WebSphere® Information Integration

Version 9

# ASNCLP Program Reference for Replication and Event Publishing

SC19-1018-00

WebSphere® Information Integration

IBM

Version 9

# ASNCLP Program Reference for Replication and Event Publishing

SC19-1018-00

**Note**

Before using this information and the product that it supports, be sure to read the general information under "Notices and trademarks" on page 185.

# Contents

## Chapter 4. ASNCLP commands for the Replication Alert Monitor . . . . . . 145

## Accessing information about IBM . . 183

## Notices and trademarks . . . . . . . 185

## Index . . . . . . . . . . . . . . . 189

# Chapter 1. Getting started with the ASNCLP program

The ASNCLP program is a command-line interface for administration of SQL replication, Q replication, and event publishing.

The replication programs store information about your configurations in control tables. The ASNCLP commands create, modify, and remove this information.

For example, the ASNCLP program provides a command to create a Q subscription. The output of a command is a SQL script that inserts information into the control tables about the source, target, queues, and other options.

Each ASNCLP command works with only a subset of information. You can use multiple commands together to generate the SQL for an entire configuration.

Before you run the ASNCLP program, you need to configure your environment. The following topics discuss how to do this:

- "Supported operating systems"
- "Setting up a Java environment to run the ASNCLP program" on page 2
- "Binding z/OS packages for the ASNCLP program" on page 2

Once your environment is configured, you'll need to know a few basics about using the ASNCLP program. These are discussed in the following topics:

- "Running the ASNCLP commands in interactive mode" on page 3
- "Running the ASNCLP commands using an input file" on page 3

You can learn more about how to build an ASNCLP input script for your replication or publishing configuration by reading these topics:

- "Sample ASNCLP scripts for setting up SQL replication" on page 6
- "Sample ASNCLP scripts for Q replication" on page 53
- "Sample ASNCLP scripts for setting up event publishing" on page 132
- "Sample ASNCLP scripts for setting up the Replication Alert Monitor" on page 146

## Supported operating systems

The ASNCLP program runs on the Linux®, AIX®, Solaris, and Windows® operating-systems. The ASNCLP commands will not run natively on z/OS® or OS/400®.

The ASNCLP commands will generate replication definitions for all operating system environments that are supported by the replication products: z/OS, OS/390®, OS/400 (SQL replication only), UNIX® (AIX, Solaris, HP-UX), Linux, 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 **db2 connect** statement to each of the servers.

**Restriction:** The ASNCLP program does not support z/VM® or VSE because DB2® in these operating-system environments does not support the replication architecture for DB2™ Version 8 and later.

## Setting up a Java environment to run the ASNCLP program

The ASNCLP program runs in a Java Runtime Environment, and you must add certain jar files in your Java CLASSPATH environment variable to run the ASNCLP program.

**Procedure**

To set up a Java Runtime environment to run the ASNCLP program, specify the following jar files in your Java CLASSPATH environment variable:

- *INSTDIR*\java\Common.jar
- *INSTDIR*\tools\db2cmn.jar
- *INSTDIR*\tools\db2replapis.jar
- *INSTDIR*\tools\db2qreplapis.jar
- *INSTDIR*\tools\jt400.jar

Where *INSTDIR* is the DB2 instance directory. On Linux and UNIX operating systems, the instance directory is the *INSTDIR*/sqllib directory, where *INSTDIR* is the home directory of the instance owner. On Windows operating systems, the instance directory is the \sqllib directory where DB2 was installed.

### Examples

**Windows**: To set the CLASSPATH environment variable from a Windows command prompt:

```
set CLASSPATH=%
CLASSPATH%;
c:\sqllib\java\Common.jar;
c:\sqllib\tools\db2cmn.jar;
c:\sqllib\tools\db2replapis.jar;
c:\sqllib\tools\db2qreplapis.jar;
c:\sqllib\tools\jt400.jar;
```

**Linux, UNIX**: To set the CLASSPATH environment variable from a UNIX command prompt:

```
export
CLASSPATH=$CLASSPATH
:/u/myinst/sqllib/java/Common.jar;
:/u/myinst/sqllib/tools/db2cmn.jar;
:/u/myinst/sqllib/tools/db2replapis.jar;
:/u/myinst/sqllib/tools/db2qreplapis.jar;
:/u/myinst/sqllib/tools/jt400.jar;
```

## Binding z/OS packages for the ASNCLP program

Before you use the ASNCLP program with DB2 UDB for z/OS, you must bind the basic DRDA and CLI packages to the DB2 subsystem that you will be working with.

**Before you begin**

Before you can bind the z/OS packages, you must connect to the z/OS server.

**Procedure**

To bind the basic z/OS packages for the ASNCLP program, open an operating system command prompt and issue the following command:

```
db2 bind @ddcsmvs.lst isolation ur blocking all
db2 bind @db2cli.lst  isolation ur blocking all
```

If you do not perform this bind, the first time you use the ASNCLP program with a DB2 UDB for z/OS server, the ASNCLP program might return the following error message:

```
ASN1560E  The replication action ended in error. An SQL error was encountered.
SQL Message: "[IBM][CLI Driver][DB2] SQL0805N  Package
"package_name" was not found.  SQLSTATE=51002
```

## Running the ASNCLP commands in interactive mode

You can run the ASNCLP commands in interactive mode from a command prompt.

**Procedure**

To run the ASNCLP commands in interactive mode:

1. Open an operating system command prompt and issue the following command:

   ```
   ASNCLP
   ```

   The ASNCLP command starts the ASNCLP program and changes the command prompt to `Repl >`.
2. Issue any of the ASNCLP commands. For example: To set the Q Capture server to the database *aliasname*, issue the following command:

   ```
   SET SERVER CAPTURE TO DBALIAS aliasname
   ```
3. To exit the ASNCLP program, issue the following command:

   ```
   quit
   ```

To get help for the ASNCLP program, issue the following command from an operating system command prompt:

```
ASNCLP ?
```

## Running the ASNCLP commands using an input file

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

**Procedure**

To run the ASNCLP commands in batch mode using an input file:

1. Create an input file that contains the ASNCLP commands that you want to run. Commands in the input file must be delimited by the semicolon (;) and can span multiple lines. You can also add comments to the input file by beginning the comment line with a number (#) sign.
2. Open an operating system command prompt and issue the following command:

   ```
   ASNCLP -f myfile.in
   ```

   In the example the input-file name is myfile.in and can consist of any valid file name plus an extension. You can also specify a full file path and file name. For example:

```
ASNCLP -f c:\temp\myfile.in
```

The ASNCLP command starts the ASNCLP program, which processes all of the commands in the input file until it encounters an error or the end of the file.

If your input file does not contain the quit command, you can exit the ASNCLP program by issuing the following command:

```
quit
```

# Chapter 2. ASNCLP commands for SQL replication

The ASNCLP commands for SQL replication define and change objects such as control tables, registrations, and subscription sets.

“Sample ASNCLP scripts for setting up SQL replication” on page 6 demonstrates how you can combine SQL replication commands to create an ASNCLP setup script.

Table 1 lists the ASNCLP commands for SQL replication and links to topics that describe each command.

*Table 1. ASNCLP commands for SQL replication*

| If you want to ... | Use this command |
|---|---|
| Add columns to an existing member | “ALTER MEMBER ADD COLS command” on page 9 |
| Change the properties of a registration | “ALTER REGISTRATION command” on page 10 |
| Change the properties of a subscription set | “ALTER SUBSCRIPTION SET command” on page 13 |
| Establish a session for SQL replication | “ASNCLP SESSION SET TO command (SQL replication)” on page 15 |
| Create control tables | “CREATE CONTROL TABLES FOR command (SQL replication)” on page 16 |
| Create a subscription-set member | “CREATE MEMBER command” on page 18 |
| Create a registration | “CREATE REGISTRATION command” on page 24 |
| Create a SQL statement that is processed with an existing subscription set | “CREATE STMT command” on page 29 |
| Create a subscription set | “CREATE SUBSCRIPTION SET command” on page 30 |
| Drop control tables | “DROP CONTROL TABLES ON command” on page 32 |
| Delete a subscription-set member | “DROP MEMBER command” on page 34 |
| Delete a registration | “DROP REGISTRATION command” on page 34 |
| Delete SQL statements for an existing subscription set | “DROP STMT command” on page 35 |
| Delete a subscription set | “DROP SUBSCRIPTION SET command” on page 36 |
| Control a manual full refresh for offline load procedures | “OFFLINE LOAD command” on page 36 |
| Promote a registration | “PROMOTE REGISTRATION command” on page 37 |
| Promote a subscription set | “PROMOTE SUBSCRIPTION SET command” on page 39 |
| Set a source and target Capture schema for all task commands | “SET CAPTURE SCHEMA command (SQL replication)” on page 41 |
| Specify whether to drop the table space when you drop the replication object that it contains | “SET DROP command (SQL replication)” on page 42 |
| Set the log file name for the ASNCLP program | “SET LOG command” on page 43 |
| Specify a name for the output files that contain the SQL scripts | “SET OUTPUT command (SQL replication)” on page 43 |
| Set up customization rules for creating table space objects | “SET PROFILE command (SQL replication)” on page 44 |
| Specify whether to automatically run the SQL statements before the ASNCLP commands process the next task command | “SET RUN SCRIPT command (SQL replication)” on page 48 |

*Table 1. ASNCLP commands for SQL replication (continued)*

| If you want to ... | Use this command |
|---|---|
| Specify the server (database) used in the ASNCLP session, authentication information, and other required parameters for connecting to the server | "SET SERVER command (SQL replication)" on page 50 |
| Enable and disable the tracing for the ASNCLP commands | "SET TRACE command" on page 52 |

## Sample ASNCLP scripts for setting up SQL replication

This sample contains five ASNCLP scripts for setting up a basic SQL replication environment. It includes Capture control tables, a registration, Apply control tables, a subscription set, and a subscription-set member.

Each ASNCLP script generates one or more SQL scripts to create a replication object. Because some replication objects depend on the existence of other objects, run the ASNCLP scripts and the SQL scripts that they generate in the following order:

1. Capture control tables
2. Registration
3. Apply control tables
4. Subscription set (generates two SQL scripts that create definitions at the Capture control server and the Apply control server).
5. Subscription-set member (generates two SQL scripts that create definitions at the Capture control server and the Apply control server, which is also the target server).

Table 2 on page 9 below the sample describes each SQL script.

This sample has a section for each ASNCLP script, which you can copy to a text file and run by using the `ASNCLP -f` *`filename`* command. Within the code sample in each section, details about each group of commands are preceded by a comment character (#).

### ASNCLP script 1 (Capture control tables)

This script generates SQL statements that create Capture control tables at the SAMPLE database. It includes commands for the following tasks:

1 Setting the environment
2 Creating Capture control tables
3 Ending the ASNCLP session

```
# 1 Setting the environment
# The scope of environment commands is the entire ASNCLP script file unless
# the commands are overridden by another value.
# In the SET SERVER command, the user ID and password are optional. If you omit
# these keywords, the ASNCLP will use the implicit ID and password for connecting
# to the database.
# The SET OUTPUT command generates a SQL script, capctrl.sql.
# The SET LOG command sets one log file, capctrl.err, to record results for the
# ASNCLP script.
# The SET RUN SCRIPT LATER option allows you to review the SQL scripts before
# they are run.
SET SERVER CAPTURE TO DB SAMPLE ID DB2ADMIN PASSWORD "passw0rd";
SET OUTPUT CAPTURE SCRIPT "capctrl.sql";
SET LOG "capctrl.err";
SET RUN SCRIPT LATER;

# 2 Creating Capture control tables
# This command generates a SQL script that connects to the SAMPLE database and
# creates the control tables. The UOW table will be created in the TSUOW100
# table space, and the other control tables will be created in the TSASN100
# table space.

CREATE CONTROL TABLES FOR CAPTURE SERVER IN UW UOW TSUOW100 OTHERS TSASN100;

# 3 Ending the ASNCLP session.

QUIT;
```

## ASNCLP script 2 (registration)

This script generates SQL statements that register the STAFF table at the SAMPLE database for replication. It includes commands for the following tasks:

1 Setting the environment
2 Registering a source table
3 Ending the ASNCLP session.

```
# 1 Setting the environment
# The SET OUTPUT command generates a SQL script, register.sql.

SET SERVER CAPTURE TO DB SAMPLE ID DB2ADMIN PASSWORD "passw0rd";
SET OUTPUT CAPTURE SCRIPT "register.sql";
SET LOG "register.err";
SET RUN SCRIPT LATER;

# 2 Registering a source table.
# The CREATE REGISTRATION command specifies the STAFF table. The
# DIFFERENTIAL REFRESH STAGE keywords specify to update the target table
# periodically when the source table changes, and create a change-date (CD)
# table named CDSTAFF. Because the COLS keyword is not specified, all columns
# in the source table are registered.

CREATE REGISTRATION (DB2ADMIN.STAFF) DIFFERENTIAL REFRESH STAGE CDSTAFF;

# 3 Ending the ASNCLP session.
QUIT;
```

## ASNCLP script 3 (Apply control tables)

This script generates SQL statements that create the Apply control tables at the TARGET database. It includes commands for the following tasks:

1 Setting the environment
2 Creating Apply control tables
3 Ending the ASNCLP session

```
# 1 Setting the environment
# The SET OUTPUT command generates a SQL script, appctrl.sql.
SET SERVER CONTROL TO DB TARGET ID DB2ADMIN PASSWORD "passw0rd";
SET OUTPUT CONTROL SCRIPT "appctrl.sql";
SET LOG "appctrl.err";
SET RUN SCRIPT LATER;

# 2 Creating Apply control tables
# These statements generate a SQL script that connects to the TARGET database
# and creates the control tables.

CREATE CONTROL TABLES FOR APPLY CONTROL SERVER IN UW OTHERS TSASN100;
```

```
# 3 Ending the ASNCLP session.

QUIT;
```

## ASNCLP script 4 (subscription set)

This script generates SQL statements that create a subscription set. It includes commands for the following tasks:

1 Setting the environment
2 Creating a subscription set
3 Ending the ASNCLP session.

```
# 1 Setting the environment
# Two SET SERVER commands are used because the subscription set definitions
# are stored in both the Capture control tables and Apply control tables.
# The SET OUTPUT command generates two scripts: capsubset.sql, which inserts
# set definitions into the Capture control tables, and appsubset.sql, which
# inserts set definitions into the Apply control tables.

SET SERVER CAPTURE TO DB SAMPLE ID DB2ADMIN PASSWORD "passw0rd";
SET SERVER CONTROL TO DB TARGET ID DB2ADMIN PASSWORD "passw0rd";
SET OUTPUT CAPTURE SCRIPT "capsubset.sql" CONTROL SCRIPT "appsubset.sql";
SET LOG "subset.err";
SET RUN SCRIPT LATER;

# 2 Creating a subscription set
# The CREATE SUBSCRIPTION SET command creates a set named SET00 with an Apply
# qualifier of AQ00. It specifies that the set be activated indefinitely on a
# specified start date and time, and that the set be processed every minute.

CREATE SUBSCRIPTION SET SETNAME SET00 APPLYQUAL AQ00 ACTIVATE YES
TIMING INTERVAL 1 START DATE "2006-10-22" TIME "09:00:00.000000";

# 3 Ending the ASNCLP session.

QUIT;
```

## ASNCLP script 5 (subscription-set member)

This script generates SQL statements that create a subscription-set member. It includes commands for the following tasks:

1 Setting the environment
2 Creating a subscription-set member
3 Ending the ASNCLP session.

```
# 1 Setting the environment
# The SET OUTPUT command generates two SQL scripts, capmember.sql and
# appmember.sql. Because the target server is the same as the Apply control
# server, no separate script is generated for the target server. These
# scripts define the member at the Capture control server and Apply
# control server.

SET SERVER CAPTURE TO DB SAMPLE ID DB2ADMIN PASSWORD "passw0rd";
SET SERVER CONTROL TO DB TARGET ID DB2ADMIN PASSWORD "passw0rd";
SET SERVER TARGET TO DB TARGET ID DB2ADMIN PASSWORD "passw0rd";
SET OUTPUT CAPTURE SCRIPT "capmember.sql" CONTROL SCRIPT "appmember.sql"
SET LOG "member.err";
SET RUN SCRIPT LATER;

# 2 Creating a subscription-set member.
# The SET PROFILE command creates a profile, TBSPROFILE, to store options
# for the tablespace that is used by the target table . The CREATE MEMBER command
# specifies the SET00 set, AQ00 Apply qualifier, and STAFF source table. The
# TRGSTAFF target table is specified as a user copy with all columns registered.

SET PROFILE TBSPROFILE FOR OBJECT TARGET TABLESPACE OPTIONS UW USING
FILE "/tmp/db/ts/TSTRG.TS" SIZE 700 PAGES;
CREATE MEMBER IN SETNAME SET00 APPLYQUAL AQ00 ACTIVATE YES SOURCE STAFF
TARGET NAME TRGSTAFF DEFINITION IN TSTRG00 CREATE USING PROFILE TBSPROFILE
TYPE USERCOPY COLS ALL REGISTERED;

# 3 Ending the ASNCLP session.

QUIT;
```

### Output of the scripts

In addition to the log files, this sample produces six SQL script files in the same directory where you run the ASNCLP program. Table 2 describes the files.

*Table 2. SQL script files that are created by the sample ASNCLP scripts*

| Output file | Contains SQL to ... |
|---|---|
| capctrl.sql | Create Capture control tables |
| register.sql | Register a source table |
| appctrl.sql | Create Apply control tables |
| capsubset.sql | Insert definitions for a subscription set into the Capture control tables |
| appsubset.sql | Insert definitions for a subscription set into the Apply control tables |
| capmember.sql | Insert definitions for a subscription-set member into the Capture control tables |
| appmember.sql | Insert definitions for a subscription-set member into the Apply control tables |

## ALTER MEMBER ADD COLS command

Use the ALTER MEMBER ADD COLS command to add columns to an existing member in an existing subscription set.

### Syntax

```
►►──ALTER MEMBER ADD COLS──IN──SETNAME──setname──APPLYQUAL──applyqual──SOURCE──────────────────────►

►──┬──────────────┬──objname──TARGET──┬──────────────┬──objname─────────────────────────────────────►
   └─objowner──.──┘                   └─objowner──.──┘

►─COLS──(──EXPRESSION──"source-col-or-expr"──┬───────────────────────┬──)──────────────────────────►◄
                                              └─TARGET──name──┬───┬──┘
                                                              └─+─┘
```

### Parameters

**SETNAME** *setname*
Specifies the subscription-set name.

**APPLYQUAL** *applyqual*
Specifies the Apply qualifier for the subscription set.

**SOURCE** *objowner.objname*
Specifies the source object's owner and name.

**TARGET** *objowner.objname*
Specifies the target object's owner and name.

**COLS**
Specifies the columns to add. You can specify multiple columns by using commas and parentheses.

**EXPRESSION** *"source-col-or-expr"*
Specifies an expression for the column. The double quotation marks are required.

**TARGET** *name*
Specifies the target's column name.

**+** Specifies that the column is part of the primary key.

### Usage notes

- For update-anywhere subscription sets, the columns are added to the members for both replication directions (master-to-replica and replica-to-master).
- The Capture schema for the target table is inherited from the subscription set.

### Example

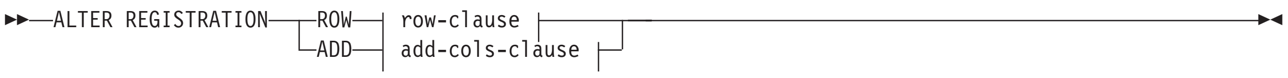
To add column NEWSTAFF to the existing subscription set SET00 :

```
ALTER MEMBER ADD COLS IN SETNAME SET00 APPLYQUAL AQ00 SOURCE DB2ADMIN.STAFF
TARGET DB2ADMIN.TRGSTAFF COLS (EXPRESSION "source-col-or-expr" TARGET NEWSTAFF)
```

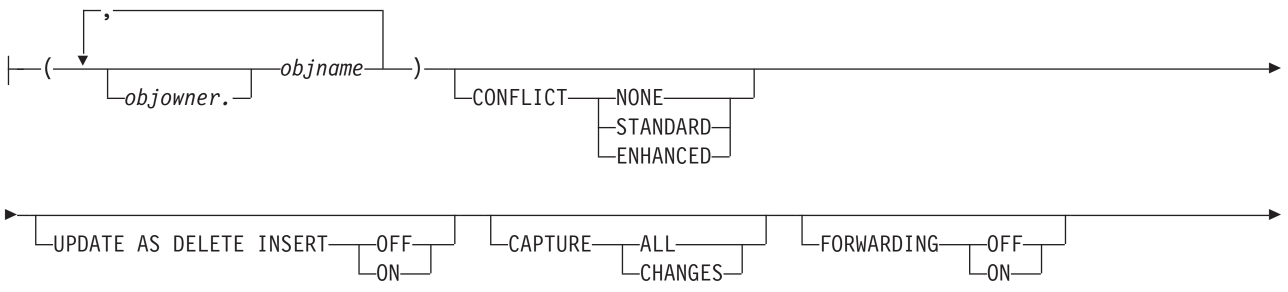
## ALTER REGISTRATION command

Use the ALTER REGISTRATION command to alter a registration row in the IBMSNAP_REGISTER table and to add new columns to a registered source.

### Syntax



**row-clause:**



```
►──┬─────────────────────┬──┬─────────────────────────────┬──────────────────────────────┤
   └─FULL REFRESH──┬─ON──┤  └─STOP── ON── ERROR──┬─ON──┤
                   └─OFF─┘                       └─OFF─┘
```

**add-cols-clause:**

```
├──┬───────────┬──objname──┬──────────────────────────────────────────────────┬──────────►
   └─objowner.─┘           │            ┌─,──────────────────────────────┐    │
                           └─COLS──(──▼─colname──┬──────────────────┬──┴─)─┘
                                                 └─IMAGE──┬─AFTER──┤
                                                          ├─BOTH───┤
                                                          └─BEFORE─┘

►──┬──────────────────────┬─────────────────────────────────────────────────────────────┤
   └─PREFIX──┬─X──────────┤
             └─befimgpref─┘
```

## Parameters

**ROW**
Specify to alter a registration row in the IBMSNAP_REGISTER table.

**ADD**
Specify to add new columns from a source object to a registration. This parameter only applies if the source object is a table or nickname.

*objowner*
Specifies the owner of the registered source object (table, view, or nickname). You can specify multiple objects.

*objname*
Specifies the name of the registered source object (table, view, or nickname). You can specify multiple objects.

**CONFLICT**
Specifies the conflict-detection level.

**NONE**
No conflict detection. Conflicting updates between the master table and the replica table will not be detected. This option is not recommended for update-anywhere replication. This is the default.

**STANDARD**
Moderate conflict detection. During each Apply cycle, the Apply program compares the key values in the master's CD table with those in the replica's CD table. If the same key value exists in both CD tables, it is a conflict. In the case of a conflict, the Apply program will undo the transaction that was previously committed at the replica by reading from the replica's CD table and keeping only the changes that originated at the master.

**ENHANCED**
Conflict detection that provides the best data integrity among the master and its replicas. As with standard detection, the Apply program compares the key values in the master's CD table with those in the replica's CD table during each Apply cycle. If the same key value exists in both CD tables, it

is a conflict. However, with enhanced detection, the Apply program waits for all in-flight transactions to commit before checking for conflicts. To ensure that it catches all in-flight transactions, the Apply program locks all target tables in the subscription set against further transactions and begins conflict detection after all changes are captured in the CD table. In case of a conflict, the Apply program will undo the transaction that was previously committed at the replica by reading from the replica's CD table and keeping only the changes that originated at the master.

**UPDATE AS DELETE INSERT**

**ON**
Specify to capture updates as delete-insert pairs.

**OFF**
Specify to capture updates as updates. This is the default.

**CAPTURE**

**ALL**
Specify to capture everything.

**CHANGES**
Specify to capture only changes.

**FORWARDING**

**OFF**
Specify not to forward changes from this source.

**ON**
Specify to forward changes from this source.

**FULL REFRESH**

**ON**
Specify to allow full refreshes for this source.

**OFF**
Specify to not allow full refreshes for this source.

**STOP ON ERROR**

**ON**
Specify to stop the Capture program if it detects an error for this registration.

**OFF**
Specify to not stop the Capture program if it detects an error for this registration.

**COLS**
Specifies the columns that you want to register.

*colname*
Specifies a list of the columns that you want to register.

**IMAGE**

**AFTER**
Specify to register only after-image columns.

**BOTH**
Specify to register both after-image and before-image columns.

**BEFORE**
Specify to register only before-image columns.

**PREFIX**

- If you specify **IMAGE AFTER**, the prefix will be null and the source will not allow any before-image columns.
- If you specify **IMAGE BOTH** or **IMAGE BEFORE** and do not specify **PREFIX**, a default value of `X` is used as a prefix for the before images. If you specify **PREFIX**, that value is used.
- If you choose **IMAGE BOTH** and do not specify a prefix, the before-imaged prefix will be `X`.

You cannot alter an existing before-image prefix using the ALTER REGISTRATION ROW command. However, you can add that prefix to a new before-image column. If the existing before-image prefix is null and you want to add a before-image column to the existing registration, you can specify the before-image prefix using the ALTER REGISTRATION ADD command. If you do not specify the prefix, the ASNCLP program sets it to a default value of `X`.

### Usage notes

The parameters in this command do not have default values.

If you add a column to a CD table when the registered source also has an internal CCD table associated with it, you must:

- Use the ALTER ADD REGISTRATION COL command to add a column to the CD table
- Use the ALTER ADD SUBSCRIPTION MEMBER COL command to add a column to the internal CCD table. If you do not do this step, you will not be able to add the column to any target table that is dependent on the registered source.

### Example 1

To alter a registration row for DB2ADMIN.STAFF that captures updates as delete-insert pairs:

```
ALTER REGISTRATION ROW (DB2ADMIN.STAFF) UPDATE AS DELETE INSERT ON
```

### Example 2

To alter a registration by adding a new column C002 to table DB2ADMIN.STAFF:

```
ALTER REGISTRATION ADD DB2ADMIN.STAFF COLS (C002 IMAGE BOTH)
```

## ALTER SUBSCRIPTION SET command

Use the ALTER SUBSCRIPTION SET command to alter certain values for a subscription set.

### Syntax

```
►►──ALTER SUBSCRIPTION SET──SETNAME──setname──APPLYQUAL──applyqual──SETTYPE──┬─R──────────────┬──────────►
                                                                             ├─U─┬──────────┬─┤
                                                                             │   ├─F ONLY─┤ │
                                                                             │   └─S ONLY─┘ │
                                                                             └─P──────────────┘
```

```
►──┬───────────────────────┬──┬────────────────────────────────────────────────────────────┬──────────►
   └─ACTIVATE──┬─NO───┬────┘  └─TIMING──┬─EVENT──eventname──────────────────────────────┬─┘
               ├─YES──┤                 ├─INTERVAL──minutes─────────────────────────────┤
               └─ONCE─┘                 ├─BOTH──EVENT──eventname── INTERVAL──minutes────┤
                                        └─CONTINUOUS────────────────────────────────────┘

►──┬──────────────────────┬──┬─────────────────────────────┬───────────────────────────────────────►◄
   └─BLOCKING──minutes────┘  └─COMMIT COUNT──┬─n────┬──────┘
                                             └─NULL─┘
```

## Parameters

**SETNAME** *setname*
Specifies the subscription-set name.

**APPLYQUAL** *applyqual*
Specifies the Apply qualifier for the subscription set.

**SETTYPE**
Specifies the subscription set type.

**R** Specifies a read-only set. This is the default.

**U** Specifies an update-anywhere set. The default is both F and S directions.

**F ONLY**
Specifies an update-anywhere set in the F direction only, where the source table is the replica and the target table is the master.

**S ONLY**
Specifies an update-anywhere set in the S direction only, where the source table is the mast table or the other source, and the target table is the replica or other copy.

**P** Specifies a peer-to-peer set.

**ACTIVATE**
Specifies whether to activate the subscription set.

**NO**
Specify to not activate the subscription set. This is the default.

**YES**
Specify to activate the subscription set.

**ONCE**
Specify to activate the subscription set for one Apply cycle, then deactivate the subscription set.

**TIMING**
Specifies the timing for the subscription set.

**EVENT** *eventname*
Specifies the event that when posted to the IBMSNAP_SUBS_EVENT table, causes the Apply program to process the subscription set.

**INTERVAL** *minutes*
Specifies the interval for the Apply program to process the subscription set. The default interval is 20 minutes.

**BOTH**
Specifies that this subscription set uses both event and interval timing.

**CONTINUOUS**
Specifies that the Apply program should process the subscription set continuously. This keyword is equivalent to specifying an interval of zero minutes.

**BLOCKING** *minutes*
Specifies a threshold limit to regulate the amount of data to fetch and apply. This keyword controls the MAX_SYNCH_MINUTES column of the IBMSNAP_SUB_SET table.

**COMMIT COUNT** *n*
Specifies the number of transactions that the Apply program should process before issuing a SQL COMMIT statement for the subscription set. Specify a NULL value to have the Apply program issue just one COMMIT statement for the subscription set after it processes the entire set.

### Example 1

To alter the SET00 subscription set to a read-only subscription set type using the AQ00 Apply qualifier and to change the timing interval from 20 minutes to 15 minutes:

```
ALTER SUBSCRIPTION SET SETNAME SET00 APPLYQUAL AQ00 SETTYPE R
 ACTIVATE YES TIMING INTERVAL 15 COMMIT COUNT NULL
```

### Example 2

To alter the SET00 subscription set so that it activates once and sets the source table as the replica and the target table as the master:

```
ALTER SUBSCRIPTION SET SETNAME SET00 APPLYQUAL AQ00 SETTYPE U
 F ONLY ACTIVATE ONCE COMMIT COUNT 5
```

## ASNCLP SESSION SET TO command (SQL replication)

Use the ASNCLP SESSION SET TO command to define an ASNCLP session for SQL replication.

### Syntax

```
►►—ASNCLP SESSION SET TO—SQL REPLICATION————————————————————————————————————►◄
```

### Parameters

**SQL REPLICATION**
Specify to set the ASNCLP session to SQL replication. This ASNCLP session only accepts SQL replication syntax.

### Usage notes

Issue the ASNCLP SESSION SET command before all other commands in an ASNCLP session. If you do not issue the ASNCLP SESSION SET command, the ASNCLP program defaults to SQL replication.

### Example

To set the ASNCLP session to SQL replication:

```
ASNCLP SESSION SET TO SQL REPLICATION
```

## CREATE CONTROL TABLES FOR command (SQL replication)

Use the CREATE CONTROL TABLES FOR command to create a new set of Capture, Apply, or Replication Alert Monitor control tables.

### Syntax

```
►►──CREATE CONTROL TABLES FOR──┬─CAPTURE SERVER─────────┬──┬──────────────────────────────────────────┬──►◄
                               ├─APPLY CONTROL SERVER───┤  └─IN──┬─ZOS──┤ zos-ts-clause ├──────────┬─┘
                               └─MONITOR CONTROL SERVER─┘        ├─UW──┤ uw-ts-clause ├───────────┤
                                                                 └─NONIBM──┤ federated-clause ├───┘
```

**zos-ts-clause:**

```
├──┬──────────────────────────────────────────────────────────────────────┬──────────────────────►
   └─UOW DB dbname──┬─tsname─────────────────┬──┤ prof-clause ├──┘
                    └─NAMING PREFIX──prefix─┘

►──┬──────────────────────────────────────────────────────────────────────┬──────────────────────►
   └─ALERTS DB dbname──┬─tsname─────────────────┬──┤ prof-clause ├──┘
                       └─NAMING PREFIX──prefix─┘

►──┬──────────────────────────────────────────────────────────────────────┬──────────────────────►
   └─PAGE LOCK DB dbname──┬─tsname─────────────────┬──┤ prof-clause ├──┘
                          └─NAMING PREFIX──prefix─┘

►──┬──────────────────────────────────────────────────────────────────────┬──────────────────────┤
   └─ROW LOCK DB dbname──┬─tsname─────────────────┬──┤ prof-clause ├──┘
                         └─NAMING PREFIX──prefix─┘
```

**uw-ts-clause:**

```
├──┬──────────────────────────────────────────────────────┬──────────────────────────────────────►
   └─UOW──┬─tsname─────────────────┬──┤ prof-clause ├──┘
          └─NAMING PREFIX──prefix─┘

►──┬──────────────────────────────────────────────────────┬──────────────────────────────────────┤
   └─OTHERS──┬─tsname─────────────────┬──┤ prof-clause ├──┘
             └─NAMING PREFIX──prefix─┘
```

**federated-clause:**

```
├──┬──────────────────────────────────────────────────────┬──SCHEMA──schemaname──────────────────┤
   └─OTHERS──┬─tsname─────────────────┬──┤ prof-clause ├──┘
             └─NAMING PREFIX──prefix─┘
```

**prof-clause:**

```
├──┬─────────────────────────────────┬───────────────────────────────────────────────────────────┤
   ├─CREATE USING PROFILE──pname─┤
   └─REUSE───────────────────────┘
```

## Parameters

**CAPTURE SERVER**
Specify to create replication control tables for the Capture server.

**APPLY CONTROL SERVER**
Specify to create replication control tables for the Apply control server.

**MONITOR CONTROL SERVER**
Specify to create replication control tables for the Monitor control server.

**IN** Specifies the table space. If you do not specify the **IN** clause, the **CREATE CONTROL TABLES** command uses the DB2 defaults for table spaces.

**ZOS**
Specifies z/OS or OS/390.

**UW**
Specifies Linux, UNIX, or Windows.

**NONIBM**
Specifies federated data source such as Oracle or Informix.

**Federated-clause**

**OTHERS**
Specifies the table space for all replication control tables whenever the tables are created in a non-DB2 database. You specify a table space name or a segment name for only those remote sources that support them.

**SCHEMA**
Specifies the remote schema name for a federated replication source server. The default is the remote user ID. If the schema is in lower or mixed case on the federated data source, you must use double quotation marks around the string to ensure that it is not converted to upper case. Lower case names and quotation marks are recommended for Informix sources.

**UOW**
Specifies the table space for the unit-of-work (UOW) table.

**ALERTS**
Specifies an existing database on z/OS to create the control tables in. This keyword is valid only when creating monitor control servers.

**PAGE LOCK**
Specifies the table space for replication control tables that require page-level locking. The table must be in an existing database.

**ROW LOCK**
Specifies the table space for replication control tables that require row-level locking. The table must be in an existing database.

**DB** *dbname*
**z/OS:** Specifies the name of an existing database. You must specify the database name, even if you set the database name in the profile.

**OTHERS**
Specifies the table space for all replication control tables except the UOW table.

*tsname*
Specifies the table space name for the monitor alerts table. The *tsname* input can be a heterogeneous segment or table space name.

**NAMING PREFIX** *prefix*
Specifies a naming prefix for the control tables.

**CREATE USING PROFILE** *pname*
Specify to create the control tables using the *pname* profile. If you specify the **CREATE USING PROFILE** parameter, the ASNCLP program uses *tsname* as the key (for z/OS, the key is *dbname.tsname*).

**REUSE**
Specify to reuse the current table space or index. You must issue the **CREATE USING PROFILE** parameter before you can use the **REUSE** parameter. When you specify the **REUSE** parameter, the ASNCLP program checks if the table space or index exists for the *tsname*:

- If the table space or index exists, the ASNCLP program resets the flags and passes the fully populated object.
- If the table space or index does not exist, the ASNCLP program displays a syntax error saying that the **CREATE USING PROFILE** parameter is expected.

### Example 1

To create the Capture control tables and to name the UOW table space TSUOW100 and all other table spaces TSASN100:

```
CREATE CONTROL TABLES FOR CAPTURE SERVER IN UW UOW TSUOW100 OTHERS TSASN100
```

### Example 2

To create the Apply control tables and to name all table spaces except the UOW table space TSASN100:

```
CREATE CONTROL TABLES FOR APPLY CONTROL SERVER IN UW OTHERS TSASN100
```

## CREATE MEMBER command

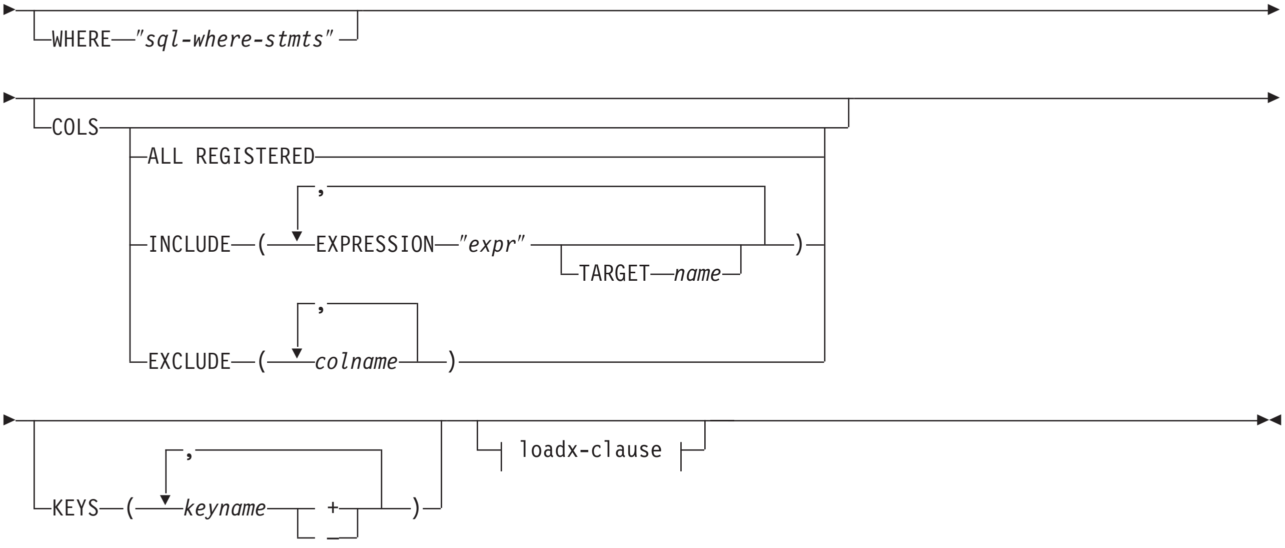
Use the CREATE MEMBER command to add a subscription-set member to an existing subscription set.

Adding a member to a set includes:

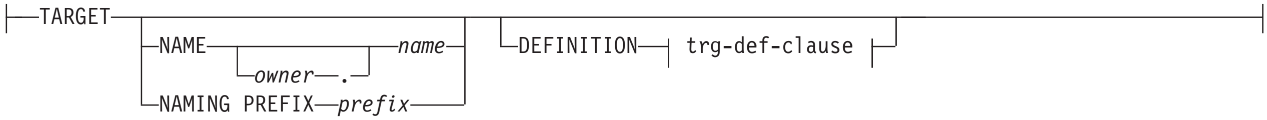
- Creating the mapping between the source and target tables (database objects).
- Creating the mapping between the source and target columns.
- Creating the target table (database object), if it doesn't already exist.
- Creating the target index, if necessary.
- Setting the KEYS value for the index.

### Syntax

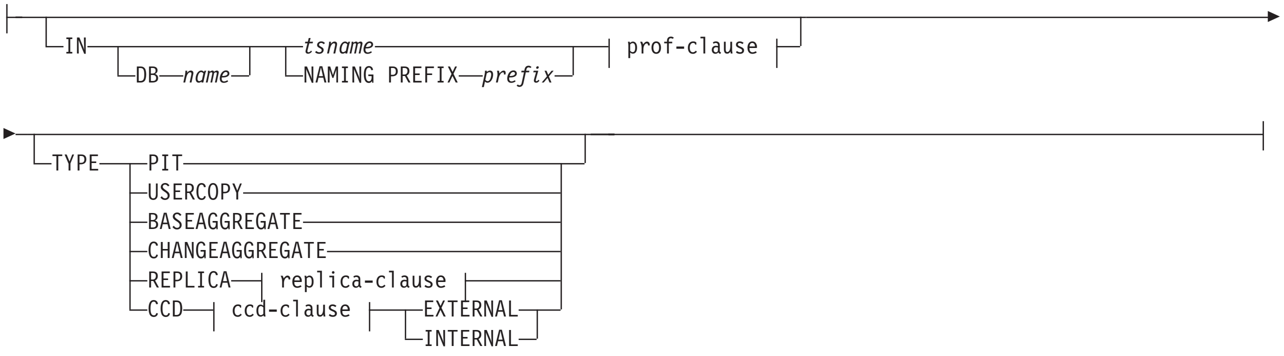
```
►►──CREATE MEMBER──IN──SETNAME──setname──APPLYQUAL──applyq──┬──────────────────────┬──SOURCE──────────►
                                                            └─ACTIVATE──┬─NO───┬─┘
                                                                        ├─YES──┤
                                                                        └─ONCE─┘

►──┬─────────────┬──objname──┬─────────────────┬──┬──────────────────────────┬──────────────────►
   └─objowner──.─┘           └─┤ target-clause ├─┘  └─TGT KEY CHANGE──┬─OFF─┬─┘
                                                                      └─ON──┘
```



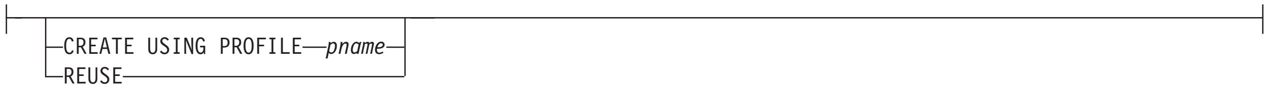
**target-clause:**



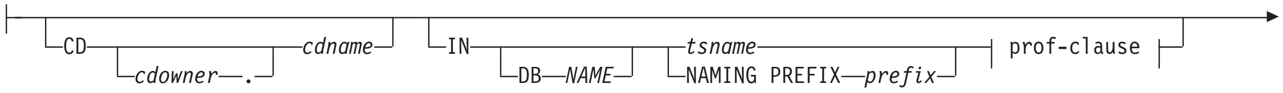
**trg-def-clause:**



**prof-clause:**



**replica-clause:**



```
►─┬───────────────────────────────────┬──┬─────────────────────────┬──┬─────────────────────────┬──────────────►
  └─UPDATE AS DELETE INSERT─┬─OFF─┬─┘  └─FORWARDING─┬─OFF─┬─┘  └─FULL REFRESH─┬─ON──┬─┘
                            └─ON──┘                  └─ON──┘                    └─OFF─┘

►─┬─────────────────────────┬──────────────────────────────────────────────────────────────────────────────┤
  └─STOP ON ERROR─┬─ON──┬─┘
                  └─OFF─┘
```

**ccd-clause:**

```
├──┬─AS SOURCE──────────────────────────────┬─CONDENSED─┬─ON──┬──────────────────────────────────────┤
   ├─WITH UOW COLS─┬─ALL──────────────────┤            └─OFF─┘
   │               └─┤ cols-clause ├──────┤
   └─COMPLETE─┬─ON──┬──────────────────────┘
              └─OFF─┘
```

**cols-clause:**

```
            ┌─,───────┐
            ▼         │
├──(────colname───┴──)──────────────────────────────────────────────────────────────────┤
```

**loadx-clause:**

```
├──┬────────────────────────────────────────────────────────────────────────┬─────────────────────────┤
   └─LOADX──TYPE─┬─NO ASNLOAD──────────────────────────────────────────┬─┘
                 ├─USER DEFINED────────────────────────────────────────┤
                 ├─CROSSLOADER──LOAD SRC NICKNAME──owner.tablename──┤
                 ├─LOAD EXPORT─────────────────────────────────────────┤
                 └─IMPORT EXPORT───────────────────────────────────────┘
```

## Parameters

**SETNAME** *setname*
Specifies the subscription-set name.

**APPLYQUAL** *applyqual*
Specifies the Apply qualifier for the subscription set.

**ACTIVATE**
Specifies whether to activate the subscription set.

**NO**
Specify to not activate the subscription set. This is the default.

**YES**
Specify to activate the subscription set.

**ONCE**
Specify to activate the subscription set for one Apply cycle, then deactivate the subscription set.

**SOURCE** *objowner***.***objname*
Specifies the source object name and owner.

**TGT KEY CHANGE**
Specifies whether the target key can change.

**OFF**
Specifies that the key value cannot change. This is the default.

**ON**
Specifies that the key value can change.

**WHERE** *"sql-where-stmts"*
Specifies the WHERE clause that will be evaluated for this member. The double quotation marks are required.

**COLS**
Specifies the columns to include in the target table.

**ALL REGISTERED**
Specify to include all registered columns.

**INCLUDE**
Specifies the columns to include.

**EXPRESSION** *"expr"*
Specifies the source column or expression. Specify multiple columns or expressions using commas and parentheses.

**TARGET** *name*
Specifies the name of the target column.

**EXCLUDE (***colname***)**
Exclude the specified columns.

**KEYS** *keyname*
Specifies the key names. Include a plus sign (+) for ascending keys and a minus sign (-) for descending keys.

target-clause:

**TARGET**
Specifies the target object.

**NAME** *owner***.***name*
Specifies the target object owner and name.

**NAMING PREFIX** *prefix*
Specifies the prefix to use to generate a target-table name.

**DEFINITION**
Specifies the database, table space, and target-table type.

trg-def-clause:

**IN** Specifies the table space for the target table. If you do not specify the **IN** clause, the command uses the DB2 defaults for table spaces.

**DB** *name*
Specifies the name of the database that contains the target table and its table space. You must specify the database name, even if you set the database name in the profile.

*tsname*
Specifies the name of the table space. For z/OS, the name includes the database name (for example, *"dbname.tsname"*). This command does not create the database. You can specify a heterogeneous segment or table space name, but it must already exist.

**NAMING PREFIX** *prefix*
Specifies a naming prefix to use to create the table space.

**TYPE**
Specifies the type of target table.

**PIT**
Specifies a point-in-time table.

**USERCOPY**
Specifies a user-copy table.

**BASEAGGREGATE**
Specifies a base-aggregate table. This table contains data aggregated from the source or point-in-time table at intervals.

**CHANGEAGGREGATE**
Specifies a change-aggregate table. This table contains data based on changes to a source table (CD or internal CCD table).

**REPLICA**
Specifies a replica table for update-anywhere replication.

**CCD**
Specifies a consistent-change data (CCD) table.

**EXTERNAL**
Specifies that the CCD table is external.

**INTERNAL**
Specifies that the CCD table is internal.

prof-clause:

**CREATE USING PROFILE** *pname*
Specify to use the *tsname* value as the key (for z/OS, the key is *dbname.tsname*).

**REUSE**
Specify to reuse the current table space or index. You must issue the **CREATE USING PROFILE** parameter before you can use the **REUSE** parameter. When you specify the **REUSE** parameter, the ASNCLP program checks if the table space or index exists for the *tsname*:

- If the table space or index exists, the ASNCLP program resets the flags and passes the fully populated object to the API.
- If the table space or index does not exist, the ASNCLP program displays a syntax error saying that the **CREATE USING PROFILE** parameter is expected.

replica-clause:

**CD** *cdowner***.***cdname*
Specifies the name of the object owner and the name of the CD table for the replica table.

**UPDATE AS DELETE INSERT**
Specifies how to handle SQL UPDATE statements.

**OFF**
Specify to capture updates as updates. This is the default.

**ON**
Specify to capture updates as delete-insert pairs.

**FORWARDING**
Specifies whether to forward captured changes to other replicas.

**OFF**
Specify to not forward captured changed.

**ON**
Specify to forward captured changes.

**FULL REFRESH**
Specifies whether to perform a full refresh for the replica table.

**ON**
Specify to perform a full refresh. This is the default.

**OFF**
Specify not to perform a full refresh.

**STOP ON ERROR**
Specifies whether the Capture program is to stop when it encounters an error.

**ON**
Specify to stop the Capture program if a Capture error occurs. This is the default.

**OFF**
Specify to continue the Capture program if a Capture error occurs.

ccd-clause:

**AS SOURCE**
Specifies that the CCD table is a source.

**WITH UOW COLS**

**ALL**
Specifies that the CCD table includes columns from the IBMSNAP_UOW table.

**COMPLETE**
Specifies whether the CCD table is complete.

**ON**
Specifies that the CCD table includes all data. This is the default.

**OFF**
Specifies that the CCD table includes only changes.

**CONDENSED**
Specifies whether to condense the CCD table.

**ON**
Specifies that the CCD table includes only the most recent change for each row. This is the default.

**OFF**
Specifies that the CCD table includes a change history for each row.

cols-clause:

*colname*
Specifies which of the UOW columns should be included in the CCD table. These columns include: IBMSNAP_APPLY_QUAL, IBMSNAP_AUTHID, IBMSNAP_AUTHTKN, IBMSNAP_REJ_CODE, and IBMSNAP_UOWID.

loadx-clause:

**LOADX TYPE**
Specifies the load type to use with this member.

**NO ASNLOAD**
Specify to not use the ASNLOAD for this member.

**USER DEFINED**
Specify to use a user-defined or user-modified ASNLOAD exit.

**CROSSLOADER LOAD SRC NICKNAME** *owner*.*tablename*
Specify the *owner* and *tablename* to use with the LOAD from CURSOR utility for this member.

**LOAD EXPORT**
**Linux, UNIX, and Windows:** Specify to use an EXPORT/LOAD combination for this member.

**IMPORT EXPORT**
**Linux, UNIX, and Windows:** Specify to use an EXPORT/IMPORT combination for this member.

### Usage notes

- The target object is not required for the command, but the command does require a target object so that the ASNCLP program can derive the target name.
- You cannot specify the conflict-detection level for replica-table autoregistration because it is inherited from the master table.
- You cannot specify capturing updates as delete-insert pairs for CCD table autoregistration because there is no Capture program for these tables.
- If the subscription set is empty when you issue this command, the command uses a default value of YES for the **ACTIVATE** keyword.

### Example

To create a member in the SET00 subscription set and TSUOW100 table space using TBSPROFILE profile from the STAFF source table to the TRGSTAFF target:

```
CREATE MEMBER IN SETNAME SET00 APPLYQUAL AQ00 SOURCE DB2ADMIN.STAFF
TARGET NAME DB2ADMIN.TRGSTAFF DEFINITION IN TSUOW100 CREATE USING PROFILE TBSPROFILE
```

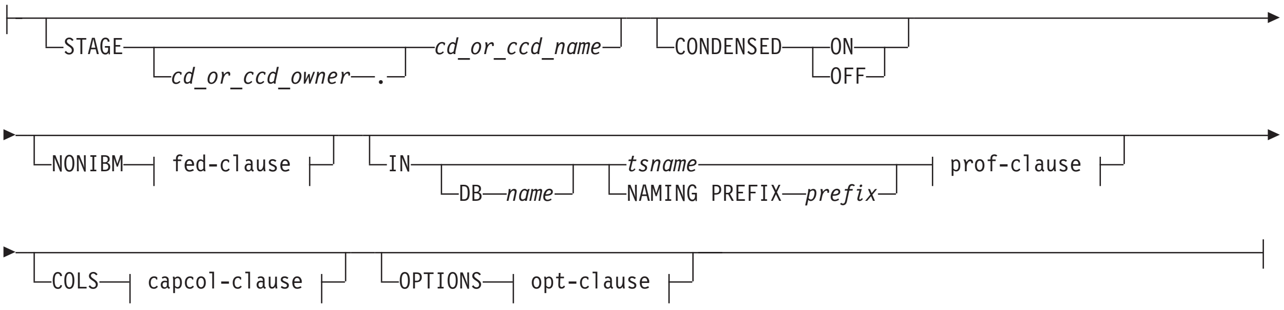
## CREATE REGISTRATION command

Use the CREATE REGISTRATION command to register a source table, view, or nickname so that it can be used for replication. You can use this command to create multiple registrations using one command.

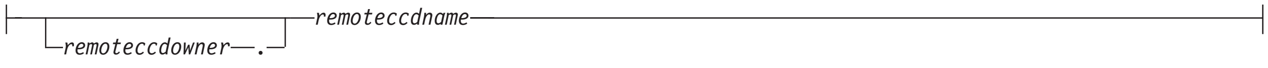
### Syntax

```
                          ┌─,──────────────────────────────────────────────────────────────────┐
                          ▼                                                                    │
►►──CREATE REGISTRATION──(──┬──────────────┬──objname──┬──────────────────────────────────────────┬─┴──)──────────►
                            └─objowner──.──┘           └─RMTJRN LIB──libname──NAME──journalname──┘

►──┬─DIFFERENTIAL REFRESH──┤ diff-ref-clause ├─┬─────────────────────────────────────────────────────►◄
   └─FULL REFRESH ONLY─────────────────────────┘
```

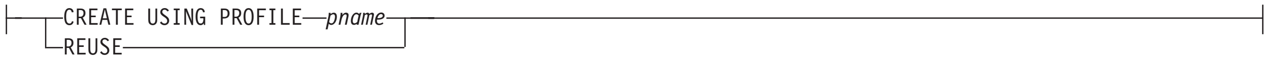
**diff-ref-clause:**



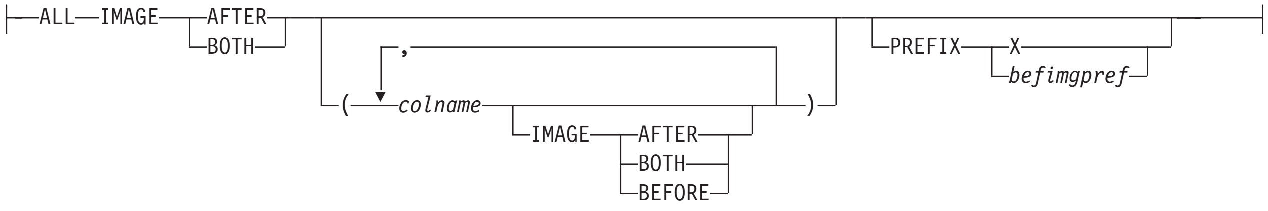
**fed-clause:**



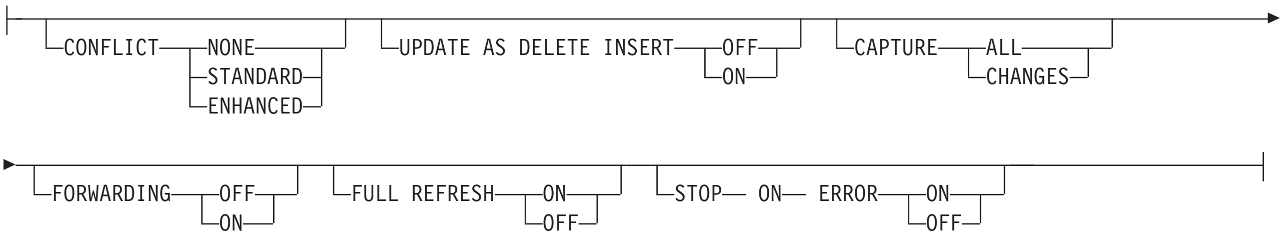
**prof-clause:**



**capcol-clause:**



**opt-clause:**



## Parameters

*objowner*
: Specifies the owner of the source object (table, view, or nickname) to register. You can specify multiple objects.

*objname*
: Specifies the name of the source object (table, view, or nickname) to register. You can specify multiple objects.

**LIB** *libname*
Specifies the AS/400 library name.

**NAME** *journalname*
Specifies the AS/400 journal name.

**DIFFERENTIAL REFRESH**
Specify to update the target table periodically as the source object changes.

**FULL REFRESH ONLY**
Specify to do a full refresh only, instead of applying changes.

diff-ref-clause:

**STAGE** *cd_or_ccd_owner***.***cd_or_ccd_name*
Specifies the CD table owner and name. For non-DB2 sources, specifies the CCD table owner and name.

**Note**: If the object name is a view, then there can be multiple CD table names. Do not include this parameter because the command will generate view names for you. In this case, the ASNCLP program ignores any values you specify for this parameter.

**CONDENSED**

**ON**
Specify to retain the most current data value.

**OFF**
Specify to retain a history of data.

**Note:**

- Must be set to **OFF** if the source is non-DB2.
- This parameter is ignored for a CD table; CD tables are always noncondensed.

**NONIBM**
Specifies the non-DB2 options.

*remoteccdowner***.**
Specifies the CCD table owner in the non-DB2 database.

*remoteccdname*
Specifies the CCD table name in the non-DB2 database.

**IN** Specifies the CD or CCD table space. If you do not specify the **IN** clause, the command uses the DB2 defaults for table spaces.

**DB** *name*
Specifies the name of an existing database where the CD or CCD table will be created. You must specify the database name, even if you set the database name in the profile.

*tsname*
Specifies the table space name. For z/OS, the name includes the database name (for example, "dbname.tsname"). You can specify a heterogeneous segment or table space name, but it must already exist.

**NAMING PREFIX** *prefix*
Specifies a naming prefix for the control tables.

prof-clause:

**CREATE USING PROFILE** *pname*
Specify to create the registration using a profile.

**REUSE**
Specify to reuse the current table space or index. You must issue the **CREATE USING PROFILE** parameter before you can use the **REUSE** parameter. When you specify the **REUSE** parameter, the ASNCLP program checks if the table space or index exists for the *tsname*:

- If the table space or index exists, the ASNCLP program resets the flags and passes the fully populated object to the API.
- If the table space or index does not exist, the ASNCLP program displays a syntax error saying that the **CREATE USING PROFILE** parameter is expected.

**COLS**
Specifies the columns that you want to register.

**Note:** This command only applies if the object is table. If the object is view, you can not have a registered sub set of columns.

capcol-clause:

**ALL**
Specifies that you want to register all columns. This is the default.

**IMAGE**

**AFTER**
Specify to register only after-image columns.

**BOTH**
Specify to register both after-image and before-image columns.

*colname*
Specifies a list of the columns that you want to register.

**PREFIX**

- If you specify **IMAGE AFTER**, the prefix will be null and the source will not allow any before-image columns.
- If you specify **IMAGE BOTH** or **IMAGE BEFORE** and do not specify **PREFIX**, a default value of `X` is used as a prefix for the before images. If you specify a **PREFIX**, that value is used.

You cannot alter an existing before-image prefix using the ALTER REGISTRATION ROW command. However, you can add that prefix to a new before-image column. If the existing before-image prefix is null and you want to add a before-image column to the existing registration, you can specify the before-image prefix using the ALTER REGISTRATION ADD command. If you do not specify the prefix, the ASNCLP program sets it to a default value of `X`.

opt-clause:

**CONFLICT**
Specifies the conflict-detection level.

**NONE**
No conflict detection. Conflicting updates between the master table and the replica table will not be detected. This option is not recommended for update-anywhere replication. This is the default.

**STANDARD**

Moderate conflict detection. During each Apply cycle, the Apply program compares the key values in the master's CD table with those in the replica's CD table. If the same key value exists in both CD tables, it is a conflict. In case of a conflict, the Apply program will undo the transaction that was previously committed at the replica by reading from the replica's CD table and keeping only the changes that originated at the master.

**ENHANCED**

Conflict detection that provides the best data integrity among the master and its replicas. As with standard detection, the Apply program compares the key values in the master's CD table with those in the replica's CD table during each Apply cycle. If the same key value exists in both CD tables, it is a conflict. However, with enhanced detection, the Apply program waits for all inflight transactions to commit before checking for conflicts. To ensure that it catches all inflight transactions, the Apply program locks all target tables in the subscription set against further transactions and begins conflict detection after all changes are captured in the CD table. In case of a conflict, the Apply program will undo the transaction that was previously committed at the replica by reading from the replica's CD table and keeping only the changes that originated at the master.

**UPDATE AS DELETE INSERT**

**ON**

Specify to capture updates as delete-insert pairs.

**OFF**

Specify to capture updates as updates. This is the default.

**CAPTURE**

**ALL**

Specify to capture everything. This is the default.

**CHANGES**

Specify to capture only changes.

**FORWARDING**

**OFF**

Specify not to forward changes from this source. This is the default.

**ON**

Specify to forward changes from this source.

**FULL REFRESH**

**ON**

Specify to allow full refreshes for this source. This is the default.

**OFF**

Specify not to allow full refreshes for this source.

**STOP ON ERROR**

**ON**

Specify not to stop the Capture program if it detects an error for this registration. This is the default.

**OFF**

Specify to stop the Capture program if it detects an error for this registration.

## Usage notes

If multiple objects are registered at one time:

- The CD table or CCD table object owner and name clause is ignored; the command generates its own defaults.
- The table space specifications apply to all registrations.
- The OPTIONS values are common across all registrations.
- If the source object is view, the command decides whether the source can be registered as differential or full refresh and the user input will be ignored.

## Example 1

To create a registration for DB2ADMIN.STAFF that only does full refreshes:

```
CREATE REGISTRATION (DB2ADMIN.STAFF) FULL REFRESH ONLY
```

## Example 2

To create a registration for DB2ADMIN.STAFF that updates the target table as the source objects change, registers after-image columns C002 and C003, and registers both after-image and before-image columns C000 and C001:

```
CREATE REGISTRATION (DB2ADMIN.STAFF) DIFFERENTIAL REFRESH STAGE CDSTAFF
COLS (C000 IMAGE BOTH, C001 IMAGE BOTH, C002 IMAGE AFTER, C003 IMAGE AFTER) PREFIX X
```

# CREATE STMT command

Use the CREATE STMT command to create a statement for an existing subscription set. This command lets you add a SQL statement or a stored procedure that Apply will process to the subscription set.

## Syntax

```
►►──CREATE STMT──IN──SETNAME──setname──APPLYQUAL──applyqual──┬─────────────────┬──────────────►
                                                             └─SETTYPE──┬─R─┬──┘
                                                                        ├─U─┤
                                                                        └─P─┘

►──┬──────────────────┬──┬─────────────────────┬──EXECUTE──┬─AT SOURCE────────┬──────────────►
   ├─SQL──"statement"─┤  └─NUMBER──stmtnumber──┘           ├─AFTER AT TARGET──┤
   └─PROC──"procname"─┘                                    └─BEFORE AT TARGET─┘

►──┬─────────────────────┬──────────────────────────────────────────────────────────────────►◄
   └─SQLSTATES──"states"─┘
```

## Parameters

**SETNAME** *setname*
Specifies the subscription-set name.

**APPLYQUAL** *applyqual*
Specifies the Apply qualifier for the subscription set.

**SETTYPE**
Specifies the subscription-set type.

**R** Specifies a read-only set. This is the default.

**U** Specifies an update-anywhere set.

**P** Specifies a peer-to-peer set.

**SQL** *"statement"*
Specifies an SQL statement. The double quotation marks are required.

**PROC** *"procname"*
Specifies a stored procedure name. The double quotation marks are required.

**NUMBER** *stmtnumber*
Specifies the statement number to assign to this SQL statement or stored procedure. The default is (the value for the STMT_NUMBER column in the IBMSNAP_SUBS_STMT table) + 1.

**EXECUTE**
Specifies where and when to execute the statement or procedure.

**AT SOURCE**
Specify to execute the statement or procedure at the source server.

**AFTER AT TARGET**
Specify to execute the statement or procedure at the target server after the Apply program processes the subscription set.

**BEFORE AT TARGET**
Specify to execute the statement or procedure at the target server before the Apply program processes the subscription set.

**SQLSTATES** *"states"*
Specifies the SQL states that are accepted as normal during execution of the statement or procedure. The double quotation marks are required.

### Example 1

To create a statement for the SET00 subscription set that executes an SQL statement at the source:

```
CREATE STMT IN SETNAME SET00 APPLYQUAL AQ00 SQL "statement" EXECUTE AT SOURCE
```

### Example 2

To create a statement for the SET00 subscription set that executes the stored procedure at the target server before the Apply program processes the subscription set:

```
CREATE STMT IN SETNAME SET00 APPLYQUAL AQ00 PROC "procname" EXECUTE BEFORE AT TARGET
```

## CREATE SUBSCRIPTION SET command

Use the CREATE SUBSCRIPTION SET command to create an empty subscription set.

### Syntax

```
►►──CREATE SUBSCRIPTION SET──SETNAME──setname──APPLYQUAL──applyqual──┬─────────────────────────┬──────────────►
                                                                      └─ACTIVATE───┬─NO───┬──┘
                                                                                   ├─YES──┤
                                                                                   └─ONCE─┘
```

```
►─┬───────────────┬──┬──────────────────────────────────────────────────────┬──────────────►
  └─SETTYPE──┬─R─┬┘  └─TIMING──┬─EVENT──eventname──────────────────────────┬─┘
             ├─U─┤             ├─INTERVAL──minutes─────────────────────────┤
             └─P─┘             ├─BOTH──EVENT──eventname── INTERVAL──minutes─┤
                               └─CONTINUOUS────────────────────────────────┘

►─┬────────────────────────────────────────────────────────┬─┬──────────────────────────────────┬──►
  └─START DATE──"yyyy-mm-dd" ──TIME──"hh:mm:ss.ffffff" ─┘ └─NONIBM SOURCE SERVER──srvrname─┘

►─┬───────────────────┬──────────────────────────────────────────────────────────────────────►◄
  └─COMMIT COUNT──n─┘
```

## Parameters

**SETNAME** *setname*
Specifies the subscription-set name.

**APPLYQUAL** *applyqual*
Specifies the Apply qualifier for the subscription set.

**ACTIVATE**
Specifies whether to activate the subscription set.

**NO**
Specify to not activate the subscription set. This is the default.

**YES**
Specify to activate the subscription set.

**ONCE**
Specify to activate the subscription set for one Apply cycle, then deactivate the subscription set.

**SETTYPE**
Specifies the subscription-set type.

**R** Specifies a read-only set. This is the default.

**U** Specifies an update-anywhere set.

**P** Specifies a peer-to-peer set.

**TIMING**
Specifies the timing for the subscription set.

**EVENT** *eventname*
Specifies the event that when posted to the IBMSNAP_SUBS_EVENT table, causes the Apply program to process the subscription set.

**INTERVAL** *minutes*
Specifies the interval for the Apply program to process the subscription set. The default interval is 20 minutes.

**BOTH**
Specifies that this subscription set uses both event and interval timing.

**CONTINUOUS**
Specifies that the Apply program should process the subscription set continuously. This keyword is equivalent to specifying an interval of zero minutes.

**START DATE** *"yyyy-mm-dd"*
Specifies the date to activate the subscription set. The double quotation marks are required.

**TIME** *"hh:mm:ss.ffffff"*
Specifies the time to activate the subscription set. The double quotation marks are required.

**NONIBM SOURCE SERVER** *srvrname*
Specifies the name of the non-DB2 source server.

**COMMIT COUNT** *n*
Specifies the number of transactions that the Apply program should process before issuing a SQL COMMIT statement for the subscription set. The default value is NULL, which means that the Apply program issues just one COMMIT statement for the subscription set after it processes the entire set. Do not specify the **COMMIT COUNT** option if you want the default behavior.

### Usage notes

- This command can create only empty subscription sets, whereas the Replication Center allows you to create empty subscription sets or add members to the set while creating it.
- A Capture schema is required, even though the set is empty.
- Because the set is empty, the default for activating the set is **NO**.
- To add a member to an existing subscription set, use the CREATE MEMBER command.
- To add a statement to the set, issue the CREATE SUBSCRIPTION SET STMTS command.

### Example 1

To create a subscription set SET00 that activates on 2006-11-22 at 09:00:00.000000:

```
CREATE SUBSCRIPTION SET SETNAME SET00 APPLYQUAL AQ00 ACTIVATE YES TIMING INTERVAL 1
START DATE "2006-11-22" TIME "09:00:00.000000"
```

### Example 2

To create a subscription set SET00 that activates for one Apply cycle on 2006-11-22 at 09:00:00.000000:

```
CREATE SUBSCRIPTION SET SETNAME SET00 APPLYQUAL AQ00 ACTIVATE ONCE TIMING CONTINUOUS
START DATE "2006-11-22" TIME "09:00:00.000000" NONIBM SOURCE SERVER SAMPLE
```

## DROP CONTROL TABLES ON command

Use the DROP CONTROL TABLES ON command to drop a set of Capture, Apply, or Monitor control tables.

This command does not drop replication control tables on an OS/400 system.

### Syntax

```
►►──DROP CONTROL TABLES ON──┬─CAPTURE SERVER─────────┬──┬──────────────────────────┬──────────────────►
                            ├─APPLY CONTROL SERVER───┤  └─ARCHLEVEL──┬─0801─┬─┘
                            └─MONITOR CONTROL SERVER─┘               ├─0201─┤
                                                                     └─0805─┘

►──┬────────────────────┬──────────────────────────────────────────────────────────────────────────►◄
   └─NONIBM SCHEMA──name─┘
```

## Parameters

**CAPTURE SERVER**
Specify to drop the Capture control tables.

**APPLY CONTROL SERVER**
Specify to drop the Apply control tables.

**MONITOR CONTROL SERVER**
Specify to drop the Monitor control tables.

**ARCHLEVEL**
Specifies the replication architecture level for the control tables that you want to drop.

**0801**
Specifies the Version 8 architecture level. For the Monitor control tables, the architecture level is always 0801.

**z/OS:** 0801 specifies control tables created on a z/OS system running in version 8 compatibility mode.

**0201**
Specifies the architecture level for Version 5, Version 6, or Version 7.

**0805**
Specifies the control tables created on a z/OS system running in new-function mode

**NONIBM SCHEMA** *name*
Specifies the remote schema name to use for heterogeneous replication. The following non-DB2 data sources are supported:

- Oracle
- Sybase
- Microsoft SQL Server
- Informix®
- Teredata

## Usage notes

- The SET DROP command affects this command.
- This command drops the table spaces that the control tables are in if they do not contain any other objects.
- **Recommendation**: If the pre-Version 8 tables contain any data, migrate them instead of dropping them.

## Example 1

To drop the Version 5 Capture control tables:

```
DROP CONTROL TABLES ON CAPTURE SERVER ARCHLEVEL 0201
```

## Example 2

To drop the Version 8 Apply control tables:

```
DROP CONTROL TABLES ON APPLY CONTROL SERVER ARCHLEVEL 0801
```

# DROP MEMBER command

Use the DROP MEMBER command to drop a member from an existing subscription set.

## Syntax

```
►►──DROP MEMBER──FROM──SETNAME──setname──APPLYQUAL──applyqual──SOURCE──┬───────────────┬─objname─────────►
                                                                        └─objowner──.──┘

►─TARGET──┬───────────────┬─objname───────────────────────────────────────────────────────────────────►◄
          └─objowner──.──┘
```

## Parameters

**SETNAME** *setname*
Specifies the subscription-set name.

**APPLYQUAL** *applyqual*
Specifies the Apply qualifier for the subscription set.

**SOURCE** *objowner.objname*
Specifies the source object's owner and name.

**TARGET** *objowner.objname*
Specifies the target object's owner and name.

## Usage notes

- For update-anywhere subscription sets, members for both replication directions (master-to-replica and replica-to-master) are dropped.
- The values specified in the SET DROP command determine whether the target table space is also dropped depends on the SET DROP command.
- Whether the target table is also dropped depends on the environment command:
  - If the target table has dependent subscription sets, it is not dropped and the autoregistration information is not deleted.
  - If there are no dependent subscription sets, the target table is dropped depending on the SET SERVER command. The autoregistration information is deleted.

## Example

To drop a member from the SET00 subscription set:

```
DROP MEMBER FROM SETNAME SET00 APPLYQUAL AQ00 SOURCE DB2ADMIN.STAFF
TARGET DB2ADMIN.TRGSTAFF;
```

# DROP REGISTRATION command

Use the DROP REGISTRATION command to drop one or more registrations.

## Syntax

```
                     ┌─,─────────────────────────┐
                     ▼                           │
►►──DROP REGISTRATION──(──┬────────────┬─objname──┴──)──────────────────────────────────────────────►◄
                          └─objowner──┘
```

## Parameters

*objowner*.
Specifies the owner of the source object (table, view, or nickname) to drop.

*objname*
Specifies the name of the source object (table, view, or nickname) to drop.

## Usage notes

- The SET DROP command affects whether associated table spaces of the CD tables will be dropped when the objects are dropped.
- If the object is a view, only the CD views are dropped.
- For nicknames, this command does not drop the associated table spaces.

## Example 1

To drop the registration for DB2ADMIN.STAFF:

```
DROP REGISTRATION (DB2ADMIN.STAFF)
```

## Example 2

To drop the registration for DB2ADMIN.STAFF and DB2ADMIN.EMPLOYEE:

```
DROP REGISTRATION (DB2ADMIN.STAFF, DB2ADMIN.EMPLOYEE)
```

# DROP STMT command

Use the DROP STMT command to drop SQL statements from an existing subscription set.

## Syntax

```
►►──DROP STMT──FROM──SETNAME──setname──APPLYQUAL──applyqual──┬─────────────────┬──────────────────►
                                                             └─SETTYPE──┬─R─┬─┘
                                                                        ├─U─┤
                                                                        └─P─┘

            ┌─,───────┐
            ▼         │
►─NUMBER──(───stmtnumber──┴──)──────────────────────────────────────────────────────────────────►◄
```

## Parameters

**SETNAME** *setname*
Specifies the subscription-set name.

**APPLYQUAL** *applyqual*
Specifies the Apply qualifier for the subscription set.

**SETTYPE**
Specifies the subscription-set type.

**R** Specifies a read-only set. This is the default.

**U** Specifies an update-anywhere set.

**P** Specifies a peer-to-peer set.

**NUMBER** *stmtnumber*
Specifies the statement number to drop. You can specify multiple numbers using commas and parentheses.

### Usage notes

- You cannot drop statements that are added to a subscription set for heterogeneous replication. These statements have the value G for the BEFORE_OR_AFTER column of the IBMSNAP_SUBS_STMTS table.

### Example

To drop a statement from the subscription set SET00:

```
DROP STMT FROM SETNAME SET00 APPLYQUAL AQ00 NUMBER (5)
```

## DROP SUBSCRIPTION SET command

Use the DROP SUBSCRIPTION SET command to drop an existing subscription set for a specified Apply qualifier.

### Syntax

```
►►──DROP SUBSCRIPTION SET──SETNAME──setname──APPLYQUAL──applyqual──────────────────────────────────►◄
```

### Parameters

**SETNAME** *setname*
Specifies the subscription-set name.

**APPLYQUAL** *applyqual*
Specifies the Apply qualifier for the subscription set.

### Usage notes

- If the subscription set has members, all members and statements will be dropped.
- See the "DROP MEMBER command" on page 34 for the rules that affect the dropped objects.

### Example

To drop the subscription set SET00:

```
DROP SUBSCRIPTION SET SETNAME SET00 APPLYQUAL AQ00
```

## OFFLINE LOAD command

Use the OFFLINE LOAD command to control a manual full refresh for offline load procedures.

You must first run the OFFLINE LOAD BEFORE command to prepare for an offline load. This will generate the scripts to deactivate the relevant subscription sets. After you have completed your offline load, you then need to run the OFFLINE LOAD AFTER command to reactivate the subscription set and reset the IBMSNAP_PRUNCNTL and IBMSNAP_SIGNAL tables

## Syntax

```
►►──OFFLINE LOAD──┬─BEFORE─┬──SETNAME──setname──APPLYQUAL──applyqual─────────────────────────►◄
                  └─AFTER──┘
```

## Parameters

**BEFORE**
Specifies that you want to modify your replication environment in preparation for running an offline load for the target tables.

**AFTER**
Specifies that you want to modify your replication environment after running an offline load for the target tables.

**SETNAME** *setname*
Specifies the subscription-set name.

**APPLYQUAL** *applyqual*
Specifies the Apply qualifier for the subscription set.

## Example 1

To run the OFFLINE LOAD BEFORE command and to generate the scripts to deactivate the subscription set SET00:

```
OFFLINE LOAD BEFORE SETNAME SET00 APPLYQUAL AQ00
```

## Example 2

To run the OFFLINE LOAD AFTER command and to reactivate the subscription set SET00 and to reset the IBMSNAP_PRUNCNTL SET and IBMSNAP_SIGNAL tables:

```
OFFLINE LOAD AFTER SETNAME SET00 APPLYQUAL AQ00
```

# PROMOTE REGISTRATION command

Use the PROMOTE REGISTRATION command to promote existing registrations.

## Syntax

```
                         ┌─,─────────────────────────┐
                         ▼                           │
►►──PROMOTE REGISTRATION──(──┬──────────────┬─objname─┴──)──┬──────────────────────┬──────────────►◄
                             └─objowner──.─┘                └─USING──┤ new-clause ├─┘
```

### new-clause:

```
├──┬─────────────────────┬──┬─────────────────────────────┬──┬─TABLE──┤ tbl-clause ├─┬──────────────┤
   └─SOURCE DB──aliasname─┘  └─CAPTURE SCHEMA──schemaname─┘  └─VIEW──┤ view-clause ├─┘
```

### tbl-clause:

```
├──┬────────────────────┬──┬──────────────────────────────────────────┬─────────────────────────────┤
   └─CD SCHEMA──cdschema─┘  └─CREATE SOURCE WITH SCHEMA──tableschema─┘
```

**view-clause:**

```
├──┬────────────────────────────────────────────────────────────────┬──────────────►
   └─CD SCHEMA FOR──┬──────────────────┬──┬──────────────────────────┬─┘
                    └─VIEW──viewschema─┘  └─SOURCE TABLE──tableschema─┘

►──┬──────────────────────────────────────────────────────────────────────────┬──────┤
   └─CREATE SOURCE VIEW──┬───────────────────────────────┬──USING SCHEMA──viewschema─┘
                         └─WITH UNREGISTERED BASE TABLES─┘
```

## Parameters

*objowner***.**
Specifies the owner of the source object (table, view, or nickname) to promote. You can specify multiple objects.

*objname*
Specifies the name of the source object (table, view, or nickname) to promote. You can specify multiple objects.

new-clause:

**SOURCE DB** *aliasname*
Specifies the new source database alias for the promoted object. This database is where you will run the generated script.

**CAPTURE SCHEMA** *schemaname*
Specifies the Capture schema to use when promoting a registration.

**TABLE**
Specifies a CD table.

**VIEW**
Specifies a CD view.

tbl-clause:

**CD SCHEMA** *cdschema*
Specifies the new CD-table schema name for the promoted object.

**CREATE SOURCE WITH SCHEMA** *tableschema*
Specifies the new source-table schema name to use when promoting the underlying table.

view-clause:

**CD SCHEMA FOR**

**VIEW** *viewschema*
Specifies the new CD-view schema name for the promoted object.

**SOURCE TABLE** *tableschema*
Specifies the new CD-table schema name for the promoted object.

**CREATE SOURCE VIEW**
Specify to promote the view on the new source.

**WITH UNREGISTERED BASE TABLES**
Specify to promote underlying base tables that are not registered.

**USING SCHEMA** *viewschema*
Specifies the new source-view schema name to use when promoting the underlying view and the unregistered base tables.

### Usage notes

- If you do not specify the **USING** new-clause parameter, this command uses the existing values for the object.
- This command uses the following rules when generating the SQL scripts:
  - All views and tables referenced by the registered views exist on the new server.
  - All registered source tables referenced by the registered views are already promoted to the new server.
  - The **WITH UNREGISTERED BASE TABLES** clause promotes only the unregistered base tables of the view. It does not promote the registered base tables. You must promote the registered base tables separately before promoting the registered view.
  - The same new schema name will be used for both the underlying base tables and the view.
- The command does not support a new source CD schema when promoting subscription sets; do not change the CD schema when promoting registrations.

### Example 1

To promote the registration for DB2ADMIN.STAFF using the SAMPLE database and ASN1 schema:

```
PROMOTE REGISTRATION (DB2ADMIN.STAFF) USING SOURCE DB SAMPLE TABLE CD SCHEMA ASN1
```

### Example 2

To promote the registration for DB2ADMIN.STAFF and to name the new CD-table schema STAFF:

```
PROMOTE REGISTRATION (DB2ADMIN.STAFF) USING VIEW CD SCHEMA FOR SOURCE TABLE STAFF
```

## PROMOTE SUBSCRIPTION SET command

Use the PROMOTE SUBSCRIPTION SET command to recreate an existing subscription set in another replication environment.

### Syntax

```
►►──PROMOTE SUBSCRIPTION SET──SETNAME──setname──APPLYQUAL──applyqual──┬──────────────────────────┬────────►◄
                                                                      └─USING──┤ new-clause ├────┘
```

**new-clause::**

```
├──┬────────────────────────────────────────────────────────────────────────────┬──────────────────────►
   └─CAPTURE SCHEMA FOR──┬──────────────────────┬──┬────────────────────────┬───┘
                         └─SOURCE──sourcename───┘  └─REPLICA──replicaname───┘
```

```
►─┬──────────────────────────────────────────────────────────────────────────────────────┬─►
  └─DB FOR──┬─────────────────────┬──┬─────────────────────┬──┬───────────────────────┬─┘
            └─SOURCE──sourcealias─┘  └─TARGET──targetalias─┘  └─CONTROL──controlalias─┘

►─┬────────────────────────┬─┬───────────────────┬─┬─────────────────────────────┬─►
  └─APPLYQUAL──newapplyqual─┘ └─SETNAME──newsetname─┘ └─SOURCE SCHEMA──newsourcename─┘

►─┬──────────────────────────────────────────────────────────────────┬─────────────────┤
  └─TARGET──┬──────────────────────┬─┬─────────────────────────┬─┘
            └─SCHEMA──newtargetname─┘ └─CD SCHEMA──newcdschema─┘
```

## Parameters

**SETNAME** *setname*
Specifies the subscription-set name.

**APPLYQUAL** *applyqual*
Specifies the Apply qualifier for the subscription set.

**USING**
Specifies the information for the promoted subscription set.

new-clause:

**CAPTURE SCHEMA FOR**
Specifies the new Capture schema.

**SOURCE** *sourcename*
Specifies the new Capture schema at the source.

**REPLICA** *replicaname*
Specifies the new Capture schema at the source for a replica.

**DB FOR**
Specifies the new database alias.

**SOURCE** *sourcealias*
Specifies the new source database alias for the promoted object. This database is where you will run the generated script.

**TARGET** *targetalias*
Specifies the new target database alias for the promoted object. This database is where you will run the generated script.

**CONTROL** *controlalias*
Specifies the new Apply control database alias for the promoted object. This database is where you will run the generated script.

**APPLYQUAL** *newapplyqual*
Specifies the new Apply qualifier.

**SETNAME** *newsetname*
Specifies the new subscription-set name.

**SOURCE SCHEMA** *newsourcename*
Specifies the new source schema name.

**TARGET**
Specifies the schemas for the target.

**SCHEMA** *newtargetname*
Specifies the new target schema name.

**CD SCHEMA** *newcdschema*
Specifies the new target-CD schema name.

### Usage notes

- If you do not specify a USING clause, this command uses the existing values.
- The command does not support a new source CD schema when promoting subscription sets, so you should not change the CD schema when you promote registrations.

### Example

To promote an existing subscription set SET00:

```
PROMOTE SUBSCRIPTION SET SETNAME SET00 APPLYQUAL AQ00 USING CAPTURE SCHEMA
FOR SOURCE ASN2 SETNAME SET01 SOURCE SCHEMA SAMPLE1 TARGET SCHEMA TARGET1
CD SCHEMA ASN3
```

## SET CAPTURE SCHEMA command (SQL replication)

Use the SET CAPTURE SCHEMA command to set a source and target Capture schema for all task commands. The default Capture schema is ASN. You can use this command to change the default.

This command allows you to omit the Capture schema settings in the task commands.

### Syntax

```
►►──SET CAPTURE SCHEMA──┬─SOURCE─┬──┬─TO──┬─DEFAULT─┬─┬───────────────────────────────────►◄
                        └─TARGET─┘  │     └─NULLS───┘ │
                                    └─capschema───────┘
```

### Parameters

**SOURCE**
Specifies the Capture schema at the source. The schema can be any valid DB2 schema name.

**TARGET**
Specifies the Capture schema at the target (used for autoregistration of replica or CCD target tables). The schema can be any valid DB2 schema name.

**DEFAULT**
Specify to set the Capture schema to ASN and to reset any previous SET CAPTURE SCHEMA commands.

**NULLS**
Specify to set the Q Capture schema to NULL.

*capschema*
Specifies the name of a schema that generates the Capture control tables.

### Example 1

To set the Capture schema to ASN by default:

```
SET CAPTURE SCHEMA SOURCE TO DEFAULT
```

### Example 2

To set the Capture schema to ASN1:

```
SET CAPTURE SCHEMA SOURCE ASN1
```

## SET DROP command (SQL replication)

Use the SET DROP command to determine whether to drop the table space when you drop the database object (replication control tables, registrations, or subscription-set members).

**Note:** The drop options affect multiple objects (that is, they are at the environment-command level), whereas the create options are at an object level (that is, they are at the task-command level).

### Syntax

```
►►──SET DROP──TARGET──┬─ALWAYS─┬──────────────────────────────────────────────►◄
                      └─NEVER──┘


►►──SET DROP──┬─CD─────────────┬──TABLESPACE──┬─WHEN EMPTY─┬──────────────────►◄
              ├─CCD────────────┤              └─NEVER──────┘
              ├─TARGET─────────┤
              └─CONTROL TABLES─┘
```

### Parameters

**TARGET**
Specifies whether you want to drop the target tables with the subscription.

**ALWAYS**
Always drop the target table.

**NEVER**
Never drop the target table.

**DROP**
Specifies what you want to drop with the subscription.

**CD**
Change data table

**CCD**
Consistent-change-data table

**TARGET**
Target table

**CONTROL TABLES**
Capture, Apply, or Monitor control tables

These options are relevant only for operating-system environments for which the commands create the table spaces. You can always specify the drop flag for each of these object types.

**TABLESPACE**
Specifies when to drop the table space that contains the specified object.

**WHEN EMPTY**
Drop the table space only when it is empty.

**NEVER**
Never drop the table space.

### Usage notes

The drop subscription-set member command decides whether to drop an autoregistered target table. If the autoregistration has dependent subscriptions, the command does not drop the target table and does not drop the registration; otherwise, the registration and the target table are dropped only if the SET DROP TARGET ALWAYS command allows it.

### Example 1

To always drop the target table's table space when the subscription is dropped:

```
SET DROP TARGET ALWAYS
```

### Example 2

To drop the CCD table space when it is empty:

```
SET DROP CCD TABLESPACE WHEN EMPTY
```

## SET LOG command

Use the SET LOG command to define the log file for the ASNCLP session. The log file contains informational warning and error messages.

### Syntax

```
►►—SET LOG—"logfilename"——————————————————————————————————————————————►◄
```

### Parameters

*"logfilename"*
Specifies the output log file name. The default file name is replmsg.log.

### Usage notes

- If the files already exist, the ASNCLP program will append to them.
- The double quotation marks in the command syntax are required.

### Example

To name the output log file cnsrc.err:

```
SET LOG "cnsrc.err"
```

## SET OUTPUT command (SQL replication)

Use the SET OUTPUT command to define output files for the ASNCLP session. The output files contain the SQL statements needed to set up replication.

### Syntax

```
►►──SET OUTPUT──┬──────────────────────────────┬──┬───────────────────────────────┬──────────────────►
                └─CAPTURE SCRIPT──"capfname"──┘  └─CONTROL SCRIPT──"cntlfname"──┘

►─┬────────────────────────────┬──┬─────────────────────────────┬─────────────────────────────────►◄
  └─TARGET SCRIPT──"trgfname"─┘  └─MONITOR SCRIPT──"monfname"─┘
```

### Parameters

**CAPTURE SCRIPT** *"capfname"*
Specifies the output file name for SQL scripts that run at the Capture server. The default file name is replcap.sql.

**CONTROL SCRIPT** *"cntlfname"*
Specifies the output file name for SQL scripts that run at the Apply control server. The default file name is replctl.sql.

**TARGET SCRIPT** *"trgfname"*
Specifies the output file name for SQL scripts that run at the target server. The default file name is repltrg.sql.

**MONITOR SCRIPT** *"monfname"*
Specifies the output file name for scripts that run at the Monitor control server. The default file name is replmonitor.sql.

### Usage notes

- If you do not need an output file, run the SET OUTPUT command and specify "" for the file name.
- If a script already exists, the new script appends to the current script.
- The double quotation marks in the command syntax are required.

### Example 1

To name the output Apply control script file control.sql:

```
SET OUTPUT CONTROL SCRIPT "control.sql"
```

### Example 2

To name the output monitor script file monitor.sql:

```
SET OUTPUT MONITOR SCRIPT "monitor.sql"
```

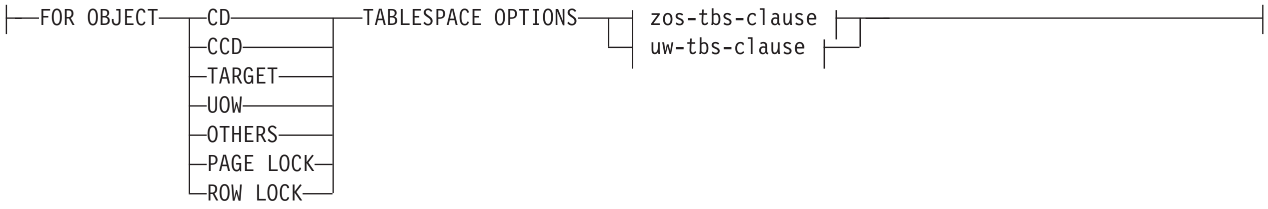
## SET PROFILE command (SQL replication)

Use the SET PROFILE command to customize rules for creating table space objects. After you issue a SET PROFILE command, all subsequent task commands inherit the table space DDL specifications defined by the command. You can associate a profile with a task command by specifying the profile's name in the task command.

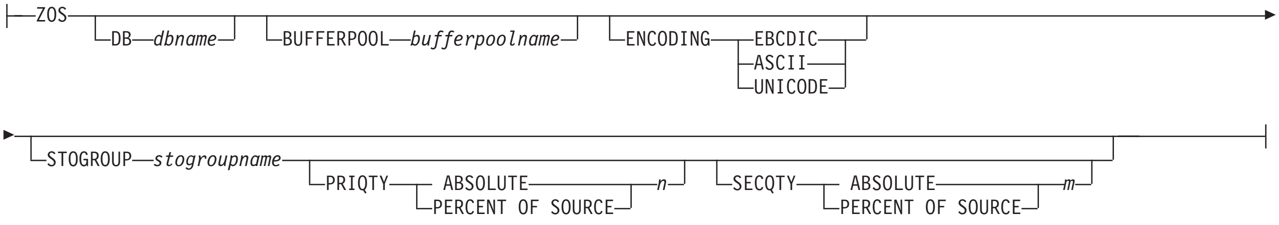
### Syntax

```
►►──SET PROFILE──profilename──┬─┤ prof-clause ├─┬──────────────────────────────────────────────►◄
                              └─UNDO────────────┘
```

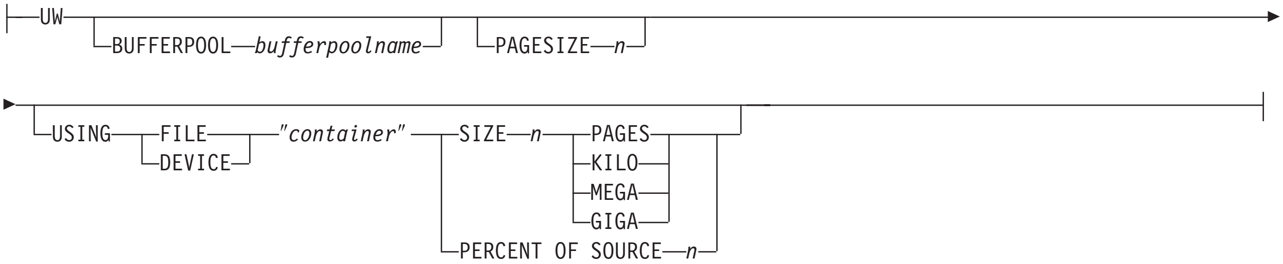
**prof-clause:**



**zos-tbs-clause:**



**uw-tbs-clause:**



## Parameters

**PROFILE** *profilename*
Specifies the profile name.

**UNDO**
Specify to undo a specific profile.

prof-clause:

**FOR OBJECT**
Specify to set an object for the table space options:

**CD**
Change data table

**CCD**
Consistent change data table

**TARGET**
Target table

**UOW**
Unit-of-work table

**OTHERS**
All other control tables, except the UOW table

**PAGE LOCK**
z/OS: All tables that follow the page locking mechanism

**ROW LOCK**
z/OS: All tables that follow the row locking mechanism

**TABLESPACE OPTIONS**
Specify to set the table space options. You can specify table space options for z/OS or Linux, UNIX, and Windows.

**z/OS:** No support for table space lock size because the replication API infers the correct value in most cases.

**Linux, UNIX, and Windows:**

- The ASNCLP program supplies the MANAGED BY DATABASE clause.
- No support for LONG table spaces.
- No support for heterogeneous replication environments.

zos-tbs-clause:

**DB** *dbname*
Specifies the name of the z/OS database to connect to. This parameter does not specify the subsystem name; use the SET SERVER command to set the subsystem name to connect to.

**BUFFERPOOL** *bufferpoolname*
Specifies the buffer pool name.

**ENCODING**
Specifies the encoding scheme (EBCDIC, ASCII, or UNICODE). The default is EBCDIC.

**STOGROUP** *stogroupname*
Specifies a storage group name.

**PRIQTY**
Specify to set the minimum primary space allocation for a DB2-managed data set for a table space.

**SECQTY**
Specify to set the minimum secondary space allocation for a DB2-managed data set for a table space.

**ABSOLUTE**
Specifies an actual value in kilobytes (denoted as *n* or *m* in the syntax diagram) for space allocation. See the information about the CREATE TABLESPACE command in the *DB2 UDB for z/OS V8 SQL Reference* (SC18-7426-00) for more details.

**PERCENT OF SOURCE**
Specifies the percentage of the source table size, as indicated by:

- **z/OS:** The column “npages” in SYSIBM.SYSTABLES
- **Linux, UNIX, and Windows:** The column “npages” in SYSSTAT.TABLES

This method will work only if the column holds the correct value for this table, which can be achieved by running the “db2 runstats on table a.b.” command or by manually updating the DB2 catalog.

uw-tbs-clause:

**BUFFERPOOL** *bufferpoolname*
Specifies the buffer pool name.

**PAGESIZE** *n*
Specifies the page size of the table space.

**Restriction:** The page size of the table space must match the page size of the buffer pool.

**FILE**
Specifies the container path string for the File. For example, for UNIX you can set the container path to /tmp/db/ts/ and for Windows, you can set the container path to D:\tmp\db\ts\.

**DEVICE**
Specifies the container path string for the device. For example, for UNIX you can set the container path to /tmp/db/ts/ and for Windows, you can set the container path to D:\tmp\db\ts\.

*"container"*
Specifies the name of the container. The ASNCLP program will generate and append the table space name to the specified path when you run a task command such as CREATE REGISTRATION. The double quotation marks in the syntax are mandatory.

**SIZE** *n*
Specifies the size of the container:

**PAGES**
Actual number of pages

**KILO**
Kilobytes

**MEGA**
Megabytes

**GIGA**
Gigabytes

## Usage notes

- You cannot specify your own naming convention for CD table names or table spaces because the task commands generate default values.
- This command is not used for heterogeneous replication environments because the task commands do not create table spaces on remote servers.
- OS/400 systems do not have table spaces that require special DDL.
- The task commands allow you to specify a table space clause so that you can use an existing table space. The task commands do not provide an index clause because indexes are always created (except in certain cases when creating target tables).
- The scope of the profile lasts only as long as the current session. Once you quit the ASNCLP session, the profile information is not saved for the next session.

## Example 1

To create a profile TBSPROFILE that sets the table space options for the target control tables:

```
SET PROFILE TBSPROFILE FOR OBJECT TARGET TABLESPACE OPTIONS UW
USING FILE "c:\TSTRG.TS" SIZE 700 PAGES
```

### Example 2

To undo the profile TBSPROFILE:

```
SET PROFILE TBSPROFILE UNDO
```

## SET RUN SCRIPT command (SQL replication)

Use the SET RUN SCRIPT command to control whether to automatically run SQL statements that are generated by each ASNCLP task command before processing the next command or to manually run them later in a DB2 command prompt.

"Using SET RUN SCRIPT options" helps you understand when to run commands immediately and when to run them later.

### Syntax

```
►►──SET RUN SCRIPT──┬─LATER──────────────────────────┬──────────────────────────────────────────►◄
                    └─NOW──STOP ON SQL ERROR──┬─ON──┬┘
                                              └─OFF─┘
```

### Parameters

**LATER**
Specify to run the SQL scripts at a later time. If you specify to run them later, you must run the generated SQL script manually at a DB2 command prompt by using the following command:

```
db2 -tvf filename
```

where *filename* is the name of the SQL script file.

**Federated sources:** Use the following command to run the script for federated (non-DB2) sources:

```
db2 -td# -vf filename
```

**NOW**
Specify to automatically execute the SQL scripts.

**STOP ON SQL ERROR**
Specifies whether to stop running the SQL scripts if an error occurs.

**ON**
Specify to stop processing the ASNCLP commands when the first SQL statement fails. All previous SQL statements related to this command will be rolled back. If the source scripts run correctly and have been committed, and the target scripts have an error, only the target scripts will be rolled back. The committed source statements will not be rolled back.

**OFF**
Specify to process the ASNCLP commands and run all of the SQL statements, regardless of errors.

### Using SET RUN SCRIPT options

Some ASNCLP CREATE commands require that one or more replication objects exist before the command can be processed. For example, you cannot create subscriptions until control tables exist.

These dependencies can influence whether you use the NOW or LATER options. In general, the following guidelines apply:

- If you want to create different types of objects in a single ASNCLP script, you are likely to need to use SET RUN SCRIPT NOW.
- If you have multiple ASNCLP scripts, each creating one or more instances of an object, you can use either NOW or LATER. If you use LATER, you are likely to need to run the generated SQL from one ASNCLP script before processing subsequent ASNCLP scripts.
- In some situations, objects of the same type require that SET RUN NOW be used.

Figure 1 shows these dependencies for SQL replication.

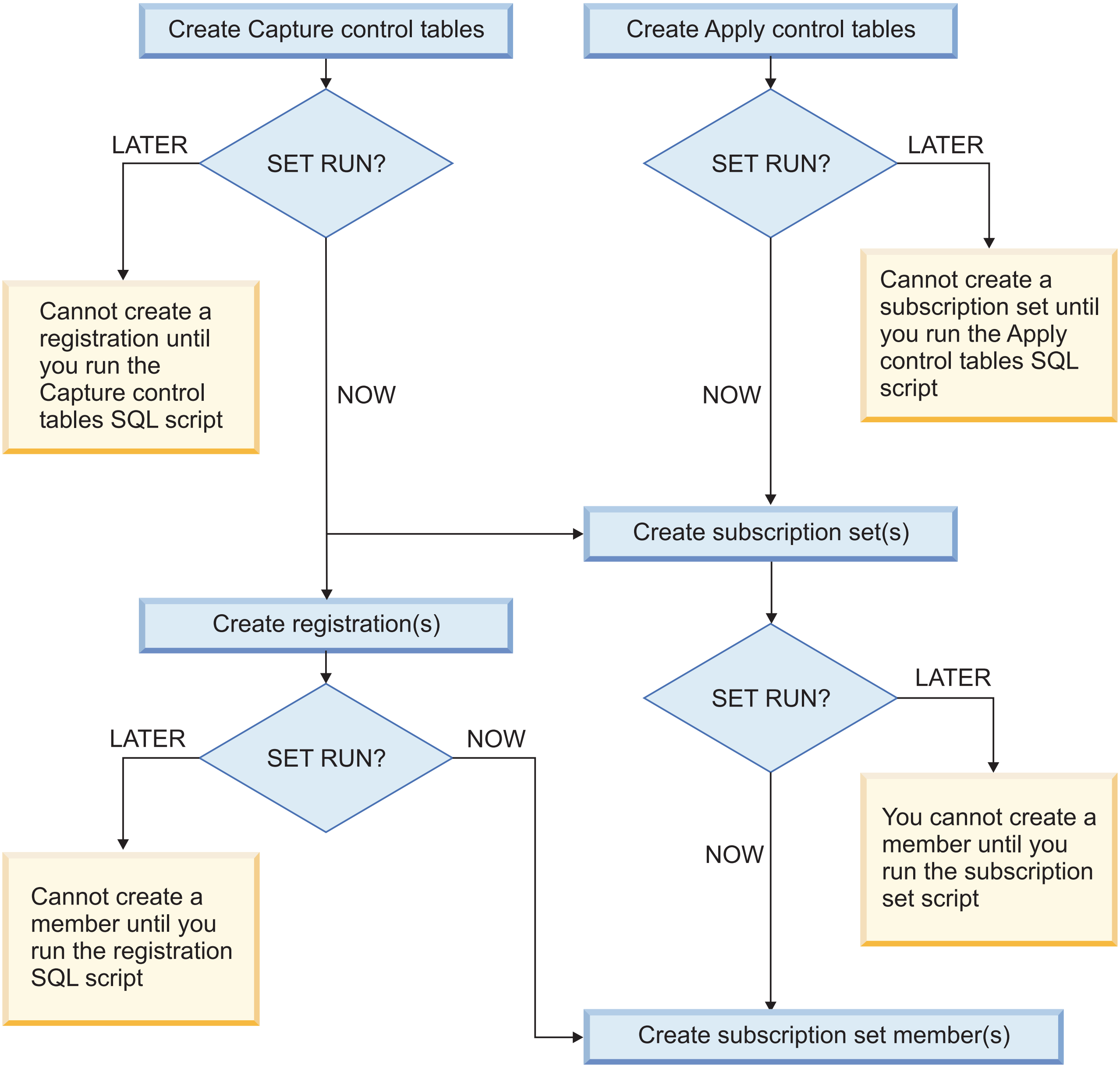

Create Capture control tables
Create Apply control tables
LATER
SET RUN?
SET RUN?
LATER
Cannot create a subscription set until you run the Apply control tables SQL script
Cannot create a registration until you run the Capture control tables SQL script
NOW
NOW
Create subscription set(s)
Create registration(s)
LATER
SET RUN?
LATER
NOW
SET RUN?
You cannot create a member until you run the subscription set script
NOW
Cannot create a member until you run the registration SQL script
Create subscription set member(s)

*Figure 1. Dependencies between ASNCLP commands for SQL replication.* This diagram shows the dependencies between ASNCLP CREATE commands that are used to set up SQL replication. It assumes all objects use the default schema of ASN.

### Usage notes

- Use SET RUN SCRIPT LATER when you want to verify the SQL scripts before you run them to create or update your replication configuration.
- Use SET RUN SCRIPT LATER if you want to create SQL script files on one operating system, but run them on another. For example, you might want to run ASNCLP on Windows and copy the SQL script files to z/OS since ASNCLP does not run on the native z/OS operating system.
- This command supports scripts to set up heterogeneous replication. Federated registration generates a script that creates a trigger on the IBMSNAP_PRUNCNTL table to prune from all CCD tables. This trigger is dropped and recreated for each registration by including all of the previous registration information along with the current registration. If each registration script is not executed before the next registration script is run, the prune control trigger in the database does not have the CCD information for the previous registration, and the trigger will be out of sync with the actual registered objects in the database. This problem can be solved by using the SET RUN SCRIPT NOW option for the input file.

### Example 1

To run the SQL scripts at a later time:

```
SET RUN SCRIPT LATER
```

### Example 2

To automatically run the SQL scripts but stop processing the ASNCLP commands if an error occurs:

```
SET RUN SCRIPT NOW STOP ON SQL ERROR ON
```

## SET SERVER command (SQL replication)

Use the SET SERVER command to specify the remote iSeries source server, Capture control server, Apply control server, or target server to use in the ASNCLP session. After you set a server name, all subsequent commands in the session will apply to this server until you change the server with this command.

The SET SERVER command is required for the following task commands:

**All control table commands**
Set the Capture control server or Apply control server before creating or dropping replication control tables.

**All registration commands (including promote)**
Set the Capture control server before running the registration commands. For iSeries, you must also set the remote source server.

**All subscription commands (including promote)**
Set the Capture control, Apply control, and target servers before running the subscription commands, unless one or more servers are not needed. For example, because the ALTER SUBSCRIPTION SET and ALTER SUBSCRIPTION SET MEMBER commands modify only control tables on the Apply control server, you do not need to set the Capture control servers for these commands. For iSeries, you must set the remote source server.

## Syntax

```
►►──SET SERVER──┬─ALL───────────┬──────────────────────────────────────────────────────────►
                ├─REMOTE SOURCE─┤
                ├─CAPTURE───────┤
                ├─CONTROL───────┤
                └─TARGET────────┘

►─ TO──┬─NULLS───────────────────────────────────────────────────────────────────────────►◄
       └─┬─DB──dbalias──────────┬──┬─────────────────────────┬─┤ other-options ├─┘
         └─DBALIAS──aliasname───┘  ├─DBNAME──dbname──────────┤
                                   └─NONIBM SERVER──remsrvr──┘
```

**other-options:**

```
├──┬──────────────────────────────┬──┬──────────────────────────────┬──────────────────────┤
   └─AS400 HOSTNAME──"hostname"───┘  └─ID──userid──PASSWORD──pwd───┘
```

## Parameters

**ALL**
Specify to set the database for all servers (remote source server, Capture control server, Apply control server, target server).

**REMOTE SOURCE**
iSeries: Specify to set the database as a remote source server.

**CAPTURE**
Specify to set the database as a Capture control server.

**CONTROL**
Specify to set the database as an Apply control server.

**TARGET**
Specify to set the database as a target server.

**NULLS**
Specify to set the server name to NULL. This option resets a previously set server name.

**DB** *dbalias*
Specifies the database alias name.

**DBALIAS** *aliasname*
Linux, UNIX, or Windows: Specifies the database alias name.

**DBNAME** *dbname*
z/OS: Specifies the database name.

**NONIBM SERVER** *remsrvr*
Capture control servers and target servers only: Specifies the remote server name for a non-DB2 source or target. This parameter is valid only for Capture control servers and target servers, not for Apply control servers.

**AS400 HOSTNAME** *"hostname"*
Specifies the OS/400 host name, typically an IP address or name.

**ID** *userid*
Specifies the user ID to use to connect to the database.

**PASSWORD** *pwd*
Specifies the password to use to connect to the database.

### Usage notes

- Use the NONIBM SERVER clause to set up replication using non-DB2 data sources and targets such as Oracle and Sybase. The environment command saves the database server information, but does not perform the actual db2 connect command. The environment command assigns a database alias to a logical replication server. The ASNCLP program attempts the connection to determine the platform and build the appropriate objects for the task commands.
- If you issue multiple environment commands, the most recent command overrides the current settings for a given remote source server, Capture control server, Apply control server, or target server. That is, you can associate only one value for each of these servers, but these values need not be the same.

### Example 1

To set all servers to the database SAMPLE:

```
SET SERVER ALL TO DB SAMPLE ID DB2ADMIN PASSWORD "passw0rd"
```

### Example 2

To set the Capture control server to the database SAMPLE:

```
SET SERVER CAPTURE TO DB SAMPLE ID DB2ADMIN PASSWORD "passw0rd"
```

## SET TRACE command

Use the SET TRACE command to enable and disable the internal trace for the ASNCLP commands.

### Syntax

```
►►──SET TRACE──┬─OFF─┬─────────────────────────────────────────────────────►◄
               └─ON──┘
```

### Parameters

**OFF**
Specify to turn off the trace.

**ON**
Specify to turn on the trace.

### Usage notes

- The trace is written to stdout and stderr.

### Example 1

To turn off the internal trace for the ASNCLP program:

```
SET TRACE OFF
```

# Chapter 3. ASNCLP commands for Q replication and event publishing

The ASNCLP commands for Q replication are divided into shared commands, commands for unidirectional replication, commands for multidirectional replication, and commands for event publishing.

This section contains the following topics:

- "Sample ASNCLP scripts for Q replication"
- "Shared ASNCLP commands for Q replication and event publishing" on page 73
- "ASNCLP commands for unidirectional Q replication" on page 96
- "ASNCLP commands for multidirectional Q replication" on page 112
- "ASNCLP commands for event publishing" on page 131

## Sample ASNCLP scripts for Q replication

The following sample scripts show you how to put together ASNCLP commands to set up unidirectional, bidirectional, and peer-to-peer Q replication.

- "Sample ASNCLP scripts for setting up unidirectional Q replication"
- "Sample ASNCLP scripts for setting up bidirectional Q replication" on page 57
- "Sample ASNCLP scripts for setting up peer-to-peer Q replication (two servers)" on page 62
- "Sample ASNCLP scripts for setting up peer-to-peer Q replication (three servers)" on page 67

### Sample ASNCLP scripts for setting up unidirectional Q replication

This sample contains four ASNCLP scripts for setting up a unidirectional Q replication environment. It includes Q Capture and Q Apply control tables, a replication queue map, and a Q subscription.

ASNCLP scripts typically generate one or more SQL scripts to create replication objects. Because some replication objects depend on the existence of other objects, run the ASNCLP scripts and the SQL scripts that they generate in the following order:

1. Q Capture and Q Apply control tables
2. Replication queue map
3. Q subscription

The final ASNCLP script checks the WebSphere MQ environment for Q replication and does not generate SQL statements. You must create the control tables and queue map before you run this script.

This sample has a section for each ASNCLP script, which you can copy to a text file and run by using the `ASNCLP -f` *`filename`* command. Within the code sample in each section, details about each group of commands are preceded by a comment character (#).

Table 3 on page 57 below the sample describes the SQL scripts that are generated.

For help creating the WebSphere MQ objects that are used in these scripts, see Graphical tool for generating WebSphere MQ setup scripts for Q replication and event publishing and WebSphere MQ setup scripts for Q replication.

## ASNCLP script 1 (Q Capture and Q Apply control tables)

This script generates SQL statements that create Q Capture control tables at the SAMPLE database and Q Apply control tables at the TARGET database. It includes commands for the following tasks:

1 Setting the environment
2 Creating Q Capture control tables
3 Creating Q Apply control tables
4 Ending the ASNCLP session

```
# 1 Setting the environment.
# In the SET SERVER command, the user ID and password are optional. If you omit
# these keywords, the ASNCLP will use the implicit ID and password for connecting
# to the database.
# The SET LOG command directs ASNCLP messages to the log file qcontrol.err.
# The SET OUTPUT command creates two SQL scripts: qcapctrl.sql, which creates
# Q Capture control tables at the SAMPLE database, and qappctrl.sql, which creates
# Q Apply control tables at the TARGET database.
# The SETQMANAGER commands are required for creating Q replication control tables.
# The SET RUN SCRIPT LATER option allows you to review the SQL scripts before
# they are run.

ASNCLP SESSION SET TO Q REPLICATION;
SET LOG "qcontrol.err";
SET SERVER CAPTURE TO DB SAMPLE ID DB2ADMIN PASSWORD "passw0rd";
SET QMANAGER "QM1" FOR CAPTURE SCHEMA;
SET CAPTURE SCHEMA SOURCE ASN1;
SET SERVER TARGET TO DB TARGET ID DB2ADMIN PASSWORD "passw0rd";
SET QMANAGER "QM2" FOR APPLY SCHEMA;
SET APPLY SCHEMA ASN1;
SET OUTPUT CAPTURE SCRIPT "qcapctrl.sql" TARGET SCRIPT "qappctrl.sql";
SET RUN SCRIPT LATER;

# 2 Creating Q Capture control tables.
# The command specifies a restart queue and administration queue, doubles the
# default amount of memory available to build transactions to 64 MB, and reduces
# the default interval for recording performance information to 600000 milliseconds
# (one minute).
CREATE CONTROL TABLES FOR CAPTURE SERVER USING
RESTARTQ "ASN1.QM1.RESTARTQ" ADMINQ "ASN1.QM1.ADMINQ"
MEMORY LIMIT 64 MONITOR INTERVAL 600000 IN UW TBSPACE TSQCAP;

# 3 Creating Q Apply control tables.
# This command specifies a password file, asnpwd.aut. The Q Apply progam uses this
# file to connect to the Q Capture server when it loads the target table.

CREATE CONTROL TABLES FOR APPLY SERVER USING PWDFILE "asnpwd.aut"
IN UW TBSPACE TSQAPP;

# 4 Ending the ASNCLP session.

QUIT;
```

## ASNCLP script 2 (replication queue map)

This script generates SQL statements to create a replication queue map. It includes commands for the following tasks:

1 Setting the environment
2 Creating a replication queue map
3 Ending the ASNCLP session

```
# 1 Setting the environment.
# The SET OUTPUT command creates two SQL scripts: qcapqmap.sql, which adds
# definitions for the queue map to the Q Capture control tables, and
# qappmap.sql, which adds definitions for the queue map to the Q Apply
# control tables.

ASNCLP SESSION SET TO Q REPLICATION;
SET LOG "rqmap.err";
SET SERVER CAPTURE TO DB SAMPLE ID DB2ADMIN PASSWORD "passw0rd";
SET CAPTURE SCHEMA SOURCE ASN1;
SET SERVER TARGET TO DB TARGET ID DB2ADMIN PASSWORD "passw0rd";
SET APPLY SCHEMA ASN1;
SET OUTPUT CAPTURE SCRIPT "qcapmap.sql" TARGET SCRIPT "qappmap.sql";
SET RUN SCRIPT LATER;

# 2 Creating a replication queue map.
# This command generates SQL to create a replication queue map,
# SAMPLE_ASN1_TO_TARGET_ASN1. It specifies a remote administation
# queue and receive queue at the Q Apply server, and a send queue at
# the Q Capture server. The command also sets the number of agent threads
# for the Q Apply program to 8 (half of the default 16), and specifies that
# heartbeat messages be sent every 5 seconds.

CREATE REPLQMAP SAMPLE_ASN1_TO_TARGET_ASN1 USING
ADMINQ "ASN1.QM1.ADMINQ" RECVQ "ASN1.QM1_TO_QM2.DATAQ"
SENDQ "ASN1.QM1_TO_QM2.DATAQ" NUM APPLY AGENTS 8 HEARTBEAT INTERVAL 5;

# 3 Ending the ASNCLP session.

QUIT;
```

## ASNCLP script 3 (Q subscription)

This script generates SQL statements to create a Q subscription. It specifies a source table, EMPLOYEE, at the SAMPLE database, and a new target table, TGTEMPLOYEE, at the TARGET database. The script includes commands for the following tasks:

1 Setting the environment
2 Creating a Q subscription
3 Ending the ASNCLP session

```
# 1 Setting the environment.
# The SET OUTPUT command creates two SQL scripts: qcapsub.sql, which adds
# definitions for the Q subscription to the Q Capture control tables, and
# qappsub.sql, which adds definitions for the Q subscription to the Q Apply
# control tables.

ASNCLP SESSION SET TO Q REPLICATION;
SET LOG "qsub.err";
SET SERVER CAPTURE TO DB SAMPLE ID DB2ADMIN PASSWORD "passw0rd";
SET CAPTURE SCHEMA SOURCE ASN1;
SET SERVER TARGET TO DB TARGET ID DB2ADMIN PASSWORD "passw0rd";
SET APPLY SCHEMA ASN1;
SET OUTPUT CAPTURE SCRIPT "qcapsub.sql" TARGET SCRIPT "qappsub.sql";
SET RUN SCRIPT LATER;

# 2 Creating the Q subscription
# This command generates SQL to create a Q subscription named EMPLOYEE0001
# that specifies the EMPLOYEE table as a source. The TARGET NAME keywords
# are used without the EXISTS or NAMING PREFIX keywords, resulting in a target
# table name of TGTEMPLOYEE. The EMPNO column, which is the primary key for the
# EMPLOYEE table, is specified as the key for replication. The command also
# specifies that the Q Apply program load the target table (LOAD PHASE I) using
# the EXPORT and IMPORT utilities (LOAD TYPE 2).

CREATE QSUB USING REPLQMAP SAMPLE_ASN1_TO_TARGET_ASN1
(SUBNAME EMPLOYEE0001 EMPLOYEE OPTIONS HAS LOAD PHASE I
TARGET NAME EMPLOYEE KEYS (EMPNO) LOAD TYPE 2);

# 3 Ending the ASNCLP session.

QUIT;
```

## ASNCLP script 4 (check WebSphere MQ environment)

This script does not generate SQL. Instead, it checks whether the queue managers and queues that were specified in the other scripts exist, and whether the objects have the correct properties for Q replication. Then it checks the message flow between the queues in the replication queue map by sending test messages. The script includes commands for the following tasks:

1 Setting the environment
2 Checking the queue managers and queues
3 Sending test messages
4 Ending the ASNCLP session

```
# 1 Setting the environment.
# No SET RUN statement is required. The commands run immediately and send
# results to the command window and log.

ASNCLP SESSION SET TO Q REPLICATION;
SET LOG "qchecks.err";
SET SERVER CAPTURE TO DB SAMPLE ID DB2ADMIN PASSWORD "passw0rd";
SET QMANAGER "QM1" FOR CAPTURE SCHEMA;
SET CAPTURE SCHEMA SOURCE ASN1;
SET SERVER TARGET TO DB TARGET ID DB2ADMIN PASSWORD "passw0rd";
SET QMANAGER "QM2" FOR APPLY SCHEMA;
SET APPLY SCHEMA ASN1;

# 2 Checking the queue managers and queues.
# These commands check whether the queue managers and queues exist, and validate
# their settings against the requirements for Q replication. If errors are detected,
# you must correct them before you start the Q Capture and Q Apply programs.

VALIDATE WSMQ ENVIRONMENT FOR CAPTURE SCHEMA;
VALIDATE WSMQ ENVIRONMENT FOR APPLY SCHEMA;
VALIDATE WSMQ ENVIRONMENT FOR REPLQMAP SAMPLE_ASN1_TO_TARGET_ASN1;

# 3 Sending test messages.
# This command puts a test message on the send queue, ASN1.QM1_TO_QM2.DATAQ,
# and tries to get the message from the receive queue, ASN1.QM1_TO_QM2.DATAQ.
# The command also puts a test message on the Q Apply administration queue,
# ASN1.QM1.ADMINQ, and tries to get the message from the Q Capture administration
# queue, ASN1.QM1.ADMINQ.

VALIDATE WSMQ MESSAGE FLOW FOR REPLQMAP SAMPLE_ASN1_TO_TARGET_ASN1;

# 4 Ending the ASNCLP session.

QUIT;
```

## Output of the scripts

In addition to the log files, this example produces six SQL script files in the same directory where you run the ASNCLP program. Table 3 on page 57 describes the files.

*Table 3. SQL script files that are created by the sample ASNCLP scripts*

| Output file | Contains SQL to ... |
|---|---|
| qcapctrl.sql | Create Q Capture control tables |
| qappctrl.sql | Create Q Apply control tables |
| qcapqmap.sql | Insert definitions for a replication queue map into the Q Capture control tables |
| qappqmap.sql | Insert definitions for a replication queue map into the Q Apply control tables |
| qcapqsub.sql | Insert definitions for a Q subscription into the Q Capture control tables |
| qappqsub.sql | Insert definitions for a Q subscription into the Q Apply control tables |

## Sample ASNCLP scripts for setting up bidirectional Q replication

This sample contains six ASNCLP scripts for setting up a bidirectional Q replication environment. It includes Q Capture and Q Apply control tables at both servers, replication queue maps in both directions, and two bidirectional Q subscriptions.

The scenario is a standby configuration with two databases, SAMPLE (the primary server) and SAMPLE2 (the standby server). One table, EMPLOYEE, will be replicated in both directions between the two databases. The Q Capture and Q Apply programs at the SAMPLE database have the schema RED. The two corresponding programs at the SAMPLE2 database have the schema BLUE.

The ASNCLP scripts create eight SQL scripts. Table 4 on page 61 below the sample describes each SQL script.

Because some replication objects depend on the existence of other objects, run the ASNCLP scripts and the SQL scripts that they generate in the following order:

1. Q Capture and Q Apply control tables at SAMPLE
2. Q Capture and Q Apply control tables at SAMPLE2
3. Replication queue map from SAMPLE to SAMPLE2
4. Replication queue map from SAMPLE2 to SAMPLE
5. Q subscriptions

You can copy the commands for each ASNCLP script to a text file, modify the values, and run the script by using the `ASNCLP -f` *`filename`* command. Within the code sample in each section, details about each group of commands are preceded by a comment character (#).

For help creating the WebSphere MQ objects that are used in these scripts, see Graphical tool for generating WebSphere MQ setup scripts for Q replication and event publishing and WebSphere MQ setup scripts for Q replication.

## ASNCLP script 1

This script creates control tables at the SAMPLE database. It includes commands for the following tasks:

1 Setting the environment
2 Creating Q Capture and Q Apply control tables at the SAMPLE database
3 Ending the ASNCLP session

```
# 1 Setting the environment
# The SET OUTPUT MULTIDIR command creates one SQL script, SAMPLE.sql, that is
# automatically named after the database. The script contains SQL statements to
# create both Q Capture and Q Apply control tables.
# The SET LOG command directs ASNCLP messages to one log file, bidir1.err.
# The SET RUN SCRIPT LATER option allows you to review the SQL scripts before
# they are run.

ASNCLP SESSION SET TO Q REPLICATION;
SET OUTPUT MULTIDIR;
SET LOG "bidir1.err";
SET RUN SCRIPT LATER;

# 2 Creating Q Capture and Q Apply control tables at SAMPLE
# To use the script, change the ID and PASSWORD values.
# Both the Q Capture and Q Apply control tables will have the schema RED.

SET SERVER CAPTURE TO DB SAMPLE ID DB2ADMIN PASSWORD "passw0rd";
SET CAPTURE SCHEMA SOURCE RED;
SET QMANAGER "QM1" FOR CAPTURE SCHEMA;
CREATE CONTROL TABLES FOR CAPTURE SERVER USING
RESTARTQ "RED.QM1.RESTARTQ" ADMINQ "RED.QM1.ADMINQ";
SET SERVER TARGET TO DB SAMPLE ID DB2ADMIN PASSWORD "passw0rd";
SET APPLY SCHEMA RED;
SET QMANAGER "QM1" FOR APPLY SCHEMA;
CREATE CONTROL TABLES FOR APPLY SERVER USING PWDFILE "asnpwd.aut";

# 3 Ending the ASNCLP session

QUIT;
```

## ASNCLP script 2

This script creates control tables at the SAMPLE2 database. It includes commands for the following tasks:

1 Setting the environment
2 Creating Q Capture and Q Apply control tables at the SAMPLE2 database
3 Ending the ASNCLP session

```
# 1 Setting the environment
# The SET OUTPUT MULTIDIR command creates one SQL script, SAMPLE2.sql. The script
# contains SQL statements to create both Q Capture and Q Apply control tables.

ASNCLP SESSION SET TO Q REPLICATION;
SET OUTPUT MULTIDIR;
SET LOG "bidir2.err";
SET RUN SCRIPT LATER;

# 2 Creating Q Capture and Q Apply control tables at SAMPLE2
# Both the Q Capture and Q Apply control tables will have the schema BLUE.

SET SERVER CAPTURE TO DB SAMPLE2 ID DB2ADMIN PASSWORD "passw0rd";
SET CAPTURE SCHEMA SOURCE BLUE;
SET QMANAGER "QM2" FOR CAPTURE SCHEMA;
CREATE CONTROL TABLES FOR CAPTURE SERVER USING
```

```
RESTARTQ "BLUE.QM2.RESTARTQ" ADMINQ "BLUE.QM2.ADMINQ";
SET SERVER TARGET TO DB SAMPLE2 ID DB2ADMIN PASSWORD "passw0rd";
SET APPLY SCHEMA BLUE;
SET QMANAGER "QM2" FOR APPLY SCHEMA;
CREATE CONTROL TABLES FOR APPLY SERVER USING PWDFILE "asnpwd.aut";

# 3 End the ASNCLP session.

QUIT;
```

## ASNCLP script 3

This script creates a replication queue map from SAMPLE to SAMPLE2. It includes commands for the following tasks:

1 Setting the environment
2 Creating a replication queue map from SAMPLE to SAMPLE2
3 Ending the ASNCLP session

```
# 1 Setting the environment
# Two SET SERVER commands are required because the replication queue map from
# SAMPLE to SAMPLE2 is defined in the Q Capture control tables at SAMPLE and
# the Q Apply control tables at SAMPLE2.
# The SET OUTPUT command specifies two SQL scripts: rqmred1.sql, which adds
# definitions to SAMPLE, and rqmblue1.sql, which adds definitions to SAMPLE2.

ASNCLP SESSION SET TO Q REPLICATION;
SET SERVER CAPTURE TO DB SAMPLE ID DB2ADMIN PASSWORD "passw0rd";
SET CAPTURE SCHEMA SOURCE RED;
SET SERVER TARGET TO DB SAMPLE2 ID DB2ADMIN PASSWORD "passw0rd";
SET APPLY SCHEMA BLUE;
SET OUTPUT CAPTURE SCRIPT "rqmred1.sql" TARGET SCRIPT "rqmblue1.sql";
SET LOG "bidir3.err";
SET RUN SCRIPT LATER;

# 2 Creating a replication queue map
# The CREATE REPLQMAP command specifies an administration queue and receive queue
# within the queue manager QM2 that is used for SAMPLE2, and a send queue within
# the queue manager QM1 that is used for SAMPLE.

CREATE REPLQMAP SAMPLE_RED_TO_SAMPLE2_BLUE USING
ADMINQ "BLUE.QM1.ADMINQ" RECVQ "BLUE.QM1_TO_QM2.DATAQ"
SENDQ "RED.QM1_TO_QM2.DATAQ";

# 3 Ending the ASNCLP session

QUIT;
```

## ASNCLP script 4

This script creates a replication queue map from SAMPLE2 to SAMPLE. It includes commands for the following tasks:

1 Setting the environment
2 Creating a replication queue map from SAMPLE2 to SAMPLE
3 Ending the ASNCLP session

```
# 1 Setting the environment
# Two SET SERVER commands are required because the replication queue map from
# SAMPLE2 to SAMPLE is defined in the Q Capture control tables at SAMPLE2 and
# the Q Apply control tables at SAMPLE.
# The SET OUTPUT command specifies two SQL scripts, rqmblue2.sql, which adds
# definitions to SAMPLE2, and rqmred2.sql, which adds definitions to SAMPLE.

ASNCLP SESSION SET TO Q REPLICATION;
```

```
SET SERVER CAPTURE TO DB SAMPLE2 ID DB2ADMIN PASSWORD "passw0rd";
SET CAPTURE SCHEMA SOURCE BLUE;
SET SERVER TARGET TO DB SAMPLE ID DB2ADMIN PASSWORD "passw0rd";
SET APPLY SCHEMA RED;
SET OUTPUT CAPTURE SCRIPT "rqmblue2.sql" TARGET SCRIPT "rqmred2.sql";
SET LOG "bidir4.err";
SET RUN SCRIPT LATER;

# 2 Creating a replication queue map
# The CREATE REPLQMAP command specifies an administration queue and receive queue
# within the queue manager QM1 that is used for SAMPLE, and a send queue within
# the queue manager QM2 that is used for SAMPLE2.

CREATE REPLQMAP SAMPLE2_BLUE_TO_SAMPLE_RED USING
ADMINQ "RED.QM2.ADMINQ" RECVQ "RED.QM2_TO_QM1.DATAQ"
SENDQ "BLUE.QM1_TO_QM2.DATAQ";

# 3 Ending the ASNCLP session

QUIT;
```

## ASNCLP script 5

This ASNCLP script contains commands for the Q subscriptions between the SAMPLE database and the SAMPLE2 database. You invoke this script by using the LOAD MULTIDIR REPL SCRIPT command in ASNCLP script 6. The script includes commands for the following tasks:

1 Setting the subgroup
2 Setting servers for the subgroup
3 Identifying the matching Q Capture and Q Apply schema at each server
4 Specifying replication queue maps to connect the servers in both directions
5 Specifying the table to be replicated (one copy at each server)
6 Creating the Q subscriptions

```
# No environment commands are required for this script. These commands are included
# in ASNCLP script 6, which invokes this script.

# 1 Setting the subgroup

SET SUBGROUP "bidirgroup";

# 2 Setting servers for the subgroup

SET SERVER MULTIDIR TO DB "SAMPLE";
SET SERVER MULTIDIR TO DB "SAMPLE2";

# 3 Identifying the matching schema of the Q Capture and Q Apply
# control tables at each server

SET MULTIDIR SCHEMA "SAMPLE".RED;
SET MULTIDIR SCHEMA "SAMPLE2".BLUE;

# 4 Specifying the replication queue maps that connect the two servers
# in both directions

SET CONNECTION SOURCE "SAMPLE".RED TARGET "SAMPLE2".BLUE REPLQMAP
"SAMPLE_RED_TO_SAMPLE2_BLUE";
SET CONNECTION SOURCE "SAMPLE2".BLUE TARGET "SAMPLE".RED REPLQMAP
"SAMPLE2_BLUE_TO_SAMPLE_RED";

# 5 Specifying the table to be replicated (one copy at each server)
# The SET TABLES command specifies only one table, RED.EMPLOYEE at the SAMPLE
# database. This prompts the command to generate SQL statements to create a matching
# table at the SAMPLE2 database, BLUE.TGTEMPLOYEE.
```

```
SET TABLES (SAMPLE.RED.RED.EMPLOYEE);

# 6 Creating the Q subscriptions
# The command uses two FROM NODE clauses to specify a conflict rule of C (check
# changed columns) and a conflict action of F (force changes into the target) for
# the SAMPLE database. For SAMPLE2 (the standby server), the conflict rule is
# A (check all columns) and the conflict action is I (ignore conflicts).

CREATE QSUB SUBTYPE B
FROM NODE SAMPLE.RED SOURCE ALL CHANGED ROWS Y HAS LOAD PHASE I
TARGET CONFLICT RULE C CONFLICT ACTION F
FROM NODE SAMPLE2.BLUE SOURCE ALL CHANGED ROWS N HAS LOAD PHASE E
TARGET CONFLICT RULE A CONFLICT ACTION I;

# No QUIT statement is required. The ASNCLP program reads this statement in
# script 6.
```

## ASNCLP script 6

The final script uses the LOAD MULTIDIR REPL SCRIPT command to invoke ASNCLP script 5 for creating the Q subscriptions. It includes commands for the following tasks:

1 Setting the environment
2 Invoking the script that creates Q subscriptions
3 Ending the ASNCLP session

```
# 1 Setting the environment
# The SET OUTPUT MULTIDIR command creates two SQL scripts that are automatically
# named after the databases, SAMPLE.sql and SAMPLE2.sql. Run each SQL script at the
# database for which it is named.
# IMPORTANT: Move or rename the existing SAMPLE.sql and SAMPLE2.sql scripts that
# were generated for creating control tables, or the statements for creating
# Q subscriptions will be appended to the end of the files.

ASNCLP SESSION SET TO Q REPLICATION;
SET OUTPUT MULTIDIR;
SET LOG "bidir5.err";
SET RUN SCRIPT LATER;

# 2 Invoking the script that creates Q subscriptions
# Before you run this script, save ASNCLP script 5 in a file, bidirqsubs.in.

LOAD MULTIDIR REPL SCRIPT "/home/files/asnclp/bidirqsubs.in";

# 3 Ending the ASNCLP session

QUIT;
```

## Output of the scripts

In addition to the log files, this example produces eight SQL script files in the same directory where you run the ASNCLP program. Table 4 describes the files.

*Table 4. SQL script files that are created by the sample ASNCLP scripts*

| Output file | Contains SQL to ... |
|---|---|
| SAMPLE.sql | Create Q Capture and Q Apply control tables at the SAMPLE database. |
| SAMPLE2.sql | Create Q Capture and Q Apply control tables at the SAMPLE2 database. |

*Table 4. SQL script files that are created by the sample ASNCLP scripts (continued)*

| Output file | Contains SQL to ... |
|---|---|
| rqmred1.sql | Add definitions to the Q Capture control tables at SAMPLE for the replication queue map SAMPLE_RED_TO_SAMPLE2_BLUE. |
| rqmblue1.sql | Add definitions to the Q Apply control tables at SAMPLE2 for the replication queue map SAMPLE_RED_TO_SAMPLE2_BLUE. |
| rqmred2.sql | Add definitions to the Q Capture control tables at SAMPLE for the replication queue map SAMPLE2_BLUE_TO_SAMPLE_RED. |
| rqmblue2.sql | Add definitions to the Q Apply control tables at SAMPLE2 for the replication queue map SAMPLE2_BLUE_TO_SAMPLE_RED. |
| SAMPLE.sql | Add Q subscription definitions to the Q Capture and Q Apply control tables at SAMPLE. |
| SAMPLE2.sql | Add Q subscription definitions to the Q Capture and Q Apply control tables at SAMPLE2. |

## Sample ASNCLP scripts for setting up peer-to-peer Q replication (two servers)

This sample contains six ASNCLP scripts for setting up a peer-to-peer Q replication environment with two servers. It includes Q Capture and Q Apply control tables at both servers, replication queue maps in both directions, and two peer-to-peer Q subscriptions.

The scenario for these samples involves two databases, SAMPLE and SAMPLPEER. One table, DEPARTMENT, will be replicated in both directions between the two databases. The Q Capture and Q Apply programs at the SAMPLE database have the schema GREEN. The two corresponding programs at the SAMPLPEER database have the schema MAGENTA.

The ASNCLP scripts create eight SQL scripts. Table 5 on page 66 below the sample describes each SQL script.

Because some replication objects depend on the existence of other objects, run the ASNCLP scripts and the SQL scripts that they generate in the following order:

1. Q Capture and Q Apply control tables at SAMPLE
2. Q Capture and Q Apply control tables at SAMPLPEER
3. Replication queue map from SAMPLE to SAMPLPEER
4. Replication queue map from SAMPLPEER to SAMPLE
5. Q subscriptions

You can copy the commands for each ASNCLP script to a text file, modify the values, and run the script by using the `ASNCLP -f` *`filename`* command. Within the code sample in each section, details about each group of commands are preceded by a comment character (#).

For help creating the WebSphere MQ objects that are used in these scripts, see Graphical tool for generating WebSphere MQ setup scripts for Q replication and event publishing and WebSphere MQ setup scripts for Q replication.

## ASNCLP script 1

This script creates control tables at the SAMPLE database. It includes commands for the following tasks:

1 Setting the environment
2 Creating Q Capture and Q Apply control tables at SAMPLE
3 Ending the ASNCLP session

```
# 1 Setting the environment
# The SET OUTPUT MULTIDIR command creates one SQL script, SAMPLE.sql, that is
# automatically named after the database. The script contains SQL statements to
# create both Q Capture and Q Apply control tables.
# The SET LOG command directs ASNCLP messages to one log file, p2p2-log1.err.
# The SET RUN SCRIPT LATER option allows you to review the SQL scripts before
# they are run.

ASNCLP SESSION SET TO Q REPLICATION;
SET OUTPUT MULTIDIR;
SET LOG "p2p2-log1.err";
SET RUN SCRIPT LATER;

# 2 Creating Q Capture and Q Apply control tables at SAMPLE
# To use the script, change the ID and PASSWORD values.
# Both the Q Capture and Q Apply control tables will have the schema GREEN.

SET SERVER CAPTURE TO DB SAMPLE ID DB2ADMIN PASSWORD "passw0rd";
SET CAPTURE SCHEMA SOURCE GREEN;
SET QMANAGER "QM1" FOR CAPTURE SCHEMA;
CREATE CONTROL TABLES FOR CAPTURE SERVER USING
RESTARTQ "GREEN.QM1.RESTARTQ" ADMINQ "GREEN.QM1.ADMINQ";
SET SERVER TARGET TO DB SAMPLE ID DB2ADMIN PASSWORD "passw0rd";
SET APPLY SCHEMA GREEN;
SET QMANAGER "QM1" FOR APPLY SCHEMA;
CREATE CONTROL TABLES FOR APPLY SERVER USING PWDFILE "asnpwd.aut";

# 3 Ending the ASNCLP session

QUIT;
```

## ASNCLP script 2

This script creates control tables at the SAMPLPEER database. It includes commands for the following tasks:

1 Setting the environment
2 Creating Q Capture and Q Apply control tables at SAMPLPEER
3 Ending the ASNCLP session

```
# 1 Setting the environment
# The SET OUTPUT MULTIDIR command creates one SQL script, SAMPLPEER.sql. The
# script contains SQL statements to create both Q Capture and Q Apply control
# tables.

ASNCLP SESSION SET TO Q REPLICATION;
SET OUTPUT MULTIDIR;
SET LOG "p2p2-log2.err";
SET RUN SCRIPT LATER;

# 2 Creating Q Capture and Q Apply control tables at SAMPLPEER
# Both the Q Capture and Q Apply control tables will have the schema MAGENTA.

SET SERVER CAPTURE TO DB SAMPLPEER ID DB2ADMIN PASSWORD "passw0rd";
SET CAPTURE SCHEMA SOURCE MAGENTA;
SET QMANAGER "QM2" FOR CAPTURE SCHEMA;
CREATE CONTROL TABLES FOR CAPTURE SERVER USING
RESTARTQ "MAGENTA.QM2.RESTARTQ" ADMINQ "MAGENTA.QM2.ADMINQ";
SET SERVER TARGET TO DB SAMPLPEER ID DB2ADMIN PASSWORD "passw0rd";
SET APPLY SCHEMA MAGENTA;
SET QMANAGER "QM2" FOR APPLY SCHEMA;
CREATE CONTROL TABLES FOR APPLY SERVER USING PWDFILE "asnpwd.aut";

# 3 End the ASNCLP session.

QUIT;
```

## ASNCLP script 3

This script creates a replication queue map from SAMPLE to SAMPLPEER. It includes commands for the following tasks:

1 Setting the environment
2 Creating a replication queue map from SAMPLE to SAMPLPEER
3 Ending the ASNCLP session

```
# 1 Setting the environment
# Two SET SERVER commands are required because the replication queue map from
# SAMPLE to SAMPLPEER is defined in the Q Capture control tables at SAMPLE and
# the Q Apply control tables at SAMPLPEER.
# The SET OUTPUT command specifies two SQL scripts: rqmgreen1.sql, which adds
# definitions to SAMPLE, and rqmmagenta1.sql, which adds definitions to SAMPLPEER.

ASNCLP SESSION SET TO Q REPLICATION;
SET SERVER CAPTURE TO DB SAMPLE ID DB2ADMIN PASSWORD "passw0rd";
SET CAPTURE SCHEMA SOURCE GREEN;
SET SERVER TARGET TO DB SAMPLPEER ID DB2ADMIN PASSWORD "passw0rd";
SET APPLY SCHEMA MAGENTA;
SET OUTPUT CAPTURE SCRIPT "rqmgreen1.sql" TARGET SCRIPT "rqmmagenta1.sql";
SET LOG "p2p2-log3.err";
SET RUN SCRIPT LATER;

# 2 Creating a replication queue map
# The CREATE REPLQMAP command specifies an administration queue and receive queue
# within the queue manager QM2 that is used for SAMPLPEER, and a send queue within
# the queue manager QM1 that is used for SAMPLE.

CREATE REPLQMAP SAMPLE_GREEN_TO_SAMPLPEER_MAGENTA USING
ADMINQ "MAGENTA.QM1.ADMINQ" RECVQ "MAGENTA.QM1_TO_QM2.DATAQ"
SENDQ "GREEN.QM1_TO_QM2.DATAQ";

# 3 Ending the ASNCLP session

QUIT;
```

## ASNCLP script 4

This script creates a replication queue map from SAMPLPEER to SAMPLE. It includes commands for the following tasks:

1 Setting the environment
2 Creating a replication queue map from SAMPLPEER to SAMPLE
3 Ending the ASNCLP session

```
# 1 Setting the environment
# Two SET SERVER commands are required because the replication queue map from
# SAMPLPEER to SAMPLE is defined in the Q Capture control tables at SAMPLPEER
# and the Q Apply control tables at SAMPLE.
# The SET OUTPUT command specifies two SQL scripts, rqmmagenta2.sql, which adds
# definitions to SAMPLPEER, and rqmgreen1.sql, which adds definitions to SAMPLE.

ASNCLP SESSION SET TO Q REPLICATION;
SET SERVER CAPTURE TO DB SAMPLPEER ID DB2ADMIN PASSWORD "passw0rd";
SET CAPTURE SCHEMA SOURCE MAGENTA;
SET SERVER TARGET TO DB SAMPLE ID DB2ADMIN PASSWORD "passw0rd";
SET APPLY SCHEMA GREEN;
SET OUTPUT CAPTURE SCRIPT "rqmmagenta2.sql" TARGET SCRIPT "rqmgreen2.sql";
SET LOG "p2p2-log4.err";
SET RUN SCRIPT LATER;

# 2 Creating a replication queue map
# The CREATE REPLQMAP command specifies an administration queue and receive
# queue within the queue manager QM1 that is used for SAMPLE, and a send queue
# within the queue manager QM2 that is used for SAMPLPEER.

CREATE REPLQMAP SAMPLPEER_MAGENTA_TO_SAMPLE_GREEN USING
ADMINQ "GREEN.QM2.ADMINQ" RECVQ "GREEN.QM2_TO_QM1.DATAQ"
SENDQ "MAGENTA.QM1_TO_QM2.DATAQ";

# 3 Ending the ASNCLP session

QUIT;
```

## ASNCLP script 5

This ASNCLP script contains commands for the Q subscriptions between the SAMPLE database and the SAMPLPEER database. You invoke this script by using the LOAD MULTIDIR REPL SCRIPT command in ASNCLP script 6. The script includes commands for the following tasks:

1 Setting the subgroup
2 Setting servers for the subgroup
3 Identifying the matching Q Capture and Q Apply schema at each server
4 Specifying queue maps that connect the servers in both directions
5 Specifying the table to be replicated (one copy at each server)
6 Creating the Q subscriptions

```
# No environment commands are required for this script. These commands are
# included in ASNCLP script 6, which invokes this script.

# 2 Setting the subgroup

SET SUBGROUP "p2p2group";

# 3 Setting servers for the subgroup

SET SERVER MULTIDIR TO DB "SAMPLE";
SET SERVER MULTIDIR TO DB "SAMPLPEER";

# 4 Identifying the matching schema of the Q Capture and Q Apply
# control tables at each server

SET MULTIDIR SCHEMA "SAMPLE".GREEN;
SET MULTIDIR SCHEMA "SAMPLPEER".MAGENTA;

# 5 Specifying the replication queue maps that connect the two servers
# in both directions

SET CONNECTION SOURCE "SAMPLE".GREEN TARGET "SAMPLPEER".MAGENTA REPLQMAP
```

```
"SAMPLE_GREEN_TO_SAMPLPEER_MAGENTA";
SET CONNECTION SOURCE "SAMPLPEER".MAGENTA TARGET "SAMPLE".GREEN REPLQMAP
"SAMPLPEER_MAGENTA_TO_SAMPLE_GREEN";

# 6 Specifying the table to be replicated (one copy at each server)
# The SET TABLES command specifies only one table, GREEN.DEPARTMENT at the
# SAMPLE database. This prompts the command to generate SQL statements to create
# a matching table at the SAMPLPEER database, MAGENTA.TGTEMPLOYEE.

SET TABLES (SAMPLE.GREEN.GREEN.EMPLOYEE);

# 7 Creating the Q subscriptions
# A single CREATE QSUB command generates commands to create two peer-to-peer
# Q subscriptions between SAMPLE and SAMPLPEER.

CREATE QSUB SUBTYPE P;

# No QUIT statement is required. The ASNCLP program reads this statement in
# script 6.
```

## ASNCLP script 6

The final script uses the LOAD MULTIDIR REPL SCRIPT command to invoke ASNCLP script 5 for creating the Q subscriptions. It includes commands for the following tasks:

1 Setting the environment
2 Invoking the script that creates Q subscriptions
3 Ending the ASNCLP session

```
# 1 Setting the environment
# The SET OUTPUT MULTIDIR command creates two SQL scripts that are automatically
# named after the databases, SAMPLE.sql and SAMPLPEER.sql. Run each SQL script
# at the database for which it is named.
# IMPORTANT: Move or rename the existing SAMPLE.sql and SAMPLPEER.sql scripts
# that were generated for creating control tables, or the statements for creating
# Q subscriptions will be appended to the end of the files.

ASNCLP SESSION SET TO Q REPLICATION;
SET OUTPUT MULTIDIR;
SET LOG "p2p2-log5.err";
SET RUN SCRIPT LATER;

# 2 Invoking the script that creates Q subscriptions
# Before you run this script, save ASNCLP script 5 in a file, p2p2qsubs.in.

LOAD MULTIDIR REPL SCRIPT "/home/files/asnclp/p2p2qsubs.in";

# 3 Ending the ASNCLP session

QUIT;
```

## Output of the scripts

In addition to the log files, this example produces eight SQL script files in the same directory where you run the ASNCLP program. Table 5 describes the files.

*Table 5. SQL script files that are created by the sample ASNCLP scripts*

| Output file | Contains SQL to ... |
|---|---|
| SAMPLE.sql | Create Q Capture and Q Apply control tables at the SAMPLE database. |

*Table 5. SQL script files that are created by the sample ASNCLP scripts (continued)*

| Output file | Contains SQL to ... |
|---|---|
| SAMPLPEER.sql | Create Q Capture and Q Apply control tables at the SAMPLPEER database. |
| rqmgreen1.sql | Add definitions to the Q Capture control tables at SAMPLE for the replication queue map SAMPLE_GREEN_TO_ SAMPLPEER_MAGENTA. |
| rqmmagenta1.sql | Add definitions to the Q Apply control tables at SAMPLPEER for the replication queue map SAMPLE_GREEN_TO_ SAMPLPEER_MAGENTA. |
| rqmgreen2.sql | Add definitions to the Q Capture control tables at SAMPLE for the replication queue map SAMPLPEER_MAGENTA_TO_ SAMPLE_GREEN. |
| rqmmagenta2.sql | Add definitions to the Q Apply control tables at SAMPLPEER for the replication queue map SAMPLPEER_MAGENTA_TO_ SAMPLE_GREEN. |
| SAMPLE.sql | Add Q subscription definitions to the Q Capture and Q Apply control tables at SAMPLE. |
| SAMPLPEER.sql | Add Q subscription definitions to the Q Capture and Q Apply control tables at SAMPLPEER. |

## Sample ASNCLP scripts for setting up peer-to-peer Q replication (three servers)

This sample contains six ASNCLP scripts for setting up peer-to-peer Q replication with three servers. It includes Q Capture and Q Apply control tables at each of the three servers, replication queue maps in both directions between each server, and six Q subscriptions between the servers.

The scenario involves three databases, SAMPLE, SAMPLE2, and SAMPLE3. One table, STAFF, will be replicated between the three databases. The Q Capture and Q Apply programs at the SAMPLE database have the schema GRAY. At SAMPLE2 the programs have the schema BROWN, and at SAMPLE3 the schema is YELLOW.

The ASNCLP scripts create 12 SQL scripts. Table 6 on page 72 below the sample describes each SQL script.

Because some replication objects depend on the existence of other objects, run the ASNCLP scripts and the SQL scripts that they generate in the following order:

1. Q Capture and Q Apply control tables at SAMPLE
2. Q Capture and Q Apply control tables at SAMPLE2
3. Q Capture and Q Apply control tables at SAMPLE3
4. Replication queue maps
5. Q subscriptions

You can copy the commands for each ASNCLP script to a text file, modify the values, and run the script by using the `ASNCLP -f filename` command. Within the code sample in each section, details about each group of commands are preceded by a comment character (#).

For help creating the WebSphere MQ objects that are used in these scripts, see Graphical tool for generating WebSphere MQ setup scripts for Q replication and event publishing and WebSphere MQ setup scripts for Q replication.

## ASNCLP script 1

This script creates control tables at the SAMPLE database. It includes commands for the following tasks:

1 Setting the environment
2 Creating Q Capture and Q Apply control tables at the SAMPLE database
3 Ending the ASNCLP session

```
# 1 Setting the environment
# The SET OUTPUT MULTIDIR command creates one SQL script that is automatically
# named after the database, SAMPLE.sql. The script contains SQL statements to
# create both Q Capture and Q Apply control tables.
# The SET LOG command directs ASNCLP messages to one log file, p2p3-log1.err.
# The SET RUN SCRIPT LATER option allows you to review the SQL scripts before
# they are run.

ASNCLP SESSION SET TO Q REPLICATION;
SET OUTPUT MULTIDIR;
SET LOG "p2p3-log1.err";
SET RUN SCRIPT LATER;

# 2 Creating Q Capture and Q Apply control tables at SAMPLE
# To use the script, change the ID and PASSWORD values.
# Both the Q Capture and Q Apply control tables will have the schema GRAY.

SET SERVER CAPTURE TO DB SAMPLE ID DB2ADMIN PASSWORD "passw0rd";
SET CAPTURE SCHEMA SOURCE GRAY;
SET QMANAGER "QM1" FOR CAPTURE SCHEMA;
CREATE CONTROL TABLES FOR CAPTURE SERVER USING
RESTARTQ "GRAY.QM1.RESTARTQ" ADMINQ "GRAY.QM1.ADMINQ";
SET SERVER TARGET TO DB SAMPLE ID DB2ADMIN PASSWORD "passw0rd";
SET APPLY SCHEMA GRAY;
SET QMANAGER "QM1" FOR APPLY SCHEMA;
CREATE CONTROL TABLES FOR APPLY SERVER USING PWDFILE "asnpwd.aut";

# 3 Ending the ASNCLP session

QUIT;
```

## ASNCLP script 2

This script creates control tables at the SAMPLE2 database. It includes commands for the following tasks:

1 Setting the environment
2 Creating Q Capture and Q Apply control tables at the SAMPLE2 database
3 Ending the ASNCLP session

```
# 1 Setting the environment
# The SET OUTPUT MULTIDIR command creates one SQL script, SAMPLE2.sql.
# The script contains SQL statements to create both Q Capture and Q Apply
# control tables.
```

```
ASNCLP SESSION SET TO Q REPLICATION;
SET OUTPUT MULTIDIR;
SET LOG "p2p3-log2.err";
SET RUN SCRIPT LATER;

# 2 Creating Q Capture and Q Apply control tables at SAMPLE2
# Both the Q Capture and Q Apply control tables will have the schema BROWN.

SET SERVER CAPTURE TO DB SAMPLE2 ID DB2ADMIN PASSWORD "passw0rd";
SET CAPTURE SCHEMA SOURCE BROWN;
SET QMANAGER "QM2" FOR CAPTURE SCHEMA;
CREATE CONTROL TABLES FOR CAPTURE SERVER USING
RESTARTQ "BROWN.QM2.RESTARTQ" ADMINQ "BROWN.QM2.ADMINQ";
SET SERVER TARGET TO DB SAMPLE2 ID DB2ADMIN PASSWORD "passw0rd";
SET APPLY SCHEMA BROWN;
SET QMANAGER "QM2" FOR APPLY SCHEMA;
CREATE CONTROL TABLES FOR APPLY SERVER USING PWDFILE "asnpwd.aut";

# 3 End the ASNCLP session.

QUIT;
```

## ASNCLP script 3

This script creates control tables at the SAMPLE3 database. It includes commands for the following tasks:

1 Setting the environment
2 Creating Q Capture and Q Apply control tables at the SAMPLE3 database
3 Ending the ASNCLP session

```
# 1 Setting the environment
# The SET OUTPUT MULTIDIR command creates one SQL script, SAMPLE3.sql.
# The script contains SQL statements to create both Q Capture and Q Apply
# control tables.

ASNCLP SESSION SET TO Q REPLICATION;
SET OUTPUT MULTIDIR;
SET LOG "p2p3-log3.err";
SET RUN SCRIPT LATER;

# 2 Creating Q Capture and Q Apply control tables at SAMPLE3
# Both the Q Capture and Q Apply control tables will have the schema YELLOW.

SET SERVER CAPTURE TO DB SAMPLE2 ID DB2ADMIN PASSWORD "passw0rd";
SET CAPTURE SCHEMA SOURCE YELLOW;
SET QMANAGER "QM3" FOR CAPTURE SCHEMA;
CREATE CONTROL TABLES FOR CAPTURE SERVER USING
RESTARTQ "YELLOW.QM3.RESTARTQ" ADMINQ "YELLOW.QM3.ADMINQ";
SET SERVER TARGET TO DB SAMPLE2 ID DB2ADMIN PASSWORD "passw0rd";
SET APPLY SCHEMA YELLOW;
SET QMANAGER "QM3" FOR APPLY SCHEMA;
CREATE CONTROL TABLES FOR APPLY SERVER USING PWDFILE "asnpwd.aut";

# 3 End the ASNCLP session.

QUIT;
```

## ASNCLP script 4

This script creates six replication queue maps, from SAMPLE to SAMPLE2 and SAMPLE3, from SAMPLE2 to SAMPLE and SAMPLE3, and from SAMPLE3 to SAMPLE and SAMPLE2:

**1** Creating the replication queue maps
**2** Ending the ASNCLP session

```
# 2 Creating the replication queue maps
# Two servers and two schemas need to be set for each CREATE REPLQMAP command.
# The SET OUTPUT commands create a separate SQL script for each queue map.

ASNCLP SESSION SET TO Q REPLICATION;
SET LOG "p2p3-log4.err";
SET RUN SCRIPT LATER;

# First queue map
SET SERVER CAPTURE TO DB SAMPLE ID DB2ADMIN PASSWORD "passw0rd";
SET CAPTURE SCHEMA SOURCE GRAY;
SET SERVER TARGET TO DB SAMPLE2 ID DB2ADMIN PASSWORD "passw0rd";
SET APPLY SCHEMA BROWN;
SET OUTPUT CAPTURE SCRIPT "rqm1.sql" TARGET SCRIPT "rqm1.sql";
CREATE REPLQMAP SAMPLE_GRAY_TO_SAMPLE2_BROWN USING
ADMINQ "BROWN.QM2.ADMINQ" RECVQ "BROWN.QM1_TO_QM2.DATAQ"
SENDQ "GRAY.QM1_TO_QM2.DATAQ";

# Second queue map
SET SERVER CAPTURE TO DB SAMPLE2 ID DB2ADMIN PASSWORD "passw0rd";
SET CAPTURE SCHEMA SOURCE BROWN;
SET SERVER TARGET TO DB SAMPLE ID DB2ADMIN PASSWORD "passw0rd";
SET APPLY SCHEMA GRAY;
SET OUTPUT CAPTURE SCRIPT "rqm2.sql" TARGET SCRIPT "rqm2.sql";
CREATE REPLQMAP SAMPLE2_BROWN_TO_SAMPLE_GRAY USING
ADMINQ "GRAY.QM1.ADMINQ" RECVQ "GRAY.QM2_TO_QM1.DATAQ"
SENDQ "BROWN.QM2_TO_QM1.DATAQ";

# Third queue map
# The SET SERVER CAPTURE and SET CAPTURE SCHEMA commands for SAMPLE2.BROWN
# are still in effect.
SET SERVER TARGET TO DB SAMPLE3 ID DB2ADMIN PASSWORD "passw0rd";
SET APPLY SCHEMA YELLOW;
SET OUTPUT CAPTURE SCRIPT "rqm3.sql" TARGET SCRIPT "rqm3.sql";
CREATE REPLQMAP SAMPLE2_BROWN_TO_SAMPLE3_YELLOW USING
ADMINQ "YELLOW.QM3.ADMINQ" RECVQ "YELLOW.QM2_TO_QM3.DATAQ"
SENDQ "BROWN.QM2_TO_QM3.DATAQ";

# Fourth queue map
SET SERVER CAPTURE TO DB SAMPLE3 ID DB2ADMIN PASSWORD "passw0rd";
SET CAPTURE SCHEMA SOURCE YELLOW;
SET SERVER TARGET TO DB SAMPLE2 ID DB2ADMIN PASSWORD "passw0rd";
SET APPLY SCHEMA BROWN;
SET OUTPUT CAPTURE SCRIPT "rqm4.sql" TARGET SCRIPT "rqm4.sql";
CREATE REPLQMAP SAMPLE3_YELLOW_TO_SAMPLE2_BROWN USING
ADMINQ "BROWN.QM2.ADMINQ" RECVQ "BROWN.QM3_TO_QM2.DATAQ"
SENDQ "YELLOW.QM3_TO_QM2.DATAQ";

# Fifth queue map
# The SET SERVER CAPTURE and SET CAPTURE SCHEMA commands for SAMPLE3.YELLOW
# are still in effect.
SET SERVER TARGET TO DB SAMPLE ID DB2ADMIN PASSWORD "passw0rd";
SET APPLY SCHEMA GRAY;
SET OUTPUT CAPTURE SCRIPT "rqm5.sql" TARGET SCRIPT "rqm5.sql";
CREATE REPLQMAP SAMPLE3_YELLOW_TO_SAMPLE_GRAY USING
ADMINQ "GRAY.QM1.ADMINQ" RECVQ "GRAY.QM3_TO_QM1.DATAQ"
SENDQ "YELLOW.QM3_TO_QM1.DATAQ";

# Sixth queue map
SET SERVER CAPTURE TO DB SAMPLE ID DB2ADMIN PASSWORD "passw0rd";
SET CAPTURE SCHEMA SOURCE GRAY;
SET SERVER TARGET TO DB SAMPLE3 ID DB2ADMIN PASSWORD "passw0rd";
SET APPLY SCHEMA YELLOW;
SET OUTPUT CAPTURE SCRIPT "rqm6.sql" TARGET SCRIPT "rqm6.sql";
```

```
CREATE REPLQMAP SAMPLE_GRAY_TO_SAMPLE3_YELLOW USING
ADMINQ "YELLOW.QM3.ADMINQ" RECVQ "YELLOW.QM1_TO_QM3.DATAQ"
SENDQ "GRAY.QM1_TO_QM3.DATAQ";

#  2  Ending the ASNCLP session

QUIT;
```

## ASNCLP script 5

This ASNCLP script contains commands for the Q subscriptions between the three servers. You invoke this script by using the LOAD MULTIDIR REPL SCRIPT command in ASNCLP script 6. The script includes commands for the following tasks:

1 Setting the subgroup
2 Setting servers for the subgroup
3 Identifying the matching Q Capture and Q Apply schema at each server
4 Specifying the replication queue maps that connect the three servers
5 Specifying the table to be replicated (one copy at each server)
6 Creating the Q subscriptions

```
# No environment commands are required for this script. These commands are included
# in ASNCLP script 6, which invokes this script.

#  2  Setting the subgroup

SET SUBGROUP "p2p3group";

#  3  Setting servers for the subgroup

SET SERVER MULTIDIR TO DB "SAMPLE";
SET SERVER MULTIDIR TO DB "SAMPLE2";
SET SERVER MULTIDIR TO DB "SAMPLE3";

#  4  Identifying the matching schema of the Q Capture and Q Apply
# control tables at each server

SET MULTIDIR SCHEMA "SAMPLE".GRAY;
SET MULTIDIR SCHEMA "SAMPLE2".BROWN;
SET MULTIDIR SCHEMA "SAMPLE3".YELLOW;

#  5  Specifying the replication queue maps that connect the two servers
# in both directions

SET CONNECTION SOURCE "SAMPLE".GRAY TARGET "SAMPLE2".BROWN REPLQMAP
"SAMPLE_GRAY_TO_SAMPLE2_BROWN";
SET CONNECTION SOURCE "SAMPLE".GRAY TARGET "SAMPLE3".YELLOW REPLQMAP
"SAMPLE_GRAY_TO_SAMPLE3_YELLOW";
SET CONNECTION SOURCE "SAMPLE2".BROWN TARGET "SAMPLE".GRAY REPLQMAP
"SAMPLE2_BROWN_TO_SAMPLE_GRAY";
SET CONNECTION SOURCE "SAMPLE2".BROWN TARGET "SAMPLE3".YELLOW REPLQMAP
"SAMPLE2_BROWN_TO_SAMPLE3_YELLOW";
SET CONNECTION SOURCE "SAMPLE3".YELLOW TARGET "SAMPLE".GRAY REPLQMAP
"SAMPLE3_YELLOW_TO_SAMPLE_GRAY";
SET CONNECTION SOURCE "SAMPLE3".YELLOW TARGET "SAMPLE2".BROWN REPLQMAP
"SAMPLE3_YELLOW_TO_SAMPLE2_BROWN";

#  6  Specifying the table to be replicated (one copy at each server)
# The SET TABLES command specifies only one table, GRAY.STAFF at the SAMPLE
# database. This prompts the command to generate SQL statements to create a
# matching table at the SAMPLE2 database, BROWN.TGTSTAFF, and at the SAMPLE3
# database, YELLOW.TGTSTAFF.

SET TABLES (SAMPLE.GRAY.GRAY.STAFF);
```

```
# 7 Creating the Q subscriptions
# A single CREATE QSUB command generates commands to create six peer-to-peer
# Q subscriptions between SAMPLE, SAMPLE2, and SAMPLE3.

CREATE QSUB SUBTYPE P;

# No QUIT statement is required. The ASNCLP program reads this statement in
# script 5.
```

## ASNCLP script 6

The final script uses the LOAD MULTIDIR REPL SCRIPT command to invoke ASNCLP script 5 for creating the Q subscriptions. It includes commands for the following tasks:

1 Setting the environment
2 Invoking the script that creates Q subscriptions
3 Ending the ASNCLP session

```
# 1 Setting the environment
# The SET OUTPUT MULTIDIR command creates three SQL scripts that are
# automatically named after the databases, SAMPLE.sql, SAMPLE2.sql, and
# SAMPLE3.sql. Run each SQL script at the database for which it is named.
# IMPORTANT: Move or rename the existing SAMPLE.sql, SAMPLE2.sql, and
# SAMPLE3.sql scripts that were generated for creating control tables,
# or the statements for creating Q subscriptions will be appended
# to the end of the files.

ASNCLP SESSION SET TO Q REPLICATION;
SET OUTPUT MULTIDIR;
SET LOG "p2p3-log5.err";
SET RUN SCRIPT LATER;

# 2 Invoking the script that creates Q subscriptions
# Before you run this script, save ASNCLP script 5 in a file, p2p3qsubs.in.

LOAD MULTIDIR REPL SCRIPT "/home/files/asnclp/p2p3qsubs.in";

# 3 Ending the ASNCLP session

QUIT;
```

## Output of the scripts

In addition to the log files, this example produces 12 SQL script files in the same directory where you run the ASNCLP program. Table 6 describes the files.

*Table 6. SQL script files that are created by the sample ASNCLP scripts*

| Output file | Contains SQL to ... |
|---|---|
| SAMPLE.sql | Create Q Capture and Q Apply control tables at the SAMPLE database. |
| SAMPLE2.sql | Create Q Capture and Q Apply control tables at the SAMPLE2 database. |
| SAMPLE3.sql | Create Q Capture and Q Apply control tables at the SAMPLE23 database. |
| rqm1.sql | Define the queue map SAMPLE_GRAY_TO_SAMPLE2_BROWN |
| rqm2.sql | Define the queue map SAMPLE2_BROWN_TO_SAMPLE_GRAY |

Table 6. SQL script files that are created by the sample ASNCLP scripts (continued)

| Output file | Contains SQL to ... |
|---|---|
| rqm3.sql | Define the queue map SAMPLE2_BROWN_TO_ SAMPLE3_YELLOW |
| rqm4.sql | Define the queue map SAMPLE3_YELLOW_TO_ SAMPLE2_BROWN |
| rqm5.sql | Define the queue map SAMPLE3_YELLOW_TO_SAMPLE_GRAY |
| rqm6.sql | Define the queue map SAMPLE_GRAY_TO_SAMPLE3_YELLOW |
| SAMPLE.sql | Add Q subscription definitions to the Q Capture and Q Apply control tables at SAMPLE. |
| SAMPLE2.sql | Add Q subscription definitions to the Q Capture and Q Apply control tables at SAMPLE2. |
| SAMPLE3.sql | Add Q subscription definitions to the Q Capture and Q Apply control tables at SAMPLE3. |

## Shared ASNCLP commands for Q replication and event publishing

Some ASNCLP commands are used for event publishing, unidirectional replication, and multidirectional replication. For example, you use the SET SESSION, CREATE CONTROL TABLES, and SET QMANAGER commands to help set up a Q replication or event publishing configuration.

Table 7 lists the shared commands for Q replication and event publishing and links to topics that describe each command.

Table 7. Shared ASNCLP commands for Q replication and event publishing

| If you want to ... | Use this command |
|---|---|
| Add a column to a Q subscription | "ALTER ADD COLUMN command" on page 74 |
| Change a replication queue map | "ALTER REPLQMAP command" on page 75 |
| Establish a session for Q replication | "ASNCLP SESSION SET TO command (Q replication)" on page 76 |
| Create the control tables for the Q Capture and Q Apply programs | "CREATE CONTROL TABLES FOR command (Q replication)" on page 77 |
| Create a replication queue map | "CREATE REPLQMAP command" on page 82 |
| Drop the control tables for the Q Capture and Q Apply programs | "DROP CONTROL TABLES ON command" on page 83 |
| Delete a replication queue map | "DROP REPLQMAP command" on page 84 |
| Insert a LOADDONE signal into the IBMQREP_SIGNAL table for a manual load | "LOAD DONE command" on page 84 |
| Set the Q Apply schema for all task commands | "SET APPLY SCHEMA command" on page 84 |
| Set the Q Capture schema for all task commands | "SET CAPTURE SCHEMA command" on page 85 |
| Define the log file for the ASNCLP program | "SET LOG command" on page 86 |

*Table 7. Shared ASNCLP commands for Q replication and event publishing (continued)*

| If you want to ... | Use this command |
|---|---|
| Specify custom parameters for database objects to be created implicitly | "SET PROFILE command" on page 87 |
| Set the WebSphere MQ queue manager | "SET QMANAGER command" on page 90 |
| Specify whether to automatically run each task command from an input file before the ASNCLP program processes the next task command | "SET RUN SCRIPT command (Q replication)" on page 91 |
| Specify the server (database) used in the ASNCLP session, authentication information, and other required parameters for connecting to the server | "SET SERVER command (multidirectional Q replication)" on page 129 |
| Enable and disable the trace for the ASNCLP commands | "SET TRACE command" on page 93 |
| Display the environment set during the session | "SHOW SET ENV command" on page 93 |
| Start a Q subscription | "START QSUB command" on page 93 |
| Stop a Q subscription | "STOP QSUB command" on page 94 |
| Verify that the required WebSphere MQ objects exist and have the correct properties for schemas, queue maps, and Q subscriptions. | "VALIDATE WSMQ ENVIRONMENT FOR command" on page 94 |
| Send test messages that validate the message flow between the WebSphere MQ queues that are specified for a replication queue map. | "VALIDATE WSMQ MESSAGE FLOW FOR REPLQMAP command" on page 95 |

## ALTER ADD COLUMN command

Use the ALTER ADD COLUMN command to add a column to a Q subscription.

### Syntax

```
                                     ┌─,───────┐
►►──ALTER ADD COLUMN USING SIGNAL──(──▼─colname─┴── )──┬─QSUB──subname──USING REPQMAP──qmapname─┬──────►◄
                                                       └─XML PUB──pubname───────────────────────┘
```

### Parameters

*colname*
Specifies one or more columns (separated by a comma) to add to the definition of the active Q subscription or XML publication.

**QSUB** *subname*
Specifies the name of the Q subscription.

**USING REPQMAP** *qmapname*
Specifies the name of the replication queue map used by the Q subscription.

**XML PUB** *pubname*
Specifies the name of the XML publication.

### Usage notes

- This command adds a column to the subscribed set of columns on the source table for an active Q subscription or XML publication.
- The column needs to exist in the source table already and should not be part of any existing Q subscription or XML publication.

- The Q subscription or XML publication must be active.
- The column must be nullable or have a default value on the source table.
- The column name on the target table will be implicitly named the same as the column name on the source table.
- For LONG VARCHAR or GRAPHIC types, the DATA CHANGES INCLUDE VARCHAR COLUMNS option must be enabled. VARCHAR COLUMNS are variable length character columns. The DATA CHANGES INCLUDE VARCHAR COLUMNS is an option set on the source table by altering the table's attributes using SQL.
- There is a limit of 20 columns that can be inserted into the statement.

### Example

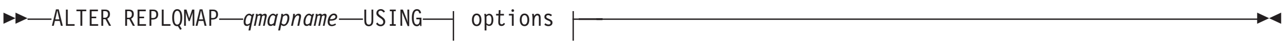
To alter a Q subscription by adding columns PHONE and ADDRESS to the EMPLOYEE0001 Q subscription:

```
ALTER ADD COLUMN USING SIGNAL (PHONE, ADDRESS) QSUB EMPLOYEE0001
USING REPQMAP SAMPLE_ASN_TO_TARGETDB_ASN
```

## ALTER REPLQMAP command

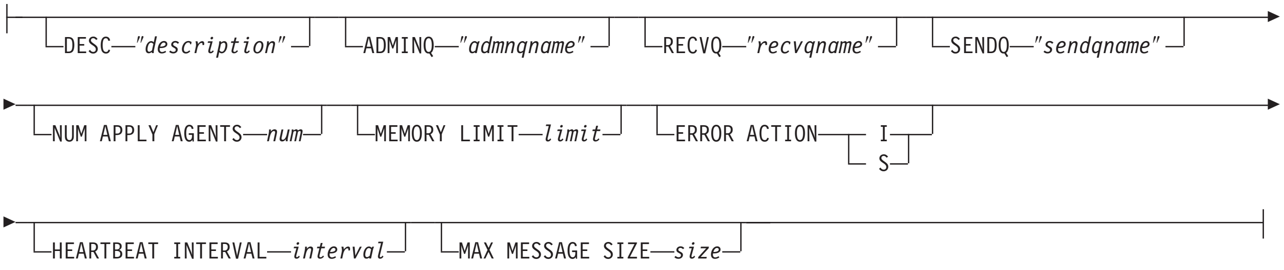
Use the ALTER REPLQMAP command to customize attributes for an existing replication queue map.

### Syntax



```
ALTER REPLQMAP qmapname USING options
```

**options:**



```
DESC "description"
ADMINQ "admnqname"
RECVQ "recvqname"
SENDQ "sendqname"
NUM APPLY AGENTS num
MEMORY LIMIT limit
ERROR ACTION I
             S
HEARTBEAT INTERVAL interval
MAX MESSAGE SIZE size
```

### Parameters

*qmapname*
Specifies the name of the replication queue map.

**DESC** *"description"*
Specifies the description of the replication queue map.

**ADMINQ** *"adminqname"*
Specifies the name of the administration queue at the Q Apply server.

**Note:** If the Q Capture and Q Apply programs share a single queue manager, the two programs may share an administration queue.

**RECVQ** *"recvqname"*
Specifies the name of the receive queue used by the Q Apply program.

**SENDQ** *"sendqname"*
Specifies the name of the send queue used by the Q Capture program. It is used for moving messages to a Q Apply program.

**NUM APPLY AGENTS** *num*
Specifies the number of threads used to concurrently apply transactions from the specified receive queue.

**MEMORY LIMIT** *limit*
Specifies the maximum number of megabytes used per receive queue to buffer incoming transactions.

**ERROR ACTION**
Action to perform if Q Apply fails to apply the change.

**I** The Q Capture program will invalidate all Q subscriptions for the queue in error but keep publishing on the other queues.

**S** The Q Capture program terminates when an error is detected on this queue.

**HEARTBEAT INTERVAL** *interval*
Specifies the interval (in seconds) between heartbeat messages sent by the Q Capture program to both a user application and the Q Apply program when there are no transactions to publish.

**MAX MESSAGE SIZE** *size*
Specifies the maximum size (in kilobytes) of the buffer that is used for sending messages over the send queue.

### Example

To alter the SAMPLE_ASN1_TO_TARGETDB_ASN1 replication queue map and set the threads to 4, set the maximum memory limit to 10 megabytes, stop the Q Capture program if an error occurs, set the heartbeat interval to 4, and set the maximum buffer size to 5 kilobytes:

```
ALTER REPLQMAP SAMPLE_ASN1_TO_TARGETDB_ASN1 USING NUM APPLY AGENTS 4
MEMORY LIMIT 10 ERROR ACTION S HEARTBEAT INTERVAL 4 MAX MESSAGE SIZE 5
```

## ASNCLP SESSION SET TO command (Q replication)

Use the ASNCLP SESSION SET TO command to establish an ASNCLP session for Q replication.

### Syntax

```
►►──ASNCLP SESSION SET TO──Q REPLICATION──────────────────────────────────────────►◄
```

### Parameters

**Q REPLICATION**
Specify to set the ASNCLP session to Q replication. This ASNCLP session only accepts Q replication syntax.

### Usage notes

- Issue the ASNCLP SESSION SET command before all other commands in an ASNCLP session. If you do not issue the ASNCLP SESSION SET command, the ASNCLP program defaults to SQL replication.

- You can only issue commands that apply to the type of replication that you specify.

### Example 1

To set the ASNCLP session to Q replication:

```
ASNCLP SESSION SET TO Q REPLICATION
```

## CREATE CONTROL TABLES FOR command (Q replication)

Use the CREATE CONTROL TABLES FOR command to set up Q Capture and Q Apply control tables. For event publishing, Q Apply control tables are not needed.

For bidirectional and peer-to-peer replication, run the SET MULTIDIR SCHEMA command before you use this command. The Q Capture and Q Apply programs must use the same schema on each server.

### Syntax

```
►►──CREATE CONTROL TABLES FOR──┬─CAPTURE SERVER── USING──┤ capparms-clause ├─┬──────────────────►
                               └─APPLY SERVER─┬─────────────────────────────┬─┘
                                              └─USING──┤ appparms-clause ├──┘

►──┬────────────────────────────────────────────┬──┬──────────────────────────────────┬──────────►◄
   └─IN──┬─ZOS──┤ zos-ts-clause ├──────────┬─┘  └─ZOS──── INDEX──┤ zos-idx-clause ├──┘
         ├─UW──┤ uw-ts-clause ├────────────┤
         └─FEDERATED──┤ fed-ts-clause ├────┘
```

#### capparms-clause:

```
├──RESTARTQ──"rstqname" ──ADMINQ──"admqname" ──┬───────────────────────────┬──┬─────────────────────┬──►
                                               └─STARTMODE──┬─ WARMSI─┬─┘  └─MEMORY LIMIT──limit─┘
                                                            ├─ COLD───┤
                                                            └─ WARMNS─┘

►──┬──────────────────┬──┬─────────────────────────────┬──┬─────────────────────────┬──────────────►
   └─AUTOSTOP──┬─ N─┬─┘  └─MONITOR INTERVAL──interval─┘  └─MONITOR LIMIT──monlimit─┘
               └─ Y─┘

►──┬─────────────────────────┬──┬─────────────────────────┬──┬─────────────────────────────────┬────►
   └─TRACE LIMIT──trclimit──┘  └─SIGNAL LIMIT──siglimit─┘  └─PRUNE INTERVAL──prninterval─┘

►──┬─────────────────────────────────┬──┬──────────────────┬──┬───────────────────┬──┬──────────────┬──►
   └─SLEEP INTERVAL──sleepinterval─┘  └─LOGREUSE──┬─ N─┬─┘  └─LOGSTDOUT──┬─ N─┬─┘  └─TERM──┬─ Y─┬─┘
                                                  └─ Y─┘                └─ Y─┘              └─ N─┘

►──┬───────────────────────────────┬─────────────────────────────────────────────────────────────┤
   └─CAPTURE PATH──"capture_path" ─┘
```

#### apparms-clause:

```
├──┬──────────────────────────┬──┬────────────────────────┬──┬─────────────────────────────┬──────►
   └─MONITOR LIMIT──monlimit─┘  └─TRACE LIMIT──trclimit─┘  └─MONITOR INTERVAL──interval─┘
```

```
►─┬─────────────────────────────┬──┬──────────────────┬──┬───────────────────────────────┬──────►
  └─PRUNE INTERVAL─prninterval─┘  └─AUTOSTOP─┬─N─┬─┘  └─SLEEP INTERVAL─sleepinterval─┘
                                             └─Y─┘

►─┬───────────────────┬──┬────────────────────┬──┬──────────────────────────┬──┬──────────────┬──►
  └─LOGREUSE─┬─N─┬─┘  └─LOGSTDOUT─┬─N─┬─┘  └─APPLY PATH─"apply_path"─┘  └─TERM─┬─Y─┬─┘
             └─Y─┘                └─Y─┘                                        └─N─┘

►─┬─────────────────────┬──┬───────────────────────┬──┬─────────────────────────┬──────────────┤
  └─PWDFILE─"filename"─┘  └─DEADLOCK RETRIES─num─┘  └─SIGNAL LIMIT─siglimit─┘
```

**zos-ts-clause:**

```
├─┬──────────────────────────────────────────────────────────────────────────────┬────────────►
  └─PAGE LOCK─┬──────────────┬─┬─tsname──────────────────┬─┤ prof-clause ├─┘
              └─DB─dbname─┘ └─NAMING PREFIX─prefix─┘

►─┬──────────────────────────────────────────────────────────────────────────────┬────────────┤
  └─ROW LOCK─┬──────────────┬─┬─tsname──────────────────┬─┤ prof-clause ├─┘
             └─DB─dbname─┘ └─NAMING PREFIX─prefix─┘
```

**uw-ts-clause:**

```
├─┬───────────────────────────────────────────────────────────┬───────────────────────────────┤
  └─TBSPACE─┬─tsname──────────────────┬─┤ prof-clause ├─┘
            └─NAMING PREFIX─prefix─┘
```

**fed-ts-clause:**

```
├─┬─────────────────────────┬─RMT SCHEMA─name────────────────────────────────────────────────┤
  └─TBSPACE───tsname───┘
```

**prof-clause:**

```
├─┬────────────────────────────────────────────────┬───────────────────────────────────────────┤
  └─CREATE─┬──────────────────────────────────┬─┘
           └─ USING PROFILE─pname─┘
```

**zos-idx-clause:**

```
├─┬──────────┬─ USING PROFILE─pname──────────────────────────────────────────────────────────┤
  └─CREATE─┘
```

## Parameters

**CAPTURE SERVER**
Specify to create Q Capture control tables.

**APPLY SERVER**
Specify to create Q Apply control tables.

**FEDERATED**
Specify to create control tables in an Oracle, Sybase, Informix, or Microsoft SQL

Server database, and to create nicknames for these control tables in the Q Apply server. Some control tables are created in the Q Apply server.

capparms-clause:

**RESTARTQ** *"rstqname"*
Specifies the restart queue that the Q Capture program uses.

**ADMINQ** *"admqname"*
Specifies the administration queue that the Q Capture program uses.

**STARTMODE**
Specifies what kind of start the Q Capture program will perform.

**WARMSI**
Specify for the Q Capture program to perform a warm start. If the Q Capture program is starting for the first time, it will perform a cold start.

**COLD**
Specify for the Q Capture program to perform a cold start.

**WARMNS**
Specify for the Q Capture program to attempt a warm start if information is available. If the information is not available, the Q Capture program will stop.

**MEMORY LIMIT** *limit*
Specifies the maximum amount (in MB) of memory that the Q Capture program can use to build transactions.

**AUTOSTOP**

**N** The Q Capture or Q Apply program does not stop after it reaches the end of the active log and finds no transactions.

**Y** The Q Capture or Q Apply program stops after it reaches the end of the active log and finds no transactions.

**MONITOR INTERVAL** *interval*
Specifies how frequently (in milliseconds) the Q Capture program inserts rows into the IBMQREP_CAPMON table.

**MONITOR LIMIT** *monlimit*
Specifies how long (in minutes) a row can remain in the IBMQREP_CAPMON and IBMQREP_CAPQMON tables before it becomes eligible for pruning. All rows in these tables that are older than the specified value are pruned at the next pruning cycle.

**TRACE LIMIT** *trclimit*
Specifies how long (in minutes) a row can remain in the IBMQREP_CAPTRACE table before it becomes eligible for pruning. All rows that are older than the specified value are pruned at the next pruning cycle.

**SIGNAL LIMIT** *siglimit*
Specifies how long (in minutes) a row can remain in the IBMQREP_SIGNAL table before it becomes eligible for pruning. All rows that are older than the specified value are pruned at the next pruning cycle.

**PRUNE INTERVAL** *prninterval*
Specifies how frequently (in seconds) the IBMQREP_CAPMON, IBMQREP_CAPQMON, IBMQREP_CAPTRACE, and IBMQREP_SIGNAL tables are pruned.

**SLEEP INTERVAL** *sleepinterval*
Specifies the number of milliseconds that the Q Capture program sleeps when it finishes processing the active log and determines that the buffer is empty.

**LOGREUSE**

**N** The Q Capture program appends messages to the log file, even after the Q Capture program restarts.

**Y** The Q Capture program reuses the log file by first truncating the current log file and then starting a new log when the Q Capture program restarts.

**LOGSTDOUT**

**N** The Q Capture program only sends messages to the log file.

**Y** The Q Capture program sends messages to both the log file and the standard output (stdout).

**TERM**

**Y** The Q Capture program terminates if DB2 stops. This value is the default.

**N** The Q Capture program continues running if DB2 stops with MODE(QUIESCE).

**CAPTURE_PATH** *"capture_path"*
Specifies the location of the work files that the Q Capture program uses. On z/OS systems, the location can be an MVS data set high-level qualifier with //. The default is NULL.

appparms-clause:

**MONITOR LIMIT** *monlimit*
Specifies how long (in minutes) a row can remain in the IBMQREP_APPLYMON table before it becomes eligible for pruning. All rows that are older than the specified value are pruned at the next pruning cycle.

**TRACE LIMIT** *trclimit*
Specifies how long (in minutes) a row can remain in the IBMQREP_APPLYTRACE table before it becomes eligible for pruning. All rows that are older than the specified value are pruned at the next pruning cycle.

**MONITOR INTERVAL** *interval*
Specifies how frequently (in milliseconds) the Q Apply program inserts rows into the IBMQREP_APPLYMON table.

**PRUNE INTERVAL** *prninterval*
Specifies how frequently (in seconds) the IBMQREP_APPLYMON and IBMQREP_APPLYTRACE tables are pruned.

**AUTOSTOP**

**N** The Q Apply program does not stop after all queues are emptied once.

**Y** The Q Apply program stops after all queues are emptied once.

**LOGREUSE**

**N** The Q Apply program appends messages to the log file, even after the Q Apply program is restarted.

**Y** The Q Apply program reuses the log file by first truncating the current log file and then starting a new log when the Q Apply program is restarted.

**LOGSTDOUT**

**N** The Q Apply program sends messages only to the log file.

**Y** The Q Apply program sends messages to the log file and the standard output (stdout).

**APPLY PATH** *"apply_path"*
Specifies the location of the work files the Q Apply program uses. The default path is the directory where the asnqapp command was run.

**TERM**

**Y** The Q Apply program stops if DB2 stops.

**N** The Q Apply program continues running if DB2 stops with MODE(QUIESCE).

**PWDFILE** *"filename"*
Specifies the name of the password file.

**DEADLOCK RETRIES** *num*
Specifies the number of retries for SQL deadlock errors.

zos-ts-clause:

**PAGE LOCK**
Specify for replication control tables that require page-level locking.

**ROW LOCK**
Specify for replication control tables that require row-level locking.

**DB** *dbname*
Specifies the name of the database that contains the table space where the control tables will be created.

*tsname*
Specifies the name of the table space for the z/OS control tables.

**NAMING PREFIX** *prefix*
Specifies a prefix to add to the name of the table space.

uw-ts-clause:

**TBSPACE**

*tsname*
Specifies the name of the table space that is used for the control tables on Linux, UNIX, or Windows.

**NAMING PREFIX** *prefix*
Specifies a prefix to add to the name of the table space.

fed-ts-clause:

**TBSPACE** *tsname*
Specifies the name of an existing Oracle table space, Sybase segment, Informix dbspace, or Microsoft SQL Server file group that is used for the control tables.

**RMT SCHEMA**
The remote schema that the Q Apply program uses to create control tables on the non-DB2 database. The default is the remote authorization ID.

**CREATE**
Specify to create a table space. When this parameter is used without the USING PROFILE keyword, the table space is assumed to exist and the control tables are created in this table space.

**USING PROFILE** *pname*
Specifies the name of a profile to use to customize the table space attributes.

## Example 1

To create Q Apply control tables and to specify a monitor limit of 3 minutes and a trace limit of 9 minutes:

```
CREATE CONTROL TABLES FOR APPLY SERVER USING MONITOR LIMIT 3 TRACE LIMIT 9
```

## Example 2

To create Q Capture control tables:

```
CREATE CONTROL TABLES FOR CAPTURE SERVER USING
RESTARTQ "ASN1.QM1.RESTARTQ" ADMINQ "ASN1.QM1.ADMINQ"
```

## Example 3

To create Q Apply control tables for replication to an Oracle target with a remote authorization ID of ORACLE_ID:

```
CREATE CONTROL TABLES FOR APPLY SERVER IN FEDERATED RMT SCHEMA ORACLE_ID
```

# CREATE REPLQMAP command

Use the CREATE REPLQMAP command to create a replication queue map for Q subscriptions.

## Syntax

```
►►──CREATE REPLQMAP──qmapname──┬──────────────────────────┬──USING ADMINQ──"admnqname" ──────────────────────►
                               └─DESC──"description" ─────┘

►─RECVQ──"recvqname" ──SENDQ──"sendqname" ──┬─────────────────────────┬──┬────────────────────────┬──────────►
                                            └─NUM APPLY AGENTS──num───┘  └─MEMORY LIMIT──limit────┘

►──┬───────────────────────┬──┬──────────────────────────────┬──┬─────────────────────────┬──────────────────►◄
   └─ERROR ACTION──┬─I─┬───┘  └─HEARTBEAT INTERVAL──interval─┘  └─MAX MESSAGE SIZE──size──┘
                   └─S─┘
```

## Parameters

*qmapname*
Specifies the name of the replication queue map.

**DESC** *"description"*
Specifies the description of the replication queue map.

**ADMINQ** *"adminqname"*
Specifies the name of the administration queue at the Q Apply server.

**Note:** If the Q Capture and Q Apply programs share a single queue manager, the two programs may share an administration queue.

**RECVQ** *"recvqname"*
Specifies the name of the receive queue used by the Q Apply program.

**SENDQ** *"sendqname"*
Specifies the name of the send queue used by the Q Capture program.

**NUM APPLY AGENTS** *num*
Specifies the number of threads used for concurrently applying transactions from the specified receive queue.

**MEMORY LIMIT** *limit*
Specifies the maximum number of megabytes used per receive queue for buffering incoming transactions.

**ERROR ACTION**
Action to perform if Q Apply fails to apply the change.

**I** The Q Capture program will invalidate all Q subscriptions for the queue in error but keep publishing on the other queues.

**S** The Q Capture program terminates when an error is detected on this queue.

**HEARTBEAT INTERVAL** *interval*
Specifies the interval (in seconds) between heartbeat messages sent by the Q Capture program to the Q Apply program when there are no transactions to publish.

**MAX MESSAGE SIZE** *size*
Specifies the maximum size (in kilobytes) of the buffer used for sending messages over the send queue.

### Example

To create a replication queue map SAMPLE_ASN1_TO_TARGETDB_ASN1:

```
CREATE REPLQMAP SAMPLE_ASN1_TO_TARGETDB_ASN1 USING ADMINQ "ASN1.QM1.ADMINQ"
RECVQ "ASN1.QM1_TO_QM2.DATAQ" SENDQ "ASN1.QM1_TO_QM2.DATAQ"
```

## DROP CONTROL TABLES ON command

Use the DROP CONTROL TABLES ON command to drop the Q Capture and Q Apply control tables.

### Syntax

```
►►──DROP CONTROL TABLES ON──┬─CAPTURE SERVER─┬──────────────────────────────────────────►◄
                            └─APPLY SERVER───┘
```

### Parameters

**CAPTURE SERVER**
Specify to drop the Q Capture control tables.

**APPLY SERVER**
Specify to drop the Q Apply control tables.

### Usage notes

This command is used in conjunction with the SET SERVER command to indicate the location of the control tables.

### Example

To drop the Q Capture control tables:

```
DROP CONTROL TABLES ON CAPTURE SERVER
```

## DROP REPLQMAP command

Use the DROP REPLQMAP command to delete existing replication queue maps.

**Restriction:** Q subscriptions that use the replication queue map must first be deleted.

### Syntax

```
►►──DROP REPLQMAP──qmapname───────────────────────────────────────────────────────────────►◄
```

### Parameters

*qmapname*
Specifies the name of the replication queue map to delete.

### Example

To delete the SAMPLE_ASN1_TO_TARGETDB_ASN1 replication queue map:

```
DROP REPLQMAP SAMPLE_ASN1_TO_TARGETDB_ASN1
```

## LOAD DONE command

Use the LOAD DONE command to insert a LOADDONE signal into the IBMQREP_SIGNAL table. This allows you to inform the Q Capture program that you have finished loading the target table and that Q Apply can start applying spilled changes to the target table. You need to issue the LOAD DONE command only if you are doing a manual load. If Q Apply is doing the load, this signal is not necessary.

### Syntax

```
►►──LOAD DONE── QSUB──┬─SUBNAME──subname────────────┬──────────────────────────────────────►◄
                      └─FOR SUBNAME LIKE──"%text%"──┘
```

### Parameters

**SUBNAME** *subname*
Specifies the name of the Q subscription for the LOADDONE signal.

**FOR SUBNAME LIKE** *"%text%"*
Specify to signal that the load is done for Q subscriptions that match the expression in the LIKE clause. The following example shows a LIKE clause:

```
LOAD DONE QSUB FOR SUBNAME LIKE "%table%"
```

### Example

To signal the EMPLOYEE0001 Q subscription that a load is done:

```
LOAD DONE QSUB SUBNAME EMPLOYEE0001
```

## SET APPLY SCHEMA command

Use the SET APPLY SCHEMA command to set a default Q Apply schema for all task commands.

### Syntax

```
►►──SET APPLY SCHEMA──┬─TO DEFAULT──┬────────────────────────────────────────────►◄
                      └─applyschema─┘
```

### Parameters

**TO DEFAULT**
Specify to set the Q Apply schema to ASN and to reset any previous SET APPLY SCHEMA commands.

*applyschema*
Specifies the Q Apply schema name.

### Example 1

To reset the default Q Apply schema to ASN:

```
SET APPLY SCHEMA TO DEFAULT
```

### Example 2

To set the default Q Apply schema to ASN1:

```
SET APPLY SCHEMA ASN1
```

## SET CAPTURE SCHEMA command

Use the SET CAPTURE SCHEMA command to set a default source Q Capture schema for all task commands. The default Q Capture schema is ASN. You can use this command to change the default.

This command allows you to omit the Q Capture schema settings in the task commands.

### Syntax

```
►►──SET CAPTURE SCHEMA SOURCE──┬─TO──┬─DEFAULT─┬─┬───────────────────────────────►◄
                               │     └─NULLS───┘ │
                               └─capschema───────┘
```

### Parameters

**SOURCE**
Specifies the Q Capture schema. The schema can be any valid DB2 schema name.

**DEFAULT**
Specify to set the Q Capture schema to ASN and to reset any previous SET CAPTURE SCHEMA commands.

**NULLS**
Specify to set the Q Capture schema to NULL.

*capschema*
Specifies the Q Capture schema name.

### Example 1

To reset the default Q Capture schema to ASN:

```
SET CAPTURE SCHEMA SOURCE TO DEFAULT
```

### Example 2

To set the default Q Capture schema to ASN1:

```
SET CAPTURE SCHEMA SOURCE ASN1
```

## SET LOG command

Use the SET LOG command to define the log file for the ASNCLP session. The log file contains informational, warnings, and errors messages

### Syntax

```
►►──SET LOG──"logfilename"──────────────────────────────────────────────────────►◄
```

### Parameters

*"logfilename"*
Specifies the output log file name. The default log file name is qreplmsg.log.

### Usage notes

- If the files already exist, the ASNCLP program will append to them.
- The double quotation marks in the command syntax are required.

### Example

To name the output log file qmaplog.err for creating replication queue maps:

```
SET LOG "qmaplog.err"
```

## SET OUTPUT command

Use the SET OUTPUT command to define output files for the ASNCLP program. The output files contain the SQL statements needed to set up Q replication and event publishing.

### Syntax

```
►►──SET OUTPUT──┬────────────────────────────┬──┬──────────────────────────┬──────────────►◄
                └─CAPTURE SCRIPT──"capfname"─┘  └─TARGET SCRIPT──"trgfname"─┘
```

### Parameters

**CAPTURE SCRIPT** *"capfname"*
Specifies the output file name for SQL scripts that run at the Q Capture server.

**TARGET SCRIPT** *"trgfname"*
Specifies the output file name for SQL scripts that run at the Q Apply, or target server.

### Usage notes

- If a script already exists, the new script appends to the current script.
- The double quotation marks in the command syntax are required.

### Example 1

To name the target script output file ″target.sql″:

```
SET OUTPUT TARGET SCRIPT "target.sql"
```

## SET PROFILE command

Use the SET PROFILE command to specify custom parameters for table spaces or indexes that are created by the ASNCLP program. After you issue a SET PROFILE command, you can associate a profile with a task command by specifying the profile's name in the task command.

### Syntax

```
►►──SET PROFILE──profilename──┬─┤ prof-clause ├─┬──────────────────────────────────────►◄
                              └─UNDO────────────┘
```

**prof-clause:**

```
├──FOR OBJECT──┬─TARGET──────┬──┬─TABLESPACE OPTIONS──┬─┤ zos-tbls-clause ├─┬─┬───────────┤
               ├─QCNTL TBLS──┤  │                     └─┤ uw-tbs-clause ├───┘ │
               ├─PAGE LOCK───┤  └─INDEX OPTIONS────┤ zos-idx-clause ├─────────┘
               └─ROW LOCK────┘
```

**zos-tbs-clause:**

```
├──ZOS──┬──────────┬──┬────────────────────────────┬──┬─────────────────────────┬──────────────►
        └─DB─dbname┘  └─BUFFERPOOL─bufferpoolname─┘  └─ENCODING──┬─EBCDIC──┬──┘
                                                                 ├─ASCII───┤
                                                                 └─UNICODE─┘

►──┬──────────────────────────────────────────────────────────────────────────────────────────┬──┤
   └─STOGROUP─stogroupname──┬───────────────────────────────────────────┬──┬───────────────────────────────────────────┬─┘
                            └─PRIQTY──┬─ABSOLUTE─n────────────────┬─────┘  └─SECQTY──┬─ABSOLUTE─m────────────────┬─────┘
                                      ├─PERCENT OF SOURCE─n───────┤                  ├─PERCENT OF SOURCE─m───────┤
                                      └─PERCENT OF SOURCE ALLOC─n─┘                  └─PERCENT OF SOURCE ALLOC─m─┘
```

**uw-tbs-clause:**

```
├──UW──┬─────────────────────────────┬──┬─────────────┬──────────────────────────────────────►
       └─BUFFERPOOL─bufferpoolname───┘  └─PAGESIZE─n──┘

►──┬─────────────────────────────────────────────────────────────────┬──────────────────────┤
   └─USING──┬─FILE───┬──"container" ──┬─SIZE─n──┬─PAGES─┬──────────┬─┘
            └─DEVICE─┘                │         ├─KILO──┤          │
                                      │         ├─MEGA──┤          │
                                      │         └─GIGA──┘          │
                                      ├─PERCENT OF SOURCE─n────────┤
                                      └─PERCENT OF SOURCE ALLOC─m──┘
```

### zos-idx-clause:

```
├──ZOS──┬──────────────────────────┬────────────────────────────────────────────────────────────────────►
        └─BUFFERPOOL──bufferpoolname─┘

►──┬──────────────────────────────────────────────────────────────────────────────────────────────────┬──┤
   └─STOGROUP──stogroupname──┬───────────────────────────────────────┬─┬────────────────────────────────────────┬─┘
                             └─PRIQTY──┬─ABSOLUTE──n────────────────┬─┘ └─SECQTY──┬─ABSOLUTE──m────────────────┬─┘
                                       ├─PERCENT OF SOURCE──n───────┤             ├─PERCENT OF SOURCE──m───────┤
                                       └─PERCENT OF SOURCE ALLOC──n─┘             └─PERCENT OF SOURCE ALLOC──m─┘
```

## Parameters

**PROFILE** *profilename*
Specifies the profile name.

**UNDO**
Specify to undo a specific profile.

**FOR OBJECT**
Specifies the object for which you are setting table space or index options:

**TARGET**
Target table

**QCNTL TBLS**
Q replication control tables

**PAGE LOCK**
**z/OS:** All tables that follow the page locking mechanism

**ROW LOCK**
**z/OS:** All tables that follow the row locking mechanism

**TABLESPACE OPTIONS**
Specify to set table space options.

**INDEX OPTIONS**
Specify to set index options.

**DB** *dbname*
Specifies the name of the z/OS database to connect to.

**BUFFERPOOL** *bufferpoolname*
Specifies the buffer pool name.

**ENCODING**
Specifies the encoding scheme (EBCDIC, ASCII, or UNICODE). The default is EBCDIC.

**STOGROUP** *stogroupname*
Specifies a storage group name.

**PRIQTY**
Specifies the minimum primary space allocation for a DB2-managed data set for a table space.

**SECQTY**
Specifies the minimum secondary space allocation for a DB2-managed data set for a table space.

**ABSOLUTE**
Specifies an actual value in kilobytes (denoted as *n* or *m* in the syntax diagram) for space allocation. See the CREATE TABLESPACE command in the *DB2 UDB for z/OS V8 SQL Reference* (SC18-7426-00) for more details.

**PERCENT OF SOURCE**

Specifies the percentage (denoted as *n* or *m* in the syntax diagram) of the source table size for space allocation. See the CREATE TABLESPACE command in the *DB2 UDB for z/OS V8 SQL Reference* (SC18-7426-00) for more details.

**PERCENT OF SOURCE ALLOC**

The number (denoted as *n* or *m* in the syntax diagram) specifies that the space allocation is at least that percentage of the source table allocation (not current space usage) of the related source table in z/OS. If it is used in conjunction with the PRIQTY keyword, the number specifies the minimum primary space allocation. If used in conjunction with the SECQTY keyword, the number specifies the minimum secondary space allocation. See the CREATE TABLESPACE command in the *DB2 UDB for z/OS V8 SQL Reference* (SC18-7426-00) for more details.

**PAGESIZE** *n*

Specifies the page size of the table space.

**Restriction:** The page size of the table space must match the page size of the buffer pool.

**FILE**

Specifies the container path string for the file. For example, for Linux or UNIX you can set the container path to /tmp/db/ts/ and for Windows, you can set the container path to D:\tmp\db\ts\.

**DEVICE**

Specifies the container path string for the device. For example, for Linux or UNIX you can set the container path to /tmp/db/ts/ and for Windows, you can set the container path to D:\tmp\db\ts\.

*"container"*

Specifies the name of the container.

**SIZE** *n*

Specifies the size of the container:

**PAGES**

Actual number of pages

**KILO**

Kilobytes

**MEGA**

Megabytes

**GIGA**

Gigabytes

## Usage notes

- The scope of the profile lasts only as long as the current session. Once you quit the ASNCLP session, the profile information is not saved for the next session.

## Example 1

To create a profile IDXPROFILE that specifies a table space with an 8 kilobytes page size and a 2 gigabyte container for target tables that are created by the ASNCLP program:

```
SET PROFILE IDXPROFILE FOR OBJECT TARGET TABLESPACE OPTIONS UW PAGESIZE 8
USING FILE "container" SIZE 2 GIGA
```

### Example 2

To create a profile TBSPROFILE that sets the index options for tables that follow the page locking mechanism:

```
SET PROFILE TBSPROFILE FOR OBJECT PAGE LOCK INDEX OPTIONS ZOS DB TARGETDB
STOGROUP MYSTOGROUP PRIQTY PERCENT OF SOURCE 70
```

### Example 3

To undo the profile TBSPROFILE:

```
SET PROFILE TBSPROFILE UNDO
```

## SET QMANAGER command

Use the SET QMANAGER command to set the WebSphere MQ queue manager.

### Syntax

```
►►─SET QMANAGER─"mgrname" ─FOR──┬─CAPTURE SCHEMA─────────────────┬──────────────────────────►◄
                                ├─APPLY SCHEMA───────────────────┤
                                └─MULTIDIR─servername.schemaname─┘
```

### Parameters

*"mgrname"*
Specifies the name of the WebSphere MQ queue manager.

**CAPTURE SCHEMA**
Specify to set the queue manager for the Q Capture control tables.

**APPLY SCHEMA**
Specify to set the queue manager for the Q Apply control tables.

**MULTIDIR**
Specify to set the queue manager for the bidirectional or peer-to-peer replication server.

*servername*
Specifies the name of the server (database).

*schemaname*
Specifies the schema of the control tables.

### Example 1

To set the queue manager QM1 for the Q Capture program:

```
SET QMANAGER "QM1" FOR CAPTURE SCHEMA
```

### Example 2

To set the queue manager QM2 for the Q Apply program:

```
SET QMANAGER "QM2" FOR APPLY SCHEMA
```

### Example 3

To set the queue manager QM3 for a server TESTDB.BLUE that is used in bidirectional or peer-to-peer replication:

```
SET QMANAGER "QM3" FOR MULTIDIR TESTDB.BLUE
```

## SET RUN SCRIPT command (Q replication)

Use the SET RUN SCRIPT command to control whether to automatically run SQL statements that are generated by each ASNCLP task command before processing the next command or to manually run them later in a DB2 command prompt.

"Using SET RUN SCRIPT options" helps you understand when to run commands immediately and when to run them later.

### Syntax

```
►►──SET RUN SCRIPT──┬─LATER──────────────────────────┬────────────────────────────────────────────►◄
                    └─NOW──STOP ON SQL ERROR──┬─ON──┬┘
                                              └─OFF─┘
```

### Parameters

**LATER**
Specify to run the SQL scripts at a later time. If you specify to run them later, you must run generated SQL script manually at a DB2 command prompt by using the following command:

```
db2 -tvf <filename>
```

where filename is the name of the SQL script file.

**NOW**
Specify to automatically execute the SQL scripts.

**STOP ON SQL ERROR**
Specifies whether to stop running the SQL scripts if an error occurs.

**ON**
Specify to stop processing the ASNCLP commands when the first SQL statement fails. All previous SQL statements related to this command will be rolled back. If the source scripts run correctly and have been committed, and the target scripts have an error, only the target scripts will be rolled back. The committed source statements will not be rolled back.

**OFF**
Specify to process the ASNCLP commands and run all of the SQL statements, regardless of errors.

### Using SET RUN SCRIPT options

Some ASNCLP CREATE commands require that one or more replication objects exist before the command can be processed. For example, you cannot create Q subscriptions or XML publications until control tables exist.

These dependencies can influence whether you use the NOW or LATER options. In general, the following guidelines apply:

- If you want to create different types of objects in a single ASNCLP script, you are likely to need to use SET RUN SCRIPT NOW.
- If you have multiple ASNCLP scripts, each creating one or more instances of an object, you can use either NOW or LATER. If you use LATER, you are likely to need to run the generated SQL from one ASNCLP script before processing subsequent ASNCLP scripts.

- In some situations, objects of the same type require that SET RUN NOW be used.

Figure 2 shows these dependencies for Q replication.

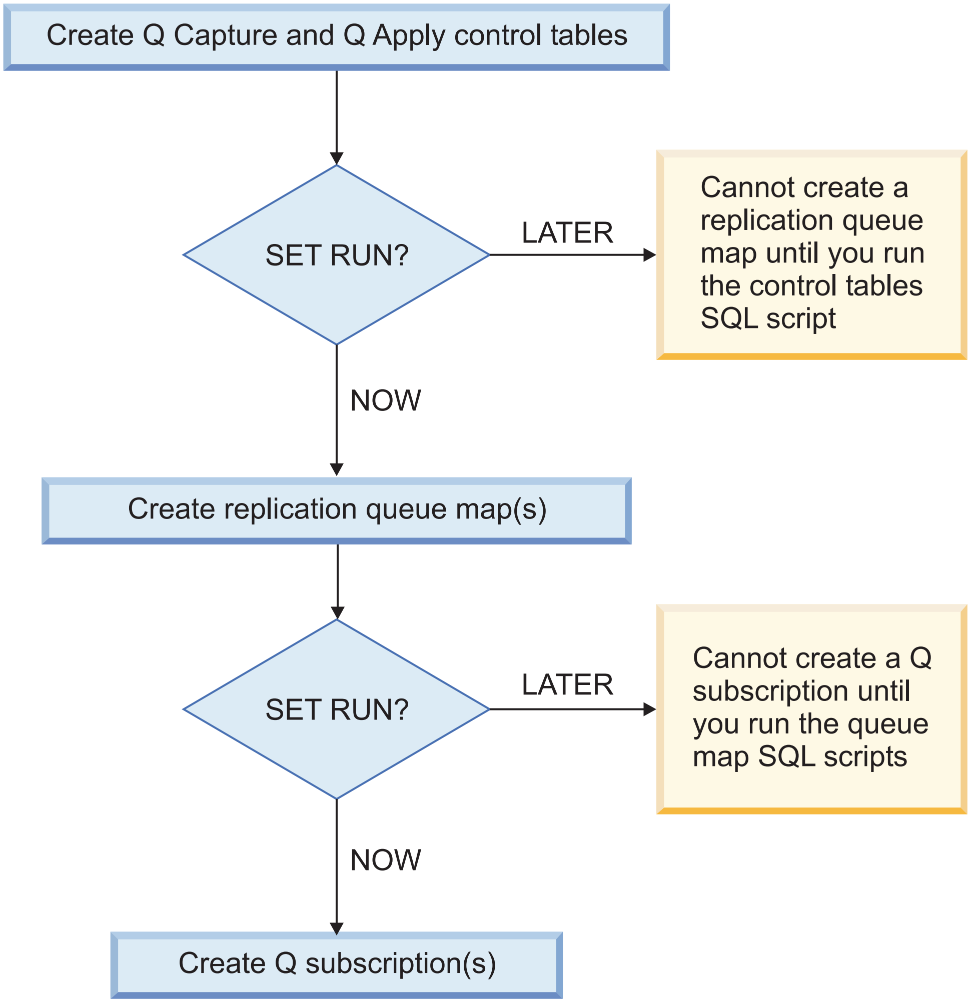

Create Q Capture and Q Apply control tables
SET RUN?
LATER
Cannot create a replication queue map until you run the control tables SQL script
NOW
Create replication queue map(s)
SET RUN?
LATER
Cannot create a Q subscription until you run the queue map SQL scripts
NOW
Create Q subscription(s)

*Figure 2. Dependencies between ASNCLP commands for Q replication.* This diagram shows the dependencies between ASNCLP CREATE commands that are used to set up Q replication. It assumes all objects use the default schema of ASN. The dependencies for Q Capture controls tables, publishing queue maps, and XML publications that are used in event publishing are the same.

## Usage notes

- Use SET RUN SCRIPT LATER when you want to verify the SQL scripts before you run them to create or update your replication configuration.
- Use SET RUN SCRIPT LATER if you want to create SQL script files on one operating system, but run them on another. For example, you might want to run ASNCLP on Windows and copy the SQL script files to z/OS since ASNCLP does not run on the native z/OS operating system.

## Example 1

To run the SQL scripts at a later time:

```
SET RUN SCRIPT LATER
```

### Example 2

To automatically run the SQL scripts but stop processing the ASNCLP commands if an error occurs:

```
SET RUN SCRIPT NOW STOP ON SQL ERROR ON
```

## SET TRACE command

Use the SET TRACE command to enable and disable the internal trace for the ASNCLP commands.

### Syntax

```
►►──SET TRACE──┬─OFF─┬────────────────────────────────────────────────────────────►◄
               └─ON──┘
```

### Parameters

**OFF**
Specify to turn off the trace.

**ON**
Specify to turn on the trace.

### Usage notes

- All output is sent to the console. For readability, save the output to a file.

### Example

To turn on the internal trace for the ASNCLP program:

```
SET TRACE ON
```

## SHOW SET ENV command

The SHOW SET ENV command displays the environment set during the session. The console displays the environment.

### Syntax

```
►►──SHOW SET ENV──────────────────────────────────────────────────────────────────►◄
```

### Example

To display the environment set during an ASNCLP session:

```
SHOW SET ENV
```

## START QSUB command

Use the START QSUB command to start a Q subscription.

### Syntax

```
►►──START── QSUB──┬─SUBNAME──subname───────────┬──────────────────────────────────►◄
                  └─FOR SUBNAME LIKE──"%text%"─┘
```

### Parameters

**SUBNAME** *subname*
Specifies the name of the Q subscription to start.

**FOR SUBNAME LIKE** *"%text%"*
Specify to start Q subscriptions that match the expression in the LIKE clause. The following example shows a LIKE clause:

```
START QSUB FOR SUBNAME LIKE "%table%"
```

### Example

To start a Q subscription:

```
START QSUB SUBNAME EMPLOYEE0001
```

## STOP QSUB command

Use the STOP QSUB command to stop a Q subscription.

### Syntax

```
►►──STOP── QSUB──┬─SUBNAME──subname──────────┬──────────────────────────────────────────────────►◄
                 └─FOR SUBNAME LIKE──"%text%"─┘
```

### Parameters

**SUBNAME** *subname*
Specifies the name of the Q subscription to stop.

**FOR SUBNAME LIKE** *"%text%"*
Specify to stop Q subscriptions that match the expression in the LIKE clause. The following example shows a LIKE clause:

```
STOP QSUB FOR SUBNAME LIKE "%table%"
```

### Example

To stop a Q subscription:

```
STOP QSUB SUBNAME EMPLOYEE0001
```

## VALIDATE WSMQ ENVIRONMENT FOR command

Use the VALIDATE WSMQ ENVIRONMENT FOR command to verify that the required WebSphere MQ objects exist and have the correct properties for Q replication schemas, queue maps, and Q subscriptions.

### Syntax

```
►►──VALIDATE WSMQ ENVIRONMENT FOR──────────────────────────────────────────────────────────────────►

►──┬─CAPTURE SCHEMA──────────────────────────────────────────────────────────────┬──────────────►◄
   ├─APPLY SCHEMA────────────────────────────────────────────────────────────────┤
   ├─PUBQMAP──publishing_queue_map_name──────────────────────────────────────────┤
   ├─REPLQMAP──replication_queue_map_name────────────────────────────────────────┤
   └─QSUB──q_subscription_name──USING REPLQMAP──replication_queue_map_name───────┘
```

### Parameters

**CAPTURE SCHEMA**
Specify to validate the queue manager, restart queue, and administration queue that are defined for a Q Capture schema.

**APPLY SCHEMA**
Specify to validate the queue manager that is defined for a Q Apply schema.

**PUBQMAP**
Specify to validate the send queue that is specified for a publishing queue map.

**REPLQMAP**
Specify to validate the send queue, receive queue, and Q Apply administration queue that are specified for a replication queue map.

**QSUB**
Specify to validate the model queue that is defined to create spill queues for a Q subscription.

### Usage notes

Messages that describe the results of the tests are sent to the standard output (stdout).

### Example 1

To validate the send queue, receive queue, and Q Apply administration queue that are specified for a replication queue map SAMPLE_ASN_TO_TARGET_ASN:

```
VALIDATE WSMQ ENVIRONMENT FOR REPLQMAP SAMPLE_ASN_TO_TARGET_ASN
```

### Example 2

To validate the model queue that is specified for the Q Subscription EMPLOYEE0001 that uses the replication queue map SAMPLE_ASN_TO_TARGET_ASN:

```
VALIDATE WSMQ ENVIRONMENT FOR QSUB EMPLOYEE0001
USING REPLQMAP SAMPLE_ASN_TO_TARGET_ASN
```

## VALIDATE WSMQ MESSAGE FLOW FOR REPLQMAP command

Use the VALIDATE WSMQ MESSAGE FLOW FOR REPLQMAP command to send test messages that validate the message flow between the WebSphere MQ queues that are specified for a replication queue map.

### Syntax

```
►►──VALIDATE WSMQ MESSAGE FLOW FOR REPLQMAP──queue_map_name──────────────────────────────────►◄
```

### Parameters

*queue_map_name*
Specifies the name of an existing replication queue map.

### Usage notes

The command puts a test message on the send queue and attempts to get the message from the receive queue. It also puts a test message on the Q Apply administration queue and attempts to get the message from the Q Capture administration queue. Messages that describe the results of the tests are sent to the standard output (stdout).

### Example

To test the message flow between queues that are part of a replication queue map named SAMPLE_ASN_TO_TARGET_ASN:

```
VALIDATE WSMQ MESSAGE FLOW FOR REPLQMAP SAMPLE_ASN_TO_TARGET_ASN
```

## ASNCLP commands for unidirectional Q replication

The ASNCLP commands for unidirectional Q replication set the Q Capture and Q Apply server, define, change, and delete Q subscriptions, and specify output files.

“Sample ASNCLP scripts for setting up unidirectional Q replication” on page 53 demonstrates how you can combine unidirectional Q replication commands to create an ASNCLP setup script.

Table 8 lists the ASNCLP commands for unidirectional Q replication and links to topics that describe each command.

*Table 8. ASNCLP commands for unidirectional Q replication*

| If you want to ... | Use this command |
|---|---|
| Change a Q subscription | “ALTER QSUB command (unidirectional replication)” |
| Create a Q subscription | “CREATE QSUB command (unidirectional replication)” on page 99 |
| Delete a Q subscription | “DROP QSUB command (unidirectional replication)” on page 109 |
| • Specify whether to drop the target table when you delete a Q subscription<br>• Specify whether to drop the table space when you drop the target table or control tables | “SET DROP command (unidirectional replication)” on page 110 |
| Define output files that contain SQL statements to set up unidirectional Q replication. | “SET OUTPUT command” on page 86 |
| Specify the Q Capture server or Q Apply server to use in the ASNCLP session for unidirectional replication. | “SET SERVER command (Q replication and event publishing)” on page 111 |

### ALTER QSUB command (unidirectional replication)

Use the ALTER QSUB command to change the properties of a Q subscription for unidirectional Q replication.

### Syntax

```
►►──ALTER QSUB──subname──REPLQMAP──mapname──┬──────────────────────────┬──┬────────────────────┬──────►
                                            └─USING REPLQMAP──mapname──┘  └─DESC──description──┘
```

```
►─┬──────────────────────────────────────────┬──────────────────────────────►◄
  └─USING OPTIONS───┬┤ other-opt-clause ├┬──┘
                    └┤ add-cols-clause  ├┘
```

**other-opt-clause:**

```
├──┬──────────────────────────────────────┬──┬──────────────────────┬─────────►
   └─SEARCH CONDITION──"search_condition"─┘  └─ALL CHANGED ROWS──┬─N─┬┘
                                                                 └─Y─┘

►──┬─────────────────────┬──┬─────────────────────┬──┬─────────────────────┬──►
   └─HAS LOAD PHASE──┬─N─┬┘  └─SUPPRESS DELETES──┬─N─┬┘  └─CONFLICT ACTION──┬─I─┬┘
                     ├─I─┤                       └─Y─┘                      ├─F─┤
                     └─E─┘                                                  ├─D─┤
                                                                            ├─S─┤
                                                                            └─Q─┘

►──┬───────────────────┬──┬─────────────────────────┬──┬────────────────┬─────┤
   └─ERROR ACTION──┬─S─┬┘  └─OKSQLSTATES──"sqlstates"─┘  └─LOAD TYPE──┬─0─┬┘
                   ├─D─┤                                            ├─1─┤
                   └─Q─┘                                            ├─2─┤
                                                                    └─3─┘
```

**add-cols-clause:**

```
├──┬────────────────────────────────────────────────┬────────────────────────┤
   │                ┌─,────────────────────────┐     │
   │                ▼                          │     │
   └─ADD COLS── (───trgcolname──┬────────────┬─┴── )─┘
                                └─srccolname─┘
```

## Parameters

**QSUB** *subname*
Specifies the name of the Q subscription.

**REPLQMAP** *mapname*
Specifies the name of the replication queue map for the Q subscription.

**USING REPLQMAP** *mapname*
Specify to alter the Q subscription and to use a different replication queue map.

**DESC** *description*
Specifies a description of the Q subscription.

other-opt-clause:

**SEARCH CONDITION** *"search_condition"*
Specifies a search condition for filtering changes to replicate. The change is not sent if the predicate is false. This is an annotated select WHERE clause, where there must be a colon before the column names of the table to be replicated. The following example shows a WHERE clause:

```
ALTER QSUB myqsub REPLQMAP replqmap10 USING OPTIONS SEARCH CONDITION
"WHERE :MYKEY > 1000"
```

**ALL CHANGED ROWS**
Specifies the data sending option.

**N** Send a row only if a subscribed column in the source table changes.

**Y** Send a row when any column in the source table changes.

**HAS LOAD PHASE**
Specifies whether the target table for the Q subscription will be loaded with data from the source.

**N** No load phase at the target. This is the default.

**I** Specifies an automatic load. The Q Apply program calls the LOAD from CURSOR utility, EXPORT/IMPORT utility, or EXPORT/LOAD utility, depending on the type of load that is specified in the LOAD_TYPE keyword, and on the platform of the Q Apply server and Q Capture server. This option is not valid for Q subscriptions that specify stored procedures as targets.

**E** Specifies a manual load. An application other than the Q Apply program loads the target table. In this case, you insert the LOADDONE signal (using the LOADDONE command) into the IBMQREP_SIGNAL table at the Q Capture server to inform the Q Capture program that the application is done loading.

**SUPPRESS DELETES**
Specifies whether to send rows that were deleted from the source table.

**N** Send deleted rows.

**Y** Do not send deleted rows.

**CONFLICT ACTION**
Specifies what action to take if a conflict occurs.

**I** Ignore.

**F** Force: This action requires the send option **CHANGED COLS ONLY = 'N'**.

**D** Disable the Q subscription.

**S** Stop Q Apply.

**Q** Stop reading from queue.

**ERROR ACTION**
Specifies what action to take if an error occurs.

**S** Stop Q Apply without applying the transaction.

**D** Disable subscription and notify Q Capture.

**Q** Stop reading from queue.

**OKSQLSTATES** *"sqlstates"*
Specifies a list of SQL statements within double quotation marks that are not to be considered as errors when applying changes to this table.

**LOAD TYPE**
Specifies a type of load.

**0** Choose the best type automatically.

**1** Use LOAD from CURSOR only.

**2** Use EXPORT and IMPORT only.

**3** Use EXPORT and LOAD only.

**ADD COLS** *(trgcolname srccolname)*

Specify to add one or more columns to the Q subscription. If *trgcolname* and *srccolname* are the same, only specify the *trgcolname*.

## Example 1

To alter a Q subscription for unidirectional replication and change the load type to an automatic load, send deleted rows, and stop reading from the queue if an error occurs:

```
ALTER QSUB EMPLOYEE0001 REPLQMAP SAMPLE_ASN1_TO_TARGETDB_ASN1
USING OPTIONS ALL CHANGED ROWS N HAS LOAD PHASE I
SUPPRESS DELETES N CONFLICT ACTION F ERROR ACTION Q LOAD TYPE 1
```

## Example 2

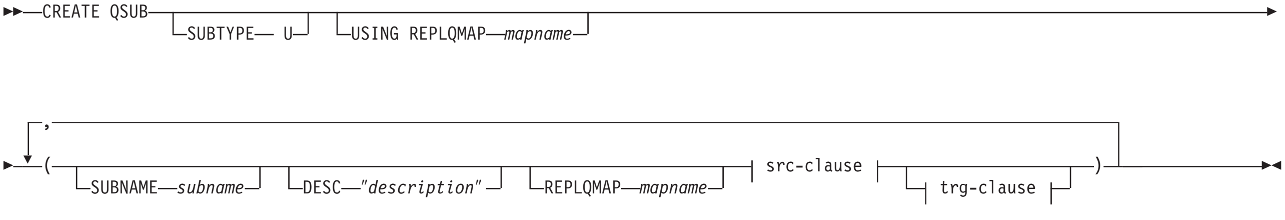
To alter a Q subscription for unidirectional replication by adding two columns that you want to begin replicating from the source table:

```
ALTER QSUB EMPLOYEE0001 REPLQMAP SAMPLE_ASN1_TO_TARGETDB_ASN1
USING OPTIONS ADD COLS (BONUS,COMM)
```

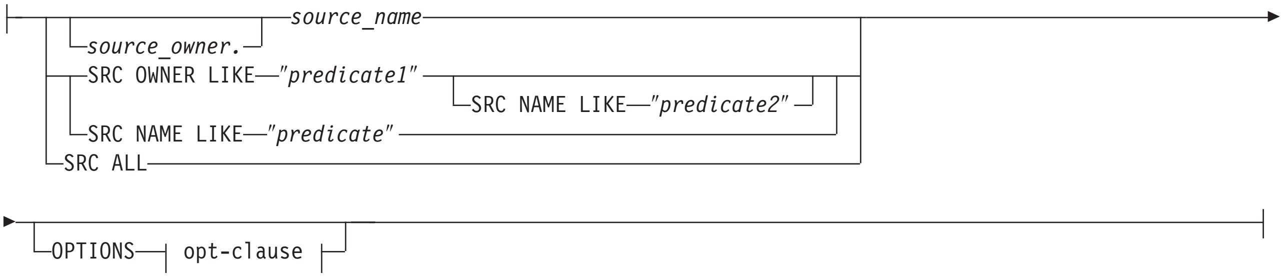
# CREATE QSUB command (unidirectional replication)

Use the CREATE QSUB command to create a Q subscription for unidirectional replication.

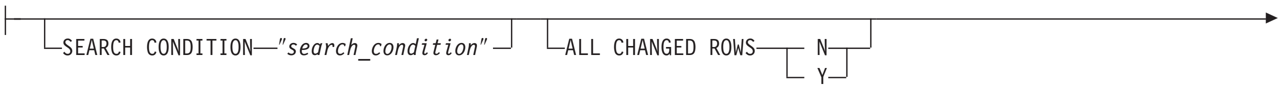
## Syntax



**src-clause:**


source_owner.
source_name
SRC OWNER LIKE "predicate1"
SRC NAME LIKE "predicate2"
SRC NAME LIKE "predicate"
SRC ALL
OPTIONS
opt-clause

**opt-clause:**



```
►─┬─────────────────────────────────────────────────┬──┬────────────────────────────┬──────────────┤
  └─HAS LOAD PHASE──┬─ N ───────────────────────────┤  └─SUPPRESS DELETES──┬─ N─┬──┘
                    └─┬─ I─┬──┬────────────────────┬┘                      └─ Y─┘
                      └─ E─┘  └─SPILL_MODELQ──name─┘
```

**trg-clause:**

```
├──┬───────┬──┬─────────────────────────────────────────┬─────────────────────────────────────────►
   └─EXIST─┘  └─TARGET─┬─────────────────────────────────┤
                       ├─NAME──┬────────────┬─target_name─┤
                       │       └─owner──.──┘             │
                       └─NAMING PREFIX──prefix────────────┘

►─┬─────────────────────────────────────────┬──┬──────────────────────────────────────┬──────────►
  └─TABLE OWNER──┬─target_owner─────────────┤  └─TABLE NAME──┬─target_name──────────┤
                 ├─NAMING PREFIX──prefix────┤                ├─NAMING PREFIX──prefix─┤
                 ├─SAME AS SOURCE───────────┤                └─SAME AS SOURCE────────┘
                 └─SAME AS USERID───────────┘

►─┬────────────────────────────────┬──────────────────────────────────────────────────────────────►
  └─FEDERATED──┤ fed-clause ├──────┘

►─┬─────────────────────────────────────────────────────────────────────────────────┬─────────────►
  └─IN──┬─────────────┬──┬─tsname────────────────┬──┬──────────────────────┬────────┘
        └─DB──name────┘  └─NAMING PREFIX──prefix─┘  └─┤ prof-clause ├──────┘

►─┬──────────────────────────────────────┬────────────────────────────────────────────────────────►
  └─TYPE───┬─USERTABLE──────────────────┤
           ├─STOREDPROC─────────────────┤
           ├─NICKNAME───────────────────┤
           └─CCD──┤ ccd-clause ├────────┘

►─┬────────────────────────────────────────────────────────────────────────┬──────────────────────►
  └─TRGCOLS──┬─ALL─────────────────────────────────────────────────────────┤
             │                  ┌─,─────────────────────────────┐          │
             │                  ▼                               │          │
             ├─INCLUDE── (──────trgcolname──┬────────────────┬──┴─── )─────┤
             │                              └─srccolname─────┘             │
             │                  ┌─,─────────────┐                          │
             │                  ▼               │                          │
             └─EXCLUDE── (──────trgcolname──────┴─── )────────────────────┘

►─┬────────────────────────────────────────────────────────────────────────────┬──────────────────►
  │                                                  ┌─,─────────────────┐     │
  │                                                  ▼                   │     │
  └─KEYS──┬──────────────────────────────┬─── (──────keyname──┬──────┬───┴── )─┘
          ├─indexowner.indexname─────────┤                    ├─ +───┤
          └─NAMING PREFIX──prefix────────┘                    └─ -───┘

►─┬──────────────────────────────────────────┬────────────────────────────────────────────────────►
  └─ZOS INDEX CREATE USING PROFILE──pname────┘
```

```
►─┬──────────────────────────────────────────────────────────────┬─┬─────────────────────────┬──────────►
  ├─NICKNAME──┬─owner.nickname──────────────────────────────────┤ └─CONFLICT ACTION──┬─ I─┬─┘
  │           └─NAMING PREFIX──prefix─┘                         │                   ├─ F─┤
  └─NEW NICKNAME RMT SERVERNAME──srvname──┬─owner.nickname──────┘                   ├─ D─┤
                                          └─NAMING PREFIX──prefix─┘                 ├─ S─┤
                                                                                    └─ Q─┘

►─┬────────────────────┬─┬──────────────────────────────┬─┬───────────────────┬──────────────────────┤
  └─ERROR ACTION──┬─ S─┬─┘ └─OKSQLSTATES──"sqlstates"──┘ └─LOAD TYPE──┬─ 0─┬─┘
                  ├─ D─┤                                              ├─ 1─┤
                  └─ Q─┘                                              ├─ 2─┤
                                                                      └─ 3─┘
```

**fed-clause:**

```
├──┬──────────────────────────┬─nickname target name────────────────────────────────────────────────┤
   └─nickname target owner.──┘
```

**ccd-clause:**

```
├──┬─────────────────────┬─┬─────────────────────┬─┬────────────────────────────────────────┬────────┤
   └─CONDENSED──┬─ON──┬──┘ └─COMPLETE──┬─ON──┬──┘ └─WITH UOW COLS──┬─ALL────────────────────┤
                └─OFF─┘                └─OFF─┘                     │    ┌─,───────────┐      │
                                                                   │    ▼             │      │
                                                                   └─(───colname──────┴──)─┘
```

**prof-clause:**

```
├──┬─────────────────────────────────────────────┬──────────────────────────────────────────────────┤
   └─CREATE──┬─────────────────────────────┬─────┘
             └─USING PROFILE──pname────────┘
```

## Parameters

**SUBTYPE U**
Specifies unidirectional replication.

**USING REPLQMAP** *mapname*
Specifies the name of the replication queue map that is used by all of the Q subscriptions in this command. This is the replication queue map that will be used by all of the Q subscriptions in a mass scenario, or if replication queue maps are not specified with the parenthesis for each Q subscription.

**SUBNAME** *subname*
Specifies the name of the Q subscription.

**DESC** *"description"*
Specifies a description of the Q subscription.

**REPLQMAP** *mapname*
Specifies the name of the replication queue map for the Q subscription.

src-clause:

*source_owner***.***source_name*
Specifies the source table's schema and name.

**SRC OWNER LIKE** *"predicate1"*

Specify to choose all tables with a schema that matches the expression in the LIKE statement. The following example shows a LIKE statement:

```
CREATE QSUB USING REPLQMAP ABCDPUBQMAP
(SRC OWNER LIKE "ASN%");

CREATE QSUB USING REPLQMAP ABCDPUBQMAP
(SRC OWNER LIKE "JDOE" SRC NAME LIKE "%TAB%");
```

**SRC NAME LIKE**

Specify to choose all tables with a name that matches the expression in the LIKE statement. The following example shows a LIKE statement:

```
CREATE QSUB USING REPLQMAP ABCDPUBQMAP
(SRC OWNER LIKE "ASN%");

CREATE QSUB USING REPLQMAP ABCDPUBQMAP
(SRC OWNER LIKE "JDOE" SRC NAME LIKE "%TAB%");
```

**SRC ALL**

Specify to choose all tables, with the exception of DB2 catalog views, that exist on the Q Capture server.

opt-clause:

**SEARCH CONDITION***"search_condition"*

Specifies a search condition for filtering changes to replicate. The change is not sent if the predicate is false. This is an annotated select WHERE clause, where there must be a colon before the column names of the table to be replicated. The following example shows a WHERE clause:

```
CREATE QSUB USING REPLQMAP ASNMAP
(SUBNAME mysubname ALLTYPE1 OPTIONS SEARCH CONDITION
"WHERE :MYKEY > 1000")
```

**ALL CHANGED ROWS**

Specifies the data sending option.

**N** Send a row only if a subscribed column in the source table changes.

**Y** Send a row when any column in the source table changes.

**HAS LOAD PHASE**

Specifies whether the target table for the Q subscription will be loaded with data from the source.

**N** No load phase at the target. This is the default.

**I** Specifies an automatic load. The Q Apply program calls the LOAD from CURSOR utility, EXPORT/IMPORT utility, or EXPORT/LOAD utility, depending on the type of load that is specified in the LOAD_TYPE keyword, and on the platform of the Q Apply server and Q Capture server. This option is not valid for Q subscriptions that specify stored procedures as targets.

**E** Specifies a manual load. An application other than the Q Apply program loads the target table. In this case, you use the LOADDONE command to insert the signal into the IBMQREP_SIGNAL table at the Q Capture server to inform the Q Capture program that the application is done loading.

**SPILL_MODELQ** *name*

Specifies the name of the model queue that is used as a spill queue for this Q subscription. On z/OS, you might want to create separate spill queues for Q subscriptions if the page set for the model queue is not large enough to handle transactions from multiple Q subscriptions during a load.

**SUPPRESS DELETES**
Specifies whether to send rows that were deleted from the source table.

**N** Send deleted rows.

**Y** Do not send deleted rows.

trg-clause:

**EXIST**
Specifies that the target table exists.

- If you specify **EXIST** but do not provide a target table name, the ASNCLP program will look for the default table *TGT-<SOURCE TABLE NAME>*.
- If you specify **EXIST** and a single **TARGET NAME**, and you use **SOURCE ALL** or **SOURCE NAME LIKE**, then all of the source tables will be mapped to that single specified existing target table.
- If you do not specify **EXIST**, and you use **SOURCE ALL** or **SOURCE NAME LIKE**, then the source tables will be paired with target tables that use the default name *TGT-<SOURCE TABLE NAME>*.

**TARGET**
Specifies options for the target table owner and name.

**NAME** *target_owner.target_name*
Specifies the target table's name and optionally the table schema.

**NAMING PREFIX**
Specifies the prefix to use to name the target table. The default is TGT. You can specify any other prefix, for example, if you specify CLP as a prefix and the source table is T1, the target table would be called CLPT1.

**TABLE OWNER**
Specifies options for the target table owner.

*target_owner*
Specifies to use the schema of the target table.

**NAMING PREFIX**
Specifies the prefix to use to name the target table. The default is TGT. You can specify any other prefix, for example, if you specify CLP as a prefix and the source table is T1, the target table would be called CLPT1.

**SAME AS SOURCE**
Specifies to use the same owner as the corresponding source table.

**SAME AS USERID**
Specifies to use the current user ID.

**TABLE NAME**
Specifies options for the target table name.

*target_name*
Specifies the name that you want to use for the target table.

**NAMING PREFIX**
Specifies the prefix to use to name the target table. For example, if you specify CLP as a prefix and the source table is T1, the target table would be called CLPT1.

**SAME AS SOURCE**
Specifies to name the target table the same as the corresponding source table.

**IN**

**DB** *name*
Specifies the name of the logical database for the table space (required for z/OS).

*tsname*
Specifies the name of the table space for the target table.

**Federated targets:** Specifies an existing table space (Oracle), segment (Sybase), dbspace (Informix), or file group (Microsoft SQL Server).

**NAMING PREFIX** *prefix*
Specifies the prefix to use to name the table space.

**TYPE**

**USERTABLE**
Specifies a table as the target.

**STOREDPROC**
Specifies a stored procedure as the target.

**NICKNAME**
Specifies a nickname as the target.

**CCD**
Specifies a consistent-change data (CCD) table as the target.

**TRGCOLS**

**ALL**
Specify to replicate all columns from the source table.

**INCLUDE**
Specifies the column definitions if the target table does not exist. Specifies the columns in the target table that will be replicated.

*trgcolname*
Specify to add a column definition to the target table that uses the provided name and the properties of a source column with the same name. In the following example, both the source and target table have the columns *one*, *two*, and *three*.

```
CREATE QSUB SUBTYPE U USING REPLQMAP replqmap9
(SUBNAME sub9 dpropr64.srctable
EXIST TARGET NAME dpropr64.trgtable
TRGCOLS INCLUDE (one, two))
```

*srccolname*
Specify to add a column definition to the target table that uses the provided name and to use the properties of the source column for the target column properties.

**EXCLUDE**
Specify to exclude the source column from the target table definition. This keyword can be used only when the source and target tables have the same column names, or when you are creating a new target table. The following example shows how you would use the CREATE QSUB command with this option. In the example, the source table columns are *one*, *two*, and *three*.

```
CREATE QSUB USING REPLQMAP replqmap10
(SUBNAME sub10 dpropr64.srctable exist target name
dpropr64.tgttable trgcols
exclude(three))
```

*trgcolname*
Specify to exclude a column definition from the target table that uses the provided name and the properties of a source column with the same name.

**KEYS**
Specifies one or more key columns that replication uses to determine the uniqueness of a row. If no key is specified, replication tries to determine its own key by looking first for a primary key within the set of replicated columns, then for a unique constraint, and then for a unique index. If none of these exists, replication will use all subscribed, valid columns as key columns for replication. (Some subscribed columns, such as LOB columns, cannot be used as keys.)

*indexowner***.***indexname*
Specifies the index owner and name.

**NAMING PREFIX** *prefix*
Specifies the prefix to use to name the index.

*keyname*
Specifies the name of the columns that are included in the index.

**+** Ascending order.

**-** Descending order.

**ZOS INDEX CREATE USING PROFILE** *pname*
Specifies the name of the index profile for customizing a z/OS index.

**NICKNAME**
Specifies the nickname for the Q Apply program to use to load rows into the target table with the LOAD from CURSOR utility.

*owner***.***nickname*
Specifies the source owner and nickname.

**NAMING PREFIX** *prefix*
Specifies the prefix to use to name the nickname.

**NEW NICKNAME RMT SERVERNAME** *srvname*
Specifies the name of the remote server if the ASNCLP program creates the nickname for loading.

**CONFLICT ACTION**
Specifies what action to take if a conflict occurs.

**I** Ignore.

**F** Force: This action requires the send option **CHANGED COLS ONLY = ’N’**.

**D** Disable the Q subscription.

**S** Stop Q Apply.

**Q** Stop reading from queue.

**ERROR ACTION**
Specifies what action to take if an error occurs.

**S** Stop Q Apply without applying the transaction.

**D** Disable subscription and notify Q Capture.

**Q** Stop reading from the receive queue.

**OKSQLSTATES** *"sqlstates"*
Specifies a list of SQL statements within double quotation marks that are not to be considered as errors when applying changes to this table.

**LOAD TYPE**
Specifies a type of load.

**0** Choose the best type automatically.

**1** Use LOAD from CURSOR only.

**2** Use EXPORT/IMPORT only.

**3** Use EXPORT/LOAD only.

fed-clause

*nickname target owner*
If you specify the FEDERATED keyword, you can optionally provide an owner for the nickname that is created for a federated target.

*nickname target name*
If you specify the FEDERATED keyword, you can optionally provide a name for the nickname that is created for a federated target.

ccd-clause

**CONDENSED**
Specify one of the following values:

**ON** Specifies that the CCD table is condensed. A condensed CCD table contains one row for every key value in the source table and contains only the latest value for the row.

**OFF** Specifies that the CCD table is noncondensed. A noncondensed CCD table contains multiple rows with the same key value, one row for every change that occurs to the source table.

**COMPLETE**
Specify one of the following values:

**ON** Specifies that the CCD table is complete. A complete CCD table contains every row of interest from the source table and is initialized with a full set of source data.

**OFF** Specifies that the CCD table is noncomplete. A noncomplete CCD table contains only changes to the source table and starts with no data.

**WITH UOW COLS**
Specify one of the following values:

**ALL** Specifies that the CCD table contains all four unit-of-work (UOW) columns: IBMSNAP_AUTHID, IBMSNAP_AUTHTKN, IBMSNAP_PLANID, IBMSNAP_UOWID.

*colname*
Specify one or more unit-of-work (UOW) columns for the CCD table.

prof-clause:

**CREATE**
Specify to create a table space.

**USING PROFILE** *pname*
Specifies the name of the profile to use to create the table space.

## Usage notes

- The **REPLQMAP** keyword is mandatory. You can specify either `CREATE QSUB USING REPLQMAP` *mapname* or `CREATE QSUB (SUBNAME subname REPLQMAP` *mapname*`)`.
- If a target table is specified and **SRC ALL** or **SRC NAME LIKE** was specified, all the source tables will attempt to subscribe to target tables with the same name.
- If the **TABLE OWNER** or **TABLE NAME** keywords are not specified, the default owner is the owner of the corresponding source table and the default name is *TGT-<SOURCE TABLE NAME>*
- DB value for Logical Database is mandatory for target tables on z/OS platforms. It must be specified in the profile.
- If a mass subscription is used (for example, using the **SRC OWNER LIKE** or **SRC NAME LIKE** clause) the specified *target_owner*.*target_name* clause is not valid if the target table does not exist. Only default or a naming prefix are allowed for generated target tables.

## Example 1

The following example shows the commands that are needed to set the environment and profiles for a CREATE QSUB command for unidirectional replication. In this example, both the Q Capture program and Q Apply program run in the same z/OS subsystem and share a queue manager.

```
ASNCLP SESSION SET TO Q REPLICATION;

SET SERVER CAPTURE to dbALIAS EC06V71A DBNAME stlec1 ID ADMF001  password "xx";
SET SERVER TARGET to dbALIAS EC06V71A DBNAME stlec1 ID ADMF001  password "xxx";

SET CAPTURE SCHEMA SOURCE QDECODER;
SET APPLY SCHEMA QDECODER;

SET QMANAGER "CSQ1" FOR CAPTURE SCHEMA;
SET QMANAGER "CSQ1" FOR APPLY SCHEMA;

SET PROFILE "UITRGTS" FOR OBJECTS TARGET INDEX OPTIONS ZOS
  BUFFERPOOL BP1  STOGROUP "DPROSTGQ"
  PRIQTY ABSOLUTE 100 SECQTY ABSOLUTE 50;

SET PROFILE "UTRGTS" FOR OBJECT TARGET TABLESPACE OPTIONS ZOS
   DB "JUTRGDB"
  BUFFERPOOL BP4
   ENCODING UNICODE
   STOGROUP "DPROSTG"
   PRIQTY ABSOLUTE 100  SECQTY ABSOLUTE 50;

SET OUTPUT CAPTURE SCRIPT "capfile6.sql" TARGET SCRIPT "tgtfile.sql";
SET RUN SCRIPT NOW STOP ON SQL ERROR ON;
```

## Example 2

This example demonstrates the use of a naming prefix for the target table (XNEW) and tablespace for the target table (Y). The example also shows the use of ″like″ statements to specify the source table for the Q subscription.

```
CREATE QSUB USING REPLQMAP QDECODERQM (SRC OWNER LIKE "DSN8710%" SRC NAME LIKE
"%EMP%" TARGET TABLE NAME NAMING PREFIX XNEW IN DB D1CDG01 NAMING PREFIX Y);
```

## Example 3

This example shows how to use a tablespace profile (USING PROFILE UTRGTS) for the target table tablespace when the target tables do not exist.

```
CREATE QSUB USING REPLQMAP QDECODERQM (SRC OWNER LIKE "DSN8710%" SRC NAME LIKE
"%EMP%" TARGET TABLE NAME NAMING PREFIX XNEW2 IN DB D1CDG01 EMPTBSP2 CREATE USING
PROFILE UTRGTS);
```

## Example 4

This example shows that no IN clause is required when the target table exists.

```
CREATE QSUB USING REPLQMAP QDECODERQM (SRC OWNER LIKE "DSN8710%" SRC NAME LIKE
"%EMP%" EXIST TARGET TABLE OWNER NAMING PREFIX X);
```

## Example 5

This example creates all of the target tables in one tablespace (RST1).

```
CREATE QSUB USING REPLQMAP QDECODERQM (SRC OWNER LIKE "DSN8710%" SRC NAME LIKE
"%EMP%" TARGET TABLE NAME XNEW IN DB D1CDG01 RTS1);
```

## Example 6

In this example, the target table exists, the target owner is ABC, and target table prefix is XNEW.

```
CREATE QSUB USING REPLQMAP QDECODERQM (SRC OWNER LIKE "DSN8710%" SRC NAME LIKE
"%EMP%" TARGET TABLE OWNER ABC TABLE NAME NAMING PREFIX XNEW );
```

## Example 7

This example shows the use of a target owner prefix (ABC).

```
CREATE QSUB USING REPLQMAP QDECODERQM (SRC OWNER LIKE "DSN8710%" SRC NAME LIKE
"%EMP%" TARGET TABLE OWNER NAMING PREFIX ABC TABLE NAME NAMING PREFIX XNEW );
```

## Example 8

In this example the source and target owner names are the same. For this to occur, the target must be in a different database or subsystem than the source.

```
CREATE QSUB USING REPLQMAP QDECODERQM (SRC OWNER LIKE "DSN8710%" SRC NAME LIKE
"%EMP%" TARGET TABLE OWNER SAME AS SOURCE TABLE NAME SAME AS SOURCE );
```

## Example 9

This example does not use the environment and profile from "Example 1" on page 107. It creates a Q subscription for unidirectional replication using the replication queue map SAMPLE_ASN1_TO_TARGETDB_ASN1 and specifies that the Q Apply program loads the target tables using the EXPORT and IMPORT utilities. It also that the column EMPNO be used as the key for replication.

```
CREATE QSUB USING REPLQMAP SAMPLE_ASN1_TO_TARGETDB_ASN1
(SUBNAME EMPLOYEE0001 EMPLOYEE OPTIONS HAS LOAD PHASE I
TARGET NAME TGTEMPLOYEE KEYS (EMPNO) LOAD TYPE 2);
```

## Example 10

This example creates a Q subscription with the following characteristics:

- From a DB2 table (EMPLOYEE)
- To a Sybase table (TGT_EMPLOYEE) that the command will create

- In the existing Sybase segment SEG_EMPLOYEE
- Using the SAMPLE_ASN_TO_FEDDB_ASN replication queue map
- Specifying a nickname name of EMPNICKNAME

```
CREATE QSUB USING REPLQMAP SAMPLE_ASN_TO_FEDDB_ASN (SUBNAME FEDQSUB
EMPLOYEE TARGET NAME TGTEMPLOYEE FEDERATED EMPNICKNAME);
```

## DROP QSUB command (unidirectional replication)

Use the DROP QSUB command to delete a Q subscription for unidirectional Q replication

### Syntax

```
►►──DROP──┬─────┬──QSUB──┬───────────────────────────┬──────────────────────────────────►
          └─ALL─┘        └─USING REPLQMAP──mapname───┘

►──┬───────────────────────────────────────────────────────┬───────────────────────────►◄
   │  ┌─,─────────────────────────────────────────────┐   │
   │  ▼                                               │   │
   ├──(──SUBNAME──subname──┬──────────────────────────┬──)──┤
   │                       └─USING REPLQMAP──mapname──┘     │
   └─FOR SUBNAME LIKE──"predicate"─────────────────────────┘
```

### Parameters

**ALL**
Specify to delete all Q subscriptions.

**USING REPLQMAP** *mapname*
Specify to delete all of the Q subscriptions that use the specified replication queue map.

**SUBNAME** *subname*
Specifies the name of the Q subscription to delete.

> **USING REPLQMAP** *mapname*
> Specifies the name of the replication queue map that is used by the Q subscription that you want to delete.

**FOR SUBNAME LIKE** *"predicate"*
Specify to delete all of the Q subscriptions that match the expression in the LIKE statement. The following example shows a LIKE statement:

```
DROP QSUB USING REPLQMAP ABCDREPLQMAP
(FOR SUBNAME LIKE "ASN%");
```

### Usage notes

The keyword **ALL** can only be used alone and cannot be combined with any other options.

### Example

To delete a Q subscription for unidirectional replication:

```
DROP QSUB (SUBNAME EMPLOYEE0001 USING REPLQMAP SAMPLE_ASN1_TO_TARGETDB_ASN1)
```

## SET DROP command (unidirectional replication)

Use the SET DROP command to specify whether to drop the target table and its table space when you delete a Q subscription for unidirectional replication. You also use this command to specify whether to drop the table spaces for control tables.

### Syntax

```
►►─SET DROP TARGET──┬─NEVER──┬────────────────────────────────────────────────────────►◄
                    └─ALWAYS─┘
```

```
►►─SET DROP──┬─TARGET─────────┬─TABLESPACE──┬─WHEN EMPTY─┬────────────────────────────►◄
             └─CONTROL TABLES─┘             └─NEVER──────┘
```

### Parameters

**TARGET**
Specifies if you want to drop the target tables with the subscription.

**ALWAYS**
Always drop the target table.

**NEVER**
Never drop the target table.

**DROP**
Specify what you want to drop when you delete a Q subscription.

**TARGET**
Target table.

**CONTROL TABLES**
Q Capture and Q Apply control tables.

**TABLESPACE**
Specifies whether the table space should be dropped when the target table or control tables that it contains is dropped.

**WHEN EMPTY**
Drop the table space only when it is empty.

**NEVER**
Never drop the table space.

### Example 1

To always drop the target table when the Q subscription is deleted:

```
SET DROP TARGET ALWAYS
```

### Example 2

To never drop the control tables' table space when the control tables get dropped:

```
SET DROP CONTROL TABLES TABLESPACE NEVER
```

## SET OUTPUT command

Use the SET OUTPUT command to define output files for the ASNCLP program. The output files contain the SQL statements needed to set up Q replication and event publishing.

### Syntax

```
►►──SET OUTPUT──┬────────────────────────────┬──┬──────────────────────────┬──────────────►◄
                └─CAPTURE SCRIPT──"capfname"─┘  └─TARGET SCRIPT──"trgfname"─┘
```

### Parameters

**CAPTURE SCRIPT** *"capfname"*
Specifies the output file name for SQL scripts that run at the Q Capture server.

**TARGET SCRIPT** *"trgfname"*
Specifies the output file name for SQL scripts that run at the Q Apply, or target server.

### Usage notes

- If a script already exists, the new script appends to the current script.
- The double quotation marks in the command syntax are required.

### Example 1

To name the target script output file "target.sql":

```
SET OUTPUT TARGET SCRIPT "target.sql"
```

## SET SERVER command (Q replication and event publishing)

Use the SET SERVER command to specify the Q Capture server or Q Apply server (also referred to as a target server) to use in the ASNCLP session. After you set a server name, all subsequent commands in the session will apply to this server until you change the server with this command.

### Syntax

```
►►──SET SERVER──┬─CAPTURE─┬──TO──────────────────────────────────────────────────────────►
                └─TARGET──┘

►──┬─NULLS─────────────────────────────────────────────────────────────────────────────────┬──►◄
   └─┬─DB──dbalias────────┬──┬────────────────┬──┬─────────────────────────┬──┤ other-options ├─┘
     └─DBALIAS──aliasname─┘  └─DBNAME──dbname─┘  └─NONIBM SERVER──remsrvr─┘
```

**other-options:**

```
├──┬──────────────────────────────┬─────────────────────────────────────────────────────┤
   └─ID──userid── PASSWORD──pwd───┘
```

### Parameters

**CAPTURE**
Specify to set the database as a Q Capture server.

**TARGET**
Specify to set the database as a Q Apply server.

**NULLS**
Specify to set the server name to NULL. This option resets a previously set server name.

**DB** *dbalias*
Specifies the database alias name.

**DBALIAS** *aliasname*
**Linux, UNIX, Windows:** Specifies the database alias name.

**DBNAME** *dbname*
**z/OS:** Specifies the database name.

**NONIBM SERVER**
The remote server name for a federated target. The target can be Oracle, Sybase, Informix, or Microsoft SQL Server. This option is only valid for target servers.

**ID** *userid*
Specifies the user ID to use to connect to the database.

**PASSWORD** *pwd*
Specifies the password to use to connect to the database.

### Example 1

To set the Q Capture server to the database SAMPLE:

```
SET SERVER CAPTURE TO DB SAMPLE
```

### Example 2

To set the target server to the z/OS database TARGETDB:

```
SET SERVER TARGET TO DBNAME TARGETDB
```

### Example 3

To set the target server to an Oracle database ORACLEDB:

```
SET SERVER TARGET TO NONIBM SERVER ORACLEDB
```

## ASNCLP commands for multidirectional Q replication

The ASNCLP commands for multidirectional replication define, change, and drop the objects that are unique to bidirectional and peer-to-peer Q replication.

The following topics demonstrate how you can combine multidirectional Q replication commands to create ASNCLP setup scripts:

- "Sample ASNCLP scripts for setting up bidirectional Q replication" on page 57
- "Sample ASNCLP scripts for setting up peer-to-peer Q replication (two servers)" on page 62
- "Sample ASNCLP scripts for setting up peer-to-peer Q replication (three servers)" on page 67

Table 9 on page 113 lists the ASNCLP commands for event publishing and links to topics that describe each command.

Table 9. ASNCLP commands for multidirectional Q replication

| Description | Command |
|---|---|
| Change a Q subscription for bidirectional replication | "ALTER QSUB command (bidirectional replication)" |
| Change a Q subscription for peer-to-peer replication | "ALTER QSUB command (peer-to-peer replication)" on page 116 |
| Create a Q subscription for bidirectional replication | "CREATE QSUB command (bidirectional replication)" on page 117 |
| Create a Q subscription for peer-to-peer replication | "CREATE QSUB command (peer-to-peer replication)" on page 120 |
| Delete the subgroup that you set by using the SET SUBGROUP command. | "DROP SUBGROUP command (multidirectional Q replication)" on page 122 |
| Delete a Q subscription for bidirectional replication | "DROP SUBTYPE command (bidirectional replication)" on page 123 |
| Delete a Q subscription for peer-to-peer replication between two servers | "DROP SUBTYPE command (peer-to-peer replication)" on page 124 |
| Invoke ASNCLP program scripts used to set up multidirectional replication | "LOAD MULTIDIR REPL SCRIPT command (multidirectional Q replication)" on page 124 |
| Connect the servers that are used for bidirectional or peer-to-peer replication. | "SET CONNECTION command (multidirectional Q replication)" on page 126 |
| Specify the Q Capture and Q Apply schema on a server that is used for multidirectional replication | "SET MULTIDIR SCHEMA command (multidirectional Q replication)" on page 127 |
| Define output files that contain SQL scripts for multidirectional replication | "SET OUTPUT command (multidirectional Q replication)" on page 127 |
| Set a reference table to identify a Q subscription that you want to change or delete. | "SET REFERENCE TABLE command (multidirectional Q replication)" on page 128 |
| Specify the server that contains both Q Capture and Q Apply control tables to use in the ASNCLP session | "SET SERVER command (multidirectional Q replication)" on page 129 |
| Specify the name of the subgroup, a collection of Q subscriptions between servers that are used for multidirectional replication | "SET SUBGROUP command (multidirectional Q replication)" on page 130 |
| Specify the tables that participate in a bidirectional or peer-to-peer configuration | "SET TABLES command (multidirectional Q replication)" on page 130 |

## ALTER QSUB command (bidirectional replication)

Use the ALTER QSUB command to change the properties of one or both bidirectional Q subscriptions for a single logical table.

### Syntax

```
►►──ALTER QSUB SUBTYPE B──────────────────────────────────────────────────────────►

►─┬──────────────────────────────┬──┬────────────────────────────┬──┬────────────────────────────┬─►
  └─FROM NODE──servername.schemaname─┘  └─SOURCE──┤ source-clause ├─┘  └─TARGET──┤ target-clause ├─┘

►─┬──────────────────────────────┬──┬────────────────────────────┬──┬────────────────────────────┬─►◄
  └─FROM NODE──servername.schemaname─┘  └─SOURCE──┤ source-clause ├─┘  └─TARGET──┤ target-clause ├─┘
```

**source-clause:**

```
|--+-----------------------------+--+---------------------------+--------------------|
   '-ALL CHANGED ROWS--+-N-+-----'  '-HAS LOAD PHASE--+-N-+-----'
                       '-Y-'                          +-I-+
                                                      '-E-'
```

**target-clause:**

```
|--+------------------------+--+------------------------------+--+--------------------------+-->
   '-CONFLICT RULE--+-K-+---'  '-CONFLICT ACTION--+-I-+-------'  '-ERROR ACTION--+-Q-+------'
                    +-C-+                         +-F-+                          +-D-+
                    '-A-'                         +-D-+                          '-S-'
                                                  +-S-+
                                                  '-Q-'

>--+--------------------+--+-------------------------------+----------------------|
   '-LOAD TYPE--+-0-+---'  '-OKSQLSTATES--"sqlstates"------'
                +-2-+
                '-3-'
```

## Parameters

**SUBTYPE B**
Specifies bidirectional Q subscriptions.

**FROM NODE** *server***.***schemaname*
Identifies one of the two bidirectional Q subscriptions by specifying the server and schema of its source table.

source-clause:

**ALL CHANGED ROWS**
Specifies the data sending option.

**N** Send a row only if a subscribed column in the source table changes.

**Y** Send a row when any column in the source table changes.

**HAS LOAD PHASE**
Specifies whether the target table for the Q subscription will be loaded with data from the source.

**N** No load phase at the target. This is the default.

**I** Specifies an automatic load. The Q Apply program calls the EXPORT and IMPORT utilities or EXPORT and LOAD utilities, depending on the type of load that is specified in the LOAD_TYPE keyword and on the platform of the Q Apply server and Q Capture server.

**E** Specifies a manual load. An application other than the Q Apply program loads the target table. In this case, you insert the LOADDONE signal (using the LOADDONE command) into the IBMQREP_SIGNAL table at the Q Capture server to inform the Q Capture program that the application is done loading.

target-clause:

**CONFLICT RULE**

**K** Check only key values.

**C** Check changed non-key values in addition to key values.

**A** Check all values for updates.

**CONFLICT ACTION**
Specifies what action to take if a conflict occurs.

**I** Ignore.

**F** The Q Apply program tries to force the change. This requires that the Q Capture program send all columns, so the CHANGED_COLS_ONLY value must be set to N (no) in the IBMQREP_SUBS table.

**D** Disable the Q subscription.

**S** Stop Q Apply.

**Q** Stop reading from queue.

**ERROR ACTION**
Specifies what action to take if an error occurs.

**S** Stop Q Apply without applying the transaction.

**D** Disable the Q subscription and notify Q Capture.

**Q** Stop reading from queue.

**OKSQLSTATES** *"sqlstates"*
Specifies a list of SQL statements within double quotation marks that are not to be considered as errors when applying changes to this table.

**LOAD TYPE**
Specifies a type of load.

**0** Choose the best type automatically.

**2** Use EXPORT and IMPORT only.

**3** Use EXPORT and LOAD only.

## Usage notes

You must use the LOAD MULTIDIR REPL SCRIPT command to invoke the script file that contains this command and the other commands that are required to set its context.

## Example

The following script changes the Q subscriptions for the EMPLOYEE table at SAMPLE and SAMPLE2. For the Q subscription whose source table is at SAMPLE (FROM NODE SAMPLE.RED), the load option will be changed to manual load. For the other Q subscription, the error action is changed to disable the Q subscription and notify the Q Capture program if an error occurs.

To identify the Q subscriptions, the first commands identify the subgroup, the servers in the subgroup, and the reference table RED.EMPLOYEE.

```
SET SUBGROUP "BIDIRGROUP";

SET SERVER MULTIDIR TO DB "SAMPLE";
SET SERVER MULTIDIR TO DB "SAMPLE2";

SET REFERENCE TABLE USING SCHEMA "SAMPLE".RED USES TABLE RED.EMPLOYEE;

ALTER QSUB SUBTYPE B
FROM NODE SAMPLE.RED SOURCE HAS LOAD PHASE E
FROM NODE SAMPLE2.BLUE TARGET ERROR ACTION D;
```

## ALTER QSUB command (peer-to-peer replication)

Use the ALTER QSUB command to change the properties of the peer-to-peer Q subscriptions for a single logical table.

### Syntax

```
►►──ALTER QSUB──SUBTYPE── P──SOURCE──┤ source-clause ├──┬────────────────────────────────┬──────────►◄
                                                        └─TARGET──┤ target-clause ├─┘
```

**source-clause:**

```
├──┬──────────────────────────┬──────────────────────────────────────────────────────────────────────┤
   └─HAS LOAD PHASE──┬─ N─┬─┘
                     ├─ I─┤
                     └─ E─┘
```

**target-clause:**

```
├──┬────────────────────────┬──┬──────────────────────┬──┬──────────────────────────────┬──────────┤
   └─ERROR ACTION──┬─ Q─┬─┘  └─LOAD TYPE──┬─ 0─┬─┘  └─OKSQLSTATES──"sqlstates"─┘
                   ├─ D─┤                 ├─ 2─┤
                   └─ S─┘                 └─ 3─┘
```

### Parameters

**SUBTYPE P**
Specifies a peer-to-peer Q subscription.

source-clause:

**HAS LOAD PHASE**
Specifies whether the target table for the Q subscription will be loaded with data from the source.

**N** No load phase at the target. This is the default.

**I** Specifies an automatic load. The Q Apply program calls the EXPORT and IMPORT utilities or EXPORT and LOAD utilities, depending on the type of load that is specified in the LOAD_TYPE keyword, and on the platform of the Q Apply server and Q Capture server.

**E** Specifies a manual load. An application other than the Q Apply program loads the target table. In this case, you insert the LOADDONE signal (using the LOADDONE command) into the IBMQREP_SIGNAL table at the Q Capture server to inform the Q Capture program that the application is done loading.

target-clause:

**ERROR ACTION**

**D** Disable subscription and notify the Q Capture program.

**S** Stop the Q Apply program without applying the transaction.

**Q** Stop reading from the receive queue.

**LOAD TYPE**
Specifies a type of load.

**0** Choose the best type automatically.

**2** Use EXPORT and IMPORT only.

**3** Use EXPORT and LOAD only.

**OKSQLSTATES** *"sqlstates"*
Specifies a list of SQL statements within double quotation marks that are not to be considered as error when applying changes to this table.

### Usage notes

You must use the LOAD MULTIDIR REPL SCRIPT command to invoke the script file that contains this command and the other commands that are required to set its context.

### Example

The following script changes the Q subscriptions for the STAFF table at SAMPLE, SAMPLE2, and SAMPLE3 in a peer-to-peer configuration with three servers. The command specifies an automatic load that uses the EXPORT and IMPORT utilities and sets the error action to disable the Q subscription and notify the Q Capture program if an error occurs.

To identify the Q subscriptions, the first commands identify the subgroup, the servers in the subgroup, and the reference table GRAY.STAFF.

```
SET SUBGROUP "P2P3GROUP";

SET SERVER MULTIDIR TO DB "SAMPLE";
SET SERVER MULTIDIR TO DB "SAMPLE2";
SET SERVER MULTIDIR TO DB "SAMPLE3";

SET REFERENCE TABLE USING SCHEMA "SAMPLE".GRAY USES TABLE GRAY.STAFF;

ALTER QSUB SUBTYPE P SOURCE HAS LOAD PHASE I TARGET ERROR ACTION D LOAD TYPE 2;
```

## CREATE QSUB command (bidirectional replication)

Use the CREATE QSUB command to create two Q subscriptions for a single logical table that participates in bidirectional replication.

### Syntax

```
►►──CREATE QSUB SUBTYPE B──────────────────────────────────────────────────────►

►──┬─────────────────────────────────────────────────────────────────────────┬─►
   └─FROM NODE──servername.schemaname── SOURCE──┤ source-clause ├──TARGET──┤ target-clause ├─┘

►──┬─────────────────────────────────────────────────────────────────────────┬─►◄
   └─FROM NODE──servername.schemaname── SOURCE──┤ source-clause ├──TARGET──┤ target-clause ├─┘
```

**source-clause:**

```
├──┬───────────────────────────┬──┬──────────────────────────┬──────────────────────────────────────────┤
   └─ALL CHANGED ROWS──┬─ N─┬──┘  └─HAS LOAD PHASE──┬─ N─┬──┘
                       └─ Y─┘                       ├─ I─┤
                                                    └─ E─┘
```

**target-clause:**

```
├──┬─────────────────────────┬──┬────────────────────────────┬──┬─────────────────────────┬──────────────►
   └─CONFLICT RULE──┬─ K─┬──┘  └─CONFLICT ACTION──┬─ I─┬──┘  └─ERROR ACTION──┬─ Q─┬──┘
                    ├─ C─┤                         ├─ F─┤                     ├─ D─┤
                    └─ A─┘                         ├─ D─┤                     └─ S─┘
                                                   ├─ S─┤
                                                   └─ Q─┘

►──┬──────────────────────┬──┬────────────────────────────────┬───────────────────────────────────────────┤
   └─LOAD TYPE──┬─ 0─┬───┘  └─OKSQLSTATES──"sqlstates"──────┘
                ├─ 2─┤
                └─ 3─┘
```

## Parameters

**SUBTYPE B**

Specifies bidirectional Q subscriptions.

**FROM NODE** *servername***.***schemaname*

A FROM NODE statement is required if you want to specify options for one or both of the Q subscriptions. If you omit FROM NODE, both Q subscriptions will be created with the following default options:

- ALL_CHANGED_ROWS=N
- BEFORE_VALUES=N
- CHANGED_COLS_ONLY=Y
- HAS_LOADPHASE=I
- CONFLICT_ACTION=K
- CONFLICT_RULE=I
- ERROR_ACTION=Q

In the FROM NODE statement, you specify a server name and schema to identify the logical table that is the source for the Q subscription.

source-clause:

**ALL CHANGED ROWS**

Specifies the data sending option.

**N** Send a row only if a subscribed column in the source table changes.

**Y** Send a row when any column in the source table changes.

**HAS LOAD PHASE**

Specifies whether the target table for the Q subscription will be loaded with data from the source.

**N** No load phase at the target. This is the default.

**I** Specifies an automatic load. The Q Apply program calls the EXPORT and IMPORT utilities or EXPORT and LOAD utilities, depending on the type of load that is specified in the LOAD_TYPE keyword and on the platform of the Q Apply server and Q Capture server.

**E** Specifies a manual load. An application other than the Q Apply program loads the target table. In this case, you insert the LOADDONE signal (using the LOADDONE command) into the IBMQREP_SIGNAL table at the Q Capture server to inform the Q Capture program that the application is done loading.

target-clause:

**CONFLICT RULE**

**K** Check only key values.

**C** Check changed non-key values in addition to key values.

**A** Check all values for updates.

**CONFLICT ACTION**

**I** Ignore.

**F** The Q Apply program tries to force the change. This requires that the Q Capture program send all columns, so the CHANGED_COLS_ONLY value must be set to N (no) in the IBMQREP_SUBS table.

**D** Disable the Q subscription.

**S** Stop the Q Apply program.

**Q** Stop reading from the receive queue.

**ERROR ACTION**
Specifies what action to take if an error occurs.

**Q** Stop reading from the receive queue.

**D** Disable the Q subscription and notify the Q Capture program.

**S** Stop the Q Apply program without applying the transaction.

**OKSQLSTATES** *"sqlstates"*
Specifies a list of SQL statements within double quotation marks that are not to be considered as errors when applying changes to this table.

**LOAD TYPE**
Specifies the utilities that the Q Apply program uses to load the target.

**0** Choose the best type automatically.

**2** Use EXPORT and IMPORT only.

**3** Use EXPORT and LOAD only.

## Usage notes

Table 10 on page 120 shows the permitted combinations for BEFORE_VALUES and CHANGE_COLS_ONLY depending on the values of CONFLICT_RULE and CONFLICT_ACTION.

**Recommendation:** Always use the ASNCLP or Replication Center to change the value of CONFLICT_RULE and CONFLICT_ACTION. The administration tools will automatically set the correct value for BEFORE_VALUES and CHANGE_COLS_ONLY. Neither of these attributes can be set explicitly using the administration tools.

*Table 10. Required attributes for BEFORE_VALUES and CHANGE_COLS_ONLY depending on the values of CONFLICT_RULE and CONFLICT_ACTION*

| CONFLICT RULE | CONFLICT ACTION | BEFORE VALUES | CHANGE COLS ONLY |
|---|---|---|---|
| K | I, S, D, or Q | N | Y |
| K | F | N | N |
| C | I, S, D, or Q | Y | Y |
| C | F | Y | N |
| A | I, S, D, or Q | Y | N |

### Example

The following commands create two Q subscriptions for bidirectional replication between the SAMPLE and SAMPLE2 servers. The commands specify an automatic load at both servers. At SAMPLE, a CONFLICT_RULE of C (check changed key and non-key values) and a CONFLICT_ACTION of F (force the change) are specified. At SAMPLE2, a CONFLICT_RULE of A (check all values for updates) and a CONFLICT_ACTION of I (ignore) are specified.

To identify the Q subscriptions, the first commands identify the subgroup, the servers and schemas in the subgroup, and the two replication queue maps. The SET TABLES command specifies the RED.EMPLOYEE table at the SAMPLE database, which will generate statements to create a matching table at SAMPLE2.

```
SET SUBGROUP "bidirgroup"

SET SERVER MULTIDIR TO DB "SAMPLE";
SET SERVER MULTIDIR TO DB "SAMPLE2";

SET MULTIDIR SCHEMA "SAMPLE".RED;
SET MULTIDIR SCHEMA "SAMPLE2".BLUE;

SET CONNECTION SOURCE "SAMPLE".RED TARGET "SAMPLE2".BLUE REPLQMAP
"SAMPLE_RED_TO_SAMPLE2_BLUE";
SET CONNECTION SOURCE "SAMPLE2".BLUE TARGET "SAMPLE".RED REPLQMAP
"SAMPLE2_BLUE_TO_SAMPLE_RED";

SET TABLES (SAMPLE.RED.RED.EMPLOYEE);

CREATE QSUB SUBTYPE B
FROM NODE SAMPLE.RED SOURCE SOURCE HAS LOAD PHASE I
TARGET CONFLICT RULE C CONFLICT ACTION F
FROM NODE SAMPLE2.BLUE SOURCE SOURCE HAS LOAD PHASE I
TARGET CONFLICT RULE A CONFLICT ACTION I
```

## CREATE QSUB command (peer-to-peer replication)

Use the CREATE QSUB command to create a set of Q subscriptions for a single logical table that participates in peer-to-peer replication.

### Syntax

```
►►─CREATE QSUB─SUBTYPE─ P─┬─────────────────────────────┬─┬─────────────────────────────┬─────────►◄
                          └─SOURCE─┤ source-clause ├─┘ └─TARGET─┤ target-clause ├─┘
```

### source-clause:

```
├──┬──────────────────────────┬──────────────────────────────────────────────────────┤
   └─HAS LOAD PHASE──┬─ N─┬─┘
                     ├─ I─┤
                     └─ E─┘
```

### target-clause:

```
├──┬─────────────────────────┬──┬─────────────────────┬──┬──────────────────────────────┬─────────────┤
   └─ERROR ACTION──┬─ Q─┬─┘  └─LOAD TYPE──┬─ 0─┬─┘  └─OKSQLSTATES─"sqlstates"─┘
                   ├─ D─┤                 ├─ 2─┤
                   └─ S─┘                 └─ 3─┘
```

## Parameters

**SUBTYPE P**
Specifies Q subscriptions for peer-to-peer replication.

source-clause:

**HAS LOAD PHASE**
Specifies whether the tables that are specified in the Q subscriptions will be loaded with data from one of the peer copies of the table.

**N** No load phase. This is the default.

**I** Specifies an automatic load. The Q Apply program calls the EXPORT and IMPORT utilities or EXPORT and LOAD utilities, depending on the type of load that is specified in the LOAD TYPE keyword, and on the platform of the Q Apply server and Q Capture server.

**E** Specifies a manual load. An application other than the Q Apply program loads the target table. In this case, you insert the LOADDONE signal (using the LOADDONE command) into the IBMQREP_SIGNAL table at the Q Capture server to inform the Q Capture program that the application is done loading.

target-clause:

**ERROR ACTION**
Specifies what action to take if an error occurs.

**Q** Stop reading from the receive queue.

**D** Disable subscription and notify the Q Capture program.

**S** Stop the Q Apply program without applying the transaction.

**LOAD TYPE**
Specifies a type of load.

**0** Choose the best type automatically.

**2** Use EXPORT and IMPORT only.

**3** Use EXPORT and LOAD only.

**OKSQLSTATES** *"sqlstates"*
Specifies a list of SQL statements within double quotation marks that are not to be considered as error when applying changes to this table.

## Usage notes

- Convergence columns and triggers will be created on the tables that participate in the peer-to-peer replication setup.
- For peer-to-peer replication with convergence, only the attributes shown in Table 11 are allowed (and are implicitly assigned).

*Table 11. Attributes for peer-to-peer replication with convergence*

| Conflict Rule | Conflict Action | Before Values | Change Cols Only |
|---|---|---|---|
| V | F | N | N |

## Example

The following script creates Q subscriptions for the STAFF table at SAMPLE, SAMPLE2, and SAMPLE3 in a peer-to-peer configuration with three servers. The Q subscriptions specify no load phase and an error action that prompts the Q Apply program to stop reading from the receive queue if an error occurs.

To identify the Q subscriptions, the first commands identify the subgroup, the servers and schemas in the subgroup, and the replication queue maps. The SET TABLES command specifies GRAY.STAFF at the SAMPLE database, which will generate SQL statements to create matching tables at the other two servers.

```
SET SUBGROUP "p2p3group";

SET SERVER MULTIDIR TO DB "SAMPLE";
SET SERVER MULTIDIR TO DB "SAMPLE2";
SET SERVER MULTIDIR TO DB "SAMPLE3";

SET MULTIDIR SCHEMA "SAMPLE".GRAY;
SET MULTIDIR SCHEMA "SAMPLE2".BROWN;
SET MULTIDIR SCHEMA "SAMPLE3".YELLOW;

SET CONNECTION SOURCE "SAMPLE".GRAY TARGET "SAMPLE2".BROWN REPLQMAP
"SAMPLE_GRAY_TO_SAMPLE2_BROWN";
SET CONNECTION SOURCE "SAMPLE".GRAY TARGET "SAMPLE3".YELLOW REPLQMAP
"SAMPLE_GRAY_TO_SAMPLE3_YELLOW";
SET CONNECTION SOURCE "SAMPLE2".BROWN TARGET "SAMPLE".GRAY REPLQMAP
"SAMPLE2_BROWN_TO_SAMPLE_GRAY";
SET CONNECTION SOURCE "SAMPLE2".BROWN TARGET "SAMPLE3".YELLOW REPLQMAP
"SAMPLE2_BROWN_TO_SAMPLE3_YELLOW";
SET CONNECTION SOURCE "SAMPLE3".YELLOW TARGET "SAMPLE".GRAY REPLQMAP
"SAMPLE3_YELLOW_TO_SAMPLE_GRAY";
SET CONNECTION SOURCE "SAMPLE3".YELLOW TARGET "SAMPLE2".BROWN REPLQMAP
"SAMPLE3_YELLOW_TO_SAMPLE2_BROWN";

SET TABLES (SAMPLE.GRAY.GRAY.STAFF);

CREATE QSUB SUBTYPE P SOURCE HAS LOAD PHASE N TARGET ERROR ACTION Q;
```

# DROP SUBGROUP command (multidirectional Q replication)

Use the DROP SUBGROUP command to delete the subgroup that you set by using the SET SUBGROUP command.

## Syntax

```
►►──DROP SUBGROUP────────────────────────────────────────────────────────────►◄
```

### Usage notes

When you delete a subgroup, all Q subscriptions within the group are also deleted.

### Example 1

The following script drops the bidirectional subgroup BIDIGROUP. First it sets the subgroup, then sets the two servers in the group. The SET MULTIDIR SCHEMA command specifies the shared Q Capture and Q Apply schema RED at one of the servers to further identify the Q subscriptions that are dropped at both servers along with the subgroup.

```
SET SUBGROUP "BIDIRGROUP";

SET SERVER MULTIDIR TO DB "SAMPLE";
SET SERVER MULTIDIR TO DB "SAMPLE1";

SET MULTIDIR SCHEMA "SAMPLE".RED

DROP SUBGROUP;
```

## DROP SUBTYPE command (bidirectional replication)

Use the DROP SUBTYPE command to delete both bidirectional Q subscriptions for a single logical table.

The command creates SQL statements to connect to both servers in the bidirectional configuration and delete the Q subscription from their control tables.

### Syntax

```
►►──DROP SUBTYPE────B──QSUBS──────────────────────────────────────────────────────────►◄
```

### Parameters

**B** Specifies bidirectional Q replication.

**QSUBS**
Specifies that all of the Q subscriptions that are defined with the same SET SUBGROUP command will be deleted.

### Usage notes

- No tables or table spaces are ever dropped.
- You must use the LOAD MULTIDIR REPL SCRIPT command to invoke the script file that contains this command and the other commands that are required to set its context.

### Example

The following commands delete the Q subscription for the EMPLOYEE table at SAMPLE and SAMPLE2. To identify the Q subscription, the first commands identify the subgroup, the servers in the subgroup, and the reference table RED.EMPLOYEE.

```
SET SUBGROUP "BIDIRGROUP";

SET SERVER MULTIDIR TO DB "SAMPLE";
SET SERVER MULTIDIR TO DB "SAMPLE2";
SET REFERENCE TABLE USING SCHEMA "SAMPLE".RED USES TABLE RED.EMPLOYEE;

DROP SUBTYPE B QSUBS;
```

## DROP SUBTYPE command (peer-to-peer replication)

Use the DROP SUBTYPE command to delete the peer-to-peer Q subscriptions for a single logical table.

The command creates SQL statements to connect to all servers in the peer-to-peer configuration and delete the Q subscription from their control tables.

### Syntax

```
►►──DROP──SUBTYPE P──QSUBS─────────────────────────────────────────────────────────────►◄
```

### Parameters

**SUBTYPE P**
Specifies a peer-to-peer Q subscription.

### Usage notes

- No tables or table spaces are ever dropped.
- Convergence columns and triggers will remain on the tables that previously participated in a peer-to-peer replication scenario.
- You must use the LOAD MULTIDIR REPL SCRIPT command to invoke the script file that contains this command and the other commands that are required to set its context.

### Example

The following script deletes the Q subscription for the STAFF table at SAMPLE, SAMPLE2, and SAMPLE3. To identify the Q subscription, the first commands identify the subgroup, the servers in the subgroup, and the reference table GRAY.STAFF.

```
SET SUBGROUP "P2P3GROUP";

SET SERVER MULTIDIR TO DB "SAMPLE";
SET SERVER MULTIDIR TO DB "SAMPLE2";
SET SERVER MULTIDIR TO DB "SAMPLE3";

SET REFERENCE TABLE USING SCHEMA "SAMPLE".GRAY USES TABLE GRAY.STAFF;

DROP SUBTYPE P QSUBS;
```

## LOAD MULTIDIR REPL SCRIPT command (multidirectional Q replication)

Use the LOAD MULTIDIR REPL SCRIPT command to invoke ASNCLP program scripts used to set up peer-to-peer and bidirectional replication.

### Syntax

```
►►──LOAD MULTIDIR REPL SCRIPT──┬─"filelocation/filename"─┬──────────────────────────────►◄
                               └─"filelocation\filename"─┘
```

## Parameters

*filelocation*
Specifies the absolute path where the input file is located. If no directory is specified, the current directory is assumed.

*filename*
Specifies the name of the bidirectional or peer-to-peer replication input file.

## Usage notes

- Only definitions pertaining to one subgroup can be placed in one bidirectional or peer-to-peer replication script.
- Several scripts can be invoked to set up several subgroups if each one is invoked with its own LOAD MULTIDIR REPL SCRIPT call.
- Several LOAD MULTIDIR REPL SCRIPT statements can exist in one ASNCLP program input file.

## Example

The following is a sample script used to invoke four bidirectional or peer-to-peer scripts:

```
LOAD MULTIDIR REPL SCRIPT "3nodes\3Node0.in";
LOAD MULTIDIR REPL SCRIPT "3nodes\3Node1.in";
LOAD MULTIDIR REPL SCRIPT "3nodes\3Node2.in";
LOAD MULTIDIR REPL SCRIPT "3nodes\3Node3.in";
```

**Note:** This script creates four subgroups. Each subgroup definition is placed into a bidirectional or peer-to-peer script (for example, 3Node0.in).

The following is a sample bidirectional or peer-to-peer script (3Node0.in):

```
# Give the subgroup a name.
set subgroup "3Node0";

# Set the servers (databases) that will participate in this subgroup.
set server multidir to db "testdb";
set server multidir to db "testdb1";
set server multidir to db "testdb2";

# Specify the Q Capture/Q Apply schema for the catalogs used on those servers.
set multidir schema "testdb".BLUE;
set multidir schema "testdb1".RED;
set multidir schema "testdb2".YELLOW;

# Specify the replication queue maps used to join the catalogs together
set connection SOURCE "testdb".BLUE TARGET "testdb1".RED replqmap "BLUEtoRED";
set connection SOURCE "testdb".BLUE TARGET "testdb2".YELLOW replqmap "BLUEtoYELLOW";
set connection SOURCE "testdb1".RED TARGET "testdb".BLUE replqmap "REDtoBLUE";
set connection SOURCE "testdb1".RED TARGET "testdb2".YELLOW replqmap "REDtoYELLOW";
set connection SOURCE "testdb2".YELLOW TARGET "testdb".BLUE replqmap "YELLOWtoBLUE";
set connection SOURCE "testdb2".YELLOW TARGET "testdb1".RED replqmap "YELLOWtoRED";

# Specify the tables to participate in this subgroup (1 per server).
set tables("testdb".BLUE.BLUE.AllTypes0, "testdb1".RED.RED.AllTypes0,
"testdb2".YELLOW.YELLOW.AllTypes0);
# Create the subgroup
create qsub subtype p;
```

This bidirectional or peer-to-peer script creates a subgroup “3Node0”. All of the information required to generate the subgroup’s Q subscriptions is located in this one input file.

## SET CONNECTION command (multidirectional Q replication)

Use the SET CONNECTION command to connect the two servers that are used for bidirectional or peer-to-peer replication.

### Syntax

```
►►──SET CONNECTION──┬───────────────────────────┬──SOURCE──sourceservername.sourceschemaname──────────►
                    └─SUBNAME──subscriptionname─┘

►─TARGET──targetservername.targetschemaname──REPLQMAP──mapname────────────────────────────────────────►◄
```

### Parameters

**SUBNAME** *subscriptionname*
Specifies the name of the Q subscription between the two servers (from source to target) that are specified in the connection. If more than one Q subscription is created between the two servers, the first Q subscription will carry the name as specified, and every subsequent Q subscription will have an incremental number appended to it.

**SOURCE**

*sourceservername*
Specifies the name of the source server.

*sourceschemaname*
Specifies the schema of the control tables at the source server.

**TARGET**

*targetservername*
Specifies the name of the target server.

*targetschemaname*
Specifies the schema of the control tables at the target server.

**REPLQMAP** *mapname*
Specifies the name of the replication queue map that connects the Q Capture program at the source server with the Q Apply program at the target server.

### Usage notes

To make a connection between two servers, you must run the SET CONNECTION command twice because both servers act as a source and a target. See the example below.

### Example

To set the connection between the servers BLUE and RED that are used for peer-to-peer replication servers:

```
SET CONNECTION SOURCE TESTDB.BLUE
TARGET TESTDB1.RED REPLQMAP BLUE.TO.RED;

SET CONNECTION SOURCE TESTDB1.RED
TARGET TESTDB.BLUE REPLQMAP RED.TO.BLUE
```

## SET MULTIDIR SCHEMA command (multidirectional Q replication)

Use the SET MULTIDIR SCHEMA command to set the same schema for the Q Capture and Q Apply control tables on a server that is used for bidirectional or peer-to-peer replication. If you do not specify the server or schema, the ASNCLP program defaults to ASN.

### Syntax

```
►►──SET MULTIDIR SCHEMA──servername.schemaname──────────────────────────────────────────────►◄
```

### Parameters

*servername*
Specifies the name of the server that contains the Q Capture and Q Apply control tables.

*schemaname*
Specifies the schema for the Q Capture and Q Apply control tables on a server that is used for bidirectional or peer-to-peer replication.

### Example

To set the multidirectional schema for bidirectional or peer-to-peer replication to BLUE on the server TESTDB:

```
SET MULTIDIR SCHEMA TESTDB.BLUE
```

## SET OUTPUT command (multidirectional Q replication)

Use the SET OUTPUT command to define output files for the ASNCLP program. The output files contain the SQL statements needed to set up multidirectional Q replication.

### Syntax

```
►►──SET OUTPUT──┬──────────┬──┬─────────────────────────────┬───────────────────────────────►◄
                └─MULTIDIR─┘  └─MONITOR SCRIPT──"monfname"─┘
```

### Parameters

**MULTIDIR**
Specify to name the output files after the databases that the SQL scripts run on.

**MONITOR SCRIPT** *"monfname"*
Specifies the output file name for scripts that run at the Monitor control server. The default file name is replmonitor.sql.

### Usage notes

- If a script already exists, the new script appends to the current script.
- MULTIDIR does not require a file name because the ASNCLP program automatically names the output SQL scripts based on the names of the databases that the SQL scripts run on.
- The double quotation marks in the command syntax are required.

### Example 1

To name the SQL script output files based on the names of the databases that the SQL script runs on:

```
SET OUTPUT MULTIDIR
```

## SET REFERENCE TABLE command (multidirectional Q replication)

Use the SET REFERENCE TABLE command to identify a Q subscription for bidirectional or peer-to-peer replication. You specify this command before you use the ALTER QSUB or DROP SUBTYPE commands to change or drop the Q subscriptions.

### Syntax

```
►►—SET REFERENCE TABLE— USING SCHEMA—server.schema—USES TABLE—tableowner.tablename————————————►◄
```

### Parameters

**USING SCHEMA**

*server*
Specifies the name of the server that contains the table.

*schema*
Specifies the schema of the control tables in which this table is specified as a source and target.

**USES TABLE**

*tableowner*
Specifies the table schema.

*tablename*
Specifies the table name.

### Example 1

The following script sets the reference table RED.DEPARTMENT at the server SAMPLE to identify and change the Q subscription for the DEPARTMENT table at SAMPLE and SAMPLE1.

```
SET SUBGROUP "BIDIRGROUP";

SET SERVER MULTIDIR TO DB "SAMPLE";
SET SERVER MULTIDIR TO DB "SAMPLE1";

SET REFERENCE TABLE USING SCHEMA "SAMPLE".RED USES TABLE RED.DEPARTMENT;

ALTER QSUB SUBTYPE B SOURCE HAS LOAD PHASE I TARGET ERROR ACTION S;
```

### Example 2

The following script sets the reference table RED.EMPLOYEE at the server SAMPLE to identify and drop the Q subscription for the EMPLOYEE table at SAMPLE, SAMPLE1, and SAMPLE2.

```
SET SUBGROUP "P2P3GROUP";

SET SERVER MULTIDIR TO DB "SAMPLE";
SET SERVER MULTIDIR TO DB "SAMPLE1";
SET SERVER MULTIDIR TO DB "SAMPLE2";

SET REFERENCE TABLE USING SCHEMA "SAMPLE".RED USES TABLE RED.EMPLOYEE;

DROP SUBTYPE P QSUBS;
```

## SET SERVER command (multidirectional Q replication)

Use the SET SERVER command to specify the server that contains both Q Capture and Q Apply control tables (MULTIDIR) to use in the ASNCLP session. After you set a server name, all subsequent commands in the session will apply to this server until you change the server with this command.

### Syntax

```
►►──SET SERVER──┬─CAPTURE──┬──TO──┬─NULLS──────────────────────────────────────────────────────────┬──►◄
                ├─TARGET───┤      └─┬─DB──dbalias──────────┬──┬─────────────────┬──┤ other-options ├─┘
                └─MULTIDIR─┘        └─DBALIAS──aliasname───┘  └─DBNAME──dbname──┘
```

**other-options:**

```
├──┬───────────────────────────┬──────────────────────────────────────────────────────────┤
   └─ID──userid──PASSWORD──pwd─┘
```

### Parameters

**CAPTURE**
Specify to set the database as a Q Capture server.

**TARGET**
Specify to set the database as a Q Apply server (also referred to as target server).

**MULTIDIR**
Specify to set the database as a bidirectional or peer-to-peer replication server. For z/OS, this is the subsystem location name.

**NULLS**
Specify to set the server name to NULL. This option resets a previously set server name.

**DB** *dbalias*
Specifies the database alias name.

**DBALIAS** *aliasname*
**Linux, UNIX, Windows:** Specifies the database alias name.

**DBNAME** *dbname*
**z/OS:** Specifies the database name.

**ID** *userid*
Specifies the user ID to use to connect to the database.

**PASSWORD** *pwd*
Specifies the password to use to connect to the database.

### Usage notes

When using bidirectional or peer-to-peer replication, you must explicitly set the MULTIDIR option.

### Example

To set the bidirectional or peer-to-peer replication server to the database TESTDB:

```
SET SERVER MULTIDIR TO DB TESTDB
```

## SET SUBGROUP command (multidirectional Q replication)

Use the SET SUBGROUP command to specify a name for a collection of Q subscriptions that are involved in bidirectional or peer-to-peer replication.

### Syntax

```
►►—SET SUBGROUP—subgroup-name———————————————————————————————————————►◄
```

### Parameters

*subgroupname*
: Specifies the name of the collection of Q subscriptions for bidirectional or peer-to-peer replication.

### Example

To set the subgroup BLUEandRED:

```
SET SUBGROUP BLUEandRED
```

## SET TABLES command (multidirectional Q replication)

Use the SET TABLES command to specify the tables that participate in a single bidirectional or peer-to-peer subscription (each listed table is both a source and a target for the Q subscription).

### Syntax

```
►►—SET TABLES————————————————————————————————————————————————————————►


                                           ┌─,─────────────────────────────────────┐
                                           ▼                                       │
►—(—server.schema.table_owner.table_name—,———server.schema.table_owner.table_name—┴— )————————►◄
```

### Parameters

*server*
: Specifies the name of the server (database) that contains the table.

*schema*
: Specifies the schema of the control tables in which this table is specified as a source or target.

*tableowner*
: Specifies the schema of the table.

*tablename*
Specifies the name of the table.

### Usage notes

- You must specify at least one table.
  - The first table must be located at the starting peer (peer-to-peer replication) or primary server (bidirectional replication), and it must already exist.
  - If you specify additional tables that already exist at the other servers, the ASNCLP program will check to see if they exist. If the tables do not exist, they will be created based on the first table.
- You must specify a CREATE QSUB command after identifying the tables for the Q subscription with the SET TABLES command.
- To create a set of Q subscriptions for peer-to-peer or bidirectional replication using the tables specified in the SET TABLES command, you must issue a CREATE QSUB command before the next SET TABLES command. That is, each SET TABLES command will override the previous one until you issue a CREATE QSUB statement.

### Example 1

In this example, the table specified in parentheses is BLUE.TABLE3 on the testdb server with a Q Capture and Q Apply schema of BLUE. There are two other servers in the peer-to-peer configuration: testdb1 with a shared schema of RED and testdb2 with a shared schema of GREEN. New tables will be generated on testdb1 and testdb2 with the names RED.TGTTABLE3 and GREEN.TGTTABLE3 because no tables were specified explicitly for the RED and GREEN servers.

```
SET TABLES ("testdb".BLUE.BLUE.TABLE3);
CREATE QSUB SUBTYPE P;
```

### Example 2

In this example, the first table specified in the SET TABLES command is RCTEST2.TABLE2 on the testdb server with a Q Capture and Q Apply schema of BLUE. New tables will be generated on testdb1 and testdb2 with the name of RCTEST3.XYZ and RCBLUE.AllTypes0 because the two other tables are specified explicitly.

```
SET TABLES ("testdb".BLUE.RCTEST2.TABLE2,"testdb1".RED.RCTEST3.XYZ,
"testdb2".YELLOW.RCBLUE.AllTypes0);
CREATE QSUB SUBTYPE P;
```

## ASNCLP commands for event publishing

The ASNCLP commands for event publishing define and change publishing queue maps and XML publications. The commands also can be used to start and stop XML publications.

"Sample ASNCLP scripts for setting up event publishing" on page 132 demonstrates how you can combine event publishing commands to create an ASNCLP setup script.

Table 12 on page 132 lists the ASNCLP commands for event publishing and links to topics that describe each command.

*Table 12. ASNCLP commands for event publishing*

| If you want to ... | Use this command |
|---|---|
| Change a publishing queue map | “ALTER PUBQMAP command” on page 135 |
| Change an XML publication | “ALTER XML PUB command” on page 136 |
| Create a publishing queue map | “CREATE PUBQMAP command” on page 138 |
| Create an XML publication | “CREATE XML PUB command” on page 139 |
| Delete a publishing queue map | “DROP PUBQMAP command” on page 142 |
| Delete an XML publication | “DROP XML PUB command” on page 143 |
| Start an XML publication | “START XML PUB command” on page 143 |
| Stop an XML publication | “STOP XML PUB command” on page 144 |

## Sample ASNCLP scripts for setting up event publishing

This sample contains four ASNCLP scripts for setting up a basic event publishing environment. It includes Q Capture control tables, a publishing queue map, and an XML publication.

ASNCLP scripts typically generate one or more SQL scripts to create publishing objects. Because some publishing objects depend on the existence of other objects, run the ASNCLP scripts and the SQL scripts that they generate in the following order:

1. Q Capture control tables
2. Publishing queue map
3. Q subscription

The final ASNCLP script checks the WebSphere® MQ environment for event publishing and does not generate SQL statements. You must create the control tables and queue map before you run this script.

This sample has a section for each ASNCLP script, which you can copy to a text file and run by using the `ASNCLP -f` *`filename`* command. Within the code sample in each section, details about each group of commands are preceded by a comment character (#).

Table 13 on page 135 below the sample describes the SQL scripts that are generated.

For help creating the WebSphere MQ objects that are used in these scripts, see Graphical tool for generating WebSphere MQ setup scripts for Q replication and event publishing and WebSphere MQ setup scripts for Q replication.

### ASNCLP script 1 (Q Capture control tables)

This script generates SQL statements that create Q Capture control tables at the SAMPLE database. It includes commands for the following tasks:

**1** Setting the environment
**2** Creating Q Capture control tables
**3** Ending the ASNCLP session

```
# 1 Setting the environment.
# The SET LOG command directs ASNCLP messages to one log file, ep.err.
# The SET OUTPUT command creates an SQL script, epcontrol.sql.
# The SET RUN SCRIPT LATER option allows you to review the SQL scripts before
# they are run.

ASNCLP SESSION SET TO Q REPLICATION;
SET SERVER CAPTURE TO DB SAMPLE ID DB2ADMIN PASSWORD "passw0rd";
SET QMANAGER "QM1" FOR CAPTURE SCHEMA;
SET CAPTURE SCHEMA SOURCE EP1;
SET OUTPUT CAPTURE SCRIPT "epcontrol.sql";
SET RUN SCRIPT LATER;

# 2 Creating Q Capture control tables.
# The command specifies a restart queue and Q Capture administration queue.
# It also reduces the time that the Q Capture program pauses after reaching the
# end of the DB2 recovery log from the default of 5000 milliseconds (5 seconds)
# to 1000 milliseconds.

CREATE CONTROL TABLES FOR CAPTURE SERVER USING
RESTARTQ "EP1.QM1.RESTARTQ" ADMINQ "EP1.QM1.ADMINQ"
SLEEP INTERVAL 1000;

# 3 Ending the ASNCLP session.

QUIT;
```

### ASNCLP script 2 (publishing queue map)

This script generates SQL statements that create a publishing queue map. It includes commands for the following tasks:

1 Setting the environment
2 Creating a publishing queue map
3 Ending the ASNCLP session

```
# 1 Setting the environment.
# The SET OUTPUT command creates a SQL script, pqmap.sql, which adds definitions
# for the queue map to the Q Capture control tables.

ASNCLP SESSION SET TO Q REPLICATION;
SET LOG "pqmap.err";
SET SERVER CAPTURE TO DB SAMPLE ID DB2ADMIN PASSWORD "passw0rd";
SET CAPTURE SCHEMA SOURCE EP1;
SET OUTPUT CAPTURE SCRIPT "pqmap.sql";
SET RUN SCRIPT LATER;

# 2 Creating a publishing queue map.
# This command generates SQL to create a publishing queue map,
# SAMPLE_ASN1_TO_SUBSCRIBER. It specifies a send queue at the Q Capture
# server. The command also specifies that the content of each message will
# be a single row, that the memory buffer for each message (MAX MESSAGE SIZE)
# will be 128 KB (double the default), and that that heartbeat messages will be
# sent every 5 seconds.
CREATE PUBQMAP SAMPLE_ASN1_TO_SUBSCRIBER USING
SENDQ "EP1.QM1.PUBDATAQ" MESSAGE CONTENT TYPE R
MAX MESSAGE SIZE 128 HEARTBEAT INTERVAL 5;

# 5 Ending the ASNCLP session.

QUIT;
```

### ASNCLP script 3 (XML publication)

This script generates SQL statements to create an XML publication. It specifies a source table, DEPARTMENT, at the SAMPLE database. The script includes commands for the following tasks:

1 Setting the environment
2 Creating an XML publication
3 Ending the ASNCLP session

```
# 1 Setting the environment.
# The SET OUTPUT command creates a SQL script, xmlpub.sql, that adds definitions
# for the XML publication to the Q Capture control tables.

ASNCLP SESSION SET TO Q REPLICATION;
SET LOG "xmlpub.err";
SET SERVER CAPTURE TO DB SAMPLE ID DB2ADMIN PASSWORD "passw0rd";
SET CAPTURE SCHEMA SOURCE EP1;
SET OUTPUT CAPTURE SCRIPT "xmlpub.sql";
SET RUN SCRIPT LATER;

# 2 Creating the XML publication
# This command generates SQL to create an XML publication named DEPARTMENT0001.
# It specifies the DEPARTMENT table as a source. Messages will be sent when any
# column in the source table changes. DELETE operations at the source table will
# not prompt a message to be sent.

CREATE XML PUB USING PUBQMAP SAMPLE_EP1_TO_SUBSCRIBER
(PUBNAME "DEPARTMENT0001" DB2ADMIN.DEPARTMENT ALL CHANGED ROWS Y
SUPPRESS DELETES Y);

# 5 Ending the ASNCLP session.

QUIT;
```

## ASNCLP script 4 (check WebSphere MQ environment)

This script does not generate SQL. Instead, it checks whether the queue manager and queues that were specified in the other scripts exist and whether the objects have the correct properties for event publishing. The script includes commands for the following tasks:

1 Setting the environment
2 Checking the queue managers and queues
3 Ending the ASNCLP session

```
# 1 Setting the environment.
# No SET RUN statement is required. The commands run immediately and send results
# to the command window and log.

ASNCLP SESSION SET TO Q REPLICATION;
SET LOG "epchecks.err";
SET SERVER CAPTURE TO DB SAMPLE ID DB2ADMIN PASSWORD "passw0rd";
SET QMANAGER "QM1" FOR CAPTURE SCHEMA;
SET CAPTURE SCHEMA SOURCE EP1;

# 2 Checking the queue manager and queues.
# These commands check whether the queue manager and queues exist and validate
# their settings. Any errors must be corrected before you start the Q Capture
# program.

VALIDATE WSMQ ENVIRONMENT FOR CAPTURE SCHEMA;
VALIDATE WSMQ ENVIRONMENT FOR PUBQMAP SAMPLE_ASN1_TO_SUBSCRIBER;

# 3 Ending the ASNCLP session.

QUIT;
```

### Output of the script

In addition to the log file, ep.err, this example produces three SQL script files in the same directory where you run the ASNCLP program. Table 13describes the files.

*Table 13. SQL script files that are created by the sample ASNCLP scripts*

| Output file | Contains SQL to ... |
|---|---|
| epcontrol.sql | Create Q Capture control tables |
| pqmap.sql | Create a publishing queue map |
| xmlpub.sql | Create an XML publication |

## ALTER PUBQMAP command

Use the ALTER PUBQMAP command to change attributes for an existing publishing queue map.

### Syntax

```
►►──ALTER PUBQMAP──qmapname──USING──┤ options ├──────────────────────────────────►◄
```

**options:**

```
├──┬─────────────────────┬──┬─────────────────────────────┬──┬──────────────────────┬──►
   └─DESC──"description"─┘  └─MESSAGE CONTENT TYPE──┬─T─┬─┘  └─SENDQ──sendqname─────┘
                                                    └─R─┘

►──┬──────────────────────┬──┬───────────────────────────────┬──┬──────────────────────────┬──►
   └─ERROR ACTION──┬─I─┬──┘  └─HEARTBEAT INTERVAL──interval──┘  └─MAX MESSAGE SIZE──size──┘
                   └─S─┘

►──┬──────────────────────────┬──────────────────────────────────────────────────────────┤
   └─HEADER──┬─NONE────┬──────┘
             └─MQ RFH2─┘
```

### Parameters

*qmapname*
Specifies the name of the publishing queue map.

**DESC** *"description"*
Specifies the description of the publishing queue map.

**MESSAGE CONTENT TYPE**
Specifies whether messages put on the queue will contain an entire database transaction or only a row operation.

**T** Messages contain all of the row operations (update, insert, or delete) within a DB2 transaction, and information about the transaction. This is the default.

**R** Messages contain a single update, insert, or delete operation, and information about the DB2 transaction to which it belongs.

**SENDQ** *sendqname*
Specify to updates the send queue used by the publishing queue map.

**ERROR ACTION**
Specify to tell the Q Capture program what to do when the send queue stops accepting messages. For example, the queue might be full, or the queue manager might have reported a severe error for this queue.

**I** The Q Capture program invalidates all Q subscriptions and XML publications for this queue but continues to put messages on other queues. This is the default.

**S** The Q Capture program stops when an error is detected on this queue.

**HEARTBEAT INTERVAL** *interval*
Specifies the interval (in seconds) between heartbeat messages sent by the Q Capture program to a subscribing application when there are no transactions to publish.

**MAX MESSAGE SIZE** *size*
Specifies the maximum size (in kilobytes) of the buffer that is used for sending messages over the send queue.

**HEADER**
Specifies whether you want a JMS-compliaint MQ RFH2 header added to all XML messages that use the send queue that is specified in this publishing queue map.

**NONE**
Specify to send only the XML publication message with no special headers.

**MQ RFH2**
Specify to attach a special header to the XML message that will contain the topic name that you specify as part of an XML publication.

### Example

To alter the SAMPLE_ASN1_TO_SUBSCRIBER publishing queue map and change the message type from row to transaction, stop the Q Capture program if an error occurs, specify 6 seconds between heartbeat messages, and set the maximum size of the buffer to 64 kilobytes for sending messages over the send queue:

```
ALTER PUBQMAP SAMPLE_ASN1_TO_SUBSCRIBER USING MESSAGE CONTENT TYPE T ERROR ACTION S
HEARTBEAT INTERVAL 6 MAX MESSAGE SIZE 64
```

## ALTER XML PUB command

Use the ALTER XML PUB command to alter an XML publication.

### Syntax

```
►►─ALTER XML PUB─pubname─FOR─┬───────────────┬─source_name─┬─────────────────────┬──────────►
                             └─source_owner.─┘             └─DESC─"description"─┘

►─┬──────────────────┬─┬───────────────────────────┬──────────────────────────────────────►◄
  └─PUBQMAP─qmapname─┘ └─OPTIONS─┤ opt-clause ├─┘
```

**opt-clause:**

```
├─┬──────────────────────────────┬─┬────────────────────────┬─┬──────────────────────┬─────►
  └─SEARCH CONDITION─"search_cond"─┘ └─ALL CHANGED ROWS─┬─N─┬─┘ └─BEFORE VALUES─┬─N─┬─┘
                                                     └─Y─┘                 └─Y─┘
```

```
►─┬────────────────────────┬──┬────────────────────────┬──┬────────────────┬──────────────────────────┤
  └─CHANGED COLS ONLY──┬─Y─┬┘  └─SUPPRESS DELETES──┬─N─┬┘  └─TOPIC──"topic"─┘
                       └─N─┘                       └─Y─┘
```

## Parameters

**PUB** *pubname*
Specifies the name of the XML publication.

*source_owner*
Specifies the source table schema.

*source_name*
Specifies the source table name.

**DESC** *"description"*
Specifies a description of the XML publication.

**PUBQMAP** *qmapname*
Specifies the new name of the publishing queue map that is used by this XML publication.

other-opt-clause:

**SEARCH CONDITION** *"search_cond"*
Specifies a search condition for filtering changes to publish. The change is not sent if the predicate is false. This is an annotated select WHERE clause, where there must be a colon before the column names of the source table. The following example shows a WHERE clause:

```
ALTER XML PUB mypubname FOR ALLTYPE1 OPTIONS
SEARCH CONDITION "WHERE :MYKEY > 1000"
```

**ALL CHANGED ROWS**
Specifies a data sending option.

**Y** Send a row when any column in the source table changes.

**N** Send a row only if a subscribed column in the source table changes.

**BEFORE VALUES**
For an update operation, this keyword indicates whether the Q Capture program sends the before values of non-key columns in addition to their after values. For a delete, this keyword indicates whether the Q Capture program sends the before values of non-key columns in addition to the before values of the key columns.

**N** The Q Capture program does not send before values of non-key columns that change. If a key column changes, the Q Capture program sends both its before and after values. For delete statements involving key columns, only before values are sent. This is the default.

**Y** When there are changes to non-key columns in the source table that are part of an XML publication, the Q Capture program sends both before and after values.

**CHANGED COLS ONLY**
Specifies whether the Q Capture program publishes columns that are part of an XML publication only if they have changed. This keyword only applies to update operations.

**Y** When the Q Capture program sends an updated row, it sends only the changed columns that are part of an XML publication. This is the default.

**N** The Q Capture program sends all columns in a row that are part of an XML publication whenever any of them have changed.

**SUPPRESS DELETES**
Specifies whether to send rows that were deleted from the source table.

**N** Send deleted rows.

**Y** Do not send deleted rows.

**TOPIC** *"topic"*
Specifies the topic that will be included in the MQ RFH2 message header and used by the XML publication. You must specify the HEADER MQ RFH2 keywords when you create the publishing queue map that this XML publication uses.

### Example

To alter the XML publication MYXMLPUB by only sending a row if the subscribed column has changed, sending all columns in a row that are part of the publication whenever any of them have changed, and sending deleted rows:

```
ALTER XML PUB MYXMLPUB FOR ERIC.TSTTABLE OPTIONS ALL CHANGED ROWS N
BEFORE VALUES N CHANGED COLS ONLY N SUPPRESS DELETES N
```

## CREATE PUBQMAP command

Use the CREATE PUBQMAP command to create a publishing queue map that specifies that send queue to use for event publishing.

### Syntax

```
►►──CREATE PUBQMAP──qmapname──┬───────────────────────┬──USING SENDQ──"sendqname"──────────────►
                              └─DESC──"description"───┘

►──┬───────────────────────────┬──┬───────────────────┬──┬─────────────────────────────────┬────►
   └─MESSAGE CONTENT TYPE──┬─T─┤  └─ERROR ACTION──┬─I─┤  └─HEARTBEAT INTERVAL──interval───┘
                           └─R─┘                  └─S─┘

►──┬─────────────────────────┬──┬─────────────────────────┬───────────────────────────────────►◄
   └─MAX MESSAGE SIZE──size──┘  └─HEADER──┬─NONE────┬─────┘
                                          └─MQ RFH2─┘
```

### Parameters

*qmapname*
Specifies the name of the publishing queue map.

**DESC** *"description"*
Specifies the description of the publishing queue map.

**SENDQ** *"sendqname"*
Specifies the name of the WebSphere MQ queue to use as the send queue.

**MESSAGE CONTENT TYPE**
Specifies whether messages put on the queue will contain an entire database transaction or only a row operation.

**T** Messages contain all of the row operations (update, insert, or delete) within a DB2 transaction, and information about the transaction. This is the default.

**R** Messages contain a single update, insert, or delete operation, and information about the DB2 transaction to which it belongs.

**ERROR ACTION**
Specify to tell the Q Capture program what to do when the send queue stops accepting messages. For example, the queue might be full, or the queue manager might have reported a severe error for this queue.

**I** The Q Capture program invalidates all Q subscriptions and XML publications for this queue but continues to put messages on other queues. This is the default.

**S** The Q Capture program stops when an error is detected on this queue.

**HEARTBEAT INTERVAL** *interval*
Specifies the interval (in seconds) between heartbeat messages sent by the Q Capture program to a subscribing application when there are no transactions to publish. To disable heartbeat messages, set the heartbeat interval to 0.

**MAX MESSAGE SIZE** *size*
Specifies the maximum size (in kilobytes) of the buffer used for sending messages over the send queue.

**HEADER**
Specifies whether you want a JMS-compliaint MQ RFH2 header added to all XML messages that use the send queue that is specified in this publishing queue map.

**NONE**
Specify to send only the XML publication message with no special headers.

**MQ RFH2**
Specify to attach a special header to the XML message that will contain the topic name that you specify as part of an XML publication.

### Example

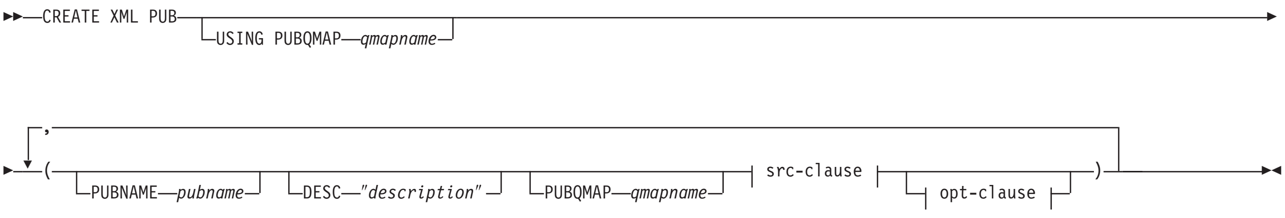
To create a publishing queue map SAMPLE_ASN1_TO_SUBSCRIBER that sets the message content type to row, Specifies 5 seconds between heartbeat messages, and sets a maximum size of buffer at 128 kilobytes for sending messages over the send queue:

```
CREATE PUBQMAP SAMPLE_ASN1_TO_SUBSCRIBER USING
SENDQ "ASN1.QM1.PUBDATAQ" MESSAGE CONTENT TYPE R
ERROR ACTION I HEARTBEAT INTERVAL 5 MAX MESSAGE SIZE 128
```

## CREATE XML PUB command

Use the CREATE XML PUB command to create an XML publication.

### Syntax



**src-clause:**

```
├──┬─┬───────────────┬─source_name────────────────────┬──┬──────────────────────┬──┤
   │ └─source_owner.─┘                                │  └─COLS──┤ col-cause ├─┘
   ├─SRC OWNER LIKE─"predicate1"─┬───────────────────────────┬─┤
   │                             └─SRC NAME LIKE─"predicate2"─┘ │
   │ └─SRC NAME LIKE─"predicate"──────────────────────────────┘ │
   └─SRC ALL──────────────────────────────────────────────────┘
```

**col-cause:**

```
├──┬─ALL─────────────────────────────┬─┬────────────────────────────────┬──────┤
   │              ┌─,──────┐         │ │                ┌─,──────┐      │
   ├─INCLUDE── ( ─▼─colname─┴── ) ───┤ └─ISKEYS── ( ─▼─colname─┴── ) ──┘
   │              ┌─,──────┐         │
   └─EXCLUDE── ( ─▼─colname─┴── ) ───┘
```

**opt-clause:**

```
├──┬────────────────────────────────┬─┬──────────────────────────┬─┬─────────────────────────┬──►
   └─SEARCH CONDITION─"search_cond"─┘ └─ALL CHANGED ROWS──┬─N─┬──┘ └─BEFORE VALUES──┬─N─┬──┘
                                                          └─Y─┘                     └─Y─┘

►─┬───────────────────────────┬─┬──────────────────────┬─┬────────────────────────┬──►
  └─CHANGED COLS ONLY──┬─Y─┬──┘ └─HAS LOAD PHASE──┬─N─┬─┘ └─SUPPRESS DELETES──┬─N─┬─┘
                       └─N─┘                      └─E─┘                      └─Y─┘

►─┬─────────────────┬──────────────────────────────────────────────────────────┤
  └─TOPIC─"topic"───┘
```

## Parameters

**USING PUBQMAP** *qmapname*
: Specifies the publishing queue map that is used by all subsequent XML publications that are created by this command.

**PUBNAME** *pubname*
: Specifies the name of the XML publication.

**DESC** *"description"*
: Specifies a description of the XML publication.

**PUBQMAP** *qmapname*
: Specifies the publishing queue map that is used by this XML publication. If you do not specify the **USING PUBQMAP** keyword, you must define the **PUBQMAP** keyword for every XML publication that you define.

src-clause:

*source_owner*
: Specifies the schema of the source table.

*source_name*
: Specifies the name of the source table.

**SRC OWNER LIKE** *"predicate1"*
Specify to choose all tables with a schema that matches the expression in the LIKE statement. The following examples show LIKE statements:

```
CREATE XML PUB USING PUBQMAP ABCDPUBQMAP
(SRC OWNER LIKE "ASN%");

CREATE XML PUB USING PUBQMAP ABCDPUBQMAP
(SRC OWNER LIKE "JDOE" SRC NAME LIKE "%TAB%");
```

**SRC NAME LIKE** *"predicate2"*
Specify to choose all tables with a name that matches the expression in the LIKE statement. The following example shows a LIKE statement:

```
CREATE XML PUB USING PUBQMAP ABCDPUBQMAP
(SRC NAME LIKE "%4%")
```

**SRC ALL**
Specify to choose all tables, with the exception of DB2 catalog views, that exist on the Q Capture server.

col-cause:

**ALL**
Specify to publish all columns in the source table.

**INCLUDE (***colname***)**
Specifies what columns to publish. You can specify multiple columns.

**EXCLUDE (***colname***)**
Specifies what columns not to publish. You can specify multiple columns.

**ISKEY (***colname***)**
Indicates whether the column is part of the key to use for publishing. Any column or set of columns that are unique at the source can be used. If no column is specified as a key, the Q Capture program looks for a primary key within the set of published columns, then for a unique constraint, and then for a unique index. If none of these exists, Q Capture will use all published, valid columns as key columns for publishing. (Some columns, such as LOB columns, cannot be used as keys.)

opt-clause:

**SEARCH CONDITION** *"search_cond"*
Specifies a search condition for filtering changes to publish. The change is not sent if the predicate is false. This is an annotated select WHERE clause, which requires a colon before the column names. The following example shows a WHERE clause:

```
CREATE XML PUB USING PUBQMAP ASNMAP
(PUBNAME mypubname ALLTYPE1 SEARCH CONDITION
"WHERE :MYKEY > 1000")
```

**ALL CHANGED ROWS**
Specifies a data sending option.

**Y** Send a row when any column in the source table changes.

**N** Send a row only if a subscribed column in the source table changes.

**BEFORE VALUES**
For an update operation, this keyword indicates whether the Q Capture program sends the before values of non-key columns in addition to their after

values. For a delete, this keyword indicates whether the Q Capture program sends the before values of non-key columns in addition to the before values of the key columns.

**N** The Q Capture program does not send before values of non-key columns that change. If a key column changes, the Q Capture program sends both its before and after values. For delete statements involving key columns, only before values are sent. This is the default.

**Y** When there are changes to non-key columns in the source table that are part of an XML publication, the Q Capture program sends both before and after values.

**CHANGED COLS ONLY**

This keyword indicates whether the Q Capture program publishes columns that are part of an XML publication only if they have changed. This field applies to update operations only.

**Y** When the Q Capture program sends an updated row, it sends only the changed columns that are part of an XML publication. This is the default.

**N** The Q Capture program sends all columns in a row that are part of an XML publication whenever any of them has changed.

**HAS LOAD PHASE**

Specifies whether the target table for the XML publication will be loaded with data from the source.

**N** No load phase at the target. This is the default.

**E** External load: Specifies a manual load by an application other outside of replication. In this case, you insert the LOADDONE signal (using the LOADDONE command) into the IBMQREP_SIGNAL table at the Q Capture server to inform the Q Capture program that the application is done loading.

**SUPPRESS DELETES**

Specifies whether to send rows that were deleted from the source table.

**N** Send deleted rows.

**Y** Do not send deleted rows.

**TOPIC** *"topic"*

Specifies the topic that will be included in the MQ RFH2 message header and used by the XML publication. You must specify the HEADER MQ RFH2 keywords when you create the publishing queue map that this XML publication uses.

### Example

To create an XML publication using publishing queue map SAMPLE_ASN1_TO_SUBSCRIBER that publishes a row when any column in the source table changes and does not publish rows that were deleted from the source table:

```
CREATE XML PUB USING PUBQMAP SAMPLE_ASN1_TO_SUBSCRIBER (PUBNAME "EMPLOYEE0001"
DB2ADMIN.EMPLOYEE ALL CHANGED ROWS Y BEFORE VALUES Y CHANGED COLS ONLY Y
HAS LOAD PHASE N SUPPRESS DELETES Y)
```

## DROP PUBQMAP command

Use the DROP PUBQMAP command to delete an existing publishing queue map.

**Restriction:** The XML publications that are using the publishing queue map must first be deleted.

### Syntax

```
►►──DROP PUBQMAP──qmapname──────────────────────────────────────────────────►◄
```

### Parameters

*qmapname*
Specifies the name of the publishing queue map to drop.

### Example

To drop the SAMPLE_ASN1_TO_SUBSCRIBER publishing queue map:

```
DROP PUBQMAP SAMPLE_ASN1_TO_SUBSCRIBER
```

## DROP XML PUB command

Use the DROP XML PUB command to delete an XML publication.

### Syntax

```
►►──DROP XML PUB──┬─ALL─────────────────────────────┬──────────────────────►◄
                  │  ┌─,────────────────────────┐   │
                  │  ▼                          │   │
                  ├─(──PUBNAME──pubname──)──────┴───┤
                  └─FOR PUBNAME LIKE──"predicate"───┘
```

### Parameters

**ALL**
Specify to delete all of the XML publications for the schema and server set through the SET commands.

**PUBNAME** *pubname*
Specifies the name of an XML publication to delete.

**FOR PUBNAME LIKE** *"predicate"*
Specify to delete all XML publications that match the LIKE statement. The following example shows a LIKE statement:

```
DROP XML PUB FOR PUBNAME LIKE "pubname02%"
```

### Example

To delete an XML publication:

```
DROP XML PUB (PUBNAME MYXMLPUB)
```

## START XML PUB command

Use the START XML PUB command to start an XML publication.

### Syntax

```
►►──START──XML PUB──┬─PUBNAME──pubname───────────┬──────────────────────────────────►◄
                    └─FOR PUBNAME LIKE──"%text%"─┘
```

### Parameters

**PUBNAME** *pubname*
: Specifies the name of the XML publication to start.

**FOR PUBNAME LIKE** *"%text%"*
: Specify to start XML publications that match the expression in the LIKE clause. The following example shows a LIKE clause:

```
START XML PUB FOR PUBNAME LIKE "%table%"
```

### Example

To start an XML publication:

```
START XML PUB PUBNAME MYXMLPUB
```

## STOP XML PUB command

Use the STOP XML PUB command to stop an XML publication.

### Syntax

```
►►──STOP──XML PUB──┬─PUBNAME──pubname───────────┬──────────────────────────────────►◄
                   └─FOR PUBNAME LIKE──"%text%"─┘
```

### Parameters

**PUBNAME** *pubname*
: Specifies the name of the XML publication to stop.

**FOR PUBNAME LIKE** *"%text%"*
: Specify to stop XML publications that match the expression in the LIKE clause. The following example shows a LIKE clause:

```
STOP XML PUB FOR PUBNAME LIKE "%table%"
```

### Example

To stop an XML publication:

```
STOP XML PUB PUBNAME MYXMLPUB
```

# Chapter 4. ASNCLP commands for the Replication Alert Monitor

The ASNCLP commands for the Replication Alert Monitor define and change objects such as control tables, contacts, alert conditions, and suspensions.

"Sample ASNCLP scripts for setting up the Replication Alert Monitor" on page 146 demonstrates how you can combine Replication Alert Monitor commands to create an ASNCLP setup script.

Table 14 lists the ASNCLP commands for the Replication Alert Monitor and links to topics that describe each command.

*Table 14. ASNCLP commands for the Replication Alert Monitor*

| If you want to ... | Use this command |
|---|---|
| Change alert conditions for the Apply program | "ALTER ALERT CONDITIONS FOR APPLY command" on page 148 |
| Change alert conditions for the Capture program | "ALTER ALERT CONDITIONS FOR CAPTURE command" on page 151 |
| Change alert conditions for the Q Apply program | "ALTER ALERT CONDITIONS FOR QAPPLY command" on page 153 |
| Change alert conditions for the Q Capture program | "ALTER ALERT CONDITIONS FOR QCAPTURE command" on page 155 |
| Change contact information for notifications | "ALTER CONTACT command" on page 157 |
| Change a contact group | "ALTER GROUP command" on page 158 |
| Change a monitor suspension | "ALTER MONITOR SUSPENSION command" on page 159 |
| Change a monitor suspension template | "ALTER MONITOR SUSPENSION TEMPLATE command" on page 160 |
| Create alert conditions for the Apply program | "CREATE ALERT CONDITIONS FOR APPLY command" on page 161 |
| Create alert conditions for the Capture program | "CREATE ALERT CONDITIONS FOR CAPTURE command" on page 163 |
| Create alert conditions for the Q Apply program | "CREATE ALERT CONDITIONS FOR QAPPLY command" on page 165 |
| Create alert conditions for the Q Capture program | "CREATE ALERT CONDITIONS FOR QCAPTURE command" on page 167 |
| Create contact information for notifications | "CREATE CONTACT command" on page 168 |
| Create the control tables for the Monitor program | "CREATE CONTROL TABLES FOR command (Replication Alert Monitor)" on page 169 |
| Create a contact group | "CREATE GROUP command" on page 171 |
| Create a monitor suspension | "CREATE MONITOR SUSPENSION command" on page 172 |
| Create a monitor suspension template | "CREATE MONITOR SUSPENSION TEMPLATE command" on page 173 |
| Delegate an existing contact to a new contact | "DELEGATE CONTACT command" on page 174 |

*Table 14. ASNCLP commands for the Replication Alert Monitor (continued)*

| If you want to ... | Use this command |
|---|---|
| Delete alert conditions for the Apply program | "DROP ALERT CONDITIONS FOR APPLY command" on page 175 |
| Delete alert conditions for the Capture program | "DROP ALERT CONDITIONS FOR CAPTURE command" on page 175 |
| Delete alert conditions for the Q Apply program | "DROP ALERT CONDITIONS FOR QAPPLY command" on page 176 |
| Delete alert conditions for the Q Capture program | "DROP ALERT CONDITIONS FOR QCAPTURE command" on page 176 |
| Delete an existing contact | "DROP CONTACT command" on page 176 |
| Delete a contact group | "DROP GROUP command" on page 177 |
| Delete a monitor suspension | "DROP MONITOR SUSPENSION command" on page 177 |
| Delete a monitor suspension template | "DROP MONITOR SUSPENSION TEMPLATE command" on page 178 |
| List monitor suspensions | "LIST MONITOR SUSPENSION command" on page 178 |
| List monitor suspension templates | "LIST MONITOR SUSPENSION TEMPLATE command" on page 178 |
| Specify the server (database) used in the ASNCLP session, authentication information, and other required parameters for connecting to the server | "SET SERVER command" on page 179 |
| Substitute one existing contact with another existing contact | "SUBSTITUTE CONTACT command" on page 180 |

## Sample ASNCLP scripts for setting up the Replication Alert Monitor

This sample contains two ASNCLP scripts for setting up the Replication Alert Monitor. It includes Monitor control tables, a contact, and alert conditions.

ASNCLP scripts typically generate one or more SQL scripts to create replication objects. Because some replication objects depend on the existence of other objects, run the ASNCLP scripts and the SQL scripts that they generate in the following order:

1. Monitor control tables
2. Contact and alert conditions
3. Suspension template and suspension

Table 15 on page 148 below the sample describes each SQL script.

This sample has a section for each ASNCLP script, which you can copy to a text file and run by using the `ASNCLP -f` *filename* command. Within the code sample in each section, details about each group of commands are preceded by a comment character (#).

### ASNCLP script 1 (Monitor control tables)

This script generates SQL statements that create Monitor control tables at the SAMPLE database. It includes commands for the following tasks:

```
# 1 Setting the environment.
# The ASNCLP SESSION SET command is not required for the monitor because
# the session must be set to SQL replication and this is the default if no
# command is entered.
# The SET LOG command directs ASNCLP messages to one log file, moncontrol.err.
# The SET OUTPUT command creates an SQL script, moncontrol.sql.
# The SET RUN SCRIPT LATER option allows you to review the SQL scripts before
# they are run.

SET OUTPUT MONITOR SCRIPT "moncontrol.sql";
SET LOG "moncontrol.err";
SET SERVER MONITOR TO DB SAMPLE ID DB2ADMIN PASSWORD "passw0rd";
SET RUN SCRIPT LATER;

# 2 Creating Monitor control tables.

CREATE CONTROL TABLES FOR MONITOR CONTROL SERVER
IN UW OTHERS TSMON1;

# 3 Ending the ASNCLP session.

QUIT;
```

## ASNCLP script 2 (contact and alert conditions)

This script generates SQL statements that define alert conditions for the Monitor and a contact to be alerted when a condition is met. It includes commands for the following tasks:

```
# 1 Setting the environment
# Three SET SERVER commands are required in this script: You set the Monitor
# server to specify which set of Monitor control tables will store information
# about the contact and alert conditions. You set the Capture and target servers
# to specify which servers will be monitored for the alert conditions that you
# will define.
# The SET OUTPUT command creates an SQL script, conalert.sql.

SET OUTPUT MONITOR SCRIPT "conalert.sql";
SET LOG "conalert.err";
SET SERVER MONITOR TO DB SAMPLE ID DB2ADMIN PASSWORD "passw0rd";
SET SERVER CAPTURE TO DB SAMPLE ID DB2ADMIN PASSWORD "passw0rd";
SET SERVER TARGET TO DB TARGET ID DB2ADMIN PASSWORD "passw0rd";
SET RUN SCRIPT LATER;

# 2 Specifying a contact
# The CREATE CONTACT command defines a contact name and specifies that alerts
# be sent to an email address.

CREATE CONTACT repladmin EMAIL "repladmin@us.ibm.com" DESCRIPTION
"Replication administrator";


# 3 Creating alert conditions.
# These commands create alert conditions for the Q Capture program that runs
# at the monitored server SAMPLE and the Q Apply program that runs at the
# monitored server TARGET. The Q Capture conditions trigger an alert if Q Capture
# is down or if any errors or warnings occur. The LATENCY condition triggers
# an alert if the average Q Capture latency exceeds 2 seconds. The Q Apply
# conditions trigger an alert if Q Apply is down or if any errors or warnings
# occur. The EXCEPTIONS condition triggers an alert if a row is added to the
# IBMQREP_EXCEPTIONS table, signaling a SQL error or conflict.

CREATE ALERT CONDITIONS FOR QCAPTURE SCHEMA ASN1 MONITOR QUALIFIER MONQUAL
NOTIFY CONTACT repladmin (STATUS DOWN, ERRORS, WARNINGS, LATENCY 2);
CREATE ALERT CONDITIONS FOR QAPPLY SCHEMA ASN1 MONITOR QUALIFIER MONQUAL
NOTIFY CONTACT repladmin (STATUS DOWN, ERRORS, WARNINGS, EXCEPTIONS);

# 5 Ending the ASNCLP session.

QUIT;
```

### ASNCLP script 3 (suspension template and suspension)

This script generates SQL statements that create a monitor suspension template to define a repeating pattern of monitor suspensions, and a monitor suspension to put the template into effect. It includes commands for the following tasks:

```
# 1 Setting the environment
# Two SET SERVER commands are required in this script: You set the Monitor
# server to specify which set of Monitor control tables will store information
# about the template and suspension. You set the Capture server to specify
# the server where monitoring will be periodically suspended.
# The SET OUTPUT command creates an SQL script, suspend.sql.

SET OUTPUT MONITOR SCRIPT "suspend.sql";
SET LOG "suspend.err";
SET SERVER MONITOR TO DB SAMPLE ID DB2ADMIN PASSWORD "passw0rd";
SET SERVER APPLY TO DB TARGET ID DB2ADMIN PASSWORD "passw0rd";
SET RUN SCRIPT LATER;

# 2 Creating a suspension template
# The CREATE MONITOR SUSPENSION TEMPLATE command creates a template that suspends
# the monitor program during the lunch hour every day.

CREATE MONITOR SUSPENSION TEMPLATE LUNCH START TIME 12:00:00
REPEATS DAILY FOR DURATION 1 HOUR;

# 3 Creating a suspension
# The CREATE MONITOR SUSPSENSION command creates a suspension names S1
# that specifies that monitoring at the TARGET database will be suspended
# every day from 2006-12-10 to 2007-12-31. The command uses the template
# LUNCH to start the suspension at 12:00:00 for a period of hour.

CREATE MONITOR SUSPENSION NAME S1 FOR SERVER TARGET STARTING DATE 2006-12-10
USING TEMPLATE LUNCH ENDING DATE 2007-12-31;

# 5 Ending the ASNCLP session.

QUIT;
```

### Output of the script

In addition to the log file, monitor.err, this example produces three SQL script files in the same directory where you run the ASNCLP program. Table 15 describes the files and where they run.

*Table 15. SQL script files that are created by the sample ASNCLP scripts*

| Output file | Contains SQL to ... |
|---|---|
| moncontrol.sql | Create Monitor control tables |
| conalert.sql | Define a contact and alert conditions |
| suspend.sql | Create a suspension template and suspension |

## ALTER ALERT CONDITIONS FOR APPLY command

Use the ALTER ALERT CONDITIONS FOR APPLY command to alter alert conditions for the Apply program.

### Syntax

```
►►—ALTER ALERT CONDITIONS FOR APPLY—QUALIFIER—qual-name—┬——————————————————┬——————————►
                                                        └SET NAME—set-name┘
```

```
►─MONITOR-QUALIFIER─mon-qual─┬─────────────────┬─(─┬────────────────────────┬─ )──────────►◄
                             └┤ notify-clause ├─┘   ├┤ add-or-remove-clause ├┤
                                                    └┤ change-clause ├───────┘
```

**notify-clause:**

```
├──NOTIFY──┬─CONTACT─contact-name─┬──────────────────────────────────────────────┤
           ├─GROUP─group-name─────┤
           └─OPERATOR CONSOLE─────┘
```

**add-or-remove-clause:**

```
                  ┌─,────────────────────────────┐
                  ▼                              │
├──┬─ADD────┬─────┬─STATUS DOWN─────────────────┬┴──────────────────────────────┤
   └─REMOVE─┘     ├─ERRORS──────────────────────┤
                  ├─WARNINGS────────────────────┤
                  ├─SUBSCRIPTIONS FAILING───────┤
                  ├─SUBSCRIPTIONS DELAYED─time──┤
                  ├─SUBSCRIPTIONS INACTIVE──────┤
                  ├─SUBSCRIPTIONS REFRESHED─────┤
                  ├─TRANSACTIONS REJECTED───────┤
                  ├─REWORKED ROWS─rows──────────┤
                  └─LATENCY─end-end-latency─────┘
```

**change-clause:**

```
              ┌─,────────────────────────────┐
              ▼                              │
├──CHANGE─────┬─SUBSCRIPTIONS DELAYED─time──┬┴──────────────────────────────────┤
              ├─REWORKED ROWS─rows──────────┤
              └─LATENCY─end-end-latency─────┘
```

## Parameters

**APPLY QUALIFIER** *qual-name*
Specifies the Apply qualifier.

**SET NAME** *set-name*
Specifies the subscription set name. If you do not specify a subscription set name, all of the set names in the Apply qualifier will be assumed.

**MONITOR QUALIFIER** *mon-qual*
Specifies the Monitor qualifier.

**NOTIFY**
Specifies the contact or group of contacts to notify when the alert condition occurs.

**CONTACT** *contact-name*
Specifies the contact to notify.

**GROUP** *group-name*
Specifies the group to notify.

**OPERATOR CONSOLE**
Specifies that alert notifications are sent to the z/OS console. This option is valid only if the monitor server is on a z/OS subsystem.

**ADD**
Specify to add an alert condition.

**REMOVE**
Specify to remove an alert condition.

**CHANGE**
Specify to change an alert condition.

**STATUS DOWN**
Specifies whether the Monitor program uses the asnacmd status command to verify that the Apply program is running. The asnacmd status command uses the DB2 Administration Server for non-OS/400 systems. If the Apply program is not running, an alert is sent.

**ERRORS**
Specifies that the Monitor program checks if any error messages were logged in the IBMSNAP_APPLYTRACE table, specifically, any rows that have a value of ERROR for the OPERATION column. If any row is fetched, the DESCRIPTION column is included in the alert.

**WARNINGS**
Specifies that the Monitor program checks if any warnings were logged in the IBMSNAP_APPLYTRACE table, specifically, any rows that have a value of WARNING for the OPERATION column. If any row is fetched, the DESCRIPTION column is included in the alert.

**SUBSCRIPTIONS FAILING**
Specifies whether the Monitor program checks if processed subscription sets finished in error. These subscription set have rows in the IBMSNAP_APPLYTRAIL table with a value of -1 in the STATUS column.

**SUBSCRIPTIONS DELAYED** *time*
Specifies whether the Monitor program checks if subscription sets were processed too late. The determination is based on the following formula: (LAST_RUN + user threshold in seconds > CURRENT TIMESTAMP).

**SUBSCRIPTIONS INACTIVE**
Specifies whether the Monitor program looks for subscription sets made inactive by the Apply program. Such sets are identified by a value of 0 for the ACTIVATE column and -1 for the STATUS column of the IBMSNAP_SUBS_SET table.

**SUBSCRIPTIONS REFRESHED**
Specifies whether the Monitor programs checks if a full refresh has been processed since the last Monitor cycle. See the FULL_REFRESH column in the IBMSNAP_APPLYTRAIL table for this information (rows from the IBMSNAP_APPLYTRAIL table whose values for FULL_REFRESH are 'Y'). If any row is fetched, an alert is sent.

**TRANSACTIONS REJECTED**
Specifies that the Monitor program checks if any conflict has been detected by the Apply program when updating the source table and the replica tables. This check is valid only for subscriptions in an update-anywhere replication environment. See the IBMSNAP_APPLYTRAIL table for this information. If any row is fetched, an alert is sent.

**REWORKED ROWS**
Specifies whether the Monitor program checks if any rows were inserted into

the IBMSNAP_APPLYTRAIL table since the last Monitor cycle for rows reworked in the target table. If the number of rows fetched exceeds the specified value, an alert is sent.

**LATENCY** *end-end-latency*
Specifies whether the Monitor program checks if the total time required to process the data end-to-end (including time it took to capture it) is too high. If the value from the IBMSNAP_APPLYTRAIL table exceeds the specified value, an alert is sent.

### Usage notes

- Specify the alert conditions in parentheses and separate them with commas.
- If you specify the same alert condition twice, the ASNCLP program issues an error.

### Example

To alter an alert condition for the Apply program by removing the condition WARNINGS and no longer alerting the contact REPLADMIN when the condition occurs:

```
ALTER ALERT CONDITIONS FOR APPLY QUALIFIER MYAPPLY01 MONITOR QUALIFIER MONQUAL
NOTIFY REPLADMIN (REMOVE WARNINGS)
```

## ALTER ALERT CONDITIONS FOR CAPTURE command

Use the ALTER ALERT CONDITIONS FOR CAPTURE command to alter alert conditions for the Capture program.

### Syntax

```
►►─ALTER ALERT CONDITIONS FOR CAPTURE─┬──────────────────────┬─MONITOR-QUALIFIER─mon-qual──────────►
                                      └─SCHEMA─cap-schema─┘

►─┬───────────────────┬─(─┬──────────────────────────┬ )────────────────────────────────────────►◄
  └─┤ notify-clause ├─┘   ├─┤ add-or-remove-clause ├─┤
                          └─┤ change-clause ├────────┘
```

#### notify-clause:

```
├──NOTIFY──┬─CONTACT─contact-name─┬──────────────────────────────────────────────────────────────┤
           ├─GROUP─group-name─────┤
           └─OPERATOR CONSOLE─────┘
```

#### add-or-remove-clause:

```
                  ┌─,────────────────────────────┐
                  ▼                              │
├──┬─ADD────┬──────┬─STATUS DOWN────────────────┬┴─────────────────────────────────────────────┤
   └─REMOVE─┘      ├─STATUS LAST COMMIT─time-secs─┤
                   ├─ERRORS─────────────────────┤
                   ├─WARNINGS───────────────────┤
                   ├─CURRENT LATENCY─latency────┤
                   ├─HISTORIC LATENCY─latency───┤
                   └─MEMORY─memory──────────────┘
```

**change-clause:**

```
             ┌─,─────────────────────────────┐
             ▼                               │
├──CHANGE────┬─STATUS LAST COMMIT──time-secs─┬┴──────────────────────────────────────────────────────┤
             ├─CURRENT LATENCY──latency──────┤
             ├─HISTORIC LATENCY──latency─────┤
             └─MEMORY──memory────────────────┘
```

## Parameters

**SCHEMA** *cap-schema*
Specifies the Capture schema for the server that you are monitoring. The default is ASN.

**MONITOR QUALIFIER** *mon-qual*
Specifies the Monitor qualifier.

**NOTIFY**
Specifies the contact or group of contacts to notify when the alert condition occurs.

**CONTACT** *contact-name*
Specifies the contact to notify.

**GROUP** *group-name*
Specifies the group to notify.

**OPERATOR CONSOLE**
Specifies that alert notifications are sent to the z/OS console. This option is valid only if the monitor server is on a z/OS subsystem.

**ADD**
Specify to add an alert condition.

**REMOVE**
Specify to remove an alert condition.

**CHANGE**
Specify to change an alert condition.

**STATUS DOWN**
Specifies whether the Monitor program uses the asnccmd status command to verify that the Capture program is running. The asnccmd status command uses the DB2 Administration Server. If the Capture program is not running, an alert is sent.

**STATUS LAST COMMIT** *time-secs*
Specifies that the Monitor program calculates the difference between the values of the CURRENT TIMESTAMP and CURR_COMMIT_TIME columns of the IBMSNAP_RESTART table. This option has more delay than the STATUS DOWN option, but can be useful if you don't run the DB2 Administration Server at the monitored server. If the calculated difference is greater than the number of seconds specified, an alert is sent.

**ERRORS**
Specifies that the Monitor program checks if any error messages were logged in the IBMSNAP_CAPTRACE table, specifically, any rows that have a value of ERROR for the OPERATION column. If any row is fetched, the DESCRIPTION column is included in the alert.

**WARNINGS**
Specifies that the Monitor program checks if any warnings were logged in the IBMSNAP_CAPTRACE table, specifically, any rows that have a value of WARNING for the OPERATION column. If any row is fetched, the DESCRIPTION column is included in the alert.

**CURRENT LATENCY** *latency*
Specifies that the Monitor program calculates the current latency using the values of the CURR_COMMIT_TIME and MAX_COMMIT_TIME columns in the IBMSNAP_RESTART table. If the latency is greater than the number of seconds specified, an alert is sent.

**HISTORIC LATENCY** *latency*
Specifies that the Monitor program calculates the current latency using the values of the MONITOR_TIME and SYNCHTIME columns in the IBMSNAP_CAPMON table. If the latency is greater than the number of seconds specified, an alert is sent.

**MEMORY** *memory*
Specifies whether the Monitor program selects rows from the IBMSNAP_CAPMON table that were inserted since the last Monitor cycle to verify if the CURRENT_MEMORY column exceeded the specified value.

### Usage notes

- Specify the alert conditions in parentheses and separate them with commas.
- If you specify the same alert condition twice, the ASNCLP program issues an error.

### Example

To alter an alert condition for the Capture program by removing the condition MEMORY and no longer alerting the contact REPLADMIN when the condition occurs:

```
ALTER ALERT CONDITIONS FOR CAPTURE SCHEMA ASN1 MONITOR QUALIFIER MONQUAL
NOTIFY CONTACT REPLADMIN (REMOVE MEMORY 60)
```

## ALTER ALERT CONDITIONS FOR QAPPLY command

Use the ALTER ALERT CONDITIONS FOR QAPPLY command to alter alert conditions for the Q Apply program.

### Syntax

```
►►──ALTER ALERT CONDITIONS FOR QAPPLY──┬─────────────────┬──MONITOR-QUALIFIER──monitor-qualifier──────────►
                                       └─SCHEMA──schema──┘

►──┬───────────────────┬──(──┬─────────────────────────┬──)──────────────────────────────────────────────►◄
   └─┤ notify-clause ├─┘     ├─┤ add-or-remove-clause ├─┤
                             └─┤ change-clause ├────────┘
```

**notify-clause:**

```
├──NOTIFY──┬──CONTACT──contact-name──┬──────────────────────────────────────────────────────────────────┤
           ├──GROUP──group-name──────┤
           └──OPERATOR CONSOLE───────┘
```

**add-or-remove-clause:**

```
              ┌─,──────────────────────────────┐
              ▼                                │
├──┬─ADD────┬────┬─ STATUS DOWN──────────────┬──┴──────────────────────────────────────────┤
   └─CHANGE─┘    ├─ERRORS────────────────────┤
                 ├─WARNINGS──────────────────┤
                 ├─LATENCY──seconds──────────┤
                 ├─EELATENCY──seconds────────┤
                 ├─MEMORY──megabytes─────────┤
                 ├─EXCEPTIONS────────────────┤
                 ├─SPILL QUEUES DEPTH──percentage─┤
                 └─QUEUE DEPTH──queue-percent─────┘
```

**change-clause:**

```
            ┌─,──────────────────────────────┐
            ▼                                │
├──CHANGE─────┬─LATENCY──seconds──────────┬──┴──────────────────────────────────────────┤
              ├─EELATENCY──seconds────────┤
              ├─MEMORY──megabytes─────────┤
              ├─SPILL QUEUES DEPTH──percentage─┤
              └─QUEUE DEPTH──queue-percent─────┘
```

## Parameters

**SCHEMA** *schema*
Specifies the Q Apply schema that qualifies the process to be monitored. The default is ASN.

**MONITOR QUALIFIER** *monitor-qualifier*
Specifies the monitor qualifier that groups the alert conditions:

**ADD**
Specify to add an alert condition.

**REMOVE**
Specify to remove an alert condition.

**CHANGE**
Specify to change an alert condition.

**STATUS DOWN**
Specifies that the Monitor program will use the asnqccmd status command to verify if the Q Apply program is down.

**ERRORS**
Specifies that the Monitor program check if error messages were logged in the IBMQREP_APPLYTRACE table.

**WARNINGS**
Specifies that the Monitor program checks if any warnings were logged in the IBMSNAP_CAPTRACE table, specifically, any rows that have a value of WARNING for the OPERATION column. If any row is fetched, the DESCRIPTION column is included in the alert.

**LATENCY** *seconds*
Specifies that an alert will be sent when the difference in seconds of MONITOR_TIME and CURRENT_LOG_TIME in the IBMQREP_APPLYMON table exceeds the number of seconds specified.

**EELATENCY** *seconds*
Specifies that an alert will be sent when the value of the column END2END_LATENCY (in milliseconds) in the IBMQREP_APPLYMON table exceeds the number of milliseconds specified.

**MEMORY** *megabytes*
Specifies that the Monitor process will select rows from the IBMQREP_APPLYMON table that were inserted since the last Monitor cycle to verify if the CURRENT_MEMORY column exceeded the number of megabytes specified.

**EXCEPTIONS**
Specifies that an alert will be sent if there is any row in the IBMQREP_EXCEPTIONS table.

**SPILL QUEUES DEPTH** *percentage*
Specifies that the Monitor program will check whether the percentage of fullness of the spill queue is greater than specified percentage. The Monitor program checks this percentage only when any Q subscription is on the load state (the value of the STATE column in the IBMQREP_TARGETS table is L, D, F, or E).

**QUEUE DEPTH** *queue_percent*
Specifies that an alert will be sent when the specified percentage of the given queue is full.

notify-clause:

**CONTACT** *contact_name*
Specifies the contact to notify when a defined alert condition is detected.

**GROUP** *group_name*
Specifies the group to notify when a defined alert condition is detected.

**OPERATOR CONSOLE**
Specifies that alert notifications are sent to the z/OS console. This option is valid only if the monitor server is on a z/OS subsystem.

### Example

To alter an alert condition for the Q Apply program by removing the condition EXCEPTIONS and no longer alerting the contact REPLADMIN when the condition occurs:

```
ALTER ALERT CONDITIONS FOR QAPPLY MONITOR QUALIFIER MONQUAL
 NOTIFY REPLADMIN (REMOVE EXCEPTIONS)
```

## ALTER ALERT CONDITIONS FOR QCAPTURE command

Use the ALTER ALERT CONDITIONS FOR QCAPTURE command to alter the alert conditions for the Q Capture program.

### Syntax

```
►►──ALTER ALERT CONDITIONS FOR QCAPTURE──┬─────────────────┬──MONITOR-QUALIFIER──monitor-qualifier──────►
                                         └─SCHEMA──schema──┘
```

```
►─┬─────────────────┬─(─┬──┤ add-or-remove-clause ├─┬─ )──────────────────────────►◄
  └─┤ notify-clause ├─┘   └──┤ change-clause ├────────┘
```

**notify-clause:**

```
├──NOTIFY──┬─CONTACT──contact-name─┬──────────────────────────────────────────────────┤
           ├─GROUP──group-name─────┤
           └─OPERATOR CONSOLE──────┘
```

**add-or-remove-clause:**

```
             ┌─,───────────────────────────────┐
             ▼                                 │
├──┬─ADD────┬───┬─ STATUS DOWN────────────────┬┴──────────────────────────────────────┤
   └─CHANGE─┘   ├─ERRORS──────────────────────┤
                ├─WARNINGS────────────────────┤
                ├─LATENCY──seconds────────────┤
                ├─MEMORY──megabytes───────────┤
                ├─TRANSACTION SIZE──megabytes─┤
                ├─SUBSCRIPTIONS INACTIVE──────┤
                └─QUEUE DEPTH──queue-percent──┘
```

**change-clause:**

```
          ┌─,───────────────────────────────┐
          ▼                                 │
├──CHANGE───┬─LATENCY──seconds────────────┬─┴─────────────────────────────────────────┤
            ├─MEMORY──megabytes───────────┤
            ├─TRANSACTION SIZE──megabytes─┤
            └─QUEUE DEPTH──queue-percent──┘
```

## Parameters

**SCHEMA** *schema*
Specifies the Q Capture schema that qualifies the process to be monitored. The default is ASN.

**MONITOR QUALIFIER** *monitor-qualifier*
Specifies the monitor qualifier that groups the alert conditions.

**ADD**
Specify to add an alert condition.

**REMOVE**
Specify to remove an alert condition.

**CHANGE**
Specify to change an alert condition.

**STATUS DOWN**
Specifies that the Monitor program will use the asnqccmd status command to verify if the Q Capture program is down.

**ERRORS**
Specifies that the Monitor program check if error messages were logged in the IBMQREP_CAPTRACE table.

**WARNINGS**
Specifies that the Monitor program checks if any warnings were logged in the IBMSNAP_CAPTRACE table, specifically, any rows that have a value of WARNING for the OPERATION column. If any row is fetched, the DESCRIPTION column is included in the alert.

**LATENCY** *seconds*
Specifies that an alert will be sent when the difference in seconds of MONITOR_TIME and CURRENT_LOG_TIME in the IBMQREP_CAPMON table exceeds the number of seconds specified.

**MEMORY** *megabytes*
Specifies that the Monitor process will select rows from the IBMQREP_CAPMON table that were inserted since the last Monitor cycle to verify if the CURRENT_MEMORY column exceeded the number of megabytes specified.

**TRANSACTION SIZE** *megabytes*
Specifies that the Monitor process will select rows for the IBMSNAP_CAPMON table to verify if any transaction size exceeded the number of megabytes specified.

**SUBSCRIPTIONS INACTIVE**
Specifies that an alert will be sent when the value of the STATE column in the IBMQREP_SUBS table is I.

**QUEUE DEPTH** *queue-percent*
Specifies that an alert will be sent when the specified percentage of the given queue is full.

notify-clause:

**CONTACT** *contact_name*
Specifies the contact to notify when a defined alert condition is detected.

**GROUP** *group-name*
Specifies the group to notify when a defined alert condition is detected.

**OPERATOR CONSOLE**
Specifies that alert notifications are sent to the z/OS console. This option is valid only if the monitor server is on a z/OS subsystem.

### Example

To alter an alert condition for the Q Capture program by removing the condition MEMORY and no longer alerting the contact REPLADMIN when the condition occurs:

```
ALTER ALERT CONDITIONS FOR QCAPTURE SCHEMA ASN1 MONITOR QUALIFIER MONQUAL
 NOTIFY CONTACT REPLADMIN (REMOVE MEMORY 60)
```

## ALTER CONTACT command

Use the ALTER CONTACT command to alter contact information, such as the contact name and mail address, that the Replication Alert Monitor program uses for notifications when a replication alert condition is detected.

### Syntax

```
►►──ALTER CONTACT──contact-name──┬─EMAIL─┬──"email-address"──┬──────────────────────────────┬──────►◄
                                 └─PAGE──┘                   └─DESCRIPTION──"description"─┘
```

### Parameters

**CONTACT** *contact-name*
Specifies the name of the contact. The contact must exist.

**EMAIL** *"email-address"*
Specifies the primary e-mail address for the contact. The double quotation marks are required.

**PAGE** *"email-address"*
Specifies the pager address for the contact. The double quotation marks are required.

**DESCRIPTION** *"description"*
Specifies a brief description for the contact. The double quotation marks are required.

### Example

To alter a contact REPLADMIN by changing the e-mail address to repladmin@ibm.com:

```
ALTER CONTACT REPLADMIN EMAIL "repladmin@ibm.com"
```

## ALTER GROUP command

Use the ALTER GROUP command to alter a group of replication monitor contacts.

### Syntax

```
►►──ALTER GROUP──group-name──┬──────────────────────────────┬──────────────────────────►
                             └─DESCRIPTION──"description"─┘


               ┌─,──────────────┐
               ▼                │
►──┬─NEW CONTACTS───contact-name1─┴────────────┬───────────────────────────►◄
   │             ┌─,──────────────────────┐   │
   │             ▼                        │   │
   └─CONTACTS───contact-name2──┬─ADD────┬─┴───┘
                               └─REMOVE─┘
```

### Parameters

*group-name*
Specifies the name of the group. The group must exist.

**DESCRIPTION** *"description"*
Specifies a brief description for the group. The double quotation marks are required.

**NEW CONTACTS** *contact-name1*
Specifies a comma-separated list of contacts that belong to this group. This list overwrites the existing list of contacts for the group.

**CONTACTS** *contact-name2*

**ADD**
Specifies a comma-separated list of contacts to add to this group.

**REMOVE**
Specifies a comma-separated list of contacts to remove from this group.

### Example

To alter a group MAINTENANCE by removing a contact PERFORMANCE:

```
ALTER GROUP MAINTENANCE CONTACTS PERFORMANCE REMOVE
```

## ALTER MONITOR SUSPENSION command

Use the ALTER MONITOR SUSPENSION command to specify a different template for the monitor suspension, to change the start or end date for using the template, or to change the start or end date for suspending the monitor program if you do not use a template.

### Syntax

```
►►─ALTER MONITOR SUSPENSION─name─┬───────────────────────────┬──┬─────────────────────┬──────────────►
                                 └─TEMPLATE─template_name─┘  └─STARTING DATE─date─┘

►─┬───────────────────┬──────────────────────────────────────────────────────────────────────────►◄
  └─ENDING DATE─date─┘
```

### Parameters

**TEMPLATE**
Specifies the template that you want to use for this suspension.

**STARTING DATE**
Specifies one of two different values, depending on whether you use a template for the suspension:

**With template**
Specifies the date that you want to start using the monitor suspension template.

**Without template**
Specifies the date on which the monitor program will be suspended. Use YYYY-MM-DD format.

**ENDING DATE**
Specifies one of two different values, depending on whether you use a template for the suspension:

**With template**
Specifies the date that you want to stop using the monitor suspension template.

**Without template**
Specifies the date when the monitor suspension ends. Use YYYY-MM-DD format.

### Usage notes

To initiate the change, use the asnmcmd reinit command, or stop and start the monitor program.

### Example 1

To change the suspension S1 so that it uses a different template, SATURDAY, and applies the template starting 2006-12-09:

```
ALTER MONITOR SUSPENSION NAME S1 TEMPLATE SATURDAY STARTING DATE 2006-12-09
```

### Example 2

To change the suspension S2 so that it uses a template, LUNCH1, starting 2007-01-01 and ending 2007-06-30:

```
ALTER MONITOR SUSPENSION NAME S2 TEMPLATE LUNCH1 STARTING DATE 2007-01-01
ENDING DATE 2007-06-30
```

## ALTER MONITOR SUSPENSION TEMPLATE command

Use the ALTER MONITOR SUSPENSION TEMPLATE command to change the frequency and duration of periods that the monitor program is suspended.

### Syntax

```
►►──ALTER MONITOR SUSPENSION TEMPLATE──template_name──┬──────────────────────┬────────────────────────►
                                                      └─START TIME──HH:MM:SS─┘

►─REPEATS──┤ occurrence-clause ├─────────────────────────────────────────────────────────────────────►◄
```

**occurrence-clause:**

```
                 ┌─MINUTES─┐
├──┬─DAILY──FOR DURATION──n──┴─HOURS───┴──────────────────────────┬────────────────────────────────┤
   │                                                 ┌─MINUTES─┐  │
   └─WEEKLY──DAY OF WEEK──┬─SUNDAY────┬──FOR DURATION──n──┼─HOURS───┼──┘
                          ├─MONDAY────┤                    └─DAYS────┘
                          ├─TUESDAY───┤
                          ├─WEDNESDAY─┤
                          ├─THURSDAY──┤
                          ├─FRIDAY────┤
                          └─SATURDAY──┘
```

### Parameters

**START TIME**
Specifies the time at which the monitor program will be suspended. Use HH:MM:SS format. The default value is 00:00:00.

**REPEATS**
Specifies which days the monitor program will be suspended and for how long.

### Usage notes

To initiate the change, use the asnmcmd reinit command, or stop and start the monitor program.

### Example 1

To change a template so that it suspends the monitor program from 00:00:00 to 03:00:00 every SUNDAY for one year:

```
ALTER MONITOR SUSPENSION TEMPLATE sunday START TIME 00:00:00 REPEATS WEEKLY
DAY OF WEEK SUNDAY FOR DURATION 3 HOURS
```

### Example 2

To lengthen a template that suspends the monitor program during the lunch hour every day to 90 minutes:

```
ALTER MONITOR SUSPENSION TEMPLATE lunch START TIME 12:00:00 REPEATS DAILY
FOR DURATION 90 MINUTES
```

## CREATE ALERT CONDITIONS FOR APPLY command

Use the CREATE ALERT CONDITIONS FOR APPLY command to create alert conditions for the Apply program. Each entry represents a condition that the Replication Alert Monitor program looks for. If the condition is true, the Monitor program sends an alert to the corresponding contact or group, or to the operator console.

### Syntax

```
►►──CREATE ALERT CONDITIONS FOR APPLY──QUALIFIER──qual-name──┬─────────────────────┬──────────────►
                                                              └─SET NAME──set-name─┘

►─MONITOR-QUALIFIER──mon-qual──NOTIFY──┬─CONTACT──contact-name─┬──────────────────────────────►◄
                                       ├─GROUP──group-name─────┤
                                       └─OPERATOR CONSOLE──────┘


►►──(──┬─STATUS DOWN──────────────────┬─)──────────────────────────────────────────────────►◄
       ├─ERRORS───────────────────────┤
       ├─WARNINGS─────────────────────┤
       ├─SUBSCRIPTIONS FAILING────────┤
       ├─SUBSCRIPTIONS DELAYED──time──┤
       ├─SUBSCRIPTIONS INACTIVE───────┤
       ├─SUBSCRIPTIONS REFRESHED──────┤
       ├─TRANSACTIONS REJECTED────────┤
       ├─REWORKED ROWS──rows──────────┤
       └─LATENCY──end-end-latency─────┘
```

### Parameters

**APPLY QUALIFIER** *qual-name*
Specifies the Apply qualifier.

**SET NAME** *set-name*
Specifies the subscription set name. If you do not specify a subscription set name, all of the set names in the Apply qualifier will be assumed.

**MONITOR QUALIFIER** *mon-qual*
Specifies the Monitor qualifier.

**NOTIFY**
Specifies the contact or group of contacts to notify when the alert condition occurs.

**CONTACT** *contact-name*
Specifies the contact to notify.

**GROUP** *group-name*
Specifies the group to notify.

**OPERATOR CONSOLE**
Specifies that alert notifications are sent to the z/OS console. This option is valid only if the monitor server is on a z/OS subsystem.

**STATUS DOWN**
Specifies whether the Monitor program uses the asnacmd status command to verify that the Apply program is running. The asnacmd status command uses the DB2 Administration Server for non-OS/400 systems. If the Apply program is not running, an alert is sent.

**ERRORS**
Specifies that the Monitor program checks if any error messages were logged in the IBMSNAP_APPLYTRACE table, specifically, any rows that have a value of ERROR for the OPERATION column. If any row is fetched, the DESCRIPTION column is included in the alert.

**WARNINGS**
Specifies that the Monitor program checks if any warnings were logged in the IBMSNAP_APPLYTRACE table, specifically, any rows that have a value of WARNING for the OPERATION column. If any row is fetched, the DESCRIPTION column is included in the alert.

**SUBSCRIPTIONS FAILING**
Specifies whether the Monitor program checks if processed subscription sets finished in error. These subscription set have rows in the IBMSNAP_APPLYTRAIL table with a value of -1 in the STATUS column.

**SUBSCRIPTIONS DELAYED** *time*
Specifies whether the Monitor program checks if subscription sets were processed too late. The determination is based on the following formula: (LAST_RUN + user threshold in seconds > CURRENT TIMESTAMP).

**SUBSCRIPTIONS INACTIVE**
Specifies whether the Monitor program looks for subscription sets made inactive by the Apply program. Such sets are identified by a value of 0 for the ACTIVATE column and -1 for the STATUS column of the IBMSNAP_SUBS_SET table.

**SUBSCRIPTIONS REFRESHED**
Specifies whether the Monitor programs checks if a full refresh has been processed since the last Monitor cycle. See the FULL_REFRESH column in the IBMSNAP_APPLYTRAIL table for this information (rows from the IBMSNAP_APPLYTRAIL table whose values for FULL_REFRESH are ’Y’). If any row is fetched, an alert is sent.

**TRANSACTIONS REJECTED**
Specifies that the Monitor program checks if any conflict has been detected by the Apply program when updating the source table and the replica tables. This check is valid only for subscriptions in an update-anywhere replication environment. See the IBMSNAP_APPLYTRAIL table for this information. If any row is fetched, an alert is sent.

**REWORKED ROWS** *rows*
Specifies whether the Monitor program checks if any rows were inserted into

the IBMSNAP_APPLYTRAIL table since the last Monitor cycle for rows reworked in the target table. If the number of rows fetched exceeds the specified value, an alert is sent.

**LATENCY** *end-end-latency*
Specifies whether the Monitor program checks if the total time required to process the data end-to-end (including time it took to capture it) is too high. If the value from the IBMSNAP_APPLYTRAIL table exceeds the specified value, an alert is sent.

### Usage notes

- Specify the alert conditions in parentheses and separate them with commas.
- If you specify the same alert condition twice, the ASNCLP program issues an error.

### Example

To create alert conditions for the Apply program that sends an alert to the contact REPLADMIN when a condition occurs:

```
CREATE ALERT CONDITIONS FOR APPLY QUALIFIER MYAPPLY01 MONITOR QUALIFIER MONQUAL
NOTIFY CONTACT REPLADMIN (STATUS DOWN, ERRORS, WARNINGS, SUBSCRIPTIONS FAILING,
SUBSCRIPTIONS DELAYED 300, SUBSCRIPTIONS INACTIVE, SUBSCRIPTIONS REFRESHED,
TRANSACTIONS REJECTED, REWORKED ROWS 2, LATENCY 360)
```

## CREATE ALERT CONDITIONS FOR CAPTURE command

Use the CREATE ALERT CONDITIONS FOR CAPTURE command to create alert conditions for the Capture program. Each entry represents a condition that the Replication Alert Monitor program looks for. If the condition is true, the Monitor program sends an alert to the corresponding contact or group, or to the operator console.

### Syntax

```
►►──CREATE ALERT CONDITIONS FOR CAPTURE──┬────────────────────────┬──MONITOR QUALIFIER──mon-qual──────────►
                                         └─SCHEMA──cap-schema─────┘


                              ┌─,──────────────────────────────────┐
                              ▼                                    │
►─NOTIFY──┬─CONTACT──contact-name─┬── (────┬─STATUS DOWN────────────────────┬──┴── )──────────────────────►◄
          ├─GROUP──group-name─────┤        ├─STATUS LAST COMMIT──time-secs──┤
          └─OPERATOR CONSOLE──────┘        ├─ERRORS─────────────────────────┤
                                           ├─WARNINGS───────────────────────┤
                                           ├─CURRENT LATENCY──latency───────┤
                                           ├─HISTORIC LATENCY──latency──────┤
                                           └─MEMORY──memory─────────────────┘
```

### Parameters

**SCHEMA** *cap-schema*
Specifies the Capture schema for the server that you are monitoring. The default is ASN.

**MONITOR QUALIFIER** *mon-qual*
Specifies the Monitor qualifier.

**NOTIFY**
Specifies the contact or group of contacts to notify when the alert condition occurs.

**CONTACT** *contact-name*
Specifies the contact to notify.

**GROUP** *group-name*
Specifies the group to notify.

**OPERATOR CONSOLE**
Specifies that alert notifications are sent to the z/OS console. This option is valid only if the monitor server is on a z/OS subsystem.

**STATUS DOWN**
Specifies whether the Monitor program uses the asnccmd status command to verify that the Capture program is running. The asnccmd status command uses the DB2 Administration Server. If the Capture program is not running, an alert is sent.

**STATUS LAST COMMIT** *time-secs*
Specifies that the Monitor program calculates the difference between the values of the CURRENT TIMESTAMP and CURR_COMMIT_TIME columns of the IBMSNAP_RESTART table. This option has more delay than the STATUS DOWN option, but can be useful if you do not run the DB2 Administration Server at the monitored server. If the calculated difference is greater than the number of seconds specified, an alert is sent.

**ERRORS**
Specifies that the Monitor program checks if any error messages were logged in the IBMSNAP_CAPTRACE table, specifically, any rows that have a value of ERROR for the OPERATION column. If any row is fetched, the DESCRIPTION column is included in the alert.

**WARNINGS**
Specifies that the Monitor program checks if any warnings were logged in the IBMSNAP_CAPTRACE table, specifically, any rows that have a value of WARNING for the OPERATION column. If any row is fetched, the DESCRIPTION column is included in the alert.

**CURRENT LATENCY** *latency*
Specifies that the Monitor program calculates the current latency using the values of the CURR_COMMIT_TIME and MAX_COMMIT_TIME columns in the IBMSNAP_RESTART table. If the latency is greater than the number of seconds specified, an alert is sent.

**HISTORIC LATENCY** *latency*
Specifies that the Monitor program calculates the current latency using the values of the MONITOR_TIME and SYNCHTIME columns in the IBMSNAP_CAPMON table. If the latency is greater than the number of seconds specified, an alert is sent.

**MEMORY** *memory*
Specifies whether the Monitor program selects rows from the IBMSNAP_CAPMON table that were inserted since the last Monitor cycle to verify if the CURRENT_MEMORY column exceeded the specified value.

## Usage notes

If you specify the same alert condition twice, the ASNCLP program issues an error.

### Example

To create alert conditions for the Capture program that sends an alert to the contact REPLADMIN when a condition occurs:

```
CREATE ALERT CONDITIONS FOR CAPTURE QUALIFIER MYAPPLY01 MONITOR QUALIFIER MONQUAL
NOTIFY CONTACT REPLADMIN (STATUS DOWN, ERRORS, WARNINGS, SUBSCRIPTION FAILING,
SUBSCRIPTION DELAYED 300, SUBSCRIPTIONS INACTIVE, SUBSCRIPTIONS REFRESHED,
TRANSACTION REJECTED, REWORKED ROWS 2, LATENCY 360)
```

## CREATE ALERT CONDITIONS FOR QAPPLY command

Use the CREATE ALERT CONDITIONS FOR QAPPLY command to create alert conditions for the Q Apply program. Each entry represents a condition that the Replication Alert Monitor program looks for. If the condition is true, the Monitor program sends an alert to the corresponding contact or group, or to the operator console.

### Syntax

```
►►──CREATE ALERT CONDITIONS FOR QAPPLY──┬───────────────┬──MONITOR-QUALIFIER──monitor-qualifier──────►
                                        └─SCHEMA──schema─┘


                                         ┌─,──────────────────────────┐
                                         ▼                            │
►──┬─────────────────────────────┬──(───┬─STATUS DOWN──────────────┬─┴── )──────────────►◄
   └─┤ notification-list-definition ├─┘  ├─ERRORS───────────────────┤
                                         ├─WARNINGS─────────────────┤
                                         ├─LATENCY──seconds─────────┤
                                         ├─EELATENCY──seconds───────┤
                                         ├─MEMORY──megabytes────────┤
                                         ├─EXCEPTIONS───────────────┤
                                         ├─SPILL QUEUES DEPTH──percentage─┤
                                         └─QUEUE DEPTH──queue-percent─────┘
```

**notification-list-definition:**

```
├──NOTIFY──┬─CONTACT──contact-name─┬──────────────────────────────────────────┤
           ├─GROUP──group-name─────┤
           └─OPERATOR CONSOLE──────┘
```

### Parameters

**SCHEMA** *schema*
Specifies the Q Apply schema that qualifies the process to be monitored. The default is ASN.

**MONITOR QUALIFIER** *monitor-qualifier*
Specifies the monitor qualifier that groups the alert conditions:

**STATUS DOWN**
Specifies that the Monitor program will use the asnqccmd status command to verify if the Q Apply program is down.

**ERRORS**
Specifies that the Monitor program check if error messages were logged in the IBMQREP_APPLYTRACE table.

**WARNINGS**
Specifies that the Monitor program checks if any warnings were logged in the IBMSNAP_CAPTRACE table, specifically, any rows that have a value of WARNING for the OPERATION column. If any row is fetched, the DESCRIPTION column is included in the alert.

**LATENCY** *seconds*
Specifies that an alert will be sent when the difference in seconds of MONITOR_TIME and CURRENT_LOG_TIME in the IBMQREP_APPLYMON table exceeds the number of seconds specified.

**EELATENCY** *seconds*
Specifies that an alert will be sent when the value of the column END2END_LATENCY (in milliseconds) in the IBMQREP_APPLYMON table exceeds the number of milliseconds specified.

**MEMORY** *megabytes*
Specifies that the Monitor process will select rows from the IBMQREP_APPLYMON table that were inserted since the last Monitor cycle to verify if the CURRENT_MEMORY column exceeded the number of megabytes specified.

**EXCEPTIONS**
Specifies that an alert will be sent if there is any row in the IBMQREP_EXCEPTIONS table.

**SPILL QUEUES DEPTH** *percentage*
Specifies that the Monitor program will check whether the percentage of fullness of the spill queue is greater than specified percentage. The Monitor program checks this percentage only when any Q subscription is on the load state (the value of the STATE column in the IBMQREP_TARGETS table is L, D, F, or E).

**QUEUE DEPTH** *queue-percent*
Specifies that an alert will be sent when the specified percentage of the given queue is full.

notification-list-definition:

**CONTACT** *contact_name*
Specifies the contact to notify when a defined alert condition is detected.

**GROUP** *group-name*
Specifies the group to notify when a defined alert condition is detected.

**OPERATOR CONSOLE**
Specifies that alert notifications are sent to the z/OS console. This option is valid only if the monitor server is on a z/OS subsystem.

## Example

To create alert conditions for the Q Apply program that send an alert to the contact REPLADMIN when a condition occurs:

```
CREATE CONDITIONS FOR QAPPLY MONITOR QUALIFIER MONQUAL
 NOTIFY CONTACT REPLADMIN (STATUS DOWN, ERRORS, WARNINGS
LATENCY 360, EXCEPTIONS)
```

## CREATE ALERT CONDITIONS FOR QCAPTURE command

Use the CREATE ALERT CONDITIONS FOR QCAPTURE command to create alert conditions for the Q Capture program. Each entry represents a condition that the Replication Alert Monitor program looks for. If the condition is true, the Monitor program sends an alert to the corresponding contact or group, or to the operator console.

### Syntax

```
►►──CREATE ALERT CONDITIONS FOR QCAPTURE──┬───────────────┬──MONITOR QUALIFIER──monitor-qualifier──────►
                                          └─SCHEMA──schema─┘


                                           ┌─,────────────────────────┐
                                           ▼                          │
►─┤ notification-list-definition ├──(────┬─STATUS DOWN──────────────┬─┴─ )──────────────────────────►◄
                                         ├─ERRORS───────────────────┤
                                         ├─WARNINGS─────────────────┤
                                         ├─LATENCY──seconds─────────┤
                                         ├─MEMORY──megabytes────────┤
                                         ├─TRANSACTION SIZE──megabytes─┤
                                         ├─SUBSCRIPTIONS INACTIVE───┤
                                         └─QUEUE DEPTH──queue_percent─┘
```

**notification-list-definition:**

```
├──NOTIFY───┬─CONTACT──contact-name─┬──────────────────────────────────────────────────────┤
            ├─GROUP──group-name─────┤
            └─OPERATOR CONSOLE──────┘
```

### Parameters

**SCHEMA** *schema*
Specifies the Q Capture schema that qualifies the process to be monitored. The default is ASN.

**MONITOR QUALIFIER** *monitor-qualifier*
Specifies the monitor qualifier that groups the alert conditions.

**STATUS DOWN**
Specifies that the Monitor program will use the asnqccmd status command to verify if the Q Capture program is down.

**ERRORS**
Specifies that the Monitor program check if error messages were logged in the IBMQREP_CAPTRACE table.

**WARNINGS**
Specifies that the Monitor program checks if any warnings were logged in the IBMSNAP_CAPTRACE table, specifically, any rows that have a value of WARNING for the OPERATION column. If any row is fetched, the DESCRIPTION column is included in the alert.

**LATENCY** *seconds*
Specifies that an alert will be sent when the difference in seconds of

MONITOR_TIME and CURRENT_LOG_TIME in the IBMQREP_CAPMON table exceeds the number of seconds specified.

**MEMORY** *megabytes*
Specifies that the Monitor process will select rows from the IBMQREP_CAPMON table that were inserted since the last Monitor cycle to verify if the CURRENT_MEMORY column exceeded the number of megabytes specified.

**TRANSACTION SIZE** *megabytes*
Specifies that the Monitor process will select rows for the IBMSNAP_CAPMON table to verify if any transaction size exceeded the number of megabytes specified.

**SUBSCRIPTIONS INACTIVE**
Specifies that an alert will be sent when the value of the STATE column in the IBMQREP_SUBS table is I.

**QUEUE DEPTH** *queue-percent*
Specifies that an alert will be sent when the specified percentage of the given queue is full.

notification-list-definition:

**CONTACT** *contact_name*
Specifies the contact to notify when a defined alert condition is detected.

**GROUP** *group-name*
Specifies the group to notify when a defined alert condition is detected.

**OPERATOR CONSOLE**
Specifies that alert notifications are sent to the z/OS console. This option is valid only if the monitor server is on a z/OS subsystem.

### Example

To create alert conditions for the Q Capture program that sends an alert to the contact REPLADMIN when a condition occurs:

```
CREATE ALERT CONDITIONS FOR QCAPTURE SCHEMA ASN1 MONITOR QUALIFIER MONQUAL
NOTIFY CONTACT REPLADMIN (STATUS DOWN, ERRORS, WARNINGS, LATENCY 30, MEMORY 60)
```

## CREATE CONTACT command

Use the CREATE CONTACT command to create contact information, such as the contact name and e-mail address, that the Replication Alert Monitor program uses for notifications when a replication alert condition is detected. You can optionally associate a contact to a pre-existing group.

### Syntax

```
     ┌,────────────────────────────────────────────────┐
     ▼                                                 │
►►───CREATE CONTACT──contact-name──┬───────────────────┬┴──┬─EMAIL─┬──"email-address"──────────►
                                   └─GROUP──group-name─┘   └─PAGE──┘
```

```
►─┬───────────────────────────┬────────────────────────────────────────────────────────►◄
  └─DESCRIPTION─"description"─┘
```

## Parameters

**CONTACT** *contact-name*
Specifies the name of the contact. This name cannot match another contact already defined.

**GROUP** *group-name*
Specifies the name of the group to add the contact to. The group must be already defined.

**EMAIL** *"email-address"*
Specifies the primary e-mail address for the contact. The double quotation marks are required.

**PAGE** *"email-address"*
Specifies the pager address for the contact. The double quotation marks are required.

**DESCRIPTION** *"description"*
Specifies a brief description for the contact. The double quotation marks are required.

## Example

To create a contact REPLADMIN with an e-mail address repladmin@us.ibm.com:

```
CREATE CONTACT REPLADMIN EMAIL "repladmin@us.ibm.com"
DESCRIPTION "replication administration"
```

# CREATE CONTROL TABLES FOR command (Replication Alert Monitor)

Use the CREATE CONTROL TABLES FOR command to create a new set of Replication Alert Monitor control tables.

## Syntax

```
►►─CREATE CONTROL TABLES FOR───MONITOR CONTROL SERVER─┬──────────────────────────────────┬─►◄
                                                      └─IN──┬─ZOS─┤ zos-ts-clause ├──────┤
                                                            ├─UW─┤ uw-ts-clause ├────────┤
                                                            └─NONIBM──┤ fed-ts-clause ├──┘
```

**zos-ts-clause:**

```
├─┬──────────────────────────────────────────────────────────────────┬──────────────────►
  └─ALERTS DB dbname──┬─tsname──────────────────┬─┤ prof-clause ├─┘
                      └─NAMING PREFIX─prefix─┘

►─┬──────────────────────────────────────────────────────────────────┬──────────────────►
  └─PAGE LOCK DB dbname──┬─tsname──────────────────┬─┤ prof-clause ├─┘
                         └─NAMING PREFIX─prefix─┘
```

```
►─┬──────────────────────────────────────────────────────────────┬─────────────────────────────────────┤
  └ROW LOCK DB dbname──┬tsname──────────────────┬┤ prof-clause ├─┘
                       └NAMING PREFIX──prefix───┘
```

**uw-ts-clause:**

```
├─┬───────────────────────────────────────────────────────┬────────────────────────────────────────────┤
  └OTHERS──┬tsname──────────────────┬┤ prof-clause ├──────┘
           └NAMING PREFIX──prefix───┘
```

**fed-ts-clause:**

```
├─┬───────────────────────────────────────────────────────┬SCHEMA──schemaname─────────────────────────┤
  └OTHERS──┬tsname──────────────────┬┤ prof-clause ├──────┘
           └NAMING PREFIX──prefix───┘
```

**prof-clause:**

```
├─┬─────────────────────────────┬──────────────────────────────────────────────────────────────────────┤
  ├CREATE USING PROFILE──pname──┤
  └REUSE────────────────────────┘
```

## Parameters

**MONITOR CONTROL SERVER**
Specify to create replication control tables for the Monitor control server.

**IN** Specifies the table space. If you do not specify the **IN** clause, the **CREATE CONTROL TABLES** command uses the DB2 defaults for table spaces.

**ZOS**
Specifies z/OS or OS/390.

**UW**
Specifies UNIX or Windows.

**NONIBM**
Specifies non-DB2 data sources.

**ALERTS**
Specifies an existing database on z/OS to create the control tables in. This keyword is valid only when creating monitor control servers.

**PAGE LOCK**
Specifies the table space for replication control tables that require page-level locking. The table must be in an existing database.

**ROW LOCK**
Specifies the table space for replication control tables that require row-level locking. The table must be in an existing database.

**DB** *dbname*
z/OS: Specifies the name of an existing database. You must specify the database name, even if you set the database name in the profile. This command does not create the database.

**OTHERS**
Specifies the table space for all replication control tables except the UOW table.

*tsname*
: Specifies the table space name for the monitor alerts table. The *tsname* input can be a heterogeneous segment or table space name.

**NAMING PREFIX** *prefix*
: Specifies a naming prefix for the control tables.

**SCHEMA** *schemaname*
: Specifies the remote schema name for heterogeneous replication. The default is the remote user ID. For non-DB2 databases, you can specify a table space name or a segment name for those remote sources that support them.

**CREATE USING PROFILE** *pname*
: Specify to create the control tables using the *pname* profile. If you specify the **CREATE USING PROFILE** parameter, the ASNCLP program uses *tsname* as the key (For z/OS, the key is *dbname.tsname*).

**REUSE**
: Specify to reuse the current DDL object. You must issue the **CREATE USING PROFILE** parameter before you can use the **REUSE** parameter. When you specify the **REUSE** parameter, the ASNCLP program checks if the DDL object exists for the *tsname*:

- If the DDL object exists, the ASNCLP program resets the flags and passes the fully populated DDL.
- If the DDL object does not exist, the ASNCLP program displays a syntax error saying that the **CREATE USING PROFILE** parameter is expected.

### Example 1

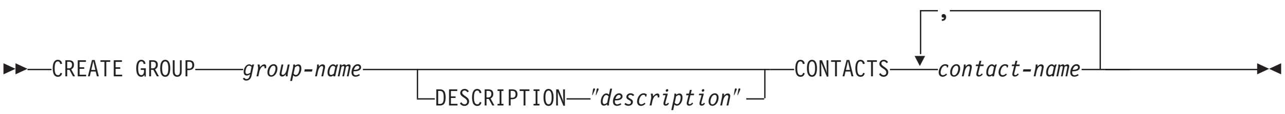
To create the Monitor control tables:

```
CREATE CONTROL TABLES FOR MONITOR CONTROL SERVER
```

## CREATE GROUP command

The CREATE GROUP command creates a group of replication monitor contacts.

### Syntax



### Parameters

*group-name*
: Specifies the name of the group. This name cannot match another group already defined. This parameter is required.

**DESCRIPTION** *"description"*
: Specifies a brief description for the group. The double quotation marks are required.

**CONTACTS** *contact-name*
: Specifies a comma-separated list of contacts that belong to this group.

### Example

To create a group MAINTENANCE that contains contacts REPLADMIN and PERFORMANCE:

```
CREATE GROUP MAINTENANCE CONTACTS REPLADMIN, PERFORMANCE
```

## CREATE MONITOR SUSPENSION command

Use the CREATE MONITOR SUSPENSION command to suspend the monitor program. You can specify a start and end date or use a template that defines a repeating pattern of suspensions.

### Syntax

```
►►──CREATE MONITOR SUSPENSION──name──FOR──┬─SERVER──server_name──┬──STARTING DATE──date──────────────────►
                                          └─ALIAS──server_alias──┘

►──┬────────────────────────────────┬──ENDING DATE──date──┬──────────────────────────────┬──────────────►◄
   ├─USING TEMPLATE──template_name──┤                     └─ENDING TIME──ending_time─────┘
   └─STARTING TIME──starting_time───┘
```

### Parameters

**SERVER**
Specifies the name of the DB2 database where you want to suspend the monitor program.

**z/OS:** This value represents the DB2 subsystem location name.

**ALIAS**
**Linux, UNIX, Windows:** The DB2 alias for the database where you want to suspend the monitor program.

**STARTING DATE**
Specifies one of two different values, depending on whether you use a template for the suspension:

**With template**
Specifies the date that you want to start using the monitor suspension template.

**Without template**
Specifies the date on which the monitor program will be suspended. Use YYYY-MM-DD format.

**USING TEMPLATE**
Specifies that you want to use a template to set the start time and other characteristics of the suspension. You define the template by using the CREATE MONITOR SUSPENSION TEMPLATE command.

**STARTING TIME**
Specifies the time when the monitor suspension begins. Use HH:MM:SS format. The default is 00:00:00.

**ENDING DATE**
Specifies one of two different values, depending on whether you use a template for the suspension:

**With template**

Specifies the date that you want to stop using the monitor suspension template.

**Without template**

Specifies the date when the monitor suspension ends. Use YYYY-MM-DD format.

**ENDING TIME**

Specifies one of two different values, depending on whether you use a template for the suspension:

**With template**

Specifies the time that you want to stop using the monitor suspension template.

**Without template**

Specifies the time when the monitor suspension ends.

Use HH:MM:SS format for the ending time. The default is 00:00:00.

### Example 1

To create a suspension S1 on the monitor control server QSRVR1 that uses the template SUNDAY:

```
CREATE MONITOR SUSPENSION NAME S1 FOR SERVER QSRVR1 STARTING DATE 2006-12-10
USING TEMPLATE SUNDAY ENDING DATE 2007-12-31
```

### Example 2

To create a suspension S2 on the monitor control server QSRVR2 that does not use a template but suspends the monitor during the month of December:

```
CREATE MONITOR SUSPENSION NAME S2 FOR SERVER QSRVR2 STARTING DATE 2006-11-30
STARTING TIME 00:00:00 ENDING DATE 2006-12-31 ENDING TIME 24:00:00
```

## CREATE MONITOR SUSPENSION TEMPLATE command

Use the CREATE MONITOR SUSPENSION TEMPLATE command to define the frequency and duration of periods that the monitor program is suspended.

### Syntax

```
►►──CREATE MONITOR SUSPENSION TEMPLATE──template_name──┬───────────────────────┬──────────►
                                                       └─START TIME──HH:MM:SS──┘

►─REPEATS──┤ occurrence-clause ├───────────────────────────────────────────────────────►◄
```

**occurrence-clause:**

```
                                 ┌─MINUTES─┐
├──┬─DAILY──FOR DURATION──n──────┴─HOURS───┴─────────────────────────────────┬──────────────────────────────┤
   │                                                          ┌─MINUTES─┐    │
   └─WEEKLY──DAY OF WEEK──┬─SUNDAY────┬──FOR DURATION──n──────┼─HOURS───┼────┘
                          ├─MONDAY────┤                       └─DAYS────┘
                          ├─TUESDAY───┤
                          ├─WEDNESDAY─┤
                          ├─THURSDAY──┤
                          ├─FRIDAY────┤
                          └─SATURDAY──┘
```

## Parameters

**START TIME**
Specifies the time at which the monitor program will be suspended, in HH:MM:SS (hours:minutes:seconds) format. The default value is 00:00:00.

**REPEATS**
Specifies which days the monitor program will be suspended, and for how long.

## Example 1

To create a template that suspends the monitor program from 00:00:00 to 04:00:00 every Sunday:

```
CREATE MONITOR SUSPENSION TEMPLATE SUNDAY START TIME 00:00:00 REPEATS WEEKLY
DAY OF WEEK SUNDAY FOR DURATION 4 HOURS
```

## Example 2

To create a template that suspends the monitor program during the lunch hour every day:

```
CREATE MONITOR SUSPENSION TEMPLATE LUNCH START TIME 12:00:00 REPEATS DAILY
FOR DURATION 1 HOUR
```

# DELEGATE CONTACT command

Use the DELEGATE CONTACT command to delegate an existing contact to a new contact for a specific period of time.

## Syntax

```
►►──DELEGATE CONTACT────contact-name1──── TO──contact-name2──FROM──"start-date" ──TO──"end-date" ──────────►◄
```

## Parameters

**CONTACT** *contact-name1*
Specifies the name of the contact to be delegated. The contact must exist.

**TO** *contact-name2*
Specifies the new contact for all alert conditions (if any) that refer to the contact being delegated. The contact must exist.

**FROM** *"start-date"*
Specifies the date when the delegation starts. The date is sensitive to the DB2 locale. The double quotation marks are required.

**TO** *"end-date"*
Specifies the date when the delegation ends. The date is sensitive to the DB2 locale. The double quotation marks are required.

### Example

To delegate alerts from one (REPLADMIN) contact to another (PERFORMACE) for a given period of time:

```
DELEGATE CONTACT REPLADMIN TO PERFORMACE FROM "2007-11-22" TO "2007-12-06"
```

## DROP ALERT CONDITIONS FOR APPLY command

Use the DROP ALERT CONDITIONS FOR APPLY command to drop alert conditions for the Apply program.

### Syntax

```
►►──DROP ALERT CONDITIONS FOR APPLY QUALIFIER──apply-qual──MONTITOR QUALIFIER──mon-qual──────────────►◄
```

### Parameters

**APPLY QUALIFIER** *qual-name*
Specifies the Apply qualifier.

**MONITOR QUALIFIER** *mon-qual*
Specifies the Monitor qualifier.

### Example

To drop alert conditions for the Apply program:

```
DROP ALERT CONDITIONS FOR APPLY QUALIFIER MYAPPLY01 MONITOR QUALIFIER MONQUAL
```

## DROP ALERT CONDITIONS FOR CAPTURE command

Use the DROP ALERT CONDITIONS FOR CAPTURE command to drop alert conditions for the Capture program.

### Syntax

```
►►──DROP ALERT CONDITIONS FOR CAPTURE──SCHEMA──cap-schema──MONTITOR QUALIFIER──mon-qual──────────────►◄
```

### Parameters

**SCHEMA** *cap-schema*
Specifies the Capture schema for the server that you are monitoring.

**MONITOR QUALIFIER** *mon-qual*
Specifies the Monitor qualifier.

### Example

To drop alert conditions for the Capture program:

```
DROP ALERT CONDITIONS FOR CAPTURE SCHEMA ASN1 MONITOR QUALIFIER MONQUAL
```

## DROP ALERT CONDITIONS FOR QAPPLY command

Use the DROP ALERT CONDITIONS FOR QAPPLY command to drop alert conditions for the Q Apply program.

### Syntax

```
►►──DROP ALERT CONDITIONS FOR QAPPLY SCHEMA──schema──MONITOR QUALIFIER──monitor-qualifier──────────────►◄
```

### Parameters

**SCHEMA** *schema*
Specifies the Q Apply schema that qualifies the process to be monitored.

**MONITOR QUALIFIER** *monitor-qualifier*
Specifies the monitor qualifier grouping the alert conditions.

### Example

To drop alert conditions for the Q Apply program:

```
DROP ALERT CONDITIONS FOR QAPPLY SCHEMA ASN1 MONITOR QUALIFIER MONQUAL
```

## DROP ALERT CONDITIONS FOR QCAPTURE command

Use the DROP ALERT CONDITIONS FOR QCAPTURE command to drop alert conditions for the Q Capture program.

### Syntax

```
►►──DROP ALERT CONDITIONS FOR QCAPTURE SCHEMA──schema──MONITOR QUALIFIER──monitor-qualifier────────────►◄
```

### Parameters

**SCHEMA** *schema*
Specifies the Q Capture schema that qualifies the process to be monitored.

**MONITOR QUALIFIER** *monitor-qualifier*
Specifies the monitor qualifier that groups the alert conditions.

### Example

To drop alert conditions for the Q Capture program:

```
DROP ALERT CONDITIONS FOR QCAPTURE SCHEMA ASN1 MONITOR QUALIFIER MONQUAL
```

## DROP CONTACT command

Use the DROP CONTACT command to drop an existing contact.

### Syntax

```
►►──DROP CONTACT──contact-name1──┬────────────────────────────────┬──────────►◄
                                 └─SUBSTITUTE WITH──contact-name2─┘
```

### Parameters

**CONTACT** *contact-name1*
  Specifies the name of the contact. The contact must exist.

**SUBSTITUTE WITH** *contact-name2*
  Specifies the name of a contact. The contact must exist. If the contact being deleted is referenced by any alert conditions, then the alert conditions will now reference the contact represented in this clause.

### Usage notes

If you drop a contact that is the only one referred by an alert condition, this command returns an error. In this case, you must either delete the alert condition before you drop the contact, or use the SUBSTITUTE WITH clause.

### Example

To drop a contact REPLADMIN:

```
DROP CONTACT REPLADMIN
```

## DROP GROUP command

Use the DROP GROUP command to drop a group of replication monitor contacts.

### Syntax

```
►►──DROP GROUP──group-name──────────────────────────────────────────────────►◄
```

### Parameters

*group-name*
  Specifies the name of the group. The group must exist.

### Usage notes

If you drop a group that is the only one referred to by an alert condition, and there are no individual contacts referred to by the alert condition, this command returns an error.

### Example

To drop a group MAINTENANCE:

```
DROP GROUP MAINTENANCE
```

## DROP MONITOR SUSPENSION command

Use the DROP MONITOR SUSPENSION command to delete a suspension from the monitor control tables.

### Syntax

```
►►──DROP MONITOR SUSPENSION──name──────────────────────────────────────────────────►◄
```

### Parameters

*name*
Specifies the template that you want to delete.

### Usage notes

After you remove the suspension, reinitialize the monitor or stop and start the monitor to prompt it to read its control tables and end the suspension.

### Example

To delete the suspension S1:

```
DROP MONITOR SUSPENSION NAME S1
```

## DROP MONITOR SUSPENSION TEMPLATE command

Use the DROP MONITOR SUSPENSION TEMPLATE command to delete a template from the monitor control tables.

### Syntax

```
►►──DROP MONITOR SUSPENSION TEMPLATE──template_name──────────────────────────────────►◄
```

### Parameters

*template_name*
Specifies the name of an existing template.

### Example

To drop the template named that is named sunday:

```
DROP MONITOR SUSPENSION TEMPLATE sunday
```

## LIST MONITOR SUSPENSION command

Use the LIST MONITOR SUSPENSION command to generate a list of suspensions that are defined on a monitor control server. The command sends a report that shows the suspension name and other properties to the standard output (stdout).

### Syntax

```
►►──LIST MONITOR SUSPENSION──────────────────────────────────────────────────────────►◄
```

## LIST MONITOR SUSPENSION TEMPLATE command

Use the LIST MONITOR SUSPENSION TEMPLATE command to generate a list of suspension templates on a monitor control server. The command sends a report that shows the template name and other properties to the standard output (stdout).

### Syntax

```
►►──LIST MONITOR SUSPENSION TEMPLATE──────────────────────────────────────────────────────►◄
```

### Example

The following example shows the output of the LIST MONITOR SUSPENSION TEMPLATE command:

```
TEMPLATE_NAME         START_TIME  FREQUENCY          DURATION  UNITS
------------------    ----------  ---------------    --------  -------
daytemp1              12:00:00    DAILY              4         HOURS
wednesdaytemp2        00:00:00    WEDNESDAY          2         DAYS
minutestemp3          17:30:00    SUNDAY             30        MINUTES
```

## SET OUTPUT command (monitor)

Use the SET OUTPUT command to define output files for the ASNCLP program. The output files contain the SQL statements needed to set up Q replication and event publishing.

### Syntax

```
►►──SET OUTPUT──┬────────────────────────────┬─────────────────────────────────────────►◄
                └─MONITOR SCRIPT──"monfname"─┘
```

### Parameters

**MONITOR SCRIPT** *"monfname"*
Specifies the output file name for scripts that run at the Monitor control server. The default file name is replmonitor.sql.

### Usage notes

- If a script already exists, the new script appends to the current script.
- The double quotation marks in the command syntax are required.

### Example 1

To name the monitor script output file "monitor.sql":

```
SET OUTPUT MONITOR SCRIPT "monitor.sql"
```

## SET SERVER command

Use the SET SERVER command to specify the server (database) used in the ASNCLP session. You can specify authentication information and other required parameters for connecting to the server.

You should always set the Monitor control server before running the monitor administration commands.

## Syntax

```
►►──SET SERVER── MONITOR──TO──┬─NULLS───────────────────────────────────────────────────────────┬──────►◄
                              └─┬─DB──dbalias─────────┬──┬──────────────────┬─┤ other-options ├─┘
                                └─DBALIAS──aliasname──┘  └─DBNAME──dbname───┘
```

### other-options:

```
├──┬──────────────────────────────┬──────────────────────────────────────────────────────────────────┤
   └─ID──userid──PASSWORD──pwd────┘
```

## Parameters

**MONITOR**
Specify to set the database as a monitor server.

**NULLS**
Specify to set the server name to NULLS. This option resets a previously set server name.

**DB** *dbalias*
Specifies the database alias name.

**DBALIAS** *aliasname*
Linux, UNIX, or Windows: Specifies the database alias name.

**DBNAME** *dbname*
z/OS: Specifies the database name.

**ID** *userid*
Specifies the user ID to use to connect to the database.

**PASSWORD** *pwd*
Specifies the password to use to connect to the database.

## Example

To set the monitor server to the SAMPLE database:

```
SET SERVER MONITOR TO DB SAMPLE
```

# SUBSTITUTE CONTACT command

Use the SUBSTITUTE CONTACT command to substitute one existing contact with another existing contact.

## Syntax

```
►►──SUBSTITUTE CONTACT────contact-name1──── WITH──contact-name2──────────────────────────────────►◄
```

## Parameters

*contact-name1*
Specifies the name of the contact to be substituted. The contact must exist.

**WITH** *contact-name2*
Specifies the new contact for all alert conditions (if any) that refer to the contact being substituted. The contact must exist.

## Example

To substitute one contact (REPLADMIN) for another (PERFORMACE):

```
SUBSTITUTE CONTACT REPLADMIN WITH PERFORMACE
```

# Accessing information about IBM

IBM® has several methods for you to learn about products and services.

You can find the latest information on the Web at www.ibm.com/software/data/integration/db2ii/support.htmlINSERT HAWK LINK HERE:

- Product documentation in PDF and online information centers
- Product downloads and fix packs
- Release notes and other support documentation
- Web resources, such as white papers and IBM Redbooks™
- Newsgroups and user groups
- Book orders

To access product documentation, go to this site:

publib.boulder.ibm.com/infocenter/db2help/topic/

INSERT HAWK LINK HERE

You can order IBM publications online or through your local IBM representative.

- To order publications online, go to the IBM Publications Center at www.ibm.com/shop/publications/order.
- To order publications by telephone in the United States, call 1-800-879-2755.

To find your local IBM representative, go to the IBM Directory of Worldwide Contacts at www.ibm.com/planetwide.

# Contacting IBM

You can contact IBM by telephone for customer support, software services, and general information.

## Customer support

To contact IBM customer service in the United States or Canada, call 1-800-IBM-SERV (1-800-426-7378).

## Software services

To learn about available service options, call one of the following numbers:

- In the United States: 1-888-426-4343
- In Canada: 1-800-465-9600

## General information

To find general information in the United States, call 1-800-IBM-CALL (1-800-426-2255).

Go to www.ibm.com for a list of numbers outside of the United States.

## Accessible documentation

Documentation is provided in XHTML format, which is viewable in most Web browsers.

XHTML allows you to view documentation according to the display preferences that you set in your browser. It also allows you to use screen readers and other assistive technologies.

Syntax diagrams are provided in dotted decimal format. This format is available only if you are accessing the online documentation using a screen reader.

## Providing comments on the documentation

Please send any comments that you have about this information or other documentation.

Your feedback helps IBM to provide quality information. You can use any of the following methods to provide comments:

- Send your comments using the online readers' comment form at www.ibm.com/software/awdtools/rcf/.
- Send your comments by e-mail to comments@us.ibm.com. Include the name of the product, the version number of the product, and the name and part number of the information (if applicable). If you are commenting on specific text, please include the location of the text (for example, a title, a table number, or a page number).

# Notices and trademarks

## Notices

This information was developed for products and services offered in the U.S.A.

IBM may not offer the products, services, or features discussed in this document in other countries. Consult your local IBM representative for information on the products and services currently available in your area. Any reference to an IBM product, program, or service is not intended to state or imply that only that IBM product, program, or service may be used. Any functionally equivalent product, program, or service that does not infringe any IBM intellectual property right may be used instead. However, it is the user's responsibility to evaluate and verify the operation of any non-IBM product, program, or service.

IBM may have patents or pending patent applications covering subject matter described in this document. The furnishing of this document does not grant you any license to these patents. You can send license inquiries, in writing, to:

IBM Director of Licensing
IBM Corporation
North Castle Drive
Armonk, NY 10504-1785 U.S.A.

For license inquiries regarding double-byte (DBCS) information, contact the IBM Intellectual Property Department in your country or send inquiries, in writing, to:

IBM World Trade Asia Corporation
Licensing 2-31 Roppongi 3-chome, Minato-ku
Tokyo 106-0032, Japan

The following paragraph does not apply to the United Kingdom or any other country where such provisions are inconsistent with local law: INTERNATIONAL BUSINESS MACHINES CORPORATION PROVIDES THIS PUBLICATION ″AS IS″ WITHOUT WARRANTY OF ANY KIND, EITHER EXPRESS OR IMPLIED, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF NON-INFRINGEMENT, MERCHANTABILITY OR FITNESS FOR A PARTICULAR PURPOSE. Some states do not allow disclaimer of express or implied warranties in certain transactions, therefore, this statement may not apply to you.

This information could include technical inaccuracies or typographical errors. Changes are periodically made to the information herein; these changes will be incorporated in new editions of the publication. IBM may make improvements and/or changes in the product(s) and/or the program(s) described in this publication at any time without notice.

Any references in this information to non-IBM Web sites are provided for convenience only and do not in any manner serve as an endorsement of those Web sites. The materials at those Web sites are not part of the materials for this IBM product and use of those Web sites is at your own risk.

IBM may use or distribute any of the information you supply in any way it believes appropriate without incurring any obligation to you.

Licensees of this program who wish to have information about it for the purpose of enabling: (i) the exchange of information between independently created programs and other programs (including this one) and (ii) the mutual use of the information which has been exchanged, should contact:

IBM Corporation
J46A/G4
555 Bailey Avenue
San Jose, CA 95141-1003 U.S.A.

Such information may be available, subject to appropriate terms and conditions, including in some cases, payment of a fee.

The licensed program described in this document and all licensed material available for it are provided by IBM under terms of the IBM Customer Agreement, IBM International Program License Agreement or any equivalent agreement between us.

Any performance data contained herein was determined in a controlled environment. Therefore, the results obtained in other operating environments may vary significantly. Some measurements may have been made on development-level systems and there is no guarantee that these measurements will be the same on generally available systems. Furthermore, some measurements may have been estimated through extrapolation. Actual results may vary. Users of this document should verify the applicable data for their specific environment.

Information concerning non-IBM products was obtained from the suppliers of those products, their published announcements or other publicly available sources. IBM has not tested those products and cannot confirm the accuracy of performance, compatibility or any other claims related to non-IBM products. Questions on the capabilities of non-IBM products should be addressed to the suppliers of those products.

All statements regarding IBM's future direction or intent are subject to change or withdrawal without notice, and represent goals and objectives only.

This information is for planning purposes only. The information herein is subject to change before the products described become available.

This information contains examples of data and reports used in daily business operations. To illustrate them as completely as possible, the examples include the names of individuals, companies, brands, and products. All of these names are fictitious and any similarity to the names and addresses used by an actual business enterprise is entirely coincidental.

COPYRIGHT LICENSE:

This information contains sample application programs in source language, which illustrate programming techniques on various operating platforms. You may copy, modify, and distribute these sample programs in any form without payment to IBM, for the purposes of developing, using, marketing or distributing application programs conforming to the application programming interface for the operating platform for which the sample programs are written. These examples have not been thoroughly tested under all conditions. IBM, therefore, cannot guarantee or imply reliability, serviceability, or function of these programs.

Each copy or any portion of these sample programs or any derivative work, must include a copyright notice as follows:

© (your company name) (year). Portions of this code are derived from IBM Corp. Sample Programs. © Copyright IBM Corp. _enter the year or years_. All rights reserved.

If you are viewing this information softcopy, the photographs and color illustrations may not appear.

## Trademarks

IBM trademarks and certain non-IBM trademarks are marked at their first occurrence in this document.

See http://www.ibm.com/legal/copytrade.shtml for information about IBM trademarks.

The following terms are trademarks or registered trademarks of other companies:

Java™ and all Java-based trademarks and logos are trademarks or registered trademarks of Sun Microsystems, Inc. in the United States, other countries, or both.

Microsoft®, Windows, Windows NT®, and the Windows logo are trademarks of Microsoft Corporation in the United States, other countries, or both.

Intel®, Intel Inside® (logos), MMX and Pentium® are trademarks of Intel Corporation in the United States, other countries, or both.

UNIX is a registered trademark of The Open Group in the United States and other countries.

Linux is a trademark of Linus Torvalds in the United States, other countries, or both.

Other company, product or service names might be trademarks or service marks of others.

# Index

Printed in USA

SC19-1018-00