Command Line Processor (CHCCLP) (Version 10.2.1) 6
Getting started with the CHCCLP program 7
Supported operating systems 8
Running CHCCLP 9
Running the CHCCLP commands in interactive mode 10
To run the CHCCLP commands in interactive mode 11
Working with command results in interactive mode 12
Running the CHCCLP commands in batch mode 13
To run CHCCLP commands in batch mode 14
To run a script file with substitution variables 15
Working with command results in batch mode 16
Running the CHCCLP commands in embedded mode 17
To set up EmbeddedScript 18
Working with EmbeddedScript 19
Working with command results in embedded mode 22
Sample programs 25
Overriding the locale 28
Enabling verbose command output 29
Setting and working with context 30
CHCCLP Commands 32
Understanding command syntax 33
Environment commands 34
chcclp session set to cdc 35
clear context 36
exit 37
help 38
print 39
quit 40
save session 41
set debug 42
set variable 44
set verbose 45
show context 46
show session 47
show variables 48
show version 49
Access Server commands 50
connect server 51
disconnect server 52
modify server 53
show server 55
Datastore commands 56
connect datastore 57
define external target datastore 60
disconnect datastore 62
list database schemas 63
list database tables 64

list databases . . . 65
list datastores . . . 66
list datatypes . . . 67
list encodings . . . 68
list journal control fields . . . 69
select datastore . . . 70
show datastore . . . 72
Tables commands . . . 74
add replication table . . . 75
list index columns . . . 77
list replication tables . . . 79
list table columns . . . 81
list table indexes . . . 83
readd replication table . . . 85
remove replication table . . . 87
Subscription commands . . . 89
add subscription . . . 90
delete subscription . . . 93
describe subscription . . . 94
export subscription . . . 95
import subscription . . . 97
list subscriptions . . . 100
lock subscription . . . 102
modify latency thresholds . . . 103
modify subscription . . . 105
modify subscription datastage properties . . . 108
modify subscription user exit . . . 111
promote subscription . . . 113
select subscription . . . 116
show latency thresholds . . . 117
show subscription . . . 118
show subscription datastage properties . . . 120
show subscription user exit . . . 121
unlock subscription . . . 122
Table mapping commands . . . 124
add rule set . . . 125
add table mapping . . . 128
delete rule set . . . 134
delete table mapping . . . 135
export rule set . . . 136
export table mapping . . . 138
flag refresh . . . 139
list refresh order . . . 141
list rule set tables . . . 142
list rule sets . . . 143
list table mappings . . . 144
mark capture point . . . 146

modify refresh order . . . 148
modify rule set . . . 150
modify table mapping . . . 153
park table mapping . . . 157
promote rule set . . . 158
promote table mapping . . . 160
reassign table mapping . . . 163
select rule set . . . 164
select table mapping . . . 165
show rule set . . . 167
show table mapping . . . 169
Mapping details commands . . . 171
add data translation . . . 172
add derived column . . . 173
clear user exit . . . 176
delete data translation . . . 177
delete derived column . . . 178
filter source column . . . 179
list column encodings . . . 181
list column mappings . . . 183
list source columns . . . 185
map column . . . 187
modify column encoding . . . 190
modify data translation . . . 192
modify derived column . . . 193
modify operations . . . 196
modify user exit cdll . . . 204
modify user exit function . . . 206
modify user exit javaclass . . . 208
modify user exit storedproc . . . 211
show data translation . . . 213
show derived column . . . 214
show operations . . . 216
show user exit . . . 218
unmap column . . . 220
Replication commands . . . 221
clear datastore events . . . 222
clear subscription events . . . 223
end replication . . . 224
list datastore events . . . 227
list subscription events . . . 228
list subscription performance metrics . . . 229
list table performance metrics . . . 231
monitor replication . . . 233
monitor subscription activity . . . 234
monitor subscription busy tables . . . 235
monitor subscription latency . . . 236

monitor subscription performance . . . 237
monitor subscription refresh . . . 238
monitor table performance . . . 239
show datastore event details . . . 241
show subscription event details . . . 242
start mirroring . . . 243
start refresh . . . 245
Access Manager commands . . . 247
add access connection . . . 248
add access datastore . . . 250
add access user . . . 251
change login password . . . 253
close access user connections . . . 254
delete access connection . . . 255
delete access datastore . . . 256
delete access user . . . 257
list access datastores . . . 258
list access users . . . 259
modify access connection . . . 260
modify access datastore . . . 262
modify access user . . . 264
show access connection . . . 266
show access datastore . . . 268
show access user . . . 269
Support commands . . . 271
run support assistant . . . 272
Management Console Commands reference . . . 273
Using commands in Management Console . . . 274
Opening command environments . . . 275
To issue commands in a Windows environment . . . 276
To issue commands in a UNIX or Linux environment . . . 277
To issue commands in a Telnet environment . . . 278
Management Console commands . . . 279
addTable - Add Table to Catalog . . . 281
assignTable - Assign Source Table . . . 282
connectAgent - Connect to Datastore . . . 285
connectServer - Connect to Access Server . . . 286
deAssignTable - Unassign Target Table . . . 288
describe - Describe Selected Tables . . . 289
disconnectAgent - Disconnect from Datastore . . . 290
disconnectServer - Disconnect from Access Server . . . 291
errorHandle - Define Error Handler . . . 292
execute - Run Batch File . . . 293
exit - Close Command Environment . . . 294
getAgents - Get Accessible Datastores . . . 295
getSubscriptionActivity - Get Subscription Activity . . . 296
getSubscriptions - Get Subscriptions for Datastore . . . 297

getSubscriptionStatus - Get Subscription Status . . . 298
getSubscriptionTables - Get Subscription Tables . . . 299
getTableStatus - Get Source Table Status . . . 300
help - Display Command Help . . . 301
reAddTable - Re-add Table in Catalog . . . 302
removeTable - Remove Table from Catalog . . . 303
setReplication - Set Replication Method . . . 304
setTableStatus - Set Source Table Status . . . 306
startRefresh - Start Refresh . . . 307
startMirror - Start Mirroring . . . 309
stopAll - Stop All Replication Activities . . . 311
stopMirror - Stop Mirroring . . . 312
stopRefresh - Stop Refresh . . . 314
subscribeTable - Select Table to Subscription . . . 316
unSubscribeTable - Deselect Table from Subscription . . . 317
Scripting support . . . 318
Creating a script file . . . 319
Compiling a script file . . . 321
To compile a script file (Windows) . . . 322
To compile a script file (Linux or UNIX) . . . 323
To compile a script file (Telnet) . . . 324
Running a batch file . . . 325
User roles and permissions . . . 326
User roles . . . 327
User permissions . . . 328
Troubleshooting . . . 332
Using the IBM Support Assistant (ISA DC) . . . 333
To use ISA DC to collect data for a product problem (command line) . . . 334
To use ISA DC to collect data for a product problem (GUI) . . . 337
To use ISA DC to collect data for a question or an enhancement request (command line) . . . 339
To use ISA DC to collect data for a question or an enhancement request (GUI) . . . 341
Troubleshooting and contacting IBM Support . . . 343

Getting started with the CHCCLP program

CHCCLP is the command line interface for IBM®InfoSphere® Change Data Capture version 10.2 or later datastores. CHCCLP allows users to script InfoSphere CDC functionality, such as connecting to datastores, creating subscriptions, mapping tables, modifying mapping details, and controlling and monitoring replication.
The CHCCLP program can process the commands in one of three modes:

- **Interactive**—Run CHCCLP commands one at a time from an interactive command prompt.
- **Batch**—Assemble CHCCLP commands in a script file that is processed by a single invocation of the program. Batch mode uses the same syntax as Interactive mode.
- **Embedded**—Run CHCCLP commands through a Java™ programming interface.

Related concepts

- Running the CHCCLP commands in interactive mode
- Running the CHCCLP commands in batch mode
- Running the CHCCLP commands in embedded mode

Getting started with the CHCCLP program

CHCCLP is the command line interface for IBM®InfoSphere® Change Data Capture version 10.2 or later datastores. CHCCLP allows users to script InfoSphere CDC functionality, such as connecting to datastores, creating subscriptions, mapping tables, modifying mapping details, and controlling and monitoring replication.
The CHCCLP program can process the commands in one of three modes:

- **Interactive**—Run CHCCLP commands one at a time from an interactive command prompt.
- **Batch**—Assemble CHCCLP commands in a script file that is processed by a single invocation of the program. Batch mode uses the same syntax as Interactive mode.
- **Embedded**—Run CHCCLP commands through a Java™ programming interface.

Related concepts

- Running the CHCCLP commands in interactive mode
- Running the CHCCLP commands in batch mode
- Running the CHCCLP commands in embedded mode

Supported operating systems

The CHCCLP program runs on Linux, UNIX, Windows, z/OS®, and UNIX System Services (USS) on z/OS.

The CHCCLP commands generate replication definitions for all operating system environments that are supported by InfoSphere® CDC: z/OS, Linux, UNIX, and Windows. You must have connectivity to each server for which you are generating replication definitions; that is, you must be able to issue a database connection statement to each of the servers.

Note: Additional configuration steps are required to enable the CHCCLP to run natively on z/OS or on USS.

Running CHCCLP

If you want to use the CHCCLP simplified command line tool on a server other than the one on which Management Console or Access Server are currently installed and running, you can install copies of Access Server on any server from which you want to issue commands; the additional installations of Access Server do not need to be running for you to use the CHCCLP tool.

Note: Your command prompt or terminal window might need to have its size adjusted to larger than 80 characters in order to display command output without wrapping.

Running CHCCLP on Windows

- Open the Command Line Processor from the Windows Start menu. By default, the Management Console installer creates the following Start menu shortcut: IBM® InfoSphere® Change Data Capture/Management Console/Command Line ProcessorThis method can only be used to launch CHCCLP in interactive mode. The same shortcut is created by the Access Server installer.
- Alternatively, open an operating system command prompt, change to the installation folder for Management Console or Access Server, and in the bin folder, issue the following command: `chcclp`

The CHCCLP command starts the CHCCLP program and changes the command prompt to Repl >.

Running CHCCLP on Linux or UNIX

- Open an operating system command prompt, change to the installation folder for Access Server, and in the bin folder, issue the following command: `./chcclp`

The CHCCLP command starts the CHCCLP program and changes the command prompt to Repl >.

Running CHCCLP on z/OS

You can enable the CHCCLP command-line program to run on UNIX System Services (USS) for z/OS®.

1. Install the IBM SDK for z/OS, Java™ Technology Edition, Version 6 on USS. You can find the SDK at http://www.ibm.com/servers/eserver/zseries/software/java.
2. Set the JAVA_HOME environment variable to point to the Java installation, and CHC_HOME environment variable to point to the installation folder for CHCCLP:

```
export JAVA_HOME=/usr/lpp/java/J6.0

export CHC_HOME=/usr/lpp/chc
```

3. Update the PATH environment variable to include CHCCLP:`export`

```
PATH=$PATH:$CHC_HOME/bin
```

4. In the UNIX System Services prompt, issue the following command: `chcclp`

In this section, you will learn:

- **Running the CHCCLP commands in interactive mode**
- **Running the CHCCLP commands in batch mode**
- **Running the CHCCLP commands in embedded mode**

Running the CHCCLP commands in interactive mode

You can run the CHCCLP commands in interactive mode from a command prompt.
See also:

- **To run the CHCCLP commands in interactive mode**
- **Working with command results in interactive mode**

Parent topic: Running CHCCLP

To run the CHCCLP commands in interactive mode

Procedure

1. Launch CHCCLP on your operating system. The CHCCLP command starts the CHCCLP program in interactive mode by default and changes the command prompt to Repl >.

2. Issue any of the CHCCLP commands.
3. To exit the CHCCLP program, issue either of the following commands:`quit;`
   or
   ```
   exit;
   ```

Parent topic: Running the CHCCLP commands in interactive mode

Working with command results in interactive mode

In interactive mode, commands display their output in the prompt window. There are six results for a command:

- **Success with no results**—The command executes with no output and the interactive prompt returns to `Repl >`. Most commands that add or modify configuration use this method. For example, the command add subscription does not display any output if the subscription was added successfully.
- **Success with a text value**—The interactive prompt displays a formatted output string for a command. Commands like show datastore event details use this method. The command documentation identifies this result type as ResultStringValue.
- **Success with a list of values**—The interactive prompt displays a list of strings. The command help uses this output. The command documentation identifies this result type as ResultStringList.
- **Success with a table of key pairs or value pairs**—The interactive prompt displays a table of key pairs or value pairs containing the output of the command. Commands like show subscription use this method. The command documentation identifies this result type as ResultStringKeyValues.
- **Success with a table of records**—The interactive prompt displays a table of results with column headers and a row for each record. Commands like list subscriptions use this method. The command documentation identifies this result type as ResultStringTable.
- **Failure**—The command fails to execute and displays an error code and message.

Parent topic: Running the CHCCLP commands in interactive mode

Running the CHCCLP commands in batch mode

You can run the CHCCLP commands in batch mode by using an input file.

A CHCCLP input file is known as a script. A CHCCLP script contains the same commands used in interactive mode. Each command ends with a semicolon (;). The script can also contain comments. These lines start with a pound sign (#). The first command, excluding comments, in a script must be “CHCCLP session set to cdc” in order to identify the script for InfoSphere® CDC.

Commands are executed in sequence. If a command returns an error, batch processing terminates.

See also:

- **To run CHCCLP commands in batch mode**
- **To run a script file with substitution variables**
- **Working with command results in batch mode**

Parent topic: Running CHCCLP

To run CHCCLP commands in batch mode

Procedure

1. Follow the instructions for your operating system to open a system command prompt and locate the CHCCLP program.
2. Issue the following command: `CHCCLP –f <file>`
   where <file> is the fully qualified file name.

Parent topic: Running the CHCCLP commands in batch mode

To run a script file with substitution variables

Procedure

1. Create a script file containing a series of commands. For any parameter values for which you want to provide input at execution time, define a variable name and use %variableName% in place of the actual value. For example:

   ```
   CHCCLP session set to cdc;
   connect server
    hostname %hostname% port %port%
   username %username% password %password%;
   connect datastore name %source% context source;
   connect datastore name %target% context target;
   ...
   disconnect server;
   exit;
   ```

2. Open an operating system command prompt and issue the following command, providing variableName:value arguments for each substitution variable in the script:`CHCCLP –f script.txt hostname:myserver port:10101`

   ```
   username:user1 password:pass1 source:ds1 target:ds2
   ```

Parent topic: Running the CHCCLP commands in batch mode

Working with command results in batch mode

In batch mode, commands display their output in the prompt window. There are six results for a command:

- **Success with no results**—The command executes with no output and the interactive prompt returns to `Repl >`. Most commands that add or modify configuration use this method. For example, the command add subscription does not display any output if the subscription was added successfully.
- **Success with a text value**—The interactive prompt displays a formatted output string for a command. Commands like show datastore event details use this method. The command documentation identifies this result type as ResultStringValue.
- **Success with a list of values**—The interactive prompt displays a list of strings. The command help uses this output. The command documentation identifies this result type as ResultStringList.
- **Success with a table of key pairs or value pairs**—The interactive prompt displays a table of key pairs or value pairs containing the output of the command. Commands like show subscription use this method. The command documentation identifies this result type as ResultStringKeyValues.
- **Success with a table of records**—The interactive prompt displays a table of results with column headers and a row for each record. Commands like list subscriptions use this method. The command documentation identifies this result type as ResultStringTable.
- **Failure**—The command fails to execute and displays an error code and message.

Parent topic: Running the CHCCLP commands in batch mode

Running the CHCCLP commands in embedded mode

You can interact with the command line processor using EmbeddedScript in a Java™ program. EmbeddedScript accepts the same commands as interactive or batch mode and provides the ability to execute commands and retrieve results.
See also:

- **To set up EmbeddedScript**
- **Working with EmbeddedScript**
- **Working with command results in embedded mode**
- **Sample programs**

Parent topic: Running CHCCLP

Related concepts

- Getting started with the CHCCLP program

To set up EmbeddedScript

Procedure

1. Open your Java™ programming environment, create a new Java project, and add the following library from an Access Server installation:`<access server installation>/lib/CHCCLP.jar`
   If you want to copy the CHCCLP.jar file into your Java programming environment, rather than pointing to CHCCLP.jar in the Access Server installation, you must copy all jar files from <access server installation>/lib.

2. Create a new class, and add the following import statement:`import com.ibm.replication.cdc.scripting.*;`
   Additional imports may be required depending on your code.

Parent topic: Running the CHCCLP commands in embedded mode

Working with EmbeddedScript

The EmbeddedScript class provides support for starting a scripting session, executing scripting commands, retrieving results, and ending the session using the Java™ programming language.

Modifier and Type / Method	Description

void	`open()` throws EmbeddedScriptExceptionOpens a connection to the command line processor.
	`EmbeddedScript script = new EmbeddedScript();`
	`try`
	`{`
	`   script.open();`
	`   ...`
	`}`
	`catch (EmbeddedScriptException e)`
	`{`
	`   ...`
	`}`
	`finally`
	`{`
	`   script.close();`
	`}`

void	`execute() throws EmbeddedScriptException`Executes a script command. Statements do not need to be terminated with semicolon (;) when executing commands using EmbeddedScript. If the command succeeds, the results can be accessed by calling `getResult()`. If the command fails, an exception is thrown.
	`EmbeddedScript script = new EmbeddedScript();`
	
	`try`
	`{`
	`   script.open();`
	`   script.execute(connect server username user1 password pass1);`
	`   Result result = script.getResult();`
	`   ...`
	`}`
	`catch (EmbeddedScriptException e)`
	`{`
	` int resultCode = e.getResultCode();`
	` String message = e.getResultMessage();`
	`}`
	`finally`
	`{`
	`   script.close();`
	`}`

void	`close()`Closes from the command line processor session. Closing is required and ensures that the command line processor successfully disconnects from datastores and Access Server, if those connections are not closed using scripting commands. `EmbeddedScript script = new EmbeddedScript();` `try` `{` `script.open();` `...` `}` `catch (EmbeddedScriptException e)` `{` `...` `}` `finally` `{` `script.close();` `}`
Result	`getResult()`Returns the last result. Results can be printed to the console using `result.display(System.out)` or cast to a specific result type: ResultNullResultStringKeyValuesResultStringListResultStringTableResultStringValue`try` `{` `script.execute(list datastores);` `Result result = script.getResult();` `if (result instanceof ResultStringTable)` `{` `...` `}` `}` `catch (EmbeddedScriptException e)` `{` `...` `}`
int	`getResultCode()`Returns the result code for statement execution.
String	`getResultMessage()`Returns the last result message. The message may be null if the command executes successfully.
int	`getStatus()`Returns the status of the command as EmbeddedScript.SUCCESS or EmbeddedScript.FAILURE.
String	`getVerboseMessages()`If verbose is enabled, returns any messages that have been logged during command execution. To enable verbose mode, execute: `script.execute(set verbose);`

The EmbeddedScriptException class provides access to the result error code and result error message

Header	Header
int	`getResultCode()`Returns the result code for statement execution.
String	`getResultCodeAndMessage()`Returns a formatted result code and message.
String	`getResultMessage()`Returns the last result message. The message may be null if the command executes successfully.

Parent topic: Running the CHCCLP commands in embedded mode

Working with command results in embedded mode

The EmbeddedScript method `getResult()` returns the result of the last command that was executed There are five types of Result supported by EmbeddedScript. In order to cast a result to its specific type, check the result object to determine which class it is, or call `getType()`. While the documentation indicates the type of result returned for each command, it is recommended to confirm that the type is the correct instance before using it.

Using `instanceof` to check the type:

```
script.execute(list subscriptions);

Result result = script.getResult();

if (result instanceof ResultStringTable)

{

  ...

}
```

Using `getType()` to check the type:

```
script.execute(list subscriptions);

Result result = script.getResult();

switch (result.getType())

{

case Result.TABLE:

  ...

  break;

}
```

The key methods for each of the result types are as follows:

- `ResultNull` for commands with no output.
  - **String**
    - `toString()`Returns an empty string

- `ResultStringKeyValues` for commands that return a collection of string-based key-value pairs.
  - **String[]**
    - `getKeys()`
    - Returns an array containing the keys in the result.
  - **String[]**
    - `getNames()`
    - Returns the column names for the result.
  - **int**
    - `getRowCount()`
    - Returns the number of keys in the result.
  - **String**
    - `getValue(String key)`
    - Returns the value for a given key, or null if the key cannot be found.
  - **String**
    - `toString()`
    - Returns a formatted text representation of the result.
- `ResultStringList` for commands that return a list of strings.
  - **int**

    - `getRowCount()`
    - Returns the number of strings in the result.
  - **String**
    - `getValueAt(int row) throws IndexOutOfBoundsException`
    - Returns the value for a row index. Row indexes start from zero.
  - **String**
    - `toString()`
    - Returns a formatted text representation of the result.
- `ResultStringTable` for commands that return a table of records.
  - **int**
    - `getColumnCount()`
    - Returns the number of columns in the table.
  - **String**
    - `getColumnAt(int column) throws IndexOutOfBoundsException`
    - Returns the name of a column for the column index. Column indexes start from zero.
  - **int**
    - `getRowCount()`
    - Returns the number of rows in the result.
  - **String**
    - `getValueAt(int row, int column) throws ArrayIndexOutOfBoundsException`
    - Returns the value for a row and column index. Row indexes start from zero. Column indexes start from zero.
  - **String**
    - `getValueAt(int row, String columnName) throws ArrayIndexOutOfBoundsException`
    - Returns the value for a row and named column. Row indexes start from zero. Returns null if no value is found.
    -
      ```
      int rowCount = result.getRowCount();
      for (int row = 0; row < rowCount; row++)
      {
        String value = result.getValueAt(row, DATASTORE);
        …
      }
      ```
  - **int**
    - `lookupRow(String lookupColumnName, String lookupValue)`
    - Returns the row index where the named lookup column contains the given lookup value.
    -
      ```
      int row = result.lookupRow(LASTNAME, SMITH);
      String firstName = result.getValueAt(row, FIRSTNAME);
      String middleName = result.getValueAt(row, MIDDLENAME);
      ```
  - **String**
    - `lookupValue(String lookupColumnName, String keyColumnName, String keyValue)`
    - Returns the string value in the named lookup column where the named key column contains a key value.

  -
    ```
    String firstName = result.lookupValue(FIRSTNAME, LASTNAME,
     SMITH);
    String middleName = result.lookupValue(MIDDLENAME, LASTNAME,
     SMITH);
    ```
- **String**
  - `toString()`
  - Returns a formatted text representation of the result.

- `ResultValue` for commands that return a string value.
  - **String**
    - `getValue()`
    - Returns the value.
  - **String**
    - `toString()`
    - Returns a formatted text representation of the result.

The results for a command may contain elements that are translated. For example, an application run in Japanese will display column names, keys, and possibly values in Japanese. Applications written using EmbeddedScript must account for language differences, if those applications are run in multiple languages.

Parent topic: Running the CHCCLP commands in embedded mode

Sample programs

The following example connects to Access Server and two datastores, creates a subscription, maps a few tables, and starts replication.

```
import com.ibm.replication.cdc.scripting.EmbeddedScript;
import com.ibm.replication.cdc.scripting.EmbeddedScriptException;
import java.text.MessageFormat;

public class Sample1
{
  public static void main(String[] args)
  {
     EmbeddedScript script = new EmbeddedScript();

     try
     {
        script.open();

        script.execute(connect server username user1 password pass1);
        script.execute(connect datastore name DS1 context source);
        script.execute(connect datastore name DS2 context target);
        script.execute(add subscription name SUB1);

        String mapping = add table mapping
           + sourceSchema {0} sourceTable {1}
           + targetSchema {2} targetTable {3};

        script.execute(MessageFormat.format(mapping, new Object[] {
           USER1, TABLE_1, USER1, TABLE_1 }));
        script.execute(MessageFormat.format(mapping, new Object[] {
           USER1, TABLE_2, USER1, TABLE_2 }));
        script.execute(MessageFormat.format(mapping, new Object[] {
           USER1, TABLE_3, USER1, TABLE_3 }));

        script.execute(start mirroring);
        script.execute(disconnect server);
     }
     catch (EmbeddedScriptException e)
     {
        System.out.println(e.getResultCodeAndMessage());
     }
     finally
     {
        script.close();
     }
  }
}
```

The following example connects to Access Server and two datastores, gets the list of subscriptions with monitor summary and starts any that are not currently mirroring.

```
import com.ibm.replication.cdc.scripting.EmbeddedScript;
import com.ibm.replication.cdc.scripting.EmbeddedScriptException;
import com.ibm.replication.cdc.scripting.Result;
import com.ibm.replication.cdc.scripting.ResultStringTable;
public class Sample2
{
  public static void main(String[] args)
  {
     EmbeddedScript script = new EmbeddedScript();

     try
     {
        script.open();

        script.execute(connect server username user1 password pass1);
        script.execute(connect datastore name DS1 context source);
        script.execute(connect datastore name DS2 context target);
        script.execute(monitor replication);

        Result result = script.getResult();
        if (result.getType() == Result.TABLE)
        {
           ResultStringTable table = (ResultStringTable) result;

           // Prints the table to the console window.
           table.display(System.out);

           String subscription;
           String state;

           for (int row = 0, rows = table.getRowCount(); row < rows; row++)
           {
              subscription = table.getValueAt(row, SUBSCRIPTION);
              state = table.getValueAt(row, STATE);
              if (state.equals(Inactive))
              {
                 System.out.println(Starting  + subscription);
                 script.execute(start mirroring name  + subscription);
              }
           }
        }

        script.execute(disconnect server);
     }
     catch (EmbeddedScriptException e)
     {
        System.out.println(e.getResultCodeAndMessage());
     }
     finally
     {
        script.close();
     }
  }
}
```

Parent topic: Running the CHCCLP commands in embedded mode

Overriding the locale

You can override the locale for CHCCLP. The following list indicates the languages that are supported by InfoSphere® CDC:

- English (en)
- Japanese (ja)
- Korean (ko)
- Simplified Chinese (zh_CN)

To change the locale, open an operating system command prompt and issue the following command:

```
CHCCLP –L <locale>
```

where <locale> is the abbreviation for the supported language.

Enabling verbose command output

Verbose mode displays additional output for many commands. By default, verbose mode is not enabled.

Execute either of the following commands to turn on verbose mode:

```
set verbose;
```

```
set debug verbose yes;
```

Execute the following command to turn off verbose mode:

```
set debug verbose no;
```

In verbose mode, commands that normally run as “Success with no results” display a success code and message. Commands may additionally report progress as the command executes.

Setting and working with context

The context defines the Access Server, source and target datastores, subscription and table mapping that are currently active in the script. The context is similar to selection in Management Console. For example, in order to view or modify column mappings in Management Console, you must select a table mapping in the current subscription. In scripting, you must set the context to a table mapping. Setting a table mapping context requires a subscription to be the current context, and the subscription context requires a source and target datastore context.
To view the current context, execute the show context command.

```
CONTEXT           CURRENT OBJECT
----------------- ---------------------------------
Access Server     localhost@10101
Source Datastore  DS1
Target Datastore  DS2
Subscription      SUB1
Table Mapping     CDC.TABLE_1 - CDC.TABLE_1
```

Setting the Access Server context

When you connect to Access Server, the context is automatically set. Disconnecting from Access Server clears all context settings.

Setting the datastore context

When you connect to a datastore, the context is automatically set to the source and target, depending on the capabilities of the datastore. For example, connecting to a DB2® datastore sets the source and target context to that datastore, because the DB2 datastore can be used as a source and target in subscriptions. Connecting to an InfoSphere® DataStage® datastore sets the target context, because the InfoSphere DataStage datastore can only be used as a target.Specifying or changing the context can be performed during connection or afterwards. You do not have to disconnect and reconnect to change the context, and you can change the context as often as you need.
To connect to a DB2 datastore and set it to source and target context, execute the connect datastore command:

```
connect datastore name DB2DS;
```

If you now execute the add subscription command, the subscription will be created with DS2DS datastore as the source and target datastores for the subscription. If you want to create a subscription between two different datastores, you can either connect to those datastores and identify your intention using the following commands:

```
connect datastore DB2DS1 context source;
connect datastore DB2DS2 context target;
```

or you can connect to both datastores and specify the context after:

```
connect datastore DB2DS1;
connect datastore DB2DS2;
select datastore DB2DS1 context source;
select datastore DB2DS2 context target;
```

You can also execute the following, where the first connect statement will set the context for the source and target to DB2DS1, and the second connect will override the target context to DB2DS2:

```
connect datastore DB2DS1;
connect datastore DB2DS2 context target;
```

When you specify the context during connection, the context will only be set if the connection succeeds. If you need to change the context after you have connected, use the select datastore command. If the context is invalid for the datastore, an error will be reported, but the datastore connection will not be closed. For example, if you execute connect to an InfoSphere DataStage datastore with source context, the connection will succeed, but the context switch to source will fail.

When you disconnect from a datastore, the source and target context (as applicable) is cleared if that datastore was currently specified as the context.

Setting the subscription context

When you add a new subscription, its source and target datastores are controlled by the current context and the new subscription is automatically set as the current context. The subscription context can be changed at any time by executing the select subscription command:`select subscription name SUB1;`

When you delete a subscription, the subscription context is cleared.

Setting the table mapping context

When you add a new table mapping, the table mapping is added to the subscription that is the current context and the new table mapping is automatically set as the current context. The table mapping context can be changed at any time by executing the command:`select table mapping…`

The select table mapping command has various optional parameters to identify which table mapping to select. You do not need to specify all parameters. The command will succeed if a unique table mapping is identified, otherwise the command will report an error. For example, to select a table mapping based on the name of a source table, execute the following command:

```
select table mapping sourcetable TABLE1;
```

When you delete a table mapping, the table mapping context is cleared.

Related reference

- disconnect server
- connect datastore
- add subscription
- select subscription
- select table mapping

CHCCLP Commands

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.

In this section, you will learn:

- **Understanding command syntax**
- **Environment commands**
- **Access Server commands**
- **Datastore commands**
- **Tables commands**
- **Subscription commands**
- **Table mapping commands**
- **Mapping details commands**
- **Replication commands**
- **Access Manager commands**
- **Support commands**

Understanding command syntax

Command formats

For each command, the following items of information are provided:

- **Syntax and syntax diagram**—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. 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 variable value.
- Square brackets ([ ]) indicate an optional parameter. If you omit the parameter, InfoSphere® CDC will use a default value.

Using quotation marks in CHCCLP commands

Use double quotation marks around parameter values that contain special characters or syntax keywords, or around empty strings.

The following text is an example of setting a description for a subscription with spaces:

```
add subscription name TEST description "A test subscription";
```

If the parameter value contains double quotation marks, use single quotation marks around the value. Parameter values cannot contain a mixture of single and double quotes.

The following text is an example of setting descriptions that contain single or double quotes:

```
add subscription name TEST description 'A "test" subscription';
```

```
add subscription name TEST description "A 'test' subscription";
```

Parent topic:CHCCLP Commands

Environment commands

See also:

- **chcclp session set to cdc**
- **clear context**
- **exit**
- **help**
- **print**
- **quit**
- **save session**
- **set debug**
- **set variable**
- **set verbose**
- **show context**
- **show session**
- **show variables**
- **show version**

Parent topic:CHCCLP Commands

chcclp session set to cdc

Use this command to identify that the script executes InfoSphere® CDC commands. This command must be the first statement in a script file. It is not required when running commands in the interactive prompt.

Syntax diagram

.arrow, .syntaxarrow { fill: none; stroke: black; } .arrowheadStartEnd, .arrowheadRepSep, .arrowheadRepSepReturn { stroke: black; fill: black; } .arrowheadSeq, .arrowheadStartChoice, .arrowheadAfterChoice, .arrowheadStartRepGroup, .arrowheadEndRepGroup, .arrowheadRev { stroke: none; fill: none; } rect { fill: none; stroke: none; } rect.fragref,rect.syntaxfragref { fill: none; stroke: black; } text { fill: #000000; fill-opacity: 1; font-family: IBM Plex Sans,Arial Unicode MS,Arial,Helvetica; font-style: normal; font-weight: normal; font-size: 8pt; stroke: #000000; stroke-width: 0.1; } text.var, text.syntaxvar {font-style:italic;} chcclp session set to cdc

Syntax

```
chcclp session set to cdc;
```

Parameters

None.

Required role

Any

Result

None.

Sample

Identifies InfoSphere CDC scripts when executing from a script file.

```
chcclp session set to cdc;
connect server username user1 password password1;
connect datastore name DS1;
disconnect datastore;
disconnect server;
exit;
```

Parent topic: Environment commands

clear context

Use this command to clear the datastore, subscription or table mapping context.

Syntax diagram

.arrow, .syntaxarrow { fill: none; stroke: black; } .arrowheadStartEnd, .arrowheadRepSep, .arrowheadRepSepReturn { stroke: black; fill: black; } .arrowheadSeq, .arrowheadStartChoice, .arrowheadAfterChoice, .arrowheadStartRepGroup, .arrowheadEndRepGroup, .arrowheadRev { stroke: none; fill: none; } rect { fill: none; stroke: none; } rect.fragref,rect.syntaxfragref { fill: none; stroke: black; } text { fill: #000000; fill-opacity: 1; font-family: IBM Plex Sans,Arial Unicode MS,Arial,Helvetica; font-style: normal; font-weight: normal; font-size: 8pt; stroke: #000000; stroke-width: 0.1; } text.var, text.syntaxvar {font-style:italic;} clear contexttype datastoresubscriptiontableMapping

Syntax

```
clear context [type <value>];
```

Parameters

- [type <value>]

- Specifies which context to clear.Valid values:
  - **datastore**—Clears the current datastore context.
  - **subscription**—Clears the current subscriptoin context.
  - **tableMapping**—Clears the current tableMapping context.

Required role

Any

Result

This command returns a value of 0 if the command was successful and a negative value if the command fails. When verbose mode is enabled, the command prints success message CDC1204.

Sample

Clears the current datastore context.

```
connect server username user1 password password1;
connect datastore name DS1 context source;
connect datastore name DS2 context target;
select subscription name SUB1;
show context;
clear context type datastore;
show context;
disconnect server;
exit;
```

Parent topic: Environment commands

exit

Use this command to exit the interactive prompt or script. Access Server and datastores will be disconnected automatically.

Syntax diagram

.arrow, .syntaxarrow { fill: none; stroke: black; } .arrowheadStartEnd, .arrowheadRepSep, .arrowheadRepSepReturn { stroke: black; fill: black; } .arrowheadSeq, .arrowheadStartChoice, .arrowheadAfterChoice, .arrowheadStartRepGroup, .arrowheadEndRepGroup, .arrowheadRev { stroke: none; fill: none; } rect { fill: none; stroke: none; } rect.fragref,rect.syntaxfragref { fill: none; stroke: black; } text { fill: #000000; fill-opacity: 1; font-family: IBM Plex Sans,Arial Unicode MS,Arial,Helvetica; font-style: normal; font-weight: normal; font-size: 8pt; stroke: #000000; stroke-width: 0.1; } text.var, text.syntaxvar {font-style:italic;} exit

Syntax

```
exit;
```

Parameters

None. **Required role**

Any **Result**

None. **Sample**

Exits the interactive prompt. Access Server will be automatically disconnected.

```
connect server username user1 password password1;
```

```
exit;
```

Quits the interactive prompt. The connection to Access Server is automatically disconnected.

```
connect server username user1 password password1;
```

```
quit;
```

Parent topic: Environment commands

help

Use this command to display a list of available commands, a list of commands that belong to a category or information about a specific command.

Syntax diagram

.arrow, .syntaxarrow { fill: none; stroke: black; } .arrowheadStartEnd, .arrowheadRepSep, .arrowheadRepSepReturn { stroke: black; fill: black; } .arrowheadSeq, .arrowheadStartChoice, .arrowheadAfterChoice, .arrowheadStartRepGroup, .arrowheadEndRepGroup, .arrowheadRev { stroke: none; fill: none; } rect { fill: none; stroke: none; } rect.fragref,rect.syntaxfragref { fill: none; stroke: black; } text { fill: #000000; fill-opacity: 1; font-family: IBM Plex Sans,Arial Unicode MS,Arial,Helvetica; font-style: normal; font-weight: normal; font-size: 8pt; stroke: #000000; stroke-width: 0.1; } text.var, text.syntaxvar {font-style:italic;} help

Syntax

```
help <value>;
```

Required role

Any

Parameters

- <value>

- Specifies either the specific command or the category of commands for which you want help displayed. Partial names will list all matching commands.

Required role

Any

Result

The list of available commands or the help for a specified command.

Sample

Provides a list of all available commands.

```
help;
```

Provides information about the show datastore command.

```
help show datastore;
```

Lists the commands that contain the word column.

```
help column;
```

Parent topic: Environment commands

print

Use this command to print a message to the output.

Syntax diagram

.arrow, .syntaxarrow { fill: none; stroke: black; } .arrowheadStartEnd, .arrowheadRepSep, .arrowheadRepSepReturn { stroke: black; fill: black; } .arrowheadSeq, .arrowheadStartChoice, .arrowheadAfterChoice, .arrowheadStartRepGroup, .arrowheadEndRepGroup, .arrowheadRev { stroke: none; fill: none; } rect { fill: none; stroke: none; } rect.fragref,rect.syntaxfragref { fill: none; stroke: black; } text { fill: #000000; fill-opacity: 1; font-family: IBM Plex Sans,Arial Unicode MS,Arial,Helvetica; font-style: normal; font-weight: normal; font-size: 8pt; stroke: #000000; stroke-width: 0.1; } text.var, text.syntaxvar {font-style:italic;} print"message"

Syntax

```
print <message>;
```

Parameters

- <message>

- Specifies the string value that you want printed to the output.

Required role

Any

Result

A message printed to the command-line processor output window.

Sample

Prints a message to the output window.

```
print Connecting to Access Server...;
connect server username user1 password pass1;
exit;
```

Parent topic: Environment commands

quit

Use this command to exit the interactive mode.

Syntax diagram

.arrow, .syntaxarrow { fill: none; stroke: black; } .arrowheadStartEnd, .arrowheadRepSep, .arrowheadRepSepReturn { stroke: black; fill: black; } .arrowheadSeq, .arrowheadStartChoice, .arrowheadAfterChoice, .arrowheadStartRepGroup, .arrowheadEndRepGroup, .arrowheadRev { stroke: none; fill: none; } rect { fill: none; stroke: none; } rect.fragref,rect.syntaxfragref { fill: none; stroke: black; } text { fill: #000000; fill-opacity: 1; font-family: IBM Plex Sans,Arial Unicode MS,Arial,Helvetica; font-style: normal; font-weight: normal; font-size: 8pt; stroke: #000000; stroke-width: 0.1; } text.var, text.syntaxvar {font-style:italic;} quit

Syntax

```
quit;
```

Parameters

None.

Required role

Any

Result

None.

Sample

Quits the interactive prompt. The connection to Access Server is automatically disconnected.

```
connect server username user1 password password1;
quit;
```

Exits the interactive prompt. The connection to Access Server is automatically disconnected.

```
connect server username user1 password password1;
exit;
```

Parent topic: Environment commands

save session

Use this command to save the statements that were executed during the current command line processor session to file.

Syntax diagram

.arrow, .syntaxarrow { fill: none; stroke: black; } .arrowheadStartEnd, .arrowheadRepSep, .arrowheadRepSepReturn { stroke: black; fill: black; } .arrowheadSeq, .arrowheadStartChoice, .arrowheadAfterChoice, .arrowheadStartRepGroup, .arrowheadEndRepGroup, .arrowheadRev { stroke: none; fill: none; } rect { fill: none; stroke: none; } rect.fragref,rect.syntaxfragref { fill: none; stroke: black; } text { fill: #000000; fill-opacity: 1; font-family: IBM Plex Sans,Arial Unicode MS,Arial,Helvetica; font-style: normal; font-weight: normal; font-size: 8pt; stroke: #000000; stroke-width: 0.1; } text.var, text.syntaxvar {font-style:italic;} save sessionfilenamename

Syntax

```
save session filename <name>;
```

Parameters

- filename <name>

- Specifies the full path and name of the file where the statements will be saved. Use double quotation marks to surround the filename when it contains spaces. On Windows, use \\ for a single backslash.

Required role

Any

Result

This command returns a value of 0 if the command was successful and a negative value if the command fails. When verbose mode is enabled, the command prints success message CDC1006.

Sample

(Windows) Saves the statements executed during the session to file.

```
connect server
username user1 password password1;
connect datastore name DS1;
disconnect datastore;
disconnect server;
save session filename c:\\output\\script1.txt;
exit;
```

(Linux/UNIX) Saves the statements executed during the session to file.

```
connect server
username user1 password password1;
connect datastore name DS1;
disconnect datastore;
disconnect server;
save session filename //output//script1.txt;
exit;
```

Parent topic: Environment commands

set debug

Use this command to set the internal debug options for the scripting environment.

Syntax diagram

.arrow, .syntaxarrow { fill: none; stroke: black; } .arrowheadStartEnd, .arrowheadRepSep, .arrowheadRepSepReturn { stroke: black; fill: black; } .arrowheadSeq, .arrowheadStartChoice, .arrowheadAfterChoice, .arrowheadStartRepGroup, .arrowheadEndRepGroup, .arrowheadRev { stroke: none; fill: none; } rect { fill: none; stroke: none; } rect.fragref,rect.syntaxfragref { fill: none; stroke: black; } text { fill: #000000; fill-opacity: 1; font-family: IBM Plex Sans,Arial Unicode MS,Arial,Helvetica; font-style: normal; font-weight: normal; font-size: 8pt; stroke: #000000; stroke-width: 0.1; } text.var, text.syntaxvar {font-style:italic;} set debugverbosefalse/notrue/yestrace_context_changesfalse/no true/yestrace_parse_exceptionfalse/notrue/yestrace_parse_resultsfalse/notrue/yes trace_execution_timetrue/yesfalse/no

Syntax

```
set debug [verbose <value>] [trace_context_changes <change>]

[trace_parse_exception <exception>] [trace_parse_results <result>]

[trace_execution_time <execution>];
```

Parameters

- [verbose <value>]

- Turns on verbose output for script execution.Valid values:
  - **true/yes**—Enables verbose output. Either yes or true are valid values..
  - **false/no**—Does not enable verbose output. Either no or false are valid values.

  Default value: false/no

- [trace_context_changes <change>]

- Traces context changes.Valid values:
  - **true/yes**—Enables tracing for context change. Either yes or true are valid values..
  - **false/no**—Does not enable tracing for context change. Either no or false are valid values.

  Default value: false/no

- [trace_parse_exception <exception>]

- Traces parser exceptions.Valid values:
  - **true/yes**—Enables tracing for parser exception. Either yes or true are valid values..
  - **false/no**—Does not enable tracing for parser exception. Either no or false are valid values.

  Default value: false/no

- [trace_parse_results <result>]

- Traces parse results.Valid values:
  - **true/yes**—Enables tracing for parse results. Either yes or true are valid values..
  - **false/no**—Does not enable tracing for parse results. Either no or false are valid values.

  Default value: false/no

- **[trace_execution_time <execution>]**
  - Traces the start and end time for a command.Valid values:
    - **true/yes**—Enables tracing for execution time. Either yes or true are valid values..
    - **false/no**—Does not enable tracing for execution time. Either no or false are valid values.

Required role

Any

Result

This command returns a value of 0 if the command was successful and a negative value if the command fails. When verbose mode is enabled, the command prints success message CDC1002.

Sample

Toggles verbose output on and off.

```
set debug verbose true;
...
set debug verbose false;
```

Parent topic: Environment commands

set variable

Use this command to store a variable by name for use in the script. To use a variable as a value parameter, use %name%. If a value already exists for the named parameter, the set variable command will update it with the new value. To display all variables in the current session, use the show variables command.

Syntax diagram

.arrow, .syntaxarrow { fill: none; stroke: black; } .arrowheadStartEnd, .arrowheadRepSep, .arrowheadRepSepReturn { stroke: black; fill: black; } .arrowheadSeq, .arrowheadStartChoice, .arrowheadAfterChoice, .arrowheadStartRepGroup, .arrowheadEndRepGroup, .arrowheadRev { stroke: none; fill: none; } rect { fill: none; stroke: none; } rect.fragref,rect.syntaxfragref { fill: none; stroke: black; } text { fill: #000000; fill-opacity: 1; font-family: IBM Plex Sans,Arial Unicode MS,Arial,Helvetica; font-style: normal; font-weight: normal; font-size: 8pt; stroke: #000000; stroke-width: 0.1; } text.var, text.syntaxvar {font-style:italic;} set variablenamevariablevaluestore

Syntax

```
set variable name <variable> value <store>;
```

Parameters

- name <variable>

- Specifies the name of the variable.

- value <store>

- Specifies the value to store.

Required role

Any

Result

This command returns a value of 0 if the command was successful and a negative value if the command fails. When verbose mode is enabled, the command prints success message CDC1003.

Sample

Uses variable to provide the Access Server login credentials and to identify the source and target datastores.

```
set variable name username value user1;
set variable name password value pass1;
set variable name sourceds value DS1;
set variable name targetds value DS2:
connect server username %username% password %password%;
connect datastore name %sourceds% context source;
connect datastore name %targetds% context target;
...
disconnect datastore name %sourceds%;
disconnect datastore name %targetds%;
disconnect server;
exit;
```

Parent topic: Environment commands

set verbose

Use this command to turn on verbose mode for script execution.Verbose mode can also be turned on and off using the set debug command. When verbose mode is turned on, commands may provide additional information about their progress and a result message in the output window.

Syntax diagram

.arrow, .syntaxarrow { fill: none; stroke: black; } .arrowheadStartEnd, .arrowheadRepSep, .arrowheadRepSepReturn { stroke: black; fill: black; } .arrowheadSeq, .arrowheadStartChoice, .arrowheadAfterChoice, .arrowheadStartRepGroup, .arrowheadEndRepGroup, .arrowheadRev { stroke: none; fill: none; } rect { fill: none; stroke: none; } rect.fragref,rect.syntaxfragref { fill: none; stroke: black; } text { fill: #000000; fill-opacity: 1; font-family: IBM Plex Sans,Arial Unicode MS,Arial,Helvetica; font-style: normal; font-weight: normal; font-size: 8pt; stroke: #000000; stroke-width: 0.1; } text.var, text.syntaxvar {font-style:italic;} set verbose

Syntax

```
set verbose;
```

Parameters

None.

Required role

Any

Result

This command returns a value of 0. When verbose mode is enabled, the command prints success message CDC1004.

Sample

Turns on verbose mode for commands.

```
set verbose;
```

Parent topic: Environment commands

show context

Use this command to show the current context for the scripting environment. Context settings are used by default when individual scripting commands do not override them. For example, the source and target datastore context are used to identify the datastores when creating a new subscription.

Syntax diagram

.arrow, .syntaxarrow { fill: none; stroke: black; } .arrowheadStartEnd, .arrowheadRepSep, .arrowheadRepSepReturn { stroke: black; fill: black; } .arrowheadSeq, .arrowheadStartChoice, .arrowheadAfterChoice, .arrowheadStartRepGroup, .arrowheadEndRepGroup, .arrowheadRev { stroke: none; fill: none; } rect { fill: none; stroke: none; } rect.fragref,rect.syntaxfragref { fill: none; stroke: black; } text { fill: #000000; fill-opacity: 1; font-family: IBM Plex Sans,Arial Unicode MS,Arial,Helvetica; font-style: normal; font-weight: normal; font-size: 8pt; stroke: #000000; stroke-width: 0.1; } text.var, text.syntaxvar {font-style:italic;} show context

Syntax

```
show context;
```

Parameters

None.

Required role

Any

Result

A ResultStringTable containing the current context settings.

```
CONTEXT            CURRENT OBJECT
------------------ ----------------------------------
Access Server      localhost@10101
Source Datastore   DS1
Target Datastore   DS2
Subscription       SUB1
Table Mapping      CDC.TABLE_1 - CDC.TABLE_1
```

Sample

Displays the current context settings.

```
connect server username user1 password password1;
connect datastore name DS1 context source;
connect datastore name DS2 context target;
add subscription name SUB1;
show context;
disconnect server;
exit;
```

Parent topic: Environment commands

show session

Use this command to view the set of commands that have been executed in the scripting session.

Syntax diagram

.arrow, .syntaxarrow { fill: none; stroke: black; } .arrowheadStartEnd, .arrowheadRepSep, .arrowheadRepSepReturn { stroke: black; fill: black; } .arrowheadSeq, .arrowheadStartChoice, .arrowheadAfterChoice, .arrowheadStartRepGroup, .arrowheadEndRepGroup, .arrowheadRev { stroke: none; fill: none; } rect { fill: none; stroke: none; } rect.fragref,rect.syntaxfragref { fill: none; stroke: black; } text { fill: #000000; fill-opacity: 1; font-family: IBM Plex Sans,Arial Unicode MS,Arial,Helvetica; font-style: normal; font-weight: normal; font-size: 8pt; stroke: #000000; stroke-width: 0.1; } text.var, text.syntaxvar {font-style:italic;} show session

Syntax

```
show session;
```

Parameters

None. **Required role**

Any **Result**

The set of commands that have been executed during the scripting session.

Sample

Displays the statements executed during the session to the output window.

```
connect server username user1 password password1;
connect datastore name DS1;
disconnect datastore;
disconnect server;
show session;
exit;
```

Parent topic: Environment commands

show variables

Use this command to view the current variables available for use as value parameters using the syntax %name%.The command lists variables set as command line arguments when running CHCCLP and variables set using the set variable command.

Syntax diagram

.arrow, .syntaxarrow { fill: none; stroke: black; } .arrowheadStartEnd, .arrowheadRepSep, .arrowheadRepSepReturn { stroke: black; fill: black; } .arrowheadSeq, .arrowheadStartChoice, .arrowheadAfterChoice, .arrowheadStartRepGroup, .arrowheadEndRepGroup, .arrowheadRev { stroke: none; fill: none; } rect { fill: none; stroke: none; } rect.fragref,rect.syntaxfragref { fill: none; stroke: black; } text { fill: #000000; fill-opacity: 1; font-family: IBM Plex Sans,Arial Unicode MS,Arial,Helvetica; font-style: normal; font-weight: normal; font-size: 8pt; stroke: #000000; stroke-width: 0.1; } text.var, text.syntaxvar {font-style:italic;} show variables

Syntax

```
show variables;
```

Parameters

None.

Required role

Any

Result

A ResultStringTable of the VARIABLE NAME and VALUE pairs.

Sample

Shows the current variables.

```
set variable name username value user1;

set variable name password value pass1;

set variable name sourceds value DS1;

set variable name targetds value DS2:

show variables;

exit;
```

Parent topic: Environment commands

show version

Use this command to display the command line processor name and version.

Syntax diagram

.arrow, .syntaxarrow { fill: none; stroke: black; } .arrowheadStartEnd, .arrowheadRepSep, .arrowheadRepSepReturn { stroke: black; fill: black; } .arrowheadSeq, .arrowheadStartChoice, .arrowheadAfterChoice, .arrowheadStartRepGroup, .arrowheadEndRepGroup, .arrowheadRev { stroke: none; fill: none; } rect { fill: none; stroke: none; } rect.fragref,rect.syntaxfragref { fill: none; stroke: black; } text { fill: #000000; fill-opacity: 1; font-family: IBM Plex Sans,Arial Unicode MS,Arial,Helvetica; font-style: normal; font-weight: normal; font-size: 8pt; stroke: #000000; stroke-width: 0.1; } text.var, text.syntaxvar {font-style:italic;} show version

Syntax

```
show version;
```

Parameters

None.

Required role

Any

Result

Displays the version of the command line processor that is currently running.

Sample

```
show version;

exit;
```

Parent topic: Environment commands

Access Server commands

See also:

- **connect server**
- **disconnect server**
- **modify server**
- **show server**

Parent topic:CHCCLP Commands

connect server

Use this command to connect to an Access Server.

Syntax diagram

.arrow, .syntaxarrow { fill: none; stroke: black; } .arrowheadStartEnd, .arrowheadRepSep, .arrowheadRepSepReturn { stroke: black; fill: black; } .arrowheadSeq, .arrowheadStartChoice, .arrowheadAfterChoice, .arrowheadStartRepGroup, .arrowheadEndRepGroup, .arrowheadRev { stroke: none; fill: none; } rect { fill: none; stroke: none; } rect.fragref,rect.syntaxfragref { fill: none; stroke: black; } text { fill: #000000; fill-opacity: 1; font-family: IBM Plex Sans,Arial Unicode MS,Arial,Helvetica; font-style: normal; font-weight: normal; font-size: 8pt; stroke: #000000; stroke-width: 0.1; } text.var, text.syntaxvar {font-style:italic;} connect serverhostnamehostportnumberusernamenamepasswordvalue

Syntax

```
connect server [hostname <host>] [port <number>] username <name> password <value>;
```

Parameters

- [hostname <host>]

- Specifies the hostname where Access Server is installed.Default value: localhost

- [port <number>]

- Specifies the port number used to connect to Access Server.Default value: 10101

- username <name>

- Specifies the username that will be used to log on to Access Server.

- password <value>

- Specifies the password for the username that will be used to log on to Access Server.

Required role

Any

Result

This command returns a value of 0 if the command was successful and a negative value if the command fails. When verbose mode is enabled, the command prints success message CDC1101.

Sample

Connects and disconnects from Access Server.

```
connect server
hostname mymachine port 10101
username user1 password password1;
disconnect server;
exit;
```

Parent topic: Access Server commands

disconnect server

Use this command to disconnect from an Access Server. Datastores will be disconnected automatically.

Syntax diagram

.arrow, .syntaxarrow { fill: none; stroke: black; } .arrowheadStartEnd, .arrowheadRepSep, .arrowheadRepSepReturn { stroke: black; fill: black; } .arrowheadSeq, .arrowheadStartChoice, .arrowheadAfterChoice, .arrowheadStartRepGroup, .arrowheadEndRepGroup, .arrowheadRev { stroke: none; fill: none; } rect { fill: none; stroke: none; } rect.fragref,rect.syntaxfragref { fill: none; stroke: black; } text { fill: #000000; fill-opacity: 1; font-family: IBM Plex Sans,Arial Unicode MS,Arial,Helvetica; font-style: normal; font-weight: normal; font-size: 8pt; stroke: #000000; stroke-width: 0.1; } text.var, text.syntaxvar {font-style:italic;} disconnect server

Syntax

```
disconnect server;
```

Parameters

None.

Required role

Any

Result

This command returns a value of 0 if the command was successful and a negative value if the command fails. When verbose mode is enabled, the command prints success message CDC1102.

Sample

Disconnects from Access Server.

```
connect server username user1 password password1;
disconnect server;
exit;
```

Parent topic: Access Server commands

modify server

Use this command to modify the attributes for Access Server.Using this command, you can enable message tracing and logging. The options are automatically turned off when you restart Access Server. Use show server to display the current options.

Syntax diagram

.arrow, .syntaxarrow { fill: none; stroke: black; } .arrowheadStartEnd, .arrowheadRepSep, .arrowheadRepSepReturn { stroke: black; fill: black; } .arrowheadSeq, .arrowheadStartChoice, .arrowheadAfterChoice, .arrowheadStartRepGroup, .arrowheadEndRepGroup, .arrowheadRev { stroke: none; fill: none; } rect { fill: none; stroke: none; } rect.fragref,rect.syntaxfragref { fill: none; stroke: black; } text { fill: #000000; fill-opacity: 1; font-family: IBM Plex Sans,Arial Unicode MS,Arial,Helvetica; font-style: normal; font-weight: normal; font-size: 8pt; stroke: #000000; stroke-width: 0.1; } text.var, text.syntaxvar {font-style:italic;} modify servertracingfalse/notrue/yesloggingfalse/notrue/yes

Syntax

```
modify server [tracing <trace>] [logging <log>];
```

Parameters

- [tracing <trace>]

- Enables or disables Access Server message tracing.Valid values:
  - **true/yes**—Message tracing is enabled. Either yes or true are valid values..
  - **false/no**—Message tracing is disabled. Either no or false are valid values..

  Default value: false/no

- [logging <log>]

- Enables or disables Access Server logging.Valid values:
  - **true/yes**—Logging is enabled. Either yes or true are valid values..
  - **false/no**—Logging is disabled. Either no or false are valid values..

  Default value: false/no

Required role

Any

Result

This command returns a value of 0 if the command was successful and a negative value if the command fails. When verbose mode is enabled, the command prints success message CDC1103.

Sample

Turns on message tracing. Use run support assistant to collect the trace after completing the task.

```
connect server username user1 password password1;
modify server tracing true;
...
modify server tracing false;
run support assistant filename c:\output\sa.zip;
disconnect server;
exit;
```

Parent topic: Access Server commands

Related reference

- show server
- run support assistant

show server

Use this command to show the properties of the Access Server to which you are connected.

Syntax diagram

.arrow, .syntaxarrow { fill: none; stroke: black; } .arrowheadStartEnd, .arrowheadRepSep, .arrowheadRepSepReturn { stroke: black; fill: black; } .arrowheadSeq, .arrowheadStartChoice, .arrowheadAfterChoice, .arrowheadStartRepGroup, .arrowheadEndRepGroup, .arrowheadRev { stroke: none; fill: none; } rect { fill: none; stroke: none; } rect.fragref,rect.syntaxfragref { fill: none; stroke: black; } text { fill: #000000; fill-opacity: 1; font-family: IBM Plex Sans,Arial Unicode MS,Arial,Helvetica; font-style: normal; font-weight: normal; font-size: 8pt; stroke: #000000; stroke-width: 0.1; } text.var, text.syntaxvar {font-style:italic;} show server

Syntax

```
show server;
```

Parameters

None.

Required role

Any

Result

A ResultStringKeyValues object containing the PROPERTY and VALUE attributes of the server.

```
PROPERTY                   VALUE
-------------------------- --------------------------
Version:                   10.2
Hostname:                  localhost
Port:                      10101
Username:                  user1
Message tracing:           Off
Logging:                   Off
```

Sample

Connects to Access Server and shows its properties.

```
connect server username user1 password password1;
show server;
disconnect server;
exit;
```

Parent topic: Access Server commands

Datastore commands

See also:

- **connect datastore**
- **define external target datastore**
- **disconnect datastore**
- **list database schemas**
- **list database tables**
- **list databases**
- **list datastores**
- **list datatypes**
- **list encodings**
- **list journal control fields**
- **select datastore**
- **show datastore**

Parent topic:CHCCLP Commands

connect datastore

Use this command to connect to a datastore and optionally indicate whether the datastore will be used as a source or target for subsequent commands. If your Access Server administrator has configured the datastore for multiuser mode, you can only connect to the datastore once from Management Console, the command-line interface, or API, using the same login credentials. Use the select datastore command to change the context between source and target after you have connected.

Syntax diagram

Syntax 1: Connects a datastore .arrow, .syntaxarrow { fill: none; stroke: black; } .arrowheadStartEnd, .arrowheadRepSep, .arrowheadRepSepReturn { stroke: black; fill: black; } .arrowheadSeq, .arrowheadStartChoice, .arrowheadAfterChoice, .arrowheadStartRepGroup, .arrowheadEndRepGroup, .arrowheadRev { stroke: none; fill: none; } rect { fill: none; stroke: none; } rect.fragref,rect.syntaxfragref { fill: none; stroke: black; } text { fill: #000000; fill-opacity: 1; font-family: IBM Plex Sans,Arial Unicode MS,Arial,Helvetica; font-style: normal; font-weight: normal; font-size: 8pt; stroke: #000000; stroke-width: 0.1; } text.var, text.syntaxvar {font-style:italic;} connect datastorenamedatastoreusernamenamepasswordvaluecontext sourcetarget

Syntax 2: Connects to a datastore with full login credentials. The syntax is applicable when command line processor is running with an embedded Access Server. Once a connection is established, subsequent connections to the same hostname and port must use the same name within a CHCCLP session. .arrow, .syntaxarrow { fill: none; stroke: black; } .arrowheadStartEnd, .arrowheadRepSep, .arrowheadRepSepReturn { stroke: black; fill: black; } .arrowheadSeq, .arrowheadStartChoice, .arrowheadAfterChoice, .arrowheadStartRepGroup, .arrowheadEndRepGroup, .arrowheadRev { stroke: none; fill: none; } rect { fill: none; stroke: none; } rect.fragref,rect.syntaxfragref { fill: none; stroke: black; } text { fill: #000000; fill-opacity: 1; font-family: IBM Plex Sans,Arial Unicode MS,Arial,Helvetica; font-style: normal; font-weight: normal; font-size: 8pt; stroke: #000000; stroke-width: 0.1; } text.var, text.syntaxvar {font-style:italic;} connect datastorenamedatastore usernamenamepasswordvaluehostnamehostportnumbermultiusertrue/yesfalse/no contextsourcetarget

Syntax

Syntax 1: Connects a datastore
```
connect datastore name <datastore> [username <name>] [password <value>]
[context <type>];
```

Syntax 2: Connects to a datastore with full login credentials. The syntax is applicable when command line processor is running with an embedded Access Server. Once a connection is established, subsequent connections to the same hostname and port must use the same name within a CHCCLP session.
```
connect
datastore name <datastore> username <name> password <value>
hosname <host> port <number> [multiuser <lock>][context <type>];
```

Parameters

Syntax 1: Connects a datastore

- name <datastore>
- Specifies the name of the datastore that you want to connect to.

- [username <name>]
- Specifies the datastore login username. This value is required when datastore connection credentials are not set in Access Manager.

- [password <value>]
- Specifies the datastore login password. This value is required when datastore connection credentials are not set in Access Manager.

- [context <type>]
- Identifies that the datastore will be set to the source or target context. If the context parameter is omitted, the datastore is set to both source and target, depending on the capabilities of the datastore. The context parameter can be changed at any point without requiring a new connection to the datastore. Valid values:
  - **source**—Sets the datastore to be a source datastore.
  - **target**—Sets the datastore to be a target datastore.

  Default value: source and target

Syntax 2: Connects to a datastore with full login credentials. The syntax is applicable when command line processor is running with an embedded Access Server. Once a connection is established, subsequent connections to the same hostname and port must use the same name within a CHCCLP session.

- name <datastore>
- Specifies the name of the datastore that you want to connect to.

- username <name>
- Specifies the datastore login username.

- password <value>
- Specifies the datastore login password.

- hostname <host>
- Specify the host name where datastore is installed.

- port <number>
- Specifies the port number used to connect to datastore.Valid range: 1 to 65535

- [multiuser <lock>]
- Specifies that subscriptions must be locked before editing.Valid values:
  - **true/yes**—Specifies that subscriptions must be locked before editing.
  - **false/no**—Specifies that subscriptions do not have to be locked before editing.

  Default value: source and target

- [context <type>]
- Identifies that the datastore will be set to the source or target context. If the context parameter is omitted, the datastore is set to both source and target, depending on the capabilities of the datastore. The context parameter can be changed at any point without requiring a new connection to the datastore. Valid values:
  - **source**—Sets the datastore to be a source datastore.
  - **target**—Sets the datastore to be a target datastore.

  Default value: source and target

Required role

Any

Result

This command returns a value of 0 if the command was successful and a negative value if the command fails. When verbose mode is enabled, the command prints success message CDC1201. **Sample**

Connects and disconnects from a datastore. The context will default to both source and target.

```
connect server username user1 password password1;
connect datastore name DS1;
disconnect datastore;
disconnect server;
exit;
```

Connects and disconnects from a datastore with login credentials, and specifies that the datastore will be used as the source datastore for subsequent commands. To view the current datastore context, use the "show context" command.

```
connect server username user1 password password1;
connect datastore name DS1 username dsuser1 password dspass1 context source;
show context;
disconnect datastore name DS1;
disconnect server;
exit;
```

Connects and disconnects from a datastore with full login credentials, and specifies that the datastore will be used as the source datastore for subsequent commands. To view the current datastore context, use the "show context" command. The syntax is applicable for datastore connection when the command line processor is running with an embedded Access Server.

```
connect datastore name DS1 hostname ds1machine port 11211 username dbuser
password dbpass context source;
show context;
disconnect datastore context source;
exit;
```

Parent topic: Datastore commands

Related reference

- select datastore

define external target datastore

Use this command to define an external target datastore for the session.

Syntax diagram

.arrow, .syntaxarrow { fill: none; stroke: black; } .arrowheadStartEnd, .arrowheadRepSep, .arrowheadRepSepReturn { stroke: black; fill: black; } .arrowheadSeq, .arrowheadStartChoice, .arrowheadAfterChoice, .arrowheadStartRepGroup, .arrowheadEndRepGroup, .arrowheadRev { stroke: none; fill: none; } rect { fill: none; stroke: none; } rect.fragref,rect.syntaxfragref { fill: none; stroke: black; } text { fill: #000000; fill-opacity: 1; font-family: IBM Plex Sans,Arial Unicode MS,Arial,Helvetica; font-style: normal; font-weight: normal; font-size: 8pt; stroke: #000000; stroke-width: 0.1; } text.var, text.syntaxvar {font-style:italic;} define external target datastorenamedatastoredbPlatformDB2_iDB2_z Java hostnamehostportnumberusernameuserpasswordvalue

Syntax

```
define external target datastore name <datastore> dbPlatform <platform> hostname

<host> port <number> [username <user>] [password <value>];
```

Parameters

- **name <datastore>**
  - Specifies the name of the external datastore that you want to define.
- **dbPlatform <platform>**
  - Specifies the operating system for the external target datastore to which you want to connect.Valid values:
    - DB2_i
    - DB2_z
    - Java
- **hostname <host>**
  - Specifies the host name or the IP address of the external target datastore to which you want to connect.
- **port <number>**
  - Specifies the port number of the external target datastore to which you want to connect.
- **[username <user>]**
  - Specifies the name of the database user that owns the metadata.
- **[password <value>]**
  - Specifies the password to access the metadata database.

Required role

- Administrator
- System administrator

Result

This command returns a value of 0 if the command was successful and a negative value if the command fails. When verbose mode is enabled, the command prints success message CDC1201.

Sample

Defines an external datastore.

```
connect server username user1 password password1;

define external target datastore name EXTDS1 dbPlatform Java hostname host1 port 11001 username dbuser1

password dbpassword1;

disconnect server;
```

```
exit;
```

Parent topic: Datastore commands

disconnect datastore

Use this command to disconnect from a datastore.

Syntax diagram

.arrow, .syntaxarrow { fill: none; stroke: black; } .arrowheadStartEnd, .arrowheadRepSep, .arrowheadRepSepReturn { stroke: black; fill: black; } .arrowheadSeq, .arrowheadStartChoice, .arrowheadAfterChoice, .arrowheadStartRepGroup, .arrowheadEndRepGroup, .arrowheadRev { stroke: none; fill: none; } rect { fill: none; stroke: none; } rect.fragref,rect.syntaxfragref { fill: none; stroke: black; } text { fill: #000000; fill-opacity: 1; font-family: IBM Plex Sans,Arial Unicode MS,Arial,Helvetica; font-style: normal; font-weight: normal; font-size: 8pt; stroke: #000000; stroke-width: 0.1; } text.var, text.syntaxvar {font-style:italic;} disconnect datastorenamedatastore

Syntax

```
disconnect datastore [name <datastore>];
```

Parameters

- [name <datastore>]

- Specifies the name of the datastore that will be disconnected. If a value is not specified, you will be disconnected from the datastore that is currently set to source and target context.

Required role

Any

Result

This command returns a value of 0 if the command was successful and a negative value if the command fails. When verbose mode is enabled, the command prints success message CDC1202.

Sample

Connects and disconnects from the datastore currently set to source and target context.

```
connect server username user1 password password1;

connect datastore name DS1;

disconnect datastore;

disconnect server;

exit;
```

Parent topic: Datastore commands

list database schemas

Use this command to list the database schemas.

Syntax diagram

.arrow, .syntaxarrow { fill: none; stroke: black; } .arrowheadStartEnd, .arrowheadRepSep, .arrowheadRepSepReturn { stroke: black; fill: black; } .arrowheadSeq, .arrowheadStartChoice, .arrowheadAfterChoice, .arrowheadStartRepGroup, .arrowheadEndRepGroup, .arrowheadRev { stroke: none; fill: none; } rect { fill: none; stroke: none; } rect.fragref,rect.syntaxfragref { fill: none; stroke: black; } text { fill: #000000; fill-opacity: 1; font-family: IBM Plex Sans,Arial Unicode MS,Arial,Helvetica; font-style: normal; font-weight: normal; font-size: 8pt; stroke: #000000; stroke-width: 0.1; } text.var, text.syntaxvar {font-style:italic;} list database schemasnamedatastoredatabasename

Syntax

```
list database schemas [name <datastore>] [database <name>];
```

Parameters

- [name <datastore>]

- Specifies the name of the datastore.If a name is not provided, the datastore that is currently set to source and target context will be used.

- [database <name>]

- Specifies a filter for the name of the database when required by the datastore.

Required role

- Administrator
- System administrator

Result

A ResultStringTable object containing the list of schemas.

Sample

List the database schemas.

```
connect server username user1 password password1;
connect datastore name DS1;
list database schemas;
disconnect server;
exit;
```

Parent topic: Datastore commands

list database tables

Use this command to list the database tables of the specified schema.

Syntax diagram

.arrow, .syntaxarrow { fill: none; stroke: black; } .arrowheadStartEnd, .arrowheadRepSep, .arrowheadRepSepReturn { stroke: black; fill: black; } .arrowheadSeq, .arrowheadStartChoice, .arrowheadAfterChoice, .arrowheadStartRepGroup, .arrowheadEndRepGroup, .arrowheadRev { stroke: none; fill: none; } rect { fill: none; stroke: none; } rect.fragref,rect.syntaxfragref { fill: none; stroke: black; } text { fill: #000000; fill-opacity: 1; font-family: IBM Plex Sans,Arial Unicode MS,Arial,Helvetica; font-style: normal; font-weight: normal; font-size: 8pt; stroke: #000000; stroke-width: 0.1; } text.var, text.syntaxvar {font-style:italic;} list database tablesnamedatastoredatabasenameschemaschema

Syntax

```
list database tables [name <datastore>] [database <name>] schema <schema>;
```

Parameters

- [name <datastore>]

- Specifies the name of the datastore.If a name is not provided, the datastore that is currently set to source and target context will be used.

- [database <name>]

- Specifies a filter for the name of the database when required by the datastore.

- schema <schema>

- Specifies a filter for the name of the schema.

Required role

- Administrator
- System administrator

Result

A ResultStringTable object containing the list of tables for a schema. **Sample**

List database tables.

```
connect server username user1 password password1;

connect datastore name DS1;

list database tables schema USER1;

disconnect server;

exit;
```

Parent topic: Datastore commands

list databases

Use this command to list the databases of the datastore.

Syntax diagram

.arrow, .syntaxarrow { fill: none; stroke: black; } .arrowheadStartEnd, .arrowheadRepSep, .arrowheadRepSepReturn { stroke: black; fill: black; } .arrowheadSeq, .arrowheadStartChoice, .arrowheadAfterChoice, .arrowheadStartRepGroup, .arrowheadEndRepGroup, .arrowheadRev { stroke: none; fill: none; } rect { fill: none; stroke: none; } rect.fragref,rect.syntaxfragref { fill: none; stroke: black; } text { fill: #000000; fill-opacity: 1; font-family: IBM Plex Sans,Arial Unicode MS,Arial,Helvetica; font-style: normal; font-weight: normal; font-size: 8pt; stroke: #000000; stroke-width: 0.1; } text.var, text.syntaxvar {font-style:italic;} list databasesnamedatastore

Syntax

```
list databases [name <datastore>];
```

Parameters

- [name <datastore>]

- Specifies the name of the datastore.If a name is not provided, the datastore that is currently set to source and target context will be used.

Required role

- Administrator
- System administrator

Result

A ResultStringTable object containing the list of databases. **Sample**

List the databases.

```
connect server username user1 password password1;

connect datastore name DS1;

list databases;

disconnect server;

exit;
```

Parent topic: Datastore commands

list datastores

Use this command to list the datastores that are available to the user who connected to Access Server.

Syntax diagram

.arrow, .syntaxarrow { fill: none; stroke: black; } .arrowheadStartEnd, .arrowheadRepSep, .arrowheadRepSepReturn { stroke: black; fill: black; } .arrowheadSeq, .arrowheadStartChoice, .arrowheadAfterChoice, .arrowheadStartRepGroup, .arrowheadEndRepGroup, .arrowheadRev { stroke: none; fill: none; } rect { fill: none; stroke: none; } rect.fragref,rect.syntaxfragref { fill: none; stroke: black; } text { fill: #000000; fill-opacity: 1; font-family: IBM Plex Sans,Arial Unicode MS,Arial,Helvetica; font-style: normal; font-weight: normal; font-size: 8pt; stroke: #000000; stroke-width: 0.1; } text.var, text.syntaxvar {font-style:italic;} list datastores

Syntax

```
list datastores;
```

Parameters

None.

Required role

Any

Result

A ResultStringTable object containing the list of datastores that the user can access.

Sample

Connects to Access Server and lists the datastores that the user has access to.

```
connect server username user1 password password1;

list datastores;

disconnect server;

exit;
```

Parent topic: Datastore commands

list datatypes

Use this command to list the datatypes available with the datastore.

Syntax diagram

.arrow, .syntaxarrow { fill: none; stroke: black; } .arrowheadStartEnd, .arrowheadRepSep, .arrowheadRepSepReturn { stroke: black; fill: black; } .arrowheadSeq, .arrowheadStartChoice, .arrowheadAfterChoice, .arrowheadStartRepGroup, .arrowheadEndRepGroup, .arrowheadRev { stroke: none; fill: none; } rect { fill: none; stroke: none; } rect.fragref,rect.syntaxfragref { fill: none; stroke: black; } text { fill: #000000; fill-opacity: 1; font-family: IBM Plex Sans,Arial Unicode MS,Arial,Helvetica; font-style: normal; font-weight: normal; font-size: 8pt; stroke: #000000; stroke-width: 0.1; } text.var, text.syntaxvar {font-style:italic;} list datatypesnamedatastore

Syntax

```
list datatypes [name <datastore>];
```

Parameters

- [name <datastore>]

- Specifies the name of the datastore.If a name is not provided, the datastore that is currently set to source context will be used.

Required role

- Administrator
- System administrator

Result

A ResultStringTable object containing the list of datatypes available for the datastore, indicating which can be used in derived columns. **Sample**

Lists the datatypes available with the datastore.

```
connect server username user1 password password1;

connect datastore name DS1;

list datatypes;

disconnect server;

exit;
```

Parent topic: Datastore commands

list encodings

Use this command to list the character encodings supported by the datastore.

Syntax diagram

.arrow, .syntaxarrow { fill: none; stroke: black; } .arrowheadStartEnd, .arrowheadRepSep, .arrowheadRepSepReturn { stroke: black; fill: black; } .arrowheadSeq, .arrowheadStartChoice, .arrowheadAfterChoice, .arrowheadStartRepGroup, .arrowheadEndRepGroup, .arrowheadRev { stroke: none; fill: none; } rect { fill: none; stroke: none; } rect.fragref,rect.syntaxfragref { fill: none; stroke: black; } text { fill: #000000; fill-opacity: 1; font-family: IBM Plex Sans,Arial Unicode MS,Arial,Helvetica; font-style: normal; font-weight: normal; font-size: 8pt; stroke: #000000; stroke-width: 0.1; } text.var, text.syntaxvar {font-style:italic;} list encodingsnamedatastore

Syntax

```
list encodings [name <datastore>];
```

Parameters

- [name <datastore>]

- Specifies the name of the datastore.If a name is not provided, the datastore that is currently set to source and target context will be used.

Required role

- Administrator
- System administrator

Result

A ResultStringTable object containing the list of encoding available for the datastore.

Sample

Lists the character encodings supported by the datastore.

```
connect server username user1 password password1;

connect datastore name DS1;

list encodings;

disconnect server;

exit;
```

Parent topic: Datastore commands

list journal control fields

Use this command to list the journal control fields.

Syntax diagram

.arrow, .syntaxarrow { fill: none; stroke: black; } .arrowheadStartEnd, .arrowheadRepSep, .arrowheadRepSepReturn { stroke: black; fill: black; } .arrowheadSeq, .arrowheadStartChoice, .arrowheadAfterChoice, .arrowheadStartRepGroup, .arrowheadEndRepGroup, .arrowheadRev { stroke: none; fill: none; } rect { fill: none; stroke: none; } rect.fragref,rect.syntaxfragref { fill: none; stroke: black; } text { fill: #000000; fill-opacity: 1; font-family: IBM Plex Sans,Arial Unicode MS,Arial,Helvetica; font-style: normal; font-weight: normal; font-size: 8pt; stroke: #000000; stroke-width: 0.1; } text.var, text.syntaxvar {font-style:italic;} list journal control fieldsnamedatastore

Syntax

```
list journal control fields [name <datastore>];
```

Parameters

- [name <datastore>]

- Specifies the name of the datastore.If a name is not provided, the datastore that is currently set to source context will be used.

Required role

- Administrator
- System administrator

Result

A ResultStringTable object containing the list of journal control fields for mapping columns.

Sample

Lists the journal control fields.

```
connect server username user1 password password1;

connect datastore name DS1;

list journal control fields;

disconnect server;

exit;
```

Parent topic: Datastore commands

select datastore

Use this command to select the datastore to context. The datastore will be used for subsequent commands. Set the name to   to clear the current source and target datastore context. The source and target datastore contexts must be specified in order to create a subscription. Use the show context or list datastores commands to view which datastores are the source and target.

Syntax diagram

.arrow, .syntaxarrow { fill: none; stroke: black; } .arrowheadStartEnd, .arrowheadRepSep, .arrowheadRepSepReturn { stroke: black; fill: black; } .arrowheadSeq, .arrowheadStartChoice, .arrowheadAfterChoice, .arrowheadStartRepGroup, .arrowheadEndRepGroup, .arrowheadRev { stroke: none; fill: none; } rect { fill: none; stroke: none; } rect.fragref,rect.syntaxfragref { fill: none; stroke: black; } text { fill: #000000; fill-opacity: 1; font-family: IBM Plex Sans,Arial Unicode MS,Arial,Helvetica; font-style: normal; font-weight: normal; font-size: 8pt; stroke: #000000; stroke-width: 0.1; } text.var, text.syntaxvar {font-style:italic;} select datastorename""namecontextsourcetarget

Syntax

```
select datastore name <name> [context <type>];
```

Parameters

- name <name>

- Specifies the name of the datastore to be selected for use with subsequent commands.Set the name to to clear the current source and target datastore context.

- [context <type>]

- Identifies that the datastore will be set to the source or target context. If the context parameter is omitted, the datastore is set to both source and target, depending on the capabilities of the datastore. The context can be changed at any point without requiring a new connection to the datastore.
  Valid values:
  - **source**—Sets the datastore to be a source datastore.
  - **target**—Sets the datastore to be a target datastore.

  Default value: source and target

Required role

Any

Result

This command returns a value of 0 if the command was successful and a negative value if the command fails. When verbose mode is enabled, the command prints success message CDC1203.

Sample

Connects to two datastores, specifying the context during connection, and creates a new subscription between them.

```
connect server username user1 password password1;
connect datastore name DS1 context source;
connect datastore name DS2 context target;
add subscription name SUB1;
disconnect datastore name DS1;
disconnect datastore name DS2;
disconnect server;
```

```
exit;
```

Connects to two datastores, sets one to the source and one to the target after connection, and creates a new subscription between them.

```
connect server username user1 password password1;

connect datastore name DS1;

connect datastore name DS2;

select datastore name DS1 context source;

select datastore name DS2 context target;

add subscription name SUB1;

disconnect datastore name DS1;

disconnect datastore name DS2;

disconnect server;

exit;
```

Parent topic: Datastore commands

Related reference

- show context
- list datastores

show datastore

Use this command to show the properties of a datastore.

Syntax diagram

.arrow, .syntaxarrow { fill: none; stroke: black; } .arrowheadStartEnd, .arrowheadRepSep, .arrowheadRepSepReturn { stroke: black; fill: black; } .arrowheadSeq, .arrowheadStartChoice, .arrowheadAfterChoice, .arrowheadStartRepGroup, .arrowheadEndRepGroup, .arrowheadRev { stroke: none; fill: none; } rect { fill: none; stroke: none; } rect.fragref,rect.syntaxfragref { fill: none; stroke: black; } text { fill: #000000; fill-opacity: 1; font-family: IBM Plex Sans,Arial Unicode MS,Arial,Helvetica; font-style: normal; font-weight: normal; font-size: 8pt; stroke: #000000; stroke-width: 0.1; } text.var, text.syntaxvar {font-style:italic;} show datastorenamedatastore

Syntax

```
show datastore [name <datastore>];
```

Parameters

- [name <datastore>]

- Specifies the name of the datastore whose properties will be displayed.If a name is not provided, the datastore that is currently set to source and target context will be used.

Required role

- Administrator
- System administrator

Result

A ResultStringKeyValue table containing the properties of the datastore.

```
PROPERTY
         VALUE
-------------------------- --------------------------
Name:                      DS1
Description:               DB2 LUW on Windows VM
Host Name:                 myhost.mycompany.com
Port:                      10005
Version:                   V10R2M0T0BCDC_WTPGMVMN_22
Platform:                  Java VM
Database:                  DB2 for LUW
Type:                      DUAL
Table Identification:      Schema (owner), table name
```

Sample

Shows the properties of two datastore. The first show datastore will use the current context to identify and display properties for DS2. The second show datastore overrides the context and displays the properties for DS1.

```
connect server username user1
password password1;
connect datastore name DS1;
connect datastore name DS2;
show datastore;
show datastore name DS1;
disconnect datastore name DS1;
disconnect datastore name DS2;
disconnect server;

exit;
```

Parent topic:Datastore commands

Tables commands

- **add replication table**
- **list index columns**
- **list replication tables**
- **list table columns**
- **list table indexes**
- **readd replication table**
- **remove replication table**

Parent topic:CHCCLP Commands

add replication table

Use this command to add a table to the datastore for use as a source table for replication in a subscription.Adding tables to the datastore is optional. They are automatically added when the add table mapping command is executed.

Syntax diagram

.arrow, .syntaxarrow { fill: none; stroke: black; } .arrowheadStartEnd, .arrowheadRepSep, .arrowheadRepSepReturn { stroke: black; fill: black; } .arrowheadSeq, .arrowheadStartChoice, .arrowheadAfterChoice, .arrowheadStartRepGroup, .arrowheadEndRepGroup, .arrowheadRev { stroke: none; fill: none; } rect { fill: none; stroke: none; } rect.fragref,rect.syntaxfragref { fill: none; stroke: black; } text { fill: #000000; fill-opacity: 1; font-family: IBM Plex Sans,Arial Unicode MS,Arial,Helvetica; font-style: normal; font-weight: normal; font-size: 8pt; stroke: #000000; stroke-width: 0.1; } text.var, text.syntaxvar {font-style:italic;} add replication tablenamedatastoredatabasenameschemaschematable table

Syntax

```
add replication table [name <datastore>] [database <name>] schema <schema>

table <table>;
```

Parameters

- [name <datastore>]

- Specifies the name of the datastore.If a name is not provided, the datastore that is currently set to source context will be used.

- [database <name>]

- Specifies the name of the database containing the table to be added.

- schema <schema>

- Specifies the name of the schema.

- table <table>

- Specifies the name of the table.

Required role

- Administrator
- System administrator

Result

This command returns a value of 0 if the command was successful and a negative value if the command fails. When verbose mode is enabled, the command prints success messaged CDC1303. **Sample**

Makes two tables available for replication.

```
connect server username user1 password password1;

connect datastore name DS1 context source;

connect datastore name DS2 context target;

add replication table schema USER1 table TABLE1;

add replication table schema USER1 table TABLE2;

list replication tables schema USER1;

disconnect server;

exit;
```

Parent topic: Tables commands

list index columns

Use this command to view the columns and their sorting orders of an index.

Syntax diagram

.arrow, .syntaxarrow { fill: none; stroke: black; } .arrowheadStartEnd, .arrowheadRepSep, .arrowheadRepSepReturn { stroke: black; fill: black; } .arrowheadSeq, .arrowheadStartChoice, .arrowheadAfterChoice, .arrowheadStartRepGroup, .arrowheadEndRepGroup, .arrowheadRev { stroke: none; fill: none; } rect { fill: none; stroke: none; } rect.fragref,rect.syntaxfragref { fill: none; stroke: black; } text { fill: #000000; fill-opacity: 1; font-family: IBM Plex Sans,Arial Unicode MS,Arial,Helvetica; font-style: normal; font-weight: normal; font-size: 8pt; stroke: #000000; stroke-width: 0.1; } text.var, text.syntaxvar {font-style:italic;} list index columns namedatastoredatabasenameschemaschematable tableindexindex

Syntax

```
list index columns [name <datastore>] [database <name>] schema <schema>

table <table> index <index>;
```

Parameters

- [name <datastore>]

- Specifies the name of the datastore.If a name is not provided, the datastore that is currently set to source and target context will be used.

- [database <name>]

- Specifies the name of the database.

- schema <schema>

- Specifies the name of the schema.

- table <table>

- Specifies the name of the table.

- index <index>

- Specifies the full name of the index using the format schema.indexname.

Required role

- Operator
- Administrator
- System administrator

Result

A ResultStringTable containing the columns in the index.Columns in index USER1.P_TABLE1 for table USER1.TABLE1.

```
COLUMN NAME      SORT ORDER

---------------- ----------------

CUSTOMER_NUM     Ascending
```

Sample

Lists the columns in the index. Use the list table indexes command to view the available indexes.

```
connect server username user1 password password1;

connect datastore name DS1 context source;

connect datastore name DS2 context target;

list table indexes schema USER1 table TABLE1;

list index columns schema USER1 table TABLE1 index USER1.P_TABLE1;
```

```
disconnect server;

exit;
```

Parent topic:Tables commands

list replication tables

Use this command to list the tables in your source datastore that are in subscriptions or available for use as source tables in a subscription.

Syntax diagram

.arrow, .syntaxarrow { fill: none; stroke: black; } .arrowheadStartEnd, .arrowheadRepSep, .arrowheadRepSepReturn { stroke: black; fill: black; } .arrowheadSeq, .arrowheadStartChoice, .arrowheadAfterChoice, .arrowheadStartRepGroup, .arrowheadEndRepGroup, .arrowheadRev { stroke: none; fill: none; } rect { fill: none; stroke: none; } rect.fragref,rect.syntaxfragref { fill: none; stroke: black; } text { fill: #000000; fill-opacity: 1; font-family: IBM Plex Sans,Arial Unicode MS,Arial,Helvetica; font-style: normal; font-weight: normal; font-size: 8pt; stroke: #000000; stroke-width: 0.1; } text.var, text.syntaxvar {font-style:italic;} list replication tablesnamedatastoredatabasenameschemaschema

Syntax

```
list replication tables [name <datastore>] [database  <name>] [schema <schema>];
```

Parameters

- [name <datastore>]

- Specifies the name of the datastore for which all tables will be listed. If a name is not provided, the datastore that is currently set to source context will be used.

- [database <name>]

- Specifies a filter for the name of the database.If database is omitted, the command lists all tables.

- [schema <schema>]

- Specifies a filter for the name of the schema. If schema is omitted, the command lists all tables.

Result

A ResultStringTable containing the tables in the datastore that have been made available for replication.Replication tables for datastore DS1.

```
SCHEMA           REPLICATION TABLE
---------------- -------------------
USER1            TABLE1
USER1            TABLE2
USER1            TABLE3
USER1            TABLE4
USER1            TABLE5
```

Sample

Lists the tables available for replication in schema USER1.

```
connect server username user1 password password1;
connect datastore name DS1 context source;
connect datastore name DS2 context target;
list replication tables schema USER1;
disconnect server;
exit;
```

Parent topic: Tables commands

list table columns

Use this command to display the structure of a database table.

Syntax diagram

.arrow, .syntaxarrow { fill: none; stroke: black; } .arrowheadStartEnd, .arrowheadRepSep, .arrowheadRepSepReturn { stroke: black; fill: black; } .arrowheadSeq, .arrowheadStartChoice, .arrowheadAfterChoice, .arrowheadStartRepGroup, .arrowheadEndRepGroup, .arrowheadRev { stroke: none; fill: none; } rect { fill: none; stroke: none; } rect.fragref,rect.syntaxfragref { fill: none; stroke: black; } text { fill: #000000; fill-opacity: 1; font-family: IBM Plex Sans,Arial Unicode MS,Arial,Helvetica; font-style: normal; font-weight: normal; font-size: 8pt; stroke: #000000; stroke-width: 0.1; } text.var, text.syntaxvar {font-style:italic;} list table columns name datastoredatabase databaseschema schema table table

Syntax

```
list table columns [name <datastore>] [database <database>] schema <schema>

table <table>;
```

Parameters

- [name <datastore>]

- Specifies the name of the datastore.If a name is not provided, the datastore that is currently set to source and target context will be used.

- [database <database>]

- Specifies the name of the database.

- schema <schema>

- Specifies the name of the schema.

- table <table>

- Specifies the name of the table.

Result

A ResultStringTable containing the columns in the table.Columns for table USER1.TABLE1.

```
COLUMN NAME    DATA TYPE   LENGTH   PRECISION  SCALE    KEY       NULLABLE

-------------- ----------- -------- ---------- -------- --------- ----------

CUSTOMER_NUM   NUMBER      10       10         0        Yes       No

ZIP            VARCHAR2    10       10         0        No        No

NAME           VARCHAR2    30       30         0        No        Yes

CREDIT_LIMIT   NUMBER      10       10         0        No        Yes

ADDR_LN1       VARCHAR2    30       30         0        No        Yes

ADDR_LN2       VARCHAR2    30       30         0        No        Yes

CITY           VARCHAR2    25       25         0        No        Yes

STATE          CHAR        2        2          0        No        Yes

PHONE          CHAR        12       12         0        No        Yes

FAX            CHAR        12       12         0        No        Yes

EMAIL          VARCHAR2    40       40         0        No        Yes
```

Sample

Lists the columns in a table.

```
connect server username user1 password password1;

connect datastore name DS1;
```

```
list table columns schema USER1 table TABLE1;
disconnect server;
exit;
```

Parent topic: Tables commands

list table indexes

Use this command to display the indexes available for a database table.

Syntax diagram

.arrow, .syntaxarrow { fill: none; stroke: black; } .arrowheadStartEnd, .arrowheadRepSep, .arrowheadRepSepReturn { stroke: black; fill: black; } .arrowheadSeq, .arrowheadStartChoice, .arrowheadAfterChoice, .arrowheadStartRepGroup, .arrowheadEndRepGroup, .arrowheadRev { stroke: none; fill: none; } rect { fill: none; stroke: none; } rect.fragref,rect.syntaxfragref { fill: none; stroke: black; } text { fill: #000000; fill-opacity: 1; font-family: IBM Plex Sans,Arial Unicode MS,Arial,Helvetica; font-style: normal; font-weight: normal; font-size: 8pt; stroke: #000000; stroke-width: 0.1; } text.var, text.syntaxvar {font-style:italic;} list table indexesnamedatastoredatabasedatabaseschemaschematable table

Syntax

```
list table indexes [name <datastore>] [database <database>] schema <schema>

table <table>;
```

Parameters

- [name <datastore>]

- Specifies the name of the datastore.If a name is not provided, the datastore that is currently set to source and target context will be used.

- [database <database>]

- Specifies the name of the database.

- schema <schema>

- Specifies the name of the schema.

- table <table>;

- Specifies the name of the table.

Required role

- Operator
- Administrator
- System administrator

Result

A ResultStringTable containing the indexes available for a table.Indexes for table USER1.TABLE1.

```
INDEX NAME       PRIMARY INDEX    UNIQUE INDEX

---------------- ---------------- ----------------

USER1.P_TABLE1   Yes              Yes
```

Sample

Lists the indexes for a table.

```
connect server username user1 password password1;

connect datastore name DS1;

list table indexes schema USER1 table TABLE1;

disconnect server;

exit;
```

Parent topic: Tables commands

readd replication table

Use this command to re-add a table to the datastore so that the datastore metadata matches the database structure.After executing the readd replication table command, any subscriptions containing the table must be described using the describe subscription command. Any table mappings that use the table as a source must be reassigned using the reassign table mapping command.

Syntax diagram

.arrow, .syntaxarrow { fill: none; stroke: black; } .arrowheadStartEnd, .arrowheadRepSep, .arrowheadRepSepReturn { stroke: black; fill: black; } .arrowheadSeq, .arrowheadStartChoice, .arrowheadAfterChoice, .arrowheadStartRepGroup, .arrowheadEndRepGroup, .arrowheadRev { stroke: none; fill: none; } rect { fill: none; stroke: none; } rect.fragref,rect.syntaxfragref { fill: none; stroke: black; } text { fill: #000000; fill-opacity: 1; font-family: IBM Plex Sans,Arial Unicode MS,Arial,Helvetica; font-style: normal; font-weight: normal; font-size: 8pt; stroke: #000000; stroke-width: 0.1; } text.var, text.syntaxvar {font-style:italic;} readd replication tablenamedatastoredatabasenameschemaschema tabletable

Syntax

```
readd replication table [name <datastore>] [database <name>] schema <schema>

table <table>;
```

Parameters

- [name <datastore>]

- Specifies the name of the datastore.If a name is not provided, the datastore that is currently set to source context will be used.

- [database <name>]

- Specifies the database for the table.

- schema <schema>

- Specifies the schema for the table.

- table <table>

- Specifies the name of the table.

Required role

- Administrator
- System administrator

Result

This command returns a value of 0 if the command was successful and a negative value if the command fails. When verbose mode is enabled, the command prints success message CDC1306.

Sample

Updates the metadata definition of a database table.

```
connect server username user1 password password1;

connect datastore name DS1;

list table columns schema USER1 table TABLE1;

readd replication table schema USER1 table TABLE1;

list table columns schema USER1 table TABLE1;

disconnect server;

exit;
```

Parent topic: Tables commands

remove replication table

Use this command to remove a table from replication. A table cannot be removed if it is currently used as a source table in a subscription.

Syntax diagram

.arrow, .syntaxarrow { fill: none; stroke: black; } .arrowheadStartEnd, .arrowheadRepSep, .arrowheadRepSepReturn { stroke: black; fill: black; } .arrowheadSeq, .arrowheadStartChoice, .arrowheadAfterChoice, .arrowheadStartRepGroup, .arrowheadEndRepGroup, .arrowheadRev { stroke: none; fill: none; } rect { fill: none; stroke: none; } rect.fragref,rect.syntaxfragref { fill: none; stroke: black; } text { fill: #000000; fill-opacity: 1; font-family: IBM Plex Sans,Arial Unicode MS,Arial,Helvetica; font-style: normal; font-weight: normal; font-size: 8pt; stroke: #000000; stroke-width: 0.1; } text.var, text.syntaxvar {font-style:italic;} remove replication tablenamedatastoredatabasenameschemaschema tabletable

Syntax

```
remove replication table [name <datastore>] [database <name>] schema <schema>
table <table>;
```

Parameters

- [name <datastore>]
- Specifies the name of the datastore from which the table will be removed. If a name is not provided, the datastore that is currently set to source context will be used.

- [database <name>]
- Specifies the name of the database.

- schema <schema>
- Specifies the name of the schema.

- table <table>
- Specifies the name of the table to be removed.

Required role

- Administrator
- System administrator

Result

This command returns a value of 0 if the command was successful and a negative value if the command fails. When verbose mode is enabled, the command prints success message CDC1307. **Sample**

Removes a table from replication.

```
connect server username user1 password password1;
connect datastore name DS1;
list table columns schema USER1 table TABLE1;
remove replication table schema USER1 table TABLE1;
disconnect server;
exit;
```

Parent topic: Tables commands

Subscription commands

- **add subscription**
- **delete subscription**
- **describe subscription**
- **export subscription**
- **import subscription**
- **list subscriptions**
- **lock subscription**
- **modify latency thresholds**
- **modify subscription**
- **modify subscription datastage properties**
- **modify subscription user exit**
- **promote subscription**
- **select subscription**
- **show latency thresholds**
- **show subscription**
- **show subscription datastage properties**
- **show subscription user exit**
- **unlock subscription**

Parent topic:CHCCLP Commands

add subscription

Use this command to create a new subscription and set it to the current context.The source and target datastore contexts identify the datastores for the subscription. To set the datastore context, use the select datastore command. To view the current datastore context, use the show context command.

Syntax diagram

.arrow, .syntaxarrow { fill: none; stroke: black; } .arrowheadStartEnd, .arrowheadRepSep, .arrowheadRepSepReturn { stroke: black; fill: black; } .arrowheadSeq, .arrowheadStartChoice, .arrowheadAfterChoice, .arrowheadStartRepGroup, .arrowheadEndRepGroup, .arrowheadRev { stroke: none; fill: none; } rect { fill: none; stroke: none; } rect.fragref,rect.syntaxfragref { fill: none; stroke: black; } text { fill: #000000; fill-opacity: 1; font-family: IBM Plex Sans,Arial Unicode MS,Arial,Helvetica; font-style: normal; font-weight: normal; font-size: 8pt; stroke: #000000; stroke-width: 0.1; } text.var, text.syntaxvar {font-style:italic;} add subscription namesubscriptiondescriptiondescriptionsourceidsource transferableworktargetsourcetcphostnamehostfirewallportnumberpersistencyfalse/no true/yesmirrorContinueOnErrordefaultendcontinuerefreshContinueOnErrordefault endcontinuerelatedSubscriptions,value*""engineCommunicationtcpjms maintainHistorytargetsourceminimizeNetworkLoadfalse/notrue/yes

Syntax

```
add subscription name <subscription> [description <description>] [sourceid <source>]
[transferablework <transfer>] [tcphostname <host>] [firewallport <number>]
[persistency <state>] [mirrorContinueOnError <mirror>] [refreshContinueOnError
<refresh>] [relatedSubscriptions <value>] [engineCommunication <method>]
[maintainHistory <history>] [minimizeNetworkLoad <load>];
```

Parameters

- name <subscription>

- Specifies the name of the new subscription.The subscription name must be unique on the source datastore.

- [description <description>]

- Specifies the description of the subscription. Default value: empty string ().

- [sourceid <source>]

- Specifies the source identifier for the subscription on the target datastore.
- The identifier must be unique on the target.
- If no identifier is provided, the default value for the Source ID will be the first eight characters of the subscription name.

- [transferablework <transfer>]

- Selects the datastore to handle transferable work. Valid values:
  - **source**—InfoSphere® CDC will attempt to perform the encoding conversion workload on the server where your source datastore is installed.
  - **target**—InfoSphere CDC will attempt to perform the encoding conversion workload on the server where your target datastore is installed.

  Default value: target

- [tcphostname <host>]

- Specifies the TCP host name for the target datastore. If no TCP host name is provided, the host name is set to auto-select based on the target datastore.

- [firewallport <number>]
- Specifies the firewall port number used to connect to the target datastore through a firewall.

- [persistency <state>]
- Marks the subscription as persistent, if supported by the datastore. Valid values:
  - **true/yes**—Specifies that the subscription will be marked persistent. Either yes or true are valid values.
  - **false/no**—Specifies that the subscription will not be marked persistent. Either no or false are valid values.

  Default value: false/no

- [mirrorContinueOnError <mirror>]
- Indicates if mirroring should continue on error, where supported by the datastore.   Valid values:
  - **default**—Specifies that mirroring will end based on the datastore settings.
  - **end**—Overrides the datastore settings and specifies that mirroring will end after an apply error on the target database.
  - **continue**—Overrides the datastore settings and specifies that mirroring will continue after an apply error on the target database.

  Default value: default

- [refreshContinueOnError <refresh>]
- Indicates if refresh should continue on error, where supported by the datastore. Valid values:
  - **default**—Specifies that refresh will end based on the datastore settings.
  - **end**—Overrides the datastore settings and specifies that a refresh will end after an apply error on the target database.
  - **continue**—Overrides the datastore settings and specifies that a refresh will continue after an apply error on the target database.

  Default value: default

- [relatedSubscriptions <value>]
- Specifies the related subscriptions for recursion prevention. Use for none, * for all, or a comma-delimited list of subscription source ids.

- [engineCommunication <method>]
- Specifies the communication protocol between engines, where supported.Valid values:
  - **tcp**—Specifies that TCP/IP will be the communication protocol used.
  - **jms**—Specifies that JMS will be the communication protocol used.

  Default value: tcp

- [maintainHistory <history>]
- Specify whether the source datastore will replicate history tables or the target database will maintain history, when system or bitemporal tables are supported by the source and target datastores.Valid values:
  - **source**—Specifies that the source datastore will replicate history tables.
  - **target**—Specifies that the target datastore will maintain history.

  Default value: target

- **[minimizeNetworkLoad <load>]**
  - Specify whether to minimize the network load where possible, even if it may result in increased replication latency. This option is only available when both the source and target datastores support the functionality.Valid values:
    - **true/yes**—Specifies that LOB columns will only be sent when a change is detected.
    - **false/no**—Specifies that LOB columns will be sent regardless of a change occurring.

Required role

- Administrator
- System administrator

Result

This command returns a value of 0 if the command was successful and a negative value if the command fails. When verbose mode is enabled, the command prints success message CDC1301.

Sample

Creates a new subscription.

```
connect server username user1 password password1;
connect datastore name DS1 context source;
connect datastore name DS2 context target;
add subscription name SUB1;
disconnect server;
exit;
```

Parent topic: Subscription commands

delete subscription

Use this command to delete a subscription.

Syntax diagram

.arrow, .syntaxarrow { fill: none; stroke: black; } .arrowheadStartEnd, .arrowheadRepSep, .arrowheadRepSepReturn { stroke: black; fill: black; } .arrowheadSeq, .arrowheadStartChoice, .arrowheadAfterChoice, .arrowheadStartRepGroup, .arrowheadEndRepGroup, .arrowheadRev { stroke: none; fill: none; } rect { fill: none; stroke: none; } rect.fragref,rect.syntaxfragref { fill: none; stroke: black; } text { fill: #000000; fill-opacity: 1; font-family: IBM Plex Sans,Arial Unicode MS,Arial,Helvetica; font-style: normal; font-weight: normal; font-size: 8pt; stroke: #000000; stroke-width: 0.1; } text.var, text.syntaxvar {font-style:italic;} delete subscriptionnamesubscription

Syntax

```
delete subscription [name <subscription>];
```

Parameters

- [name <subscription>]

- Specifies the name of the subscription.If a name is not provided, the subscription that is currently identified as the context will be used. To view the current context, use the show context command.

Required role

- Administrator
- System administrator

Result

This command returns a value of 0 if the command was successful and a negative value if the command fails. When verbose mode is enabled, the command prints success message CDC1311. **Sample**

Deletes a subscription by name.

```
connect server username user1 password password1;
connect datastore name DS1 context source;
connect datastore name DS2 context target;
delete subscription name SUB1;
disconnect server;
exit;
```

Deletes a subscription based on the current context.

```
connect server username user1 password password1;
connect datastore name DS1 context source;
connect datastore name DS2 context target;
select subscription name SUB1;
delete subscription;
disconnect server;
exit;
```

Parent topic: Subscription commands

describe subscription

Use this command to start the describe process to notify the target datastore about changes to a subscription.

Syntax diagram

.arrow, .syntaxarrow { fill: none; stroke: black; } .arrowheadStartEnd, .arrowheadRepSep, .arrowheadRepSepReturn { stroke: black; fill: black; } .arrowheadSeq, .arrowheadStartChoice, .arrowheadAfterChoice, .arrowheadStartRepGroup, .arrowheadEndRepGroup, .arrowheadRev { stroke: none; fill: none; } rect { fill: none; stroke: none; } rect.fragref,rect.syntaxfragref { fill: none; stroke: black; } text { fill: #000000; fill-opacity: 1; font-family: IBM Plex Sans,Arial Unicode MS,Arial,Helvetica; font-style: normal; font-weight: normal; font-size: 8pt; stroke: #000000; stroke-width: 0.1; } text.var, text.syntaxvar {font-style:italic;} describe subscriptionnamesubscription

Syntax

```
describe subscription [name <subscription>];
```

Parameters

- [name <subscription>]

- Specifies the name of the subscription.If a name is not provided, the subscription that is currently identified as the context will be used. To view the current context, use the show context command.

Required role

- Administrator
- System administrator

Result

This command returns a value of 0 if the command was successful and a negative value if the command fails. When verbose mode is enabled, the command prints success message CDC1304.

Sample

Describes changes to a subscription to the target datastore.

```
connect server username user1
password password1;
connect datastore name DS1 context source;
connect datastore name DS2 context target;
describe subscription name SUB1;
disconnect server;
exit;
```

Parent topic: Subscription commands

export subscription

Use this command to export a subscription definition into an XML file. The XML file can be loaded using the import subscription command or using the import wizard in Management Console. Export requires additional connections to the source and target datastores and cannot run when multiuser configuration is enabled. Use Management Console to export when multiuser configuration is in use.

Syntax diagram

.arrow, .syntaxarrow { fill: none; stroke: black; } .arrowheadStartEnd, .arrowheadRepSep, .arrowheadRepSepReturn { stroke: black; fill: black; } .arrowheadSeq, .arrowheadStartChoice, .arrowheadAfterChoice, .arrowheadStartRepGroup, .arrowheadEndRepGroup, .arrowheadRev { stroke: none; fill: none; } rect { fill: none; stroke: none; } rect.fragref,rect.syntaxfragref { fill: none; stroke: black; } text { fill: #000000; fill-opacity: 1; font-family: IBM Plex Sans,Arial Unicode MS,Arial,Helvetica; font-style: normal; font-weight: normal; font-size: 8pt; stroke: #000000; stroke-width: 0.1; } text.var, text.syntaxvar {font-style:italic;} export subscriptionnamesubscriptionfilename file

Syntax

```
export subscription [name <subscription>] [filename <file>];
```

Parameters

- [name <subscription>]

- Specifies the name of the subscription.If a name is not provided, the subscription that is currently identified as the context will be used. To view the current context, use the show context command.

- [filename <file>]

- Specifies the fully qualified output XML file name for the subscription export. Use double quotation marks to surround the file name.
  .xml will be appended to the file name, if it does not end with the XML extension.
  If a file name is not provided, the output file name will default to subscription's name.

Required role

- Administrator
- System administrator

Result

This command returns a value of 0 if the command was successful and a negative value if the command fails. When verbose mode is enabled, the command prints success message CDC1326.

Sample

Selects a subscription and exports it to file.

```
connect server username user1 password password1;

connect datastore name DS1 context source;

connect datastore name DS2 context target;

select subscription name SUB1;

export subscription filename c:\\files\\sub1.xml;

disconnect server;

exit;
```

Parent topic: Subscription commands

import subscription

Use this command to import the subscription defined in the XML file to a new subscription or to an existing subscription. The XML file can be created using the export subscription command or using the export wizard in Management Console. Import requires additional connections to the source and target datastores and cannot run when multiuser configuration is enabled. Use Management Console to import when multiuser configuration is in use.

Syntax diagram

.arrow, .syntaxarrow { fill: none; stroke: black; } .arrowheadStartEnd, .arrowheadRepSep, .arrowheadRepSepReturn { stroke: black; fill: black; } .arrowheadSeq, .arrowheadStartChoice, .arrowheadAfterChoice, .arrowheadStartRepGroup, .arrowheadEndRepGroup, .arrowheadRev { stroke: none; fill: none; } rect { fill: none; stroke: none; } rect.fragref,rect.syntaxfragref { fill: none; stroke: black; } text { fill: #000000; fill-opacity: 1; font-family: IBM Plex Sans,Arial Unicode MS,Arial,Helvetica; font-style: normal; font-weight: normal; font-size: 8pt; stroke: #000000; stroke-width: 0.1; } text.var, text.syntaxvar {font-style:italic;} import subscriptionfilenamefiletypenewexistingReplaceAll existingKeepExtranamesubscription sourceididentifiersourceDatastoresource targetDatastoretargetoldSourceSchemaold sourcenewSourceSchemanew source oldTargetSchemaold targetnewTargetSchemanew target

Syntax

```
import subscription [filename <file>] [type <import>] [name <subscription>]

[sourceid <identifier>] [sourceDatastore <source>] [targetDatastore <target>];

[oldSourceSchema <old source>] [newSourceSchema <new source>] [oldTargetSchema

<old target>] [newTargetSchema <new target>];
```

Parameters

- [filename <file>]

- Specifies the name of the XML file that contains the subscription definition to be imported.

- [type <import>]

- Specify whether you want to import the subscription defined in the XML file to a new subscription or to an existing subscription. If you are importing to an existing subscription, you will need to specify whether to replace all mappings in the existing subscription, or only replace the mappings from the imported subscription.
  Valid values:
  - **new**—Specifies that you are importing to a new subscription.
  - **existingReplaceAll**—Specifies that you are importing to an existing subscription and all mappings in the existing subscription will be replaced.
  - **existingKeepExtra**—Specifies that you are importing to an existing subscription and that only the mappings from the imported subscription will be replaced in the existing subscription.

- [name <subscription>]

- Specifies the name of the subscription to import. If a value is not specified, it will default to the subscription name from the XML definition.

- [sourceid <identifier>]
  - Specifies the source identifier for the new subscription on the target datastore.
  - The identifier must be unique on the target.
  - If no identifier is provided, the default value for the Source ID will be the first eight characters of the subscription name.

- [sourceDatastore <source>]
  - Specifies the name of the source datastore for the imported subscription. If a value is not specified, it will default to the datastore that is currently set to source context.

- [targetDatastore <target>]
  - Specifies the name of the target datastore for the imported subscription. If a value is not specified, it will default to the datastore that is currently set to target context.

- [oldSourceSchema <old source>]
  - Specify the table owner in the original subscription on the old source datastore. Use the format database.schema when required by the datastore. Only schemas that are changing between the original and imported subscription need to be specified. For each old source schema, you must provide the new source schema.This parameter can be specified more than once.

- [newSourceSchema <new source>]
  - Specify the table owner in the imported subscription on the new source datastore. Use the format database.schema when required by the datastore. Only schemas that are changing between the original and imported subscription need to be specified. For each old source schema, you must provide the new source schema.This parameter can be specified more than once.

- [oldTargetSchema <old target>]
  - Specify the table owner in the original subscription on the old target datastore. Use the format database.schema when required by the datastore. Only schemas that are changing between the original and imported subscription need to be specified. For each old target schema, you must provide the new target schema.This parameter can be specified more than once.

- [newTargetSchema <new target>]
  - Specify the table owner in the imported subscription on the new target datastore. Use the format database.schema when required by the datastore. Only schemas that are changing between the original and imported subscription need to be specified. For each old target schema, you must provide the new target schema.This parameter can be specified more than once.

Required role

- Administrator
- System administrator

Result

This command returns a value of 0 if the command was successful and a negative value if the command fails. When verbose mode is enabled, the command prints success message CDC1327.

Sample

Imports a subscription saved in sub1.xml into a new subscription called SUB2.

```
connect server username user1 password password1;

connect datastore name DS1 context source;

connect datastore name DS2 context target;
```

```
import subscription filename c:\\files\\sub1.xml type new name SUB2;
disconnect server;
exit;
```

Parent topic:Subscription commands

list subscriptions

Use this command to list the subscriptions that are currently loaded.

Syntax diagram

.arrow, .syntaxarrow { fill: none; stroke: black; } .arrowheadStartEnd, .arrowheadRepSep, .arrowheadRepSepReturn { stroke: black; fill: black; } .arrowheadSeq, .arrowheadStartChoice, .arrowheadAfterChoice, .arrowheadStartRepGroup, .arrowheadEndRepGroup, .arrowheadRev { stroke: none; fill: none; } rect { fill: none; stroke: none; } rect.fragref,rect.syntaxfragref { fill: none; stroke: black; } text { fill: #000000; fill-opacity: 1; font-family: IBM Plex Sans,Arial Unicode MS,Arial,Helvetica; font-style: normal; font-weight: normal; font-size: 8pt; stroke: #000000; stroke-width: 0.1; } text.var, text.syntaxvar {font-style:italic;} list subscriptionsfilteralldatastorenamedatastore *

Syntax

```
list subscriptions [filter <type>] [name <datastore>];
```

Parameters

- [filter <type>]

- Specifies how the subscriptions will be selected. If a datastore filter is selected and the name is not provided, the current context will be used. Valid values:
  - **all**—Specificies that all subscriptions will be selected.
  - **datastore**—Specifies that the subscriptions for a specified datastore will be selected.

  Default value: all

- [name <datastore>]

- Specifies the name of the datastore when the filter is set to datastore.

Required role

Any

Result

A ResultStringTable object containing the list of subscriptions that are currently loaded.

Sample

Lists all of the subscriptions for the connected datastores.

```
connect server username user1 password password1;
connect datastore name DS1 context source;
connect datastore name DS2 context target;
list subscriptions;
disconnect server;
exit;
```

Lists all of the subscriptions where datastore DS1 is either the source or target.

```
connect server username user1 password password1;
connect datastore name DS1;
list subscriptions filter datastore name DS1;
disconnect server;
exit;
```

Lists all of the subscriptions between source datastore DS1 and target datastore DS2.

```
connect server username user1 password password1;
connect datastore name DS1;
connect datastore name DS2;
```

```
list subscriptions filter datastore;
disconnect server;
exit;
```

Parent topic:Subscription commands

lock subscription

Use this command to lock a subscription for editing. Locking is only available when the datastore is configured for multiuser mode in Access Manager. Use the show subscription command to view whether the subscription is already locked and which user locked it.

Syntax diagram

.arrow, .syntaxarrow { fill: none; stroke: black; } .arrowheadStartEnd, .arrowheadRepSep, .arrowheadRepSepReturn { stroke: black; fill: black; } .arrowheadSeq, .arrowheadStartChoice, .arrowheadAfterChoice, .arrowheadStartRepGroup, .arrowheadEndRepGroup, .arrowheadRev { stroke: none; fill: none; } rect { fill: none; stroke: none; } rect.fragref,rect.syntaxfragref { fill: none; stroke: black; } text { fill: #000000; fill-opacity: 1; font-family: IBM Plex Sans,Arial Unicode MS,Arial,Helvetica; font-style: normal; font-weight: normal; font-size: 8pt; stroke: #000000; stroke-width: 0.1; } text.var, text.syntaxvar {font-style:italic;} lock subscriptionnamesubscription

Syntax

```
lock subscription [name <subscription>];
```

Parameters

- [name <subscription>]

- Specifies the name of the subscription.If a name is not provided, the subscription that is currently identified as the context will be used. To view the current context, use the show context command.

Required role

- Operator
- Administrator
- System administrator

Result

This command returns a value of 0 if the command was successful and a negative value if the command fails. When verbose mode is enabled, the command prints success message CDC1322. **Sample**

Locks the selected subscription.

```
connect server username user1 password password1;

connect datastore name DS1 context source;

connect datastore name DS2 context target;

select subscription name SUB1;

lock subscription;

disconnect server;

exit;
```

Parent topic: Subscription commands

modify latency thresholds

Use this command to modify the latency thresholds for a subscription.

Syntax diagram

.arrow, .syntaxarrow { fill: none; stroke: black; } .arrowheadStartEnd, .arrowheadRepSep, .arrowheadRepSepReturn { stroke: black; fill: black; } .arrowheadSeq, .arrowheadStartChoice, .arrowheadAfterChoice, .arrowheadStartRepGroup, .arrowheadEndRepGroup, .arrowheadRev { stroke: none; fill: none; } rect { fill: none; stroke: none; } rect.fragref,rect.syntaxfragref { fill: none; stroke: black; } text { fill: #000000; fill-opacity: 1; font-family: IBM Plex Sans,Arial Unicode MS,Arial,Helvetica; font-style: normal; font-weight: normal; font-size: 8pt; stroke: #000000; stroke-width: 0.1; } text.var, text.syntaxvar {font-style:italic;} modify latency thresholdsnamesubscriptionenabledfalse/notrue/yes warning0timeproblemminutes

Syntax

```
modify latency thresholds [name <subscription>] enabled <value> [warning <time>]

[problem <minutes>];
```

Parameters

- [name <subscription>]

- Specifies the name of the subscription.If a name is not provided, the subscription that is currently identified as the context will be used. To view the current context, use the show context command.

- enabled <value>

- Enables latency thresholds. The problem and warning threshold parameters are not required when you disable latency thresholds.Valid values:
  - **true/yes**—Enables latency thresholds. Either yes or true are valid values.
  - **false/no**—Disables latency thresholds. Either no or false are valid values.

- [warning <time>]

- Specifies the warning threshold in minutes. Use zero to set a problem threshold and no warning threshold.

- [problem <minutes>]

- Specifies the problem threshold in minutes. This value must be greater than the value set in the warning threshold.

Required role

- Administrator
- System administrator

Result

This command returns a value of 0 if the command was successful and a negative value if the command fails. When verbose mode is enabled, the command prints success message CDC1305.

Sample

Specifies a problem threshold of 2 minutes and a warning threshold of 1 minute.

```
connect server username user1 password password1;

connect datastore name DS1 context source;

connect datastore name DS2 context target;

select subscription name SUB1;

modify latency thresholds enabled true problem 2 warning 1;

disconnect server;
```

```
exit;
```

Specifies a problem threshold of 5 minutes and no warning threshold.

```
connect server username user1 password password1;
connect datastore name DS1 context source;
connect datastore name DS2 context target;
select subscription name SUB1;
modify latency thresholds enabled true problem 5 warning 0;
disconnect server;
exit;
```

Disables latency thresholds.

```
connect server username user1 password password1;
connect datastore name DS1 context source;
connect datastore name DS2 context target;
select subscription name SUB1;
modify latency thresholds enabled false;
disconnect server;
exit;
```

Parent topic: Subscription commands

modify subscription

Use this command to modify the properties of an existing subscription.

Syntax diagram

.arrow, .syntaxarrow { fill: none; stroke: black; } .arrowheadStartEnd, .arrowheadRepSep, .arrowheadRepSepReturn { stroke: black; fill: black; } .arrowheadSeq, .arrowheadStartChoice, .arrowheadAfterChoice, .arrowheadStartRepGroup, .arrowheadEndRepGroup, .arrowheadRev { stroke: none; fill: none; } rect { fill: none; stroke: none; } rect.fragref,rect.syntaxfragref { fill: none; stroke: black; } text { fill: #000000; fill-opacity: 1; font-family: IBM Plex Sans,Arial Unicode MS,Arial,Helvetica; font-style: normal; font-weight: normal; font-size: 8pt; stroke: #000000; stroke-width: 0.1; } text.var, text.syntaxvar {font-style:italic;} modify subscription namesubscriptiondescriptiondescription transferableworksourcetargettcphostnamehostfirewallportnumberpersistencyfalse/no true/yesmirrorContinueOnErrordefaultendcontinuerefreshContinueOnErrordefault endcontinuerelatedSubscriptions,value*""engineCommunicationtcpjms maintainHistorytargetsourceminimizeNetworkLoadfalse/notrue/yes

Syntax

```
modify subscription name <subscription> [description <description>]

[transferablework <transfer>] [tcphostname <host>] [firewallport number]

[persistency <state>] [mirrorContinueOnError <mirror>] [refreshContinueOnError

<refresh>] [relatedSubscriptions <value>] [engineCommunication <method>]

[maintainHistory <history>] [minimizeNetworkLoad <load>];
```

Parameters

- name <subscription>
- Specifies the name of the subscription.If a name is not provided, the subscription that is currently identified as the context will be used. To view the current context, use the show context command.

- [description <description>]
- Specifies the description of the subscription.
- You can specify to clear the description.

- [transferablework <transfer>]
- Selects the datastore to handle transferable work. Valid values:
  - **source**—InfoSphere® CDC will attempt to perform the encoding conversion workload on the server where your source datastore is installed.
  - **target**—InfoSphere CDC will attempt to perform the encoding conversion workload on the server where your target datastore is installed.

- [tcphostname <host>]
- Specifies the TCP host name for the target datastore. If no TCP host name is provided, the host is set to auto-select based on the target datastore.

- [firewallport number]
- Specifies the firewall port number used to connect to the target datastore.

- [persistency <state>]
- Marks the subscription as persistent, if supported by the datastore. Valid values:
  - **true/yes**—Marks the subscription as persistent. Either yes or true are valid values.

- **false/no**—Does not mark the subscription as persistent. Either no or false are valid values.

  Default value: false/no

- **[mirrorContinueOnError <mirror>]**
  - Indicates if mirroring of the subscription should continue when an error is encountered, where supported by the datastore. Valid values:
    - **default**—Specifies that the datastore's setting is used.
    - **end**—Specifies that mirroring will end after an apply error on the target database.
    - **continue**—Specifies that mirroring will continue after an apply error on the target database.
- **[refreshContinueOnError <refresh>]**
  - Indicates if the refresh of the subscription should continue when an error is encountered, where supported by the datastore. Valid values:
    - **default**—Specifies that the datastore's setting is used.
    - **end**—Specifies that a refresh will end after an apply error on the target database.
    - **continue**—Specifies that a refresh will continue after an apply error on the target database.
- **[relatedSubscriptions <value>]**
  - Specifies the related subscriptions for recursion prevention. Valid values:
    - Use an empty string () to indicate no related subscriptions
    - Use an asterisk (*) to indicate all subscriptions
    - Enter a comma-delimited list of subscription names
- **[engineCommunication <method>]**
  - Specifies the communication protocol between engines, where supported.Valid values:
    - **tcp**—Specifies that TCP/IP will be the communication protocol used.
    - **jms**—Specifies that JMS will be the communication protocol used.
- **[maintainHistory <history>]**
  - Specify whether the source datastore will replicate history tables or the target database will maintain history, when system or bitemporal tables are supported by the source and target datastores.Valid values:
    - **source**—Specifies that the source datastore will replicate history tables.
    - **target**—Specifies that the target datastore will maintain history.
- **[minimizeNetworkLoad <load>]**
  - Specify whether to minimize the network load where possible, even if it may result in increased replication latency. This option is only available when both the source and target datastores support the functionality.Valid values:
    - **true/yes**—Specifies that LOB columns will only be sent when a change is detected.
    - **false/no**—Specifies that LOB columns will be sent regardless of a change occurring.

Required role

- Administrator
- System administrator

Result

This command returns a value of 0 if the command was successful and a negative value if the command fails. When verbose mode is enabled, the command prints success message CDC1302. **Sample**

Selects a subscription and modifies its error attributes.

```
connect server username user1 password password1;

connect datastore name DS1 context source;

connect datastore name DS2 context target;

select subscription name SUB1;

modify subscription mirrorContinueOnError end refreshContinueOnError end;

disconnect server;

exit;
```

Parent topic: Subscription commands

modify subscription datastage properties

Use this command to modify the InfoSphere® DataStage® properties for a subscription.

Syntax diagram

Syntax 1 (Direct Connect)

.arrow, .syntaxarrow { fill: none; stroke: black; } .arrowheadStartEnd, .arrowheadRepSep, .arrowheadRepSepReturn { stroke: black; fill: black; } .arrowheadSeq, .arrowheadStartChoice, .arrowheadAfterChoice, .arrowheadStartRepGroup, .arrowheadEndRepGroup, .arrowheadRev { stroke: none; fill: none; } rect { fill: none; stroke: none; } rect.fragref,rect.syntaxfragref { fill: none; stroke: black; } text { fill: #000000; fill-opacity: 1; font-family: IBM Plex Sans,Arial Unicode MS,Arial,Helvetica; font-style: normal; font-weight: normal; font-size: 8pt; stroke: #000000; stroke-width: 0.1; } text.var, text.syntaxvar {font-style:italic;} modify subscription datastage propertiesnamesubscription batchSizeMaxRows100000rowsbatchSizeTimeSeconds600timeclobTruncationSize 8000characterblobTruncationSize8000binaryprojectNameprojectjobNamejob connectionKeykeyautoStartJobtrue/yesfalse/no

Syntax 2 (Flat File)

.arrow, .syntaxarrow { fill: none; stroke: black; } .arrowheadStartEnd, .arrowheadRepSep, .arrowheadRepSepReturn { stroke: black; fill: black; } .arrowheadSeq, .arrowheadStartChoice, .arrowheadAfterChoice, .arrowheadStartRepGroup, .arrowheadEndRepGroup, .arrowheadRev { stroke: none; fill: none; } rect { fill: none; stroke: none; } rect.fragref,rect.syntaxfragref { fill: none; stroke: black; } text { fill: #000000; fill-opacity: 1; font-family: IBM Plex Sans,Arial Unicode MS,Arial,Helvetica; font-style: normal; font-weight: normal; font-size: 8pt; stroke: #000000; stroke-width: 0.1; } text.var, text.syntaxvar {font-style:italic;} modify subscription datastage propertiesbatchSizeMaxRows100000 rowsbatchSizeTimeSeconds600timeclobTruncationSize8000character blobTruncationSize8000binaryincludeRecordCounttrue/yesfalse/no

Syntax

Syntax 1 (Direct Connect)

```
modify subscription datastage properties [name <subscription>] [batchSizeMaxRows
<rows>] [batchSizeTimeSeconds <time>] [clobTruncationSize <character>]
[blobTruncationSize <binary>] [projectName <project>] [jobName <job>] [connectionKey
<key>] [autoStartJob <start>];
```

Syntax 2 (Flat File)

```
modify subscription datastage properties [batchSizeMaxRows <rows>]
[batchSizeTimeSeconds <time>] [clobTruncationSize <character>]
[blobTruncationSize <binary>] [includeRecordCount <count>];
```

Parameters

Syntax 1 (Direct Connect)

- [name <subscription>]

- Specifies the name of the subscription.If a name is not provided, the subscription that is currently identified as the context will be used. To view the current context, use the show context command.

- **[batchSizeMaxRows <rows>]**
  - Specifies the number of rows that can be changed before subscription data is sent to InfoSphere DataStage.
- **[batchSizeTimeSeconds <time>]**
  - Specifies the amount of elapsed time before subscription data is sent to InfoSphere DataStage.
- **[clobTruncationSize <character>]**
  - Specifies the truncation point for large character data (in characters).
- **[blobTruncationSize <binary>]**
  - Specifies the truncation point for large binary data (in bytes).
- **[projectName <project>]**
  - Specifies one part of a unique identifier for the job that is being sent to InfoSphere DataStage using the direct connect option.
- **[jobName <job>]**
  - Specifies another part of a unique identifier for the job that is being sent to InfoSphere DataStage using the direct connect option.
- **[connectionKey <key>]**
  - Identifies the connection key information used by InfoSphere DataStage to authenticate the job sending data using the direct connect option.
- **[autoStartJob <start>]**
  - Specifies that the job will auto-start. This parameter requires that InfoSphere DataStage server and the InfoSphere CDC for InfoSphere DataStage server are both installed on the same machine.Valid values:
    - **true/yes**—Enables auto-start for the job. Either yes or true are valid values.
    - **false/no**—Does not enable auto-start for the job. Either no or false are valid values.

Syntax 2 (Flat File)

- **[name <subscription>]**
  - Specifies the name of the subscription.If a name is not provided, the subscription that is currently identified as the context will be used. To view the current context, use the show context command.
- **[batchSizeMaxRows <rows>]**
  - Specifies the number of rows that can be changed before subscription data is sent to InfoSphere DataStage.
- **[batchSizeTimeSeconds <time>]**
  - Specifies the amount of elapsed time before subscription data is sent to InfoSphere DataStage.
- **[clobTruncationSize <character>]**
  - Specifies the truncation point for large character data (in characters).
- **[blobTruncationSize <binary>]**
  - Specifies the truncation point for large binary data (in bytes).
- **[includeRecordCount <count>]**
  - Specifies that the record count will be included in the filename.Valid values:
    - **true/yes**—Specifies that the record count will be included in the file name. Either yes or true are valid values.
    - **false/no**—Specifies that the record count will not be included in the file name. Either no or false are valid values.

Required role

- Administrator
- System administrator

Result

This command returns a value of 0 if the command was successful and a negative value if the command fails. When verbose mode is enabled, the command prints success message CDC1302.

Sample

Selects a subscription with InfoSphere DataStage Flat Files table mappings and modifies the batch settings.

```
connect server username user1 password password1;
connect datastore name DS1 context source;
connect datastore name DS2 context target;
select subscription name SUB1;
modify subscription datastage properties
 batchSizeMaxRows 1000 batchSizeTimeSeconds 60;
disconnect server;
exit;
```

Selects a subscription with InfoSphere DataStage Direct Connect table mappings and modifies the project information.

```
connect server username user1 password password1;
connect datastore name DS1 context source;
connect datastore name DS2 context target;
select subscription name SUB1;
modify subscription datastage properties projectName MAIN jobName SUB1;
disconnect server;
exit;
```

Parent topic: Subscription commands

modify subscription user exit

Use this command to modify user exits for a subscription.

Syntax diagram

.arrow, .syntaxarrow { fill: none; stroke: black; } .arrowheadStartEnd, .arrowheadRepSep, .arrowheadRepSepReturn { stroke: black; fill: black; } .arrowheadSeq, .arrowheadStartChoice, .arrowheadAfterChoice, .arrowheadStartRepGroup, .arrowheadEndRepGroup, .arrowheadRev { stroke: none; fill: none; } rect { fill: none; stroke: none; } rect.fragref,rect.syntaxfragref { fill: none; stroke: black; } text { fill: #000000; fill-opacity: 1; font-family: IBM Plex Sans,Arial Unicode MS,Arial,Helvetica; font-style: normal; font-weight: normal; font-size: 8pt; stroke: #000000; stroke-width: 0.1; } text.var, text.syntaxvar {font-style:italic;} modify subscription user exitnamesubscriptiontypejavaclassclassname"" classparameter""name

Syntax

```
modify subscription user exit [name <subscription>] [type <function>]

[classname <class>] [parameter <name>];
```

Parameters

- [name <subscription>]

- Specifies the name of the subscription.If a name is not provided, the subscription that is currently identified as the context will be used. To view the current context, use the show context command.

- [type <function>]

- Specifies the function type of the user exit.Valid value: javaclass

- [classname <class>]

- Specifies the name of the Java™ class user exit that implements the SubscriptionUserExitIF interface. If a class name is not provided, the user exit will be cleared.

- [parameter <name>]

- Specifies the parameters that are to be used with the user exit program.

Required role

- Administrator
- System administrator

Result

This command returns a value of 0 if the command was successful and a negative value if the command fails. When verbose mode is enabled, the command prints success message CDC1309. **Sample**

Selects a subscription and sets a user exit.

```
connect server username user1 password password1;

connect datastore name DS1 context source;

connect datastore name DS2 context target;

select subscription name SUB1;

modify subscription user exit classname CustomHandler;

disconnect server;

exit;
```

Parent topic: Subscription commands

promote subscription

Use this command to promote the specified subscription to a new subscription or to an existing one.

Syntax diagram

.arrow, .syntaxarrow { fill: none; stroke: black; } .arrowheadStartEnd, .arrowheadRepSep, .arrowheadRepSepReturn { stroke: black; fill: black; } .arrowheadSeq, .arrowheadStartChoice, .arrowheadAfterChoice, .arrowheadStartRepGroup, .arrowheadEndRepGroup, .arrowheadRev { stroke: none; fill: none; } rect { fill: none; stroke: none; } rect.fragref,rect.syntaxfragref { fill: none; stroke: black; } text { fill: #000000; fill-opacity: 1; font-family: IBM Plex Sans,Arial Unicode MS,Arial,Helvetica; font-style: normal; font-weight: normal; font-size: 8pt; stroke: #000000; stroke-width: 0.1; } text.var, text.syntaxvar {font-style:italic;} promote subscription namesubscriptiontypenewexistingReplaceAll existingKeepExtrapromotionNamenamesourceididentifiersourceDatastoresource targetDatastoretargetoldSourceSchemaold sourcenewSourceSchemanew source oldTargetSchemaold targetnewTargetSchemanew target

Syntax

```
promote subscription [name <subscription>] type <import> [promotionName <name>]

[sourceid <identifier>] [sourceDatastore <source>] [targetDatastore <target>]

[oldSourceSchema <old source>] [newSourceSchema <new source>] [oldTargetSchema

<old target>] [newTargetSchema <new target>];
```

Parameters

- **[name <subscription>]**
  - Specifies the name of the subscription.If a name is not provided, the subscription that is currently identified as the context will be used. To view the current context, use the show context command.
- **type <import>**
  - Specify whether you want to promote the subscription to a new subscription or to an existing one. If promote into existing one, specifies to replace all mappings in the existing subscription, or only replace mappings of the subscription promoted from.Valid values:
    - **new**—Specifies that you are importing to a new subscription.
    - **existingReplaceAll**—Specifies that you are importing to an existing subscription and all mappings in the existing subscription will be replaced.
    - **existingKeepExtra**—Specifies that you are importing to an existing subscription and that only the mappings from the imported subscription will be replaced in the existing subscription.
- **[promotionName <name>]**
  - Specify the name of the subscription to promote into. If not specified, default to the subscription name that promoted from.
- **[sourceid <identifier>]**
  - Specifies the source identifier for the new subscription on the target datastore.
  - The identifier must be unique on the target.
  - If no identifier is provided, the default value for the Source ID will be the first eight characters of the subscription name.

- [sourceDatastore <source>]
- Specifies the name of the source datastore for the imported subscription. If a value is not specified, it will default to the datastore that is currently set to source context.

- [targetDatastore <target>]
- Specifies the name of the target datastore for the imported subscription. If a value is not specified, it will default to the datastore that is currently set to target context.

- [oldSourceSchema <old source>]
- Specify the table owner in the original subscription on the old source datastore. Use the format database.schema when required by the datastore. Only schemas that are changing between the original and promoted subscription need to be specified. For each old source schema, you must provide the new source schema.This parameter can be specified more than once.

- [newSourceSchema <new source>]
- Specify the table owner in the promotion subscription on the new source datastore. Use the format database.schema when required by the datastore. Only schemas that are changing between the original and promoted subscription need to be specified. For each old source schema, you must provide the new source schema.This parameter can be specified more than once.

- [oldTargetSchema <old target>]
- Specify the table owner in the original subscription on the old target datastore. Use the format database.schema when required by the datastore. Only schemas that are changing between the original and promoted subscription need to be specified. For each old target schema, you must provide the new target schema.This parameter can be specified more than once.

- [newTargetSchema <new target>]
- Specify the table owner in the promotion subscription on the new target datastore. Use the format database.schema when required by the datastore. Only schemas that are changing between the original and promoted subscription need to be specified. For each old target schema, you must provide the new target schema.This parameter can be specified more than once.

Required role

- Administrator
- System administrator

Result

This command returns a value of 0 if the command was successful and a negative value if the command fails. When verbose mode is enabled, the command prints success message CDC1328.

Sample

Promote a subscription SUB1 into a new subscription SUB2.

```
connect server username user1 password password1;
connect datastore name DS1 context source;
connect datastore name DS2 context target;
promote subscription name SUB1 type new promotionName SUB2;
disconnect server;
exit;
```

Parent topic: Subscription commands

select subscription

Use this command to select the subscription to context. The subscription will be used for subsequent commands.Use the show context command or the list subscriptions command to view which subscription is currently the context.

Syntax diagram

.arrow, .syntaxarrow { fill: none; stroke: black; } .arrowheadStartEnd, .arrowheadRepSep, .arrowheadRepSepReturn { stroke: black; fill: black; } .arrowheadSeq, .arrowheadStartChoice, .arrowheadAfterChoice, .arrowheadStartRepGroup, .arrowheadEndRepGroup, .arrowheadRev { stroke: none; fill: none; } rect { fill: none; stroke: none; } rect.fragref,rect.syntaxfragref { fill: none; stroke: black; } text { fill: #000000; fill-opacity: 1; font-family: IBM Plex Sans,Arial Unicode MS,Arial,Helvetica; font-style: normal; font-weight: normal; font-size: 8pt; stroke: #000000; stroke-width: 0.1; } text.var, text.syntaxvar {font-style:italic;} select subscriptionnamesubscription

Syntax

```
select subscription name <subscription>;
```

Parameters

- name <subscription>

- Specifies the name of the subscription.If a name is not provided, the subscription that is currently identified as the context will be used. To view the current context, use the show context command.

Required role

Any

Result

This command returns a value of 0 if the command was successful and a negative value if the command fails. When verbose mode is enabled, the command prints success message CDC1308.

Sample

Selects a subscription and shows that it is the current context.

```
connect server username user1 password password1;

connect datastore name DS1 context source;

connect datastore name DS2 context target;

select subscription name SUB1;

show context;

disconnect server;

exit;
```

Parent topic: Subscription commands

show latency thresholds

Use this command to display latency thresholds for a subscription.

Syntax diagram

.arrow, .syntaxarrow { fill: none; stroke: black; } .arrowheadStartEnd, .arrowheadRepSep, .arrowheadRepSepReturn { stroke: black; fill: black; } .arrowheadSeq, .arrowheadStartChoice, .arrowheadAfterChoice, .arrowheadStartRepGroup, .arrowheadEndRepGroup, .arrowheadRev { stroke: none; fill: none; } rect { fill: none; stroke: none; } rect.fragref,rect.syntaxfragref { fill: none; stroke: black; } text { fill: #000000; fill-opacity: 1; font-family: IBM Plex Sans,Arial Unicode MS,Arial,Helvetica; font-style: normal; font-weight: normal; font-size: 8pt; stroke: #000000; stroke-width: 0.1; } text.var, text.syntaxvar {font-style:italic;} show latency thresholdsnamesubscription

Syntax

```
show latency thresholds [name <subscription>];
```

Parameters

- [name <subscription>]

- Specifies the name of the subscription.If a name is not provided, the subscription that is currently identified as the context will be used. To view the current context, use the show context command.

Required role

- Operator
- Administrator
- System administrator

Result

A ResultStringKeyValues object which shows the warning and problem latency threshold settings, when they are configured. **Sample**

Displays the latency threshold settings for a subscription.

```
connect server username user1 password password1;

connect datastore name DS1 context source;

connect datastore name DS2 context target;

show latency thresholds name SUB1;

disconnect server;

exit;
```

Parent topic: Subscription commands

show subscription

Use this command to display the properties of a subscription.

Syntax diagram

.arrow, .syntaxarrow { fill: none; stroke: black; } .arrowheadStartEnd, .arrowheadRepSep, .arrowheadRepSepReturn { stroke: black; fill: black; } .arrowheadSeq, .arrowheadStartChoice, .arrowheadAfterChoice, .arrowheadStartRepGroup, .arrowheadEndRepGroup, .arrowheadRev { stroke: none; fill: none; } rect { fill: none; stroke: none; } rect.fragref,rect.syntaxfragref { fill: none; stroke: black; } text { fill: #000000; fill-opacity: 1; font-family: IBM Plex Sans,Arial Unicode MS,Arial,Helvetica; font-style: normal; font-weight: normal; font-size: 8pt; stroke: #000000; stroke-width: 0.1; } text.var, text.syntaxvar {font-style:italic;} show subscriptionnamesubscription

Syntax

```
show subscription [name <subscription>];
```

Parameters

- [name <subscription>]

- Specifies the name of the subscription.If a name is not provided, the subscription that is currently identified as the context will be used. To view the current context, use the show context command.

Required role

- Operator
- Administrator
- System administrator

Result

A ResultStringKeyValues object with the attributes of the subscription.

```
PROPERTY
       VALUE
-------------------------- --------------------------
Name:                      SUB1
Description:
Source Datastore:          DS1
Target Datastore:          DS2
Source ID:                 SUB1
TCP Host:
Firewall Port:
Persistency:               false
Transferable Work:         target
Propagation Control:
```

Sample

Displays the properties of a subscription.

```
connect server username user1 password password1;
connect datastore name DS1 context source;
connect datastore name DS2 context target;
select subscription name SUB1;
show subscription;
disconnect server;
exit;
```

Parent topic: Subscription commands

show subscription datastage properties

Use this command to view the InfoSphere® DataStage® properties for a subscription with an InfoSphere DataStage target.

Syntax diagram

.arrow, .syntaxarrow { fill: none; stroke: black; } .arrowheadStartEnd, .arrowheadRepSep, .arrowheadRepSepReturn { stroke: black; fill: black; } .arrowheadSeq, .arrowheadStartChoice, .arrowheadAfterChoice, .arrowheadStartRepGroup, .arrowheadEndRepGroup, .arrowheadRev { stroke: none; fill: none; } rect { fill: none; stroke: none; } rect.fragref,rect.syntaxfragref { fill: none; stroke: black; } text { fill: #000000; fill-opacity: 1; font-family: IBM Plex Sans,Arial Unicode MS,Arial,Helvetica; font-style: normal; font-weight: normal; font-size: 8pt; stroke: #000000; stroke-width: 0.1; } text.var, text.syntaxvar {font-style:italic;} show subscription datastage propertiesnamesubscription

Syntax

```
show subscription datastage properties [name <subscription>];
```

Parameters

- [name <subscription>]

- Specifies the name of the subscription.If a name is not provided, the subscription that is currently identified as the context will be used. To view the current context, use the show context command.

Required role

- Operator
- Administrator
- System administrator

Result

A ResultStringKeyValues object with the InfoSphere DataStage attributes of the subscription. **Sample**

Modifies the InfoSphere DataStage configuration properties for a subscription that contains flat file or direct connect mappings and displays the new values.

```
connect server username user1 password password1;

connect datastore name DS1 context source;

connect datastore name DS2 context target;

select subscription name SUB1;

modify subscription datastage properties

 batchSizeMaxRows 1000 batchSizeTimeSeconds 60;

show subscription datastage properties;

disconnect server;

exit;
```

Parent topic: Subscription commands

show subscription user exit

Use this command to display the user exits for a subscription.

Syntax diagram

.arrow, .syntaxarrow { fill: none; stroke: black; } .arrowheadStartEnd, .arrowheadRepSep, .arrowheadRepSepReturn { stroke: black; fill: black; } .arrowheadSeq, .arrowheadStartChoice, .arrowheadAfterChoice, .arrowheadStartRepGroup, .arrowheadEndRepGroup, .arrowheadRev { stroke: none; fill: none; } rect { fill: none; stroke: none; } rect.fragref,rect.syntaxfragref { fill: none; stroke: black; } text { fill: #000000; fill-opacity: 1; font-family: IBM Plex Sans,Arial Unicode MS,Arial,Helvetica; font-style: normal; font-weight: normal; font-size: 8pt; stroke: #000000; stroke-width: 0.1; } text.var, text.syntaxvar {font-style:italic;} show subscription user exitnamesubscription

Syntax

```
show subscription user exit [name <subscription>];
```

Parameters

- [name <subscription>]

- Specifies the name of the subscription.If a name is not provided, the subscription that is currently identified as the context will be used. To view the current context, use the show context command.

Required role

- Operator
- Administrator
- System administrator

Result

A ResultStringKeyValues object with the user exit configuration for a subscription.

Sample

Modifies the user exit settings for a subscription and displays its new values.

```
connect server username user1 password password1;

connect datastore name DS1 context source;

connect datastore name DS2 context target;

select subscription name SUB1;

modify subscription user exit classname CustomHandler;

show subscription user exit;

disconnect server;

exit;
```

Parent topic: Subscription commands

unlock subscription

Use this command to unlock a subscription.Locking is only available when the datastore is configured for multiuser mode in Access Manager. Use the show subscription command to view which user locked the subscription.

Syntax diagram

.arrow, .syntaxarrow { fill: none; stroke: black; } .arrowheadStartEnd, .arrowheadRepSep, .arrowheadRepSepReturn { stroke: black; fill: black; } .arrowheadSeq, .arrowheadStartChoice, .arrowheadAfterChoice, .arrowheadStartRepGroup, .arrowheadEndRepGroup, .arrowheadRev { stroke: none; fill: none; } rect { fill: none; stroke: none; } rect.fragref,rect.syntaxfragref { fill: none; stroke: black; } text { fill: #000000; fill-opacity: 1; font-family: IBM Plex Sans,Arial Unicode MS,Arial,Helvetica; font-style: normal; font-weight: normal; font-size: 8pt; stroke: #000000; stroke-width: 0.1; } text.var, text.syntaxvar {font-style:italic;} unlock subscription namesubscriptioncommentvalue

Syntax

```
unlock subscription [name <subscription>] [comment <value>];
```

Parameters

- [name <subscription>]

- Specifies the name of the subscription.If a name is not provided, the subscription that is currently identified as the context will be used. To view the current context, use the show context command.

- [comment <value>]

- Specifies an optional comment to record with the unlocking of the subscription.

Required role

- Operator
- Administrator
- System administrator

Result

This command returns a value of 0 if the command was successful and a negative value if the command fails. When verbose mode is enabled, the command prints success message CDC1323. **Sample**

Locks a subscription for a configuration change and unlocks it when the change is complete.

```
connect server username user1 password password1;
connect datastore name DS1 context source;
connect datastore name DS2 context target;
select subscription name SUB1;
lock subscription;
map table sourceschema USER1 sourcetable MISC_SUPPLY targetschema USER1 targettable MISC_SUPPLY;
unlock subscription comment Mapped the MISC_SUPPLY table.;
disconnect server;
exit;
```

Parent topic: Subscription commands

Table mapping commands

- **add rule set**
- **add table mapping**
- **delete rule set**
- **delete table mapping**
- **export rule set**
- **export table mapping**
- **flag refresh**
- **list refresh order**
- **list rule set tables**
- **list rule sets**
- **list table mappings**
- **mark capture point**
- **modify refresh order**
- **modify rule set**
- **modify table mapping**
- **park table mapping**
- **promote rule set**
- **promote table mapping**
- **reassign table mapping**
- **select rule set**
- **select table mapping**
- **show rule set**
- **show table mapping**

Parent topic:CHCCLP Commands

add rule set

Use this command to define rules that identify which tables are in scope for replication.

Syntax diagram

.arrow, .syntaxarrow { fill: none; stroke: black; } .arrowheadStartEnd, .arrowheadRepSep, .arrowheadRepSepReturn { stroke: black; fill: black; } .arrowheadSeq, .arrowheadStartChoice, .arrowheadAfterChoice, .arrowheadStartRepGroup, .arrowheadEndRepGroup, .arrowheadRev { stroke: none; fill: none; } rect { fill: none; stroke: none; } rect.fragref,rect.syntaxfragref { fill: none; stroke: black; } text { fill: #000000; fill-opacity: 1; font-family: IBM Plex Sans,Arial Unicode MS,Arial,Helvetica; font-style: normal; font-weight: normal; font-size: 8pt; stroke: #000000; stroke-width: 0.1; } text.var, text.syntaxvar {font-style:italic;} add rule setnamesubscriptionruleNamesetschemanamestructureOnly false/notrue/yesincludeAllfalse/notrue/yesincludePatternincludeincludeMatchexact startsWithcontainsendsWithexcludePatternexcludeexcludeMatchexactstartsWith containsendsWith

Syntax

```
add rule set [name <subscription>] ruleName <set> schema <name> [structureOnly
<data>] [includeAll <allow>] [includePattern <include>] [includeMatch <type>]
[excludePattern <exclude>] [excludeMatch <match>];
```

Parameters

- [name <subscription>]
- Specifies the name of the subscription.If a name is not provided, the datastore that is currently set to source and target context will be used.

- ruleName <set>
- Specifies the name of the rule set.Rule set names must be unique within a subscription.

- schema <name>
- Specifies the schema for the new rule set.

- [structureOnly <data>]
- Specifies whether to replicate only structural (DDL), not data (DML), changes for the tables included in the rule.Valid values:
  - **true/yes**—Only structural (DDL) changes are replicated. Either yes or true are valid values.
  - **false/no**—Both structural (DDL) and data (DML) changes are replicated. Either no or false are valid values.

  Default value: false/no

- [includeAll <allow>]
- Specifies whether to allow all tables to be considered for inclusion in the rule set.Valid values:
  - **true/yes**—All tables are to be considered for inclusion in the rule set. Either yes or true are valid values.
  - **false/no**—Not all tables are to be considered for inclusion in the rule set. Either no or false are valid values.

  Default value: false/no

- [includePattern <include>]

- Specifies the criteria by which tables will be judged for inclusion from the rule set. Patterns are case-sensitive.This parameter can be specified more than once.

- includeMatch <type>]

- Specifies the pattern matching type used for table inclusion.This parameter can be specified more than once.
  Valid values:
  - exact
  - startsWith
  - contains
  - endsWith

- [excludePattern <exclude>]

- Specifies the criteria by which tables will be judged for exclusion in from the rule set. Patterns are case-sensitive.This parameter can be specified more than once.

- [excludeMatch <match>]

- Specifies the pattern matching type used for table exclusion.This parameter can be specified more than once.
  Valid values:
  - exact
  - startsWith
  - contains
  - endsWith

Required role

- Administrator
- System administrator

Result

This command returns a value of 0 if the command was successful and a negative value if the command fails. When verbose mode is enabled, the command prints success message CDC1701.

Sample

SAMPLE 1: Creates a subscription and adds a rule set to include tables that starts with A or B for the given schema and exclude tables that ends with C.

```
connect server
username user1 password password1;
connect datastore name DS1 context source;
connect datastore name DS2 context target;
add subscription name SUB1;
add rule set ruleName RULE1 schema USER1 structureOnly false
 includePattern A includeMatch startsWith
 includePattern B includeMatch startsWith
 excludePattern C excludeMatch endsWith;
list rule sets;
disconnect server;
exit;
```

SAMPLE 2: Creates a subscription and adds a rule set to include all tables for the given schema and exclude table X and table Y.

```
connect server username user1 password password1;
connect datastore name DS1 context source;
connect datastore name DS2 context target;
add subscription name SUB1;
add rule set ruleName RULE1 schema USER1 structureOnly false
 includeAll yes
 excludePattern X excludeMatch exact
 excludePattern Y excludeMatch exact;
list rule sets;
disconnect server;
exit;
```

Parent topic: Table mapping commands

add table mapping

Use this command to map a source table to a target table and set the table mapping to the current context.

Syntax diagram

Syntax 1: Maps a source table to a target table.
.arrow, .syntaxarrow { fill: none; stroke: black; } .arrowheadStartEnd, .arrowheadRepSep, .arrowheadRepSepReturn { stroke: black; fill: black; } .arrowheadSeq, .arrowheadStartChoice, .arrowheadAfterChoice, .arrowheadStartRepGroup, .arrowheadEndRepGroup, .arrowheadRev { stroke: none; fill: none; } rect { fill: none; stroke: none; } rect.fragref,rect.syntaxfragref { fill: none; stroke: black; } text { fill: #000000; fill-opacity: 1; font-family: IBM Plex Sans,Arial Unicode MS,Arial,Helvetica; font-style: normal; font-weight: normal; font-size: 8pt; stroke: #000000; stroke-width: 0.1; } text.var, text.syntaxvar {font-style:italic;} add table mappingnamesubscriptionsourceDatabasesource sourceSchemaschemasourceTabletabletargetDatabasetargettargetSchema schema2targetTablenametargetIndexModeautoDetectindexallSearchablecolumns targetIndexNameindextargetIndexColumnscolumntypestandardadaptive summarizationauditconsolidationOneToOneconsolidationOneToManymethodmirror refreshpreventRecursionfalse/notrue/yes

Syntax 2: Maps a source table to an InfoSphere DataStage.
.arrow, .syntaxarrow { fill: none; stroke: black; } .arrowheadStartEnd, .arrowheadRepSep, .arrowheadRepSepReturn { stroke: black; fill: black; } .arrowheadSeq, .arrowheadStartChoice, .arrowheadAfterChoice, .arrowheadStartRepGroup, .arrowheadEndRepGroup, .arrowheadRev { stroke: none; fill: none; } rect { fill: none; stroke: none; } rect.fragref,rect.syntaxfragref { fill: none; stroke: black; } text { fill: #000000; fill-opacity: 1; font-family: IBM Plex Sans,Arial Unicode MS,Arial,Helvetica; font-style: normal; font-weight: normal; font-size: 8pt; stroke: #000000; stroke-width: 0.1; } text.var, text.syntaxvar {font-style:italic;} add table mappingnamesubscriptionsourceDatabasesource sourceSchemaschemasourceTabletabletypedirectConnectflatFilerecordFormat singleRecordmultiRecorddirectory""pathcustomFormattercustom

Syntax 3: Adds a table to the datastore for use as a source table for replication in a subscription. The syntax should be used when target datastore is external. .arrow, .syntaxarrow { fill: none; stroke: black; } .arrowheadStartEnd, .arrowheadRepSep, .arrowheadRepSepReturn { stroke: black; fill: black; } .arrowheadSeq, .arrowheadStartChoice, .arrowheadAfterChoice, .arrowheadStartRepGroup, .arrowheadEndRepGroup, .arrowheadRev { stroke: none; fill: none; } rect { fill: none; stroke: none; } rect.fragref,rect.syntaxfragref { fill: none; stroke: black; } text { fill: #000000; fill-opacity: 1; font-family: IBM Plex Sans,Arial Unicode MS,Arial,Helvetica; font-style: normal; font-weight: normal; font-size: 8pt; stroke: #000000; stroke-width: 0.1; } text.var, text.syntaxvar {font-style:italic;} add table mappingnamesubscription sourceDatabasedatabasesourceSchemaschemasourceTabletable

Syntax 4: Maps a source table to a target table. The syntax should be used when source datastore is unknown. .arrow, .syntaxarrow { fill: none; stroke: black; } .arrowheadStartEnd, .arrowheadRepSep, .arrowheadRepSepReturn { stroke: black; fill: black; } .arrowheadSeq, .arrowheadStartChoice, .arrowheadAfterChoice, .arrowheadStartRepGroup, .arrowheadEndRepGroup, .arrowheadRev { stroke: none; fill: none; } rect { fill: none; stroke: none; } rect.fragref,rect.syntaxfragref { fill: none; stroke: black; } text { fill: #000000; fill-opacity: 1; font-family: IBM Plex Sans,Arial Unicode MS,Arial,Helvetica; font-style: normal; font-weight: normal; font-size: 8pt; stroke: #000000; stroke-width: 0.1; } text.var, text.syntaxvar {font-style:italic;} add table mappingnamesubscriptionsourceDatabasedatabase sourceSchemaschemasourceTabletabletargetDatabasetargetDatabase targetSchemaschema2targetTabletargetTabletargetIndexModemode targetIndexNameindextargetIndexColumnscolumntypestandardadaptive summarizationauditconsolidationOneToOneconsolidationOneToMany

Syntax

Syntax 1: Maps a source table to a target table.

```
add table mapping [name <subscription>] [sourceDatabase <source>] sourceSchema
<schema> sourceTable <table> [targetDatabase <target>] targetSchema <schema2>
targetTable <name> [targetIndexMode <mode>] [targetIndexName <index>]
[targetIndexColumns <column>] [type <type>] [method <method>] [preventRecursion
<value>];
```

Syntax 2 - Maps a source table to an InfoSphere® DataStage® target.

```
add table mapping [name <subscription>] [sourceDatabase <source>] sourceSchema
<schema> sourceTable <table> [type <type>] [recordFormat <format>]
[directory <path>] [customFormatter <custom>];
```

Syntax 3 - Adds a table to the datastore for use as a source table for replication in a subscription. The syntax should be used when target datastore is external.

```
add table mapping [name <subscription>] [sourceDatabase <database>]
sourceSchema <schema> sourceTable <table>;
```

Syntax 4: Maps a source table to a target table. The syntax should be used when source datastore is unknown.`add table mapping [name <subscription>] [sourceDatabase <database>]`

```
sourceSchema
<schema> sourceTable <table> [[targetDatabase <target>] targetSchema
<schema2> targetTable <name>] [targetIndexMode <mode>] [targetIndexName <index>]
[targetIndexColumns <column>] [type <type>];
```

Parameters

Syntax 1: Maps a source table to a target table.

- [name <subscription>]
- Specifies the name of the subscription.If a name is not provided, the subscription that is currently identified as the context will be used. To view the current context, use the show context command.

- [sourceDatabase <source>]
- Specifies the database for the source table.

- sourceSchema <schema>
- Specifies the schema for the source table.

- **sourceTable <table>**
  - Specifies the name of the source table.
- **[targetDatabase <target>]**
  - Specifies the database for the target table.
- **targetSchema <schema2>**
  - Specifies the schema for the target table.
- **targetTable <name>**
  - Specifies the name of the target table.
- **[targetIndexMode <mode>]**
  - Specifies the mode used to locate a row in the target table.Valid values:
    - **autoDetect**—Specifies that an index will be selected automatically.
    - **index**—Specifies that an index will be used to identify a row in the target table.
    - **allSearchable**—Specifies that all target columns will be searched to identify which columns are suitable to uniquely identify rows. The results of the search are used to build a WHERE clause which uniquely identifies the row on the target column to apply data.
    - **columns**—Specifies that a column will be used to identify a row in the target table.

    Default value: default
- **[targetIndexName <index>]**
  - Specifies the target index name when the targetIndexMode is set to `index`.
- **[targetIndexColumns <column>]**
  - Specifies a comma-delimited list of columns used to identify a row when the targetIndexMode is set to `column`.
- **[type <type>]**
  - Specifies the table mapping type.Valid values:
    - **standard**—Specifies that the table will be mapped using standard replication.
    - **adaptive**—Specifies that the table will be mapped using Adaptive Apply.
    - **summarization**—Specifies that the table will be mapped to summarize data.
    - **audit**—Specifies that the table will be mapped using LiveAudit™
    - **consolidationOneToOne**—Specifies that the table will be mapped to consolidate data (one-to-one)
    - **consolidationOneToMany**—Specifies that the table will be mapped to consolidate data (one-to-many)

    Default value: standard
- **[method <method>]**
  - Specifies the replication method for the table mapping.
    Valid values:
    - **mirror**—Immediately replicates changes made to the source table to the target table or accumulate source table changes and replicate at a later time.
    - **refresh**—Replicates a snapshot of the source table to the target table.

    Default value: mirror
- **[preventRecursion <value>]**
  - Specifies whether to prevent from replicating changes back to the source database when a subscription is configured for bidirectional replication.Valid values:

- **true/yes**—Prevents recursion.
- **false/no**—Does not prevent recursion.

Default value: false/no

Syntax 2 - InfoSphere DataStage target

- [name <subscription>]
- Specifies the name of the subscription.If a name is not provided, the subscription that is currently identified as the context will be used. To view the current context, use the show context command.

- [sourceDatabase <source>]
- Specifies the database for the source table.

- sourceSchema <schema>
- Specifies the schema for the source table.

- sourceTable <table>
- Specifies the name of the source table.

- [type <type>]
- Specifies the table mapping type.Valid values:
  - **directConnect**—Specifies that the table will be mapped to InfoSphere DataStage using Direct Connect.
  - **flatFile**—Specifies that the table will be mapped to InfoSphere DataStage using Flat Files.

- [recordFormat <format>]
- Specifies the record format for InfoSphere DataStage mappings.Valid values:
  - **singleRecord**—Specifies that an update operation is sent as a single row.
  - **multiRecord**—Specifies that an update operation is sent as two rows.

  Default value: singleRecord

- [directory <path>]
- Specifies the directory for files in a InfoSphere DataStage Flat File mapping. Default value:

- [customFormatter <custom>]
- Specifies a custom formatter for InfoSphere DataStage flat file output.

Syntax 3: Adds a table to the datastore for use as a source table for replication in a subscription. The syntax should be used when target datastore is external.

- [name <subscription>]
- Specifies the name of the subscription.If a name is not provided, the subscription that is currently identified as the context will be used. To view the current context, use the show context command.

- [sourceDatabase <source>]
- Specifies the database for the source table.

- sourceSchema <schema>
- Specifies the schema for the source table.

- sourceTable <table>
- Specifies the name of the source table.

Syntax 4: Maps a source table to a target table. The syntax should be used when source datastore is unknown.

- [name <subscription>]
- Specifies the name of the subscription.If a name is not provided, the subscription that is currently identified as the context will be used. To view the current context, use the show context command.

- **[sourceDatabase <source>]**
  - Specifies the database for the source table.
- **sourceSchema <schema>**
  - Specifies the schema for the source table.
- **sourceTable <table>**
  - Specifies the name of the source table.
- **[targetDatabase <target>]**
  - Specifies the database for the target table.
- **targetSchema <schema2>**
  - Specifies the schema for the target table.
- **targetTable <name>**
  - Specifies the name of the target table.
- **[targetIndexMode <mode>]**
  - Specifies the mode used to locate a row in the target table.Valid values:
    - **autoDetect**—Specifies that an index will be selected automatically.
    - **index**—Specifies that an index will be used to identify a row in the target table.
    - **allSearchable**—Specifies that all target columns will be searched to identify which columns are suitable to uniquely identify rows. The results of the search are used to build a WHERE clause which uniquely identifies the row on the target column to apply data.
    - **columns**—Specifies that a column will be used to identify a row in the target table.

    Default value: default
- **[targetIndexName <index>]**
  - Specifies the target index name when the targetIndexMode is set to `index`.
- **[targetIndexColumns <column>]**
  - Specifies a comma-delimited list of columns used to identify a row when the targetIndexMode is set to `column`.
- **[type <type>]**
  - Specifies the table mapping type.Valid values:
    - **standard**—Specifies that the table will be mapped using standard replication.
    - **adaptive**—Specifies that the table will be mapped using Adaptive Apply.
    - **summarization**—Specifies that the table will be mapped to summarize data.
    - **audit**—Specifies that the table will be mapped using LiveAudit
    - **consolidationOneToOne**—Specifies that the table will be mapped to consolidate data (one-to-one)
    - **consolidationOneToMany**—Specifies that the table will be mapped to consolidate data (one-to-many)

    Default value: standard

Required role

- Administrator
- System administrator

Result

This command returns a value of 0 if the command was successful and a negative value if the command fails. When verbose mode is enabled, the command prints success message CDC1402.

Sample

Creates a subscription and maps two tables.

```
connect server username user1 password password1;
connect datastore name DS1 context source;
connect datastore name DS2 context target;
add subscription name SUB1;
add table mapping
 sourceSchema USER1 sourceTable TABLE1
 targetSchema USER1 targetTable TABLE1;
add table mapping
 sourceSchema USER1 sourceTable TABLE2
 targetSchema USER1 targetTable TABLE2;
list table mappings;
disconnect server;
exit;
```

Selects the target index column when mapping a table.

```
connect server username user1 password password1;
connect datastore name DS1 context source;
connect datastore name DS2 context target;
select subscription name SUB2;
add table mapping
 sourceSchema USER1 sourceTable CUSTOMER
 targetSchema USER1 targetTable CUSTOMER
 targetIndexMode columns targetIndexColumns CUSTID;
show table mapping;
disconnect server;
exit;
```

Parent topic: Table mapping commands

delete rule set

Use this command to delete a rule set.

Syntax diagram

.arrow, .syntaxarrow { fill: none; stroke: black; } .arrowheadStartEnd, .arrowheadRepSep, .arrowheadRepSepReturn { stroke: black; fill: black; } .arrowheadSeq, .arrowheadStartChoice, .arrowheadAfterChoice, .arrowheadStartRepGroup, .arrowheadEndRepGroup, .arrowheadRev { stroke: none; fill: none; } rect { fill: none; stroke: none; } rect.fragref,rect.syntaxfragref { fill: none; stroke: black; } text { fill: #000000; fill-opacity: 1; font-family: IBM Plex Sans,Arial Unicode MS,Arial,Helvetica; font-style: normal; font-weight: normal; font-size: 8pt; stroke: #000000; stroke-width: 0.1; } text.var, text.syntaxvar {font-style:italic;} delete rule setruleNamename

Syntax

```
delete rule set [ruleName name];
```

Parameters

- [ruleName name]

- Specifies the name of the rule set.If a name is not provided, the rule set that is currently identified as the context will be used. To view the current context, use the show context command.

Required role

- Administrator
- System administrator

Result

This command returns a value of 0 if the command was successful and a negative value if the command fails. When verbose mode is enabled, the command prints success message CDC1702.

Sample

Sample 1: Deletes a rule set by name.

```
connect server username user1 password password1;
connect datastore name DS1 context source;
connect datastore name DS2 context target;
select subscription name SUB1;
delete rule set ruleName rule1;
disconnect server;
exit;
```

Sample 2: Deletes a rule set based on the current context.

```
connect server username user1
password password1;
connect datastore name DS1 context source;
connect datastore name DS2 context target;
select subscription name SUB1;
selete rule set ruleName rule1;
delete rule set;
disconnect server;
exit;
```

Parent topic: Table mapping commands

delete table mapping

Use this command to delete a table mapping. The table mapping must be first selected as the current context. Use the list table mappings command or the show context command to view the current table mapping context.

Syntax diagram

.arrow, .syntaxarrow { fill: none; stroke: black; } .arrowheadStartEnd, .arrowheadRepSep, .arrowheadRepSepReturn { stroke: black; fill: black; } .arrowheadSeq, .arrowheadStartChoice, .arrowheadAfterChoice, .arrowheadStartRepGroup, .arrowheadEndRepGroup, .arrowheadRev { stroke: none; fill: none; } rect { fill: none; stroke: none; } rect.fragref,rect.syntaxfragref { fill: none; stroke: black; } text { fill: #000000; fill-opacity: 1; font-family: IBM Plex Sans,Arial Unicode MS,Arial,Helvetica; font-style: normal; font-weight: normal; font-size: 8pt; stroke: #000000; stroke-width: 0.1; } text.var, text.syntaxvar {font-style:italic;} delete table mapping

Syntax

```
delete table mapping;
```

Parameters

None. **Required role**

- Administrator
- System administrator

Result

This command returns a value of 0 if the command was successful and a negative value if the command fails. When verbose mode is enabled, the command prints success message CDC1408. **Sample**

Selects a table mapping and deletes it.

```
connect server username user1 password password1;

connect datastore name DS1 context source;

connect datastore name DS2 context target;

select subscription name SUB1;

select table mapping sourceSchema USER1 sourceTable TABLE1;

list table mappings;

delete table mapping;

disconnect server;

exit;
```

Parent topic: Table mapping commands

export rule set

Use this command to export a definition of the rule set in context into an XML file. The XML file can be loaded using the import subscription command or using the import wizard in Management Console. Export requires additional connections to the source and target datastores and cannot run when multiuser configuration is enabled. Use Management Console to export when multiuser configuration is in use.

Syntax diagram

.arrow, .syntaxarrow { fill: none; stroke: black; } .arrowheadStartEnd, .arrowheadRepSep, .arrowheadRepSepReturn { stroke: black; fill: black; } .arrowheadSeq, .arrowheadStartChoice, .arrowheadAfterChoice, .arrowheadStartRepGroup, .arrowheadEndRepGroup, .arrowheadRev { stroke: none; fill: none; } rect { fill: none; stroke: none; } rect.fragref,rect.syntaxfragref { fill: none; stroke: black; } text { fill: #000000; fill-opacity: 1; font-family: IBM Plex Sans,Arial Unicode MS,Arial,Helvetica; font-style: normal; font-weight: normal; font-size: 8pt; stroke: #000000; stroke-width: 0.1; } text.var, text.syntaxvar {font-style:italic;} export rule set ruleNamenamefilenamefile

Syntax

```
export rule set [ruleName name] [filename file];
```

Parameters

- [ruleName name]

- Specifies the name of the rule set. . If a name is not provided, the rule set that is currently identified as the context will be used. To view the current context, use the show context command.

- [filename file]

- Specifies the fully qualified output XML file name for the subscription export.
- Use double quotes to surround the file name. .xml will be appended to the file name, if it does not end with the XML extension. If file name is not provided, the output file name will default to subscription's name.

Required role

- Administrator
- System administrator

Result

This command returns a value of 0 if the command was successful and a negative value if the command fails. When verbose mode is enabled, the command prints success message CDC1331.

Sample

Selects a rule set and exports it to file.

```
connect server username user1 password password1;
connect datastore name DS1 context source;
connect datastore name DS2 context target;
select subscription name SUB1;
select rule set rulename RULE1;
export rule set filename c:\\files\\sub1.xml;
disconnect server;
exit;
```

Parent topic: Table mapping commands

export table mapping

Use this command to export a definition of the table mapping in context into an XML file. The XML file can be loaded using the import subscription command or using the import wizard in Management Console. Export requires additional connections to the source and target datastores and cannot run when multiuser configuration is enabled. Use Management Console to export when multiuser configuration is in use.

Syntax diagram

.arrow, .syntaxarrow { fill: none; stroke: black; } .arrowheadStartEnd, .arrowheadRepSep, .arrowheadRepSepReturn { stroke: black; fill: black; } .arrowheadSeq, .arrowheadStartChoice, .arrowheadAfterChoice, .arrowheadStartRepGroup, .arrowheadEndRepGroup, .arrowheadRev { stroke: none; fill: none; } rect { fill: none; stroke: none; } rect.fragref,rect.syntaxfragref { fill: none; stroke: black; } text { fill: #000000; fill-opacity: 1; font-family: IBM Plex Sans,Arial Unicode MS,Arial,Helvetica; font-style: normal; font-weight: normal; font-size: 8pt; stroke: #000000; stroke-width: 0.1; } text.var, text.syntaxvar {font-style:italic;} export table mapping filenamename

Syntax

```
export table mapping [filename name];
```

Parameters

- [filename name]

- Specifies the fully qualified output XML file name for the subscription export. Use double quotes to surround the file name. .xml will be appended to the file name, if it does not end with the XML extension. If file name is not provided, the output file name will default to subscription's name.

Required role

- Administrator
- System administrator

Result

This command returns a value of 0 if the command was successful and a negative value if the command fails. When verbose mode is enabled, the command prints success message CDC1330.

Sample

Selects a table mapping and exports it to file.

```
connect server username user1 password password1;
connect datastore name DS1 context source;
connect datastore name DS2 context target;
select subscription name SUB1;
select table mapping sourceTable TABLE_1;
export table mapping filename c:\\files\\sub1.xml;
disconnect server;
exit;
```

Parent topic: Table mapping commands

flag refresh

Use this command to flag a table in a subscription for refresh

Syntax diagram

Syntax 1: Flags a table for refresh .arrow, .syntaxarrow { fill: none; stroke: black; } .arrowheadStartEnd, .arrowheadRepSep, .arrowheadRepSepReturn { stroke: black; fill: black; } .arrowheadSeq, .arrowheadStartChoice, .arrowheadAfterChoice, .arrowheadStartRepGroup, .arrowheadEndRepGroup, .arrowheadRev { stroke: none; fill: none; } rect { fill: none; stroke: none; } rect.fragref,rect.syntaxfragref { fill: none; stroke: black; } text { fill: #000000; fill-opacity: 1; font-family: IBM Plex Sans,Arial Unicode MS,Arial,Helvetica; font-style: normal; font-weight: normal; font-size: 8pt; stroke: #000000; stroke-width: 0.1; } text.var, text.syntaxvar {font-style:italic;} flag refreshtypestandarddifferentialdifferentialWithLogdifferentialLogOnly enableSubsetfalse/notrue/yessourcesqlsourcetargetsqltarget

Syntax 2: Flags the specified rule table for refresh. .arrow, .syntaxarrow { fill: none; stroke: black; } .arrowheadStartEnd, .arrowheadRepSep, .arrowheadRepSepReturn { stroke: black; fill: black; } .arrowheadSeq, .arrowheadStartChoice, .arrowheadAfterChoice, .arrowheadStartRepGroup, .arrowheadEndRepGroup, .arrowheadRev { stroke: none; fill: none; } rect { fill: none; stroke: none; } rect.fragref,rect.syntaxfragref { fill: none; stroke: black; } text { fill: #000000; fill-opacity: 1; font-family: IBM Plex Sans,Arial Unicode MS,Arial,Helvetica; font-style: normal; font-weight: normal; font-size: 8pt; stroke: #000000; stroke-width: 0.1; } text.var, text.syntaxvar {font-style:italic;} flag refreshschemaschematablename

Syntax

Syntax 1: Flags a table for refresh
```
flag refresh [type <refresh>] [enableSubset <flag>] [sourcesql <source>]
[targetsql <target>];
```
Syntax 2: Flags the specified rule table for refresh.
```
flag refresh schema <schema> table <name>;
```

Parameters

Syntax 1: Flags a table for refresh

- [type <refresh>]

- pecifies the type of refresh to be performed.Valid values:
  - **standard**—Specifies a standard refresh.
  - **differential**—Specifies a differential refresh, where supported by the datastore.
  - **differentialWithLog**—Specifies a differential refresh, where supported by the datastore, and also creates a log table to track all changes during the refresh.
  - **differentialLogOnly**——Creates and populates a log table to identify all differences between the source and target tables, where supported by the datastore.

  Default value: standard

- [enableSubset <flag>]

- Specifies that SQL WHERE clauses will be used to perform a row subset refresh, where supported by the datastore.Valid values:

- **true/yes**—Specifies that SQL WHERE clauses will be used. Either yes or true are valid values.
- **false/no**—Specifies that SQL WHERE clauses will not be used. Either no or false are valid values.

Default value: false/no

- [sourcesql <source>]

- Specifies the SQL WHERE clause for the selected source table for a row subset refresh.

- [targetsql <target>]

- Specifies the SQL WHERE clause for the selected target table for row subset refresh.

Syntax 2: Flags the specified rule table for refresh.

- schema <schema>

- Specifies the schema name of the table.

- table <name>

- Specifies the name of the table.

Required role

- Operator
- Administrator
- System administrator

Result

This command returns a value of 0 if the command was successful and a negative value if the command fails. When verbose mode is enabled, the command prints success message CDC1404.

Sample

Selects a table mapping and flags it for refresh.

```
connect server username user1 password password1;
connect datastore name DS1 context source;
connect datastore name DS2 context target;
select subscription name SUB1;
select table mapping sourceSchema USER1 sourceTable TABLE1;
flag refresh;
list table mappings;
disconnect server;
exit;
```

Flags a rule table for refresh.

```
connect server username user1 password password1;
connect datastore name DS1 context source;
connect datastore name DS2 context target;
select subscription name SUB1;
flag refresh schema USER1 table TABLE1;
list rule set tables;
disconnect server;
exit;
```

Parent topic: Table mapping commands

list refresh order

Use this command to list the refresh order for the source tables in a subscription.

Syntax diagram

.arrow, .syntaxarrow { fill: none; stroke: black; } .arrowheadStartEnd, .arrowheadRepSep, .arrowheadRepSepReturn { stroke: black; fill: black; } .arrowheadSeq, .arrowheadStartChoice, .arrowheadAfterChoice, .arrowheadStartRepGroup, .arrowheadEndRepGroup, .arrowheadRev { stroke: none; fill: none; } rect { fill: none; stroke: none; } rect.fragref,rect.syntaxfragref { fill: none; stroke: black; } text { fill: #000000; fill-opacity: 1; font-family: IBM Plex Sans,Arial Unicode MS,Arial,Helvetica; font-style: normal; font-weight: normal; font-size: 8pt; stroke: #000000; stroke-width: 0.1; } text.var, text.syntaxvar {font-style:italic;} list refresh ordernamesubscription

Syntax

```
list refresh order [name <subscription>];
```

Parameters

- [name <subscription>]

- Specifies the name of the subscription.If a name is not provided, the subscription that is currently identified as the context will be used. To view the current context, use the show context command.

Required role

- Administrator
- System administrator

Result

A ResultStringTable of tables in the subscription with their refresh group and sequence number.

```
GROUP            SOURCE TABLE     GROUP ORDER      SEQUENCE
---------------- ---------------- ---------------- ----------------
Group 1          CDC.TABLE_2      1                0
Group 1          CDC.TABLE_4      1                1
Ungrouped        CDC.TABLE_1      0                0
Ungrouped        CDC.TABLE_3      0                0
Ungrouped        CDC.TABLE_5      0                0
```

Sample

Lists the refresh order of the tables in a subscription.

```
connect server username user1 password password1;
connect datastore name DS1 context source;
connect datastore name DS2 context target;
select subscription name SUB1;
list refresh order;
disconnect server;
exit;
```

Parent topic: Table mapping commands

list rule set tables

Use this command to view a list of all tables that match the criteria for all rule sets defined for this subscription.

Syntax diagram

.arrow, .syntaxarrow { fill: none; stroke: black; } .arrowheadStartEnd, .arrowheadRepSep, .arrowheadRepSepReturn { stroke: black; fill: black; } .arrowheadSeq, .arrowheadStartChoice, .arrowheadAfterChoice, .arrowheadStartRepGroup, .arrowheadEndRepGroup, .arrowheadRev { stroke: none; fill: none; } rect { fill: none; stroke: none; } rect.fragref,rect.syntaxfragref { fill: none; stroke: black; } text { fill: #000000; fill-opacity: 1; font-family: IBM Plex Sans,Arial Unicode MS,Arial,Helvetica; font-style: normal; font-weight: normal; font-size: 8pt; stroke: #000000; stroke-width: 0.1; } text.var, text.syntaxvar {font-style:italic;} list rule set tablesname subscription

Syntax

```
list rule set tables [name subscription];
```

Parameters

- [name subscription]

- Specifies the name of the subscription.If a name is not provided, the subscription that is currently identified as the context will be used. To view the current context, use the show context command.

Required role

- Administrator
- System administrator

Result

A ResultStringTable of tables that match the criteria for all rule sets defined for this subscription.
```
SCHEMA          TABLE NAME      STATUS          STRUCTURE ONLY
```

Sample

List all tables that match the criteria for all rule sets defined for this subscription.

```
connect server username user1 password password1;

connect datastore name DS1 context source;

connect datastore name DS2 context target;

select subscription name SUB1;

list rule set tables;

disconnect server;

exit;
```

Parent topic: Table mapping commands

list rule sets

Use this command to view a list of rule sets in the subscription.

Syntax diagram

.arrow, .syntaxarrow { fill: none; stroke: black; } .arrowheadStartEnd, .arrowheadRepSep, .arrowheadRepSepReturn { stroke: black; fill: black; } .arrowheadSeq, .arrowheadStartChoice, .arrowheadAfterChoice, .arrowheadStartRepGroup, .arrowheadEndRepGroup, .arrowheadRev { stroke: none; fill: none; } rect { fill: none; stroke: none; } rect.fragref,rect.syntaxfragref { fill: none; stroke: black; } text { fill: #000000; fill-opacity: 1; font-family: IBM Plex Sans,Arial Unicode MS,Arial,Helvetica; font-style: normal; font-weight: normal; font-size: 8pt; stroke: #000000; stroke-width: 0.1; } text.var, text.syntaxvar {font-style:italic;} list rule setsnamesubscription

Syntax

```
list rule sets [name subscription];
```

Parameters

- [name subscription]

- Specifies the name of the subscription.If a name is not provided, the subscription that is currently identified as the context will be used. To view the current context, use the show context command.

Required role

- Operator
- Administrator
- System administrator

Result

A ResultStringTable of list of rule sets in the subscription.

```
RULE SET NAME    SCHEMA
INCLUDE TABLES   EXCLUDE TABLES   STRUCTURE ONLY   CONTEXT
  RULE1            USER1            *                B                 Yes
RULE SET NAME    SCHEMA           INCLUDE TABLES   EXCLUDE TABLES   STRUCTURE ONLY   CONTEXT
  RULE1            USER1            *                B                 Yes
```

Sample

Lists the rule sets in a subscription.

```
connect server username user1 password password1;
connect datastore name DS1 context source;
connect datastore name DS2 context target;
list rule sets name SUB1;
disconnect server;
exit;
```

Parent topic: Table mapping commands

list table mappings

Use this command to list the table mappings in a subscription.Starting mirroring or refresh can change the status of a table mapping. Use the reload parameter to refresh the list of table mappings.

Syntax diagram

.arrow, .syntaxarrow { fill: none; stroke: black; } .arrowheadStartEnd, .arrowheadRepSep, .arrowheadRepSepReturn { stroke: black; fill: black; } .arrowheadSeq, .arrowheadStartChoice, .arrowheadAfterChoice, .arrowheadStartRepGroup, .arrowheadEndRepGroup, .arrowheadRev { stroke: none; fill: none; } rect { fill: none; stroke: none; } rect.fragref,rect.syntaxfragref { fill: none; stroke: black; } text { fill: #000000; fill-opacity: 1; font-family: IBM Plex Sans,Arial Unicode MS,Arial,Helvetica; font-style: normal; font-weight: normal; font-size: 8pt; stroke: #000000; stroke-width: 0.1; } text.var, text.syntaxvar {font-style:italic;} list table mappingsnamesubscriptionmethodmirrorrefreshstatusactive refreshparkedincompleteOnlyfalse/notrue/yesreloadfalse/notrue/yes

Syntax

```
list table mappings [name <subscription>] [method <type>] [status <state>]

[incompleteOnly <unmapped>] [reload <state>];
```

Parameters

- **[name <subscription>]**
  - Specifies the name of the subscription.If a name is not provided, the subscription that is currently identified as the context will be used. To view the current context, use the show context command.
- **[method <type>]**
  - Optionally filters table mapping by their replication method.Valid values:
    - **mirror**—Displays only the table mappings that have a replication method of mirroring.
    - **refresh**—Displays only the table mappings that have a replication method of refresh.
- **[status <state>]**
  - Optionally filters table mappings by their replication status.Valid values:
    - **active**—Displays only the table mappings that have a replication status of active.
    - **refresh**—Displays only the table mappings that have a replication status of refresh.
    - **parked**—Displays only the table mappings that have a replication status of parked.
- **[incompleteOnly <unmapped>]**
  - Optionally displays table mappings that are not mapped to a target.Valid values:
    - **true/yes**—Displays only the table mappings that are not mapped to a target. Either yes or true are valid values.
    - **false/no**—Displays all table mappings, whether they are mapped to a target or not. Either no or false are valid values.

Default value: false/no

- [reload <state>]

- Optionally reloads table mappings.Valid values:
  - **true/yes**—Reloads table mapping information from the datastores. Either yes or true are valid values.
  - **false/no**—Returns existing table mapping definitions and only requests them from the datastores when they haven't been previously loaded. Either no or false are valid values.

  Default value: false/no

Required role

- Operator
- Administrator
- System administrator

Result

A ResultStringTable of the table mappings for the subscription.Table mappings for subscription SUB1.

```
SOURCE TABLE   TARGET TABLE   MAPPING TYPE   METHOD   STATUS    CONTEXT
-------------- -------------- -------------- -------- --------- ---------
CDC.TABLE_1    CDC.TABLE_1    Standard       Mirror   Refresh
CDC.TABLE_2    CDC.TABLE_2    Standard       Mirror   Refresh
CDC.TABLE_3    CDC.TABLE_3    Standard       Mirror   Refresh
CDC.TABLE_4    CDC.TABLE_4    Standard       Mirror   Refresh
CDC.TABLE_5    CDC.TABLE_6    Standard       Mirror   Refresh
```

Sample

Lists the table mappings in a subscription.

```
connect server username user1 password password1;
connect datastore name DS1 context source;
connect datastore name DS2 context target;
list table mappings name SUB1;
disconnect server;
exit;
```

Lists tables flagged for refresh in the subscription.

```
connect server username user1 password password1;
connect datastore name DS1 context source;
connect datastore name DS2 context target;
list table mappings name SUB1 status refresh;
disconnect server;
exit;
```

Parent topic: Table mapping commands

mark capture point

Use this command to set the bookmark to the head-of-log for a source table at the point in time when the command was issued.
Marking the capture point sets all records in the table as having been processed and prompts capture to begin mirroring with any new changes.

Syntax diagram

Syntax 1: Mark a capture pointNote: The table must be first selected as the current context.
.arrow, .syntaxarrow { fill: none; stroke: black; } .arrowheadStartEnd, .arrowheadRepSep, .arrowheadRepSepReturn { stroke: black; fill: black; } .arrowheadSeq, .arrowheadStartChoice, .arrowheadAfterChoice, .arrowheadStartRepGroup, .arrowheadEndRepGroup, .arrowheadRev { stroke: none; fill: none; } rect { fill: none; stroke: none; } rect.fragref,rect.syntaxfragref { fill: none; stroke: black; } text { fill: #000000; fill-opacity: 1; font-family: IBM Plex Sans,Arial Unicode MS,Arial,Helvetica; font-style: normal; font-weight: normal; font-size: 8pt; stroke: #000000; stroke-width: 0.1; } text.var, text.syntaxvar {font-style:italic;} mark capture point

Syntax 2: Mark a table capture point on the rule table before mirroring. .arrow, .syntaxarrow { fill: none; stroke: black; } .arrowheadStartEnd, .arrowheadRepSep, .arrowheadRepSepReturn { stroke: black; fill: black; } .arrowheadSeq, .arrowheadStartChoice, .arrowheadAfterChoice, .arrowheadStartRepGroup, .arrowheadEndRepGroup, .arrowheadRev { stroke: none; fill: none; } rect { fill: none; stroke: none; } rect.fragref,rect.syntaxfragref { fill: none; stroke: black; } text { fill: #000000; fill-opacity: 1; font-family: IBM Plex Sans,Arial Unicode MS,Arial,Helvetica; font-style: normal; font-weight: normal; font-size: 8pt; stroke: #000000; stroke-width: 0.1; } text.var, text.syntaxvar {font-style:italic;} mark capture pointschemaschema tablename

Syntax

Syntax 1: Mark a capture point`mark capture point;`
Syntax 2: Mark a table capture point on the rule table before mirroring.`mark capture point schema schema table name;`

Parameters

Syntax 1: Mark a capture pointNone.
Syntax 2: Mark a table capture point on the rule table before mirroring.

- schema schema
    - Specifies the name of the schema.

- table name
    - Specifies the name of the table.

Required role

- Operator
- Administrator
- System administrator

Result

This command returns a value of 0 if the command was successful and a negative value if the command fails. When verbose mode is enabled, the command prints success message CDC1403.

Sample

Marks the capture point for a table in the selected subscription.

```
connect server username user1 password password1;
connect datastore name DS1 context source;
connect datastore name DS2 context target;
select subscription name SUB1;
select table mapping
sourceSchema USER1 sourceTable TABLE_1
targetSchema USER1 targetTable TABLE_1;
mark capture point;
list table mappings;
disconnect server;
exit;
```

Parent topic: Table mapping commands

modify refresh order

Use this command to modify the refresh order for the source tables in a subscription.Use the list refresh order command to view the current refresh order for a subscription.

Syntax diagram

.arrow, .syntaxarrow { fill: none; stroke: black; } .arrowheadStartEnd, .arrowheadRepSep, .arrowheadRepSepReturn { stroke: black; fill: black; } .arrowheadSeq, .arrowheadStartChoice, .arrowheadAfterChoice, .arrowheadStartRepGroup, .arrowheadEndRepGroup, .arrowheadRev { stroke: none; fill: none; } rect { fill: none; stroke: none; } rect.fragref,rect.syntaxfragref { fill: none; stroke: black; } text { fill: #000000; fill-opacity: 1; font-family: IBM Plex Sans,Arial Unicode MS,Arial,Helvetica; font-style: normal; font-weight: normal; font-size: 8pt; stroke: #000000; stroke-width: 0.1; } text.var, text.syntaxvar {font-style:italic;} modify refresh ordernamesubscriptiondatabase databaseschema schematable tablegroup 0numbersequence 0sequence

Syntax

```
modify refresh order [name <subscription>] [database  <database>] schema  <schema>

table  <table> group  <number> sequence  <sequence>;
```

Parameters

- **[name <subscription>]**
  - Specifies the name of the subscription.If a name is not provided, the subscription that is currently identified as the context will be used. To view the current context, use the show context command.
- **[database <database>]**
  - Specifies the database for the source table.This parameter can be specified more than once.
- **schema <schema>**
  - Specifies the schema for the source table.
- **table <table>**
  - Specifies the name of the source table.This parameter can be specified more than once.
- **group <number>**
  - Specifies the group number for the table. Use 0 for ungrouped tables, or values greater than zero to specify a group number.This parameter can be specified more than once.
    Default value: 0
- **sequence <sequence>**
  - Specifies the sequence number for the table within a group. Group 0 tables are not sequenced and must use sequence number 0.This parameter can be specified more than once.
    Default value: 0

Required role

- Administrator
- System administrator

Result

This command returns a value of 0 if the command was successful and a negative value if the command fails. When verbose mode is enabled, the command prints success message CDC1310.

Sample

Selects two tables to be refreshed before all other tables in the subscription by putting them in a group.

```
connect server username user1 password password1;

connect datastore name DS1 context source;

connect datastore name DS2 context target;

select subscription name SUB1;

modify refresh order

  schema USER1 table TABLE_2 group 1 sequence 0

  schema USER1 table TABLE_4 group 1 sequence 1;

list refresh order;

disconnect server;

exit;
```

Parent topic: Table mapping commands

modify rule set

Use this command to rename a rule set or modifies the properties of an existing rule set. If you use this command to modify rule patterns, you should give the complete criteria as the input parameters, since all existing patterns for both table inclusion and exclusion will be overwritten.

Syntax diagram

.arrow, .syntaxarrow { fill: none; stroke: black; } .arrowheadStartEnd, .arrowheadRepSep, .arrowheadRepSepReturn { stroke: black; fill: black; } .arrowheadSeq, .arrowheadStartChoice, .arrowheadAfterChoice, .arrowheadStartRepGroup, .arrowheadEndRepGroup, .arrowheadRev { stroke: none; fill: none; } rect { fill: none; stroke: none; } rect.fragref,rect.syntaxfragref { fill: none; stroke: black; } text { fill: #000000; fill-opacity: 1; font-family: IBM Plex Sans,Arial Unicode MS,Arial,Helvetica; font-style: normal; font-weight: normal; font-size: 8pt; stroke: #000000; stroke-width: 0.1; } text.var, text.syntaxvar {font-style:italic;} modify rule setruleNamesetnewNamenewschemanamestructureOnly false/notrue/yesincludeAllfalse/notrue/yesincludePatternincludeincludeMatchexact startsWithcontainsendsWithexcludePatternexcludeexcludeMatchexactstartsWith containsendsWith

Syntax

```
modify rule set [ruleName <set>] [newName <new>] [schema <name>] [structureOnly
<data>] [includeAll <allow>] [includePattern <include>] [includeMatch <type>]
[excludePattern <exclude>] [excludeMatch <match>];
```

Parameters

- [ruleName <set>]
- Specifies the name of the rule set.Rule set names must be unique within a subscription.

- [newName <new>]
- Specifies the new name for the rule set. Rule set names must be unique within a subscription.

- [schema <name>]
- Specifies the schema for the new rule set.

- [structureOnly <data>]
- Specifies whether to replicate only structural (DDL), not data (DML), changes for the tables included in the rule.Valid values:
  - **true/yes**—Only structural (DDL) changes are replicated. Either yes or true are valid values.
  - **false/no**—Both structural (DDL) and data (DML) changes are replicated. Either no or false are valid values.

  Default value: false/no

- [includeAll <allow>]
- Specifies whether to allow all tables to be considered for inclusion in the rule set.Valid values:
  - **true/yes**—All tables are to be considered for inclusion in the rule set. Either yes or true are valid values.

- **false/no**—Not all tables are to be considered for inclusion in the rule set. Either no or false are valid values.

  Default value: false/no

- [includePattern <include>]

- Specifies the criteria by which tables will be judged for inclusion from the rule set. Patterns are case-sensitive.This parameter can be specified more than once.

- includeMatch <type>]

- Specifies the pattern matching type used for table inclusion.This parameter can be specified more than once.

  Valid values:
  - exact
  - startsWith
  - contains
  - endsWith

- [excludePattern <exclude>]

- Specifies the criteria by which tables will be judged for exclusion in from the rule set. Patterns are case-sensitive.This parameter can be specified more than once.

- [excludeMatch <match>]

- Specifies the pattern matching type used for table exclusion.This parameter can be specified more than once.

  Valid values:
  - exact
  - startsWith
  - contains
  - endsWith

Required role

- Administrator
- System administrator

Result

This command returns a value of 0 if the command was successful and a negative value if the command fails. When verbose mode is enabled, the command prints success message CDC1703.

Sample

Sample 1: Selects a rule set within a subscription, modifies the rule name and schema for the rule set.

```
connect server username user1 password password1;
connect datastore name DS1 context source;
connect datastore name DS2 context target;
select subscription name SUB1;
select rule set ruleName RULE1;
modify rule set newName RULE2 schema USER2;
list rule sets;
disconnect server;
exit;
```

Sample 2: Modifies the table inclusion and exclusion criteria of the rule set.

```
connect server username user1 password password1;
connect datastore name DS1 context source;
connect datastore name DS2 context target;
```

```
select subscription name SUB1;
modify rule set ruleName RULE1
 includeAll yes
 excludePattern X excludeMatch exact
 excludePattern Y excludeMatch exact;
list rule sets;
disconnect server;
exit;
```

Parent topic: Table mapping commands

modify table mapping

Use this command to change the replication method of a single table mapping. The table mapping must be first selected as the current context.

Syntax diagram

Syntax 1: Replication Method

.arrow, .syntaxarrow { fill: none; stroke: black; } .arrowheadStartEnd, .arrowheadRepSep, .arrowheadRepSepReturn { stroke: black; fill: black; } .arrowheadSeq, .arrowheadStartChoice, .arrowheadAfterChoice, .arrowheadStartRepGroup, .arrowheadEndRepGroup, .arrowheadRev { stroke: none; fill: none; } rect { fill: none; stroke: none; } rect.fragref,rect.syntaxfragref { fill: none; stroke: black; } text { fill: #000000; fill-opacity: 1; font-family: IBM Plex Sans,Arial Unicode MS,Arial,Helvetica; font-style: normal; font-weight: normal; font-size: 8pt; stroke: #000000; stroke-width: 0.1; } text.var, text.syntaxvar {font-style:italic;} modify table mappingmethodmirrorrefreshpreventRecursion true/yes false/no

Syntax 2: Target Index

.arrow, .syntaxarrow { fill: none; stroke: black; } .arrowheadStartEnd, .arrowheadRepSep, .arrowheadRepSepReturn { stroke: black; fill: black; } .arrowheadSeq, .arrowheadStartChoice, .arrowheadAfterChoice, .arrowheadStartRepGroup, .arrowheadEndRepGroup, .arrowheadRev { stroke: none; fill: none; } rect { fill: none; stroke: none; } rect.fragref,rect.syntaxfragref { fill: none; stroke: black; } text { fill: #000000; fill-opacity: 1; font-family: IBM Plex Sans,Arial Unicode MS,Arial,Helvetica; font-style: normal; font-weight: normal; font-size: 8pt; stroke: #000000; stroke-width: 0.1; } text.var, text.syntaxvar {font-style:italic;} modify table mappingmethodmirrorrefreshtargetIndexModeautoDetect indexallSearchablecolumnstargetIndexNamenametargetIndexColumnscolumn

Syntax 3: Row Filtering

.arrow, .syntaxarrow { fill: none; stroke: black; } .arrowheadStartEnd, .arrowheadRepSep, .arrowheadRepSepReturn { stroke: black; fill: black; } .arrowheadSeq, .arrowheadStartChoice, .arrowheadAfterChoice, .arrowheadStartRepGroup, .arrowheadEndRepGroup, .arrowheadRev { stroke: none; fill: none; } rect { fill: none; stroke: none; } rect.fragref,rect.syntaxfragref { fill: none; stroke: black; } text { fill: #000000; fill-opacity: 1; font-family: IBM Plex Sans,Arial Unicode MS,Arial,Helvetica; font-style: normal; font-weight: normal; font-size: 8pt; stroke: #000000; stroke-width: 0.1; } text.var, text.syntaxvar {font-style:italic;} modify table mappingrowFilterfilterselecttrue/yesfalse/no

Syntax 4: Conflict Detection and Resolution

.arrow, .syntaxarrow { fill: none; stroke: black; } .arrowheadStartEnd, .arrowheadRepSep, .arrowheadRepSepReturn { stroke: black; fill: black; } .arrowheadSeq, .arrowheadStartChoice, .arrowheadAfterChoice, .arrowheadStartRepGroup, .arrowheadEndRepGroup, .arrowheadRev { stroke: none; fill: none; } rect { fill: none; stroke: none; } rect.fragref,rect.syntaxfragref { fill:

none; stroke: black; } text { fill: #000000; fill-opacity: 1; font-family: IBM Plex Sans,Arial Unicode MS,Arial,Helvetica; font-style: normal; font-weight: normal; font-size: 8pt; stroke: #000000; stroke-width: 0.1; } text.var, text.syntaxvar {font-style:italic;} modify table mappingdetectionColumnslistresolutionMethodmethodvalue comparison

Syntax 5: InfoSphere® DataStage® targets
.arrow, .syntaxarrow { fill: none; stroke: black; } .arrowheadStartEnd, .arrowheadRepSep, .arrowheadRepSepReturn { stroke: black; fill: black; } .arrowheadSeq, .arrowheadStartChoice, .arrowheadAfterChoice, .arrowheadStartRepGroup, .arrowheadEndRepGroup, .arrowheadRev { stroke: none; fill: none; } rect { fill: none; stroke: none; } rect.fragref,rect.syntaxfragref { fill: none; stroke: black; } text { fill: #000000; fill-opacity: 1; font-family: IBM Plex Sans,Arial Unicode MS,Arial,Helvetica; font-style: normal; font-weight: normal; font-size: 8pt; stroke: #000000; stroke-width: 0.1; } text.var, text.syntaxvar {font-style:italic;} modify table mappingrecordFormatsingleRecordmultiRecorddirectory pathcustomFormattercustom

Syntax

Syntax 1: Replication Method

```
modify table mapping method <type> [preventRecursion <recursion>];
```

Syntax 2: Target Index

```
modify table mapping [targetIndexMode <mode>] [targetIndexName <name>]
 [targetIndexColumns <column>];
```

Syntax 3: Row Filtering

```
modify table mapping [rowFilter <filter>] [select <match>];
```

Syntax 4: Conflict Detection and Resolution

```
modify table mapping [detectionColumns <list>] [resolutionMethod <method>]
[value <comparison>];
```

Syntax 5 - InfoSphere DataStage targets

```
modify table mapping [recordFormat <format>] [directory <path>]
[customFormatter <custom>];
```

Parameters

Syntax 1: Replication Method

- method <type>

- Specifies the replication method for the table mapping.Valid values:
  - **mirror**—Immediately replicates changes made to the source table to the target table or accumulate source table changes and replicate at a later time.
  - **refresh**—Replicates a snapshot of the source table to the target table.

- [preventRecursion <recursion>]

- Specifies whether to prevent from replicating changes back to the source database when a subscription is configured for bidirectional replication.Valid values:
  - **true/yes**—Specifies that recursion will be prevented. Either yes or true are valid values.

    - **false/no**—Specifies that recursion will not be prevented. Either no or false are valid values.

Syntax 2: Target Index

- [targetIndexMode <mode>]
  - Specifies the mode used to locate a row in the target table.Valid values:
    - **autoDetect**—Specifies that an index will be selected automatically.
    - **index**—Specifies that an index will be used to identify a row in the target table.
    - **allSearchable**—Specifies that all target columns will be searched to identify which columns are suitable to uniquely identify rows. The results of the search are used to build a WHERE clause which uniquely identifies the row on the target column to apply data.
    - **columns**—Specifies that a column will be used to identify a row in the target table.

- [targetIndexName <name>]
  - Specifies the target index name when the target index mode is set to index.

- [targetIndexColumns <column>]
  - Specifies a comma-delimited list of columns used to identify a row when the targetIndexMode is set to `column`.

Syntax 3: Row Filtering

- [rowFilter <filter>]
  - Specifies the row-filtering expression. Row filter will be cleared if expression is set to empty string.

- [select <match>]
  - Specifies whether the filter selects or omits rows that match the expression. Valid values:
    - **true/yes**—Specifies that the filter selects rows that match the expression. Either yes or true are valid values.
    - **false/no**—Specifies that the filter omits rows that match the expression. Either no or false are valid values.

Syntax 4: Conflict Detection and Resolution

- [detectionColumns <list>]
  - Specifies a comma-delimited list of columns used for conflict detection.

- [resolutionMethod <method>]
  - Specifies the method used to resolve conflicts.Valid values:
    - **none**—Specifies that conflict detection resolution is not enabled.
    - **sourceWins**—Specifies that the source will win conflicts.
    - **targetWins**—Specifies that the target will win conflicts.
    - **largestValueWins**—Specifies that the largest value will win conflicts.
    - **smallestValueWins**—Specifies that the smallest value will win conflicts.
    - **userExit**—Specifies that conflicts will be resolved with a user exit.

- [value <comparison>]
  - Specifies the comparison column name if conflict resolution method is largestValueWins or smallestValueWins, or specifies the user exit name (with path) if conflict resolution method is userExit.

Syntax 5 - InfoSphere DataStage targets

- [recordFormat <format>]

- Specifies the record format for InfoSphere DataStage mappings.Valid values:
  - **singleRecord**—Specifies that an update operation is sent as a single row.
  - **multiRecord**—Specifies that an update operation is sent as two rows.

  Default value: singleRecord

- [directory <path>]

- Specifies the directory for files in a InfoSphere DataStage flat file mapping.

- [customFormatter <custom>]

- Specifies a custom formatter for InfoSphere DataStage flat file output.

Required role

- Administrator
- System administrator

Result

This command returns a value of 0 if the command was successful and a negative value if the command fails. When verbose mode is enabled, the command prints success message CDC1405.

Sample

Specifies that the selected table will be set to refresh only and not be mirrored.

```
connect server username user1 password password1;

connect datastore name DS1 context source;

connect datastore name DS2 context target;

select subscription name SUB1;

select table mapping sourceSchema USER1 sourceTable TABLE_1;

modify table mapping method refresh;

show table mapping;

disconnect server;

exit;
```

Changes the detection of a row on the target table to use all searchable columns.

```
connect server username user1 password password1;

connect datastore name DS1 context source;

connect datastore name DS2 context target;

select subscription name SUB1;

select table mapping sourceSchema USER1 sourceTable TABLE_1;

modify table mapping targetIndexMode allSearchable;

show table mapping;

disconnect server;

exit;
```

Parent topic: Table mapping commands

park table mapping

Use this command to park a table from replication. The table mapping must be first selected as the current context.

Syntax diagram

.arrow, .syntaxarrow { fill: none; stroke: black; } .arrowheadStartEnd, .arrowheadRepSep, .arrowheadRepSepReturn { stroke: black; fill: black; } .arrowheadSeq, .arrowheadStartChoice, .arrowheadAfterChoice, .arrowheadStartRepGroup, .arrowheadEndRepGroup, .arrowheadRev { stroke: none; fill: none; } rect { fill: none; stroke: none; } rect.fragref,rect.syntaxfragref { fill: none; stroke: black; } text { fill: #000000; fill-opacity: 1; font-family: IBM Plex Sans,Arial Unicode MS,Arial,Helvetica; font-style: normal; font-weight: normal; font-size: 8pt; stroke: #000000; stroke-width: 0.1; } text.var, text.syntaxvar {font-style:italic;} park table mapping

Syntax

```
park table mapping;
```

Parameters

None. **Required role**

- Operator
- Administrator
- System administrator

Result

This command returns a value of 0 if the command was successful and a negative value if the command fails. When verbose mode is enabled, the command prints success message CDC1406. **Sample**

Parks the selected table mapping.

```
connect server username user1 password password1;

connect datastore name DS1 context source;

connect datastore name DS2 context target;

select subscription name SUB1;

select table mapping

sourceSchema USER1 sourceTable TABLE_1

targetSchema USER1 targetTable TABLE_1;

park table mapping;

list table mappings;

disconnect server;

exit;
```

Parent topic: Table mapping commands

promote rule set

Use this command to promote the specified rule set to a new subscription or to an existing subscription. Promotion requires additional connections to the source and target datastores and cannot run when multiuser configuration is enabled. Use Management Console to promote when multiuser configuration is in use.

Syntax diagram

.arrow, .syntaxarrow { fill: none; stroke: black; } .arrowheadStartEnd, .arrowheadRepSep, .arrowheadRepSepReturn { stroke: black; fill: black; } .arrowheadSeq, .arrowheadStartChoice, .arrowheadAfterChoice, .arrowheadStartRepGroup, .arrowheadEndRepGroup, .arrowheadRev { stroke: none; fill: none; } rect { fill: none; stroke: none; } rect.fragref,rect.syntaxfragref { fill: none; stroke: black; } text { fill: #000000; fill-opacity: 1; font-family: IBM Plex Sans,Arial Unicode MS,Arial,Helvetica; font-style: normal; font-weight: normal; font-size: 8pt; stroke: #000000; stroke-width: 0.1; } text.var, text.syntaxvar {font-style:italic;} promote rule set ruleNamesettypenewexistingReplaceAll existingKeepExtrapromotionNamenamesourceid identifiersourceDatastoresource targetDatastoretargetoldSourceSchemaoldnewSourceSchema new

Syntax

```
promote rule set [ruleName <set>] type <type> [promotionName <name>] [sourceid
<identifier>] [sourceDatastore <source>] [targetDatastore <target>] [oldSourceSchema
<old>] [newSourceSchema <new>];
```

Parameters

- **[ruleName <set>]**
  - Specifies the name of the rule set.If a name is not provided, the rule set that is currently identified as the context will be used. To view the current context, use the show context command.
- **type <type>**
  - Specify whether you want to promote the rule set to a new subscription or to an existing one. Valid values:
    - **new**—Promotes the rule set into a new subscription.
    - **existingReplaceAll**—Promotes the rule set into an existing subscription and replaces all of the mappings and rules sets.
    - **existingKeepExtra**—Promotes the rule set into an existing subscription and replaces only the rule set that was promoted.
- **promotionName <name>]**
  - Specify the name of the subscription into which the rule set will be promoted. If no name is specified, the default value is the subscription name that the rule set was promoted from.
- **[sourceid <identifier>]**
  - Specifies the source identifier for the new subscription on the target datastore.
  - The identifier must be unique on the target.
  - If no identifier is provided, the default value for the Source ID will be the first eight characters of the subscription name.
- **[sourceDatastore <source>]**
  - Specifies the name of the source datastore for the promotion subscription. If a value is not specified, it will default to the datastore that is currently set to

source context.

- [targetDatastore <target>]

- Specifies the name of the target datastore for the promotion subscription. If a value is not specified, it will default to the datastore that is currently set to target context.

- [oldSourceSchema <old>]

- Specifies the name of the schema of the original datastore that contains the original tables in the promoted rule set.

- [newSourceSchema <new>]

- Specifies the name of the schema of the new datastore that contains the new tables to be included in the rule set.

Required role

- Administrator
- System administrator

Result

This command returns a value of 0 if the command was successful and a negative value if the command fails. When verbose mode is enabled, the command prints success message CDC1332.

Sample

Selects a rule set and promotes it to a new subscription.

```
connect server username user1
password password1;
connect datastore name DS1 context source;
connect datastore name DS2 context target;
select subscription name SUB1;
select rule set rulename RULE1;
promote rule set type new promotionname SUB2;
disconnect server;
exit;
```

Parent topic: Table mapping commands

promote table mapping

Promote the table mapping in context to a new subscription or to an existing subscription. Promotion requires additional connections to the source and target datastores and cannot run when multiuser configuration is enabled. Use Management Console to promote when multiuser configuration is in use.

Syntax diagram

.arrow, .syntaxarrow { fill: none; stroke: black; } .arrowheadStartEnd, .arrowheadRepSep, .arrowheadRepSepReturn { stroke: black; fill: black; } .arrowheadSeq, .arrowheadStartChoice, .arrowheadAfterChoice, .arrowheadStartRepGroup, .arrowheadEndRepGroup, .arrowheadRev { stroke: none; fill: none; } rect { fill: none; stroke: none; } rect.fragref,rect.syntaxfragref { fill: none; stroke: black; } text { fill: #000000; fill-opacity: 1; font-family: IBM Plex Sans,Arial Unicode MS,Arial,Helvetica; font-style: normal; font-weight: normal; font-size: 8pt; stroke: #000000; stroke-width: 0.1; } text.var, text.syntaxvar {font-style:italic;} promote table mapping typenewexistingReplaceAllexistingKeepExtra promotionName subscriptionsourceid identifiersourceDatastoresource targetDatastoretargetoldSourceSchemaold sourcenewSourceSchemanew source oldTargetSchemaold targetnewTargetSchema new target

Syntax

```
promote table mapping type <type>

promote table mapping type <promote> [promotionName <subscription>]

[sourceid <identifier>] [sourceDatastore <source>] [targetDatastore <target>];

[oldSourceSchema <old source>] [newSourceSchema <new source>] [oldTargetSchema

<old target>] [newTargetSchema <new target>];
```

Parameters

- **[type <promote>]**
  - Specify whether you want to promote the table mapping to a new subscription or to an existing one. If you are promoting into an existing one, specifies whether to replace all mappings in the existing subscription or only replace the mapping that you are promoting.Valid values:
    - **new**—Specifies that you are promoting to a new subscription.
    - **existingReplaceAll**—Specifies that you are promoting to an existing subscription and all mappings in the existing subscription will be replaced.
    - **existingKeepExtra**—Specifies that you are promoting to an existing subscription and that only the mappings from the imported subscription will be replaced in the existing subscription.
- **[promotionName <subscription>]**
  - Specifies the name of the subscription to promote into.If not specified, the name will default to the name of the subscription that was promoted from.
- **[sourceid <identifier>]**
  - Specifies the source identifier for the new subscription on the target datastore.
  - The identifier must be unique on the target.
  - If no identifier is provided, the default value for the Source ID will be the first eight characters of the subscription name.
- **[sourceDatastore <source>]**

  - Specifies the name of the source datastore for the promotion subscription. If a value is not specified, it will default to the datastore that is currently set to source context.

- [targetDatastore <target>]
  - Specifies the name of the target datastore for the promotion subscription. If a value is not specified, it will default to the datastore that is currently set to target context.

- [oldSourceSchema <old source>]
  - Specifies the table owner in the original subscription on the old source datastore. Use the format database.schema when required by the datastore. Only schemas that are changing between the original and promoted subscription need to be specified. For each old source schema, you must provide the new source schema.This parameter can be specified more than once.

- [newSourceSchema <new source>]
  - Specifies the table owner in the promotion subscription on the new source datastore. Use the format database.schema when required by the datastore. Only schemas that are changing between the original and promoted subscription need to be specified. For each old source schema, you must provide the new source schema.This parameter can be specified more than once.

- [oldTargetSchema <old target>]
  - Specifies the table owner in the original subscription on the old target datastore. Use the format database.schema when required by the datastore. Only schemas that are changing between the original and promotion subscription need to be specified. For each old target schema, you must provide the new target schema.This parameter can be specified more than once.

- [newTargetSchema <new target>]
  - Specifies the table owner in the promotion subscription on the new target datastore. Use the format database.schema when required by the datastore. Only schemas that are changing between the original and promotion subscription need to be specified. For each old target schema, you must provide the new target schema.This parameter can be specified more than once.

Required role

- Administrator
- System administrator

Result

This command returns a value of 0 if the command was successful and a negative value if the command fails. When verbose mode is enabled, the command prints success message CDC1329.

Sample

Selects a table mapping and promotes it to a new subscription.

```
connect server username user1 password password1;
connect datastore name DS1 context source;
connect datastore name DS2 context target;
select subscription name SUB1;
select table mapping sourceTable TABLE_1;
promote table mapping type new promotionname SUB2;
```

```
disconnect server;

exit;
```

Parent topic: Table mapping commands

reassign table mapping

Use this command to reassign the selected table. The table mapping must be first selected as the current context. Reassign updates the target datastore’s metadata for the table mapping.

Syntax diagram

.arrow, .syntaxarrow { fill: none; stroke: black; } .arrowheadStartEnd, .arrowheadRepSep, .arrowheadRepSepReturn { stroke: black; fill: black; } .arrowheadSeq, .arrowheadStartChoice, .arrowheadAfterChoice, .arrowheadStartRepGroup, .arrowheadEndRepGroup, .arrowheadRev { stroke: none; fill: none; } rect { fill: none; stroke: none; } rect.fragref,rect.syntaxfragref { fill: none; stroke: black; } text { fill: #000000; fill-opacity: 1; font-family: IBM Plex Sans,Arial Unicode MS,Arial,Helvetica; font-style: normal; font-weight: normal; font-size: 8pt; stroke: #000000; stroke-width: 0.1; } text.var, text.syntaxvar {font-style:italic;} reassign table mapping

Syntax

```
reassign table mapping;
```

Parameters

None.

Required role

- Administrator
- System administrator

Result

This command returns a value of 0 if the command was successful and a negative value if the command fails. When verbose mode is enabled, the command prints success message CDC1407.

Sample

Reassigns the table mapping.

```
connect server username user1 password password1;

connect datastore name DS1 context source;

connect datastore name DS2 context target;

select subscription name SUB1;

select table mapping

sourceSchema USER1 sourceTable TABLE_1

targetSchema USER1 targetTable TABLE_1;

reassign table mapping;

disconnect server;

exit;
```

Parent topic: Table mapping commands

select rule set

Use this command to set the rule set as active for subsequent commands. Rule sets are identified by the name on the command. Use the show context command to view the current rule set context.

Syntax diagram

.arrow, .syntaxarrow { fill: none; stroke: black; } .arrowheadStartEnd, .arrowheadRepSep, .arrowheadRepSepReturn { stroke: black; fill: black; } .arrowheadSeq, .arrowheadStartChoice, .arrowheadAfterChoice, .arrowheadStartRepGroup, .arrowheadEndRepGroup, .arrowheadRev { stroke: none; fill: none; } rect { fill: none; stroke: none; } rect.fragref,rect.syntaxfragref { fill: none; stroke: black; } text { fill: #000000; fill-opacity: 1; font-family: IBM Plex Sans,Arial Unicode MS,Arial,Helvetica; font-style: normal; font-weight: normal; font-size: 8pt; stroke: #000000; stroke-width: 0.1; } text.var, text.syntaxvar {font-style:italic;} select rule setnamesubscriptionruleNameset

Syntax

```
select rule set [name <subscription>] ruleName <set>;
```

Parameters

- [name <subscription>]

- Specifies the name of the subscription.If a name is not provided, the subscription that is currently identified as the context will be used. To view the current context, use the show context command.

- ruleName <set>

- Specifies the name of the rule set.Rule set names must be unique within a subscription.

Required role

- Operator
- Administrator
- System administrator

Result

This command returns a value of 0 if the command was successful and a negative value if the command fails. When verbose mode is enabled, the command prints success message CDC1423. **Sample**

Selects a rule set and shows that it is the current context.

```
connect server username user1
password password1;
connect datastore name DS1 context source;
connect datastore name DS2 context target;
select subscription name SUB1;
select rule set ruleName rule1;
show context;
disconnect server;
exit;
```

Parent topic: Table mapping commands

select table mapping

Use this command to select a table mapping to context. The table mapping will be used for subsequent commands. Table mappings are identified using the source and target parameters provided with the command. An error will be reported if there is not a unique match.
Use the show context command to view the current table mapping context.

Syntax diagram

.arrow, .syntaxarrow { fill: none; stroke: black; } .arrowheadStartEnd, .arrowheadRepSep, .arrowheadRepSepReturn { stroke: black; fill: black; } .arrowheadSeq, .arrowheadStartChoice, .arrowheadAfterChoice, .arrowheadStartRepGroup, .arrowheadEndRepGroup, .arrowheadRev { stroke: none; fill: none; } rect { fill: none; stroke: none; } rect.fragref,rect.syntaxfragref { fill: none; stroke: black; } text { fill: #000000; fill-opacity: 1; font-family: IBM Plex Sans,Arial Unicode MS,Arial,Helvetica; font-style: normal; font-weight: normal; font-size: 8pt; stroke: #000000; stroke-width: 0.1; } text.var, text.syntaxvar {font-style:italic;} select table mappingnamesubscriptionsourceDatabasedatabase sourceSchemaschemasourceTabletabletargetDatabasedatabase2targetSchema schema2targetTabletable2

Syntax

```
select table mapping [name <subscription>] [sourceDatabase <database>]

[sourceSchema <schema>] [sourceTable <table>] [targetDatabase <database2>]

[targetSchema <schema2>] [targetTable <table2>];
```

Parameters

- [name <subscription>]
- Specifies the name of the subscription.If a name is not provided, the subscription that is currently identified as the context will be used. To view the current context, use the show context command.

- [sourceDatabase <database>]
- Specifies the database for the source table.

- [sourceSchema <schema>]
- Specifies the schema for the source table.

- [sourceTable <table>]
- Specifies the name of the source table.

- [targetDatabase <database2>]
- Specifies the database for the source table.

- [targetSchema <schema2>]
- Specifies the schema for the target table.

- [targetTable <table2>]
- Specifies the name of the target table.

Required role

- Monitor

Result

This command returns a value of 0 if the command was successful and a negative value if the command fails. When verbose mode is enabled, the command prints success message CDC1401. **Sample**

Selects a table mapping by fully specifying the source and target tables.

```
connect server username user1 password password1;

connect datastore name DS1 context source;

connect datastore name DS2 context target;

select subscription name SUB1;

select table mapping

sourceSchema USER1 sourceTable TABLE_1

targetSchema USER1 targetTable TABLE_1;

show context;

disconnect server;

exit;
```

Selects a table mapping by the source table name. If more than one table matches, an error will be reported.

```
connect server username user1 password password1;

connect datastore name DS1 context source;

connect datastore name DS2 context target;

select subscription name SUB1;

select table mapping sourceTable TABLE_1;

show context;

disconnect server;

exit;
```

Parent topic: Table mapping commands

show rule set

Use this command to show the properties of a rule set. The rule set must be first selected as the current context.

Syntax diagram

.arrow, .syntaxarrow { fill: none; stroke: black; } .arrowheadStartEnd, .arrowheadRepSep, .arrowheadRepSepReturn { stroke: black; fill: black; } .arrowheadSeq, .arrowheadStartChoice, .arrowheadAfterChoice, .arrowheadStartRepGroup, .arrowheadEndRepGroup, .arrowheadRev { stroke: none; fill: none; } rect { fill: none; stroke: none; } rect.fragref,rect.syntaxfragref { fill: none; stroke: black; } text { fill: #000000; fill-opacity: 1; font-family: IBM Plex Sans,Arial Unicode MS,Arial,Helvetica; font-style: normal; font-weight: normal; font-size: 8pt; stroke: #000000; stroke-width: 0.1; } text.var, text.syntaxvar {font-style:italic;} show rule set ruleNameset

Syntax

```
show rule set [ruleName <set>];
```

Parameters

- [ruleName <set>]

- Specifies the name of the rule set.If a name is not provided, the rule set that is currently identified as the context will be used. To view the current context, use the show context command.

Required role

- Operator
- Administrator
- System administrator

Result

A ResultStringKeyValues object with the attributes of the rule set.

```
PROPERTY
VALUE
Rule Set Name              RULE1
Schema                     USER1
Structural Changes Only    Yes


Include Tables
 B                        Exact Match


Exclude Tables
 C                        Ends With
```

Sample

Shows the properties of a rule set.

```
connect server username user1 password password1;
connect datastore name DS1 context source;
connect datastore name DS2 context target;
select subscription name SUB1;
select rule set ruleName rule1;
show rule set;
disconnect server;
exit;
```

Parent topic: Table mapping commands

show table mapping

Use this command to show the properties of a table mapping. The table mapping must be first selected as the current context.

Syntax diagram

.arrow, .syntaxarrow { fill: none; stroke: black; } .arrowheadStartEnd, .arrowheadRepSep, .arrowheadRepSepReturn { stroke: black; fill: black; } .arrowheadSeq, .arrowheadStartChoice, .arrowheadAfterChoice, .arrowheadStartRepGroup, .arrowheadEndRepGroup, .arrowheadRev { stroke: none; fill: none; } rect { fill: none; stroke: none; } rect.fragref,rect.syntaxfragref { fill: none; stroke: black; } text { fill: #000000; fill-opacity: 1; font-family: IBM Plex Sans,Arial Unicode MS,Arial,Helvetica; font-style: normal; font-weight: normal; font-size: 8pt; stroke: #000000; stroke-width: 0.1; } text.var, text.syntaxvar {font-style:italic;} show table mapping

Syntax

```
show table mapping;
```

Parameters

None.

Required role

- Operator
- Administrator
- System administrator

Result

A ResultStringKeyValues object with the attributes of the table mapping.

```
PROPERTY                     VALUE
---------------------------- ----------------------------
Source Table:                CDC.TABLE_1
Target Table:                CDC.TABLE_1
Mapping Type:                Standard
Method:                      Mirror
Status:                      Refresh

TARGET KEY
 Index Mode:                 Auto Detect

REFRESH
 Row Subset Refresh:
 Source WHERE clause:
 Target WHERE clause:

ROW-FILTERING
 Row-filtering Expression:
 Select/Omit Rows:

CONFLICTS
 Conflict Detection Columns:
 Conflict Resolution Method: None
 Value Comparison Column:
 User Exit (with Path):
```

Sample

Shows the properties of a table mapping.

```
connect server username user1 password password1;
connect datastore name DS1 context source;
connect datastore name DS2 context target;
select subscription name SUB1;
select table mapping
sourceSchema USER1 sourceTable TABLE_1
targetSchema USER1 targetTable TABLE_1;
show table mapping;
disconnect server;
exit;
```

Parent topic: Table mapping commands

Mapping details commands

- **add data translation**
- **add derived column**
- **clear user exit**
- **delete data translation**
- **delete derived column**
- **filter source column**
- **list column encodings**
- **list column mappings**
- **list source columns**
- **map column**
- **modify column encoding**
- **modify data translation**
- **modify derived column**
- **modify operations**
- **modify user exit cdll**
- **modify user exit function**
- **modify user exit javaclass**
- **modify user exit storedproc**
- **show data translation**
- **show derived column**
- **show operations**
- **show user exit**
- **unmap column**

Parent topic:CHCCLP Commands

add data translation

Use this command to add a data translation for the target column in the table mapping that is the current context.

Syntax diagram

.arrow, .syntaxarrow { fill: none; stroke: black; } .arrowheadStartEnd, .arrowheadRepSep, .arrowheadRepSepReturn { stroke: black; fill: black; } .arrowheadSeq, .arrowheadStartChoice, .arrowheadAfterChoice, .arrowheadStartRepGroup, .arrowheadEndRepGroup, .arrowheadRev { stroke: none; fill: none; } rect { fill: none; stroke: none; } rect.fragref,rect.syntaxfragref { fill: none; stroke: black; } text { fill: #000000; fill-opacity: 1; font-family: IBM Plex Sans,Arial Unicode MS,Arial,Helvetica; font-style: normal; font-weight: normal; font-size: 8pt; stroke: #000000; stroke-width: 0.1; } text.var, text.syntaxvar {font-style:italic;} add data translationtargetColumnnamebeforevalueaftertranslated

Syntax

```
add data translation targetColumn <name> before <value> after <translated>;
```

Parameters

- targetColumn <name>

- Specifies the name of the target column.

- before <value>

- Specifies the before value. Set this parameter to the string <NULL> to indicate that you want to translate null to a specific value.

- after <translated>

- Specifies the after (translated) value. Set this parameter to the string <NULL> to indicate that you want to translate a specific value to null.

Required role

- Administrator
- System administrator

Result

This command returns a value of 0 if the command was successful and a negative value if the command fails. When verbose mode is enabled, the command prints success message CDC1416.

Sample

Translates state codes to full names.

```
connect server username user1 password password1;

connect datastore name DS1 context source;

connect datastore name DS2 context target;

select subscription name SUB1;

select table mapping sourceSchema USER1 sourceTable LOCATION;

add data translation targetColumn STATE before CA after California;

add data translation targetColumn STATE before NY after New York;

show data translation targetColumn STATE;

disconnect server;

exit;
```

Parent topic: Mapping details commands

add derived column

Use this command to add a derived column in the table mapping that is the current context.

Syntax diagram

.arrow, .syntaxarrow { fill: none; stroke: black; } .arrowheadStartEnd, .arrowheadRepSep, .arrowheadRepSepReturn { stroke: black; fill: black; } .arrowheadSeq, .arrowheadStartChoice, .arrowheadAfterChoice, .arrowheadStartRepGroup, .arrowheadEndRepGroup, .arrowheadRev { stroke: none; fill: none; } rect { fill: none; stroke: none; } rect.fragref,rect.syntaxfragref { fill: none; stroke: black; } text { fill: #000000; fill-opacity: 1; font-family: IBM Plex Sans,Arial Unicode MS,Arial,Helvetica; font-style: normal; font-weight: normal; font-size: 8pt; stroke: #000000; stroke-width: 0.1; } text.var, text.syntaxvar {font-style:italic;} add derived columnnamecolumndescription""descriptiondatatypetype length0bytesscale0scalenullablefalse/notrue/yesdateTimeFormat NONEMDYDMY YMDJULISOUSAEURJISHMSdateTimeSeparator spaceforwardSlashhyphenperiod commacolonevaluateafterbothexpressionname

Syntax

```
add derived column name <column> [description <description>] datatype <type>

[length <bytes>] [scale <scale>] [nullable <value>] [dateTimeFormat <format>]

[dateTimeSeparator  <separator>][evaluate <frequency>] expression <name>;
```

Parameters

- name <column>

- Specifies the name of the derived column.

- [description <description>]

- Specifies the description about the purpose of the derived column.Default value: empty string ().

- datatype <type>

- Specifies the data type for the derived column.

- [length <bytes>]

- Specifies the length of the column in bytes.Default value: 0

- [scale <scale>]

- Specifies the scale of the numerical data type.Default value: 0

- [nullable <value>]

- Specifies if the column is nullable.Valid values:
  - **true/yes**—Enables the column as nullable. Either yes or true are valid values.
  - **false/no**—Does not enable the column as nullable. Either no or false are valid values.

  Default value: false/no

- [dateTimeFormat <frequency>]

- Specifies the format of the date/time data type.Valid values:
  - **NONE**—
  - **MDY**—Specifies the input format is mmddyy
  - **DMY**—Specifies the input format is ddmmyy
  - **YMD**—Specifies the input format is yymmdd
  - **JUL**—Specifies the input format is yyjjj, where jjj represent the sequence number of a day in the calendar year. jjj must be between 1, which represents

January 1st, and 366, which represents December 31st in a leap year. For jjj values less than 100, you must specify the leading zero or zeros. For example, the Julian date for February 4th is 035, which represents the 35th day of the year
    - **ISO**—Specifies the input format is the International Organization for Standardization date format (CCYY/MM/DD HH.MM.SS).
    - **USA**—Specifies the input format is the United States date format (MM/DD/CCYY HH:MM AM/PM).
    - **EUR**—Specifies the input format is the European date format (DD/MM/CCYY HH.MM.SS).
    - **JIS**—Specifies the input format is Japanese Industrial Standard (CCYY/MM/DD HH:MM:SS).
    - **HMS**—Specifies the input format is HH MM SS.

- **[dateTimeSeparator <separator>]**
  - Specifies the character that will be used as a separator when formatting date/time data types.Valid values:
    - **space**—Specifies that a space will be used as a separator.
    - **forwardSlash**—Specifies that a forward slash will be used as a separator.
    - **hyphen**—Specifies that a hyphen will be used as a separator.
    - **period**—Specifies that a period will be used as a separator.
    - **comma**—Specifies that a comma will be used as a separator.
    - **colon**—Specifies that a colon will be used as a separator.
- **[evaluate <frequency>]**
  - Specifies the evaluation frequency you want for this derived column.Valid values:
    - **after**—Select this option when you want InfoSphere® CDC to evaluate the expression in the derived column for the after image of the source table.
    - **both**—Select this option when you want InfoSphere CDC to evaluate the expression in the derived column for both the before images and after images of the source table.

    Default value: after
- **expression <name>**
  - Specifies the expression for the derived column.

Required role

- Administrator
- System administrator

Result

This command returns a value of 0 if the command was successful and a negative value if the command fails. When verbose mode is enabled, the command prints success message CDC1410.

Sample

Adds a derived column that changes the PARTNAME value to upper case.

```
connect server username user1 password password1;

connect datastore name DS1 context source;

connect datastore name DS2 context target;

select subscription name SUB1;

select table mapping sourceSchema USER1 sourceTable TABLE_1;

add derived column name PART_NAME

 datatype VARCHAR length 100
 expression %UPPER(PARTNAME);
list source columns;
disconnect server;
exit;
```

Parent topic: Mapping details commands

clear user exit

Use this command to clear user exits for the table mapping.

Syntax diagram

.arrow, .syntaxarrow { fill: none; stroke: black; } .arrowheadStartEnd, .arrowheadRepSep, .arrowheadRepSepReturn { stroke: black; fill: black; } .arrowheadSeq, .arrowheadStartChoice, .arrowheadAfterChoice, .arrowheadStartRepGroup, .arrowheadEndRepGroup, .arrowheadRev { stroke: none; fill: none; } rect { fill: none; stroke: none; } rect.fragref,rect.syntaxfragref { fill: none; stroke: black; } text { fill: #000000; fill-opacity: 1; font-family: IBM Plex Sans,Arial Unicode MS,Arial,Helvetica; font-style: normal; font-weight: normal; font-size: 8pt; stroke: #000000; stroke-width: 0.1; } text.var, text.syntaxvar {font-style:italic;} clear user exit

Syntax

```
clear user exit;
```

Parameters

None. **Required role**

- Administrator
- System administrator

Result

Sample

Selects table mapping and clears user exits.

```
connect server username user1 password password1;
connect datastore name DS1 context source;
connect datastore name DS2 context target;
select subscription name SUB1;
select table mapping sourceSchema USER1 sourceTable TABLE_1 targetSchema USER1 targetTable TABLE_1;
clear user exit;
disconnect server;
exit;
```

Parent topic: Mapping details commands

delete data translation

Use this command to delete one or all data translation definition for the target column in the table mapping that is the current context.

Syntax diagram

.arrow, .syntaxarrow { fill: none; stroke: black; } .arrowheadStartEnd, .arrowheadRepSep, .arrowheadRepSepReturn { stroke: black; fill: black; } .arrowheadSeq, .arrowheadStartChoice, .arrowheadAfterChoice, .arrowheadStartRepGroup, .arrowheadEndRepGroup, .arrowheadRev { stroke: none; fill: none; } rect { fill: none; stroke: none; } rect.fragref,rect.syntaxfragref { fill: none; stroke: black; } text { fill: #000000; fill-opacity: 1; font-family: IBM Plex Sans,Arial Unicode MS,Arial,Helvetica; font-style: normal; font-weight: normal; font-size: 8pt; stroke: #000000; stroke-width: 0.1; } text.var, text.syntaxvar {font-style:italic;} delete data translationtargetColumnnamebeforevalue

Syntax

```
delete data translation targetColumn <name> [before <value>];
```

Parameters

- targetColumn <name>

- Specifies the name of the target column.

- [before <value>]

- Specifies the before value for the data translation to be deleted. If this value is not provided, all data translations for the column will be deleted.

Required role

- Administrator
- System administrator

Result

This command returns a value of 0 if the command was successful and a negative value if the command fails. When verbose mode is enabled, the command prints success message CDC1417.

Sample

Deletes a data translation for a column.

```
connect server username user1 password password1;

connect datastore name DS1 context source;

connect datastore name DS2 context target;

select subscription name SUB1;

select table mapping sourceSchema USER1 sourceTable PART_NUMBERS;

delete data translation targetColumn PART_CODE before A1;

show data translation targetColumn PART_CODE;

disconnect server;

exit;
```

Parent topic: Mapping details commands

delete derived column

Use this command to delete a derived column in the table mapping that is the current context.

Syntax diagram

.arrow, .syntaxarrow { fill: none; stroke: black; } .arrowheadStartEnd, .arrowheadRepSep, .arrowheadRepSepReturn { stroke: black; fill: black; } .arrowheadSeq, .arrowheadStartChoice, .arrowheadAfterChoice, .arrowheadStartRepGroup, .arrowheadEndRepGroup, .arrowheadRev { stroke: none; fill: none; } rect { fill: none; stroke: none; } rect.fragref,rect.syntaxfragref { fill: none; stroke: black; } text { fill: #000000; fill-opacity: 1; font-family: IBM Plex Sans,Arial Unicode MS,Arial,Helvetica; font-style: normal; font-weight: normal; font-size: 8pt; stroke: #000000; stroke-width: 0.1; } text.var, text.syntaxvar {font-style:italic;} delete derived columnname column

Syntax

```
delete derived column name <column>;
```

Parameters

- name <column>

- Specifies the name of the derived column.

Required role

- Administrator
- System administrator

Result

This command returns a value of 0 if the command was successful and a negative value if the command fails. When verbose mode is enabled, the command prints success message CDC1411. **Sample**

Deletes a derived column. The column must be unmapped before it can be deleted.

```
connect server username user1 password password1;

connect datastore name DS1 context source;

connect datastore name DS2 context target;

select subscription name SUB1;

select table mapping sourceSchema USER1 sourceTable TABLE_1;

unmap column targetColumn PART_NAME;

delete derived column PART_NAME;

list source columns;

disconnect server;

exit;
```

Parent topic: Mapping details commands

filter source column

Use this command to filter a source column for replication in the table mapping that is the current context. This command can also be used to mark a source column as a critical column.

Syntax diagram

.arrow, .syntaxarrow { fill: none; stroke: black; } .arrowheadStartEnd, .arrowheadRepSep, .arrowheadRepSepReturn { stroke: black; fill: black; } .arrowheadSeq, .arrowheadStartChoice, .arrowheadAfterChoice, .arrowheadStartRepGroup, .arrowheadEndRepGroup, .arrowheadRev { stroke: none; fill: none; } rect { fill: none; stroke: none; } rect.fragref,rect.syntaxfragref { fill: none; stroke: black; } text { fill: #000000; fill-opacity: 1; font-family: IBM Plex Sans,Arial Unicode MS,Arial,Helvetica; font-style: normal; font-weight: normal; font-size: 8pt; stroke: #000000; stroke-width: 0.1; } text.var, text.syntaxvar {font-style:italic;} filter source column namecolumnreplicatetrue/yesfalse/nocriticaltrue/yes false/no

Syntax

```
filter source column name <column> replicate <filter> [critical <value>];
```

Parameters

- name <column>

- Specifies the name of the source column.

- replicate <filter>

- Filters the column for replication.Valid values:
  - **true/yes**—Specifies that the column will be filtered. Either yes or true are valid values.
  - **false/no**—Specifies that the column will not be filtered. Either no or false are valid values.

- [critical <value>]

- Marks the column as a critical column.Valid values:
  - **true/yes**—Specifies that the column will be marked as a critical column. Either yes or true are valid values.
  - **false/no**—Specifies that the column will not be marked as a critical column. Either no or false are valid values.

Required role

- Administrator
- System administrator

Result

This command returns a value of 0 if the command was successful and a negative value if the command fails. When verbose mode is enabled, the command prints success message CDC1413. **Sample**

Deselects a column from replication.

```
connect server username user1 password password1;

connect datastore name DS1 context source;

connect datastore name DS2 context target;

select subscription name SUB1;

select table mapping sourceSchema USER1 sourceTable EMPLOYEE;

list source columns;
```

```
filter source column name SALARY replicate no critical no;
list source columns;
disconnect server;
exit;
```

Parent topic: Mapping details commands

list column encodings

Use this command to list column MBCS encodings for a source or target table in the table mapping that is the current context.

Syntax diagram

.arrow, .syntaxarrow { fill: none; stroke: black; } .arrowheadStartEnd, .arrowheadRepSep, .arrowheadRepSepReturn { stroke: black; fill: black; } .arrowheadSeq, .arrowheadStartChoice, .arrowheadAfterChoice, .arrowheadStartRepGroup, .arrowheadEndRepGroup, .arrowheadRev { stroke: none; fill: none; } rect { fill: none; stroke: none; } rect.fragref,rect.syntaxfragref { fill: none; stroke: black; } text { fill: #000000; fill-opacity: 1; font-family: IBM Plex Sans,Arial Unicode MS,Arial,Helvetica; font-style: normal; font-weight: normal; font-size: 8pt; stroke: #000000; stroke-width: 0.1; } text.var, text.syntaxvar {font-style:italic;} list column encodingstypesourcetarget

Syntax

```
list column encodings [type <value>];
```

Parameters

- [type <value>]

- Specifies whether to list column encodings for the source or target tableValid values:
  - **source**—Shows the column encodings for the source table.
  - **target**—Shows the column encodings for the target table.

  Default: source

Required role

- Operator
- Administrator
- System administrator

Result

A ResultStringTable of the database and source encoding for the columns for a source table in the table mapping.

```
Column MBCS encoding conversion for source table CUSTOMER.

SOURCE COLUMN    DATABASE ENCODING   SOURCE ENCODING
---------------- ------------------- -----------------
CUSTOMER_NUM
ZIP              windows-1252        windows-1252
NAME             windows-1252        windows-1252
CREDIT_LIMIT
ADDR_LN1         windows-1252        windows-1252
ADDR_LN2         windows-1252        windows-1252
CITY             windows-1252        windows-1252
STATE            windows-1252        windows-1252
PHONE            windows-1252        windows-1252
FAX              windows-1252        windows-1252
EMAIL            windows-1252        windows-1252
```

Sample

Lists the database and actual encodings for the columns in the source and target tables in a table mapping.

```
connect server username user1 password password1;
connect datastore name DS1 context source;
connect datastore name DS2 context target;
select subscription name SUB1;
select table mapping sourceSchema USER1 sourceTable CUSTOMER;
list column encodings type source;
list column encodings type target;
disconnect server;
exit;
```

Parent topic:Mapping details commands

list column mappings

Use this command to list the column mappings for the table mapping in the table mapping that is the current context.

Syntax diagram

.arrow, .syntaxarrow { fill: none; stroke: black; } .arrowheadStartEnd, .arrowheadRepSep, .arrowheadRepSepReturn { stroke: black; fill: black; } .arrowheadSeq, .arrowheadStartChoice, .arrowheadAfterChoice, .arrowheadStartRepGroup, .arrowheadEndRepGroup, .arrowheadRev { stroke: none; fill: none; } rect { fill: none; stroke: none; } rect.fragref,rect.syntaxfragref { fill: none; stroke: black; } text { fill: #000000; fill-opacity: 1; font-family: IBM Plex Sans,Arial Unicode MS,Arial,Helvetica; font-style: normal; font-weight: normal; font-size: 8pt; stroke: #000000; stroke-width: 0.1; } text.var, text.syntaxvar {font-style:italic;} list column mappings showDataTypefalse/notrue/yes

Syntax

```
list column mappings [showDataType <value>];
```

Parameters

- [showDataType <value>]

- Enables the output to include column data types.Valid values:
  - **true/yes**—Includes column data type information in the output. Either yes or true are valid values.
  - **false/no**—Does not include column data type information in the output. Either no or false are valid values.

  Default value: false/no

Required role

- Operator
- Administrator
- System administrator

Result

A ResultStringTable containing the column mappings for a table mapping.`Column mappings for table mapping USER1.CUSTOMER – USER1.CUSTOMER.`

```
SOURCE COLUMN    TARGET COLUMN    INITIAL VALUE
---------------- ---------------- ----------------
CUSTOMER_NUM     CUSTOMER_NUM
ZIP              ZIP
NAME             NAME
CREDIT_LIMIT     CREDIT_LIMIT
ADDR_LN1         ADDR_LN1
ADDR_LN2         ADDR_LN2
CITY             CITY
STATE            STATE
PHONE            PHONE
FAX              FAX
EMAIL            EMAIL
```

Sample

Lists the column mappings for the selected table mapping.

```
connect server username user1 password password1;

connect datastore name DS1 context source;

connect datastore name DS2 context target;

select subscription name SUB1;

select table mapping sourceSchema USER1 sourceTable CUSTOMER;

list column mappings;

disconnect server;

exit;
```

Parent topic: Mapping details commands

list source columns

Use this command to list column structures and column selections for the source table of the table mapping that is the current context.

Syntax diagram

.arrow, .syntaxarrow { fill: none; stroke: black; } .arrowheadStartEnd, .arrowheadRepSep, .arrowheadRepSepReturn { stroke: black; fill: black; } .arrowheadSeq, .arrowheadStartChoice, .arrowheadAfterChoice, .arrowheadStartRepGroup, .arrowheadEndRepGroup, .arrowheadRev { stroke: none; fill: none; } rect { fill: none; stroke: none; } rect.fragref,rect.syntaxfragref { fill: none; stroke: black; } text { fill: #000000; fill-opacity: 1; font-family: IBM Plex Sans,Arial Unicode MS,Arial,Helvetica; font-style: normal; font-weight: normal; font-size: 8pt; stroke: #000000; stroke-width: 0.1; } text.var, text.syntaxvar {font-style:italic;} list source columns

Syntax

```
list source columns;
```

Parameters

None. **Required role**

- Operator
- Administrator
- System administrator

Result

A ResultStringTable containing the columns for a source table in a table mapping.

```
Source columns for table USER1.CUSTOMER.


SOURCE COLUMN    DATA TYPE        DERIVED        REPLICATE      CRITICAL

---------------- ---------------- -------------- -------------- --------------

CUSTOMER_NUM     NUMBER           No             Yes            No

ZIP              VARCHAR2         No             Yes            No

NAME             VARCHAR2         No             Yes            No

CREDIT_LIMIT     NUMBER           No             Yes            No

ADDR_LN1         VARCHAR2         No             Yes            No

ADDR_LN2         VARCHAR2         No             Yes            No

CITY             VARCHAR2         No             Yes            No

STATE            CHAR             No             Yes            No

PHONE            CHAR             No             Yes            No

FAX              CHAR             No             Yes            No

EMAIL            VARCHAR2         No             Yes            No
```

Sample

Lists the source columns in the table and any derived columns that have been added to it.

```
connect server username user1 password password1;

connect datastore name DS1 context source;

connect datastore name DS2 context target;

select subscription name SUB1;

select table mapping sourceSchema USER1 sourceTable CUSTOMER;

list source columns;
```

```
disconnect server;

exit;
```

Parent topic: Mapping details commands

map column

Use this command to map a source column, journal control field, or expression to a target column in the table mapping that is the current context. For unmapped target columns, the initial value can be set to a constant, blank, null, zero, database default, or current date, depending on the column definition.

Syntax diagram

.arrow, .syntaxarrow { fill: none; stroke: black; } .arrowheadStartEnd, .arrowheadRepSep, .arrowheadRepSepReturn { stroke: black; fill: black; } .arrowheadSeq, .arrowheadStartChoice, .arrowheadAfterChoice, .arrowheadStartRepGroup, .arrowheadEndRepGroup, .arrowheadRev { stroke: none; fill: none; } rect { fill: none; stroke: none; } rect.fragref,rect.syntaxfragref { fill: none; stroke: black; } text { fill: #000000; fill-opacity: 1; font-family: IBM Plex Sans,Arial Unicode MS,Arial,Helvetica; font-style: normal; font-weight: normal; font-size: 8pt; stroke: #000000; stroke-width: 0.1; } text.var, text.syntaxvar {font-style:italic;} map columntargetColumnnametypesourceColumnconstantexpression journalinitialValueaccumulationdeductionvaluevalue

Syntax

```
map column targetColumn <name> [type <mapping>] value <value> [summarizationType

<type>];
```

Parameters

- targetColumn <name>
- Specifies the target column to map.

- [type <mapping>]
- Specifies the type of the column mapping.Valid values:
  - **sourceColumn**—Maps a source column to a target column.
  - **constant**—Populates the target column with a constant value.
  - **expression**—Maps an expression to a target column.
  - **journal**—Maps a journal control field to a target column.
  - **initialValue**—Populates the target column with an initial value.
  - **accumulation**—Ensures that numeric changes applied to the target column are directly proportional to changes applied to the corresponding source columns
  - **deduction**—Ensures that numeric changes applied to the target columns are inversely proportional to changes applied to mapped source columns.

  Default value: sourceColumn

- value <value>
- Specifies the following information based on the value assigned in the type parameter.
  - If sourceColumn was assigned, <value> must be the source column name.
  - If constant was assigned, <value> must be the constant value.
  - If journal was assigned, <value> must be the journal control field name.
  - If expression was assigned, <value> must be the derived expression.
  - If initialValue was assigned, <value> must one of the following expected values: blank, zero, null, databaseDefault, currentDate

  - If accumulation was assigned, <value> must be the accumulation source column name or expression for a summarization mapping.
  - If deduction was assigned, <value> must be the deduction source column name or expression for a summarization mapping.

- [summarizationType <type>]

- Specifies the type of the value parameter. This parameter is only applicable when the type parameter has been set to accumulation or deduction.
- Valid values:
  - **sourceColumn**
  - **expression**

Required role

- Administrator
- System administrator

Result

This command returns a value of 0 if the command was successful and a negative value if the command fails. When verbose mode is enabled, the command prints success message CDC1409.

Sample

Maps column names between source and target tables containing slightly different column names. When tables are mapped, the datastore will automatically map columns with identical names and compatible data types.

```
connect server username user1 password password1;
connect datastore name DS1 context source;
connect datastore name DS2 context target;
select subscription name SUB1;
select table mapping sourceSchema USER1 sourceTable CUSTOMER;
map column targetColumn CUSTOMER_NO value CUSTNO;
map column targetColumn CUSTOMER_NAME value NAME;
map column targetColumn PHONE type expression
 value '%CONCAT(AREA_CODE, -, PHONE_NUMBER)';
list column mappings;
disconnect server;
exit;
```

Maps a target column to a journal control field.

```
connect server username user1 password password1;
connect datastore name DS1 context source;
connect datastore name DS2 context target;
select subscription name SUB1;
select table mapping sourceSchema USER1 sourceTable CUSTOMER;
map column targetColumn MODUSER type journal value &USER;
list column mappings; disconnect server;
exit;
```

Sets an initial value for a target column.

```
connect server username user1 password password1;
connect datastore name DS1 context source;
connect datastore name DS2 context target;
select subscription name SUB1;
select table mapping sourceSchema USER1 sourceTable PRODUCTS;
map column targetColumn PRCODE type initialValue value zero;
```

```
list column mappings;

disconnect server;

exit;
```

Parent topic: Mapping details commands

modify column encoding

Use this command to modify column encodings for a source or target column in the table mapping that is the current context.

Syntax diagrams

Syntax 1: .arrow, .syntaxarrow { fill: none; stroke: black; } .arrowheadStartEnd, .arrowheadRepSep, .arrowheadRepSepReturn { stroke: black; fill: black; } .arrowheadSeq, .arrowheadStartChoice, .arrowheadAfterChoice, .arrowheadStartRepGroup, .arrowheadEndRepGroup, .arrowheadRev { stroke: none; fill: none; } rect { fill: none; stroke: none; } rect.fragref,rect.syntaxfragref { fill: none; stroke: black; } text { fill: #000000; fill-opacity: 1; font-family: IBM Plex Sans,Arial Unicode MS,Arial,Helvetica; font-style: normal; font-weight: normal; font-size: 8pt; stroke: #000000; stroke-width: 0.1; } text.var, text.syntaxvar {font-style:italic;} modify column encodingsourceColumnnametypedefaultbinaryspecify valuevalue

Syntax 2:
.arrow, .syntaxarrow { fill: none; stroke: black; } .arrowheadStartEnd, .arrowheadRepSep, .arrowheadRepSepReturn { stroke: black; fill: black; } .arrowheadSeq, .arrowheadStartChoice, .arrowheadAfterChoice, .arrowheadStartRepGroup, .arrowheadEndRepGroup, .arrowheadRev { stroke: none; fill: none; } rect { fill: none; stroke: none; } rect.fragref,rect.syntaxfragref { fill: none; stroke: black; } text { fill: #000000; fill-opacity: 1; font-family: IBM Plex Sans,Arial Unicode MS,Arial,Helvetica; font-style: normal; font-weight: normal; font-size: 8pt; stroke: #000000; stroke-width: 0.1; } text.var, text.syntaxvar {font-style:italic;} modify column encodingtargetColumnnametypedefaultbinaryspecify valuevalue

Syntax

Syntax 1:`modify column encoding [sourceColumn <name>] type <override> [value <value>];`

Syntax 2:

```
modify column encoding [targetColumn <name>] type <override> [value <value>];
```

Parameters

Syntax 1:

- [sourceColumn <name>]

- Specifies the name of the source column.

- type <override>

- Specifies the MBCS encoding conversion action.Valid values:
  - **default**—Uses the database default encoding.
  - **binary**—Replicates data with no changes to the encoding.
  - **specify**—Allows you to specify an IANA encoding.

- [value <value>]

- Specifies the IANA name of the character encoding that is to be used when the specify value is assigned in the type parameter.

Syntax 2:

- [targetColumn <name>]

- Specifies the name of the target column.

- type <override>

- Specifies the MBCS encoding conversion action.Valid values:
  - **default**—Uses the database default encoding.
  - **binary**—Replicates data with no changes to the encoding.
  - **specify**—Allows you to specify an IANA encoding.

- [value <value>]

- Specifies the IANA name of the character encoding that is to be used when the specify value is assigned in the type parameter.

Required role

- Administrator
- System administrator

Result

This command returns a value of 0 if the command was successful and a negative value if the command fails. When verbose mode is enabled, the command prints success message CDC1414.

Sample

Overrides the database encoding for a column and identifies that the data is binary.

```
connect server username user1 password password1;

connect datastore name DS1 context source;

connect datastore name DS2 context target;

select subscription name SUB1;

select table mapping sourceSchema USER1 sourceTable CUSTOMER;

modify column encoding sourceColumn FLAG type binary;

list column encodings type source;

disconnect server;

exit;
```

Overrides the database encoding for a column and identifies that is stores UTF-8 data.

```
connect server username user1 password password1;

connect datastore name DS1 context source;

connect datastore name DS2 context target;

select subscription name SUB1;

select table mapping sourceSchema USER1 sourceTable CUSTOMER;

modify column encoding targetColumn DESCRIPTION type specify value UTF-8;

list column encodings type target;

disconnect server;

exit;
```

Parent topic: Mapping details commands

modify data translation

Use this command to modify a data translation definition for the target column in the table mapping that is the current context.

Syntax diagram

.arrow, .syntaxarrow { fill: none; stroke: black; } .arrowheadStartEnd, .arrowheadRepSep, .arrowheadRepSepReturn { stroke: black; fill: black; } .arrowheadSeq, .arrowheadStartChoice, .arrowheadAfterChoice, .arrowheadStartRepGroup, .arrowheadEndRepGroup, .arrowheadRev { stroke: none; fill: none; } rect { fill: none; stroke: none; } rect.fragref,rect.syntaxfragref { fill: none; stroke: black; } text { fill: #000000; fill-opacity: 1; font-family: IBM Plex Sans,Arial Unicode MS,Arial,Helvetica; font-style: normal; font-weight: normal; font-size: 8pt; stroke: #000000; stroke-width: 0.1; } text.var, text.syntaxvar {font-style:italic;} modify data translationtargetColumnnamebeforetranslationaftervalue

Syntax

```
modify data translation targetColumn <name> before <translation> after <value>;
```

Parameters

- targetColumn <name>

- Specifies the name of the target column.

- before <translation>

- Specifies the before value for the data translation to be modified.

- after <value>

- Specifies the after (translated) value to be set. Set this parameter to the string <NULL> to indicate that you want to translate a specific value to null.

Required role

- Administrator
- System administrator

Result

This command returns a value of 0 if the command was successful and a negative value if the command fails. When verbose mode is enabled, the command prints success message CDC1418.

Sample

Modifies the data translation for a column.

```
connect server username user1 password password1;

connect datastore name DS1 context source;

connect datastore name DS2 context target;

select subscription name SUB1;

select table mapping sourceSchema USER1 sourceTable PART_NUMBERS;

modify data translation targetColumn PART_CODE before A1 after 0000A1;

show data translation targetColumn PART_CODE;

disconnect server;

exit;
```

Parent topic: Mapping details commands

modify derived column

Use this command to modify a derived column in the table mapping that is the current context.

Syntax diagram

.arrow, .syntaxarrow { fill: none; stroke: black; } .arrowheadStartEnd, .arrowheadRepSep, .arrowheadRepSepReturn { stroke: black; fill: black; } .arrowheadSeq, .arrowheadStartChoice, .arrowheadAfterChoice, .arrowheadStartRepGroup, .arrowheadEndRepGroup, .arrowheadRev { stroke: none; fill: none; } rect { fill: none; stroke: none; } rect.fragref,rect.syntaxfragref { fill: none; stroke: black; } text { fill: #000000; fill-opacity: 1; font-family: IBM Plex Sans,Arial Unicode MS,Arial,Helvetica; font-style: normal; font-weight: normal; font-size: 8pt; stroke: #000000; stroke-width: 0.1; } text.var, text.syntaxvar {font-style:italic;} modify derived columnname columndescriptionpurposedatatypetype lengthbytesscalescalenullabletrue/yesfalse/nodateTimeFormat NONEMDYDMY YMDJULISOUSAEURJISHMSdateTimeSeparator spaceforwardSlashhyphenperiod commacolonevaluateafterbothexpressionexpression

Syntax

```
modify derived column name <column> [description <purpose>] [datatype <type>]
[length <bytes>] [scale <scale>] [nullable <value>] [dateTimeFormat <format>]
[dateTimeSeparator  <separator>][evaluate <frequency>] [expression <expression>];
```

Parameters

- **name <column>**
  - Specifies the name of the derived column.
- **[description <purpose>]**
  - Specifies the description about the purpose of the derived column.You can specify to clear the description.
- **[datatype <type>]**
  - Specifies the data type for the derived column.
- **[length <bytes>]**
  - Specifies the length of the column in bytes.
- **[scale <scale>]**
  - Specifies the scale of the numerical data type.
- **[nullable <value>]**
  - Specifies if the column is nullable.Valid values:
    - **true/yes**—Specifies that the column is nullable. Either yes or true are valid values.
    - **false/no**—Specifies that the column is not nullable. Either no or false are valid values.
- **[dateTimeFormat <frequency>]**
  - Specifies the format of the date/time data type.Valid values:
    - **NONE**—
    - **MDY**—Specifies the input format is mmddyy
    - **DMY**—Specifies the input format is ddmmyy
    - **YMD**—Specifies the input format is yymmdd
    - **JUL**—Specifies the input format is yyjjj, where jjj represent the sequence number of a day in the calendar year. jjj must be between 1, which represents

January 1st, and 366, which represents December 31st in a leap year. For jjj values less than 100, you must specify the leading zero or zeros. For example, the Julian date for February 4th is 035, which represents the 35th day of the year
  - **ISO**—Specifies the input format is the International Organization for Standardization date format (CCYY/MM/DD HH.MM.SS).
  - **USA**—Specifies the input format is the United States date format (MM/DD/CCYY HH:MM AM/PM).
  - **EUR**—Specifies the input format is the European date format (DD/MM/CCYY HH.MM.SS).
  - **JIS**—Specifies the input format is Japanese Industrial Standard (CCYY/MM/DD HH:MM:SS).
  - **HMS**—Specifies the input format is HH MM SS.

- **[dateTimeSeparator <separator>]**
  - Specifies the character that will be used as a separator when formatting date/time data types.Valid values:
  - **space**—Specifies that a space will be used as a separator.
  - **forwardSlash**—Specifies that a forward slash will be used as a separator.
  - **hyphen**—Specifies that a hyphen will be used as a separator.
  - **period**—Specifies that a period will be used as a separator.
  - **comma**—Specifies that a comma will be used as a separator.
  - **colon**—Specifies that a colon will be used as a separator.
- **[evaluate <frequency>]**
  - Specifies the evaluation frequency you want for this derived column.Valid values:
  - **after**—Select this option when you want InfoSphere® CDC to evaluate the expression of the source derived column for only the after image of an update operation.
  - **both**—Select this option when you want InfoSphere CDC to evaluate the expression of the source derived column for both the before and after images of an update operation.
- **[expression <expression>]**
  - Specifies the expression for the derived column.

Required role

- Administrator
- System administrator

Result

This command returns a value of 0 if the command was successful and a negative value if the command fails. When verbose mode is enabled, the command prints success message CDC1412.

Sample

Changes the expression for a derived column.

```
connect server username user1 password password1;

connect datastore name DS1 context source;

connect datastore name DS2 context target;

select subscription name SUB1;

select table mapping sourceSchema USER1 sourceTable TABLE_1;

modify derived column PART_NAME

 expression '%UPPER(%CONCAT(PARTNO, -, PARTNAME))';
```

```
show derived column name PART_NAME;

disconnect server;

exit;
```

Parent topic:Mapping details commands

modify operations

Use this command to modify the row or table-level operations that will be applied on the target for a table mapping. The table mapping must first be selected as the current context.

Syntax diagram

Syntax 1 (Modifies row or table-level operations for Standard table mapping type) .arrow, .syntaxarrow { fill: none; stroke: black; } .arrowheadStartEnd, .arrowheadRepSep, .arrowheadRepSepReturn { stroke: black; fill: black; } .arrowheadSeq, .arrowheadStartChoice, .arrowheadAfterChoice, .arrowheadStartRepGroup, .arrowheadEndRepGroup, .arrowheadRev { stroke: none; fill: none; } rect { fill: none; stroke: none; } rect.fragref,rect.syntaxfragref { fill: none; stroke: black; } text { fill: #000000; fill-opacity: 1; font-family: IBM Plex Sans,Arial Unicode MS,Arial,Helvetica; font-style: normal; font-weight: normal; font-size: 8pt; stroke: #000000; stroke-width: 0.1; } text.var, text.syntaxvar {font-style:italic;} modify operationsonInsertinsertRowdoNotInsertonUpdateupdateRow doNotUpdateonDeletedeleteRowdoNotDeleteonCleardeleteAlldeleteSelected doNotDeleteexpressionclausesqlAfterRefreshrefreshsqlAfterTruncatetruncate

Syntax 2 (Modifies row or table-level operations for Adaptive Apply table mapping type) .arrow, .syntaxarrow { fill: none; stroke: black; } .arrowheadStartEnd, .arrowheadRepSep, .arrowheadRepSepReturn { stroke: black; fill: black; } .arrowheadSeq, .arrowheadStartChoice, .arrowheadAfterChoice, .arrowheadStartRepGroup, .arrowheadEndRepGroup, .arrowheadRev { stroke: none; fill: none; } rect { fill: none; stroke: none; } rect.fragref,rect.syntaxfragref { fill: none; stroke: black; } text { fill: #000000; fill-opacity: 1; font-family: IBM Plex Sans,Arial Unicode MS,Arial,Helvetica; font-style: normal; font-weight: normal; font-size: 8pt; stroke: #000000; stroke-width: 0.1; } text.var, text.syntaxvar {font-style:italic;} modify operationsonInsertupdateOrInsertRowdoNotInsertonUpdate updateOrInsertRowdoNotUpdateonDeletedeleteRowIfExistsdoNotDeleteonClear deleteAlldeleteSelecteddoNotDeleteexpressionclauselogChangedOnInserttrue/yes false/nologChangedOnUpdatetrue/yesfalse/nologChangedOnDeletetrue/yesfalse/no sqlAfterRefreshrefreshsqlAfterTruncatetruncate

Syntax 3 (Modifies row or table-level operations for Audit table mapping type) .arrow, .syntaxarrow { fill: none; stroke: black; } .arrowheadStartEnd, .arrowheadRepSep, .arrowheadRepSepReturn { stroke: black; fill: black; } .arrowheadSeq, .arrowheadStartChoice, .arrowheadAfterChoice, .arrowheadStartRepGroup, .arrowheadEndRepGroup, .arrowheadRev { stroke: none; fill: none; } rect { fill: none; stroke: none; } rect.fragref,rect.syntaxfragref { fill: none; stroke: black; } text { fill: #000000; fill-opacity: 1; font-family: IBM Plex Sans,Arial Unicode MS,Arial,Helvetica; font-style: normal; font-weight: normal; font-size: 8pt; stroke: #000000; stroke-width: 0.1; } text.var, text.syntaxvar {font-style:italic;} modify operationsonInsertdoNotInsertauditonUpdatedoNotUpdate auditAfterauditBeforeAfteronDeletedoNotDeleteauditonCleardoNotDeleteaudit sqlAfterRefreshrefreshsqlAfterTruncatetruncate

Syntax 4 (Modifies row or table-level operations for Summarization table mapping type) .arrow, .syntaxarrow { fill: none; stroke: black; } .arrowheadStartEnd, .arrowheadRepSep, .arrowheadRepSepReturn { stroke: black; fill: black; } .arrowheadSeq, .arrowheadStartChoice, .arrowheadAfterChoice, .arrowheadStartRepGroup, .arrowheadEndRepGroup, .arrowheadRev { stroke: none; fill: none; } rect { fill: none; stroke: none; } rect.fragref,rect.syntaxfragref { fill: none; stroke: black; } text { fill: #000000; fill-opacity: 1; font-family: IBM Plex Sans,Arial Unicode MS,Arial,Helvetica; font-style: normal; font-weight: normal; font-size: 8pt; stroke: #000000; stroke-width: 0.1; } text.var, text.syntaxvar {font-style:italic;} modify operationsonInsertupdateOrInsertRowonUpdate updateOrInsertRowonDeleteupdateOrInsertRowonCleardeleteAlldeleteSelected doNotDeleteexpressionclausesqlAfterRefreshrefreshsqlAfterTruncatetruncate

Syntax 5 (Modifies row or table-level operations for Consolidation One-to-One table mapping type) .arrow, .syntaxarrow { fill: none; stroke: black; } .arrowheadStartEnd, .arrowheadRepSep, .arrowheadRepSepReturn { stroke: black; fill: black; } .arrowheadSeq, .arrowheadStartChoice, .arrowheadAfterChoice, .arrowheadStartRepGroup, .arrowheadEndRepGroup, .arrowheadRev { stroke: none; fill: none; } rect { fill: none; stroke: none; } rect.fragref,rect.syntaxfragref { fill: none; stroke: black; } text { fill: #000000; fill-opacity: 1; font-family: IBM Plex Sans,Arial Unicode MS,Arial,Helvetica; font-style: normal; font-weight: normal; font-size: 8pt; stroke: #000000; stroke-width: 0.1; } text.var, text.syntaxvar {font-style:italic;} modify operationsonInsertupdateOrInsertRowonUpdate updateOrInsertRowonDeletedoNotDeleteonCleardeleteAlldeleteSelected doNotDeleteexpressionclausesqlAfterRefreshrefreshsqlAfterTruncatetruncate

Syntax 6 (Modifies row or table-level operations for Consolidation One-to-Many table mapping type) .arrow, .syntaxarrow { fill: none; stroke: black; } .arrowheadStartEnd, .arrowheadRepSep, .arrowheadRepSepReturn { stroke: black; fill: black; } .arrowheadSeq, .arrowheadStartChoice, .arrowheadAfterChoice, .arrowheadStartRepGroup, .arrowheadEndRepGroup, .arrowheadRev { stroke: none; fill: none; } rect { fill: none; stroke: none; } rect.fragref,rect.syntaxfragref { fill: none; stroke: black; } text { fill: #000000; fill-opacity: 1; font-family: IBM Plex Sans,Arial Unicode MS,Arial,Helvetica; font-style: normal; font-weight: normal; font-size: 8pt; stroke: #000000; stroke-width: 0.1; } text.var, text.syntaxvar {font-style:italic;} modify operationsonInsertdoNotInsertupdateAllonUpdateupdateAll onDeletedoNotDeleteonCleardeleteAlldeleteSelecteddoNotDeleteexpressionclause sqlAfterRefreshrefreshsqlAfterTruncatetruncate

Syntax

Syntax 1 (Modifies row or table-level operations for Standard table mapping type)

```
modify operations [onInsert <insert>] [onUpdate <update>] [onDelete <delete>]
[onClear <clear>] [expression <clause>] [sqlAfterRefresh <refresh>]
[sqlAfterTruncate <truncate>];
```

Syntax 2 (Modifies row or table-level operations for Adaptive Apply table mapping type)`modify operations [onInsert <insert>] [onUpdate <update>] [onDelete <delete>]`

`[onClear <clear>] [expression <clause>] [logChangedOnInsert <message_insert>]`

`[logChangedOnUpdate <message_update>] [logChangedOnDelete <message_delete>]`

`[sqlAfterRefresh <refresh>] [sqlAfterTruncate <truncate>];`

Syntax 3 (Modifies row or table-level operations for Audit table mapping type)`modify operations [onInsert <insert>] [onUpdate <update>] [onDelete <delete>]`

`[onClear <clear>] [sqlAfterRefresh <refresh>] [sqlAfterTruncate <truncate>];`

Syntax 4 (Modifies row or table-level operations for Summarization table mapping type)`modify operations [onInsert <insert>] [onUpdate <update>] [onDelete <delete>]`

`[onClear <clear>] [expression <clause>] [sqlAfterRefresh <refresh>]`

`[sqlAfterTruncate <truncate>];`

Syntax 5 (Modifies row or table-level operations for Consolidation One-to-One table mapping type)`modify operations [onInsert <insert>] [onUpdate <update>] [onDelete <delete>]`

`[onClear <clear>] [expression <clause>] [sqlAfterRefresh <refresh>]`

`[sqlAfterTruncate <truncate>];`

Syntax 6 (Modifies row or table-level operations for Consolidation One-to-Many table mapping type)`modify operations [onInsert <insert>] [onUpdate <update>] [onDelete <delete>]`

`[onClear <clear>] [expression <clause>] [sqlAfterRefresh <refresh>]`

`[sqlAfterTruncate <truncate>];`

Parameters

Syntax 1 (Modifies row or table-level operations for Standard table mapping type)

- **[onInsert <insert>]**
  - Specifies an operation which will be applied to the target table when a row is inserted in the source table.Valid values:
  - **insertRow**—Inserts the row in the target table.
  - **doNotInsert**—Does not insert the row in the target table.
- **[onUpdate <update>]**
  - Specifies an operation which will be applied to the target table when a row is updated in the source table.Valid values:
  - **updateRow**—Updates the row on the target table.
  - **doNotUpdate**—Does not update the row on the target table.
- **[onDelete <delete>]**
  - Specifies an operation which will be applied to the target table when a row is deleted in the source table.Valid values:
  - **deleteRow**—Deletes the row on the target table.
  - **doNotDelete**—Does not delete the row on the target table.
- **[onClear <clear>]**
  - Specifies an operation which will be applied to the target table when a table-level clear or refresh has been performed on source.Valid values:
  - **deleteAll**—Deletes all rows in the target table in response to a table-level clear or refresh operation.
  - **deleteSelected**—Deletes selected rows in the target table in response to a table-level clear or refresh operation.
  - **doNotDelete**—Prevents from applying a delete to a mapped target table.
- **[expression <clause>]**
  - The SQL WHERE clause for deleting the selected rows. InfoSphere® CDC only deletes or truncates the rows for which the condition is true.

- **[sqlAfterRefresh <refresh>]**
  - Specifies an additional SQL statement that execute after applying a table refresh operation to the target table.
- **[sqlAfterTruncate <truncate>]**
  - Specifies an additional SQL statement that execute after applying a table truncate/clear operation to the target table.

Syntax 2 (Modifies row or table-level operations for Adaptive Apply table mapping type)

- **[onInsert <insert>]**
  - Specifies an operation which will be applied to the target table when a row is inserted in the source table.Valid values:
    - **updateOrInsertRow**—Updates or inserts the row on the target table.
    - **doNotInsert**—Does not insert the row in the target table.
- **[onUpdate <update>]**
  - Specifies an operation which will be applied to the target table when a row is updated in the source table.Valid values:
    - **updateOrInsertRow**—Updates or inserts the row on the target table.
    - **doNotUpdate**—Does not update the row on the target table.
- **[onDelete <delete>]**
  - Specifies an operation which will be applied to the target table when a row is deleted in the source table.Valid values:
    - **deleteRowIfExists**—Deletes the row, if it exists, on the target table.
    - **doNotUpdate**—Does not update the row, if it exists, on the target table.
- **[onClear <clear>]**
  - Specifies an operation which will be applied to the target table when a table-level clear or refresh has been performed on source.Valid values:
    - **deleteAll**—Deletes all rows in the target table in response to a table-level clear or refresh operation.
    - **deleteSelected**—Deletes selected rows in the target table in response to a table-level clear or refresh operation
    - **doNotDelete**—Prevents InfoSphere CDC from applying a delete to a mapped target table
- **[expression <clause>]**
  - The SQL WHERE clause for deleting the selected rows. InfoSphere CDC only deletes or truncates the rows for which the condition is true.
- **[logChangedOnInsert <message_insert>]**
  - Generates a log message when there is an insert on the source.Valid values:
    - **true/yes**—Generates a log message when there is an insert on the source. Either yes or true are valid values.
    - **false/no**——Does not generate a log message when there is an insert on the source. Either no or false are valid values.
- **[logChangedOnUpdate <message_update>]**
  - Generates a log message when there is an update on the source.Valid values:
    - **true/yes**—Generates a log message when there is an update on the source. Either yes or true are valid values.
    - **false/no**——Does not generate a log message when there is an update on the source. Either no or false are valid values.

- **[logChangedOnDelete <message_delete>]**
  - Generates a log message when there is a delete on the source.Valid values:
    - **true/yes**—Generates a log message when there is a delete on the source. Either yes or true are valid values.
    - **false/no**——Does not generate a log message when there is a delete on the source. Either no or false are valid values.
- **[sqlAfterRefresh <refresh>]**
  - Specifies an additional SQL statement that execute after applying a table refresh operation to the target table.
- **[sqlAfterTruncate <truncate>]**
  - Specifies an additional SQL statement that execute after applying a table truncate/clear operation to the target table.

Syntax 3 (Modifies row or table-level operations for Audit table mapping type)

- **[onInsert <insert>]**
  - Specifies an operation which will be applied to the target table when a row is inserted in the source table.Valid values:
    - **doNotInsert**—Does not insert the row in the target table.
    - **audit**—Audits the operations applied to the target table when row inserted on the source.
- **[onUpdate <update>]**
  - Specifies an operation which will be applied to the target table when a row is updated in the source table.Valid values:
    - **doNotUpdate**—Does not update the row on the target table.
    - **auditAfter**—Specifies that the target table only audits the after image when there is a change to a row in the source table
    - **auditBeforeAfter**—Specifies that the target table audits the before and after images when there is a change to a row in the source table
- **[onDelete <delete>]**
  - Specifies an operation which will be applied to the target table when a row is deleted in the source table.Valid values:
    - **doNotUpdate**—Does not update the row on the target table.
    - **audit**—Audits the operations applied to the target table when row deleted on the source.
- **[onClear <clear>]**
  - Specifies an operation which will be applied to the target table when a table-level clear or refresh has been performed on source.Valid values:
    - **doNotDelete**—Prevents InfoSphere CDC from applying a delete to a mapped target table.
    - **audit**—Audits the operations applied to the target table in response to a table-level clear or refresh operation.
- **[sqlAfterRefresh <refresh>]**
  - Specifies an additional SQL statement that execute after applying a table refresh operation to the target table.
- **[sqlAfterTruncate <truncate>]**
  - Specifies an additional SQL statement that execute after applying a table truncate/clear operation to the target table.

Syntax 4 (Modifies row or table-level operations for Summarization table mapping type)

- **[onInsert <insert>]**
    - Specifies an operation which will be applied to the target table when a row is inserted in the source table.Valid values:
        - **updateOrInsertRow**—Updates or inserts the row on the target table.
- **[onUpdate <update>]**
    - Specifies an operation which will be applied to the target table when a row is updated in the source table.Valid values:
        - **updateOrInsertRow**—Updates or inserts the row on the target table.
- **[onDelete <delete>]**
    - Specifies an operation which will be applied to the target table when a row is deleted in the source table.Valid values:
        - **updateOrInsertRow**—Updates or inserts the row on the target table.
- **[onClear <clear>]**
    - Specifies an operation which will be applied to the target table when a table-level clear or refresh has been performed on source.Valid values:
        - **deleteAll**—Deletes all rows in the target table in response to a table-level clear or refresh operation.
        - **deleteSelected**—Deletes selected rows in the target table in response to a table-level clear or refresh operation.
        - **doNotDelete**—Prevents InfoSphere CDC from applying a delete to a mapped target table.
- **[expression <clause>]**
    - The SQL WHERE clause for deleting the selected rows. InfoSphere CDC only deletes or truncates the rows for which the condition is true.
- **[sqlAfterRefresh <refresh>]**
    - Specifies an additional SQL statement that execute after applying a table refresh operation to the target table.
- **[sqlAfterTruncate <truncate>]**
    - Specifies an additional SQL statement that execute after applying a table truncate/clear operation to the target table.

Syntax 5 (Modifies row or table-level operations for Consolidation One-to-One table mapping type)

- **[onInsert <insert>]**
    - Specifies an operation which will be applied to the target table when a row is inserted in the source table.Valid values:
        - **updateOrInsertRow**—Updates or inserts the row on the target table.
- **[onUpdate <update>]**
    - Specifies an operation which will be applied to the target table when a row updated in the source table.Valid values:
        - **updateOrInsertRow**—Updates or inserts the row on the target table.
- **[onDelete <delete>]**
    - Specifies an operation which will be applied to the target table when a row deleted in the source table.Valid values:
        - **doNotDelete**—Does not delete the row on the target table.
- **[onClear <clear>]**
    - Specifies an operation which will be applied to the target table when a table-level clear or refresh has been performed on source.Valid values:

  - **deleteAll**—Deletes all rows in the target table in response to a table-level clear or refresh operation.
  - **deleteSelected**—Deletes selected rows in the target table in response to a table-level clear or refresh operation.
  - **doNotDelete**—Prevents InfoSphere CDC from applying a delete to a mapped target table.

- [expression <clause>]
- The SQL WHERE clause for deleting the selected rows. InfoSphere CDC only deletes or truncates the rows for which the condition is true.

- [sqlAfterRefresh <refresh>]
- Specifies an additional SQL statement that execute after applying a table refresh operation to the target table.

- [sqlAfterTruncate <truncate>]
- Specifies an additional SQL statement that execute after applying a table truncate/clear operation to the target table.

Syntax 6 (Modifies row or table-level operations for Consolidation One-to-Many table mapping type)

- [onInsert <insert>]
- Specifies an operation which will be applied to the target table when a row is inserted in the source table.Valid values:
  - **doNotInsert**—Does not insert the row in the target table.
  - **updateAll**—

- [onUpdate <update>]
- Specifies an operation which will be applied to the target table when a row is updated in the source table.Valid values:
  - **updateAll**—

- [onDelete <delete>]
- Specifies an operation which will be applied to the target table when a row is deleted in the source table.Valid values:
  - **doNotDelete**—Does not delete the row on the target table.

- [onClear <clear>]
- Specifies an operation which will be applied to the target table when a table-level clear or refresh has been performed on source.Valid values:
  - **deleteAll**—Deletes all rows in the target table in response to a table-level clear or refresh operation.
  - **deleteSelected**—Deletes selected rows in the target table in response to a table-level clear or refresh operation.
  - **doNotDelete**—Prevents InfoSphere CDC from applying a delete to a mapped target table.

- [expression <clause>]
- The SQL WHERE clause for deleting the selected rows. InfoSphere CDC only deletes or truncates the rows for which the condition is true.

- [sqlAfterRefresh <refresh>]
- Specifies an additional SQL statement that execute after applying a table refresh operation to the target table.

- [sqlAfterTruncate <truncate>]
- Specifies an additional SQL statement that execute after applying a table truncate/clear operation to the target table.

Required role

- Administrator
- System administrator

Result

Sample

Selects table mapping and specifies the operations that will be applied to the target table in response to a row-level operations on the source.

```
connect server username user1
password password1;
connect datastore name DS1 context source;
connect datastore name DS2 context target;
select subscription name SUB1;
select table mapping sourceSchema USER1 sourceTable TABLE_1;
modify operations onInsert insertRow onUpdate doNotUpdate onDelete doNotDelete;
disconnect server;
exit;
```

Parent topic: Mapping details commands

modify user exit cdll

Use this command to modify C/C++ DLL user exits for the table mapping.

Syntax diagram

.arrow, .syntaxarrow { fill: none; stroke: black; } .arrowheadStartEnd, .arrowheadRepSep, .arrowheadRepSepReturn { stroke: black; fill: black; } .arrowheadSeq, .arrowheadStartChoice, .arrowheadAfterChoice, .arrowheadStartRepGroup, .arrowheadEndRepGroup, .arrowheadRev { stroke: none; fill: none; } rect { fill: none; stroke: none; } rect.fragref,rect.syntaxfragref { fill: none; stroke: black; } text { fill: #000000; fill-opacity: 1; font-family: IBM Plex Sans,Arial Unicode MS,Arial,Helvetica; font-style: normal; font-weight: normal; font-size: 8pt; stroke: #000000; stroke-width: 0.1; } text.var, text.syntaxvar {font-style:italic;} modify user exit cdlldllnamenamebeforeInsertbefore_insertafterInsert after_insertbeforeUpdatebefore_updateafterUpdateafter_updatebeforeDelete before_deleteafterDeleteafter_deletebeforeRefreshbefore_refreshafterRefresh after_refreshbeforeTruncatebefore_truncateafterTruncateafter_truncate

Syntax

```
modify user exit cdll [dllname <name>] [beforeInsert <before_insert>] [afterInsert
<after_insert>] [beforeUpdate <before_update>] [afterUpdate <after_update>]
[beforeDelete <before_delete>] [afterDelete <after_delete>] [beforeRefresh
<before_refresh>] [afterRefresh <after_refresh>] [beforeTruncate <before_truncate>]
[afterTruncate <after_truncate>];
```

Parameters

- **[dllname <name>]**
  - Specifies the DLL name.
- **[beforeInsert <before_insert>]**
  - Specifies the name of the function which will be called before replicating an insert operation.
- **[afterInsert <after_insert>]**
  - Specifies the name of the function which will be called after replicating an insert operation.
- **[beforeUpdate <before_update>]**
  - Specifies the name of the function which will be called before replicating an update operation.
- **[afterUpdate <after_update>]**
  - Specifies the name of the function which will be called after replicating an update operation.
- **[beforeDelete <before_delete>]**
  - Specifies the name of the function which will be called before replicating a delete operation.
- **[afterDelete <after_delete>]**
  - Specifies the name of the function which will be called after replicating a delete operation.
- **[beforeRefresh <before_refresh>]**
  - Specifies the name of the function which will be called before replicating a refresh operation.

- [afterRefresh <after_refresh>]
- Specifies the name of the function which will be called after replicating a refresh operation.

- [beforeTruncate <before_truncate>]
- Specifies the name of the function which will be called before replicating a truncate operation.

- [afterTruncate <after_truncate>]
- Specifies the name of the function which will be called after replicating a truncate operation.

Required role

- Administrator
- System administrator

Result

This command returns a value of 0 if the command was successful and a negative value if the command fails. When verbose mode is enabled, the command prints success message CDC1420.

Sample

Selects table mapping and sets C/C++ DLL user exit.

```
connect server username user1 password
password1;
connect datastore name DS1 context source;
connect datastore name DS2 context target;
select subscription name SUB1;
select table mapping sourceSchema USER1 sourceTable TABLE_1 targetSchema USER1
targetTable TABLE_1;
modify user exit cdll dllname userExitDllName beforeInsert function1 afterDelete
function2;
disconnect server;
exit;
```

Parent topic: Mapping details commands

modify user exit function

Use this command to modify Standard Function user exit for the table mapping.

Syntax diagram

.arrow, .syntaxarrow { fill: none; stroke: black; } .arrowheadStartEnd, .arrowheadRepSep, .arrowheadRepSepReturn { stroke: black; fill: black; } .arrowheadSeq, .arrowheadStartChoice, .arrowheadAfterChoice, .arrowheadStartRepGroup, .arrowheadEndRepGroup, .arrowheadRev { stroke: none; fill: none; } rect { fill: none; stroke: none; } rect.fragref,rect.syntaxfragref { fill: none; stroke: black; } text { fill: #000000; fill-opacity: 1; font-family: IBM Plex Sans,Arial Unicode MS,Arial,Helvetica; font-style: normal; font-weight: normal; font-size: 8pt; stroke: #000000; stroke-width: 0.1; } text.var, text.syntaxvar {font-style:italic;} modify user exit functionbeforeInsertbefore_insertafterInsertafter_insert beforeUpdatebefore_updateafterUpdateafter_updatebeforeDeletebefore_delete afterDeleteafter_deletebeforeRefreshbefore_refreshafterRefreshafter_refresh beforeTruncatebefore_truncateafterTruncateafter_truncate

Syntax

```
modify user exit function [beforeInsert <before_insert>] [afterInsert
<after_insert>] [beforeUpdate <before_update>] [afterUpdate <after_update>]
[beforeDelete <before_delete>] [afterDelete <after_delete>] [beforeRefresh
<before_refresh>] [afterRefresh <after_refresh>] [beforeTruncate <before_truncate>]
[afterTruncate <after_truncate>];
```

Parameters

- [beforeInsert <before_insert>]
- Specifies the name of the program which will be called before replicating an insert operation.

- [afterInsert <after_insert>]
- Specifies the name of the program which will be called after replicating an insert operation.

- [beforeUpdate <before_update>]
- Specifies the name of the program which will be called before replicating an update operation.

- [afterUpdate <after_update>]
- Specifies the name of the program which will be called after replicating an update operation.

- [beforeDelete <before_delete>]
- Specifies the name of the program which will be called before replicating a delete operation.

- [afterDelete <after_delete>]
- Specifies the name of the program which will be called after replicating a delete operation.

- [beforeRefresh <before_refresh>]
- Specifies the name of the program which will be called before replicating a refresh operation.

- [afterRefresh <after_refresh>]
- Specifies the name of the program which will be called after replicating a refresh operation.

- [beforeTruncate <before_truncate>]
- Specifies the name of the program which will be called before replicating a truncate operation.

- [afterTruncate <after_truncate>]
- Specifies the name of the program which will be called after replicating a truncate operation.

Required role

- Administrator
- System administrator

Result

This command returns a value of 0 if the command was successful and a negative value if the command fails. When verbose mode is enabled, the command prints success message CDC1420.

Sample

Selects table mapping and sets Standard Function user exit.

```
connect server username user1 password password1;
connect datastore name DS1 context source;
connect datastore name DS2 context target;
select subscription name SUB1;
select table mapping sourceSchema USER1 sourceTable TABLE_1 targetSchema USER1 targetTable TABLE_1;
modify user exit function beforeInsert function1 afterDelete function2;
disconnect server;
exit;
```

Parent topic: Mapping details commands

modify user exit javaclass

Use this command to modify Java Class user exits for the table mapping.

Syntax diagram

.arrow, .syntaxarrow { fill: none; stroke: black; } .arrowheadStartEnd, .arrowheadRepSep, .arrowheadRepSepReturn { stroke: black; fill: black; } .arrowheadSeq, .arrowheadStartChoice, .arrowheadAfterChoice, .arrowheadStartRepGroup, .arrowheadEndRepGroup, .arrowheadRev { stroke: none; fill: none; } rect { fill: none; stroke: none; } rect.fragref,rect.syntaxfragref { fill: none; stroke: black; } text { fill: #000000; fill-opacity: 1; font-family: IBM Plex Sans,Arial Unicode MS,Arial,Helvetica; font-style: normal; font-weight: normal; font-size: 8pt; stroke: #000000; stroke-width: 0.1; } text.var, text.syntaxvar {font-style:italic;} modify user exit javaclassclassnamenameparameterparameter beforeInserttrue/yesfalse/noafterInserttrue/yesfalse/nobeforeUpdatetrue/yesfalse/no afterUpdatetrue/yesfalse/nobeforeDeletetrue/yesfalse/noafterDeletetrue/yesfalse/no beforeRefreshtrue/yesfalse/noafterRefreshtrue/yesfalse/nobeforeTruncatetrue/yes false/noafterTruncatetrue/yesfalse/no

Syntax

```
modify user exit javaclass [classname <name>] [parameter <parameter>] [beforeInsert
<before_insert>] [afterInsert <after_insert>] [beforeUpdate <before_update>]
[afterUpdate <after_update>] [beforeDelete <before_delete>] [afterDelete
<after_delete>] [beforeRefresh <before_refresh>] [afterRefresh <after_refresh>]
[beforeTruncate <before_truncate>] [afterTruncate <after_truncate>];
```

Parameters

- **[classname <name>]**
  - Specifies the name of the Java class that implements the UserExitIF interface.
- **[parameter <parameter>]**
  - Specifies the parameters that are to be used with the user exit program.
- **[beforeInsert <before_insert>]**
  - Specifies if user exit will run before replicating an insert operation.Valid values:
    - **true/yes**—Specifies that the user exit will run before replicating an insert operation. Either yes or true are valid values.
    - **false/no**—Specifies that the user exit will not run before replicating an insert operation. Either no or false are valid values.
- **[afterInsert <after_insert>]**
  - Specifies if user exit will run after replicating an insert operation.Valid values:
    - **true/yes**—Specifies that the user exit will run after replicating an insert operation. Either yes or true are valid values.
    - **false/no**—Specifies that the user exit will not run after replicating an insert operation. Either no or false are valid values.
- **[beforeUpdate <before_update>]**
  - Specifies if user exit will run before replicating an update operation.Valid values:
    - **true/yes**—Specifies that the user exit will run before replicating an update operation. Either yes or true are valid values.
    - **false/no**—Specifies that the user exit will not run before replicating an update operation. Either no or false are valid values.

- **[afterUpdate <after_update>]**
  - Specifies if user exit will run after replicating an update operation.Valid values:
    - **true/yes**—Specifies that the user exit will run after replicating an update operation. Either yes or true are valid values.
    - **false/no**—Specifies that the user exit will not run after replicating an update operation. Either no or false are valid values.
- **[beforeDelete <before_delete>]**
  - Specifies if user exit will run before replicating a delete operation.Valid values:
    - **true/yes**—Specifies that the user exit will run before replicating a delete operation. Either yes or true are valid values.
    - **false/no**—Specifies that the user exit will not run before replicating a delete operation. Either no or false are valid values.
- **[afterDelete <after_delete>]**
  - Specifies if user exit will run after replicating a delete operation.Valid values:
    - **true/yes**—Specifies that the user exit will run after replicating a delete operation. Either yes or true are valid values.
    - **false/no**—Specifies that the user exit will not run after replicating a delete operation. Either no or false are valid values.
- **[beforeRefresh <before_refresh>]**
  - Specifies if user exit will run before replicating a refresh operation.Valid values:
    - **true/yes**—Specifies that the user exit will run before replicating a refresh operation. Either yes or true are valid values.
    - **false/no**—Specifies that the user exit will not run before replicating a refresh operation. Either no or false are valid values.
- **[afterRefresh <after_refresh>]**
  - Specifies if user exit will run after replicating a refresh operation.Valid values:
    - **true/yes**—Specifies that the user exit will run after replicating a refresh operation. Either yes or true are valid values.
    - **false/no**—Specifies that the user exit will not run after replicating a refresh operation. Either no or false are valid values.
- **[beforeTruncate <before_truncate>]**
  - Specifies if user exit will run before replicating a truncate operation.Valid values:
    - **true/yes**—Specifies that the user exit will run before replicating a truncate operation. Either yes or true are valid values.
    - **false/no**—Specifies that the user exit will not run before replicating a truncate operation. Either no or false are valid values.
- **[afterTruncate <after_truncate>]**
  - Specifies if user exit will run after replicating a truncate operation.Valid values:
    - **true/yes**—Specifies that the user exit will run after replicating a truncate operation. Either yes or true are valid values.
    - **false/no**—Specifies that the user exit will not run after replicating a truncate operation. Either no or false are valid values.

Required role

- Administrator
- System administrator

Result

This command returns a value of 0 if the command was successful and a negative value if the command fails. When verbose mode is enabled, the command prints success message CDC1420.

Sample

Selects table mapping and sets Java Class user exit.

```
connect server username user1 password password1;
connect datastore name DS1 context source;
connect datastore name DS2 context target;
select subscription name SUB1;
select table mapping sourceSchema USER1 sourceTable TABLE_1 targetSchema USER1 targetTable TABLE_1;
modify user exit javaclass classname CustomHandlerClass parameter 100 beforeInsert yes afterDelete yes;
disconnect server;
exit;
```

Parent topic: Mapping details commands

modify user exit storedproc

Use this command to modify Stored Procedure user exits for the table mapping.

Syntax diagram

.arrow, .syntaxarrow { fill: none; stroke: black; } .arrowheadStartEnd, .arrowheadRepSep, .arrowheadRepSepReturn { stroke: black; fill: black; } .arrowheadSeq, .arrowheadStartChoice, .arrowheadAfterChoice, .arrowheadStartRepGroup, .arrowheadEndRepGroup, .arrowheadRev { stroke: none; fill: none; } rect { fill: none; stroke: none; } rect.fragref,rect.syntaxfragref { fill: none; stroke: black; } text { fill: #000000; fill-opacity: 1; font-family: IBM Plex Sans,Arial Unicode MS,Arial,Helvetica; font-style: normal; font-weight: normal; font-size: 8pt; stroke: #000000; stroke-width: 0.1; } text.var, text.syntaxvar {font-style:italic;} modify user exit storedprocschemanamebeforeInsertbefore_insert afterInsertafter_insertbeforeUpdatebefore_updateafterUpdateafter_update beforeDeletebefore_deleteafterDeleteafter_deletebeforeRefreshbefore_refresh afterRefreshafter_refreshbeforeTruncatebefore_truncateafterTruncateafter_truncate

Syntax

```
modify user exit storedproc [schema <name>][beforeInsert <before_insert>]
[afterInsert <after_insert>] [beforeUpdate <before_update>] [afterUpdate
<after_update>] [beforeDelete <before_delete>] [afterDelete <after_delete>]
[beforeRefresh <before_refresh>] [afterRefresh <after_refresh>] [beforeTruncate
<before_truncate>] [afterTruncate <after_truncate>];
```

Parameters

- **[schema <name>]**
  - Specifies the schema name.
- **[beforeInsert <before_insert>]**
  - Specifies the name of the stored procedure, which runs before replicating an insert operation.
- **[afterInsert <after_insert>]**
  - Specifies the name of the stored procedure, which runs after replicating an insert operation.
- **[beforeUpdate <before_update>]**
  - Specifies the name of the stored procedure, which runs before replicating an update operation.
- **[afterUpdate <after_update>]**
  - Specifies the name of the stored procedure, which runs after replicating an update operation.
- **[beforeDelete <before_delete>]**
  - Specifies the name of the stored procedure, which runs before replicating a delete operation.
- **[afterDelete <after_delete>]**
  - Specifies the name of the stored procedure, which runs after replicating a delete operation.
- **[beforeRefresh <before_refresh>]**
  - Specifies the name of the stored procedure, which runs before replicating a refresh operation.

- [afterRefresh <after_refresh>]
  - Specifies the name of the stored procedure, which runs after replicating a refresh operation.

- [beforeTruncate <before_truncate>]
  - Specifies the name of the stored procedure, which runs before replicating a truncate operation.

- [afterTruncate <after_truncate>]
  - Specifies the name of the stored procedure, which runs after replicating a truncate operation.

Required role

- Administrator
- System administrator

Result

This command returns a value of 0 if the command was successful and a negative value if the command fails. When verbose mode is enabled, the command prints success message CDC1420.

Sample

Selects table mapping and sets Stored Procedure user exit.

Parent topic: Mapping details commands

show data translation

Use this command to display the data translation settings for a target column.

Syntax diagram

.arrow, .syntaxarrow { fill: none; stroke: black; } .arrowheadStartEnd, .arrowheadRepSep, .arrowheadRepSepReturn { stroke: black; fill: black; } .arrowheadSeq, .arrowheadStartChoice, .arrowheadAfterChoice, .arrowheadStartRepGroup, .arrowheadEndRepGroup, .arrowheadRev { stroke: none; fill: none; } rect { fill: none; stroke: none; } rect.fragref,rect.syntaxfragref { fill: none; stroke: black; } text { fill: #000000; fill-opacity: 1; font-family: IBM Plex Sans,Arial Unicode MS,Arial,Helvetica; font-style: normal; font-weight: normal; font-size: 8pt; stroke: #000000; stroke-width: 0.1; } text.var, text.syntaxvar {font-style:italic;} show data translationtargetColumnname

Syntax

```
show data translation targetColumn <name>;
```

Parameters

- targetColumn <name>

- Specifies the name of the target column.

Required role

- Operator
- Administrator
- System administrator

Result

A ResultStringTable containing the before and after value translations for a column.

```
Data translation for target column STATE.


BEFORE           AFTER

---------------- ----------------

CA               California

NY               New York
```

Sample

Shows the data translations for a column.

```
connect server username user1 password password1;

connect datastore name DS1 context source;

connect datastore name DS2 context target;

select subscription name SUB1;

select table mapping sourceSchema USER1 sourceTable LOCATION;

show data translation targetColumn STATE;

disconnect server;

exit;
```

Parent topic: Mapping details commands

show derived column

Use this command to display the properties of a derived column in the table mapping that is the current context.

Syntax diagram

.arrow, .syntaxarrow { fill: none; stroke: black; } .arrowheadStartEnd, .arrowheadRepSep, .arrowheadRepSepReturn { stroke: black; fill: black; } .arrowheadSeq, .arrowheadStartChoice, .arrowheadAfterChoice, .arrowheadStartRepGroup, .arrowheadEndRepGroup, .arrowheadRev { stroke: none; fill: none; } rect { fill: none; stroke: none; } rect.fragref,rect.syntaxfragref { fill: none; stroke: black; } text { fill: #000000; fill-opacity: 1; font-family: IBM Plex Sans,Arial Unicode MS,Arial,Helvetica; font-style: normal; font-weight: normal; font-size: 8pt; stroke: #000000; stroke-width: 0.1; } text.var, text.syntaxvar {font-style:italic;} show derived columnnamecolumn

Syntax

```
show derived column name <column>;
```

Parameters

- name <column>

- Specifies the name of the derived column.

Required role

- Operator
- Administrator
- System administrator

Result

A ResultStringKeyValues object containing the properties of the derived column.

```
PROPERTY                  VALUE
------------------------- -------------------------
Name:                     PART_NAME
Description:
Data Type:                VARCHAR
Length:                   100
Scale:                    0
Nullable:                 false
Evaluation Frequency:     after
Expression:               %UPPER(PARTNAME)
```

Sample

Shows the properties of a derived column.

```
connect server username user1 password password1;
connect datastore name DS1 context source;
connect datastore name DS2 context target;
select subscription name SUB1;
select table mapping sourceSchema USER1 sourceTable TABLE_1;
add derived column PART_NAME datatype
 VARCHAR length 100 expression %UPPER(PARTNAME);
show derived column name PART_NAME;
disconnect server;
exit;
```

Parent topic: Mapping details commands

show operations

Use this command to view the row and table-level operations that will be applied on the target for a table mapping. The table mapping must be first selected as the current context.

Syntax diagram

.arrow, .syntaxarrow { fill: none; stroke: black; } .arrowheadStartEnd, .arrowheadRepSep, .arrowheadRepSepReturn { stroke: black; fill: black; } .arrowheadSeq, .arrowheadStartChoice, .arrowheadAfterChoice, .arrowheadStartRepGroup, .arrowheadEndRepGroup, .arrowheadRev { stroke: none; fill: none; } rect { fill: none; stroke: none; } rect.fragref,rect.syntaxfragref { fill: none; stroke: black; } text { fill: #000000; fill-opacity: 1; font-family: IBM Plex Sans,Arial Unicode MS,Arial,Helvetica; font-style: normal; font-weight: normal; font-size: 8pt; stroke: #000000; stroke-width: 0.1; } text.var, text.syntaxvar {font-style:italic;} show operations

Syntax

```
show operations;
```

Parameters

None.

Required role

- Operator
- Administrator
- System administrator

Result

A ResultResultList object containing ResultStringValue object containing description and ResultStringKeyValues object containing the row and table-level operations.

```
Row and table-level operations for table mapping DB2ADMIN.TABLE_1.

PROPERTY                      VALUE
----------------------------- --------------------------

ROW-LEVEL OPERATIONS
 On Insert:                   Insert Row
 On Update:                   Do Not Update
 On Delete:                   Do Not Delete

TABLE-LEVEL OPERATIONS
 On Clear/Truncate:           Delete Selected Rows
 Delete selected rows where: ZIP='12345'
 SQL After Truncate:
 SQL After Refresh:
```

Sample

Selects a table mapping and shows row and table-level operations.

```
connect server username user1 password password1;
connect datastore name DS1 context source;
connect datastore name DS2 context target;
select subscription name SUB1;
select table mapping sourceSchema USER1 sourceTable TABLE_1 targetSchema USER1 targetTable TABLE_1;
```

```
show operations;

disconnect server;

exit;
```

Parent topic: Mapping details commands

show user exit

Use this command to view user exits for the table mapping.

Syntax diagram

.arrow, .syntaxarrow { fill: none; stroke: black; } .arrowheadStartEnd, .arrowheadRepSep, .arrowheadRepSepReturn { stroke: black; fill: black; } .arrowheadSeq, .arrowheadStartChoice, .arrowheadAfterChoice, .arrowheadStartRepGroup, .arrowheadEndRepGroup, .arrowheadRev { stroke: none; fill: none; } rect { fill: none; stroke: none; } rect.fragref,rect.syntaxfragref { fill: none; stroke: black; } text { fill: #000000; fill-opacity: 1; font-family: IBM Plex Sans,Arial Unicode MS,Arial,Helvetica; font-style: normal; font-weight: normal; font-size: 8pt; stroke: #000000; stroke-width: 0.1; } text.var, text.syntaxvar {font-style:italic;} show user exit

Syntax

```
show user exit;
```

Parameters

None.

Result

The ResultResultList object containing ResultStringValue object containing description, ResultStringKeyValues object containing user exit type and parameters (if applicable), and ResultStringTable object containing settings for user exit operations.

```
User exit for table mapping DB2ADMIN.TABLE_1 - DB2ADMIN.TARGET_TABLE_1


User Exit Type:            Java Class

Class Name:                UserExitClassName

Parameter:                 200



OPERATION        BEFORE           AFTER

---------------- ---------------- ----------------

Insert           Yes               No

Update           No               Yes

Delete           No               No

Refresh          No               No

Truncate         No               No
```

Required role

- Operator
- Administrator
- System administrator

Sample

Shows the user exit settings for a table mapping.

```
connect server username user1 password password1;

connect datastore name DS1 context source;

connect datastore name DS2 context target;

select subscription name SUB1;

select table mapping sourceSchema USER1 sourceTable TABLE_1 targetSchema USER1 targetTable TABLE_1;

show user exit;

disconnect server;
```

```
exit;
```

Parent topic: Mapping details commands

unmap column

Use this command to unmap a target column in the table mapping that is the current context.

Syntax diagram

.arrow, .syntaxarrow { fill: none; stroke: black; } .arrowheadStartEnd, .arrowheadRepSep, .arrowheadRepSepReturn { stroke: black; fill: black; } .arrowheadSeq, .arrowheadStartChoice, .arrowheadAfterChoice, .arrowheadStartRepGroup, .arrowheadEndRepGroup, .arrowheadRev { stroke: none; fill: none; } rect { fill: none; stroke: none; } rect.fragref,rect.syntaxfragref { fill: none; stroke: black; } text { fill: #000000; fill-opacity: 1; font-family: IBM Plex Sans,Arial Unicode MS,Arial,Helvetica; font-style: normal; font-weight: normal; font-size: 8pt; stroke: #000000; stroke-width: 0.1; } text.var, text.syntaxvar {font-style:italic;} unmap columntargetColumnname

Syntax

```
unmap column targetColumn <name>;
```

Parameters

- targetColumn <name>

- Specifies the target column to unmap.

Required role

- Administrator
- System administrator

Result

This command returns a value of 0 if the command was successful and a negative value if the command fails. When verbose mode is enabled, the command prints success message CDC1415. **Sample**

Unmaps a target column and displays the current column mappings.

```
connect server username user1 password password1;
connect datastore name DS1 context source;
connect datastore name DS2 context target;
select subscription name SUB1;
select table mapping sourceSchema USER1 sourceTable CUSTOMER;
unmap column targetColumn REWARDCODE;
list column mappings;
disconnect server;
exit;
```

Parent topic: Mapping details commands

Replication commands

- **clear datastore events**
- **clear subscription events**
- **end replication**
- **list datastore events**
- **list subscription events**
- **list subscription performance metrics**
- **list table performance metrics**
- **monitor replication**
- **monitor subscription activity**
- **monitor subscription busy tables**
- **monitor subscription latency**
- **monitor subscription performance**
- **monitor subscription refresh**
- **monitor table performance**
- **show datastore event details**
- **show subscription event details**
- **start mirroring**
- **start refresh**

Parent topic:CHCCLP Commands

clear datastore events

Use this command to clear the events logged for a datastore.

Syntax diagram

.arrow, .syntaxarrow { fill: none; stroke: black; } .arrowheadStartEnd, .arrowheadRepSep, .arrowheadRepSepReturn { stroke: black; fill: black; } .arrowheadSeq, .arrowheadStartChoice, .arrowheadAfterChoice, .arrowheadStartRepGroup, .arrowheadEndRepGroup, .arrowheadRev { stroke: none; fill: none; } rect { fill: none; stroke: none; } rect.fragref,rect.syntaxfragref { fill: none; stroke: black; } text { fill: #000000; fill-opacity: 1; font-family: IBM Plex Sans,Arial Unicode MS,Arial,Helvetica; font-style: normal; font-weight: normal; font-size: 8pt; stroke: #000000; stroke-width: 0.1; } text.var, text.syntaxvar {font-style:italic;} clear datastore eventsnamedatastoretypesourcetargetboth

Syntax

```
clear datastore events [name <datastore>] type <value>;
```

Parameters

- [name <datastore>]

- Specifies the name of the datastore for which events will be cleared.If a name is not provided, the datastore that is currently set to source and target context will be used.

- type <value>

- Specifies whether to clear events from the source, events from the target, or both sets of events.Valid values:
  - **source**—Clears events from the source only.
  - **target**—Clears events from the target only.
  - **both**—Clears events from both the source and target.

Required role

- Administrator
- System administrator

Result

This command returns a value of 0 if the command was successful and a negative value if the command fails. When verbose mode is enabled, the command prints success message CDC1320. **Sample**

Connects to a datastore and clears the source and target events logged at the datastore level.

```
connect server username user1 password password1;

connect datastore name DS1;

clear datastore events type both;

disconnect datastore name DS1;

disconnect server;
```

Parent topic: Replication commands

clear subscription events

Use this command to clear the events logged for a subscription.

Syntax diagram

.arrow, .syntaxarrow { fill: none; stroke: black; } .arrowheadStartEnd, .arrowheadRepSep, .arrowheadRepSepReturn { stroke: black; fill: black; } .arrowheadSeq, .arrowheadStartChoice, .arrowheadAfterChoice, .arrowheadStartRepGroup, .arrowheadEndRepGroup, .arrowheadRev { stroke: none; fill: none; } rect { fill: none; stroke: none; } rect.fragref,rect.syntaxfragref { fill: none; stroke: black; } text { fill: #000000; fill-opacity: 1; font-family: IBM Plex Sans,Arial Unicode MS,Arial,Helvetica; font-style: normal; font-weight: normal; font-size: 8pt; stroke: #000000; stroke-width: 0.1; } text.var, text.syntaxvar {font-style:italic;} clear subscription eventsnamesubscriptiontypesourcetargetboth

Syntax

```
clear subscription events [name <subscription>] type <value>;
```

Parameters

- [name <subscription>]

- Specifies the name of the subscription for which events will be cleared. If a name is not provided, the subscription that is currently identified as the context will be used. To view the current context, use the show context command.

- type <value>

- Specifies whether to clear events from the source, events from the target, or both sets of events.Valid values:
  - **source**—Clears events from the source only.
  - **target**—Clears events from the target only.
  - **both**—Clears events from both the source and target.

Required role

- Administrator
- System administrator

Result

This command returns a value of 0 if the command was successful and a negative value if the command fails. When verbose mode is enabled, the command prints success message CDC1321.

Sample

Selects a subscription and clears the source events.

```
connect server username user1 password password1;

connect datastore name DS1 context source;

connect datastore name DS2 context target;

select subscription name SUB1;

clear subscription events type source;

disconnect server;

exit;
```

Parent topic: Replication commands

end replication

Use this command to end replication for a subscription.The table mapping context is cleared after executing this command if the context is currently set to this subscription.

Syntax diagram

.arrow, .syntaxarrow { fill: none; stroke: black; } .arrowheadStartEnd, .arrowheadRepSep, .arrowheadRepSepReturn { stroke: black; fill: black; } .arrowheadSeq, .arrowheadStartChoice, .arrowheadAfterChoice, .arrowheadStartRepGroup, .arrowheadEndRepGroup, .arrowheadRev { stroke: none; fill: none; } rect { fill: none; stroke: none; } rect.fragref,rect.syntaxfragref { fill: none; stroke: black; } text { fill: #000000; fill-opacity: 1; font-family: IBM Plex Sans,Arial Unicode MS,Arial,Helvetica; font-style: normal; font-weight: normal; font-size: 8pt; stroke: #000000; stroke-width: 0.1; } text.var, text.syntaxvar {font-style:italic;} end replicationnamesubscriptionmethodnormalimmediateabort scheduledEndendTypenowlogPositionlocalTimeendValuevaluewait0seconds

Syntax

```
end replication [name <subscription>] [method <stop>] [endType <type>]

[endValue <value>] [wait <seconds>];
```

Parameters

- **[name <subscription>]**
  - Specifies the name of the subscription.If a name is not provided, the subscription that is currently identified as the context will be used. To view the current context, use the show context command.
- **[method <stop>]**
  - Specifies the method to stop replication.Valid values:
    - **normal**—InfoSphere® CDC 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.
    - **immediate**—InfoSphere CDC stops all in progress work and then ends replication. Starting replication after using this option can be slower than using the **normal** option. 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.
      Attention: Use this option if business reasons require replication to end faster than normal at the expense of a slower start when you resume replication on the subscription.
    - **abort**—InfoSphere CDC stops all in progress work and then ends replication rapidly. Starting replication after using this option can be much slower than the **normal** option. 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.
Attention: 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 subscription. A sudden business requirement for an unplanned shut down of your source system may require this option for ending replication.
- **scheduledEnd**—This option will process all committed database changes in the source database and then end replication at the indicated point with the **normal** option. This option is not available if all the tables in a subscription are currently refreshing.

Default value: normal

- **[endType <type>]**
  - Identifies the scheduled end type when the method value is **scheduledEnd**. Valid values:
    - **now**—End replication at the current source system time in your source database log.
    - **logPosition**—End replication with the **normal** option at the specified log position. InfoSphere CDC displays the format of the log position for the source datastore. This option is only available for supported source datastores.
    - **localTime**—End replication with the **normal** option at the specified log date and time. InfoSphere CDC displays the UTC offset (in minutes) of the source database.

    Default value: now
- **[endValue <value>]**
  - When the endType value is **logPosition**, this value specifies a log position for the end value. When the endType value is **localTime**, this value specifies the exact date and time in the source engine's time zone using the yyyyMMddHHmmssSSS format.
- **[wait <seconds>]**
  - Specifies how long to wait (in seconds) for replication to stop. Use -1 to wait indefinitely, 0 for no wait, or the maximum number of seconds to wait before continuing with script execution.Default value: 0

Required role

- Operator
- Administrator
- System administrator

Result

This command returns a value of 0 if the command was successful and a negative value if the command fails. When verbose mode is enabled, the command prints success message CDC1314.

Sample

Selects a subscription and ends replication using normal shut down.

```
connect server username user1 password password1;

connect datastore name DS1 context source;
connect datastore name DS2 context target;
select subscription name SUB1;
end replication;
disconnect server;
exit;
```

Selects a subscription and schedules it to end replicating.

```
connect server username user1 password password1;
connect datastore name DS1 context source;
connect datastore name DS2 context target;
select subscription name SUB1;
end replication method scheduledEnd endType now;
disconnect server;
exit;
```

Parent topic: Replication commands

list datastore events

Use this command to list the events for a datastore.

Syntax diagram

.arrow, .syntaxarrow { fill: none; stroke: black; } .arrowheadStartEnd, .arrowheadRepSep, .arrowheadRepSepReturn { stroke: black; fill: black; } .arrowheadSeq, .arrowheadStartChoice, .arrowheadAfterChoice, .arrowheadStartRepGroup, .arrowheadEndRepGroup, .arrowheadRev { stroke: none; fill: none; } rect { fill: none; stroke: none; } rect.fragref,rect.syntaxfragref { fill: none; stroke: black; } text { fill: #000000; fill-opacity: 1; font-family: IBM Plex Sans,Arial Unicode MS,Arial,Helvetica; font-style: normal; font-weight: normal; font-size: 8pt; stroke: #000000; stroke-width: 0.1; } text.var, text.syntaxvar {font-style:italic;} list datastore eventsnamedatastoretypesourcetargetcount500number

Syntax

```
list datastore events [name <datastore>] type <retrieve> [count <number>];
```

Parameters

- [name <datastore>]

- Specifies the name of the datastore for which events will be displayed.If a name is not provided, the datastore that is currently set to source and target context will be used.

- type <retrieve>

- Specifies whether to retrieve events from the source or from the target.Valid values:
  - **source**—Clears events from the source only.
  - **target**—Clears events from the target only.

- [count <number>]

- Specifies the number of events to retrieve.Valid range: 1 to 10000
  Default value: 500

Required role

Any

Result

A ResultStringTable of datastore events. Each event is identified by a row number that can be used to fetch more details using the show datastore event details command.

Sample

Lists 500 source events from a datastore.

```
connect server username user1 password password1;

connect datastore name DS1;

list datastore events type source count 500;

disconnect datastore name DS1;

disconnect server;

exit;
```

Parent topic: Replication commands

list subscription events

Use this command to list the events for a subscription.

Syntax diagram

.arrow, .syntaxarrow { fill: none; stroke: black; } .arrowheadStartEnd, .arrowheadRepSep, .arrowheadRepSepReturn { stroke: black; fill: black; } .arrowheadSeq, .arrowheadStartChoice, .arrowheadAfterChoice, .arrowheadStartRepGroup, .arrowheadEndRepGroup, .arrowheadRev { stroke: none; fill: none; } rect { fill: none; stroke: none; } rect.fragref,rect.syntaxfragref { fill: none; stroke: black; } text { fill: #000000; fill-opacity: 1; font-family: IBM Plex Sans,Arial Unicode MS,Arial,Helvetica; font-style: normal; font-weight: normal; font-size: 8pt; stroke: #000000; stroke-width: 0.1; } text.var, text.syntaxvar {font-style:italic;} list subscription eventsnamesubscriptiontypesourcetargetcount500 number

Syntax

```
list subscription events [name <subscription>] type <retrieve> [count <number>];
```

Parameters

- [name <subscription>]

- Specifies the name of the subscription.If a name is not provided, the subscription that is currently identified as the context will be used. To view the current context, use the show context command.

- type <retrieve>

- Specifies whether to retrieve events from the source or from the target.Valid values:
  - **source**—Clears events from the source only.
  - **target**—Clears events from the target only.

- [count <number>]

- Specifies the number of events to retrieve.Valid range: 1 to 10000
  Default value: 500

Required role

Any

Result

A ResultStringTable of subscription events. Each event is identified by a row number that can be used to fetch more details using the show subscription event details command.

Sample

Selects a subscription and lists 500 source and target subscription events.

```
connect server username user1 password password1;

connect datastore name DS1 context source;

connect datastore name DS2 context target;

list subscription events name SUB1 type source count 500;

list subscription events name SUB1 type target count 500;

disconnect server;

exit;
```

Parent topic: Replication commands

list subscription performance metrics

Use this command to view a list of the available performance metrics of the subscription.

Syntax diagram

.arrow, .syntaxarrow { fill: none; stroke: black; } .arrowheadStartEnd, .arrowheadRepSep, .arrowheadRepSepReturn { stroke: black; fill: black; } .arrowheadSeq, .arrowheadStartChoice, .arrowheadAfterChoice, .arrowheadStartRepGroup, .arrowheadEndRepGroup, .arrowheadRev { stroke: none; fill: none; } rect { fill: none; stroke: none; } rect.fragref,rect.syntaxfragref { fill: none; stroke: black; } text { fill: #000000; fill-opacity: 1; font-family: IBM Plex Sans,Arial Unicode MS,Arial,Helvetica; font-style: normal; font-weight: normal; font-size: 8pt; stroke: #000000; stroke-width: 0.1; } text.var, text.syntaxvar {font-style:italic;} list subscription performance metricsnamesubscription

Syntax

```
list subscription performance metrics [name <subscription>];
```

Parameters

- [name <subscription>]

- Specifies the name of the subscription.If a name is not provided, the subscription that is currently identified as the context will be used. To view the current context, use the show context command.

Required role

Any

Result

The ResultStringTable object containing list of available performance metrics for a subscription.`subscription.`

```
METRIC                                               ID
--------------------------------------------------- ------
Datastore - DatastoreName
   Time check missed count                          1001
   Free memory - bytes                              1002
   Maximum memory - bytes                           1003
   Total memory - bytes                             1004
   Garbage collections - count                      1005
   Garbage collection CPU time - milliseconds       1006
   Global memory manager - bytes                    1007
   Network errors - count                           2401
```

Sample

Select a subscription and lists its available performance metrics.

```
connect server username user1 password password1;
connect datastore name DS1 context source;
connect datastore name DS2 context target;
select subscription name SUB1;
list subscription performance metrics;
disconnect server;
exit;
```

Parent topic: Replication commands

list table performance metrics

Use this command to view a list of the available table level performance metrics of the subscription.

Syntax diagram

.arrow, .syntaxarrow { fill: none; stroke: black; } .arrowheadStartEnd, .arrowheadRepSep, .arrowheadRepSepReturn { stroke: black; fill: black; } .arrowheadSeq, .arrowheadStartChoice, .arrowheadAfterChoice, .arrowheadStartRepGroup, .arrowheadEndRepGroup, .arrowheadRev { stroke: none; fill: none; } rect { fill: none; stroke: none; } rect.fragref,rect.syntaxfragref { fill: none; stroke: black; } text { fill: #000000; fill-opacity: 1; font-family: IBM Plex Sans,Arial Unicode MS,Arial,Helvetica; font-style: normal; font-weight: normal; font-size: 8pt; stroke: #000000; stroke-width: 0.1; } text.var, text.syntaxvar {font-style:italic;} list table performance metricsnamesubscription

Syntax

```
list table performance metrics [name <subscription>];
```

Parameters

- [name <subscription>]

- Specifies the name of the subscription.If a name is not provided, the subscription that is currently identified as the context will be used. To view the current context, use the show context command.

Required role

Any

Result

The ResultStringTable object containing the list of the available table level performance metrics.

```
METRIC                                                     ID
---------------------------------------------------- ------
Source Engine
   Source engine bytes processed                       1601
   Source pre-filter inserts                           1602
   Source pre-filter updates                           1603
   Source pre-filter deletes                           1604
Target Apply
   Target database bytes processed                     1701
   Target apply inserts - count                        1702
   Target apply updates - count                        1703
   Target apply deletes - count                        1704
   Target apply slow database operations - count       1705
```

Sample

Selects a subscription and lists its available table level performance metrics.

```
connect server username user1 password password1;
connect datastore name DS1 context source;
connect datastore name DS2 context target;
select subscription name SUB1;
list table performance metrics;
disconnect server;
exit;
```

Parent topic: Replication commands

monitor replication

Use this command to monitor replication state and latency of subscriptions

Syntax diagram

.arrow, .syntaxarrow { fill: none; stroke: black; } .arrowheadStartEnd, .arrowheadRepSep, .arrowheadRepSepReturn { stroke: black; fill: black; } .arrowheadSeq, .arrowheadStartChoice, .arrowheadAfterChoice, .arrowheadStartRepGroup, .arrowheadEndRepGroup, .arrowheadRev { stroke: none; fill: none; } rect { fill: none; stroke: none; } rect.fragref,rect.syntaxfragref { fill: none; stroke: black; } text { fill: #000000; fill-opacity: 1; font-family: IBM Plex Sans,Arial Unicode MS,Arial,Helvetica; font-style: normal; font-weight: normal; font-size: 8pt; stroke: #000000; stroke-width: 0.1; } text.var, text.syntaxvar {font-style:italic;} monitor replicationfilterallcontextdatastoresubscriptionnamesubscription

Syntax

```
monitor replication [filter <value>] [name <subscription>];
```

Parameters

- [filter <value>]

- Specifies how the subscriptions to monitor will be selected. If a datastore or subscription filter is selected and the name is not provided, the current context will be used.Valid values:
  - **all**—Requests monitoring information for all available subscriptions
  - **datastore**—Filters the subscriptions by datastore.
  - **subscription**—Filters the subscriptions by name.

  Default value: all

- [name <subscription>]

- Specifies the name of the datastore or subscription based on the filter parameter. If a subscription name is provided, the subscription must match the current source and target datastores in context

Required role

Any

Result

A ResultStringTable containing the current replication state, scheduled end setting (when applicable), and current latency for the filtered set of subscriptions

Sample

Polls the datastore for its current replication state, scheduled end setting (when applicable), and current latency threshold for each subscription.

```
connect server username user1 password password1;

connect datastore name DS1;

monitor replication;

disconnect server;

exit;
```

Parent topic: Replication commands

monitor subscription activity

Use this command to monitor subscription activity statistics.

Syntax diagram

.arrow, .syntaxarrow { fill: none; stroke: black; } .arrowheadStartEnd, .arrowheadRepSep, .arrowheadRepSepReturn { stroke: black; fill: black; } .arrowheadSeq, .arrowheadStartChoice, .arrowheadAfterChoice, .arrowheadStartRepGroup, .arrowheadEndRepGroup, .arrowheadRev { stroke: none; fill: none; } rect { fill: none; stroke: none; } rect.fragref,rect.syntaxfragref { fill: none; stroke: black; } text { fill: #000000; fill-opacity: 1; font-family: IBM Plex Sans,Arial Unicode MS,Arial,Helvetica; font-style: normal; font-weight: normal; font-size: 8pt; stroke: #000000; stroke-width: 0.1; } text.var, text.syntaxvar {font-style:italic;} monitor subscription activitynamesubscription

Syntax

```
monitor subscription activity [name <subscription>];
```

Parameters

- [name <subscription>]

- Specifies the name of the subscription.If a name is not provided, the subscription that is currently identified as the context will be used. To view the current context, use the show context command.

Required role

Any

Result

A ResultStringTable containing the source and target byte and operational counts for the subscription.

Sample

Polls the activity (byte and operation counts) for a subscription and displays the source and target values.

```
connect server username user1 password password1;

connect datastore name DS1 context source;

connect datastore name DS2 context target;

select subscription name SUB1;

monitor subscription activity;

disconnect server;

exit;
```

Parent topic: Replication commands

monitor subscription busy tables

Use this command to monitor the busy tables in a subscription. The subscription must be actively mirroring in order to capture busy table information.

Syntax diagram

.arrow, .syntaxarrow { fill: none; stroke: black; } .arrowheadStartEnd, .arrowheadRepSep, .arrowheadRepSepReturn { stroke: black; fill: black; } .arrowheadSeq, .arrowheadStartChoice, .arrowheadAfterChoice, .arrowheadStartRepGroup, .arrowheadEndRepGroup, .arrowheadRev { stroke: none; fill: none; } rect { fill: none; stroke: none; } rect.fragref,rect.syntaxfragref { fill: none; stroke: black; } text { fill: #000000; fill-opacity: 1; font-family: IBM Plex Sans,Arial Unicode MS,Arial,Helvetica; font-style: normal; font-weight: normal; font-size: 8pt; stroke: #000000; stroke-width: 0.1; } text.var, text.syntaxvar {font-style:italic;} monitor subscription busy tablesnamesubscription

Syntax

```
monitor subscription busy tables [name <subscription>];
```

Parameters

- [name <subscription>]

- Specifies the name of the subscription.If a name is not provided, the subscription that is currently identified as the context will be used. To view the current context, use the show context command.

Required role

Any

Result

A ResultStringTable containing the busiest tables in the subscription. For each table, the number of bytes, and relative percentage of activity is presented.

Sample

Polls the subscription for the list of busiest tables.

```
connect server username user1 password password1;

connect datastore name DS1 context source;

connect datastore name DS2 context target;

select subscription name SUB1;

monitor subscription busy tables;

disconnect server;

exit;
```

Parent topic: Replication commands

monitor subscription latency

Use this command to monitor subscription latency statistics.

Syntax diagram

.arrow, .syntaxarrow { fill: none; stroke: black; } .arrowheadStartEnd, .arrowheadRepSep, .arrowheadRepSepReturn { stroke: black; fill: black; } .arrowheadSeq, .arrowheadStartChoice, .arrowheadAfterChoice, .arrowheadStartRepGroup, .arrowheadEndRepGroup, .arrowheadRev { stroke: none; fill: none; } rect { fill: none; stroke: none; } rect.fragref,rect.syntaxfragref { fill: none; stroke: black; } text { fill: #000000; fill-opacity: 1; font-family: IBM Plex Sans,Arial Unicode MS,Arial,Helvetica; font-style: normal; font-weight: normal; font-size: 8pt; stroke: #000000; stroke-width: 0.1; } text.var, text.syntaxvar {font-style:italic;} monitor subscription latencynamesubscription

Syntax

```
monitor subscription latency [name <subscription>];
```

Parameters

- [name <subscription>]

- Specifies the name of the subscription.If a name is not provided, the subscription that is currently identified as the context will be used. To view the current context, use the show context command.

Required role

Any

Result

A ResultStringTable containing the latency threshold and current latency value for the subscription.

Sample

Polls latency information for the subscription and displays the current value.

```
connect server username user1 password password1;

connect datastore name DS1 context source;

connect datastore name DS2 context target;

select subscription name SUB1;

monitor subscription latency;

disconnect server;

exit;
```

Parent topic: Replication commands

monitor subscription performance

Use this command to monitor subscription performance statistics.

Syntax diagram

Syntax 1 (Monitors table performance statistics for the table mapping selected as the current context) .arrow, .syntaxarrow { fill: none; stroke: black; } .arrowheadStartEnd, .arrowheadRepSep, .arrowheadRepSepReturn { stroke: black; fill: black; } .arrowheadSeq, .arrowheadStartChoice, .arrowheadAfterChoice, .arrowheadStartRepGroup, .arrowheadEndRepGroup, .arrowheadRev { stroke: none; fill: none; } rect { fill: none; stroke: none; } rect.fragref,rect.syntaxfragref { fill: none; stroke: black; } text { fill: #000000; fill-opacity: 1; font-family: IBM Plex Sans,Arial Unicode MS,Arial,Helvetica; font-style: normal; font-weight: normal; font-size: 8pt; stroke: #000000; stroke-width: 0.1; } text.var, text.syntaxvar {font-style:italic;} monitor subscription performancenamesubscriptionmetricIDslist

S Syntax

```
monitor subscription performance [name <subscription>] metricIDs <list>;
```

Parameters

- name <subscription>

- Specifies the name of the subscription.If a name is not provided, the subscription that is currently identified as the context will be used. To view the current context, use the show context command.

- metricIDs <list>

- Specifies a comma-delimited list of statistic metric IDs used to monitor the performance of the subscription.

Required role

Any

Result

The ResultStringTable object containing performance statistics for a subscription.

Sample

Polls the performance statistics (byte and operation counts) for a subscription and displays the source and target values.

```
connect server username user1 password password1;
connect datastore name DS1 context source;
connect datastore name DS2 context target;
select subscription name SUB1;
list subscription performance statistics;
monitor subscription performance metricIDs 1301,101,102,202,207,208,209,2305;
disconnect server;
exit;
```

Parent topic: Replication commands

monitor subscription refresh

Use this command to monitor the refresh status of the tables in a subscription.If the subscription is currently being refreshed, the result is a table of statistics for each table flagged for refresh. If the subscription is not actively refreshing tables, the result is the last known refresh statistics, if any are available.

Syntax diagram

.arrow, .syntaxarrow { fill: none; stroke: black; } .arrowheadStartEnd, .arrowheadRepSep, .arrowheadRepSepReturn { stroke: black; fill: black; } .arrowheadSeq, .arrowheadStartChoice, .arrowheadAfterChoice, .arrowheadStartRepGroup, .arrowheadEndRepGroup, .arrowheadRev { stroke: none; fill: none; } rect { fill: none; stroke: none; } rect.fragref,rect.syntaxfragref { fill: none; stroke: black; } text { fill: #000000; fill-opacity: 1; font-family: IBM Plex Sans,Arial Unicode MS,Arial,Helvetica; font-style: normal; font-weight: normal; font-size: 8pt; stroke: #000000; stroke-width: 0.1; } text.var, text.syntaxvar {font-style:italic;} monitor subscription refreshnamesubscription

Syntax

```
monitor subscription refresh [name <subscription>];
```

Parameters

- [name <subscription>]

- Specifies the name of the subscription.If a name is not provided, the subscription that is currently identified as the context will be used. To view the current context, use the show context command.

Required role

Any

Result

A ResultStringTable containing the list of tables currently being refreshed, or the list of tables in the last performed refresh. For each table, the statistics include the number of rows read from the source and the number applied on the target.

Sample

Polls the subscription for the statistics of the tables currently being refreshed.

```
connect server username user1 password password1;

connect datastore name DS1 context source;

connect datastore name DS2 context target;

select subscription name SUB1;

monitor subscription refresh;

disconnect server;

exit;
```

Parent topic: Replication commands

monitor table performance

Use this command to monitor table performance statistics for a table mapping selected as the current context or a rule table selected for the subscription.

Syntax diagram

Syntax 1 (Monitors table performance statistics for the table mapping selected as the current context) .arrow, .syntaxarrow { fill: none; stroke: black; } .arrowheadStartEnd, .arrowheadRepSep, .arrowheadRepSepReturn { stroke: black; fill: black; } .arrowheadSeq, .arrowheadStartChoice, .arrowheadAfterChoice, .arrowheadStartRepGroup, .arrowheadEndRepGroup, .arrowheadRev { stroke: none; fill: none; } rect { fill: none; stroke: none; } rect.fragref,rect.syntaxfragref { fill: none; stroke: black; } text { fill: #000000; fill-opacity: 1; font-family: IBM Plex Sans,Arial Unicode MS,Arial,Helvetica; font-style: normal; font-weight: normal; font-size: 8pt; stroke: #000000; stroke-width: 0.1; } text.var, text.syntaxvar {font-style:italic;} monitor table performancemetricIDslist

Syntax 2 (Monitors table performance statistics for a rule table that specified by schema and name) .arrow, .syntaxarrow { fill: none; stroke: black; } .arrowheadStartEnd, .arrowheadRepSep, .arrowheadRepSepReturn { stroke: black; fill: black; } .arrowheadSeq, .arrowheadStartChoice, .arrowheadAfterChoice, .arrowheadStartRepGroup, .arrowheadEndRepGroup, .arrowheadRev { stroke: none; fill: none; } rect { fill: none; stroke: none; } rect.fragref,rect.syntaxfragref { fill: none; stroke: black; } text { fill: #000000; fill-opacity: 1; font-family: IBM Plex Sans,Arial Unicode MS,Arial,Helvetica; font-style: normal; font-weight: normal; font-size: 8pt; stroke: #000000; stroke-width: 0.1; } text.var, text.syntaxvar {font-style:italic;} monitor table performanceschemaschematablenamemetricIDslist

Syntax

Syntax 1 (Monitors table performance statistics for the table mapping selected as the current context)`monitor table performance metricIDs <list>;`

Syntax 2 (Monitors table performance statistics for a rule table that specified by schema and name)`monitor table performance schema <schema> table <name> metricIDs <list>;`

Parameters

Syntax 1 (Monitors table performance statistics for the table mapping selected as the current context)

- metricIDs <list>

- Specifies a comma-delimited list of statistic metric IDs used to monitor the performance of the source and target tables of the table mapping.

Syntax 2 (Monitors table performance statistics for a rule table that specified by schema and name)

- schema <schema>

- Specifies the schema name of the table.

- table <name>

- Specifies the name of the table.

- metricIDs <list>

- Specifies a comma-delimited list of statistic metric IDs used to monitor the performance of the source and target tables of the table mapping.

Required role

Any

Result

The ResultStringTable object containing performance statistics for a table.

Sample

Sample 1: Polls the performance statistics (byte and operation counts) for a table mapping and displays the source and target values.

```
connect server username user1 password
password1;
connect datastore name DS1 context source;
connect datastore name DS2 context target;
select subscription name SUB1;
select table mapping sourceTable TABLE_1;
list table performance statistics;
monitor table performance metricIDs 1601,1602,1603,1604;
disconnect server;
exit;
```

Sample 2: Polls the performance statistics (byte and operation counts) for a rule table and displays the source and target values.

```
connect server username user1 password
password1;
connect datastore name DS1 context source;
connect datastore name DS2 context target;
select subscription name SUB1;
list rule set tables;
list table performance statistics;
monitor table performance schema USER1 table TABLE1 statisticIDs 1601,1602,1603,1604;
disconnect server;
exit;
```

Parent topic: Replication commands

show datastore event details

Use this command to view the details of a datastore event. The list datastore events command must be called before running this command.

Syntax diagram

.arrow, .syntaxarrow { fill: none; stroke: black; } .arrowheadStartEnd, .arrowheadRepSep, .arrowheadRepSepReturn { stroke: black; fill: black; } .arrowheadSeq, .arrowheadStartChoice, .arrowheadAfterChoice, .arrowheadStartRepGroup, .arrowheadEndRepGroup, .arrowheadRev { stroke: none; fill: none; } rect { fill: none; stroke: none; } rect.fragref,rect.syntaxfragref { fill: none; stroke: black; } text { fill: #000000; fill-opacity: 1; font-family: IBM Plex Sans,Arial Unicode MS,Arial,Helvetica; font-style: normal; font-weight: normal; font-size: 8pt; stroke: #000000; stroke-width: 0.1; } text.var, text.syntaxvar {font-style:italic;} show datastore event detailsrownumber

Syntax

```
show datastore event details row <number>;
```

Parameters

- row <number>

- Specifies the row number from the previous list datastore events result.

Required role

Any

Result

A ResultStringValue containing the details of the event.

Sample

Lists the events for a datastore and displays details about the first one.

```
connect server
username user1 password password1;
connect datastore name datastore1;
list datastore events context source;
show datastore event details row 1;
disconnect datastore;
disconnect server;
exit;
```

Parent topic: Replication commands

Related reference

- list datastore events

show subscription event details

Use this command to view the details of a subscription event. The list subscription events command must be called before running this command.

Syntax diagram

.arrow, .syntaxarrow { fill: none; stroke: black; } .arrowheadStartEnd, .arrowheadRepSep, .arrowheadRepSepReturn { stroke: black; fill: black; } .arrowheadSeq, .arrowheadStartChoice, .arrowheadAfterChoice, .arrowheadStartRepGroup, .arrowheadEndRepGroup, .arrowheadRev { stroke: none; fill: none; } rect { fill: none; stroke: none; } rect.fragref,rect.syntaxfragref { fill: none; stroke: black; } text { fill: #000000; fill-opacity: 1; font-family: IBM Plex Sans,Arial Unicode MS,Arial,Helvetica; font-style: normal; font-weight: normal; font-size: 8pt; stroke: #000000; stroke-width: 0.1; } text.var, text.syntaxvar {font-style:italic;} show subscription event detailsrownumber

Syntax

```
show subscription event details row <number>;
```

Parameters

- row <number>

- Specifies the row number from the previous list subscription events result.

Required role

Any

Result

A ResultStringValue containing the details of the event.

Sample

Lists the events for a subscription and displays details about the first one.

```
connect server username user1 password password1;
connect datastore name datastore1;
select subscription name SUB1;
list subscriptions events context source;
show subscription event details row 1;
disconnect datastore;
disconnect server;
exit;
```

Parent topic: Replication commands

Related reference

- list subscription events

start mirroring

Use this command to start mirroring a subscription.The table mapping context is cleared after executing this command if the context is currently set to this subscription. If the start mirroring command is executed with no wait and the list table mappings command is immediately executed, use the reload option to update the table statuses.
Use the list table mappings command to view the list of tables that are currently marked for mirroring.

Syntax diagram

.arrow, .syntaxarrow { fill: none; stroke: black; } .arrowheadStartEnd, .arrowheadRepSep, .arrowheadRepSepReturn { stroke: black; fill: black; } .arrowheadSeq, .arrowheadStartChoice, .arrowheadAfterChoice, .arrowheadStartRepGroup, .arrowheadEndRepGroup, .arrowheadRev { stroke: none; fill: none; } rect { fill: none; stroke: none; } rect.fragref,rect.syntaxfragref { fill: none; stroke: black; } text { fill: #000000; fill-opacity: 1; font-family: IBM Plex Sans,Arial Unicode MS,Arial,Helvetica; font-style: normal; font-weight: normal; font-size: 8pt; stroke: #000000; stroke-width: 0.1; } text.var, text.syntaxvar {font-style:italic;} start mirroringnamesubscription methodcontinuousscheduledEnd endTypenowlogPositionlocalTimeendValuevaluewait0seconds

Syntax

```
start mirroring [name <subscription>] [method <mirror>] [endType <type>]

[endValue <value>] [wait <seconds>];
```

Parameters

- [name <subscription>]
- Specifies the name of the subscription.If a name is not provided, the subscription that is currently identified as the context will be used. To view the current context, use the show context command.

- [method <mirror>]
- Specifies the method of mirroring to be used.Valid values:
  - **continuous**—Replicates changes to the target on a continuous basis.
  - **scheduledEnd**—Replicates changes to the target until to a user-specified point in the source database log and then ends replication.

  Default value: continuous

- [endType <type>]
- Identifies the scheduled end type when the method parameter is set to scheduledEnd.Valid values:
  - **now**—Specifies that replication will stop when the current time is reached in the source database log.
  - **logPosition**—Specifies that replication will stop when replication reaches a user-specified log position in the source database log.
  - **localTime**—Specifies that replication will stop at a user-specified data and time in the source database log.

  Default value: now

- [endValue <value>]

    - When the endType parameter is set to logPosition, this value specifies a log position for the end value. When the endType parameter is set to localTime, this value specifies the exact date and time in the source engine's time zone using yyyyMMddHHmmssSSS.

- **[wait <seconds>]**
    - Specifies how long to wait (in seconds) for mirroring to start. Use -1 to wait indefinitely, 0 for no wait, or the maximum number of seconds to wait before continuing with script execution.Default value: 0

Required role

- Operator
- Administrator
- System administrator

Result

This command returns a value of 0 if the command was successful and a negative value if the command fails. When verbose mode is enabled, the command prints success message CDC1312.

Sample

Selects a subscription and starts mirroring.

```
connect server username user1 password password1;
connect datastore name DS1 context source;
connect datastore name DS2 context target;
select subscription name SUB1;
start mirroring;
disconnect server;
exit;
```

Parent topic: Replication commands

start refresh

Use this command to start refreshing a subscription.The table mapping context is cleared after executing this command if the context is currently set to this subscription. If the start refresh command is executed with no wait and the list table mappings command is immediately executed, use the reload option to update the table statuses.
Use the list table mappings command to view the list of tables that are currently flagged for refresh.

Syntax diagram

.arrow, .syntaxarrow { fill: none; stroke: black; } .arrowheadStartEnd, .arrowheadRepSep, .arrowheadRepSepReturn { stroke: black; fill: black; } .arrowheadSeq, .arrowheadStartChoice, .arrowheadAfterChoice, .arrowheadStartRepGroup, .arrowheadEndRepGroup, .arrowheadRev { stroke: none; fill: none; } rect { fill: none; stroke: none; } rect.fragref,rect.syntaxfragref { fill: none; stroke: black; } text { fill: #000000; fill-opacity: 1; font-family: IBM Plex Sans,Arial Unicode MS,Arial,Helvetica; font-style: normal; font-weight: normal; font-size: 8pt; stroke: #000000; stroke-width: 0.1; } text.var, text.syntaxvar {font-style:italic;} start refreshnamesubscription wait0seconds

Syntax

```
start refresh [name <subscription>] [wait <seconds>];
```

Parameters

- [name <subscription>]

- Specifies the name of the subscription.If a name is not provided, the subscription that is currently identified as the context will be used. To view the current context, use the show context command.

- [wait <seconds>]

- Specifies how long to wait (in seconds) for refresh to start. Use -1 to wait indefinitely, 0 for no wait, or the maximum number of seconds to wait before continuing with script execution.Default value: 0

Required role

- Operator
- Administrator
- System administrator

Result

This command returns a value of 0 if the command was successful and a negative value if the command fails. When verbose mode is enabled, the command prints success message CDC1313.

Sample

Selects a subscription and starts refresh.

```
connect server username user1 password password1;

connect datastore name DS1 context source;

connect datastore name DS2 context target;

select subscription name SUB1;

start refresh;

disconnect server;

exit;
```

Parent topic: Replication commands

Access Manager commands

See also:

- **add access connection**
- **add access datastore**
- **add access user**
- **change login password**
- **close access user connections**
- **delete access connection**
- **delete access datastore**
- **delete access user**
- **list access datastores**
- **list access users**
- **modify access connection**
- **modify access datastore**
- **modify access user**
- **show access connection**
- **show access datastore**
- **show access user**

Parent topic:CHCCLP Commands

add access connection

Use this command to add a new connection between a datastore and user in Access Server.

Syntax diagram

.arrow, .syntaxarrow { fill: none; stroke: black; } .arrowheadStartEnd, .arrowheadRepSep, .arrowheadRepSepReturn { stroke: black; fill: black; } .arrowheadSeq, .arrowheadStartChoice, .arrowheadAfterChoice, .arrowheadStartRepGroup, .arrowheadEndRepGroup, .arrowheadRev { stroke: none; fill: none; } rect { fill: none; stroke: none; } rect.fragref,rect.syntaxfragref { fill: none; stroke: black; } text { fill: #000000; fill-opacity: 1; font-family: IBM Plex Sans,Arial Unicode MS,Arial,Helvetica; font-style: normal; font-weight: normal; font-size: 8pt; stroke: #000000; stroke-width: 0.1; } text.var, text.syntaxvar {font-style:italic;} add access connectionusernameuserdatastorenamedbUserdatabase dbPassword passwordalwaysPromptfalse/notrue/yesshowParameterValuesfalse/no true/yeswriteProtectParametersfalse/notrue/yessaveParameterstrue/yesfalse/no

Syntax

```
add access connection username user datastore name dbUser database dbPassword
password [alwaysPrompt prompt] [showParameterValues show] [writeProtectParameters
write] [saveParameters save];
```

Parameters

- **username user**
  - Specifies the name of the user.
- **datastore name**
  - Specifies the name of the datastore.
- **dbUser database**
  - Specifies the database user name.
- **dbPassword password**
  - Specifies the database password
- **[alwaysPrompt prompt]**
  - Specifies whether the user should be prompted for login credentials.Valid values:
    - **true/yes**—Prompts user for login credentials. Either yes or true are valid values.
    - **false/no**—Does not prompt user for login credentials. Either no or false are valid values.

    Default value: false/no
- **[showParameterValues show]**
  - Specifies whether parameter values are displayed at login.Valid values:
    - **true/yes**—Displays parameter values at login. Either yes or true are valid values.
    - **false/no**—Does not display parameter values at login. Either no or false are valid values.

    Default value: false/no
- **[writeProtectParameters write]**
  - Specifies that parameters are write-protected.Valid values:

- **true/yes**—Write-protects parameters. Either yes or true are valid values.
- **false/no**—Does not write-protect parameters. Either no or false are valid values.

Default value: false/no

- [saveParameters save]

- Specifies that parameters can be saved.Valid values:
  - **true/yes**—Allows parameters to be saved. Either yes or true are valid values.
  - **false/no**—Does not allow parameters to be saved. Either no or false are valid values.

  Default value: false/no

Required role

Access Manager

Result

This command returns a value of 0 if the command was successful and a negative value if the command fails. When verbose mode is enabled, the command prints success message CDC1611.

Sample

Adds a new datastore connection to a user.

```
connect server username user1 password password1;
add access user name USER1;
add access connection username USER1 datastore DS1 dbUser DBUSER dbPassword mypassword;
list access datastores username USER1;
disconnect server;
exit;
```

Parent topic: Access Manager commands

add access datastore

Use this command to add a new datastore to Access Server.

Syntax diagram

.arrow, .syntaxarrow { fill: none; stroke: black; } .arrowheadStartEnd, .arrowheadRepSep, .arrowheadRepSepReturn { stroke: black; fill: black; } .arrowheadSeq, .arrowheadStartChoice, .arrowheadAfterChoice, .arrowheadStartRepGroup, .arrowheadEndRepGroup, .arrowheadRev { stroke: none; fill: none; } rect { fill: none; stroke: none; } rect.fragref,rect.syntaxfragref { fill: none; stroke: black; } text { fill: #000000; fill-opacity: 1; font-family: IBM Plex Sans,Arial Unicode MS,Arial,Helvetica; font-style: normal; font-weight: normal; font-size: 8pt; stroke: #000000; stroke-width: 0.1; } text.var, text.syntaxvar {font-style:italic;} add access datastorenamedatastoredescriptiondescriptionhostname nameportnumbermultiuserfalse/notrue/yes

Syntax

```
add access datastore name <datastore> [description <description>] hostname <name>
port <number> [multiuser <lock>];
```

Parameters

- name <datastore>
- Specifies the name of the datastore.

- [description <description>]
- Specifies a description of the datastore.Default value: empty string ().

- hostname <name>
- Specifies the host name where the datastore is installed.

- port <number>
- Specifies the port number used to connect to datastore.

- [multiuser <lock>]
- Specifies that subscriptions must be locked before editing.Valid values:
  - **true/yes**—Requires that subscriptions must be locked before being edited. Either yes or true are valid values.
  - **false/no**—Does not require that subscriptions be locked before being edited. Either no or false are valid values.

  Default value: false/no

Required role

Access Manager

Result

This command returns a value of 0 if the command was successful and a negative value if the command fails. When verbose mode is enabled, the command prints success message CDC1609.

Sample

Adds a new datastore to Access Server.

```
connect server username user1 password password1;
add access datastore name DS1 hostname server.host.com port 11111;
show access datastore name DS1;
disconnect server;
exit;
```

Parent topic: Access Manager commands

add access user

Use this command to add a new user to Access Server.

Syntax diagram

.arrow, .syntaxarrow { fill: none; stroke: black; } .arrowheadStartEnd, .arrowheadRepSep, .arrowheadRepSepReturn { stroke: black; fill: black; } .arrowheadSeq, .arrowheadStartChoice, .arrowheadAfterChoice, .arrowheadStartRepGroup, .arrowheadEndRepGroup, .arrowheadRev { stroke: none; fill: none; } rect { fill: none; stroke: none; } rect.fragref,rect.syntaxfragref { fill: none; stroke: black; } text { fill: #000000; fill-opacity: 1; font-family: IBM Plex Sans,Arial Unicode MS,Arial,Helvetica; font-style: normal; font-weight: normal; font-size: 8pt; stroke: #000000; stroke-width: 0.1; } text.var, text.syntaxvar {font-style:italic;} add access usernameuserfullnamenamedescription""description password""loginrolesysadminmonitoroperatoradminmanagerfalse/notrue/yes changePasswordfalse/notrue/yespasswordExpiresfalse/notrue/yes

Syntax

```
add access user name <user> [fullname <name>] [description <description>] [password

<login>] [role <level>] [manager <manager>] [changePassword <change>]

[passwordExpires <expire>];
```

Parameters

- name <user>

- Specifies the name of the user.

- [fullname <name>]

- Specifies the full name for the user.Default value: empty string ().

- [description <description>]

- Specifies a description for the user.Default value: empty string ().

- [password <login>]

- Specifies a login password for the user.Default value: empty string ().

- [role <level>]

- Specifies a role for the user.Valid values:
  - **monitor**—Specifies that a user assigned to this role is a Monitor and only has access to the Monitoring perspective in Management Console.
  - **operator**—Specifies that a user assigned to this role is an Operator and only has access to the Monitoring perspective in Management Console.
  - **admin**—Specifies that a user assigned to this role is an Administrator and can perform all available operations in Management Console, but cannot modify system parameters. Users assigned to this role can access both the Monitoring and Configuration perspectives
  - **sysadmin**—Specifies that a user assigned to this role is a System Administrator and can perform all available operations in Management Console. Only users that require full operational access to the Monitoring and Configuration perspectives should be assigned to this role. System Administrators can also modify system parameters to calibrate their replication environment

  Default value: sysadmin

- [manager <manager>]

- Specifies whether the user is a datastore and user account manager. Only system administrators can be managers.Valid values:
  - **true/yes**—Specifies that the user is a datastore and user account manager. Either yes or true are valid values.
  - **false/no**—Specifies that the user is not a datastore and user account manager. Either no or false are valid values.

  Default value: false/no

- [changePassword <change>]

- Specifies whether the user is required to change their password when they next log in.Valid values:
  - **true/yes**—Requires users to change their password at next login. Either yes or true are valid values.
  - **false/no**—Does not require users to change their password at next login. Either no or false are valid values.

  Default value: false/no

- [passwordExpires <expire>]

- Specifies whether the password expires according to Access Server policy. Valid values:
  - **true/yes**—Overrides any existing password expiry policies. Either yes or true are valid values.
  - **false/no**—Specifies that passwords will expire. Either no or false are valid values.

  Default value: false/no

Required role

Access Manager

Result

This command returns a value of 0 if the command was successful and a negative value if the command fails. When verbose mode is enabled, the command prints success message CDC1607.

Sample

Adds a new user to Access Server.

```
connect server username user1 password password1;
add access user name USER1 manager yes changepassword yes;
show access user name USER1;
disconnect server;
exit;
```

Parent topic: Access Manager commands

change login password

Use this command to update the Access Server login password for a user.

Syntax diagram

.arrow, .syntaxarrow { fill: none; stroke: black; } .arrowheadStartEnd, .arrowheadRepSep, .arrowheadRepSepReturn { stroke: black; fill: black; } .arrowheadSeq, .arrowheadStartChoice, .arrowheadAfterChoice, .arrowheadStartRepGroup, .arrowheadEndRepGroup, .arrowheadRev { stroke: none; fill: none; } rect { fill: none; stroke: none; } rect.fragref,rect.syntaxfragref { fill: none; stroke: black; } text { fill: #000000; fill-opacity: 1; font-family: IBM Plex Sans,Arial Unicode MS,Arial,Helvetica; font-style: normal; font-weight: normal; font-size: 8pt; stroke: #000000; stroke-width: 0.1; } text.var, text.syntaxvar {font-style:italic;} change login passwordnameuseroldPasswordoldnewPasswordnew

Syntax

```
change login password [name user] oldPassword old newPassword new;
```

Parameters

- [name user]

- Specifies the name of the user.
- If the name parameter is not provided, the password is updated for the user currently logged in.

- oldPassword old

- Specifies the old password.

- newPassword new

- Specifies the new password.

Required role

Any

Result

This command returns a value of 0 if the command was successful and a negative value if the command fails. When verbose mode is enabled, the command prints success message CDC1613.

Sample

Changes the login password for a user.

```
connect server username user1 password password1;
change login password oldPassword password11 newPassword password22;
disconnect server;
exit;
```

Parent topic: Access Manager commands

close access user connections

Use this command to terminate open Access Server connections for a user.

Syntax diagram

.arrow, .syntaxarrow { fill: none; stroke: black; } .arrowheadStartEnd, .arrowheadRepSep, .arrowheadRepSepReturn { stroke: black; fill: black; } .arrowheadSeq, .arrowheadStartChoice, .arrowheadAfterChoice, .arrowheadStartRepGroup, .arrowheadEndRepGroup, .arrowheadRev { stroke: none; fill: none; } rect { fill: none; stroke: none; } rect.fragref,rect.syntaxfragref { fill: none; stroke: black; } text { fill: #000000; fill-opacity: 1; font-family: IBM Plex Sans,Arial Unicode MS,Arial,Helvetica; font-style: normal; font-weight: normal; font-size: 8pt; stroke: #000000; stroke-width: 0.1; } text.var, text.syntaxvar {font-style:italic;} close access user connectionsnameuser

Syntax

```
close access user connections name <user>;
```

Parameters

- name <user>

- Specifies the name of the user.

Required role

Access Manager **Result**

This command returns a value of 0 if the command was successful and a negative value if the command fails. When verbose mode is enabled, the command prints success message CDC1618. **Sample**

Parent topic: Access Manager commands

delete access connection

Use this command to delete a connection between a datastore and user in Access Server.

Syntax diagram

.arrow, .syntaxarrow { fill: none; stroke: black; } .arrowheadStartEnd, .arrowheadRepSep, .arrowheadRepSepReturn { stroke: black; fill: black; } .arrowheadSeq, .arrowheadStartChoice, .arrowheadAfterChoice, .arrowheadStartRepGroup, .arrowheadEndRepGroup, .arrowheadRev { stroke: none; fill: none; } rect { fill: none; stroke: none; } rect.fragref,rect.syntaxfragref { fill: none; stroke: black; } text { fill: #000000; fill-opacity: 1; font-family: IBM Plex Sans,Arial Unicode MS,Arial,Helvetica; font-style: normal; font-weight: normal; font-size: 8pt; stroke: #000000; stroke-width: 0.1; } text.var, text.syntaxvar {font-style:italic;} delete access connectionusernameuserdatastorename

Syntax

```
delete access connection username user datastore name;
```

Parameters

- username user

- Specifies the name of the user.

- datastore name

- Specifies the name of the datastore.

Required role

Access Manager

Result

This command returns a value of 0 if the command was successful and a negative value if the command fails. When verbose mode is enabled, the command prints success message CDC1612.

Sample

Removes a datastore connection from a user in Access Server.

```
connect server username user1 password password1;
delete access connection username USER1 datastore DS1;
list access datastores username USER1;
disconnect server;
exit;
```

Parent topic: Access Manager commands

delete access datastore

Use this command to delete a datastore in Access Server.

Syntax diagram

.arrow, .syntaxarrow { fill: none; stroke: black; } .arrowheadStartEnd, .arrowheadRepSep, .arrowheadRepSepReturn { stroke: black; fill: black; } .arrowheadSeq, .arrowheadStartChoice, .arrowheadAfterChoice, .arrowheadStartRepGroup, .arrowheadEndRepGroup, .arrowheadRev { stroke: none; fill: none; } rect { fill: none; stroke: none; } rect.fragref,rect.syntaxfragref { fill: none; stroke: black; } text { fill: #000000; fill-opacity: 1; font-family: IBM Plex Sans,Arial Unicode MS,Arial,Helvetica; font-style: normal; font-weight: normal; font-size: 8pt; stroke: #000000; stroke-width: 0.1; } text.var, text.syntaxvar {font-style:italic;} delete access datastorenamedatastore

Syntax

```
delete access datastore name datastore;
```

Parameters

- name datastore

- Specifies the name of the datastore.

Required role

Access Manager

Result

This command returns a value of 0 if the command was successful and a negative value if the command fails. When verbose mode is enabled, the command prints success message CDC1610.

Sample

Removes a datastore from Access Server.

```
connect server username user1 password password1;
delete access datastore name DS1;
list access datastores;
disconnect server;
exit;
```

Parent topic: Access Manager commands

delete access user

Use this command to delete a user in Access Server.

Syntax diagram

.arrow, .syntaxarrow { fill: none; stroke: black; } .arrowheadStartEnd, .arrowheadRepSep, .arrowheadRepSepReturn { stroke: black; fill: black; } .arrowheadSeq, .arrowheadStartChoice, .arrowheadAfterChoice, .arrowheadStartRepGroup, .arrowheadEndRepGroup, .arrowheadRev { stroke: none; fill: none; } rect { fill: none; stroke: none; } rect.fragref,rect.syntaxfragref { fill: none; stroke: black; } text { fill: #000000; fill-opacity: 1; font-family: IBM Plex Sans,Arial Unicode MS,Arial,Helvetica; font-style: normal; font-weight: normal; font-size: 8pt; stroke: #000000; stroke-width: 0.1; } text.var, text.syntaxvar {font-style:italic;} delete access usernameuser

Syntax

```
delete access user name user;
```

Parameters

- name user

- Specifies the name of the user.

Required role

Access Manager

Result

This command returns a value of 0 if the command was successful and a negative value if the command fails. When verbose mode is enabled, the command prints success message CDC1608.

Sample

Deletes a user from Access Server.

```
connect server username user1 password password1;

delete access user name USER1;

list access users;

disconnect server;

exit;
```

Parent topic: Access Manager commands

list access datastores

Use this command to list the datastores in Access Server.

Syntax diagram

.arrow, .syntaxarrow { fill: none; stroke: black; } .arrowheadStartEnd, .arrowheadRepSep, .arrowheadRepSepReturn { stroke: black; fill: black; } .arrowheadSeq, .arrowheadStartChoice, .arrowheadAfterChoice, .arrowheadStartRepGroup, .arrowheadEndRepGroup, .arrowheadRev { stroke: none; fill: none; } rect { fill: none; stroke: none; } rect.fragref,rect.syntaxfragref { fill: none; stroke: black; } text { fill: #000000; fill-opacity: 1; font-family: IBM Plex Sans,Arial Unicode MS,Arial,Helvetica; font-style: normal; font-weight: normal; font-size: 8pt; stroke: #000000; stroke-width: 0.1; } text.var, text.syntaxvar {font-style:italic;} list access datastoresusernamename

Syntax

```
list access datastores [username name];
```

Parameters

- [username name]

- Specifies a filter that displays datastores associated with the user.
- When the username parameter is omitted, all datastores are listed.

Required role

Access Manager

Result

The ResultStringTable object containing list of datastores.

Sample

Sample 1: Lists the datastores in Access Server.

```
connect server username user1 password password1;
list access datastores;
disconnect server;
exit;
```

Sample 2: Lists the datastores for a specific user in Access Server.

```
connect server username user1 password password1;
list access datastores name USER1;
disconnect server;
exit;
```

Parent topic: Access Manager commands

list access users

Use this command to view the users configured in Access Server.

Syntax diagram

.arrow, .syntaxarrow { fill: none; stroke: black; } .arrowheadStartEnd, .arrowheadRepSep, .arrowheadRepSepReturn { stroke: black; fill: black; } .arrowheadSeq, .arrowheadStartChoice, .arrowheadAfterChoice, .arrowheadStartRepGroup, .arrowheadEndRepGroup, .arrowheadRev { stroke: none; fill: none; } rect { fill: none; stroke: none; } rect.fragref,rect.syntaxfragref { fill: none; stroke: black; } text { fill: #000000; fill-opacity: 1; font-family: IBM Plex Sans,Arial Unicode MS,Arial,Helvetica; font-style: normal; font-weight: normal; font-size: 8pt; stroke: #000000; stroke-width: 0.1; } text.var, text.syntaxvar {font-style:italic;} list access usersdatastorename

Syntax

```
list access users [datastore name];
```

Parameters

- [datastore name]

- Specifies a filter that displays users associated with the datastore. When the datastore parameter is omitted, all users are listed.

Required role

Access Manager **Result**

The ResultStringTable object containing list of access users. **Sample**

Sample 1: Lists the users in Access Server.

```
connect server username user1 password password1;
list access users;
disconnect server;
exit;
```

Sample 2: Lists the users with connection to a specific datastore in Access Server.

```
connect server username user1 password password1;
list access users datastore DS1;
disconnect server;
exit;
```

Parent topic: Access Manager commands

modify access connection

Use this command to modify a connection between a datastore and user in Access Server.

Syntax diagram

.arrow, .syntaxarrow { fill: none; stroke: black; } .arrowheadStartEnd, .arrowheadRepSep, .arrowheadRepSepReturn { stroke: black; fill: black; } .arrowheadSeq, .arrowheadStartChoice, .arrowheadAfterChoice, .arrowheadStartRepGroup, .arrowheadEndRepGroup, .arrowheadRev { stroke: none; fill: none; } rect { fill: none; stroke: none; } rect.fragref,rect.syntaxfragref { fill: none; stroke: black; } text { fill: #000000; fill-opacity: 1; font-family: IBM Plex Sans,Arial Unicode MS,Arial,Helvetica; font-style: normal; font-weight: normal; font-size: 8pt; stroke: #000000; stroke-width: 0.1; } text.var, text.syntaxvar {font-style:italic;} modify access connectionusernameuserdatastorenamedbUserdatabase dbPassword passwordalwaysPromptfalse/notrue/yesshowParameterValuesfalse/no true/yeswriteProtectParametersfalse/notrue/yessaveParameterstrue/yesfalse/no

Syntax

```
modify access connection username user datastore name [dbUser database]
[dbPassword password] [alwaysPrompt prompt] [showParameterValues show]
[writeProtectParameters write] [saveParameters save];
```

Parameters

- **username user**
  - Specifies the name of the user.
- **datastore name**
  - Specifies the name of the datastore.
- **[dbUser database]**
  - Specifies the database user name.
- **[dbPassword password]**
  - Specifies the database password
- **[alwaysPrompt prompt]**
  - Specifies whether the user should be prompted for login credentials.Valid values:
    - **true/yes**—Prompts user for login credentials. Either yes or true are valid values.
    - **false/no**—Does not prompt user for login credentials. Either no or false are valid values.

    Default value: false/no
- **[showParameterValues show]**
  - Specifies whether parameter values are displayed at login.Valid values:
    - **true/yes**—Displays parameter values at login. Either yes or true are valid values.
    - **false/no**—Does not display parameter values at login. Either no or false are valid values.

    Default value: false/no
- **[writeProtectParameters write]**
  - Specifies that parameters are write-protected.Valid values:

- **true/yes**—Write-protects parameters. Either yes or true are valid values.
- **false/no**—Does not write-protect parameters. Either no or false are valid values.

Default value: false/no

- [saveParameters save]

- Specifies that parameters can be saved.Valid values:
  - **true/yes**—Allows parameters to be saved. Either yes or true are valid values.
  - **false/no**—Does not allow parameters to be saved. Either no or false are valid values.

Default value: true/yes

Required role

Access Manager

Result

This command returns a value of 0 if the command was successful and a negative value if the command fails. When verbose mode is enabled, the command prints success message CDC1617.

Sample

Modifies the connection settings to a datastore for a user.

```
connect server username user1 password password1;
modify access connection username USER1 datastore DS1 dbUser DBUSER dbPassword password12;
list access datastores username USER1;
disconnect server;
exit;
```

Parent topic: Access Manager commands

modify access datastore

Use this command to modify a datastore inAccess Server

Syntax diagram

.arrow, .syntaxarrow { fill: none; stroke: black; } .arrowheadStartEnd, .arrowheadRepSep, .arrowheadRepSepReturn { stroke: black; fill: black; } .arrowheadSeq, .arrowheadStartChoice, .arrowheadAfterChoice, .arrowheadStartRepGroup, .arrowheadEndRepGroup, .arrowheadRev { stroke: none; fill: none; } rect { fill: none; stroke: none; } rect.fragref,rect.syntaxfragref { fill: none; stroke: black; } text { fill: #000000; fill-opacity: 1; font-family: IBM Plex Sans,Arial Unicode MS,Arial,Helvetica; font-style: normal; font-weight: normal; font-size: 8pt; stroke: #000000; stroke-width: 0.1; } text.var, text.syntaxvar {font-style:italic;} modify access datastorenamedatastoredescriptiondescriptionhostname nameportnumbermultiuserfalse/notrue/yes

Syntax

```
modify access datastore name <datastore> [description <description>] [hostname

<name>] [port <number>] [multiuser <lock>];
```

Parameters

- **name <datastore>**
  - Specifies the name of the datastore.
- **[description <description>]**
  - Specifies a description of the datastore.Default value: empty string ().
- **[hostname <name>]**
  - Specifies the host name where the datastore is installed.
- **[port <number>]**
  - Specifies the port number used to connect to datastore.
- **[multiuser <lock>]**
  - Specifies that subscriptions must be locked before editing.Valid values:
    - **true/yes**—Requires that subscriptions must be locked before being edited. Either yes or true are valid values.
    - **false/no**—Does not require that subscriptions be locked before being edited. Either no or false are valid values.

    Default value: false/no

Required role

Access Manager **Result**

This command returns a value of 0 if the command was successful and a negative value if the command fails. When verbose mode is enabled, the command prints success message CDC1616. **Sample**

Modifies the properties of a datastore in Access Server.

```
connect server username user1
password password1;

modify access datastore name DS1 hostname server2.host.com port 22222;

show access datastore name DS1;

disconnect server;

exit;
```

Parent topic: Access Manager commands

modify access user

Use this command to modify a user to Access Server.

Syntax diagram

.arrow, .syntaxarrow { fill: none; stroke: black; } .arrowheadStartEnd, .arrowheadRepSep, .arrowheadRepSepReturn { stroke: black; fill: black; } .arrowheadSeq, .arrowheadStartChoice, .arrowheadAfterChoice, .arrowheadStartRepGroup, .arrowheadEndRepGroup, .arrowheadRev { stroke: none; fill: none; } rect { fill: none; stroke: none; } rect.fragref,rect.syntaxfragref { fill: none; stroke: black; } text { fill: #000000; fill-opacity: 1; font-family: IBM Plex Sans,Arial Unicode MS,Arial,Helvetica; font-style: normal; font-weight: normal; font-size: 8pt; stroke: #000000; stroke-width: 0.1; } text.var, text.syntaxvar {font-style:italic;} modify access usernameuserfullnamenamedescription""description password""loginrolesysadminmonitoroperatoradminmanagerfalse/notrue/yes changePasswordfalse/notrue/yespasswordExpiresfalse/notrue/yes

Syntax

```
modify access user <user> [fullname <name>] [description <description>] [password
<login>] [role <level>] [manager <manager>] [changePassword <change>]
[passwordExpires <expire>];
```

Parameters

- name <user>

- Specifies the name of the user.

- [fullname <name>]

- Specifies the full name for the user.Default value: empty string ().

- [description <description>]

- Specifies a description for the user.Default value: empty string ().

- [password <login>]

- Specifies a login password for the user.Default value: empty string ().

- [role <level>]

- Specifies a role for the user.Valid values:
  - **monitor**—Specifies that a user assigned to this role is a Monitor and only has access to the Monitoring perspective in Management Console.
  - **operator**—Specifies that a user assigned to this role is an Operator and only has access to the Monitoring perspective in Management Console.
  - **admin**—Specifies that a user assigned to this role is an Administrator and can perform all available operations in Management Console, but cannot modify system parameters. Users assigned to this role can access both the Monitoring and Configuration perspectives
  - **sysadmin**—Specifies that a user assigned to this role is a System Administrator and can perform all available operations in Management Console. Only users that require full operational access to the Monitoring and Configuration perspectives should be assigned to this role. System Administrators can also modify system parameters to calibrate their replication environment

  Default value: sysadmin

- [manager <manager>]

- Specifies whether the user is a datastore and user account manager. Only system administrators can be managers.Valid values:
  - **true/yes**—Specifies that the user is a datastore and user account manager. Either yes or true are valid values.
  - **false/no**—Specifies that the user is not a datastore and user account manager. Either no or false are valid values.

  Default value: false/no

- [changePassword <change>]

- Specifies whether the user is required to change their password when they next log in.Valid values:
  - **true/yes**—Requires users to change their password at next login. Either yes or true are valid values.
  - **false/no**—Does not require users to change their password at next login. Either no or false are valid values.

  Default value: false/no

- [passwordExpires <expire>]

- Specifies whether the password expires according to Access Server policy. Valid values:
  - **true/yes**—Overrides any existing password expiry policies. Either yes or true are valid values.
  - **false/no**—Specifies that passwords will expire. Either no or false are valid values.

  Default value: false/no

Required role

Access Manager

Result

This command returns a value of 0 if the command was successful and a negative value if the command fails. When verbose mode is enabled, the command prints success message CDC1615.

Sample

Sample 1: Modifies the properties of a user in Access Server.

```
connect server username user1 password password1;
modify access user name USER1 changepassword yes;
show access user name USER1;
disconnect server;
exit;
```

Sample 2: Resets a disabled and locked account.

```
connect server username user1 password password1;
modify access user name USER1 disabled no locked no;
show access user name USER1;
disconnect server;
exit;
```

Parent topic: Access Manager commands

show access connection

Use this command to view the properties of a connection between a user and datastore in Access Server.

Syntax diagram

.arrow, .syntaxarrow { fill: none; stroke: black; } .arrowheadStartEnd, .arrowheadRepSep, .arrowheadRepSepReturn { stroke: black; fill: black; } .arrowheadSeq, .arrowheadStartChoice, .arrowheadAfterChoice, .arrowheadStartRepGroup, .arrowheadEndRepGroup, .arrowheadRev { stroke: none; fill: none; } rect { fill: none; stroke: none; } rect.fragref,rect.syntaxfragref { fill: none; stroke: black; } text { fill: #000000; fill-opacity: 1; font-family: IBM Plex Sans,Arial Unicode MS,Arial,Helvetica; font-style: normal; font-weight: normal; font-size: 8pt; stroke: #000000; stroke-width: 0.1; } text.var, text.syntaxvar {font-style:italic;} show access connectionusernameuserdatastorename

Syntax

```
show access connection username user datastore name;
```

Parameters

- username user

- Specifies the name of the user.

- datastore name

- Specifies the name of the datastore.

Required role

Access Manager

Result

The ResultStringKeyValues object containing properties of the access connection.

```
PROPERTY                            VALUE
----------------------------------- --------------------------


CONNECTION PARAMETERS
 Datastore:                         DatastoreName
 User:                              username
 Database Login:                    dbLogin
 Database Password:                 ********


CONNECTION/RETRY OPTIONS
 Always Show Connection Dialog:     No
 Show Parameter Values:             No
 Write-Protect Parameters:          No
 Allow Connection Parameter Saving: Yes
```

Sample

Shows the connection information for a user to a datastore in Access Server.

```
connect server username user1 password password1;
show access connection username USER1 datastore DS1;
disconnect server;
exit;
```

Parent topic: Access Manager commands

show access datastore

Use this command to view the properties of a datastore in Access Server.

Syntax diagram

.arrow, .syntaxarrow { fill: none; stroke: black; } .arrowheadStartEnd, .arrowheadRepSep, .arrowheadRepSepReturn { stroke: black; fill: black; } .arrowheadSeq, .arrowheadStartChoice, .arrowheadAfterChoice, .arrowheadStartRepGroup, .arrowheadEndRepGroup, .arrowheadRev { stroke: none; fill: none; } rect { fill: none; stroke: none; } rect.fragref,rect.syntaxfragref { fill: none; stroke: black; } text { fill: #000000; fill-opacity: 1; font-family: IBM Plex Sans,Arial Unicode MS,Arial,Helvetica; font-style: normal; font-weight: normal; font-size: 8pt; stroke: #000000; stroke-width: 0.1; } text.var, text.syntaxvar {font-style:italic;} show access datastorenamedatastore

Syntax

```
show access datastore name datastore;
```

Parameters

- name datastore

- Specifies the name of the datastore.

Required role

Access Manager

Result

The ResultStringKeyValues object containing properties of the access datastore.

```
PROPERTY                   VALUE
-------------------------- --------------------------
Name:                      DataStoreName
Description:               Created - May 9, 11:14am
Host Name:                 hostName
Port:                      10901
Version:                   V10R2M0T0BCDC_WTPGMVMN_25
Platform:                  Java VM
Database:                  JDBC
Type:                      Dual
Multiuser:                 No
```

Sample

Shows the properties of a datastore in Access Server.

```
connect server username user1 password password1;
show access datastore name DS1;
disconnect server;
exit;
```

Parent topic: Access Manager commands

show access user

Use this command to view the properties of a user in Access Server.

Syntax diagram

.arrow, .syntaxarrow { fill: none; stroke: black; } .arrowheadStartEnd, .arrowheadRepSep, .arrowheadRepSepReturn { stroke: black; fill: black; } .arrowheadSeq, .arrowheadStartChoice, .arrowheadAfterChoice, .arrowheadStartRepGroup, .arrowheadEndRepGroup, .arrowheadRev { stroke: none; fill: none; } rect { fill: none; stroke: none; } rect.fragref,rect.syntaxfragref { fill: none; stroke: black; } text { fill: #000000; fill-opacity: 1; font-family: IBM Plex Sans,Arial Unicode MS,Arial,Helvetica; font-style: normal; font-weight: normal; font-size: 8pt; stroke: #000000; stroke-width: 0.1; } text.var, text.syntaxvar {font-style:italic;} show access usernameuser

Syntax

```
show access user name user;
```

Parameters

- name user

- Specifies the name of the user.

Required role

Access Manager

Result

The ResultStringKeyValues object containing properties of the access user.

```
PROPERTY                  VALUE
------------------------- -------------------------

INFORMATION
 Name:                    userName
 Full Name:
 Description:

ACCESS
 Role:                    System Administrator
 Manager:                 Yes

STATUS
 Account Disabled:        No
 Account Locked:          No
 Password Change:         No
 Password Expires:        Yes
```

Sample

Shows the properties of a user in Access Server.

```
connect server username user1 password password1;
show access user name USER1;
disconnect server;
exit;
```

Parent topic: Access Manager commands

Support commands

See also:

- **run support assistant**

Parent topic:CHCCLP Commands

run support assistant

Use this command to run the InfoSphere® CDC Support Assistant to gather scripting environment and Access Server traces and logs.

Syntax diagram

.arrow, .syntaxarrow { fill: none; stroke: black; } .arrowheadStartEnd, .arrowheadRepSep, .arrowheadRepSepReturn { stroke: black; fill: black; } .arrowheadSeq, .arrowheadStartChoice, .arrowheadAfterChoice, .arrowheadStartRepGroup, .arrowheadEndRepGroup, .arrowheadRev { stroke: none; fill: none; } rect { fill: none; stroke: none; } rect.fragref,rect.syntaxfragref { fill: none; stroke: black; } text { fill: #000000; fill-opacity: 1; font-family: IBM Plex Sans,Arial Unicode MS,Arial,Helvetica; font-style: normal; font-weight: normal; font-size: 8pt; stroke: #000000; stroke-width: 0.1; } text.var, text.syntaxvar {font-style:italic;} run support assistantfilenamename

Syntax

```
run support assistant filename <name>;
```

Parameters

- filename <name>

- Specifies the fully qualified output zip filename for the Support Assistant collection.Use double quotation marks to surround the filename. .zip will be appended to the filename, if it does not end with the zip extension.

Required role

Any

Result

This command returns a value of 0 if the command was successful and a negative value if the command fails. When verbose mode is enabled, the command prints success message CDC1005.

Sample

Performs operations while Access Server tracing is enabled and runs support assistant to collect the traces.

```
connect server username user1 password password1;

modify server tracing yes;

connect datastore name DS1 context source;

connect datastore name DS2 context target;

select subscription name SUB1;

...

modify server tracing no;

run support assistant filename c:\\output\\sa.zip;

disconnect server;

exit;
```

Parent topic: Support commands

Management Console Commands reference

Note: The CHCCLP feature is intended to replace the existing command line functionality, which is deprecated and will be removed in a future release. New scripting activities should use CHCCLP rather than the Management Console Commands when possible.

This section describes the Management Console interactive command prompt environment that you can use on a client workstation to configure and initiate transformational replication between source and target datastores. Commands are submitted through a command prompt in Windows or UNIX, or a Telnet environment.

Although Management Console provides the full range of support, commands can be used when your replication configuration is relatively simple to manage. Commands can be issued interactively or included in script files. This guide also describes how script files are created and compiled into a batch file. You can then run the batch file to perform a sequence of commands. Depending on individual preferences, the interactive command prompt environment may be chosen instead of the application.

In this section, you will learn:

- **Using commands in Management Console**
- **Management Console commands**
- **Scripting support**
- **User roles and permissions**

Using commands in Management Console

As an alternative to defining a replication configuration through the Management Console application, Management Console commands can be used to perform administrative and operational functions. Although InfoSphere® CDC also supports some of these commands, command syntax is not consistent across all InfoSphere CDC product offerings. In addition, InfoSphere CDC commands must be issued locally on source or target systems that may be geographically dispersed. By comparison, Management Console commands can be applied to a specified datastore. All commands can be interactively submitted on a workstation at the command prompt in Windows and UNIX, or in a Telnet environment.
In this section, you will learn:

- **Opening command environments**

Parent topic: Management Console Commands reference

Related concepts

- Opening command environments

Opening command environments

To interactively submit Management Console commands, you must open an interactive environment where these commands are recognized. This section explains how to open a Windows, UNIX, or Telnet environment on a workstation where Management Console (Windows) or Access Server (UNIX) has been installed.

See also:

- **To issue commands in a Windows environment**
- **To issue commands in a UNIX or Linux environment**
- **To issue commands in a Telnet environment**

Parent topic: Using commands in Management Console

To issue commands in a Windows environment
Procedure

1. Open a command prompt window.
2. Navigate to the root of the Management Console installation directory.
3. Issue the following command:`online.bat`
   You can now invoke Management Console commands at the prompt.

Parent topic: Opening command environments

Related reference
- exit - Close Command Environment

To issue commands in a UNIX or Linux environment

Procedure

1. Log on to the UNIX or Linux system where Access Server is installed.
2. Navigate to the following directory:`/<Access_Server_Installation_Directory>/bin`
3. Issue the following command:`./online`
   You can now invoke Management Console commands at the prompt.

Parent topic: Opening command environments

Related reference

- exit - Close Command Environment

To issue commands in a Telnet environment

Procedure

1. Perform the following instructions for the operating system where you want to issue the commands:**Windows**:
   Open a command prompt and navigate to the Management Console installation directory and then issue the following command:
   ```
   online.bat  -t  <port>
   ```
   where <port> is an available port number on the local workstation that will be used by the Telnet command environment.
   UNIX and Linux:
   Navigate to the following directory:
   ```
   <Access_Server_Installation_Directory>\bin
   ```
   Issue the following command:
   ```
   ./online  -t  <port>
   ```
   where <port> is an available port number on the local workstation that will be used by the Telnet command environment.

2. Open Telnet and specify the host name and port of the local workstation that you reserved in the previous step:`open gsmith.mycompany.com 1012`

Results

You can now invoke Management Console commands at the prompt.
Parent topic: Opening command environments

Related reference

- exit - Close Command Environment

Management Console commands

You should note the following conventions in the definition of the command parameters:

- Angle brackets < > indicate mandatory parameters.
- Square brackets [ ] show optional arguments. You can select any or none of the parameters, depending on requirements.
- A vertical bar | separates items in a list of parameters or options when the choices are limited only to the parameters and options listed. When the vertical bar appears in a list enclosed by SQUARE BRACKETS, the choices are limited to the items in the list, but the user still has the option not to use any parameters or options.
- Ellipsis ... mean that a parameter or option can be repeated more than once.

Further considerations:

- Command names are case-sensitive, and must be entered as documented in this appendix. Command parameters (for instance, datastore names, table names, and so on) may be case-sensitive on certain platforms. In the examples, command parameters are expressed in uppercase characters.
- All command invocations must end with a semicolon before pressing the Enter key.
- For commands with no parameters, you must specify an empty parameter list after the command name (in other words, ()).
- For optional command parameters ([ ]), default values are used if these parameters are omitted in an invocation.
- Special values must be specified in lowercase characters.
- In each interactive command prompt environment, the connectServer command must be issued before invoking any of the other Management Console commands (except help - Display Command Help) documented in the following sections. Conversely, no Management Console commands can be issued in the command environment after invoking disconnectServer - Disconnect from Access Server.
- Command syntax is verified, and syntax errors that are detected are reported after command invocation. Specific names (such as datastore names, subscription names, and so on) identified in a command invocation are not verified.

In this section, you will learn:

- **addTable - Add Table to Catalog**
- **assignTable - Assign Source Table**
- **connectAgent - Connect to Datastore**
- **connectServer - Connect to Access Server**
- **deAssignTable - Unassign Target Table**
- **describe - Describe Selected Tables**
- **disconnectAgent - Disconnect from Datastore**
- **disconnectServer - Disconnect from Access Server**
- **errorHandle - Define Error Handler**
- **execute - Run Batch File**
- **exit - Close Command Environment**
- **getAgents - Get Accessible Datastores**

- **getSubscriptionActivity - Get Subscription Activity**
- **getSubscriptions - Get Subscriptions for Datastore**
- **getSubscriptionStatus - Get Subscription Status**
- **getSubscriptionTables - Get Subscription Tables**
- **getTableStatus - Get Source Table Status**
- **help - Display Command Help**
- **reAddTable - Re-add Table in Catalog**
- **removeTable - Remove Table from Catalog**
- **setReplication - Set Replication Method**
- **setTableStatus - Set Source Table Status**
- **startRefresh - Start Refresh**
- **startMirror - Start Mirroring**
- **stopAll - Stop All Replication Activities**
- **stopMirror - Stop Mirroring**
- **stopRefresh - Stop Refresh**
- **subscribeTable - Select Table to Subscription**
- **unSubscribeTable - Deselect Table from Subscription**

Parent topic: Management Console Commands reference

Related reference

- help - Display Command Help
- disconnectServer - Disconnect from Access Server

addTable - Add Table to Catalog

This command adds a table to the source tables available for replication for the specified datastore.

Adding a source table to the tables available for replication is discussed in more detail in the Management Console documentation.

Syntax

```
addTable(<agent>,<path>,<table>);
```

Parameters

- <agent>

- The name of the datastore that will have a source table added to its catalog.
- The datastore that you specify must be either a dual or source datastore. For more information about datastores, see your Management Console documentation.

- <path>

- Depending on the type of datastore, the owner of the source table or the library where the source table is located.

- <table>

- The name of the source table that will be added to the catalog.

Result

None.

Example

`addTable(ORAPUBS, GSMITH, EMP);` The source table named EMP owned by GSMITH will be added to the catalog for the datastore ORAPUBS.

`addTable(PRODPUBS, MASTDB.BMOORE, SALES);` The source table named SALES (in the database MASTDB) owned by BMOORE will be added to the catalog for the datastore PRODPUBS.

`addTable(HQAGENT, CUSTLIB, CUSTOMER);` The source table named CUSTOMER located in library CUSTLIB will be added to the catalog for the datastore HQAGENT.

Parent topic: Management Console commands

assignTable - Assign Source Table

This command assigns a source table that has been described (by the describe command) to the specified target table.
For general information about table assignments and mapping tables, see the Management Console documentation.

Syntax

```
assignTable(<agent>,<publisherid>, <pubpath>, <pubtable>, <subdb>, <subpath>,
<subtable>, <submember>, <indexlib>, [, index | , noindex]);
```

Parameters

- <agent>
- The name of the datastore that will have a source table assigned to a described target table.
- The datastore that you specify must be either a dual or target datastore. For more information about datastores, see the Management Console documentation.

- <publisherid>
- The identifier for the source server.
- This identifier is specified when creating a new subscription through Management Console.

- <pubpath>
- Depending on the type of datastore, the owner of the source table or the library where the source table is located.

- <pubtable>
- The name of the source table that you want to assign to the target table.

- <subdb>
- The name of the database that contains the target table.
- If a database name is not required, set this parameter to the special value of *null.

- <subpath>
- Depending on the type of datastore, the owner of the target table or the library where the target table is located.

- <subtable>
- The name of the target table that you want to assign to the target table.

- <submember>
- Indicates how members in source and target tables are assigned.
- If you specify the name of a member in the target table, all members in the source table will be merged into a single member in the multi-member target table.
- Alternatively, this parameter can be set to one of the following special values:
  - `member`—Corresponding members in the source and target tables will be assigned. This value should be specified if the multi-member target table has the same member structure as the multi-member source table.
  - `only`—Merges all members in the source table to the single-member target table.
  - `*null`—Indicates that this parameter is not applicable.

- <indexlib>

- The library containing the unique index for the target table.
- If this parameter is not applicable, set it to the special value of `*null`.

- index

- The name of the unique index for the target table.
- If this parameter is not applicable, set it to the special value of `*null`.

- noindex

- Indicates the operation performed by the command when a unique index for the target table is not specified.
- This parameter is applicable only when the index parameter is omitted (the name of a unique index or `*null` is not specified) in a command invocation. In this case, it provides confirmation whether or not to proceed with the table assignment.
- For example, under Live Audit replication, a unique index is not required for the target table. By specifying the appropriate value for this parameter, you can confirm that the assignment must still be performed.
- You can specify one of the following special values:
  - y—Confirms the table assignment with the specified parameters.
  - n—Rejects the table assignment with the specified parameters. Instead, the deAssignTable command is performed using some of the parameters that have been specified.
- If this parameter is omitted, the default setting is n.

Result

None.

Example

```
assignTable(ORASUBS, NTORAPUB,
GSMITH, EMP, *null, GSMITH, DESTEMP, *null, *null, *null);
```

The source table EMP owned by GSMITH will be assigned to the target table DESTEMP also owned by GSMITH.

The assignment is performed without a unique index for the target table DESTEMP.

```
assignTable(ORASUBS,
NTORAPUB, BJONES, CUST, DESTDB, BJONES, DESTCUST, *null,
*null, *null);
```

The source table CUST owned by BJONES will be assigned to the target table DESTCUST (also owned by BJONES) that is located in the database DESTDB.

The assignment is performed without a unique index for the target table DESTCUST.

```
assignTable(BRANCHAGENT, HQSYS, CUSTLIB, CUSTOMER,
*null, DESTLIB, CUST, MEMBER, DESTLIB, CUSTIX);
```

The multi-member source table CUSTOMER located in the library CUSTLIB will be assigned to the multi-member target table CUST located in the library DESTLIB.

Corresponding members in the source and target tables will be assigned.

The unique index CUSTIX for the target table is located in the library DESTLIB.

```
assignTable(REGION1, PUB, SALESDB, SALES, *null,
DESTDB, DESTSALES, *null, *null, y);
```

The source table SALES located in the library SALESDB will be assigned to the target table DESTSALES located in the library DESTDB.

The assignment is performed without a unique index for the target table DESTSALES.

Parent topic: Management Console commands

Related reference

- deAssignTable - Unassign Target Table

connectAgent - Connect to Datastore

This command establishes a connection to the specified datastore that has been defined in the Access Manager perspective and can be accessed by the Management Console user identified in the connectServer command.
Successfully issuing this command allows you to use other commands where the datastore name has to be identified.
To list the datastores that can be specified in this command, use getAgents command.
Establishing a connection to a datastore through the Management Console is discussed in more detail in the Management Console documentation.

Syntax

```
connectAgent(<agent>);
```

Parameters

- <agent>

- The name of the datastore to which you will connect.

Result

None.

Example

`connectAgent(ORAPUBS);`A connection to the datastore named ORAPUBS will be established.

Parent topic: Management Console commands

Related reference

- connectServer - Connect to Access Server
- getAgents - Get Accessible Datastores

connectServer - Connect to Access Server

This command establishes a connection to Access Server running on the specified server and listening for TCP/IP communications on the specified port number. This command must be issued before other supported Management Console commands (except the help - command) can be invoked.
From a single command prompt environment, only one connection to an Access Server can be maintained at the same time. You must use the disconnectServer command to end the connection to the Access Server. However, if you have more than one command environment active at the same time, simultaneous connections to different Access Servers is possible (one connection per command environment).
Note: If you have connected to an Access Server through Management Console, you must still issue this command and identify the same Access Server. In other words, a connection established to an Access Server through Management Console is not recognized in a command environment.
Establishing a connection to an Access Server through Management Console is discussed in more detail in Management Console documentation.

Syntax

```
connectServer(<user>, <password>, <host>, <port>);
```

Parameters

- <user>
- A Management Console user defined in Access Server.

- <password>
- The password associated with the Management Console user specified through the first parameter (user).
- If no password is associated with the Management Console user, specify two double quotation marks () for this parameter.Note: If this command is included in a script file, the specified password will be encrypted after the script file has been successfully compiled into a batch file that can be executed.

- <host>
- The host name of the server where Access Server is running.
- If Access Server is running on the local system, set this parameter to localhost.

- <port>
- The port number that will be used to establish a connection to Access Server.

Result

None.

Example

`connectServer(GSMITH, APPLEJUICE, HOST1, 10101);`The Management Console user GSMITH with the password APPLEJUICE will establish a connection to Access Server running on the server HOST1, and listening for TCP/IP communications on port number 10101.
`connectServer(EAHIGGINS, , LOCALHOST, 4545);`The Management Console user EAHIGGINS will establish a connection to Access Server running on the local system, and listening for TCP/IP communications on port number 4545. No password is associated with the user EAHIGGINS.

Parent topic: Management Console commands

Related concepts

- Scripting support
- Compiling a script file

Related reference

- connectAgent - Connect to Datastore
- help - Display Command Help
- disconnectServer - Disconnect from Access Server
- execute - Run Batch File
- disconnectAgent - Disconnect from Datastore
- getAgents - Get Accessible Datastores

deAssignTable - Unassign Target Table

This command unassigns the specified source table from the currently assigned target table.

Syntax

```
deAssignTable(<agent>, <publisherid>, <path>, <table>);
```

Parameters

- <agent>

- The name of the datastore that will have a target table unassigned from the specified source table.
- The datastore that you specify must be either a dual or target datastore. For more information about datastores, see Management Console documentation.

- <publisherid>

- The identifier for the source server.
- This identifier is specified when creating a new subscription through Management Console.

- <path>

- Depending on the type of datastore, the owner of the source table or the library where the source table is located.

- <table>

- The name of the source table that you want to unassign from the target table.

Result

None.

Example

`deAssignTable(ORASUBS, NTORAPUB, GSMITH, EMP);`The source table EMP owned by GSMITH will be deassigned from the target table that was assigned to it.

`deAssignTable(PRODSUBS, PRODPUB, MASTDB.BMOORE, SALES);`The source table SALES (in the database MASTDB) owned by BMOORE will be deassigned from the target table that was assigned to it.

`deAssignTable(BRANCHAGENT, HQSYS, CUSTLIB, CUSTOMER);`The multi-member source table CUSTOMER located in the library CUSTLIB will be deassigned from the multi-member target table that was assigned to it.

Parent topic: Management Console commands

describe - Describe Selected Tables

This command sends descriptions of all source tables selected to the specified subscription.

Source table descriptions have to be sent to the target environment after a change has been made to the table definition within Management Console or before replicating table data for the first time. For example, if you change column selections for a source table, the table definition is updated to reflect these changes. Before data from this table can be replicated, a new description of the source table has to be sent to the subscription environment to update the current table definition.

The subscription that you reference in this command must have been defined through Management Console, as there is no command available in the API to create a new subscription.

Syntax

```
describe(<agent>, <subscription>);
```

Parameters

- <agent>

- The name of the datastore that will have descriptions of source tables selected to one of its subscriptions sent by this command.
- The datastore that you specify must be either a dual or source datastore. For more information about datastores, see Managing datastores.

- <subscription>

- The name of the subscription that will have descriptions of its selected source tables sent by this command.

Result

None.

Example

`describe(ORAPUBS, NTORASUB);`Descriptions of source tables selected to the NTORASUB subscription will be sent to the target environment.

`describe(HQAGENT, BRANCHSYS);`Descriptions of source tables selected to the BRANCHSYS subscription will be sent to the target environment.

Parent topic: Management Console commands

disconnectAgent - Disconnect from Datastore

This command ends the connection to the specified datastore that has been defined in Access Manager and can be accessed by the Management Console user identified in connectServer - Connect to Access Server.
To list the datastores that can be specified in this command, use getAgents - Get Accessible Datastores.
Ending a connection to a datastore through Management Console is discussed in more detail in the Management Console documentation.

Syntax

```
disconnectAgent(<agent>);
```

Parameters

- <agent>

- The name of the datastore that will be disconnected.

Result

None.

Example

`disconnectAgent(ORAPUBS);`The connection to the datastore named ORAPUBS will be ended.

Parent topic: Management Console commands

Related reference

- connectServer - Connect to Access Server
- getAgents - Get Accessible Datastores

disconnectServer - Disconnect from Access Server

This command ends the connection to the Access Server established through connectServer - Connect to Access Server.

After successfully issuing this command, you cannot invoke other Management Console commands (except the help - Display Command Help) in the same command environment.

Since a connection to only one Access Server can be maintained in each command environment, this command does not require any parameters.

Note: Existing connections to datastores established through the connectAgent command are automatically ended when disconnecting from Access Server. This is equivalent to issuing disconnectAgent - Disconnect from Datastore for each connected datastore.

Syntax

```
disconnectServer();
```

Parameters

None.

Result

None.

Example

`disconnectServer();`The current connection to the Access Server will be ended.

Existing connections to datastores established in the command environment will also be ended.

Parent topic: Management Console commands

Related reference

- connectServer - Connect to Access Server
- help
- disconnectAgent - Disconnect from Datastore

errorHandle - Define Error Handler

This command determines whether batch file processing should stop or continue upon encountering an error.

This command is primarily intended for use in a script file that includes one or more execute commands. An invocation applies to all subsequent execute commands prior to the next use of this command.

Syntax

```
errorHandle([mode]);
```

Parameters

- mode

- Indicates whether batch file processing should stop or continue when an error is encountered.
- This is an optional parameter. You can specify one of the following special values:
  - `continue`—batch file processing initiated by execute commands will continue when an error is encountered.
  - `stop`—batch file processing initiated by execute commands will stop when an error is encountered.
- If this parameter is omitted, the default setting is `stop`.

Result

None.

Example

`errorHandle();`Batch file processing will stop when an error is encountered.
`errorHandle(continue);`Batch file processing will continue when an error is encountered.

Parent topic: Management Console commands

execute - Run Batch File

This command runs the specified batch file that was produced as a result of compiling a script file.
If you include this command in a script file, it should not reference the batch file in which it is contained, as this would lead to recursive executions without termination.

Syntax

```
execute(<file>);
```

Parameters

- <file>

- The name of the existing batch file that you want to run.
- If you specify the name of the batch file without providing the full path, the batch file must be located in the same folder where the `online.bat` file resides.

Result

None.

Example

`execute(EA.BAT);`The batch file `ea.bat` located in the same folder where the `online.bat` file resides, and a sequence of Management Console commands will be performed.
`execute(C:\EA\BATCH\SUBSCRIBE.TXT);` The batch file `subscribe.txt` will run, and a sequence of Management Console commands will be performed.

Parent topic: Management Console commands

Related concepts

- Scripting support

exit - Close Command Environment

This command closes the command environment where this command was issued. This command ensures that connections established to Access Server and datastores are completely disconnected before closing the command environment. If this command is encountered in a script file, batch processing stops and all subsequent commands in the script file are not performed.

Syntax

```
exit();
```

Parameters

None.

Result

None.

Example

`exit();`The command environment where this command was issued will be closed.

Parent topic: Management Console commands

Related concepts

- Opening command environments

getAgents - Get Accessible Datastores

This command lists the datastores that have been defined in Access Manager and can be accessed by the Management Console user identified in connectServer - Connect to Access Server.

Use this command to identify the datastores that you can reference in other commands that require a datastore name to be specified.

Syntax

```
getAgents();
```

Parameters

None.

Result

This command returns the names of datastores that can be accessed and referenced in other commands.

Example

`getAgents();`The datastores that can be accessed and referenced in other commands will be listed.

Parent topic: Management Console commands

Related reference

- connectServer - Connect to Access Server

getSubscriptionActivity - Get Subscription Activity

This command returns replication activity information for one or more subscriptions. The subscription that you may reference in this command must have been defined through Management Console, as there is no command to create a new subscription.
Displaying replication activity information through Management Console is discussed in more detail in Management Console documentation.

Syntax

```
getSubscriptionActivity(<agent> [, subscription]);
```

Parameters

- <agent>
- The name of the datastore that will have replication activity information for one or more of its subscriptions returned by this command.
- The datastore that you specify must be either a dual or source datastore. For more information about datastores, see Management Console documentation.

- subscription
- The name of the subscription that will have replication activity information returned by this command.
- This is an optional parameter. If it is omitted, information will be returned about all subscriptions defined for the specified datastore.

Result

This command returns a state that identifies the current replication activity for each subscription.

Example

`getSubscriptionActivity(ORAPUBS);`Replication activity information about all subscriptions defined for the datastore ORAPUBS will be retrieved and displayed.
`getSubscriptionActivity(HQAGENT, BRANCHSYS);`Replication activity information about the subscription BRANCHSYS defined for the datastore HQAGENT will be retrieved and displayed.

Parent topic: Management Console commands

getSubscriptions - Get Subscriptions for Datastore

This command lists the subscriptions that have been defined for the specified datastore.

Use this command to identify the subscriptions that you can reference in other commands that require a subscription name to be specified.

Syntax

```
getSubscriptions(<agent>);
```

Parameters

- <agent>

- The name of the datastore for which you want to list existing subscriptions.

Result

This command returns the names of subscriptions defined for the specified datastore.

Example

`getSubscriptions(ORAPUBS);`The subscriptions defined for the datastore ORAPUBS will be listed.

Parent topic: Management Console commands

getSubscriptionStatus - Get Subscription Status

This command returns replication status information for one or more subscriptions. The subscription that you may reference in this command must have been defined through Management Console, as there is no Management Console command to create a new subscription.
Displaying replication status information through Management Console is discussed in more detail in the Management Console documentation.

Syntax

```
getSubscriptionStatus(<agent> [, subscription]);
```

Parameters

- <agent>

- The name of the datastore that will have replication status information for one or more of its subscriptions returned by this command.
- The datastore that you specify must be either a dual or source datastore. For more information about datastores, see the Management Console documentation.

- subscription

- The name of the subscription that will have its replication status returned by this command.
- This is an optional parameter. If it is omitted, information will be returned about all subscriptions defined for the specified datastore.

Result

This command returns a state that identifies the current replication status for each subscription.

Example

`getSubscriptionStatus(ORAPUBS);`Replication status information about all subscriptions defined for the datastore `ORAPUBS` will be retrieved and displayed.
`getSubscriptionStatus(HQAGENT, BRANCHSYS);`Replication status information about the subscription `BRANCHSYS` defined for the datastore `HQAGENT` will be retrieved and displayed.

Parent topic: Management Console commands

getSubscriptionTables - Get Subscription Tables

This command lists the source tables that are currently selected to the specified subscription.

Use this command to identify the source tables that you can reference in other commands that require a source table name to be specified.

Syntax

```
getSubscriptionTables(<agent>, <subscription> [, path]);
```

Parameters

- <agent>

- The name of the datastore for which you want to list selected source tables.
- The datastore that you specify must be either a dual or source datastore.

- <subscription>

- The name of the subscription for which you want to list selected source tables.

- path

- Depending on the type of datastore, the owner of the source tables or the library where the source tables are located.
- This parameter is optional. If it is omitted, all source tables selected to the specified subscription are listed.

Result

This command returns the names of source tables selected to the specified subscription.

Example

`getSubscriptionTables(ORAPUBS, NTORASUB);`All source tables selected to the subscription NTORASUB will be listed.

`getSubscriptionTables(HQAGENT, BRANCHSYS, EAHIGGINS);`Source tables owned by EAHIGGINS that are selected to the subscription BRANCHSYS will be listed.

`getSubscriptionTables(PRODPUBS, PRODSUB, MASTDB.BMOORE);`Source tables (in the database MASTDB) owned by BMOORE that are selected to the subscription PRODSUB will be listed.

`getSubscriptionTables(HQAGENT, BRANCHSYS, CUSTLIB);`Source tables located in library CUSTLIB that are selected to the subscription BRANCHSYS will be listed.

Parent topic: Management Console commands

getTableStatus - Get Source Table Status

This command retrieves the status of a source table selected to the specified subscription.

The status of a source table can be displayed in Management Console, and is discussed in more detail in Management Console documentation.

Syntax

```
getTableStatus(<agent>, <subscription>, <path>, <v>);
```

Parameters

- <agent>

- The name of the datastore that has the source table currently selected to the specified subscription.
- The source that you specify must be either a dual or source datastore.

- <subscription>

- The name of the subscription to which the source table is selected.

- <path>

- Depending on the type of datastore, the owner of the source table or the library where the source table is located.

- <table>

- The name of the source table that will have its status returned by this command.

Result

This command returns either idle, refresh, or active.

Example

`getTableStatus(ORAPUBS, NTORASUB, GSMITH, EMP);`The current status of the source table named EMP owned by GSMITH that is currently selected to the subscription NTORASUB will be returned.

`getTableStatus(PRODPUBS, PRODSUB, MASTDB.BMOORE, SALES);` The current status of the source table named SALES (in the database MASTDB) owned by BMOORE that is currently selected to the subscription PRODSUB will be returned.

`getTableStatus(HQAGENT, BRANCHSYS, CUSTLIB, CUSTOMER);`The current status of the source table named CUSTOMER located in library CUSTLIB that is currently selected to the subscription BRANCHSYS will be returned.

Parent topic: Management Console commands

help - Display Command Help

This command displays help information about the specified command.
This command can be issued before or after establishing a connection to Access Server.

Syntax

```
help([command]);
```

Parameters

- command

- The name of the command for which you want to display help information.
- This parameter is optional. If it is omitted, a list of Management Console commands is displayed without help information.
- Command names are case-sensitive, and must be entered as documented.

Result

This command lists the Management Console commands (if no parameter is specified) or displays help information for the specified command.

Example

`help();`All Management Console commands will be listed.
`help(connectAgent);` Help information for the connectAgent command will be displayed.

Parent topic: Management Console commands

reAddTable - Re-add Table in Catalog

This command changes the metadata definition of an existing table in the catalog. You must issue this command if you have physically changed the table in some way in the database. For example, adding a new column or modifying a column name are changes that affect the physical table. This command ensures that the table definition within Management Console is updated accordingly.
Note: Mirroring of the named table to all subscriptions must be stopped before the table can be successfully re-added.
Re-adding a catalog table through Management Console is discussed in more detail in the Management Console documentation.

Syntax

```
reAddTable(<agent>, <path>, <table>);
```

Parameters

- <agent>
- The name of the datastore that will have a table re-added to its catalog.
- The datastore that you specify must be either a dual or source datastore.

- <path>
- Depending on the type of datastore, the owner of the catalog table or the library where the catalog table is located.

- <table>
- The name of the table that will be re-added in the catalog.

Result

None. **Example**

`reAddTable(ORAPUBS, GSMITH, EMP);`The table named `EMP` owned by `GSMITH` will be re-added in the catalog for the datastore `ORAPUBS`.

`reAddTable(PRODPUBS, MASTDB.BMOORE, SALES);` The table named `SALES` (in the database `MASTDB`) owned by `BMOORE` will be re-added in the catalog for the datastore `PRODPUBS`.

`reAddTable(HQAGENT, CUSTLIB, CUSTOMER);`The table named `CUSTOMER` located in library `CUSTLIB` will be re-added in the catalog for the datastore `HQAGENT`.

Parent topic: Management Console commands

removeTable - Remove Table from Catalog

This command removes a table from the catalog for the specified datastore.

Syntax

```
removeTable(<agent>, <path>, <table> [, endjrn]);
```

Parameters

- <agent>

- The name of the datastore that will have a table removed from its catalog.
- The datastore that you specify must be either a dual or source datastore.

- <path>

- Depending on the type of datastore, the owner of the catalog table or the library where the catalog table is located.

- <table>

- The name of the table that will be removed from the catalog.

- endjrn

- Indicates whether active journaling of the specified table will be stopped.
- Set this parameter to Y to stop active journaling of the table or N to continue active journaling of the table.
- This parameter is optional and only applies to DB2® for i tables.

Result

None.

Example

`removeTable(ORAPUBS, GSMITH, EMP);` The table named `EMP` owned by `GSMITH` will be removed from the catalog for the datastore `ORAPUBS`.

`removeTable(PRODPUBS, MASTDB.BMOORE, SALES);` The table named `SALES` (in the database `MASTDB`) owned by `BMOORE` will be removed from the catalog for the datastore `PRODPUBS`.

`removeTable(HQAGENT, CUSTLIB, CUSTOMER, Y);` The table named `CUSTOMER` located in library `CUSTLIB` will be removed from the catalog for the datastore `HQAGENT`.

Active journaling of the table will be stopped.

Parent topic: Management Console commands

setReplication - Set Replication Method

This command sets the method that you want to use to replicate the specified source table.

For platforms that support table updates by unique key or relative record number (RRN), this command also allows you to set the method of applying replicated updates to the subscription table.

The subscription that you reference in this command must have been defined through Management Console, as there is no command to create a new subscription.

Setting the replication or update methods for a source table through Management Console is discussed in more detail in Management Console documentation.

Syntax

```
setReplication(<agent>, <subscription>, <path>, <table>, <jrnpath>, <jrnname>,

<repmethod> [, updmethod]);
```

Parameters

- <agent>
- The name of the datastore that will have the replication or update methods for a selected source table changed by this command.
- The datastores that you specify must be either a dual or source datastore. For more information about datastores, see the Management Console documentation.

- <subscription>
- The name of the subscription to which the source table is selected.

- <path>
- Depending on the type of datastore, the owner of the source table or the library where the source table is located.

- <table>
- The name of the source table that will have its replication or update methods changed by this command.

- <jrnpath>
- Depending on the type of datastore, the owner of the journal or the library where the journal is located.
- This parameter only applies to datastores that allow you to specify the journal where source table updates are maintained, and when the selected replication method is mirror.
- Set this parameter to "" in order to specify the journal owner or location identified through a recognized InfoSphere® CDC system parameter.

- <jrnname>
- The name of the journal where source table updates are maintained.
- This parameter only applies to datastores that allow you to specify the journal where source table updates are maintained, and when the selected replication method is mirror.
- Set this parameter to "" in order to specify the journal name identified through a recognized InfoSphere CDC system parameter

- <repmethod>
- This parameter must be set to refresh or mirror. For descriptions of these replication methods, see the Management Console documentation.

- The method that you want to use to replicate the specified source table.

- updmethod

- The method that you want to use to apply replicated updates to the subscription table.
- This is an optional parameter that can be specified when working with platforms that support updates either by unique key or relative record number (RRN). When specified, this parameter must be set to `key` or `rrn`. The default setting is `key`. For descriptions of these update methods, see the Management Console documentation.Note: You cannot specify an update method without also specifying a replication method.

Result

None.

Example

`setReplication(ORAPUBS, NTORASUB, GSMITH, EMP, GSMITH, EMPJRN, mirror);`The replication method for the source table named EMP owned by GSMITH that is currently selected to the subscription NTORASUB will be set to mirror.
The journal named EMPJRN owned by GSMITH is used for source table updates.

`setReplication(PRODPUBS, PRODSUB, MASTDB.BMOORE, SALES, MASTDB.BMOORE, SALESJRN, refresh);` The replication method for the source table named SALES (in the database MASTDB) owned by BMOORE that is currently selected to the subscription PRODSUB will be set to refresh.
The journal named SALESJRN (in the database MASTDB) owned by BMOORE is used for source table updates.

`setReplication(HQAGENT, BRANCHSYS, CUSTLIB, CUSTOMER, refresh, rrn);`The replication method for the source table named CUSTOMER located in library CUSTLIB that is currently selected to the subscription BRANCHSYS will be set to refresh.
The update method for this table will be set to `rrn`.

Parent topic: Management Console commands

setTableStatus - Set Source Table Status

This command sets the status of a source table selected to the specified subscription.
The subscription that you reference in this command must have been defined through Management Console, as there is no command to create a new subscription.
Setting the status of a source table through Management Console is discussed in more detail in Management Console documentation.

Syntax

```
setTableStatus(<agent>, <subscription>, <path>, <table>, <status>,);
```

Parameters

- <agent>
- The name of the datastore that will have a selected source table set to a recognized state.
- The datastore that you specify must be either a dual or source datastore. For more information about datastores, see the Management Console documentation.

- <subscription>
- The name of the subscription to which the source table is selected.

- <path>
- Depending on the type of datastore, the owner of the source table or the library where the source table is located.

- <table>
- The name of the source table that will be set to a recognized state.

- <status>
- The state that will be assigned to the source table specified in the command.
- This parameter must be set to either idle, refresh, or active. For definitions of these states, see the Management Console documentation.

Result

None.

Example

`setTableStatus(ORAPUBS, NTORASUB, GSMITH, EMP, idle);`The source table named EMP owned by GSMITH that is currently selected to the subscription NTORASUB will be set to an idle state.
`setTableStatus(PRODPUBS, PRODSUB, MASTDB.BMOORE, SALES, refresh);` The source table named SALES (in the database MASTDB) owned by BMOORE that is currently selected to the subscription PRODSUB will be set to a refresh state.
`setTableStatus(HQAGENT, BRANCHSYS, CUSTLIB, CUSTOMER, active);`The source table named CUSTOMER located in library CUSTLIB that is currently selected to the subscription BRANCHSYS will be set to an active state.

Parent topic: Management Console commands

startRefresh - Start Refresh

This command starts refresh activities for one or all subscriptions defined for a datastore.
Refresh activities can be started for all selected source tables or just those tables that have a status of REFRESH.
The subscriptions that you reference in this command must have been defined through Management Console, as there is no command to create new subscriptions. Starting refresh activities through Management Console is discussed in more detail in Management Console documentation.

Syntax

```
startRefresh(<agent>, <subscription> [, tabrfsh]);
```

Parameters

- <agent>

- The name of the datastore that will have refresh activities started by this command.
- The datastore that you specify must be either a dual or source datastore. For more information about datastores, see the Management Console documentation.

- <subscription>

- The subscriptions that will have refresh activities started by this command.
- You can specify a subscription name or the special value of `*all` to reference all subscriptions defined for the specified datastore.

- tabrfsh

- Indicates whether all or certain selected tables are included in refresh activities.
- This is an optional parameter. You can specify one of the following special values:
  - flagged—selected source tables that have a status of refresh are included in refresh activities.
  - *all—all selected source tables are included in refresh activities regardless of table status.

  If this parameter is omitted, the default setting is flagged.
  Note: Depending on the value of the second parameter (subscription), this parameter applies to either a single subscription or all subscriptions.

Result

None.

Example

`startRefresh(ORAPUBS, NTORASUB);`Refresh activities are started for source tables that have a status of REFRESH and are selected to the NTORASUB subscription.

`startRefresh(ORAPUBS, *all, flagged);` Refresh activities are started for source tables that have a table status of REFRESH and are selected to any subscription defined for the ORAPUBS datastore.

`startRefresh(HQAGENT, BRANCHSYS, *all);`Refresh activities are started for all source tables selected to the BRANCHSYS subscription.

Parent topic: Management Console commands

Related reference

- setTableStatus - Set Source Table Status

startMirror - Start Mirroring

This command starts mirroring activities for one or all subscriptions defined for a datastore.
Continuous or net change mirroring can be chosen when this command is invoked. If continuous mirroring is selected, any updates to a source table are propagated in real time to the subscription database. With continuous mirroring, the subscription database is always consistent with the source database. This approach is appropriate for implementations where changes must be accessible virtually immediately on the subscription database. Net change mirroring is used when you want to accumulate source table changes and transfer them periodically. After the transfer is complete, all mirroring activity ends. This approach should be adopted if it is important that transfers occur during off-peak periods or if instantaneous updates are not necessary.
The subscriptions that you reference in this command must have been defined through Management Console, as there is no command to create new subscriptions. Starting mirroring activities through Management Console is discussed in more detail in theManagement Console documentation.

Syntax

```
startMirror(<agent>, <subscription> [, mode]);
```

Parameters

- <agent>

- The name of the datastore that will have mirroring activities started by this command.
- The datastore that you specify must be either a dual or source datastore. For more information about datastores, see the Management Console documentation.

- <subscription>

- The subscriptions that will have mirroring activities started by this command.
- You can specify a subscription name or the special value of `*all` to reference all subscriptions defined for the specified datastore.

- mode

- Indicates whether continuous or net change mirroring is used when this command is invoked.
- This is an optional parameter. You can specify one of the following special values:
  - continuous—source table updates are mirrored to the subscription database in real-time.
  - net—accumulated table updates are mirrored to the subscription database when this command is invoked.

  If this parameter is omitted, the default setting is continuous.
  Note: Depending on the value of the second parameter (subscription), this parameter applies to either a single subscription or all subscriptions.

Result

None. **Example**

`startMirror(ORAPUBS, NTORASUB);`Continuous mirroring is started for source tables that are selected to the NTORASUB subscription.
`startMirror(ORAPUBS, *all, net);` Net change mirroring is started for

source tables that are selected to any subscription defined for the ORAPUBS datastore.
`startMirror(HQAGENT,`
`BRANCHSYS, continuous);`Continuous mirroring is started for source tables that are selected to the BRANCHSYS subscription.

Parent topic: Management Console commands

stopAll - Stop All Replication Activities

This command stops all active replication activities involving the specified datastore. This command immediately stops all replication activities. Replication is not stopped in a controlled manner to allow current operations to complete. Therefore, you should only use this command in extreme circumstances where it is necessary to stop all replication activities at the time the command is issued.

Note: This command does not end the established connection to the specified datastore.

Stopping all replication activities through Management Console is discussed in more detail in the Management Console documentation.

Syntax

```
stopAll(<agent>);
```

Parameters

- <agent>

- The name of the datastore that will have all active replication activities stopped by this command.
- The datastore that you specify must be either a dual or source datastore.

Result

None.

Example

`stopAll(ORAPUBS);`All active replication activities associated with the datastore ORAPUBS will be stopped.

Parent topic: Management Console commands

stopMirror - Stop Mirroring

This command stops mirroring activities for one or all subscriptions defined for a datastore.
Mirroring activities can be stopped immediately or in a controlled manner. A controlled stop will shut down mirroring activities cleanly. This is the normal way to end mirroring. Mirroring will end after the operation has completed. An immediate stop will shut down mirroring activities as quickly as possible, and should only be used in extreme circumstances. No attempt is made to shut down mirroring activities cleanly. Therefore, ensure that the current record is applied on the subscription database, and clean up system resources such as database connections, shared memory, and user queues.
The subscriptions that you reference in this command must have been defined through Management Console, as there is no command to create new subscriptions. Stopping mirroring activities through Management Console is discussed in more detail in theManagement Console documentation.

Syntax

```
stopMirror(<agent>, <subscription> [, mode]);
```

Parameters

- <agent>

- The name of the datastore that will have mirroring activities started by this command.
- The datastore that you specify must be either a dual or source datastore. For more information about datastores, see the Management Console documentation.

- <subscription>

- The subscriptions that will have mirroring activities stopped by this command.
- You can specify a subscription name or the special value of *all to reference all subscriptions defined for the specified datastore.

- mode

- Indicates whether mirroring activities are stopped immediately or in a controlled manner.
- This is an optional parameter. You can specify one of the following special values:
  - immediate—mirroring is immediately stopped.
  - controlled—mirroring is stopped in a controlled manner.
- If this parameter is omitted, the default setting is controlled.
- Most of the time you should specify controlled. Specify immediate only if you have attempted to stop a mirroring in a controlled manner, but the datastore replication engine is not responding.
- Depending on the value of the second parameter (subscription), this parameter applies to either a single subscription or all subscriptions.

Result

None. **Example**

`stopMirror(ORAPUBS, NTORASUB);`Mirroring is stopped in a controlled manner for source tables selected to the NTORASUB subscription.
`stopMirror(ORAPUBS, *all, immediate);` Mirroring is immediately stopped for source tables that are selected to any subscription defined for the datastore

ORAPUBS.
`stopMirror(HQAGENT,`
`BRANCHSYS, controlled);`Mirroring is stopped in a controlled manner for all source tables selected to the BRANCHSYS subscription.

Parent topic: Management Console commands

stopRefresh - Stop Refresh

This command stops refresh activities for one or all subscriptions defined for a datastore.

Refresh activities can be stopped immediately or in a controlled manner. A controlled stop will shut down refresh activities properly. This is the normal way to end a refresh. A refresh will end after the operation has completed. An immediate stop will shut down refresh activities as quickly as possible, and should only be used in extreme circumstances. No attempt is made to shut down a refresh operation properly. Therefore, ensure that the current record is applied on the subscription database, and clean up system resources such as database connections, shared memory, and user queues.

The subscriptions that you reference in this command must have been defined through the Management Console application, as there is no command to create new subscriptions.

Stopping refresh activities through Management Console is discussed in more detail in theManagement Console documentation.

Syntax

```
stopRefresh(<agent>, <subscription> [, mode]);
```

Parameters

- <agent>

- The name of the datastore that will have refresh activities stopped by this command.
- The datastore that you specify must be either a dual or source datastore. For more information about datastores, see the Management Console documentation.

- <subscription>

- The subscriptions that will have refresh activities stopped by this command.
- You can specify a subscription name or the special value of *all to reference all subscriptions defined for the specified datastore.

- mode

- Indicates whether refresh activities are stopped immediately or in a controlled manner.
- This is an optional parameter. You can specify one of the following special values:
  - immediate—refresh operation is immediately stopped.
  - controlled—refresh operation is stopped in a controlled manner.
- If this parameter is omitted, the default setting is controlled.
- Most of the time you should specify controlled. Specify immediate only if you have attempted to stop a refresh operation in a controlled manner, but the datastore replication engine is not responding.
- Depending on the value of the second parameter (subscription), this parameter applies to either a single subscription or all subscriptions.

Result

None. **Example**

`stopRefresh(ORAPUBS, NTORASUB);`Refresh activities are stopped in a controlled manner for source tables selected to the NTORASUB subscription.

`stopRefresh(ORAPUBS, *all,`

`immediate);` Refresh activities are immediately stopped for source tables that are selected to any subscription defined for the datastore ORAPUBS.
`stopRefresh(HQAGENT, BRANCHSYS, controlled);`Refresh activities are stopped in a controlled manner for all source tables selected to the BRANCHSYS subscription.

Parent topic: Management Console commands

subscribeTable - Select Table to Subscription

This command selects a table in the catalog to the specified subscription.
The subscription that you reference in this command must have been defined through Management Console, as there is no command to create a new subscription.

Syntax

```
subscribeTable(<agent>, <subscription>, <path>, <table>);
```

Parameters

- <agent>
- The name of the datastore that will have a table in its catalog selected to the specified subscription.
- The datastore that you specify must be either a dual or source datastore.

- <subscription>
- The name of the subscription to which the table in the catalog will be selected.

- <path>
- Depending on the type of datastore, the owner of the catalog table or the library where the catalog table is located.

- <table>
- The name of the table in the catalog that will be selected to the specified subscription.

Result

None.

Example

`subscribeTable(ORAPUBS, NTORASUB, GSMITH, EMP);`The catalog table named EMP owned by GSMITH will be selected to the subscription NTORASUB.

`subscribeTable(PRODPUBS, PRODSUB, MASTDB.BMOORE, SALES);` The catalog table named SALES (in the database MASTDB) owned by BMOORE will be selected to the subscription PRODSUB.

`subscribeTable(HQAGENT, BRANCHSYS, CUSTLIB, CUSTOMER);`The catalog table named CUSTOMER located in library CUSTLIB will be selected to the subscription BRANCHSYS.

Parent topic: Management Console commands

unSubscribeTable - Deselect Table from Subscription

This command deselects a source table from the specified subscription.
The subscription that you reference in this command must have been defined through Management Console, as there is no command to create a new subscription.

Syntax

```
unSubscribeTable(<agent>, <subscription>, <path>, <table>);
```

Parameters

- <agent>
- The name of the datastore that will have a source table deselected from one of its subscriptions.
- The datastore that you specify must be either a dual or source datastore.

- <subscription>
- The name of the subscription from which the source table will be deselected.

- <path>
- Depending on the type of datastore, the owner of the source table or the library where the source table is located.

- <table>
- The name of the source table that will be deselected from the specified subscription.

Result

None.

Example

`unSubscribeTable(ORAPUBS, NTORASUB, GSMITH, EMP);`The source table named EMP owned by GSMITH will be deselected from the subscription NTORASUB.

`unSubscribeTable(PRODPUBS, PRODSUB, MASTDB.BMOORE, SALES);` The source table named SALES (in the database MASTDB) owned by BMOORE will be deselected from the subscription PRODSUB.

`unSubscribeTable(HQAGENT, BRANCHSYS, CUSTLIB, CUSTOMER);`The source table named CUSTOMER located in library CUSTLIB will be deselected from the subscription BRANCHSYS.

Parent topic: Management Console commands

Scripting support

A script file contains a sequence of Management Console commands that is compiled to support batch file processing. A script file cannot include logic statements (for example, conditional branching and looping), and so the sequence of commands that is performed is the same each time its corresponding batch file is executed.

Running a sequence of Management Console commands in batch mode is accomplished by performing the following steps:

1. Creating a Script File.
2. Compiling a Script File.
3. Running a Batch File.

In this section, you will learn:

- **Creating a script file**
- **Compiling a script file**
- **Running a batch file**

Parent topic: Management Console Commands reference

Creating a script file

There are no restrictions on the name, extension, or location of the script file. Although a simple text editor like Notepad will typically be used to create a script file, any plain text editor can be used to create script files.

Script files must contain a strict sequence of Management Console commands. Logic statements are not supported in script files. Each command in a script file must end with a semicolon (;).

Comments

In addition to Management Console commands, you can include comments. In script files, a comment line starts with two consecutive forward slash characters (//) and continues to the end of the line. A comment line or block is not terminated by a recognized character or character sequence.

When compiling a script file, comments are ignored.

Commands

All Management Console commands that you can interactively submit may be included in a script file. However, you should avoid including an execute command that references the script file in which the command is contained. This results in recursive batch file executions without termination.

In addition, you should only include commands that you can perform based on your current role.

Sample Script File

The following sample script file can be modified for your working environment:

```
// This script file starts replication for tables that have been
// selected to a subscription, successfully described, and assigned
// to subscription tables.
//
// Connect to Access Server.
connectServer(EAHIGGINS, PINEAPPLE, LOCALHOST, 10101);
//
// Connect to datastore.
connectAgent(HQAGENT);
//
// In the event that replication is in progress, stop all replication
// activities.
stopAll(HQAGENT);
//
// Perform an initial refresh.
startRefresh(HQAGENT, BRANCHSYS, *all);
//
// Start mirroring.
startMirror(HQAGENT, BRANCHSYS, continuous);
//
// If an error is encountered during a batch file run, continue
// processing.
errorHandle(continue);
//
// Run a batch file that displays subscription activity and status
// information.
```

```
execute(EA.BAT);
//
// Disconnect from datastore.
disconnectAgent(HQAGENT);
//
// Disconnect from Access Server.
disconnectServer();
```

After creating a script file, it must be compiled.

Parent topic: Scripting support

Related concepts

- Compiling a script file
- Management Console commands
- User roles and permissions

Compiling a script file

After creating a script file, it must be compiled to produce a batch file. This step verifies command syntax, but does not verify specific names (such as datastore names, subscription names, and so on) identified in a command invocation. If the script file contains connectServer - Connect to Access Server, compiling also encrypts the Management Console user password in the batch file. This prevents the password from being seen by other users. When the batch file is executed, the encrypted password is internally decrypted before connectServer - Connect to Access Server is invoked.

If the script file you are compiling contains execute - Run Batch File commands, you must individually compile each script file (in no specific order) identified in the execute - Run Batch File commands. Compiling the script file containing the execute - Run Batch File commands does not automatically compile nested script files.

After compiling a script file, do not modify the generated batch file that you will run. If you edit the batch file after compilation, you will not be able to run this file.

Note: Script files must be compiled outside an interactive command environment.

See also:

- **To compile a script file (Windows)**
- **To compile a script file (Linux or UNIX)**
- **To compile a script file (Telnet)**

Parent topic: Scripting support

Related concepts

- Creating a script file
- Running a batch file

Related reference

- connectServer - Connect to Access Server
- execute - Run Batch File

To compile a script file (Windows)

About this task

Perform the following steps based on the operating system running on the workstation where Management Console has been installed.

Procedure

1. Open a command prompt window.
2. Change the current directory to the location where Management Console was installed.Tip: To avoid having to specify long path names when compiling script files, create and compile your script files in the Management Console installation directory. However, do not overwrite any of the existing installation files in this directory.
3. Issue the following command:`online.bat  -c  <scriptfile><outputfile>`
   where <scriptfile> is the full path name of the script file that you want to compile, and <outputfile> is the full path name of the output file that is produced as a result of compiling the script file.

Example

Example:

```
online.bat -c c:\scripts\script1.txt c:\scripts\script1c.txt
```

Parent topic: Compiling a script file

To compile a script file (Linux or UNIX)

About this task

Perform the following steps based on the operating system running on the workstation where Access Server has been installed.

Procedure

1. Open a command prompt window
2. Change the current directory to the location where Access Server was installed.
3. Issue the following command: `bin/online -c <scriptfile><outputfile>`
   where <scriptfile> is the full path name of the script file that you want to compile, and <outputfile> is the full path name of the output file that is produced as a result of compiling the script file.

Example

Example:

```
bin/online -c script1 script1c
```

Parent topic: Compiling a script file

To compile a script file (Telnet)

About this task

Perform the following steps based on the operating system running on the workstation where Management Console has been installed.

Procedure

1. Perform the steps under Windows outside of the Telnet command environment that you may have opened to submit commands interactively. If necessary, use exit - Close Command Environment to end the command environment.
2. If the compile is successful, the specified batch file will be created. This is the file that you will run.

Parent topic: Compiling a script file

Related concepts

- Opening command environments
- Running a batch file

Related tasks

- To compile a script file (Windows)

Related reference

- exit - Close Command Environment

Running a batch file

After successfully compiling a script file, you can now run the compiled script file that was produced as a result of the compilation. You can use execute - Run Batch File, or the online program to run the compiled script file.

execute Command

This command can be submitted in an interactive command environment or included in a script file.

online.bat/dtsconsole Programs

Outside an interactive command environment, you can run a compiled script file by using the online.bat or online program. This option allows you to use a system scheduler to run the batch file at specific times. The command to run the online.bat or online program can be specified in the scheduler to run the named batch file in an unattended mode.

Note: The procedure described in this section assumes that you have just completed the steps in Compiling a Script File.

Perform the following step based on the operating system running on the workstation where Management Console or Access Server has been installed:

Windows

Issue the following command:

```
online.bat  -b  <scriptfile>
```

where<scriptfile> is the full path name of the script file that was produced as a result of compiling the script file.

Example:

```
online.bat -b c:\scripts\script1c
```

Linux or UNIX

Issue the following command:

```
bin/online -b <scriptfile>
```

where <scriptfile> is the full path name of the script file that was produced as a result of compiling the script file.

Example:

```
bin/online.bat -b script1c
```

Telnet

Perform the steps under Windows.

Parent topic: Scripting support

Related concepts

- Compiling a script file
- Opening command environments
- Creating a script file

Related reference

- execute - Run Batch File

User roles and permissions

This section identifies and briefly describes the four user roles. It then lists the user permissions for each command or action.

In this section, you will learn:

- **User roles**
- **User permissions**

Parent topic: Management Console Commands reference

User roles

A Management Console user belongs to one of the following four roles:

- **TS System administrator**—The user can view, configure, and initiate replication.
- **TS Administrator**—The user can view, configure, and initiate replication, but cannot change InfoSphere® CDC system parameters.
- **TS Operator**—The user can view and initiate replication.
- **TS Monitor**—The user can only view replication.

For more information about user roles, see your Management Console documentation.

To determine the role to which a user is currently assigned, see the `getRole` method in the `UserProfile` interface. For more information, see InfoSphere CDC API reference - Javadocs.

Parent topic: User roles and permissions

User permissions

This section identifies those methods in the Management Console API that are only accessible by specific user roles. The remainder of the methods in the API do not require a specific user role. For more information, see User roles.

Note: The permissions for each method apply to a Management Console user connected to an Access Server. The `User` property specified when invoking the connect method in the `DataSource` interface determines the user that will be affected by the specified permissions. For more information on this method and interface, see InfoSphere® CDC API reference - Javadocs.

The following table uses these conventions for user permissions.

- **Unlimited**—The user has permission to invoke the method.
- **No permission**—The user does not have permission to invoke the method. If the user attempts to invoke restricted methods will result in the `ApiException` exception being thrown. In cases where a method is out of scope (such as a method that is equivalent to an Access Manager function) or called automatically for all user roles, this symbol will also indicate that a method is not applicable at the user level.

Table 1. Permissions for methods

Method	TS System Administrator	TS Administrator	TS Operator	TS Monitor
Opening Command Environments	Unlimited	Unlimited	Unlimited	Unlimited
addTable - Add Table to Catalog	Unlimited	Unlimited	No permission	No permission
assignTable - Assign Subscription Table	Unlimited	Unlimited	No permission	No permission
connectAgent - Connect to Replication Agent	Unlimited	Unlimited	Unlimited	Unlimited
connectServer - Connect to Access Server	Unlimited	Unlimited	Unlimited	Unlimited
deAssignTable - De-assign Subscription Table	Unlimited	Unlimited	No permission	No permission

describe - Describe Selected Tables	Unlimited	Unlimited	Unlimited	No permission
disconnectAgent - Disconnect from Replication Agent	Unlimited	Unlimited	No permission	No permission
disconnectServer - Disconnect from Access Server	Unlimited	Unlimited	No permission	No permission
errorHandle - Define Error Handler	Unlimited	Unlimited	No permission	No permission
execute - Run Batch File	Unlimited	Unlimited	No permission	No permission
exit - Close Command Environment	Unlimited	Unlimited	No permission	No permission
getAgents - Get Accessible Replication Agents	Unlimited	Unlimited	No permission	No permission
getSubscriptionActivity - Get Subscription Activity	Unlimited	Unlimited	No permission	No permission
getSubscriptions - Get Subscriptions for Replication Agent	Unlimited	Unlimited	No permission	No permission
getSubscriptionStatus - Get Subscription Status	Unlimited	Unlimited	No permission	No permission
getSubscriptionTables - Get Subscription Tables	Unlimited	Unlimited	No permission	No permission

getTableStatus - Get Publication Table Status	Unlimited	Unlimited	No permission	No permission
help - Display Command Help	Unlimited	Unlimited	Unlimited	Unlimited
reAddTable - Re-add Table in Catalog	Unlimited	Unlimited	No permission	No permission
removeTable - Remove Table from Catalog	Unlimited	Unlimited	No permission	No permission
setReplication - Set Replication Method	Unlimited	Unlimited	No permission	No permission
setTableStatus - Set Publication Table Status	Unlimited	Unlimited	No permission	No permission
startRefresh - Start Refresh	Unlimited	Unlimited	Unlimited	No permission
startMirror - Start Mirroring	Unlimited	Unlimited	Unlimited	No permission
stopAll - Stop All Replication Activities	Unlimited	Unlimited	Unlimited	No permission
stopMirror - Stop Mirroring	Unlimited	Unlimited	Unlimited	No permission
stopRefresh - Stop Refresh	Unlimited	Unlimited	Unlimited	No permission
subscribeTable - Select Table to Subscription	Unlimited	Unlimited	No permission	No permission
unSubscribeTable - De-select Table from Subscription	Unlimited	Unlimited	No permission	No permission
Compiling a Script File	Unlimited	Unlimited	Unlimited	Unlimited

Running a Batch File	Unlimited	Unlimited	Unlimited	Unlimited

Parent topic: User roles and permissions

Troubleshooting

If you encounter issues while running InfoSphere® CDC, you have a number of options for tracking and troubleshooting issues to help with problem resolution.
There are three methods that you can use in InfoSphere CDC for tracking and troubleshooting issues:
- Data Collection with the IBM® Support Assistant (ISA DC)
- Management Console Support Assistant
- The dmsupportinfo command, which is executed on the replication engine

If you are trying to troubleshoot issues with InfoSphere CDC version 10.2 or later on Linux, UNIX and Windows operating systems, you should use the ISA DC tool unless otherwise instructed by IBM Technical Support.
In this section, you will learn:

- **Using the IBM Support Assistant (ISA DC)**
- **Troubleshooting and contacting IBM Support**

Using the IBM Support Assistant (ISA DC)

You can use the IBM® Support Assistant Data Collection tool (ISA DC) to collect InfoSphere® CDC data to provide to IBM Technical Support to assist you in troubleshooting issues with InfoSphere CDC, to request a product enhancement or to ask a question about InfoSphere CDC.

ISA DC can be used with InfoSphere CDC replication engines that are version 10.2 or later, except InfoSphere CDC for z/OS®.

The ISA DC tool is included in the InfoSphere CDC installation process, and does not require configuration. The executable files are located in the isa folder in the InfoSphere CDC directory. Simply run the isadc.bat, isadc.sh or index.html file, as appropriate, to launch the tool.

Prerequisites and considerations for using ISA DC

The following prerequisite must be satisfied on the machine on which ISA DC will be run, in order to successfully use the tool:

- IBM JRE/JDK version 1.6 or later

The following issues should be taken into consideration before you attempt to use ISA DC:

- ISA DC cannot be run remotely. It must be run on the machine where the instance is configured.
- ISA DC cannot be used to collect data from InfoSphere CDC for z/OS.
- If InfoSphere CDC is installed but you have not configured an instance or are unable to configure an instance, ISA DC can still be used to collect minimal data to assist IBM Technical Support in resolving the issue.

See also:

- **To use ISA DC to collect data for a product problem (command line)**
- **To use ISA DC to collect data for a product problem (GUI)**
- **To use ISA DC to collect data for a question or an enhancement request (command line)**
- **To use ISA DC to collect data for a question or an enhancement request (GUI)**

Parent topic: Troubleshooting

To use ISA DC to collect data for a product problem (command line)

Procedure

1. Launch the IBM® Support Assistant.Run the isadc.bat or isadc.sh file, located in the isa\isadc folder in the root directory of the InfoSphere® CDC instance.

2. Enter 1 to accept the license agreement and press Enter.After the license agreement has been accepted, it will not be shown again.

3. Provide a file name and press Enter.The name provided will be given to the .zip file containing the data collection results.
   If you do not want to assign a name to the data collection results, press Enter and a default name will be used.

4. Enter 1 to confirm your chosen file name and press Enter to continue.
5. Enter 1 to run the InfoSphere Change Data CaptureSupport Assistant Data Collector and press Enter.The Welcome page is displayed.

6. Read the Welcome page information and enter 1 to proceed. Press Enter.
7. Enter 1 to collect data for a product problem and press Enter.
8. Enter 1 to process your input and continue collecting data. Press Enter.If you want to cancel the collection, enter 2 and press Enter.
   If you want to go back and change your input for the previous step, enter 3 and press Enter.

9. Select the name of the InfoSphere CDC instance for which data will be collected. If you have multiple instances of InfoSphere CDC configured, you will be asked to select which instance for which you want to collect. Enter the corresponding number for the instance name and press Enter.
   If you have a single InfoSphere CDC instance configured, it will be selected automatically and this step will be skipped.
   Even if you do not have an instance configured, ISA DC will still collect what data is available. If no instance is configured, you can skip to Step 14.

10. Enter 1 to process your input and continue collecting data. Press Enter.If you want to cancel the collection, enter 2 and press Enter.
    If you want to go back and change your input for the previous step, enter 3 and press Enter.

11. If your selected instance is not running, you will be alerted by ISA DC. As only minimal data is available if the instance is stopped, it is preferable that the instance be running during data collection.Try to start your instance. When the instance is running, enter 1 and press Enter.
    If you cannot start your instance and want to continue the data collection process, enter 2 and press Enter.

12. Enter 1 to process your input and continue collecting data. Press Enter.If you want to cancel the collection, enter 2 and press Enter.
    If you want to go back and change your input for the previous step, enter 3 and press Enter.

13. If the instance is running, you will be asked for information regarding when the problem occurred.
    A. Enter the date and time when you think the problem began and press Enter. This information must be entered in the following format: yyyy-mm-dd hh:mm:ss

    B. Enter 1 to process your input and continue collecting data. Press Enter.If you want to cancel the collection, enter 2 and press Enter.
    If you want to go back and change your input for the previous step, enter 3 and press Enter.

    C. Determine the period of time for which the data will be collected and press Enter.The amount specified will be applied as a before value and an after value to the date and time specified previously. For example, if you select 1 Day as the time period, data will be collect for 24 hours before the specified date and time and for the 24 hours after the specified date and time.

    D. Enter 1 to process your input and continue collecting data. Press Enter.If you want to cancel the collection, enter 2 and press Enter.
    If you want to go back and change your input for the previous step, enter 3 and press Enter.

14. Select the method to transfer the data collection archive file and press Enter. Choose one of the following options:
    - **Send using secure transfer to IBM Support (HTTPS)**—Sends the .zip file to IBM Support using a secure protocol.
    - **Send using FTP to IBM Support (unencrypted)**—Sends the .zip file to IBM Support using an unencrypted protocol.
    - **Send using FTP to another location (unencrypted)**—Sends the .zip file to a recipient of your choice, using an unencrypted protocol.
    - **End the collection without sending**—Ends the data collection and creates the .zip file, but does not transfer it.

15. Enter 1 to process your input and continue collecting data. Press Enter.If you want to cancel the collection, enter 2 and press Enter.
    If you want to go back and change your input for the previous step, enter 3 and press Enter.

16. If you chose to end the collection without sending the output, ISA DC will notify your when the .zip file has been successfully created. Enter 1 and press Enter to exit the application.
17. If you chose to transfer the file using HTTPS, follow these steps:
    A. If you want to receive a confirmation email when the upload was successful,

enter an email address and press Enter. If you do not want to receive confirmation, press Enter to continue.

B. Enter 1 to process your input and continue collecting data. Press Enter.If you want to cancel the collection, enter 2 and press Enter.
If you want to go back and change your input for the previous step, enter 3 and press Enter.

C. Enter the PMR number that was given to you by IBM Technical Support and press Enter. Ensure that the PMR number follows the required naming convention of PMRNumber.BranchNumber.CountryCode. If an unknown PMR number is entered, you will be asked to correct the PMR number and re-send the data.

D. Enter 1 to process your input and continue collecting data. Press Enter.If you want to cancel the collection, enter 2 and press Enter.
If you want to go back and change your input for the previous step, enter 3 and press Enter.

18. If you chose to transfer the file to IBM Technical Support using unencrypted FTP, follow these steps:
    A. Enter the PMR number that was given to you by IBM Technical Support and press Enter. Ensure that the PMR number follows the required naming convention of PMRNumber.BranchNumber.CountryCode. If an unknown PMR number is entered, you will be asked to correct the PMR number and re-send the data.

    B. Enter 1 to process your input and continue collecting data. Press Enter.If you want to cancel the collection, enter 2 and press Enter.
    If you want to go back and change your input for the previous step, enter 3 and press Enter.

19. If you chose to transfer the file using unencrypted FTP, follow these steps:
    A. Enter the FTP host name and press Enter.
    B. Enter the user name and press Enter.
    C. Enter the password for the user name and press Enter.
    D. Enter the path for the directory on the FTP server and press Enter.
    E. Enter 1 to process your input and continue collecting data. Press Enter.If you want to cancel the collection, enter 2 and press Enter.
    If you want to go back and change your input for the previous step, enter 3 and press Enter.

20. When you receive notice that the operation has completed successfully, enter 1 and press Enter to exit the application.

Parent topic: Using the IBM Support Assistant (ISA DC)

To use ISA DC to collect data for a product problem (GUI)

Procedure

1. Launch the IBM® Support Assistant.Run the index.html file, located in the isa\isadc folder in the root directory of the InfoSphere® CDC instance.

2. Read the license agreement and click OK to accept it.After the license agreement has been accepted, it will not be shown again.

3. Click Start.The Welcome screen opens.

4. Click OK.
5. Select A product problem from the drop down box.
6. Click OK.
7. Select the name of an InfoSphere CDC instance from the drop down list and click OK.If you have multiple instances of InfoSphere CDC configured, you will be asked to select which instance for which you want to collect.
   If you have a single InfoSphere CDC instance configured, it will be selected automatically and this step will be skipped.

8. If your selected instance is not running, you will be alerted by ISA DC. As only minimal data is available if the instance is stopped, it is preferable that the instance be running during data collection.Try to start your instance. When the instance is running, select Yes, I have started the instance from the drop down box and click OK.
   If you cannot start your instance and want to continue the data collection process, select No, continue with minimal data collection from the drop down box and click OK.

9. If the instance is running, you will be asked for information regarding when the problem occurred. Enter the date and time when you think the problem began and click OK.This information must be entered in the following format: yyyy-mm-dd hh:mm:ss.

10. Determine the period of time for which the data will be collected and click OK. Choose one of the following values:
    - 6 hours
    - 12 hours
    - 1 Day
    - 2 Days
    - 7 Days
    The amount specified will be applied as a before value and an after value to the date and time specified previously. For example, if you select `1 Day` as the time period, data will be collect for 24 hours before the specified date and time and for the 24 hours after the specified date and time.

11. If you chose to end the collection without sending the output, select Do not transfer data to IBM. ISA DC will notify you when the .zip file has been successfully created.
12. If you want to transfer the data to IBM using a secure transfer (HTTPS), select the Transfer to IBM option.
    A. Choose the HTTPS transfer type option.
    B. Enter the PMR number that was given to you by IBM Technical Support. Ensure that the PMR number follows the required naming convention of PMRNumber.BranchNumber.CountryCode. If an unknown PMR number is entered, you will be asked to correct the PMR number and re-send the data.

    C. Enter your email address.
    D. Click Transfer.
13. If you want to transfer the data to IBM using unencrypted FTP, select the Transfer to IBM option.
    A. Choose the FTP transfer type option.
    B. Enter the PMR number that was given to you by IBM Technical Support. Ensure that the PMR number follows the required naming convention of PMRNumber.BranchNumber.CountryCode. If an unknown PMR number is entered, you will be asked to correct the PMR number and re-send the data.

    C. Click Transfer.
14. If you choose to send the data to a location other than IBM using unencrypted FTP, click Transfer to another server via FTP
    A. Enter the email address or IP address of the recipient in the Hotmail/IP Address field.
    B. Enter the user name.
    C. Enter the password.
    D. Enter the path for the directory on the FTP server.
    E. Click Transfer.
15. When you receive notice that the operation has completed successfully, click Browse directory if you want to see the file you created or click Start New Collection to collect more data.To exit the application, close your browser tab or window.

Parent topic: Using the IBM Support Assistant (ISA DC)

To use ISA DC to collect data for a question or an enhancement request (command line)

Procedure

1. Launch the IBM® Support Assistant.Run the isadc.bat or isadc.sh file, located in the isa\isadc folder in the root directory of the InfoSphere® CDC instance.

2. Enter 1 to accept the license agreement and press Enter.After the license agreement has been accepted, it will not be shown again.

3. Provide a file name and press Enter.The name provided will be given to the .zip file containing the data collection results.
   If you do not want to assign a name to the data collection results, press Enter and a default name will be used.

4. Enter 1 to confirm your chosen file name and press Enter to continue.
5. Enter 1 to run the InfoSphere Change Data CaptureSupport Assistant Data Collector and press Enter.The Welcome page is displayed.

6. Read the Welcome page information and enter 1 to proceed. Press Enter.
7. Enter 2 to collect data for a question or an enhancement request and press Enter.
8. Enter 1 to process your input and continue collecting data. Press Enter.If you want to cancel the collection, enter 2 and press Enter.
   If you want to go back and change your input for the previous step, enter 3 and press Enter.

9. Select the method to transfer the data collection archive file and press Enter. Choose one of the following options:
   - **Send using secure transfer to IBM Support (HTTPS)**—Sends the .zip file to IBM Support using a secure protocol.
   - **Send using FTP to IBM Support (unencrypted)**—Sends the .zip file to IBM Support using an unencrypted protocol.
   - **Send using FTP to another location (unencrypted)**—Sends the .zip file to a recipient of your choice, using an unencrypted protocol.
   - **End the collection without sending**—Ends the data collection and creates the .zip file, but does not transfer it.

10. Enter 1 to process your input and continue collecting data. Press Enter.If you want to cancel the collection, enter 2 and press Enter.
    If you want to go back and change your input for the previous step, enter 3 and press Enter.

11. If you chose to end the collection without sending the output, ISA DC will notify your when the .zip file has been successfully created. Enter 1 and press Enter to exit the application.
12. If you chose to transfer the file using HTTPS, follow these steps:
    A. If you want to receive a confirmation email when the upload was successful,

    enter an email address and press Enter. If you do not want to receive confirmation, press Enter to continue.
  B. Enter 1 to process your input and continue collecting data. Press Enter.If you want to cancel the collection, enter 2 and press Enter.
     If you want to go back and change your input for the previous step, enter 3 and press Enter.

  C. Enter the PMR number that was given to you by IBM Technical Support and press Enter. Ensure that the PMR number follows the required naming convention of PMRNumber.BranchNumber.CountryCode. If an unknown PMR number is entered, you will be asked to correct the PMR number and re-send the data.

  D. Enter 1 to process your input and continue collecting data. Press Enter.If you want to cancel the collection, enter 2 and press Enter.
     If you want to go back and change your input for the previous step, enter 3 and press Enter.

13. If you chose to transfer the file to IBM Technical Support using unencrypted FTP, follow these steps:
  A. Enter the PMR number that was given to you by IBM Technical Support and press Enter. Ensure that the PMR number follows the required naming convention of PMRNumber.BranchNumber.CountryCode. If an unknown PMR number is entered, you will be asked to correct the PMR number and re-send the data.

  B. Enter 1 to process your input and continue collecting data. Press Enter.If you want to cancel the collection, enter 2 and press Enter.
     If you want to go back and change your input for the previous step, enter 3 and press Enter.

14. If you chose to transfer the file using unencrypted FTP, follow these steps:
  A. Enter the FTP host name and press Enter.
  B. Enter the user name and press Enter.
  C. Enter the password for the user name and press Enter.
  D. Enter the path for the directory on the FTP server and press Enter.
  E. Enter 1 to process your input and continue collecting data. Press Enter.If you want to cancel the collection, enter 2 and press Enter.
     If you want to go back and change your input for the previous step, enter 3 and press Enter.

15. When you receive notice that the operation has completed successfully, enter 1 and press Enter to exit the application.

Parent topic: Using the IBM Support Assistant (ISA DC)

To use ISA DC to collect data for a question or an enhancement request (GUI)

Procedure

1. Launch the IBM® Support Assistant.Run the index.html file, located in the isa\isadc folder in the root directory of the InfoSphere® CDC instance.

2. Read the license agreement and click OK to accept it.After the license agreement has been accepted, it will not be shown again.

3. Click Start.The Welcome screen opens.

4. Click OK.
5. Select A question or enhancement request from the drop down box.
6. Click OK.
7. If you chose to end the collection without sending the output, select Do not transfer data to IBM. ISA DC will notify you when the .zip file has been successfully created.
8. If you want to transfer the data to IBM using a secure transfer (HTTPS), select the Transfer to IBM option.
   A. Choose the HTTPS transfer type option.
   B. Enter the PMR number that was given to you by IBM Technical Support. Ensure that the PMR number follows the required naming convention of PMRNumber.BranchNumber.CountryCode. If an unknown PMR number is entered, you will be asked to correct the PMR number and re-send the data.

   C. Enter your email address.
   D. Click Transfer.
9. If you want to transfer the data to IBM using unencrypted FTP, select the Transfer to IBM option.
   A. Choose the FTP transfer type option.
   B. Enter the PMR number that was given to you by IBM Technical Support. Ensure that the PMR number follows the required naming convention of PMRNumber.BranchNumber.CountryCode. If an unknown PMR number is entered, you will be asked to correct the PMR number and re-send the data.

   C. Click Transfer.
10. If you choose to send the data to a location other than IBM using unencrypted FTP, click Transfer to another server via FTP
    A. Enter the email address or IP address of the recipient in the Hotmail/IP Address field.
    B. Enter the user name.
    C. Enter the password.
    D. Enter the path for the directory on the FTP server.
    E. Click Transfer.
11. When you receive notice that the operation has completed successfully, click Browse directory if you want to see the file you created or click Start New Collection to collect more data.To exit the application, close your browser tab or

window.

Parent topic: Using the IBM Support Assistant (ISA DC)

Troubleshooting and contacting IBM Support

The following support page contains the latest troubleshooting information and details on how to open a service request with IBM® Support:
- http://www.ibm.com/software/data/infosphere/support/change-data-capture/

For contact information in your region:
- http://www.ibm.com/planetwide/

Parent topic: Troubleshooting