properties

- property name: "solid_passthrough_read"; values: "NONE", "CONDITIONAL", "FORCE"
- property name: "solid_passthrough_write"; values: "NONE", "CONDITIONAL", "FORCE"

Changing default settings with the ADMIN COMMAND

The **SqlPassthroughRead** and **SqlPassthroughWrite** parameters are of type R/W; the parameter value can be changed with the ADMIN COMMAND and the change takes effect immediately.

```
ADMIN COMMAND 'parameter Passthrough.<parameter name>=<value>';
```

where

`parameter name` is `SqlPassthroughRead` or `SqlPassthroughWrite`

`value` is NONE, CONDITIONAL, or FORCE.

5.3.3 Tracing and monitoring SQL passthrough

The solidDB server provides means for tracing and monitoring the status of SQL passthrough.

ADMIN COMMAND 'trace { on | off |} passthrough'

The `ADMIN COMMAND 'trace on passthrough'` provides tracing information about the SQL passthrough connections and the loading of the ODBC driver.

- Loading of the ODBC driver: the driver name and status of the load
- Status of connections to the backend: connect/reconnect/disconnect/broken

ADMIN COMMAND 'passthrough status'

The `ADMIN COMMAND 'passthrough status'` provides status information about the SQL passthrough connections:

- NO REMOTE SERVER - no remote server object defined
- NOT CONNECTED - not connected, no errors
- CONNECTED - connected
- LOGIN FAILED - failed at login
- CONNECTION BROKEN - connection broken

Example

```
ADMIN COMMAND 'passthrough status';
RC TEXT
-- ----
0  CONNECTED
```

Performance counters

The following performance counters provide information about the SQL passthrough connections and statements.

Table 25. Perfmon counters

Perfmon Variable	Description
Passthru open connections	Number of SQL passthrough connections to backend
Passthru open statements	Number of prepared statements to backend
Passthru reads	Number of executed read-type statements that return rows (for example, `SELECT` statements)
Passthru non reads	Number of executed write-type statements that return rows (for example, `INSERT` statements)
Passthru commits	Number of committed statements
Passthru rollbacks	Number of rollback statements
Passthru result cnv	Number of fetched (read) rows for which conversion between backend and solidDB data types have been performed. For example, conversion is needed if the data type in backend is CHAR(5) and VARCHAR in solidDB.
Passthru param cnv	Number of statements for which conversion between statement parameters have been performed
Passthru failures	Number of statements that could not be prepared in backend
Passthru reprepared	Number of statements that have been reprepared because write-type statements other that INSERT, UPDATE, and DELETE have been executed in the backend. Repreparation is needed in such cases to ensure that the table definitions have not been changed, which in turn would cause errors with the prepared statements.

For details on how to use the performance counters, see section *Monitoring solidDB* in the *IBM solidDB Administrator Guide*.

5.4 SQL passthrough and High Availability

The SQL passthrough can be used with solidDB High Availability.

Basic connectivity

In normal operation, all the passthrough processing takes place in the Primary server, unless load balancing is used.

Transparent Connectivity with load balancing

When load balancing is enacted (PREFERRED_ACCESS=READ_MOSTLY), the Secondary server is capable of performing the passthrough on read statements. All the non-read statements are directed to the Primary server by using the normal transaction hand-over mechanism.

Server failovers

SQL passthrough operates correctly in the presence of HotStanby failovers, following the normal rules:

- All the transaction that are mid-way (active) are aborted in a failover.
- All transactions that are committed are also successfully committed in the backend.

Using transparent failover

If transparent connectivity is used, failure transparency level must be set to TF_LEVEL=SESSION to retain the session passthrough mode over the connection failover. In the other case (TF_LEVEL=CONNECTION), the session-specific passthrough mode is lost and passthrough operates as on a new connection.

Backend failovers

SQL passthrough does not support backend failovers.

5.5 SQL passthrough failure handling

In failure situations, an error is returned to the at the passthrough request.

Backend fails or shuts down

If the backend data server fails or shuts down (all connections are terminated), the next passthrough request fails on the broken connection and an error is returned to the user. The status of passthrough changes to CONNECTION BROKEN and all active connections from the solidDB server are closed.

To recover:

1. Restart the backend data server.
2. Execute a statement to be passed through.

   The first passthrough request invokes a new successful connection and the passthrough state changes to CONNECTED.

If after the backend failure you set the **Passthrough.PassthroughEnabled** parameter to `no`, errors are not returned when statements to be passed through are executed.

6 Data aging

Data aging is the action of removing table rows from the frontend while preserving them in the backend. This can be used to control main memory usage in the solidDB frontend database by deleting unnecessary data, while preserving the data in the backend database.

The Universal Cache data aging solution is application driven. Applications control what data is to be aged and perform the aging, that is, remove the unnecessary data from the frontend database.

The rules of aging vary from application to application. To use data aging effectively, design the subscriptions and table mappings in such a way that data that has been aged in the frontend database is not propagated back to the frontend database during subsequent refreshes or mirroring from the backend database to the frontend database.

6.1 Principles of operation

Data aging is controlled with SQL statements that define how deletion of data is handled during replication. You can perform data aging per session or transaction.

Overview of data aging

When an application is ready to age data, it invokes a delete capture mode in the solidDB server. The delete capture mode disables the propagation of deletes from the frontend to the backend. The delete capture mode is set with the SQL statement SET [TRANSACTION] DELETE CAPTURE NONE; starting from the next transaction, data changes (deletes) are not propagated on that session or transaction.

After the data has been removed with DELETE statements, the delete capture mode is set back to normal. To prevent deleting the data from the backend database, refreshes from the frontend to the backend are blocked permanently for those tables that have had rows aged.

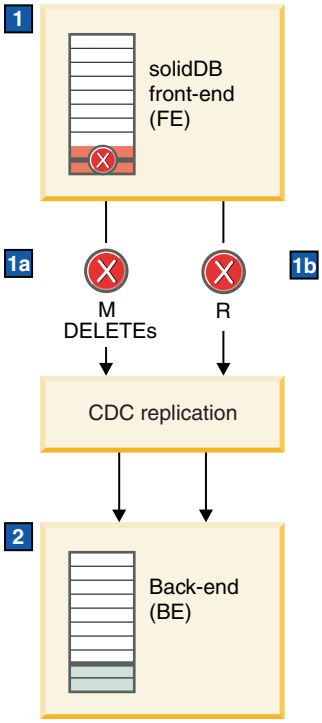

Figure 10. Data aging architecture

1. Data aging mode is enabled with the SET DELETE CAPTURE NONE statement and data is deleted with DELETE statements.
   a. Mirroring of DELETE statements is disabled.
   b. Refresh is disabled permanently for tables in which data has been aged.
2. Rows that were deleted in the frontend database are preserved in the backend data server.

solidDB behavior during data aging

There are two data capture modes that can be set in the solidDB server:

- SET [TRANSACTION] DELETE CAPTURE NONE: Data aging mode is set on: starting from the next transaction, data changes (deletes) are not propagated on this session or transaction.
- SET [TRANSACTION] DELETE CAPTURE CHANGES: Data aging mode is set off: starting from the next transaction, data changes (deletes) are propagated on this session or transaction.

If the data capture mode is set for a session, it is valid until it is changed back to a normal mode (DELETE CAPTURE CHANGES).

To preserve consistency in the frontend and backend databases, execution of certain solidDB statements is restricted during data aging. Other database operations, such as transaction logging or database recovery are not affected.

When the session or transaction is in the DELETE CAPTURE NONE mode, the solidDB server behaves as follows:

- INSERT and UPDATE statements are not allowed.

  This is because transactions containing either one or both would produce potentially inconsistent results in the backend database. The same restriction

applies to statements executed in procedures and triggers. Executing INSERT and UPDATE statements produces an error.

- DELETE statements are allowed.

  Because deleted rows are not propagated to the backend, any portion of the frontend data can be deleted without the danger of removing it from the backend database.
- All other statements, except DDL statements, are executed normally. DDL statements are not allowed.
- The DELETE CAPTURE NONE mode affects only the connection or connections that it is enacted on; database operations performed on other frontend connections (sessions) are not affected.
- The DELETE CAPTURE NONE mode does not affect normal transaction logging, database recovery, or High Availability (HotStandby) operation.

Data aging and solidDB High Availability

If data aging is taking place and HotStandby failure occurs, the active aging transactions (transactions with DELETE statements) are aborted. The aborted transactions need to be executed again. Otherwise, the behavior depends on the connectivity mode:

- **Basic Connectivity**: In Basic Connectivity, the application has to reconnect to the new Primary database and set the DELETE CAPTURE NONE mode for the session.
- **Transparent Connectivity**:
  - If the failure transparency level (TF_LEVEL) is set to SESSION, the application may continue without any preparation steps.
  - If the failure transparency level (TF_LEVEL) is set to CONNECTION, the application has to set the delete capture mode of the session before continuing.
  - Load balancing does not have effect on data aging.

InfoSphere CDC behavior during and after data aging

Subscriptions where solidDB is a source datastore

Refreshes of aged tables from frontend to backend are blocked.

To be able to remove data from the frontend but preserve it in the backend, refreshes from the front end to the backend are blocked for those tables that have had rows aged. This is because when you start a refresh on a subscription, InfoSphere CDC sends a complete copy of the data in the source tables to the target tables. If refreshes were allowed, the data that was deleted from the frontend would also be deleted in the backend.

Note: Both the **Refresh** feature and the automatic refresh performed in the beginning of **Mirroring** are blocked.

Subscriptions where solidDB is a target datastore

Refreshes and mirroring from the back end to frontend are not affected. This means that data that was deleted from the frontend could be returned to the frontend when Refresh or Mirroring from backend to frontend is started.

To prevent this kind of recursion of data from the backend to the frontend:

- Design your subscriptions and table mappings in such a way that data that can become obsolete in the frontend is not mapped to be propagated from the backend to the frontend.
- Do not use Mirroring in the subscriptions from the backend to the frontend.

If you want to return aged data back to the frontend, you need to perform a Refresh from the backend to the frontend. This also removes the blocking of refreshes from frontend to backend for those tables that have been aged.

6.2 Using data aging

To age data in the solidDB frontend, you use SQL statements to first set the data capture mode to a state where deletes are not propagated to the backend, and then delete the rows with the obsolete or unnecessary data. After you have deleted the data, you can change the data capture mode in the solidDB server to allow propagation of data normally.

Before you begin

To use data aging, you need to have your Universal Cache environoment up and running.

It is assumed that there are no subscriptions from the backend to the frontend, or, the subscriptions and table mappings are designed in such a way that there is no risk of returning aged data from the backend to the frontend.

Procedure

1. **Set the data capture mode to DELETE CAPTURE NONE.**
   - To enable data aging for the session, issue the following command:
     ```
     SET DELETE CAPTURE NONE
     ```
   - To enable data aging for the next transaction, issue the following command:
     ```
     SET TRANSACTION DELETE CAPTURE NONE
     ```
2. **Drop (delete) the rows with the obsolete data.**
3. **Disable data aging by setting the data capture mode to DELETE CAPTURE CHANGES.**
   - To disable data aging for the session, issue the following command:
     ```
     SET DELETE CAPTURE CHANGES
     ```
   - If you used the `SET TRANSACTION DELETE CAPTURE NONE` statement, the mode is changed back to DELETE CAPTURE CHANGES after you have committed your transaction.

What to do next

See 7.3.1, “Using the Aging stored procedure,” on page 84 for creating automated data aging scripts.

7 Tools and utilities

The solidDB server and InfoSphere CDC for solidDB installation packages include tools and utilities for automating and scripting common tasks in the setup and usage of InfoSphere CDC replication technology. For example, the tools and utilities can be used to script creation of InfoSphere CDC instances and subscriptions or automate data aging and refreshes.

The tools are provided as a set of sample applications, scripts, and stored procedures that can be used from the operating system command prompt.

InfoSphere CDC for solidDB package - tools and utilities

Table 26. InfoSphere CDC for solidDB package - tools and utilities

Tool or utility	Purpose	Location
Instance and subscription management tools • dminstancemanager • dmsubcriptionmanager	Command-line utilities facilitating InfoSphere CDC instance and subscription management	`<InfoSphere CDC for solidDB installation directory>\samples\ucutils`
ucautomation automation framework (Perl)	Perl-based sample scripts and library modules for automating installation and configuration tasks, including creating datastores, subscriptions, and mappings	`<InfoSphere CDC for solidDB installation directory>\samples\ucautomation`
ucdeploy sample application	Sample application to demonstrate how to set up InfoSphere CDC replication using scripting.	`<InfoSphere CDC for solidDB installation directory>\samples\ucdeploy`
uchsbmonitor sample script	Sample script for monitoring and restarting a set of subscriptions, for example, in High Availability (HotStandby) configurations	`<InfoSphere CDC for solidDB installation directory>\samples\uchsbmonitor`
ucpassthrough sample application	Sample application to demonstrate how to set up InfoSphere CDC replication with the SQL passthrough feature	`<InfoSphere CDC for solidDB installation directory>\samples\ucpassthrough`

solidDB package - stored procedures

Table 27. solidDB package - stored procedures

Stored procedure	Purpose	Location
Data aging • create_automatic_aging.sql • start_automatic_aging.sql • stop_automatic_aging.sql	Automates aging of data in a solidDB database based on user-defined aging rules	`<solidDB installation directory>\procedures`
Refresh • create_refresh_package.sql	Enables refreshes to be started programmatically from an application, without interacting with an InfoSphere CDC instance	`<solidDB installation directory>\procedures`

7.1 Perl automation framework

The Perl automation framework provides a variety of Perl-based sample scripts and library modules for automating installation, configuration, and subscription handling tasks in Linux and Windows environments.

The Perl automation framework is available in the `<InfoSphere CDC for solidDB installation directory>\samples\ucautomation` directory.

- **include** – Perl modules
- **perldoc** – Perl documentation (PODs) on each library module
- **samples** – Sample scripts for creating and controlling subscriptions between two solidDB instances and between solidDB and DB2 instances

The framework provides automation for the following:

- Install components of Universal Cache or InfoSphere CDC replication
- Create databases of various types (solidDB, Informix, DB2 for Linux, UNIX, and Windows) and execute SQL statements against them
- Create and start InfoSphere CDC instances for any of the supported database types
- Create datastores, subscriptions, and mappings and start mirroring
- Cleanup of the environment by deleting any created components

To use the automation framework, you need

- a Linux or Windows environment, and
- a working installation of Perl, available, for example, through http://www.perl.com.

For a detailed description of how to set up the environment and how to use the automation framework, see the `readme` file in the `<InfoSphere CDC for solidDB installation directory>\samples\ucautomation` directory.

7.2 Instance and subscription management tools

The **dminstancemanager** and **dmsubcriptionmanager** tools enable you to script the creation, removal, and modifying of InfoSphere CDC instances and subscriptions.

The tools are available in the `<InfoSphere CDC for solidDB installation directory>\samples\ucutils` directory.

- **dminstancemanager** – creates, removes, modifies and queries status of InfoSphere CDC instances
- **dmsubcriptionmanager** – creates and removes subscriptions or adds table mappings to existing subscriptions
- **ucenv** – configures the environment for the use of the utilities

The **dminstancemanager** and **dmsubcriptionmanager** tools can be used with any InfoSphere CDC engine, not just with InfoSphere CDC for solidDB. The **ucenv** script is used for defining which InfoSphere CDC engine **dminstancemanager** uses.

For a detailed description of how to set up the environment for the use of the **dminstancemanager** and **dmsubcriptionmanager** utilities and how to use them, see the `readme.txt` in the `<InfoSphere CDC for solidDB installation directory>\samples\ucutils` directory.

See also the **ucdeploy** and **ucpassthrough** sample applications, available in the `<InfoSphere CDC for solidDB installation directory>\samples` directory. These samples use the **dminstancemanager** and **dmsubcriptionmanager** tools creating InfoSphere CDC instances and replication subscriptions.

7.2.1 ucdeploy – Configuration and setup sample

The **ucdeploy** sample creates two solidDB databases (frontend and backend), the corresponding InfoSphere CDC instances and datastores, and the subscription between them. The sample then starts mirroring on the subscription, demonstrating how data is replicated from the frontend database to the backend database.

The **ucdeploy** sample uses the **dminstancemanager** utility to create the frontend and backend instances and the **dmsubcriptionmanager** utility to create the subscription. The sample also utilizes standard InfoSphere CDC `dm`-commands, for example, to start mirroring on the subscription.

The **ucdeploy** sample is available in the `<InfoSphere CDC for solidDB installation directory>\samples\ucdeploy` directory.

For a detailed description of how to use the sample, see the `readme.txt` in the same directory.

7.2.2 ucpassthrough – SQL passthrough setup sample

The **ucpassthrough** sample creates two solidDB databases (frontend and backend), the corresponding InfoSphere CDC instances and datastores, and the subscription between them. The sample then inserts data into the backend database using the SQL passthrough feature.

The **ucpassthrough** sample uses the **dminstancemanager** utility to create the frontend and backend instances and the **dmsubcriptionmanager** utility to create the subscription. The sample also utilizes standard InfoSphere CDC `dm`-commands, for example, to start the instances. SQL statements are used to pass through statements that insert and read data in the backend database.

The **ucpassthrough** sample script is available in the `<InfoSphere CDC for solidDB installation directory>\samples\ucpassthrough` directory. For a detailed description of how to use the sample, see the `readme.txt` in the same directory.

7.2.3 uchsbmonitor – HSB subscription monitoring sample

The **uchsbmonitor** sample is Perl script that monitors subscriptions in High Availability setups. The sample program restarts Mirroring if the subscriptions have stopped due to a failover or switchover event.

For example, in cases where a Primary server that is a target datastore fails, replication on subscriptions ends. For recovery, replication on the subscription needs to be restarted.

The **uchsbmonitor** sample script `hsbmonitor.pl` is available in the `<InfoSphere CDC for solidDB installation directory>\samples\uchsbmonitor` directory.

The syntax to run the script is:

```
perl hsbmonitor.pl -s src -t tgt <subscription_name>
```

where

- `src` – name of the source instance
- `tgt` – name of the target instance
- `<subscription_name>` – name of the subscription to be monitored

7.3 SQL stored procedures for data aging and refresh

The stored procedures included in the solidDB package enable automating data aging and refreshes.

The Aging procedure deletes rows in a solidDB database based on user-defined aging rules. The Aging procedure can be activated at solidDB startup so that it performs automatic data aging in the background.

The Refresh stored procedure enables you to start a refresh programmatically from an application, without interacting with an InfoSphere CDC instance.

7.3.1 Using the Aging stored procedure

The Aging procedure *SQL_START_AUTOMATIC_AGING* is an SQL stored procedure that executes user-defined DELETE statements in the solidDB database. The user defines the aging rules in the form of DELETE statements which are maintained in a table AUX_AUTOMATIC_DELETES. The AUX_AUTOMATIC_DELETES table is created automatically by the procedure.

Aging rules

You create and modify the aging rules in the AUX_AUTOMATIC_DELETES table using normal SQL statements. The rules can be removed, added, or changed at runtime.

Table 28. AUX_AUTOMATIC_DELETES table definition

Column	Data type	Description
id	INTEGER PRIMARY KEY	Identifier for the aging rule
statement	LONG VARCHAR NOT NULL	The value must be a full DELETE statement. Any other statement will cause the procedure to fail. Only one statement is allowed per row.
exec_period	INTEGER NOT NULL	Defines the aging interval in seconds
next_exec_date	TIMESTAMP	Defines the next time the rule is executed The procedure calculates the value by adding the value of exec_period to the current time of the execution. If the user gives the value when creating the rule, the first delete operation will take place at the time specified. If the value is not given, the statement will be executed at the next available opportunity.

Any type of DELETE statement can be used as the aging rule. Each row in the AUX_AUTOMATIC_DELETES table corresponds to a single rule. Several rules may be inserted in the table, each executing with its own frequency.

The formulation of the rules depend on the application design. Two examples are described below:

- **Example 1: aging rule is based on a column that contains information on the aging state**

  If in a table called 'table_1' the rows to be aged can be identified by value 'DONE' in a column 'state', the rule statement would be:

  ```
  DELETE FROM table_1 WHERE state='DONE';
  ```

- **Example 2: aging rule is based on the date**

  If in a table called 'table_2', all those rows can be aged for which the date is older that the current date, the rule statement would be:

  ```
  DELETE FROM table_2 WHERE DATE<CURDATE();
  ```

Procedure lifecycle

The procedure does not have any parameters. It runs an internal loop: at each iteration, it reads the rules, executes the applicable rules, and then calculates and updates the next execution time of a rule by adding the value (in seconds) of exec_period to the current execution time. By default, the procedure sleeps for 1 second between each iteration. The sleep interval may be changed by editing the procedure code.

The procedure is typically run as a background job. The exit mechanism is based on a table created by the procedure, called AUX_AUTOMATIC_DELETES_BREAK. At each iteration of the internal loop, the procedure checks whether there are any rows in the AUX_AUTOMATIC_DELETES_BREAK table. If there is at least one row in it, the procedure exits. At the next startup, the procedure removes all the rows from the AUX_AUTOMATIC_DELETES_BREAK table.

Table 29. AUX_AUTOMATIC_DELETES_BREAK table definition

Column	Data type	Description
break	INTEGER	Any existing rows causes the aging procedure to exit

Scripts for creating and running the Aging procedure

The solidDB package includes SQL scripts for creating and running the stored procedure. The scripts are available in the `procedures` directory under the solidDB installation directory.

Table 30. Scripts for creating and running Aging procedure

Script	Usage
create_automatic_aging.sql	Creates the stored procedure
start_automatic_aging.sql	Calls the stored procedure
stop_automatic_aging.sql	Stops the stored procedure

Creating the Aging procedure

To create the Aging procedure:

1. If there are any backend to frontend subscriptions involving the tables that are to be aged in the frontend, remove or stop those subscriptions.

   Alternatively, the databases could be designed in such a way that InfoSphere CDC row filtering can be used to prevent recursion of aged data. For an example, see 7.3.3, "Example: Automating data aging for bidirectional subscriptions," on page 89.
2. Create the procedure by running the script `create_automatic_aging.sql`.

   For example, solsql can be used to run the script.

   ```
   solsql -f "C:\solidDB\procedures\create_automatic_aging.sql" "tcp 2315" dba dba
   ```

Starting and running the Aging procedure

After you have created the procedure, you need to start the procedure and define the aging rules. The aging rules can also be modified at runtime.

1. Start the aging procedure.
   - Run the script `start_automatic_aging.sql`.

     This will start the aging procedure in the background.

     or
   - Include the `start_automatic_aging.sql` script at the solidDB server startup, using the `-x executeandnoexit` command-line option.

     ```
     solid -x executeandnoexit:start_automatic_aging.sql
     ```
2. Define the aging rules by populating the AUX_AUTOMATIC_DELETES table.

   For example, to age data in the table 'table_1' based on the value of the 'state' column every 5 seconds, issue the following command:

   ```
   INSERT INTO aux_automatic_deletes (id, statement, exec_period) values
   (1, 'DELETE FROM table_1 WHERE state=''DONE''', 5);
   COMMIT WORK;
   ```

Stopping the Aging procedure

The Aging procedure can be stopped in the following ways:

- Run the script `stop_automatic_aging.sql`.
- Add a row in the AUX_AUTOMATIC_DELETES_BREAK table by issuing the following command:

  ```
  INSERT INTO aux_automatic_deletes_break (1);
  COMMIT WORK;
  ```
- Use the `ADMIN COMMAND 'backgroundjob'` command to control the procedure.

7.3.2 Using the Refresh stored procedure

The Refresh procedure *TS_REFRESH_CDC_SUBSCRIPTION* is an SQL stored procedure that initiates a Refresh on a subscription.

Overview of the Refresh procedure

To be able to initiate a refresh through the solidDB connection, the Access Server login data must be set with an InfoSphere CDC for solidDB command `dmsetaccessserverparams` before the Refresh procedure is started.

When the procedure is called, it checks the existence and the refresh status of the subscription.

- If the refresh can be started, the procedure call blocks until the refresh is finished. Depending on the size of the refreshed data, the call might block for a long time.

  If the call does not return, normal timeouts apply.
- If the refresh cannot be started, an error is returned.

The state of the refresh is maintained in a table called TS_REFRESH, which is created automatically by the InfoSphere CDC for solidDB when the instance is created. When the procedure is started, it changes the status to '1' (Refresh in progress).After the refresh is finished, InfoSphere CDC for solidDB updates the state to '2' (Refresh finished). If the refresh fails, InfoSphere CDC for solidDB reports the error in the table.

Table 31. TS_REFRESH table definition

Column	Data type	Description
subscription_name	VARCHAR (20) PRIMARY KEY	The subscription name
state	INTEGER NOT NULL	The state of the refresh: • **-1** — error • **0** — refresh requested • **1** — refresh in progress • **2** — refresh finished
error_description	VARCHAR(255)	The error description • Problem loading Access Server Parameters • Access Server username not set • Access Server password not set • Access Server host address not set • Access Server port number not set • Error creating connection to Access Server • Error connecting to Access Server • Connection to Access Server does not exist • Failed to get publishers • Failed to find a matching subscription • Subscription does not exist • Error polling on refresh
inserts_performed	BIG INT	Number of rows for committed inserts during refresh The number of inserts per commit depends on the value that is set with the InfoSphere CDC system parameter “refresh_commit_after_max_operations” on page 169. Default value is 0.

Limitations

The Refresh stored procedure does not support referential integrity. If your tables include foreign keys and you have set the InfoSphere CDC for solidDB system parameter **refresh_with_referential_integrity** to true, the Refresh stored procedure cannot start a refresh. Instead of using the Refresh stored procedure, you must initiate refreshes manually using the Management Console or with the **dmrefresh** command.

Creating the Refresh procedure

The solidDB package includes an SQL script for creating the stored procedure. The script is available in the procedures directory under the solidDB installation directory.

Script	Usage
create_refresh_package.sql	Creates the stored procedure

To create the Refresh procedure:

1. Ensure that you have created the subscriptions and your frontend and backend data servers and the InfoSphere CDC components are running normally.
2. Create the Refresh procedure by running the script create_refresh_package.sql.

   You can use **solsql** to run the script, as shown in the following example:

   ```
   solsql -f "C:\solidDB\procedures\create_refresh_package.sql" "tcp 2315" dba dba
   ```
3. Define the login data for the Access Server using the InfoSphere CDC for solidDB command dmsetaccessserverpararms.

   The syntax for the dmsetaccessserverpararms command is:

   ```
   dmsetaccessserverparams [-u <username>] [-p <password>] [-H <hostname>] [-P <port>]
   ```

   For example:

   ```
   dmsetaccessserverparams -u dba -p dba - H 192.167.3.3 -P 10101
   ```

Running the Refresh procedure

To run the Refresh procedure:

1. Ensure that there is Mirroring (continuous) ongoing in a frontend to backend subscription.
2. Call the Refresh procedure with the following syntax:

   ```
   CALL ts_refresh_cdc_subscription ('subscription_name');
   ```

   For example:

   ```
   CALL ts_refresh_cdc_subscription ('current_invoices');
   ```

Monitoring the status of refresh

You can check the progress of the refresh by viewing the TS_REFRESH table for the state of refresh and the number of refreshed rows (inserts_performed).

For example:

```
SELECT * from TS_REFRESH;

SUBSCRIPTION_NAME   STATE  ERROR_DESCRIPTION   INSERTS_PERFORMED
-----------------   -----  -----------------   -----------------
```

```
current_invoices        1                                   2000

1 rows fetched.
```

Stopping the stored procedure

The procedure call blocks until the refresh executes successfully. If you want to stop the procedure, use the `ADMIN COMMAND 'throwout'` to force an exit.

Normal timeouts apply also:

- If a *query timeout* is set, the call will timeout on the query timeout. By default, there is no timeout.

  For example:

  – In ODBC, set the query timeout with the ODBC statement attribute SQL_ATTR_QUERY_TIMEOUT (in seconds).

  – In JDBC, set the query timeout with the statement method setQueryTimeout() (in seconds).
- If a *connection timeout* is set, the connection is lost after the timeout expires.

  For details on the timeout behavior, see appendix *Timeout controls* in the *IBM solidDB Programmer Guide*.

7.3.3 Example: Automating data aging for bidirectional subscriptions

This example describes how the Aging procedure can be used together with InfoSphere CDC row filtering to automate data aging in a bidirectional subscription setup.

If your setup includes bidirectional subscriptions, you must design your applications and subscriptions in such a way that those rows that are removed (aged) from the frontend are not returned back to the frontend when a backend to frontend refresh or mirroring is used.

One possibility is to use the Aging procedure to delete data in the frontend while InfoSphere CDC row filtering is set up to prevent replication of aged rows back to the frontend.

In this example, the application controls which data can be aged by maintaining information about the aging status of the data; it flags the rows to be deleted. The actual deletes of data are performed using the Aging procedure. The InfoSphere CDC row filtering is then set up so that rows which are flagged to be deleted are not replicated from the backend to the frontend.

Example of setting up data aging with bidirectional subscriptions

Note: In this example, it is assumed that a new column can be added to the tables. This is not mandatory; depending on the database design, existing columns could be used to identify the aged rows.

1. **Set up the environment to support data aging.**
   - Add a column 'aged' which can contain the value '0' (not aged) or '1' (aged).
   - Design the application to set the rows to be aged to have value '1' in the 'aged' column.
2. **Create and start the Aging procedure in the solidDB server.**

For details, see 7.3.1, “Using the Aging stored procedure,” on page 84.

3. **Set up the subscriptions from frontend to backend and vice versa.**
4. **In the backend to frontend subscriptions, set up the row filtering.**

   Create a row filtering rule that replicates only those rows that have the value <1 in the 'aged' column.

   For instruction on how set filters, see section *Filtering rows and columns* in the IBM InfoSphere Change Data Capture version 6.5 Information Center.
5. **Create the aging rules by adding DELETE statements in the AUX_AUTOMATIC_DELETES table.**

   For example, to create a rule that deletes all rows flagged for delete in table_1, execute the following INSERT statement:

```
INSERT INTO aux_automatic_deletes (id, statement, exec_period) values
(1, 'DELETE FROM table_1 WHERE aged=1', 10);
```

Result

While the application is running, it flags certain rows in the database for aging ('aged' = 1). Those rows are replicated to the backend, together with the changed aging state. As the Aging procedure is running, it deletes the flagged rows from the frontend tables. These rows will not be replicated from the backend to the frontend as the row filtering ('aged' < 1) prevents that.

Replicating data from backend to front end

All forms of replication from the backend to the front end are allowed: continuous mirroring, a refresh initiated with the InfoSphere CDC tools, or a refresh initiated by the application by using the Refresh procedure .

However, since InfoSphere CDC replication is asynchronous in nature (the changes made in the frontend are not effective immediately in the backend, and vice versa), the following restrictions apply for this example:

- If the rows to be aged are being modified in the backend, mirroring from the backend to the frontend is not allowed until the aging activity has finished.
- If the Refresh procedure is used, ensure that the values in the 'aging' column have been replicated to the backend before you issue a refresh. If you issue a refresh before replication has finished, the data you aged in frontend may return to the backend.

You can check the progress of the refresh by viewing the number of refreshed rows in the TS_REFRESH table.

For example:

```
SELECT * from TS_REFRESH;

SUBSCRIPTION_NAME   STATE  ERROR_DESCRIPTION   INSERTS_PERFORMED
-----------------   -----  -----------------   -----------------

current_invoices        1                                  2000

1 rows fetched.
```

8 Failure handling in Universal Cache

The following sections provide an overview of different failure scenarios and the required recovery procedures, if any.

Tip: If the recovery instructions contain manual tasks, they can often be automated by using scripts or the commands available with InfoSphere CDC.

8.1 Standalone solidDB server fails

If a standalone solidDB server fails, replication on the subscriptions ends also. To recover, proceed as follows:

Procedure

1. Restart the solidDB server manually and recover the database.

   For instructions, see the section *Administering* in the *IBM solidDB Administrator Guide*.
2. Restart the InfoSphere CDC instance.

   For instructions, see the section 10.4, "Starting and stopping InfoSphere CDC," on page 114.
3. Restart replication on the subscriptions.

   For instructions, see section *Starting and ending replication on subscriptions* in the IBM InfoSphere Change Data Capture version 6.5 Information Center.

Results

When restarted, replication on the subscriptions resumes and the databases are re-synchronized. Replication continues normally.

8.2 InfoSphere CDC instance fails

If the InfoSphere CDC instance fails, replication on the subscriptions ends also. To recover, proceed as follows:

1. Restart the InfoSphere CDC instance.

   For instructions, see the section 10.4, "Starting and stopping InfoSphere CDC," on page 114.
2. Restart replication on the subscriptions.

   For instructions, see section *Starting and ending replication on subscriptions* in the IBM InfoSphere Change Data Capture version 6.5 Information Center.

Results

When restarted, replication on the subscriptions resumes and the databases are re-synchronized. Replication continues normally.

If this failure happens, the solidDB server continues to process transactions until it reaches the limit specified by the **LogReader.MaxLogSize** parameter.

8.3 solidDB server in HA mode (HotStandby) fails

The following sections describe failure scenarios in a HotStandby configuration.

Primary solidDB server fails

If the primary solidDB server fails, a high availability manager, like the High-Availability Controller (HAC), performs a failover to the secondary solidDB server as a standard procedure. If the 2-Safe protocol is used, the database and log states are fully preserved. The applications perceive a failover time of less than one second.

- If the solidDB database is a source datastore (write-only cache where the data is replicated only from the frontend to the backend), the InfoSphere CDC instance reconnects automatically to the new primary, and replication continues.
- If the solidDB database is a target datastore (read-only or read-write cache), replication on the subscriptions ends. The subscription needs to be restarted with the Management Console or with the InfoSphere CDC command `dmstartmirror`.

  For instructions, see section *Starting and ending replication on subscriptions* in the *InfoSphere Change Data Capture Management Console, Administration Guide*.

During the scenario above, the InfoSphere CDC instance is in operation all the time.

Tip: For more information about the HA (HotStandby) functionality and the High-Availability Controller (HAC), see the *IBM solidDB High Availability User Guide*.

Secondary solidDB frontend fails

In the case of secondary frontend failure, no manual intervention is needed.

If the secondary frontend fails, the secondary frontend node is recovered in a normal way that is specific to the installation (for example, automatically rebooted). HAC automatically performs the rest of the recovery. The failure is not visible to applications or to theInfoSphere CDC instance.

8.4 Communication link between primary solidDB server and InfoSphere CDC for solidDB instance fails

If the communication link between the primary solidDB server and the InfoSphere CDC for solidDB instance fails, replication on the subscriptions ends also. However, the failure of the link alone is considered unlikely.

To recover, proceed as follows:

1. Restart the InfoSphere CDC instance.

   For instructions, see the section 10.4, “Starting and stopping InfoSphere CDC,” on page 114.
2. Restart replication on the subscriptions.

   For instructions, see section *Starting and ending replication on subscriptions* in the IBM InfoSphere Change Data Capture version 6.5 Information Center.

Results

When restarted, replication on the subscriptions resume and the databases are re-synchronized. Replication continues normally.

If this failure happens, the solidDB server continues to process transactions until it reaches the limit specified by the **LogReader.MaxLogSize** parameter.

8.5 Backend server or backend node fails

If the backend server or the backend node fails, replication on the subscriptions ends also. To recover, proceed as follows:

Procedure

1. Restart the backend server and recover the database.
2. Restart the InfoSphere CDC instance.
3. Restart mirroring (replication) on the subscriptions.

   Note: It may be possible to automate the above steps using the backend specific tools and procedures.

Results

When restarted, the replication resumes and the databases are resynchronized. Replication continues normally.

If this failure happens, the solidDB front end continues to process transactions until it reaches the limit specified by the **LogReader.MaxLogSize** parameter.

8.6 Backend primary server fails

If the the backend primary server fails, or if the whole backend node fails, the recovery must be handled according to rules and tools of the backend product in question. The solidDB server does not offer any means to correct the situation.

After the backend server is running as a new primary, an exact copy of the InfoSphere CDC instance is restarted at the surviving node. You must reconfigure the subscription with the InfoSphere CDC tools to reconnect the InfoSphere CDC instances in question. The new subscription must continue from a full refresh (in both directions) before the mirroring can start.

In some cases, the state of the subscription replication might be lost, and a full refresh is needed.

9 Troubleshooting

This section provides instructions and guidelines on how to prevent or troubleshoot common problems while configuring or using Universal Cache.

Initial connections are not successful

The components for Universal Cache must be installed and configured in the order described in section *Overview of installation and configuration steps*. Review the steps below and ensure that the installation and configuration steps were followed.

Installation and configuration order

- Frontend solidDB server
- InfoSphere CDC for solidDB
- Backend data server
- InfoSphere CDC for the backend data server
- Access Server
- Management Console

Dependencies between components used in replication

To set up replication between databases, you need define and create various entities and components which are dependent on each other. These entities and components must be created in the following order and modified or deleted in the reverse order. For more details and instructions, see the IBM InfoSphere Change Data Capture version 6.5 Information Center.

1. Databases
2. InfoSphere CDC instances
3. Datastores
4. Subscriptions
5. Table mappings

Making changes to replication subscriptions

If you need to make changes to your replication subscriptions, you must first end replication on your subscriptions. For more details and instructions, see section *Ending replication on a subscription* in the IBM InfoSphere Change Data Capture version 6.5 Information Center.

Subscriptions fail after performing `hsb netcopy` followed by a switchover

In High Availability (HotStandby) configurations, subscriptions where the solidDB database is the source datastore might fail if a switchover is performed shortly after **`hsb netcopy`**.

The subscriptions might fail, for example, in the following cases:

1. After a failure or a maintenance break, primary server (node 1) and secondary server (node 2) are synchronized using `ADMIN COMMAND 'hsb netcopy'`.
2. Replication continues against the primary server (node 1) for few transactions.
3. The primary server (node 1) fails and switchover changes the secondary server (node 2) to be the new primary server.
4. Subscriptions fail and replication against the new primary server (node 2) cannot be restarted.

Causes

The command `ADMIN COMMAND 'hsb netcopy'` does not copy any log files. Subsequently, because InfoSphere CDC replication is asynchronous in nature, InfoSphere CDC for solidDB might not have processed all the transactions up to the point from which the **`hsb netcopy`** was made. This means that the log position InfoSphere CDC for solidDB tries to use after the switchover might not be valid – the log entry for the last transaction on node 1 before the **`hsb netcopy`** might not exist on the new primary (node 2).

Workaround

To ensure that InfoSphere CDC for solidDB has access to a valid log entry in the new primary server (node 2) after a switchover:

- Before performing **`hsb netcopy`**, copy the log files from the primary server (node 1) to the secondary server (node 2). This ensures that InfoSphere CDC for solidDB has access to the log positions of the transactions that were executed before the **`hsb netcopy`** was made.

  or

- Do not perform switchover shortly after **`hsb netcopy`** or wait for several transactions to be replicated to the backend database before performing the switchover. This ensures that log positions in the primary server (node 1) and secondary server (node 2) are synchronized.

  or

- If the switchover has already taken place (for example, due to a failure of node 1):

  1. Recover the old primary server (node 1).
  2. Perform a switchover to return the old primary server (node 1) back to a primary server.
  3. Restart replication on the subscription.

  Before performing another switchover (to make node 2 the new primary server), wait for several transactions to be replicated. This ensures that log positions in the primary server (node 1) and secondary server (node 2) are synchronized.

InfoSphere CDC for solidDB connection to solidDB server times out

InfoSphere CDC for solidDB connections to the solidDB server can be idle for long periods of time, causing connection idle timeouts. By default, the solidDB server timeout for idle connections is set to 480 minutes (specified with the **Srv.ConnectTimeOut** parameter).

Workaround:

Set the connection idle timeout for the InfoSphere CDC for solidDB connection to infinite by using the non-standard solidDB JDBC connection property **solid_idle_timeout_min=0**. The InfoSphere CDC for solidDB connection settings are specified with the InfoSphere CDC configuration tool (**dmconfigurets**), using the **Database area** > **Advanced** button in Windows operating systems or the **Configure advanced parameters** > **Modify settings** option in Linux and UNIX operating systems.

Note: The timeout setting specified for the InfoSphere CDC for solidDB instance does not impact the server setting (**Srv.ConnectTimeOut**) for other connections.

Bidirectional replication does not work between solidDB and DB2

Symptom

Bidirectional replication between solidDB and DB2 for Linux, UNIX, and Windows does not work. You can create the subscriptions and table mappings successfully but data changes in the solidDB database are not replicated to the DB2 database.

In some cases, starting replication (**Start Mirroring**) fails with the following type of error message:

```
Error 1465 BIDI5 Mar 13, 2013 3:04:11 PM
--- Subscription BIDI5 is terminating abnormally.

Error 1713 BIDI5 Mar 13, 2013 3:04:11 PM
IBM InfoSphere Change Data Capture to BIDI5 is initiating shutdown due
to failure on the local system. See the previous messages for
additional information.

Error 2913 BIDI5 Mar 13, 2013 3:04:11 PM LOAD operation is not supported
by IBM InfoSphere Change Data Capture. Please refresh the the table [
DB2ADMIN.T42] when LOAD operation ends.
```

Recovery

To use bidirectional replication with DB2 for Linux, UNIX, and Windows, set the InfoSphere CDC for DB2 system parameter **ddl_awareness** to `false`.

You can set system parameters in two ways:

- On the computer where your InfoSphere CDC for DB2 replication engine is installed, issue the following command:

  ```
  dmset -I <INSTANCE_NAME> ddl_awareness=false
  ```

  For example:

  ```
  dmset -I backend_DB2 ddl_awareness=false
  ```

- In the **Configuration** perspective of the Management Console, right-click on the datastore and select **Properties** > **System Parameters**. Click **Add** and enter the parameter name and its value (**ddl_awareness=false**).

10 InfoSphere CDC for solidDB reference

This section contains detailed instructions about how to install and configure the IBM InfoSphere Change Data Capture for IBM solidDB replication engine. This section also includes commands specific to the InfoSphere CDC replication engine, as well as other reference information.

This section corresponds to the documents called *IBM InfoSphere Change Data Capture End-User Documentation* that are delivered with the InfoSphere CDC components for the other data servers.

When setting up Universal Cache or InfoSphere CDC replication between solidDB servers, follow the system level installation and configuration instructions, referring to this section as necessary.

In this section, the term InfoSphere CDC is used for referring to InfoSphere CDC for solidDB.

10.1 About InfoSphere CDC

IBM InfoSphere Change Data Capture (InfoSphere CDC) is a replication solution that allows you to replicate data to or from supported databases. It can also receive replicated data from supported databases based on table mapping details defined during configuration.

InfoSphere CDC allows you to maintain a replicated database that can be used to reduce processing overheads and network traffic. Replication can be carried out continuously or periodically on a net change basis. When data is transferred from a source server, it can be remapped or transformed in the target environment.

10.1.1 InfoSphere CDC for solidDB system requirements

Disk space requirements

Table 32. Disk space requirements

Disk space
InfoSphere CDC source system: • **100 GB**—Default value for the **Staging Store Disk Quota** for each instance of InfoSphere CDC. Use the InfoSphere CDC configuration tool to configure disk space for this quota. • **5 GB**—For installation files, data queues, and log files. • **Global disk quota**—Disk space is required on your source system for this quota which is used to store in-scope change data that has not been committed in your database. The amount of disk space required is determined by your replication environment and the workload of your source database. Use the **mirror_global_disk_quota_gb** system parameter to configure the amount of disk space used by this quota. InfoSphere CDC target system: • **1 GB**—The minimum amount of disk space allowed for the **Staging Store Disk Quota** for each instance of InfoSphere CDC. The minimum value for this quota is sufficient for all instances created on your target system. Use the InfoSphere CDC configuration tool to configure the disk space for this quota. • **5 GB**—For installation files, data queues, and log files. • **Global disk quota**—Disk space is required on your target system for this quota which is used to store LOB data received from your InfoSphere CDC source system. The amount of disk space required is determined by your replication environment and the amount of LOB data you are replicating. To improve performance, InfoSphere CDC will only persist LOB data to disk if RAM is not available on your target system. Use the **mirror_global_disk_quota_gb** system parameter to configure the amount of disk space used by this quota.

InfoSphere CDC may require additional disk space in the following situations:

- You are running large batch transactions in the database on your source system.
- You are configuring multiple subscriptions and one of your subscriptions is latent. In this type of scenario, InfoSphere CDC on your source system may persist transaction queues to disk if RAM is not available.
- You are replicating large LOB data types.
- You are replicating "wide" tables that have hundreds of columns.
- You are performing regular back ups of your metadata with the **dmbackupmd** command-line utility.

RAM requirements

Table 33. RAM requirements

RAM
Each instance of InfoSphere CDC requires memory for the Java Virtual Machine (JVM). The following default values for memory are assigned: • **1024 MB of RAM**—Default value for each 64-bit instance of InfoSphere CDC. • **512 MB of RAM**—Default value for each 32-bit instance of InfoSphere CDC. Use the InfoSphere CDC configuration tool to configure the memory for each instance of InfoSphere CDC. **Note:** InfoSphere CDC is predominantly a Java-based application. However, some portions of it are written in C. These portions of InfoSphere CDC are not subject to the memory limits specified for the JVM.

Although InfoSphere CDC memory requirements will fluctuate, you must work with your system administrator to ensure the allocated memory for each instance of the product is available at all times. This may involve deployment planning since other applications with memory requirements may be installed on the same server with InfoSphere CDC. Using values other than the defaults or allocating more RAM than is physically available on your server should only be undertaken after considering the impacts on product performance.

InfoSphere CDC source deployments may require additional RAM in the following scenarios:

- You are replicating large LOB data types with your InfoSphere CDC source deployment. These data types are sent to target while being retrieved from the source database. The target waits until all LOBs (for each record) are received before applying a row. LOBs are stored in memory as long as there is adequate RAM, otherwise they are written to disk on the target.
- You are replicating "wide" tables with hundreds of columns.
- You are performing large batch transactions in your source database rather than online transaction processing (OLTP).

Port requirements

InfoSphere CDC requires that you allocate a set of ports for communications with other components in the replication environment. The ports must be accessible through firewalls, although you do not require access to the internet.

Table 34. Port requirements

Protocol	Default port	Purpose
TCP	11101	Accepts connections from: • Management Console • Other installations of InfoSphere CDC as a source of replication • Command line utilities

Assessing disk space and memory requirements

InfoSphere CDC requires disk space and memory when it processes change data from your source database. In order to process change data efficiently and replicate these changes to your target system, it is very important that InfoSphere CDC has adequate disk space and memory for each of the components described in this section.

Disk space requirements for the staging store

The InfoSphere CDC staging store is located on your source system and is a cache of change data read from the database logs. The size of the staging store will increase as the product accumulates change data, and therefore you must plan your source environment accordingly, particularly disk space.

The disk space allocated to the staging store is controlled by the **Staging Store Disk Quota** value that is set when you create an instance with the InfoSphere CDC configuration tool. In most cases, the default value is appropriate for InfoSphere CDC source systems. Since the staging store is only used on source systems, you can reduce this value to the minimum of 1 GB if you are configuring a target instance of InfoSphere CDC.

Note: You can also allocate disk space to the staging store with the **staging_store_disk_quota_gb** system parameter in Management Console.

Memory requirements for the JVM (Java Virtual Machine)

As a Java-based product, InfoSphere CDC requires you to allocate the maximum amount of memory (RAM) to be used by the Java Virtual Machine (JVM). This prevents InfoSphere CDC from using all of the available memory on the system where it is installed. The **Maximum Memory Allowed** value is set on a per-instance basis for each instance you create for your source or target database. In most cases the default values are appropriate for 32-bit and 64-bit instances. However, if your database is processing an extremely heavy workload, you may have to adjust the default values. The RAM allocated must be physically available on your system.

Disk space requirements for the global disk quota

The global disk quota on your source and target systems is used for all capture components including temporary files, transaction queues, and LOBs which are staged on the target before being applied. InfoSphere CDC will manage disk space utilization across all components as required.

Most databases have a mechanism that allows you to roll back or undo changes to your database by storing uncommitted changes. Similarly, InfoSphere CDC uses this disk quota to store in-scope change data that has not been committed in your database. Once the database transaction is committed, the disk space used by the transaction is released. Long running open transactions will contribute to the amount of disk space used.

You can configure the amount disk space that is allocated to this quota with the **mirror_global_disk_quota_gb** system parameter. The default setting of this system parameter is such that InfoSphere CDC will only stop replicating after this disk quota exhausts all available disk space on your system. If you would prefer InfoSphere CDC to stop replicating after it uses a specific amount of disk space, you can specify the value with this system parameter in Management Console.

Sizing considerations for the staging store:

This topic outlines scenarios that will increase the disk requirements for the staging store on your source system. All of these scenarios should be kept in mind when you are planning the disk space requirements for your replication environment.

Latent subscriptions

The amount of data within the staging store is related to the latency of your subscriptions. InfoSphere CDC measures latency as the amount of time that passes between when data changes on a source table and when it changes on the target table. For example, if an application inserts and commits a row into the source table at 10:00 and InfoSphere CDC applies that row to the target table at 10:15, then the latency for the subscription is 15 minutes.

When all of your subscriptions are mirroring and have very little latency, the volume of data that needs to be kept in the staging store will be relatively small. If all of your subscriptions are mirroring but some are latent, the staging store will contain all the data generated by the logs for the latent subscriptions during the entire time they are mirroring. For example, if the difference in latency between the

least latent subscription and the most latent subscription is 3 hours, and your database generates 100 GB of log data per hour, the staging store will require approximately 300 GB of disk storage space.

Inactive subscriptions

An inactive (not currently replicating) subscription that contains tables with a replication method of Mirror will continue to accumulate change data in the staging store from the current point back to the point where mirroring was stopped. For this reason, you should delete subscriptions that are no longer required, or change the replication method of all tables in the subscription to Refresh to prevent the accumulation of change data in the staging store on your source system.

Continuous Capture

Continuous Capture is designed to accommodate those replication environments in which it is necessary to separate the reading of the database logs from the transmission of the logical database operations. This is useful when you want to continue processing log data even if replication and your subscriptions stop due to issues such as network communication failures over a fragile network, target server maintenance, or some other issue. You can enable or disable Continuous Capture without stopping subscriptions.

Continuous Capture results in additional disk utilization on the source machine in order to accumulate change data from the database log file when these are not being replicated to the target machine. This change data is stored in the staging store. The additional disk utilization due to the accumulation of change data in the staging store should be evaluated and understood before deciding to use this feature in your replication environment.

10.1.2 Required database, user accounts, and schemas

Creating a solidDB database

When you configure InfoSphere CDC, you are prompted for the host name and port number of the solidDB server you want InfoSphere CDC to connect to and replicate data. Before installing InfoSphere CDC, ensure that this solidDB database exists and that you have created and set up a database user that has access to it.

Setting up a solidDB account with SYS_ADMIN_ROLE privileges

Create and set up a solidDB user and assign DBA privileges to this user. For InfoSphere CDC to connect to your solidDB database, you need to create a solidDB user account and assign SYS_ADMIN_ROLE privileges to this user. When you configure InfoSphere CDC, you are prompted for the host name and port number of the solidDB server you want InfoSphere CDC to connect to and the user name and password of the solidDB user that has access to this database.

Creating a solidDB schema

Create a schema or choose an existing schema for your InfoSphere CDC database metadata tables. You will have to specify this schema when you configure InfoSphere CDC.

Setting up a Windows user account for InfoSphere CDC

If you are installing InfoSphere CDC on a Windows system, you must set up a new, or decide on an existing Windows account that you will use to install, configure, or upgrade InfoSphere CDC.

Setting up a Linux or UNIX user account for InfoSphere CDC

If you are installing InfoSphere CDC on a Linux or UNIX system, you must set up a new, or decide on an existing Linux or UNIX account that you will use to install, configure, or upgrade InfoSphere CDC. You can install InfoSphere CDC in the directory of your choice, however, it must be owned by the Linux or UNIX account.

10.1.3 Single byte and multibyte character support

InfoSphere CDC supports replication of both single byte and multibyte character sets.

Single byte character support

InfoSphere CDC performs single byte character support (SBCS) code page conversions transparently. This means that you do not have to be aware of the code pages that are being used on each system. InfoSphere CDC is able to perform the conversions automatically by examining user configuration parameters.

Multibyte character support

InfoSphere CDC supports the replication of multibyte character sets (MBCS) such as Japanese or Chinese, which cannot be represented in a single-byte. The most common MBCS implementation is double-byte character sets (DBCS).

The specification for MBCS dictates that data will be applied as is to the mapped column on the target system when you have configured a specific translation. This is possible when the database has a single-byte character set configured (regardless of the actual character set of the data) but this cannot be assured when the character set is multibyte.

InfoSphere CDC will respect the mappings and apply the data according to the configuration set. There will be no validation that the character set can be inserted correctly into the column. You must be aware of the character sets on the database and select the appropriate values when selecting character set translations for their data. When you set an encoding conversion in Management Console, InfoSphere CDC applies the data to the target database in the exact form it was received.

Implications for multibyte character support in solidDB databases

The encoding of solidDB character data types depends on the database mode, *Unicode* or *partial Unicode*.

Unicode mode (`General.InternalCharencoding = utf8`)

- Character data types (CHAR, VARCHAR, and so on) are stored in UTF-8.
- Wide character data types (WCHAR, WVARCHAR, and so on) are stored in UTF-16.

Partial Unicode mode (`General.InternalCharencoding = raw`)

- Character data types use no particular encoding; instead, the data is stored in byte strings with the assumption that user applications are aware of this and handle the conversion as necessary.
- Wide character data types are stored in UTF-16.

When a new instance of InfoSphere CDC for solidDB is created, the default encoding is set according to the default solidDB database mode which is partial Unicode. By default, the encoding of character data type columns is always set to ISOLatin1.

- If your database mode is Unicode, you need to set the encoding of character data type columns (CHAR, VARCHAR, and so on) to UTF-8.
- If your database mode is partial Unicode and your application encoding is not set to ISOLatin1, you need to set the encoding of character data type columns (CHAR, VARCHAR, and so on) to the encoding used in the application environment.

Table 35. Default (partial Unicode) and Unicode encoding settings for character and wide character data type columns

Column type	Default encoding (partial Unicode)	Required encoding for Unicode databases
Character data types (CHAR, VARCHAR, and so on)	ISOLatin1	UTF-8
Wide character data types (WCHAR, WVARCHAR, and so on)	UTF-16BE	UTF-16BE

User exits and multibyte character sets

Java class user exits in InfoSphere CDC support multibyte character sets (MBCS). Multibyte character sets are converted to Java strings (UTF-16).

10.2 Installing InfoSphere CDC

This section provides step-by-step instructions on how to install InfoSphere CDC.

10.2.1 Installing InfoSphere CDC using an interactive installation

You can install InfoSphere CDC on a Windows server or an UNIX or Linux server.

To install InfoSphere CDC (Windows)

Procedure

1. Double-click the installation file. The InfoSphere CDC installation wizard opens.
2. Click **Next**.
3. If you agree to the license terms, select **I accept the terms in the license agreement** and then click **Next**.
4. Select the folder where you want to install InfoSphere CDC and click **Next**.
5. If you have a previous installation of InfoSphere CDC, the installation will prompt you to upgrade the installation. Click **OK** to upgrade the installation.
6. Select the location for the product icons and click **Next**.
7. Review the installation summary and click **Install**.

8. Optionally, select **Launch Configuration Tool** to launch the configuration tool after the installation. The configuration tool allows you to add an instance of InfoSphere CDC.
9. Click **Done** to exit the installation.

To install InfoSphere CDC (UNIX and Linux)

About this task

Note: If you have X-Windows installed, the installation program will launch the configuration tool in a graphical environment. The configuration process is similar to Windows except you do not have to start and stop instances.

Procedure

1. Log on to the account you set up for InfoSphere CDC.
2. Copy the InfoSphere CDC installation file for your Linux platform.
3. Make the installation program executable.
4. Run the installation program by typing the name of the installation file.
5. Press `Enter` on the **Introduction** screen to display the license agreement. Follow the instructions on the screen to navigate through the license agreement.
6. To accept the license agreement, type `1`.
7. Type the absolute path to your installation directory or press `Enter` to accept the default.

   Note: The directory that you specify must be owned by the account you are using for the installation. If the installation program cannot create the directory, you are prompted to specify a different directory.
8. Review the installation summary. Press `Enter` to start the installation.
9. After completing the installation, InfoSphere CDC gives you the option of launching the configuration tool for InfoSphere CDC.
10. Type `1` to launch the configuration tool.

10.2.2 Installing InfoSphere CDC using a silent installation

A silent installation allows you to automatically install InfoSphere CDC by specifying a command with various parameters. You can use this type of installation method for large-scale deployments of InfoSphere CDC by embedding the silent installation command in a script.

To perform a silent installation of InfoSphere CDC (UNIX and Linux)

Procedure

1. Log on to the account you set up for InfoSphere CDC.
2. Copy the InfoSphere CDC installation file.
3. Make the installation program executable.
4. Install InfoSphere CDC and generate a response file with the following command:

   ```
   <setup.bin> -r <response-file>
   ```
5. On another system, perform the silent installation by running the following command:

   ```
   <setup.bin> -i silent -f <response-file>
   ```

   where:

- <response-file> is the full path to the installation file.

10.3 Configuring InfoSphere CDC

After installing InfoSphere CDC, the installation program launches a configuration tool. The configuration tool allows you to configure InfoSphere CDC for your environment. You must configure InfoSphere CDC before you can start replication.

10.3.1 Configuring InfoSphere CDC instances (Windows)

You can add, edit, or delete an instance of InfoSphere CDC. Use the InfoSphere CDC configuration tool to work with instances.

To add a new instance of InfoSphere CDC (Windows)

Procedure

1. If you are configuring the first instance of InfoSphere CDC after installation, you can proceed to Step 3 of this procedure.
2. At the command prompt, launch the configuration tool by issuing the following command in the specified directory:

   ```
   \<InfoSphere CDC Installation Directory>\bin\dmconfigurets
   ```
3. At the welcome message, click **OK** to continue.
4. On the **IBM InfoSphere CDC New Instance** dialog box, you can configure the following options in the **Instance** area:

Option	Description
Name	Type a name for your InfoSphere CDC instance. This name must be unique.
Server Port	Type the port number which InfoSphere CDC uses for communication with client workstations running Management Console and other servers. **Note:** This port number cannot be used by other applications installed on the same server. You will use this port number when specifying access parameters for your datastore in the Access Manager perspective in Management Console. InfoSphere CDC displays a default TCP/IP port of 11101. For more information, see your Management Console documentation. **Note:** If you install several instances on the same node, the port number for each instance must be unique.
Auto-Discovery Port	Select the box and type the UDP port number that is used for auto-discovery broadcasts sent from Access Server. For more information about auto-discovery, see your Management Console documentation.
Maximum Memory Allowed	Type the maximum amount of RAM you want to allocate for InfoSphere CDC. You must allocate a minimum of 64 MB for each instance you configure. By default, there is 512 MB of RAM allocated for a 32 bit instance and 1024 MB of RAM allocated for a 64 bit instance.

Option	Description
Staging Store Disk Quota (GB)	Enter the maximum amount of disk space that will be utilized by the InfoSphere CDC staging store on your source system. The default value is 100 GB and minimum value is 1 GB. Specify 1 GB if you are creating an instance that will be used as a target of replication. This reduces the disk resources that InfoSphere CDC requires on your target system.
Bit-Version	Select the bit-version of your database by selecting one of the following options: • 32 bit • 64 bit These options are not enabled if you are installing InfoSphere CDC on a 32-bit server.

5. In the **Windows Service** area, you can specify the account that will be used to start InfoSphere CDC services. Select one of the following options:

Option	Description
Local System account	Start InfoSphere CDC services through the local system administrator account.
This account	Start InfoSphere CDC services through the specified user account. The account must be specified in the format *<domain>\<user name>*, where *<domain>* is the name of a domain in your environment, and *<user name>* is a valid login user name in the specified domain. If your computer is not part of a domain, you can specify *<computer name>\<user name>*. In the **Password** and **Confirm Password** boxes, type the password currently associated with the selected Windows user account. If you change the password for the Windows user account after installing InfoSphere CDC, you will have to use the Windows Services dialog to change the password currently set for each InfoSphere CDC service.

6. In the **Database** area, you can configure access to the database that contains the tables for replication. To complete this step, you will require system administrator privileges. You can then add a datastore in the Access Manager perspective in Management Console and provide users access to this database. For more information, see your Management Console documentation.

Option	Description
User name	Type the user name for the specified database.

Option	Description
Password	Type the password for the specified database.
Metadata Schema	Select the schema for the database that will be used for the InfoSphere CDC metadata tables. As a default, the user name entered above is used. You can specify any schema except those in use by other installed instances of InfoSphere CDC for the given database. You must set up or decide on this schema as part of the installation prerequisites. **Note:** Make sure to use UPPERCASE letters for the metadata schema. By default, all the schema names (catalog names) in solidDB are in uppercase.
Advanced	The **Advanced** button enables you to modify configuration parameters for the solidDB JDBC driver. For more information about the JDBC driver parameters, see the *IBM solidDB Programmer Guide*. **Tip:** • In HA setup, the parameter **`solid_tf_level`** is by default set to `CONNECTION`. • In SMA setup, the parameter **`solid_shared_memory`** is by default set to `yes`. • To enable use of operating-system-based external authentication, set the following properties: – solid_use_strong_encryption=yes – solid_gskit_path=*location_of_GSKit_library* **Important:** To authenticate users using the operating-system-based authentication mechanisms, the IBM Global Security Kit (GSKit) must be enable on both the server and client computers.

7. In the **Server** area, you can configure the solidDB server that you want to replicate data to or from and which contains all of the tables for replication. You can configure either single server or HA configuration (HotStandby).

Option	Description
Single server	Type the host name and port number for the specified solidDB server.
Enable SMA	Select the check box if you are using solidDB with shared memory access (SMA).
HA Configuration (HotStandby)	Type the host names and port numbers for the specified Primary and Secondary solidDB servers.

8. Click **OK** to save your configuration settings for the InfoSphere CDC instance.

9. If InfoSphere CDC has detected an unsupported encoding, a dialog will open asking you to select an alternate encoding from a list.

   You can filter the list of alternate encodings by clicking one of the following buttons:

   - **Closest match**—Displays the alternated encodings that are the closest match to the data.
   - **Comparable encodings byte length**—Displays the alternate encodings in order of byte length.
   - **All**–Displays all alternate encodings.

   Select an encoding from the list and click **OK**.

   If you click **Cancel**, an error message will be displayed and the instance will not be created.

What to do next

After you complete the configuration, you can start InfoSphere CDC.

To edit an instance of InfoSphere CDC (Windows)

Procedure

1. Stop InfoSphere CDC if it is started by using the `dmshutdown` command.
2. Launch the configuration tool at the command prompt by issuing the following command in the specified directory:

   ```
   \<InfoSphere CDC Installation Directory>\bin\dmconfigurets
   ```

3. In the **Instances** area, select the instance that you want to modify and click **Stop** if the instance is started.
4. In the **Instances** area, select an instance and click **Edit**.

   The **InfoSphere CDC Edit Instance** dialog opens.
5. You can modify any of the values on this dialog box that you specified when adding an instance.
6. Click **Apply** to save your changes and then click **Close**.

   The configuration tool will modify the instance.
7. In the **Instances** area, select the instance that you modified and click **Start** to start the instance.

To delete an instance of InfoSphere CDC (Windows)

Procedure

1. Stop InfoSphere CDC if it is started by using the `dmshutdown` command.
2. At the command prompt, launch the configuration tool by issuing the following command in the specified directory:

   ```
   \<InfoSphere CDC Installation Directory>\bin\dmconfigurets
   ```

3. In the **Instances** area, select the instance that you want to delete and click **Stop** if the instance is started.
4. In the **Instances** area, select an instance and click **Delete**.
5. Click **Yes** to permanently delete the instance.

10.3.2 Configuring InfoSphere CDC instances (UNIX and Linux)

You can add, edit, or delete an instance of InfoSphere CDC. Use the InfoSphere CDC configuration tool to work with instances.

To add a new instance of InfoSphere CDC (UNIX and Linux)

Procedure

1. If you are configuring the first instance of InfoSphere CDC after installation, you can proceed to Step 3 of this procedure.
2. At the command prompt, launch the configuration tool by issuing the following command in the specified directory:
   `/<InfoSphere CDC Installation Directory>/bin/dmconfigurets`
3. At the welcome message, press **Enter** to continue.
4. Type 2 and press **Enter** to add a new instance of InfoSphere CDC.
5. Type a name for your InfoSphere CDC instance and press **Enter**. The instance name must be unique.
6. Type the port number which InfoSphere CDC uses for communication with client workstations running Management Console and other servers. InfoSphere CDC displays a default port of `11101`. Press **Enter**.

   Note: This port number cannot be used by other applications installed on the same server. You will use this port number when specifying access parameters for your datastore in the Access Manager perspective in Management Console. For more information, see your Management Console documentation.

   Note: If you install several instances on the same node, the port number for each instance must be unique.
7. If you are using the auto-discovery feature in Access Manager, then enable the this feature by typing the UDP port number that you set in Access Server. InfoSphere CDC uses this UDP port number for auto-discovery broadcasts sent from Access Server. Otherwise, press **Enter** to disable this feature.
8. Type the amount of physically available RAM you want to allocate for InfoSphere CDC. You must allocate a minimum of 64 MB for each instance you configure. By default, there is 512 MB of RAM allocated for a 32 bit instance and 1024 MB of RAM allocated for a 64 bit instance.
9. Select the solidDB server configuration type you want to configure.

Option	Description
Single server	Type 1 and press **Enter**.
HA Configuration (HotStandby)	Type 2 and press **Enter**.

10. Type the host name and port number according to your configuration type.

Option	Description
Single server	1. Type the host name for the specified server and press **Enter**. 2. Type the port number for the specified server and press **Enter**. Default is 1964.

Option	Description
HA Configuration (HotStandby)	1. Type the host name for the specified Primary server and press **Enter**. 2. Type the port number for the specified Primary server and press **Enter**. Default is 1964. 3. Type the host name for the specified Secondary server and press **Enter**. 4. Type the port number for the specified Secondary server and press **Enter**. Default is 1964. **Note:** The default port number for Primary and Secondary is the same as it is assumed that the Primary and Secondary are located on different nodes. If, for example, for evaluation purposes, your Primary and Secondary servers are located on the same node, the default port number for both cannot be the same.

11. Select to enable the use of solidDB with shared memory access (SMA) as necessary.

Option	Description
Use default settings	Type `n` and press **Enter**.
Enable SMA	Type `y` and press **Enter**.

12. Configure advanced parameters (JDBC parameters) as necessary.

Option	Description
Use default settings	Type `n` and press **Enter**.
Modify settings	1. Type `y` and press **Enter** 2. Enter the parameter settings using the syntax `<parameter>=<value>;<parameter>=<value>;...` **Tip:** • In HA setup, the parameter **`solid_tf_level`** is by default set to `CONNECTION`. • In SMA setup, the parameter **`solid_shared_memory`** is by default set to `yes`. • To enable use of operating-system-based external authentication, set the following properties: – solid_use_strong_encryption=yes – solid_gskit_path=*location_of_GSKit_library* **Important:** To authenticate users using the operating-system-based authentication mechanisms, the IBM Global Security Kit (GSKit) must be enable on both the server and client computers.

13. Type the user name for the specified database and press **Enter**.
14. Type the password for the specified database and press **Enter**. The configuration tool will now search the database for schemas.
15. Type the number that corresponds to the metadata schema that you would like to use and press **Enter**.

16. Type the path to the directory that will be used for bulk inserts into the database. Press **Enter**. Both your solidDB database and InfoSphere CDC must have read and write permissions for this directory.

    Notes:
    - You should use a different directory for each instance of InfoSphere CDC.
    - This directory may contain database tables for replication. You should take this into consideration when determining user access to this directory.
17. If InfoSphere CDC detects an unsupported encoding, an error message will be displayed and you will be asked to choose an alternate encoding.
    a. Enter `y` to proceed.

       Note: If you enter `n` and press **Enter** to cancel, the instance will not be created.
    b. Enter a value to choose how the alternate encodings will be displayed:
       - **1**—Displays the available alternate encodings that are the closest match to the database.
       - **2**—Displays the available alternate encodings in order of byte length.
       - **3**—Displays all available alternate encodings.
    c. Enter the number for the encoding to be used and press **Enter**.
18. The configuration tool creates the InfoSphere CDC instance and prompts you to start the instance. Type `y` to start the instance.

    Note: The configuration tool will prompt you if your configuration is about to overwrite the metadata for an existing instance.

To edit an instance of InfoSphere CDC (UNIX and Linux)

Procedure

1. Stop InfoSphere CDC if it is started by using the `dmshutdown` command.
2. Launch the configuration tool by issuing the following command in the specified directory:

   ```
   /<InfoSphere CDC Installation Directory>/bin/dmconfigurets
   ```
3. Type `1` and press **Enter** to list the installed instances of InfoSphere CDC. Record the name of the instance you want to modify.
4. Type `3` and press **Enter** to modify an instance of InfoSphere CDC.
5. Type the instance name that you want to modify and press **Enter**.

   The configuration tool allows you to edit a number of values that you specified when adding an instance.
6. After making your changes, type `5` and press **Enter** to apply your changes and return to the main menu. Type `6` and press **Enter** to discard your changes.

To delete an instance of InfoSphere CDC (UNIX and Linux)

Procedure

1. Stop InfoSphere CDC if it is started by using the `dmshutdown` command.
2. Launch the configuration tool by issuing the following command in the specified directory:

   ```
   /<InfoSphere CDC Installation Directory>/bin/dmconfigurets
   ```
3. Type `1` and press **Enter** to list the installed instances of InfoSphere CDC. Record the name of the instance you want to delete.
4. Type `4` and press **Enter** to delete an instance of InfoSphere CDC.

5. Type the instance name that you want to delete and press **Enter**.

10.4 Starting and stopping InfoSphere CDC

This section provides step-by-step instruction on how to start and stop InfoSphere CDC instances.

10.4.1 Starting InfoSphere CDC

When you install InfoSphere CDC on a supported Windows server, you can start it manually after the initial configuration. Starting InfoSphere CDC starts the services in Windows. The services will automatically start after a reboot.

When you install InfoSphere CDC on a supported Linux server, you can issue a command to start it. After installing InfoSphere CDC, start it so that you can create a datastore for this instance in Management Console.

To start InfoSphere CDC (Windows)

Procedure

1. At the command prompt, launch the configuration tool by issuing the following command in the specified directory:

   ```
   \<InfoSphere CDC Installation Directory>\bin\dmconfigurets
   ```

2. In the **Instances** area, select the instance that you want to start and click **Start**.

   The configuration tool starts the instance of InfoSphere CDC.

What to do next

You can also use the Windows Services dialog to start and stop InfoSphere CDC services.

To start InfoSphere CDC (UNIX and Linux)

Procedure

Depending on the operating system you are running InfoSphere CDC, issue one of the following start commands:

- `dmts32 -I <instance_name>`
- `dmts64 -I <instance_name>`

10.4.2 Stopping InfoSphere CDC

It may be necessary to stop InfoSphere CDC when you want to change the configuration settings using the InfoSphere CDC configuration tool.

On Windows, stopping InfoSphere CDC stops the services in Windows. The services will automatically start again after a reboot.

On UNIX and Linux, you can issue a command to stop InfoSphere CDC. Use the command prior to taking a server or database offline for maintenance purposes or for upgrading InfoSphere CDC.

To stop InfoSphere CDC (Windows)

Procedure

1. Launch the configuration tool by issuing the following command in the specified directory:

   ```
   /<InfoSphere CDC Installation Directory>/bin/dmconfigurets
   ```

2. In the **Instances** area, select the instance that you want to stop and click **Stop**.
   The configuration tool stops the instance of InfoSphere CDC.

What to do next

You can also use the Windows Services dialog to start and stop InfoSphere CDC services.

To stop InfoSphere CDC (UNIX and Linux)

Procedure

1. End replication on all subscriptions in Management Console. For more information about how to end replication on subscriptions, see your Management Console documentation.
2. Depending on how you want to stop InfoSphere CDC, issue one of the following stop commands:

Option	Description
dmshutdown -I <*instance_name*>	Use this command to gracefully shut down InfoSphere CDC. If you have multiple active InfoSphere CDC installations on the same Linux server, and you want to shut them all down, run this command from the installation directory for each InfoSphere CDC instance.
dmterminate -I <*instance_name*>	Use this command to terminate all InfoSphere CDC processes for all instances running on a Linux server. Use this command when you cannot completely shut down InfoSphere CDC using the `dmshutdown` command.

10.4.3 Enabling SQL statements in Management Console

InfoSphere CDC lets you execute SQL statements after it applies a table-level clear or refresh operation to a target table. You can specify SQL statements in the **Additional SQL** dialog box in Management Console. By default, this feature is disabled in InfoSphere CDC for security reasons. You can enable this feature by creating a table called TS_SQL_EXECAUTH in the database where you installed InfoSphere CDC. The structure of the table is unimportant, and you must create the table using the same schema as the metadata tables during the configuration of InfoSphere CDC. For more information about specifying SQL statements in Management Console, see "Specifying SQL to control refresh operations" in your Management Console documentation.

To enable SQL statements in Management Console

Procedure

1. Locate the database on the target server that you created for InfoSphere CDC. Depending on how you are using InfoSphere CDC, this is the database you want InfoSphere CDC to replicate to or from.

   Note: During installation, InfoSphere CDC places metadata tables in the database necessary for InfoSphere CDC processes.
2. If you want to enable the specification of SQL statements, create a table named TS_SQL_EXECAUTH in the database.

Note: The table can have any structure and must be created in the schema you specified when you configured InfoSphere CDC.

10.5 Datatypes supported by InfoSphere CDC

When you map source and target columns for replication, you should know which data types are compatible.

10.5.1 Supported data types

This section identifies the data types that InfoSphere CDC can replicate. All solidDB data types are supported for replication.

- bigint
- binary
- blob
- char
- clob
- date
- decimal
- double precision
- float
- integer
- long varbinary
- long varchar
- nchar
- nclob
- numeric
- nvarchar
- real
- smallint
- time
- timestamp
- tinyint
- varbinary
- varchar
- wchar
- wvarchar

10.5.2 Supported mappings

This section indicates the supported Management Console mappings for supported data types.

Published data types	Supported mapping
bigint	Any numeric, binary or LOB data type
binary	Any binary or LOB data type
blob	Any binary or LOB data type
char	Any character, variable character, CLOB, binary, or other LOB data type

Published data types	Supported mapping
clob	Any character, variable character, CLOB, binary, or other LOB data type
date	Any data type
decimal	Any numeric, binary or LOB data type
double precision	Any numeric, binary or LOB data type
float	Any numeric, binary or LOB data type
integer	Any numeric, binary or LOB data type
long varbinary	Any binary or LOB data type
long varchar	Any character, variable character, CLOB, binary, or other LOB data type
nchar	Any character, variable character, CLOB, binary, or other LOB data type
nclob	Any character, variable character, CLOB, binary, or other LOB data type
nvarchar	Any character, variable character, CLOB, binary, or other LOB data type
numeric	Any numeric, binary or LOB data type
real	Any numeric, binary or LOB data type
smallint	Any numeric, binary or LOB data type
tim	Any time data type
timestamp	Any date, time, or timestamp data types
tinyint	Any numeric, binary or LOB data type
varbinary	Any binary or LOB data type
varchar	Any character, variable character, CLOB, binary, or other LOB data type

10.6 InfoSphere CDC metadata tables

InfoSphere CDC maintains a set of tables that represent data about your current replication configuration. These tables are created in the schema and database that you specify in the configuration tool and should be part of the backup strategy for your database. InfoSphere CDC will not replicate these tables. Do not modify the contents of these tables unless requested to do so by your IBM representative.

The names of the metadata tables created by InfoSphere CDC are as follows:

- TS_AUTH

  Note: For all users you added in the Access Manager perspective in Management Console, make sure you give GRANT SELECT privileges to the TS_AUTH metadata table. For more information about how to add users in the Access Manager perspective in Management Console, see your Management Console documentation.
- TS_BOOKMARK
- TS_CONFAUD

  The conflict resolution audit table records information about conflicts that were resolved using conflict detection and resolution.
- TS_AGED_TABLES

This metadata table is specific to InfoSphere CDC for solidDB. It maintains information about the aging status of tables in the solidDB frontend.

- TS_REFRESH

  This metadata table is specific to InfoSphere CDC for solidDB. It maintains information about the refresh status of data in the solidDB frontend.

10.7 Commands for InfoSphere CDC

This section discusses the commands available with InfoSphere CDC. Using these commands you can control replication, manage your tables for replication, monitor replication, and perform various other tasks.

10.7.1 Using the InfoSphere CDC commands

You can issue InfoSphere CDC commands at a command line prompt or as part of a batch file or shell script. Commands are located in the `bin` directory of your InfoSphere CDC installation directory. Navigate to this directory to run the commands.

Note: To list the available flags for a command and a short description of each flag, type the name of the command at a command prompt with the `-?` flag and press **Enter**. For example, `dmterminate -?`.

Command formats

For each command, the following items of information are provided:

- **Syntax**—Identifies the name of the command and lists the command parameters.
- **Parameters**—Describes each parameter in the command and identifies the values that can be specified.
- **Result**—Indicates the values that are returned by the command if it is successful. These values can be useful for scripting. This section also specifies the information that is displayed on the screen, if any, as a result of executing the command.
- **Examples**—Provides one or more examples of invoking the command.

Parameter formats

Note the following conventions in the definition of the command parameters:

- Angle brackets (< >) indicate a **mandatory** parameter.
- Square brackets ([ ]) indicate an **optional** parameter. If you omit the parameter, InfoSphere CDC uses a default value.
- A vertical bar (|) separating one or more parameters indicate that only one of the parameters in the list can be used. When one or more vertical bars appear in a list of parameters that is enclosed by square brackets [ ], the choices are limited to the parameters in the list, but you have the option to not specify any of the parameters.
- Ellipsis (...) means that a parameter or option can be repeated more than once.
- Unless otherwise noted, the commands apply to all operating systems.

10.7.2 Setting the TSINSTANCE environment variable

Before using the commands, you can set the `TSINSTANCE` environment variable to the name of your InfoSphere CDC instance.

After you set the TSINSTANCE environment variable, you no longer have to specify the instance name when issuing commands.

Windows platform

Issue the following command at the command prompt:

```
SET TSINSTANCE=<instance_name>
```

where:

- <instance_name> is the name of your InfoSphere CDC instance.

Linux platform

Issue the following command:

```
EXPORT TSINSTANCE=<instance_name>
```

where:

- <instance_name> is the name of your InfoSphere CDC instance.

10.7.3 Controlling replication commands

This section contains commands that control replication in InfoSphere CDC.

dmendreplication - End Replication

Use this command to end refresh or mirroring on the specified subscriptions.

Ending replication allows you to prepare for transitional activities in your business environment and allows you to move to the next step in your business processes. Here are some examples of transitional activities in your business environment that may require an end to replication:

- Initiating a database backup.
- Performing a regularly scheduled reboot of your source database server.
- Quiescing your database in preparation for an upgrade.
- Weekly batch processing has just completed.
- Preparing for offline maintenance activities.

If you are replicating data continuously with Continuous mirroring and business reasons arise that require an end to replication, InfoSphere CDC provides multiple options that suit most business needs. If your business requirements dictate that replication must end at a particular point in your source database log because the target database must be in a known state when replication ends, you can choose from the following Scheduled End to replication options:

- -se parameter—When specified without –t, this parameter ends replication at the current time in the source database log.
- -t parameter—When specified with –se, this parameter ends replication at a user-specified date and time.

An example of a scenario that might require these options is that you are populating a reporting instance and you need stable (non-changing) data in your reporting instance during the day. At the end of the day when you shut down your application, you can choose one of the Scheduled End (Net Change) options to update the reporting instance with data from the current day as well.

If business requirements do not require a specific end point but do require a time frame for ending replication, InfoSphere CDC provides escalating options (Normal, Immediate, and Abort) that end replication more rapidly at the expense of a slower start when resuming replication. For example, a routine end to replication with no particular urgency may require the Normal option, whereas a sudden business need to end replication rapidly may require the Abort option. A routine reboot of a SAN might be appropriate for the Normal option, whereas a sudden and unexpected hardware or application failure may require the Abort option.

If you initiate an end to replication and business reasons warrant a change in the desired time frame, you can reschedule the end of replication by specifying a new date and time, a new position in the database log, or choose another option for ending replication.

Ending replication is also necessary if you want to update and make changes to your subscription by:

- Adding a table mapping to the subscription.
- Deleting a table mapping from the subscription.
- Temporarily removing a table mapping from the subscription (parking a table).
- Modifying mapping details such as source and target column mappings, derived columns, data translations, row and column selections, user exits, and so on.
- Updating the properties of a subscription when the structure of your source or target tables change.

This command also includes an asynchronous option for scripting (`-nw` parameter) that can be used with `-se` to allow your script to continue executing without waiting for the Scheduled End to replication.

You can also start and end replication in Management Console.

To stop an instance after ending replication on all subscriptions, use the **dmshutdown** command.

Syntax

```
dmendreplication [-I <INSTANCE_NAME>] [-c|-i|-a|-se [-t <date and time>]
[-w|-nw]] <-A|-s <SUBSCRIPTION NAME ...> [-L <locale>]
```

Parameters

-I <INSTANCE_NAME>
Specifies the InfoSphere CDC instance for which you want to end replication. Alternatively, you can specify the TSINSTANCE environment variable in place of this value.

-c Specifies that InfoSphere CDC ends replication on the specified subscriptions with the Normal option. InfoSphere CDC will use this option by default if you do not specify `–se`, `-i`, or `–a`.

This option completes in progress work and then ends replication. If a refresh is in progress, Normal will complete the refresh for the current table before replication ends.

Normal is the most appropriate option for most business requirements and is the preferred method for ending replication in most situations.

-i Specifies that InfoSphere CDC ends replication on the specified subscriptions with the Immediate option.

This option stops all in progress work and then ends replication. Starting replication after using this option can be slower than using `-c`. If a refresh is in progress, the refresh for the current table will be interrupted and then replication will end.

You should ensure that all dependent source database logs are available before ending replication using the Immediate option. InfoSphere CDC may need to reprocess all the dependent source logs when you restart the subscription. If InfoSphere CDC is currently processing a long running transaction when you end replication with Immediate, InfoSphere CDC may have to resume replication from the earliest open transaction in the database logs. Use the **dmshowlogdependency** command to determine which logs are required.

Note: Use this option if business reasons require replication to end faster than `-c` at the expense of a slower start when you resume replication on the specified subscriptions.

-a Specifies that InfoSphere CDC ends replication on the specified subscriptions with the Abort option.

This option stops all in progress work and then ends replication rapidly. Starting replication after using this option can be much slower than using `-c`. A refresh in progress will be interrupted and the target will stop processing any data that has not been committed before replication ends.

You should ensure that all dependent source database logs are available before ending replication using the Abort option. InfoSphere CDC may need to reprocess all the dependent source logs when you restart the subscription. If InfoSphere CDC is currently processing a long running transaction when you end replication with Abort, InfoSphere CDC may have to resume replication from the earliest open transaction in the database logs. Use the **dmshowlogdependency** command to determine which logs are required.

Note: Use this option if your business reasons require a rapid end to replication and you are willing to tolerate a much slower start when you resume replication on the specified subscriptions.

A sudden business requirement for an unplanned shutdown of your source system may require this option for ending replication.

-se

Specifies that InfoSphere CDC will end replication normally at the current source system time in the source database log with the Scheduled End option. The source system time when replication will end is set when you issue this command.

If you specify the following parameters with `-se`, replication will end at a specific date and time or log position:

- –t—End replication at a specific date and time in your source database log.

Note: As latency between the source and target increases, the amount of time required to end replication will also increase.

-t <date and time>

Indicates the date and time in the source database log when replication will end when using `–se`. When specifying a value for this parameter, use the following format:

```
"yyyy-MM-dd HH:mm"
```

This parameter is optional when you specify `–se`.

-w Indicates that this command will wait for replication to end when you use –se. –w is the default setting for a Scheduled End to replication.

If you are scripting the command with this parameter, your script must wait for -se processing to complete before it continues to execute.

Note: This parameter does not apply if you specify –c, -i, or –a. InfoSphere CDC will always wait if you specify –c, -i, or –a when ending replication.

-nw
Indicates that this command will not wait for replication to end if you specify -se. If you are scripting this command, this parameter allows your script to continue executing (asynchronous) if -se processing is not complete.

-A Indicates that InfoSphere CDC ends replication on all subscriptions. Use –s to end replication on one or more subscriptions.

-s <SUBSCRIPTION NAME>
Indicates the subscriptions where InfoSphere CDC will end replication.

To specify multiple subscriptions, list the subscriptions separated by a space. For example:

```
Subscription1 Subscription2 Subscription3
```

You must specify a value for this parameter or use –A for all subscriptions.

-L <locale>
The name of the locale used for the InfoSphere CDC instance. The default is the locale of the machine where InfoSphere CDC is installed.

Result

This command returns a value of 0 if the command was successful and a non-zero value if the command fails.

Examples

```
dmendreplication -I MYINSTANCE -c -s FINANCE
```

InfoSphere CDC ends replication with the Normal option for the FINANCE subscription in the specified instance.

```
dmendreplication -I MYINSTANCE –se –t "2010-02-05-00-00" FINANCE -nw
```

InfoSphere CDC ends replication with the Scheduled End option for the FINANCE subscription at the specified time in the source database log. The command exits before Scheduled End processing is complete.

```
dmendreplication -I MYINSTANCE –a –s SUBSCRIPTION1 SUBSCRIPTION2
```

InfoSphere CDC ends replication with the Abort option for SUBSCRIPTION1 and SUBSCRIPTION2 in the specified instance.

dmrefresh - Refresh Subscription

Use this command to refresh the specified subscriptions. When you refresh a subscription, InfoSphere CDC ensures that the target tables are synchronized with the source tables. Typically, you would refresh target tables when you have set the replication method to **Refresh** on your tables.

However, you can also refresh target tables that have a replication method set to **Mirror** and a status of **Active** or **Refresh**. When you refresh a table configured for

mirroring, InfoSphere CDC refreshes the target table so that it is synchronized with the source table and then marks a table capture point as the starting point for mirroring.

This command exits after it has successfully refreshed the specified subscriptions. If you terminate this program while it is still running, InfoSphere CDC ends replication immediately for the specified subscriptions.

Syntax

```
dmrefresh -I <instance_name> [-a|-f] <-A|-s <subscription_names> ...> [-L <locale>]
```

Parameters

-I <instance_name>
Specifies the InfoSphere CDC instance for which you want to refresh one or more subscriptions. Alternatively, you can specify the TSINSTANCE environment variable in place of this value.

-a Specifies that InfoSphere CDC refreshes all target tables in the subscription.

-f Specifies that InfoSphere CDC refreshes only target tables that are flagged for refresh. If you omit both the `-a` and `-f` options, InfoSphere CDC assumes `-f` by default.

-A Specifies that InfoSphere CDC refreshes all subscriptions.

-s <subscription_names>
Specifies that InfoSphere CDC refreshes the indicated subscription. To specify multiple subscriptions, list the subscriptions separated by a space.

-L <locale>
The name of the locale used for the InfoSphere CDC instance. The default is the locale of the machine where InfoSphere CDC is installed.

Result

This command returns a value of 0 if the command was successful and a non-zero value if the command fails.

Examples

```
dmrefresh -I new_instance -a -s Finance
```

InfoSphere CDC refreshes all target tables in the **Finance** subscription.

dmstartmirror - Start Mirroring

Issue this command from your InfoSphere CDC source to start mirroring on the specified subscriptions. This command starts mirroring for any table with a replication method of **Mirror** and a status of **Refresh** or **Active**. Tables with a replication method of Mirror and a status of Refresh are refreshed before mirroring begins.

InfoSphere CDC provides two types of mirroring for source tables that are mapped to target tables: Continuous (`-c` parameter) and Scheduled End (Net Change) (`-n` parameter). The type of mirroring you select depends on your business needs.

As its name implies, Continuous mirroring replicates changes to the target on a continuous basis. Use this type of mirroring when business requirements dictate that you need replication to be running continuously and you do not have a clearly defined reason to end replication at the present time.

Scheduled End (Net Change) mirroring replicates changes (to the target) up to a user-specified point in the source database log and then ends replication. Use this type of mirroring when business requirements dictate that you only replicate your data periodically and you have a clearly defined end point for the state of your target database when replication ends.

Scheduled End (Net Change) mirroring allows you to end replication at the following points in your source database log:

- -n parameter—When specified without –t, this parameter ends replication at the current time in the source database log.
- -t parameter—When specified with –n, this parameter ends replication at a user-specified date and time.

These user specified end points ensure that your target database is in a known state when replication ends.

Syntax

```
dmstartmirror [-I <INSTANCE_NAME>] [-c|-n [-t <date and time>]
[-w|-nw]]  <-A|-s <SUBSCRIPTION NAME ...> [-L <locale>]
```

Parameters

-I <INSTANCE_NAME>
Specifies the InfoSphere CDC instance for which you want to start mirroring. Alternatively, you can specify the TSINSTANCE environment variable in place of this value.

-c Specifies that InfoSphere CDC will start Continuous mirroring on the specified subscriptions.

If you do not specify –c or -n, InfoSphere CDC will start Continuous mirroring by default on the specified subscriptions.

-n Specifies that InfoSphere CDC mirrors all committed database changes in the source database and then ends replication normally at the current source system time in the database log with the Scheduled End option.

-t <date and time>
Indicates the date and time in the source database log when replication will end when using –n.

When specifying a value for this parameter, use the following format:

```
"yyyy-MM-dd HH:mm"
```

This parameter is optional when you specify –n.

-w Indicates that this command will wait for replication to end when you use –n. –w is the default setting for a Scheduled End to replication.

If you are scripting the command with this parameter, your script must wait for -n processing to complete before it continues to execute.

This parameter does not apply if you specify –c for Continuous mirroring.

-nw
Indicates that this command will not wait for replication to end if you specify -n.

If you are scripting this command, this parameter allows your script to continue executing (asynchronous) if -n processing is not complete.

This parameter does not apply if you specify –c for Continuous mirroring.

-A Indicates that InfoSphere CDC starts mirroring for all subscriptions.

Use –s to start mirroring for one or more subscriptions.

-s <SUBSCRIPTION NAME>
Indicates the subscriptions where InfoSphere CDC will start mirroring. To specify multiple subscriptions, list the subscriptions separated by a space. For example:

```
Subscription1 Subscription2 Subscription3
```

You must specify a value for this parameter or use **–A** for all subscriptions.

-L <locale>
The name of the locale used for the InfoSphere CDC instance. The default is the locale of the machine where InfoSphere CDC is installed.

Result

This command returns a value of 0 if the command was successful and a non-zero value if the command fails.

Examples

```
dmstartmirror -I MYINSTANCE -c -s FINANCE
```

InfoSphere CDC starts continuous mirroring for the FINANCE subscription.

```
dmstartmirror -I MYINSTANCE –n –t "2010-02-05-00-00" FINANCE -nw
```

InfoSphere CDC starts mirroring with the Scheduled End option for the FINANCE subscription in the MYINSTANCE instance. Replication will end at the specified time in the source database log. The command will exit before Scheduled End processing is complete.

10.7.4 Database transaction log commands

This section contains commands that help you manage your database transaction log or bookmarks.

dmdecodebookmark - Display Verbose Bookmark Information

Use this command to display verbose information about a bookmark.

Syntax

```
dmdecodebookmark -I <instance_name> (-b | -f) [-L <locale>]
```

Parameters

-I <instance_name>
The name of the InfoSphere CDC instance. You can set the TSINSTANCE environment variable to the name of your InfoSphere CDC instance. After this is complete, you no longer have to specify the instance when issuing commands.

-b <bookmark>
The bookmark as a hexadecimal-encoded string.

-f <bookmark_file>
The bookmark file as a binary file.

-L <locale>
The name of the locale used for the InfoSphere CDC instance. The default is the locale of the machine where InfoSphere CDC is installed.

Result

This command returns a value of 0 if the command was successful and a non-zero value if the command fails.

Examples

```
dmdecodebookmark -f bookmark.txt
```

InfoSphere CDC displays information about the `bookmark.txt` file.

dmsetbookmark - Set Bookmark

Use this command on your InfoSphere CDC source system to set the replication position (bookmark) in the stream of change data for a subscription. You can obtain the replication position for a subscription with the **dmshowbookmark** command, which is executed on your InfoSphere CDC target system. More information on the InfoSphere CDC stream of change data is provided in the following paragraphs.

InfoSphere CDC parses the data from your database logs and creates a stream of change data to process on the source and eventually apply on the target. The stream of change data is sorted in the order in which the data was committed in the source database, whereas the data in your database logs is sorted in the order in which the individual action was done in the source database.

For example, two transactions named T1 and T2 may be ordered like this in your source database log:

```
T1: Insert1
T2: Insert1
T2: Insert2
T2: Commit
T1: Commit
```

As you can see, data is sorted in the database log according to when the individual action was done in your source database.

However, the InfoSphere CDC stream of change data will order the two transactions like this:

```
T2: Insert1
T2: Insert2
T2: Commit
T1: Insert1
T1: Commit
```

Data is sorted according to when the data is committed in your source database.

Syntax

```
dmsetbookmark [-I <INSTANCE_NAME>] -s <SUBSCRIPTION_NAME ...> (-b <bookmark> | -f
<bookmark_file_name>) [-a] [-L <locale>]
```

Parameters

-I <INSTANCE_NAME>
The name of the InfoSphere CDC instance. You can set the TSINSTANCE

environment variable to the name of your InfoSphere CDC instance. After this is complete, you no longer have to specify the instance when issuing commands.

-s <SUBSCRIPTION_NAME>
The name of the subscription for which InfoSphere CDC sets a bookmark.

-b <bookmark>
Indicates the name of the binary or XML file that contains all replication position (bookmark) information which determines the point in the database log where you want InfoSphere CDC to resume mirroring. When mirroring resumes, InfoSphere CDC will start capturing change data at the replication position indicated in the file. You can specify an absolute path for the location of the file. If you do not specify an absolute path, you must place the file in the InfoSphere CDC installation directory. InfoSphere CDC will auto-detect the binary or XML format of the file.

Specifies the bookmark which determines the point in the database log where you want InfoSphere CDC to resume mirroring. The next time InfoSphere CDC mirrors, it will scrape at the given position. The bookmark is a hex encoded string that is obtained from the `dmshowbookmark` command.

-l <bookmark>
Bookmark indicating the new scraping point. The bookmark is a string obtained from the `dmdecodebookmark` command. For more information, see "dmdecodebookmark - Display Verbose Bookmark Information" on page 125.

-f <bookmark_file>
Specifies the binary file containing a bookmark which determines the point in the database log where you want InfoSphere CDC to resume mirroring. The next time InfoSphere CDC mirrors, it will scrape at the given position. The bookmark file is a binary file that stores the position.

-a Sets all tables in the subscriptions (except for parked tables) as active as of the new scraping point.

-L <locale>
The name of the locale used for the InfoSphere CDC instance. The default is the locale of the machine where InfoSphere CDC is installed.

Result

This command returns a value of 0 if the command was successful and a non-zero value if the command fails

Examples

```
dmsetbookmark -I MYINSTANCE -b 2FC5GJHKLKSJLKJL458K9K809IK9
-s FINANCE
```

InfoSphere CDC sets a bookmark position on the **Finance** subscription for the specified instance. This command specifies that mirroring will resume at the indicated position in the database log.

dmshowbookmark - Display Bookmark Information

Use this command on your InfoSphere CDC target system to obtain the replication position (bookmark) in the stream of change data for a subscription. After generating the replication position information with this command, you can use the **dmsetbookmark** command on the source system to set the replication position for a subscription. More information on the InfoSphere CDC stream of change data is provided in the following paragraphs.

InfoSphere CDC parses the data from your database logs and creates a stream of change data to process on the source and eventually apply on the target. The stream of change data is sorted in the order in which the data was committed in the source database, whereas the data in your database logs is sorted in the order in which the individual action was done in the source database.

For example, two transactions named T1 and T2 may be ordered like this in your source database log:

```
T1: Insert1
T2: Insert1
T2: Insert2
T2: Commit
T1: Commit
```

As you can see, data is sorted in the database log according to when the individual action was done in your source database.

However, the InfoSphere CDC stream of change data will order the two transactions like this:

```
T2: Insert1
T2: Insert2
T2: Commit
T1: Insert1
T1: Commit
```

Data is sorted according to when the data is committed in your source database.

Syntax

```
dmshowbookmark [-I <INSTANCE_NAME>] -s <SOURCE_ID>
[-f <bookmark_file_name>] [-x <bookmark_file_name>] [-v] [-L <locale>]
```

Parameters

-I <INSTANCE_NAME>
The name of the InfoSphere CDC instance. You can set the TSINSTANCE environment variable to the name of your InfoSphere CDC instance. After this is complete, you no longer have to specify the instance when issuing commands.

-s <SOURCE_ID>
Specifies the source ID of the subscription for which you want to obtain the replication position (bookmark).

-f <bookmark_file_name>
Specifies the name of the binary file that will be generated by this command. The generated file contains information about the replication position (bookmark) for the specified subscription.

You can specify an absolute path for the location where you want to create the file. If you do not specify an absolute path, the file is created in the InfoSphere CDC installation directory.

Use the `-f` parameter in the **dmsetbookmark** command to read the binary file generated by this parameter.

-x <bookmark_file_name>
Specifies the name of the XML file that will be generated by this command. The generated file contains information about the replication position (bookmark) for the specified subscription.

You can specify an absolute path for the location where you want to create the file. If you do not specify an absolute path, the file is created in the InfoSphere CDC installation directory.

Use the `-f` parameter in the **dmsetbookmark** command to read the XML file generated by this parameter.

[-v]
Displays verbose information about the replication position (bookmark), including a hexadecimal-encoded string. The amount of information displayed depends on the type and version of the source engine. The hexadecimal-encoded string is always displayed. It is a subset of what the `dmdecodebookmark` command displays. If not specified, only a hexadecimal-encoded sting is displayed.

-L <locale>
The name of the locale used for the InfoSphere CDC instance. The default is the locale of the machine where InfoSphere CDC is installed.

Result

This command returns a value of 0 if the command was successful and a non-zero value if the command fails.

Examples

```
dmshowbookmark -I MYINSTANCE -s MASTER -f bookmark
```

InfoSphere CDC obtains the replication position (bookmark) information for the specified instance and the MASTER source ID. Replication position (bookmark) information is contained in the bookmark binary file which will be placed in the InfoSphere CDC installation directory since no absolute path has been specified.

```
dmshowbookmark -I MYINSTANCE -s FINANCE -x mybookmarks
```

InfoSphere CDC obtains the replication position (bookmark) information for the specified instance and the FINANCE source ID. Replication position (bookmark) information is contained in the `mybookmarks` XML file which will be placed in the InfoSphere CDC installation directory since no absolute path has been specified.

dmshowlogdependency - Show Log Dependency

Use this command to display information about the database logs that are used by InfoSphere CDC and are required for replication. Use this command to implement a log retention policy. With this command you can display the following information:

- A list of all the database logs that are required for the specified instance.
- The earliest open transaction in the database log for the specified instance.
- The database log that the specified instance of InfoSphere CDC is currently reading on the source.
- The database log for the subscription that the specified instance of InfoSphere CDC is currently applying on the target.

You must issue this command on your InfoSphere CDC source system.

Syntax

```
dmshowlogdependency -I <instance_name> (-c | -t | -l)
(-s <subscription_name> | -A) [-v] [-L <locale>]
```

Parameters

-I <instance_name>
The name of the InfoSphere CDC instance. You can set the TSINSTANCE environment variable to the name of your InfoSphere CDC instance. After this is complete, you no longer have to specify the instance when issuing commands.

-c Considers the current position instead of the restart position.

-t Displays the current database log for a subscription that the specified InfoSphere CDC instance is currently reading. This is the log containing the current position confirmed by the target.

-l Displays the current source database log that the specified InfoSphere CDC instance is currently reading. This is the log containing the current scraping position.

-s <SUBSCRIPTION_NAME>
Specifies the name of the subscription for which you want to display the database log that InfoSphere CDC is currently reading. Use this parameter in conjunction with the `-t` parameter to display the database log.

-A Specifies all subscriptions in the specified InfoSphere CDC instance.

-v Specifies verbose output (otherwise, the output is formatted for scripting).

-L <locale>
The name of the locale used for the InfoSphere CDC instance. The default is the locale of the machine where InfoSphere CDC is installed.

Result

This command returns a value of 0 if the command was successful and a non-zero value if the command fails.

Examples

```
dmshowlogdependency -I MYINSTANCE -i -s MYSUBSCRIPTIONNAME
```

Displays the complete list of required database logs for the specified instance and subscription.

```
dmshowlogdependency -I MYINSTANCE -A
```

Displays the complete list of required database logs for all subscriptions in the specified instance.

10.7.5 Exporting and importing configuration commands

This section contains commands that allow you to export and/or import your InfoSphere CDC global configuration.

dmexportconfiguration - Export InfoSphere CDC Configuration

Use this command to export the configuration details you had set when you had installed an instance of InfoSphere CDC. Configuration details are sent to an XML configuration file. You can use the `dmimportconfiguration` command to import the XML file that you create with this command into another instance of InfoSphere CDC.

Note: This command does not export subscription-specific settings that are configured in Management Console. Subscription-specific settings can be exported to an XML file in Management Console. For more information, see your Management Console documentation.

This command is interactive and will prompt you for your password. You cannot use this command in a script.

Syntax

```
dmexportconfiguration <path_to_configuration_file> [-L <locale>]
```

Parameters

<path_to_configuration_file>
The relative or absolute path to the XML configuration file that you want to export. The relative path is relative to your installation of InfoSphere CDC.

-L <locale>
The name of the locale used for the InfoSphere CDC instance. The default is your machine's locale.

Result

This command returns a value of 0 if the command was successful and a non-zero value if the command fails.

Examples

```
dmexportconfiguration c:\configurations\configuration.xml
```

InfoSphere CDC exports the XML file to specified relative path.

dmimportconfiguration - Import InfoSphere CDC Configuration

Use this command to import InfoSphere CDC configuration settings from an XML file you created with the `dmexportconfiguration` command.

Note: You can script this command and use an InfoSphere CDC silent installation to deploy InfoSphere CDC on multiple systems.

Syntax

```
dmimportconfiguration <path_to_configuration_file> [-L <locale>]
```

Parameters

<path_to_configuration_file>
The relative or absolute path to the XML configuration file that you are importing. The relative path is relative to your installation of InfoSphere CDC.

-L <locale>
The name of the locale used for the InfoSphere CDC instance. The default is the locale of the machine where InfoSphere CDC is installed.

Result

This command returns a value of 0 if the command was successful and a non-zero value if the command fails.

Examples

```
dmimportconfiguration c:\configurations\configuration.xml
```

InfoSphere CDC imports the XML configuration file from the specified absolute path.

10.7.6 Managing tables for replication commands

This section contains commands that help you manage the tables that you want to replicate with InfoSphere CDC.

dmdescribe - Describe Source Tables

Use this command to send source table metadata changes over to the target.

This command exits after it has successfully described the specified subscriptions.

Syntax

```
dmdescribe -I <instance_name> <-A|-s <subscription_names> ...> [-L <locale>]
```

Parameters

-I <instance_name>
Specifies the InfoSphere CDC instance for which you want to send source table mapping changes over to the target. Alternatively, you can specify the TSINSTANCE environment variable in place of this value.

-A Specifies that InfoSphere CDC will send source table mapping changes made to all subscriptions over to the target.

-s <subscription_names>
Specifies that InfoSphere CDC sends source metadata changes for the indicated subscriptions over to the target. To specify multiple subscriptions, list the subscriptions separated by a space.

-L <locale>
The name of the locale used for the InfoSphere CDC instance. The default is the locale of the machine where InfoSphere CDC is installed.

Result

This command returns a value of 0 if the command was successful and a non-zero value if the command fails.

Examples

```
dmdescribe -I NEWINSTANCE -s FINANCE
```

InfoSphere CDC sends source metadata changes in the Finance subscription over to the target for the specified instance.

dmflagforrefresh - Flag for Refresh

Use this command to flag a source table for refresh. When you flag a table for refresh, you are selecting the tables that you want to refresh at a future point in time. Use this procedure when you have selected **Refresh** as your replication method on a subscription.

Syntax

```
dmflagforrefresh -I <instance_name> -s <subscription_names>
<-A|-t <schema>.<table> ...> [-L <locale>]
```

Parameters

-I <instance_name>
Specifies the name of the InfoSphere CDC instance. Alternatively, you can specify the TSINSTANCE environment variable in place of this value.

-s <subscription_names>
Specifies the name of the subscription. List the subscriptions if you specify more than one.

-A Specifies that InfoSphere CDC flags all source tables for refresh in the subscription.

-t <schema>.<table>
Specifies the name of a source table in the subscription that InfoSphere CDC flags for refresh. You must specify the table name in the format *schema.table*. List the tables if you specify more than one.

-L <locale>
The name of the locale used for the InfoSphere CDC instance. The default is your machine's locale.

Result

This command returns a value of 0 if the command was successful and a non-zero value if the command fails.

Examples

```
dmflagforrefresh -I myinstance -s Finance -A
```

InfoSphere CDC flags for refresh all source tables in the **Finance** subscription for the specified instance.

dmmarktablecapturepoint - Mark a table capture point on a source table

Use this command to mark a table capture point on a source table and move the table to active state. If you changed the table before executing this command, those changes will not be replicated.

Mark a table capture point on a source table when you want to override an existing position in the stream of changed data. This is possible when you have already synchronized (refreshed) your source and target tables using an application other than Management Console (for example, using the import or export capabilities of your database platform) and you know the point in time your source and target are synchronized with each other. InfoSphere CDC mirrors changes to the target table from the current position in the stream of changed data. This position is set by InfoSphere CDC when you select **Mirror (Change Data Capture)** after mapping your tables in the Map Tables wizard. If you want to override the position set by InfoSphere CDC, then you can manually mark a table capture point in Management Console. When you decide to start mirroring on the subscription, InfoSphere CDC identifies the position you have set as the point in time from which to capture and replicate database changes to the target.

Syntax

```
dmmarktablecapturepoint -I <instance_name> -s <subscription_names>
<-A|-t <schema>.<table> ...> [-L <locale>]
```

Parameters

-I <instance_name>
Specifies the name of the InfoSphere CDC instance. Alternatively, you can specify the TSINSTANCE environment variable in place of this value.

-s <subscription_names>
Specifies the subscription name. List the subscriptions if you specify more than one.

-A Specifies that InfoSphere CDC overrides an existing position in the stream of changed data on all source tables in the subscription.

-t <schema>.<table>
Specifies the name of a source table in the subscription on which InfoSphere CDC marks a table capture point. You must specify the table name in the format *schema.table*. List the tables if you specify more than one.

-L <locale>
The name of the locale used for the InfoSphere CDC instance. The default is your machine's locale.

Result

This command returns a value of 0 if the command was successful and a non-zero value if the command fails.

Examples

```
dmmarktablecapturepoint -I myinstance -s Finance -A
```

InfoSphere CDC overrides an existing position in the stream of changed data on all source tables in the **Finance** subscription.

```
dmmarktablecapturepoint -I myinstance -s Finance -t myschema.mytable
```

InfoSphere CDC moves the specified table to active in the **Finance** subscription.

dmpark - Park Table

Use this command to park a source table. By parking a source table, you tell InfoSphere CDC that you do not want to capture changes for that particular table in a subscription. When you park a table, InfoSphere CDC does not replicate any subsequent changes you make on the source table, which may result in inconsistent source and target tables.

Note: Before you can park a source table, if you are mirroring the table to the target, then you need to end replication on the subscription. For more information, see "dmendreplication - End Replication" on page 119.

Syntax

```
dmpark -I <instance_name> -s <subscription_names> <-A|-t <schema>.<table> ...>
[-L <locale>]
```

Parameters

-I <instance_name>
Specifies the name of the InfoSphere CDC instance. Alternatively, you can specify the TSINSTANCE environment variable in place of this value.

-s <subscription_names>
Specifies the subscription name. List the subscriptions if you specify more than one.

-A Specifies that InfoSphere CDC parks all source tables in the subscription.

-t <schema>.<table>
Specifies the name of a source table in the subscription that InfoSphere CDC parks. You must specify the table name in the format *schema.table*. List the tables if you specify more than one.

-L <locale>
The name of the locale used for the InfoSphere CDC instance. The default is your machine's locale.

Result

This command returns a value of 0 if the command was successful and a non-zero value if the command fails.

Examples

```
dmpark -I myinstance -s Finance -A
```

InfoSphere CDC parks all source tables in the **Finance** subscription.

dmreaddtable - Update Source Table Definition

Use this command to update the definition of a source table in InfoSphere CDC metadata. Run this command after you have changed the definition of a source table using your RDBMS.

Syntax

```
dmreaddtable -I <instance_name> <-A|-t <schema>.<table> ...> [-a] [-L <locale>]
```

Parameters

-I <instance_name>
Specifies the name of the InfoSphere CDC instance. Alternatively, you can specify the TSINSTANCE environment variable in place of this value.

-A Specifies that InfoSphere CDC updates definitions for all source tables that are available for replication.

-t <schema>.<table>
Specifies the name of a source table in the subscription for which InfoSphere CDC updates the definition. You must specify the table name in the format *schema.table*. List the tables if you want to specify more than one.

-a Specifies that the active state of a table is kept after the table definition is updated.

-L <locale>
The name of the locale used for the InfoSphere CDC instance. The default is your machine's locale.

Result

This command returns a value of 0 if the command was successful and a non-zero value if the command fails.

Examples

```
dmreaddtable -I new_instance -A
```

InfoSphere CDC updates definitions for all source tables that are available for replication.

dmreassigntable - Update Target Table Definition

Use this command to update the definition of a target table in InfoSphere CDC metadata. Run this command after you have changed the definition of a target table using your RDBMS.

Syntax

```
dmreassigntable -I <instance_name> -s <subscription_names>
<-A|-t <schema>.<table> ...> [-L <locale>]
```

Parameters

-I <instance_name>
Specifies the name of the InfoSphere CDC instance. Alternatively, you can specify the TSINSTANCE environment variable in place of this value.

-s <subscription_names>
Specifies the InfoSphere CDC subscription that contains the table. List the subscriptions if you specify more than one.

-A Specifies that InfoSphere CDC updates definitions for all target tables in the subscription.

-t <schema>.<table>
Specifies the name of a target table in the subscription for which InfoSphere CDC updates the definition. You must specify the table name in the format *schema.table*. List the tables if you specify more than one.

-L <locale>
The name of the locale used for the InfoSphere CDC instance. The default is your machine's locale.

Result

This command returns a value of 0 if the command was successful and a non-zero value if the command fails.

Examples

```
dmreassigntable -I new_instance -s Finance -A
```

InfoSphere CDC updates definitions for all target tables in the **Finance** subscription.

dmsetreplicationmethod - Set Replication Method

Use this command to change the replication method for tables in a subscription. When running this command, InfoSphere CDC changes the status of any **Active** tables to **Refresh**.

Note: Before you run this command, you must end replication on the subscription.

Syntax

```
dmsetreplicationmethod -I <instance_name> <-r|-m> -s <subscription_names>
<-A|-t <schema>.<table> ...> [-L <locale>]
```

Parameters

-I <instance_name>
Specifies the name of the InfoSphere CDC instance. Alternatively, you can specify the TSINSTANCE environment variable in place of this value.

-m Specifies that tables will use **Mirror (Change Data Capture)** as the replication method.

-r Specifies that tables will use **Refresh (Snapshot)** as the replication method.

-s <subscription_names>
Specifies the name of the subscriptions.

-A Specifies that all tables in the subscription will use the indicated replication method.

-t <schema>.<table>
Specifies the name of a source table in the subscription that will use the indicated replication method. You must specify the table name in the format *schema.table*. List the tables if you specify more than one.

-L <locale>
The name of the locale used for the InfoSphere CDC instance. The default is your machine's locale.

Result

This command returns a value of 0 if the command was successful and a non-zero value if the command fails.

Examples

```
dmsetreplicationmethod -I myinstance -r -s Finance -A
```

All tables in the **Finance** subscription will use **Refresh** as the replication method in the specified InfoSphere CDC instance.

```
dmsetreplicationmethod -I new_instance -m -s Finance -t acct.taxcodes
```

The source table *acct.taxcodes* in the **Finance** subscription will use **Mirror** as the replication method in the specified InfoSphere CDC instance.

10.7.7 Monitoring replication commands

This section contains commands that help you monitor replication in InfoSphere CDC:

dmclearevents - Clear Events

Use this command to delete events from the **Event Log** view in Management Console.

Syntax

```
dmclearevents -I <instance_name> [-S|-T-|-B] <-A|-s <subscription_names> ...>
[-L <locale>]
```

Parameters

-I <instance_name>
Specifies the name of the InfoSphere CDC instance. Alternatively, you can specify the TSINSTANCE environment variable in place of this value.

-S Specifies that InfoSphere CDC clears events from the source.

-T Specifies that InfoSphere CDC clears events from both the source and target. If none of the `S`, `T`, and `B` options are specified, InfoSphere CDC assumes `B` by default.

-B Specifies the name of a source table in the subscription on which InfoSphere CDC sets a log position. You must specify the table name in the format *schema.table*. List the tables if you specify more than one.

-A Specifies that InfoSphere CDC clears events for all subscriptions.

-s <subscription_names>
Specifies that InfoSphere CDC clears events for the indicated subscription. List the subscriptions if you specify more than one.

-L <locale>
The name of the locale used for the InfoSphere CDC instance. The default is your machine's locale.

Result

This command returns a value of 0 if the command was successful and a non-zero value if the command fails.

Examples

```
dmclearevents -I myinstance -S -A
```

InfoSphere CDC clears events from the source for **all** subscriptions for the specified instance.

```
dmclearevents -I myinstance -T -s Finance Marketing
```

InfoSphere CDC clears events from both the source and target for the **Finance** and **Marketing** subscriptions for the specified instance.

dmgetsubscriptionstatus - Get Subscription Status

Use this command to retrieve information indicating the current state of subscriptions and to send the results to standard output.

Syntax

```
dmgetsubscriptionstatus -I <instance_name> [-p] <-A|-s <subscription_name> ...>
[-L <locale>]
```

Parameters

-I <instance_name>
Specifies the name of the InfoSphere CDC instance. Alternatively, you can specify the TSINSTANCE environment variable in place of this value.

-p Specifies that InfoSphere CDC sends state information to standard output.

-A Specifies that InfoSphere CDC retrieves state information for all subscriptions.

-s <subscription_name>
Specifies the name of the subscription for which state information is retrieved. List the subscriptions if you specify more than one.

-L <locale>
The name of the locale used for the InfoSphere CDC instance. The default is your machine's locale.

Result

This command returns one of the following:

- `0`—If the specified subscriptions have a state of **Inactive**.
- `1`—If any of the specified subscriptions have a state other than **Inactive**.
- A negative value—If an error has occurred while retrieving status information.

Examples

```
dmgetsubscriptionstatus -I myinstance -p -A
```

InfoSphere CDC retrieves state information for all subscriptions and sends the results to standard output for the specified instance.

dmshowevents - Display InfoSphere CDC events

Use this command to display InfoSphere CDC events to standard output. You can use this command as an alternative to showing InfoSphere CDC events in the Event Log view in Management Console.

The output of this command shows events in chronological order with the most recent event shown first in the list.

Syntax

```
dmshowevents -I <instance_name> <-a|-s <subscription> ...
|-t <source_ID> ...|-s <subscription> ... -t <source_ID> ...> [-h] [-c max_msg]
[-L <locale>]
```

or

```
dmshowevents -I <instance_name> <-a|-s <subscription>|-t
<source_ID>> ...> [-h] [-c max_msg] [-L <locale>]
```

Parameters

-I <instance_name>
Specifies the name of the InfoSphere CDC instance. Alternatively, you can specify the TSINSTANCE environment variable in place of this value.

-a Specifies that InfoSphere CDC shows events for all subscriptions.

-s <subscription>
Specifies the name of the source subscription for which InfoSphere CDC shows events. List the subscriptions if you specify more than one.

-t <source_ID>
Specifies the source ID for which InfoSphere CDC shows events. List the source IDs if you specify more than one.

-h Specifies that InfoSphere CDC displays a header before the list of events. This option helps you identify each item of information that is displayed for each event.

-c max_msg
Specifies the maximum number of events that InfoSphere CDC displays. If you omit this parameter or you specify a value greater than the total number of events, InfoSphere CDC displays all events for the specified subscriptions and/or source IDs.

- **Minimum Setting**—0. No events are shown.
- **Maximum Setting**—2147483647

-L <locale>
The name of the locale used for the InfoSphere CDC instance. The default is the locale of the machine where InfoSphere CDC is installed.

Result

This command returns a value of 0 if the command was successful and a non-zero value if the command fails.

Examples

```
dmshowevents -I new_instance -s Finance
```

InfoSphere CDC displays all events for the **Finance** subscription for the specified instance.

```
dmshowevents -I myinstance –a –h
```

InfoSphere CDC displays all events for all subscriptions. A header is displayed before the list of events for the specified instance.

```
dmshowevents -I newinstance –s Finance –t Atlanta –s Marketing –h –c 20
dmshowevents -I myinstance –s Finance Marketing –t Atlanta –h –c 20
```

InfoSphere CDC displays the most recent 20 events for the **Finance** and **Marketing** subscriptions and for the Atlanta source ID. A header is displayed before the list of events for the specified instance.

Sample output

```
EVENTTIME|EVENTSOURCE|ORIGINATOR|EVENTID|SEVERITY|EVENTPROGRAM|EVENTTEXT

2006-04-21 17:23:08.817|T|ATLANTA|95|Information|class com.datamirror.ts.target.
publication.c|Transformation Server Communications ending.

2006-04-21 17:23:08.614|T|ATLANTA|1538|Information|class com.datamirror.ts.target.
publication.c|---Transformation Server for ATLANTA terminating normally.

2006-04-21 17:23:08.333|T|ATLANTA|1537|Information|class com.datamirror.ts.target.
publication.c|Describe conversation with ATLANTA completed successfully.

2006-04-21 17:23:07.911|T|ATLANTA|1536|Information|class com.datamirror.ts.target.
publication.c|Describe conversation started by ATLANTA.

2006-04-21 17:23:07.333|T|ATLANTA|1531|Information|class com.datamirror.ts.target.
publication.c|Communication with ATLANTA successfully started on Data channel.

2006-04-21 17:23:06.973|T|ATLANTA|1534|Information|class com.datamirror.ts.engine.a
|Code page conversation from the source database's code page 1252 to the target
database's code page Cp1252 for ATLANTA will be performed by the Remote system
```

Fields in each record are separated by vertical bars (|). These fields are identified in the first line of the output. In the *EVENTSOURCE* field, *S* indicates source and *T* indicates target.

10.7.8 Other commands

This section contains miscellaneous commands that allow you to determine the version of InfoSphere CDC, verify communications, shutdown, and terminate InfoSphere CDC (UNIX servers only), set system parameters, and backup your metadata.

dmbackupmd - Backup Metadata

Use this command to create a backup of the InfoSphere CDC metadata database which contains information about your current replication configuration. You should always back up your metadata when there are changes to your subscription configuration and table status. You can only back up your metadata while InfoSphere CDC is running.

The backup of the metadata database is created in `<Installation_directory>/instance/<instance_name>/conf/backup` for UNIX and Linux and in `<Installation_directory>\instance\<instance_name>\conf\backup` for Windows. The files in the backup directory should be stored on separate media for possible recovery.

Syntax

```
dmbackupmd -I <instance_name> [-L <locale>]
```

Parameters

-I <instance_name>
Specifies the name of the InfoSphere CDC instance. Alternatively, you can specify the TSINSTANCE environment variable in place of this value.

-L <locale>
The name of the locale used for the InfoSphere CDC instance. The default is the locale of the machine where InfoSphere CDC is installed.

Result

This command returns a value of 0 if the command was successful and a non-zero value if the command fails.

dmconfigurets - Configure InfoSphere CDC

Use this command to launch the InfoSphere CDC configuration tool. You can use this tool to create instances and configure your installation of InfoSphere CDC.

Syntax

```
dmconfigurets [-L <locale>]
```

Parameters

-L <locale>
The name of the locale used for the InfoSphere CDC instance. The default is your machine's locale.

Result

This command returns a value of 0 if the command was successful and a non-zero value if the command fails.

dmmdcommander

This command is for internal use only.

dmmdconsole

This command is for internal use only.

dmclearstagingstore - Remove cached operations from the staging store

Use this command to remove all the contents from the InfoSphere CDC staging store on your source system. The staging store is used to provide a cache of change data that is read from the database logs. There may be times when the contents of the staging store are no longer valid and InfoSphere CDC will give instructions to clear the staging store with this command.

Syntax

```
dmclearstagingstore [-I <INSTANCE_NAME>] [-L <locale>]
```

Parameters

-I <instance_name>
Specifies the name of the InfoSphere CDC instance. Alternatively, you can specify the TSINSTANCE environment variable in place of this value.

-L <locale>
The name of the locale used for the InfoSphere CDC instance. The default is your machine's locale.

Result

This command returns a value of 0 if the operation was successful. If it fails, this command returns a non-zero value.

dmgetstagingstorestatus - Retrieve Staging Store status

Use this command to retrieve status information for the InfoSphere CDC staging store on your source system and the Continuous Capture feature.

Syntax

```
dmgetstagingstorestatus [-I <INSTANCE_NAME>] [-L <locale>]
```

Parameters

-I <instance_name>
Specifies the name of the InfoSphere CDC instance. Alternatively, you can specify the TSINSTANCE environment variable in place of this value.

-L <locale>
The name of the locale used for the InfoSphere CDC instance. The default is your machine's locale.

Related reference:

"dmdisablecontinuouscapture - Disable Continuous Capture" on page 143

dmenablecontinuouscapture - Enable Continuous Capture

Use this command to enable Continuous Capture for your staging store.

Continuous Capture allows the InfoSphere CDC log reader to continue operating when communication with the target datastore is interrupted due to network difficulties or other issues. Upon resumption of communication with the target, Continuous Capture will reduce the latency between the source and target datastores.

Syntax

```
dmenablecontinuouscapture [-I <INSTANCE_NAME>] [-L <locale>]
```

Parameters

-I <instance_name>
Specifies the name of the InfoSphere CDC instance. Alternatively, you can specify the TSINSTANCE environment variable in place of this value.

-L <locale>
The name of the locale used for the InfoSphere CDC instance. The default is your machine's locale.

Related reference:

"dmdisablecontinuouscapture - Disable Continuous Capture"

dmdisablecontinuouscapture - Disable Continuous Capture

Use this command to disable Continuous Capture for your staging store.

Syntax

```
dmdisablecontinuouscapture [-I <INSTANCE_NAME>] [-L <locale>]
```

Parameters

-I <instance_name>
Specifies the name of the InfoSphere CDC instance. Alternatively, you can specify the TSINSTANCE environment variable in place of this value.

-L <locale>
The name of the locale used for the InfoSphere CDC instance. The default is your machine's locale.

Related reference:

"dmenablecontinuouscapture - Enable Continuous Capture" on page 142

dmset - Set InfoSphere CDC System Parameter

Use this command to view or change InfoSphere CDC system parameters. You can also change system parameters in Management Console. For more information, see your Management Console documentation.

Note: You can set any system parameter using this command. However, it will only display system parameters that are set to non-default values.

Syntax

```
dmset -I <instance_name> [<parameter_name>[=[<parameter_value>]]] [-L <locale>]
```

Parameters

-I <instance_name>
Specifies the name of the InfoSphere CDC instance. Alternatively, you can specify the TSINSTANCE environment variable in place of this value.

<parameter_name>
Specifies the name of the InfoSphere CDC system parameter.

<parameter_value>
Specifies the value that you want to assign to the system parameter.

-L <locale>
The name of the locale used for the InfoSphere CDC instance. The default is your machine's locale.

Result

This command returns a value of 0 if the command was successful and a non-zero value if the command fails.

Examples

```
dmset -I myinstance
```

Displays all of the system parameters that are set to non-default values.

```
dmset -I myinstance global_unicode_as_char=false
```

Sets the `global_unicode_as_char` system parameter to false.

```
dmset -I myinstance global_unicode_as_char
```

Displays the current value of the specified parameter.

```
dmset -I myinstance stop_replication=
```

Deletes the `stop_replication` system parameter.

dmsetaccessserverparams - Set Access Server parameters

Use this command to define the access and login data to the Access Server. This command is needed when using the Refresh stored procedure.

Syntax

```
dmsetaccessserverparams [-u <username>] [-p <password>] [-H <hostname>] [-P <port>]
```

Parameters

-u <username>
Specifies the Access Server user.

-p <password>
Specifies the password for the Access Server user.

-H <hostname>
Specifies the host name (system name) or full IP address of the workstation running Access Server.

-P <port>
Specifies a unique TCP/IP port number that is used to connect to Access Server. You specify this port number when you install the Access Server and when you log on to the Management Console. The default value is 10101.

-L <locale>
The name of the locale used for the InfoSphere CDC instance. The default is the locale of the machine where InfoSphere CDC is installed.

When using `dmsetaccessserverparams` for the first time, specify all the parameters. If you do not specify the parameter values, the following default values are set:

- **User** - Admin
- **Password** - "" (blank)
- **Host** - localhost
- **Port** - 10101

After you have set the values, you can modify them by issuing the command again, including all or some of the parameters.

Result

This command returns a value of 0 if the command was successful and a non-zero value if the command fails.

Examples

```
dmsetaccessserverparams -u dba -p dba -H localhost -P 10101
```

Sets the access and login data for the user 'dba' with password 'dba'.

```
dmsetaccessserverparams - H newmachine
```

Changes the host name of the workstation running Access Server.

dmshowversion - Show InfoSphere CDC Version

Use this command to display the InfoSphere CDC version and build number. Run this command before you contact your IBM representative, so that you can provide the version and build number of InfoSphere CDC that you are running.

Syntax

```
dmshowversion [-L <locale>]
```

Parameters

-L <locale>
The name of the locale used for the InfoSphere CDC instance. The default is your machine's locale.

Result

This command returns a value of 0 if the command was successful and a non-zero value if the command fails.

dmshutdown - Shut Down InfoSphere CDC

Use this command to stop an instance of InfoSphere CDC and end replication on all subscriptions that use the instance as a source. This command is often used prior to taking a server or database offline for maintenance purposes or upgrading InfoSphere CDC.

Note: As a best practice before you run this command and to ensure that it completes successfully, use the dmendreplication command to end replication on all subscriptions that use the specified instance as a source and as a target. This command will not complete successfully if target subscriptions are still running.

To end replication on subscriptions that use the specified instance as a target, you can use the –a parameter which will generate an error when forcefully ending replication on subscriptions that use the specified instance as the target.

If this command does not end InfoSphere CDC processes and stop the specified instance, use the **dmterminate** command on the UNIX and Linux platforms to force a complete shut down. On Windows platforms, use the **dmterminate** command to stop the service.

Syntax

```
dmshutdown [-I <INSTANCE_NAME>] [-c|-i|-a|-se [-t <date and time>|-p <log position>]
[-L <locale>]
```

Parameters

-I <instance_name>
Specifies the name of the InfoSphere CDC instance. Alternatively, you can specify the TSINSTANCE environment variable in place of this value.

-c Specifies that InfoSphere CDC stops the specified instance and ends replication on all subscriptions that use the instance as a source with the Normal option. InfoSphere CDC will use this option by default if you do not specify –se, -i, or –a.

This option completes in progress work and then ends replication. If a refresh is in progress, Normal will complete the refresh for the current table before replication ends.

Normal is the most appropriate option for most business requirements and is the preferred method for ending replication in most situations.

-i Specifies that InfoSphere CDC stops the specified instance and ends replication on all subscriptions that use the instance as a source with the **Immediate** option.

This option stops all in progress work and then ends replication. Starting replication after using this option can be slower than using -c. If a refresh is in progress, the refresh for the current table will be interrupted and then replication will end.

Note: Use this option if business reasons require replication to end faster than -c at the expense of a slower start when you resume replication on the specified subscriptions.

-a Specifies that InfoSphere CDC stops the specified instance and ends replication on all subscriptions that use the instance as a source or target with the Abort option. Subscriptions that use the specified instance as a target will end replication with an error.

This option stops all in progress work and then ends replication rapidly. Starting replication after using this option can be much slower than using -c. A refresh in progress will be interrupted and the target will stop processing any data that has not been committed before replication ends.

Note: Use this option if your business reasons require a rapid end to replication and you are willing to tolerate a much slower start when you resume replication on the specified subscriptions.

A sudden business requirement for an unplanned shutdown of your source system may require this option for ending replication.

Note: As a best practice, use the **dmendreplication** command to end replication on all subscriptions that use the instance specified in this command as a source or target.

-se
Specifies that InfoSphere CDC will stop the specified instance and end replication normally at the current source system time in the source database log with the Scheduled End option. Replication will end on subscriptions that use the specified instance as a source. The source system time when replication will end is set when you issue this command.

Note: As latency between the source and target increases, the amount of time required to end replication will also increase.

-t <date and time>
Indicates the date and time in the source database log when replication will end when using –se.

When specifying a value for this parameter, use the following format:

```
"yyyy-MM-dd HH:mm"
```

This parameter is optional when you specify –se.

-L <locale>
The name of the locale used for the InfoSphere CDC instance. The default is the locale of the machine where InfoSphere CDC is installed.

Result

This command returns a value of 0 if the command was successful and a non-zero value if the command fails.

Related reference:

“dmterminate - Terminate InfoSphere CDC Processes” on page 148

dmsupportinfo - Collect support information

Note: You should only run this command when the Management Console Support Assistant cannot connect to yourInfoSphere CDC datastore because it is not running or it will not run. For more information on the Support Assistant, see *Management Console - Administration Guide*.

When requested by IBM Support, use this command to collect InfoSphere CDC environment information in a generated `.zip` file that is used to diagnose and troubleshoot your support issue.

Once the command has completed collecting information and generating the `.zip` file, the output will display the full path and name of the `.zip` file. If you run this command multiple times, the generated `.zip` files are numbered randomly. Note that you are responsible for deleting the generated `.zip` files when they are no longer required.

Syntax

```
dmsupportinfo [-I <INSTANCE_NAME>] [-t <"yyyy-MM-dd hh:mm:ss to yyyy-MM-dd
 hh:mm:ss">] [-L <locale>]
```

Parameters

-I <INSTANCE_NAME>
Specifies the name of the InfoSphere CDC instance. Alternatively, you can specify the TSINSTANCE environment variable in place of this value.

-t <"yyyy-MM-dd hh:mm:ss to yyyy-MM-dd hh:mm:ss">
Specifies the date and time range (relative to the time zone of the operating system where you issue this command) used by InfoSphere CDC to retrieve environment information.

Note: As a best practice, specify a date and time range that only captures the time period when you experienced problems. This allows for easier problem diagnosis and reduces the size of the files retrieved.

-L <locale>
The name of the locale used for the InfoSphere CDC instance. The default is the locale of the machine where InfoSphere CDC is installed.

Result

This command returns a value of 0 if the command was successful and a non-zero value if the command fails.

Examples

```
dmsupportinfo -I PRODUCTION -t "2009-12-03 08:00:00 to 2009-12-03 12:00:00"
```

Retrieves support information for the Production instance from 8:00 AM to 12:00 PM on December 3, 2009. This is the time range when you experienced support issues with this instance of InfoSphere CDC.

dmterminate - Terminate InfoSphere CDC Processes

Note: This command is not supported on Windows.

Use this command to terminate all InfoSphere CDC processes, for all instances running on a UNIX or Linux server that you cannot completely shut down with the `dmshutdown` command. InfoSphere CDC terminates only processes that are started by the UNIX account used to run this command.

You can use this command prior to taking a server or database offline for maintenance purposes or upgrading InfoSphere CDC to the latest version.

Use the `dmshutdown` command to gracefully shut down InfoSphere CDC. If `dmshutdown` is unable to completely shut down InfoSphere CDC, then use `dmterminate` to terminate any active InfoSphere CDC processes that still remain after issuing `dmshutdown`.

Syntax

```
dmterminate [-L <locale>]
```

Parameters

-L <locale>
The name of the locale used for the InfoSphere CDC instance. The default is your machine's locale.

Result

This command returns a value of 0 if the command was successful and a non-zero value if the command fails.

dmts32 - Start InfoSphere CDC

Use this command to start a 32-bit instance of InfoSphere CDC.

Syntax

```
dmts32 -I <instance_name> [-L <locale>]
```

Parameters

-I <instance_name>
Specifies the InfoSphere CDC instance for which you want to start.

-L <locale>
The name of the locale used for the InfoSphere CDC instance. The default is the locale of the machine where InfoSphere CDC is installed.

Result

This command returns a value of 0 if the command was successful and a non-zero value if the command fails.

Examples

```
dmts32 -I -I myinstance
```

InfoSphere CDC starts for the specified instance.

dmts64 - Start InfoSphere CDC

Use this command to start a 64-bit instance of InfoSphere CDC.

Syntax

```
dmts64 -I <instance_name> [-L <locale>]
```

Parameters

-I <instance_name>
Specifies the InfoSphere CDC instance for which you want to start.

-L <locale>
The name of the locale used for the InfoSphere CDC instance. The default is the locale of the machine where InfoSphere CDC is installed.

Result

This command returns a value of 0 if the command was successful and a non-zero value if the command fails.

Examples

```
dmts64 -I myinstance
```

InfoSphere CDC starts for the specified instance.

10.8 User exits for InfoSphere CDC

A user exit lets you define a set of actions that InfoSphere CDC can run before or after a database event occurs on a specified table. User exits allow you to customize your environment to meet your business requirements.

After compiling your Java Class or Stored Procedure user exit, you can configure the user exit in Management Console. For more information about configuring user exits, see "Configuring user exits" in your Management Console documentation.

The Javadoc (API) information that is installed with InfoSphere CDC provides detailed class and interface specifications for the Java Class user exits that are available in InfoSphere CDC. For each interface, the supported methods that can be called are identified.

The Javadoc (API) documentation for the user exits is located in the following directory: <*system drive*>:\<installation directory>\docs\api. To open the help in your browser, click index.html.

Sample user exits are provided with InfoSphere CDC. You can extend or modify these samples to suit your environment.

10.8.1 Stored procedure user exits for table and row level operations

A stored procedure is a program (or procedure) which is physically stored within a database. The advantage of a stored procedure is that when it is run, in response to a user request, it is run directly by the database engine, which usually runs on a separate database server and is generally faster at processing database requests.

After writing and compiling user exit programs, you can specify at which user exit point you want to invoke the user exit (either before or after a row-level or before or after a table-level operation) on the **User Exits** tab of Management Console.

10.8.2 Defining a stored procedure user exit

When defining a stored procedure in InfoSphere CDC, consider the following:

- Overloaded stored procedures are not supported.
- Stored procedures must have at least two parameters, which must be the first two defined, in the following order:
  - result—an integer output parameter, used to return any error codes to the event log.
  - returnMsg—a character output parameter, used to return error messages to be logged.

10.8.3 Stored procedure user exit database connections

The stored procedure user exit program and InfoSphere CDC use the same shared connection as the default method to connect to the database. This setting ensures that, by default, changes to tables made by InfoSphere CDC are visible to the stored procedure user exit program.

10.8.4 Retrieving data with a stored procedure user exit

You can retrieve data from your source table by passing system parameters to your stored procedure. You can retrieve the following type of data:

- **Retrieve system values (s$)**—when passed to a stored procedure, the s$ prefix makes system values available from the source database to your stored procedure. For example, s$entry identifies the entry point at which InfoSphere CDC had run the user exit.
- **Retrieve journal control fields (j$)**—when passed to a stored procedure, the j$ prefix makes journal control fields available from the source database to your stored procedure. For example, j$USER identifies the user ID of the person that made the update on the source table. This is useful if you are using the stored procedure to audit table or row-level operations that have occurred on the source table.
- **Retrieve data values**—depending on the prefix you pass to the stored procedure, you can retrieve data from the source database and make it available to your stored procedure. For example, you can use b$ to retrieve the before image of the source column.

Each of these values can be used as input parameters for the stored procedure user exit that you write. The format used to retrieve data is slightly different depending on the product that you are using:

- For InfoSphere CDC, the format is *<x>$<value>*

where *<x>* represents the prefix and *<value>* represents the name of the value to be retrieved.

Retrieving system values using the s$ prefix

This prefix is used to retrieve system values. The table below presents and briefly describes these values.

Prefix and Value	Data Type	Description
s$entry	NUMBER	Represents the entry point from where the stored procedure was invoked. You can invoke a stored procedure from the following entry points: • **1**—indicates that InfoSphere CDC has invoked the stored procedure before a table clear (truncate) operation • **2**—indicates that InfoSphere CDC has invoked the stored procedure after a table clear (truncate) operation • **3**—indicates that InfoSphere CDC has invoked the stored procedure before a row insert operation • **4**—indicates that InfoSphere CDC has invoked the stored procedure after a row insert operation • **5**—indicates that InfoSphere CDC has invoked the stored procedure before a row update operation • **6**—indicates that InfoSphere CDC has invoked the stored procedure after a row update operation • **7**—indicates that InfoSphere CDC has invoked the stored procedure before a row delete operation • **8**—indicates that InfoSphere CDC has invoked the stored procedure after a row delete operation • **9**—indicates that InfoSphere CDC has invoked the stored procedure before a table refresh operation • **10**—indicates that InfoSphere CDC has invoked the stored procedure after a table refresh operation

Prefix and Value	Data Type	Description
s$srcSysId	VARCHAR	Identifies uniquely the location of the source data.
s$srcTabId	VARCHAR	Represents the name of the source table in the source database that sends replicated data to the target.
s$tgtTabId	VARCHAR	Represents the name of the target table in the target database that receives replicated data from the source.

Retrieving journal control fields using the j$ prefix

This prefix is used to retrieve information about the operation that occurred on the source system. You can use **jb$** with InfoSphere CDC to retrieve the same information.

The available values are listed:

Prefix and Value	Data Type	Description
j$CCID	VARCHAR	Identifies the transaction with the insert, update, or delete operation.
j$CODE	VARCHAR	Identifies the type of journal or log entry, either “U” for a refresh operation or “R” for mirroring.
j$CTRR or j$CNTRRN	VARCHAR	Identifies the relative record number of the source table that recorded the journal/log entry. **Note:** CTRR or CNTRRN contains meaningful information when you invoke your stored procedure on the insert entries that make up the refresh.
j$ENTT or j$ENTTYP	VARCHAR	Generates journal or log codes that identify the operation type on the source system.
j$JRN or j$JOURNAL	VARCHAR	The name of the journal/log where InfoSphere CDC is reading insert, update, or delete operations from.
j$JOB	VARCHAR	Identifies the name of the job that made the insert, update, or delete on the source system.
j$MBR or j$MEMBER	VARCHAR	Identifies the name of the source table or its alias.

Prefix and Value	Data Type	Description
j$NBR or j$JOBNO	VARCHAR	Identifies the process ID of the program on the source table that is making the insert, update, or delete operation.
j$PGM or j$PROGRAM	VARCHAR	Identifies the name of the program on the source system that made the insert, update or delete operation.
j$SEQN or j$SEQNO	VARCHAR	Identifies the sequence number of the insert, update, or delete operation in the journal or log.
j$SYNM or j$SYSTEM	VARCHAR	Identifies the host name of the source system.
j$USER	VARCHAR	Identifies the database user name that made the insert, update, or delete operation on the source.
j$USPF	VARCHAR	Identifies the operating system user name that made the insert, update, or delete operation on the source.
j$TSTP or j$TIMSTAMP	VARCHAR	Identifies the date and time of when the insert, update, or delete operation or refresh was made on the source. In environments that support microsecond precision, the date and time format of this journal control field is YYYY-MM-DD-HH:MM:SS.UUUUUU. Otherwise, InfoSphere CDC sets the microsecond component UUUUUU to zeroes or does not include it at all.

Retrieving data values using b$, a$, k$, and d$ prefixes

Four prefixes are used to retrieve data:

Prefix	Mode	Description

b$<source column name>	Input	Used to retrieve the before image of the data in a source column. The before image is the original image from the source table column before any transformation is applied to it.
		
		For example, you may have made the following UPDATE to your source table:
		`UPDATE source_table`
		`set MYCOLUMN = 2`
		`where MYCOLUMN = 1;`
		
		This will set 2 on all rows where MYCOLUMN was 1 before the execution of this SQL statement.
		
		When you define a stored procedure and decide that you want the stored procedure to retrieve the before image of MYCOLUMN, you would specify the following:
		`b$MYCOLUMN;`
		
		This returns a value of 1.

Prefix	Mode	Description

a$<source column name>	Input	Used to retrieve the after image of the data in a source column. The after image is the translated data from the source table column. For example, the data that was translated by a derived expression.
		
		For example, you may have made the following UPDATE to your source table:
		`UPDATE source_table`
		`set MYCOLUMN = 2`
		`where MYCOLUMN = 1;`
		
		This will set 2 on all rows where MYCOLUMN was 1 before the execution of this SQL statement.
		
		When you define a stored procedure and decide that you want the stored procedure to retrieve the after image of MYCOLUMN, you would specify the following:
		`a$MYCOLUMN;`
		
		This returns a value of 2.

k$<target key column name>	Input	Used to access the target table to find the rows that need to be modified. **Note:** Key columns are not available for auditing.
d$<target column name>	Input/Output	Used to retrieve data values after transformation, which will be used to update the table in the target database. Only these values may be modified by the stored procedure.

10.8.5 Example of a stored procedure user exit

The following code snippet is an example of a stored procedure user exit.

Code	Comments
create or replace procedure PROD.AUDIT_STPROC (result OUT INT, returnMsg OUT CHAR, s$entry IN NUMBER, s$srcSysId IN CHAR, s$srcTabId IN CHAR, s$tgtTabId IN CHAR, j$ENTT IN CHAR, a$IDNO IN NUMBER, a$PRICE IN NUMBER, a$DESC IN CHAR, a$LONGDESC IN CHAR, a$TRANSDATE IN DATE, d$IDNO IN NUMBER, d$PRICE IN NUMBER, d$DESC IN CHAR, d$LONGDESC IN CHAR, d$TRANSDATE IN DATE)	The parameters you declare and want to pass to your stored procedure must be valid data types. The following parameters are mandatory and must be declared in your stored procedure: **result**—Returns a value of '0' which indicates the stored procedure was successful or an error which may be an integer. **returnMsg**—Returns an error message in the Event Log. The following parameters have been declared in this stored procedure: • **s$entry**—retrieves the entry point at which the stored procedure was called. In this example, InfoSphere CDC calls the user exit at every entry point. • **s$srcSysId**—retrieves the location of source data • **s$srcSysId**—retrieves the location of source data • **s$srcTabId**—retrieves the name of the source table • **s$srcTabId**—retrieves the name of the source table • **s$tgtTabId**—retrieves the name of the target table • **s$tgtTabId**—retrieves the name of the target table • **j$ENTT**—retrieves the journal code that indicates the type of operation that took place on the source table • **j$ENTT**—retrieves the journal code that indicates the type of operation that took place on the source table • **a$**—retrieves the after image of the IDNO, PRICE, DESC, LONGDESC, and TRANSDATE source columns • **a$**—retrieves the after image of the IDNO, PRICE, DESC, LONGDESC, and TRANSDATE source columns • **d$**—retrieves the transformed data of the IDNO, PRICE, DESC, LONGDESC, and TRANSDATE target columns • **d$**—retrieves the transformed data of the IDNO, PRICE, DESC, LONGDESC, and TRANSDATE target columns
IS ENTRYPOINT VARCHAR(50); BEGIN CASE s$entry WHEN 16 THEN ENTRYPOINT := 'User Exit program called Before Insert'; WHEN 1048576 THEN ENTRYPOINT := 'User Exit program called After Insert'; WHEN 64 THEN ENTRYPOINT := 'User Exit program called Before Update'; WHEN 4194304 THEN ENTRYPOINT := 'User Exit program called After Update'; END CASE;	This stored procedure user exit can be invoked from these entry points.

Code	Comments
`insert into PROD.AUDIT_TABLE1 values ( s$entry, s$srcSysId, s$srcTabId, s$tgtTabId, j$ENTT, a$IDNO, a$PRICE, a$DESC, a$LONGDESC, a$TRANSDATE, d$IDNO, d$PRICE, d$DESC, d$LONGDESC, d$TRANSDATE, ENTRYPOINT);`	This stored procedure user exit will INSERT these values into *PROD.AUDIT_TABLE1*.
`result := 1; returnMsg := 'OK'; END AUDIT_STPROC;`	This stored procedure user exit was successful. **Note:** If your stored procedure returns a '0', then a message is generated to the event log.

10.8.6 Sample user exits for InfoSphere CDC

InfoSphere CDC provides sample user exits that you can extend or modify to suit your environment. The samples are found in `samples.jar` which is located in the samples directory in your InfoSphere CDC installation directory. The Java file contains the following samples:

- **ArchiveLogPathUserExitSample.java** — returns the fully qualified path (including file name and extension) to the archive log file. This sample is located in com.datamirror.ts.target.publication.userexit.sample.
- **CRUserExitSample.java**—a conflict resolution user exit that can be used with tables having a primary key column of any data type or a numeric column with any data type. This sample is located in `com.datamirror.ts.target.publication.userexit.cdr`.
- **DEUserExitSample.java** — used in expressions using the %USERFUNC column function. It calculates the sum of the user-supplied parameters (in the expression) and returns the sum incremented by 1. This sample is located in `com.datamirror.ts.derivedexpressionmanager`.
- **SPUserExitSample.java** — calls a stored procedure with the image coming from the source. This sample is located in `com.datamirror.ts.target.publication.userexit.sample`.
- **UserExitSample.java** — subscribes to replication events to retrieve the details of the events which took place. This sample is located in `com.datamirror.ts.target.publication.userexit.sample`.
- **UserExitSample1.java** — records new rows inserted into a table on the target and stores them in a text file. The user specifies the name of the text file as a parameter. This sample is located in `com.datamirror.ts.target.publication.userexit.sample`.

Note the following:

- To run the sample user exits without modifying them, you must specify the fully qualified path to the compiled user exit in Management Console. For example, `com.datamirror.ts.target.publication.userexit.sample.UserExitSample`.
- Compiled sample user exits are located in the `ts.jar` file which is found in the `lib` directory in your InfoSphere CDC installation directory. Note that the compiled user exits in the `ts.jar` file have a `*.class` extension.
- If you want to modify the sample user exits, you must compile the user exit after you make changes to the source code.
- The user exit class must also be in your classpath.

For more information about how to specify Java class or Stored Procedure user exits in Management Console, see your Management Console documentation.

To compile the sample user exits (Windows)

Procedure

1. Stop InfoSphere CDC.
2. Unzip the `samples.jar` file into the `lib` folder in your InfoSphere CDC installation folder. Make sure you maintain the folder structure when unzipping the jar file.

   After unzipping the jar file, you will have a folder structure like the following:

   ```
   <InfoSphere CDC installation folder>\lib\com\datamirror\ts\target
   \publication\userexit\sample
   ```
3. Make your changes to the sample user exit.
4. Compile the modified user exit. For example, if you want to compile `UserExitSample.java`, open a command window, navigate to the `lib` folder and issue the following command:

   ```
   javac  -classpath ts.jar;. com\datamirror\ts\target\publication\userexit\sample
   \UserExitSample.java
   ```

   If this command runs successfully, there will be no output on your screen.

   Note: Your system must have the Java JDK to run this command.
5. After running the command successfully, navigate to the following directory and confirm that you have created a `UserExitSample.class` file:

   ```
   <InfoSphere CDC installation directory>\lib\com\datamirror\ts\target
   \publication\userexit\sample
   ```
6. Start InfoSphere CDC.
7. The final step to configure the user exit is to specify the fully qualified path to *UserExitSample* in Management Console. For example:

   ```
   com.datamirror.ts.target.publication.userexit.sample.UserExitSample
   ```

   Note: Do not specify the `.class` extension.

What to do next

For more information about how to specify Java class user exits in Management Console, see your Management Console documentation.

Note: If you plan to use the sample user exits in production environments, you will have to test the samples before they are deployed. IBM does not assume responsibility for adverse results caused by modified or customized user exit classes.

To compile the sample user exits (UNIX and Linux)

Procedure

1. Stop InfoSphere CDC.
2. Unzip the `samples.jar` file into the `lib` directory in your InfoSphere CDC installation directory. Make sure you maintain the directory structure when unzipping the jar file.

   After unzipping the jar file, you will have a directory structure like the following:

   ```
   <InfoSphere CDC installation directory>/lib/com/datamirror/ts/target
   /publication/userexit/sample
   ```
3. Make your changes to the sample user exit.

4. Compile the modified user exit. For example, if you want to compile `UserExitSample.java`, open a command window, navigate to the `lib` directory and issue the following command:

   ```
   javac  -classpath ts.jar:. com/datamirror/ts/target/publication/userexit/sample
   /UserExitSample.java
   ```

   If this command runs successfully, there will be no output on your screen.

   Note: Your system must have the Java JDK to run this command.
5. After running the command successfully, navigate to the following directory and confirm that you have created a `UserExitSample.class` file:

   ```
   <InfoSphere CDC installation directory>/lib/com/datamirror/ts/target
   /publication/userexit/sample
   ```
6. Start InfoSphere CDC.
7. The final step to configure the user exit is to specify the fully qualified path to `UserExitSample` in Management Console. For example:

   ```
   com.datamirror.ts.target.publication.userexit.sample.UserExitSample
   ```

   Note: Do not specify the `.class` extension.

What to do next

For more information about how to specify Java class user exits in Management Console, see your Management Console documentation.

Note: If you plan to use the sample user exits in production environments, you will have to test the samples before they are deployed. IBM does not assume responsibility for adverse results caused by modified or customized user exit classes.

10.8.7 InfoSphere CDC API reference – Javadocs

The API reference is available in Javadoc format in your InfoSphere CDC installation directory.

To view the API reference, navigate to the api directory below and click the `index.html` file to open the Javadoc documentation in your browser:

- Windows—`<InfoSphere CDC installation directory>\docs\api`
- UNIX and Linux—`<InfoSphere CDC installation directory>/docs/api`

10.9 Conflict resolution audit table

When InfoSphere CDC resolves a conflict between the source and target tables, it records information about the resolution in the TS_CONFAUD table. InfoSphere CDC creates this table in the target metadata location that is specified during configuration of InfoSphere CDC.

In this section, you will learn:

10.9.1 Structure of the conflict resolution audit table

You can use the TS_CONFAUD table to track how conflict resolution affected your target table. For example, you can query the AFTERIMG column to see when a change was made to the target table. Then you can review the contents of the

BEFOREIMG and AFTERIMG columns to see the change on the source table that resulted in the data on the target table. This can help in identifying problems in your conflict resolution strategy.

Conflict detection and resolution is configured in Management Console. For more information, see your Management Console documentation.

The structure of the TS_CONFAUD table is as follows.

Column	Description
CNFTIME	The date and time on the target when the conflict was detected.
SRCTIME	The time the conflicting data was applied to the source table.
SRCSYSID	The source ID of the subscription.
SRCSCHEMA	The schema or library name for the source table.
SRCNAME	The name of the source table.
SRCMEMBER	This field is blank.
TGTSCHEMA	The schema or library for the target table.
TGTNAME	The name of the target table.
OPTYPE	The row-level operation on the source that caused the conflict. The value is one of: • **1**—a row was inserted into the source table. • **2**—a row was updated on the source table. • **3**—a row was deleted from the source table.
CNFTYPE	The type of conflict that was detected. The value is one of: • **1**—a row was inserted into the source table. The key for that row already exists in the target table. • **2**—a row was updated or deleted on the source table. The key for that row does not exist in the target table. • **3**—a row was updated or deleted on the source table. The images of the source and target tables do not match. • **4**—an unexpected conflict was detected.
RESMTD	The conflict resolution method that was used. The value is one of: • **1**—source wins • **2**—target wins • **3**—largest value wins • **4**—smallest value wins • **5**—user exit If the resolution method was **None**, then a row will not be entered into this table. See your InfoSphere CDC documentation for more information about these methods.
CNFRES	Indicates if the conflict was resolved. The value is one of: • **Y**—the conflict was resolved. • **N**—the conflict was not resolved.

Column	Description
BEFOREIMG	A representation of the row in the source table before it was changed. See 10.9.2, “Row image format” for more information about the format of this column.
BEFORETRNC	Indicates if the before image stored in BEFOREIMG was truncated. The value is one of: • **Y**—the value was truncated. • **N**—the value was not truncated.
AFTERIMG	A representation of the row in the source table after it was changed. See 10.9.2, “Row image format” for more information about the format of this column.
AFTERTRNC	Indicates if the after image stored in AFTERIMG was truncated. The value is one of: • **Y**—the value was truncated. • **N**—the value was not truncated.
TGTIMG	A representation of the row in the target table before replication occurred. See 10.9.2, “Row image format” for more information about the format of this column.
TGTTRNC	Indicates if the image stored in TGTIMG was truncated. The value is one of: • **Y**—the value was truncated. • **N**—the value was not truncated.
WINIMG	A representation of the final row in the target table after conflict resolution has occurred. See 10.9.2, “Row image format” for more information about the format of this column.
WINTRNC	Indicates if the image stored in WINIMG was truncated. The value is one of: • **Y**—the value was truncated. • **N**—the value was not truncated.

10.9.2 Row image format

The BEFOREIMG, AFTERIMG, TGTIMG, and WINIMG columns in the audit table show representations of a row in either the source or target table.

The images in these columns are limited by the maximum length of VARCHAR data on your target metadata database. The images contain all of the values in the row, except for data in raw, binary, or LOB columns. The data from each column is presented in the following format:

```
(length:value)
```

In the format above, *value* is the data in the column and *length* is the number of characters used to represent the data. The images display numeric data as character strings and NULL values as (null).

The row images match the column order in the source table and the conflict resolution audit table. These images may be truncated if the image is longer than the maximum length of VARCHAR data in the target metadata database. If a table's key column is not the first column in the table, then it may be truncated.

10.9.3 Truncated images

If a row image is longer than the maximum length of a VARCHAR column, then they will be truncated. There is a column in the audit table that indicates if each image column has been truncated. For example, if WINTRNC is Y, then the value of WINIMG was truncated. The format of the truncated column is:

```
(-length:value)
```

In the format above, *value* is the truncated value and *length* is the number of characters in the truncated string.

10.9.4 Unaudited data types

The audit table does not include columns of the following data types in its images:

- IMAGE
- NTEXT
- TEXT

If the source or target table contains rows with these data types, then the image simply overlooks them. Binary data will appear in the images as hex-encoded characters. The image does not store any information from unsupported columns.

10.10 Configuring user exits

A user exit lets you define a set of actions that InfoSphere CDC can run before or after a database event occurs on a specified table. When using InfoSphere CDC, a database event is defined as either a row-level operation or as a table-level operation. Row-level operations include an insert, update, or a delete. Table-level operations include a refresh or a truncate operation. For example, you can configure a row-level user exit program that sends an alert after InfoSphere CDC replicates a delete operation on a particular target table.

User exits can be grouped as either a Before User Exit or an After User Exit:

- **Before User Exit**—runs before InfoSphere CDC replicates any row-level or table-level operations to the target table.
- **After User Exit**—runs after InfoSphere CDC replicates any row-level or table-level operations to the target table.

The following list identifies common scenarios for developing a user exit program before or after row or table-level operations:

- Customize when InfoSphere CDC replicates a row-level operation to the target table. For example, you can develop logic for insert, update, or delete operations so that these occur based on some specified criteria, such as the original invoice date. InfoSphere CDC can run the user exit and apply the row-level operation (insert, update, or delete) to the appropriate target table based on the original invoice date, such as, January 2004, February 2004, November 2006, and so on.
- Disable the default row-level or table-level operations, and replace them by invoking a user exit program that performs custom operations. For example, in response to a table-level truncate operation, you can develop a user exit that lets you do a soft delete rather than a hard delete on the target table.

10.10.1 To configure a user exit for a Java class

About this task

For Java class user exits, method names are predefined. This means that you can only enable and disable user exit programs. You need to configure a user exit in Java that implements the UserExitIF interface class provided by InfoSphere CDC for solidDB.

Procedure

1. Click **Configuration** > **Subscriptions**.
2. Select the subscription.
3. Click the **Table Mappings** view and select the table mapping.
4. Right-click and select **Edit Mapping Details**.
5. Click the **User Exits** tab.
6. Select **Java Class** from the **User Exit Type** list.
7. Type the name of the Java class user exit that implements the **UserExitIF** interface in the **Class Name** box.

   For example, you may have imported the UserExitIF interface, and the user exit program class that implements this interface in your function has the following definition: public class UE1 implements UserExitIF

   In the **Class Name** box, you need to type:

Option	Description
UE1	If it is a stand-alone class
<Java package>.UE1	If the class is included in a Java package (for example, com.datamirror.interface.UE1)

   The files you generate from compiling the user exit program must be located in a library or folder that is referenced by the CLASSPATH environment variable.
8. Type the parameters that you want to make available to the user exit program in the **Parameter** box.

   You can access the parameters in the user exit program class by invoking the **getParameter( )** method during the initialization process. There are no conventions for specifying the parameters. The values you type in this box are free-form. The string of parameter values cannot exceed 255 characters in length.
9. Type the name of the user exit programs you want InfoSphere CDC to call beside one or more of the following operations:

Option	Description
Before Insert	InfoSphere CDC runs the user exit before replicating an insert operation.
After Insert	InfoSphere CDC runs the user exit after replicating an insert operation.
Before Update	InfoSphere CDC runs the user exit before replicating an update operation.
After Update	InfoSphere CDC runs the user exit after replicating an update operation.
Before Delete	InfoSphere CDC runs the user exit before replicating a delete operation.

Option	Description
After Delete	InfoSphere CDC runs the user exit after replicating a delete operation.
Before Refresh	InfoSphere CDC runs the user exit before replicating a refresh operation.
After Refresh	InfoSphere CDC runs the user exit after replicating a refresh operation.
Before Truncate	InfoSphere CDC runs the user exit before replicating a truncate operation.
After Truncate	InfoSphere CDC runs the user exit after replicating a truncate operation.

10. Click **Apply**.

10.11 System parameters for InfoSphere CDC for solidDB

System parameters let you control the behavior of InfoSphere CDC. If your replication environment requires a particular configuration, then you can use system parameters to modify the behavior of default operations in InfoSphere CDC. The default system parameter settings are appropriate for most installations. Maintain these default settings until you become familiar with the configuration of InfoSphere CDC.

InfoSphere CDC provides system parameters that control the behavior of your source and target datastores.

Note: If you make changes to a system parameter during active replication, you must stop and restart replication for the changes to take effect.

10.11.1 General product system parameters

General product system parameters let you control basic features of InfoSphere CDC and information you may have specified during installation.

mirror_auto_restart_interval_minutes

Use this parameter to indicate how frequently (in minutes) InfoSphere CDC will attempt to automatically restart continuous mirroring for persistent subscriptions. InfoSphere CDC will continue in its attempts to restart mirroring at a defined interval until it is either successful, an unrecoverable error occurs or the process is manually stopped.

Applies To—Source

Default Setting—0 minutes

Minimum Setting—0 minutes

Maximum Setting—60 minutes

retrieve_credentials

This system parameter defines whether InfoSphere CDC tries to fetch the backend login data for SQL passthrough purposes in Universal Cache.

Set this parameter to one of the following:

- true—Indicates that InfoSphere CDC tries to fetch the backend login data for SQL passthrough purposes in Universal Cache.
- false—Indicates that InfoSphere CDC does not try to fetch the backend login data for SQL passthrough purposes in Universal Cache. The 'false' setting is needed if the backend data server is DB2 for iSeries or DB2 for z/OS.

Applies To—Source

Default Setting—true

10.11.2 Notification system parameters

Notification system parameters let you control if you should generate InfoSphere CDC messages in the **Event Log** for specific events.

events_max_retain

Use this system parameter to control the maximum number of events that InfoSphere CDC will store in the event log. If InfoSphere CDC generates more events than the specified maximum, then the earliest events will be deleted.

Default Setting—10000

Minimum Setting— -1

Maximum Setting—2147483647

global_shutdown_after_no_heartbeat_response_minutes

Use this system parameter to specify the duration, in minutes, of communication inactivity before active InfoSphere CDC processes for a subscription are stopped. If a value outside the acceptable range is specified, the default setting is used.

Applies To—Source

Default Setting—15 minutes

Minimum Setting—3 minutes

Maximum Setting—999 minutes

global_conversion_not_possible_warning

Use this system parameter to control whether or not InfoSphere CDC generates a warning in the Management Console **Event Log** in the following situations:

- Data conversion is not possible for a specific data value.
- Converted data types are encountered that are out of range.

Set this parameter to one of the following:

true—generates a warning in the **Event Log** if data conversion is not possible for a specific data value or converted data types are encountered that are out of range.

false—does not generate a warning in the **Event Log** if data conversion is not possible for a specific data value or converted data types are encountered that are out of range.

Applies To—Target

Default Setting—False

implicit_transformation_warning

Use this system parameter when you want InfoSphere CDC to generate a warning message in the Event Log to identify possible problems with data transformations on the target table. When this system parameter is enabled, data transformations that meet the criteria for warnings are logged to the Event Log. For example, InfoSphere CDC will generate a warning if it had to truncate data in a column with variable length encoding in order to fit it into the target column.

Each time you restart InfoSphere CDC, InfoSphere CDC will generate one warning for each column in a table even if the conversions may be different each time. You should check the Event Log for new warnings related to data conversion to ensure data consistency between your source and target tables.

Note: This system parameter applies only when the **convert_not_nullable_column** parameter has been set to true.

Set this parameter to one of the following:

- **true**—Generates a warning message in the Event Log the first time an issue is detected on a per column per table basis.
- **false**—No warning message is generated in the Event Log.

Applies To—Target

Default Setting—True

Related reference:

"convert_not_nullable_column" on page 172

10.11.3 Maximize throughput system parameters

InfoSphere CDC system parameters allow you to significantly reduce the workload of the target database during mirroring. The InfoSphere CDC apply process groups transactions on the target to reduce the workload. Every commit on the target database will correspond with a commit on the source. However, it may not perform every commit that was done on the source. For example, if the source does three small transactions containing one operation each, the target may commit all three operations as part of a single transaction. You can use this grouping of system parameters to significantly reduce the resources required by the target database. The default settings are appropriate for most databases, but if your target system has limited resources and an increase in latency is acceptable, you can adjust the settings appropriately.

mirror_commit_after_max_transactions

This system parameter specifies the maximum number of transactions that are grouped together before a commit. Normally, commits issued to the target database are in response to commits issued by applications running on the source. You can use this system parameter to manage commits by controlling how often they are issued to the target database. This approach can be used to reduce the overhead of frequent commits to the database.

Applies To—Target

Default Setting—10

Minimum Setting—1

mirror_commit_after_max_seconds

This system parameter specifies the amount of time, in seconds, before committing small transactions to the target database. Normally, commits issued to the target database are in response to commits issued by applications running on the source. You can use this system parameter to manage commits by controlling how often they are issued to the target database. This approach can be used to reduce the overhead of frequent commits to the database.

Applies To—Target

Default Setting—1 second

Minimum Setting—1

mirror_commit_after_max_operations

This system parameter specifies the number of operations that must be applied to the target database before a commit is issued. Normally, commits issued to the target database are in response to commits issued by applications running on the source. You can use this system parameter to manage commits by controlling how often they are issued to the target database. This approach can be used to reduce the overhead of frequent commits to the database.

Applies To—Target

Default Setting—1000

Minimum Setting—1

mirror_commit_on_transaction_boundary

This system parameter indicates whether or not the commits that InfoSphere CDC does on the target database will always correspond with a commit that occurred on the source database. If you choose to ignore the commitment control of the source database, InfoSphere CDC allows you to see the partial results of large transactions.

Set this parameter to one of the following:

- true—Does not ignore the commitment control of the source database. Only records in a committed transaction are mirrored to the target. This setting provides true transaction consistency by ensuring that only committed transactions are sent to the target.
- false—Ignores the commitment control of the source database. This value disables commitment control for transaction processing. No attempt to maintain transaction consistency is performed during mirroring.

Applies To—Target

Default Setting—true

mirror_interim_commit_threshold

By default, InfoSphere CDC guarantees transactional delivery of change data to the target. This ensures that if any data from a source transaction is committed to the target, then all other in scope operations from that source transaction are committed as well.

In cases where large transactions are done on the source system, it can be more efficient to break a large transaction into smaller transactions when applying data to the target. You can configure this behavior using this system parameter.

The default value of 0 for this system parameter indicates that the product guarantees transactional delivery of change data to the target. A value greater than 0 indicates that you want InfoSphere CDC to break up large transactions into smaller transactions. For example, a value of 2000 indicates that InfoSphere CDC will split up large source transactions so that each transaction committed to the target database contains no more than 2000 operations.

Applies To—Target

Default Setting—0

Minimum Setting—0

refresh_commit_after_max_operations

This system parameter identifies the number of rows comprising each transaction during refresh. To reduce the workload on the target database during refresh, InfoSphere CDC periodically commits the changes to the target database rather than performing the refresh as a single large transaction.

Applies To—Target

Default Setting—1000

Minimum Setting—1

10.11.4 Encoding system parameters

For some system parameters, you can set the default method for treating data in defined Unicode columns, and you can set the default character encoding for your database.

global_unicode_as_char

This system parameter indicates the default method of treating data in defined Unicode columns. For each InfoSphere CDC installation on a server, this system parameter defines the system default method of treating data in Unicode columns. If a Unicode column is set to the system default, the current system default method, as defined by this system parameter, is used.

Set this parameter to one of the following:

- true—InfoSphere CDC treats all data in Unicode columns as single-byte characters. Use this setting when Unicode columns contain single-byte character data.
- false—InfoSphere CDC treats all data in Unicode columns as a continuous bit stream. Use this setting when Unicode columns contain non-single-byte

character data. Setting this system parameter to false ensures that InfoSphere CDC handles non-single-byte character data in the same way as previous InfoSphere CDC releases.

Note: Setting this parameter to false does not ensure that replicated non-single-byte character data in Unicode columns are represented properly on the target. For replicated non-single-byte character data, you may have to apply user exit programs or other customization to properly represent data in Unicode columns. For more information about user exit programs, see the *InfoSphere CDC End-User Documentation* for your platform.

Applies To—Source

Default Setting—false

10.11.5 Disk resource system parameters

Some system parameters control memory usage in InfoSphere CDC. For improved performance, if you are able to allocate more than the default value of 512 MB for the InfoSphere CDC Java Virtual Machine, then you can adjust the disk resource system parameters to use the increased memory.

mirror_global_disk_quota_gb

Use this system parameter to globally set a disk quota (in GB) for all capture components including temporary files, transaction queues, and LOBs which are staged on the target before being applied. InfoSphere CDC will manage disk space utilization across all components as required.

Most databases have a mechanism that allows you to roll back or undo changes to your database by storing uncommitted changes. Similarly, InfoSphere CDC uses the disk quota controlled by this system parameter to store in scope change data that has not been committed in your database. Once the database transaction is committed, the disk space used by the transaction is released.

Note: InfoSphere CDC stores the data in memory to improve performance and will only persist the data to disk if memory is unavailable.

The default setting of this system parameter is such that the product will only stop replicating after this disk quota exhausts all available disk space on your system. If you would prefer InfoSphere CDC to stop replicating after it uses a specific amount of disk space, you can set the value with this system parameter.

Applies To—Source and Target

Default Setting—9223372036854775807

Maximum Setting—9223372036854775807

Minimum Setting—1

mirror_memory_txqueue_total_mb

This system parameter controls the amount of memory used to stage data on the source. For optimal performance, this system parameter should be large enough to hold the largest amount of uncommited data that will ever exist in the source database.

Applies To—Source

Default Setting—15 Megabytes

mirror_memory_txqueue_each_mb

This system parameter controls the amount of memory used to stage data on the source. For optimal performance, this system parameter should be large enough to hold the data for the largest transactions that occur on the source.

Applies To—Source

Default Setting—3 Megabytes

global_memory_lob_cache_mb

This system parameter controls the amount of memory that will be used to stage LOB values on the target. For optimal performance, this value should be large enough to hold the entire data for the largest LOB values that will be replicated.

Default Setting—2 Megabytes

Applies To—Target

mirror_queue_for_buffers_between_cdc_threads_operations

This system parameter controls the ability of InfoSphere CDC's log scraping to take advantage of multiple processors. The default setting is acceptable for most situations. You can increase this value for highly scalable environments.

Applies To—Source

Default Setting—100 entries

Minimum Setting—100 entries

staging_store_can_run_independently

Use this system parameter to determine if subscriptions will exclusively use the InfoSphere CDC staging store to accumulate change data or if they will be allowed to use independent log readers and log parsers to receive data directly from the database logs.

Set this parameter to one of the following values:

- **true**—Specifies that subscriptions can use either the staging store to accumulate change data or use independent log readers and log parsers to receive data directly from the database logs.
- **false**—Specifies that subscriptions will use the InfoSphere CDC staging store to accumulate change data.

Changes to this system parameter value will only take effect after the replication engine is restarted.

If you change the value from true to false, you will need to clear the staging store before starting replication.

Applies To—Source

Default Setting—true

staging_store_disk_quota_gb

Use this system parameter to specify the maximum amount of disk space (in GB) that will be utilized by the InfoSphere CDC staging store on your source system.

Applies To—Source

Default Setting—100 (GB)

Maximum Setting—2147483647 (GB)

Minimum Setting—1 (GB)

10.11.6 Apply process system parameters

Some system parameters adjust the way InfoSphere CDC applies rows, column data, and error handling.

convert_not_nullable_column

Use this system parameter to control how InfoSphere CDC behaves when it applies a null value to a non-nullable column. When set to true (default value), the null value will be replaced with a default value (based on the data type of the column) and InfoSphere CDC will generate a warning in the event log. When set to false, InfoSphere CDC applies the null value directly to the non-nullable column which will result in an error and cause a shutdown of InfoSphere CDC on the target database.

Set this parameter to one of the following:

- **true**—Null value will be replaced with default value for non-nullable column, and a warning event will be logged.
- **false**—Null data will be applied as-is to non-nullable column and database will usually return an error code.

Applies To—Source and Target

Default Setting—True

global_max_batch_size

Use this system parameter to specify the maximum number of rows that InfoSphere CDC can place in an array and apply to the target database during refresh or mirroring. InfoSphere CDC collects rows and places them in an array (in memory) while receiving table-level operations from the source system. InfoSphere CDC applies the rows from the array when there is a change to a different table, when there is a new table-level operation, or when the maximum number of rows in an array has been reached.

You can use this parameter during mirroring only if **mirror_end_on_error** is true and **mirror_expected_errors_list** is empty. Use only during a refresh if **refresh_end_on_error** is true and **refresh_expected_errors_list** is empty. Before InfoSphere CDC places rows into an array, it allocates memory for the maximum number of rows you specify and multiplies this integer by the maximum length of a row. If the maximum number of rows is too large, then InfoSphere CDC cannot allocate enough memory and will shut down. Management Console references this area to present replication latency information.

Applies To—Target

Default Setting—25 rows

Maximum Setting—2147483647 rows

Minimum Setting—1 row

mirror_end_on_error

Use this system parameter to indicate if you want to end mirroring after an apply error occurs on the target database.

Set this parameter to one of the following:

- true—End mirroring after an apply error on the target database.
- false—Do not end mirroring after an apply error on the target database.

Applies To—Target

Default Setting—true

refresh_end_on_error

Use this system parameter to indicate if you want to end a refresh after an apply error occurs.

Set this parameter to one of the following:

- true—End a refresh after an apply error occurs.
- false—Do not end a refresh after an apply error occurs.

Applies To—Target

Default Setting—true

refresh_with_referential_integrity

Use this system parameter to indicate whether or not you want the data removed from all the target tables being refreshed before repopulating any of them. This is most useful when there are referential integrity constraints on the tables being refreshed.

Set this parameter to one of the following:

- true—Indicates that InfoSphere CDC will first remove all data in reverse of the specified refresh order. When specifying the refresh order, in general parent tables should appear before the referenced child tables.
- false—Indicates that InfoSphere CDC will not first remove all data from your tables and will refresh the tables in the specified order.

Applies To—Source

Default Setting—false

solid_fast_refresh_apply_pipes

Use this system parameter to improve the performance of fast refreshes. The fast refresh feature reduces the amount of time needed to replicate a large amount of data from the backend data server to the solidDB frontend.

Set this parameter to match the number of processors (cores) in your system.

Applies To—Target

Default Setting—2

Related reference:

"solid_fast_refresh_on"

solid_fast_refresh_on

Use this system parameter to control the fast refresh feature. The fast refresh feature reduces the amount of time needed to replicate a large amount of data from the backend data server to the solidDB frontend.

Set this parameter to one of the following:

- true—Indicates that fast refresh is enabled.
- false—Indicates that fast refresh is disabled.

Applies To—Target

Default Setting—false

Related reference:

"solid_fast_refresh_apply_pipes" on page 173

userexit_max_lob_size_kb

Use this system parameter to set the maximum size of LOB data (in kb) that InfoSphere CDC can pass to a user exit.

Applies To—Target

Default Setting—128 KB

Maximum Setting—9223372036854775807 KB

Minimum Setting—1 KB

Appendix A. Log Reader parameters

The Log Reader parameters appear in the `[LogReader]` section of the client-side `solid.ini` configuration file.

Table 36. Log Reader parameters

[LogReader]	Description	Factory Value	Access Mode
LogReaderEnabled	By using this parameter, you can enable or disable the log reader capability. In Universal Cache and InfoSphere CDC replication setups, this parameter must be set to `yes`.	no	RO
MaxLogSize	This parameter defines the size of the protected portion of the disk-based transaction log. When the log files are removed (for example after a backup), at least the specified amount of the log data is retained. The protected portion of the log facilitates a possible catchup after a failure case when the replication has not been active for some time. The actual log size may exceed the **`MaxLogSize`** value, if the log files are not removed. Catchup is possible as long as the propagator log position is within the existing log. The minimum value is 5 (5 MB). If you attempt to define a smaller log size, it is automatically changed to 5 MB. The maximum possible log size is practically unlimited. Unit: megabytes.	10240	RW

Table 36. Log Reader parameters (continued)

[LogReader]	Description	Factory Value	Access Mode
MaxSpace	This parameter defines the maximum number of log records buffered before slowdown. The log records are buffered in an in-memory log reader buffer. The size of a log record is that of the (binary) row size, plus a few bytes of additional metadata overhead. When the buffer fills up, throughput throttling is applied in the solidDB server: the operations are blocked until there is room in the logreader buffer. The throttling only takes place when the log reading is active. If there is no log reader activity, the solidDB server continues the processing and log files are preserved at least until the defined **MaxLogSize** limit is reached (see above).	100000	RW
MaxMemLogSize	Maximum size of the Log Reader logfile in memory, when logging is not enabled (**Logging.LogEnabled = no**). After maximum size is reached, logreader catchup might not be possible anymore. Unit: megabytes.	1 MB	RW
Silent	If set to `Yes`, the Log Reader activities are not output to `solmsg.out`. Possible values are `yes` and `no`.	no	RW/Startup
UseThrottling	Controls whether the log reader uses throttling to block operations until there is space in the log reader buffer.	yes	RW/Startup

Appendix B. SQL passthrough parameters

The SQL passthrough parameters appear in the [Passthrough] section of the client-side solid.ini configuration file.

Table 37. SQL passthrough parameters

[Passthrough]	Description	Factory value	Access mode
ComplexNumNonindexedConstr	This parameter specifies the minimum number of non-indexed WHERE clause constraints in a complex statement. If a statement has less non-indexed constraints of the following type, the statement is not complex and it is not passed through to the back end: the WHERE clause constraint does not resolve with index, the index does not exist, or the optimizer chooses different index for constraint. Value 0 (zero) means that number of non-indexed constraints is not used when estimating if the statement is complex. This parameter is effective only when the passthrough mode is CONDITIONAL. Use the performance counter *Passthrough complex by num non indexed constraints* to monitor the number of statements that are passed through when this parameter is set.	0	RW
ComplexNumOrderedRows	This parameter specifies the minimum estimated number of rows which must be sorted in a complex statement. If a statement has less than the estimated number of sortable rows, the statement is not complex and it is not passed through to the back end. Value 0 (zero) means that number of sortable rows is not used when estimating if the statement is complex. This parameter is effective only when the passthrough mode is CONDITIONAL. Use the performance counter *Passthrough complex by num ordered rows* to monitor the number of statements that are passed through when this parameter is set.	0	RW
ComplexNumTables	This parameter specifies the minimum number of tables in a complex statement. If a statement has less tables than specified with this parameter, the statement is not complex and it is not passed through to the backend. Value 0 (zero) means that number of tables is not used when estimating if the statement is complex. This parameter is effective only when the passthrough mode is CONDITIONAL. Use the performance counter *Passthrough complex by num tables* to monitor the number of statements that are passed through when this parameter is set.	0	RW

Table 37. SQL passthrough parameters (continued)

[Passthrough]	Description	Factory value	Access mode

ErrorMapFileName	Specifies the file path and file name for mapping backend native error codes to solidDB error codes.
	`<file_path><file_name>`
	
	For example:
	`[Passthrough]`
	`ErrorMapFileName=myfiles/db2tosoliderrors.txt`
	
	If **ErrorMapFileName** is not defined or the error is not mapped, the native backend error codes are mapped to solidDB error 13456 (Passthrough backend error: SQLState=<value>, NativeError=<backend error identifier>, MessageText=<backend error description>).
	
	The entries in the mapping file have the following format:
	`<backend_error> <solidDB error>  ; rest of the line is comment`
	
	As in the `solid.ini` configuration file, semicolon can be used to add comments.
	
	Example:
	`; this file maps DB2 native errors to solidDB native errors`
	`-207 13015 ; column not found`
	`-407 13110 ; NULL not allowed for non NULL column`
	`; end of errormappings`
	
	For more examples on the mapping files, see the `samples/sqlpassthrough` directory in the solidDB installation directory.
		No factory value.	RW/ Startup

Force32bitODBCHandles	The **Force32bitODBCHandles** parameter is needed in 64-bit environments when the backend data server is DB2 for Linux, UNIX, and Windows and the IBM Data Server Driver for CLI and ODBC is used with direct linking. When set to `yes`, solidDB server treats the ODBC handles as 32-bit integers instead of the 64-bit void pointers that are native on the 64-bit platforms.	no	RW/ Startup
IgnoreOnDisabled	The **IgnoreOnDisabled** parameter defines how the application program perceives the fact that passthrough is disabled. If the value is `yes`, all the statements related to passthrough (`SET PASSTHROUGH ...`) are ignored. If the value is `no`, an error is return on any effort to execute those statements. Possible values are `yes` and `no`.	yes	R/W
PassthroughEnabled	The **PassthroughEnabled** parameter defines whether SQL passthrough is enabled or not. • If passthrough is enabled but it cannot be initialized (for example, driver is not found), errors are returned on each effort to pass a statement to the backend. • If the backend server is shut down in a controlled way, the value of the **PassthroughEnabled** parameter can be set dynamically to `no`. The behavior exposed to the applications is then defined with the **IgnoreOnDisabled** parameter. Possible values are `yes` and `no`.	no	RW/ Startup
RemoteServerDriverPath	The **RemoteServerDriverPath** parameter specifies the driver manager path or the driver path for the backend data server specific ODBC driver that solidDB is linked to.		RW/ Startup

Table 37. SQL passthrough parameters (continued)

[Passthrough]	Description	Factory value	Access mode
RemoteServerDSN	The **RemoteServerDSN** parameter specifies the data source name (if driver manager is used) or the connect string for the backend data server specific ODBC driver that solidDB is linked to. The connect string must in the format of the ODBC call SQLConnect(), as ServerNam.		RW/ Startup
SqlPassthroughRead	The **SqlPassthroughRead** parameter defines how read statements are passed from the solidDB server to the backend. Possible values are 'None', 'Conditional', and 'Force'.	none	R/W
SqlPassthroughWrite	The **SqlPassthroughWrite** parameter defines how write statements are passed from the solidDB server to the backend. Possible values are `none`, `conditional`, and `force`.	none	R/W

Appendix C. ODBC data type support in SQL passthrough

SQL passthrough supports all standard SQL data types that are supported by the solidDB server.

Proprietary data types that are specific to the backend data server are not supported.

Supported data types

Table 38. Supported data types

SQL type identifier [1]	Typical SQL data type [2]	Typical type description
SQL_CHAR	CHAR(n)	Character string of fixed string length *n*
SQL_VARCHAR	VARCHAR(n)	Variable-length character string with a maximum string length *n*
SQL_LONGVARCHAR	LONG VARCHAR	Variable length character data The maximum length is data source–dependent.[9]
SQL_WCHAR	WCHAR(n)	Unicode character string of fixed string length *n*
SQL_WVARCHAR	VARWCHAR(n)	Unicode variable-length character string with a maximum string length *n*
SQL_WLONGVARCHAR	LONGWVARCHAR	Unicode variable-length character data. The maximum length is data source–dependent.
SQL_DECIMAL	DECIMAL(p,s)	Signed, exact, numeric value with a precision of at least *p* and scale *s*. The maximum precision is driver-defined. 1 <= p <= 15; s <= p[4]
SQL_NUMERIC	NUMERIC(p,s)	Signed, exact, numeric value with a precision *p* and scale *s* 1 <= p <= 15; s <= p [4]
SQL_SMALLINT	SMALLINT	Exact numeric value with precision 5 and scale 0 (signed: –32,768 <= n <= 32,767, unsigned: 0 <= n <= 65,535)[3].
SQL_INTEGER	INTEGER	Exact numeric value with precision 10 and scale 0 (signed: –2[31] <= n <= 2[31] – 1, unsigned: 0 <= n <= 2[32] – 1)[3].
SQL_REAL	REAL	Signed, approximate, numeric value with a binary precision 24 (zero or absolute value 10[–38] to 10[38]).
SQL_FLOAT	FLOAT(p)	Signed, approximate, numeric value with a binary precision of at least p. The maximum precision is driver-defined.[5]
SQL_DOUBLE	DOUBLE PRECISION	Signed, approximate, numeric value with a binary precision 53 (zero or absolute value 10[–308] to 10[308]).
SQL_BIT==> SQL_INTEGER	BIT	Single bit binary data.[8]

Table 38. Supported data types (continued)

SQL type identifier [1]	Typical SQL data type [2]	Typical type description
SQL_TINYINT	TINYINT	Exact numeric value with precision 3 and scale 0 (signed: –128 <= n <= 127, unsigned: 0 <= n <= 255)[3].
SQL_BIGINT	BIGINT	Exact numeric value with precision 19 (if signed) or 20 (if unsigned) and scale 0 (signed: –2[63] <= n <= 2[63] – 1, unsigned: 0 <= n <= 2[64] – 1)[3],[9].
SQL_BINARY	BINARY(n)	Binary data of fixed length *n* [9]
SQL_VARBINARY	VARBINARY(n)	Variable length binary data of maximum length *n* The maximum is set by the user.[9]
SQL_LONGVARBINARY	LONG VARBINARY	Variable length binary data Maximum length is data source–dependent.[9]
SQL_TYPE_DATE[6]	DATE	Year, month, and day fields, conforming to the rules of the Gregorian calendar. For details, see Constraints of the Gregorian Calendar in the Microsoft *ODBC Programmer's Reference*.
SQL_TYPE_TIME[6]	TIME(p)	Hour, minute, and second fields, with valid values for hours of 00 to 23, valid values for minutes of 00 to 59, and valid values for seconds of 00 to 61. Precision *p* indicates the seconds precision.
SQL_TYPE_TIMESTAMP[6]	TIMESTAMP(p)	Year, month, day, hour, minute, and second fields, with valid values as defined for the DATE and TIME data types.

Converted data types

Table 39. Converted data types

SQL type identifier [1]	Typical SQL data type [2]	Typical type description
SQL_BIT==> SQL_INTEGER	BIT	Single bit binary data.[8]

Unsupported SQL standard data types

Table 40. Unsupported SQL standard data types

SQL type identifier [1]	Typical SQL data type [2]	Typical type description
SQL_TYPE_UTCDATETIME	UTCDATETIME	Year, month, day, hour, minute, second, utchour, and utcminute fields. The utchour and utcminute fields have 1/10 microsecond precision.
SQL_TYPE_UTCTIME	UTCTIME	Hour, minute, second, utchour, and utcminute fields. The utchour and utcminute fields have 1/10 microsecond precision.
SQL_INTERVAL_MONTH[7]	INTERVAL MONTH(p)	Number of months between two dates; *p* is the interval leading precision.

Table 40. Unsupported SQL standard data types (continued)

SQL type identifier [1]	Typical SQL data type [2]	Typical type description
SQL_INTERVAL_YEAR[7]	INTERVAL YEAR(p)	Number of years between two dates; *p* is the interval leading precision.
SQL_INTERVAL_YEAR_TO_MONTH[7]	INTERVAL YEAR(p) TO MONTH	Number of years and months between two dates; *p* is the interval leading precision.
SQL_INTERVAL_DAY[7]	INTERVAL DAY(p)	Number of days between two dates; *p* is the interval leading precision.
SQL_INTERVAL_HOUR[7]	INTERVAL HOUR(p)	Number of hours between two date/times; *p* is the interval leading precision.
SQL_INTERVAL_MINUTE[7]	INTERVAL MINUTE(p)	Number of minutes between two date/times; *p* is the interval leading precision.
SQL_INTERVAL_SECOND[7]	INTERVAL SECOND(p,q)	Number of seconds between two date/times; *p* is the interval leading precision and q is the interval seconds precision.
SQL_INTERVAL_DAY_TO_HOUR[7]	INTERVAL DAY(p) TO HOUR	Number of days/hours between two date/times; *p* is the interval leading precision.
SQL_INTERVAL_DAY_TO_MINUTE[7]	INTERVAL DAY(p) TO MINUTE	Number of days/hours/minutes between two date/times; *p* is the interval leading precision.
SQL_INTERVAL_DAY_TO_SECOND[7]	INTERVAL DAY(p) TO SECOND(q)	Number of days/hours/minutes/ seconds between two date/times; p is the interval leading precision and q is the interval seconds precision.
SQL_INTERVAL_HOUR_TO_MINUTE[7]	INTERVAL HOUR(p) TO MINUTE	Number of hours/minutes between two date/times; *p* is the interval leading precision.
SQL_INTERVAL_HOUR_TO_SECOND[7]	INTERVAL HOUR(p) TO SECOND(q)	Number of hours/minutes/ seconds between two date/times; *p* is the interval leading precision and *q* is the interval seconds precision.
SQL_INTERVAL_MINUTE_TO_SECOND[7]	INTERVAL MINUTE(p) TO SECOND(q)	Number of minutes/seconds between two date/times; *p* is the interval leading precision and *q* is the interval seconds precision.
SQL_GUID	GUID	Fixed length GUID

[1] This is the value returned in the DATA_TYPE column by a call to SQLGetTypeInfo.

[2] This is the value returned in the NAME and CREATE PARAMS column by a call to SQLGetTypeInfo. The NAME column returns the designation (for example, CHAR) whereas the CREATE PARAMS column returns a comma-separated list of creation parameters such as precision, scale, and length.

[3] An application uses SQLGetTypeInfo or SQLColAttribute to determine whether a particular data type or a particular column in a result set is unsigned.

[4] SQL_DECIMAL and SQL_NUMERIC data types differ only in their precision. The precision of a DECIMAL(p,s) is an implementation-defined decimal precision that is no less than p, whereas the precision of a NUMERIC(p,s) is exactly equal to p.

[5] Depending on the implementation, the precision of SQL_FLOAT can be either 24 or 53: if it is 24, the SQL_FLOAT data type is the same as SQL_REAL; if it is 53, the SQL_FLOAT data type is the same as SQL_DOUBLE.

[6] In ODBC 3.x, the SQL date, time, and timestamp data types are SQL_TYPE_DATE, SQL_TYPE_TIME, and SQL_TYPE_TIMESTAMP, respectively; in ODBC 2.x, the data types are SQL_DATE, SQL_TIME, and SQL_TIMESTAMP.

[7] For more information about the interval SQL data types, see Interval Data Types in the Microsoft *ODBC Programmer's Reference*.

[8] The SQL_BIT data type has different characteristics than the BIT type in SQL-92.

[9] This data type has no corresponding data type in SQL-92.

Appendix D. Formatting rules for the backend ODBC driver connect string (RemoteServerDSN parameter)

In SQL passthrough setups, the connect string for the backend ODBC driver is defined with the **RemoteServerDSN** parameter in the [Passthrough] section of the solid.ini configuration file. The format of the connect string depends on the ODBC driver.

Generic rules

- If the connect string contains semicolons (;), the connect string must be given in double quotation marks, without any spaces between the first equal sign and the double quotation mark.

  For example:

  ```
  [Passthrough]
  RemoteServerDSN="Driver={IBM DB2 ODBC DRIVER};Dat
     abase=my_ids;Hostname=9.212.253.10;Port=9088;protocol=TCPIP;"
  ```

- If the connect string needs to include the username and password for the backend database, %s can be used as a placeholder to mark where the username and password should appear. %s indicates that at connection time, the username and password are read from the SYS_SERVER system table.

  For example: IDS ODBC Driver with database *my_ids* at port *9088*:

  ```
  RemoteServerDSN=my_ids:Port=9088;%s,%s
  ```

  At connection time, the %s for username and %s for password are replaced with username *Admin* and password *pwd123* that are stored in the SYS_SERVER system table.

  ```
  RemoteServerDSN=my_ids:Port=9088;Admin,pwd123
  ```

Notices

This product is protected by U.S. patents 6144941, 7136912, 6970876, 7139775, 6978396, 7266702, 7406489, 7502796, and 7587429.

This product is assigned the U.S. Export Control Classification Number ECCN=5D992b.

This information was developed for products and services offered in the U.S.A.

IBM may not offer the products, services, or features discussed in this document in other countries. Consult your local IBM representative for information on the products and services currently available in your area. Any reference to an IBM product, program, or service is not intended to state or imply that only that IBM product, program, or service may be used. Any functionally equivalent product, program, or service that does not infringe any IBM intellectual property right may be used instead. However, it is the user's responsibility to evaluate and verify the operation of any non-IBM product, program, or service.

IBM may have patents or pending patent applications covering subject matter described in this document. The furnishing of this document does not grant you any license to these patents. You can send license inquiries, in writing, to:

IBM Director of Licensing
IBM Corporation
North Castle Drive
Armonk, NY 10504-1785
U.S.A.

For license inquiries regarding double-byte (DBCS) information, contact the IBM Intellectual Property Department in your country or send inquiries, in writing, to:

Intellectual Property Licensing
Legal and Intellectual Property Law
IBM Japan Ltd.
1623-14, Shimotsuruma, Yamato-shi
Kanagawa 242-8502 Japan

The following paragraph does not apply to the United Kingdom or any other country where such provisions are inconsistent with local law: INTERNATIONAL BUSINESS MACHINES CORPORATION PROVIDES THIS PUBLICATION "AS IS" WITHOUT WARRANTY OF ANY KIND, EITHER EXPRESS OR IMPLIED, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF NON-INFRINGEMENT, MERCHANTABILITY OR FITNESS FOR A PARTICULAR PURPOSE. Some states do not allow disclaimer of express or implied warranties in certain transactions, therefore, this statement may not apply to you.

This information could include technical inaccuracies or typographical errors. Changes are periodically made to the information herein; these changes will be incorporated in new editions of the publication. IBM may make improvements and/or changes in the product(s) and/or the program(s) described in this publication at any time without notice.

Any references in this information to non-IBM Web sites are provided for convenience only and do not in any manner serve as an endorsement of those Web sites. The materials at those Web sites are not part of the materials for this IBM product and use of those Web sites is at your own risk.

IBM may use or distribute any of the information you supply in any way it believes appropriate without incurring any obligation to you.

Licensees of this program who wish to have information about it for the purpose of enabling: (i) the exchange of information between independently created programs and other programs (including this one) and (ii) the mutual use of the information which has been exchanged, should contact:

IBM Canada Limited
Office of the Lab Director
8200 Warden Avenue
Markham, Ontario
L6G 1C7
CANADA

Such information may be available, subject to appropriate terms and conditions, including in some cases, payment of a fee.

The licensed program described in this document and all licensed material available for it are provided by IBM under terms of the IBM Customer Agreement, IBM International Program License Agreement or any equivalent agreement between us.

Any performance data contained herein was determined in a controlled environment. Therefore, the results obtained in other operating environments may vary significantly. Some measurements may have been made on development-level systems and there is no guarantee that these measurements will be the same on generally available systems. Furthermore, some measurements may have been estimated through extrapolation. Actual results may vary. Users of this document should verify the applicable data for their specific environment.

Information concerning non-IBM products was obtained from the suppliers of those products, their published announcements or other publicly available sources. IBM has not tested those products and cannot confirm the accuracy of performance, compatibility or any other claims related to non-IBM products. Questions on the capabilities of non-IBM products should be addressed to the suppliers of those products.

All statements regarding IBM's future direction or intent are subject to change or withdrawal without notice, and represent goals and objectives only.

This information is for planning purposes only. The information herein is subject to change before the products described become available.

This information contains examples of data and reports used in daily business operations. To illustrate them as completely as possible, the examples include the names of individuals, companies, brands, and products. All of these names are fictitious and any similarity to the names and addresses used by an actual business enterprise is entirely coincidental.

COPYRIGHT LICENSE:

This information contains sample application programs in source language, which illustrate programming techniques on various operating platforms. You may copy, modify, and distribute these sample programs in any form without payment to IBM, for the purposes of developing, using, marketing or distributing application programs conforming to the application programming interface for the operating platform for which the sample programs are written. These examples have not been thoroughly tested under all conditions. IBM, therefore, cannot guarantee or imply reliability, serviceability, or function of these programs. The sample programs are provided "AS IS", without warranty of any kind. IBM shall not be liable for any damages arising out of your use of the sample programs.

Each copy or any portion of these sample programs or any derivative work, must include a copyright notice as follows:

© your company name) (year). Portions of this code are derived from IBM Corp. Sample Programs.

© Copyright IBM Corp. _enter the year or years_. All rights reserved.

If you are viewing this information softcopy, the photographs and color illustrations may not appear.

Trademarks

IBM, the IBM logo, ibm.com, Solid, solidDB, InfoSphere, DB2, Informix, and WebSphere are trademarks or registered trademarks of International Business Machines Corp., registered in many jurisdictions worldwide. Other product 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.

Java and all Java-based trademarks and logos are trademarks or registered trademarks of Oracle and/or its affiliates.

Linux is a registered trademark of Linus Torvalds in the United States, other countries, or both.

Microsoft and Windows are trademarks of Microsoft Corporation in the United States, other countries, or both.

UNIX is a registered trademark of The Open Group in the United States and other countries.

Other product and service names might be trademarks of IBM or other companies.

SC27-3847-05