

IBM InfoSphere Data Replication  
Version 10.1.3

*ASNCLP Program Reference for  
Replication and Event Publishing*





IBM InfoSphere Data Replication  
Version 10.1.3

*ASNCLP Program Reference for  
Replication and Event Publishing*



**Note**

Before using this information and the product that it supports, read the information in “Notices and trademarks” on page 331.

---

# Contents

## Chapter 1. Getting started with the ASNCLP program . . . . . 1

Before you run the ASNCLP program . . . . .	1
Supported operating systems . . . . .	2
Setting up a Java environment to run the ASNCLP program . . . . .	2
Binding z/OS packages for the ASNCLP program	3
ASNCLP configuration file . . . . .	3
Use of double quotation marks in ASNCLP commands . . . . .	5
Running the ASNCLP commands in batch mode . . . . .	5
Running the ASNCLP commands in interactive mode	7
Running the ASNCLP commands in execute-immediately mode . . . . .	7

## Chapter 2. ASNCLP commands for SQL Replication . . . . . 9

Sample ASNCLP script for setting up SQL Replication . . . . .	10
Sample ASNCLP script for setting up SQL Replication from a view . . . . .	11
ALTER DATASTAGE DEFINITION FOR command	13
ALTER MEMBER ADD COLS command . . . . .	14
ALTER REGISTRATION command . . . . .	15
ALTER SUBSCRIPTION SET command . . . . .	18
ASNCLP SESSION SET TO command (SQL Replication) . . . . .	19
CREATE CONTROL TABLES FOR command (SQL Replication) . . . . .	20
CREATE DATASTAGE DEFINITION FOR command	22
CREATE MEMBER command . . . . .	23
CREATE REGISTRATION command . . . . .	33
CREATE STMT command . . . . .	38
CREATE SUBSCRIPTION SET command . . . . .	40
DROP CONTROL TABLES ON command . . . . .	42
DROP DATASTAGE DEFINITION FOR . . . . .	43
DROP MEMBER command . . . . .	43
DROP REGISTRATION command . . . . .	44
DROP STMT command . . . . .	45
DROP SUBSCRIPTION SET command . . . . .	46
OFFLINE LOAD command . . . . .	46
PROMOTE REGISTRATION command . . . . .	47
PROMOTE SUBSCRIPTION SET command . . . . .	49
SET CAPTURE SCHEMA command (SQL Replication) . . . . .	51
SET DROP command (SQL Replication) . . . . .	52
SET LOG command . . . . .	53
SET OUTPUT command (SQL Replication) . . . . .	53
SET PROFILE command (SQL Replication) . . . . .	54
SET RUN SCRIPT command (SQL Replication) . . . . .	58
SET SERVER command (SQL Replication) . . . . .	61
SET TRACE command . . . . .	64

## Chapter 3. Sample ASNCLP scripts for Q Replication . . . . . 67

Sample ASNCLP scripts for setting up unidirectional Q Replication . . . . .	67
Sample ASNCLP scripts for setting up unidirectional Q Replication from a Classic data source . . . . .	68
Sample ASNCLP scripts for setting up bidirectional Q Replication . . . . .	71
Sample ASNCLP scripts for setting up peer-to-peer Q Replication (two servers) . . . . .	72
Sample ASNCLP scripts for setting up peer-to-peer Q Replication (three servers) . . . . .	74
Sample ASNCLP script for promoting unidirectional configurations . . . . .	76
Sample ASNCLP scripts for promoting peer-to-peer configurations . . . . .	77

## Chapter 4. ASNCLP commands for unidirectional Q Replication . . . . . 79

ALTER ADD COLUMN command (unidirectional replication) . . . . .	80
ALTER CAPTURE PARAMETERS command (Classic replication) . . . . .	82
ALTER CONFIGURATION APPLY command . . . . .	83
ALTER QSUB command (unidirectional replication)	84
ALTER REPLQMAP command . . . . .	89
ASNCLP SESSION SET TO command . . . . .	91
CREATE CONTROL TABLES FOR command . . . . .	92
CREATE MQ SCRIPT command . . . . .	98
CREATE QSUB command (unidirectional replication) . . . . .	101
CREATE REPLQMAP command . . . . .	122
CREATE SCHEMASUB command . . . . .	124
CREATE SUBSCRIPTION OPTIONS command . . . . .	127
DROP CONTROL TABLES ON command . . . . .	129
DROP QSUB command . . . . .	129
DROP REPLQMAP command . . . . .	132
DROP SCHEMASUB command . . . . .	133
DROP SUBSCRIPTION OPTIONS command . . . . .	133
LIST QSUB command (Q Replication) . . . . .	134
LIST REPLQMAP command (Q Replication) . . . . .	135
LIST APPLY SCHEMA command . . . . .	137
LIST CAPTURE SCHEMA command . . . . .	138
LIST SCHEMASUB command . . . . .	139
LOAD DONE command . . . . .	139
PROMOTE QSUB command (unidirectional replication) . . . . .	141
PROMOTE REPLQMAP command . . . . .	142
REINIT SCHEMASUB command . . . . .	143
SET APPLY SCHEMA command . . . . .	144
SET CAPTURE SCHEMA command . . . . .	145
SET DROP command (unidirectional replication)	145
SET LOG command . . . . .	147
SET OUTPUT command . . . . .	147

SET PROFILE command . . . . .	148
SET QMANAGER command . . . . .	151
SET RUN SCRIPT command . . . . .	152
SET SERVER command . . . . .	155
SET TRACE command . . . . .	159
SHOW SET ENV command . . . . .	159
START QSUB command . . . . .	159
START SCHEMASUB command . . . . .	162
STOP QSUB command . . . . .	163
STOP SCHEMASUB command . . . . .	165
VALIDATE WSMQ ENVIRONMENT FOR command . . . . .	165
VALIDATE WSMQ MESSAGE FLOW FOR REPLQMAP command . . . . .	166

## Chapter 5. ASNCLP commands for multidirectional Q Replication . . . . 169

Deprecated commands . . . . .	171
DROP SUBTYPE command (bidirectional replication) . . . . .	171
DROP SUBTYPE command (peer-to-peer replication) . . . . .	172
LOAD MULTIDIR REPL SCRIPT command (multidirectional Q Replication) . . . . .	172
SET MULTIDIR SCHEMA command (multidirectional Q Replication) . . . . .	174
SET SERVER command (multidirectional Q Replication) . . . . .	174
SET TABLES command (multidirectional Q Replication) . . . . .	175
ALTER ADD COLUMN command (multidirectional replication) . . . . .	178
ALTER QSUB command (bidirectional replication)	179
ALTER QSUB command (peer-to-peer replication)	182
ALTER REPLQMAP command . . . . .	184
ASNCLP SESSION SET TO command . . . . .	186
CREATE MQ SCRIPT command . . . . .	187
CREATE QSUB command (bidirectional replication) . . . . .	190
CREATE QSUB command (peer-to-peer replication)	197
CREATE REPLQMAP command . . . . .	204
CREATE SCHEMASUB command . . . . .	206
CREATE SUBSCRIPTION OPTIONS command . . . . .	208
DROP CONTROL TABLES ON command . . . . .	210
DROP REPLQMAP command . . . . .	211
DROP QSUB command . . . . .	212
DROP SCHEMASUB command . . . . .	214
DROP SUBGROUP command (multidirectional Q Replication) . . . . .	215
DROP SUBSCRIPTION OPTIONS command . . . . .	216
LIST APPLY SCHEMA command . . . . .	216
LIST CAPTURE SCHEMA command . . . . .	217
LIST SCHEMASUB command . . . . .	218
LOAD DONE command . . . . .	219
PROMOTE QSUB command (multidirectional replication) . . . . .	220
PROMOTE REPLQMAP command . . . . .	221
REINIT SCHEMASUB command . . . . .	223
SET APPLY SCHEMA command . . . . .	224
SET BIDI NODE command . . . . .	224
SET CAPTURE SCHEMA command . . . . .	226

SET CONNECTION command (multidirectional Q Replication) . . . . .	227
SET ENFORCE MATCHING CONSTRAINTS command (multidirectional Q Replication). . . . .	228
SET LOG command . . . . .	229
SET OUTPUT command (multidirectional Q Replication) . . . . .	229
SET PEER NODE command . . . . .	230
SET PROFILE command . . . . .	232
SET QMANAGER command . . . . .	236
SET REFERENCE TABLE command (multidirectional Q Replication) . . . . .	236
SET RUN SCRIPT command . . . . .	237
SET SUBGROUP command (multidirectional Q Replication) . . . . .	240
SET TRACE command . . . . .	241
SHOW SET ENV command . . . . .	241
START QSUB command . . . . .	242
START SCHEMASUB command . . . . .	244
STOP QSUB command . . . . .	245
STOP SCHEMASUB command . . . . .	247
VALIDATE WSMQ ENVIRONMENT FOR command . . . . .	248
VALIDATE WSMQ MESSAGE FLOW FOR REPLQMAP command . . . . .	249

## Chapter 6. ASNCLP commands for Event Publishing . . . . . 251

Sample ASNCLP scripts for setting up Event Publishing . . . . .	252
ALTER ADD COLUMN command (Event Publishing) . . . . .	253
ALTER PUBQMAP command . . . . .	254
ALTER PUB command . . . . .	255
CREATE CONTROL TABLES FOR command . . . . .	258
CREATE MQ SCRIPT command (Event Publishing)	264
CREATE PUBQMAP command . . . . .	266
CREATE PUB command . . . . .	268
DROP CONTROL TABLES ON command . . . . .	272
DROP PUBQMAP command . . . . .	273
DROP PUB command . . . . .	274
LIST PUBS command . . . . .	274
LIST PUBQMAPS command . . . . .	275
LIST CAPTURE SCHEMA command . . . . .	276
PROMOTE PUB command . . . . .	277
PROMOTE PUBQMAP command . . . . .	279
SET CAPTURE SCHEMA command . . . . .	280
SET LOG command . . . . .	281
SET OUTPUT command . . . . .	281
SET QMANAGER command . . . . .	282
SET RUN SCRIPT command . . . . .	283
SET SERVER command (Event Publishing) . . . . .	285
SET TRACE command . . . . .	288
SHOW SET ENV command . . . . .	288
START PUB command . . . . .	288
STOP PUB command . . . . .	289
VALIDATE WSMQ ENVIRONMENT FOR command . . . . .	289

**Chapter 7. ASNCLP commands for the Replication Alert Monitor . . . . . 291**

Sample ASNCLP script for setting up the Replication Alert Monitor . . . . . 292

ALTER ALERT CONDITIONS FOR APPLY command . . . . . 293

ALTER ALERT CONDITIONS FOR CAPTURE command . . . . . 296

ALTER ALERT CONDITIONS FOR QAPPLY command . . . . . 298

ALTER ALERT CONDITIONS FOR QCAPTURE command . . . . . 300

ALTER CONTACT command . . . . . 302

ALTER GROUP command . . . . . 303

ALTER MONITOR SUSPENSION command . . . . . 304

ALTER MONITOR SUSPENSION TEMPLATE command . . . . . 305

CREATE ALERT CONDITIONS FOR APPLY command . . . . . 306

CREATE ALERT CONDITIONS FOR CAPTURE command . . . . . 308

CREATE ALERT CONDITIONS FOR QAPPLY command . . . . . 310

CREATE ALERT CONDITIONS FOR QCAPTURE command . . . . . 312

CREATE CONTACT command . . . . . 313

CREATE CONTROL TABLES FOR command. . . . . 314

CREATE GROUP command . . . . . 316

CREATE MONITOR SUSPENSION command . . . . . 317

CREATE MONITOR SUSPENSION TEMPLATE command . . . . . 318

DELEGATE CONTACT command . . . . . 319

DROP ALERT CONDITIONS FOR APPLY command . . . . . 320

DROP ALERT CONDITIONS FOR CAPTURE command . . . . . 320

DROP ALERT CONDITIONS FOR QAPPLY command . . . . . 321

DROP ALERT CONDITIONS FOR QCAPTURE command . . . . . 321

DROP CONTACT command . . . . . 321

DROP GROUP command . . . . . 322

DROP MONITOR SUSPENSION command . . . . . 322

DROP MONITOR SUSPENSION TEMPLATE command . . . . . 323

LIST MONITOR SUSPENSION command . . . . . 323

LIST MONITOR SUSPENSION TEMPLATE command . . . . . 323

SET OUTPUT command. . . . . 324

SET SERVER command . . . . . 324

SUBSTITUTE CONTACT command . . . . . 326

**Contacting IBM . . . . . 327**

**How to read syntax diagrams . . . . . 329**

**Notices and trademarks . . . . . 331**

**Index . . . . . 335**





---

## Chapter 1. Getting started with the ASNCLP program

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

The ASNCLP program generates SQL scripts that insert or modify information into the control tables about replication sources, targets, queues, and other options. You can use multiple commands together to generate the SQL for an entire configuration. Three types of commands are available:

### Task commands

These commands create, modify, list, or remove replication objects such as control tables and queue maps. They also start objects such as Q subscriptions and publications.

### Environment commands

These commands define the environment for task commands. For example, they define the servers where objects are created, set defaults for task commands, and identify output files for messages that are issued when the ASNCLP processes task commands.

### Validation commands

These commands validate some aspects of the runtime environment for Q Capture and Q Apply. For example, they can validate the attributes of WebSphere MQ objects for replication or publishing.

The ASNCLP program can process these commands in one of three modes:

<b>Batch</b>	Assemble ASNCLP commands in a script file that is processed by a single invocation of the program
<b>Interactive</b>	Run ASNCLP commands one at a time from a command prompt
<b>Execute immediately</b>	Run independent operational commands such as START QSUB

The best way to get started writing ASNCLP scripts is to work from the examples found in these topics:

- Sample ASNCLP scripts for setting up SQL Replication
- Sample ASNCLP scripts for Q Replication
- Sample ASNCLP scripts for setting up Event Publishing
- Sample ASNCLP scripts for setting up the Replication Alert Monitor

---

## Before you run the ASNCLP program

Before you run the ASNCLP program, you might need to take some configuration steps depending on the operating system on which the program runs and the servers to which it connects.

## Supported operating systems

The ASNCLP program runs on Linux, UNIX, Windows, z/OS, and UNIX System Services (USS) on z/OS®. The ASNCLP program does not run natively on System i®.

The ASNCLP commands generate replication definitions for all operating system environments that are supported by the replication products: z/OS, System i (SQL Replication only), Linux, UNIX, and Windows. You must have connectivity to each server for which you are generating replication definitions; that is, you must be able to issue a database connection statement to each of the servers.

**Note:** Additional configuration steps are required to enable the ASNCLP to run natively on z/OS or on USS. For details, see *Optional: Enabling the ASNCLP program to run with JCL* or *Optional: Enabling the ASNCLP program to run on USS* in the *Information Management Software for z/OS Solutions Information Center*.

**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 environment. Your PATH environment variable must contain a path to a Java runtime environment in order to run the ASNCLP program.

Starting with Version 9.7 Fix Pack 2, the ASNCLP program automatically sets the path to a Java runtime environment (JVM) that was installed along with DB2 before it processes commands.

For all DB2 products except the IBM® Data Server Runtime Client, the DB2 Database for Linux, UNIX, and Windows installation process automatically installs the SDK for Java. If you need to install the SDK, go to the "IBM developer kits" page on IBM developerWorks®: <http://www.ibm.com/developerworks/java/jdk/index.html>. The IBM SDK for Java must support the code page under which you plan to run the ASNCLP program. When a code set is not supported by the IBM SDK for an operating system, you must run the ASNCLP on another operating system whose SDK supports the code page. For example, character set 5026(Cp290) is not supported by the IBM JDK for HP-UX and Solaris, so you must run the ASNCLP from Linux, AIX®, or Windows.

Use the following procedure if the PATH environment variable does not contain a path to a Java runtime environment.

### Procedure

Add the following path to your PATH environment variable:

```
INSTDIR\java\jdk
```

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

## Examples

### Linux UNIX

To set the PATH environment variable from a UNIX command prompt:

```
export
PATH=$PATH
:/u/INSTDIR/sql1lib/java/jdk
```

### Windows

To set the PATH environment variable from a Windows command prompt:

```
set PATH=%PATH%;%INSTDIR\sql1lib\java\jdk
```

**Note:** On Windows, if the database manager stores the value of the **JDK\_PATH** database configuration parameter as `c:\program files\ibm\sql1lib\java\jdk`, the space in program files can cause a problem for the ASNCLP program. To avoid this problem, change the value of **JDK\_PATH** to `c:\progra~1\ibm\sql1lib\java\jdk`. For example:

```
db2 update dbm cfg using JDK_PATH c:\progra~1\ibm\sql1lib\java\jdk
```

## Binding z/OS packages for the ASNCLP program

### z/OS

Before you use the ASNCLP program with DB2 for z/OS, you must bind the basic DRDA<sup>®</sup> 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 DB2 subsystem on 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:

```
bind @ddcmvs.lst blocking all sqlerror continue
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 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
```

## ASNCLP configuration file

To access Classic or Oracle sources, the ASNCLP program requires connectivity information to be provided through a configuration file.

You can also use a configuration file when the ASNCLP is running on UNIX System Services for z/OS (USS). When you run ASNCLP on USS, you also have the choice of specifying connection information in a communication database in the same manner that is required when running ASNCLP natively on z/OS. For more details, see *Optional: Enabling the ASNCLP program to run with JCL*.

The ASNCLP configuration file contains a group of lines for each data source that the ASNCLP needs to access. Each grouping has a unique name for the group followed by lines that specify the connection information. The unique name is used in ASNCLP scripts to identify a source.

## Syntax

Specify the server information in the configuration file in the following format:

```
[NAME]
Type=source_type
Data source=data_source_name
Host=host_name
Port=port_number
Codepage=code_page
...
```

## Parameters

[NAME]

Specifies a unique name for a configuration. You provide this name in ASNCLP scripts so that the ASNCLP program can connect to the data source. You can define multiple servers in a single configuration file by indicating the beginning of a new server definition in the enclosing brackets (for example, [NAME2]).

**Important:** The value cannot be longer than eight characters.

*Type*

Specifies the type of server:

### Classic replication

Specify Type=classic.

### ASNCLP on USS

If the server is DB2 for z/OS or DB2 for Linux, UNIX, and Windows and you are running the ASNCLP on USS, specify Type=DB2.

### Oracle sources

Specify Type=oracle.

*Data source*

Specifies the location of the source data:

### Classic replication

Specifies the name of the query processor on the Classic data server.

### ASNCLP on USS

If you are running the ASNCLP on USS, for DB2 sources this parameter specifies the DB2 for z/OS location name or DB2 for Linux, UNIX, and Windows database name.

### Oracle sources

Specifies the name of the Oracle database.

*Host*

Specifies the host name or IP address of the data server where the *data\_source\_name* resides.

*Port*

Port is the port number of the server where the data source resides.

### Codepage

Codepage is an optional parameter for Classic sources that describes the code page of the data.

### Example 1

The following example shows a configuration file that is used on USS to specify a connection to a DB2 for z/OS subsystem:

```
[DB2ZOS]
Type=DB2
Data source=dsn7
Host=stplex4a.svl.ibm.com
Port=2080
```

### Example 2

The following example shows a configuration file with multiple server definitions:

```
[server_1]
Type=classic
Data source=CACSAMP1
Host=123.123.123.1
Port=8096
[server_2]
Type=classic
Data source=CACSALES
Host=145.145.231.87
Port=8095
```

### Usage notes

You can save the configuration file to any location. The default file name is `asnservers.ini`.

You must use the **SET SERVER** command to provide the ASNCLP program with the location of the configuration file. The following example shows that the `asnservers.ini` configuration file is saved in the `/home/db2inst/sqllib/classic_files/` directory.

```
SET SERVER capture TO CONFIG SERVER cacsamp1 FILE
"/home/db2inst/sqllib/classic_files/asnservers.ini" ID my_user_id
PASSWORD "my_password";
```

## Use of double quotation marks in ASNCLP commands

If you want to preserve case or use special characters in names or passwords that are input as values to ASNCLP keywords, you can use double quotation marks ("").

By default, the ASNCLP program changes input values to upper case unless the values are enclosed in double quotation marks. So, for example, if the schema you are using to create control tables is `MySchema`, you should input the value as `"MySchema"` in commands.

For passwords, the ASNCLP program does not support special characters such as `@` or `#` unless the password is enclosed in double quotation marks. A password such as `my@pwd` would cause an error, but `"my@pwd"` is valid.

---

## Running the ASNCLP commands in batch mode

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

An ASNCLP input file is known as a script. An ASNCLP script typically contains a mix of environment and task commands, with the environment commands usually at the start of the script. Each command ends with a semicolon (;). The script can also contain comments. These lines start with a pound sign (#).

The best way to get started writing ASNCLP scripts is to work from the examples found in these topics:

- Sample ASNCLP scripts for setting up SQL Replication
- Sample ASNCLP scripts for Q Replication
- Sample ASNCLP scripts for setting up Event Publishing
- Sample ASNCLP scripts for setting up the Replication Alert Monitor

When the ASNCLP program processes a script, it compiles the ASNCLP commands into SQL statements that are written to a file. These SQL statements create, modify, or remove replication objects such as control tables and subscriptions. The ASNCLP program can run these SQL statements as they are generated or you can choose to have ASNCLP program generate only the SQL files so that you can run the SQL statements later.

If you choose to have SQL statements run as they are generated, the SQL statements for each task command are committed before the next task statement is compiled. You can choose to have ASNCLP program stop processing the ASNCLP script when it detects a potential SQL error or stop processing the SQL script when it receives an actual SQL error. Or you can have the ASNCLP program continue processing a script even if potential or actual SQL errors occur. The latter option allows you to fix errors without having to delete or comment out previously successful task commands. See the SET RUN SCRIPT topics for how to select this option, and How the ASNCLP handles errors while processing scripts for more detail on how these options in the SET RUN SCRIPT commands affect ASNCLP error behavior.

## Procedure

To run the ASNCLP commands in batch mode by 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.

**Tip:** You can specify that the ASNCLP program ignores some errors that it encounters when creating objects that already exist by using the **SET RUN SCRIPT LATER GENERATE SQL FOR EXISTING YES** command.

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

```
quit
```

---

## 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 in execute-immediately mode

The execute-immediately mode is useful when you need to issue a single command. You can use the **START QSUB** and **STOP QSUB** commands, **START PUB** and **STOP PUB**, and **LIST** commands in execute-immediately mode.

### Before you begin

The ASNCLP command that you execute cannot rely on previous commands. The command must be self-contained. For example, many commands rely on the **SET SERVER** command to define where objects are created.

### Restrictions

Execute-immediately mode is not available when the ASNCLP runs natively on z/OS with JCL.

### Procedure

To execute an ASNCLP command in execute-immediately mode:

1. Open an operating system command prompt.
2. Run the ASNCLP command:

```
ASNCLP -exe my_command
```

Replace *my\_command* with the ASNCLP command that you want to immediately execute.

The following command is an example of starting a Q subscription for a Classic replication source:

```
asnclp -exe START QSUB SUBNAME sub1 CAP SERVER OPTIONS CONFIG SERVER classic1  
FILE asnservers.ini ID id1 PASSWORD passwd1
```



---

## 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 script for setting up SQL Replication” on page 10 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*

<b>If you want to ...</b>	<b>Use this command</b>
Add columns to an existing member	“ALTER MEMBER ADD COLS command” on page 14
Change the properties of a registration	“ALTER REGISTRATION command” on page 15
Change the properties of a subscription set	“ALTER SUBSCRIPTION SET command” on page 18
Establish a session for SQL Replication	“ASNCLP SESSION SET TO command (SQL Replication)” on page 19
Create control tables	“CREATE CONTROL TABLES FOR command (SQL Replication)” on page 20
Create a subscription-set member	“CREATE MEMBER command” on page 23
Create a registration	“CREATE REGISTRATION command” on page 33
Create a SQL statement that is processed with an existing subscription set	“CREATE STMT command” on page 38
Create a subscription set	“CREATE SUBSCRIPTION SET command” on page 40
Drop control tables	“DROP CONTROL TABLES ON command” on page 42
Delete a subscription-set member	“DROP MEMBER command” on page 43
Delete a registration	“DROP REGISTRATION command” on page 44
Delete SQL statements for an existing subscription set	“DROP STMT command” on page 45
Delete a subscription set	“DROP SUBSCRIPTION SET command” on page 46
Control a manual full refresh for offline load procedures	“OFFLINE LOAD command” on page 46
Promote a registration	“PROMOTE REGISTRATION command” on page 47
Promote a subscription set	“PROMOTE SUBSCRIPTION SET command” on page 49
Set a source and target Capture schema for all task commands	“SET CAPTURE SCHEMA command (SQL Replication)” on page 51
Specify whether to drop the table space when you drop the replication object that it contains	“SET DROP command (SQL Replication)” on page 52
Set the log file name for the ASNCLP program	“SET LOG command” on page 53
Specify a name for the output files that contain the SQL scripts	“SET OUTPUT command (SQL Replication)” on page 53
Set up customization rules for creating table space objects	“SET PROFILE command (SQL Replication)” on page 54
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 58

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 61
Enable and disable the tracing for the ASNCLP commands	“SET TRACE command” on page 64

## Sample ASNCLP script for setting up SQL Replication

This sample contains an ASNCLP script for setting up a basic SQL Replication environment.

The script uses the EMPLOYEE table in the DB2 for Linux, UNIX, and Windows SAMPLE database. To create the SAMPLE database, use the **db2samp1** command. The script generates SQL statements that create Capture and Apply control tables, a registration for the EMPLOYEE table, a subscription set and a subscription-set member.

You can copy the ASNCLP script to a text file and run it by using the ASNCLP -f *filename* command. First change all occurrences of DB2ADMIN to the schema of the EMPLOYEE table in your SAMPLE database. Within the code sample, details about each group of commands are preceded by a comment character (#).

### ASNCLP script

The script performs the following actions:

- 1** Setting the RUN NOW option
- 2** Setting up the source server
- 3** Registering the EMPLOYEE table
- 4** Setting up the target server
- 5** Creating the subscription set
- 6** Creating a target object profile
- 7** Creating the subscription-set member
- 8** Ending the ASNCLP session

```
# 1 Setting the RUN NOW option
# This option prompts the ASNCLP to generate SQL scripts for creating
# replication objects and then run the scripts before generating the next
# SQL script. This option is required for this sample because, for example,
# the Capture control tables must be created before you can define a registration
# within them.
```

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

```
# 2 Setting up the source server
# Specifies the SAMPLE database as the Capture server and creates the
# Capture control tables.
```

```
SET SERVER CAPTURE TO DB SAMPLE;
CREATE CONTROL TABLES FOR CAPTURE SERVER;
```

```
# 3 Registering the EMPLOYEE table
# This command registers the EMPLOYEE table in the SAMPLE database and specifies
# that a change-data (CD) table, CDEMPLOYEE, be created to "stage" or hold
# replicated rows until the Apply program fetches them. The DIFFERENTIAL
# REFRESH option prompts the Apply program to update the target table periodically
```

```

# as the source table changes.
CREATE REGISTRATION (DB2ADMIN.EMPLOYEE) DIFFERENTIAL REFRESH STAGE CDEMPLOYEE;

# 5 Setting up the target server
# For this script we also use the SAMPLE database as the control server and
# target server.
SET SERVER CONTROL TO DB SAMPLE;
SET SERVER TARGET TO DB SAMPLE;
CREATE CONTROL TABLES FOR APPLY CONTROL SERVER;

# 6 Creating the subscription set
# The TIMING INTERVAL 1 option specifies that the Apply program process
# the set every minute.

CREATE SUBSCRIPTION SET SETNAME SET00 APPLYQUAL AQ00
ACTIVATE YES
TIMING INTERVAL 1 START DATE "2011-04-12" TIME "15:00:00.000000";

# 7 Creating a target object profile
# The profile specifies a container for the target table space that will be
# created for the target table. If you are running the script on Linux or
# UNIX, specify a Linux or UNIX filepath instead of c:\db2data\TSTRG.TS

SET PROFILE TBSPROFILE FOR OBJECT TARGET TABLESPACE OPTIONS UW USING
FILE "c:\db2data\TSTRG.TS" SIZE 700 PAGES;

# 8 Creating the subscription-set member
# The CREATE MEMBER command specifies the registered table EMPLOYEE
# as the replication source and creates a target table, TGEMPLOYEE.
# It also specifies that a new table space, TSTRG00, be created. TGEMPLOYEE
# is specified as a user copy table with all columns registered.

CREATE MEMBER IN SETNAME SET00 APPLYQUAL AQ00 ACTIVATE YES SOURCE
DB2ADMIN.EMPLOYEE
TARGET NAME DB2ADMIN.TGEMPLOYEE DEFINITION IN TSTRG00 CREATE USING
PROFILE TBSPROFILE
TYPE USERCOPY COLS ALL REGISTERED;

# 9 Ending the ASNCLP session

QUIT;

```

---

## Sample ASNCLP script for setting up SQL Replication from a view

This sample contains an ASNCLP script for setting up SQL Replication from a view over a source table. It also includes SQL statements for creating the sample view.

The scripts use the EMPLOYEE and DEPARTMENT tables in the DB2 for Linux, UNIX, and Windows SAMPLE database. To create the SAMPLE database, use the **db2samp1** command.

### Script to create sample view

The sample view performs two transformations on data in the EMPLOYEE table:

- Takes values from the FIRSTNAME and LASTNAME columns and concatenates them into a new FULLNAME column.
- Uses a CASE expression to determine whether the employee is listed as "ELIGIBLE" or "INELIGIBLE" in a TUITION\_ASSISTANCE column.

The view script also obtains the name of the employee's department by performing a join of the EMPLOYEE and DEPARTMENT tables.

```

CREATE VIEW EMPLOYEE_TRANSFORM AS
SELECT
AA.EMPNO,
CONCAT(AA.LASTNAME,CONCAT(' ',SUBSTR(AA.FIRSTNAME,1,1))) AS FULLNAME,
CASE
WHEN AA.EDLEVEL > 12 THEN 'ELIGIBLE'
ELSE 'INELIGIBLE'
END AS TUITION_ASSISTANCE,
AA.WORKDEPT,

BB.DEPTNAME
FROM DB2ADMIN.EMPLOYEE AA, DB2ADMIN.DEPARTMENT BB
WHERE BB.DEPTNO=AA.WORKDEPT;

```

Copy the SQL script into a file named `view.sql`. Change all occurrences of `DB2ADMIN` to the schema of the `EMPLOYEE` and `DEPARTMENT` tables in your `SAMPLE` database. Then save the file and run it by using the following command:

```
db2 -vtf view.sql
```

## ASNCLP script

This script generates SQL statements that create Capture and Apply control tables, a registration for the base `EMPLOYEE` table and another registration for the `EMPLOYEE_TRANSFORM` view, and a subscription set and member.

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

The script performs the following actions:

- 1** Setting the RUN NOW option
- 2** Setting up the source server
- 3** Registering the base `EMPLOYEE` table
- 4** Registering the `EMPLOYEE_TRANSFORM` view
- 5** Setting up the target server
- 6** Creating the subscription set
- 7** Creating a target object profile
- 8** Creating the subscription-set member
- 9** Ending the ASNCLP session

```

# 1 Setting the RUN NOW option
# This option prompts the ASNCLP to generate SQL scripts for creating
# replication objects and then run the scripts before generating the next
# SQL script. This option is required for this sample because, for example,
# the Capture control tables must be created before you can define a registration
# within them.

```

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

```

# 2 Setting up the source server
# Specifies the SAMPLE database as the Capture server and creates the
# Capture control tables.

```

```

SET SERVER CAPTURE TO DB SAMPLE;
CREATE CONTROL TABLES FOR CAPTURE SERVER;

```

```

# 3 Registering the base EMPLOYEE table
# To replicate from a view, you must first register the base table.
CREATE REGISTRATION (DB2ADMIN.EMPLOYEE) DIFFERENTIAL REFRESH STAGE CDEMPLOYEE;

```

```
# 4 Registering the EMPLOYEE_TRANSFORM view
```

```

# You do not specify a CD table when you register a view. The command generates
# a CD view name for you.
CREATE REGISTRATION (DB2ADMIN.EMPLOYEE_TRANSFORM) DIFFERENTIAL REFRESH;

# 5 Setting up the target server
# For this script we also use the SAMPLE database as the control server and
# target server.
SET SERVER CONTROL TO DB SAMPLE;
SET SERVER TARGET TO DB SAMPLE;
CREATE CONTROL TABLES FOR APPLY CONTROL SERVER;

# 6 Creating the subscription set
# The TIMING INTERVAL 1 option specifies that the Apply program process
# the set every minute.

CREATE SUBSCRIPTION SET SETNAME TFORM APPLYQUAL APPLYTF
ACTIVATE YES
TIMING INTERVAL 1 START DATE "2011-01-01" TIME "01:00:00.000000";

# 7 Creating a target object profile
# The profile specifies a container for the target table space that will be
# created for the target table. If you are running the script on Linux or
# UNIX, specify a Linux or UNIX filepath instead of C:\TFORM.FILE

SET PROFILE TRANSFORMSTS FOR OBJECT TARGET TABLESPACE OPTIONS UW USING
FILE "C:\TFORM.FILE" SIZE 700 PAGES;

# 8 Creating the subscription-set member
# The CREATE MEMBER command specifies the registered view EMPLOYEE_TRANSFORM
# as the replication source and creates a target table, EMPLOYEE_TUITION2.
# It also specifies that a new table space, EMPTUIT2, be created.

CREATE MEMBER IN SETNAME TFORM APPLYQUAL APPLYTF
ACTIVATE YES
SOURCE DB2ADMIN.EMPLOYEE_TRANSFORM
TARGET NAME DB2ADMIN.EMPLOYEE_TUITION2
DEFINITION IN EMPTUIT2 CREATE_USING PROFILE TRANSFORMSTS
KEYS(EMPNO +);

# 9 Ending the ASNCLP session
QUIT;

```

---

## ALTER DATASTAGE DEFINITION FOR command

Use the **ALTER DATASTAGE DEFINITION FOR** command to change the properties of an InfoSphere® DataStage® definition file (.dsx) for a consistent-change data (CCD) table that is used to feed a data warehouse.

### Syntax

►►—ALTER DATASTAGE DEFINITION FOR—SETNAME—*subscription\_set\_name*—APPLYQUAL—*apply\_qualifier*—◀◀

### Parameters

#### SETNAME

Specifies the subscription set to which the CCD member tables that are read by DataStage belong.

#### APPLYQUAL

Specifies the qualifier of the Apply program that processes the subscription set.

## Example

To change DataStage definitions for members within a subscription set called MYSET that is processed by an Apply program with the qualifier MYQUAL:

```
ALTER DATASTAGE DEFINITION FOR SETNAME "myset" APPLYQUAL "myqual";
```

---

## 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—  
▶—objowner—objname—TARGET—objname—  
▶—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

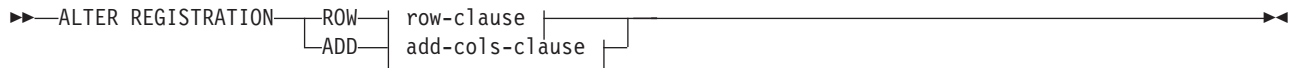
To add column NEWSTAFF to the existing subscription set SET00 :

```
ALTER MEMBER ADD COLS IN SETNAME SET00 APPLYQUAL A000 SOURCE DB2ADMIN.STAFF  
TARGET DB2ADMIN.TRGSTAFF COLS (NEWSTAFF TARGET NEWSTAFF)
```

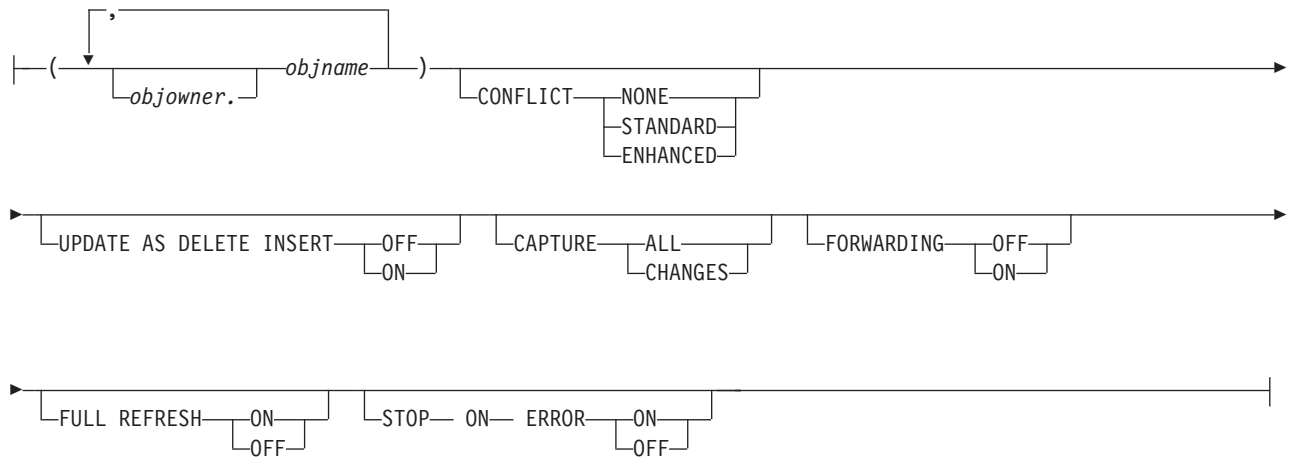
## ALTER REGISTRATION command

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

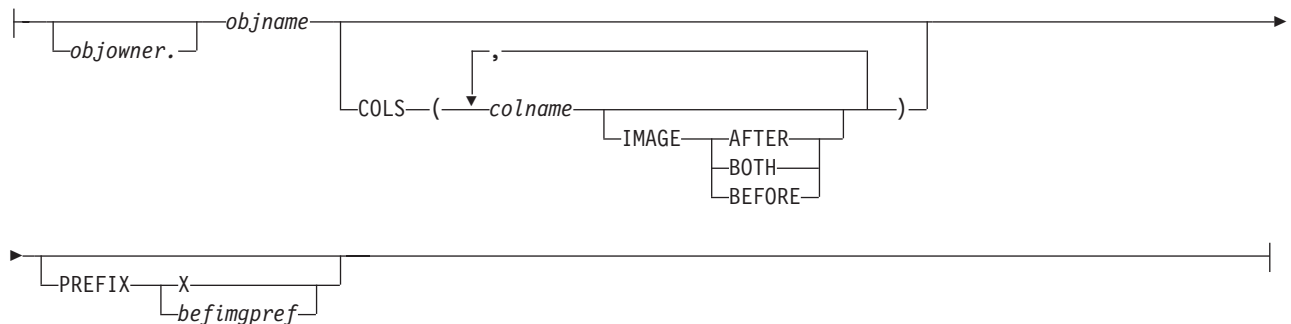
### Syntax



#### row-clause:



#### add-cols-clause:



### 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 by 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 by 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

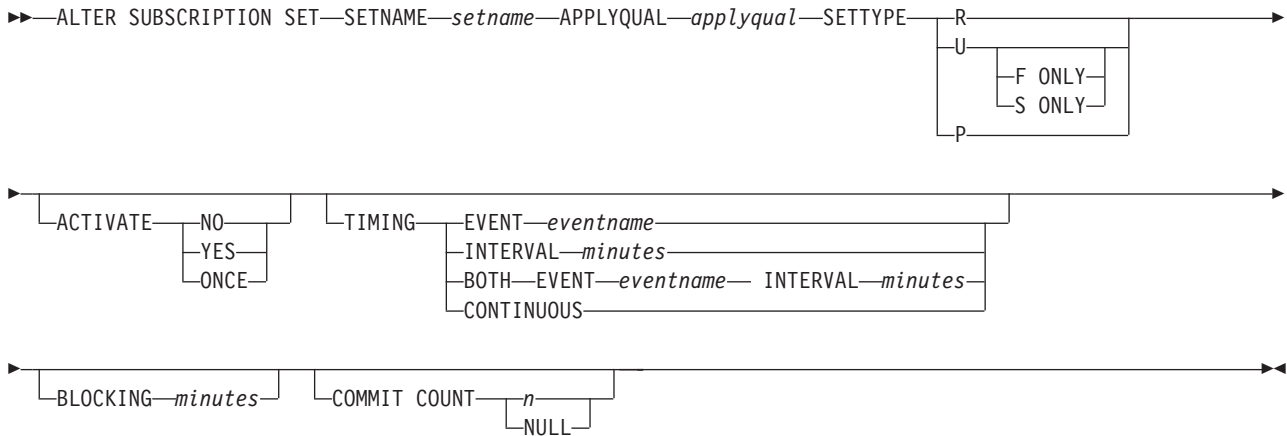
To alter a registration by adding a new column C002 to table DB2ADMIN.STAFF:  
ALTER REGISTRATION ADD DB2ADMIN.STAFF COLS (C002 IMAGE BOTH)

---

## ALTER SUBSCRIPTION SET command

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

### Syntax



### 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 master 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 within the AQ00 Apply qualifier to a read-only subscription set type 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

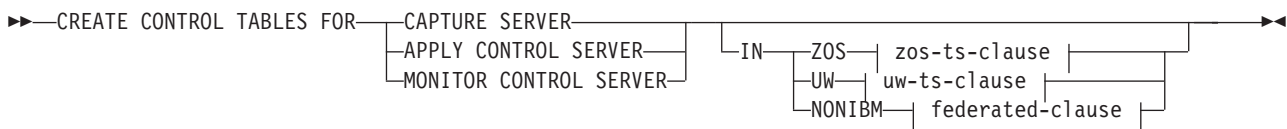
To set the ASNCLP session to SQL Replication:  
ASNCLP SESSION SET TO SQL REPLICATION

---

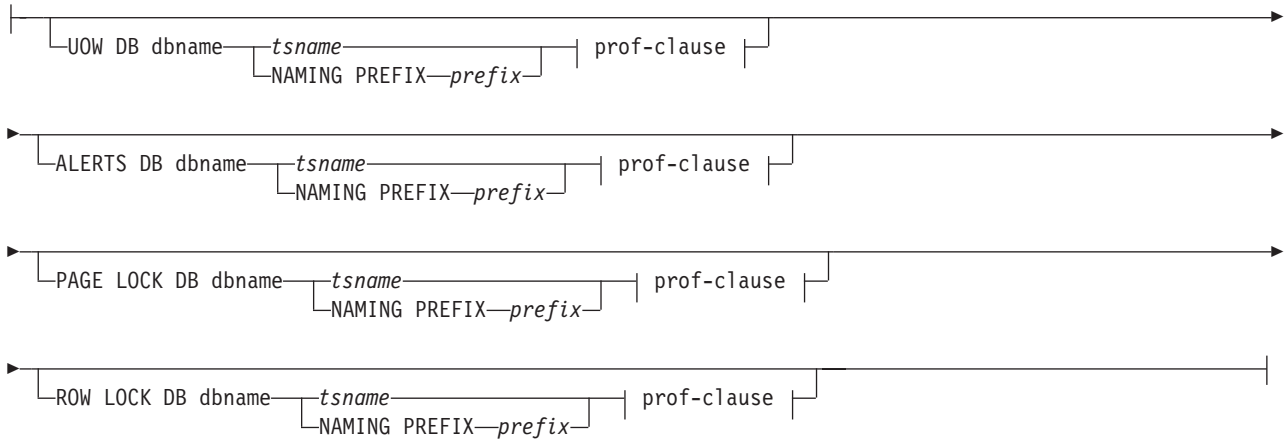
## CREATE CONTROL TABLES FOR command (SQL Replication)

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

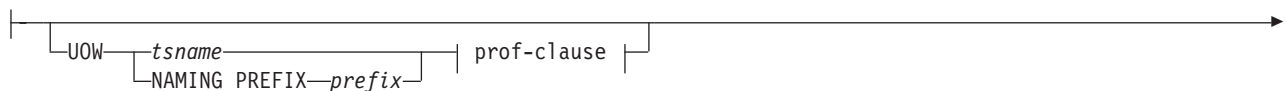
## Syntax

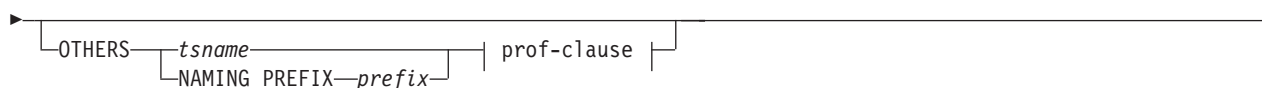


### zos-ts-clause:

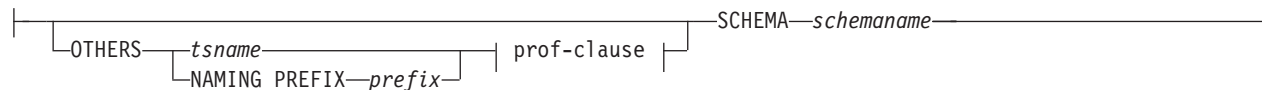


### uw-ts-clause:





**federated-clause:**



**prof-clause:**



**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 uppercase. 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 and use 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 DATASTAGE DEFINITION FOR command

Use the **CREATE DATASTAGE DEFINITION FOR** command to generate InfoSphere DataStage definition files (.dsx) that you can use to create DataStage jobs for reading data from a consistent-change data (CCD) table. The command also populates the IBMSNAP\_FEEDETL control table with information about the CCD members. DataStage reads and writes to this control table to keep track of which rows are ready to extract.

## Syntax

```
►► CREATE DATASTAGE DEFINITION FOR SETNAME subscription_set_name APPLYQUAL apply_qualifier ◀◀
```

### Parameters

#### SETNAME

Specifies the subscription set that the CCD member tables that will be read by DataStage belong to.

#### APPLYQUAL

Specifies the qualifier of the Apply program that processes the subscription set.

### Usage notes

- All member CCD tables in the set must be noncondensed.
- The IBMSNAP\_FEEDETL table must exist at the Apply control server.

### Example

To create DataStage definitions for members within a subscription set called MYSET that is processed by an Apply program with the qualifier MYQUAL:

```
SET SERVER CAPTURE TO DB SAMPLE ID user_ID PASSWORD "password";  
SET SERVER CONTROL TO DB TARGET ID user_ID PASSWORD "password";  
SET SERVER TARGET TO DB TARGET ID user_ID PASSWORD "password";
```

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

```
CREATE DATASTAGE DEFINITION FOR SETNAME "MYSET" APPLYQUAL "MYQUAL";
```

---

## 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.

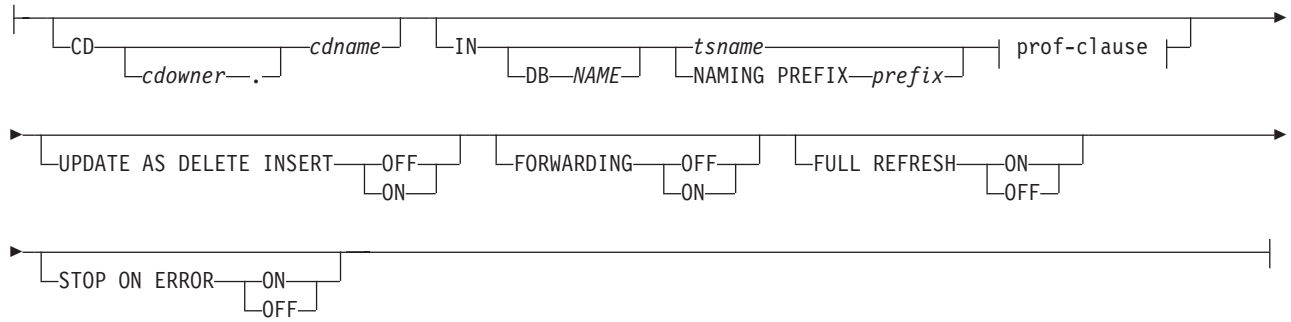
### Syntax

```
►► CREATE MEMBER IN SETNAME setname APPLYQUAL applyq [ACTIVATE {NO | YES | ONCE}] SOURCE  
[objowner.]objname [target-clause] [TGT KEY CHANGE {OFF | ON}]  
[WHERE "sql-where-stmts"]
```

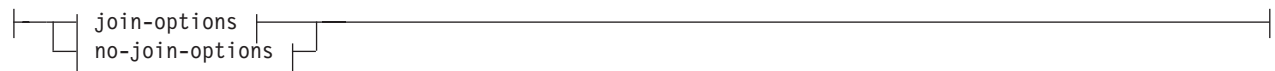




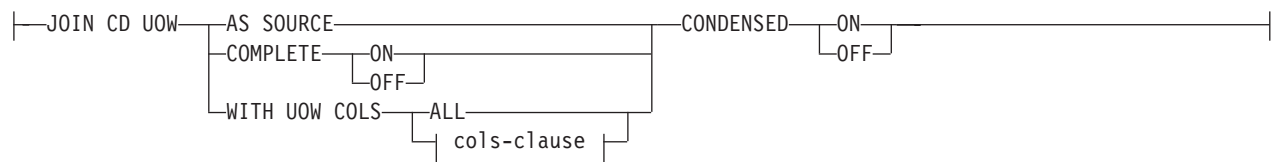
### replica-clause:



### ccd-clause:



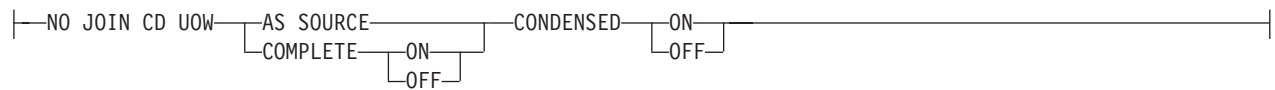
### join-options:



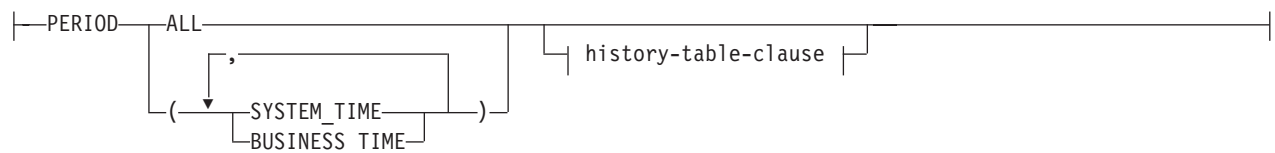
### cols-clause:



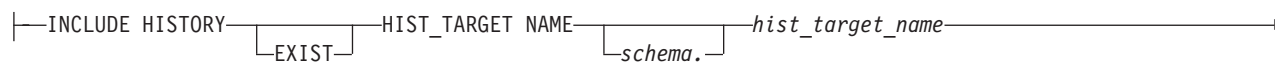
### no-join-options:



### period-clause:

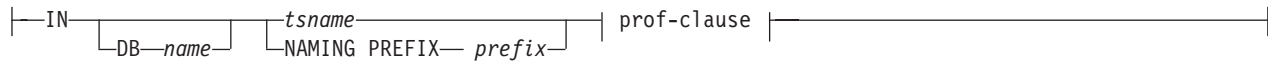


### history-table-clause:

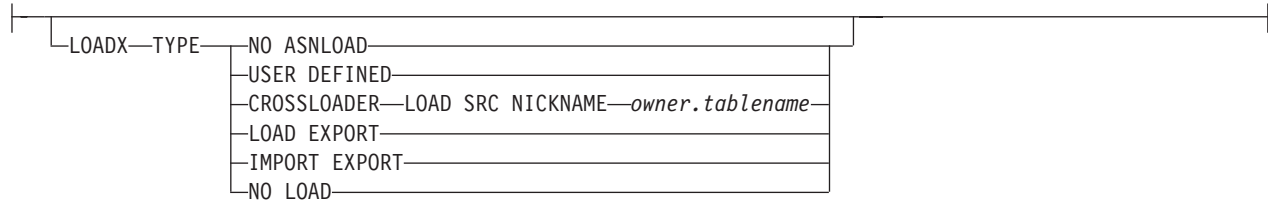




**tbspace-clause:**



**loadx-clause:**



**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** *"column\_or\_expression"*

The EXPRESSION keyword must precede the name of any source column that you want to include in the target table, or any SQL expression that you use to transform data between the source and target. Surround column names or expressions with double quotation marks ("). Separate multiple columns or expressions by using commas. The following example specifies that you want to include columns C1 and C2 from the source table:

```
COLS INCLUDE (EXPRESSION "C1", EXPRESSION "C2")
```

**TARGET** *column\_name*

You must use the TARGET keyword in the following cases:

- An expression is specified in the COLS INCLUDE statement. The TARGET keyword specifies the column or columns in the target table to which you want the results of the expression applied.
- The target table already exists, a regular source column name is used in the COLS INCLUDE statement, and the target column name is different from the source column name.

The following example specifies that you want to include two columns and an expression from the source table: column C1, column C2 mapped to a column named TGTC2 at the target, and an expression that concatenates the values in columns C3 and C4 from the source table and applies the new value into the column C3C4 at the target:

```
COLS INCLUDE (EXPRESSION "C1", EXPRESSION "C2" TARGET "TGTC2",  
EXPRESSION "C3||C4" TARGET "C3C4")
```

**EXCLUDE** (*column\_name*)

Specify to exclude one or more source columns from the target table definition. You can only use this keyword when you are creating a new target table, or for an existing target table when the source and target tables have the same column names.

**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.

**DEFINITION**

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

federated-clause

**REMOTE SCHEMA** *owner*

Specifies the schema of a new target table that is created by the ASNCLP. If this keyword is not used, the default schema is the remote authorization ID for the non-DB2 target database.

**REMOTE TABLE** *name*

Specifies the name of a new target table that is created by the ASNCLP. If this keyword is not used, the default table name is the name of the corresponding nickname in the federated database.

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 changes.

**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:

join-options:

**JOIN CD UOW**

Specifies that the CD table and IBMSNAP\_UOW table are joined to obtain commit information for transactions. The CCD table is created as type 3.

**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.

no-join-options:

**NO JOIN CD UOW**

Specifies that you do not want the CD table and IBMSNAP\_UOW table to be joined. The CCD table will be created with type 9.

**AS SOURCE**

Specifies that the CCD table is a source.

**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.

period-clause:

**PERIOD**

Specifies that the source table is a temporal table on DB2 10 for z/OS or later and you want to include some or all of the period columns in the subscription-set member.

**ALL**

Specifies that you want to include all period columns.

**SYSTEM\_TIME**

Specifies that you want to include the timestamp columns that are used with system-period temporal tables.

**BUSINESS\_TIME**

Specifies that you want to include the timestamp or date columns that are used with application-period temporal tables.

history-table-clause

**INCLUDE HISTORY**

Specifies that you want to create a subscription-set member for the history table that is associated with a temporal table with versioning on DB2 10 for z/OS or later.

**Note:** The subscription-set members for the base temporal table and its history table must be created in the same subscription set.

#### **EXIST**

Specifies that you want to create a subscription-set member for an existing history table.

#### **HIST\_TARGET NAME**

Specifies the name of the target history table. If you specify the EXIST keyword but do not specify a name, the ASNCLP program uses the history table for the target temporal table as the history target. Also use this keyword to specify the name for a new target history table that the ASNCLP creates.

tbspace-clause

#### **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 history table. If you want to use an existing table space, the target history table must be the only table that uses the table space.

##### **NAMING PREFIX** *prefix*

Specifies the prefix to use to name the table space.

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** *nickname.owner\_nickname\_name*

Specify the owner and name of the nickname to use with the LOAD from CURSOR utility for this member.

##### **LOAD EXPORT**

 Specify to use an EXPORT/LOAD combination for this member.

##### **IMPORT EXPORT**

 Specify to use an EXPORT/IMPORT combination for this member.

##### **NO LOAD**

Specify to indicate that no loading is performed 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 1: Create member for STAFF table

In this example, you create a member in the SET00 subscription set for mapping the STAFF source table to the TRGSTAFF target table. The TRGSTAFF table is created in the TSUOW100 table space and the index for the TRGSTAFF table is created according to the settings in the TBSPROFILE profile.

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

Linux UNIX Windows

### Example 2: Non-IBM target

The following commands set the environment and create a subscription set member with a Linux, UNIX, or Windows database as the Capture server and a Microsoft SQL Server target. The Apply control server is the same as the Capture server. The member has the following attributes:

- Subscription set name: SET1
- Apply qualifier: APPQUAL1
- Source owner: repldba
- Source table: EMPLOYEE
- Target nickname owner: repldba
- Target nickname: TRGEMPENICK

The commands create definitions for a new target table in the SQL Server database with a remote schema of dbo and a name of TRGEMPLOYEE.

```
SET SERVER CONTROL TO DB SAMPLE;
SET SERVER CAPTURE TO DB SAMPLE;
SET SERVER TARGET TO DB MSSQLDB NONIBM SERVER SQLSERVER;
SET OUTPUT CAPTURE SCRIPT "cap.sql";
SET OUTPUT TARGET SCRIPT "target.sql";
SET OUTPUT CONTROL SCRIPT "control.sql";
SET LOG "MEM.OUT";
CREATE MEMBER IN SETNAME SET1 APPLYQUAL APPQUAL1 ACTIVATE YES
  SOURCE repldba.EMPLOYEE TARGET NAME repldba.TRGEMPENICK
  REMOTE SCHEMA dbo REMOTE TABLE TRGEMPLOYEE;
```

### Example 3: Data distribution scenario

These commands set up a simple data distribution scenario that replicates the source table EMPLOYEE to two different target databases, TARGET1 and TARGET2. A profile is used at both target databases to specify table space characteristics for the target tables that the ASNCLP creates.

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

# Set up source database for replication
SET SERVER CAPTURE TO DB SAMPLE ID db2admin PASSWORD "mypw";
CREATE CONTROL TABLES FOR CAPTURE SERVER;

# Register source table and create corresponding CD table
CREATE REGISTRATION (db2admin.EMPLOYEE) DIFFERENTIAL REFRESH STAGE
CDEMPLOYEE;
```



```

# Set up first target database for replication
SET SERVER CONTROL TO DB TARGET1 ID db2admin PASSWORD "mypw";
SET SERVER TARGET TO DB TARGET1 ID db2admin PASSWORD "mypw";
CREATE CONTROL TABLES FOR APPLY CONTROL SERVER;

# Create subscription set from source to Target1
CREATE SUBSCRIPTION SET SETNAME SETA1 APPLYQUAL APPLY1 ACTIVATE YES
TIMING INTERVAL 1 START DATE "2011-01-01" TIME "01:00:00.000000";

# Specify table space characteristics for creating target tables at Target1
SET PROFILE TARGET1TS FOR OBJECT TARGET TABLESPACE OPTIONS UW USING
FILE "c:\TARGET1.FILE" SIZE 700 PAGES;

# Create subscription-set member from source to Target1
CREATE MEMBER IN SETNAME SETA1 APPLYQUAL APPLY1 ACTIVATE YES
SOURCE EMPLOYEE TARGET NAME EMPLOYEE
DEFINITION IN EMPTS CREATE USING PROFILE TARGET1TS
TYPE USERCOPY COLS ALL REGISTERED;

# Set up second target database for replication
SET SERVER CONTROL TO DB TARGET2 ID db2admin PASSWORD "mypw";
SET SERVER TARGET TO DB TARGET2 ID db2admin PASSWORD "mypw";
CREATE CONTROL TABLES FOR APPLY CONTROL SERVER;

# Create subscription set from source to Target2
CREATE SUBSCRIPTION SET SETNAME SETA2 APPLYQUAL APPLY2 ACTIVATE YES
TIMING INTERVAL 1 START DATE "2011-01-01" TIME "01:00:00.000000";

# Specify table space characteristics for creating target tables at Target2
SET PROFILE TARGET2TS FOR OBJECT TARGET TABLESPACE OPTIONS UW USING
FILE "c:\TARGET2.FILE" SIZE 700 PAGES;

# Create subscription-set member from source to Target2
CREATE MEMBER IN SETNAME SETA2 APPLYQUAL APPLY2 ACTIVATE YES
SOURCE EMPLOYEE TARGET NAME EMPLOYEE
DEFINITION IN EMPTS CREATE USING PROFILE TARGET2TS
TYPE USERCOPY COLS ALL REGISTERED;

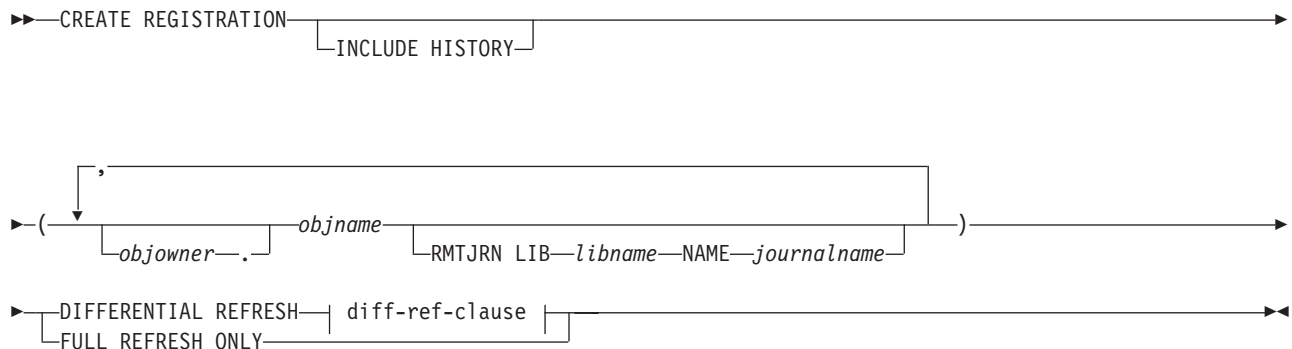
```

---

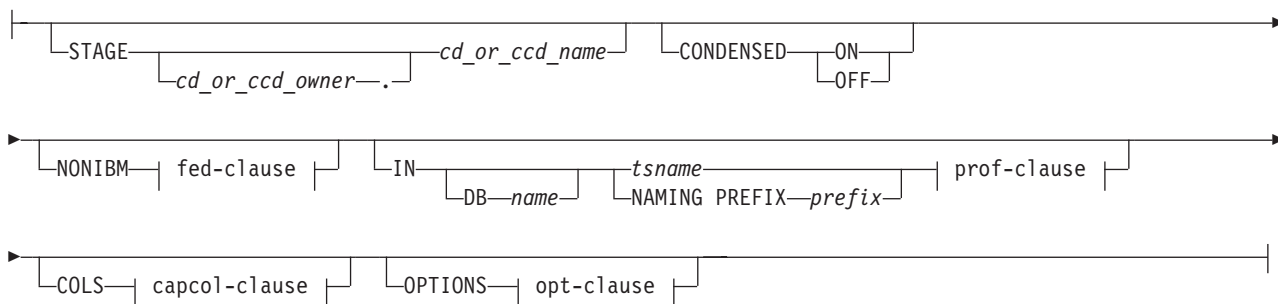
## CREATE REGISTRATION command

Use the **CREATE REGISTRATION** command to register one or more source tables, views, or nicknames for replication.

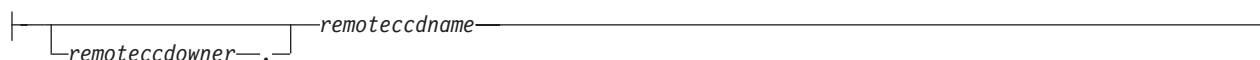
### Syntax



### diff-ref-clause:



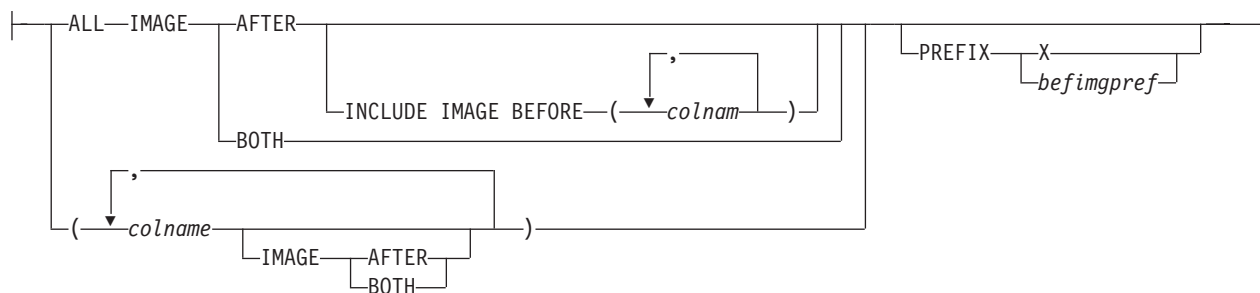
### fed-clause:



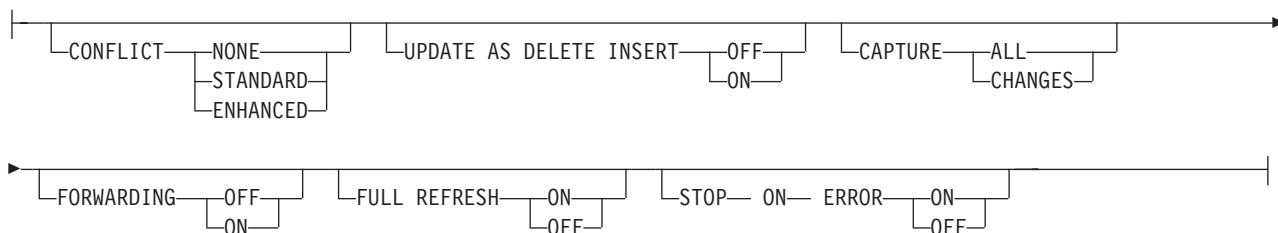
### prof-clause:



### capcol-clause:



### opt-clause:



## Parameters

### INCLUDE HISTORY

Specifies that you are registering a temporal table on DB2 10 for z/OS or later and you also want to register the associated history table.

*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*

**System i**

Specifies the System i library name.

**NAME** *journalname*

**System i**

Specifies the System i 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 by 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 a table. If the object is a view, you cannot register a subset of the 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.

**INCLUDE IMAGE BEFORE**

Specify to register before images along with after images for the listed columns.

*colname*

Specifies a list of the columns for which you want to register before images.

**IMAGE 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 by 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 by 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
```

## Example 3

To create a registration for DB2ADMIN.EMPLOYEE that updates the target table as the source objects change, registers after-images for all of the columns in the source table, and also registers before images for the SALARY and BONUS columns:

```
CREATE REGISTRATION (DB2ADMIN.EMPLOYEE) DIFFERENTIAL REFRESH  
COLS ALL IMAGE AFTER INCLUDE IMAGE BEFORE(SALARY,BONUS)PREFIX X;
```

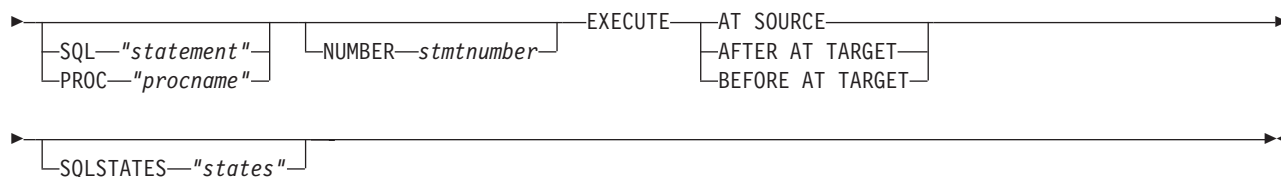
---

## 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
```



## 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 A000 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:

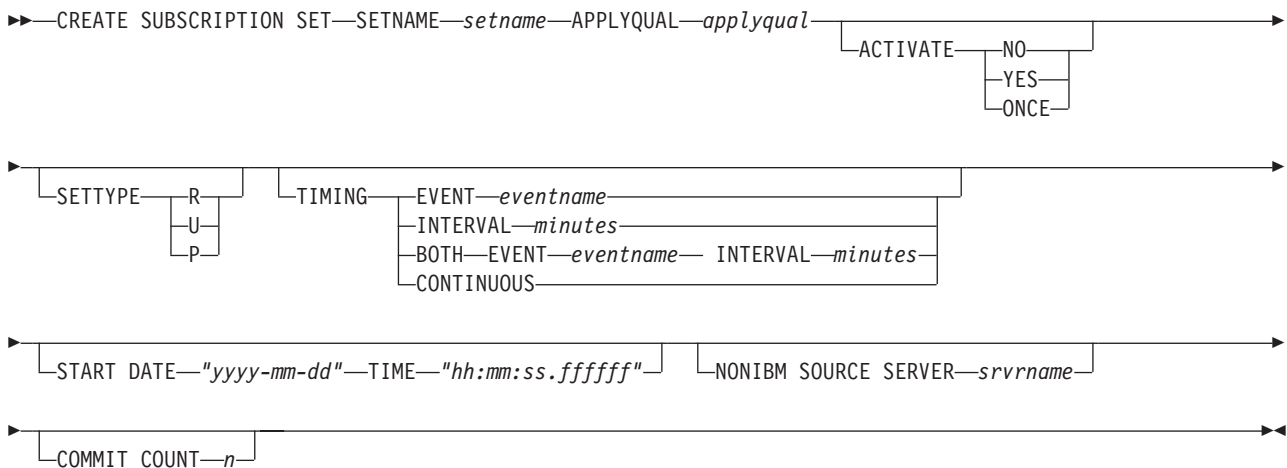
```
CREATE STMT IN SETNAME SET00 APPLYQUAL AQ00 PROC "procname" EXECUTE BEFORE AT TARGET
```

---

## CREATE SUBSCRIPTION SET command

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

### Syntax



### 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;

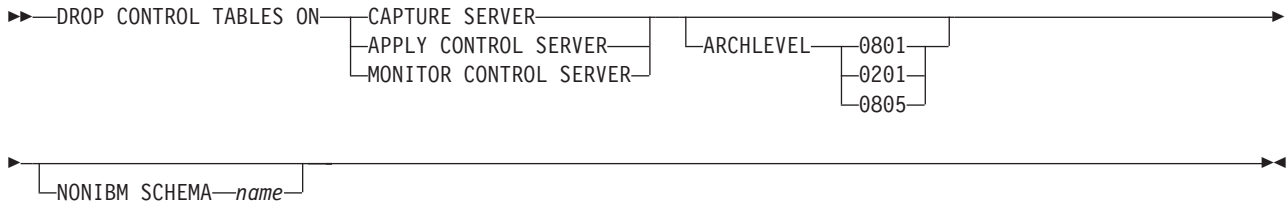
---

## 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



### 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

**z/OS** 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 DATASTAGE DEFINITION FOR

Use the **DROP DATASTAGE DEFINITION FOR** command to remove rows from the IBMSNAP\_FEEDETL control table that reference consistent-change data tables that feed InfoSphere DataStage.

### Syntax

```
►►—DROP DATASTAGE DEFINITION FOR—SETNAME—subscription_set_name—APPLYQUAL—apply_qualifier—◄◄
```

### Parameters

#### SETNAME

Specifies the subscription set to which the CCD member tables that are read by DataStage belong.

#### APPLYQUAL

Specifies the qualifier of the Apply program that processes the subscription set.

### Usage notes

This command does not remove information about the CCD tables from the IBMSNAP\_SUBS\_SET and IBMSNAP\_SUBS\_MEMBR tables, and therefore does not affect SQL Replication processing of the CCD tables. Also, the command does not delete the DataStage definition (.dsx) files that correspond to the tables. To remove the .dsx files from the DataStage project, use the InfoSphere QualityStage® and DataStage Designer.

### Example

To delete DataStage definitions for members within a subscription set called MYSET that is processed by an Apply program with the qualifier MYQUAL:

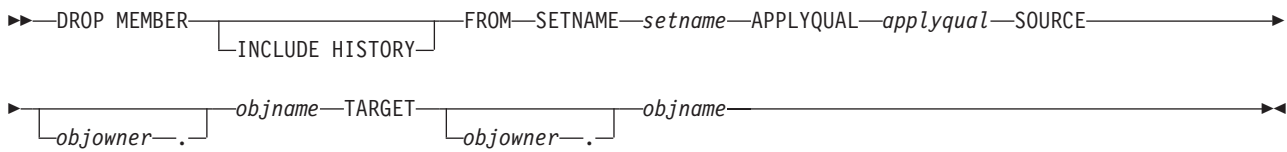
```
DROP DATASTAGE DEFINITION FOR SETNAME "MYSET" APPLYQUAL "MYQUAL";
```

---

## DROP MEMBER command

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

## Syntax



## Parameters

### INCLUDE HISTORY

Specify to delete the subscription-set member for the history table when the member for the base temporal table is deleted.

### 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:

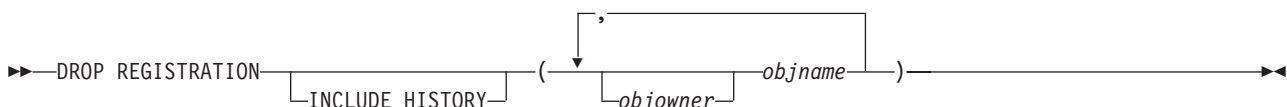
```
DROP MEMBER FROM SETNAME SET00 APPLYQUAL A000 SOURCE DB2ADMIN.STAFF  
TARGET DB2ADMIN.TRGSTAFF;
```

---

## DROP REGISTRATION command

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

## Syntax



## Parameters

*objowner*.

Specifies the owner of the source object (table, view, or nickname) for which you want to drop the registration.

*objname*

Specifies the name of the source object (table, view, or nickname) for which you want to drop the registration.

### INCLUDE HISTORY

Specify to delete the registration for the history table when the registration for the base temporal table is deleted.

## 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:

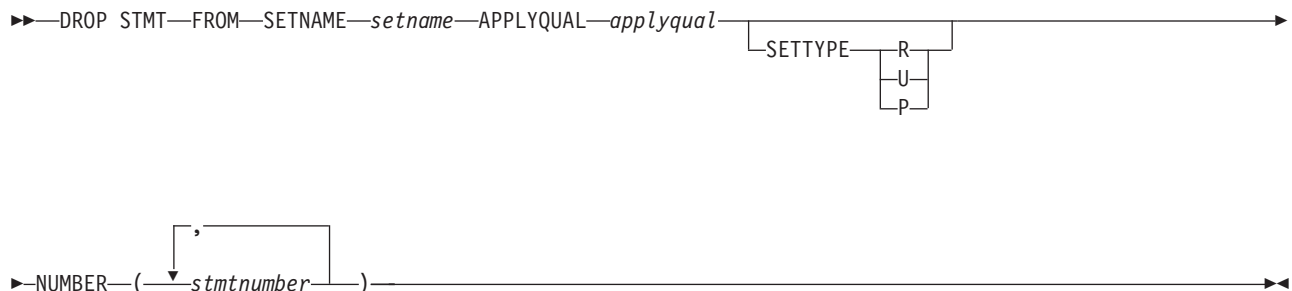
```
DROP REGISTRATION (DB2ADMIN.STAFF, DB2ADMIN.EMPLOYEE)
```

---

## DROP STMT command

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

## Syntax



## 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 43 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 | AFTER ) SETNAME setname APPLYQUAL applyqual

```

## 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 ( ( objowner . objname ) USING new-clause )

```

### new-clause:

```

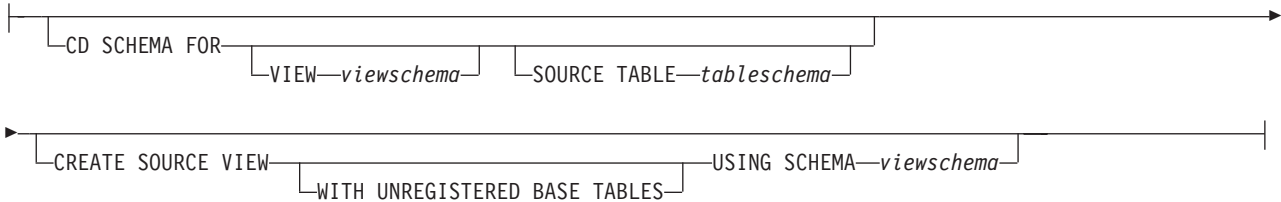
| SOURCE DB aliasname | CAPTURE SCHEMA schemaname | TABLE tbl-clause |
| | | VIEW view-clause |

```

## tbl-clause:



## view-clause:



## 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** 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

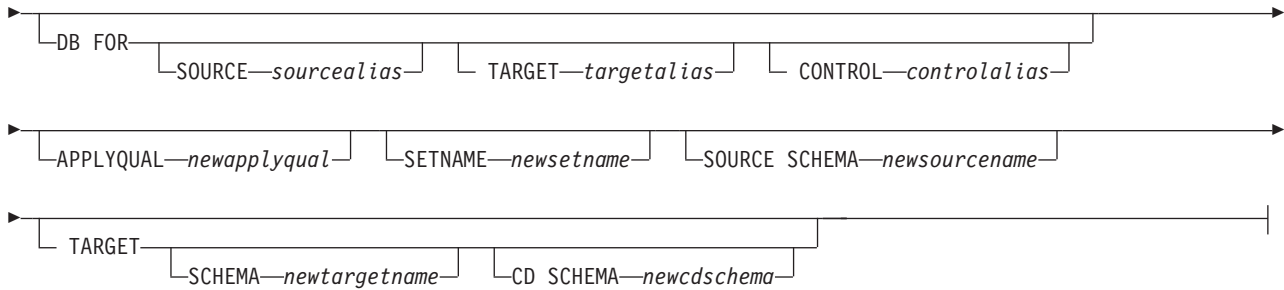
Use the **PROMOTE SUBSCRIPTION SET** command to recreate an existing subscription set in another replication environment.

### Syntax

```
►► PROMOTE SUBSCRIPTION SET SETNAME setname APPLYQUAL applyqual [ USING new-clause ]
```

#### new-clause::

```
[ CAPTURE SCHEMA FOR [ SOURCE sourcename ] [ REPLICA replicaname ] ]
```



## 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
```

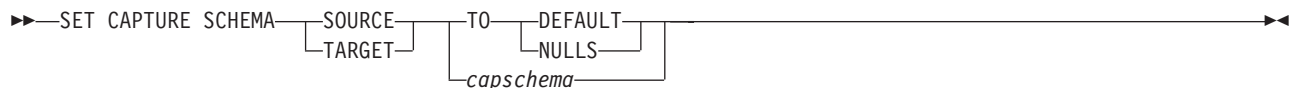
---

## 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



### 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
```

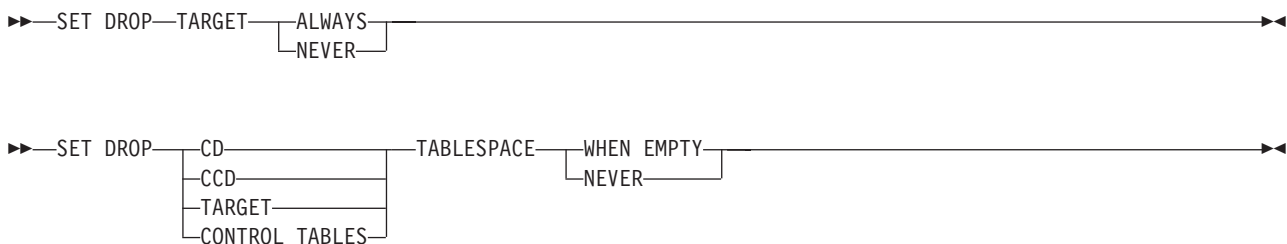
---

## 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



### 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"
```

---

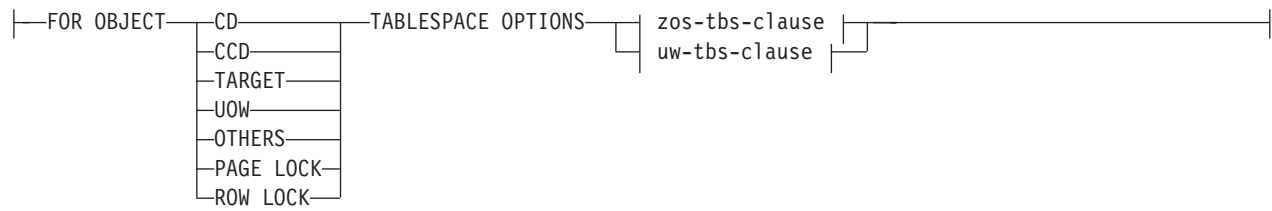
## SET PROFILE command (SQL Replication)

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.

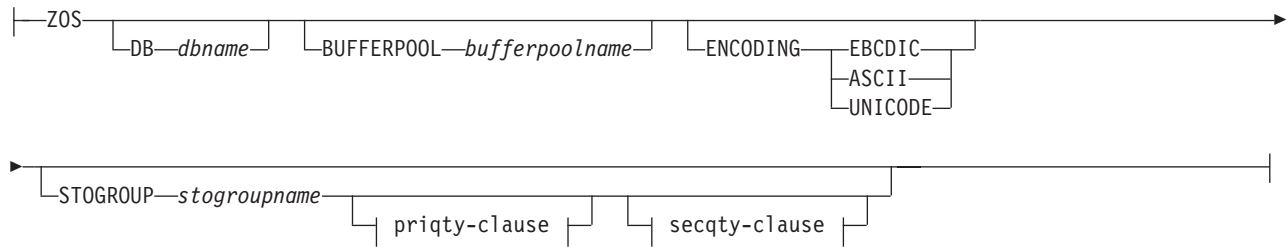
## Syntax

►►—SET PROFILE—*profilename* | *prof-clause* | UNDO

**prof-clause:**



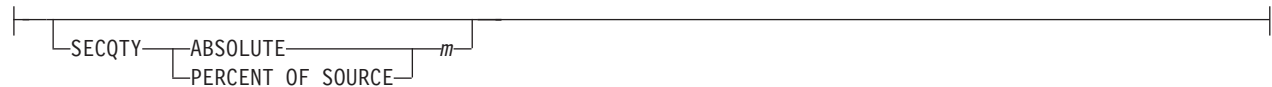
**zos-tbs-clause:**



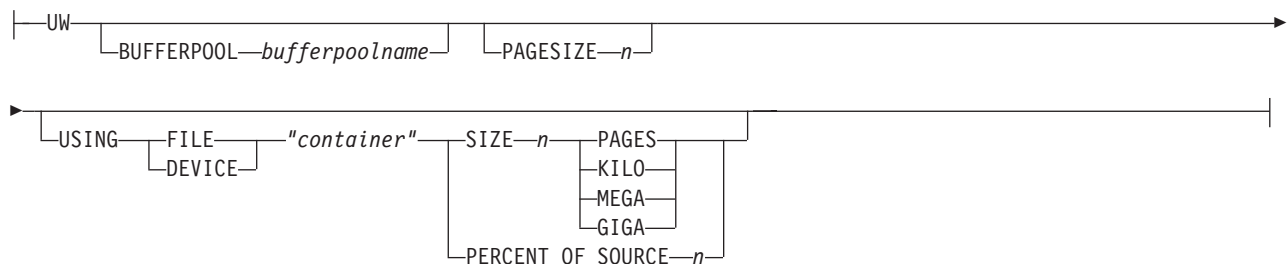
**priqty-clause:**



**secqty-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 Windows**

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

zos-tbs-clause:

**DB** *dbname*

**z/OS**

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-clause

**PRIQTY**

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

**ABSOLUTE**

Specifies an actual value in kilobytes (denoted as *n* in the syntax diagram) for primary space allocation. See the information about the **CREATE TABLESPACE** command 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 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.

secqty-clause

#### 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 *m* in the syntax diagram) for secondary space allocation. See the information about the **CREATE TABLESPACE** command 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 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.
- **System i** 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 3

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” on page 59 helps you understand when to run commands immediately and when to run them later.

### Syntax



## 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 the ASNCLP continues to process commands in the ASNCLP script file and statements in the generated SQL script file after one of the following errors:

- **ASNCLP script file:** An error while checking to predict whether the SQL statement to be generated will cause an SQL error. For example, a subscription cannot be defined in the control tables unless the control tables exist first.
- **Generated SQL script file:** An SQL error while running the SQL statements.

### ON (default)

Specify if you want the ASNCLP to stop processing commands in the ASNCLP script, and stop processing SQL statements in the generated SQL script, when the first validity check fails or SQL statement fails. If the error occurs while the ASNCLP is running the SQL script, previous SQL statements that are related to the task command with an error are rolled back.

**Note:** If the source scripts run correctly and the SQL statements in the scripts were committed but the target scripts have an SQL error, only the target scripts are rolled back. The committed source statements are not rolled back.

### OFF

Specify to process the ASNCLP commands and run all of the SQL statements, regardless of errors. You cannot use this parameter with Classic sources.

For a more complete explanation of how the ASNCLP responds to errors depending on this and other SET RUN SCRIPT options, see How the ASNCLP handles errors while processing scripts.

## 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.

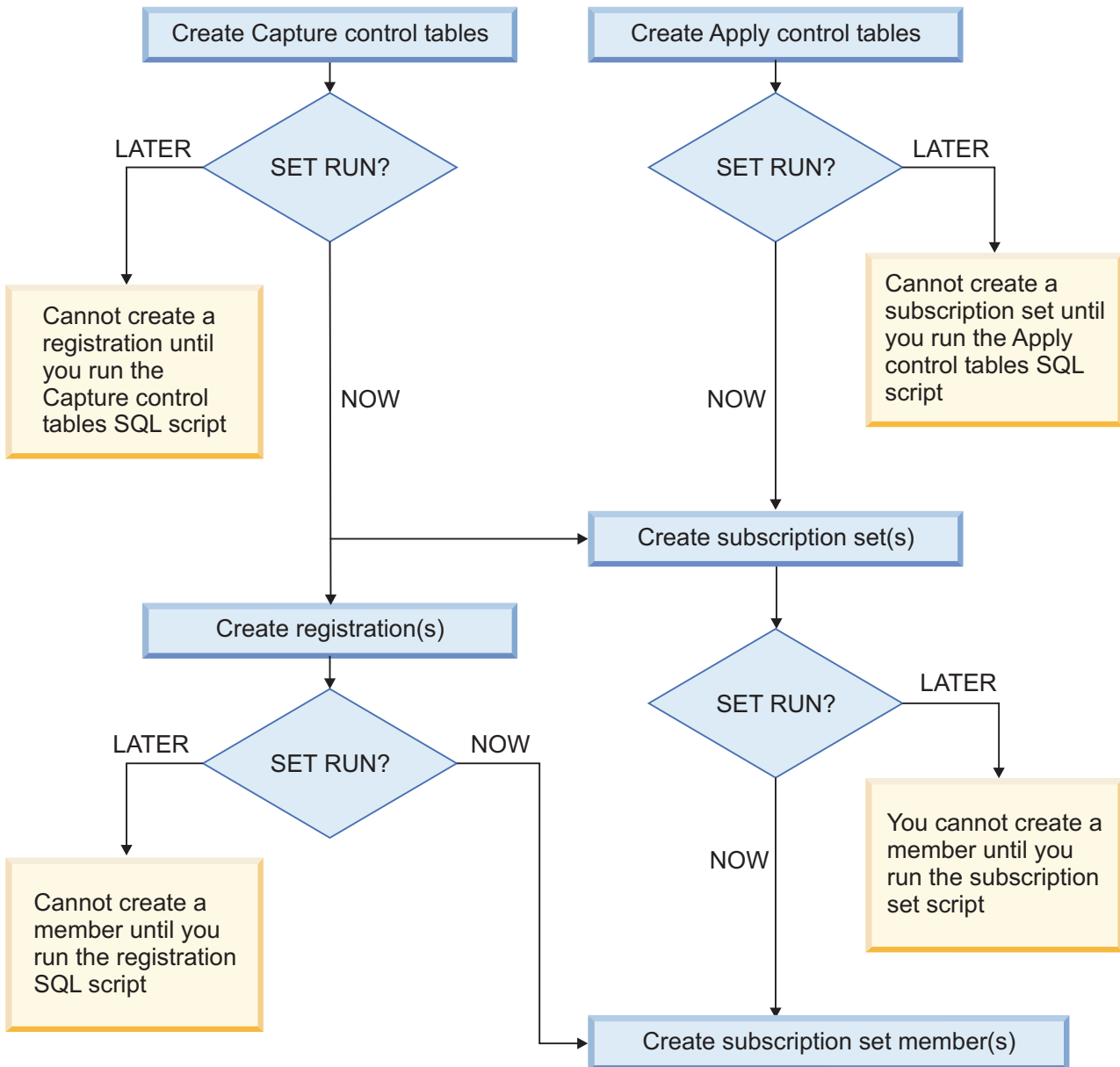


Figure 1. Dependencies between ASNCLP commands for SQL Replication. This diagram shows the dependencies between ASNCLP CREATE commands that are used to set up SQL Replication. It assumes all objects use the default schema of ASN.

## 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.
- 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 System i 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.

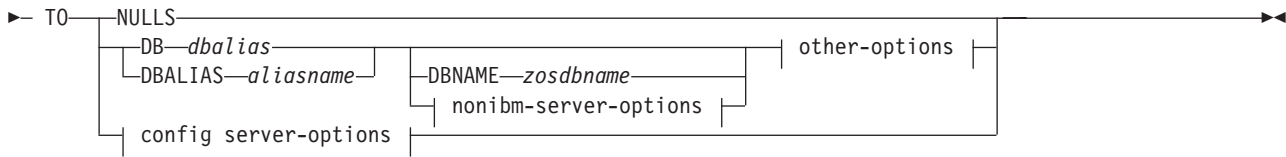
### All registration commands (including promote)

Set the Capture control server before running the registration commands. For System i, you must also set the remote source server.

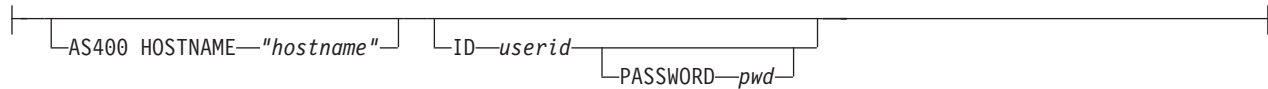
### All subscription commands (including promote)

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 System i, you must set the remote source server.

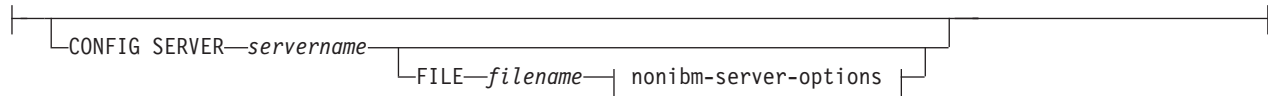
## Syntax



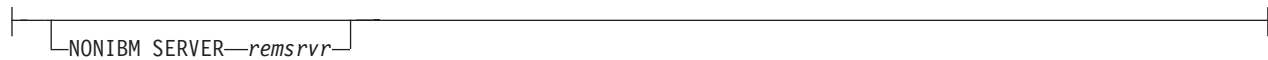
**other-options:**



**config server-options:**



**nonibm-server-options:**



**Parameters**

**ALL**

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

**REMOTE SOURCE**

**System i** 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 of a z/OS subsystem or Linux, UNIX, Windows, or System i database as cataloged on the DB2 from which the ASNCLP is invoked. This keyword is deprecated.

**DBALIAS** *aliasname*

Specifies the database alias name of a z/OS subsystem or Linux, UNIX, Windows, or System i database as cataloged on the DB2 from which the ASNCLP is invoked.

**DBNAME** *zosdbname*

**z/OS** Specifies the z/OS database name.

**Note:** DBNAME is mandatory when ASNCLP is running on z/OS and the Capture, target, or Apply control server is on z/OS. DBNAME is the name by which the DB2 database is known to local DB2 SQL applications. This name must match the name that was entered in the LOCATIONS column of the SYSIBM.LOCATIONS table in the CDB.

other-options clause:

**AS400 HOSTNAME** "*hostname*"

**System i** Specifies the System i 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. If you specify the user ID and do not specify the password, you will be prompted to enter the password.

**Note:** This keyword is not valid when the ASNCLP runs natively on z/OS because user authentication is handled through the communication database (CDB).

config server-options clause:

**CONFIG SERVER** *servername*

**DB2 sources only:** Specifies the DB2 source to connect with when the ASNCLP program is running on UNIX System Services (USS) for z/OS. The server name must match the bracketed [NAME] field that is entered in the ASNCLP configuration file.

**FILE** *filename*

Specifies the complete path and file name to the ASNCLP configuration file. If you do not use the FILE parameter, the ASNCLP program attempts to use the asnservers.ini file in the current directory, if that file exists.

nonibm server-options clause

**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.

**Note:** If the ASNCLP is running on USS, you must specify the NONIBM SERVER keyword along with the CONFIG SERVER keyword because an input file is required to connect to the source or target database.

## Usage notes

- Use the NONIBM SERVER clause to set up replication with 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"
```

### Example 3

To set the Capture control server and specify only the user ID in the command:

```
SET SERVER CAPTURE TO DB SAMPLE ID DB2ADMIN
```

You are prompted to enter the password. If you are running the commands from an input file in batch mode, the program waits for you to enter the password before the program processes the next commands.

### Example 4

In this example, the ASNCLP program is running on USS.

Given a configuration file called `sample.ini` that contains the following information:

```
[sample1]
Type=DB2
Data source=dsn7
Host=stplex4a.svl.ibm.com
Port=2080
```

Use the following command to specify the SAMPLE database as the Capture control server:

```
SET SERVER CAPTURE TO CONFIG SERVER sample1 FILE sample.ini ID id1 PASSWORD pwd1;
```

---

## 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. 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

This sample contains two ASNCLP scripts for setting up a unidirectional Q Replication environment. The first script generates commands to create WebSphere® MQ objects. The second script creates Q Capture and Q Apply control tables, a replication queue map, and a Q subscription.

You can copy the scripts to a text file, modify the values, and run the scripts by using the ASNCLP *-f filename* command. First:

- **Script 1:** Change the values for the MQHOST keywords to the IP addresses of the two systems, and ensure that the user ID that starts the ASNCLP program has permissions to execute the generated batch or shell script files.
- **Script 2:** Change db2admin and "passw0rd" to the user IDs and passwords for connecting to the two servers.

**Prerequisite:** The scripts require the replication administration tools to be at Version 9.7 Fix Pack 4.

#### ASNCLP script 1: Create WebSphere MQ objects

```
#####  
ASNCLP SESSION SET TO Q REPLICATION;  
  
CREATE MQ SCRIPT RUN NOW  
CONFIG TYPE U  
MQSERVER 1 NAME SAMPLE MQHOST "9.30.54.118",  
MQSERVER 2 NAME TARGETDB MQHOST "9.30.54.119",  
  
QUIT;  
#####
```

**Notes:** The CREATE MQ SCRIPT command generates two shell script files for Linux and UNIX systems (qrepl.sample.mq\_aixlinux.sh and qrepl.targetdb.mq\_aixlinux.sh) and two batch files for Windows systems (qrepl.sample.mq\_windows.bat and qrepl.targetdb.mq\_windows.bat). If you run the ASNCLP program on the same system as SAMPLE or TARGETDB, the RUN NOW option prompts the ASNCLP program to run the batch files or shell scripts to define the queue manager, queues, and other WebSphere MQ objects for that system. If the ASNCLP program is remote from either of the databases, you must run the appropriate batch file or shell script for these systems.

#### ASNCLP script 2: Set up Q Replication

```
#####  
ASNCLP SESSION SET TO Q REPLICATION;  
SET SERVER CAPTURE TO DBALIAS SAMPLE ID db2admin PASSWORD "passw0rd";  
SET SERVER TARGET TO DBALIAS TARGETDB ID db2admin PASSWORD "passw0rd";  
SET RUN SCRIPT NOW STOP ON SQL ERROR ON;  
  
CREATE CONTROL TABLES FOR CAPTURE SERVER;  
CREATE CONTROL TABLES FOR APPLY SERVER USING PWDFILE "asnpwd.aut";
```

```

CREATE REPLQMAP SAMPLE_ASN_TO_TARGETDB_ASN;

CREATE QSUB USING REPLQMAP SAMPLE_ASN_TO_TARGETDB_ASN
(SUBNAME EMPLOYEE0001 db2admin.EMPLOYEE OPTIONS HAS LOAD PHASE I
KEYS (EMPNO) LOAD TYPE 1);

QUIT;
#####

```

**Notes:** The commands in this script perform the following actions:

- The SET RUN SCRIPT NOW option prompts the ASNCLP program to generate SQL scripts for creating replication objects and then run the scripts. This option is required because some objects must be in place before others are created. For example, the Q Capture control tables must be created before you can define a Q subscription within them.
- The CREATE CONTROL TABLES FOR APPLY SERVER command specifies a password file, asnpwd.aut. This password file, which you can create with the asnpwd utility, contains the DB2 alias of the Q Capture server (SAMPLE). The Q Apply program uses this alias rather than a nickname to call the LOAD from CURSOR utility for loading the target table.
- For both the control tables and queue maps, the ASNCLP program by default uses the WebSphere MQ objects that were created with the CREATE MQ SCRIPT command.
- The CREATE QSUB command generates SQL to create a Q subscription named EMPLOYEE0001 that specifies the EMPLOYEE table as a source. By default the ASNCLP program generates SQL for creating a target table named 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 LOAD from CURSOR utility (LOAD TYPE 1).

---

## Sample ASNCLP scripts for setting up unidirectional Q Replication from a Classic data source

This sample contains three ASNCLP scripts for setting up a unidirectional Q Replication environment from a Classic data source. It includes 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. Table 2 on page 71 describes the SQL scripts that you create by running the samples. To create a Q subscription for a Classic source:

1. Use Classic Data Architect to create a relational mapping of the source table on the Classic server.
2. Create a Classic replication configuration file.
3. Create the Q Apply control tables
4. Update the capture parameters for the Classic data source
5. Create the replication queue map
6. Create the Q subscription

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 (#).

For help creating the WebSphere MQ objects that are used in these scripts, see WebSphere MQ setup script generator for Q Replication and Event Publishing and WebSphere MQ setup scripts for Q Replication.

## ASNCLP script 1: Create Q Apply control tables and update the capture parameters for the Classic data source

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

- 1** Setting the environment
- 2** Creating Q Apply control tables
- 3** Update the capture parameters for the Classic data source
- 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 the classicctrl.sql SQL script, which creates
# Q Apply control tables at the TARGET database.
# The SETQMANAGER commands are required for creating Q Replication control tables.

ASNCLP SESSION SET TO Q REPLICATION;
SET LOG "qcontrol.err";
SET SERVER TARGET TO DBALIAS TARGET ID DB2ADMIN PASSWORD "passw0rd";
SET QMANAGER "QM2" FOR APPLY SCHEMA;
SET APPLY SCHEMA ASN1;
SET OUTPUT TARGET SCRIPT "classicctrl.sql";

# 2 Creating Q Apply control tables
# This command specifies a password file, asnpwd.aut. The Q Apply program uses this
# file to connect to the Classic data source when it loads the target table.

CREATE CONTROL TABLES FOR APPLY SERVER IN UW TBSPACE TSQAPP;

# 3 Update the capture parameters for the Classic data source
# The following commands update the IBMQREP_CAPPARMS table to add parameters
# that specify the WebSphere MQ queue manager and queues that are used by
# the Classic capture components.

SET SERVER CAPTURE TO CONFIG SERVER classic1 FILE "asnserver.ini"
ID DB2ADMIN PASSWORD "passw0rd";
SET RUN SCRIPT NOW;
ALTER CAPTURE PARAMETERS QMGR asnqmgr RESTARTQ asnrestart ADMINQ asnadmin;

# 4 Ending the ASNCLP session

QUIT;
```

## ASNCLP script 2: Create the replication queue map

This script generates SQL statements to create a replication queue map. The script 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 the qappmap.sql SQL script,
# 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 CONFIG SERVER classic1 FILE "asnservers.ini"
ID CLASSICADMIN PASSWORD "passwd";
SET SERVER TARGET TO DBALIAS TARGET ID DB2ADMIN PASSWORD "passwd";
SET RUN SCRIPT NOW STOP ON SQL ERROR ON;
SET APPLY SCHEMA ASN1;
SET OUTPUT TARGET SCRIPT "qappmap.sql";

# 2 Creating a replication queue map
# This command generates SQL to create a replication queue map,
# CLASSIC_ASN1_TO_TARGET_ASN1. It specifies a remote administration
# 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 CLASSIC_ASN_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: Create the Q subscription

This script generates SQL statements to create a Q subscription. It specifies a source table, EMPLOYEE which is mapped to the Classic source through Classic Data Architect, and a new target table, TGTEMPLOYEE. 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 the qappsub.sql SQL script,
# 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 CONFIG SERVER classic1 FILE "asnservers.ini"
ID CLASSICADMIN PASSWORD "passwd";
SET SERVER TARGET TO DBALIAS TARGET ID DB2ADMIN PASSWORD "passwd";
SET RUN SCRIPT NOW STOP ON SQL ERROR ON;
SET APPLY SCHEMA ASN1;
SET OUTPUT TARGET SCRIPT "qappsub.sql";

# 2 Creating the Q subscription
# This command generates SQL to create a Q subscription named CLASSIC0001
# that specifies the CLASSICTABLE table as a source. The TARGET NAME keywords
# are used without the EXISTS or NAMING PREFIX keywords, resulting in a target
# table name of TGTCLASSICTABLE. The command also specifies that the Q
# Apply program load the target table (LOAD PHASE I) using LOAD TYPE 4.

CREATE QSUB USING REPLQMAP CLASSIC_ASN_TO_TARGET_ASN1
(SUBNAME CLASSIC0001 CLASSICTABLE OPTIONS HAS LOAD PHASE I
TARGET NAME CLASSICTABLE LOAD TYPE 4);

# 3 Ending the ASNCLP session

QUIT;

```

## Output of the scripts

Table 2 describes the SQL scripts that the ASNCLP sample scripts create.

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

Output file	Description
classicctrl.sql	Creates Q Apply control tables
qappqmap.sql	Inserts definitions for a replication queue map into the Q Apply control tables
qappqsub.sql	Inserts definitions for a Q subscription into the Q Apply control tables

---

## Sample ASNCLP scripts for setting up bidirectional Q Replication

This sample contains two ASNCLP scripts for setting up a bidirectional Q Replication environment. The first script generates commands to create WebSphere MQ objects at both systems. The second script creates Q Capture and Q Apply control tables at both servers, replication queue maps in both directions, and two bidirectional Q subscriptions.

The scenario involves two remote databases, SAMPLE and SAMPLE2. One table, EMPLOYEE, will be replicated in both directions between the two databases.

You can copy the scripts to a text file, modify the values, and run the scripts by using the ASNCLP *-f filename* command. First:

- **Script 1:** Change the values for the MQHOST keywords to the IP addresses of the two systems, and ensure that the user ID that starts the ASNCLP program has permissions to execute the generated batch or shell script files.
- **Script 2:** Change db2admin and "passw0rd" to the user IDs and passwords for connecting to the two servers.

**Prerequisite:** The scripts require the replication administration tools to be at Version 9.7 Fix Pack 4.

### ASNCLP script 1: Create WebSphere MQ objects

```
#####
ASNCLP SESSION SET TO Q REPLICATION;

CREATE MQ SCRIPT RUN NOW
CONFIG TYPE B
MQSERVER 1 NAME SAMPLE MQHOST "9.30.54.118",
MQSERVER 2 NAME SAMPLE2 MQHOST "9.30.54.119";

QUIT;
#####
```

**Notes:** The CREATE MQ SCRIPT command generates two shell script files for Linux and UNIX systems (qrep1.sample.mq\_aixlinux.sh and qrep1.sample2.mq\_aixlinux.sh) and two batch files for Windows systems (qrep1.sample.mq\_windows.bat and qrep1.sample2.mq\_windows.bat). If you run the ASNCLP program on the same system as SAMPLE or SAMPLE2, the RUN NOW option prompts the ASNCLP program to run the batch files or shell scripts to define the queue manager, queues, and other WebSphere MQ objects for that system. If the ASNCLP program is remote from either of the databases, you must run the appropriate batch file or shell script at these systems.

## ASNCLP script 2: Set up Q Replication

```
#####
ASNCLP SESSION SET TO Q REPLICATION;
SET BIDI NODE 1 SERVER DBALIAS SAMPLE ID db2admin PASSWORD "passw0rd" SCHEMA ASN;
SET BIDI NODE 2 SERVER DBALIAS SAMPLE2 ID db2admin PASSWORD "passw0rd" SCHEMA ASN;
SET RUN SCRIPT NOW STOP ON SQL ERROR ON;

CREATE CONTROL TABLES FOR NODE 1;
CREATE CONTROL TABLES FOR NODE 2;

CREATE REPLQMAP SAMPLE_TO_SAMPLE2 (NODE 1, NODE 2);
CREATE REPLQMAP SAMPLE2_TO_SAMPLE (NODE 2, NODE 1);

SET TABLES (SAMPLE.ASN.SMITH.EMPLOYEE);

CREATE QSUB SUBTYPE B;

QUIT;
#####
```

**Notes:** The commands in this script perform the following actions:

- The SET BIDI NODE commands specify the paired Q Capture and Q Apply servers at the SAMPLE and SAMPLE2 databases.
- The SET RUN SCRIPT NOW option prompts the ASNCLP program to generate and run the SQL to create all objects.
- The CREATE CONTROL TABLES FOR commands use the NODE 1 and NODE 2 keywords to prompt the ASNCLP program to create both Q Capture and Q Apply control tables at each server.
- In the CREATE REPLQMAP commands, the (NODE 1, NODE 2) and (NODE 2, NODE 1) syntax creates queue maps in both directions.
- For both the control tables and queue maps, the ASNCLP program by default uses the queue managers, queues, and other objects that were defined by the CREATE MQ SCRIPT command.
- The SET TABLES command specifies one table, SMITH.EMPLOYEE, at the SAMPLE database. This provides enough information to generate SQL statements for creating a matching table at SAMPLE2.

---

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

This sample contains two ASNCLP scripts for setting up a peer-to-peer Q Replication environment with two servers. The first script generates commands to create WebSphere MQ objects at both systems. The second script creates 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 involves two databases, SAMPLE and SAMPLPEER. One table, DEPARTMENT, will be replicated in both directions between the two databases.

You can copy the scripts to a text file, modify the values, and run the scripts by using the ASNCLP -f *filename* command. First:

- **Script 1:** Change the values for the MQHOST keywords to the IP addresses of the two systems, and ensure that the user ID that starts the ASNCLP program has permissions to execute the generated batch or shell script files.
- **Script 2:** Change db2admin and "passw0rd" to the user IDs and passwords for connecting to the two servers.



**Prerequisite:** The scripts require the replication administration tools to be at Version 9.7 Fix Pack 4.

## ASNCLP script 1: Create WebSphere MQ objects

```
#####
ASNCLP SESSION SET TO Q REPLICATION;

CREATE MQ SCRIPT RUN NOW
CONFIG TYPE P
MQSERVER 1 NAME SAMPLE MQHOST "9.30.54.118",
MQSERVER 2 NAME SAMPLPEER MQHOST "9.30.54.119";

QUIT;
#####
```

**Notes:** The CREATE MQ SCRIPT command generates two shell script files for Linux and UNIX systems (qrepl.sample.mq\_aixlinux.sh and qrepl.samlpeer.mq\_aixlinux.sh) and two batch files for Windows systems (qrepl.sample.mq\_windows.bat and qrepl.samlpeer.mq\_windows.bat). If you run the ASNCLP program on the same system as SAMPLE or SAMPLPEER, the RUN NOW option prompts the ASNCLP program to run the batch files or shell scripts to define the queue manager, queues, and other WebSphere MQ objects for that system. If the ASNCLP program is remote from either of the databases, you must run the appropriate batch file or shell script at these systems.

## ASNCLP script 2: Set up Q Replication

```
#####
ASNCLP SESSION SET TO Q REPLICATION;
SET PEER NODE 1 SERVER DBALIAS SAMPLE ID db2admin PASSWORD "passw0rd" SCHEMA ASN;
SET PEER NODE 2 SERVER DBALIAS SAMPLPEER ID db2admin PASSWORD "passw0rd" SCHEMA ASN;
SET RUN SCRIPT NOW STOP ON SQL ERROR ON;

CREATE CONTROL TABLES FOR NODE 1;
CREATE CONTROL TABLES FOR NODE 2;

CREATE REPLQMAP SAMPLE_TO_SAMPLPEER (NODE 1, NODE 2);
CREATE REPLQMAP SAMPLPEER_TO_SAMPLE (NODE 2, NODE 1);

SET TABLES (SAMPLE.ASN.SMITH.DEPARTMENT);

CREATE QSUB SUBTYPE P;

QUIT;
#####
```

**Notes:** The commands in this script perform the following actions:

- The SET PEER NODE commands specify the paired Q Capture and Q Apply servers at the SAMPLE and SAMPLPEER databases.
- The SET RUN SCRIPT NOW option prompts the ASNCLP program to generate and run the SQL to create all objects.
- The CREATE CONTROL TABLES FOR commands use the NODE 1 and NODE 2 keywords to prompt the ASNCLP program to create both Q Capture and Q Apply control tables at each server.
- In the CREATE REPLQMAP commands, the (NODE 1, NODE 2) and (NODE 2, NODE 1) syntax creates queue maps in both directions.
- For both the control tables and queue maps, the ASNCLP program by default uses the queue managers, queues, and other objects that were defined by the CREATE MQ SCRIPT command.

- The SET TABLES command specifies one table, SMITH.DEPARTMENT, at the SAMPLE database. This provides enough information to generate SQL statements for creating a matching table at SAMPLPEER.

---

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

This sample contains two ASNCLP scripts for setting up peer-to-peer Q Replication with three servers. The first script generates commands to create WebSphere MQ objects at all three systems. The second script 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.

You can copy the scripts to a text file, modify the values, and run the scripts by using the ASNCLP -f *filename* command. First:

- **Script 1:** Change the values for the MQHOST keywords to the IP addresses of the three systems, and ensure that the user ID that starts the ASNCLP program has permissions to execute the generated batch or shell script files.
- **Script 2:** Change db2admin and "passw0rd" to the user IDs and passwords for connecting to the three servers.

**Prerequisite:** The scripts require the replication administration tools to be at Version 9.7 Fix Pack 4.

### ASNCLP script 1: Create WebSphere MQ objects

```
#####
ASNCLP SESSION SET TO Q REPLICATION;

CREATE MQ SCRIPT RUN NOW
CONFIG TYPE P
MQSERVER 1 NAME SAMPLE MQHOST "9.30.54.118",
MQSERVER 2 NAME SAMPLE2 MQHOST "9.30.54.119",
MQSERVER 3 NAME SAMPLE2 MQHOST "9.30.54.120";

QUIT;
#####
```

**Notes:** The CREATE MQ SCRIPT command generates three shell script files for Linux and UNIX systems (qrepl.sample.mq\_aixlinux.sh, qrepl.sample2.mq\_aixlinux.sh, and qrepl.sample3.mq\_aixlinux.sh) and three batch files for Windows systems (qrepl.sample.mq\_windows.bat, qrepl.sample2.mq\_windows.bat, and qrepl.sample3.mq\_windows.bat). If you run the ASNCLP program on the same system as SAMPLE, SAMPLE2, or SAMPLE3, the RUN NOW option prompts the ASNCLP program to run the batch files or shell scripts to define the queue manager, queues, and other WebSphere MQ objects for that system. If the ASNCLP program is remote from any of the databases, do one of these things::

- Run the appropriate batch file or shell script at the remote system.
- Run the ASNCLP script on the remote system. The ASNCLP program will detect that it is local to that system and will automatically run the generated script.

### ASNCLP script 2: Set up Q Replication

```

ASNCLP SESSION SET TO Q REPLICATION;
SET PEER NODE 1 SERVER DBALIAS SAMPLE ID DB2ADMIN PASSWORD "passw0rd" SCHEMA ASN;
SET PEER NODE 2 SERVER DBALIAS SAMPLE2 ID DB2ADMIN PASSWORD "passw0rd" SCHEMA ASN;
SET PEER NODE 3 SERVER DBALIAS SAMPLE3 ID DB2ADMIN PASSWORD "passw0rd" SCHEMA ASN;
SET RUN SCRIPT NOW STOP ON SQL ERROR ON;

CREATE CONTROL TABLES FOR NODE 1 SERVER;
CREATE CONTROL TABLES FOR NODE 2 SERVER;
CREATE CONTROL TABLES FOR NODE 3 SERVER;

# First queue map (from SAMPLE to SAMPLE2)
CREATE REPLQMAP SAMPLE_ASN_TO_SAMPLE2_ASN (NODE 1, NODE 2);

# Second queue map (from SAMPLE2 to SAMPLE)
CREATE REPLQMAP SAMPLE2_ASN_TO_SAMPLE_ASN (NODE 2, NODE 1);

# Third queue map (from SAMPLE2 to SAMPLE3)
CREATE REPLQMAP SAMPLE2_ASN_TO_SAMPLE3_ASN (NODE 2, NODE 3);

# Fourth queue map (from SAMPLE3 to SAMPLE2)
CREATE REPLQMAP SAMPLE3_ASN_TO_SAMPLE2_ASN (NODE 3, NODE 2);

# Fifth queue map (from SAMPLE3 to SAMPLE)
CREATE REPLQMAP SAMPLE3_ASN_TO_SAMPLE_ASN (NODE 3, NODE 1);

# Sixth queue map (from SAMPLE to SAMPLE3)
CREATE REPLQMAP SAMPLE_ASN_TO_SAMPLE3_ASN (NODE 1, NODE 3);

SET SUBGROUP "p2p3group";

SET TABLES (SAMPLE.ASN.ELB.STAFF);

CREATE QSUB SUBTYPE P;

QUIT;

```

**Notes:** The commands in this script perform the following actions:

- The SET PEER NODE commands specify the paired Q Capture and Q Apply servers at the three databases.
- The SET RUN SCRIPT NOW option prompts the ASNCLP program to generate and run the SQL to create all objects.
- The CREATE CONTROL TABLES FOR commands use the NODE 1, NODE 2, and NODE 3 keywords to prompt the ASNCLP program to create both Q Capture and Q Apply control tables at each server.
- In the CREATE REPLQMAP commands, the (NODE 1, NODE 2) syntax creates queue maps in both directions between each server.
- For both the control tables and queue maps, the ASNCLP program by default uses the queue managers, queues, and other objects that were defined by the CREATE MQ SCRIPT command.
- The SET SUBGROUP command assigns a group name, p2p3group, to all Q subscriptions in the peer-to-peer group.
- The SET TABLES command specifies one table, ELB.STAFF, at the SAMPLE database. This provides enough information to generate SQL statements for creating matching tables at SAMPLE2 and SAMPLE3.
- A single CREATE QSUB command generates statements to create six peer-to-peer Q subscriptions between SAMPLE, SAMPLE2, and SAMPLE3.

---

## Sample ASNCLP script for promoting unidirectional configurations

This sample contains an ASNCLP script for promoting a unidirectional Q Replication configuration. You can copy an existing Q Replication or event publishing configuration to another system by *promoting* that configuration by using a set of ASNCLP scripts. These commands scan and discover the Q Replication control table and DB2 catalog table on specified source servers, and then create replication definitions. You can execute scripts containing these definitions on any destination server to recreate the replication environment there.

You can customize some of the properties of the destination environment.

Suppose you want to promote a replication environment you have created on a test server configuration to your production server configuration. The test configuration consists of Q Capture server TESTCAP and Q Apply server TESTAPP, with the following details:

- Q Capture control tables on server TESTCAP exist under schema ASN
- Q Apply control tables on server TESTAPP exist under schema ASN
- 10 replication queue maps exist between the servers, named qmap1 to qmap10
- 30 Q subscriptions exist on each queue map
- A total of 300 unidirectional Q subscriptions exist between these servers

To promote all replication queue maps and all Q subscriptions that use them from the test environment to the production one, create the following ASNCLP input script:

```
ASNCLP SESSION SET TO Q REPLICATION;
SET LOG promote-repqmap-qsub.log;

SET SERVER CAPTURE TO DBALIAS TESTCAP ID id1 PASSWORD "p1"
PROMOTE TO DBALIAS PRODCAP ID id1 PASSWORD "p1wd" SCHEMA ASN;

SET SERVER TARGET TO DBALIAS TESTAPP ID id1 PASSWORD "p1wd"
PROMOTE TO DBALIAS PRODAPP ID id1 PASSWORD "p1wd" SCHEMA ASN;

#This is the output script that will be generated by these commands
SET OUTPUT PROMOTE SCRIPT "replqmap_qsub.in";

#These two statements will be put in the generated script
SET OUTPUT CAPTURE SCRIPT "promote_capture_repqmap.sql";
SET OUTPUT TARGET SCRIPT "promote_target_repqmap.sql";

#Generate ASNCLP commands for promoting all queue maps that match this predicate
PROMOTE REPLQMAP LIKE "qmap%";

#Generate ASNCLP commands for promoting all Q subscriptions that use these
#queue maps
PROMOTE QSUB REPLQMAP LIKE "qmap%";
```

The output of this script is another ASNCLP script that is named `replqmap_qsub.in` which includes the command **SET RUN SCRIPT LATER**. Using **SET RUN SCRIPT LATER** lets you confirm or modify the script contents after it is generated and before running it. Change this **SET RUN** command to **SET RUN SCRIPT NOW STOP ON SQL ERROR ON** when you want to run the script.

Running this script by using `asnclp -f "asnclp_replqmap.in"` executes the SQL definitions and persists the information in the control tables, promoting the specified environment.

---

## Sample ASNCLP scripts for promoting peer-to-peer configurations

This sample contains three ASNCLP scripts for promoting a peer-to-peer Q Replication configuration. You can copy an existing Q Replication or Event Publishing configuration to another system by *promoting* that configuration by using a set of ASNCLP scripts. These commands scan and discover the Q Replication control table and DB2 catalog table on the specified source servers, and then create replication definitions. You can execute scripts containing these definitions on any destination server to recreate the replication environment there.

The scenario for these samples involves an existing configuration with peer-to-peer Q subscriptions between server SAMPLE, schema ASN and server TESTDB, schema BSN:

- Q Capture and Q Apply control tables exist on server SAMPLE under schema ASN, and on server TESTDB under schema BSN
- Two replication queue maps exist between SAMPLE.ASN and TESTDB.BSN:
  - RQMap1 includes send queue SQ1, receive queue RQ1, and administration queue AQ1.
  - RQMap2 includes send queue SQ2, receive queue RQ2, and administration queue AQ2.

The sample scripts promote existing objects for the peer-to-peer configuration to server SAMPLE.ASN1 and server TESTDB1.BSN1. The scripts promote both replication queue maps and all Q subscriptions that use these queue maps.

### Create the control tables on destination servers

These scripts assume that you first created Q Capture and Q Apply control tables on the promote destinations: on server SAMPLE under schema ASN1 and on server TESTDB1 under schema BSN1. Create the control tables by using the **CREATE CONTROL TABLES FOR** command or the Replication Center.

### Promote the first replication queue map

This script promotes the replication queue map that moves data from the first peer to the second peer.

```
ASNCLP SESSION SET TO Q REPLICATION;  
SET LOG promote_repqmaplog;
```

```
#Identify the first peer's Q Capture and the second peer's Q Apply. The promote-to  
#passwords are added to the generated script, but no connect is issued to the  
#promote-to servers until the generated script is run.
```

```
#Identify the Q Capture server for the first peer and the Q Apply server for the  
#second peer. The specified promote-to passwords are added to the generated script  
#so that it can successfully execute. This connection information is only used  
#when the generated script in the output script file is run.
```

```
SET SERVER CAPTURE TO DBALIAS SAMPLE ID id1 PASSWORD "p1"  
PROMOTE TO DBALIAS SAMPLE ID id1 PASSWORD "p1wd" SCHEMA ASN1;
```

```
SET SERVER TARGET TO DBALIAS TESTDB ID id1 PASSWORD "p1wd"  
PROMOTE TO DBALIAS TESTDB1 ID id1 PASSWORD "p1wd" SCHEMA BSN1;
```

```

#This command defines the file that contains the output script that is generated.
SET OUTPUT PROMOTE SCRIPT "repqmap.in";

#These two SET OUTPUT statements are put in the generated script.
SET OUTPUT CAPTURE SCRIPT "promote_capture_repqmap.sql";
SET OUTPUT TARGET SCRIPT "promote_target_repqmap.sql";

#Generate the ASNCLP commands for promoting the replication queue
#map that is named RQMap1.
PROMOTE REPLQMAP NAME RQMap1;

```

## Promote the second replication queue map

This script promotes the replication queue map that moves data from the second peer to the first peer.

```

ASNCLP SESSION SET TO Q REPLICATION;
SET LOG promote_repqmaplog;

#Identify the Q Capture server for the second peer and the Q Apply server for the
#first peer. The specified promote-to passwords are added to the generated script
#so that it can successfully execute. This connection information is only used
#when the generated script in the output script file is run.

SET SERVER CAPTURE TO DBALIAS TESTDB ID id1 PASSWORD "p1"
PROMOTE TO DBALIAS TESTDB1 ID id1 PASSWORD "p1wd" SCHEMA BSN1;

SET SERVER TARGET TO DBALIAS SAMPLE ID id1 PASSWORD "p1wd"
PROMOTE TO DBALIAS SAMPLE ID id1 PASSWORD "p1wd" SCHEMA ASN1;

```

```

#This command defines the file that contains the output script that is generated.
SET OUTPUT PROMOTE SCRIPT "repqmap.in";

#These two SET OUTPUT statements are put in the generated script
SET OUTPUT CAPTURE SCRIPT "promote_capture_repqmap.sql";
SET OUTPUT TARGET SCRIPT "promote_target_repqmap.sql";

#Generate the ASNCLP commands for promoting the replication queue
#map that is named RQMap2.
PROMOTE REPLQMAP NAME RQMap2;

```

## Run the generated scripts

Run the generated ASNCLP scripts by using the **asnclp -f repqmap.in** command from a system command prompt. Run the SQL output that is generated by these scripts.

## Promote the Q subscriptions

Promote the peer-to-peer Q subscriptions for the replication queue maps:

```

SET PEER NODE 1 SERVER DBALIAS SAMPLE ID id1 PASSWORD "p1wd" SCHEMA ASN
PROMOTE TO DBALIAS SAMPLE ID id1 PASSWORD "p1wd" SCHEMA ASN1;

SET PEER NODE 2 SERVER DBALIAS TESTDB ID id1 PASSWORD "p1wd" SCHEMA BSN
PROMOTE TO DBALIAS TESTDB1 ID id1 PASSWORD "p1wd" SCHEMA BSN1;

#Generate the ASNCLP scripts to promote all Q subscriptions that use
#replication queue maps with names that begin with the predicate RQMAP:
PROMOTE QSUB REPLQMAP LIKE "RQMAP%";

```

## Chapter 4. ASNCLP commands for unidirectional Q Replication

The ASNCLP commands for unidirectional Q Replication set the environment, define, change, and delete Q subscriptions, and specify output files. Some of the ASNCLP commands for unidirectional replication also apply to Classic replication.

“Sample ASNCLP scripts for setting up unidirectional Q Replication” on page 67 and “Sample ASNCLP scripts for setting up unidirectional Q Replication from a Classic data source” on page 68 demonstrate how you can combine ASNCLP commands to create an ASNCLP setup script.

**Note:** All of the commands for Q Replication and Classic replication require that you set the environment with the ASNCLP SESSION SET TO Q REPLICATION command.

Table 3 lists the ASNCLP commands for unidirectional Q Replication and links to topics that describe each command.

*Table 3. ASNCLP commands for unidirectional Q Replication*

If you want to ...	Use this command
Add a column to a Q subscription	ALTER ADD COLUMN command
Update the IBMQREP_CAPPARMS table when you replicate from a Classic source	“ALTER CAPTURE PARAMETERS command (Classic replication)” on page 82
Change a Q subscription	“ALTER QSUB command (unidirectional replication)” on page 84
Change a replication queue map	ALTER REPLQMAP command
Establish a session for Q Replication	ASNCLP SESSION SET TO command
Create a Q subscription	“CREATE QSUB command (unidirectional replication)” on page 101
Create the control tables for the Q Capture and Q Apply programs	CREATE CONTROL TABLES FOR command
Create a replication queue map	CREATE REPLQMAP command
Create a schema-level subscription	“CREATE SCHEMASUB command” on page 124
Create a profile for table-level Q subscriptions	“CREATE SUBSCRIPTION OPTIONS command” on page 127
Delete a Q subscription	“DROP QSUB command” on page 129
Drop the control tables for the Q Capture and Q Apply programs	DROP CONTROL TABLES ON command
Delete a replication queue map	DROP REPLQMAP command
Delete a schema-level subscription	“DROP SCHEMASUB command” on page 133
Delete a profile for table-level Q subscriptions	“DROP SUBSCRIPTION OPTIONS command” on page 133
List Q subscriptions	“LIST QSUB command (Q Replication)” on page 134
List replication queue maps	“LIST REPLQMAP command (Q Replication)” on page 135
List Q Apply schemas	LIST APPLY SCHEMA command
List Q Capture schemas	LIST CAPTURE SCHEMA command
List schema-level subscriptions	“LIST SCHEMASUB command” on page 139



Table 3. ASNCLP commands for unidirectional Q Replication (continued)

If you want to ...	Use this command
Signal that a manual load of the target table is complete	LOAD DONE command
Promote a Q subscription	PROMOTE QSUB command (unidirectional)
Promote a replication queue map	PROMOTE REPLQMAP command
Reinitialize a schema-level subscription	"REINIT SCHEMASUB command" on page 143
<ul style="list-style-type: none"> <li>Specify whether to drop the target table when you delete a Q subscription</li> <li>Specify whether to drop the table space when you drop the target table or control tables</li> </ul>	"SET DROP command (unidirectional replication)" on page 145
Set the Q Apply schema for all task commands	SET APPLY SCHEMA command
Set the Q Capture schema for all task commands	SET CAPTURE SCHEMA command
Define the log file for the ASNCLP program	SET LOG command
Define output files that contain SQL statements to set up unidirectional Q Replication	SET OUTPUT command
Specify custom parameters for database objects to be created implicitly	SET PROFILE command
Set the WebSphere MQ queue manager	SET QMANAGER command
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
Specify the Q Capture server or Q Apply server to use in the ASNCLP session for unidirectional replication.	SET SERVER command
Enable and disable the trace for the ASNCLP commands	SET TRACE command
Display the environment set during the session	SHOW SET ENV command
Start a Q subscription	START QSUB command
Start a schema-level subscription	"START SCHEMASUB command" on page 162
Stop a Q subscription	STOP QSUB command
Stop a schema-level subscription	"STOP SCHEMASUB command" on page 165
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
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

## ALTER ADD COLUMN command (unidirectional replication)

Use the **ALTER ADD COLUMN** command to add a column to a Q subscription.

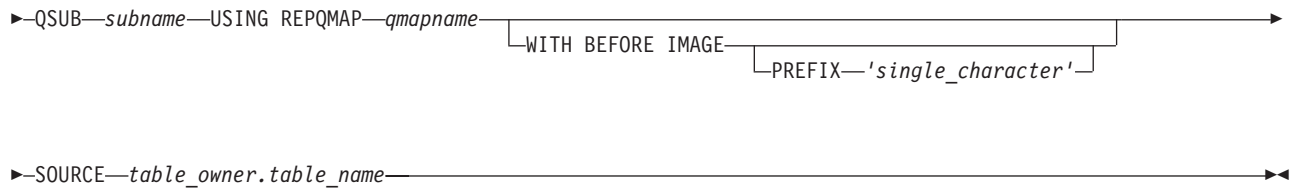
### Syntax

```

▶▶ ALTER ADD COLUMN USING SIGNAL—( colname , target_colname )

```





## Parameters

*colname*

Specifies one or more columns (separated by a comma) to add to the definition of the active Q subscription.

**TARGET** *target\_colname*

Specifies a name for the target column that is different from the source column name. The option to specify target column names is supported in the following cases:

### z/OS

The architecture level of the Q Capture or Q Apply control tables is 0907 or higher and the value of the COMPATIBILITY column in the IBMQREP\_CAPPARMS table is 0907 or higher. The architecture level is saved in the ARCH\_LEVEL column of the IBMQREP\_CAPPARMS or IBMQREP\_APPLYPARMS tables.

### Linux UNIX Windows

Source or target servers are on DB2 Version 9.7 Fix Pack 5 or higher and the value of the COMPATIBILITY column in the IBMQREP\_CAPPARMS table is 0907 or higher.

**QSUB** *subname*

Specifies the name of the Q subscription.

**WITH BEFORE IMAGE**

Specifies that the before-image value of each added column will be replicated.

**PREFIX** '*single\_character*'

Specifies the single-character prefix for each before-image column. If you do not specify a prefix, a default of X is used. If this prefix generates invalid names, other letters will be used beginning with Y until valid names are generated.

**USING REPQMAP** *qmapname*

Specifies the name of the replication queue map that is used by the Q subscription.

**SOURCE** *table\_owner.table\_name*

Specifies that the columns are added to all of the subscriptions and publications that subscribe to the source table.

## Usage notes

- The column needs to exist in the source table already and should not be part of any existing Q subscription.
- The Q subscription must be active.
- The column must be nullable or have a default value 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 attributes with SQL.

- A maximum of 20 column names can be inserted into one command.

### Example 1

To add the columns PHONE and ADDRESS to the EMPLOYEE0001 Q subscription:

```
ALTER ADD COLUMN USING SIGNAL (PHONE, ADDRESS) QSUB EMPLOYEE0001
USING REPMAP SAMPLE_ASN_TO_TARGETDB_ASN;
```

### Example 2

To add the PHONE, ADDRESS, and EMAIL columns to all Q subscriptions and publications that subscribe to the EMPLOYEE table.

```
ALTER ADD COLUMN USING SIGNAL (PHONE, ADDRESS, EMAIL) SOURCE DB2ADMIN.EMPLOYEE;
```

### Example 3

To add the PHONE column to the EMPLOYEE0001 Q subscription and specify that the column in the target table be named TRGPHONE:

```
ALTER ADD COLUMN USING SIGNAL (PHONE TARGET TRGPHONE) QSUB EMPLOYEE0001
USING REPMAP SAMPLE_ASN_TO_TARGETDB_ASN;
```

### Example 4

To add multiple columns to the EMPLOYEE0001 Q subscription and specify that the columns in the target table have different names:

```
ALTER ADD COLUMN USING SIGNAL (PHONE TARGET TRGPHONE, ID TARGET TRGID)
QSUB EMPLOYEE0001 USING REPMAP SAMPLE_ASN_TO_TARGETDB_ASN;
```

### Example 5

To add the SALARY column to the SRC10001 Q subscription, specify that the column in the target table be named TRGSALARY, and specify that a before-image version of the column also be added to the target table with a before-image prefix of Y:

```
ALTER ADD COLUMN USING SIGNAL (SALARY TARGET TRGSALARY) QSUB SRC10001
USING REPMAP REPLMAP1 WITH BEFORE IMAGE PREFIX Y;
```

---

## ALTER CAPTURE PARAMETERS command (Classic replication)

The capture operational parameters are stored in the table IBMQREP\_CAPPARMS table. Use the **ALTER CAPTURE PARAMETERS** command to update the IBMQREP\_CAPPARMS table when you replicate from a Classic source.

### Syntax

```
▶▶ALTER CAPTURE PARAMETERS—QMGR—qmgr—RESTARTQ—restartq—ADMINQ—adminq—▶▶
```

### Parameters

**QMGR** *qmgr*

Specifies the queue manager name.

**RESTARTQ** *restartq*

Specifies the name of the restart queue that is used by the publication service.

**ADMINQ** *adminq*

Specifies the name of the administration queue that is used by the publication service.

## Usage notes

- Issue this command before you define replication objects that interact with Classic data sources. Other commands that create and manipulate replication objects will not work properly if a row does not exist in the IBMQREP\_CAPPARMS table.

## Example

The following ALTER CAPTURE PARAMETERS command specifies the queue manager, restart queue, and administration queue for a Classic data source.

```
ASNCLP SESSION SET TO Q REPLICATION
SET SERVER CAPTURE CONFIG SERVER classic1
FILE asnservers.ini ID id1 PASSWORD passwd1;
ALTER CAPTURE PARAMETERS QMGR qmg1 RESTARTQ rq1 ADMINQ aq1;
```

---

## ALTER CONFIGURATION APPLY command

The **ALTER CONFIGURATION APPLY** command allows you to change the configuration of the Q Apply program after you have specified a target server and Q Apply schema.

## Syntax

```
▶▶—ALTER CONFIGURATION APPLY—SET CAPTURE SCHEMA—  
└──set—"name"—  
└──is null
```

## Parameters

**set "name"**

Specifies the new SQL Capture schema for the registrations of the CCD tables that Q apply maintains.

**is null**

Specifies that Q Apply does not maintain the registrations of its target CCD tables.

## Usage notes

- Use this command in order to configure a Q Apply program to manage a SQL Capture schema.

## Example

This example specifies that Q Apply uses the capture schema "ASN".

```
ASNCLP SESSION SET TO Q REPLICATION;
SET RUN SCRIPT NOW STOP ON SQL ERROR ON;
SET SERVER TARGET TO QAPPDB;
SET APPLY SCHEMA QAPP1;
ALTER CONFIGURATION APPLY SQL CAPTURE SCHEMA SET "ASN";
```

## 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 ]
```

```
▶ [ MANAGE TARGET CCD action ] [ USING OPTIONS
    [ other-opt-clause ]
    [ add-cols-clause ]
    [ add-period-clause ] ]
```

#### action:

```
| CREATE SQL REGISTRATION
| DROP SQL REGISTRATION
| ALTER SQL REGISTRATION FOR Q REPLICATION
```

#### other-opt-clause:

```
| [ SEARCH CONDITION "search_condition" ] [ ALL CHANGED ROWS
    [ N ]
    [ Y ] ]
```

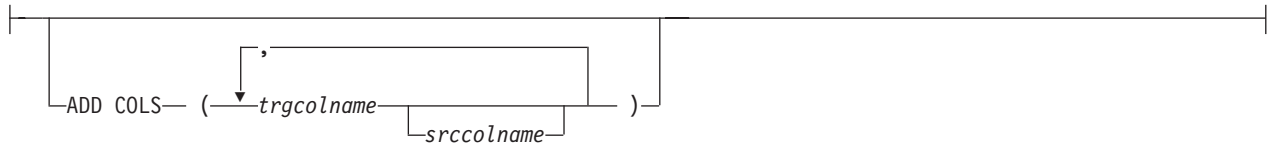
```
▶ [ HAS LOAD PHASE
    [ N ]
    [ I ]
    [ E ] ] [ SUPPRESS DELETES
    [ N ]
    [ Y ] ] [ REPLICATE ADD COLUMN
    [ N ]
    [ Y ] ]
```

```
▶ [ CHANGE CONDITION "change_condition" ] [ CONFLICT ACTION
    [ I ]
    [ F ]
    [ D ]
    [ S ]
    [ Q ] ]
```

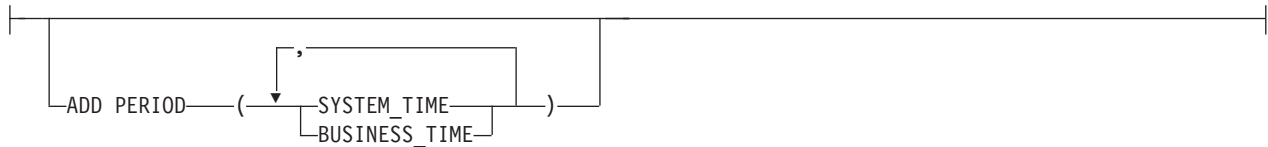
```
▶ [ ERROR ACTION
    [ S ]
    [ D ]
    [ Q ]
    [ B ] ] [ OKSQLSTATES "sqlstates" ]
```

```
▶ [ LOAD TYPE
    [ 0 ]
    [ 1 ]
    [ 2 ]
    [ 3 ]
    [ 4 ]
    [ 5 ] ] [ EXIST DATA
    [ REPLACE ]
    [ APPEND ] ]
```

### add-cols-clause:



### add-period-clause:



## 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.

action:

### **CREATE SQL REGISTRATION**

Registers the target CCD table for the Q subscription as a source for SQL replication.

### **DROP SQL REGISTRATION**

Deletes an existing registration for SQL replication. When you issue the **CREATE QSUB** command with this parameter, the ASNCLP program checks to make sure that all Q subscriptions that use this registration are inactive.

### **ALTER SQL REGISTRATION FOR Q REPLICATION**

Modifies an existing registration for SQL replication by updating the `CD_OWNER` field in the `IBMSNAP_REGISTER` table with the Q Apply schema and the `CD_TABLE` field with the name of the receive queue for the Q subscription. You can also use this action to change an existing SQL registration to a Q subscription that uses a different receive queue.

other-opt-clause:

### **SEARCH CONDITION** *"search\_condition"*

Specifies a search condition for filtering changes to replicate. You cannot use this parameter with Classic sources. 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 loads from the target. The load method depends on the LOAD TYPE keyword. This option is not valid for Q subscriptions that specify stored procedures as targets.
- E** Specifies a manual load. You can use your own load procedure or application to load the target table rather than using the Q Apply program. In this case, you use the **LOADDONE** command to indicate that the load is done.

**SUPPRESS DELETES**

Specifies whether to send rows that were deleted from the source table. This parameter is not valid for Classic sources.

- N** Send deleted rows.
- Y** Do not send deleted rows.

**REPLICATE ADD COLUMN**

Specifies whether new columns that are added to the source table should automatically be added to the Q subscription, and to the target table if they do not already exist. This function requires the Q Capture server to be at InfoSphere Replication Server for z/OS, 10.

**N (default)**

New source table columns are not automatically added to the Q subscription.

- Y** New source table columns are automatically added to the Q subscription.

**CHANGE CONDITION "change\_condition"**

Specifies a predicate that uses log record variables for filtering changes to replicate. You cannot use this parameter with Classic replication.

You can use the following log record variables:

\$OPERATION	The DML operation. Valid values are I (insert), U (update), and D (delete).
\$AUTHID	The authorization ID of a transaction.
\$AUTHTOKEN	<b>z/OS:</b> The authorization token (job name) of a transaction.
\$PLANNAME	<b>z/OS:</b> The plan name of a transaction.

For example, the following predicate specifies that Q Capture only replicate log records that were not committed by the user ASN:

```
"$AUTHID <> 'ASN'"
```

If a different predicate is specified by using the **SEARCH CONDITION** keyword, that predicate is combined with the **CHANGE CONDITION** predicate into a single

predicate by using the AND operator. For more details on the format for **CHANGE CONDITION**, see Log record variables to filter 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 the queue.

#### **ERROR ACTION**

Specifies what action to take if an error occurs.

- S** Stop Q Apply without applying the transaction.
- D** For a DB2 source, disable subscription and notify Q Capture. For a Classic source, disable subscription and notify the Classic capture components.
- Q** Stop reading from queue.
- B** When an error occurs, spill change messages for the Q subscription to a temporary spill queue until you use the **resumesub** parameter of the **MODIFY** or **asnqacmd** command to prompt Q Apply to begin applying the messages.

#### **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 method of loading the target table with data from the source.

**Note:** By default, for all of the following load types the load utilities are invoked with an option to delete all existing data in the target table before replacing it with data from the source (this is called the replace option). You can use the EXIST DATA APPEND keywords to specify that the chosen load utility is invoked with an option to append source data to the target table without deleting target table contents.

- 0** Choose the best type automatically. Not valid for Classic sources.
- 1** Use LOAD from CURSOR only. Specify this option if the source and target servers are on z/OS. Not valid for Classic sources or federated targets.

**Note:** If the Q Apply program is at Version 9.7 Fix Pack 4 or newer, you do not need to provide nickname information for this load option unless the Q subscription includes XML columns. Q Apply calls LOAD from CURSOR by specifying a cataloged DB2 alias for the source database instead of by using a nickname. You must include the DB2 alias in a password file that is created by the asnpwd utility.

- 2** Use the EXPORT and IMPORT utilities. Not valid for Classic or Oracle sources.
- 3** Use the EXPORT and LOAD utilities. Not valid for Classic or Oracle sources or for federated targets.
- 4** Select from a replication source and use the DB2 LOAD utility, or for Oracle targets use the SQL\*Loader utility.

**Oracle targets:** To use SQL\*Loader, you must create a password file by using the **asnpwd** command in the directory that is specified by the **apply\_path** parameter or the directory from which Q Apply is invoked with the following values for these keywords:

- **alias:** The Oracle `tnsnames.ora` entry that refers to the Oracle server (the same name that is used for the `NODE` option of the `CREATE SERVER` command for setting up federation).
- **id:** The remote user ID for connecting to Oracle.
- **password:** The password for connecting to Oracle.

The file must have the default name `asnpwd.aut`. Before starting the Q subscription, you should test connectivity with this command: `$> sqlplus id/password@alias`.

- 5 **Linux, UNIX, and Windows targets:** Select from a replication sources and use the DB2 IMPORT utility. The `replace` option is used by default. Use this load option when the source code page differs from the target code page. The DB2 IMPORT utility converts code pages when it is invoked with this option.

#### **ADD COLS** (*trgcolname srccolname*)

Specify to add one or more columns to the Q subscription before the replication programs begin processing the Q subscription. If *trgcolname* and *srccolname* are the same, only specify the *trgcolname*. You can also use the **ALTER ADD COLUMN** command to add columns to a Q subscription if the column does not already exist in the target table.

**Note:** Use the **ALTER ADD COLUMN** command if the Q subscription is active and is being processed by the replication programs

This parameter is not valid for Classic sources.

add-period-clause:

#### **ADD PERIOD**

Specifies that the source table is a temporal table on DB2 10 for z/OS or later and you want to include period columns in the Q subscription.

#### **SYSTEM\_TIME**

Specifies that you want to include the timestamp columns that are used with system-period temporal tables.

#### **BUSINESS\_TIME**

Specifies that you want to include the timestamp or date columns that are used with application-period temporal tables.

## **Example - Changing selected properties**

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 0
```

This example is valid only with DB2 sources.



## Example - Adding columns

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)
```

This example does not apply to Classic replication because with it you must replicate all columns. You cannot add columns.

## Example - Creating a registration for SQL replication

To alter a Q subscription that has a CCD target so that it can manage a new SQL registration by creating this registration:

```
ASNCLP SESSION SET TO Q REPLICATION;

SET RUN SCRIPT NOW STOP ON SQL ERROR ON;

SET SERVER CAPTURE TO QCAPDB;
SET SERVER TARGET TO QAPPDB;

SET CAPTURE SCHEMA SOURCE QCAP1;
SET APPLY SCHEMA QAPP1;

ALTER QSUB SUB1 REPLQMAP QCAPDB_QCAP1_TO_QAPPDB_QAPP1
MANAGE TARGET CCD CREATE SQL REGISTRATION;
```

## Example - Deleting a registration for SQL replication

To alter a Q subscription that has a CCD target by deleting the SQL registration of its target CCD:

```
ASNCLP SESSION SET TO Q REPLICATION;

SET RUN SCRIPT NOW STOP ON SQL ERROR ON;

SET SERVER CAPTURE TO QCAPDB;
SET SERVER TARGET TO QAPPDB;

SET CAPTURE SCHEMA SOURCE QCAP1;
SET APPLY SCHEMA QAPP1;

ALTER QSUB SUB1 REPLQMAP QCAPDB_QCAP1_TO_QAPPDB_QAPP1
MANAGE TARGET CCD DROP SQL REGISTRATION;
```

---

## ALTER REPLQMAP command

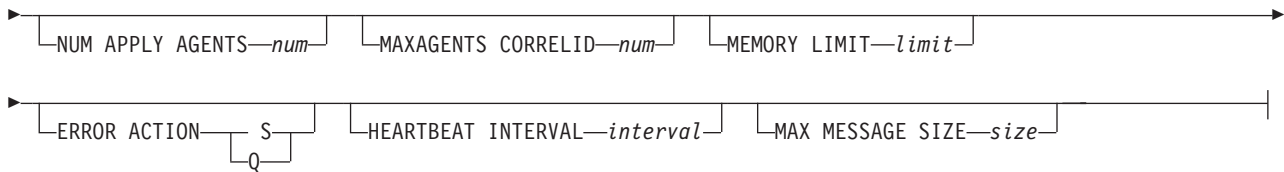
Use the **ALTER REPLQMAP** command to customize attributes for an existing replication queue map. This command applies to Q replication and Classic replication.

### Syntax

```
▶▶ ALTER REPLQMAP qmapname USING | options | ◀◀
```

#### options:

```
| [DESC—"description"] [ADMINQ—"admnqname"] [RECVQ—"recvqname"] [SENDQ—"sendqname"] |▶
```



## 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 or Classic capture components share a single queue manager with the Q Apply programs, they can share an administration queue.

**RECVQ** "*recvqname*"

Specifies the name of the receive queue that is used by the Q Apply program. See "Usage notes" on page 91 below.

**SENDQ** "*sendqname*"

Specifies the name of the send queue that is used by the Q Capture program or Classic capture components. See "Usage notes" on page 91 below.

**NUM APPLY AGENTS** *num*

Specifies the number of threads that are used to concurrently apply transactions from the specified receive queue.

**MAXAGENTS CORRELID** *num*

**z/OS** Specifies that number of threads that are used for concurrently applying transactions from the specified receive queue with the same *correlation ID*. The correlation ID identifies all transactions that were started from the same z/OS job on the Q Capture server.

The value for the **MAXAGENTS CORRELID** parameter cannot be greater than the value for the **NUM APPLY AGENTS** parameter. If **MAXAGENTS\_CORRELID** value is 1, the transactions will be applied one at a time. If the value is greater than one, for example 4, four agents will apply transactions with the same correlation ID in parallel. If the value is 0, transactions are applied in parallel by using the total number of threads specified by the **NUM APPLY AGENTS** parameter.

**MEMORY LIMIT** *limit*

Specifies the maximum number of megabytes that are used per receive queue to buffer incoming transactions.

**ERROR ACTION**

The action that the Q Capture program takes 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.

**S** The Q Capture program or the capture components stop when they detect an error on this queue.

**Q** The Q Capture program stops putting messages on any send queues that are in error and continues putting messages on other send queues. This value is not supported for Classic replication.

**HEARTBEAT INTERVAL** *interval*

Specifies the interval (in seconds) between heartbeat messages that are sent by the Q Capture program or Classic capture components 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 that is used for sending messages over the send queue. The size of the buffer must not be larger than the maximum message length (MAXMSGL) that is defined for the send queue.

**Usage notes**

You can only change the name of the send queue or receive queue if the queue map is not being used by any Q subscriptions yet. If the queue map is part of a Q subscription (active or inactive), you must take manual steps to change these queue names. See [com.ibm.swg.im.iis.repl.qrepl.doc/topics/iyrqrqmtchgqname.dita](#) for details.

**Example 1**

The following command alters the SAMPLE\_ASN1\_TO\_TARGETDB\_ASN1 replication queue map, sets the threads to 4, and invalidates all of the Q subscriptions that use the send queue for this replication queue map if an error occurs.

```
ASNCLP SESSION SET TO Q REPLICATION;
SET SERVER CAPTURE TO SAMPLE;
SET CAPTURE SCHEMA ASN1;
SET SERVER TARGET TO TARGETDB
SET APPLY SCHEMA ASN1;
ALTER REPLQMAP SAMPLE_ASN1_TO_TARGETDB_ASN1 USING NUM APPLY AGENTS 4
      ERROR ACTION I;
```

**Example 2**

The following command alters the CLASSIC\_ASN1\_TO\_TARGETDB\_ASN1 replication queue map, sets the threads to 4, sets the maximum memory limit to 10 megabytes, stops the Classic capture components if an error occurs, sets the heartbeat interval to 4, and sets the maximum buffer size to 5 kilobytes.

```
ASNCLP SESSION SET TO Q REPLICATION;
SET OUTPUT TARGET SCRIPT "replapp.sql";
SET LOG "qmap.err";
SET SERVER CAPTURE TO CONFIG SERVER server1 FILE "asnservers.ini"
      ID username PASSWORD "passw1rd";
SET SERVER TARGET TO DB TARGETDB;
SET APPLY SCHEMA ASN1;
SET RUN SCRIPT NOW STOP ON SQL ERROR ON;
ALTER REPLQMAP CLASSIC_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**

Use the **ASNCLP SESSION SET TO** command to establish an ASNCLP session for Q replication to either relational or Classic data sources.

**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. Use this parameter when you are connecting to either relational or Classic sources.

### 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
```

---

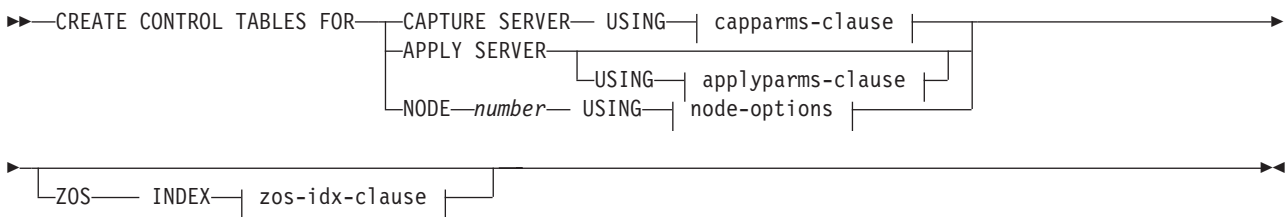
## CREATE CONTROL TABLES FOR command

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.

In Classic replication, the control tables for the Classic capture components are creating by using the Classic Data Architect.

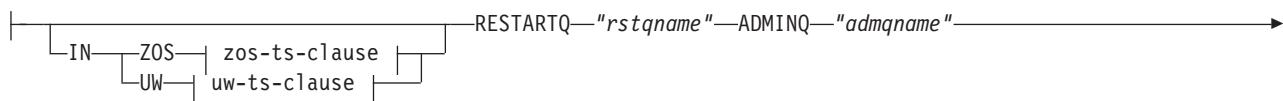
### Syntax

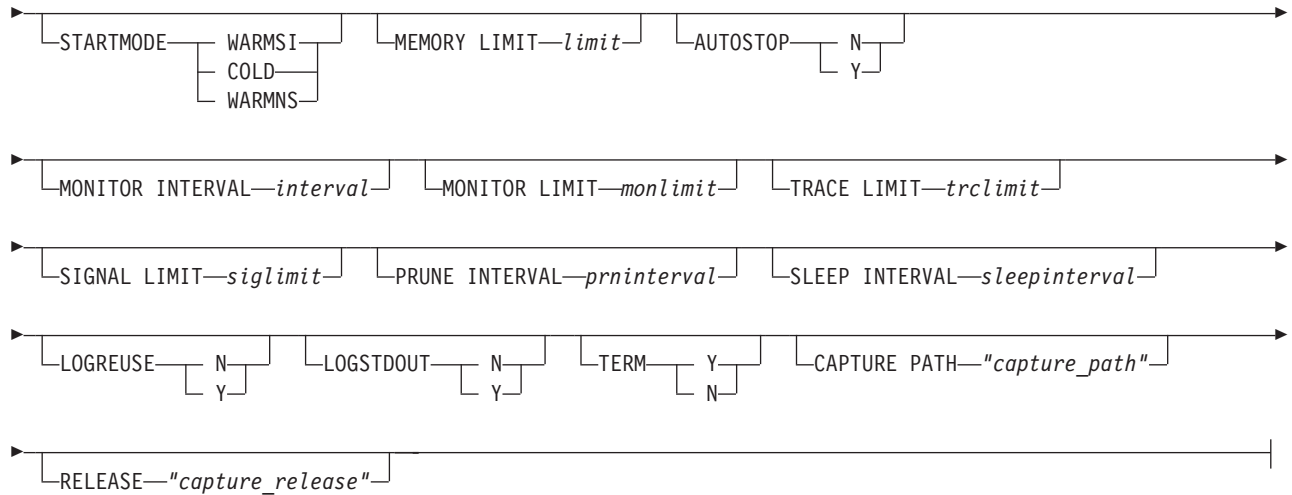


#### node-options:

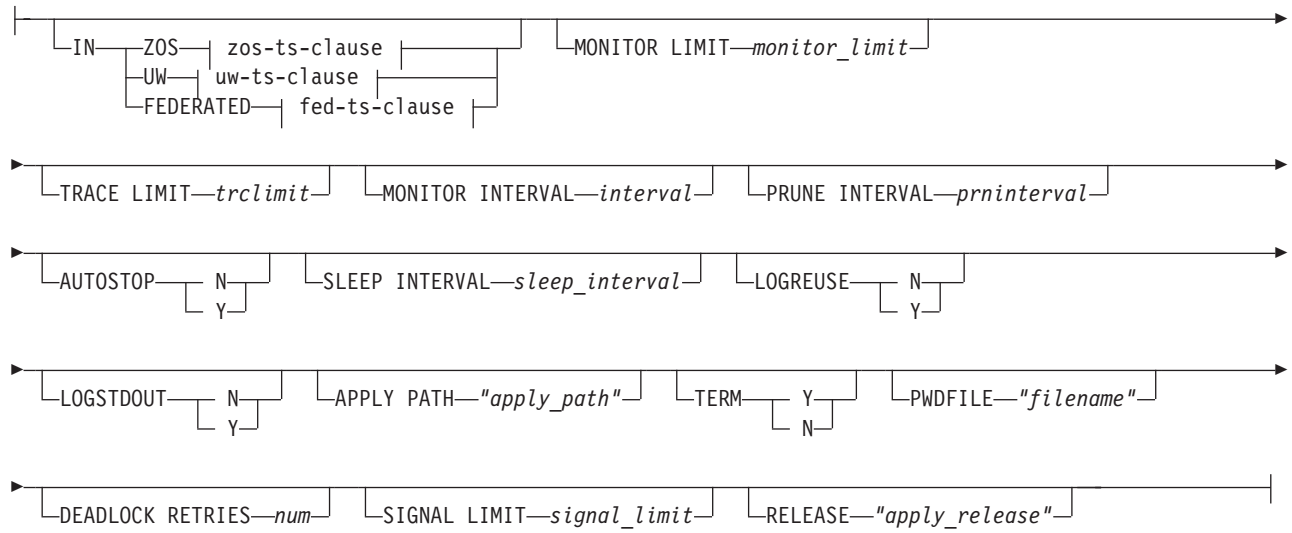


#### capparms-clause:

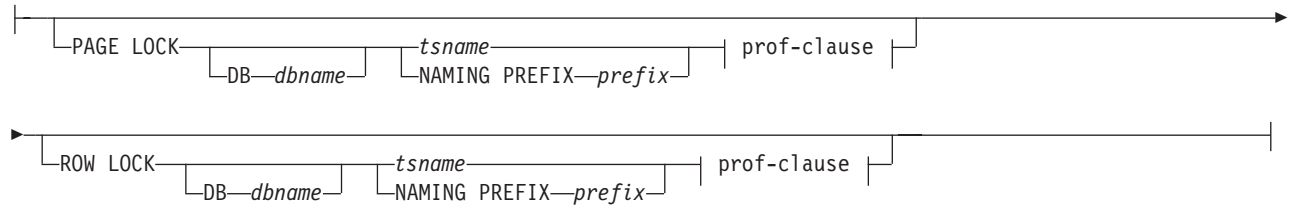




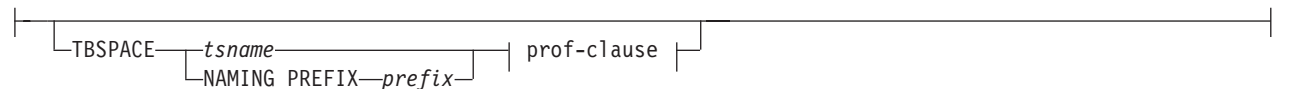
**applyparms-clause:**



**zos-ts-clause:**



**uw-ts-clause:**



### fed-ts-clause:

```
|-----RMT SCHEMA name-----|  
|-----TBSpace tname-----|
```

### prof-clause:

```
|-----CREATE-----|  
|-----USING PROFILE pname-----|
```

### zos-idx-clause:

```
|-----CREATE-----|  
|-----USING PROFILE pname-----|
```

## Parameters

### CAPTURE SERVER

Specify to create Q Capture control tables.

### APPLY SERVER

Specify to create Q Apply control tables.

### NODE

Specify to generate a script for creating both Q Capture and Q Apply control tables with the same schema on one server in a multidirectional replication configuration.

**Note:** Use this option only in conjunction with the SET BIDI NODE command for specifying the servers that are involved in multidirectional replication.

### CAPPARMS

Specify to set options for the Q Capture control tables.

### APPARMS

Specify to set options for the Q Apply control tables.

capparms-clause:

### ZOS

Specifies a z/OS system on which to create Q Capture control tables.

**UW** Specifies a Linux, UNIX, or Windows system on which to create Q Capture control tables.

### RESTARTQ *"rstqname"*

Specifies the restart queue that the Q Capture program uses.

### ADMINQ *"admname"*

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 is quiesced or stops. This value is the default.

**N** The Q Capture program continues running if DB2 is quiesced or stops.

**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.

**Linux UNIX Windows RELEASE** *"capture\_release"*

Specifies the release level of the control tables that you want to create. Allowed values are 9.7, 9.5, and 9.1. This parameter is for Linux, UNIX, and Windows only. Enclose value in double quotation marks ("). Specifying the release level enables newer replication and publishing function on an older DB2.

appparms-clause:

**ZOS**

Specifies a z/OS system in which Q Apply control tables are created.

**UW** Specifies a Linux, UNIX, or Windows system in which Q Apply control tables are created.

**FEDERATED**

Specifies a federated target, on which Q Apply control tables are created in an Oracle, Sybase, Informix, Microsoft SQL Server, or Teradata database, and nicknames are created for these control tables in the Q Apply server. Some control tables are created in the Q Apply server.

**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 is quiesced or stops.

N The Q Apply program continues running if DB2 is quiesced or stops.

**PWDFILE** *"filename"*

Specifies the name of the password file.

**DEADLOCK RETRIES** *num*

Specifies the number of retries for SQL deadlock errors.

**Linux UNIX Windows RELEASE** *"apply\_release"*

Specifies the release level of the control tables that you want to create. Allowed values are 9.7, 9.5, and 9.1. This parameter is for Linux, UNIX, and Windows only. Enclose value in double quotation marks ("). Specifying the release level enables newer replication and publishing function on an older DB2.

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. This parameter is not applicable for Teradata targets.

**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
```

### Example 4

To create Version 9.7 Q Apply control tables on a DB2 Version 9.1 database:

```
CREATE CONTROL TABLES FOR APPLY SERVER USING RELEASE "9.7"
```

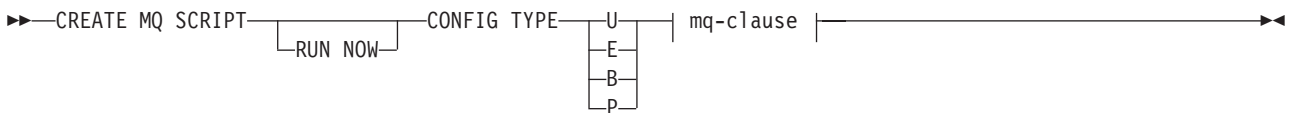
---

## CREATE MQ SCRIPT command

Use the **CREATE MQ SCRIPT** command to generate scripts for creating all of the WebSphere MQ objects that are needed for Q Replication and Event Publishing.

When you create control tables and queue maps, you can use the MQDEFAULTS keyword in these commands and the ASNCLP program will automatically use the default objects that are generated by CREATE MQ SCRIPT, bypassing the need to specify individual queue managers and queues.

### Syntax



### mq-clause:



## options:

|—MQHOST—*hostname*—| |—MQPORT—*port\_number*—| |—QMANAGER—*queue\_manager*—| |—QNAME\_QUAL—*qualifier*—|

## Parameters

### **RUN NOW**

Specifies that you want the ASNCLP program to run the generated WebSphere MQ script after it is created. The queue manager and ASNCLP program must be on the same system for you to use this option.

### **CONFIG TYPE**

Specifies the type of replication:

- U** Unidirectional
- E** Event publishing
- B** Bidirectional
- P** Peer-to-peer

mq-clause

### **MQSERVER**

A number that identifies the Q Capture server, Q Apply server, or both for multidirectional replication. The numbers differ depending on the configuration type:

#### **Unidirectional**

Use 1 to represent the Q Capture server and 2 to represent the Q Apply server. Both numbers are required.

#### **Event publishing**

Use 1 to represent the Q Capture server.

#### **Bidirectional**

Use 1 to represent one server and its paired Q Capture and Q Apply, and the number 2 to represent the other server. Both numbers are required.

#### **Peer-to-peer**

Use 1, 2, 3, and so on, depending on the number of servers in the peer-to-peer environment. At least two server numbers are required.

### **NAME**

The subsystem name or database alias of the Q Capture server, Q Apply server, or the combined Q Capture-Q Apply server for multidirectional replication.

options

### **MQHOST**

The hostname or IP address of the system that contains the queue manager that will create the WebSphere MQ objects.

### **MQPORT**

The port number that the channel listener monitors for incoming requests. If this keyword is not specified, the ASNCLP program uses the default WebSphere MQ port number 1414.

## QMANAGER

The queue manager that will be created, and that will be used to create other WebSphere MQ objects. If this keyword is not specified, the value that was specified for the **NAME** keyword is used to name the queue manager.

## QNAME\_QUAL

A qualifier that is used for the generated queue names. The default is ASN, which is the default Q Capture or Q Apply schema. This qualifier can help identify queues at the Q Capture system or Q Apply system.

## Usage notes

- **Linux UNIX Windows** The default file name for the generated script is `repl.server_name.mq`, where `server_name` is the server alias that was specified in the CREATE MQ SCRIPT command. The scripts are executable files in either the `.bat` or `.exe` format depending on whether the ASNCLP program runs on Windows or Linux-UNIX.

- **z/OS** If the ASNCLP program is running natively on z/OS, the output DD name for the generated script is OUTMQCAP, OUTMQTRG, and OUTMQX. The following lines must be included in the JCL:

```
//OUTMQCAP DD DSN=&SYSUID..ASNCLP.OUTNODE1,DISP=(NEW,CATLG,DELETE),  
//          UNIT=SYSDA,SPACE=(TRK,(30,10))  
//OUTMQTRG DD DSN=&SYSUID..ASNCLP.OUTNODE1,DISP=(NEW,CATLG,DELETE),  
//          UNIT=SYSDA,SPACE=(TRK,(30,10))
```

The generated script will be wrapped to 80 characters per line. Comments are included with changes that need to be made for z/OS.

- You can specify the CREATE MQ SCRIPT command in the same input file as other ASNCLP commands, but this command does not use the server and schema information from any previous SET commands.
- If the Q Capture and Q Apply servers are on the same system, only one script file is generated that contains all the WebSphere MQ commands.

## Example 1

To generate a script that creates WebSphere MQ objects for event publishing:

```
CREATE MQ SCRIPT CONFIG TYPE E  
MQSERVER 1 NAME SOURCEDB MQHOST "9.30.54.118" MQPORT "1414";
```

## Example 2

To generate a script that creates WebSphere MQ objects for a unidirectional replication configuration where the Q Capture and Q Apply servers are on the same system and share a local queue manager:

```
CREATE MQ SCRIPT CONFIG TYPE U  
MQSERVER 1 NAME SOURCEDB MQHOST "9.30.54.118",  
MQSERVER 2 NAME TARGETDB MQHOST "9.30.54.118";
```

## Example 3

To generate a script that creates WebSphere MQ objects for a unidirectional replication configuration where the source and target servers are remote with different queue managers (no MQPORT keywords are specified so the default ports of 1414 are used at each system):

```
CREATE MQ SCRIPT CONFIG TYPE U  
MQSERVER 1 NAME SOURCEDB MQHOST "9.30.54.118",  
MQSERVER 2 NAME TARGETDB MQHOST "9.30.54.119";
```

## Example 4

To generate a script that creates WebSphere MQ objects for a bidirectional replication configuration where the primary and standby servers are remote with different queue managers:

```
CREATE MQ SCRIPT CONFIG TYPE B
MQSERVER 1 NAME DB1 MQHOST "9.30.54.118",
MQSERVER 2 NAME DB2 MQHOST "9.30.54.119";
```

## Example 5

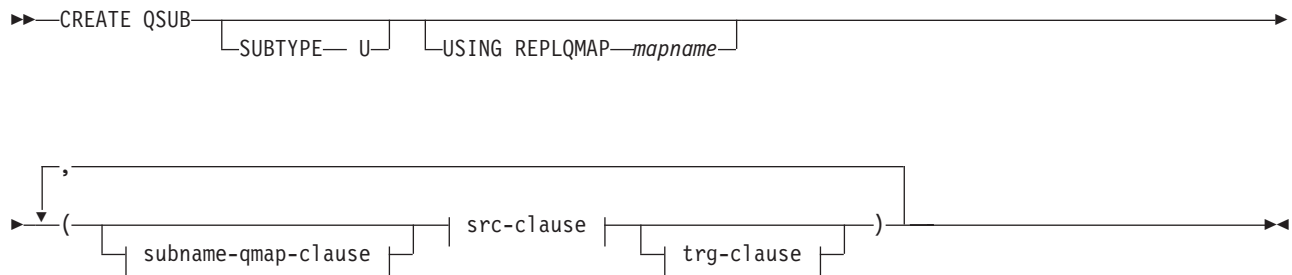
```
CREATE MQ SCRIPT CONFIG TYPE P
MQSERVER 1 NAME DB1 MQHOST "9.30.54.117",
MQSERVER 2 NAME DB2 MQHOST "9.30.54.118",
MQSERVER 3 NAME DB3 MQHOST "9.30.54.119";
```

---

## CREATE QSUB command (unidirectional replication)

Use the **CREATE QSUB** command to create a Q subscription that maps a source table to a target table. For Classic replication, a Q subscription maps a source table or view in the Classic metadata catalog to a target table.

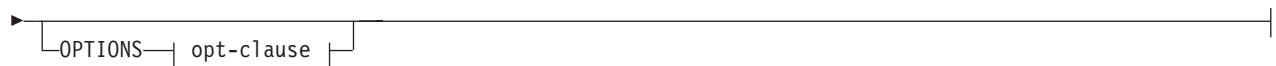
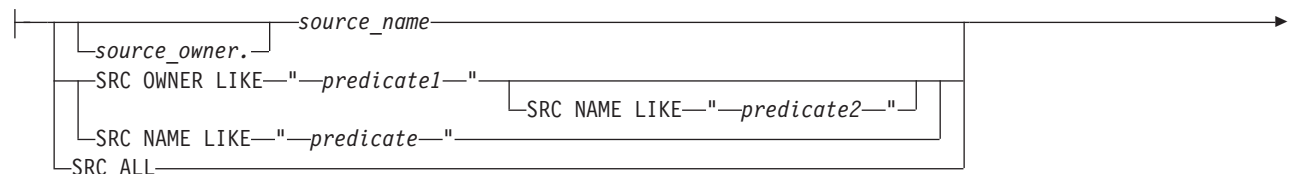
### Syntax



#### subname-qmap-clause:

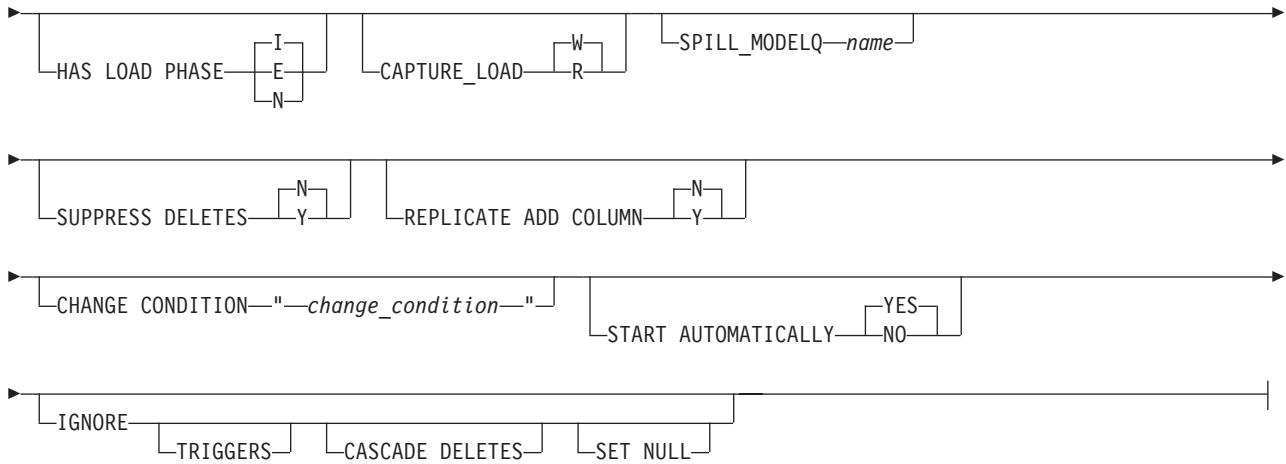


#### src-clause:

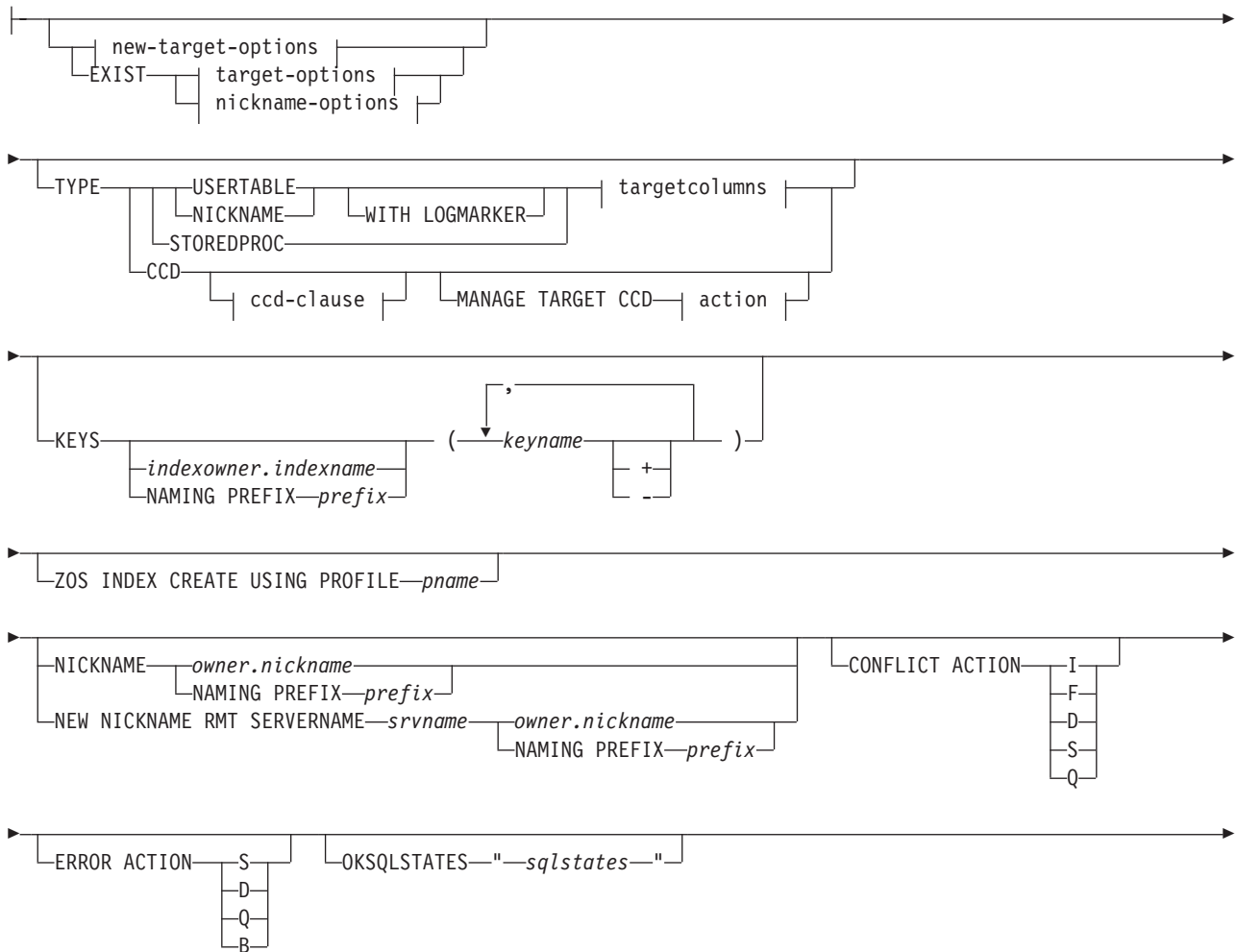


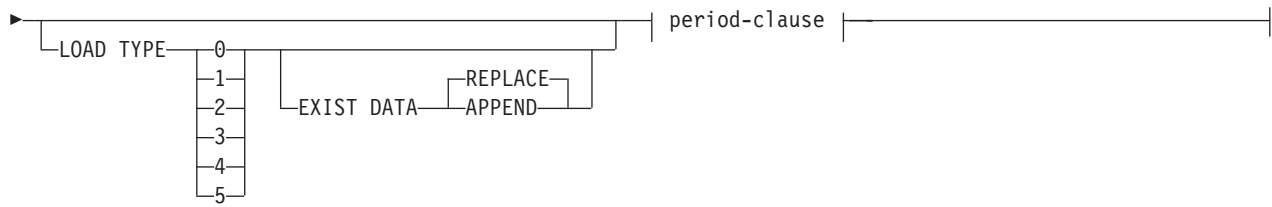
#### opt-clause:



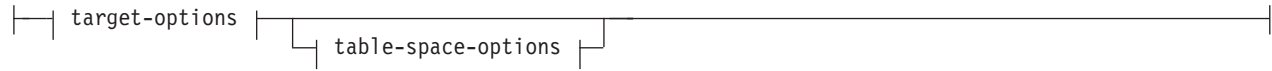


**trg-clause:**

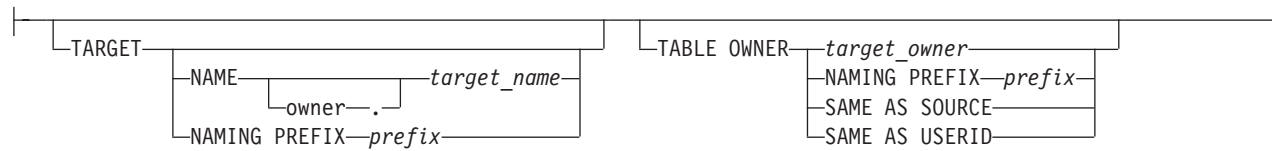




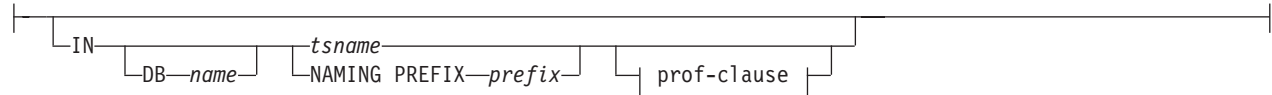
**new-target-options:**



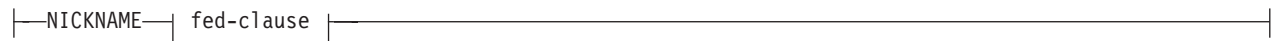
**target-options:**



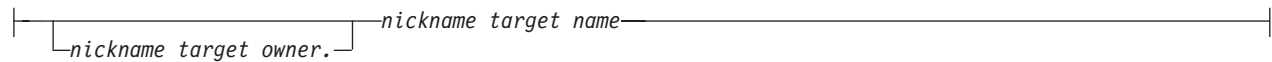
**table-space-options:**



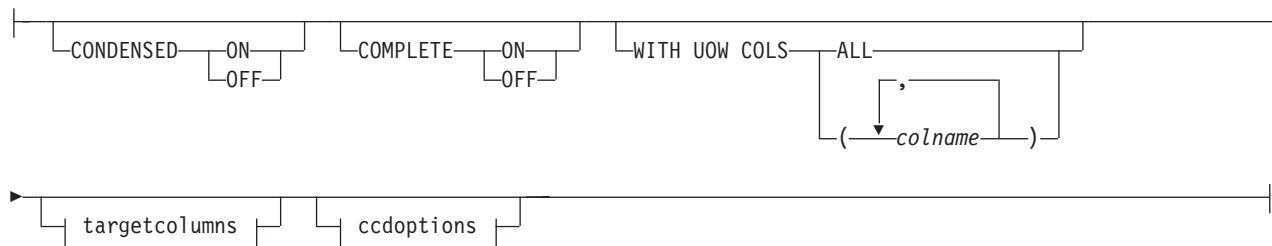
**nickname-options:**



**fed-clause:**



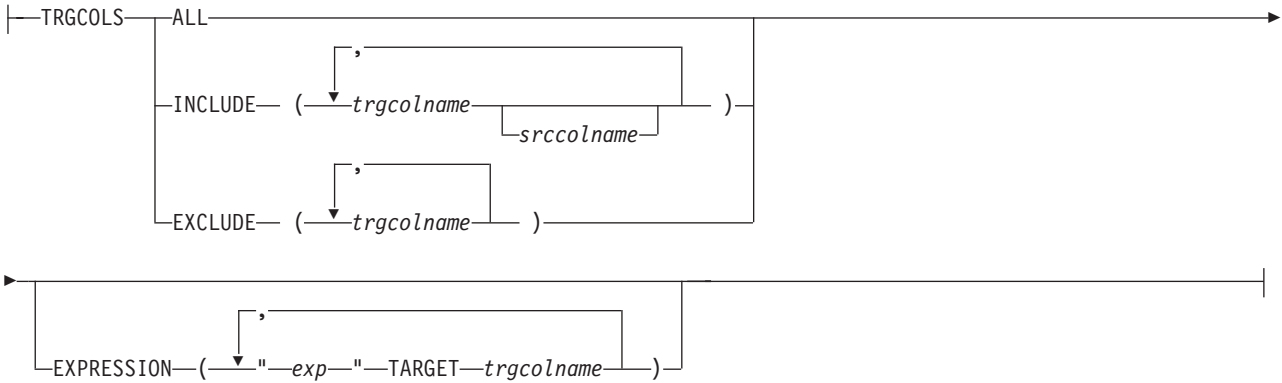
**ccd-clause:**



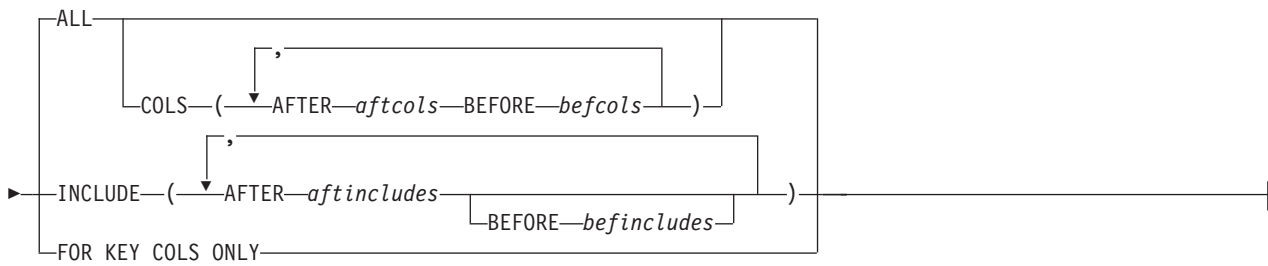
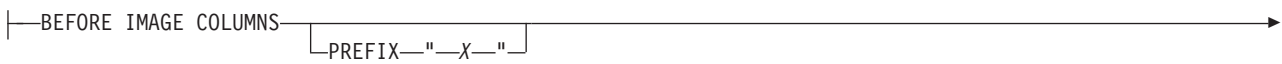
**prof-clause:**



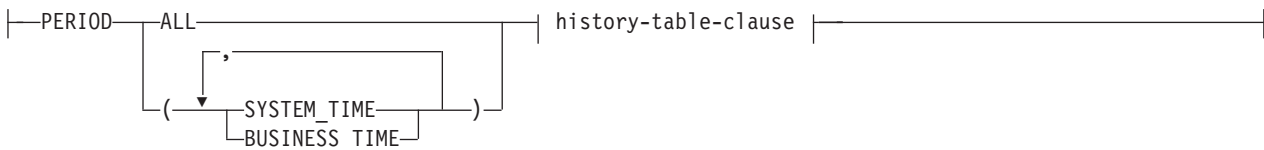
**targetcolumns:**



**ccdoptions:**



**period-clause:**



**history-table-clause:**

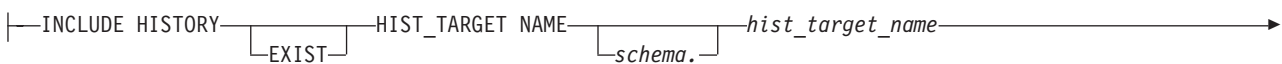




table-space-options

**action:**

CREATE SQL REGISTRATION  
ALTER SQL REGISTRATION FOR Q REPLICATION

## 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-qmap-clause

### **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 that exist on the source server. For DB2 sources, this excludes catalog views.

opt-clause:

**SEARCH CONDITION** "*search\_condition*"

Specifies a search condition for filtering changes to replicate or publish. You cannot use this parameter with Classic replication. The change is not sent if the predicate is false. "*search\_condition*" is an annotated select WHERE clause that must contain 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 (default)**

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.

**I (default)**

Specifies an automatic load. The Q Apply program loads the target. The load method depends on the LOAD TYPE keyword. This parameter 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. In this case, you use the **LOADDONE** command to indicate that the load is done.

**N** No load phase at the target.

**CAPTURE\_LOAD**

Specifies the action that the Q Capture program takes when the recovery log shows that a load operation that uses the DB2 LOAD utility occurred at the source table.

**W (default)**

Q Capture issues a warning message after the load completes.

**R** Q Capture stops and starts the Q subscription for the source table, prompting a load of the target table if one is specified for the Q subscription.

**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. This parameter is not valid for Classic replication.

**N (default)**

Send deleted rows.

**Y** Do not send deleted rows.

**REPLICATE ADD COLUMN**

Specifies whether new columns that are added to the source table should automatically be added to the Q subscription, and to the target table if they do

not already exist. This function requires the Q Capture server to be at InfoSphere Replication Server for z/OS, 10.1.

**N (default)**

New source table columns are not automatically added to the Q subscription.

**Y** New source table columns are automatically added to the Q subscription.

**CHANGE CONDITION "change\_condition"**

Specifies a predicate that uses log record variables for filtering changes to replicate. You cannot use this parameter with Classic replication.

You can use the following log record variables:

\$OPERATION	The DML operation. Valid values are I (insert), U (update), and D (delete).
\$AUTHID	The authorization ID of a transaction.
\$AUTHTOKEN	<b>z/OS:</b> The authorization token (job name) of a transaction.
\$PLANNAME	<b>z/OS:</b> The plan name of a transaction.

For example, the following predicate specifies that Q Capture only replicate log records that were not committed by the user ASN:

```
"$AUTHID <> 'ASN'"
```

If a different predicate is specified by using the **SEARCH CONDITION** keyword, that predicate is combined with the **CHANGE CONDITION** predicate into a single predicate by using the AND operator. For more details on the format for **CHANGE CONDITION**, see Log record variables to filter rows.

**START AUTOMATICALLY**

Specifies how to start the Q subscription, which is represented by the State column in the IBMQREP\_SUBS table. The State column controls whether the subscription is automatically started after starting or reinitializing the Q Capture program (subscription state N), or that the subscription must be started manually by inserting a command in the IBMQREP\_SIGNAL table (subscription state I).

**YES**

The Q subscription is started automatically (subscription state value of N). This is the default.

**NO**

The Q subscription must be started manually (subscription state value of I).

**IGNORE TRIGGERS**

Specifies that any rows that are generated by AFTER triggers at the source database will not be replicated. Use this option to avoid duplicate rows when matching triggers are already being used at the target table. If you use this option for the Q subscription, triggered changes will be ignored even if the Q Capture instance-level **igntrig** parameter is set to N.

**IGNORE CASCADE DELETES**

Specifies that when rows are deleted from child tables because of the ON DELETE CASCADE rule, the DELETE operation is not replicated. Use this option to avoid duplicate DELETE operations when ON DELETE CASCADE is already being used at the target database. If you use this option for the Q

subscription, cascaded DELETE operations will be ignored even if the Q Capture instance-level **igncasdel** parameter is set to N.

#### **IGNORE SET NULL**

Specifies that when the foreign key in a child table is set to NULL because of the ON DELETE SET NULL rule, the UPDATE operation is not replicated. Use this option to avoid duplicate UPDATE operations when ON DELETE SET NULL is already being used at the target database. If you use this option for the Q subscription, ON DELETE SET NULL operations will be ignored even if the Q Capture instance-level **ignsetnull** parameter is set to N.

**z/OS** This option is not supported on z/OS. The UPDATE operations resulting from ON DELETE SET NULL are still replicated if you specify this option on z/OS.

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*.

#### **TYPE**

##### **USERTABLE**

Specifies a table as the target.

##### **NICKNAME**

Specifies a nickname as the target.

##### **WITH LOGMARKER**

Use these keywords with the USERTABLE or NICKNAME keywords to specify a point-in-time target table or nickname. The target table or nickname must contain the column `IBMSNAP_LOGMARKER` (TIMESTAMP; nullable with default of NULL). If the ASNCLP creates the target table or nickname, this column is included. The WITH LOGMARKER keywords are only supported when the Q Apply program is at Version 9.7 Fix Pack 4 or later on Linux, UNIX, and Windows or Version 10.1 on z/OS (ARCH\_LEVEL 100Z) with the PTF that corresponds to Fix Pack 4.

**Note:** You cannot use the WITH LOGMARKER keywords if the source table has an `IBMSNAP_LOGMARKER` column. To create a three-tier configuration in which each table contains the `IBMSNAP_LOGMARKER` column, use the WITH LOGMARKER keywords when you create the Q subscription from Tier 1 to Tier 2. For the Q subscription from Tier 2 to Tier 3, use a regular column mapping to map the `IBMSNAP_LOGMARKER` column at Tier 2 to the matching column at Tier 3. This method ensures that the timestamp of when the row was changed at the source table at Tier 1 will be correctly propagated from Tier 2 to Tier 3.

**STOREDPROC**

Specifies a stored procedure as the target.

**CCD**

Specifies a consistent-change data (CCD) table as the target.

**Note:** You cannot use the TYPE CCD keywords if the source table has the IBMSNAP\_COMMITSEQ, IBMSNAP\_INTENTSEQ, IBMSNAP\_LOGMARKER, or IBMSNAP\_OPERATION columns that are used in CCD tables. To create a three-tier configuration in which each table contains these columns, use the TYPE CCD keywords when you create the Q subscription from Tier 1 to Tier 2. For the Q subscription from Tier 2 to Tier 3, use a regular column mapping to map the IBMSNAP\_% columns at Tier 2 to the matching columns at Tier 3. This method ensures that the values from the Tier 1 source recovery log that are used to populate the CCD table at Tier 2 will be correctly propagated to Tier 3.

**CREATE SQL REGISTRATION**

Registers the target CCD table for the Q subscription as a source for SQL replication.

**ALTER SQL REGISTRATION FOR Q REPLICATION**

Modifies an existing registration for SQL replication by updating the CD\_OWNER field in the IBMSNAP\_REGISTER table with the Q Apply schema and the CD\_TABLE field with the name of the receive queue for the Q subscription. You can also use this action to change an existing SQL registration to a Q subscription that uses a different receive queue.

**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. Use this keyword only to specify a nickname for loading. The nickname that is specified with this keyword is not used to reference a target table in a non-DB2 relational database.

**For Version 9.7 Fix Pack 4 or newer:** If the Q Apply program is at Version 9.7 Fix Pack 4 or newer, and the source table does not include XML columns, you do not need to specify the NICKNAME keyword for loading the target with LOAD from CURSOR. In this case, the Q Apply program invokes LOAD from CURSOR by using a cataloged DB2 alias rather than a nickname.

*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 the Q subscription and notify the Q Capture program or the Classic capture components.

**Q** Stop reading from the receive queue.

**B** When an error occurs, spill change messages for the Q subscription to a temporary spill queue until you use the **resumesub** parameter of the **MODIFY** or **asnqacmd** command to prompt Q Apply to begin applying the messages.

**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 method of loading the target table with data from the source.

**Note:** By default, for all of the following load types the load utilities are invoked with an option to delete all existing data in the target table before replacing it with data from the source (this is called the replace option). You can use the EXIST DATA APPEND keywords to specify that the chosen load utility is invoked with an option to append source data to the target table without deleting target table contents.

**0** Choose the best type automatically. Not valid for Classic sources.

**1** Use LOAD from CURSOR only. Specify this option if the source and target servers are on z/OS. Not valid for Classic sources or federated targets.

**Note:** If the Q Apply program is at Version 9.7 Fix Pack 4 or newer, you do not need to provide nickname information for this load option unless the Q subscription includes XML columns. Q Apply calls LOAD from

CURSOR by specifying a cataloged DB2 alias for the source database instead of by using a nickname. You must include the DB2 alias in a password file that is created by the `asnpwd` utility.

- 2 Use the EXPORT and IMPORT utilities. Not valid for Classic or Oracle sources.
- 3 Use the EXPORT and LOAD utilities. Not valid for Classic or Oracle sources or for federated targets.
- 4 Select from a replication source and use the DB2 LOAD utility, or for Oracle targets use the SQL\*Loader utility.

**Oracle targets:** To use SQL\*Loader, you must create a password file by using the `asnpwd` command in the directory that is specified by the `apply_path` parameter or the directory from which Q Apply is invoked with the following values for these keywords:

- **alias:** The Oracle `tnsnames.ora` entry that refers to the Oracle server (the same name that is used for the NODE option of the CREATE SERVER command for setting up federation).
- **id:** The remote user ID for connecting to Oracle.
- **password:** The password for connecting to Oracle.

The file must have the default name `asnpwd.aut`. Before starting the Q subscription, you should test connectivity with this command: `$> sqlplus id/password@alias`.

- 5 **Linux, UNIX, and Windows targets:** Select from a replication source and use the DB2 IMPORT utility. The `replace` option is used by default. Use this load option when the source code page differs from the target code page. The DB2 IMPORT utility converts code pages when it is invoked with this option.

#### EXIST DATA

Specifies whether existing data in the target table is replaced or appended to during the loading process:

##### REPLACE (default)

The load utility is invoked with the option to delete all data in the target table before replacing it with data from the source.

##### APPEND

The load utility is invoked with the option to append source data to the target table without deleting target table contents.

#### TARGET

Specifies options for the target table owner and name.

**NAME** *target\_owner.target\_name*

Specifies the target table 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 owner. 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.

**FEDERATED**

Specifies that the target table is in a non-DB2 relational database and you want replication to create a new nickname that references the target table. Use the fed-clause to specify a name and owner for the new nickname.

**Note:** Do not use this keyword if you are using an existing nickname to reference the target table. Instead, use the nickname-options clause.

**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). This parameter is not applicable for Teradata targets.

**NAMING PREFIX** *prefix*

Specifies the prefix to use to name the table space.

## nickname-options

**NICKNAME**

Specifies an existing nickname that references a target table in a non-DB2 relational target database. Use the nickname-options clause only to specify existing nicknames. Do not use both the nickname-options clause and the FEDERATED keyword; they are mutually exclusive. Use the FEDERATED keyword when you want replication to create the nickname.



If you use an existing nickname, make sure that the nickname data types are compatible with the source table according to Q Replication requirements. See Nickname data types required for federated Q Replication for more details.

**Note:** Do not use this NICKNAME keyword to specify a nickname for loading the target table with the LOAD from CURSOR utility.

fed-clause

*nickname target owner*

Specifies the owner for a new nickname that replication creates to reference a federated target, or the owner of an existing nickname.

*nickname target name*

Specifies the name of a new nickname that replication creates to reference a federated target, or the owner of an existing nickname.

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.

targetcolumns

#### **TRGCOLS**

##### **ALL**

Specify to replicate all columns from the source table.

##### **INCLUDE**

Specifies the replicated columns in the target table. If the target table does not exist, specifies the column definitions in the target table.

*trgcolname*

Specify to define a target table column 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 define a target table column that uses the properties of the specified source column, but when the target column has a different name than the source column. In the following example, the target table defines two columns *target\_one* and *target\_two* based on the properties of corresponding columns *one* and *two* in the source table:

```
CREATE QSUB SUBTYPE U USING REPLQMAP replqmap9
(SUBNAME sub9 dpropr64.srctable
EXIST TARGET NAME dpropr64.trgtable
TRGCOLS INCLUDE (target_one one, target_two two))
```

#### **EXCLUDE** (*trgcolnames*)

This keyword behaves differently depending on whether the target table exists or you are creating a new target table with the Q subscription. In the examples, the source table columns are C1, C2, and C3:

##### **New target table**

Specify to exclude the source column from the target table definition and the Q subscription. For example, in the following command column C3 is excluded from the new target table and the Q subscription:

```
CREATE QSUB USING REPLQMAP replqmap10
(SUBNAME sub10 dpropr64.srctable TARGET NAME
dpropr64.tgtable TRGCOLS EXCLUDE(C3));
```

You cannot use this keyword when you are creating a new target table with a Classic replication source.

##### **Existing target table**

Specify to exclude target columns from the Q subscription. This keyword can be used only when the source and target tables have the same column names. The target table already exists and has columns C1, C2, and C4. Column C4 will be excluded from the Q subscription:

```
CREATE QSUB USING REPLQMAP replqmap10
(SUBNAME sub10 dpropr64.srctable EXIST TARGET NAME
dpropr64.tgtable TRGCOLS EXCLUDE(C4));
```

#### **EXPRESSION** *exp*

Specifies a DB2-supported expression to which the target column is mapped.

#### **TARGET** *trgcolname*

Specifies the name of the target column that will be populated by the expression.

#### **Note about TRGCOLS and EXPRESSION usage**

The syntax for using the TRGCOLS and EXPRESSION keywords in the same command differs depending on whether the target table exists or you are creating a new target table with the Q subscription. Follow these guidelines when you use

TRGCOLS ALL and EXPRESSION, TRGCOLS INCLUDE and EXPRESSION, and TRGCOLS EXCLUDE and EXPRESSION. In the examples, the source table has the columns C1, C2, and C3:

### **New target table**

These notes apply to the use of TRGCOLS and EXPRESSION when you are creating a new target table:

#### **TRGCOLS ALL and EXPRESSION**

The new target table and the Q subscription will include all columns from the source table and the columns that are specified in the EXPRESSION clause. In this example, the target table will be created with four columns: C1, C2, C3, and EXPC3:

```
CREATE QSUB USING REPLQMAP replqmap10
(SUBNAME sub10 dpropr64.srctable TARGET NAME
dpropr64.tghtable TRGCOLS ALL
EXPRESSION ("CHAR(:C3)" TARGET EXPC3));
```

#### **TRGCOLS INCLUDE and EXPRESSION**

The new target table and the Q subscription will include the source columns that are specified in the INCLUDE clause and the columns that are specified in the EXPRESSION clause. In this example, the target table will be created with three columns: C1, C2, and EXPC3:

```
CREATE QSUB USING REPLQMAP replqmap10
(SUBNAME sub10 dpropr64.srctable TARGET NAME
dpropr64.tghtable TRGCOLS INCLUDE (C1,C2)
EXPRESSION ("CHAR(:C3)" TARGET EXPC3));
```

#### **TRGCOLS EXCLUDE and EXPRESSION**

Source columns that are specified in the EXCLUDE clause will be excluded from the target table and the Q subscription. The target table will include the columns that are specified in the EXPRESSION clause. In this example, the target table will be created with two columns: C1 and EXPC3:

```
CREATE QSUB USING REPLQMAP replqmap10
(SUBNAME sub10 dpropr64.srctable TARGET NAME
dpropr64.tghtable TRGCOLS EXCLUDE(C2,C3)
EXPRESSION ("CHAR(:C3)" TARGET EXPC3));
```

### **Existing target table**

These notes apply to the use of TRGCOLS and EXPRESSION when the target table exists:

#### **TRGCOLS ALL and EXPRESSION**

Not supported. TRGCOLS ALL means that all of the columns in the target table are mapped directly to the source table column names, so EXPRESSION cannot be used.

#### **TRGCOLS INCLUDE and EXPRESSION**

The target columns that are specified in the INCLUDE clause and any expressions that are specified in the EXPRESSION clause will be included in the Q subscription. Any columns that are specified in the INCLUDE clause should not be specified in the EXPRESSION clause. In this example, the target table has the columns C1, C2, EXPC3, and C4. The Q subscription will include the columns C1, C2, and EXPC3:

```
CREATE QSUB USING REPLQMAP replqmap10
(SUBNAME sub10 dpropr64.srctable EXIST TARGET NAME
dpropr64.tghtable TRGCOLS INCLUDE (C1,C2)
EXPRESSION ("CHAR(:C3)" TARGET EXPC3));
```

## TRGCOLS EXCLUDE and EXPRESSION

The target columns that are specified in the EXCLUDE clause will be excluded from the Q subscription. Any expressions that are specified in the EXPRESSION clause will be included in the Q subscription. Columns that are specified in the EXPRESSION clause should be excluded using the EXCLUDE clause. In this example, the target table has the columns C1, C2, EXPC3, and C4. The Q subscription will include the columns C1, C2, and EXPC3:

```
CREATE QSUB USING REPLQMAP replqmap10
(SUBNAME sub10 dpropr64.srctable EXIST TARGET NAME
dpropr64.tgttable TRGCOLS EXCLUDE(C4,C3)
EXPRESSION ("CHAR(:C3)" TARGET EXPC3));
```

ccdoptions

### BEFORE IMAGE COLUMNS

Specifies that the before-image value of each added column will be replicated.

#### **PREFIX "x"**

Specifies the prefix for each before-image column. If you do not specify a prefix, the default value of is used. If this prefix generates invalid names, other letters will be used beginning with the letter Y until valid names are generated.

#### **ALL**

Specifies that all of the after-image columns have before-image columns. This option is the default. Depending on the prefix that you choose, the DB2 database either picks before-image columns for existing targets or generates new before-image columns for new targets.

#### **COLS**

Specifies custom before-image column names.

#### **AFTER *aftercols***

Specifies the name of the after-image column in the target table.

#### **BEFORE *beforecols***

Specifies the name of the before-image column in the target table. This parameter is required. The value of **BEFORE** takes precedence over the name that is generated by the prefix for this particular column.

#### **INCLUDE**

Specifies the columns that will be part of the before-image columns.

#### **AFTER *afterincludes***

Specifies the name of the after-image column in the target table.

#### **BEFORE *beforeincludes***

Specifies the name of the before-image column. This parameter is optional. The value of **BEFORE** takes precedence over the name that is generated by the prefix for this particular column.

#### **FOR KEY COLS ONLY**

Specifies that before-image columns are generated only for the replication key columns.

period-clause:

#### **PERIOD**

Specifies that the source table is a temporal table on DB2 10 for z/OS or later and you want to include some or all of the period columns in the Q subscription.

**ALL**

Specifies that you want to include all period columns.

**SYSTEM\_TIME**

Specifies that you want to include the timestamp columns that are used with system-period temporal tables.

**BUSINESS\_TIME**

Specifies that you want to include the timestamp or date columns that are used with application-period temporal tables.

history-table-clause

**INCLUDE HISTORY**

Specifies that the source table is a temporal table with versioning on DB2 10 for z/OS or later, and that you want to create a corresponding Q subscription for the history table.

**EXIST**

Specifies that you want to create a Q subscription for an existing history table.

**HIST\_TARGET NAME**

Specifies the name of the target history table. If you specify the EXIST keyword but do not specify a name, the ASNCLP program uses the history table for the target temporal table as the history target. Also use this keyword to specify the name for a new target history table that the ASNCLP creates.

tbspace-clause

**IN****DB** *name*

Specifies the name of the logical database for the table space (required for z/OS).

*tname*

Specifies the name of the table space for the target history table. If you want to use an existing table space, the target history table must be the only table that uses the table space.

**NAMING PREFIX** *prefix*

Specifies the prefix to use to name the table space.

prof-clause:

**CREATE**

Specify to create a table space. If this keyword is not specified, the table space is treated as an existing one.

**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*.

- The **DB** value for Logical Database is mandatory for target tables on z/OS products. It must be specified in the profile.
- If a mass subscription is used (for example, if you use the **SRC OWNER LIKE** or **SRC NAME LIKE** clause), the specified *target\_owner.target\_name* clause is valid only if the target table exists. Only the default or a naming prefix are allowed for generated target tables.
- The **CREATE QSUB** command performs an additional check when you create a Q subscription for a CCD target. If you configured Q Apply to manage an SQL Capture schema, and an SQL registration exists for the target CCD in this schema, the ASNCLP issues a message that Q Apply will manage the target CCD as an SQL replication source automatically.

## 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 from a DB2 source. 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 st1ec1 ID ADMF001 password "xx";
SET SERVER TARGET to dbALIAS EC06V71A DBNAME st1ec1 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 creates a Q subscription SUB\_T1 that specifies an automatic load (LOAD TYPE 1) and creates a new nickname REPLDBA.NICK\_T1 at the Q Apply server for the LOAD from CURSOR utility. RMTSAMPLE is the remote server definition on TESTDB that points to the SAMPLE database, which is the data source for the nickname.

```
ASNCLP SESSION SET TO Q REPLICATION;
SET OUTPUT CAPTURE SCRIPT "REPLCAP.SQL" TARGET SCRIPT "REPLAPP.SQL";
SET LOG "QSUB.LOG";
SET SERVER CAPTURE TO DB SAMPLE;
SET SERVER TARGET TO DB TESTDB;
SET APPLY SCHEMA ASN;
SET CAPTURE SCHEMA SOURCE ASN;
CREATE QSUB (SUBNAME "SUB_T1" REPLQMAP SAMPLE_ASN_TO_TESTDB_ASN REPLDBA.T_TEMP
  OPTIONS HAS LOAD PHASE I TARGET NAME REPLDBA.T_TEMPNEWNEW TYPE USERTABLE
  NEW NICKNAME RMT SERVERNAME RMTSAMPLE REPLDBA.NICK_T1 LOAD TYPE 1);
```

### Example 3

This examples creates the SUB\_T2 Q subscription and specifies that the Q Apply program use an existing nickname, REPLDBA.NICK\_T2, for the LOAD from CURSOR utility.

```
ASNCLP SESSION SET TO Q REPLICATION;
SET OUTPUT CAPTURE SCRIPT "REPLCAP.SQL" TARGET SCRIPT "REPLAPP.SQL";
SET LOG "QSUB.LOG";
SET SERVER CAPTURE TO DB SAMPLE;
SET SERVER TARGET TO DB TESTDB;
SET APPLY SCHEMA ASN;
SET CAPTURE SCHEMA SOURCE ASN;
CREATE QSUB (SUBNAME "SUB_T2" REPLQMAP SAMPLE_ASN_TO_TESTDB_ASN REPLDBA.T_TEMP
OPTIONS HAS LOAD PHASE I TARGET NAME REPLDBA.T_TEMPNEWNEW TYPE USERTABLE
NICKNAME REPLDBA.NICK_T2 LOAD TYPE 1);
```

### Example 4

This example demonstrates the use of a naming prefix for the target table (XNEW) and table space 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 5

This example shows how to use a table space profile (USING PROFILE UTRGTS) for the target table space 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 6

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 7

This example creates all of the target tables in one table space (RST1).

```
CREATE QSUB USING REPLQMAP QDECODERQM (SRC OWNER LIKE "DSN8710%" SRC NAME LIKE
"%EMP%" TARGET TABLE NAME XNEW IN DB D1CDG01 RTS1);
```

### Example 8

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 9

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 10

In this example the source and target owner names are the same. For the source and target owner names to be the same, 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 11

This example does not use the environment and profile from “Example 1” on page 118. It creates a Q subscription for unidirectional replication from a DB2 source that uses the replication queue map SAMPLE\_ASN1\_TO\_TARGETDB\_ASN1 and specifies that the Q Apply program loads the target tables with the EXPORT and IMPORT utilities. It also specifies 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 12

This example creates a Q subscription from the DB2 table EMPLOYEE to the Sybase table TGT\_EMPLOYEE. The table will be created in the existing Sybase segment SEG\_EMPLOYEE by using the SAMPLE\_ASN\_TO\_FEDDB\_ASN replication queue map. The table will have the nickname of EMPNICKNAME.

```
CREATE QSUB USING REPLQMAP SAMPLE_ASN_TO_FEDDB_ASN (SUBNAME FEDQSUB
EMPLOYEE TARGET NAME TGTEMPLOYEE FEDERATED EMPNICKNAME);
```

## Example 13

This example creates a Q subscription with a new target CCD table. All of the columns in the source table are in the Q subscription and all of the columns in the target will have before-image columns.

```
ASNCLP SESSION SET TO Q REPLICATION;
SET RUN SCRIPT NOW STOP ON SQL ERROR ON;
SET SERVER CAPTURE TO DB SAMPLE;
SET SERVER TARGET TO DB SAMPLE;
SET CAPTURE SCHEMA SOURCE ASNAPP1;
SET APPLY SCHEMA ASNAPP1;
CREATE QSUB USING REPLQMAP SAMPLE_ASNAPP1_TO_SAMPLE_ASNAPP1
(SUBNAME TESTCCCDNEW DATA.EMPLOYEE TARGET NAME DATA.TGTEMPLOYEE
TYPE CCD CONDENSED ON COMPLETE ON WITH UOW COLS ALL
TRGCOLS ALL BEFORE IMAGE COLUMNS ALL);
```

## Example 14

This example creates a Q subscription with new target CCD table. All of the columns in the source table take part in the Q subscription. The command also specifies before-image columns for the key columns and a before-image prefix of Y.

```
ASNCLP SESSION SET TO Q REPLICATION;
SET RUN SCRIPT NOW STOP ON SQL ERROR ON;
SET SERVER CAPTURE TO DB SAMPLW;
SET SERVER TARGET TO DB SAMPLE;
SET CAPTURE SCHEMA SOURCE ASNAPP1;
SET APPLY SCHEMA ASNAPP1;
CREATE QSUB USING REPLQMAP SAMPLE_ASNAPP1_TO_SAMPLE_ASNAPP1
```



```
(SUBNAME TESTCCCDNEW DATA.EMPLOYEE TARGET NAME DATA.TGTEMPLOYEE
TYPE CCD CONDENSED ON COMPLETE ON WITH UOW COLS ALL
TRGCOLS ALL BEFORE IMAGE COLUMNS PREFIX "Y" FOR KEYS COLS ONLY);
```

## Example 15

This example creates a Q subscription with a new CCD target table. All of the columns in the source table are in the Q subscription. The command specifies a subset of columns that will have before images. The command also specifies the before-image column names for these columns.

```
ASNCLP SESSION SET TO Q REPLICATION;
SET RUN SCRIPT NOW STOP ON SQL ERROR ON;
SET SERVER CAPTURE TO DB SAMPLE;
SET SERVER TARGET TO DB SAMPLE;
SET CAPTURE SCHEMA SOURCE ASNAPP1;
SET APPLY SCHEMA ASNAPP1;
CREATE QSUB USING REPLQMAP SAMPLE_ASNAPP1_TO_SAMPLE_ASNAPP1
(SUBNAME TESTCCCDNEW DATA.EMPLOYEE TARGET EXIST NAME DATA.TGTEMPLOYEE
TYPE CCD CONDENSED ON COMPLETE ON WITH UOW COLS ALL
TRGCOLS ALL BEFORE IMAGE COLUMNS INCLUDE
(AFTER C1 BEFORE BEFC1, AFTER C2 BEFORE BEFC2, AFTER C3 BEFORE BEFC3));
```

## Example 16

This example creates a Q subscription with a new target CCD table. The before-image columns exist for all the replicated columns in the target. Some columns have a before-image prefix of Y while the others have no specific prefix.

```
ASNCLP SESSION SET TO Q REPLICATION;
SET RUN SCRIPT NOW STOP ON SQL ERROR ON;
SET SERVER CAPTURE TO DB SAMPLE;
SET SERVER TARGET TO DB SAMPLE;
SET CAPTURE SCHEMA SOURCE ASNAPP1;
SET APPLY SCHEMA ASNAPP1;
CREATE QSUB USING REPLQMAP SAMPLE_ASNAPP1_TO_SAMPLE_ASNAPP1
(SUBNAME TESTCCCDEXIST DATA.EMPLOYEE TARGET EXIST NAME DATA.TGTEMPLOYEE
TYPE CCD CONDENSED ON COMPLETE ON WITH UOW COLS ALL
TRGCOLS ALL BEFORE IMAGE COLUMNS PREFIX "Y" ALL COLS
(AFTER C1 BEFORE BEFC1, AFTER C2 BEFORE BEFC2));
```

## Example 17

This example creates a Q subscription with a new target CCD table. Only a subset of the columns in the target table participate in replication, and before-image columns exist only for three columns in the target table. The before-image columns do not have a specific prefix and have different names for each after-image column.

```
ASNCLP SESSION SET TO Q REPLICATION;
SET RUN SCRIPT NOW STOP ON SQL ERROR ON;
SET SERVER CAPTURE TO DB SAMPLE;
SET SERVER TARGET TO DB SAMPLE;
SET CAPTURE SCHEMA SOURCE ASNAPP1;
SET APPLY SCHEMA ASNAPP1;
CREATE QSUB USING REPLQMAP SAMPLE_ASNAPP1_TO_SAMPLE_ASNAPP1
(SUBNAME TESTCCCDEXIST DATA.EMPLOYEE TARGET EXIST NAME DATA.TGTEMPLOYEE
TYPE CCD CONDENSED ON COMPLETE ON WITH UOW COLS ALL
TRGCOLS INCLUDE (C1, C2, C3, C4, C5) BEFORE IMAGE COLUMNS INCLUDE
(AFTER C1 BEFORE BEFC1, AFTER C2 BEFORE BEFC2, AFTER C3 BEFORE BEFC3));
```

## Example 18

This example creates a Q subscription by using a target column expression that maps all of the columns that match the expression `CONCAT(:C1,:C2)` to the target column `CEXP`.

```
ASNCLP SESSION SET TO Q REPLICATION;
SET RUN SCRIPT NOW STOP ON SQL ERROR ON;
SET SERVER CAPTURE TO DB SAMPLE;
SET SERVER TARGET TO DB SAMPLE;
SET CAPTURE SCHEMA SOURCE ASNAPP1;
SET APPLY SCHEMA ASNAPP1;
CREATE QSUB USING REPLQMAP SAMPLE_ASNAPP1_TO_SAMPLE_ASNAPP1
(SUBNAME TESTEXPRESSTION DATA.EMPLOYEE TARGET NAME DATA.TGTEMPLOYEE
TRGCOLS ALL EXPRESSION ("CONCAT(:C1,:C2)" TARGET CEXP));
```

## Example 19

This example creates a Q subscription called `CLASSIC0001` for Classic replication. The `CREATE QSUB` command specifies a source table called `CLASSICTABLE` and specifies that the Q Apply program is to load a target table of the same name.

```
ASNCLP SESSION SET TO Q REPLICATION;
SET SERVER CAPTURE TO CONFIG SERVER classic1 FILE "asnserver.ini"
ID CLASSICADMIN PASSWORD "passwd";
SET SERVER TARGET TO DB TARGET ID DB2ADMIN PASSWORD "passwd";
SET RUN SCRIPT NOW STOP ON SQL ERROR ON;
SET APPLY SCHEMA ASN1;
CREATE QSUB USING REPLQMAP CLASSIC_ASN1_TO_TARGET_ASN1 (SUBNAME CLASSIC0001
CLASSICTABLE OPTIONS HAS LOAD PHASE I TARGET NAME CLASSICTABLE LOAD TYPE 4);
```

## Example 21

This example creates a Q subscription for the Oracle target table `HR.EMPLOYEE`. The nickname that references the target table, `HR.EMPNICK`, already exists on the Q Apply server.

```
ASNCLP SESSION SET TO Q REPLICATION;
SET SERVER CAPTURE TO DB SAMPLE;
SET SERVER TARGET TO DB FEDORA NONIBM SERVER V100RA;
CREATE QSUB USING REPLQMAP REPMAP1
(SUBNAME SUB1 EMPLOYEE EXIST NICKNAME HR.EMPNICK TYPE NICKNAME);
```

---

## CREATE REPLQMAP command

Use the **CREATE REPLQMAP** command to create a replication queue map for Q subscriptions.

### Syntax

```
▶▶ CREATE REPLQMAP qmapname [DESC "description"] [(node-x, node-y)]
▶ USING ADMINQ "admnqname" RECVQ "recvqname" SENDQ "sendqname" [NUM APPLY AGENTS num]
▶ [MAXAGENTS CORRELID num] [MEMORY LIMIT limit] [ERROR ACTION {S|Q}]
```

## Parameters

*qmapname*

Specifies the name of the replication queue map.

**DESC** "*description*"

Specifies the description of the replication queue map.

**NODE** *x*

In multidirectional replication, specifies the source server for this replication queue map. Use the same node number that was used in the SET BIDI NODE or SET PEER NODE command.

**NODE** *y*

In multidirectional replication, specifies the target server for this replication queue map. Use the same node number that was used in the SET BIDI NODE or SET PEER NODE command.

**ADMINQ** "*adminqname*"

Specifies the name of the administration queue at the Q Apply server.

**Note:** If the Q Capture or the Classic capture components share a single queue manager with the Q Apply program, the programs can share an administration queue.

**RECVQ** "*recvqname*"

Specifies the name of the receive queue that is used by the Q Apply program.

**SENDQ** "*sendqname*"

Specifies the name of the send queue that is used by the Q Capture program (for relational sources) or the capture components.

**NUM APPLY AGENTS** *num*

Specifies the number of threads that are used for concurrently applying transactions from the specified receive queue.

**MAXAGENTS CORRELID***num*

**z/OS** Specifies that number of threads that are used for concurrently applying transactions from the specified receive queue with the same *correlation ID*. The correlation ID identifies all transactions that were started from the same z/OS job on the Q Capture server.

The value for the **MAXAGENTS CORRELID** parameter cannot be greater than the value for the **NUM APPLY AGENTS** parameter. If **MAXAGENTS\_CORRELID** value is 1, the transactions will be applied one at a time. If the value is greater than one, for example 4, four agents will apply transactions with the same correlation ID in parallel. If the value is 0, transactions are applied in parallel by using the total number of threads specified by the **NUM APPLY AGENTS** parameter.

**MEMORY LIMIT** *limit*

Specifies the maximum number of megabytes that are used per receive queue for buffering incoming transactions.

**ERROR ACTION**

The action that the Q Capture program takes 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.

- S** The Q Capture program or the capture components stop when they detect an error on this queue.
- Q** The Q Capture program stops putting messages on any send queues that are in error and continues putting messages on other send queues. This value is not supported for Classic replication.

**HEARTBEAT INTERVAL** *interval*

Specifies the interval (in seconds) between heartbeat messages that are sent from the Q Capture program or the capture components 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 that is used for sending messages over the send queue.

### Example 1

To create a replication queue map SAMPLE\_ASN1\_TO\_TARGETDB\_ASN1 from a relational source:

```
CREATE REPLQMAP SAMPLE_ASN1_TO_TARGETDB_ASN1 USING ADMINQ "ASN1.QM1.ADMINQ"
RECVQ "ASN1.QM1_TO_QM2.DATAQ" SENDQ "ASN1.QM1_TO_QM2.DATAQ"
```

### Example 2

To create a replication queue map CLASSIC\_ASN\_TO\_TARGETDB\_ASN1 from a Classic source:

```
SET SERVER CAPTURE TO CONFIG SERVER classic1 FILE classic.ini ID id1 PASSWORD pwd1
SET SERVER TARGET TO DB ASN1
SET RUN SCRIPT NOW STOP ON SQL ERROR ON
CREATE REPLQMAP CLASSIC1_ASN_TO_TARGETDB_ASN1 USING ADMINQ "ASN1.QM1.ADMINQ"
RECVQ "CLASSIC1.QM1_TO_QM2.DATAQ" SENDQ "CLASSIC1.QM1_TO_QM2.DATAQ"
```

### Example 3

In a bidirectional replication configuration, to create a replication queue map SAMPLE\_ASN\_TO\_TARGETDB\_ASN1 to connect the Q Capture program at the SAMPLE server (node 1) with the Q Apply program at the TARGETDB server (node 2):

```
CREATE REPLQMAP SAMPLE_ASN_TO_TARGETDB_ASN1 (NODE 1, NODE 2) USING ADMINQ
"ASN1.QM1.ADMINQ" RECVQ "ASN1.QM1_TO_QM2.DATAQ" SENDQ "ASN1.QM1_TO_QM2.DATAQ"
```

## CREATE SCHEMASUB command

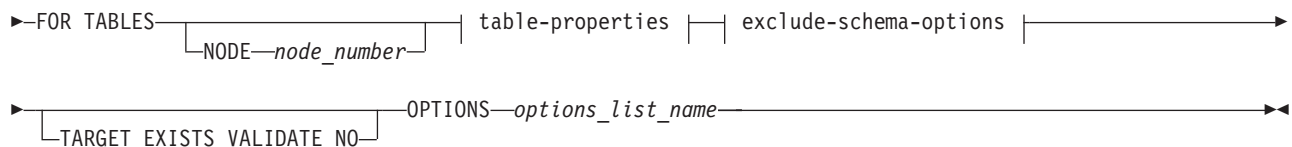
Use the **CREATE SCHEMASUB** command to create a schema-level subscription for unidirectional and bidirectional replication.

The command:

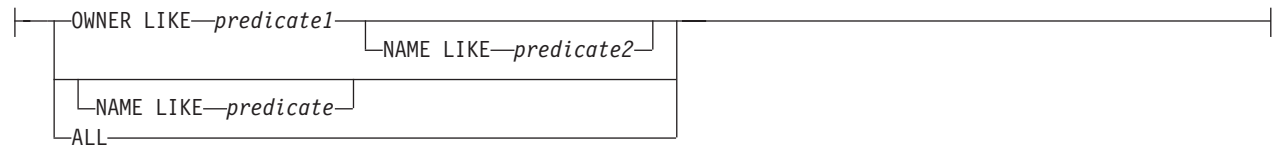
- Creates table-level Q subscriptions for all tables within the schema that meet the naming pattern that you specify.
- Saves the schema pattern so that the replication programs automatically create Q subscriptions for any tables that are added within the schema.

### Syntax

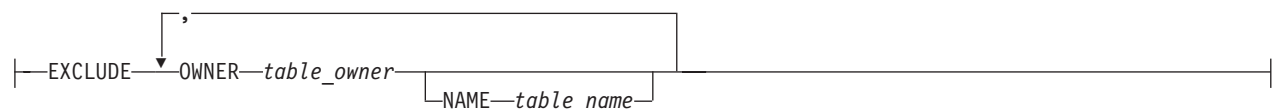
```
►►—CREATE SCHEMASUB—schema_subname—SUBTYPE—┌B—┐
└U—REPLQMAP—queue_map_name—┘
```



### table-properties:



### exclude-schema-options:



## Parameters

### SUBTYPE

Specifies the type of replication:

- U** Unidirectional. You must specify a replication queue map to be used by all Q subscriptions within the schema.
- B** Bidirectional.

For bidirectional configurations, you do not need to specify a replication queue map if only one set of queue maps (one queue map in each direction) exists between the two servers. If more than one set of queue maps exists, use the SET CONNECTION command to specify which set of queue maps to use for the schema-level subscription.

### FOR TABLES

Use FOR TABLES along with the table-properties clause to specify a pattern for selecting the schemas, and tables within the schemas, that should be included in the schema-level subscription. Follow these guidelines:

- You can use the percentage sign (%) as a wild card.
- To replicate all CREATE TABLE and DROP TABLE operations within all schemas in the database, specify the ALL keyword (which is equivalent to OWNER LIKE % NAME LIKE %, and is stored as %.%).
- Patterns for schema-level subscriptions that use the same replication queue map must not overlap so that a table matches both patterns. For example, if you specified OWNER LIKE SMITH NAME LIKE % (stored as SMITH.%) and another schema-level subscription already existed that was created with OWNER LIKE % NAME LIKE T1 (stored as %.T1), both patterns would match the table SMITH.T1 and the CREATE SCHEMASUB command would fail.
- Table-level Q subscriptions that are part of a schema-level Q subscription and use the same replication queue map should all be of the same configuration type (unidirectional or bidirectional) and have the same properties.

**NODE**

For SUBTYPE B or P. Specifies the server where the source tables to be included in the schema-level subscription reside.

**TARGET EXISTS VALIDATE NO**

Specifies that the target table exists and no validation is required for table-level Q subscriptions that are created by the ASNCLP program. This option shortens processing time with very large tables. If these keywords and the SET ENFORCING MATCHING CONSTRAINTS command are used, the TARGET EXISTS VALIDATE NO clause provided on the CREATE SCHEMASUB command takes precedence.

**Important:** If you use these keywords, the ASNCLP program assumes that the target table matches exactly with the source table.

**OPTIONS**

Specifies the name of a profile (list of options) for creating table-level Q subscriptions. You create the profile by using the CREATE SUBSCRIPTION OPTIONS command.

table-properties

**OWNER LIKE**

Specifies a single database schema or schema pattern that uses the percentage sign (%) as a wild card.

**NAME LIKE**

Specifies a single table name or table-naming pattern that uses the percentage sign (%) as a wild card.

**ALL**

Specifies that you want all schemas in the database, and all tables in the schemas, to be part of the schema-level subscription.

exclude-schema-options

**OWNER**

Specifies a schema to exclude from the schema-level subscription. For example, if there is a schema-level subscription for all tables in all schemas (using the wild card pattern %.%), but you specify EXCLUDE OWNER MSROSS, the statement CREATE TABLE MSROSS.T1 will not be replicated. A wild card is not allowed with this keyword.

**NAME**

Specifies one or more tables to exclude from the schema-level Q subscription. You can specify a single table name or table-naming pattern that uses the percentage sign (%) as a wild card.

**Usage notes**

- If you created a saved profile for creating target tables by using the SET PROFILE command, the options are used by the CREATE SCHEMASUB command when it creates target tables for table-level Q subscriptions.

**Example 1**

To create a schema-level subscription for unidirectional replication that includes all tables under the schema MSROSS:

```
CREATE SCHEMASUB SUBTYPE U REPLQMAP RQ1 FOR TABLES OWNER LIKE MSROSS;
```

## Example 2

To create a schema-level subscription for bidirectional replication that includes all schemas and tables on the SAMPLE1 database and uses the saved profile options1:

```
SET BIDI NODE 1 SERVER DBALIAS SAMPLE1;
SET BIDI NODE 2 SERVER DBALIAS SAMPLE2;
```

```
CREATE SCHEMASUB SUBTYPE B FOR TABLES NODE 1 ALL OPTIONS options1;
```

---

## CREATE SUBSCRIPTION OPTIONS command

Use the **CREATE SUBSCRIPTION OPTIONS** command to create a profile that can be used to create table-level Q subscriptions when a schema-level subscription is in place. When the Q Capture program detects a CREATE TABLE operation within the schema, it automatically creates a Q subscription and uses the options that are specified in this profile.

**Relationship with SET PROFILE command:** The options that you specify in the SET PROFILE command are used by the CREATE SCHEMASUB command to create target tables for Q subscriptions that are created by ASNCLP. The options in the SET PROFILE and CREATE SUBSCRIPTIONS OPTIONS commands do not intersect, and you can include both commands in the same input file. If both the SET PROFILE and CREATE SUBSCRIPTION OPTIONS commands are provided, the Q subscription-related attributes are picked from the CREATE SUBSCRIPTION OPTIONS command and the target table space attributes are picked from the SET PROFILE command.

### Syntax

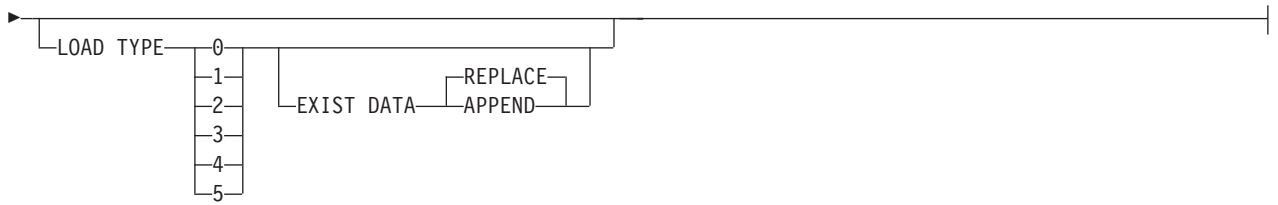
```
▶▶ CREATE SUBSCRIPTION OPTIONS options_name [ uni-properties | bidi-properties ] ▶▶
```

#### uni-properties:

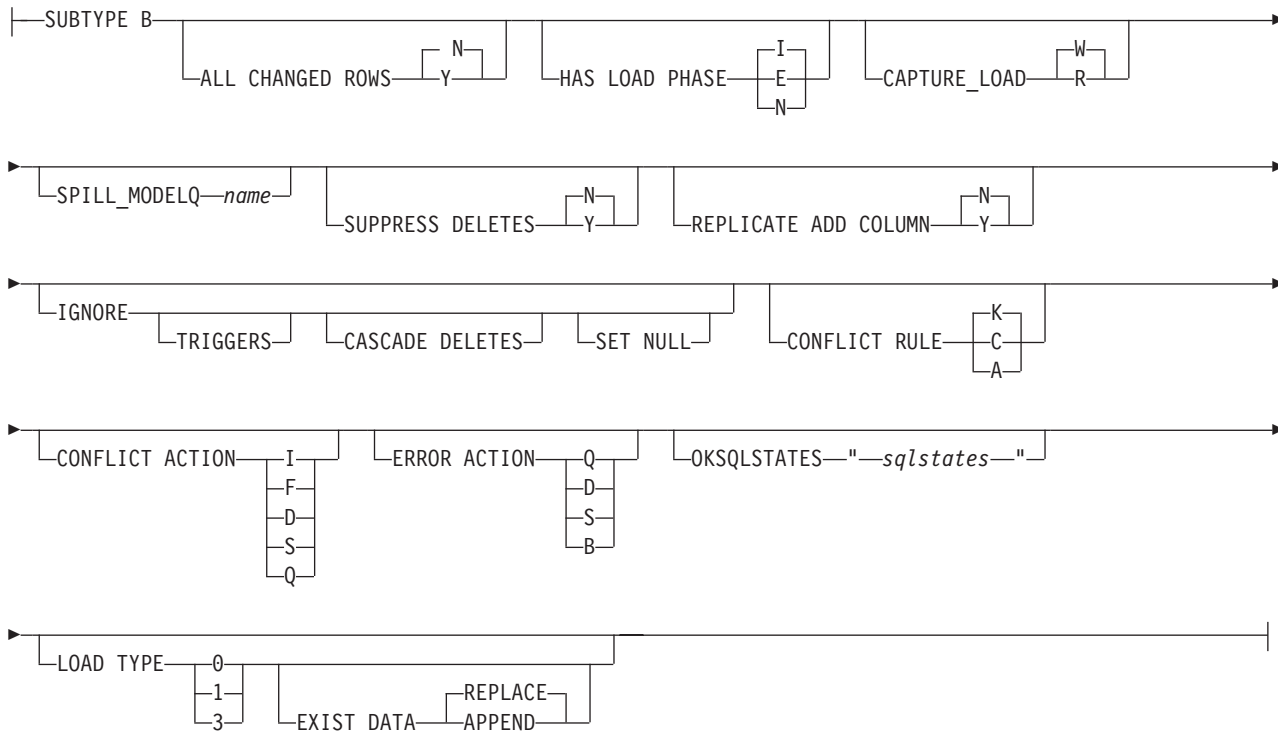
```

SUBTYPE U
  ALL CHANGED ROWS [ N | Y ]
  HAS LOAD PHASE [ I | E | N ]
  CAPTURE_LOAD [ W | R ]
  SPILL_MODELQ name
  SUPPRESS DELETES [ N | Y ]
  REPLICATE ADD COLUMN [ N | Y ]
  IGNORE [ TRIGGERS | CASCADE DELETES | SET NULL ]
  CONFLICT ACTION [ I | F | D | S | Q ]
  ERROR ACTION [ Q | D | S | B ]
  OKSQLSTATES "sqlstates"

```



**bidirectional-properties:**



**Parameters**

For descriptions of the command parameters, see the identical descriptions in one of the following topics:

- “CREATE QSUB command (unidirectional replication)” on page 101
- “CREATE QSUB command (bidirectional replication)” on page 190

**Example**

This example creates a profile called `bidioptions` that specifies properties for table-level, bidirectional Q subscriptions between the `SAMPLE` and `SAMPLE2` servers. The profile specifies a manual load phase and that cascaded delete operations should not be replicated:

```
SET BIDI NODE 1 SERVER SAMPLE;
SET BIDI NODE 2 SERVER SAMPLE2;

CREATE SUBSCRIPTION OPTIONS bidioptions
SUBTYPE B HAS LOAD PHASE E IGNORE CASCADE DELETES;
```



---

## DROP CONTROL TABLES ON command

Use the **DROP CONTROL TABLES ON** command to drop the Q Capture control tables, Q Apply control tables, or both. In Classic replication, you can use this command to drop only the Q Apply control tables.

### Syntax



### Parameters

#### **CAPTURE SERVER**

Specify to drop the Q Capture control tables.

#### **APPLY SERVER**

Specify to drop the Q Apply control tables.

#### **NODE**

Specify to drop the Q Capture and Q Apply control tables on a server in a bidirectional or peer-to-peer configuration. The server is identified by *node\_number*.

### Usage notes

This command is used in conjunction with the **SET SERVER** command to indicate the location of the control tables.

### Example: Q Capture control tables

To drop the Q Capture control tables:

```
SET SERVER TARGET TO QAPPDB;  
DROP CONTROL TABLES ON APPLY SERVER
```

### Example: Dropping both sets of control tables

To drop both Q Capture and Q Apply control tables on the SAMPLE1 and SAMPLE2 servers:

```
SET BIDI NODE 1 SERVER DBALIAS SAMPLE1;  
SET BIDI NODE 2 SERVER DBALIAS SAMPLE2;  
  
SET RUN SCRIPT NOW STOP ON SQL ERROR ON;  
  
DROP CONTROL TABLES ON NODE 1;  
DROP CONTROL TABLES ON NODE 2;
```

---

## DROP QSUB command

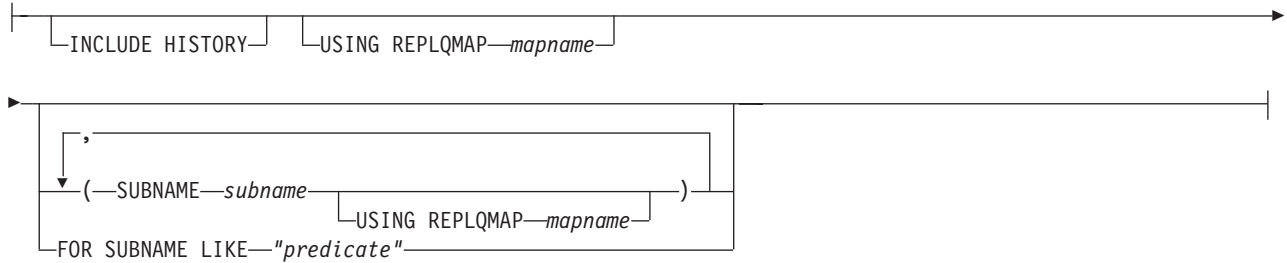
Use the **DROP QSUB** command to delete one or more Q subscriptions for unidirectional, bidirectional, or peer-to-peer Q Replication.

**Note:** Starting with Version 10 on Linux, UNIX, and Windows, use this command rather than the deprecated **DROP SUBTYPE** command to delete multidirectional Q subscriptions.

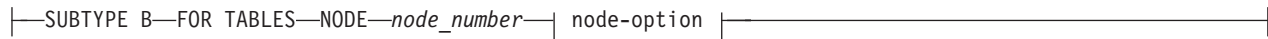
## Syntax



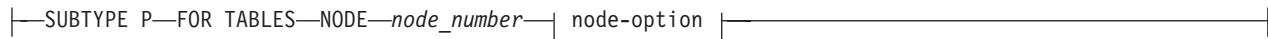
### uni-options:



### bidi-options:



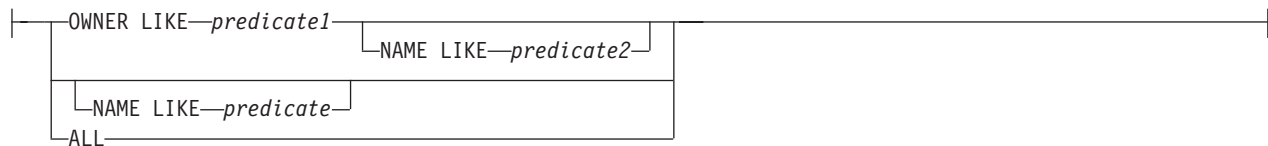
### p2p-options:



### node-option:



### source-predicate:



## Parameters

### ALL

Specify to delete all Q subscriptions. If you specify this parameter, you cannot combine it with any other parameters.

### uni-options

#### INCLUDE HISTORY

Specify to delete the Q subscription for the history table when the Q subscription for the base temporal table is deleted. If this clause is not specified, the option that was specified in the SET DROP TEMPORAL HISTORY SUB clause is used.

**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%";
```

bidi-options

**SUBTYPE B**

Specifies that you want to delete one or more bidirectional Q subscriptions.

**FOR TABLES**

Use this clause to specify one or more logical tables for which to delete paired sets of Q subscriptions.

**NODE**

Specifies a server in the bidirectional configuration that should be used to locate the logical table on which the Q subscriptions to be deleted are based.

p2p-options

**SUBTYPE P**

Specifies that you want to delete one or more peer-to-peer Q subscriptions.

**FOR TABLES**

Use this clause to specify one or more logical tables for which to delete paired sets of Q subscriptions.

**NODE**

Specifies a server in the peer-to-peer configuration that should be used to locate the logical table on which the Q subscriptions to be deleted are based.

node-options

Use these options to select one or more tables for which to delete Q subscriptions.

*source\_owner*

Specifies the schema of a single logical table.

*source\_name*

Specifies the name of a single logical table.

source-predicate

Use these options to specify multiple logical tables for which to delete Q subscriptions.

**OWNER LIKE**

Specifies a single database schema or schema pattern that uses the percentage sign (%) as a wild card.

**NAME LIKE**

Specifies a single table name or table-naming pattern that uses the percentage sign (%) as a wild card.

**ALL**

Specifies that you want to delete Q subscriptions for all schemas and all tables within those schemas.

**Example: unidirectional**

To delete a Q subscription for unidirectional replication:

```
DROP QSUB (SUBNAME EMPLOYEE0001 USING REPLQMAP SAMPLE_ASN1_TO_TARGETDB_ASN1);
```

**Example: multidirectional**

To delete all of the paired Q subscriptions for bidirectional replication under schemas that start with the letters "AIRUKU" on the SAMPLE1 and SAMPLE2 servers:

```
SET BIDI NODE 1 SERVER DBALIAS SAMPLE1;
SET BIDI NODE 2 SERVER DBALIAS SAMPLE2;
```

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

```
DROP QSUB SUBTYPE B FOR TABLES (NODE 1 OWNER LIKE "AIRUKU%");
```

**DROP REPLQMAP command**

Use the **DROP REPLQMAP** command to delete existing replication queue maps.

**Restriction:** Before you use the **DROP REPLQMAP** command, delete all Q subscriptions that use the replication queue map.

**Syntax**

```
►► DROP REPLQMAP qmapname [NODE x, -NODE y]
```

**Parameters**

*qmapname*

Specifies the name of the replication queue map to delete.

**NODE *x*, NODE *y***

Specifies to delete the replication queue map that connects two servers in one direction (**NODE *x*** and **NODE *y***) in multidirectional replication.

**Example: unidirectional**

To delete the SAMPLE\_ASN1\_TO\_TARGETDB\_ASN1 replication queue map:

```
DROP REPLQMAP SAMPLE_ASN1_TO_TARGETDB_ASN1;
```

**Example: multidirectional**

To delete both replication queue maps between the SAMPLE1 and SAMPLE2 servers in a bidirectional configuration:

```

SET BIDI NODE 1 SERVER DBALIAS SAMPLE1;
SET BIDI NODE 2 SERVER DBALIAS SAMPLE2;

SET RUN SCRIPT NOW STOP ON SQL ERROR ON;

DROP REPLQMAP repqmap1 NODE 1, NODE 2;
DROP REPLQMAP repqmap2 NODE 2, NODE 1;

```

---

## DROP SCHEMASUB command

Use the **DROP SCHEMASUB** command to delete a schema-level subscription. You can also use this command to delete all Q subscriptions that belong to the schema-level subscription.

### Syntax

```

▶▶—DROP SCHEMASUB—schema_sub_name—ALL  
NEW ONLY—▶▶

```

### Parameters

#### ALL

Specify to delete the schema-level subscription and all of the table-level Q subscriptions that belong to it.

#### NEW ONLY

Specify to delete only the schema-level subscription.

### Example 1

To delete the schema-level subscription `schema1` in a bidirectional configuration and delete all of the table-level Q subscription that belong to it:

```

SET BIDI NODE 1 SERVER SAMPLE;
SET BIDI NODE 2 SERVER SAMPLE2;

DROP SCHEMASUB schemasub1 ALL;

```

### Example 2

To delete the schema-level subscription `schema2` in a bidirectional configuration but leave all of the table-level Q subscription that belong to it:

```

SET BIDI NODE 1 SERVER SAMPLE;
SET BIDI NODE 2 SERVER SAMPLE2;

DROP SCHEMASUB schemasub2 NEW ONLY;

```

---

## DROP SUBSCRIPTION OPTIONS command

Use the **DROP SUBSCRIPTION OPTIONS** command to delete a list of Q subscription options that is used as a profile for creating table-level Q subscriptions when a schema-level subscription is in place.

**Important:** You can only use this command if the list of Q subscription options is not being used by any schema-level Q subscriptions. Any schema-level subscriptions that are using the list must be deleted before you can delete the list.

## Syntax

```
►► DROP SUBSCRIPTION OPTIONS options_name ◀◀
```

### Parameters

*options\_name*

The name of the list of Q subscription options, as specified in the CREATE SUBSCRIPTION OPTIONS command and stored in the IBMQREP\_SUBS\_PROF table at the Q Capture server.

### Example

To delete the list of Q subscription options named options1 that is used as a profile for creating Q subscriptions between the SAMPLE and SAMPLE1 servers:

```
SET BIDI NODE 1 SERVER SAMPLE;  
SET BIDI NODE 2 SERVER SAMPL1;  
  
DROP SUBSCRIPTION OPTIONS options1;
```

---

## LIST QSUB command (Q Replication)

Use the **LIST** command to list Q subscriptions.

### Syntax

```
►► LIST QSUB [FOR TABLEOWNER ownername | FOR QMAP mapname] FOR [QCAPTURE | QAPPLY] ◀◀  
[SCHEMA schema] [SERVER dbparms]
```

#### dbparms-clause:

```
[DB dbalias | DBALIAS aliasname | DBNAME dbname | ID userid | PASSWORD pwd |  
CONFIG SERVER servername | FILE filename]
```

### Parameters

**FOR TABLEOWNER** *ownername*

List only the Q subscriptions dedicated to the specified table owner name.

**FOR QMAP** *mapname*

List only the Q subscriptions used by the specified replication queue map.

**QCAPTURE**

List the Q subscription information that is defined in a single set of Q Capture control tables. Use this parameter with the **CONFIG SERVER** parameter to specify a Classic source.

**QAPPLY**

List the Q subscription information that is defined in a single set of Q Apply control tables.

**SCHEMA** *schema*

Specifies which schema to use. The default is "ASN".

dbparms-clause:

**DB** *dbalias*

Specifies the database alias name of a z/OS subsystem or Linux, UNIX, or Windows database as cataloged on the DB2 from which the ASNCLP is invoked. This keyword is deprecated.

**DBALIAS** *aliasname*

Specifies the database alias name of a z/OS subsystem or Linux, UNIX, or Windows database as cataloged on the DB2 from which the ASNCLP is invoked.

**DBNAME** *zosdbname*

**z/OS** Specifies the z/OS database name. This is a logical z/OS database name, as created on a z/OS subsystem.

**ID** *userid*

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

**PASSWORD** *pwd*

Specifies the password to use for connections.

**CONFIG SERVER** *servername*

**Classic sources:** Specifies the Classic source that the ASNCLP program connects to. The server name must match the bracketed [NAME] field that is entered in the ASNCLP configuration file. You cannot use this parameter if you are using the **TARGET** parameter.

**FILE** *filename*

Specifies the complete path and file name to the ASNCLP configuration file. If you do not use the **FILE** parameter, the ASNCLP attempts to use the `asnservers.ini` file in the current directory, if that file exists.

**Example - list by Q Capture schema**

This example lists the Q subscriptions with Q Capture schema ASN. (The SET SERVER command determines which database or subsystem the Q Capture schema is located on.)

```
LIST QSUB FOR QCAPTURE SCHEMA ASN;
```

**Example - list by Classic server schema**

This example lists the Q subscriptions on server CLASSIC1 with schema ASN.

```
LIST QSUB FOR QCAPTURE SCHEMA ASN CONFIG SERVER CLASSIC1 FILE asnservers.ini
ID id1 PASSWORD "passwd1";
```

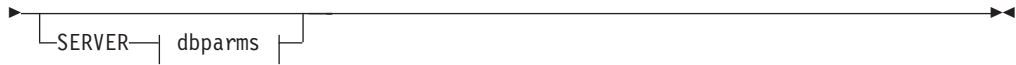
---

## LIST REPLQMAP command (Q Replication)

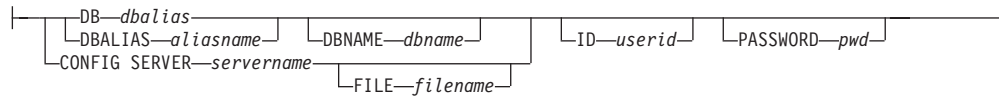
Use the **LIST REPLQMAP** command to list replication queue maps.

**Syntax**

```
►► LIST REPLQMAP FOR [QCAPTURE | QAPPLY] [SCHEMA schema]
```



**dbparms-clause:**



**Parameters**

**QCAPTURE**

List the replication queue map information that is defined in a single set of Q Capture control tables. Use this parameter with the **CONFIG SERVER** parameter to specify a Classic source.

**QAPPLY**

List the replication queue map information that is defined in a single set of Q Apply control tables.

**SCHEMA** *schema*

Specifies which schema to use. The default is "ASN".

dbparms-clause:

**DB** *dbalias*

Specifies the database alias name of a z/OS subsystem or Linux, UNIX, or Windows database as cataloged on the DB2 from which the ASNCLP is invoked. This keyword is deprecated.

**DBALIAS** *aliasname*

Specifies the database alias name of a z/OS subsystem or Linux, UNIX, or Windows database as cataloged on the DB2 from which the ASNCLP is invoked.

**DBNAME** *zosdbname*

**z/OS** Specifies the z/OS database name. This is a logical z/OS database name, as created on a z/OS subsystem.

**ID** *userid*

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

**PASSWORD** *pwd*

Specifies the password to use for connections.

**CONFIG SERVER** *servername*

**Classic sources:** Specifies which server configuration settings from the Classic replication configuration file that the ASNCLP should use to connect to the Classic server.

**FILE** *filename*

Specifies the complete path and file name to the Classic replication configuration file. If you do not use the **FILE** parameter, the ASNCLP attempts to use the `asnservers.ini` file in the current directory, if that file exists.



## Example 1

This example lists the replication queue maps with Q Capture schema ASN. (The **SET SERVER** command determines which database or subsystem the Q Capture schema is located on.)

```
LIST REPLQMAP FOR QCAPTURE SCHEMA ASN;
```

## Example 2

This example lists the replication queue maps on server CLASSIC1 with schema ASN.

```
LIST REPLQMAP FOR QCAPTURE SCHEMA ASN CONFIG SERVER CLASSIC1 FILE asnservers.ini  
ID id1 PASSWORD "passwd1";
```

---

## LIST APPLY SCHEMA command

You can use the **LIST APPLY SCHEMA** command to list the Q Apply schemas for a specified server.

### Syntax

```
▶▶—LIST APPLY SCHEMA—┬──SERVER──┴─ dbparms ─┴─▶▶
```

#### dbparms-clause:

```
├──DBALIAS—aliasname──┬──DBNAME—dbname──┬──ID—userid──┬──PASSWORD—pwd──┬──  
├──CONFIG SERVER—servername──┬──FILE—filename──┬──
```

### Parameters

dbparms-clause:

#### SERVER

Specifies the server that contains the schemas to be listed.

#### DBALIAS *aliasname*

Specifies the database alias name of a z/OS subsystem or Linux, UNIX, or Windows database as cataloged on the DB2 from which the ASNCLP is invoked.

#### DBNAME *zosdbname*

**z/OS** Specifies the z/OS database name. This is a logical z/OS database name, as created on a z/OS subsystem.

#### ID *userid*

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

#### PASSWORD *pwd*

Specifies the password to use for connections.

#### CONFIG SERVER *servername*

**Classic sources:** Specifies which server configuration settings from the Classic replication configuration file that the ASNCLP should use to connect to the Classic server.

**FILE** *filename*

Specifies the complete path and file name to the replication configuration file. If you do not use the **FILE** parameter, the ASNCLP attempts to use the `anservers.ini` file in the current directory, if that file exists. Use the **FILE** parameter with different files that are customized for different environments.

**Example**

To list the Q Capture schema on server SAMPLE:

```
LIST CAPTURE SCHEMA SERVER DBALIAS SAMPLE ID id1 PASSWORD "passwd!";
```

---

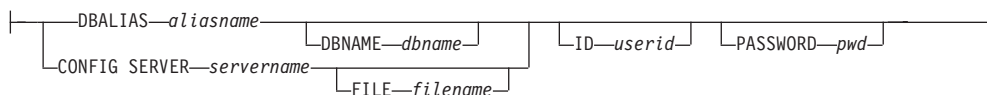
## LIST CAPTURE SCHEMA command

You can use the **LIST CAPTURE SCHEMA** command to list the Q Capture schemas for a specified server.

**Syntax**



**dbparms-clause:**



**Parameters**

dbparms-clause:

**SERVER**

Specifies the server that contains the schemas to be listed.

**DBALIAS** *aliasname*

Specifies the database alias name of a z/OS subsystem or Linux, UNIX, or Windows database as cataloged on the DB2 from which the ASNCLP is invoked.

**DBNAME** *zosdbname*

**z/OS** Specifies the z/OS database name. This is a logical z/OS database name, as created on a z/OS subsystem.

**ID** *userid*

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

**PASSWORD** *pwd*

Specifies the password to use for connections.

**CONFIG SERVER** *servername*

**Classic sources:** Specifies which server configuration settings from the Classic replication configuration file that the ASNCLP should use to connect to the Classic server.

**FILE** *filename*

Specifies the complete path and file name to the replication configuration file.

If you do not use the **FILE** parameter, the ASNCLP attempts to use the `asnservers.ini` file in the current directory, if that file exists. Use the **FILE** parameter with different files that are customized for different environments.

### Example

To list the Q Capture schema on server SAMPLE:

```
LIST CAPTURE SCHEMA SERVER DBALIAS SAMPLE ID id1 PASSWORD "passwd!";
```

---

## LIST SCHEMASUB command

The **LIST SCHEMASUB** command generates a list of all DB2 schemas on a source or target server for which a schema-level subscription is defined. It also shows whether the schema-level subscriptions are for unidirectional, bidirectional, or peer-to-peer replication.

### Syntax

►►—LIST SCHEMASUB—◄◄

### Example

To list all of the schema-level subscriptions on the SAMPLE database, which is part of a bidirectional configuration:

```
SET BIDI NODE 1 SERVER SAMPLE;  
LIST SCHEMASUB;
```

### Command output

Assume that the schema-level subscription on SAMPLE was created using the expression MSROSS%. The schema-level subscriptions on SAMPLE are MSROSS1, MSROSS2, and MSROSS3. Here is the output of the LIST SCHEMASUB command:

```
Schemas Subscription type  
MSROSS1 U  
MSROSS2 B  
MSROSS3 B
```

One schema-level subscription exists on the server for unidirectional replication, with two for bidirectional replication.

---

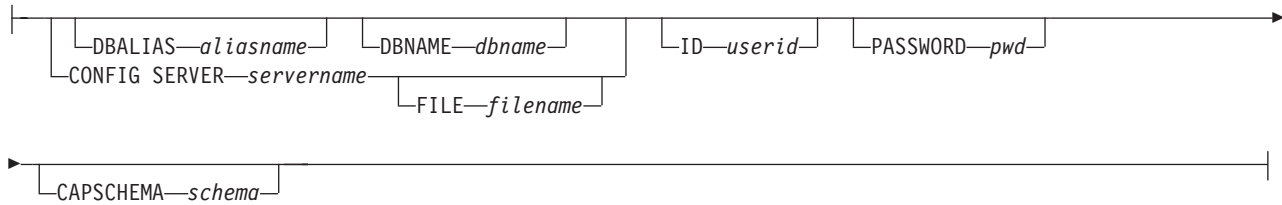
## LOAD DONE command

Use the **LOAD DONE** command to inform the Q Capture program or the Classic capture components that the target table is loaded. Issue the **LOAD DONE** command only if you are doing a manual load. If the Q Apply program is doing the load, this signal is not necessary.

### Syntax

►►—LOAD DONE— QSUB— [SUBNAME—*subname*—] [FOR SUBNAME LIKE—"*%text%*"—] [CAP SERVER OPTIONS— *classic-opt-clause* —] ◄◄

## classic-opt-clause:



## 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%"
```

### **CAP SERVER OPTIONS**

Specifies additional parameters when you issue the **LOAD DONE** command in immediate execution mode.

classic-opt-clause: These parameters only work with Classic sources.

### **DBALIAS** *aliasname*

Specifies the database alias name of a z/OS subsystem or Linux, UNIX, or Windows database as cataloged on the DB2 from which the ASNCLP is invoked.

### **DBNAME** *zosdbname*

**z/OS** Specifies the z/OS database name. This is a logical z/OS database name, as created on a z/OS subsystem.

### **ID** *userid*

Specifies the user ID to use to connect to the source database.

### **PASSWORD** *pwd*

Specifies the password to use to connect to the source database.

### **CAPSCHEMA** *schema*

Specifies the schema of the control tables of the Classic source.

### **CONFIG SERVER** *servername*

Specifies which server configuration settings from the Classic replication configuration file that the ASNCLP should use to connect to the Classic data source.

### **FILE** *filename*

Specifies the Classic replication server that the ASNCLP program connects to. The server name must match the name that is entered in the Classic replication configuration file. If you do not use the **FILE** parameter, the ASNCLP attempts to use the `asnservers.ini` file in the current directory, if that file exists.

## Example

To signal the Q Capture program or the capture components that the target table for the Q subscription EMPLOYEE0001 is loaded:

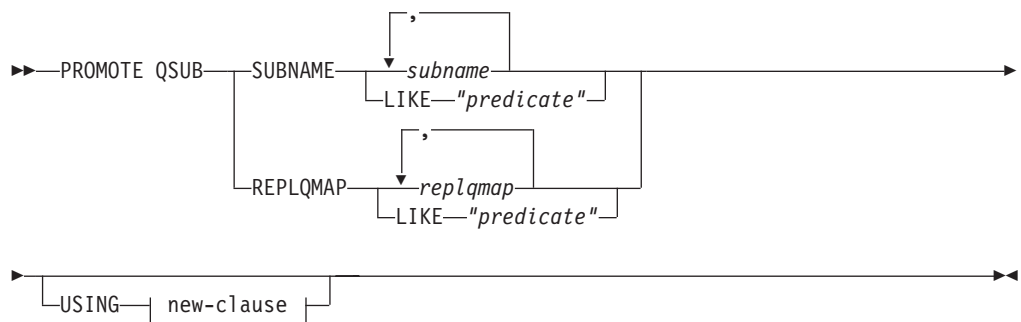
```
LOAD DONE QSUB SUBNAME EMPLOYEE0001
```

## PROMOTE QSUB command (unidirectional replication)

Use the **PROMOTE QSUB** command to build an ASNCLP script with statements that you can use to create unidirectional Q subscriptions on another set of servers. Promoting is useful for copying Q subscriptions from test systems to production systems or migrating Q subscriptions from one server to another.

You can also use this command to customize some of the properties of the promoted Q subscription, including the name of the Q Capture and Q Apply schemas and the replication queue map that is used. The promoted values of properties that cannot be customized are taken from the source Q subscription. If you need to change other properties, you can use the **ALTER QSUB** command after promoting the Q subscription to change the properties for the new Q subscription.

### Syntax



#### new-clause::



### Parameters

#### **SUBNAME** *subname*

Specifies one or more Q subscription names to promote. Use commas to separate multiple Q subscription names.

#### **LIKE** *"predicate"*

Specifies a list of Q subscription names to promote that match the predicate.

#### **REPLQMAP** *replqmap*

Specifies one or more replication queue maps. Use commas to separate multiple map names. All Q subscriptions that use the specified map or maps are promoted.

#### **LIKE** *"predicate"*

Specifies a list replication queue maps that match the predicate. All Q subscriptions that use the matching maps are promoted.

new-clause:

#### **USING SOURCE SCHEMA** *schema*

Specifies the source table schema.

**USING TARGET SCHEMA** *schema*

Specifies the target table schema. If the schema is not specified, the promoted definition uses the schema of the current target table.

**USING REPLQMAP** *newqmap*

Specifies the name of a new replication queue map that you want to use for the promoted Q subscriptions.

**Example - using a replication queue map**

To promote all Q subscriptions that use the replication queue map qmap1:

```
PROMOTE QSUB REPLQMAP "qmap1";
```

**Example - changing to a new replication queue map**

To promote all Q subscriptions that use the replication queue map qmap1 so that they use the queue map qmap2 instead:

```
PROMOTE QSUB REPLQMAP "qmap1" USING REPLQMAP "qmap2";
```

**PROMOTE REPLQMAP command**

Use the **PROMOTE REPLQMAP** command to promote one or more replication queue maps from one set of control tables to another.

If a single replication queue map is specified, you can also use this command to customize some of the properties of the promoted queue map, including the name of the replication queue map and name of the send queue. The promoted values of properties that cannot be customized are taken from the source replication queue map. If you need to change other properties, you can use the **ALTER REPLQMAP** command after promoting the replication queue map to change the properties for the new replication queue map.

**Syntax**

```

▶▶ PROMOTE REPLQMAP NAME replqmap
                        |
                        | USING new-clause
                        |
                        | LIKE "predicate"
                        |
▶▶

```

**new-clause:**

```

|
| REPLQMAP
|   |
|   | NAME newqmap
|   |
|   | map-options
|   |
|

```

**map-options:**

```

|
| ADMINQ newadminq
| SENDQ newsendq
| RECVQ newrecvq
|

```

**Parameters****NAME** *replqmap*

Specifies the name of an existing replication queue map to be promoted.

**LIKE** *"predicate"*

Specifies a list of replication queue map names that match the predicate. All replication queue map names that match the predicate will be promoted.

new-clause

**REPLQMAP**

Specifies new property values for the promoted replication queue map.

**NAME** *newqmap*

Specifies a new name for the replication queue map. If you do not specify a new name, then the current replication queue map name is used.

map-options

**ADMINQ** *newadminq*

Specifies a new name for the administration queue. If you do not specify a new name, then the current administration queue name is used.

**SENDQ** *newsendq*

Specifies a new name for the send queue. If you do not specify a new name, then the current send queue name is used.

**RECVQ** *newrecvq*

Specifies a new name for the receive queue. If you do not specify a new name, then the current receive queue name is used.

## Example 1

To promote replication queue maps that match the name "SAMPLE\_ASN":

```
PROMOTE REPLQMAP LIKE "SAMPLE_ASN%";
```

## Example 2

To promote replication queue map REPLQMAP2 and customize several properties of the promoted version of that queue map, so that the new replication queue map name is REPLQMAPNEW2, the new administration queue name is adminqnew2, the new send queue name is sendqnew2, and the new receive queue name is recvqnew2:

```
PROMOTE REPLQMAP NAME REPLQMAP2 USING REPLQMAP NAME REPLQMAPNEW2  
ADMINQ "adminqnew2" SENDQ "sendqnew2" RECVQ "recvqnew2";
```

---

## REINIT SCHEMASUB command

Use the **REINIT SCHEMASUB** command to generate a script that prompts the Q Capture program to reread any changes to the options for a schema-level subscription. You can also use this command to prompt Q Capture to reread changes to the table-level Q subscriptions within the schema.

### Syntax

```
►►—REINIT SCHEMASUB—schema_sub_name—

|          |
|----------|
| ALL      |
| NEW ONLY |

—►►
```

### Parameters

**ALL**

Specify to reinitialize a schema-level subscription and all of the table-level Q

subscriptions that belong to it. The command generates a SQL script to insert a REINIT\_SCHEMASUB signal into the IBMQREP\_SIGNAL table at the Q Capture server for the schema-level Q subscription, and REINIT\_SUB signals for the table-level Q subscriptions. You can use the SET RUN SCRIPT NOW option to immediately insert the signals.

**Note:** Reinitializing a schema-level subscription updates the options that are used for creating table-level Q subscriptions within the schema. However, the changes are used only for newly created tables. To update options for existing table-level Q subscriptions, you must reinitialize these Q subscriptions.

#### NEW ONLY

Specify to reinitialize only the schema-level subscription.

### Example

To reinitialize the schema-level Q subscription `schemasub1` and all of its table-level Q subscriptions, and also reinitialize only the schema-level subscription `schemasub2`:

```
SET BIDI NODE 1 SERVER DBALIAS SAMPLE1;
SET BIDI NODE 2 SERVER DBALIAS SAMPLE2;

SET RUN SCRIPT NOW STOP ON SQL ERROR ON;

REINIT SCHEMASUB schemasub1 ALL;
REINIT SCHEMASUB schemasub2 NEW ONLY;
```

---

## 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 applieschema ◀◀
```

### Parameters

#### TO DEFAULT

Specify to set the Q Apply schema to ASN and to reset any previous **SET APPLY SCHEMA** commands.

*applieschema*

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 schema of the source control tables for all task commands. For Classic sources, you can use only the default Q Capture schema, ASN.

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. If you are using a DB2 source, the schema can be any valid DB2 schema name. If you are using a Classic source, you must use the DEFAULT schema.

#### 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 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 {ALWAYS | NEVER | KEEP NICKNAME} ▶▶
```

```

▶▶ SET DROP {TARGET | CONTROL TABLES} TABLESPACE {WHEN EMPTY | NEVER}

```

```

▶▶ SET DROP TEMPORAL HISTORY SUB {NEVER | ALWAYS}

```

## Parameters

### TARGET

Specifies if you want to drop the target tables when you delete the Q subscription.

#### ALWAYS

Always drop the target table.

#### NEVER

Never drop the target table.

#### KEEP NICKNAME

Keep the nickname that is associated with the target table. The ASNCLP program uses this option with federated targets but ignores it for regular DB2 targets. Normally, the target nickname for federated targets is always dropped. This option can be helpful if you want to keep the nickname in case the Q subscription will be recreated later.

### 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.

### SET DROP TEMPORAL HISTORY SUB

Specifies whether you want to drop the Q subscription for a history table that is associated with a temporal table on DB2 10 for z/OS or later when you drop the Q subscription for the temporal table.

#### ALWAYS

Always drop the Q subscription for the history table.

#### NEVER

Never drop the Q subscription for the history table.

## Example 1

To always drop the target table when the Q subscription is deleted:

```
SET DROP TARGET ALWAYS;
```

## Example 2

To never drop the table space for the control tables when the control tables are dropped.

```
SET DROP CONTROL TABLES TABLESPACE NEVER;
```

## Example 3

To never drop the target table when the Q subscription is deleted and to also keep the nickname for the target table:

```
SET DROP TARGET NEVER KEEP NICKNAME;
```

---

## 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"—WITH DETAILS—▶▶
```

### Parameters

*"logfilename"*

Specifies the output log file name. The default log file name is `qreplmsg.log`.

#### WITH DETAILS

Creates an additional log file with just error messages for the run along with the "Explanation" and "User response" sections for each message. The name of the additional file is *logfilename\_1*. The contents of the standard log file remain unchanged.

### 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 1

To name the output log file `qmaplog.err` for creating replication queue maps:

```
SET LOG "qmaplog.err";
```

## Example 2

To specify that the ASNCLP program create its regular log file and an additional log file with error messages and the "Explanation" and "User response" sections for each message:

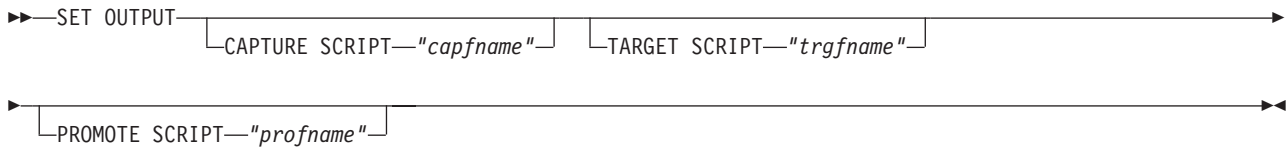
```
SET LOG "qrepllog.err" WITH DETAILS;
```

---

## 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, or the ASNCLP commands needed to promote a replication environment. You cannot use this command with non-relational sources.

## Syntax



## 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.

### PROMOTE SCRIPT "profname"

Specifies the output file name for the ASNCLP commands generated by **PROMOTE** statements. If the file name is not specified, the default file created is named qrepl\_asnc1p.in.

## 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"
```

---

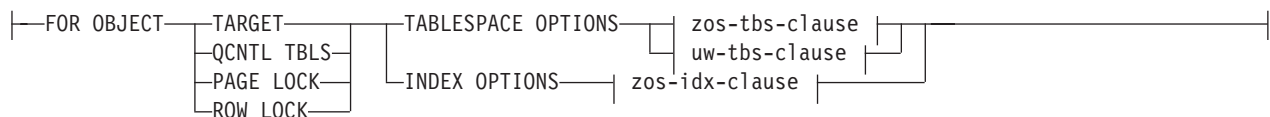
## SET PROFILE command

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.

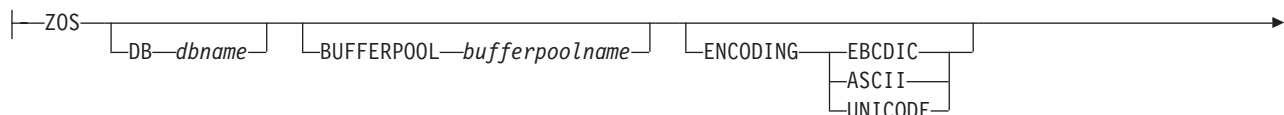
## Syntax



### prof-clause:

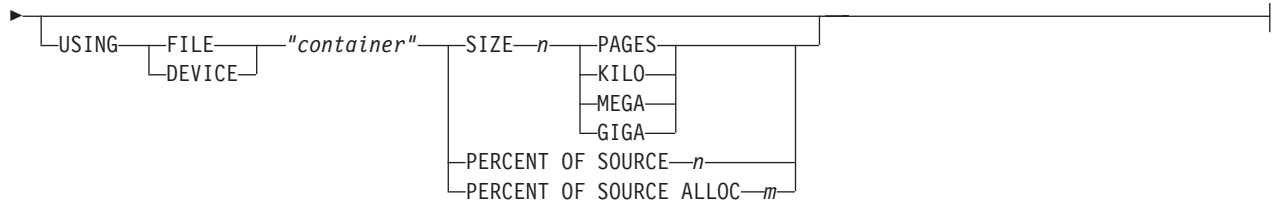
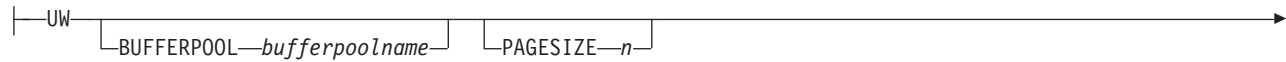


### zos-tbs-clause:

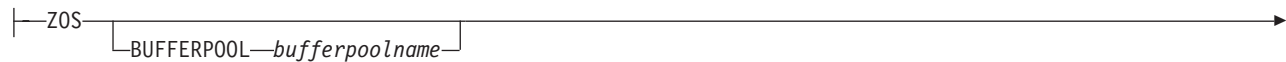




**uw-tbs-clause:**



**zos-idx-clause:**



**priqty-clause:**



**secqty-clause:**



**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 that is used by the Q Capture program, Q Apply program, or both. You cannot use this command with non-relational sources.

## Syntax

```
▶▶ SET QMANAGER "qmgrname" FOR { CAPTURE SCHEMA | APPLY SCHEMA | NODE number } ▶▶
```

### Parameters

*"qmgrname"*

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.

#### NODE

Specifies one server in a multidirectional configuration. If this keyword is specified, the ASNCLP program uses the same value for *"qmgrname"* for both the Q Capture server and Q Apply server.

### 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 QM1 for both the Q Capture and Q Apply programs on a server that is used in bidirectional or peer-to-peer replication:

```
SET QMANAGER FOR NODE 1 "QM1";
```

---

## SET RUN SCRIPT command

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. You cannot use the LATER parameter with non-relational sources.

### Syntax

```
▶▶ SET RUN SCRIPT { LATER | NOW | STOP | generate-sql-opts (ON | OFF) | ON SQL ERROR } ▶▶
```

**generate-sql-opts:**



## Parameters

### LATER

Specify to run the SQL scripts at a later time. You cannot use this parameter with Classic sources. Use this option if you want to verify your script before you run it. You can also use this option if you want to create SQL script files on one operating system, but run them on another.

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.

### NOW

Specify to automatically execute the SQL scripts.

### STOP ON SQL ERROR

Specifies whether the ASNCLP continues to process commands in the ASNCLP script file and statements in the generated SQL script file after one of the following errors:

- **ASNCLP script file:** An error while checking to predict whether the SQL statement to be generated will cause an SQL error. For example, a Q subscription cannot be defined in the control tables unless the control tables exist first.
- **Generated SQL script file:** An SQL error while running the SQL statements.

### ON (default)

Specify if you want the ASNCLP to stop processing commands in the ASNCLP script, and stop processing SQL statements in the generated SQL script, when the first validity check fails or SQL statement fails. If the error occurs while the ASNCLP is running the SQL script, previous SQL statements that are related to the task command with an error are rolled back.

**Note:** If the source scripts run correctly and the SQL statements in the scripts were committed but the target scripts have an SQL error, only the target scripts are rolled back. The committed source statements are not rolled back.

### OFF

Specify to process the ASNCLP commands and run all of the SQL statements, regardless of errors. You cannot use this parameter with Classic sources.

For a more complete explanation of how the ASNCLP responds to errors depending on this and other SET RUN SCRIPT options, see How the ASNCLP handles errors while processing scripts.

### GENERATE SQL FOR EXISTING

Specifies whether to generate SQL when ASNCLP encounters errors because of duplicate (already existing) objects when processing **CREATE** commands. This option has no effect on **DROP** commands.

**NO** The ASNCLP program does not generate SQL to create objects that already exist. This is the default.

**YES**

The ASNCLP program continues to generate SQL statements even if it encounters existing object errors for the following commands:

**CREATE CONTROL TABLES**

Another set of control tables already exist under the same schema or table spaces are specified to be created but they already exist.

**CREATE REPLQMAP**

Another replication queue map with the same name already exists.

**CREATE QSUB**

Another Q subscription with the same name already exists, a target table already exists but the option in the **CREATE QSUB** command is to create the target table, the target table already exists but the option to create the table space was specified, or a unique index with the same name already exists.

## 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 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 might 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 SCRIPT NOW be used.

Figure 2 on page 155 shows these dependencies for Q replication to a relational source. This figure does not apply to non-DB2 sources.

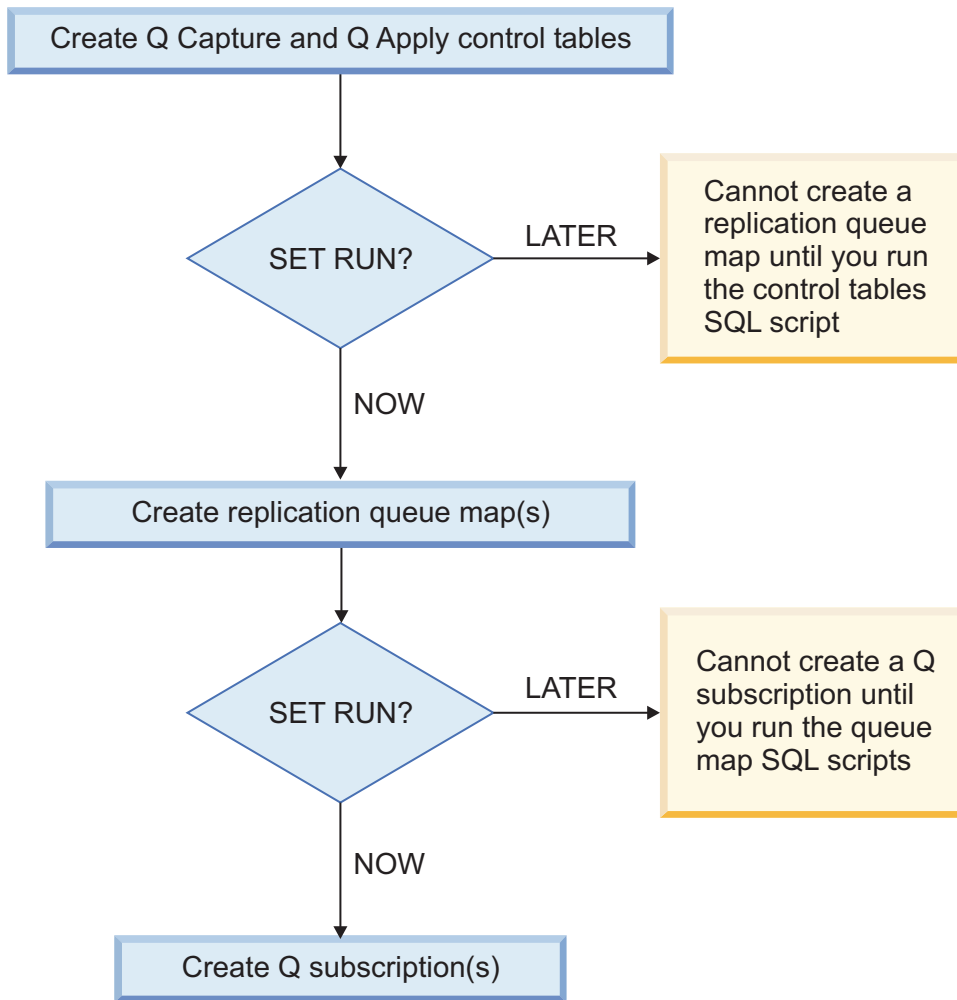


Figure 2. Dependencies between ASNCLP commands for Q replication from a DB2 source. This diagram shows the dependencies between ASNCLP CREATE commands that are used to set up Q replication. It assumes all objects use the default schema of ASN. The dependencies for Q Capture controls tables, publishing queue maps, and publications that are used in event publishing are the same.

### Example - Run immediately and stop on errors

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
```

### Example - Create SQL script and ignore errors when creating existing objects

To generate the SQL scripts instead of running them immediately, and to continue generating SQL when creating objects that already exist:

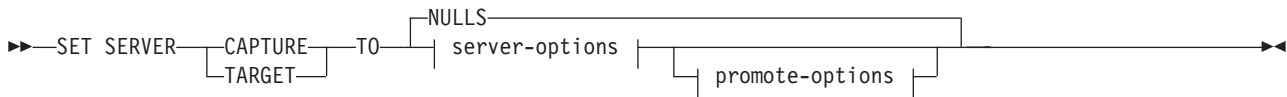
```
SET RUN SCRIPT LATER GENERATE SQL FOR EXISTING YES
```

---

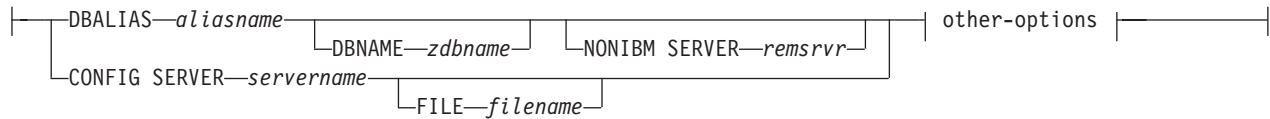
## SET SERVER command

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



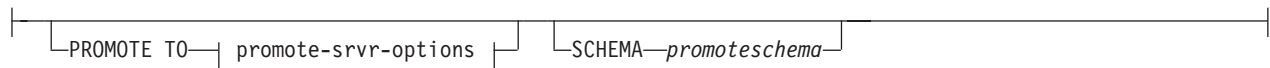
### server-options:



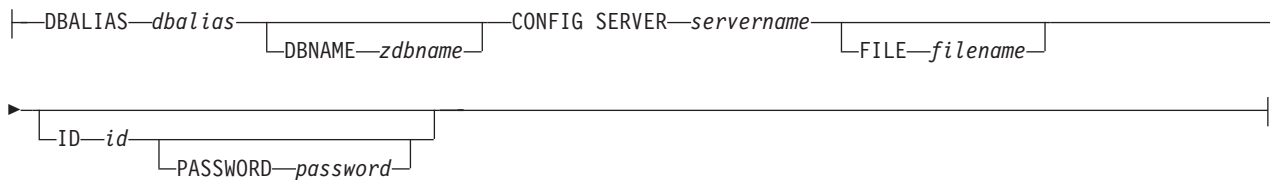
### other-options:



### promote-options:



### promote-srvr-options:



## Parameters

### CAPTURE

Specify to set the database as a Q Capture or Classic 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.

### server-options:

#### DBALIAS *aliasname*

Specifies the database alias name of a z/OS subsystem or Linux, UNIX, or Windows database as cataloged on the DB2 from which the ASNCLP is invoked.

**z/OS**

#### DBNAME *zdbname*

Specifies the database name.

**Note:** DBNAME is mandatory when ASNCLP is running on z/OS and the Q Capture or Q Apply server is on z/OS. DBNAME is a location name and is the name by which the DB2 database is known to local DB2 SQL applications. This name must match the name that was entered in the LOCATIONS column of the SYSIBM.LOCATIONS table in the CDB.

## NONIBM SERVER

**Federated targets:** The remote server name for a federated target. The target can be Informix, Microsoft SQL Server, Oracle, Sybase, or Teradata. This option is only valid for target servers.

### CONFIG SERVER *servername*

Specifies the Classic replication or Oracle source to which the ASNCLP program connects, or the DB2 source if the ASNCLP is running on UNIX System Services for z/OS. The server name must match the bracketed [NAME] field that is entered in the ASNCLP configuration file. You can also use this parameter to identify DB2 targets.

### FILE *filename*

Specifies the complete path and file name to the ASNCLP configuration file. If you do not use the **FILE** parameter, the ASNCLP program attempts to use the `asnservers.ini` file in the current directory, if that file exists.

other-options:

### 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. If you specify the user ID and do not specify the password, you will be prompted to enter the password. The password is hidden as you type.

**Note:** This keyword is not valid when the ASNCLP runs natively on z/OS because user authentication is handled through the communication database (CDB).

promote-options:

### PROMOTE TO

Promote the specified server definitions.

### SCHEMA *promoteschema*

Specifies the schema under which the server definitions will be promoted. If a schema is not specified, then the schema under which the current server definitions exist is used.

promote-srvr-options:

### DBALIAS *dbalias*

Specifies the database that will receive the promoted server definitions. If this clause is not specified and a **PROMOTE** command is included in the input file, then the **PROMOTE** command promotes the definitions to the current server.

**z/OS**

### DBNAME *zdbname*

Specifies the name of the database subsystem that will receive the promoted definitions.

### CONFIG SERVER *servername*

Specifies the replication target that the ASNCLP program connects to when

promoting definitions. The server name must match the bracketed [NAME] field that is entered in the ASNCLP configuration file.

**FILE** *filename*

Specifies the complete path and file name to the ASNCLP configuration file. If you do not use the **FILE** parameter, the ASNCLP program attempts to use the `asnservers.ini` file in the current directory, if that file exists.

**ID** *id*

Specifies the database ID where definitions will be promoted to. If not specified, the ASNCLP output script is generated without ID information.

**PASSWORD** *password*

Specifies the password to use to connect to the database. If not specified, the ASNCLP output script is generated without password information.

## Example

To set the Q Capture server to the database SAMPLE:

```
SET SERVER CAPTURE TO DBALIAS SAMPLE;
```

z/OS

## Example - z/OS

To set the target server to a z/OS database:

```
SET SERVER TARGET TO DBALIAS PRODUCTION DBNAME PRODUCTIONV9 ID id1 PASSWORD pwd1;
```

This example sets the z/OS database name to PRODUCTIONV9 and specifies the alias PRODUCTION. The user ID and password are explicitly specified because this command sets up a connection to a remote database.

## Example - federated targets

To set the target server to an Oracle database ORACLEDB:

```
SET SERVER TARGET TO DBALIAS ORADB NONIBM SERVER ORACLEDB;
```

## Example - Classic sources

Given a configuration file called `classic.ini` that contains the following information:

```
[classic1]  
Type=CLASSIC  
Data source=CACSAMP  
Host=9.30.155.156  
Port=8019
```

Use the following command to specify server `classic1` as the data server:

```
SET SERVER CAPTURE TO CONFIG SERVER classic1 FILE classic.ini ID id1 PASSWORD pwd1;
```

## Example - password prompting

To set the Capture control server and specify only the user ID in the command:

```
SET SERVER CAPTURE TO DBALIAS SAMPLE ID DB2ADMIN;
```

You are prompted to enter the password. If you are running the commands from an input file in batch mode, the program waits for you to enter the password

before the program processes the next commands. Your text is hidden when you type.

### Example - promoting configurations

To set the existing server containing definitions to be promoted and set the new server that will receive these promoted configurations:

```
SET SERVER CAPTURE TO DBALIAS SAMPLE ID id1 PASSWORD "p1wd"  
PROMOTE TO DBALIAS SAMPLE1 ID id1 PASSWORD SCHEMA ASN;
```

---

## 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:

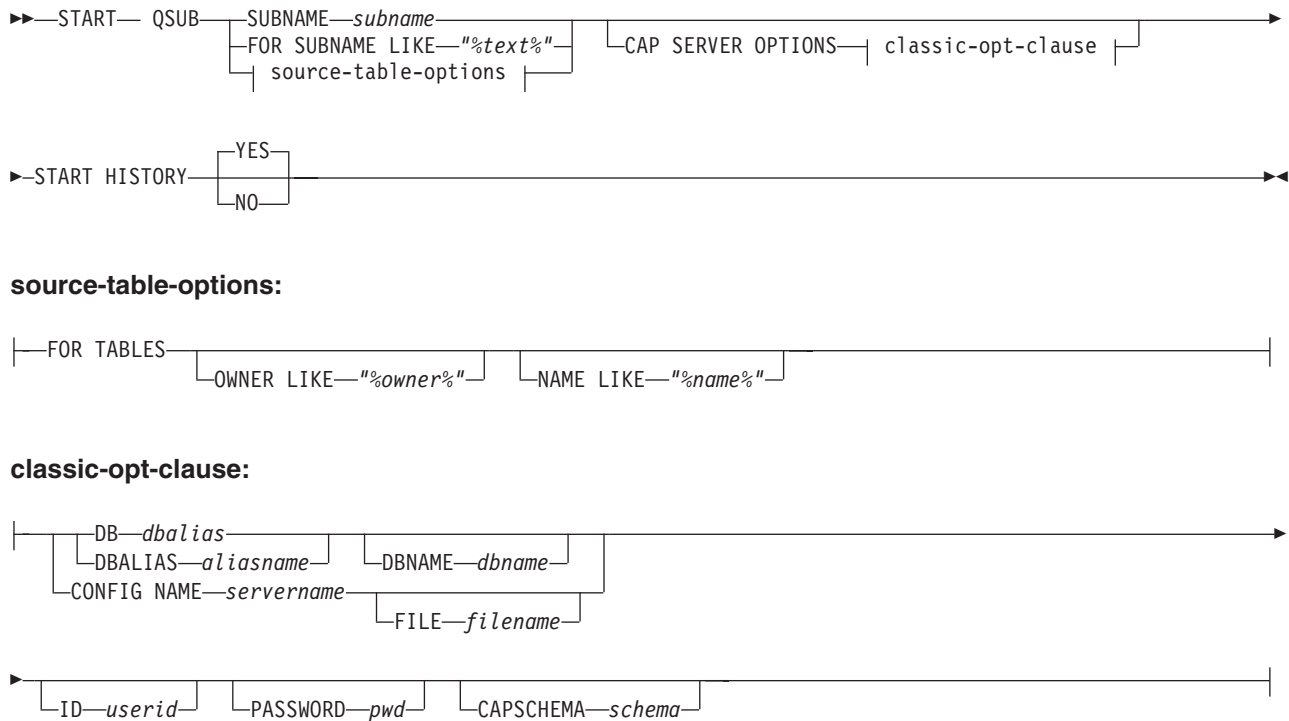
```
SHOW SET ENV
```

---

## START QSUB command

Use the **START QSUB** command to signal the Q Capture program or the Classic capture components to start one or more Q subscriptions.

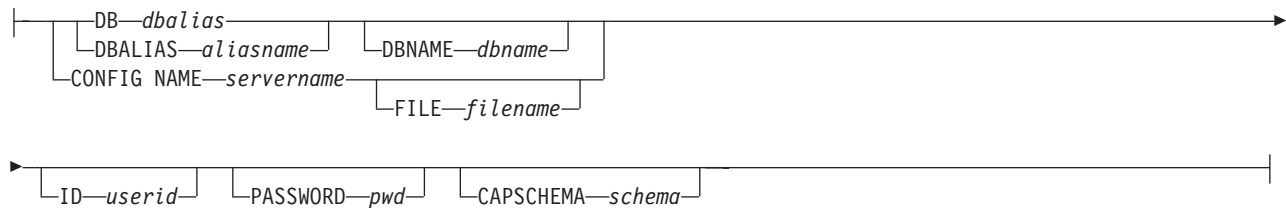
## Syntax



### source-table-options:



### classic-opt-clause:



## 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%"
```

### source-table-options

#### FOR TABLES

Use this clause to specify multiple schemas, multiple source tables, or both for which to start Q subscriptions.

#### OWNER LIKE "%owner%"

Specifies a single database schema or schema pattern that uses the percentage sign (%) as a wild card.

#### NAME LIKE "%name%"

Specifies a single table name or table-naming pattern that uses the percentage sign (%) as a wild card.

### classic-opt-clause:

These parameters work only with Classic sources. If you have already specified these parameters in a previous **SET SERVER** command, you do not have to specify them again in this command.



**DB** *dbalias*

Specifies the database alias name of a z/OS subsystem or Linux, UNIX, or Windows database as cataloged on the DB2 from which the ASNCLP is invoked. This keyword is deprecated.

**DBALIAS** *aliasname*

Specifies the database alias name of a z/OS subsystem or Linux, UNIX, or Windows database as cataloged on the DB2 from which the ASNCLP is invoked.

**DBNAME** *zosdbname*

**z/OS** Specifies the z/OS database name. This is a logical z/OS database name, as created on a z/OS subsystem.

**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.

**CAPSCHEMA** *schema*

Specifies the schema of the control tables.

**CONFIG NAME** *servername*

Specifies which server configuration settings from the Classic replication configuration file that the ASNCLP uses to connect to the Classic data server.

**FILE** *filename*

Specifies the complete path and file name to the Classic replication configuration file. If you do not use the **FILE** parameter, the ASNCLP attempts to use the `asnservers.ini` file in the current directory, if that file exists.

**START HISTORY**

Specifies whether you want to start the Q subscription for the history table when you start the Q subscription for the associated temporal table on DB2 10 for z/OS or later.

**YES (default)**

Start the Q subscription for the history table.

**NO** Do not start the Q subscription for the history table.

**Usage notes**

The CAP SERVER OPTIONS parameter overrides any settings that you specified in a previous SET command.

**Example: Classic replication with server information in START QSUB command**

To start a Q subscription from a Classic source by specifying server information in the START QSUB command:

```
START QSUB SUBNAME sub1 CAP SERVER OPTIONS CONFIG NAME classic1
FILE asnservers.ini ID id1 PASSWORD passwd1;
```

**Example: Classic replication with server information in SET SERVER command**

To start a Q subscription from a Classic source by specifying server information in a separate SET command:

```
SET SERVER CAPTURE CONFIG SERVER NAME classic1
FILE asnservers.ini ID id1 PASSWORD passwd1;
START QSUB SUBNAME sub1;
```

### Example: Starting multiple Q subscriptions on multiple servers based on schema pattern

To start all of the bidirectional Q subscriptions on the SAMPLE1 and SAMPLE2 servers that are defined under schemas that start with "AIRUKU":

```
SET BIDI NODE 1 SERVER DBALIAS SAMPLE1;
SET BIDI NODE 2 SERVER DBALIAS SAMPLE2;

SET RUN SCRIPT NOW STOP ON SQL ERROR ON;

START QSUB FOR TABLES OWNER LIKE "AIRUKU%";
```

---

## START SCHEMASUB command

Use the **START SCHEMASUB** command to generate a script that prompts the Q Capture program to start capturing DDL changes for a schema-level subscription. You can also use this command to prompt Q Capture to start capturing DML changes for the inactive and new table-level Q subscriptions within the schema.

### Syntax

```
▶▶—START SCHEMASUB—schema_sub_name—

|          |
|----------|
| ALL      |
| NEW ONLY |

—▶▶
```

### Parameters

#### ALL

Specify to start capturing DDL changes for a schema-level subscription and DML changes for all of inactive and new the table-level Q subscriptions that belong to it. The command generates a SQL script to insert a START\_SCHEMASUB signal into the IBMQREP\_SIGNAL table at the Q Capture server for the schema-level subscription, and CAPSTART signals for the table-level Q subscriptions. You can use the SET RUN SCRIPT NOW option to immediately insert the signals.

#### NEW ONLY

Specify to start only the schema-level subscription.

### Example

To start capturing DDL changes for the schema-level subscription schemasub1 and DML changes for all of its inactive and new table-level Q subscriptions, and to start capturing DDL only for the schema-level subscription schemasub2:

```
SET BIDI NODE 1 SERVER DBALIAS SAMPLE1;
SET BIDI NODE 2 SERVER DBALIAS SAMPLE2;

SET RUN SCRIPT NOW STOP ON SQL ERROR ON;

START SCHEMASUB schemasub1 ALL;
START SCHEMASUB schemasub2 NEW ONLY;
```

## STOP QSUB command

Use the **STOP QSUB** command to signal the Q Capture program or the Classic capture components to stop one or more Q subscriptions.

### Syntax

```
▶▶ STOP QSUB [SUBNAME subname | FOR SUBNAME LIKE "%text%" | source-table-options] [CAP SERVER OPTIONS | classic-opt-clause]
```

```
▶ STOP HISTORY [YES | NO]
```

#### source-table-options:

```
| FOR TABLES [OWNER LIKE "%owner%" | NAME LIKE "%name%"]
```

#### classic-opt-clause:

```
| [DB dbalias | DBALIAS aliasname | DBNAME dbname | ID userid | PASSWORD pwd | CONFIG SERVER servername | FILE filename]
```

```
▶ [CAPSCHEMA schema]
```

### 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%"
```

#### source-table-options

##### **FOR TABLES**

Use this clause to specify multiple schemas, multiple source tables, or both for which to stop Q subscriptions.

##### **OWNER LIKE** "%owner%"

Specifies a single database schema or schema pattern that uses the percentage sign (%) as a wild card.

##### **NAME LIKE** "%name%"

Specifies a single table name or table-naming pattern that uses the percentage sign (%) as a wild card.

#### classic-opt-clause:

These parameters work only with Classic sources. If you have already specified these parameters in a previous **SET SERVER** command, you do not have to specify them again in this command.

**DB** *dbalias*

Specifies the database alias name of a z/OS subsystem or Linux, UNIX, or Windows database as cataloged on the DB2 from which the ASNCLP is invoked. This keyword is deprecated.

**DBALIAS** *aliasname*

Specifies the database alias name of a z/OS subsystem or Linux, UNIX, or Windows database as cataloged on the DB2 from which the ASNCLP is invoked.

**DBNAME** *zosdbname*

**z/OS** Specifies the z/OS database name. This is a logical z/OS database name, as created on a z/OS subsystem.

**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.

**CAPSCHEMA** *schema*

Specifies the schema of the control tables.

**CONFIG SERVER** *servername*

Specifies which server configuration settings from the Classic replication configuration file that the ASNCLP uses to connect to the Classic data source.

**FILE** *filename*

Specifies the complete path and file name to the Classic replication configuration file. If you do not use the **FILE** parameter, the ASNCLP attempts to use the `asnservers.ini` file in the current directory, if that file exists.

**STOP HISTORY**

Specifies whether you want to stop the Q subscription for the history table when you stop the Q subscription for the associated temporal table on DB2 10 for z/OS or later.

**YES (default)**

Stop the Q subscription for the history table.

**NO** Do not stop the Q subscription for the history table.

## Usage notes

The `CAP SERVER OPTIONS` parameter overrides any settings that you specified in a previous `SET` command.

## Example

To stop a Q subscription:

```
STOP QSUB SUBNAME EMPLOYEE0001;
```

## Example: Stopping multiple Q subscriptions on multiple servers based on schema pattern

To stop all of the bidirectional Q subscriptions on the `SAMPLE1` and `SAMPLE2` servers that are defined under schemas that start with "AIRUKU":

```

SET BIDI NODE 1 SERVER DBALIAS SAMPLE1;
SET BIDI NODE 2 SERVER DBALIAS SAMPLE2;

SET RUN SCRIPT NOW STOP ON SQL ERROR ON;

START QSUB FOR TABLES OWNER LIKE "AIRUKU%";

```

---

## STOP SCHEMASUB command

Use the **STOP SCHEMASUB** command to generate a script that prompts the Q Capture program to stop capturing DDL changes for a schema-level subscription. You can also use this command to prompt Q Capture to stop capturing DML changes for the table-level Q subscriptions within the schema.

### Syntax

```

▶▶—STOP SCHEMASUB—schema_sub_name—ALL—NEW ONLY—▶▶

```

### Parameters

#### ALL

Specify to stop capturing DDL changes for a schema-level subscription and DML changes for all of the table-level Q subscriptions that belong to it. The command generates a SQL script to insert a STOP\_SCHEMASUB signal into the IBMQREP\_SIGNAL table at the Q Capture server for the schema-level subscription, and CAPSTOP signals for the table-level Q subscriptions. You can use the SET RUN SCRIPT NOW option to immediately insert the signals.

#### NEW ONLY

Specify to stop only the schema-level subscription.

### Example

To stop capturing DDL changes for the schema-level subscription `schemasub1` and DML changes for all of its table-level Q subscriptions, and also to stop capturing DDL for only the schema-level subscription `schemasub2`:

```

SET BIDI NODE 1 SERVER DBALIAS SAMPLE1;
SET BIDI NODE 2 SERVER DBALIAS SAMPLE2;

SET RUN SCRIPT NOW STOP ON SQL ERROR ON;

STOP SCHEMASUB schemasub1 ALL;
STOP SCHEMASUB schemasub2 NEW ONLY;

```

---

## 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
```





## Chapter 5. 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 71
- “Sample ASNCLP scripts for setting up peer-to-peer Q Replication (two servers)” on page 72
- “Sample ASNCLP scripts for setting up peer-to-peer Q Replication (three servers)” on page 74

**Note:** All of the commands for Q Replication require that you set the environment with the ASNCLP SESSION SET TO Q REPLICATION command.

Table 4 lists the ASNCLP commands for Event Publishing and links to topics that describe each command.

Table 4. ASNCLP commands for multidirectional Q Replication

Description	Command
Deprecated commands	<ul style="list-style-type: none"> <li>• “DROP SUBTYPE command (bidirectional replication)” on page 171</li> <li>• “DROP SUBTYPE command (peer-to-peer replication)” on page 172</li> <li>• “LOAD MULTIDIR REPL SCRIPT command (multidirectional Q Replication)” on page 172</li> <li>• “SET MULTIDIR SCHEMA command (multidirectional Q Replication)” on page 174</li> <li>• “SET SERVER command (multidirectional Q Replication)” on page 174</li> <li>• “SET TABLES command (multidirectional Q Replication)” on page 175</li> </ul>
Change a Q subscription for bidirectional replication	“ALTER QSUB command (bidirectional replication)” on page 179
Change a Q subscription for peer-to-peer replication	“ALTER QSUB command (peer-to-peer replication)” on page 182
Change a replication queue map	ALTER REPLQMAP command
Establish a session for Q Replication	ASNCLP SESSION SET TO command
Create the control tables for the Q Capture and Q Apply programs	CREATE CONTROL TABLES FOR command
Create a Q subscription for bidirectional replication	“CREATE QSUB command (bidirectional replication)” on page 190
Create a Q subscription for peer-to-peer replication	“CREATE QSUB command (peer-to-peer replication)” on page 197
Create a schema-level subscription	“CREATE SCHEMASUB command” on page 124

Table 4. ASNCLP commands for multidirectional Q Replication (continued)

Description	Command
Create a profile for table-level Q subscriptions	“CREATE SUBSCRIPTION OPTIONS command” on page 127
Create a replication queue map	CREATE REPLQMAP command
Drop the control tables for the Q Capture and Q Apply programs	DROP CONTROL TABLES ON command
Delete a replication queue map	DROP REPLQMAP command
Delete a schema-level subscription	“DROP SCHEMASUB command” on page 133
Delete a profile for table-level Q subscriptions	“DROP SUBSCRIPTION OPTIONS command” on page 133
Delete the subgroup that you set by using the SET SUBGROUP command.	“DROP SUBGROUP command (multidirectional Q Replication)” on page 215
Delete a Q subscription for bidirectional replication	“DROP SUBTYPE command (bidirectional replication)” on page 171
Delete a Q subscription for peer-to-peer replication between two servers	“DROP SUBTYPE command (peer-to-peer replication)” on page 172
List Q Apply schemas	LIST APPLY SCHEMA command
List Q Capture schemas	LIST CAPTURE SCHEMA command
List schema-level subscriptions	“LIST SCHEMASUB command” on page 139
Signal that a manual load of the target table is complete	LOAD DONE command
Promote Q subscriptions	PROMOTE Q SUB command (multidirectional replication)
Promote replication queue maps	PROMOTE REPLQMAP command
Reinitialize a schema-level subscription	“REINIT SCHEMASUB command” on page 143
Set the Q Apply schema for all task commands	SET APPLY SCHEMA command
Set the Q Capture schema for all task commands	SET CAPTURE SCHEMA command
Connect the servers that are used for bidirectional or peer-to-peer replication.	“SET CONNECTION command (multidirectional Q Replication)” on page 227
Specify whether the ASNCLP will enforce matching constraints between the source and target tables.	“SET ENFORCE MATCHING CONSTRAINTS command (multidirectional Q Replication)” on page 228
Define the log file for the ASNCLP program	SET LOG command
Define output files that contain SQL scripts for multidirectional replication	“SET OUTPUT command (multidirectional Q Replication)” on page 229
Specify custom parameters for database objects to be created implicitly	SET PROFILE command
Set the WebSphere MQ queue manager	SET QMANAGER command
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 236
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 240
Specify the tables that participate in a bidirectional or peer-to-peer configuration	“SET TABLES command (multidirectional Q Replication)” on page 175
Enable and disable the trace for the ASNCLP commands	SET TRACE command
Display the environment set during the session	SHOW SET ENV command

Table 4. ASNCLP commands for multidirectional Q Replication (continued)

Description	Command
Start a Q subscription	START QSUB command
Start a schema-level subscription	"START SCHEMASUB command" on page 162
Stop a Q subscription	STOP QSUB command
Stop a schema-level subscription	"STOP SCHEMASUB command" on page 165
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
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

## Deprecated commands

Some of the commands for multidirectional replication were deprecated at Version 9.7 Fix Pack 2. Replacement commands that are simpler to use were added to the ASNCLP program.

### DROP SUBTYPE command (bidirectional replication)

Use the **DROP SUBTYPE** command to delete both bidirectional Q subscriptions for a single logical table.

This command is deprecated. Starting with Version 10.1 on Linux, UNIX, and Windows, you can use the **DROP QSUB** command to delete bidirectional Q subscriptions.

#### Syntax

►► `DROP SUBTYPE B QSUBS` ◄◄

#### Parameters

##### SUBTYPE 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.

#### 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 BIDI NODE 1 SERVER DBALIAS SAMPLE;
```

```
SET BIDI NODE 2 SERVER DBALIAS 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.

This command is deprecated. Starting with Version 10.1 on Linux, UNIX, and Windows, you can use the **DROP QSUB** command to delete peer-to-peer Q subscriptions.

### 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.

### 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 PEER NODE 1 SERVER DBALIAS SAMPLE;
SET PEER NODE 2 SERVER DBALIAS SAMPLE2;
SET PEER NODE 3 SERVER DBALIAS SAMPLE3;
SET REFERENCE TABLE USING SCHEMA "SAMPLE".GRAY USES TABLE GRAY.STAFF;
DROP SUBTYPE P QSUBS;
```

## LOAD MULTIDIR REPL SCRIPT command (multidirectional Q Replication)

The **LOAD MULTIDIR REPL SCRIPT** command is deprecated. Instead, you can directly invoke ASNCLP program scripts for setting up bidirectional and peer-to-peer replication by using the **ASNCLP -f filepath** command.

**Note:** When the ASNCLP program runs natively on z/OS, the **LOAD MULTIDIR REPL SCRIPT** command is invalid. Instead, you include DD statements in the JCL that reference the location of the ASNCLP input scripts for setting up bidirectional or peer-to-peer 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 MULTIDIR SCHEMA command (multidirectional Q Replication)

The **SET MULTIDIR SCHEMA** command is deprecated. Use the SET BIDI NODE or SET PEER NODE commands to specify the shared server and schema of the Q Capture and Q Apply control tables for bidirectional or peer-to-peer replication.

**Important:** When the ASNCLP program runs natively on z/OS, the SET MULTIDIR SCHEMA syntax is invalid for bidirectional or peer-to-peer replication. You must use SET BIDI NODE or SET PEER NODE.

### 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.

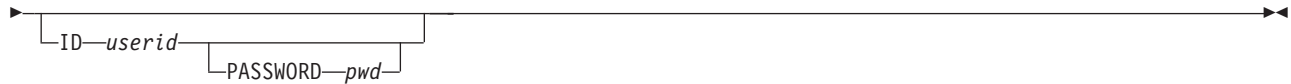
## SET SERVER command (multidirectional Q Replication)

The **SET SERVER** command for bidirectional or peer-to-peer replication is deprecated. Use the SET BIDI NODE or SET PEER NODE command instead to specify the server that contains both Q Capture and Q Apply control tables in a multidirectional configuration.

**Important:** When the ASNCLP program runs natively on z/OS, the SET SERVER syntax is invalid for bidirectional or peer-to-peer replication. You must use SET BIDI NODE or SET PEER NODE.

### Syntax

►►—SET SERVER—  
 ┌───┴───┐ ┌───┴───┐ ┌───┴───┐ ┌───┴───┐ ┌───┴───┐ ┌───┴───┐ ┌───┴───┐ ┌───┴───┐ ┌───┴───┐ ┌───┴───┐  
 CAPTURE ──┐ ┌───┴───┐ ┌───┴───┐ ┌───┴───┐ ┌───┴───┐ ┌───┴───┐ ┌───┴───┐ ┌───┴───┐ ┌───┴───┐ ┌───┴───┐  
 TARGET ───┘ ┌───┴───┐ ┌───┴───┐ ┌───┴───┐ ┌───┴───┐ ┌───┴───┐ ┌───┴───┐ ┌───┴───┐ ┌───┴───┐ ┌───┴───┐  
 MULTIDIR ───┘ ┌───┴───┐ ┌───┴───┐ ┌───┴───┐ ┌───┴───┐ ┌───┴───┐ ┌───┴───┐ ┌───┴───┐ ┌───┴───┐ ┌───┴───┐  
 └───┬───┘ └───┬───┘ └───┬───┘ └───┬───┘ └───┬───┘ └───┬───┘ └───┬───┘ └───┬───┘ └───┬───┘ └───┬───┘  
 TO ───┬───┘ ───┬───┘ ───┬───┘ ───┬───┘ ───┬───┘ ───┬───┘ ───┬───┘ ───┬───┘ ───┬───┘ ───┬───┘  
 ┌───┴───┐ ┌───┴───┐ ┌───┴───┐ ┌───┴───┐ ┌───┴───┐ ┌───┴───┐ ┌───┴───┐ ┌───┴───┐ ┌───┴───┐ ┌───┴───┐  
 NULLS ───┬───┘ ───┬───┘ ───┬───┘ ───┬───┘ ───┬───┘ ───┬───┘ ───┬───┘ ───┬───┘ ───┬───┘ ───┬───┘  
 DBALIAS—*aliasname* ───┬───┘ ───┬───┘ ───┬───┘ ───┬───┘ ───┬───┘ ───┬───┘ ───┬───┘ ───┬───┘ ───┬───┘ ───┬───┘  
 ┌───┴───┐ ┌───┴───┐ ┌───┴───┐ ┌───┴───┐ ┌───┴───┐ ┌───┴───┐ ┌───┴───┐ ┌───┴───┐ ┌───┴───┐ ┌───┴───┐  
 DBNAME—*zdbname* ───┬───┘ ───┬───┘ ───┬───┘ ───┬───┘ ───┬───┘ ───┬───┘ ───┬───┘ ───┬───┘ ───┬───┘ ───┬───┘  
 ┌───┴───┐ ┌───┴───┐ ┌───┴───┐ ┌───┴───┐ ┌───┴───┐ ┌───┴───┐ ┌───┴───┐ ┌───┴───┐ ┌───┴───┐ ┌───┴───┐  
 CONFIG SERVER—*servername* ───┬───┘ ───┬───┘ ───┬───┘ ───┬───┘ ───┬───┘ ───┬───┘ ───┬───┘ ───┬───┘ ───┬───┘ ───┬───┘  
 ┌───┴───┐ ┌───┴───┐ ┌───┴───┐ ┌───┴───┐ ┌───┴───┐ ┌───┴───┐ ┌───┴───┐ ┌───┴───┐ ┌───┴───┐ ┌───┴───┐  
 FILE—*filename* ───┬───┘ ───┬───┘ ───┬───┘ ───┬───┘ ───┬───┘ ───┬───┘ ───┬───┘ ───┬───┘ ───┬───┘ ───┬───┘



## 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.

### DBALIAS *aliasname*

Specifies the database alias name of a z/OS subsystem or Linux, UNIX, or Windows database as cataloged on the DB2 from which the ASNCLP is invoked.

### DBNAME *zosdbname*

**z/OS** Specifies the z/OS database name. This is a logical z/OS database name, as created on a z/OS subsystem.

### CONFIG SERVER *servername*

Specifies that you are using a file to provide connection information for the server. The server name must match the bracketed [NAME] field that is entered in the ASNCLP configuration file.

### FILE *filename*

Specifies the complete path and file name to the ASNCLP configuration file. If you do not use the FILE parameter, the ASNCLP program attempts to use the asnservers.ini file in the current directory, if that file exists.

### 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. If you specify the user ID and do not specify the password, you are prompted to enter the password.

## 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).

This command is deprecated. Starting with Version 10.1 on Linux, UNIX, and Windows, you can use the FOR TABLES keywords in the CREATE QSUB command instead.

## Syntax

```

▶▶ SET TABLES [option-1] [option-2]

```

### option-1:

```

|--(--server.schema.table_owner.table_name--, [server.schema.table_owner.table_name] )--|

```

### option-2:

```

|--(--NODE--number--| node-option |)--|

```

### node-option:

```

|--|
| | source_name
| | | source_owner
| | | SRC OWNER LIKE "--predicate1--"
| | | | SRC NAME LIKE "--predicate2--"
| | | SRC NAME LIKE "--predicate--"
| | | SRC ALL
|--|

```

## Parameters

### option-1:

#### *server*

Specifies the name of the database or subsystem that contains the table.

#### *schema*

Specifies the schema of the control tables in which this table is specified as a source or target.

#### *table\_owner*

Specifies the schema of the table.

#### *table\_name*

Specifies the name of the table.

### option-2:

#### **NODE**

Specifies one server in a bidirectional replication configuration. The ASNCLP uses the provided predicate to search for tables to include in the bidirectional Q subscription.

### node-option

#### *source\_owner.source\_name*

Specifies the schema and name of a table to include in the Q subscription.

#### **SRC OWNER LIKE**

Specify to choose all tables with a schema that matches the expression in the



LIKE statement. If OWNER is not specified, all eligible tables on the server that is identified by the NODE keyword are selected.

#### **SRC NAME LIKE**

Specify to choose all tables with a name that matches the expression in the LIKE statement.

#### **SRC ALL**

Specify to choose all tables that exist on the server that is identified by the NODE keyword. For DB2 sources, this excludes catalog views.

### **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;
```

### **Example 3**

This example simplifies the task of creating a large number of bidirectional Q subscriptions at the same time. The ASNCLP program looks at one server in the bidirectional configuration, identified as NODE 1, for all tables with an owner that begins with "DSN8710" and a name that includes the letters "EMP." A Q subscription is created for any table that matches this predicate.

```
SET TABLES (NODE 1 SRC OWNER LIKE "DSN8710%" SRC NAME LIKE "%EMP%");
CREATE QSUB SUBTYPE B;
```

## ALTER ADD COLUMN command (multidirectional replication)

Use the **ALTER ADD COLUMN** command to add a column to a Q subscription for multidirectional replication.

### Syntax

```
▶▶ALTER ADD COLUMN USING SIGNAL—(—colname—)—————▶▶
▶QSUB—subname—USING REPMAP—qmapname—————▶▶
    WITH BEFORE IMAGE—[PREFIX—'single_character'—]
▶SOURCE—table_owner.table_name—————▶▶
```

### Parameters

*colname*

Specifies one or more columns (separated by a comma) to add to the definition of the active Q subscription.

**QSUB** *subname*

Specifies the name of the Q subscription.

**WITH BEFORE IMAGE**

Specifies that the before-image value of each added column will be replicated.

**PREFIX** '*single\_character*'

Specifies a single-character prefix for each before-image column. If you do not specify a prefix, the default of X is used. If this prefix generates invalid names, other letters will be used beginning with Y until valid names are generated.

**USING REPMAP** *qmapname*

Specifies the name of the replication queue map that is used by the Q subscription.

**SOURCE** *table\_owner.table\_name*

Specifies that the columns are added to all of the Q subscriptions and publications for the source table.

### Usage notes

- The column needs to exist in the source table already and should not be part of any existing Q subscription.
- The Q subscription 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 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 attributes with SQL.

- A maximum of 20 columns can be inserted into the statement.
- The option to specify a different name for the target table column is not supported for multidirectional replication.

### Example 1

To add the columns PHONE and ADDRESS to the EMPLOYEE0001 Q subscription:

```
ASNCLP SET SESSION TO Q REPLICATION;
SET SERVER CAPTURE TO DB ALIAS BIDISERVER1;
SET SERVER TARGET TO DB ALIAS BIDISERVER2;
ALTER ADD COLUMN USING SIGNAL (PHONE, ADDRESS) QSUB EMPLOYEE0001
USING REPMAP BIDISERVER1_ASN_TO_BIDISERVER2_ASN;
```

### Example 2

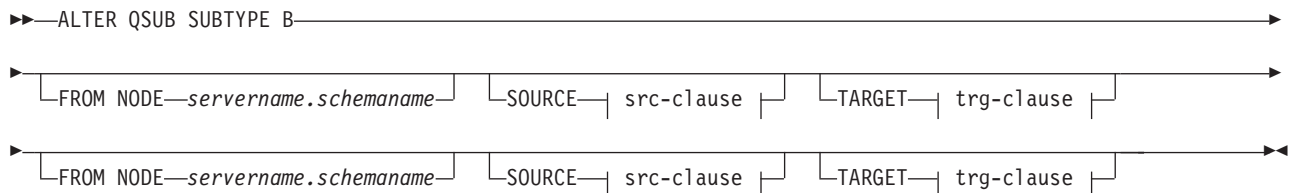
To add the PHONE, ADDRESS, and EMAIL columns to all Q subscriptions and publications for the EMPLOYEE table.

```
ASNCLP SET SESSION TO Q REPLICATION;
SET SERVER CAPTURE TO DB ALIAS P2PSERVER1;
SET SERVER TARGET TO DB ALIAS P2PSERVER2;
ALTER ADD COLUMN USING SIGNAL (PHONE, ADDRESS, EMAIL) SOURCE DB2ADMIN.EMPLOYEE;
```

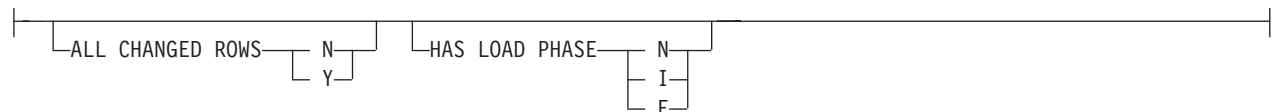
## ALTER QSUB command (bidirectional replication)

Use the **ALTER QSUB** command to change the properties of one or both bidirectional Q subscriptions for a single logical table.

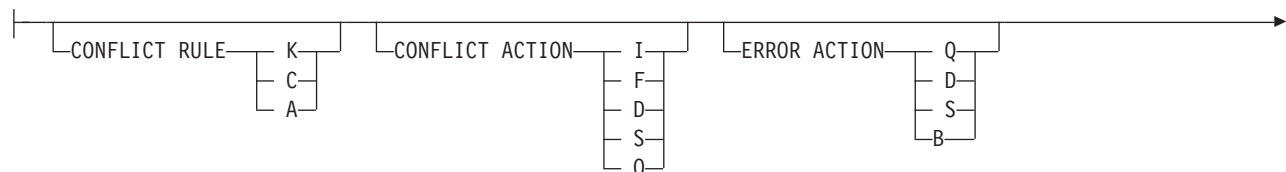
### Syntax

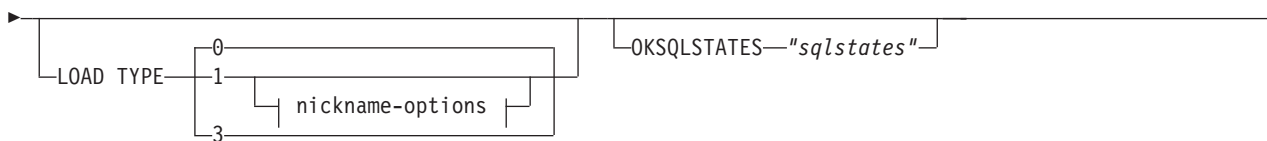


#### src-clause:



#### trg-clause:





## Parameters

### SUBTYPE B

Specifies bidirectional Q subscriptions.

### FROM NODE *servername.schemaname*

Identifies one of the two bidirectional Q subscriptions by specifying the server and schema of its source table.

src-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.

trg-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.
- B** When an error occurs, spill change messages for the Q subscription to a temporary spill queue until you use the **resumesub** parameter of the **MODIFY** or **asnqacmd** command to prompt Q Apply to begin applying the messages.

**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.
- 1** Use LOAD from CURSOR only. Specify this option if the source and target servers are on z/OS.

**Note:** If the Q Apply program is at Version 9.7 Fix Pack 4 or newer, you do not need to provide nickname information for this load option unless the Q subscription includes XML columns. Q Apply calls LOAD from CURSOR by specifying a cataloged DB2 alias for the source database instead of by using a nickname. You must include the DB2 alias in a password file that is created by the asnpwd utility.

- 3** Use EXPORT and LOAD only.

nickname-options:

**NICKNAME**

Specifies an existing 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 owner and name of an existing nickname.

**NAMING PREFIX *prefix***

Use these keywords if you are creating multiple Q subscriptions and nicknames already exist for LOAD from CURSOR. The variable *prefix* specifies a character string that is used in naming all of the nicknames, and that the ASNCLP program can use to find the nicknames. For example, if you had 10 source tables named HR.T1 through HR.T10 and 10 nicknames that reference these tables named HR.SRCNKT1 through HR.SRCNKT10, you could use the string SRCNK to enable the ASNCLP program to find the nicknames and use them in the Q subscription definitions.

**NEW NICKNAME RMT SERVERNAME *srvname***

Specifies the name of the remote server if the ASNCLP program creates the nickname for loading.

**NAMING PREFIX *prefix***

Specifies a character string that the ASNCLP program can use to generate one or more new nicknames for loading.

## 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 |—| | |
|—| D |—|
|—| S |—|
|—| B |—|
|—LOAD TYPE—| 0 |—|
|—| 1 |—| nickname-options |—|
|—| 3 |—|
```

```
▶—OKSQLSTATES—"sqlstates"—▶
```

### 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.
- B** When an error occurs, spill change messages for the Q subscription to a temporary spill queue until you use the **resumesub** parameter of the **MODIFY** or **asnqacmd** command to prompt Q Apply to begin applying the messages.

**LOAD TYPE**

Specifies the utilities that the Q Apply program uses to load the target.

- 0** Choose the best type automatically.
- 1** Use LOAD from CURSOR only. Specify this option if the source and target servers are on z/OS.

**Note:** If the Q Apply program is at Version 9.7 Fix Pack 4 or newer, you do not need to provide nickname information for this load option unless the Q subscription includes XML columns. Q Apply calls LOAD from CURSOR by specifying a cataloged DB2 alias for the source database instead of by using a nickname. You must include the DB2 alias in a password file that is created by the asnpwd utility.

- 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.

nickname-options:

**NICKNAME**

Specifies an existing 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 owner and name of an existing nickname.

**NAMING PREFIX *prefix***

Use these keywords if you are creating multiple Q subscriptions and nicknames already exist for LOAD from CURSOR. The variable *prefix*

specifies a character string that is used in naming all of the nicknames, and that the ASNCLP program can use to find the nicknames. For example, if you had 10 source tables named HR.T1 through HR.T10 and 10 nicknames that reference these tables named HR.SRCNKT1 through HR.SRCNKT10, you could use the string SRCNK to enable the ASNCLP program to find the nicknames and use them in the Q subscription definitions.

**NEW NICKNAME RMT SERVERNAME *srvname***

Specifies the name of the remote server if the ASNCLP program creates the nickname for loading.

**NAMING PREFIX *prefix***

Specifies a character string that the ASNCLP program can use to generate one or more new nicknames for loading.

**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;
```

---

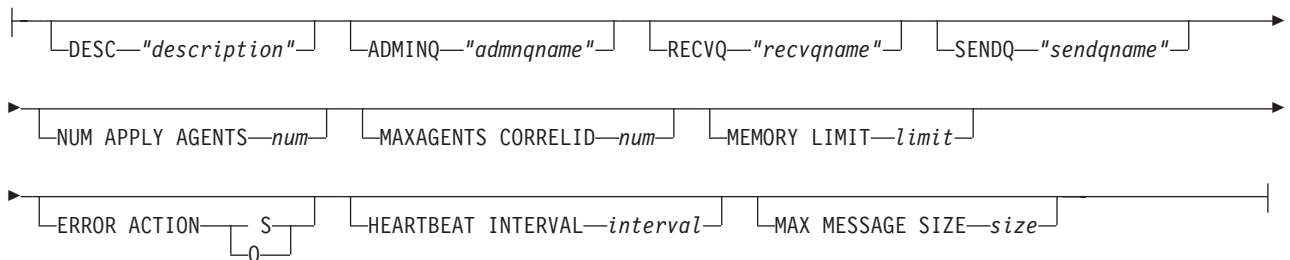
**ALTER REPLQMAP command**

Use the **ALTER REPLQMAP** command to customize attributes for an existing replication queue map. This command applies to Q replication and Classic replication.

**Syntax**

```
▶▶ ALTER REPLQMAP qmapname USING | options |
```

**options:**





## 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 or Classic capture components share a single queue manager with the Q Apply programs, they can share an administration queue.

**RECVQ** *"recvqname"*

Specifies the name of the receive queue that is used by the Q Apply program. See "Usage notes" on page 186 below.

**SENDQ** *"sendqname"*

Specifies the name of the send queue that is used by the Q Capture program or Classic capture components. See "Usage notes" on page 186 below.

**NUM APPLY AGENTS** *num*

Specifies the number of threads that are used to concurrently apply transactions from the specified receive queue.

**MAXAGENTS CORRELID***num*

**z/OS** Specifies that number of threads that are used for concurrently applying transactions from the specified receive queue with the same *correlation ID*. The correlation ID identifies all transactions that were started from the same z/OS job on the Q Capture server.

The value for the **MAXAGENTS CORRELID** parameter cannot be greater than the value for the **NUM APPLY AGENTS** parameter. If **MAXAGENTS\_CORRELID** value is 1, the transactions will be applied one at a time. If the value is greater than one, for example 4, four agents will apply transactions with the same correlation ID in parallel. If the value is 0, transactions are applied in parallel by using the total number of threads specified by the **NUM APPLY AGENTS** parameter.

**MEMORY LIMIT** *limit*

Specifies the maximum number of megabytes that are used per receive queue to buffer incoming transactions.

### ERROR ACTION

The action that the Q Capture program takes 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.

**S** The Q Capture program or the capture components stop when they detect an error on this queue.

**Q** The Q Capture program stops putting messages on any send queues that are in error and continues putting messages on other send queues. This value is not supported for Classic replication.

**HEARTBEAT INTERVAL** *interval*

Specifies the interval (in seconds) between heartbeat messages that are sent by the Q Capture program or Classic capture components 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 that is used for sending

messages over the send queue. The size of the buffer must not be larger than the maximum message length (MAXMSGL) that is defined for the send queue.

## Usage notes

You can only change the name of the send queue or receive queue if the queue map is not being used by any Q subscriptions yet. If the queue map is part of a Q subscription (active or inactive), you must take manual steps to change these queue names. See [com.ibm.swg.im.iis.repl.qrepl.doc/topics/iyrqrqmtchgqname.dita](http://com.ibm.swg.im.iis.repl.qrepl.doc/topics/iyrqrqmtchgqname.dita) for details.

## Example 1

The following command alters the SAMPLE\_ASN1\_TO\_TARGETDB\_ASN1 replication queue map, sets the threads to 4, and invalidates all of the Q subscriptions that use the send queue for this replication queue map if an error occurs.

```
ASNCLP SESSION SET TO Q REPLICATION;
SET SERVER CAPTURE TO SAMPLE;
SET CAPTURE SCHEMA ASN1;
SET SERVER TARGET TO TARGETDB
SET APPLY SCHEMA ASN1;
ALTER REPLQMAP SAMPLE_ASN1_TO_TARGETDB_ASN1 USING NUM APPLY AGENTS 4
      ERROR ACTION I;
```

## Example 2

The following command alters the CLASSIC\_ASN\_TO\_TARGETDB\_ASN1 replication queue map, sets the threads to 4, sets the maximum memory limit to 10 megabytes, stops the Classic capture components if an error occurs, sets the heartbeat interval to 4, and sets the maximum buffer size to 5 kilobytes.

```
ASNCLP SESSION SET TO Q REPLICATION;
SET OUTPUT TARGET SCRIPT "replapp.sql";
SET LOG "qmap.err";
SET SERVER CAPTURE TO CONFIG SERVER server1 FILE "asnservers.ini"
      ID username PASSWORD "passw1rd";
SET SERVER TARGET TO DB TARGETDB;
SET APPLY SCHEMA ASN1;
SET RUN SCRIPT NOW STOP ON SQL ERROR ON;
ALTER REPLQMAP CLASSIC_ASN_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

Use the **ASNCLP SESSION SET TO** command to establish an ASNCLP session for Q replication to either relational or Classic data sources.

### 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. Use this parameter when you are connecting to either relational or Classic sources.

## 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 MQ SCRIPT command

Use the **CREATE MQ SCRIPT** command to generate scripts for creating all of the WebSphere MQ objects that are needed for Q Replication and Event Publishing.

When you create control tables and queue maps, you can use the MQDEFAULTS keyword in these commands and the ASNCLP program will automatically use the default objects that are generated by CREATE MQ SCRIPT, bypassing the need to specify individual queue managers and queues.

## Syntax

```
▶▶ CREATE MQ SCRIPT [RUN NOW] CONFIG TYPE {U | E | B | P} mq-clause
```

### mq-clause:

```
MQSERVER—number—NAME—name | options
```

### options:

```
MQHOST—hostname [MQPORT—port_number] [QMANAGER—queue_manager] [QNAME_QUAL—qualifier]
```

## Parameters

### RUN NOW

Specifies that you want the ASNCLP program to run the generated WebSphere MQ script after it is created. The queue manager and ASNCLP program must be on the same system for you to use this option.

### CONFIG TYPE

Specifies the type of replication:

- U** Unidirectional
- E** Event publishing
- B** Bidirectional

**P** Peer-to-peer

mq-clause

#### **MQSERVER**

A number that identifies the Q Capture server, Q Apply server, or both for multidirectional replication. The numbers differ depending on the configuration type:

##### **Unidirectional**

Use 1 to represent the Q Capture server and 2 to represent the Q Apply server. Both numbers are required.

##### **Event publishing**

Use 1 to represent the Q Capture server.

##### **Bidirectional**

Use 1 to represent one server and its paired Q Capture and Q Apply, and the number 2 to represent the other server. Both numbers are required.

##### **Peer-to-peer**

Use 1, 2, 3, and so on, depending on the number of servers in the peer-to-peer environment. At least two server numbers are required.

#### **NAME**

The subsystem name or database alias of the Q Capture server, Q Apply server, or the combined Q Capture-Q Apply server for multidirectional replication.

options

#### **MQHOST**

The hostname or IP address of the system that contains the queue manager that will create the WebSphere MQ objects.

#### **MQPORT**

The port number that the channel listener monitors for incoming requests. If this keyword is not specified, the ASNCLP program uses the default WebSphere MQ port number 1414.

#### **QMANAGER**

The queue manager that will be created, and that will be used to create other WebSphere MQ objects. If this keyword is not specified, the value that was specified for the **NAME** keyword is used to name the queue manager.

#### **QNAME\_QUAL**

A qualifier that is used for the generated queue names. The default is ASN, which is the default Q Capture or Q Apply schema. This qualifier can help identify queues at the Q Capture system or Q Apply system.

### **Usage notes**

- **Linux UNIX Windows** The default file name for the generated script is `qrepl.server_name.mq`, where `server_name` is the server alias that was specified in the CREATE MQ SCRIPT command. The scripts are executable files in either the `.bat` or `.exe` format depending on whether the ASNCLP program runs on Windows or Linux-UNIX.
- **z/OS** If the ASNCLP program is running natively on z/OS, the output DD name for the generated script is OUTMQCAP, OUTMQTRG, and OUTMQX. The following lines must be included in the JCL:

```
//OUTMQCAP DD DSN=&SYSUID..ASNCLP.OUTNODE1,DISP=(NEW,CATLG,DELETE),
//          UNIT=SYSDA,SPACE=(TRK,(30,10))
//OUTMQTRG DD DSN=&SYSUID..ASNCLP.OUTNODE1,DISP=(NEW,CATLG,DELETE),
//          UNIT=SYSDA,SPACE=(TRK,(30,10))
```

The generated script will be wrapped to 80 characters per line. Comments are included with changes that need to be made for z/OS.

- You can specify the CREATE MQ SCRIPT command in the same input file as other ASNCLP commands, but this command does not use the server and schema information from any previous SET commands.
- If the Q Capture and Q Apply servers are on the same system, only one script file is generated that contains all the WebSphere MQ commands.

### Example 1

To generate a script that creates WebSphere MQ objects for event publishing:

```
CREATE MQ SCRIPT CONFIG TYPE E
MQSERVER 1 NAME SOURCEDB MQHOST "9.30.54.118" MQPORT "1414";
```

### Example 2

To generate a script that creates WebSphere MQ objects for a unidirectional replication configuration where the Q Capture and Q Apply servers are on the same system and share a local queue manager:

```
CREATE MQ SCRIPT CONFIG TYPE U
MQSERVER 1 NAME SOURCEDB MQHOST "9.30.54.118",
MQSERVER 2 NAME TARGETDB MQHOST "9.30.54.118";
```

### Example 3

To generate a script that creates WebSphere MQ objects for a unidirectional replication configuration where the source and target servers are remote with different queue managers (no MQPORT keywords are specified so the default ports of 1414 are used at each system):

```
CREATE MQ SCRIPT CONFIG TYPE U
MQSERVER 1 NAME SOURCEDB MQHOST "9.30.54.118",
MQSERVER 2 NAME TARGETDB MQHOST "9.30.54.119";
```

### Example 4

To generate a script that creates WebSphere MQ objects for a bidirectional replication configuration where the primary and standby servers are remote with different queue managers:

```
CREATE MQ SCRIPT CONFIG TYPE B
MQSERVER 1 NAME DB1 MQHOST "9.30.54.118",
MQSERVER 2 NAME DB2 MQHOST "9.30.54.119";
```

### Example 5

```
CREATE MQ SCRIPT CONFIG TYPE P
MQSERVER 1 NAME DB1 MQHOST "9.30.54.117",
MQSERVER 2 NAME DB2 MQHOST "9.30.54.118",
MQSERVER 3 NAME DB3 MQHOST "9.30.54.119";
```

## CREATE QSUB command (bidirectional replication)

Use the **CREATE QSUB** command to create one or more paired sets of Q subscriptions for bidirectional replication. For high availability and disaster recovery scenarios, you can use this command to create paired sets of Q subscriptions for every table in every schema on both servers.

### Syntax

```

▶▶ CREATE QSUB SUBTYPE B
    [ from-node-clause ]
    [ for-tables-clause ]
  
```

#### from-node-clause:

```

| FROM NODE servername.schemaname SOURCE [ source-clause ] [ TARGET ] [ target-clause ]
▶
| FROM NODE servername.schemaname SOURCE [ source-clause ] [ TARGET ] [ target-clause ]
  
```

#### for-tables-clause:

```

| FOR TABLES NODE node_number [ node-option ]
  
```

#### node-option:

```

| [ source_owner ] . source_name
| [ source-predicate ]
  
```

#### source-predicate:

```

| OWNER LIKE predicate1 [ NAME LIKE predicate2 ] [ TARGET EXISTS VALIDATE NO ]
| [ NAME LIKE predicate ]
| ALL
  
```

```

▶ OPTIONS options_list_name COLS [ ALL ]
| [ EXCLUDE ( column ) ]
| [ INCLUDE ( column ) ]
  
```

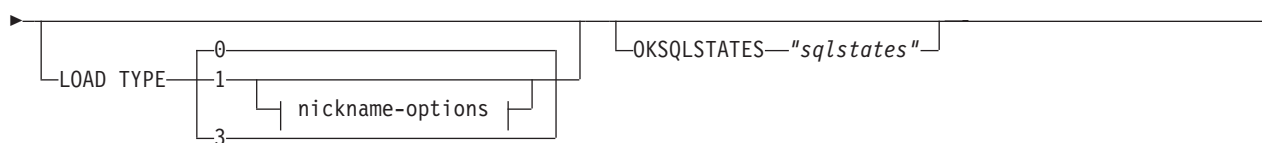
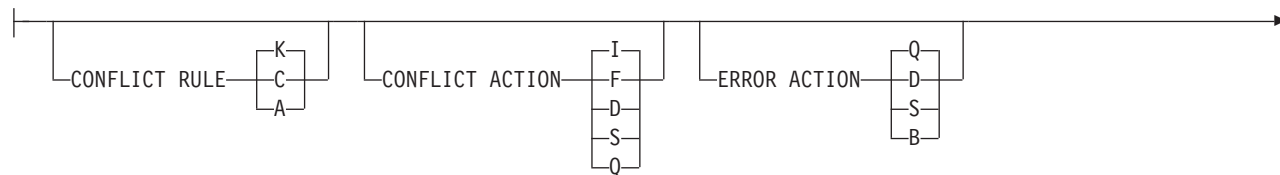
#### source-clause:

```

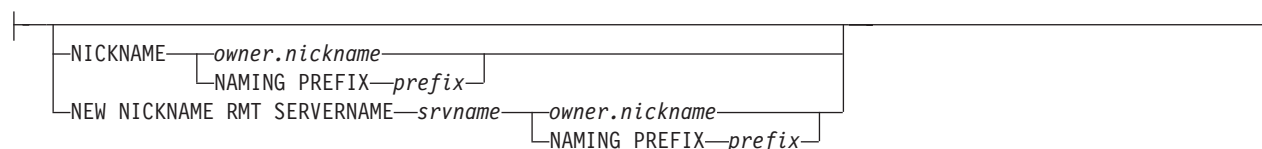
| ALL CHANGED ROWS [ N ] [ Y ]
| HAS LOAD PHASE [ I ] [ E ] [ N ]
| CAPTURE_LOAD [ W ] [ R ]
  
```



**target-clause:**



**nickname-options:**



**Parameters**

**SUBTYPE B**

Specifies bidirectional Q subscriptions.

**from-node-clause:**

**FROM NODE** *servername.schemaname*

Use this option if you are creating only one paired set of Q subscriptions. In the FROM NODE statement, you specify a server name and schema to identify the location of the logical table that is the source for the Q subscription. 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=I
- CONFLICT\_RULE=K
- ERROR\_ACTION=Q

**for-tables-clause:**

Use this clause to specify one or more logical tables for which to create paired sets of Q subscriptions. If you use this clause:

- A SET TABLES command is not needed before the CREATE QSUB command in the ASNCLP script.
- You use a Q subscription profile to specify options for the Q subscriptions by using the CREATE SUBSCRIPTION OPTIONS command. With FOR TABLES, you do not use the COLS keyword to specify a subset of columns or rows, and you do not use the source-clause and target-clause to specify Q subscription options.

**NODE**

Specifies which server in the bidirectional configuration should be used to locate the logical table on which the Q subscription is based.

node-option

Use these options to select one or more tables for which to create Q subscriptions.

*source\_owner*

Specifies the schema of a single source table.

*source\_name*

Specifies the name of a single source table.

source-predicate

Use these options to specify multiple source tables for which to create Q subscriptions.

**OWNER LIKE**

Specifies a single database schema or schema pattern that uses the percentage sign (%) as a wild card.

**NAME LIKE**

Specifies a single table name or table-naming pattern that uses the percentage sign (%) as a wild card.

**ALL**

Specifies that you want to create Q subscriptions for all schemas and all tables within those schemas.

**TARGET EXISTS VALIDATE NO**

Specifies that the target table exists and no validation is required for Q subscriptions. This option shortens processing time with very large tables.

**Important:** If you use these keywords, the ASNCLP program assumes that the target table matches exactly with the source table.

**OPTIONS**

Specifies the name of a profile (list of options) for creating Q subscriptions. You create the profile by using the CREATE SUBSCRIPTION OPTIONS command. The OPTIONS clause takes precedence over any previously provided SET PROFILE command.

**COLS**

Specifies columns to be selected.

**ALL**

Select all columns in the Q subscription. This is the default.

**EXCLUDE (column )**

Exclude the specified columns from the Q subscription. The specified columns are excluded from the target table when it is created. If the target



exists, the column names in the source table and target table must be the same. Excluded columns must be nullable, or if defined as NOT NULL then they must have a default value.

**INCLUDE (column )**

Include the specified columns in the Q subscription. If the target table is new, the table is created with the specified columns. The column names in the source table and target table must be the same.

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.

**I (default)**

Specifies an automatic load. The Q Apply program calls the LOAD from CURSOR utility 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 (by 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.

**N** No load phase at the target.

**CAPTURE\_LOAD**

Specifies the action that the Q Capture program takes when the recovery log shows that a load operation that uses the DB2 LOAD utility occurred at the source table. This parameter is only valid when the HAS LOAD PHASE option is I.

**W (default)**

Q Capture issues a warning message after the load completes.

**R** Q Capture stops and starts the Q subscription for the source table, prompting a load of the target table if one is specified for the Q subscription.

**START AUTOMATICALLY**

Specifies how to start the Q subscription, which is represented by the State column in the IBMQREP\_SUBS table. The State column controls whether the Q subscription is automatically started after starting or reinitializing the Q Capture program (subscription state N), or that the Q subscription must be started manually by inserting a command in the IBMQREP\_SIGNAL table (subscription state I).

**YES**

The Q subscription is started automatically (subscription state value of N). This is the default.

**NO** The Q subscription must be started manually (subscription state value of I).

target-clause:

#### CONFLICT RULE

- K** Check only key values.
- C** Check changed nonkey values and 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.
- B** When an error occurs, spill change messages for the Q subscription to a temporary spill queue until you use the `resumesub` parameter of the **MODIFY** or **asnqacmd** command to prompt Q Apply to begin applying the messages.

#### 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.
- 1** Use LOAD from CURSOR only. Specify this option if the source and target servers are on z/OS.

**Note:** If the Q Apply program is at Version 9.7 Fix Pack 4 or newer, you do not need to provide nickname information for this load option unless the Q subscription includes XML columns. Q Apply calls LOAD from CURSOR by specifying a cataloged DB2 alias for the source database instead of by using a nickname. You must include the DB2 alias in a password file that is created by the `asnpwd` utility.

- 3** Use EXPORT and LOAD only.

nickname-options:

#### NICKNAME

Specifies an existing 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 owner and name of an existing nickname.

**NAMING PREFIX** *prefix*

Use these keywords if you are creating multiple Q subscriptions and nicknames already exist for LOAD from CURSOR. The variable *prefix* specifies a character string that is used in naming all of the nicknames, and that the ASNCLP program can use to find the nicknames. For example, if you had 10 source tables named HR.T1 through HR.T10 and 10 nicknames that reference these tables named HR.SRCNKT1 through HR.SRCNKT10, you could use the string SRCNK to enable the ASNCLP program to find the nicknames and use them in the Q subscription definitions.

**NEW NICKNAME RMT SERVERNAME** *srvname*

Specifies the name of the remote server if the ASNCLP program creates the nickname for loading.

**NAMING PREFIX** *prefix*

Specifies a character string that the ASNCLP program can use to generate one or more new nicknames for loading.

**Usage notes**

Q subscriptions for tables that have referential integrity relationships with each other should be created at the same time (in the same CREATE QSUB command).

Table 5 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 automatically set the correct value for BEFORE\_VALUES and CHANGE\_COLS\_ONLY. Neither of these attributes can be set explicitly by using the administration tools.

Excluded columns from either source or target must be defined as nullable or not null with default columns.

Columns defined with data types ROWID and GENERATED ALWAYS are excluded automatically.

*Table 5. 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 C 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. The command generates statements to create a matching table at SAMPLE2.

```
SET SUBGROUP "bidirgroup"

SET BIDI NODE 1 SERVER DBALIAS SAMPLE SCHEMA RED;
SET BIDI NODE 2 SERVER DBALIAS SAMPLE2 SCHEMA 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 HAS LOAD PHASE I
TARGET CONFLICT RULE C CONFLICT ACTION F
FROM NODE SAMPLE2.BLUE SOURCE HAS LOAD PHASE I
TARGET CONFLICT RULE C CONFLICT ACTION I
```

### **Example: Subsetting columns - include**

The following command creates a bidirectional Q subscription that includes only columns c1, c2, c3, and c4:

```
CREATE QSUB SUBTYPE B COLS INCLUDE (C1,C2,C3,C4)
```

### **Example: Subsetting columns - exclude**

The following command creates a bidirectional Q subscription that excludes columns C1, C2, and C3:

```
CREATE QSUB SUBTYPE B SOURCE HAS LOAD PHASE I COLS EXCLUDE (C1,C2,C3)
```

### **Example: Using LOAD from CURSOR**

The following commands set the environment and then create a bidirectional Q subscription for a single table that specifies the LOAD from CURSOR utility (LOAD TYPE 1) over a nickname. Q Apply will create the nickname. The remote server name is SRCSVR1, the source table is HR.TABLE1, and the nickname that references the source table is HR.SRCNKTABLE1:

```
ASNCLP SESSION SET TO Q REPLICATION;
SET BIDI NODE 1 SERVER DBALIAS REDDB;
SET BIDI NODE 2 SERVER DBALIAS BLUEDB;

SET TABLES (REDDB.ASN.HR.TABLE1);

CREATE QSUB SUBTYPE B FROM NODE REDDB.ASN SOURCE HAS LOAD PHASE I
TARGET LOAD TYPE 1 NEW NICKNAME RMTSERVERNAME SRCSVR1 HR.SRCNKTABLE1;
```

### **Example: Multiple tables with LOAD from CURSOR**

The following commands set the environment and then create bidirectional Q subscriptions for all source tables with the schema "HR." The Q subscriptions

specify the LOAD from CURSOR utility over a nickname. The ASNCLP program will create a nickname for each source table and use the naming prefix "SRCNK" to generate the nickname names. The remote server name is SRCSVR1:

```
ASNCLP SESSION SET TO Q REPLICATION;
SET BIDI NODE 1 SERVER DBALIAS REDDB;
SET BIDI NODE 2 SERVER DBALIAS BLUEDB;

SET TABLES (NODE 1 SRC OWNER LIKE "HR%");

CREATE QSUB SUBTYPE B FROM NODE REDDB.ASN SOURCE HAS LOAD PHASE I
TARGET LOAD TYPE 1 NEW NICKNAME RMTSERVERNAME SRCSVR1 NAMING PREFIX SRCNK;
```

## LOAD from CURSOR with an existing nickname

The following commands set the environment and then create a bidirectional Q subscription for the source table HR.EMPLOYEE, specifying that the existing nickname HR.SOURCENICK should be used by the LOAD from CURSOR utility:

```
ASNCLP SESSION SET TO Q REPLICATION;
SET BIDI NODE 1 SERVER DBALIAS REDDB;
SET BIDI NODE 2 SERVER DBALIAS BLUEDB;

SET TABLES (SAMPLE.ASN.HR.EMPLOYEE);

CREATE QSUB SUBTYPE B FROM NODE REDDB.ASN SOURCE HAS LOAD PHASE I
TARGET LOAD TYPE 1 NICKNAME HR.SOURCENICK;
```

## Using FOR TABLES clause to create multiple Q subscriptions

The following example uses the CREATE SUBSCRIPTION OPTIONS command to create a profile, bidilist, for specifying options for multiple Q subscriptions. To use the profile, the FOR TABLES keywords are specified. The OWNER LIKE keywords prompt the ASNCLP program to create Q subscriptions for all tables within schemas on the SAMPLE1 server (NODE 1) that begin with the letters AIRUKU. Because FOR TABLES is used, no SET TABLES commands are required.

```
SET BIDI NODE 1 SERVER DBALIAS SAMPLE1;
SET BIDI NODE 2 SERVER DBALIAS SAMPLE2;

SET RUN SCRIPT NOW STOP ON SQL ERROR ON;

CREATE SUBSCRIPTION OPTIONS bidilist
HAS LOAD PHASE E
CONFLICT ACTION F;

CREATE QSUB SUBTYPE B FOR TABLES
(NODE 1 OWNER LIKE "AIRUKU%"
TARGET EXISTS VALIDATE NO
OPTIONS bidilist);
```

---

## CREATE QSUB command (peer-to-peer replication)

Use the **CREATE QSUB** command to create one or more paired sets of Q subscriptions for logical tables that participate in peer-to-peer replication.

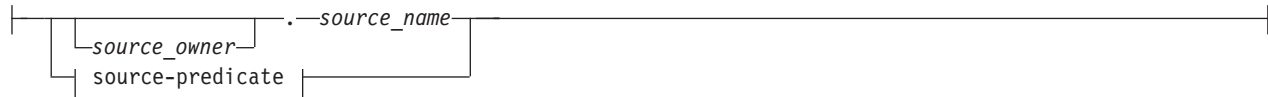
### Syntax

```
►►—CREATE QSUB—SUBTYPE P—┐
└─┬──SOURCE──┐ source-clause ┌──┬──TARGET──┐ target-clause ┌──┬──►
  └──┬──┬──┘ for-tables-clause └──┬──┬──┘
```

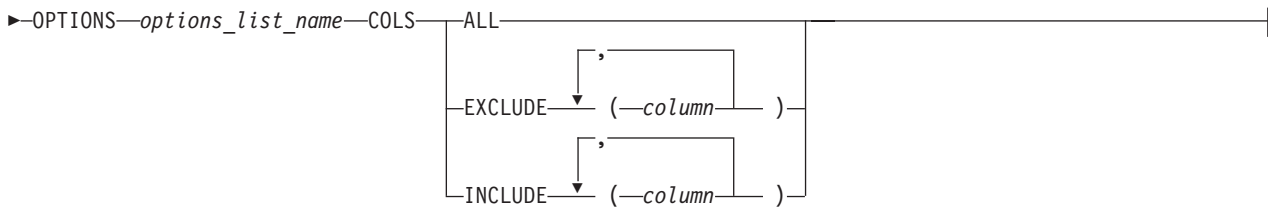
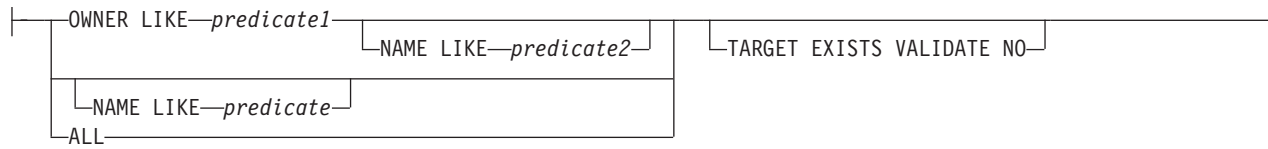
**for-tables-clause:**



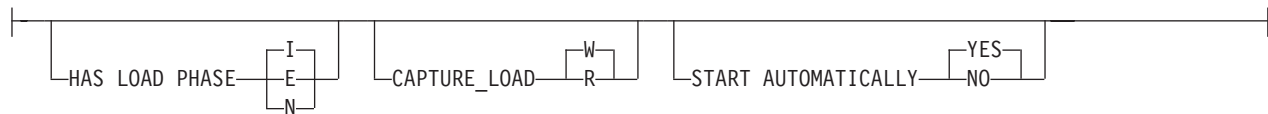
**node-option:**



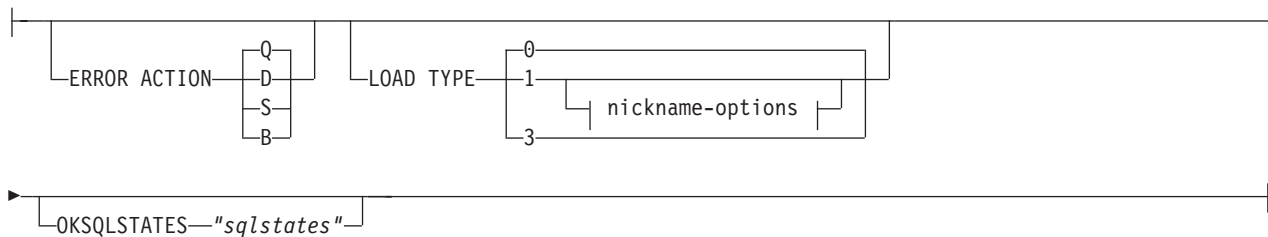
**source-predicate:**



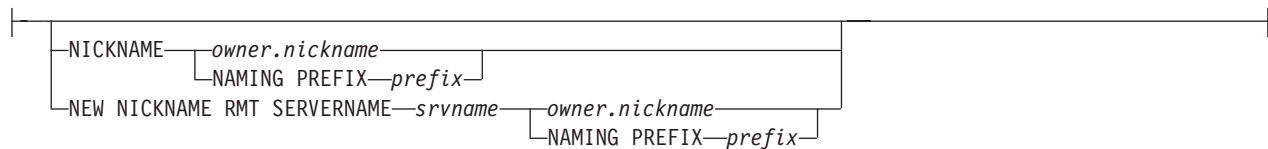
**source-clause:**



**target-clause:**



**nickname-options:**



## Parameters

### **SUBTYPE P**

Specifies Q subscriptions for peer-to-peer replication.

for-tables-clause:

Use this clause to specify one or more logical tables for which to create paired sets of Q subscriptions between the peer servers. If you use this clause:

- A SET TABLES command is not needed before the CREATE QSUB command in the ASNCLP script.
- You use a Q subscription profile to specify options for the Q subscriptions by using the CREATE SUBSCRIPTION OPTIONS command. With FOR TABLES, you do not use the COLS keyword to specify a subset of columns or rows, and you do not use the source-clause and target-clause to specify Q subscription options.

### **NODE**

Specifies which server in the peer-to-peer configuration should be used to locate the logical table on which the Q subscription is based.

node-option

Use these options to select one or more tables for which to create Q subscriptions.

*source\_owner*

Specifies the schema of a single source table.

*source\_name*

Specifies the name of a single source table.

source-predicate

Use these options to specify multiple source tables for which to create Q subscriptions.

### **OWNER LIKE**

Specifies a single database schema or schema pattern that uses the percentage sign (%) as a wild card.

### **NAME LIKE**

Specifies a single table name or table-naming pattern that uses the percentage sign (%) as a wild card.

### **ALL**

Specifies that you want to create Q subscriptions for all schemas and all tables within those schemas.

### **TARGET EXISTS VALIDATE NO**

Specifies that the target table exists and no validation is required for Q subscriptions. This option shortens processing time with very large tables.

**Important:** If you use these keywords, the ASNCLP program assumes that the target table matches exactly with the source table.

### **OPTIONS**

Specifies the name of a profile (list of options) for creating Q subscriptions. You create the profile by using the CREATE SUBSCRIPTION OPTIONS command. The OPTIONS clause takes precedence over any previously provided SET PROFILE command.

**COLS**

Specifies columns to be selected.

**ALL**

Select all columns in the Q subscription. This is the default.

**EXCLUDE (column )**

Exclude the specified columns from the Q subscription. If replication creates the target table, the specified columns are excluded. If the target exists, the column names in the source table and target table must be the same. Excluded columns must be nullable, or if defined as NOT NULL then they must have a default value.

**INCLUDE (column )**

Include the specified columns in the Q subscription. If the target table is new, the table is created with the specified columns. If the target table exists, then the specified columns are included in the table. The column names in the source table and target table must be the same.

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.

**I (default)**

Specifies an automatic load. The Q Apply program calls the LOAD from CURSOR utility 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.

**N** No load phase.

**CAPTURE\_LOAD**

**For peer-to-peer replication with two servers only:** Specifies the action that the Q Capture program takes when the recovery log shows that a load operation that uses the DB2 LOAD utility occurred at the source table.

**W (default)**

Q Capture issues a warning message after the load completes.

**R** Q Capture stops and starts the Q subscription for the source table, prompting a load of the target table if one is specified for the Q subscription.

**START AUTOMATICALLY**

Specifies how to start the Q subscription, which is represented by the State column in the IBMQREP\_SUBS table. The State column controls whether the Q subscription is automatically started after starting or reinitializing the Q Capture program (subscription state N), or that the Q subscription must be started manually by inserting a command in the IBMQREP\_SIGNAL table (subscription state I).

**YES**

The Q subscription is started automatically (subscription state value of N). This is the default.



**NO** The Q subscription must be started manually (subscription state value of I).

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.

**B** When an error occurs, spill change messages for the Q subscription to a temporary spill queue until you use the **resumesub** parameter of the **MODIFY** or **asnqacmd** command to prompt Q Apply to begin applying the messages.

**LOAD TYPE**

Specifies a type of load.

**0** Choose the best type automatically.

**1** Use LOAD from CURSOR only. Specify this option if the source and target servers are on z/OS.

**Note:** If the Q Apply program is at Version 9.7 Fix Pack 4 or newer, you do not need to provide nickname information for this load option unless the Q subscription includes XML columns. Q Apply calls LOAD from CURSOR by specifying a cataloged DB2 alias for the source database instead of by using a nickname. You must include the DB2 alias in a password file that is created by the asnpwd utility.

**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.

nickname-options:

**NICKNAME**

Specifies an existing 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 owner and name of an existing nickname.

**NAMING PREFIX *prefix***

Use these keywords if you are creating multiple Q subscriptions and nicknames already exist for LOAD from CURSOR. The variable *prefix* specifies a string that is used in naming all of the nicknames, and that the ASNCLP program can use to find the nicknames. For example, if you had 10 source tables named HR.T1 through HR.T10 and 10 nicknames that reference these tables named HR.SRCNKT1 through HR.SRCNKT10, you could use the string SRCNK to enable the ASNCLP program to find the nicknames and use them in the Q subscription definitions.

**NEW NICKNAME RMT SERVERNAME *srvname***

Specifies the name of the remote server if the ASNCLP program creates the nickname for loading.

### NAMING PREFIX *prefix*

Specifies a character string that the ASNCLP program can use to generate one or more new nicknames for loading.

### 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 6 are allowed (and are implicitly assigned).

Table 6. 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 PEER NODE 1 SERVER DBALIAS SAMPLE SCHEMA GRAY;
SET PEER NODE 2 SERVER DBALIAS SAMPLE2 SCHEMA BROWN;
SET PEER NODE 3 SERVER DBALIAS SAMPLE3 SCHEMA 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;
```

### Example: Subsetting columns - exclude

The following command creates a peer-to-peer Q subscription that excludes columns C1 and cC2:

```
CREATE QSUB SUBTYPE P COLS EXCLUDE (C1,C2)
```

## Example: Subsetting columns - include

The following command creates a peer-to-peer Q subscription that includes columns C1, C2, and C3:

```
CREATE QSUB SUBTYPE P SOURCE HAS LOAD PHASE I COLS INCLUDE (C1,C2,C3)
```

## Example: Using LOAD from CURSOR

The following commands set the environment and then create a peer-to-peer Q subscription for a single table that specifies the LOAD from CURSOR utility (LOAD TYPE 1) over a nickname. Q Apply will create the nickname. The remote server name is SRCSVR1, the source table is HR.TABLE1, and the nickname that references the source table is HR.SRCNKTABLE1:

```
ASNCLP SESSION SET TO Q REPLICATION;  
SET PEER NODE 1 SERVER DBALIAS REDDB;  
SET PEER NODE 2 SERVER DBALIAS BLUEDB;
```

```
SET TABLES (REDDB.ASN.HR.TABLE1);
```

```
CREATE QSUB SUBTYPE P FROM NODE REDDB.ASN SOURCE HAS LOAD PHASE I  
TARGET LOAD TYPE 1 NEW NICKNAME RMTSERVERNAME SRCSVR1 HR.SRCNKTABLE1;
```

## Example: Multiple tables with LOAD from CURSOR

The following commands set the environment and then create peer-to-peer Q subscriptions for all source tables with the schema "HR." The Q subscriptions specify the LOAD from CURSOR utility over a nickname. The ASNCLP program will create a nickname for each source table and use the naming prefix "SRCNK" to generate the nickname names. The remote server name is SRCSVR1:

```
ASNCLP SESSION SET TO Q REPLICATION;  
SET PEER NODE 1 SERVER DBALIAS REDDB;  
SET PEER NODE 2 SERVER DBALIAS BLUEDB;
```

```
SET TABLES (NODE 1 SRC OWNER LIKE "HR%");
```

```
CREATE QSUB SUBTYPE P FROM NODE REDDB.ASN SOURCE HAS LOAD PHASE I  
TARGET LOAD TYPE 1 NEW NICKNAME RMTSERVERNAME SRCSVR1 NAMING PREFIX SRCNK;
```

## LOAD from CURSOR with an existing nickname

The following commands set the environment and then create a peer-to-peer Q subscription for the source table HR.EMPLOYEE, specifying that the existing nickname HR.SOURCENICK should be used by the LOAD from CURSOR utility:

```
ASNCLP SESSION SET TO Q REPLICATION;  
SET PEER NODE 1 SERVER DBALIAS REDDB;  
SET PEER NODE 2 SERVER DBALIAS BLUEDB;
```

```
SET TABLES (SAMPLE.ASN.HR.EMPLOYEE);
```

```
CREATE QSUB SUBTYPE P FROM NODE REDDB.ASN SOURCE HAS LOAD PHASE I  
TARGET LOAD TYPE 1 NICKNAME HR.SOURCENICK;
```

## Using FOR TABLES clause to create multiple Q subscriptions

The following example uses the CREATE SUBSCRIPTION OPTIONS command to create a profile, p2plist, for specifying options for multiple Q subscriptions. To use the profile, the FOR TABLES keywords are specified. The OWNER LIKE keywords prompt the ASNCLP program to create Q subscriptions for all tables within

schemas on the SAMPLE1 server (NODE 1) that begin with the letters AIRUKU. Because FOR TABLES is used, no SET TABLES commands are required.

```
SET PEER NODE 1 SERVER DBALIAS SAMPLE1;
SET PEER NODE 2 SERVER DBALIAS SAMPLE2;

SET RUN SCRIPT NOW STOP ON SQL ERROR ON;

CREATE SUBSCRIPTION OPTIONS p2plist
  HAS LOAD PHASE E;

CREATE QSUB SUBTYPE P FOR TABLES
  (NODE 1 OWNER LIKE "AIRUKU%"
   TARGET EXISTS VALIDATE NO
   OPTIONS p2plist);
```

## CREATE REPLQMAP command

Use the **CREATE REPLQMAP** command to create a replication queue map for Q subscriptions.

### Syntax

```
►► CREATE REPLQMAP qmapname
  [DESC "description"] [(-NODE x-, -NODE y-)]
► USING ADMINQ "adminqname" RECVQ "recvqname" SENDQ "sendqname"
  [NUM APPLY AGENTS num]
► [MAXAGENTS CORRELID num] [MEMORY LIMIT limit] [ERROR ACTION {S|Q}]
► [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.

**NODE** *x*

In multidirectional replication, specifies the source server for this replication queue map. Use the same node number that was used in the SET BIDI NODE or SET PEER NODE command.

**NODE** *y*

In multidirectional replication, specifies the target server for this replication queue map. Use the same node number that was used in the SET BIDI NODE or SET PEER NODE command.

**ADMINQ** "*adminqname*"

Specifies the name of the administration queue at the Q Apply server.

**Note:** If the Q Capture or the Classic capture components share a single queue manager with the Q Apply program, the programs can share an administration queue.

**RECVQ** *"recvqname"*

Specifies the name of the receive queue that is used by the Q Apply program.

**SENDQ** *"sendqname"*

Specifies the name of the send queue that is used by the Q Capture program (for relational sources) or the capture components.

**NUM APPLY AGENTS** *num*

Specifies the number of threads that are used for concurrently applying transactions from the specified receive queue.

**MAXAGENTS CORRELID***num*

**z/OS** Specifies that number of threads that are used for concurrently applying transactions from the specified receive queue with the same *correlation ID*. The correlation ID identifies all transactions that were started from the same z/OS job on the Q Capture server.

The value for the **MAXAGENTS CORRELID** parameter cannot be greater than the value for the **NUM APPLY AGENTS** parameter. If **MAXAGENTS\_CORRELID** value is 1, the transactions will be applied one at a time. If the value is greater than one, for example 4, four agents will apply transactions with the same correlation ID in parallel. If the value is 0, transactions are applied in parallel by using the total number of threads specified by the **NUM APPLY AGENTS** parameter.

**MEMORY LIMIT** *limit*

Specifies the maximum number of megabytes that are used per receive queue for buffering incoming transactions.

**ERROR ACTION**

The action that the Q Capture program takes 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.

**S** The Q Capture program or the capture components stop when they detect an error on this queue.

**Q** The Q Capture program stops putting messages on any send queues that are in error and continues putting messages on other send queues. This value is not supported for Classic replication.

**HEARTBEAT INTERVAL** *interval*

Specifies the interval (in seconds) between heartbeat messages that are sent from the Q Capture program or the capture components 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 that is used for sending messages over the send queue.

## Example 1

To create a replication queue map SAMPLE\_ASN1\_TO\_TARGETDB\_ASN1 from a relational source:

```
CREATE REPLQMAP SAMPLE_ASN1_TO_TARGETDB_ASN1 USING ADMINQ "ASN1.QM1.ADMINQ"  
RECVQ "ASN1.QM1_TO_QM2.DATAQ" SENDQ "ASN1.QM1_TO_QM2.DATAQ"
```

## Example 2

To create a replication queue map CLASSIC\_ASN\_TO\_TARGETDB\_ASN1 from a Classic source:

```

SET SERVER CAPTURE TO CONFIG SERVER classic1 FILE classic.ini ID id1 PASSWORD pwd1
SET SERVER TARGET TO DB ASN1
SET RUN SCRIPT NOW STOP ON SQL ERROR ON
CREATE REPLQMAP CLASSIC1_ASN_TO_TARGETDB_ASN1 USING ADMINQ "ASN1.QM1.ADMINQ"
RECVQ "CLASSIC1.QM1_TO_QM2.DATAQ" SENDQ "CLASSIC1.QM1_TO_QM2.DATAQ"

```

### Example 3

In a bidirectional replication configuration, to create a replication queue map `SAMPLE_ASN_TO_TARGETDB_ASN1` to connect the Q Capture program at the `SAMPLE` server (node 1) with the Q Apply program at the `TARGETDB` server (node 2):

```

CREATE REPLQMAP SAMPLE_ASN_TO_TARGETDB_ASN1 (NODE 1, NODE 2) USING ADMINQ
"ASN1.QM1.ADMINQ" RECVQ "ASN1.QM1_TO_QM2.DATAQ" SENDQ "ASN1.QM1_TO_QM2.DATAQ"

```

---

## CREATE SCHEMASUB command

Use the **CREATE SCHEMASUB** command to create a schema-level subscription for unidirectional and bidirectional replication.

The command:

- Creates table-level Q subscriptions for all tables within the schema that meet the naming pattern that you specify.
- Saves the schema pattern so that the replication programs automatically create Q subscriptions for any tables that are added within the schema.

### Syntax

```

▶▶ CREATE SCHEMASUB schema_subname SUBTYPE B | U REPLQMAP queue_map_name
▶ FOR TABLES NODE node_number | table-properties | exclude-schema-options
▶ OPTIONS options_list_name
  | TARGET EXISTS VALIDATE NO

```

#### table-properties:

```

| OWNER LIKE predicate1 | NAME LIKE predicate2 |
| NAME LIKE predicate |
| ALL |

```

#### exclude-schema-options:

```

| EXCLUDE OWNER table_owner | NAME table_name |

```

## Parameters

### SUBTYPE

Specifies the type of replication:

- U** Unidirectional. You must specify a replication queue map to be used by all Q subscriptions within the schema.
- B** Bidirectional.

For bidirectional configurations, you do not need to specify a replication queue map if only one set of queue maps (one queue map in each direction) exists between the two servers. If more than one set of queue maps exists, use the SET CONNECTION command to specify which set of queue maps to use for the schema-level subscription.

### FOR TABLES

Use FOR TABLES along with the table-properties clause to specify a pattern for selecting the schemas, and tables within the schemas, that should be included in the schema-level subscription. Follow these guidelines:

- You can use the percentage sign (%) as a wild card.
- To replicate all CREATE TABLE and DROP TABLE operations within all schemas in the database, specify the ALL keyword (which is equivalent to OWNER LIKE % NAME LIKE %, and is stored as %.%).
- Patterns for schema-level subscriptions that use the same replication queue map must not overlap so that a table matches both patterns. For example, if you specified OWNER LIKE SMITH NAME LIKE % (stored as SMITH.%) and another schema-level subscription already existed that was created with OWNER LIKE % NAME LIKE T1 (stored as %.T1), both patterns would match the table SMITH.T1 and the CREATE SCHEMASUB command would fail.
- Table-level Q subscriptions that are part of a schema-level Q subscription and use the same replication queue map should all be of the same configuration type (unidirectional or bidirectional) and have the same properties.

### NODE

For SUBTYPE B or P. Specifies the server where the source tables to be included in the schema-level subscription reside.

### TARGET EXISTS VALIDATE NO

Specifies that the target table exists and no validation is required for table-level Q subscriptions that are created by the ASNCLP program. This option shortens processing time with very large tables. If these keywords and the SET ENFORCING MATCHING CONSTRAINTS command are used, the TARGET EXISTS VALIDATE NO clause provided on the CREATE SCHEMASUB command takes precedence.

**Important:** If you use these keywords, the ASNCLP program assumes that the target table matches exactly with the source table.

### OPTIONS

Specifies the name of a profile (list of options) for creating table-level Q subscriptions. You create the profile by using the CREATE SUBSCRIPTION OPTIONS command.

table-properties

**OWNER LIKE**

Specifies a single database schema or schema pattern that uses the percentage sign (%) as a wild card.

**NAME LIKE**

Specifies a single table name or table-naming pattern that uses the percentage sign (%) as a wild card.

**ALL**

Specifies that you want all schemas in the database, and all tables in the schemas, to be part of the schema-level subscription.

exclude-schema-options

**OWNER**

Specifies a schema to exclude from the schema-level subscription. For example, if there is a schema-level subscription for all tables in all schemas (using the wild card pattern %.%), but you specify EXCLUDE OWNER MSROSS, the statement CREATE TABLE MSROSS.T1 will not be replicated. A wild card is not allowed with this keyword.

**NAME**

Specifies one or more tables to exclude from the schema-level Q subscription. You can specify a single table name or table-naming pattern that uses the percentage sign (%) as a wild card.

**Usage notes**

- If you created a saved profile for creating target tables by using the SET PROFILE command, the options are used by the CREATE SCHEMASUB command when it creates target tables for table-level Q subscriptions.

**Example 1**

To create a schema-level subscription for unidirectional replication that includes all tables under the schema MSROSS:

```
CREATE SCHEMASUB SUBTYPE U REPLQMAP RQ1 FOR TABLES OWNER LIKE MSROSS;
```

**Example 2**

To create a schema-level subscription for bidirectional replication that includes all schemas and tables on the SAMPLE1 database and uses the saved profile options1:

```
SET BIDI NODE 1 SERVER DBALIAS SAMPLE1;
SET BIDI NODE 2 SERVER DBALIAS SAMPLE2;
```

```
CREATE SCHEMASUB SUBTYPE B FOR TABLES NODE 1 ALL OPTIONS options1;
```

---

## CREATE SUBSCRIPTION OPTIONS command

Use the **CREATE SUBSCRIPTION OPTIONS** command to create a profile that can be used to create table-level Q subscriptions when a schema-level subscription is in place. When the Q Capture program detects a CREATE TABLE operation within the schema, it automatically creates a Q subscription and uses the options that are specified in this profile.

**Relationship with SET PROFILE command:** The options that you specify in the SET PROFILE command are used by the CREATE SCHEMASUB command to create target tables for Q subscriptions that are created by ASNCLP. The options in the SET PROFILE and CREATE SUBSCRIPTIONS OPTIONS commands do not



intersect, and you can include both commands in the same input file. If both the SET PROFILE and CREATE SUBSCRIPTION OPTIONS commands are provided, the Q subscription-related attributes are picked from the CREATE SUBSCRIPTION OPTIONS command and the target table space attributes are picked from the SET PROFILE command.

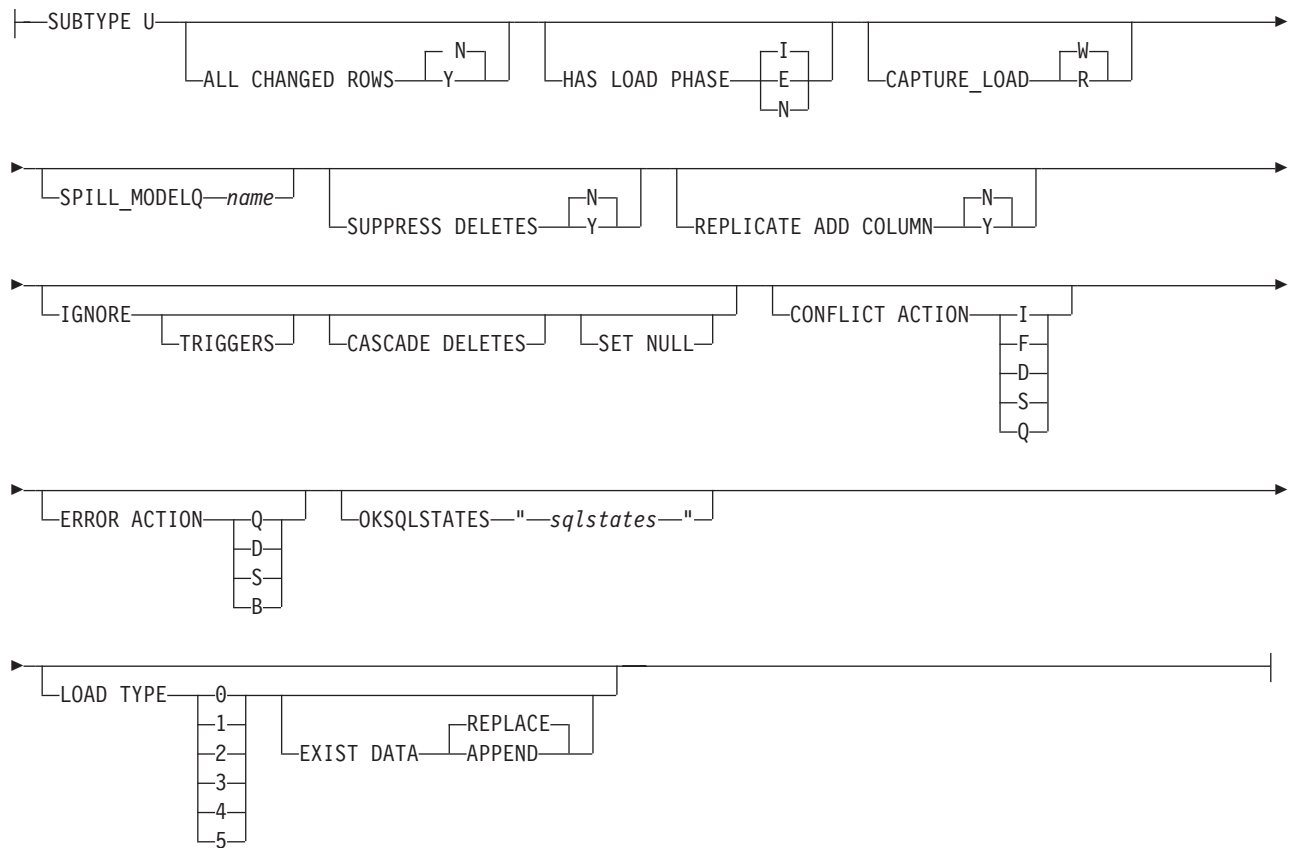
## Syntax

```

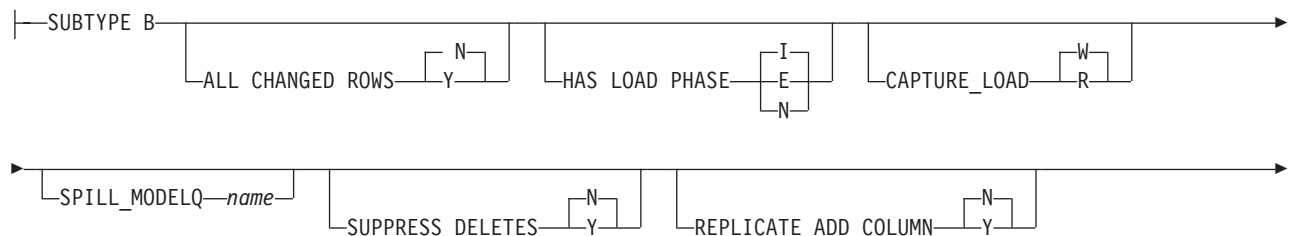
▶▶ CREATE SUBSCRIPTION OPTIONS options_name [ uni-properties | bidi-properties ]

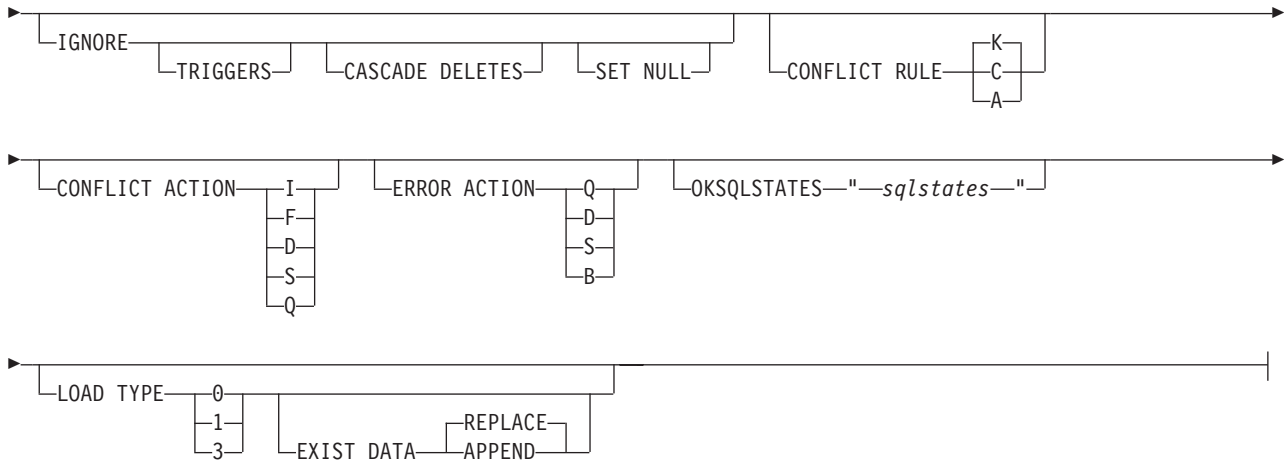
```

### uni-properties:



### bidi-properties:





## Parameters

For descriptions of the command parameters, see the identical descriptions in one of the following topics:

- “CREATE QSUB command (unidirectional replication)” on page 101
- “CREATE QSUB command (bidirectional replication)” on page 190

## Example

This example creates a profile called `bidioptions` that specifies properties for table-level, bidirectional Q subscriptions between the `SAMPLE` and `SAMPLE2` servers. The profile specifies a manual load phase and that cascaded delete operations should not be replicated:

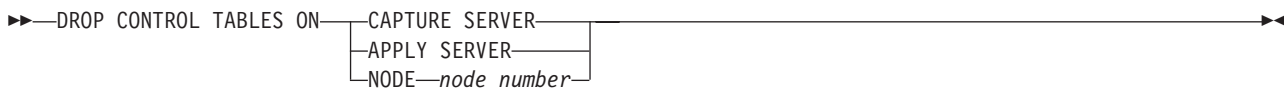
```
SET BIDI NODE 1 SERVER SAMPLE;
SET BIDI NODE 2 SERVER SAMPLE2;

CREATE SUBSCRIPTION OPTIONS bidioptions
SUBTYPE B HAS LOAD PHASE E IGNORE CASCADE DELETES;
```

## DROP CONTROL TABLES ON command

Use the **DROP CONTROL TABLES ON** command to drop the Q Capture control tables, Q Apply control tables, or both. In Classic replication, you can use this command to drop only the Q Apply control tables.

## Syntax



## Parameters

### CAPTURE SERVER

Specify to drop the Q Capture control tables.

### APPLY SERVER

Specify to drop the Q Apply control tables.

### NODE

Specify to drop the Q Capture and Q Apply control tables on a server in a bidirectional or peer-to-peer configuration. The server is identified by *node\_number*.

### Usage notes

This command is used in conjunction with the **SET SERVER** command to indicate the location of the control tables.

### Example: Q Capture control tables

To drop the Q Capture control tables:

```
SET SERVER TARGET TO QAPPDB;  
DROP CONTROL TABLES ON APPLY SERVER
```

### Example: Dropping both sets of control tables

To drop both Q Capture and Q Apply control tables on the SAMPLE1 and SAMPLE2 servers:

```
SET BIDI NODE 1 SERVER DBALIAS SAMPLE1;  
SET BIDI NODE 2 SERVER DBALIAS SAMPLE2;  
  
SET RUN SCRIPT NOW STOP ON SQL ERROR ON;  
  
DROP CONTROL TABLES ON NODE 1;  
DROP CONTROL TABLES ON NODE 2;
```

---

## DROP REPLQMAP command

Use the **DROP REPLQMAP** command to delete existing replication queue maps.

**Restriction:** Before you use the **DROP REPLQMAP** command, delete all Q subscriptions that use the replication queue map.

### Syntax

```
►►—DROP REPLQMAP—qmapname—  
└─NODE—x—, —NODE—y—┘
```

### Parameters

*qmapname*

Specifies the name of the replication queue map to delete.

**NODE** *x*, **NODE** *y*

Specifies to delete the replication queue map that connects two servers in one direction (**NODE** *x* and **NODE** *y*) in multidirectional replication.

### Example: unidirectional

To delete the SAMPLE\_ASN1\_TO\_TARGETDB\_ASN1 replication queue map:

```
DROP REPLQMAP SAMPLE_ASN1_TO_TARGETDB_ASN1;
```

## Example: multidirectional

To delete both replication queue maps between the SAMPLE1 and SAMPLE2 servers in a bidirectional configuration:

```
SET BIDI NODE 1 SERVER DBALIAS SAMPLE1;  
SET BIDI NODE 2 SERVER DBALIAS SAMPLE2;
```

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

```
DROP REPLQMAP repqmap1 NODE 1, NODE 2;  
DROP REPLQMAP repqmap2 NODE 2, NODE 1;
```

---

## DROP QSUB command

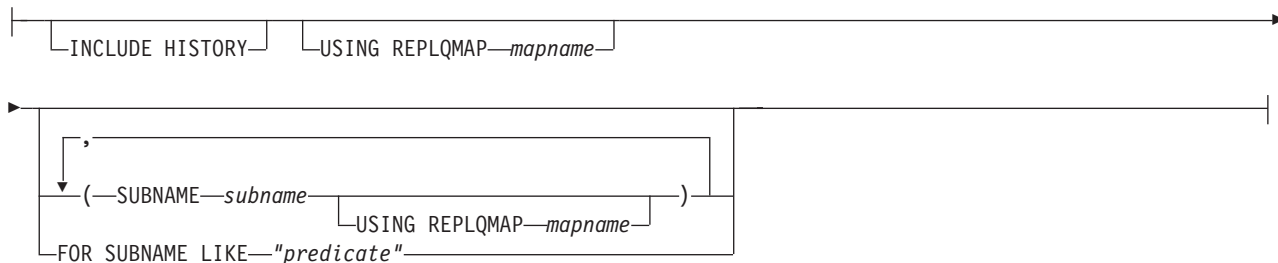
Use the **DROP QSUB** command to delete one or more Q subscriptions for unidirectional, bidirectional, or peer-to-peer Q Replication.

**Note:** Starting with Version 10 on Linux, UNIX, and Windows, use this command rather than the deprecated DROP SUBTYPE command to delete multidirectional Q subscriptions.

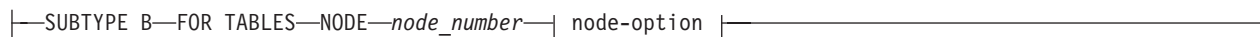
### Syntax



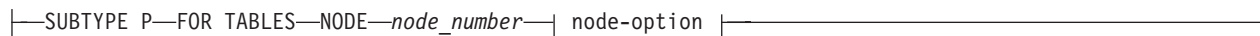
#### uni-options:



#### bidi-options:



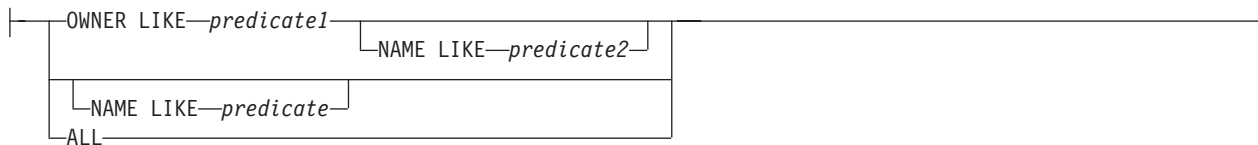
#### p2p-options:



#### node-option:



## source-predicate:



## Parameters

### ALL

Specify to delete all Q subscriptions. If you specify this parameter, you cannot combine it with any other parameters.

### uni-options

#### INCLUDE HISTORY

Specify to delete the Q subscription for the history table when the Q subscription for the base temporal table is deleted. If this clause is not specified, the option that was specified in the SET DROP TEMPORAL HISTORY SUB clause is used.

#### 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%";
```

### bidi-options

#### SUBTYPE B

Specifies that you want to delete one or more bidirectional Q subscriptions.

#### FOR TABLES

Use this clause to specify one or more logical tables for which to delete paired sets of Q subscriptions.

#### NODE

Specifies a server in the bidirectional configuration that should be used to locate the logical table on which the Q subscriptions to be deleted are based.

### p2p-options

#### SUBTYPE P

Specifies that you want to delete one or more peer-to-peer Q subscriptions.

#### FOR TABLES

Use this clause to specify one or more logical tables for which to delete paired sets of Q subscriptions.

**NODE**

Specifies a server in the peer-to-peer configuration that should be used to locate the logical table on which the Q subscriptions to be deleted are based.

node-options

Use these options to select one or more tables for which to delete Q subscriptions.

*source\_owner*

Specifies the schema of a single logical table.

*source\_name*

Specifies the name of a single logical table.

source-predicate

Use these options to specify multiple logical tables for which to delete Q subscriptions.

**OWNER LIKE**

Specifies a single database schema or schema pattern that uses the percentage sign (%) as a wild card.

**NAME LIKE**

Specifies a single table name or table-naming pattern that uses the percentage sign (%) as a wild card.

**ALL**

Specifies that you want to delete Q subscriptions for all schemas and all tables within those schemas.

**Example: unidirectional**

To delete a Q subscription for unidirectional replication:

```
DROP QSUB (SUBNAME EMPLOYEE0001 USING REPLQMAP SAMPLE_ASN1_TO_TARGETDB_ASN1);
```

**Example: multidirectional**

To delete all of the paired Q subscriptions for bidirectional replication under schemas that start with the letters "AIRUKU" on the SAMPLE1 and SAMPLE2 servers:

```
SET BIDI NODE 1 SERVER DBALIAS SAMPLE1;
SET BIDI NODE 2 SERVER DBALIAS SAMPLE2;
```

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

```
DROP QSUB SUBTYPE B FOR TABLES (NODE 1 OWNER LIKE "AIRUKU%");
```

**DROP SCHEMASUB command**

Use the **DROP SCHEMASUB** command to delete a schema-level subscription. You can also use this command to delete all Q subscriptions that belong to the schema-level subscription.

**Syntax**

```
►►—DROP SCHEMASUB—schema_sub_name—

|          |
|----------|
| ALL      |
| NEW ONLY |

—►►
```

## Parameters

### ALL

Specify to delete the schema-level subscription and all of the table-level Q subscriptions that belong to it.

### NEW ONLY

Specify to delete only the schema-level subscription.

## Example 1

To delete the schema-level subscription schema1 in a bidirectional configuration and delete all of the table-level Q subscription that belong to it:

```
SET BIDI NODE 1 SERVER SAMPLE;  
SET BIDI NODE 2 SERVER SAMPLE2;  
  
DROP SCHEMASUB schemasub1 ALL;
```

## Example 2

To delete the schema-level subscription schema2 in a bidirectional configuration but leave all of the table-level Q subscription that belong to it:

```
SET BIDI NODE 1 SERVER SAMPLE;  
SET BIDI NODE 2 SERVER SAMPLE2;  
  
DROP SCHEMASUB schemasub2 NEW ONLY;
```

---

## 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 BIDIRGROUP. 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 SUBSCRIPTION OPTIONS command

Use the **DROP SUBSCRIPTION OPTIONS** command to delete a list of Q subscription options that is used as a profile for creating table-level Q subscriptions when a schema-level subscription is in place.

**Important:** You can only use this command if the list of Q subscription options is not being used by any schema-level Q subscriptions. Any schema-level subscriptions that are using the list must be deleted before you can delete the list.

### Syntax

►►—DROP SUBSCRIPTION OPTIONS—*options\_name*—◄◄

### Parameters

*options\_name*

The name of the list of Q subscription options, as specified in the CREATE SUBSCRIPTION OPTIONS command and stored in the IBMQREP\_SUBS\_PROF table at the Q Capture server.

### Example

To delete the list of Q subscription options named options1 that is used as a profile for creating Q subscriptions between the SAMPLE and SAMPLE1 servers:

```
SET BIDI NODE 1 SERVER SAMPLE;  
SET BIDI NODE 2 SERVER SAMPL1;  
  
DROP SUBSCRIPTION OPTIONS options1;
```

---

## LIST APPLY SCHEMA command

You can use the **LIST APPLY SCHEMA** command to list the Q Apply schemas for a specified server.

### Syntax

►►—LIST APPLY SCHEMA—  
└──SERVER──| dbparms ─┘ ◄◄

### dbparms-clause:

┌──DBALIAS—*aliasname*──┐  
└──CONFIG SERVER—*servername*──┐ ┌──ID—*userid*──┐ ┌──PASSWORD—*pwd*──┐  
└──FILE—*filename*──┘

### Parameters

dbparms-clause:

#### SERVER

Specifies the server that contains the schemas to be listed.



**DBALIAS** *aliasname*

Specifies the database alias name of a z/OS subsystem or Linux, UNIX, or Windows database as cataloged on the DB2 from which the ASNCLP is invoked.

**DBNAME** *zosdbname*

**z/OS** Specifies the z/OS database name. This is a logical z/OS database name, as created on a z/OS subsystem.

**ID** *userid*

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

**PASSWORD** *pwd*

Specifies the password to use for connections.

**CONFIG SERVER** *servername*

**Classic sources:** Specifies which server configuration settings from the Classic replication configuration file that the ASNCLP should use to connect to the Classic server.

**FILE** *filename*

Specifies the complete path and file name to the replication configuration file. If you do not use the **FILE** parameter, the ASNCLP attempts to use the `asnservers.ini` file in the current directory, if that file exists. Use the **FILE** parameter with different files that are customized for different environments.

### Example

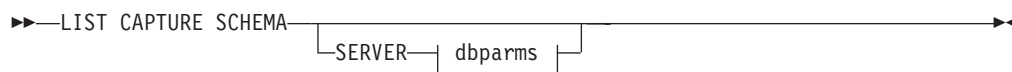
To list the Q Capture schema on server SAMPLE:

```
LIST CAPTURE SCHEMA SERVER DBALIAS SAMPLE ID id1 PASSWORD "passwd!";
```

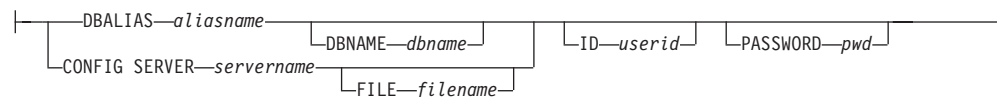
## LIST CAPTURE SCHEMA command

You can use the **LIST CAPTURE SCHEMA** command to list the Q Capture schemas for a specified server.

### Syntax



### dbparms-clause:



### Parameters

dbparms-clause:

**SERVER**

Specifies the server that contains the schemas to be listed.

**DBALIAS** *aliasname*

Specifies the database alias name of a z/OS subsystem or Linux, UNIX, or Windows database as cataloged on the DB2 from which the ASNCLP is invoked.

**DBNAME** *zosdbname*

**z/OS** Specifies the z/OS database name. This is a logical z/OS database name, as created on a z/OS subsystem.

**ID** *userid*

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

**PASSWORD** *pwd*

Specifies the password to use for connections.

**CONFIG SERVER** *servername*

**Classic sources:** Specifies which server configuration settings from the Classic replication configuration file that the ASNCLP should use to connect to the Classic server.

**FILE** *filename*

Specifies the complete path and file name to the replication configuration file. If you do not use the **FILE** parameter, the ASNCLP attempts to use the `asnservers.ini` file in the current directory, if that file exists. Use the **FILE** parameter with different files that are customized for different environments.

**Example**

To list the Q Capture schema on server SAMPLE:

```
LIST CAPTURE SCHEMA SERVER DBALIAS SAMPLE ID id1 PASSWORD "passwd!";
```

**LIST SCHEMASUB command**

The **LIST SCHEMASUB** command generates a list of all DB2 schemas on a source or target server for which a schema-level subscription is defined. It also shows whether the schema-level subscriptions are for unidirectional, bidirectional, or peer-to-peer replication.

**Syntax**

▶▶—LIST SCHEMASUB—▶▶

**Example**

To list all of the schema-level subscriptions on the SAMPLE database, which is part of a bidirectional configuration:

```
SET BIDI NODE 1 SERVER SAMPLE;
LIST SCHEMASUB;
```

**Command output**

Assume that the schema-level subscription on SAMPLE was created using the expression MSROSS%. The schema-level subscriptions on SAMPLE are MSROSS1, MSROSS2, and MSROSS3. Here is the output of the LIST SCHEMASUB command:

Schemas	Subscription type
MSROSS1	U
MSROSS2	B
MSROSS3	B

One schema-level subscription exists on the server for unidirectional replication, with two for bidirectional replication.

## LOAD DONE command

Use the **LOAD DONE** command to inform the Q Capture program or the Classic capture components that the target table is loaded. Issue the **LOAD DONE** command only if you are doing a manual load. If the Q Apply program is doing the load, this signal is not necessary.

### Syntax

```

▶▶—LOAD DONE— QSUB— SUBNAME—subname
   └─FOR SUBNAME LIKE—"%text%"—┘ └─CAP SERVER OPTIONS—┘ classic-opt-clause ┘

```

#### classic-opt-clause:

```

┌─┐ ┌─DBALIAS—aliasname—┐ ┌─DBNAME—dbname—┐ ┌─ID—userid—┐ ┌─PASSWORD—pwd—┐
└─┘ └─┘ └─┘ └─┘ └─┘ └─┘ └─┘ └─┘
┌─┐ ┌─CONFIG SERVER—servername—┐ ┌─FILE—filename—┐
└─┘ └─┘ └─┘ └─┘
┌─┐ ┌─CAPSCHEMA—schema—┐
└─┘ └─┘

```

### 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%"
```

#### CAP SERVER OPTIONS

Specifies additional parameters when you issue the **LOAD DONE** command in immediate execution mode.

classic-opt-clause: These parameters only work with Classic sources.

#### DBALIAS *aliasname*

Specifies the database alias name of a z/OS subsystem or Linux, UNIX, or Windows database as cataloged on the DB2 from which the ASNCLP is invoked.

#### DBNAME *zosdbname*

**z/OS** Specifies the z/OS database name. This is a logical z/OS database name, as created on a z/OS subsystem.

#### ID *userid*

Specifies the user ID to use to connect to the source database.

**PASSWORD** *pwd*

Specifies the password to use to connect to the source database.

**CAPSCHEMA** *schema*

Specifies the schema of the control tables of the Classic source.

**CONFIG SERVER** *servername*

Specifies which server configuration settings from the Classic replication configuration file that the ASNCLP should use to connect to the Classic data source.

**FILE** *filename*

Specifies the Classic replication server that the ASNCLP program connects to. The server name must match the name that is entered in the Classic replication configuration file. If you do not use the **FILE** parameter, the ASNCLP attempts to use the `anservers.ini` file in the current directory, if that file exists.

### Example

To signal the Q Capture program or the capture components that the target table for the Q subscription EMPLOYEE0001 is loaded:

```
LOAD DONE QSUB SUBNAME EMPLOYEE0001
```

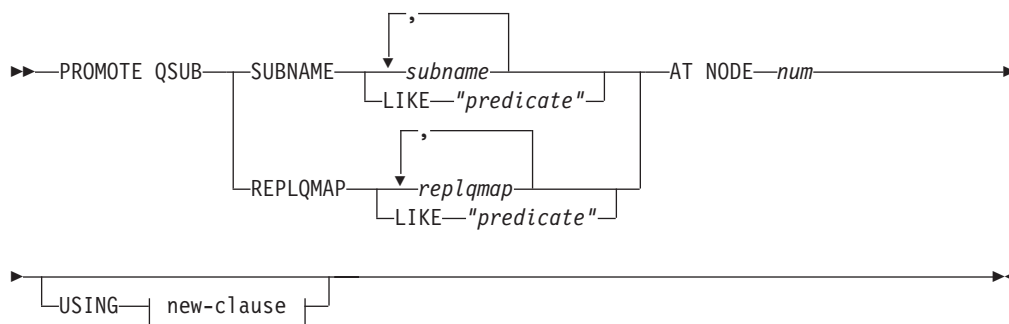
---

## PROMOTE QSUB command (multidirectional replication)

Use the **PROMOTE QSUB** command to build an ASNCLP script with statements that you can use to create Q subscriptions on another set of servers. Promoting is useful for copying Q subscriptions from test systems to production systems or migrating Q subscriptions from one server to another.

You can also use this command to customize some of the properties of the promoted Q subscription, including the name of the Q Capture and Q Apply schemas and the replication queue map that is used. The promoted values of properties that cannot be customized are taken from the source Q subscription. If you need to change other properties, you can use the **ALTER QSUB** command after promoting the Q subscription to change the properties for the new Q subscription.

### Syntax



**new-clause::**



## Parameters

**SUBNAME** *subname*

Specifies one or more Q subscription names to promote. Separate multiple Q subscription names with commas.

**LIKE** *"predicate"*

Specifies a list of Q subscription names to promote that match the predicate.

**REPLQMAP** *replqmap*

Specifies one or more replication queue maps. Separate multiple map names with commas. All Q subscriptions that use the specified map or maps are promoted.

**LIKE** *"predicate"*

Specifies a list replication queue maps that match the predicate. All Q subscriptions that use the matching maps are promoted.

**AT NODE** *num*

Specifies the node number of the configuration to be promoted. Default value is 1 for NODE 1. A "node" is a paired Q Capture-Q Apply schema at a server that is participating in bidirectional or peer-to-peer replication. For example you could have three physical computers that were involved in peer-to-peer replication, each with a database. Within each database are one or more Q Capture-Q Apply programs and their control tables that are identified by a schema. The paired schema represents a "node" in a three-way peer-to-peer configuration

new-clause:

**USING SOURCE SCHEMA** *schema*

Specifies the source table schema.

**USING TARGET SCHEMA** *schema*

Specifies the target table schema. If the schema is not specified, the promoted definition uses the schema of the current target table.

**USING REPLQMAP** *newqmap*

Specifies the name of a new replication queue map that you want to use for the promoted Q subscriptions.

## Example - matching a predicate

To promote all bidirectional Q subscriptions that match the predicate EMP at NODE 1:

```
PROMOTE QSUB SUBNAME LIKE "EMP%" AT NODE 1;
```

---

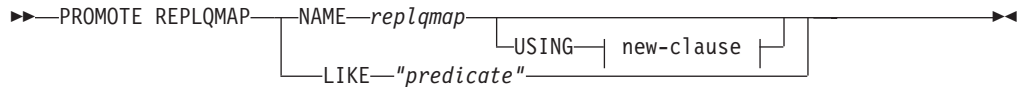
## PROMOTE REPLQMAP command

Use the **PROMOTE REPLQMAP** command to promote one or more replication queue maps from one set of control tables to another.

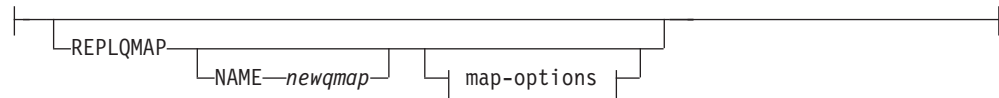
If a single replication queue map is specified, you can also use this command to customize some of the properties of the promoted queue map, including the name

of the replication queue map and name of the send queue. The promoted values of properties that cannot be customized are taken from the source replication queue map. If you need to change other properties, you can use the **ALTER REPLQMAP** command after promoting the replication queue map to change the properties for the new replication queue map.

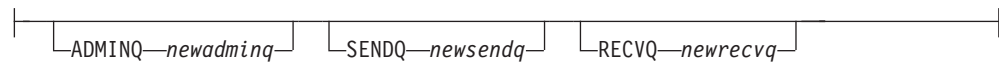
## Syntax



### new-clause:



### map-options:



## Parameters

### **NAME** *replqmap*

Specifies the name of an existing replication queue map to be promoted.

### **LIKE** *"predicate"*

Specifies a list of replication queue map names that match the predicate. All replication queue map names that match the predicate will be promoted.

### *new-clause*

#### **REPLQMAP**

Specifies new property values for the promoted replication queue map.

#### **NAME** *newqmap*

Specifies a new name for the replication queue map. If you do not specify a new name, then the current replication queue map name is used.

### *map-options*

#### **ADMINQ** *newadminq*

Specifies a new name for the administration queue. If you do not specify a new name, then the current administration queue name is used.

#### **SENDQ** *newsendq*

Specifies a new name for the send queue. If you do not specify a new name, then the current send queue name is used.

#### **RCVQ** *newrcvq*

Specifies a new name for the receive queue. If you do not specify a new name, then the current receive queue name is used.

## Example 1

To promote replication queue maps that match the name "SAMPLE\_ASN":  
PROMOTE REPLQMAP LIKE "SAMPLE\_ASN%";

## Example 2

To promote replication queue map REPLQMAP2 and customize several properties of the promoted version of that queue map, so that the new replication queue map name is REPLQMAPNEW2, the new administration queue name is adminqnew2, the new send queue name is sendqnew2, and the new receive queue name is recvqnew2:

```
PROMOTE REPLQMAP NAME REPLQMAP2 USING REPLQMAP NAME REPLQMAPNEW2  
ADMINQ "adminqnew2" SENDQ "sendqnew2" RECVQ "recvqnew2";
```

---

## REINIT SCHEMASUB command

Use the **REINIT SCHEMASUB** command to generate a script that prompts the Q Capture program to reread any changes to the options for a schema-level subscription. You can also use this command to prompt Q Capture to reread changes to the table-level Q subscriptions within the schema.

### Syntax

```
▶▶—REINIT SCHEMASUB—schema_sub_name—

|          |
|----------|
| ALL      |
| NEW ONLY |

—▶▶
```

### Parameters

#### ALL

Specify to reinitialize a schema-level subscription and all of the table-level Q subscriptions that belong to it. The command generates a SQL script to insert a REINIT\_SCHEMASUB signal into the IBMQREP\_SIGNAL table at the Q Capture server for the schema-level Q subscription, and REINIT\_SUB signals for the table-level Q subscriptions. You can use the SET RUN SCRIPT NOW option to immediately insert the signals.

**Note:** Reinitializing a schema-level subscription updates the options that are used for creating table-level Q subscriptions within the schema. However, the changes are used only for newly created tables. To update options for existing table-level Q subscriptions, you must reinitialize these Q subscriptions.

#### NEW ONLY

Specify to reinitialize only the schema-level subscription.

### Example

To reinitialize the schema-level Q subscription schemasub1 and all of its table-level Q subscriptions, and also reinitialize only the schema-level subscription schemasub2:

```
SET BIDI NODE 1 SERVER DBALIAS SAMPLE1;  
SET BIDI NODE 2 SERVER DBALIAS SAMPLE2;  
  
SET RUN SCRIPT NOW STOP ON SQL ERROR ON;  
  
REINIT SCHEMASUB schemasub1 ALL;  
REINIT SCHEMASUB schemasub2 NEW ONLY;
```

---

## 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] applieschema ▶▶
```

### Parameters

#### TO DEFAULT

Specify to set the Q Apply schema to ASN and to reset any previous **SET APPLY SCHEMA** commands.

*applieschema*

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 BIDI NODE command

Use the **SET BIDI NODE** command to specify the paired Q Capture and Q Apply control tables in a bidirectional configuration.

**Note:** Use this command instead of the deprecated SET SERVER command for bidirectional replication. Use the SET SERVER command for unidirectional replication only.

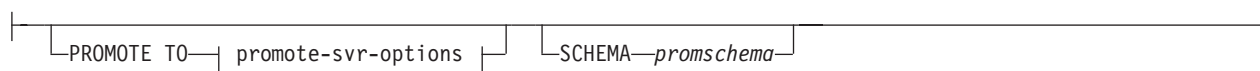
You also use **SET BIDI NODE** to specify the paired Q Capture and Q Apply control tables that will be the source of replication definitions to promote to another bidirectional server.

### Syntax

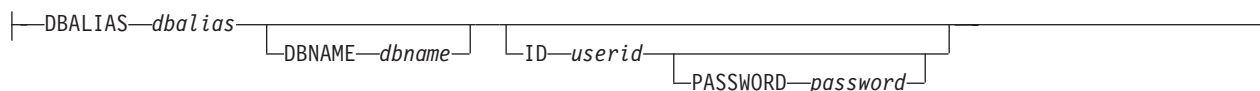
```
▶▶ SET [BIDI | BIDIRECTIONAL] NODE number SERVER [DBALIAS dbalias] [DBNAME dbname] ▶▶  
▶▶ [ID userid] [PASSWORD pwd] [SCHEMA schema] [promote-options] ▶▶
```



## promote-options:



## promote-srvr-options:



## Parameters

### **NODE** *number*

Specifies server 1 or 2 of the bidirectional configuration. A server represents a combination of server and schema.

### **SERVER**

Specifies the source database alias name. This is the database that contains the configuration that is being promoted.

### **DBALIAS** *dbalias*

Specifies the database alias name of a z/OS subsystem or Linux, UNIX, or Windows database as cataloged on the DB2 from which the ASNCLP is invoked.

### **z/OS** **DBNAME** *dbname*

Specifies the DB2 for z/OS database name.

**Note:** DBNAME is mandatory when ASNCLP is running on z/OS and the bidirectional server is on z/OS. DBNAME is a location name and is the name by which the DB2 database is known to local DB2 SQL applications. This name must match the name that was entered in the LOCATIONS column of the SYSIBM.LOCATIONS table in the CDB.

### **ID** *userid*

Specifies the user ID to use when you connect to the source database.

### **PASSWORD** *pwd*

Specifies the password to use when you connect to the source server that is specified by the **SERVER** parameter. If you specify the user ID and do not specify the password, you are prompted to enter the password. The password is hidden as you type.

**Note:** This keyword is not valid when the ASNCLP runs natively on z/OS because user authentication is handled through the communication database (CDB).

### **SCHEMA** *schema*

Specifies the schema that contains the configurations to promote from the source server. The source server is specified by the **SERVER** **DBALIAS** or **DBNAME** parameters.

## promote-options

### **PROMOTE TO** *dbalias*

Specifies the destination database alias name to receive the promoted configuration.

**SCHEMA** *promschema*

Specifies the schema of the control tables in the destination database. If the schema is not specified, then the schema in the source configuration is used in the generated scripts for the promoted configuration.

promote-srvr-options

**DBALIAS** *aliasname*

Specifies the destination database alias name.

**z/OS** **DBNAME** *dbname*

Specifies the destination database or subsystem name.

**ID** *userid*

Specifies the user ID of the database where you want to promote the configurations. This connection information is used in the generated ASNCLP scripts.

**PASSWORD** *pwd*

Specifies the password of the database where you want to promote the configurations. The user ID and password are used in the generated ASNCLP scripts that you later run to create the new configurations at the destination server.

**Example 1**

To specify the servers for a bidirectional configuration:

```
SET BIDI NODE 1 SERVER DBALIAS SAMPLE DBNAME SAMPLE SCHEMA ASN;
SET BIDI NODE 2 SERVER DBALIAS TEMPDB DBNAME TEMPDB SCHEMA ASN;
```

**Example 2**

To specify the servers to promote configurations from and the corresponding destination servers:

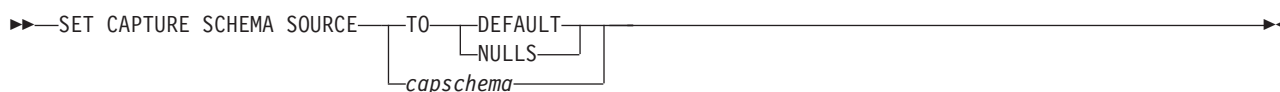
```
SET BIDI NODE 1 SERVER DBALIAS TEST01 ID id1 PASSWORD "p1wd" SCHEMA ASN
PROMOTE TO DBALIAS PRODUCTION01 ID id1 PASSWORD "pw1d" SCHEMA ASN;

SET BIDI NODE 2 SERVER DBALIAS TEST02 ID id1 PASSWORD "p1wd" SCHEMA ASN
PROMOTE TO DBALIAS TEST011 ID id1 PASSWORD "pw1d" SCHEMA ASN;
```

**SET CAPTURE SCHEMA command**

Use the **SET CAPTURE SCHEMA** command to set a default schema of the source control tables for all task commands. For Classic sources, you can use only the default Q Capture schema, ASN.

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

**Syntax**

## Parameters

### SOURCE

Specifies the Q Capture schema. If you are using a DB2 source, the schema can be any valid DB2 schema name. If you are using a Classic source, you must use the DEFAULT schema.

### 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 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 [SUBNAME—subscriptionname] SOURCE—sourceservername.sourceschemaname
▶ 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.

*target schemaname*

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. If no SET CONNECTION command is provided and a single replication queue map is found in the corresponding control table row or defined in the input script, the ASNCLP uses the value that it finds.

## 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 ENFORCE MATCHING CONSTRAINTS command (multidirectional Q Replication)

Use the **SET ENFORCE MATCHING CONSTRAINTS** command to specify whether the ASNCLP enforces matching constraints between the source and target tables. The ASNCLP by default checks that referential integrity constraints, check constraints, and unique constraints match for the source and target tables.

## Syntax

```
▶▶—SET ENFORCE MATCHING CONSTRAINTS—

|     |
|-----|
| YES |
| NO  |

—————▶▶
```

## Parameters

**YES**

Specify to enforce referential integrity constraints, check constraints, and unique constraints.

**NO** Specify to not enforce matching constraints on source and target tables.

## Usage notes

When you specify NO, you can subscribe a child table before subscribing the parent table.

## Example

```
ASNCLP SESSION SET TO Q REPLICATION;  
SET SUBGROUP "P2PSUBGROUP";  
SET PEER NODE 1 SERVER DBALIAS SAMPLE SCHEMA ASN;  
SET PEER NODE 2 SERVER DBALIAS TEMPDB SCHEMA ASN;
```

```

SET CONNECTION SOURCE "SAMPLE".ASN TARGET "TEMPDB".ASN REPLQMAP "RQ1"
SET CONNECTION SOURCE "TEMPDB".ASN TARGET "SAMPLE".ASN REPLQMAP "RQ2";
SET ENFORCE MATCHING CONSTRAINTS NO;
SET TABLES (SAMPLE.ASN.DB2OWNER.TEMP_FK, TEMPDB.ASN.DB2OWNER.TEMP_FK);
CREATE QSUB SUBTYPE P;

```

---

## 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 "logfile" [WITH DETAILS]

```

### Parameters

*"logfile"*

Specifies the output log file name. The default log file name is qreplmsg.log.

#### WITH DETAILS

Creates an additional log file with just error messages for the run along with the "Explanation" and "User response" sections for each message. The name of the additional file is *logfile\_1*. The contents of the standard log file remain unchanged.

### 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 1

To name the output log file qmaplog.err for creating replication queue maps:

```
SET LOG "qmaplog.err";
```

### Example 2

To specify that the ASNCLP program create its regular log file and an additional log file with error messages and the "Explanation" and "User response" sections for each message:

```
SET LOG "qrepllog.err" WITH DETAILS;
```

---

## 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, or the ASNCLP commands needed to promote a replication environment.

This command is not used when the ASNCLP runs natively on z/OS. The output files are defined by DD statements in the JCL.

## Syntax

```
▶▶ SET OUTPUT [MULTIDIR] [PROMOTE SCRIPT "profname"] ▶▶
```

### Parameters

#### MULTIDIR

Specify to name the output files after the databases that the SQL scripts run on.

#### PROMOTE SCRIPT "*profname*"

Specifies the output file name for the ASNCLP commands generated by PROMOTE statements. If the file name is not specified, the default file created is named `qrepl_asnclp.in`.

### 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 PEER NODE command

Use the **SET PEER NODE** command to specify the paired Q Capture and Q Apply control tables on a server in a peer-to-peer configuration.

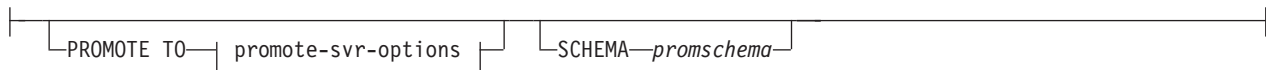
**Note:** Use this command instead of the deprecated SET SERVER command for peer-to-peer replication. Use the SET SERVER command for unidirectional replication only.

You also use **SET PEER NODE** to specify the paired Q Capture and Q Apply control tables that will be the source of replication definitions to promote to another peer-to-peer server.

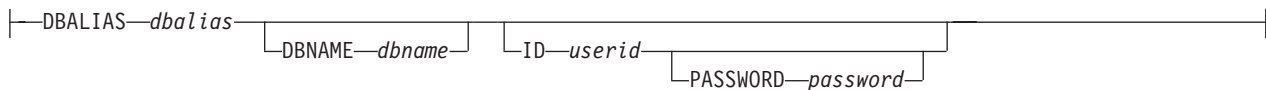
### Syntax

```
▶▶ SET [PEER [PEERTOPEER] NODE number SERVER [DBALIAS dbalias] [DBNAME dbname] ▶▶  
▶ [ID userid] [PASSWORD pwd] [SCHEMA schema] [promote-options] ▶▶
```

#### promote-options:



### promote-srvr-options:



## Parameters

### **NODE** *number*

Specify with a digit from 1 to 6 a server in the peer-to-peer configuration that defines the overall peer-to-peer context to be promoted. A server represents a set of Q Capture and Q Apply programs that are on the same server, have the same schema, and are involved in the peer-to-peer configuration. Up to six servers can be identified in a peer-to-peer configuration, each defined with a separate **SET PEER NODE** command.

### **SERVER**

Specifies the source database alias name. This is the database that contains the configuration that is being promoted.

### **DBALIAS** *aliasname*

Specifies the database alias name of a z/OS subsystem or Linux, UNIX, or Windows database as cataloged on the DB2 from which the ASNCLP is invoked.

### **DBNAME** *zosdbname*

**z/OS** Specifies the z/OS database name. This is a logical z/OS database name, as created on a z/OS subsystem.

**Note:** DBNAME is mandatory when ASNCLP is running on z/OS and the peer-to-peer server is on z/OS. DBNAME is a location name and is the name by which the DB2 database is known to local DB2 SQL applications. This name must match the name that was entered in the LOCATIONS column of the SYSIBM.LOCATIONS table in the CDB.

### **ID** *userid*

Specifies the user ID to use when you connect to the source database.

### **PASSWORD** *pwd*

Specifies the password to use when you connect to the source server. If you specify the user ID and do not specify the password, you will be prompted to enter the password. The password is hidden as you type.

**Note:** This keyword is not valid when the ASNCLP runs natively on z/OS because user authentication is handled through the communication database (CDB).

### **SCHEMA** *schema*

Specifies the source schema name.

### promote-options

### **PROMOTE TO** *dbalias*

Specifies the destination database alias name to receive the promoted configuration.

**SCHEMA** *promschema*

Specifies the schema of the control tables in the destination database. If the schema is not specified, then the schema in the source configuration is used in the generated scripts for the promoted configuration.

promote-srvr-options

**DBALIAS** *aliasname*

Specifies the destination database alias name.

**z/OS**

**DBNAME** *dbname*

Specifies the destination database name.

**ID** *userid*

Specifies the user ID of the destination database for promotion. The resulting promotion commands will not include a user ID if this parameter is not specified.

**PASSWORD** *pwd*

Specifies the password to use to connect to the destination database. The resulting promotion commands will not include a password if this parameter is not specified.

**Example 1**

To specify the three servers in a peer-to-peer configuration:

```
SET PEER NODE 1 SERVER DBALIAS GRAY DBNAME GRAY SCHEMA ASN;
SET PEER NODE 2 SERVER DBALIAS BROWN DBNAME BROWN SCHEMA ASN;
SET PEER NODE 2 SERVER DBALIAS YELLOW DBNAME YELLOW SCHEMA ASN;
```

**Example 2**

To specify the servers from which to promote configurations and the corresponding destination servers:

```
SET PEER NODE 1 SERVER DBALIAS AMERICAS ID id1 PASSWORD "p1wd" SCHEMA ASN
PROMOTE TO DBALIAS AMERICAS01 ID id1 PASSWORD "pw1d" SCHEMA ASN;
```

```
SET PEER NODE 2 SERVER DBALIAS EUROPE ID id1 PASSWORD "p1wd" SCHEMA ASN
PROMOTE TO DBALIAS EUROPE01 ID id1 PASSWORD "pw1d" SCHEMA ASN;
```

```
SET PEER NODE 3 SERVER DBALIAS ASIA ID id1 PASSWORD "p1wd" SCHEMA ASN
PROMOTE TO DBALIAS ASIA01 ID id1 PASSWORD "pw1d" SCHEMA ASN;
```

---

**SET PROFILE command**

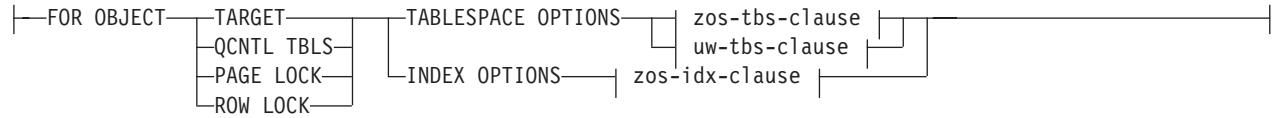
Use the **SET PROFILE** command to specify custom parameters for table spaces or indexes that are created by the ASNCLP program. After you issue a **SET PROFILE** command, you can associate a profile with a task command by specifying the profile's name in the task command.

**Syntax**

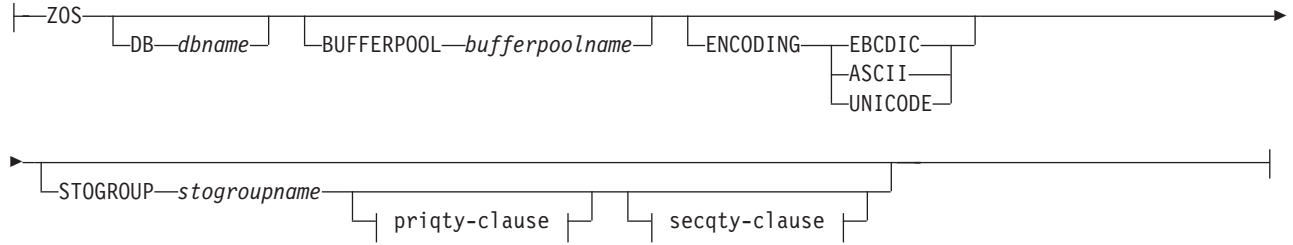
```
▶▶—SET PROFILE—profilename—┌── prof-clause ─┐──▶
                               └── UNDO ───┘
```



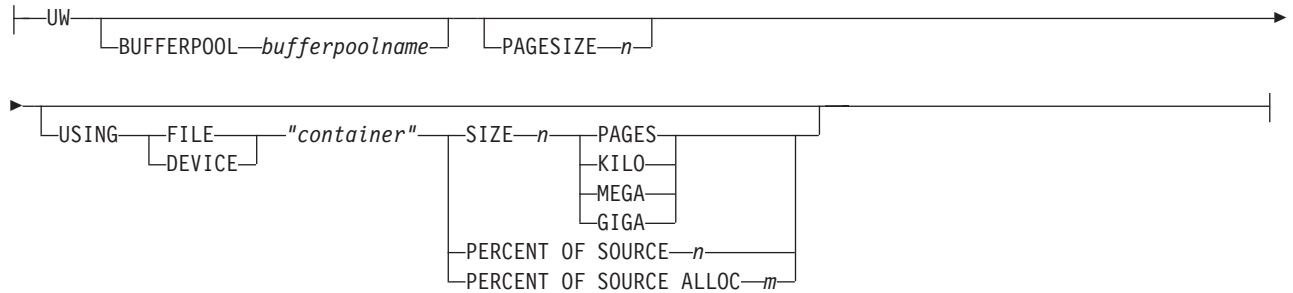
**prof-clause:**



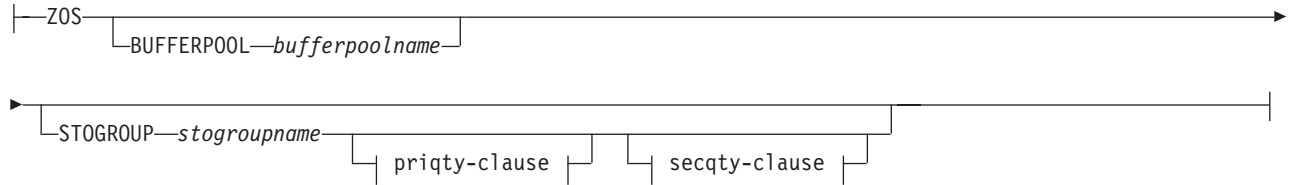
**zos-tbs-clause:**



**uw-tbs-clause:**



**zos-idx-clause:**



**priqty-clause:**



**secqty-clause:**



## 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 that is used by the Q Capture program, Q Apply program, or both. You cannot use this command with non-relational sources.

### Syntax

```
▶▶ SET QMANAGER "qmgrname" FOR 

|                |
|----------------|
| CAPTURE SCHEMA |
| APPLY SCHEMA   |
| NODE number    |

▶▶
```

### Parameters

*"qmgrname"*

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.

#### NODE

Specifies one server in a multidirectional configuration. If this keyword is specified, the ASNCLP program uses the same value for *"qmgrname"* for both the Q Capture server and Q Apply server.

### 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 QM1 for both the Q Capture and Q Apply programs on a server that is used in bidirectional or peer-to-peer replication:  
SET QMANAGER FOR NODE 1 "QM1";

---

## 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 BIDI NODE 1 SERVER DBALIAS SAMPLE;
SET BIDI NODE 2 SERVER DBALIAS 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 PEER NODE 1 SERVER DBALIAS SAMPLE;
SET PEER NODE 2 SERVER DBALIAS SAMPLE1;
SET PEER NODE 1 SERVER DBALIAS SAMPLE2;

SET REFERENCE TABLE USING SCHEMA "SAMPLE".RED USES TABLE RED.EMPLOYEE;

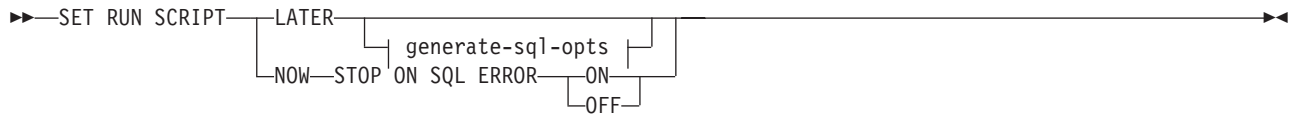
DROP SUBTYPE P QSUBS;
```

---

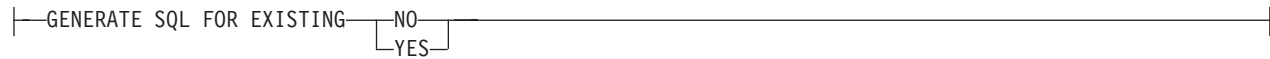
## SET RUN SCRIPT command

Use the **SET RUN SCRIPT** command to control whether to automatically run SQL statements that are generated by each ASNCPL task command before processing the next command or to manually run them later in a DB2 command prompt. You cannot use the LATER parameter with non-relational sources.

## Syntax



### generate-sql-opts:



## Parameters

### LATER

Specify to run the SQL scripts at a later time. You cannot use this parameter with Classic sources. Use this option if you want to verify your script before you run it. You can also use this option if you want to create SQL script files on one operating system, but run them on another.

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.

### NOW

Specify to automatically execute the SQL scripts.

### STOP ON SQL ERROR

Specifies whether the ASNCLP continues to process commands in the ASNCLP script file and statements in the generated SQL script file after one of the following errors:

- **ASNCLP script file:** An error while checking to predict whether the SQL statement to be generated will cause an SQL error. For example, a Q subscription cannot be defined in the control tables unless the control tables exist first.
- **Generated SQL script file:** An SQL error while running the SQL statements.

### ON (default)

Specify if you want the ASNCLP to stop processing commands in the ASNCLP script, and stop processing SQL statements in the generated SQL script, when the first validity check fails or SQL statement fails. If the error occurs while the ASNCLP is running the SQL script, previous SQL statements that are related to the task command with an error are rolled back.

**Note:** If the source scripts run correctly and the SQL statements in the scripts were committed but the target scripts have an SQL error, only the target scripts are rolled back. The committed source statements are not rolled back.

### OFF

Specify to process the ASNCLP commands and run all of the SQL statements, regardless of errors. You cannot use this parameter with Classic sources.

For a more complete explanation of how the ASNCLP responds to errors depending on this and other SET RUN SCRIPT options, see How the ASNCLP handles errors while processing scripts.

#### **GENERATE SQL FOR EXISTING**

Specifies whether to generate SQL when ASNCLP encounters errors because of duplicate (already existing) objects when processing **CREATE** commands. This option has no effect on **DROP** commands.

**NO** The ASNCLP program does not generate SQL to create objects that already exist. This is the default.

#### **YES**

The ASNCLP program continues to generate SQL statements even if it encounters existing object errors for the following commands:

#### **CREATE CONTROL TABLES**

Another set of control tables already exist under the same schema or table spaces are specified to be created but they already exist.

#### **CREATE REPLQMAP**

Another replication queue map with the same name already exists.

#### **CREATE QSUB**

Another Q subscription with the same name already exists, a target table already exists but the option in the **CREATE QSUB** command is to create the target table, the target table already exists but the option to create the table space was specified, or a unique index with the same name already exists.

### **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 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 might 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 SCRIPT NOW** be used.

Figure 3 on page 240 shows these dependencies for Q replication to a relational source. This figure does not apply to non-DB2 sources.

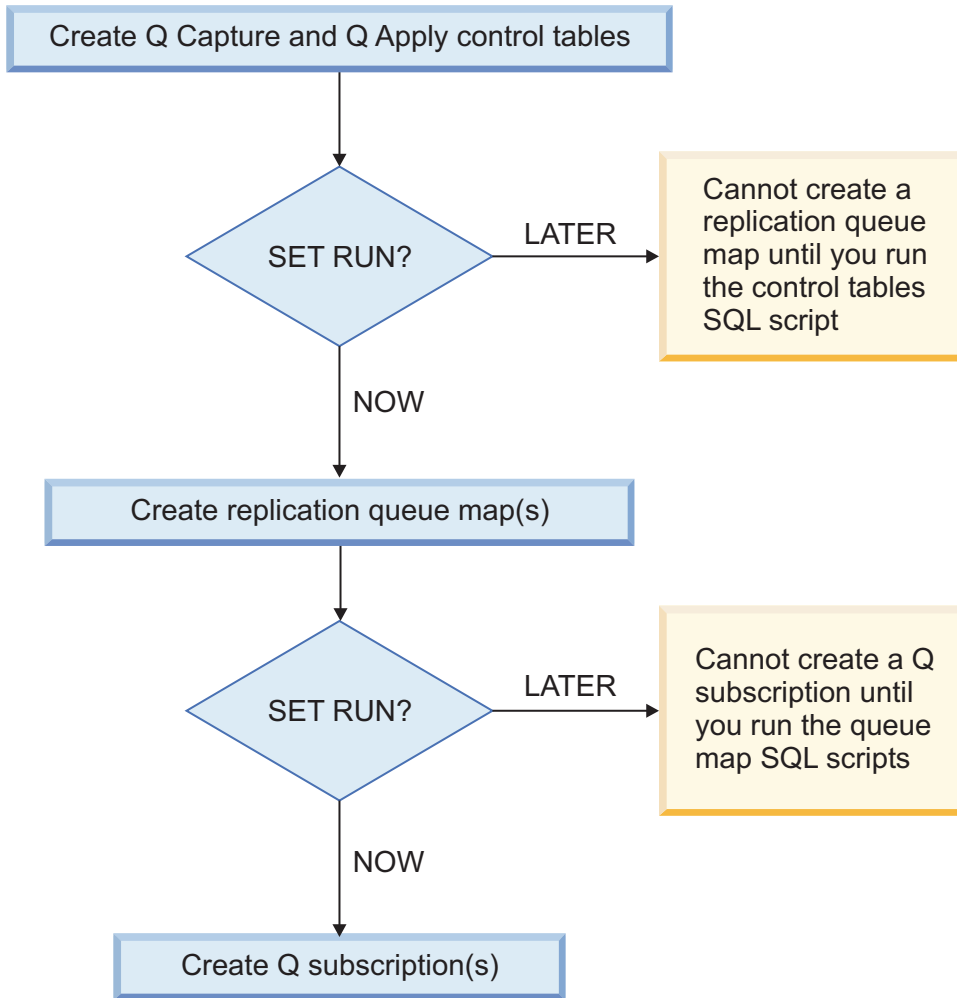


Figure 3. Dependencies between ASNCPL commands for Q replication from a DB2 source. This diagram shows the dependencies between ASNCPL CREATE commands that are used to set up Q replication. It assumes all objects use the default schema of ASN. The dependencies for Q Capture controls tables, publishing queue maps, and publications that are used in event publishing are the same.

### Example - Run immediately and stop on errors

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

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

### Example - Create SQL script and ignore errors when creating existing objects

To generate the SQL scripts instead of running them immediately, and to continue generating SQL when creating objects that already exist:

```
SET RUN SCRIPT LATER GENERATE SQL FOR EXISTING YES
```

---

## 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.

### Usage notes

If no SET SUBGROUP command is provided, the ASNCLP program generates a unique name with a number that increments for every new subgroup name that is needed.

### Example

To set the subgroup BLUEandRED:

```
SET SUBGROUP BLUEandRED
```

---

## 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 signal the Q Capture program or the Classic capture components to start one or more Q subscriptions.

### Syntax

▶▶—START— QSUB—SUBNAME—*subname*—▶▶  
┌──FOR SUBNAME LIKE—"*%text%*"──┐ ┌──CAP SERVER OPTIONS──┐ *classic-opt-clause* ┐  
└──source-table-options──┘ └──┘ └──┘

▶—START HISTORY—┐ YES ┐  
└──NO ┘▶▶

#### source-table-options:

┌──FOR TABLES──┐  
└──OWNER LIKE—"*%owner%*"──┘ └──NAME LIKE—"*%name%*"──┘

#### classic-opt-clause:

┌──DB—*dbalias*──┐  
└──DBALIAS—*aliasname*──┘ └──DBNAME—*dbname*──┘  
└──CONFIG NAME—*servername*──┘ └──FILE—*filename*──┘

▶ ┌──ID—*userid*──┘ └──PASSWORD—*pwd*──┘ └──CAPSCHEMA—*schema*──┘

## 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%*"

source-table-options

**FOR TABLES**

Use this clause to specify multiple schemas, multiple source tables, or both for which to start Q subscriptions.

**OWNER LIKE** *"%owner%"*

Specifies a single database schema or schema pattern that uses the percentage sign (%) as a wild card.

**NAME LIKE** *"%name%"*

Specifies a single table name or table-naming pattern that uses the percentage sign (%) as a wild card.

classic-opt-clause:

These parameters work only with Classic sources. If you have already specified these parameters in a previous **SET SERVER** command, you do not have to specify them again in this command.

**DB** *dbalias*

Specifies the database alias name of a z/OS subsystem or Linux, UNIX, or Windows database as cataloged on the DB2 from which the ASNCLP is invoked. This keyword is deprecated.

**DBALIAS** *aliasname*

Specifies the database alias name of a z/OS subsystem or Linux, UNIX, or Windows database as cataloged on the DB2 from which the ASNCLP is invoked.

**DBNAME** *zosdbname*

 Specifies the z/OS database name. This is a logical z/OS database name, as created on a z/OS subsystem.

**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.

**CAPSCHEMA** *schema*

Specifies the schema of the control tables.

**CONFIG NAME** *servername*

Specifies which server configuration settings from the Classic replication configuration file that the ASNCLP uses to connect to the Classic data server.

**FILE** *filename*

Specifies the complete path and file name to the Classic replication configuration file. If you do not use the **FILE** parameter, the ASNCLP attempts to use the `asnservers.ini` file in the current directory, if that file exists.

**START HISTORY**

Specifies whether you want to start the Q subscription for the history table when you start the Q subscription for the associated temporal table on DB2 10 for z/OS or later.

**YES (default)**

Start the Q subscription for the history table.

**NO** Do not start the Q subscription for the history table.

**Usage notes**

The CAP SERVER OPTIONS parameter overrides any settings that you specified in a previous SET command.

## Example: Classic replication with server information in START QSUB command

To start a Q subscription from a Classic source by specifying server information in the START QSUB command:

```
START QSUB SUBNAME sub1 CAP SERVER OPTIONS CONFIG NAME classic1
FILE asnservers.ini ID id1 PASSWORD passwd1;
```

## Example: Classic replication with server information in SET SERVER command

To start a Q subscription from a Classic source by specifying server information in a separate SET command:

```
SET SERVER CAPTURE CONFIG SERVER NAME classic1
FILE asnservers.ini ID id1 PASSWORD passwd1;
START QSUB SUBNAME sub1;
```

## Example: Starting multiple Q subscriptions on multiple servers based on schema pattern

To start all of the bidirectional Q subscriptions on the SAMPLE1 and SAMPLE2 servers that are defined under schemas that start with "AIRUKU":

```
SET BIDI NODE 1 SERVER DBALIAS SAMPLE1;
SET BIDI NODE 2 SERVER DBALIAS SAMPLE2;

SET RUN SCRIPT NOW STOP ON SQL ERROR ON;

START QSUB FOR TABLES OWNER LIKE "AIRUKU%";
```

---

## START SCHEMASUB command

Use the **START SCHEMASUB** command to generate a script that prompts the Q Capture program to start capturing DDL changes for a schema-level subscription. You can also use this command to prompt Q Capture to start capturing DML changes for the inactive and new table-level Q subscriptions within the schema.

### Syntax

```
▶▶—START SCHEMASUB—schema_sub_name—ALL—NEW ONLY—▶▶
```

### Parameters

#### ALL

Specify to start capturing DDL changes for a schema-level subscription and DML changes for all of inactive and new the table-level Q subscriptions that belong to it. The command generates a SQL script to insert a START\_SCHEMASUB signal into the IBMQREP\_SIGNAL table at the Q Capture server for the schema-level subscription, and CAPSTART signals for the table-level Q subscriptions. You can use the SET RUN SCRIPT NOW option to immediately insert the signals.

#### NEW ONLY

Specify to start only the schema-level subscription.

## Example

To start capturing DDL changes for the schema-level subscription `schemasub1` and DML changes for all of its inactive and new table-level Q subscriptions, and to start capturing DDL only for the schema-level subscription `schemasub2`:

```
SET BIDI NODE 1 SERVER DBALIAS SAMPLE1;  
SET BIDI NODE 2 SERVER DBALIAS SAMPLE2;  
  
SET RUN SCRIPT NOW STOP ON SQL ERROR ON;  
  
START SCHEMASUB schemasub1 ALL;  
START SCHEMASUB schemasub2 NEW ONLY;
```

---

## STOP QSUB command

Use the **STOP QSUB** command to signal the Q Capture program or the Classic capture components to stop one or more Q subscriptions.

### Syntax

```
▶▶ STOP QSUB SUBNAME subname  
└─┬─ FOR SUBNAME LIKE "%text%" ─┬─ CAP SERVER OPTIONS ─┬─ classic-opt-clause ─┬─  
  └─ source-table-options ─┘
```

```
▶ STOP HISTORY ┌─ YES ─┘  
               └─ NO ─┘
```

#### source-table-options:

```
┌─ FOR TABLES ─┬─ OWNER LIKE "%owner%" ─┬─ NAME LIKE "%name%" ─┬─
```

#### classic-opt-clause:

```
┌─ DB dbalias ─┬─ DBALIAS aliasname ─┬─ DBNAME dbname ─┬─ ID userid ─┬─ PASSWORD pwd ─┬─  
└─ CONFIG SERVER servername ─┬─ FILE filename ─┘
```

```
▶ ┌─ CAPSCHEMA schema ─┘
```

## 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%"
```

source-table-options

**FOR TABLES**

Use this clause to specify multiple schemas, multiple source tables, or both for which to stop Q subscriptions.

**OWNER LIKE** *"%owner%"*

Specifies a single database schema or schema pattern that uses the percentage sign (%) as a wild card.

**NAME LIKE** *"%name%"*

Specifies a single table name or table-naming pattern that uses the percentage sign (%) as a wild card.

classic-opt-clause:

These parameters work only with Classic sources. If you have already specified these parameters in a previous **SET SERVER** command, you do not have to specify them again in this command.

**DB** *dbalias*

Specifies the database alias name of a z/OS subsystem or Linux, UNIX, or Windows database as cataloged on the DB2 from which the ASNCLP is invoked. This keyword is deprecated.

**DBALIAS** *aliasname*

Specifies the database alias name of a z/OS subsystem or Linux, UNIX, or Windows database as cataloged on the DB2 from which the ASNCLP is invoked.

**DBNAME** *zosdbname*

 Specifies the z/OS database name. This is a logical z/OS database name, as created on a z/OS subsystem.

**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.

**CAPSCHEMA** *schema*

Specifies the schema of the control tables.

**CONFIG SERVER** *servername*

Specifies which server configuration settings from the Classic replication configuration file that the ASNCLP uses to connect to the Classic data source.

**FILE** *filename*

Specifies the complete path and file name to the Classic replication configuration file. If you do not use the **FILE** parameter, the ASNCLP attempts to use the `asnservers.ini` file in the current directory, if that file exists.

**STOP HISTORY**

Specifies whether you want to stop the Q subscription for the history table when you stop the Q subscription for the associated temporal table on DB2 10 for z/OS or later.

**YES (default)**

Stop the Q subscription for the history table.

**NO** Do not stop the Q subscription for the history table.

## Usage notes

The CAP SERVER OPTIONS parameter overrides any settings that you specified in a previous SET command.

## Example

To stop a Q subscription:

```
STOP QSUB SUBNAME EMPLOYEE0001;
```

## Example: Stopping multiple Q subscriptions on multiple servers based on schema pattern

To stop all of the bidirectional Q subscriptions on the SAMPLE1 and SAMPLE2 servers that are defined under schemas that start with "AIRUKU":

```
SET BIDI NODE 1 SERVER DBALIAS SAMPLE1;  
SET BIDI NODE 2 SERVER DBALIAS SAMPLE2;
```

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

```
START QSUB FOR TABLES OWNER LIKE "AIRUKU%";
```

---

## STOP SCHEMASUB command

Use the **STOP SCHEMASUB** command to generate a script that prompts the Q Capture program to stop capturing DDL changes for a schema-level subscription. You can also use this command to prompt Q Capture to stop capturing DML changes for the table-level Q subscriptions within the schema.

## Syntax

```
▶▶—STOP SCHEMASUB—schema_sub_name—

|          |
|----------|
| ALL      |
| NEW ONLY |

—▶▶
```

## Parameters

### ALL

Specify to stop capturing DDL changes for a schema-level subscription and DML changes for all of the table-level Q subscriptions that belong to it. The command generates a SQL script to insert a STOP\_SCHEMASUB signal into the IBMQREP\_SIGNAL table at the Q Capture server for the schema-level subscription, and CAPSTOP signals for the table-level Q subscriptions. You can use the SET RUN SCRIPT NOW option to immediately insert the signals.

### NEW ONLY

Specify to stop only the schema-level subscription.

## Example

To stop capturing DDL changes for the schema-level subscription schemasub1 and DML changes for all of its table-level Q subscriptions, and also to stop capturing DDL for only the schema-level subscription schemasub2:

```
SET BIDI NODE 1 SERVER DBALIAS SAMPLE1;  
SET BIDI NODE 2 SERVER DBALIAS SAMPLE2;
```

```

SET RUN SCRIPT NOW STOP ON SQL ERROR ON;

STOP SCHEMASUB schemasub1 ALL;
STOP SCHEMASUB schemasub2 NEW ONLY;

```

---

## 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
```



---

## Chapter 6. ASNCLP commands for Event Publishing

The ASNCLP commands for Event Publishing define and change publishing queue maps and publications. The commands also can be used to start and stop publications.

“Sample ASNCLP scripts for setting up Event Publishing” on page 252 demonstrates how you can combine Event Publishing commands to create an ASNCLP setup script.

Table 7 lists the ASNCLP commands for Event Publishing and links to topics that describe each command.

*Table 7. ASNCLP commands for Event Publishing*

If you want to ...	Use this command
Add a column to a publication	ALTER ADD COLUMN command
Change a publishing queue map	“ALTER PUBQMAP command” on page 254
Change a publication	“ALTER PUB command” on page 255
Create the control tables for the Q Capture program	CREATE CONTROL TABLES FOR command
Create a publishing queue map	“CREATE PUBQMAP command” on page 266
Create a publication	“CREATE PUB command” on page 268
Drop the control tables for the Q Capture program	DROP CONTROL TABLES ON command
Delete a publishing queue map	“DROP PUBQMAP command” on page 273
Delete a publication	“DROP PUB command” on page 274
List publications	“LIST PUBS command” on page 274
List publishing queue maps	“LIST PUBQMAPS command” on page 275
List Q Capture schemas	LIST CAPTURE SCHEMA command
Promote a publication	“PROMOTE PUB command” on page 277
Promote a publishing queue map	“PROMOTE PUBQMAP command” on page 279
Start a publication	“START PUB command” on page 288
Set the Q Capture schema for all task commands	SET CAPTURE SCHEMA command
Define the log file for the ASNCLP program	SET LOG command
Define output files that contain SQL statements to set up Event Publishing	SET OUTPUT command
Set the WebSphere MQ queue manager	SET QMANAGER command
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
Specify the Q Capture server to use in the ASNCLP session	SET SERVER command
Enable and disable the trace for the ASNCLP commands	SET TRACE command
Display the environment set during the session	SHOW SET ENV command
Stop a publication	“STOP PUB command” on page 289

Table 7. ASNCLP commands for Event Publishing (continued)

If you want to ...	Use this command
Verify that the required WebSphere MQ objects exist and have the correct properties for schemas, queue maps, and publications.	VALIDATE WSMQ ENVIRONMENT FOR command

## Sample ASNCLP scripts for setting up Event Publishing

This sample contains two ASNCLP scripts for setting up a basic Event Publishing environment. The first script creates WebSphere MQ objects. The second script creates Q Capture control tables, a publishing queue map, and a publication.

You can copy the scripts to a text file, modify the values, and run the scripts by using the ASNCLP *-f filename* command. First:

- **Script 1:** Change the values for the MQHOST keyword to the IP address of the SAMPLE database, and ensure that the user ID that starts the ASNCLP program has permissions to execute the generated batch or shell script file.
- **Script 2:** Change db2admin and "passw0rd" to the user ID and password for connecting to SAMPLE.

**Prerequisite:** The scripts require the replication administration tools to be at Version 9.7 Fix Pack 4.

### ASNCLP script 1: Create WebSphere MQ objects

```
#####
ASNCLP SESSION SET TO Q REPLICATION;

CREATE MQ SCRIPT RUN NOW
CONFIG TYPE E
MQSERVER 1 NAME SAMPLE MQHOST "9.30.54.118";

QUIT;
#####
```

**Notes:** The CREATE MQ SCRIPT command generates one shell script file for Linux and UNIX systems (qrepl.sample.mq\_aixlinux.sh) and one batch file for Windows systems (qrepl.sample.mq\_windows.bat). If you run the ASNCLP program on the same system as SAMPLE, the RUN NOW option prompts the ASNCLP program to run the batch file or shell script to define the queue managers, queues, and other WebSphere MQ objects. If the ASNCLP program is remote from SAMPLE, you must run the appropriate batch file or shell script at the system where SAMPLE resides.

### ASNCLP script 2: Create publishing objects

```
#####
ASNCLP SESSION SET TO Q REPLICATION;
SET SERVER CAPTURE TO DB SAMPLE ID db2admin PASSWORD "passw0rd";
SET RUN SCRIPT NOW STOP ON SQL ERROR ON;

CREATE CONTROL TABLES FOR CAPTURE SERVER;

CREATE PUBQMAP SAMPLE_ASN_TO_SUBSCRIBER;

CREATE PUB USING PUBQMAP SAMPLE_ASN_TO_SUBSCRIBER
(PUBNAME "DEPARTMENT0001" db2admin.DEPARTMENT ALL CHANGED ROWS Y
SUPPRESS DELETES Y);
```

```
QUIT;
#####
```

**Notes:** The commands in this script perform the following actions:

- The SET RUN SCRIPT NOW option prompts the ASNCLP program to generate SQL scripts for creating publishing objects and then run the scripts. This option is required because some objects must be in place before others are created. For example, the Q Capture control tables must be created before you can define a publication within them.
- For both the control tables and queue map, the ASNCLP program by default uses the WebSphere MQ objects that were created with the CREATE MQ SCRIPT command.
- The CREATE PUB command generates SQL to create a 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.

---

## ALTER ADD COLUMN command (Event Publishing)

Use the **ALTER ADD COLUMN** command to add a column to a publication.

### Syntax

```
▶▶—ALTER ADD COLUMN USING SIGNAL—(—colname—)—PUB—pubname—  

                                     └—WITH BEFORE IMAGE—┘
```

---

```
▶—SOURCE—table_owner.table_name—▶▶
```

### Parameters

*colname*

Specifies one or more columns (separated by a comma) to add to the definition of the active publication.

**PUB** *pubname*

Specifies the name of the publication.

**WITH BEFORE IMAGE**

Specifies that the before-image value of each added column will be published.

**SOURCE** *table\_owner.table\_name*

Specifies that the columns are added to all publications and Q subscriptions for the source table.

### Usage notes

- The column needs to exist in the source table already and should not be part of any existing publication.
- The publication must be active.
- The column must be nullable or have a default value 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 with SQL.

- A maximum of 20 columns can be inserted into the statement.

### Example 1

To add the columns PHONE and ADDRESS to the EMPLOYEE0001 publication:

```
ALTER ADD COLUMN USING SIGNAL (PHONE, ADDRESS) PUB EMPLOYEE0001;
```

### Example 2

To add the PHONE, ADDRESS, and EMAIL columns to all publications and Q subscriptions for the EMPLOYEE table:

```
ALTER ADD COLUMN USING SIGNAL (PHONE, ADDRESS, EMAIL) SOURCE DB2ADMIN.EMPLOYEE;
```

---

## 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 | R } [SENDQ sendqname]]
```

```
► [ERROR ACTION { S | Q } [HEARTBEAT INTERVAL interval] [MAX MESSAGE SIZE size]]
```

```
► [HEADER { NONE | MQ RFH2 } [ON CODEPAGE ERROR { SEND NO DATA | SEND RAW DATA }]]
```

### 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**

The action that the Q Capture program takes 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.

**S** The Q Capture program stops.

**Q** The Q Capture program stops putting messages on any send queues that are in error and continues putting messages on other send queues.

**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-compliant MQ RFH2 header added to all messages that use the send queue that is specified in this publishing queue map.

**NONE**

Specify to send only the publication message with no special headers.

**MQ RFH2**

Specify to attach a special header to the message that will contain the topic name that you specify as part of an publication.

**ON CODEPAGE ERROR**

Specifies whether you want to send data when code page conversion errors occur.

**SEND NO DATA**

The Q Capture program does not send the data when an error occurs during code page conversion.

**SEND RAW DATA**

The Q Capture program sends hex representation of the character data if a code page conversion error occurs.

**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 PUB command**

Use the **ALTER PUB** command to change the properties of a publication.

## Syntax

```
ALTER PUB pubname FOR source_owner. source_name DESC "description"
PUBQMAP qmapname OPTIONS opt-clause
```

### opt-clause:

```
SEARCH CONDITION "search_cond" ALL CHANGED ROWS N
CHANGE CONDITION "change_condition" BEFORE VALUES N
CHANGED COLS ONLY Y SUPPRESS DELETES N TOPIC "topic"
```

## Parameters

### **PUB** *pubname*

Specifies the name of the publication.

### *source\_owner*

Specifies the source table schema.

### *source\_name*

Specifies the source table name.

### **DESC** "*description*"

Specifies a description of the publication.

### **PUBQMAP** *qmapname*

Specifies the new name of the publishing queue map that is used by this 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 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.

#### **CHANGE CONDITION** "*change\_condition*"

Specifies a predicate that uses log record variables for filtering changes to publish.



You can use the following log record variables:

\$OPERATION	The DML operation. Valid values are I (insert), U (update), and D (delete).
\$AUTHID	The authorization ID of a transaction.
\$AUTHTOKEN	<b>z/OS:</b> The authorization token (job name) of a transaction.
\$PLANNAME	<b>z/OS:</b> The plan name of a transaction.

For example, the following predicate specifies that Q Capture only publish log records that were not committed by the user ASN:

```
$AUTHID <> 'ASN'
```

If a different predicate is specified by using the **SEARCH CONDITION** keyword, that predicate is combined with the **CHANGE CONDITION** predicate into a single predicate by using the AND operator. For more details on the format for **CHANGE CONDITION**, see Log record variables to filter rows.

#### **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 a 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 a 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 a publication. This is the default.
- N** The Q Capture program sends all columns in a row that are part of a 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 publication. You must specify the HEADER MQ RFH2 keywords when you create the publishing queue map that this publication uses.

## Example

To alter the 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 PUB MYXMLPUB FOR ERIC.TSTTABLE OPTIONS ALL CHANGED ROWS N  
BEFORE VALUES N CHANGED COLS ONLY N SUPPRESS DELETES N
```

---

## CREATE CONTROL TABLES FOR command

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.

In Classic replication, the control tables for the Classic capture components are creating by using the Classic Data Architect.

### Syntax

```
▶▶ CREATE CONTROL TABLES FOR {  
  CAPTURE SERVER USING capparms-clause |  
  APPLY SERVER USING applyparms-clause |  
  NODE-number USING node-options };
```

```
  [ZOS INDEX] zos-idx-clause;
```

#### node-options:

```
[CAPPARMS] capparms-clause [APPPARMS] applyparms-clause;
```

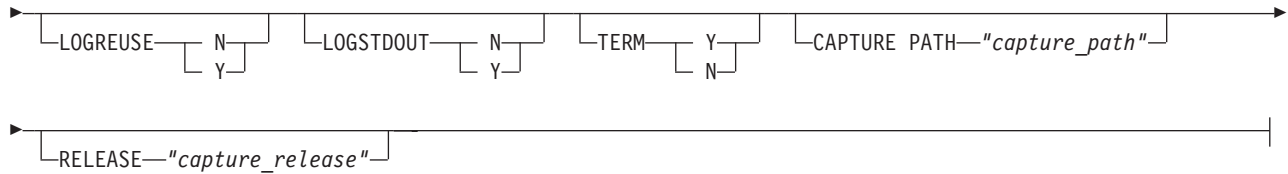
#### capparms-clause:

```
[IN {  
  [ZOS] zos-ts-clause |  
  [UW] uw-ts-clause }  
  RESTARTQ "rstqname" ADMINQ "admqname";
```

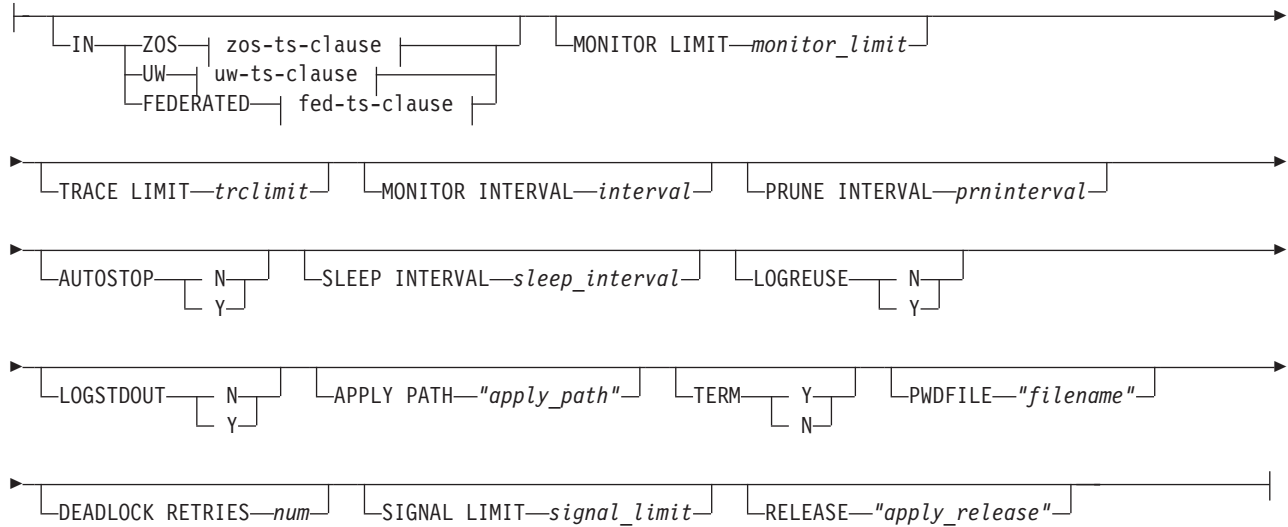
```
  [STARTMODE {  
    [WARMSI] |  
    [COLD] |  
    [WARMNS] }  
    [MEMORY LIMIT] limit [AUTOSTOP] {  
      [N] |  
      [Y] };
```

```
  [MONITOR INTERVAL] interval [MONITOR LIMIT] monlimit [TRACE LIMIT] trclimit;
```

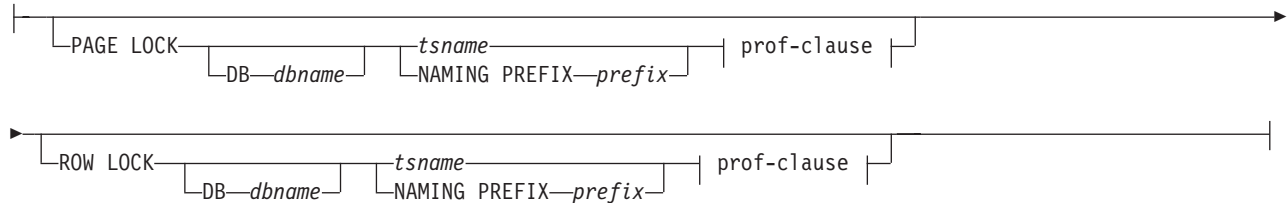
```
  [SIGNAL LIMIT] siglimit [PRUNE INTERVAL] prninterval [SLEEP INTERVAL] sleepinterval;
```



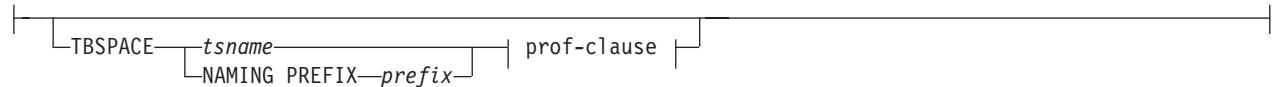
**applyparms-clause:**



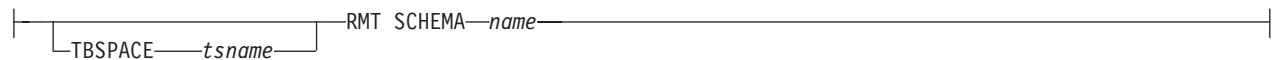
**zos-ts-clause:**



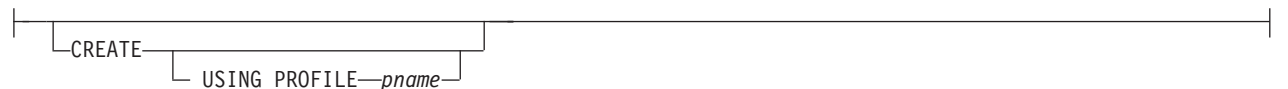
**uw-ts-clause:**



**fed-ts-clause:**



**prof-clause:**



**zos-idx-clause:**



**Parameters**

**CAPTURE SERVER**

Specify to create Q Capture control tables.

**APPLY SERVER**

Specify to create Q Apply control tables.

**NODE**

Specify to generate a script for creating both Q Capture and Q Apply control tables with the same schema on one server in a multidirectional replication configuration.

**Note:** Use this option only in conjunction with the SET BIDI NODE command for specifying the servers that are involved in multidirectional replication.

**CAPPARMS**

Specify to set options for the Q Capture control tables.

**APPARMS**

Specify to set options for the Q Apply control tables.

capparms-clause:

**ZOS**

Specifies a z/OS system on which to create Q Capture control tables.

**UW** Specifies a Linux, UNIX, or Windows system on which to create Q Capture control tables.

**RESTARTQ** "rstqname"

Specifies the restart queue that the Q Capture program uses.

**ADMINQ** "admname"

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 is quiesced or stops. This value is the default.
- N** The Q Capture program continues running if DB2 is quiesced or stops.

**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.

**Linux UNIX Windows** **RELEASE** *"capture\_release"*

Specifies the release level of the control tables that you want to create. Allowed values are 9.7, 9.5, and 9.1. This parameter is for Linux, UNIX, and Windows

only. Enclose value in double quotation marks ("). Specifying the release level enables newer replication and publishing function on an older DB2.

appparms-clause:

#### **ZOS**

Specifies a z/OS system in which Q Apply control tables are created.

**UW** Specifies a Linux, UNIX, or Windows system in which Q Apply control tables are created.

#### **FEDERATED**

Specifies a federated target, on which Q Apply control tables are created in an Oracle, Sybase, Informix, Microsoft SQL Server, or Teradata database, and nicknames are created for these control tables in the Q Apply server. Some control tables are created in the Q Apply server.

#### **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 is quiesced or stops.

**N** The Q Apply program continues running if DB2 is quiesced or stops.

**PWDFILE** *"filename"*

Specifies the name of the password file.

**DEADLOCK RETRIES** *num*

Specifies the number of retries for SQL deadlock errors.

**Linux UNIX Windows** **RELEASE** *"apply\_release"*

Specifies the release level of the control tables that you want to create. Allowed values are 9.7, 9.5, and 9.1. This parameter is for Linux, UNIX, and Windows only. Enclose value in double quotation marks ("). Specifying the release level enables newer replication and publishing function on an older DB2.

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. This parameter is not applicable for Teradata targets.

**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
```

## Example 4

To create Version 9.7 Q Apply control tables on a DB2 Version 9.1 database:

```
CREATE CONTROL TABLES FOR APPLY SERVER USING RELEASE "9.7"
```

---

## CREATE MQ SCRIPT command (Event Publishing)

Use the **CREATE MQ SCRIPT** command to generate scripts for creating all of the WebSphere MQ objects that are needed for Event Publishing.

### Syntax

```
►► CREATE MQ SCRIPT [RUN NOW] CONFIG TYPE E | mq-clause |
```

#### mq-clause:

```
| MQSERVER number NAME name | options |
```

#### options:

```
| MQHOST hostname | MQPORT port_number | QMANAGER queue_manager | QNAME_QUAL qualifier |
```

### Parameters

#### RUN NOW

Specifies that you want the ASNCLP program to run the generated WebSphere MQ script after it is created. The queue manager and ASNCLP program must be on the same system for you to use this option.

#### CONFIG TYPE

Specifies the type of configuration:

**E** Event Publishing



mq-clause

### MQSERVER

A number that identifies the Q Capture server. The numbers differ depending on the configuration type:

#### Event Publishing

Use 1 to represent the Q Capture server.

### NAME

The subsystem name or database alias of the Q Capture server.

options

### MQHOST

The hostname or IP address of the system that contains the queue manager that will create the WebSphere MQ objects.

### MQPORT

The port number that the channel listener monitors for incoming requests. If this keyword is not specified, the ASNCLP program uses the default WebSphere MQ port number 1414.

### QMANAGER

The queue manager that will be created, and that will be used to create other WebSphere MQ objects. If this keyword is not specified, the value that was specified for the **NAME** keyword is used to name the queue manager.

### QNAME\_QUAL

A qualifier that is used for the generated queue names. The default is ASN, which is the default Q Capture schema. This qualifier can help identify queues at the Q Capture system.

## Usage notes

- **Linux UNIX Windows** The default file name for the generated script is `qrepl.server_name.mq`, where `server_name` is the server alias that was specified in the CREATE MQ SCRIPT command. The scripts are executable files in either the .bat or .exe format depending on whether the ASNCLP program runs on Windows or Linux-UNIX.

- **z/OS** If the ASNCLP program is running natively on z/OS, the output DD name for the generated script is OUTMQCAP, OUTMQTRG, and OUTMQx. The following lines must be included in the JCL:

```
//OUTMQCAP DD DSN=&SYSUID..ASNCLP.OUTNODE1,DISP=(NEW,CATLG,DELETE),  
//          UNIT=SYSDA,SPACE=(TRK,(30,10))  
//OUTMQTRG DD DSN=&SYSUID..ASNCLP.OUTNODE1,DISP=(NEW,CATLG,DELETE),  
//          UNIT=SYSDA,SPACE=(TRK,(30,10))
```

The generated script will be wrapped to 80 characters per line. Comments are included with changes that need to be made for z/OS.

- You can specify the CREATE MQ SCRIPT command in the same input file as other ASNCLP commands, but this command does not use the server and schema information from any previous SET commands.

## Example 1

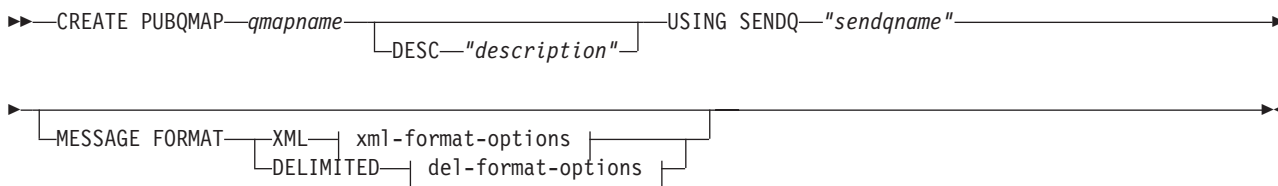
To generate a script that creates WebSphere MQ objects for event publishing:

```
CREATE MQ SCRIPT CONFIG TYPE E  
MQSERVER 1 NAME SOURCEDB MQHOST "9.30.54.118" MQPORT "1414";
```

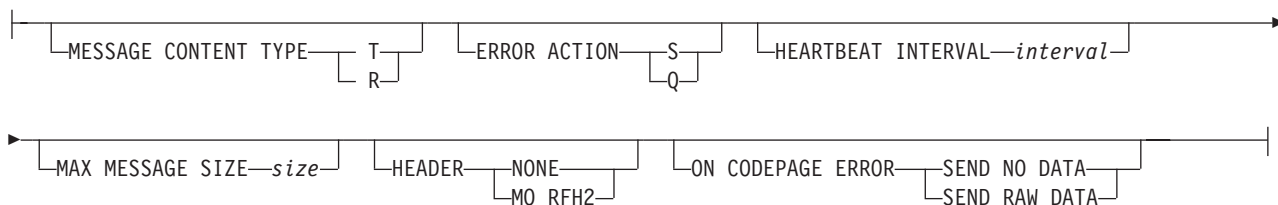
## CREATE PUBQMAP command

Use the **CREATE PUBQMAP** command to create a publishing queue map that specifies the send queue to use for event publishing and whether to send messages in XML or delimited format.

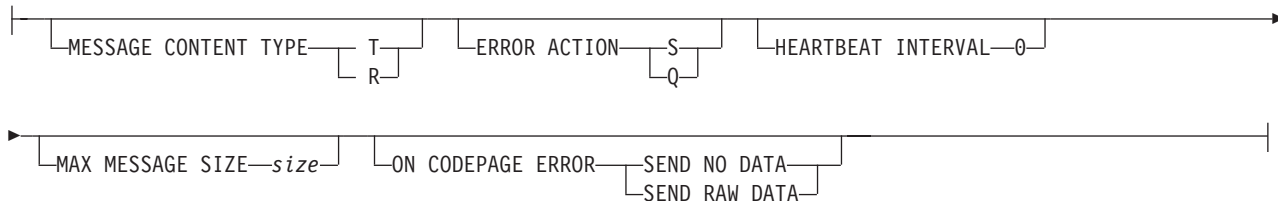
### Syntax



#### xml-format-options:



#### del-format-options:



### 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 FORMAT

Specifies whether you want to publish messages in XML format or delimited format. Use this keyword if you want to specify options for the publishing queue map; the options differ for the different message format types.

#### 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**

The action that the Q Capture program takes 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.

- S** The Q Capture program stops.

- Q** The Q Capture program stops putting messages on any send queues that are in error and continues putting messages on other send queues.

#### **HEARTBEAT INTERVAL** *interval*

**XML format only:** Specifies the interval (in seconds) between heartbeat messages that are 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. Heartbeat messages are not supported for the delimited message format, so the value of this keyword is always 0 for delimited format.

#### **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-compliant MQ RFH2 header added to all messages that use the send queue that is specified in this publishing queue map. This keyword is not supported for delimited message format.

#### **NONE**

Specify to send only the publication message with no special headers.

#### **MQ RFH2**

Specify to attach a special header to the message that will contain the topic name that you specify as part of an publication.

#### **ON CODEPAGE ERROR**

Specifies whether you want to send data when code page conversion errors occur.

#### **SEND NO DATA**

The Q Capture program does not send character data when an error occurs during code page conversion.

#### **SEND RAW DATA**

The Q Capture program sends a hexadecimal representation of character data if a code page conversion error occurs.

## **Example 1**

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 message size of 128 KB:

```
CREATE PUBQMAP SAMPLe_ASN1_TO_SUBSCRIBER USING
SENDQ "ASN1.QM1.PUBDATAQ" MESSAGE CONTENT TYPE R
HEARTBEAT INTERVAL 5 MAX MESSAGE SIZE 128
```

## Example 2

To create a publishing queue map `SAMPLE_ASN_TO_DATASTAGE` that sets the message format to delimited, the message type to row, and a maximum message size of 256 KB:

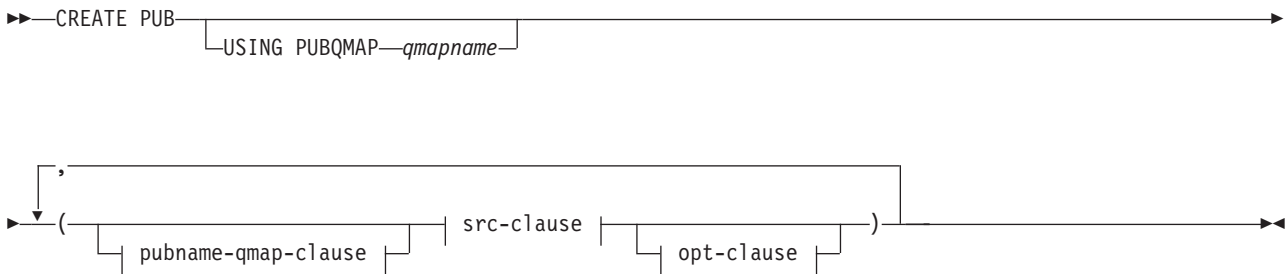
```
CREATE PUBQMAP SAMPLe_ASN_TO_DATASTAGE
USING SENDQ "ASN.QM1.DELIMDATAQ" MESSAGE FORMAT DELIMITED
MESSAGE CONTENT TYPE R HEARTBEAT INTERVAL 0 MAX MESSAGE SIZE 256
```

---

## CREATE PUB command

Use the **CREATE PUB** command to create a publication.

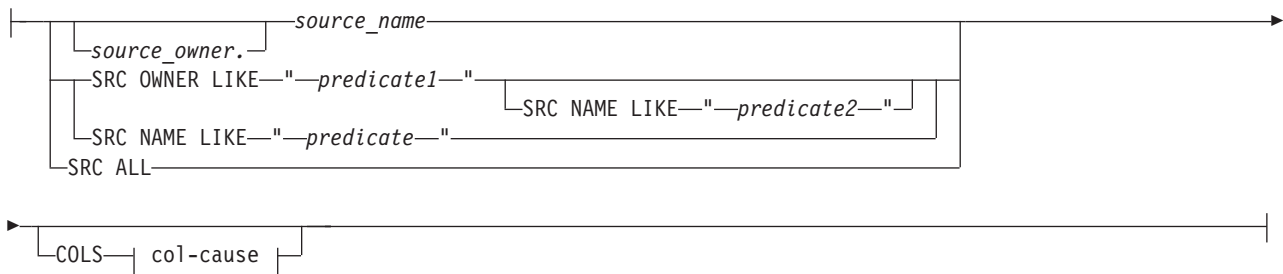
### Syntax



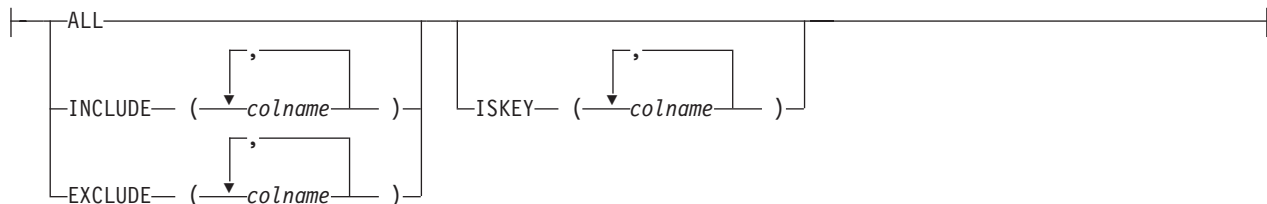
#### pubname-qmap-clause:



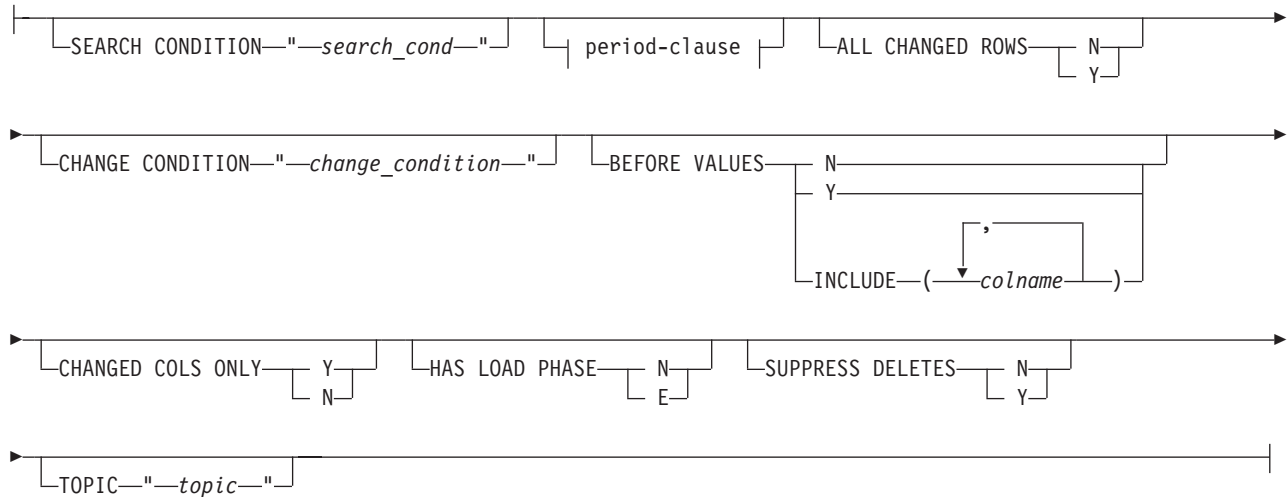
#### src-clause:



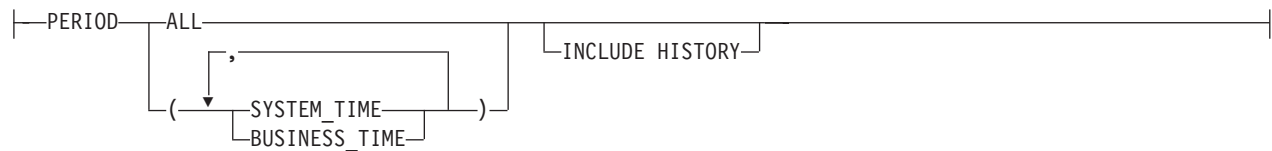
#### col-cause:



### opt-clause:



### period-clause:



## Parameters

### USING PUBQMAP *qmapname*

Specifies the publishing queue map that is used by all subsequent publications that are created by this command.

### pubname-qmap-clause:

#### PUBNAME *pubname*

Specifies the name of the publication.

#### DESC "*description*"

Specifies a description of the publication.

#### PUBQMAP *qmapname*

Specifies the publishing queue map that is used by this publication. If you do not specify the **USING PUBQMAP** keyword, you must define the **PUBQMAP** keyword for every 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 PUB USING PUBQMAP ABCDPUBQMAP
(SRC OWNER LIKE "ASN%");
```

```
CREATE 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 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-cause:

**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 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.

**CHANGE CONDITION** *"change\_condition"*

Specifies a predicate that uses log record variables for filtering changes to publish.

You can use the following log record variables:

\$OPERATION	The DML operation. Valid values are I (insert), U (update), and D (delete).
-------------	---

\$AUTHID	The authorization ID of a transaction.
\$AUTHTOKEN	<b>z/OS:</b> The authorization token (job name) of a transaction.
\$PLANNAME	<b>z/OS:</b> The plan name of a transaction.

For example, the following predicate specifies that Q Capture only publish log records that were not committed by the user ASN:

```
$AUTHID <> 'ASN'
```

If a different predicate is specified by using the **SEARCH CONDITION** keyword, that predicate is combined with the **CHANGE CONDITION** predicate into a single predicate by using the AND operator. For more details on the format for **CHANGE CONDITION**, see Log record variables to filter rows.

#### 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 nonkey 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 nonkey columns in the source table that are part of a publication, the Q Capture program sends both before and after values.

#### INCLUDE (*colname*)

Specifies the nonkey columns for which 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 a 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 a publication. This is the default.
- N** The Q Capture program sends all columns in a row that are part of a publication whenever any of them has changed.

#### HAS LOAD PHASE

Specifies whether the target table for the 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 outside of replication. In this case, you insert the **LOADDONE** signal (by 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 publication. You must specify the HEADER MQ RFH2 keywords when you create the publishing queue map that this publication uses.

period-clause:

**PERIOD**

Specifies that the source table is a temporal table on DB2 10 for z/OS or later and you want to include some or all of the period columns in the publication.

**ALL**

Specifies that you want to include all period columns.

**SYSTEM\_TIME**

Specifies that you want to include the timestamp columns that are used with system-period temporal tables.

**BUSINESS\_TIME**

Specifies that you want to include the timestamp or date columns that are used with application-period temporal tables.

**INCLUDE HISTORY**

Specifies that you want to create a publication for the history table that is associated with the base temporal table.

## Example 1

To create a publication that uses 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 PUB USING PUBQMAP SAMPLE_ASN1_TO_SUBSCRIBER (PUBNAME "EMPLOYEE001"  
DB2ADMIN.EMPLOYEE ALL CHANGED ROWS Y BEFORE VALUES Y CHANGED COLS ONLY Y  
HAS LOAD PHASE N SUPPRESS DELETES Y)
```

## Example 2

To create a publication and specify that the capture program sends before values for the nonkey columns `C10`, `C11`, and `C13`:

```
ASNCLP SESSION SET TO Q REPLICATION;  
SET RUN SCRIPT NOW STOP ON SQL ERROR ON;  
SET SERVER CAPTURE TO DB APP1DB;  
SET CAPTURE SCHEMA SOURCE SAMPLE;  
CREATE PUB USING PUBQMAP 'PUBQ1' (PUBNAME PUB1 DATA.EMPLOYEE  
OPTIONS BEFORE VALUES INCLUDE(c10, c11, c12));
```

---

## DROP CONTROL TABLES ON command

Use the **DROP CONTROL TABLES ON** command to drop the Q Capture control tables, Q Apply control tables, or both. In Classic replication, you can use this command to drop only the Q Apply control tables.



## Syntax

```
▶▶—DROP CONTROL TABLES ON—  
    |—CAPTURE SERVER—  
    |—APPLY SERVER—  
    |—NODE—node_number—  
    └──────────────────┘
```

### Parameters

#### **CAPTURE SERVER**

Specify to drop the Q Capture control tables.

#### **APPLY SERVER**

Specify to drop the Q Apply control tables.

#### **NODE**

Specify to drop the Q Capture and Q Apply control tables on a server in a bidirectional or peer-to-peer configuration. The server is identified by *node\_number*.

### Usage notes

This command is used in conjunction with the **SET SERVER** command to indicate the location of the control tables.

### Example: Q Capture control tables

To drop the Q Capture control tables:

```
SET SERVER TARGET TO QAPPDB;  
DROP CONTROL TABLES ON APPLY SERVER
```

### Example: Dropping both sets of control tables

To drop both Q Capture and Q Apply control tables on the SAMPLE1 and SAMPLE2 servers:

```
SET BIDI NODE 1 SERVER DBALIAS SAMPLE1;  
SET BIDI NODE 2 SERVER DBALIAS SAMPLE2;  
  
SET RUN SCRIPT NOW STOP ON SQL ERROR ON;  
  
DROP CONTROL TABLES ON NODE 1;  
DROP CONTROL TABLES ON NODE 2;
```

---

## DROP PUBQMAP command

Use the **DROP PUBQMAP** command to delete an existing publishing queue map.

**Restriction:** The 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:

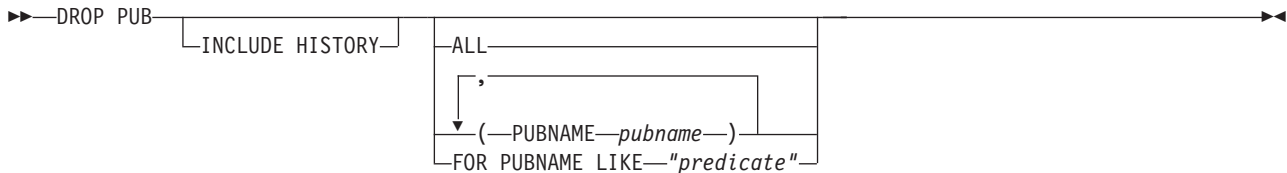
```
DROP PUBMAP SAMPLE_ASN1_TO_SUBSCRIBER
```

---

## DROP PUB command

Use the **DROP PUB** command to delete a publication.

### Syntax



### Parameters

#### ALL

Specify to delete all of the publications for the schema and server set through the SET commands.

#### PUBNAME *pubname*

Specifies the name of a publication to delete.

#### FOR PUBNAME LIKE "*predicate*"

Specify to delete all publications that match the LIKE statement. The following example shows a LIKE statement:

```
DROP PUB FOR PUBNAME LIKE "pubname02%"
```

#### INCLUDE HISTORY

Specify to delete the publication for the history table when the publication for the base temporal table is deleted.

## Example

To delete a publication:

```
DROP PUB (PUBNAME MYPUB)
```

---

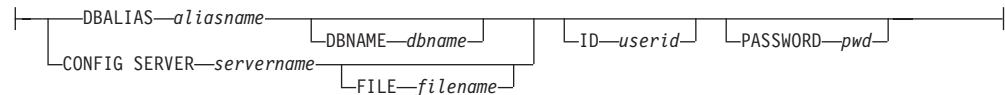
## LIST PUBS command

You can use the **LIST PUBS** command to list publications for a specified Q Capture server or schema.

### Syntax



## dbparms-clause:



## Parameters

### FOR SCHEMA *schema*

Specifies which schema to use. The default is "ASN".

## dbparms-clause:

### SERVER

Specifies the server containing the publications to list.

### DB *dbalias*

Specifies the database alias name of a z/OS subsystem or Linux, UNIX, or Windows database as cataloged on the DB2 from which the ASNCLP is invoked. This keyword is deprecated.

### DBALIAS *aliasname*

Specifies the database alias name of a z/OS subsystem or Linux, UNIX, or Windows database as cataloged on the DB2 from which the ASNCLP is invoked.

### DBNAME *zosdbname*

**z/OS** Specifies the z/OS database name. This is a logical z/OS database name, as created on a z/OS subsystem.

### ID *userid*

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

### PASSWORD *pwd*

Specifies the password to use for connections.

### CONFIG SERVER *servername*

**Classic sources:** Specifies which server configuration settings from the Classic replication configuration file that the ASNCLP should use to connect to the Classic server.

### FILE *filename*

Specifies the complete path and file name to the replication configuration file. If you do not use the **FILE** parameter, the ASNCLP attempts to use the `asnservers.ini` file in the current directory, if that file exists. Use the **FILE** parameter with different files that are customized for different environments.

---

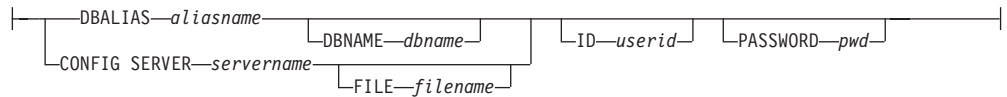
## LIST PUBQMAPS command

You can use the **LIST PUBQMAPS** command to list publication queue maps for a specified Q Capture server or schema.

## Syntax



## dbparms-clause:



## Parameters

### FOR SCHEMA *schema*

Specifies which schema to use. The default is "ASN".

## dbparms-clause:

### SERVER

Specifies the server containing the publishing queue maps to list.

### DB *dbalias*

Specifies the database alias name of a z/OS subsystem or Linux, UNIX, or Windows database as cataloged on the DB2 from which the ASNCLP is invoked. This keyword is deprecated.

### DBALIAS *aliasname*

Specifies the database alias name of a z/OS subsystem or Linux, UNIX, or Windows database as cataloged on the DB2 from which the ASNCLP is invoked.

### DBNAME *zosdbname*

**z/OS** Specifies the z/OS database name. This is a logical z/OS database name, as created on a z/OS subsystem.

### ID *userid*

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

### PASSWORD *pwd*

Specifies the password to use for connections.

### CONFIG SERVER *servername*

**Classic sources:** Specifies which server configuration settings from the Classic replication configuration file that the ASNCLP should use to connect to the Classic server.

### FILE *filename*

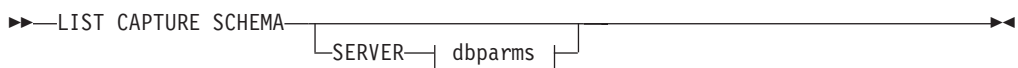
Specifies the complete path and file name to the replication configuration file. If you do not use the **FILE** parameter, the ASNCLP attempts to use the `asnservers.ini` file in the current directory, if that file exists. Use the **FILE** parameter with different files that are customized for different environments.

---

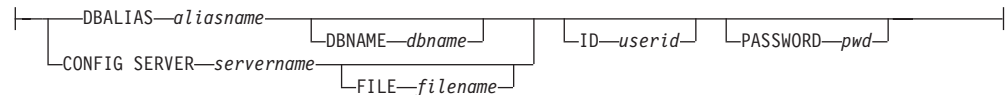
## LIST CAPTURE SCHEMA command

You can use the **LIST CAPTURE SCHEMA** command to list the Q Capture schemas for a specified server.

## Syntax



## dbparms-clause:



## Parameters

dbparms-clause:

### SERVER

Specifies the server that contains the schemas to be listed.

### DBALIAS *aliasname*

Specifies the database alias name of a z/OS subsystem or Linux, UNIX, or Windows database as cataloged on the DB2 from which the ASNCLP is invoked.

### DBNAME *zosdbname*

**z/OS** Specifies the z/OS database name. This is a logical z/OS database name, as created on a z/OS subsystem.

### ID *userid*

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

### PASSWORD *pwd*

Specifies the password to use for connections.

### CONFIG SERVER *servername*

**Classic sources:** Specifies which server configuration settings from the Classic replication configuration file that the ASNCLP should use to connect to the Classic server.

### FILE *filename*

Specifies the complete path and file name to the replication configuration file. If you do not use the **FILE** parameter, the ASNCLP attempts to use the `asnservers.ini` file in the current directory, if that file exists. Use the **FILE** parameter with different files that are customized for different environments.

## Example

To list the Q Capture schema on server SAMPLE:

```
LIST CAPTURE SCHEMA SERVER DBALIAS SAMPLE ID id1 PASSWORD "passwd!";
```

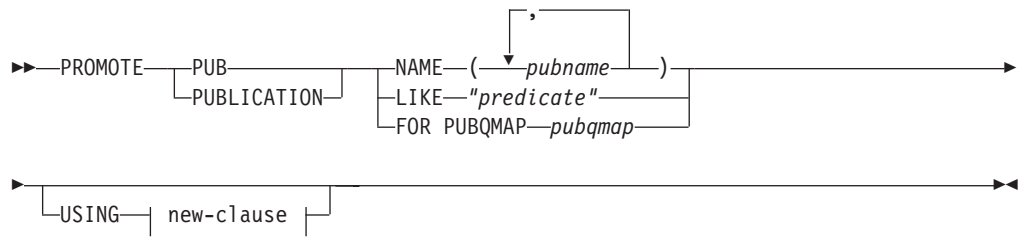
---

## PROMOTE PUB command

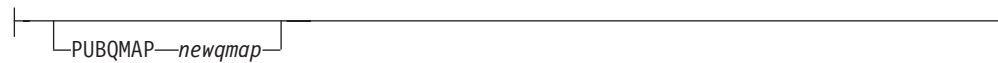
Use the **PROMOTE PUB** command to promote the definitions of one or more publications. You can use this command to customize the properties of the publication such as the name of the publication and the publishing queue map that it uses. The values for other properties are set to the same values as the current publication.

You can use the **ALTER PUB** command to change other properties after you promote the publication.

## Syntax



### new-clause::



## Parameters

### NAME *pubname*

Specifies one or more publication names to promote. Separate multiple publication names with a comma.

### LIKE *"predicate"*

Specifies part of a publication name to promote. All publications matching this predicate are promoted.

### FOR PUBQMAP *pubqmap*

Specifies an existing publishing queue map. All publications that use the publishing queue map are promoted.

new-clause:

### USING PUBQMAP *newqmap*

Specifies the name of a new publishing queue map that you want to use for the promoted publications.

## Example - matching a predicate

To promote all publications that start with the name EMP:

```
PROMOTE PUBLICATION LIKE "EMP%";
```

## Example - using a publishing queue map

To promote all publications that use the qmap1 publishing queue map :

```
PROMOTE PUBLICATION FOR PUBQMAP qmap1;
```

## Example - changing to a new publishing queue map

To promote all publications that use the publishing queue map qmap1 so that they use the queue map qmap2 instead:

```
PROMOTE PUBLICATION FOR PUBQMAP qmap1 USING PUBQMAP "qmap2";
```

## Example - naming publications

To promote publications that are named EMPLOYEE021 and EMPLOYEE032:  
PROMOTE PUB NAME (EMPLOYEE021,EMPLOYEE032);

---

## PROMOTE PUBQMAP command

Use the **PROMOTE PUBQMAP** command to promote the definitions of one or more publishing queue maps from one set of control tables to another set of control tables. You can also use this command to change some properties when the publishing queue map is promoted, such as the name of the send queue and name of the publishing queue map. The promoted values of properties that cannot be customized are taken from the source publishing queue map. If you need to change other properties, you can use the **ALTER PUBQMAP** command after promoting the publishing queue map to change the properties for the new publishing queue map.

### Syntax

```
➔➔ PROMOTE PUBQMAP NAME pubqmapname  
└──┬── USING new-clause  
└──┬── LIKE "predicate"
```

#### new-clause:

```
└──┬── PUBQMAP new-qmap  
└──┬── SENDQ new-sendq
```

### Parameters

**NAME** *pubqmapname*

Specifies the name of an existing publishing queue map to be promoted.

**USING**

Specifies new values for properties for the promoted publishing queue map.

**LIKE** *"predicate"*

Promotes all publishing queue maps that match the predicate *name*. You cannot customize the properties if you use this option.

new-clause:

**PUBQMAP** *new-qmap*

Specifies the name of the publishing queue map. If you do not specify a name, the current publishing queue map name is used.

**SENDQ** *new-sendq*

Specifies the send queue of the promoted publishing queue map. If you do not specify a send queue name, the current send queue name is used.

### Usage notes

- You must use the **SET SERVER** command with the **PROMOTE** option to set the environment for your promotions. The **SET SERVER** command allows you to specify the server that contains the publishing queue map to be promoted and to define which server the publishing queue map is promoted to.

- You cannot change the values for some properties by using the **PROMOTE PUBQMAP** command. You can later use the **ALTER PUBQMAP** command to change the value for other properties after you promote the publishing queue map.

### Example 1

To promote all publishing queue maps that match the name "SAMPLE\_ASN%":

```
PROMOTE PUBQMAP LIKE "SAMPLE_ASN%";
```

### Example 2

To promote publishing queue map PUBQMAP2, and change the name of the publishing queue map to pubqmapnew and change the name of the send queue to sendqnew2:

```
PROMOTE PUBQMAP NAME PUBQMAP2 USING PUBQMAP pubqmapnew SENDQ "sendqnew2";
```

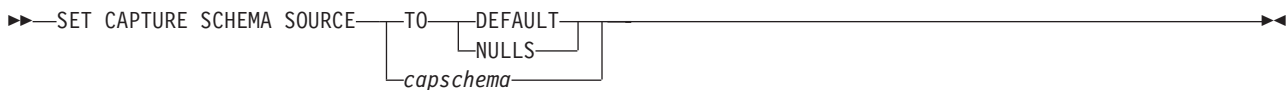
---

## SET CAPTURE SCHEMA command

Use the **SET CAPTURE SCHEMA** command to set a default schema of the source control tables for all task commands. For Classic sources, you can use only the default Q Capture schema, ASN.

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

### Syntax



### Parameters

#### SOURCE

Specifies the Q Capture schema. If you are using a DB2 source, the schema can be any valid DB2 schema name. If you are using a Classic source, you must use the DEFAULT schema.

#### 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, warning, and error messages.

### Syntax

```
▶▶—SET LOG—"logfile"——┬── WITH DETAILS ─┬──▶▶
```

### Parameters

*"logfile"*

Specifies the output log file name. The default log file name is `qrep1msg.log`.

#### WITH DETAILS

Creates an additional log file with just error messages for the run along with the "Explanation" and "User response" sections for each message. The name of the additional file is *logfile\_1*. The contents of the standard log file remain unchanged.

### 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 1

To name the output log file `qmaplog.err` for creating replication queue maps:

```
SET LOG "qmaplog.err";
```

## Example 2

To specify that the ASNCLP program create its regular log file and an additional log file with error messages and the "Explanation" and "User response" sections for each message:

```
SET LOG "qreplog.err" WITH DETAILS;
```

---

## 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, or the ASNCLP commands needed to promote a replication environment. You cannot use this command with non-relational sources.

### Syntax

```
▶▶—SET OUTPUT——┬── CAPTURE SCRIPT—"capfname" ─┬── TARGET SCRIPT—"trgfname" ─┬──▶▶
```

PROMOTE SCRIPT—"profname"

## 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.

### PROMOTE SCRIPT "*profname*"

Specifies the output file name for the ASNCLP commands generated by **PROMOTE** statements. If the file name is not specified, the default file created is named `qrepl_asnclp.in`.

## 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 QMANAGER command

Use the **SET QMANAGER** command to set the WebSphere MQ queue manager that is used by the Q Capture program, Q Apply program, or both. You cannot use this command with non-relational sources.

## Syntax

```
▶▶ SET QMANAGER—"qmgrname"—FOR—

|                     |
|---------------------|
| CAPTURE SCHEMA      |
| APPLY SCHEMA        |
| NODE— <i>number</i> |


```

## Parameters

### "*qmgrname*"

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.

### NODE

Specifies one server in a multidirectional configuration. If this keyword is specified, the ASNCLP program uses the same value for "*qmgrname*" for both the Q Capture server and Q Apply server.

## 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 QM1 for both the Q Capture and Q Apply programs on a server that is used in bidirectional or peer-to-peer replication:

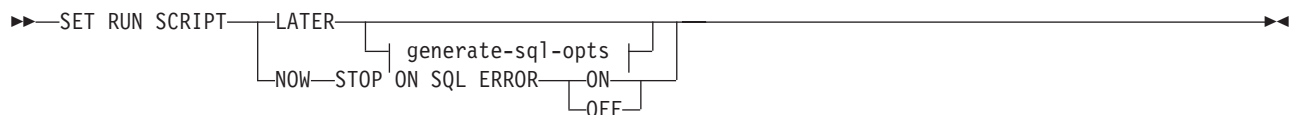
```
SET QMANAGER FOR NODE 1 "QM1";
```

---

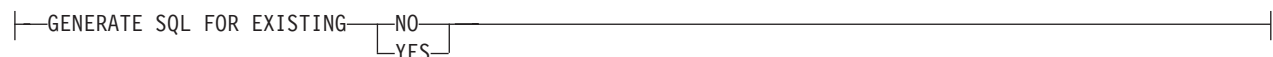
## SET RUN SCRIPT command

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. You cannot use the LATER parameter with non-relational sources.

### Syntax



### generate-sql-opts:



## Parameters

### LATER

Specify to run the SQL scripts at a later time. You cannot use this parameter with Classic sources. Use this option if you want to verify your script before you run it. You can also use this option if you want to create SQL script files on one operating system, but run them on another.

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.

### NOW

Specify to automatically execute the SQL scripts.

### STOP ON SQL ERROR

Specifies whether the ASNCLP program continues to process commands in the ASNCLP script file and statements in the generated SQL script file after one of the following errors:

- **ASNCLP script file:** An error while checking to predict whether the SQL statement to be generated will cause an SQL error. For example, a publication cannot be defined in the control tables unless the control tables exist first.
- **Generated SQL script file:** An SQL error while running the SQL statements.

**ON (default)**

Specify if you want the ASNCLP to stop processing commands in the ASNCLP script, and stop processing SQL statements in the generated SQL script, when the first validity check fails or SQL statement fails. If the error occurs while the ASNCLP is running the SQL script, previous SQL statements that are related to the task command with an error are rolled back.

**OFF**

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

For a more complete explanation of how the ASNCLP responds to errors depending on this and other SET RUN SCRIPT options, see How the ASNCLP handles errors while processing scripts.

**GENERATE SQL FOR EXISTING**

Specifies whether to generate SQL when ASNCLP encounters errors due to duplicate, or existing, objects when processing **CREATE** commands. This option has no effect on **DROP** commands.

**NO** The ASNCLP program will not generate SQL to create objects that already exist. This is the default.

**YES**

The ASNCLP program continues to generate SQL statements even if it encounters existing object errors. The following errors are ignored when you specify this option:

**CREATE CONTROL TABLES**

Another set of control tables already exist under the same schema or table spaces are specified to be created but they already exist.

**CREATE PUB**

Another publication with the same name already exists.

**CREATE PUBQMAP**

Another publishing queue map with the same name already exists.

**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 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 might 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 SCRIPT NOW be used.

### Example - Run immediately and stop on errors

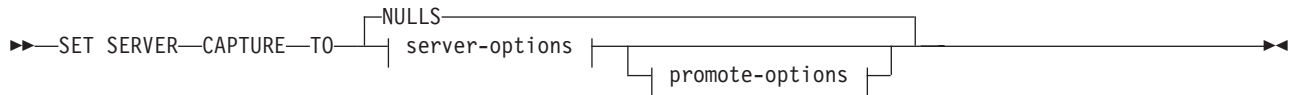
To automatically run the SQL scripts but stop processing the ASNCLP commands if an error occurs:

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

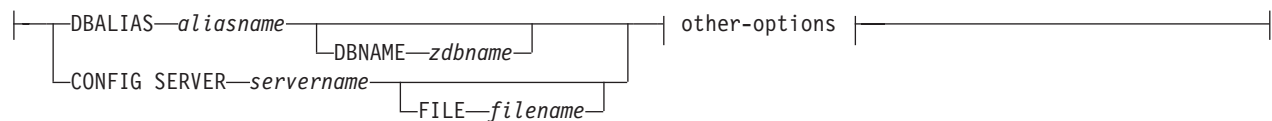
## SET SERVER command (Event Publishing)

Use the **SET SERVER** command to specify the Q Capture server to use in the ASNCLP session. After you set a server name, all subsequent commands in the session apply to this server until you change the server with this command.

### Syntax



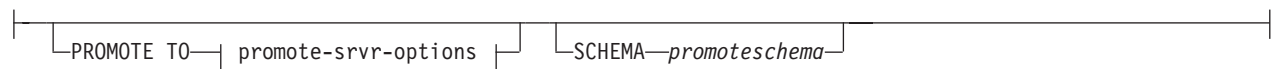
#### server-options:



#### other-options:



#### promote-options:



#### promote-srvr-options:



## Parameters

### CAPTURE

Specify to set the database as a Q Capture or Classic server.

**NULLS**

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

server-options:

**DBALIAS** *aliasname*

Specifies the database alias name of a z/OS subsystem or Linux, UNIX, or Windows database as cataloged on the DB2 from which the ASNCLP is invoked.

**z/OS**

**DBNAME** *zdbname*

Specifies the database name.

**Note:** DBNAME is mandatory when ASNCLP is running on z/OS and the Q Capture server is on z/OS. DBNAME is a location name and is the name by which the DB2 database is known to local DB2 SQL applications. This name must match the name that was entered in the LOCATIONS column of the SYSIBM.LOCATIONS table in the CDB.

**CONFIG SERVER** *servername*

**Classic sources:** Specifies the Classic source that the ASNCLP program connects to. The server name must match the bracketed [NAME] field that is entered in the ASNCLP configuration file.

**FILE** *filename*

Specifies the complete path and file name to the ASNCLP configuration file. If you do not use the **FILE** parameter, the ASNCLP program attempts to use the `asnservers.ini` file in the current directory, if that file exists.

other-options:

**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. If you specify the user ID and do not specify the password, you will be prompted to enter the password. The password is hidden as you type.

**Note:** This keyword is not valid when the ASNCLP runs natively on z/OS because user authentication is handled through the communication database (CDB).

promote-options:

**PROMOTE TO**

Promote the specified server definitions.

**SCHEMA** *promoteschema*

Specifies the schema under which the server definitions will be promoted. If a schema is not specified, then the schema under which the current server definitions exist is used.

promote-srvr-options:

**DBALIAS** *dbalias*

Specifies the database that will receive the promoted server definitions. If this

clause is not specified and a **PROMOTE** command is included in the input file, then the **PROMOTE** command promotes the definitions to the current server.

**z/OS** **DBNAME** *zdbname*

Specifies the name of the database subsystem that will receive the promoted definitions.

**CONFIG SERVER** *servername*

Specifies the replication target that the ASNCLP program connects to when promoting definitions. The server name must match the bracketed [NAME] field that is entered in the ASNCLP configuration file.

**FILE** *filename*

Specifies the complete path and file name to the ASNCLP configuration file. If you do not use the **FILE** parameter, the ASNCLP program attempts to use the `asnservers.ini` file in the current directory, if that file exists.

**ID** *id*

Specifies the database ID where definitions will be promoted to. If not specified, the ASNCLP output script is generated without ID information.

**PASSWORD** *password*

Specifies the password to use to connect to the database. If not specified, the ASNCLP output script is generated without password information.

## Example

To set the Q Capture server to the database SAMPLE:

```
SET SERVER CAPTURE TO DBALIAS SAMPLE;
```

## Example - Classic sources

Given a configuration file called `classic.ini` that contains the following information:

```
[classic1]
Type=CLASSIC
Data source=CACSAMP
Host=9.30.155.156
Port=8019
```

Use the following command to specify server `classic1` as the data server:

```
SET SERVER CAPTURE TO CONFIG SERVER classic1 FILE classic.ini ID id1 PASSWORD pwd1;
```

## Example - password prompting

To set the Capture control server and specify only the user ID in the command:

```
SET SERVER CAPTURE TO DBALIAS SAMPLE ID DB2ADMIN;
```

You are prompted to enter the password. If you are running the commands from an input file in batch mode, the program waits for you to enter the password before the program processes the next commands. Your text is hidden when you type.

## Example - promoting configurations

To set the existing server containing definitions to be promoted and set the new server that will receive these promoted configurations:

```
SET SERVER CAPTURE TO DBALIAS SAMPLE ID id1 PASSWORD "p1wd"  
PROMOTE TO DBALIAS SAMPLE1 ID id1 PASSWORD SCHEMA ASN;
```

---

## 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 PUB command

Use the **START PUB** command to start a publication.

### Syntax

```
▶▶ START PUB {PUBNAME pubname | FOR PUBNAME LIKE "predicate" } ▶▶
```



## Parameters

**PUBNAME** *pubname*

Specifies the name of the publication to start.

**FOR PUBNAME LIKE** "*predicate*"

Specify to start publications that match the expression in the LIKE clause. The following example shows a LIKE clause:

```
START PUB FOR PUBNAME LIKE "%table%"
```

## Example

To start a publication:

```
START PUB PUBNAME MYPUB
```

---

## STOP PUB command

Use the **STOP PUB** command to stop a publication.

### Syntax

```
▶▶ STOP PUB [PUBNAME pubname | FOR PUBNAME LIKE "predicate"] ▶▶
```

## Parameters

**PUBNAME** *pubname*

Specifies the name of the publication to stop.

**FOR PUBNAME LIKE** "*predicate*"

Specify to stop publications that match the expression in the LIKE clause. The following example shows a LIKE clause:

```
STOP PUB FOR PUBNAME LIKE "%table%"
```

## Example

To stop a publication:

```
STOP PUB PUBNAME MYPUB
```

---

## 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
```

---

## Chapter 7. 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 script for setting up the Replication Alert Monitor” on page 292 demonstrates how you can combine Replication Alert Monitor commands to create an ASNCLP setup script.

Table 8 lists the ASNCLP commands for the Replication Alert Monitor and links to topics that describe each command.

*Table 8. ASNCLP commands for the Replication Alert Monitor*

<b>If you want to ...</b>	<b>Use this command</b>
Change alert conditions for the Apply program	“ALTER ALERT CONDITIONS FOR APPLY command” on page 293
Change alert conditions for the Capture program	“ALTER ALERT CONDITIONS FOR CAPTURE command” on page 296
Change alert conditions for the Q Apply program	“ALTER ALERT CONDITIONS FOR QAPPLY command” on page 298
Change alert conditions for the Q Capture program	“ALTER ALERT CONDITIONS FOR QCAPTURE command” on page 300
Change contact information for notifications	“ALTER CONTACT command” on page 302
Change a contact group	“ALTER GROUP command” on page 303
Change a monitor suspension	“ALTER MONITOR SUSPENSION command” on page 304
Change a monitor suspension template	“ALTER MONITOR SUSPENSION TEMPLATE command” on page 305
Create alert conditions for the Apply program	“CREATE ALERT CONDITIONS FOR APPLY command” on page 306
Create alert conditions for the Capture program	“CREATE ALERT CONDITIONS FOR CAPTURE command” on page 308
Create alert conditions for the Q Apply program	“CREATE ALERT CONDITIONS FOR QAPPLY command” on page 310
Create alert conditions for the Q Capture program	“CREATE ALERT CONDITIONS FOR QCAPTURE command” on page 312
Create contact information for notifications	“CREATE CONTACT command” on page 313
Create the control tables for the Monitor program	“CREATE CONTROL TABLES FOR command” on page 314
Create a contact group	“CREATE GROUP command” on page 316
Create a monitor suspension	“CREATE MONITOR SUSPENSION command” on page 317
Create a monitor suspension template	“CREATE MONITOR SUSPENSION TEMPLATE command” on page 318
Delegate an existing contact to a new contact	“DELEGATE CONTACT command” on page 319

Table 8. 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 320
Delete alert conditions for the Capture program	"DROP ALERT CONDITIONS FOR CAPTURE command" on page 320
Delete alert conditions for the Q Apply program	"DROP ALERT CONDITIONS FOR QAPPLY command" on page 321
Delete alert conditions for the Q Capture program	"DROP ALERT CONDITIONS FOR QCAPTURE command" on page 321
Delete an existing contact	"DROP CONTACT command" on page 321
Delete a contact group	"DROP GROUP command" on page 322
Delete a monitor suspension	"DROP MONITOR SUSPENSION command" on page 322
Delete a monitor suspension template	"DROP MONITOR SUSPENSION TEMPLATE command" on page 323
List monitor suspensions	"LIST MONITOR SUSPENSION command" on page 323
List monitor suspension templates	"LIST MONITOR SUSPENSION TEMPLATE command" on page 323
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 324
Substitute one existing contact with another existing contact	"SUBSTITUTE CONTACT command" on page 326

## Sample ASNCLP script for setting up the Replication Alert Monitor

This sample contains an ASNCLP script for setting up the Replication Alert Monitor. It includes Monitor control tables, a contact, and alert conditions.

You can copy the sample script to a text file and run it by using the ASNCLP -f *filename* command. First, change db2admin and "passw0rd" to the user ID and password for connecting to the SAMPLE database. Within the script, details about each group of commands are preceded by a comment character (#).

### ASNCLP script

This script includes commands for the following tasks:

- 1** Setting the environment
- 2** Creating Monitor control tables
- 3** Specifying a contact
- 4** Defining alert conditions
- 5** Ending the ASNCLP session

```
# 1 Setting the environment
# Three SET SERVER commands are required in this script: You set the Monitor
# server to create the Monitor control tables and 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.
```

```
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 SAMPLE ID db2admin PASSWORD "passw0rd";
```

```

SET RUN SCRIPT NOW STOP ON SQL ERROR ON;

# 2 Creating Monitor control tables

CREATE CONTROL TABLES FOR MONITOR CONTROL SERVER;

# 3 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";

# 4 Creating alert conditions
# These commands create alert conditions for both the Q Capture program
# and the Q Apply program that run at the monitored server SAMPLE. The
# Q Capture conditions trigger an alert if Q Capture stops 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 stops, if any errors or warnings occur, or if the average end-to-end
# latency exceeds 2000 milliseconds(2 seconds). The EXCEPTIONS condition triggers
# an alert if a row is added to the IBMQREP_EXCEPTIONS table, signaling an SQL
# error or conflict. The ASNCLP SESSION SET command is needed because the alert
# conditions are for Q replication programs.

ASNCLP SESSION SET TO Q REPLICATION;
CREATE ALERT CONDITIONS FOR QCAPTURE MONITOR QUALIFIER MONQUAL
NOTIFY CONTACT repladmin (STATUS DOWN, ERRORS, WARNINGS, LATENCY 2);
CREATE ALERT CONDITIONS FOR QAPPLY MONITOR QUALIFIER MONQUAL
NOTIFY CONTACT repladmin (STATUS DOWN, ERRORS, WARNINGS, ELATENCY 2000,
EXCEPTIONS);

# 5 Ending the ASNCLP session

QUIT;

```

---

## 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—|

```

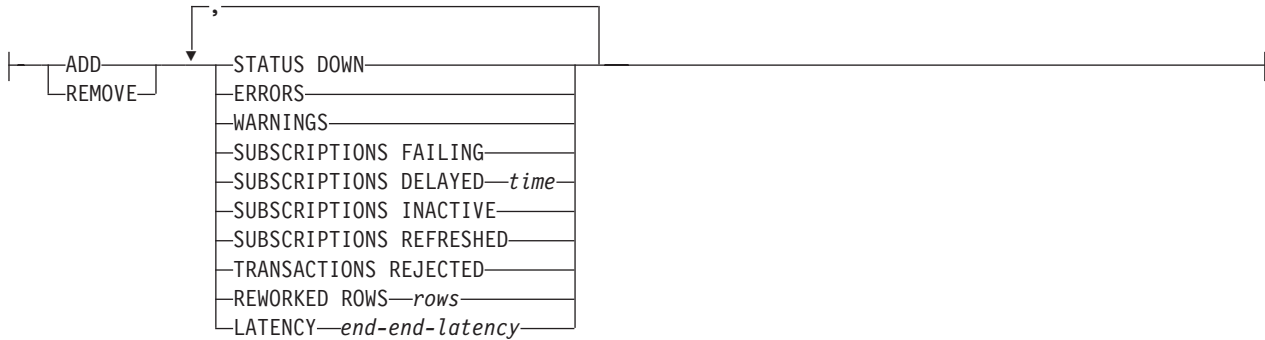
#### notify-clause:

```

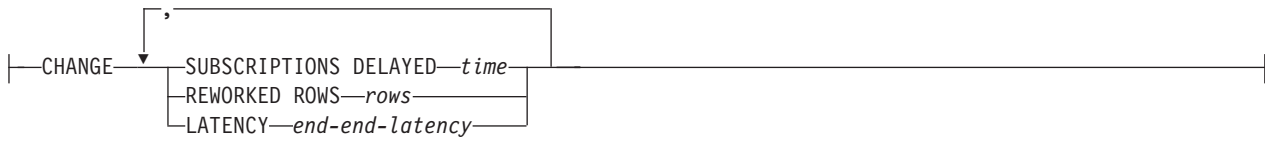
|—NOTIFY—|—————|
|—CONTACT—contact-name—|
|—GROUP—group-name—|
|—OPERATOR CONSOLE—|

```

#### add-or-remove-clause:



**change-clause:**



**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**  
**z/OS** 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:

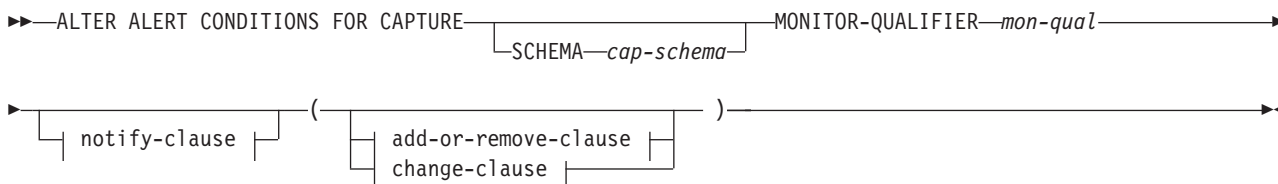
```
ALTER ALERT CONDITIONS FOR APPLY QUALIFIER MYAPPLY01 MONITOR QUALIFIER MONQUAL
NOTIFY REPLADMIN (REMOVE WARNINGS)
```

---

## ALTER ALERT CONDITIONS FOR CAPTURE command

Use the **ALTER ALERT CONDITIONS FOR CAPTURE** command to alter alert conditions for the Capture program.

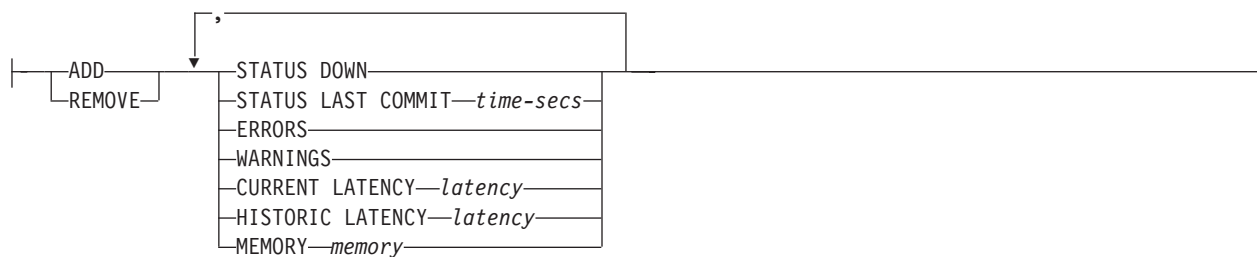
### Syntax



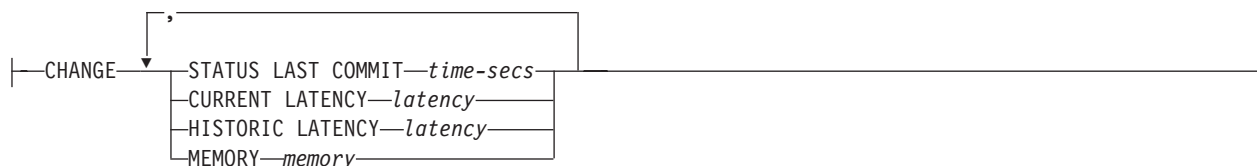
#### notify-clause:



#### add-or-remove-clause:



#### change-clause:





## 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**

**z/OS** 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 by 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 by 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**

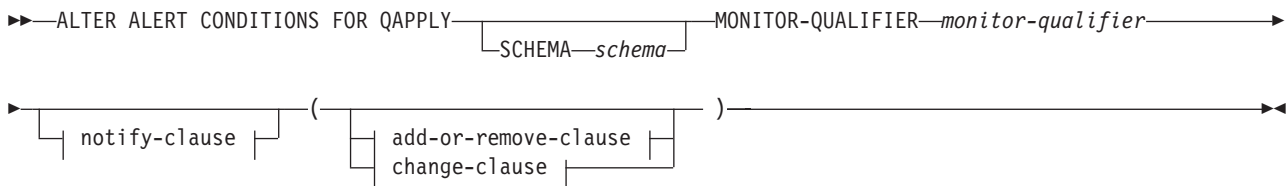
To alter an alert condition for the Capture program by removing the condition MEMORY and no longer alerting the contact REPLADMIN when the condition occurs:

```
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.

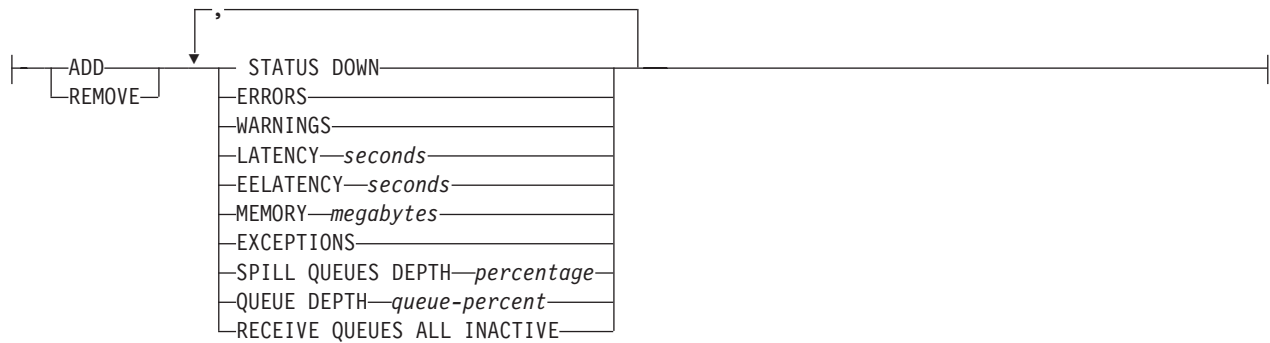
**Syntax**



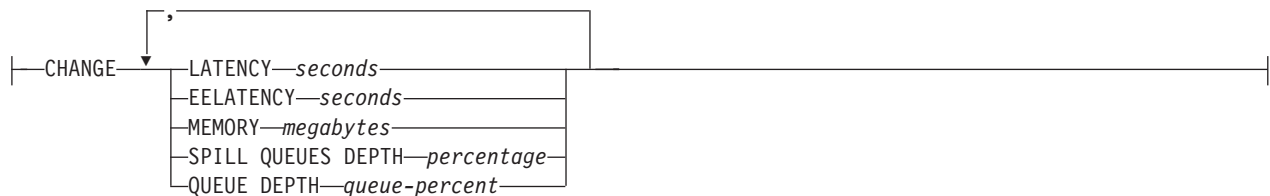
**notify-clause:**



**add-or-remove-clause:**



### change-clause:



## 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.

**RECEIVE QUEUES ALL INACTIVE**

Specifies that an alert will be sent when the value of the STATE column in the IBMQREP\_RECVQUEUES table changes to I (inactive) for any receive queue.

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**

**z/OS** 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)
```

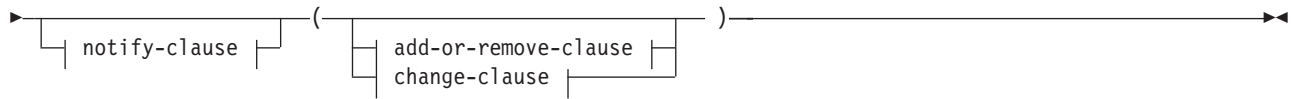
---

**ALTER ALERT CONDITIONS FOR QCAPTURE command**

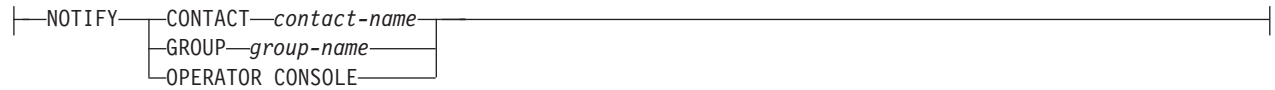
Use the **ALTER ALERT CONDITIONS FOR QCAPTURE** command to alter the alert conditions for the Q Capture program.

**Syntax**

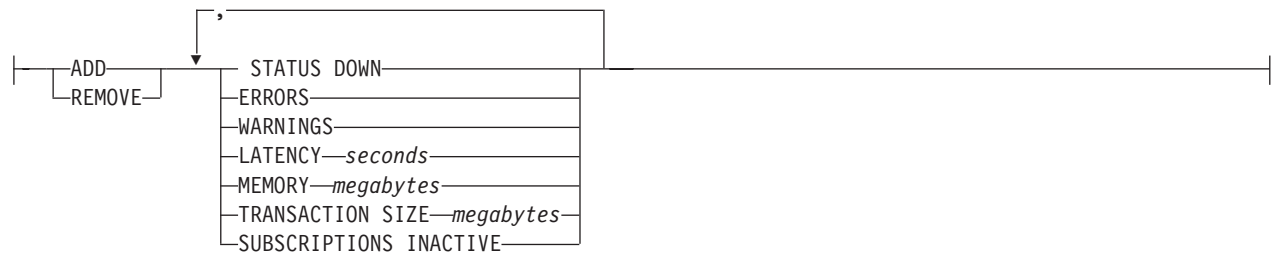
```
▶▶—ALTER ALERT CONDITIONS FOR QCAPTURE—[SCHEMA—schema]—MONITOR-QUALIFIER—monitor-qualifier—▶▶
```



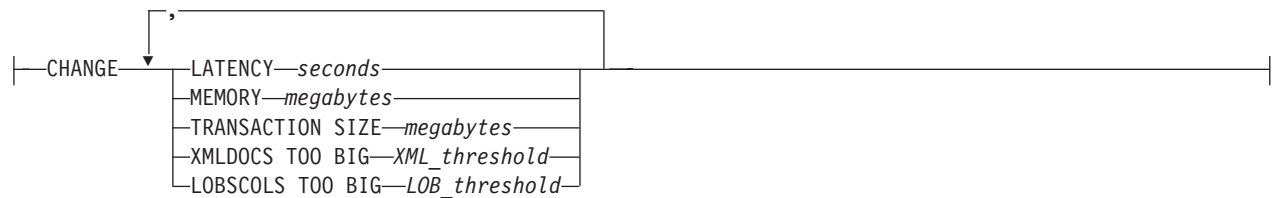
**notify-clause:**



**add-or-remove-clause:**



**change-clause:**



**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.

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

**z/OS** 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:

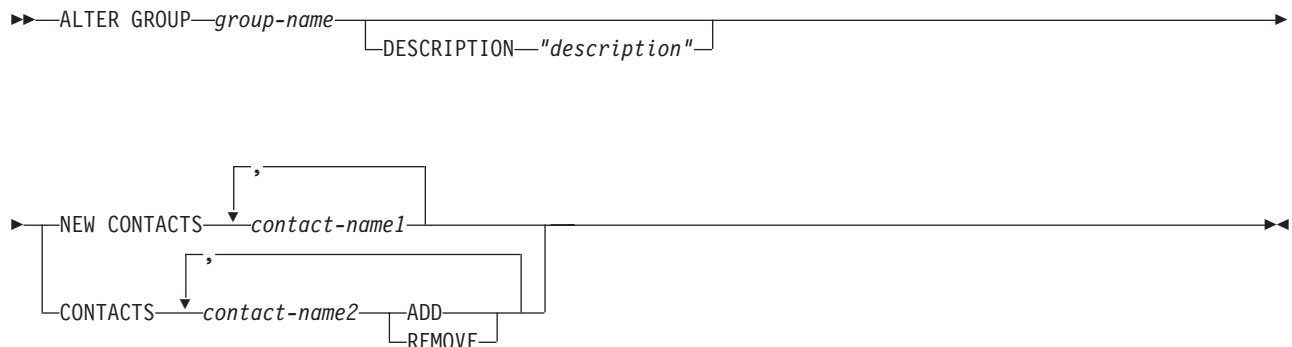
```
ALTER CONTACT REPLADMIN EMAIL "repladmin@ibm.com"
```

---

## ALTER GROUP command

Use the **ALTER GROUP** command to alter a group of replication monitor contacts.

### Syntax



## 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:

```
┌—DAILY—FOR DURATION—n—┐  
└─MINUTES—  
└─HOURS—  
┌—WEEKLY—DAY OF WEEK—┐  
└─SUNDAY—┐  
└─MONDAY—┐  
└─TUESDAY—┐  
└─WEDNESDAY—┐  
└─THURSDAY—┐  
└─FRIDAY—┐  
└─SATURDAY—┐  
└─FOR DURATION—n—┐  
└─MINUTES—  
└─HOURS—  
└─DAYS—
```

### 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:

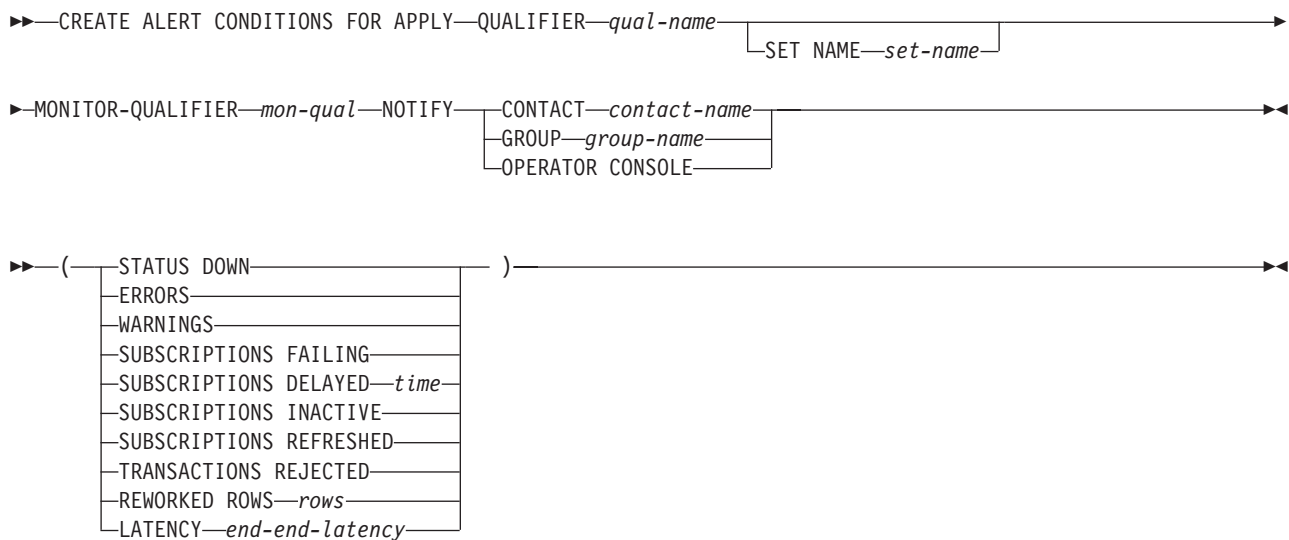
```
ALTER MONITOR SUSPENSION TEMPLATE lunch START TIME 12:00:00 REPEATS DAILY
FOR DURATION 90 MINUTES
```

---

## 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



### 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**

**z/OS** 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**

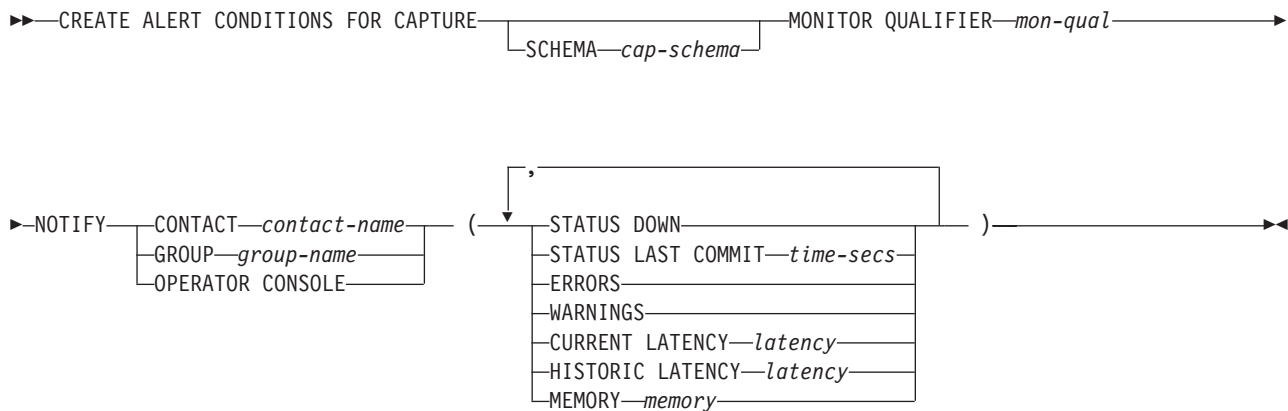
To create alert conditions for the Apply program that sends an alert to the contact REPLADMIN when a condition occurs:

```
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**



**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**

**z/OS** 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 by 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 by 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:

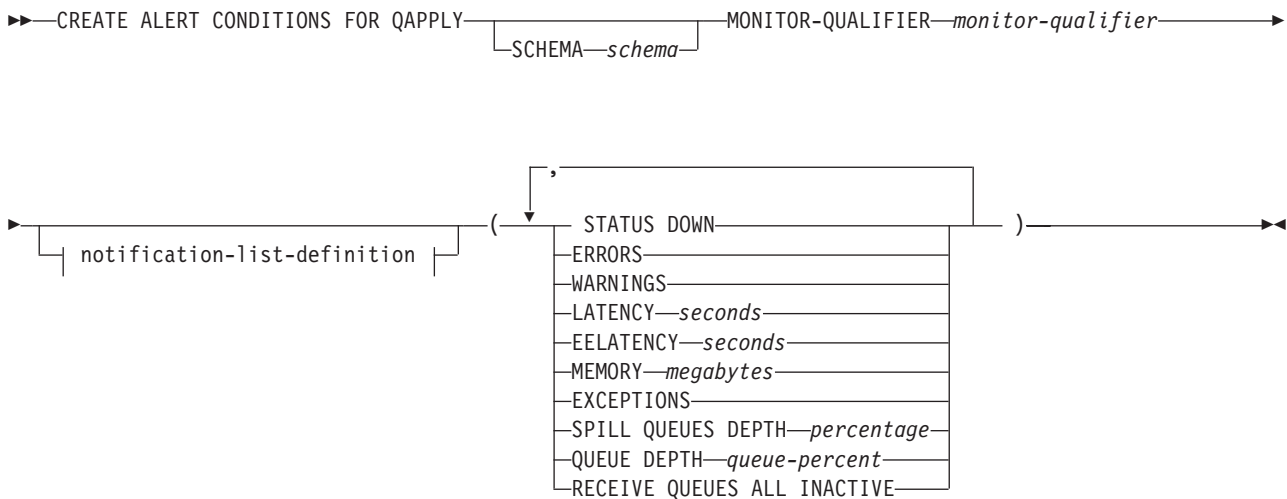
```
CREATE ALERT CONDITIONS FOR CAPTURE QUALIFIER MYAPPLY01 MONITOR QUALIFIER MONQUAL
NOTIFY CONTACT REPLADMIN (STATUS DOWN, ERRORS, WARNINGS, SUBSCRIPTION FAILING,
SUBSCRIPTION DELAYED 300, SUBSCRIPTIONS INACTIVE, SUBSCRIPTIONS REFRESHED,
TRANSACTION REJECTED, REWORKED ROWS 2, LATENCY 360)
```

---

## 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



### notification-list-definition:



### 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 `asnqcmd 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** *milliseconds*

Specifies that an alert will be sent when the average number of milliseconds that it takes for a transaction to be applied to a target table after the Q Apply program gets the transaction from a receive queue exceeds the number of milliseconds that was 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 that was 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 that was 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.

**RECEIVE QUEUES ALL INACTIVE**

Specifies that an alert will be sent when the value of the STATE column in the IBMQREP\_RECVQUEUES table changes to I (inactive) for any receive queue.

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:

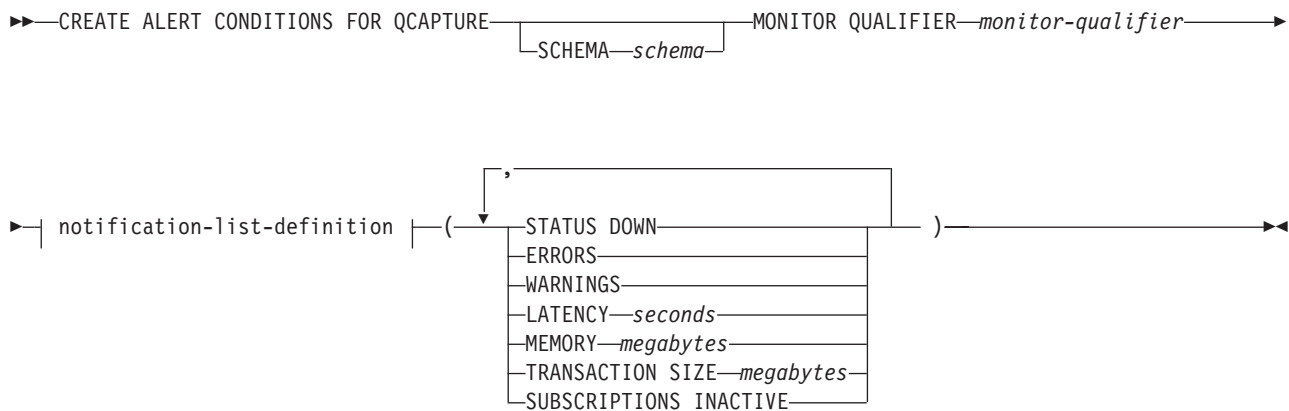
```
CREATE CONDITIONS FOR QAPPLY MONITOR QUALIFIER MONQUAL
  NOTIFY CONTACT REPLADMIN (STATUS DOWN, ERRORS, WARNINGS,
  LATENCY 360, EXCEPTIONS)
```

---

## 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



### notification-list-definition:



### 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 `asnqcmd 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.

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**

**z/OS** 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:

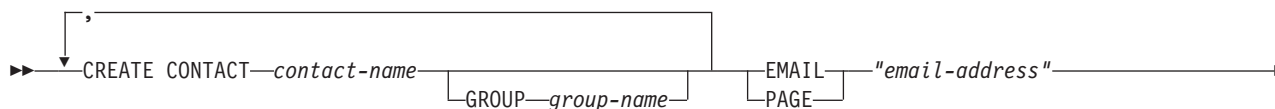
```
CREATE ALERT CONDITIONS FOR QCAPTURE SCHEMA ASN1 MONITOR QUALIFIER MONQUAL
NOTIFY CONTACT REPLADMIN (STATUS DOWN, ERRORS, WARNINGS, LATENCY 30, MEMORY 60)
```

---

## 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



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

Use the **CREATE CONTROL TABLES FOR** command to create a new set of Replication Alert Monitor control tables.

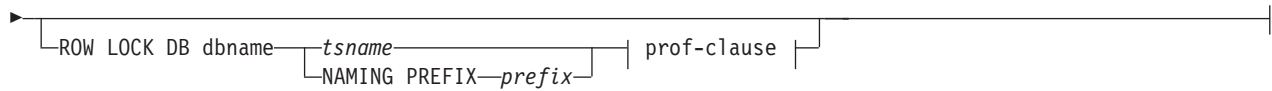
### Syntax

```
CREATE CONTROL TABLES FOR MONITOR CONTROL SERVER  
IN (ZOS | ZOS | zos-ts-clause |  
    UW | UW | uw-ts-clause |  
    NONIBM | NONIBM | fed-ts-clause |
```

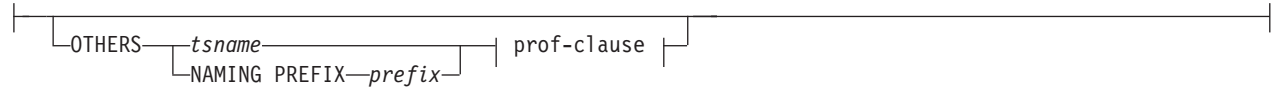
#### zos-ts-clause:

```
ALERTS DB dbname | tname | prof-clause |  
NAMING PREFIX prefix
```

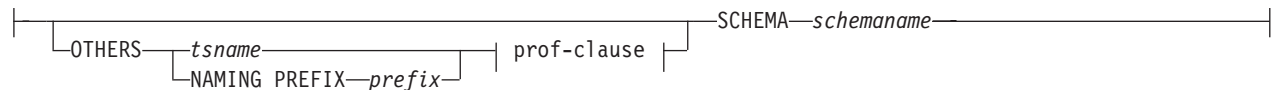
```
PAGE LOCK DB dbname | tname | prof-clause |  
NAMING PREFIX prefix
```



**uw-ts-clause:**



**fed-ts-clause:**



**prof-clause:**



**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**

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

**UW**

**Linux UNIX** Specifies UNIX or Windows.

**NONIBM**

Specifies non-DB2 data sources.

**ALERTS**

**z/OS** 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 and use 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:

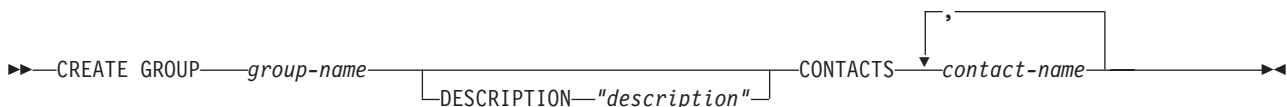
```
CREATE CONTROL TABLES FOR MONITOR CONTROL SERVER
```

---

## 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 | ALIAS server_alias } STARTING DATE date
{ USING TEMPLATE template_name | STARTING TIME starting_time } ENDING DATE date { ENDING TIME ending_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 monitored 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 monitored 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
```

---

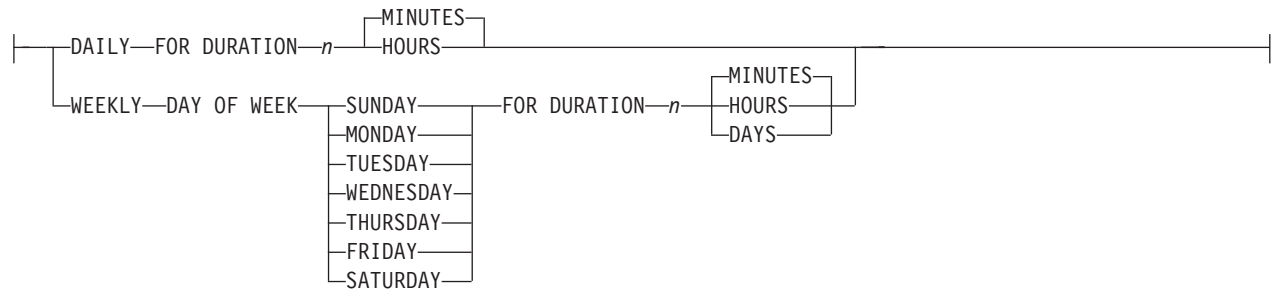
**CREATE MONITOR SUSPENSION TEMPLATE command**

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:



## 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 (PERFORMANCE) for a given period of time:

```
DELEGATE CONTACT REPLADMIN TO PERFORMANCE 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—MONITOR 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—MONITOR 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

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"
```

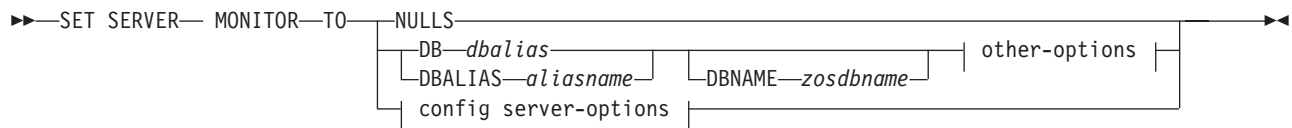
---

## SET SERVER command

Use the **SET SERVER** command to specify the database that is used as a monitor control server 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.

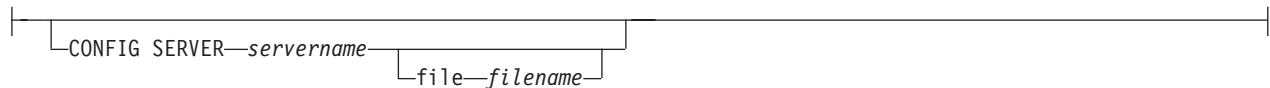
## Syntax



### other-options:



### config server-options:



## Parameters

### MONITOR

Specify to set the database as a monitor control 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 of a z/OS subsystem or Linux, UNIX, Windows, or System i database as cataloged on the DB2 from which the ASNCLP is invoked. This keyword is deprecated.

### DBALIAS *aliasname*

Specifies the database alias name of a z/OS subsystem or Linux, UNIX, Windows, or System i database as cataloged on the DB2 from which the ASNCLP is invoked.

### DBNAME *zosdbname*

**z/OS** Specifies the z/OS database name.

**Note:** DBNAME is mandatory when ASNCLP is running on z/OS and the Monitor control server is on z/OS. DBNAME is the name by which the DB2 database is known to local DB2 SQL applications. This name must match the name that was entered in the LOCATIONS column of the SYSIBM.LOCATIONS table in the CDB.

other-options clause:

### 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. If you specify the user ID and do not specify the password, you will be prompted to enter the password. The password is hidden as you type.

config server-options clause:

**CONFIG SERVER** *servername*

**UNIX System Services (USS) on z/OS:** Specifies the DB2 database to use as a monitor control server when the ASNCLP program is running on USS. The server name must match the bracketed [NAME] field that is entered in the ASNCLP configuration file.

**FILE** *filename*

Specifies the complete path and file name to the ASNCLP configuration file. If you do not use the FILE parameter, the ASNCLP program attempts to use the anservers.ini file in the current directory, if that file exists.

## 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 (PERFORMANCE):

```
SUBSTITUTE CONTACT REPLADMIN WITH PERFORMANCE
```

---

## Contacting IBM

You can contact IBM for customer support, software services, product information, and general information. You also can provide feedback to IBM about products and documentation.

The following table lists resources for customer support, software services, training, and product and solutions information.

Table 9. IBM resources

Resource	Description and location
IBM Support Portal	You can customize support information by choosing the products and the topics that interest you at <a href="http://www.ibm.com/support/entry/portal/Software/Information_Management/InfoSphere_Information_Server">www.ibm.com/support/entry/portal/Software/Information_Management/InfoSphere_Information_Server</a>
Software services	You can find information about software, IT, and business consulting services, on the solutions site at <a href="http://www.ibm.com/businesssolutions/">www.ibm.com/businesssolutions/</a>
My IBM	You can manage links to IBM Web sites and information that meet your specific technical support needs by creating an account on the My IBM site at <a href="http://www.ibm.com/account/">www.ibm.com/account/</a>
Training and certification	You can learn about technical training and education services designed for individuals, companies, and public organizations to acquire, maintain, and optimize their IT skills at <a href="http://www.ibm.com/software/sw-training/">http://www.ibm.com/software/sw-training/</a>
IBM representatives	You can contact an IBM representative to learn about solutions at <a href="http://www.ibm.com/connect/ibm/us/en/">www.ibm.com/connect/ibm/us/en/</a>

### Federation, replication, and event publishing products support

For support, go to:

- IBM InfoSphere Federation Server  
[www.ibm.com/software/data/integration/support/federation\\_server/](http://www.ibm.com/software/data/integration/support/federation_server/)
- IBM InfoSphere Replication Server  
[www.ibm.com/software/data/integration/support/replication\\_server/](http://www.ibm.com/software/data/integration/support/replication_server/)
- IBM InfoSphere Data Event Publisher  
[www.ibm.com/software/data/integration/support/data\\_event\\_publisher/](http://www.ibm.com/software/data/integration/support/data_event_publisher/)

### Classic products support

For support, go to:

- IBM InfoSphere Classic Federation Server for z/OS  
[www.ibm.com/software/data/integration/support/classic\\_federation\\_server\\_z/](http://www.ibm.com/software/data/integration/support/classic_federation_server_z/)

- IBM InfoSphere Classic Replication Server for z/OS  
www.ibm.com/software/data/infosphere/support/replication-server-z/
- IBM InfoSphere Classic Data Event Publisher for z/OS  
www.ibm.com/software/data/integration/support/data\_event\_publisher\_z/
- IBM InfoSphere Data Integration Classic Connector for z/OS  
www.ibm.com/software/data/integration/support/data\_integration\_classic\_connector\_z/

## Providing feedback

The following table describes how to provide feedback to IBM about products and product documentation.

*Table 10. Providing feedback to IBM*

Type of feedback	Action
Product feedback	You can provide general product feedback through the Consumability Survey at <a href="http://www.ibm.com/software/data/info/consumability-survey">www.ibm.com/software/data/info/consumability-survey</a>
Documentation feedback	To comment on the information center, click the Feedback link on the top right side of any topic in the information center. You can also send comments about PDF file books, the information center, or any other documentation in the following ways: <ul style="list-style-type: none"> <li>• Online reader comment form: <a href="http://www.ibm.com/software/data/rcf/">www.ibm.com/software/data/rcf/</a></li> <li>• E-mail: <a href="mailto:comments@us.ibm.com">comments@us.ibm.com</a></li> </ul>



---

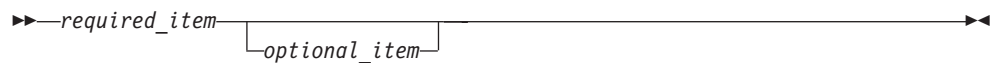
## How to read syntax diagrams

The following rules apply to the syntax diagrams that are used in this information:

- Read the syntax diagrams from left to right, from top to bottom, following the path of the line. The following conventions are used:
  - The >>--- symbol indicates the beginning of a syntax diagram.
  - The ---> symbol indicates that the syntax diagram is continued on the next line.
  - The >--- symbol indicates that a syntax diagram is continued from the previous line.
  - The --->< symbol indicates the end of a syntax diagram.
- Required items appear on the horizontal line (the main path).



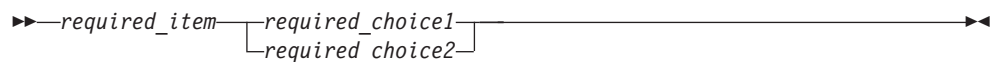
- Optional items appear below the main path.



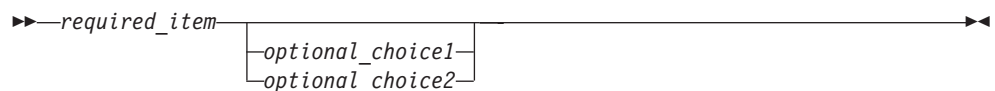
If an optional item appears above the main path, that item has no effect on the execution of the syntax element and is used only for readability.



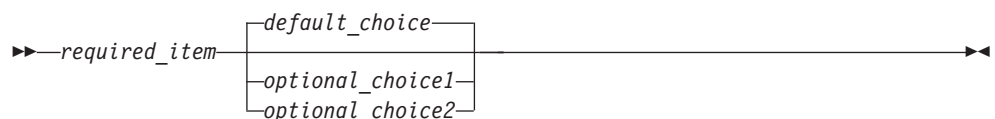
- If you can choose from two or more items, they appear vertically, in a stack. If you must choose one of the items, one item of the stack appears on the main path.



If choosing one of the items is optional, the entire stack appears below the main path.



If one of the items is the default, it appears above the main path, and the remaining choices are shown below.



- An arrow returning to the left, above the main line, indicates an item that can be repeated.

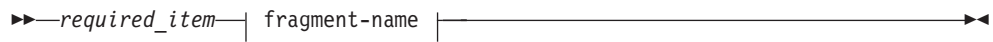


If the repeat arrow contains a comma, you must separate repeated items with a comma.

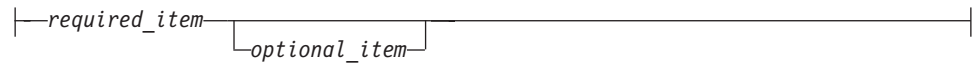


A repeat arrow above a stack indicates that you can repeat the items in the stack.

- Sometimes a diagram must be split into fragments. The syntax fragment is shown separately from the main syntax diagram, but the contents of the fragment should be read as if they are on the main path of the diagram.



**Fragment-name:**



- Keywords, and their minimum abbreviations if applicable, appear in uppercase. They must be spelled exactly as shown.
- Variables appear in all lowercase italic letters (for example, *column-name*). They represent user-supplied names or values.
- Separate keywords and parameters by at least one space if no intervening punctuation is shown in the diagram.
- Enter punctuation marks, parentheses, arithmetic operators, and other symbols, exactly as shown in the diagram.
- Footnotes are shown by a number in parentheses, for example (1).

---

## Notices and trademarks

This information was developed for products and services offered in the U.S.A.

### Notices

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 character set (DBCS) information, contact the IBM Intellectual Property Department in your country or send inquiries, in writing, to:

Intellectual Property Licensing  
Legal and Intellectual Property Law  
IBM Japan Ltd.  
1623-14, Shimotsuruma, Yamato-shi  
Kanagawa 242-8502 Japan

**The following paragraph does not apply to the United Kingdom or any other country where such provisions are inconsistent with local law:**

INTERNATIONAL BUSINESS MACHINES CORPORATION PROVIDES THIS PUBLICATION "AS IS" WITHOUT WARRANTY OF ANY KIND, EITHER EXPRESS OR IMPLIED, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF NON-INFRINGEMENT, MERCHANTABILITY OR FITNESS FOR A PARTICULAR PURPOSE. Some states do not allow disclaimer of express or implied warranties in certain transactions, therefore, this statement may not apply to you.

This information could include technical inaccuracies or typographical errors. Changes are periodically made to the information herein; these changes will be incorporated in new editions of the publication. IBM may make improvements and/or changes in the product(s) and/or the program(s) described in this publication at any time without notice.

Any references in this information to non-IBM Web sites are provided for convenience only and do not in any manner serve as an endorsement of those Web

sites. The materials at those Web sites are not part of the materials for this IBM product and use of those Web sites is at your own risk.

IBM may use or distribute any of the information you supply in any way it believes appropriate without incurring any obligation to you.

Licenses 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. The sample programs are provided "AS IS", without warranty of any kind. IBM shall not be liable for any damages arising out of your use of the sample programs.

Each copy or any portion of these sample programs or any derivative work, must include a copyright notice as follows:

© (your company name) (year). Portions of this code are derived from IBM Corp. Sample Programs. © Copyright IBM Corp. \_enter the year or years\_. All rights reserved.

If you are viewing this information softcopy, the photographs and color illustrations may not appear.

## **Trademarks**

IBM, the IBM logo, and [ibm.com](http://ibm.com) are trademarks of International Business Machines Corp., registered in many jurisdictions worldwide. Other product and service names might be trademarks of IBM or other companies. A current list of IBM trademarks is available on the Web at [www.ibm.com/legal/copytrade.shtml](http://www.ibm.com/legal/copytrade.shtml).

The following terms are trademarks or registered trademarks of other companies:

Adobe is a registered trademark of Adobe Systems Incorporated in the United States, and/or other countries.

IT Infrastructure Library is a registered trademark of the Central Computer and Telecommunications Agency which is now part of the Office of Government Commerce.

Intel, Intel logo, Intel Inside, Intel Inside logo, Intel Centrino, Intel Centrino logo, Celeron, Intel Xeon, Intel SpeedStep, Itanium, and Pentium are trademarks or registered trademarks of Intel Corporation or its subsidiaries in the United States and other countries.

Linux is a registered trademark of Linus Torvalds 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.

ITIL is a registered trademark, and a registered community trademark of the Office of Government Commerce, and is registered in the U.S. Patent and Trademark Office

UNIX is a registered trademark of The Open Group in the United States and other countries.

Cell Broadband Engine is a trademark of Sony Computer Entertainment, Inc. in the United States, other countries, or both and is used under license therefrom.

Java and all Java-based trademarks are trademarks of Sun Microsystems, Inc. in the United States, other countries, or both.

The United States Postal Service owns the following trademarks: CASS, CASS Certified, DPV, LACS<sup>Link</sup>, ZIP, ZIP + 4, ZIP Code, Post Office, Postal Service, USPS and United States Postal Service. IBM Corporation is a non-exclusive DPV and LACS<sup>Link</sup> licensee of the United States Postal Service.

Other company, product or service names may be trademarks or service marks of others.

---

# Index

## A

- ALTER ADD COLUMN command (event publishing) 253
- ALTER ADD COLUMN command (multidirectional replication) 178
- ALTER ADD COLUMN command (unidirectional replication) 80
- ALTER ALERT CONDITIONS FOR APPLY command 293
- ALTER ALERT CONDITIONS FOR CAPTURE command 296
- ALTER ALERT CONDITIONS FOR QAPPLY command 298
- ALTER ALERT CONDITIONS FOR QCAPTURE command 300
- ALTER CAPTURE PARAMETERS command (Q Replication) 82
- ALTER CONFIGURATION APPLY command 83
- ALTER CONTACT command 302
- ALTER DATASTAGE DEFINITION FOR command 13
- ALTER GROUP command 303
- ALTER MEMBER ADD COLS command (SQL Replication) 14
- ALTER MONITOR SUSPENSION command 304
- ALTER MONITOR SUSPENSION TEMPLATE command 305
- ALTER PUB command Event Publishing 256
- ALTER PUBQMAP command (Event Publishing) 254
- ALTER QSUB command (bidirectional replication) 179
- ALTER QSUB command (peer-to-peer replication) 182
- ALTER QSUB command (unidirectional Q Replication) 84
- ALTER REGISTRATION command (SQL Replication) 15
- ALTER REPLQMAP command (Q replication) 89, 184
- ALTER SUBSCRIPTION SET command (SQL Replication) 18
- ASNCLP commands 141, 220
  - ALTER ADD COLUMN (Event Publishing) 253
  - ALTER ADD COLUMN (multidirectional replication) 178
  - ALTER ADD COLUMN (unidirectional replication) 80
  - ALTER ALERT CONDITIONS FOR APPLY 293
  - ALTER ALERT CONDITIONS FOR CAPTURE 296
  - ALTER ALERT CONDITIONS FOR QAPPLY 298
  - ALTER ALERT CONDITIONS FOR QCAPTURE 300
- ASNCLP commands (*continued*)
  - ALTER CAPTURE PARAMETERS (Classic replication) 82
  - ALTER CONFIGURATION APPLY 83
  - ALTER CONTACT 302
  - ALTER GROUP 303
  - ALTER MEMBER ADD COLS (SQL Replication) 14
  - ALTER MONITOR SUSPENSION 304
  - ALTER MONITOR SUSPENSION TEMPLATE 305
  - ALTER PUB (Event Publishing) 256
  - ALTER PUB command (event publishing) 256
  - ALTER PUBQMAP (Event Publishing) 254
  - ALTER QSUB (bidirectional replication) 179
  - ALTER QSUB (peer-to-peer replication) 182
  - ALTER QSUB (unidirectional Q Replication) 84
  - ALTER REGISTRATION (SQL Replication) 15
  - ALTER REPLQMAP (Q replication) 89, 184
  - ALTER SUBSCRIPTION SET (SQL Replication) 18
  - ASNCLP SESSION SET TO (Q replication) 91, 186
  - ASNCLP SESSION SET TO (SQL Replication) 19
  - configuration file 3
  - CREATE ALERT CONDITIONS FOR APPLY 306
  - CREATE ALERT CONDITIONS FOR CAPTURE 308
  - CREATE ALERT CONDITIONS FOR QAPPLY 310
  - CREATE ALERT CONDITIONS FOR QCAPTURE 312
  - CREATE CONTACT 313
  - CREATE CONTROL TABLES (SQL Replication) 314
  - CREATE CONTROL TABLES FOR (Q replication) 92, 258
  - CREATE CONTROL TABLES FOR (SQL Replication) 20
  - CREATE GROUP 316
  - CREATE MEMBER (SQL Replication) 23
  - CREATE MONITOR SUSPENSION 317
  - CREATE MONITOR SUSPENSION TEMPLATE 318
  - CREATE MQ SCRIPT 98, 187
  - CREATE MQ SCRIPT (Event Publishing) 264
- ASNCLP commands (*continued*)
  - CREATE PUB command (event publishing) 268
  - CREATE PUBQMAP (event publishing) 266
  - CREATE QSUB (bidirectional replication) 190
  - CREATE QSUB (peer-to-peer replication) 197
  - CREATE QSUB (unidirectional Q Replication) 101
  - CREATE REGISTRATION (SQL replication) 33
  - CREATE REPLQMAP (Q replication) 122, 204
  - CREATE SCHEMASUB 124, 206
  - CREATE SUBSCRIPTION OPTIONS 127, 208
  - CREATE SUBSCRIPTION SET (SQL Replication) 40
  - DELEGATE CONTACT 319
  - DROP ALERT CONDITIONS FOR APPLY 320
  - DROP ALERT CONDITIONS FOR CAPTURE 320
  - DROP ALERT CONDITIONS FOR QAPPLY 321
  - DROP ALERT CONDITIONS FOR QCAPTURE 321
  - DROP CONTACT 322
  - DROP CONTROL TABLES (Q replication) 129, 210, 273
  - DROP CONTROL TABLES (SQL Replication) 42
  - DROP GROUP 322
  - DROP MEMBER (SQL Replication) 44
  - DROP MONITOR SUSPENSION 323
  - DROP MONITOR SUSPENSION TEMPLATE 323
  - DROP PUB (Event Publishing) 274
  - DROP PUB command (event publishing) 274
  - DROP PUBQMAP (Event Publishing) 273
  - DROP QSUB 129, 212
  - DROP REGISTRATION (SQL replication) 44
  - DROP REPLQMAP (Q replication) 132, 211
  - DROP SCHEMASUB 133, 214
  - DROP STMT (SQL Replication) 45
  - DROP SUBGROUP (multidirectional Q Replication) 215
  - DROP SUBSCRIPTION OPTIONS 133, 216
  - DROP SUBSCRIPTION SET (SQL Replication) 46
  - DROP SUBTYPE (bidirectional replication) 171



ASNCLP commands (*continued*)

- DROP SUBTYPE (peer-to-peer replication) 172
- interactive mode, using 7
- LIST APPLY SCHEMA (Q replication and event publishing) 137, 138, 216, 217, 276
- LIST CAPTURE SCHEMA (Q replication and event publishing) 138, 217, 276
- LIST MONITOR SUSPENSION 323
- LIST MONITOR SUSPENSION TEMPLATE 324
- LIST PUBQMAPS 275
- LIST PUBS 274
- LIST QSUB (Q Replication) 134
- LIST REPLQMAP (Q Replication) 135
- LIST SCHEMA (Q replication and event publishing) 138, 217, 276
- LIST SCHEMASUB 139, 218
- LOAD DONE (Q replication) 139, 219
- LOAD MULTIDIR REPL SCRIPT command (multidirectional Q Replication) 172
- multidirectional Q Replication 169
- OFFLINE LOAD (SQL Replication) 47
- PROMOTE PUB command 277
- PROMOTE PUBQMAP 279
- PROMOTE REGISTRATION (SQL Replication) 47
- PROMOTE REPLQMAP 142, 221
- PROMOTE SUBSCRIPTION SET (SQL Replication) 49
- REINIT SCHEMASUB 143, 223
- Replication Alert Monitor 291
- SET APPLY SCHEMA (Q replication) 144, 224
- SET BIDI NODE 224
- SET BIDIRECTIONAL NODE 224
- SET CAPTURE SCHEMA (Q replication) 145, 226, 280
- SET CAPTURE SCHEMA (SQL Replication) 51
- SET CONNECTION (multidirectional Q Replication) 227
- SET DROP (SQL Replication) 52
- SET DROP (unidirectional replication) 145
- SET ENFORCE MATCHING CONSTRAINTS (multidirectional Q Replication) 228
- SET LOG (Q replication) 147, 229, 281
- SET LOG (SQL Replication) 53
- SET MULTIDIR SCHEMA (multidirectional Q Replication) 174
- SET OUTPUT (monitor) 324
- SET OUTPUT (SQL Replication) 53
- SET OUTPUT (unidirectional Q replication) 148, 281
- SET OUTPUT command (multidirectional Q Replication) 229

ASNCLP commands (*continued*)

- SET PEER NODE 230
- SET PEERTOPEER NODE 230
- SET PROFILE (Q replication) 148, 232
- SET PROFILE (SQL Replication) 54
- SET QMANAGER (Q replication) 152, 236, 282
- SET REFERENCE TABLE command (multidirectional Q Replication) 237
- SET RUN SCRIPT (Event Publishing) 283
- SET RUN SCRIPT (Q replication) 152, 238
- SET RUN SCRIPT (SQL Replication) 58
- SET SERVER (Event Publishing) 285
- SET SERVER (multidirectional Q Replication) 174
- SET SERVER (Q replication and event publishing) 156
- SET SERVER (Replication Alert Monitor) 324
- SET SERVER (SQL Replication) 61
- SET SUBGROUP (multidirectional Q Replication) 241
- SET TABLES (multidirectional Q Replication) 175
- SET TRACE (Q replication) 159, 241, 288
- SET TRACE (SQL Replication) 64
- SHOW SET ENV (Q replication) 159, 241, 288
- SQL Replication 9
- START PUB (Event Publishing) 268, 288
- START PUB command (Event Publishing) 288
- START QSUB (Q replication) 160, 242
- START SCHEMASUB 162, 244
- STOP PUB (Event Publishing) 289
- STOP PUB command (Event Publishing) 289
- STOP QSUB (Q replication) 163, 245
- STOP SCHEMASUB 165, 247
- SUBSTITUTE CONTACT 326
- unidirectional Q Replication 79
- using input file 6
- VALIDATE WSMQ ENVIRONMENT FOR 165, 248, 289
- VALIDATE WSMQ MESSAGE FLOW FOR REPLQMAP 166, 249

ASNCLP program

- deprecated commands 171
- double quotation marks 5
- getting started 1

ASNCLP samples

- replicating from a view 11

ASNCLP scripts

- using 6

ASNCLP SESSION SET TO command

- SQL Replication 19

ASNCLP SESSION SET TO command (Q replication) 91, 186

## B

- batch mode
  - overview 6
- bidirectional Q Replication
  - sample scripts 71
- binding 3

## C

Classic replication

- configuration file 3

CLASSPATH environment variable 2

CREATE ALERT CONDITIONS FOR APPLY command 306

CREATE ALERT CONDITIONS FOR CAPTURE command 308

CREATE ALERT CONDITIONS FOR QAPPLY command 310

CREATE ALERT CONDITIONS FOR QCAPTURE command 312

CREATE CONTACT command 313

CREATE CONTROL TABLES command (SQL Replication) 314

CREATE CONTROL TABLES FOR command (Q replication) 92, 258

CREATE CONTROL TABLES FOR command (SQL Replication) 20

CREATE DATASTAGE DEFINITION FOR command 23

CREATE GROUP command 316

CREATE MEMBER command (SQL Replication) 23

CREATE MONITOR SUSPENSION command 317

CREATE MONITOR SUSPENSION TEMPLATE command 318

CREATE MQ SCRIPT command 98, 187

CREATE MQ SCRIPT command (event publishing) 264

CREATE PUB command (Event Publishing) 268

CREATE PUBQMAP command (Event Publishing) 266

CREATE QSUB command (bidirectional replication) 190

CREATE QSUB command (peer-to-peer replication) 197

CREATE QSUB command (unidirectional Q Replication) 101

CREATE REGISTRATION command (SQL Replication) 33

CREATE REPLQMAP command (Q replication) 122, 204

CREATE SCHEMASUB command 124, 206

CREATE STMT command (SQL Replication)
 

- ASNCLP commands 38
- CREATE STMT (SQL Replication) 38

CREATE SUBSCRIPTION OPTIONS command 127, 208

CREATE SUBSCRIPTION SET command (SQL Replication) 40

customer support
 

- contacting 327



## D

DELEGATE CONTACT command 319  
deprecated commands, ASNCLP 171  
DROP ALERT CONDITIONS FOR  
  APPLY command 320  
DROP ALERT CONDITIONS FOR  
  CAPTURE command 320  
DROP ALERT CONDITIONS FOR  
  QAPPLY command 321  
DROP ALERT CONDITIONS FOR  
  QCAPTURE command 321  
DROP CONTACT command  
  replicatoin 322  
DROP CONTROL TABLES command  
  (SQL Replication) 42  
DROP CONTROL TABLES ON command  
  (Q replication) 129, 210, 273  
DROP DATASTAGE DEFINITION FOR  
  command 43  
DROP GROUP command 322  
DROP MEMBER command (SQL  
  Replication) 44  
DROP MONITOR SUSPENSION  
  command 323  
DROP MONITOR SUSPENSION  
  TEMPLATE command 323  
DROP PUB command (Event  
  Publishing) 274  
DROP PUBQMAP command (Event  
  Publishing) 273  
DROP QSUB command 129, 212  
DROP REGISTRATION command (SQL  
  Replication) 44  
DROP REPLQMAP command (Q  
  replication) 132, 211  
DROP SCHEMASUB command 133, 214  
DROP STMT command (SQL  
  Replication) 45  
DROP SUBGROUP command  
  (multidirectional Q Replication) 215  
DROP SUBSCRIPTION OPTIONS  
  command 133, 216  
DROP SUBSCRIPTION SET command  
  (SQL Replication) 46  
DROP SUBTYPE command (bidirectional  
  replication) 171  
DROP SUBTYPE command (peer-to-peer  
  replication) 172

## E

Event Publishing  
  sample scripts 251, 252

## I

input files  
  ASNCLP commands 6  
interactive mode 7

## J

Java  
  environment  
  setting up 2

## L

legal notices 331  
LIST APPLY SCHEMA command 137,  
  138, 216, 217, 276  
LIST CAPTURE SCHEMA  
  command 138, 217, 276  
LIST MONITOR SUSPENSION  
  command 323  
LIST MONITOR SUSPENSION  
  TEMPLATE command 324  
LIST PUBQMAPS command 275  
LIST PUBS command 274  
LIST QSUB command (Q  
  Replication) 134  
LIST REPLQMAP command (Q  
  Replication) 135  
LIST SCHEMA command 138, 217, 276  
LIST SCHEMASUB command  
  (unidirectional Q Replication) 139, 218  
LOAD DONE command (Q  
  replication) 139, 219  
LOAD MULTIDIR REPL SCRIPT  
  command (multidirectional Q  
  Replication) 172

## M

multidirectional Q Replication  
  commands 169

## O

OFFLINE LOAD command  
  SQL Replication 47  
operating systems  
  supported for stored procedures 2  
Oracle  
  replication  
    configuration file 3

## P

peer-to-peer configurations  
  promoting 77  
peer-to-peer Q Replication  
  three servers 74  
  two servers 72  
PROMOTE PUB command 277  
PROMOTE PUBQMAP command 279  
PROMOTE QSUB 220  
PROMOTE QSUB (unidirectional) 141  
PROMOTE QSUB command 220  
PROMOTE QSUB command  
  (unidirectional) 141  
PROMOTE REGISTRATION command  
  (SQL Replication) 47  
PROMOTE REPLQMAP command 142,  
  221  
PROMOTE SUBSCRIPTION SET  
  command (SQL Replication) 49  
promoting  
  peer-to-peer configurations  
    samples 77  
  publications 277  
  publishing queue maps 279

promoting (*continued*)  
  unidirectional configurations 76  
promoting Q subscriptions,  
  unidirectional 141  
promoting subscriptions 220

## Q

Q subscriptions  
  ALTER QSUB 84  
  changing 84  
  promoting  
    unidirectional 141

## R

REINIT SCHEMASUB command 143,  
  223  
Replication Alert Monitor  
  ASNCLP commands 291  
  sample script 292

## S

sample script  
  Replication Alert Monitor 292  
sample scripts  
  bidirectional Q Replication 71  
  Event Publishing 251, 252  
  peer-to-peer Q Replication (three  
    servers) 74  
  peer-to-peer Q Replication (two  
    servers) 72  
  promoting peer-to-peer  
    configurations 77  
  SQL Replication 10, 11  
  unidirectional Q Replication 67, 68  
SET APPLY SCHEMA command (Q  
  replication) 144, 224  
SET BIDI NODE command 224  
SET BIDIRECTIONAL NODE  
  command 224  
SET CAPTURE SCHEMA command (Q  
  replication) 145, 226, 280  
SET CAPTURE SCHEMA command (SQL  
  Replication) 51  
SET CONNECTION command  
  (multidirectional Q Replication) 227  
SET DROP command (SQL  
  Replication) 52  
SET DROP command (unidirectional  
  replication) 145  
SET ENFORCE MATCHING  
  CONSTRAINTS command  
  (multidirectional Q Replication) 228  
SET LOG command (Q replication) 147,  
  229, 281  
SET LOG command (SQL  
  Replication) 53  
SET MULTIDIR SCHEMA command  
  (multidirectional Q replication) 174  
SET OUTPUT command (monitor) 324  
SET OUTPUT command (multidirectional  
  Q Replication) 229  
SET OUTPUT command (SQL  
  Replication) 53

- SET OUTPUT command (unidirectional Q replication) 148, 281
- SET PEER NODE command 230
- SET PEERTOPEER NODE command 230
- SET PROFILE command (Q replication) 148, 232
- SET PROFILE command (SQL Replication) 54
- SET QMANAGER command (Q replication) 152, 236, 282
- SET REFERENCE TABLE command (multidirectional Q replication) 237
- SET RUN SCRIPT command (Event Publishing) 283
- SET RUN SCRIPT command (Q replication) 152, 238
- SET RUN SCRIPT command (SQL Replication) 58
- SET SERVER command (Event Publishing) 285
- SET SERVER command (multidirectional Q Replication) 174
- SET SERVER command (Q replication and event publishing) 156
- SET SERVER command (Replication Alert Monitor) 324
- SET SERVER command (SQL Replication) 61
- SET SUBGROUP command (multidirectional Q Replication) 241
- SET TABLES command (multidirectional Q Replication) 175
- SET TRACE command (Q replication) 159, 241, 288
- SET TRACE command (SQL Replication) 64
- SHOW SET ENV command (Q replication) 159, 241, 288
- software services
  - contacting 327
- SQL Replication
  - ASNCLP commands 9
  - sample scripts 10, 11
- START PUB command
  - Event Publishing 288
- START QSUB command (Q replication) 160, 242
- START SCHEMASUB command 162, 244
- STOP PUB command
  - Event Publishing 289
- STOP QSUB command (Q replication) 163, 245
- STOP SCHEMASUB command 165, 247
- subscriptions
  - promoting 220
- SUBSTITUTE CONTACT command 326
- support
  - customer 327
- supported operating systems 2

## T

- trademarks
  - list of 331

## U

- unidirectional Q Replication
  - commands 79
  - sample scripts 67, 68

## V

- VALIDATE WSMQ ENVIRONMENT
  - FOR command 165, 248, 289
- VALIDATE WSMQ MESSAGE FLOW
  - FOR REPLQMAP command 166, 249
- views, replicating from 11

## W

- web sites
  - non-IBM 329

## Z

- z/OS
  - binding packages 3





Printed in USA

SC19-3639-00



Spine information:

IBM InfoSphere Data Replication    **Version 10.1.3**

**ASNLCP Program Reference for Replication and Event Publishing**

