IBM® DB2®

Replication Guide and Reference

Version 6

SC26-9642-00

Note

Before using this information and the product it supports, be sure to read the general information under “Appendix C. Notices” on page 421.

First Edition (June 1999)

This edition applies to the following releases and to all subsequent releases and modifications until otherwise indicated in new editions:

- DB2 DataPropagator for OS/390, a feature of Version 6 of DB2 Universal Database Server for OS/390 (DB2 UDB for OS/390) (5645-DB2)
- DataPropagator Relational Version 5 Release 1 for AS/400 (5769-DP2)
- The following DB2 Universal Database Version 6 program products:
  - DB2 Data Links Manager (5648-B90)
  - DB2 Enterprise - Extended Edition (5648-B91)
  - DB2 Query Patroller (5648-B92)
  - DB2 Universal Developer’s Edition (5648-B93)
  - DB2 Personal Developer’s Edition (5648-B94)
  - DB2 Connect Enterprise Edition (5648-B95)
  - DB2 Connect Personal Edition (5648-B96)
  - UDB Enterprise Edition (5648-B97)
  - UDB Personal Edition (5648-B98)
  - UDB Workgroup Edition (5648-B99)
  - DB2 Satellite Edition (5648-C34)

Contents

About this book

This book introduces the concepts of DB2 data replication, and it describes how to plan, configure, and administer a replication environment.

The DB2 DataPropagator product is the focus of the book. You can use it along with other products in the IBM replication solution to tailor a replication environment that suits your business needs.

You can replicate data from DB2 sources to DB2 targets. You can also replicate data between DB2 and non-IBM sources and targets. Specifically, you can use the following database management systems as sources, targets, or both:

- DB2 for AIX
- DB2 for AS/400
- DB2 for HP-UX
- DB2 for OS/2
- DB2 for OS/390
- DB2 for SCO UnixWare 7
- DB2 for Solaris
- DB2 for VM[1]
- DB2 for VSE[1]
- DB2 for Windows 95
- DB2 for Windows NT
- Informix[2]
- Microsoft Access[3]
- Microsoft Jet[3]
- Microsoft SQL Server[2]
- Oracle[2]
- Sybase[2]
- Sybase SQL Anywhere[2]

1. There is no Apply program for these products.
2. These products require DB2 DataJoiner V2 or later and the DB2 DataJoiner Replication Administration (DJRA) tool.
3. These products require DJRA.

Who should read this book

This book is written for database administrators, LAN administrators, and others who must set up and maintain a data replication environment. It assumes that you are familiar with standard database terminology, that you have experience with database design and database administration, and that you understand your applications and the data that you want to replicate.

How this book is structured

This book consists of the following parts:

Part 1: Introduction
Introduces the DB2 replication concepts and components, describes the typical replication configurations, guides you through a simple replication scenario using the DB2 Control Center, and outlines the major replication tasks that are described in the following chapters.

Part 2: Administration
Describes how to plan, set up, and maintain your replication environment.

Part 3: Operations
Describes how to operate the Capture and Apply programs for particular operating systems.

Part 4: Occasionally connected environments
Provides information about DB2 Universal Database Satellite Edition, DB2 DataPropagator for Micrsoft Jet, and DB2 mobile replication.

Part 5: The DB2 DataJoiner Replication Administration tool
Describes how to operate the DataJoiner Replication Administration (DJRA) tool.

Part 6: Reference information
Provides reference information for table structures, problem determination facilities, and messages.

Conventions

This book uses these highlighting conventions:

- **Boldface type** indicates commands or user interface controls such as names of fields, folders, icons, or menu choices.
- `Monospace type` indicates examples of text you that enter exactly as shown.
- *Italic type* indicates variables that you should replace with a value. It is used also to indicate book titles and for emphasis of words.

How to read syntax diagrams

The following rules apply to the syntax diagrams used in this book:

- Read the syntax diagrams from left to right, from top to bottom, following the path of the line.

  The ►►─── symbol indicates the beginning of a statement.

  The ───► symbol indicates that the statement syntax is continued on the next line.

  The ►─── symbol indicates that a statement is continued from the previous line.

  The ───►◄ symbol indicates the end of a statement.

  Diagrams of syntactical units other than complete statements start with the ►─── symbol and end with the ───► symbol.
- Keywords, their allowable synonyms, and reserved parameters, are either shown in uppercase or lowercase, depending on the operating system. These items must be entered exactly as shown. Variables appear in lowercase italics (for example, *column-name*). They represent user-defined parameters or suboptions.

  When entering commands, separate the parameters and keywords by at least one blank if there is no intervening punctuation.
- Enter punctuation marks (slashes, commas, periods, parentheses, quotation marks, equal signs, and so on) and numbers exactly as given.
- Footnotes are shown by a number in parentheses, for example, (1).
- Required items appear on the horizontal line (the main path).

```
►►──required_item──────────────────────────────────────────────────►◄
```

- Optional items appear below the main path.

```
►►──required_item──┬───────────────┬───────────────────────────────►◄
                   └─optional_item─┘
```

- 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.

```
►►──required_item──┬─required_choice1─┬────────────────────────────►◄
                   └─required_choice2─┘
```

  If choosing one of the items is optional, the entire stack appears below the main path.

```
►►──required_item──┬──────────────────┬──────────────────────────────────►◄
                   ├─optional_choice1─┤
                   └─optional_choice2─┘
```

Road map

If you want to ...	Refer to ...
Learn about last-minute changes to the product	The Installation Notes on the CD-ROM or the Release Notes that are installed with the products.
Learn what's new this release in DB2 replication for relational data	"What's new" on page xvii.
Learn about the components of DB2 data replication	"Chapter 1. Overview of data replication" on page 3.
Learn about DB2 data replication concepts	"Chapter 1. Overview of data replication" on page 3.
Learn about typical replication configurations	"Chapter 2. Data replication configurations" on page 19.
Work through a simple replication scenario using the DB2 Control Center on Windows NT	"Chapter 3. Data replication scenario" on page 33.
Get an overview of the types of replication tasks that you can perform	"Chapter 4. Data replication tasks" on page 49.
Design and plan your replication environment	"Chapter 5. Planning for replication" on page 59.
Learn about the IBM data replication solution	See one of the following Web sites: DB2 DataPropagator at http://www.software.ibm.com/data/dpropr/ Data replication solution at http://www.software.ibm.com/data/dbtools/datarepl.html IBM data management at http://www.software.ibm.com/data/
Read customer case studies	http://www.software.ibm.com/data/dpropr/casestudy.html
Set up your replication environment	"Chapter 6. Setting up your replication environment" on page 83.
Migrate from earlier versions of DPROPR to DB2 DataPropagator	See the online migration guide at http://www.software.ibm.com/data/dpropr/
Learn how to use DJRA	"Part 5. The DB2 DataJoiner Replication Administration tool" on page 269.
Set replication preferences in the DB2 Control Center	See the DB2 Control Center online help.

If you want to ...	Refer to ...
Define and manage replication sources and targets	“Part 2. Administration” on page 57.
Configure and operate the Capture and Apply programs	“Part 3. Operations” on page 133; see the chapter for a particular operating system.
Learn about the replication support for DB2 Universal Database for Satellite Edition	“Chapter 13. Satellite replication” on page 239.
Learn about and run DB2 DataPropagator for Microsoft Jet	“Chapter 14. Mobile replication using DB2 DataPropagator for Microsoft Jet” on page 245.
Learn about the relational database tables that control the DB2 replication process	“Chapter 21. Table structures” on page 299.
Debug error messages for the Capture and Apply programs	“Troubleshooting” on page 126.

To locate information on other topics, see “Appendix A. How the DB2 library is structured” on page 403 for a complete description of the DB2 Universal Database library.

How to send your comments

Your feedback helps IBM to provide quality information. Please send any comments that you have about this book or other DB2 DataPropagator documentation. You can use any of the following methods to provide comments:

- Send your comments from the Web. Visit the Web site at:

  http://www.software.ibm.com/data/dpropr/

  The Web site has a feedback page that you can use to enter and send comments.
- Send your comments by e-mail to comments@vnet.ibm.com. Be sure to include the name of the product, the version number of the product, and the name and part number of the book (if applicable). If you are commenting on specific text, please include the location of the text (for example, a chapter and section title, a table number, a page number, or a help topic title).

What’s new

This section introduces the major features of DB2 DataPropagator Version 6 (V6), most of which are described in more detail in this book.

- “DB2 Satellite Edition”
- “Database currency”
- “Performance features” on page xviii
- “Integration with DB2” on page xix

DB2 Satellite Edition

DB2 Universal Database Satellite Edition (DB2 Satellite Edition) brings the power of DB2 Universal Database to environments containing occasionally connected clients. It is supported on Windows 95, Windows 98, and Windows NT operating systems.

With DB2 Satellite Edition, you can replicate data between DB2 servers and some non-IBM source servers (Oracle, Sybase, Informix, Microsoft), and you gain the following benefits:

- Centralized group administration and problem determination
- Ability to easily support thousands of occasionally connected clients
- Capture and Apply programs that start and stop automatically, as required

Two new options let you run the Capture and Apply programs in batch mode:

Capture program autostop option
You can run the Capture program in batch mode by using the new AUTOSTOP invocation parameter. The Capture program will run until it captures all of the changes from the log and then it terminates.

Apply program copyonce option
You can run the Apply program in batch mode by using the new COPYONCE invocation parameter. The Apply program will try running all eligible subscription sets once and then it terminates.

Database currency

DB2 replication now provides:

LOB support
You can use DB2 DataPropagator to replicate columns with large-object (LOB) data. The Capture program flags information about changed LOB data, but does not copy the LOB data to staging tables. The Apply program copies changed LOB data directly from the source table to the target table. You can replicate LOB data between DB2 for OS/390 servers or between DB2 Universal Database servers for UNIX, Windows, and OS/2 operating systems. If you have DB2 Connect Version 6, you can also copy LOB data between DB2 for OS/390 and DB2 Universal Database servers for UNIX, Windows, and OS/2 operating systems.

ROWID support
DB2 for OS/390 V6 allows you to generate a unique identifier for each row in a table and store that identifier in a ROWID column. Using the ROWID, you can access a particular row without needing to access an index or scan the table space. Usually a ROWID identifies a row in the table that has the ROWID defined. However, by copying ROWID values from the source table to set the target rows with the same ROWID values, you can use the same ROWID value to identify both the source and target row.

Version-independent capture
The Capture program can read the log for DB2 for MVS/ESA V4, DB2 for OS/390 V5, or DB2 for OS/390 V6. Therefore you can run different versions of DB2 in a data-sharing environment, for example during version-to-version migration, and have one Capture program continue to capture transaction-consistent data.

ODBC support
This feature was added in DB2 for OS/390 V5 and updated in V6 to support a new catalog table (SQLProcedureColumns).

Unicode support
Unicode support is built into DB2 DataPropagator for UNIX, Windows, and OS/2 operating systems.

Port to Linux
DB2 DataPropagator V6 was ported to the Linux operating system.

Performance features

The following features were added to improve the performance of your replication environment:

Option to capture only columns that are available for replication
You can start the Capture program using the CHGONLY parameter if you want to capture changes only for those columns that you marked

available for replication. By default, the Capture program captures changes that are made to the source table data for all columns.

Capture program sleeptime option
: You can use the SLEEP=N invocation parameter to indicate how long you want the Capture program to sleep before it reads the log again, after it reaches the end of the log. This parameter is supported for DB2 for MVS 4.1 and later with data sharing.

Continuous block fetch by the Apply program
: The Apply program uses DB2 continuous block fetch to improve data-transfer rates and overall performance for replicating data from DB2 for OS/390 to other operating systems.

Automated use of RUNSTATS utility
: For DB2 DataPropagator on UNIX, Windows, and OS/2 operating systems, the RUNSTATS utility is run automatically after the Apply program completes a full-refresh copy to the target tables. This utility collects new statistics on the target tables and their indexes.

Integration with DB2

The replication components are integrated with DB2 Universal Database more closely than before.

DB2 DataPropagator for OS/390 feature
: When you order DB2 for OS/390 V6, you also receive a free trial version of DB2 DataPropagator for OS/390. This trial version has all the features and functions of the standard product. You can use the trial version for 90 days without paying a licensing charge. After the trial period ends, you must order a license for the DB2 DataPropagator priced feature to continue using the product.

Integrated install for OS/390
: You can install DB2 DataPropagator for OS/390 V6 using the DB2 Installer interface. The DB2 Installer is a workstation-based tool that provides an easy-to-read map for the entire installation process. The status of a task is denoted by the changed icons on the windows. Help text is provided for each field, and ″Guide Me″ help text is provided for each window.

Web Control Center
: A Java version of the DB2 Control Center is available. You can use it to perform administration tasks from a Web browser. It includes improved support for administering replication environments for DB2 for OS/390. For more information about this new interface, see the DB2 Universal Database documentation.

DataJoiner Replication Administration tool

The DataJoiner Replication Administration (DJRA) tool comes with DB2 Universal Database and it runs on Windows 95, Windows 98, and Windows NT operating systems. With DJRA, you can perform administration tasks for replication configurations involving DB2 databases, non-IBM databases, or both. Using DJRA, you can easily promote (copy) your replication criteria from one environment to another. You can also start a replication monitor to help you monitor replication activity and produce periodic reports about that activity. If your files are stored on the AS/400 platform, you can use DJRA to automatically generate relative record numbers (RRN) for replication sources so that you can replicate data without user-defined unique keys. The details about these and other benefits of DJRA are documented in this book.

Part 1. Introduction

This part of the book contains the following chapters:

Chapter 1. Overview of data replication

Replication is a process of maintaining a defined set of data in more than one location. It involves copying designated changes from one location (a source) to another (a target), and synchronizing the data in both locations. The source and target can be in logical servers (such as a DB2 database or a DB2 for OS/390 subsystem or data-sharing group) that are on the same machine or on different machines in a distributed network.

A number of IBM products enable you to replicate data. The product that is the focus of this book—DB2 DataPropagator—is a replication product for relational data. You can use it to replicate changes between any DB2 relational databases. You can also use it with other IBM products (such as DB2 DataJoiner and DataPropagator NonRelational) or non-IBM products (such as Microsoft SQL Server and Sybase SQL Server) to replicate data between a growing number of database products—both relational and nonrelational.

The replication environment that you need depends on when you want data updated and how you want transactions handled. The solution that's right for you might have a log-based or a trigger-based change-capture mechanism, or both. You also have the flexibility to choose the locations of the replication components to maximize the efficiency of your replication environment.

Before you go to Chapter 2 and begin designing your replication environment, read this chapter to familiarize yourself with the DB2 replication components and their associated concepts.

DB2 data replication components

DB2 DataPropagator consists of three main components: administration interfaces, change-capture mechanisms, and the Apply program.

- You use the administration interfaces to create control tables, which store your replication criteria.
- After you set up your replication environment, you use a change-capture mechanism to capture changes as they occur in the source database and store them temporarily in tables.
- You use the Apply program to read the tables and apply these changes to target databases, or to copy data directly from the source database to the target databases.

This section describes the control tables that manage replication requests, the logical servers that contain the replication components, as well as the main components (administration interfaces, change-capture mechanisms, and the Apply program) and how they communicate with each other.

Control tables

The replication components use control tables to communicate with each other and to manage replication requests (such as defining and managing replication sources and targets, capturing changes, replicating changes, and tracking how many changes are replicated and how many remain to be done).

The change-capture mechanisms use the following control tables: register table, unit-of-work table, pruning control table, prune lock table, critical section table, warm start table, tuning parameters table, and change data tables.

The Apply program uses the following control tables: Apply trail table, critical section table, pruning control table, prune lock table, register table, subscription set table, subscription statements table, subscription events table, subscription-targets-member table, subscription columns table, unit-of-work table, and change data tables.

Logical servers

All the replication components reside on a logical server. In this book, logical servers refer to *databases*, not to servers in the client/server sense. For the OS/390 operating system, logical servers are equivalent to *subsystems* or *data-sharing groups* (that is, the domain of a single database catalog). There are three types of logical servers:

Source server
The source server contains the change-capture mechanism, the source tables that you want to replicate, and the control tables for the Capture program.

Target server
The target server contains the target tables.

Control server
The control server contains the control tables for the Apply program.

The Apply program can reside on any of the logical servers in the network. It uses distributed DB2 technology to connect to the control, source, and target servers.

Each Apply program is associated with one control server, which you specify when you start the Apply program. Multiple Apply programs can share a control server.

The control server can be located at the source server, the target server, or any database server that the Apply program can connect to. For better performance, the control server should be located at the server where the Apply program runs because the Apply program frequently reads the control tables at the control server. However, if the source server is in a secure environment, locating the control server at the source server can improve security and let you manage and monitor subscriptions centrally.

Administration interfaces

You use the administration interfaces to create control tables, which store your replication criteria. There are two user interfaces available: DB2 Control Center and DataJoiner Replication Administration (DJRA).

DB2 Control Center

The DB2 Control Center is a database administration tool that you can use to administer the replication of data between DB2 servers. It automates many initialization functions, such as creating target tables and control tables when you specify target information.

You can use the Control Center to perform the following administration tasks for replication:

- Define DB2 tables and DB2 views as replication sources.
- Define or remove subscription sets.
- Add subscription-set members to existing subscription sets.
- Remove subscription-set members from existing subscription sets.
- Remove replication sources.
- Clone subscription sets to other servers.
- Activate and deactivate subscription sets.
- Add or delete SQL statements or a call to a procedure that will run before or after data is replicated.

DataJoiner Replication Administration (DJRA)

The DataJoiner Replication Administration (DJRA) tool is a database administration tool that you can use to perform various replication administration tasks. You can use this tool for DB2-to-DB2 replication; however, you must use it if your replication environment includes non-IBM databases.

You can use DJRA to perform the following administration tasks:

- Create the control tables and put them on your source, target, and control servers.

- Define DB2 tables, non-IBM tables, and DB2 views as replication sources.
- Alter the definitions for existing DB2 source and target tables to add new columns.
- Remove replication sources.
- Define or remove subscription sets.
- Add subscription-set members to existing subscription sets.
- Remove subscription-set members from existing subscription sets.
- Add or delete SQL statements or called procedures that run before or after data is replicated.
- Monitor the replication process.
- Copy your replication environment to another system using the promote functions.
- Load target tables off-line.

Change-capture mechanisms

The DB2 data replication solution offers these mechanisms for capturing data:

- The *Capture program* for DB2 source tables.
- *Capture triggers* for source tables in non-IBM databases, except Microsoft Access and Microsoft Jet.

The following sections describe the Capture program and triggers. For more information about how changes are replicated in Microsoft Access and Microsoft Jet databases, see “Chapter 14. Mobile replication using DB2 DataPropagator for Microsoft Jet” on page 245.

Capture Program

When the source is a DB2 table, the Capture program is used to capture changes that are made to the source. The Capture program uses the database log[4] to capture changes made to the source database and enqueues them temporarily.

The Capture program runs at the source server. Typically it runs continuously, but you can stop it while running utilities or modifying replication sources.

Tasks that you can perform with the Capture program include:

- Starting the Capture program
- Scheduling the Capture program

4. The Capture program retrieves changed and committed information from the active and archive logs on DB2 for MVS 4.1 or higher and DB2 Universal Database. Capture for VSE and VM 5.1 can read only the active log on DB2 for VSE & VM.

- Stopping the Capture program
- Suspending the Capture program temporarily
- Resuming the Capture program
- Reinitializing the Capture program
- Pruning the tables that temporarily store change data

See "Part 3. Operations" on page 133 for instructions about performing these tasks.

Capture triggers

When the source table is in a non-IBM database (other than Microsoft Access and Microsoft Jet), Capture triggers are used to capture committed changes made to the source. Capture triggers are fired when a particular database event (UPDATE, INSERT, DELETE) occurs.

DJRA automatically creates the Capture triggers. These triggers capture the changes made to defined replication source tables and store them temporarily in tables.

Apply program

The Apply program reads data directly from source tables or views to initially populate the target table. If you want changes captured, the Apply program reads the changed data that was previously captured and stored temporarily in staging tables, and applies the changes to target tables.

The Apply program generally runs at the target server, but it can run at any server in your network that can connect to the source, control, and target servers. Several Apply program instances can run on the same or different servers. Each Apply program can run using the same authorization, different authorization, or as part of a group of Apply programs where each Apply program in the group runs using the same authorization (user ID).

Tasks that you can perform with the Apply program include:
- Starting the Apply program
- Running exit routines (such as ASNLOAD to call an IBM or a vendor utility)
- Scheduling the Apply program
- Stopping the Apply program

See "Part 2. Administration" on page 57 for instructions about performing these tasks.

How the replication components communicate

The replication components are independent of each other, so they rely on information that is stored in control tables to communicate with each other. The Capture and Apply programs update control tables to indicate the progress of replication and to coordinate the processing of changes.

The replication components communicate differently depending on whether the source server is a DB2 server or a non-IBM server. For replication between DB2 servers, the Capture program captures changes that are made to data in replication source tables by reading the server *log* or *journal*. Then the Capture program places the changes into change data (CD) tables. For non-IBM sources, Capture triggers capture changes and store them in consistent-change-data (CCD) tables.

Each time that the Apply program copies data to the target database, the contents of the target database reflect the changes that were made to the source database. The Apply program works by applying updates accumulated since the Apply program last ran. The Apply program keeps track of the latest update that it makes to each target.

Log-based communication

The Capture program uses some of the control tables to indicate what changes have been made to the source database, and the Apply program uses these control tables to detect what needs to be copied to the target database.

Important: The Capture program will not capture any information until the Apply program signals it to do so, and the Apply program will not signal the Capture program to start capturing changes until you define a replication source and associated subscription sets. See "Performing the initial replication" on page 52 for more information about the steps you must perform so that the components communicate with each other and replicate changes.

The following steps describe how the Apply and Capture programs communicate in a *typical* replication scenario to ensure data integrity:
Capturing data from a source database

1. The Capture program reads the register table to determine the replication sources for which it needs to start capturing changes. If new replication sources are defined while the Capture program is running, they won't be recognized by the Capture program until you run the **reinit** command, or until you stop and restart the Capture program.
2. The Capture program monitors the DB2 log or journal to detect change records from source tables that are defined as replication sources.

3. The Capture program adds one row (or two rows if updates are saved as DELETE and INSERT operations) to the *change data (CD) table* for each change that it finds in the DB2 log or journal. Each replication source has a CD table.
4. The Capture program stores information about committed transactions in the *unit-of-work (UOW) table*. The rows in this control table identify the transactions that have been committed in the replication source server. With log-based change capture, there is one UOW table for every DB2 source server.
5. The Capture program updates the register table to record how much committed data was captured for each replication source.

Applying data to a target database

6. When the Apply program is started, it checks to see if any subscription set is due for replication. If so, the Apply program checks the register table to determine whether there are changes that need to be replicated.
7. Before the Apply program can copy the changes to the target database, it synchronizes the target with the replication source by copying all the data from the source table to the target table. This action is called a *full-refresh copy*. After the full-refresh copy, the Capture program begins capturing changes at the source.
8. The Apply program updates the pruning control table to synchronize the capture of the related source table changes in the CD table.
9. The Apply program copies the changes from the join of the CD table and the UOW table to the target table. By joining the two control tables, the Apply program ensures that it copies only the changes that were committed at the replication source.

Pruning the control tables

10. The Apply program updates the pruning control table with a value that indicates the point to which it copied changes to the target database.
11. When the Capture program prunes the CD and UOW control tables, it determines which changes were applied and deletes them from the CD table and the UOW table.

Trigger-based communication

DJRA, working through DB2 DataJoiner, creates Capture triggers at the non-IBM source table when you define the table as a replication source. Three types of triggers are created on the source table: DELETE, UPDATE, and INSERT. Also, UPDATE triggers are created on the pruning control table and the register synchronization table. The Apply program uses these control tables to detect what needs to be copied to the target database.

The following steps describe how the Capture triggers and the Apply program communicate in a *typical* replication scenario to ensure data integrity:

Capturing data from a source

1. Whenever a DELETE, UPDATE, or INSERT operation occurs at the source table that is defined as a replication source, a Capture trigger records the change in the *consistent-change-data (CCD) table*.

Applying data to a target

2. When the Apply program is started, the UPDATE trigger on the register synchronization table updates the register table to record how much committed data has been captured.
3. The Apply program gets the source table information from the register table.
4. Before the Apply program can copy the changes to the target, it synchronizes the target with the replication source by copying all the data from the source table to the target table. This action is called a *full-refresh copy*.
5. The Apply program updates the pruning control table to synchronize the capture of the related changes in the CCD table.
6. The Apply program reads the CCD table using DB2 DataJoiner nicknames, copies the changes to the target server, and applies the changes to the target table.

Pruning the control tables

7. The Apply program updates pruning control table with a value that indicates the point to which it copied changes to the target database.
8. The UPDATE trigger on the pruning control table checks all of the CCD tables that are at the source server and deletes those entries that were replicated.

DB2 data replication concepts

This section introduces some of the important concepts of DB2 data replication. You should read the whole section to get a complete overview.

Replication sources

A *Replication source* is a *user table* or view from which you want data copied. Before you can replicate data, you must define a replication source to describe the information that the change-capture mechanisms will use. When you define a replication source, you define the following attributes:

- Which columns you want to replicate
- Whether you want before-image values for a column
- Whether you want to replicate without change capture (full-refresh only)

- Whether you want updates treated as UPDATE operations or DELETE and INSERT operations
- For update-anywhere replication (where a replication source has read / write target tables), what level of conflict detection to use

Full-refresh and differential-refresh copying

The Apply program copies data from the source to the target either by full-refresh only or differential-refresh copying.

During *full-refresh only* copying, the Apply program performs these steps:

1. Empties (deletes) all of the rows from the target table
2. Reads all of the rows from the source table
3. Copies the rows to the target table

During *differential-refresh copying*, the Apply program copies only the changed data to the target table.

After-image columns and before-image columns

An *after-image* column contains the value of a data column in a source table after the value in that data column is updated. A *before-image* column contains the value of a data column in a source table before that data column is updated. When you define a replication source, you can choose to capture only the after-image or both the after-image and the before-image. Your decision will depend both on the way in which you plan to use the data and on the types of tables that you are using.

Before-image columns are useful if your applications require auditing or rollback capability. Some restrictions apply to how you use these columns, and they are discussed later in this book ("Replicating before and after images" on page 70).

Levels of conflict detection

Conflict detection pertains only to update-anywhere replication configurations. It is the process of detecting if the same row was updated in the source and target tables during the same replication cycle. With *standard conflict detection*, the Apply program searches for conflicts in rows that are already captured in the CD tables. With *enhanced conflict detection*, the Apply program locks all of the target tables, thus preventing further transactions until the current changes are captured. *Row-replica conflict detection* applies only to tables that are maintained by DataPropagator for Microsoft Jet; where conflicts are detected on a row-by-row basis instead of a transaction-by-transaction basis.

Subscription sets and subscription-set members

Before you can replicate data from the replication source, you must associate the replication source with the target to which you want the changes replicated. You define this information using subscription sets and subscription-set members. The information that you provide is stored in various replication control tables.

A *subscription set* contains the attributes of a replication subscription. When you create a subscription set, you define the following attributes:
- The relationship between the source server and the target server
- The association between the Apply qualifier and the subscription set name
- When to start replication, how often to replicate, and whether to use interval or event timing
- Data blocking, if you expect large volumes of changes

A subscription set must have one *subscription-set member* for each target table or view. When you create a subscription-set member, you define the following attributes:
- The relationship between a source table or view and a target table or view
- The structure of the target table or view
- The columns that you want replicated (subselect columns)
- The rows that you want replicated (SQL predicates)

Subscription sets ensure that all subscription-set members are treated alike during replication: either changes are applied to all targets or to none of them. The changed data for all the subscription-set members in a subscription set is replicated to the specified target tables in a single transaction. Subscription sets optimize performance because the target tables in a set are processed in one transaction against the target server. Subscription sets also preserve referential integrity.

Each subscription set is processed by one Apply program; however, each Apply program can process many subscription sets.

Apply qualifier

The *Apply qualifier* associates an Apply program and one or more subscription sets. The Apply program is also associated with one control server, which contains the control tables for the subscription sets. The control tables are shared by more than one instance of the Apply program. You specify a case-sensitive string as the value for the Apply qualifier when you define a subscription set.[5]

By using more than one Apply qualifier, you can run more than one instance of the Apply program from a single user ID. The Apply qualifier is used to identify records at the control server that define the work load of an instance of the Apply program; whereas the user ID is for authorization purposes only. For example, assume that you want to replicate data from two source databases to the target tables on your computer. The data in source table A is replicated using full-refresh copying to target table A, and the data in source table B is replicated using differential-refresh copying to target table B. You define two subscription sets (one for table A and one for table B), and you use separate Apply qualifiers to allow two instances of the Apply program to copy the data at different times.

Data manipulation

You might want to replicate only a subset of your source table, use a simple view to restructure the data from the source table to the target table, or use more complex joins and unions. You might also want to write applications to manipulate staged changes before they replicate, such as including some columns from file records.

Subsets of source tables

You can replicate certain columns or rows from the source table instead of replicating the whole source table. This process, which is sometimes known as *table partitioning*, is called *column subsetting* and *row subsetting* in this book.

Use column subsetting if you want to replicate only a subset of all of the columns from the source. This type of subsetting is appropriate, for example, if some of the columns in the source are very large, such as large objects (LOBs), or if the column data types are not supported by the intended target table.

Use row subsetting if you want to replicate only some of the rows from the source database. For example, when you are replicating data to more than one regional office, you might want to replicate only records that are relevant to that particular regional office. To subset rows, use the WHERE clause when defining the subscription-set member.

5. The Apply qualifier appears in many control tables; therefore, do not attempt to change its value after you set it.

Views as sources

Simple views are useful in data warehouse scenarios if you want to restructure copies so that data in target tables is easily queried.

Joins and unions for targets

You can create and maintain target tables with contents that are joins or unions of existing source tables.

You can use the following types of joins:

- Simple inner-joins over one or more defined replication sources, possibly in combination with other tables or views that are not themselves replication sources.
- Simple inner-joins over CCD tables that are defined as replication sources. These CCD tables can be maintained by the Apply program or they can be maintained by another application for external data sources (such as DataPropagator NonRelational).

Specifically, you can use joins and unions to manipulate data in the following ways:

- Joins of tables from a single DB2 source server (by defining a DB2 view as the join of certain tables)
- Unions of tables from one source server (by using multiple subscription-set members in a set where each member has the same target table)
- Unions of tables from multiple source servers, sometimes referred to as *multisite unions* (by creating multiple subscription-set members in multiple subscription sets because there are multiple source servers)

Target tables

When you define a subscription-set member, you must specify the type of target table that you want to use. The following types of tables are available:

- User copy tables
- Point-in-time tables
- Aggregate tables
- Consistent-change-data (CCD) tables
- Replica or row-replica tables
- User tables

The following sections describe the unique characteristics of each type of target table.

User copy tables

These tables are read-only copies of the replication source with no replication control columns added. They look like regular source tables and are a good starting point for replication. They are the most common type of target table.

Point-in-time tables

These tables are read-only copies of the replication source with a timestamp column added. The timestamp column is originally null. When changes are replicated, values are added to indicate the time when updates are made. Use these types of tables if you want to keep track of the time of changes.

Aggregate tables

These are read-only tables that use SQL column functions (such as SUM and AVG) to compute summaries of the entire contents of the source tables or of the recent changes made to the source table data. Rows are appended to aggregate tables over time. There are two kinds of aggregate tables: base aggregate tables and change aggregate tables.

Base aggregate tables summarize the contents of a source table. Use these types of tables for tracking the state of a source table on a regular basis. For example, assume that you want to know the average sales for a store on a day-by-day basis. If your source table has a row for each sales person, you could average the daily sales for everyone from the store in a base aggregate table. If the average sales for employees on Friday was $1 000, and the store was closed on Saturday and Sunday so that the average sales on each of those two days was $0, Table 1 shows the entries that would be made to a base aggregate table:

Table 1. Hypothetical entries in a base aggregate table

Friday	Saturday	Sunday
1 000	1 000	1 000

Change aggregate tables work with the change data in the control tables, not with the contents of the source table. Use these types of tables for tracking the changes made between each Apply program cycle. By setting a filter you can track recent UPDATE, INSERT, and DELETE operations collectively or individually. For example, assume that you want to know day-to-day changes for average sales for a branch. You could average the changes and store them in a change aggregate table. Therefore, if you use the average daily sales from the previous example ($1 000 on Friday, $0 on Saturday and Sunday), and store the averages of the changes in the change aggregate table, you would get the entries shown in Table 2 on page 16.

Table 2. Hypothetical entries in a change aggregate table

Friday	Saturday	Sunday
1 000	0	0

Notice that the entries in the change aggregate table differ from those in the base aggregate table, even though the source data is the same.

Consistent-change-data (CCD) tables

These read-only tables contain data from committed transactions. CCD tables contain different data if they are condensed, noncondensed, complete, or noncomplete. A *condensed CCD table* contains only the most current value for a row, while a *noncondensed CCD table* contains a history of changes to the row. The Apply program appends rows to noncondensed tables; while it only updates rows that are already in condensed tables. A *complete CCD table* contains all the rows that you want to replicate from the source table; whereas a *noncomplete CCD table* is empty when it is created and has rows appended as changes are made to the source table. Use the type of table that meets your needs:

- Condensed CCD tables are useful for staging changes to remote locations and for summarizing hot-spot updates before they are replicated to targets. *Hot-spot updates* are updates made repeatedly to the same rows over a short period of time. Condensing ensures that only the net change for a row is replicated to the target; therefore, it reduces the load on your network and prevents a hot-spot problem from occurring on the target. Use *condensed, complete CCD tables* if you want to initialize and maintain remote targets without accessing the original source each time that it is updated. Use *condensed, noncomplete CCD tables* to maintain remote targets by accessing the original source each time that it is updated.
- Noncondensed CCD tables are used for auditing purposes. To collect audit information from only the time that you start collecting change data, use *noncondensed, noncomplete CCD tables*. These tables are initially empty, and information is appended by each UPDATE, INSERT, and DELETE operation. To collect complete audit information, use *noncondensed, complete CCD tables*. These tables start out as full copies of the source tables and are appended by each UPDATE, INSERT, and DELETE operation. They retain all of the information about the source tables and can be used if you need to make historical queries (for example, as of last Tuesday, a month ago, or yesterday).

Also, use consistent-change-data (CCD) tables if you want your change data maintained by these other change-capture mechanisms, instead of the Capture program:

- DataPropagator NonRelational for change data from IMS

- Data Difference utility for change data from VSAM
- Capture triggers for change data from non-IBM data sources

Replica or row-replica tables

These are the only target tables that your applications can update directly. Changes made to replicas and row-replicas are replicated to the associated source table; the source table in turn replicates the changes to other replicas. Replicas are supported only in DB2 databases. A *row-replica table* is a special type of replica table for DB2 DataPropagator for Microsoft Jet. Use the replica table types for update-anywhere replication.

User tables

You don't actually specify a *user table* as a target; however, in update-anywhere replication, a user table is automatically a target for the replicas or row-replicas that are associated with it. The user table is the *parent replica*, and its copies are *dependent replicas*. The parent replica receives updates from a dependent replica and, if there are no conflicts detected, it replicates the changes to the other dependent replicas. The parent replica is the primary source of data. If there are any update conflicts detected, the contents of the parent replica prevail. Typically your applications access the dependent replica tables; however, they connect to the server containing the user table when the replicas are not available.

Schedule for applying updates

Synchronous replication delivers updates continually. When a change is made to the source data, it is immediately applied to the target database. A change is committed to the source database only after the change is replicated to the target database. If for some reason the change cannot be replicated to the target database, the change is not made to the source database. This type of replication is also called *real-time replication*. If your application requires synchronous updates, code your applications to update tables in a single, distributed transaction instead of using the products described in this book.

Asynchronous replication delivers updates in stages. When a change is made to the source data, it is stored temporarily for a preset interval and forwarded to the target at a later time. The interval can be a measure of time (seconds, minutes, hours) or can represent a prescribed event (midnight, or some other time of day). If changes cannot be made to a target database (for example, if the target database is down or the network is down), they are stored and applied later, in the order in which they were made to the source. This type of replication provides many benefits over synchronous replication: better use of network resources, less database contention, and the opportunity to enhance data before it reaches the target database.

DB2 DataPropagator performs asynchronous replication; therefore, changes made to the source are not made immediately to the targets. You can control how frequently the changes are applied to the target by specifying time intervals, events, or both. For environments that have occasionally connected clients, you can replicate data on demand.

Interval timing

This is the simplest method of controlling the timing of replication. To use *interval timing*, you choose a date and time for the Apply program to start replicating data to the target, and set a time interval that describes how frequently you want the data replicated. When the Apply program stops, it will not start again until the time interval passes. The time interval can be a period of time (from one minute to one year), or it can be continuous. A *continuous time interval* means that the Apply program starts replication cycles one after the other, with only a few seconds delay in-between (you can control the delay with the start parameter). The intervals that you provide are approximate. The interval actually used by the Apply program depends on the number of updates that the Apply program has to replicate and on the availability of resources (that is, database table, table space).

Event timing

This is the most precise method of controlling the timing of replication. To use *event timing*, you specify the name for an event when you define the subscription set, and you set the time when you want that event processed. Optionally, you can set an end-of-period time; the Apply program will not replicate any transactions committed after this time, but will defer their replication until a future date.

The information that you provide for event timing is stored in the subscription events table. The Apply program searches the subscription events table for the event name and the associated time and end-of-period information.

On-demand timing

For replication configurations that include occasionally connected systems, the Capture and Apply programs do not run continuously. You can replicate data to and from occasionally connected systems on demand by using the ASNCOPY or ASNSAT programs to operate the Capture and Apply components. These programs are described in “Part 4. Occasionally connected environments” on page 237.

Chapter 2. Data replication configurations

This chapter describes the typical data replication configurations and provides examples of replication solutions for common business needs. Some of these configurations show how other products can be used with DB2 DataPropagator to create a unique replication solution. The replication configurations covered here are not exhaustive because customers are continually developing new and creative implementations.

Important: DB2 replication is designed for asynchronous replication and is *not* suitable for the following situations:

- Performing real-time replication: In real-time replication, sometimes called synchronous replication, changes to the source system are made immediately to the target tables. If synchronous data delivery is essential to your application, code your application to update the application table and all of its copies with a single transaction to ensure that both the source and target are changed at the same time.
- Maintaining a backup server (hot-site backup): Do *not* use asynchronous replication to maintain a backup server that can be accessed when your primary server is down. If the source (primary) server becomes unavailable, there is no way to guarantee that all updates were made to the target (backup) server. If you must maintain a backup server, consider using other tools or features. For example, consider the following option for System/390: Peer-to-peer remote copy (PPRC) hardware feature, or extended recovery component (XRC).

Overview of replication configurations

You can combine configurations to suit your business needs. The following sections describe these typical configurations, including some variations of each:

- Data distribution
- Data consolidation
- Update anywhere
- Occasionally connected configurations

Data distribution and data consolidation are easier to set up and maintain than the other configurations.

Data distribution

In *data distribution* configurations, a primary data source resides on a source server (see Figure 1). Changes made to the data source are replicated to one or more target tables that reside anywhere in a distributed network. The target tables are read-only; therefore, you don't need to set up conflict detection because no update conflicts occur during replication. Applications can use the target tables, which are local copies, so that they don't overload the network or central server. This configuration is useful if you need to share data among several sites but you don't want to reduce the performance of your applications.

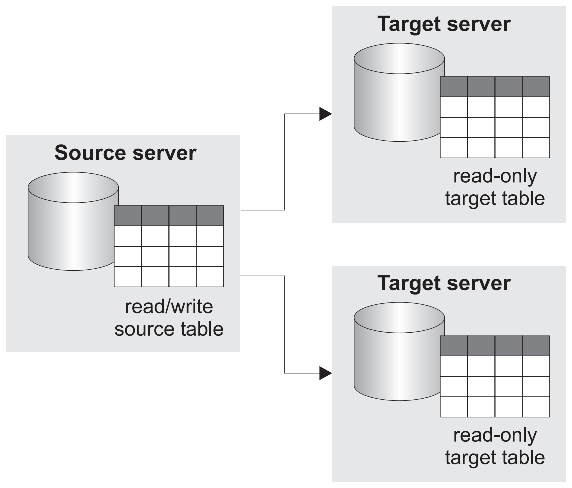


Figure 1. Data distribution. Changes made to a source table are replicated to read-only target tables.

Data consolidation

In *data consolidation* configurations, a central data server is used as a repository for data from many data sources (see Figure 2 on page 21). Therefore, this configuration consists of many source tables or views and one target table with multiple subset views. Changes made to each data source are replicated to the central data server, which is read-only.

Restriction: If you consolidate data from more than one server into a CCD target table, you must *not* use that CCD target table as a replication source for other target tables. The original servers use separate log sequences that cannot be distinguished in further replication.

Data consolidation configurations are useful for maintaining a local decision support system (DSS) so that you can analyze data without competing for production database resources. To ensure that there are no update conflicts, you must design the replication environment so that there is only one source for each data item. If each source updates a unique set of rows, you will never encounter update conflicts.

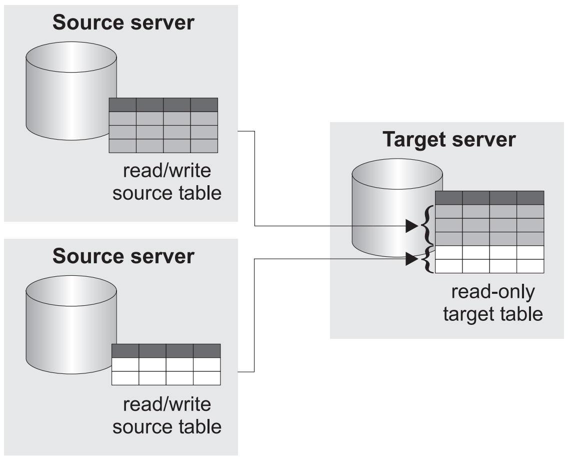


Figure 2. Data consolidation. Each source table can update a unique set of rows in a read-only target table.

Update anywhere

In *update-anywhere* configurations, a replication source has target tables that are read / write copies. Changes made to a target table are applied to the source table, which maintains the most up-to-date data. The changes to the source table are then applied to all of its target tables. Because updates can originate from either the source or target table, an update conflict might occur when the data is replicated. If you want to use this type of replication configuration, you must decide how to handle conflicts in your environment. Your options range from ignoring conflicts and rejecting any conflicting updates, to designing your application so that a conflict can never occur. By rejecting conflicting updates, you risk losing some information. If possible, try to design your system to prevent all possible conflicts (see Figure 3 on page 22). Figure 4 on page 22 shows a configuration that requires conflict detection.

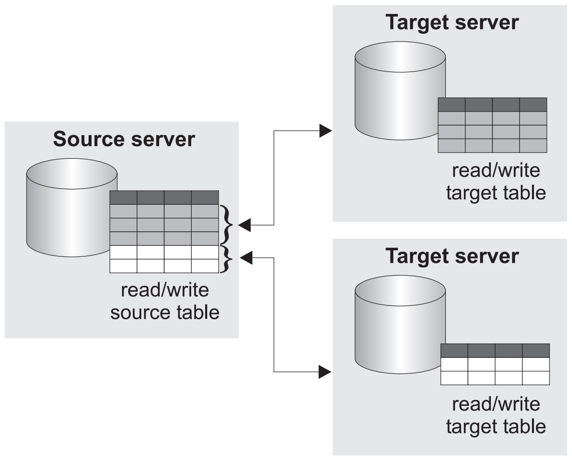


Figure 3. Update-anywhere replication without the risk of conflicts. Each read/write target table has a unique set of rows that can be updated locally; the source table at the source server maintains the most up-to-date data. This configuration does not require conflict detection.

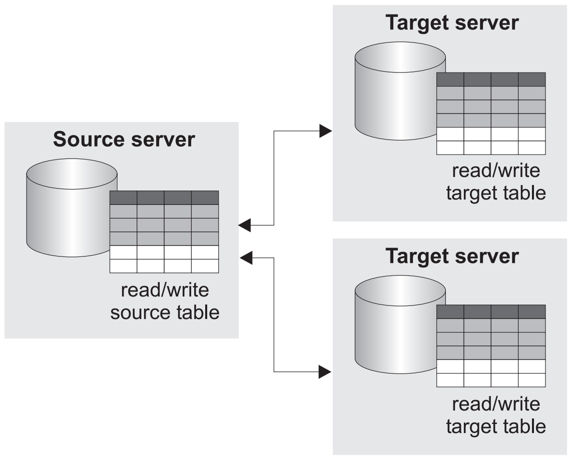


Figure 4. Update-anywhere replication with risk of conflicts. This configuration requires conflict detection because all rows can be updated at the source table or at any target table.

Occasionally connected

In *occasionally connected* configurations, you have the flexibility to connect to and transfer data to and from a primary source on demand. These types of configurations allow users to connect to the primary data source only long enough to synchronize their local database, and they don't require a continuous connection for replication administration (see Figure 5 on page 23). DB2 Universal Database Satellite Edition and DB2 DataPropagator for

Microsoft Jet support centralized administration of occasionally connected systems, but the Mobile client for DB2 replication does not.

Occasionally connected configurations are well-suited for synchronizing data on laptop computers or at computers in home offices because these configurations minimize the frequency and duration of communication-line connections, and reduce telecommunication costs. This type of configuration also works well for replicating data to onsite computers that are not constantly connected to the network (for example, if employees are in the office only 3 days a week).

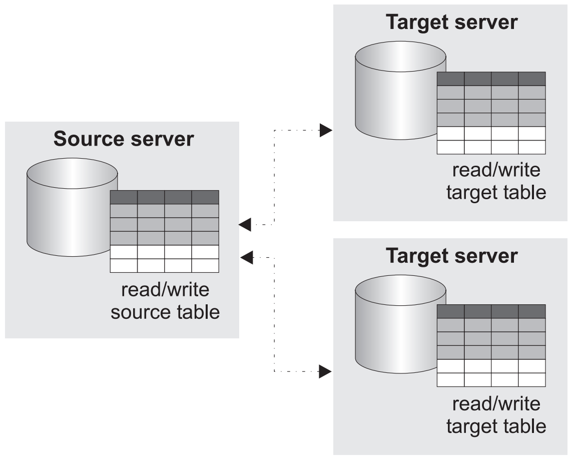


Figure 5. Occasionally connected configuration. The target servers are not continuously connected to the source server. Changes that are made to the tables are replicated when the target server is connected to the source server.

Examples of replication configurations

You can build on the typical replication configurations to come up with replication models that meet your specific needs. This section discusses examples of some common business needs and the DB2 replication solution that addresses those needs. Design issues that are unique to each replication solution are described.

Archiving audit information

Requirements: A customer in a DB2-IMS Transaction Manager (TM) environment generates audit data by writing audit information to the IMS log. New applications access DB2 through DRDA, bypassing IMS TM completely. The customer needs to track all changes to relational tables for auditing purposes to determine which users made particular changes to the data.

Replication solution: The Capture and Apply programs for DB2 DataPropagator are used to capture and store the DB2 for OS/390 changes in target tables (see Figure 6).

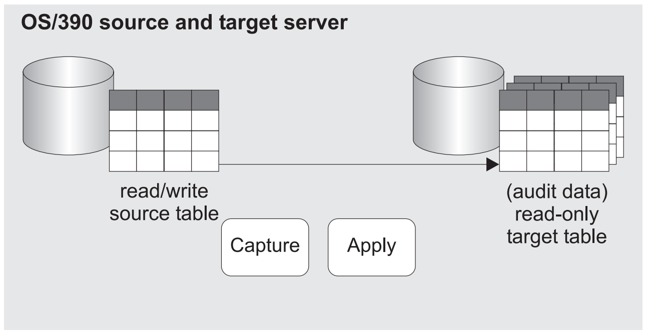

OS/390 source and target server
read/write source table
Capture
Apply
(audit data) read-only target table

Figure 6. Audit information. Audit data is replicated to a target table that can be read by the customer's application.

Design highlights: Both the before-image and the after-image values of each row are captured and stored. The authorization ID of the person who changed the data is also stored in the audit tables. All of this information is captured from the DB2 for OS/390 log.

Consolidating data from distributed databases

Requirements: A large retail chain has almost 500 stores around the country, each of which gathers purchase details through an electronic point of sale (EPOS) system. Each store keeps its data in local databases on DB2 for AIX. They transfer the data nightly to a central DB2 for OS/390 site using a pre-existing file-transfer process from the EPOS terminals. The company wants to enhance the data at the central site and then distribute it to each of the stores and to some regional offices.

Replication solution: Data changes at each retail store are captured and saved by the Capture program on DB2 for AIX (see Figure 7 on page 25). The Apply program on DB2 for OS/390 consolidates the data from all stores, summarizes the data, and distributes the summarized, consolidated data to regional offices and stores. Employees at each store can see the local trends and their performance against that of other outlets in the region. Regional managers are able to plan their marketing and distribution strategies by using the information to help set objectives for the individual stores.

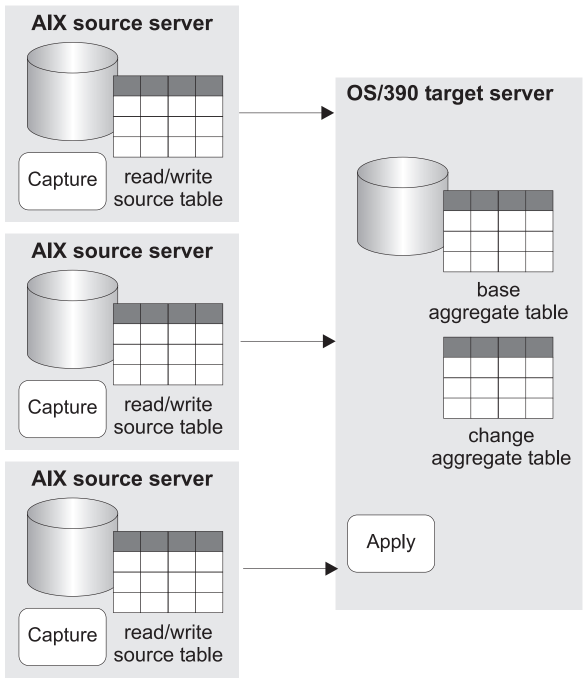


Figure 7. Consolidating data from distributed databases. Data from three source servers is replicated to two target tables on a target server.

Design highlights: The Apply program uses base aggregate and change aggregate tables to summarize the consolidated store data. The base aggregate tables summarize the contents of the source files. The change aggregate tables summarize the results of the changes made between each target refresh that is performed by the Apply program.

Distributing data to remote sites

Requirements: A small bank installed several new Windows NT client/server applications in its 85 branches. A major source of data for the new applications is the customer and financial reference data, which is derived and held at a host site in two operational systems, one on DB2 for OS/390 and the other on DB2 for AIX. If branches accessed the data directly from the host site, network traffic would be congested and the availability of the production data could be affected.

Replication solution: To minimize the network traffic, a local copy of the database is maintained at each branch (see Figure 8 on page 26). Therefore, each branch is a target server. Changes are captured from DB2 for OS/390 and DB2 for AIX, condensed in control tables on DB2 for AIX, and replicated to the branches overnight.

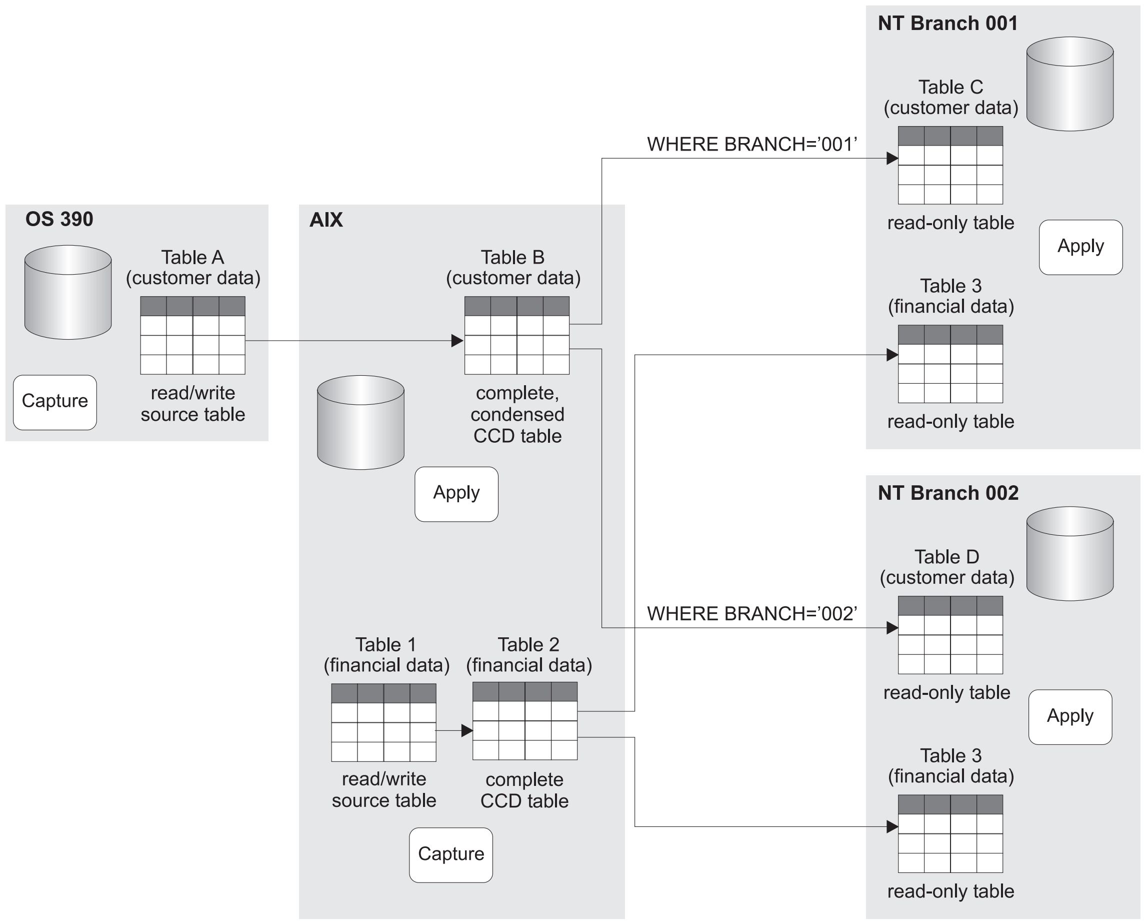


Figure 8. Distributing data to remote sites. Source data is consolidated on an AIX server and replicated to the branches. Each branch gets all of the financial data and some of the customer data. WHERE clauses are used to ensure that each branch gets the records that pertain to their own customers only.

Design highlights: One Apply program resides on AIX and replicates from DB2 for OS/390 and DB2 for AIX. There is one subscription set for replicating from DB2 for OS/390 to DB2 for AIX and one for replicating from DB2 for AIX to DB2 for AIX.

An Apply program also resides on the target servers at each branch. The Apply program on the source server runs separately from the Apply programs at the target servers. The Apply program at each of the branches replicates from the control tables on DB2 for AIX at the host site. Each of the Apply programs on the target servers has a subscription set for replicating from the host site to its local database. Each branch gets all of the financial data but only some of the customer data. WHERE clauses are used to ensure that each branch gets the records that pertain to their own customers only.

The Capture and Apply programs maintain complete, condensed CCD tables in DB2 for AIX. The administrator chose a condensed CCD table because that type of staging table contains only the most recent change made to a row, so network traffic is reduced during replication.

When the subscription sets were created for each branch, the administrator put the control server on the Windows NT server. If the administrator had put the control server on DB2 for AIX, the Apply program from each Windows NT server would need to connect to the host site over the network to read and update the control information about the subscription set, and to detect changes to its control information.

Distributing IMS data to remote sites

Requirements: A large financial institution wants to improve the flow of information from two legacy operational systems to its OS/2-based branches. It wants to provide more accurate and timely data to help loan-application research and to detect credit-card fraud. The data for loan applications is in DB2 for OS/390, and the credit card details are in an IMS system. Previous attempts to copy the legacy data consisted of an unworkable mixture of ad-hoc reports and file transfer techniques.

Replication solution: DataPropagator NonRelational is used to capture and save the changes to IMS data into CCD tables in DB2 for OS/390 (see Figure 9 on page 28). The DB2 DataPropagator Capture program is used to capture and save the changes to DB2 for OS/390 data. The data that is saved is historical—it records every change made. The Apply program runs at the branches and uses the historical data from IMS and DB2 for OS/390 to maintain DB2 for OS/2 tables.

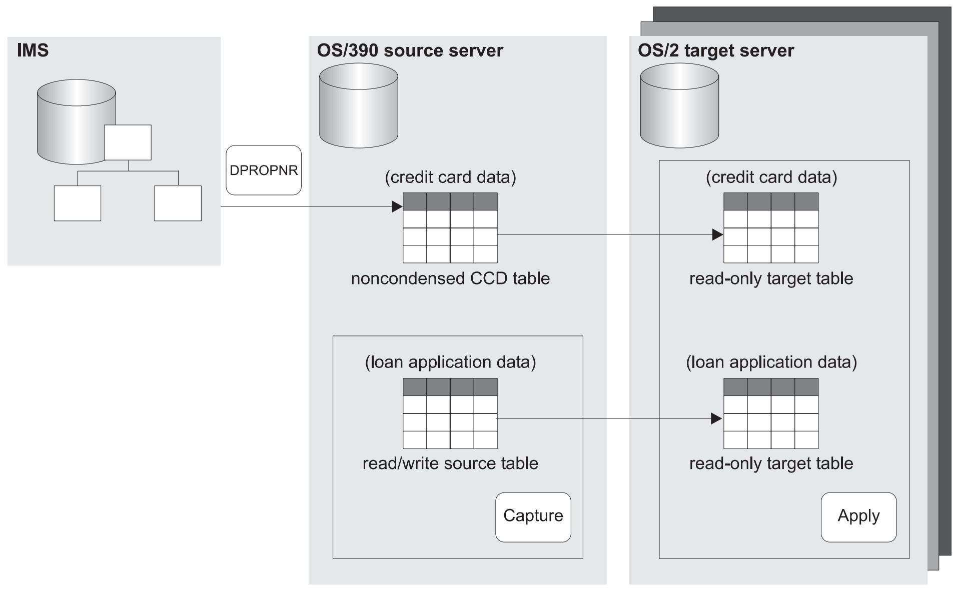


Figure 9. Distributing IMS data to relational databases. DataPropagator NonRelational replicates IMS data to target tables on an OS/390 source server. DB2 DataPropagator captures data from the OS/390 source server and replicates it to OS/2 servers.

Design highlights: DataPropagator NonRelational captures changes from the IMS log and creates a noncondensed CCD table in DB2 DataPropagator format on the OS/390 source server. DB2 DataPropagator uses this CCD table as a replication source. The Capture program on the OS/390 server captures information from the local tables that contain the credit-card and loan-application data. The Apply program on the OS/2 target server pulls the change data to the target tables.

Accessing data continuously

Requirements: An international bank wants to keep its system online 24 hours a day. Currently the system is online 23 hours 45 minutes a day. Every day the bank stops the system to quiesce it for a batch application, which requires exactly one day’s worth of data. During the 15 minutes when the system is down, the required tables are extracted. After the extraction, the system is made available for the next financial day.

Replication solution: Data changes made during the day are captured and replicated to CCD tables (see Figure 10 on page 29). The batch application was modified to process the changes in the CCD tables instead of the table extracts. The online system does not need to be stopped to provide consistent data for the batch application.

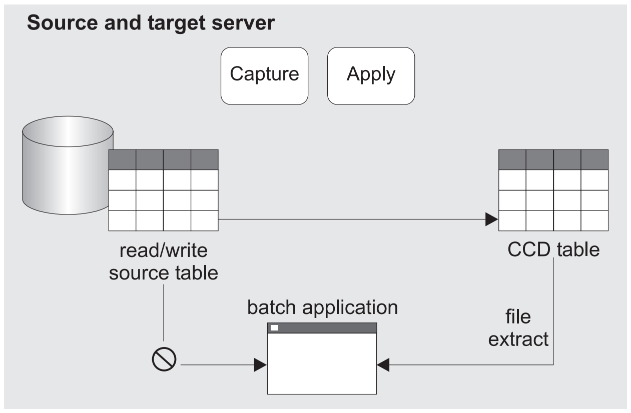


Figure 10. Batch application using replicated data. The source data is replicated to a CCD table. The batch application extracts data from the CCD table when the source table is unavailable.

Design highlights: The CCD table includes a timestamp that is used to identify the changes made during a time period (in this case, one day).

Replicating operational data to decision support systems

Requirements: A financial institution needs to replicate updates from its customer information database on DB2 for AS/400 to a decision support system that is also on DB2 for AS/400. Historical data about updates must be saved and stored with no code changes to production applications and no impact to the performance of those applications.

Replication solution: Updates are captured from the key operational tables and, on an hourly basis, replicated to CCD tables in the decision support system (see Figure 11).

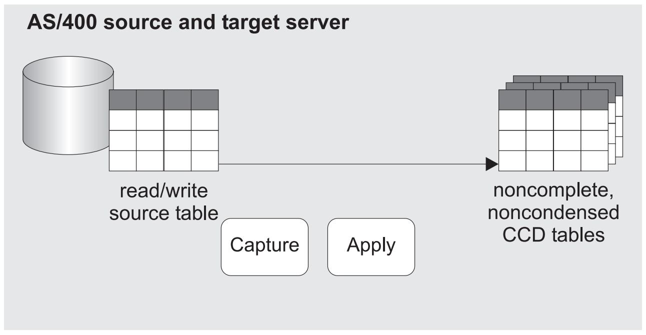

AS/400 source and target server
read/write source table
Capture
Apply
noncomplete, noncondensed CCD tables

Figure 11. Replicating operational data to decision support systems. The CCD target table is used to record all changes made to the source database.

Design highlights: The Capture and Apply programs maintain noncomplete, noncondensed CCD tables. Noncondensed CCD tables are used because they record all changes that are made to the customer information database. Furthermore, noncomplete CCD tables are used because the financial institution does not want to record the original contents of the source, it wants only the changes.

The Capture and Apply programs are given job priorities such that replication does not impact production CPU resources. The decision support system could be implemented just as easily on any of the supported target platforms and could still be ported to other platforms, if required.

Using target tables as sources of updates (update anywhere)

Requirements: A financial institution has hundreds of agents at several branches who must fill in online forms to set up and modify client accounts. The agents base the quotation rates on information that was generated at the head office and sent to the branch. The agents send reports back to the head office, and the accounts are finalized only after the information is verified at the head office. The agents would be more productive if they had access to up-to-date data, without the network problems of accessing the central database directly.

Replication solution: A special type of target table, called a *replica*, is used to set up circular subscriptions (see Figure 12 on page 31). Changes to the replica are replicated back to the primary replication source, which is a *user table*. An update that is made at one location is reflected in the databases at other locations. Agents have the current information that they need to finalize accounts while meeting with the client, and the head office has the new business data generated that day.

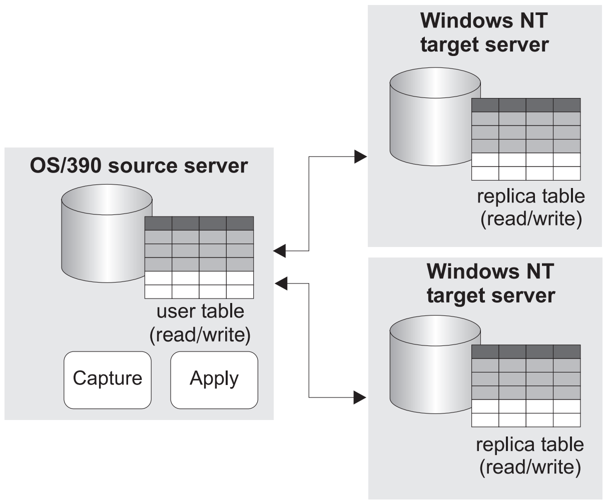


Figure 12. Update-anywhere replication example. The primary data source, or parent replica, is on an OS/390 server; and the dependent replicas are on Windows NT client systems.

Design highlights: The primary replication source is a user table. It contains the most up-to-date information.

This type of replication works best when transaction conflicts between the central database and the updatable copies can be avoided, such as when copies can update only key ranges at specific sites, or when sites can make updates only during certain time periods.

DB2 DataPropagator detects conflicts that occur when the same row is updated on the host system and on an agent's system and neither change has been replicated. If an agent made updates that are in conflict, these updates are discarded during replication to ensure data integrity. The transaction containing the conflict and all captured transactions that are found that are dependent on the conflicting transaction are backed out.

Updating data on occasionally connected systems

Requirements: An insurance company wants to equip its sales agents, who rarely visit the company's home office, with a set of offers to attract both new and existing customers—special introductory offers and personalized packages. Much of the time, the agents' computers will not be connected to the home office. When they connect to the home office, they need to get any updated information from the central database. Managing the potential backlog of changes can be an issue.

Replication solution: The sales force is supplied with laptop computers running DB2 Universal Database Satellite Edition. As a sales campaign is launched, each agent downloads the customer profiles and history, as well as the latest product offers. DB2 replication also solves the problem of keeping the information up to date. Only new and changed data rows are copied across the network.

Design highlights: DB2 Universal Database Satellite Edition is used because it meets the replication requirements and can be administered by a central administrator. An administrator at the home office sets up the replication environment, tests it, and copies it to the occasionally connected systems. The administrator also provides user IDs and passwords to the agents in the field so that they can connect to the server at the home office from their laptop computers. While they are logged on, the agents can synchronize the information on their laptop computer with the information at the source server by simply pressing a button.

Chapter 3. Data replication scenario

Use the sample scenario in this chapter to get some experience using the DB2 Control Center and the Capture and Apply programs. Follow the steps in this simple scenario to copy changes from a DB2 replication source to a target table in a DB2 database for Windows NT.

The scenario consists of the following parts:

1. “Before you begin”
2. “About this scenario” on page 34
3. “Setting up the scenario replication environment” on page 36
4. “Operating in a replication environment” on page 45

Before you begin

If you want to work through this scenario on your computer, set up your system using these steps:

1. Make sure that you have DB2 for Windows NT installed on your computer.
2. Make sure that your DB2 Control Center uses the default settings. If you have explicitly changed the default settings, some of the steps described in this scenario will not match what you see on your display.
3. Create the `C:\scripts` directory in which you will store the SQL files for replication.
4. Use the DB2 Control Center to create a new database called COPYDB, which you will use as the target and control server. To create the database, right-click the **Database** folder and follow the instructions for creating a new database with default options.
5. Use the **First Steps** icon in DB2 Universal Database (or select **Start -> Programs -> DB2 for Windows NT -> First Steps**) to create the SAMPLE database. After the database is created, close the First Steps window.

The steps in this chapter use the data in the DEPARTMENT table from the SAMPLE database. The fully qualified name is *userID*.Department; where *userID* is the user ID that created the table. Table 3 on page 34 shows the DEPARTMENT table.

Table 3. DEPARTMENT table

DEPTNO	DEPTNAME	MGRNO	ADMRDEPT	LOCATION
A00	Spiffy Computer Service	000010	A00	-
B01	Planning	000020	A00	-
C01	Information Center	000030	A00	-
D01	Development Center	-	A00	-
D11	Manufacturing Systems	000060	D01	-
D21	Administration Systems	000070	D01	-
E01	Support Services	000050	A00	-
E11	Operations	000090	E01	-
E21	Software Support	000100	E01	-

For the remainder of this exercise, use the user ID with which you created the SAMPLE and COPYDB databases. Because you created the databases, you have the authority (DBADM or SYSADM) to perform replication tasks.

About this scenario

Assume that an application that generates reports needs information that exists in the DEPARTMENT table of the SAMPLE database. Instead of using the data directly from the source table, you want to copy the changes to a target table that can be read only by the report-generating application. For ease of administration, you want to keep the target table on the same machine as the source server.

You require a simple data distribution configuration, with changes from one replication source being replicated to a single read-only copy. This section describes the design and planning issues that you need to consider before you perform any replication tasks.

Replication source

You already know that the replication source is the *userID*.DEPARTMENT table in the SAMPLE database. Before you set up your environment, you must decide what you want to replicate from that table. You decide to make all columns available for replication; and you want to save before-image values for each of them.

Tip: You might want to always include before-image values when you define a replication source. Therefore, if you later change to an update-anywhere configuration, you won't have to redefine your replication source.

Replication target

You decide that you want your replication target to be the COPYDB database, which you created using DB2 for Windows NT earlier in this chapter. Currently there is no target table in that database; you want the Control Center to create the target table according to your specifications.

Using existing target tables: When you use the Control Center, the target table is created if it doesn't exist. That is the preferred method of generating a target table because it ensures correct mapping to the replication source. You can use existing target tables if they were created by any DB2 product.

Assume that you want the target table in COPYDB to contain the following columns of information:

DEPTNO
Information from the DEPTNO column in the replication source (this column will be the primary key of the target table)

DEPTNAME
Information from the DEPTNAME column in the replication source

MGRNO
Information from the MGRNO column in the replication source

ADMRDEPT
Information from the ADMRDEPT column in the replication source

LOCATION
Information from the LOCATION column in the replication source

Because the columns in the target table simply reflect the data from the source table, and there is to be only one record in the target table for each record in the source table, you can use a *user copy* type of target table.

Replication options

For the purpose of this exercise, you decide to store the target table and the replication control tables in the default table space, USERSPACE1.

Logical server	Table space	Contents
Source server: *userid*.SAMPLE	USERSPACE1	Source replication control tables, including the CD table
Control server: *userid*.COPYDB	USERSPACE1	Replication control tables and the target table

Typically you will want to put the UOW table and the CD tables (and CCD tables if you are using them) in their own table spaces, with table or table space locking. You can put all other replication control tables together in one table space with row-level locking.

For scheduling replication, assume that you want DB2 replication to check for any changes from the source table every minute and replicate them to the target table. Although a report-generating application doesn't require that kind of turnaround, you want to test the replication environment that you set up to make sure that everything is working correctly.

Also, after each replication cycle, you want to delete any records from the Apply audit trail table that are older than one week (seven days). This pruning will prevent the table from growing too large.

You won't need to set constraints because you have a read-only target. Constraints are needed only when applications are updating a target table. In this scenario, the updates are committed at the replication source, and they must satisfy the constraints defined on that system. There is no reason for you to re-evaluate the same constraints at the target.

Setting up the scenario replication environment

After planning the replication model, you are ready to set up the replication environment.

Step 1: Customize control tables

The Control Center automatically creates control tables at the source server and at the target server. By default, it builds the control tables with default settings (table space, locking) that are suitable for testing purposes but not for production environments. To customize the control tables for your production environment, you must edit the dpcntl.udb file *before* you perform any other replication task.

To customize control tables:

1. Go to the sqllib\samples\repl\ directory.

2. Open the dpcntl.udb file. If you were in your production environment, you would edit this file to customize the control tables for your needs. For the purpose of this exercise, do not edit this file.
3. Close the dpcntl.udb file.

Step 2: Define a replication source

After you customize the control tables, go to the Control Center to define the DEPARTMENT table as a replication source.

To define a replication source:

1. In the object tree, click the **Tables** folder under the SAMPLE database. All of the tables that exist in SAMPLE appear in the contents pane.
2. Right-click the DEPARTMENT table and select **Define as Replication Source -> Custom**. A custom replication is one that lets you manipulate the data before it is applied to the source. The Define as Replication Source window opens.
3. From the Define as Replication Source window, specify that you want to use standard conflict detection. By default, all columns are available for replication with their before-image values so you don't need to change anything else in this window for this exercise. Click **OK**.
4. You have the option to run SQL now or later. Save the SQL to a file, as though you wanted to change the name of the CD table. Use these steps from the Run Now or Save SQL window:
   a. Accept the default, which is to save the SQL to a file and run it later, by clicking **OK**. The Save SQL file window opens.

      Tip: Most of the time you will want to use the default. By saving the SQL to a file, you can look at the SQL to understand what it will do, make any modifications that you require, save the file, and run it after you are confident that it will do what you expect it to do.
   b. To create a file in which to save the SQL, in the Save SQL file window, type `replsrc.sql` as the name for the file and `C:\scripts` as the directory in which you want it stored; and then click **OK**.

      Tip: By default, the SQL is saved in the `sqllib\bin` directory. When you work in your own replication environment, you will want to keep all the files in a separate directory instead of storing them with all other SQL executable programs.
   c. View the file that you created. Go to the `C:\scripts` directory and open the `replsrc.sql` file using an editor. For the purpose of this exercise, don't change anything in the file. Close the file.

Tip: When you set up your own replication environment, be careful how you edit this file. If you change the name of the CD table or the table space in which the CD table will be put, you must also modify the CREATE INDEX statement for the CD table.

5. Run the file to define the replication source:
   a. Right-click the **Replication Sources** folder and select **Run SQL files**.
   b. Specify the SQL file that you saved in step 4.b on page 37, `replsrc.sql`, and click **OK**.
6. Verify that DEPARTMENT is defined as a replication source by clicking **Replication Sources** -> **Refresh**. The table name, DEPARTMENT, appears in the contents pane of the Control Center.

The table DEPARTMENT is now defined as a replication source. When you ran the SQL file, the Control Center created the change data (CD) table for this replication source and it created the replication control tables in the default table space (USERSPACE1) for the SAMPLE database.

Step 3: Define a subscription set and a subscription-set member

After you define the source, you need to define a subscription set. A subscription set defines a relationship between the replication source (DEPARTMENT in this scenario) and a target table (that you will call DEPTCOPY in this scenario). It also defines some replication parameters.

To define a subscription set and a subscription-set member:

1. Select the **Replication Sources** object in the object tree, then right-click the **DEPARTMENT** object that appears on the right pane of the Control Center and select **Define subscription**. The Define Subscription window opens.
2. Set up the target table and subscription set:
   a. Name the subscription set that you are about to define by typing `DEPTSUB` in the **Subscription name** field.
   b. Identify the database where the target table will reside by selecting `COPYDB` in the **Target server** field.
   c. Type `DEPTQUAL` in the **Apply qualifier** field. This string identifies the definitions unique to each instance of the Apply program that will run this subscription set.

   Tip: The Apply qualifier is case-sensitive. If you want the Apply qualifier to be in lowercase characters, you must delimit it when you type it; for example `"deptqual"`. If you simply type `deptqual`, the Control Center converts the value to uppercase characters by default.

d. Specify a name for the target table by typing DEPTCOPY over the default name.
e. Specify that you want the Control Center to create the target table by selecting the **Create table** check box for the DEPTCOPY target table.
f. Click **Advanced**. The Target Type page of the Advanced Subscription Definition notebook opens.
g. Because you want to create a user copy type of target table, leave the **User Copy** radio button selected.
h. Configure the columns in the target table by clicking the **Target Columns** tab and making DEPTNO the primary key of the target table. Select the **Primary key** check box next to DEPTNO.

   Tip: You might want to expand the window to view all of the columns. Some rows have names beginning with the letter X (for example, XDEPTNO). These rows store the before-image column values that you requested.
i. Indicate that you want to replicate rows that meet certain criteria by clicking the **Rows** tab and typing the following WHERE clause:
   ```
   DEPTNO >='A00'
   ```
j. Click **OK** to save these settings and return to the Define Subscription window.

3. Define the SQL statements that will be processed when the subscription set is run:
   a. Click **SQL** to open the SQL window.
   b. Click **Add** to open the Add SQL window.
   c. Indicate that you want to delete any records in the Apply audit trail table that are older than seven days by typing the following processing statement in the **SQL statement or Call procedure** field:
      ```
      DELETE FROM ASN.IBMSNAP_APPLYTRAIL WHERE LASTRUN
      < (CURRENT TIMESTAMP - 7 DAYS)
      ```
   d. Indicate that ″row not found″ is an acceptable SQL state by typing the value 02000 in the **SQLSTATE** field and clicking **Add**. This value is added to the **Acceptable SQLSTATE values** list box.

      Tip: You can define all of the SQL states that you want to ignore.
   e. To run the SQL before the subscription set is processed, click the **At the target server before subscription is processed** radio button. In this case, you must run the SQL at the target server because the control server and target server are colocated and the Apply trail table is at the control server.
   f. Click **OK**. The SQL statement is added to the list box in the SQL window and the Add SQL window closes.

   g. Click **OK** in the SQL window to return to the Define Subscription window.
4. Click **Timing** and use the Source to Target page of the Subscription Timing notebook to specify when and how often to replicate the subscription set.
   a. Keep the default values for **Start date**, **Start time**, **Time-based**, and **Using relative timing**.
   b. Specify that you want the subscription set to run in 1-minute intervals:
      1) Use the spin button on the **Minutes** field to select 1-minute intervals (or type 1 in the field).
      2) Use the spin button on the **Hours** field to change the default number to 0 (or type `0` in the field).
   c. Click the **Data Blocking** tab, and use the spin buttons to select 1 as the number of minutes at a time that Apply will copy committed data.

      Tip: The value that you set for data blocking depends on how much free space you have on the workstation that runs the Apply program. Typically, you would use a number from 5 to 20. If you want to be very conservative, use 1 minute.
   d. Click **OK** to save these values, close the Subscription Timing notebook, and return to the Define Subscription window.
5. Submit the subscription set.
   a. Click **OK** in the Define Subscription window. The Run Now or Save SQL window opens.
   b. Specify the control server, which is the database that will contain the subscription set control information, by typing `COPYDB`. This server is the database in which you want to store the subscription control information.
   c. Accept the default option, which is to save the SQL file and run it later, by clicking **OK**. The Save SQL file window opens.
   d. Type the file name, `replsub.sql`, and the directory in which you want to store it, `C:\scripts`; and then click **OK**. The Save SQL file window closes.
6. Run the file to define the subscription set:
   a. Right-click the **Replication Subscriptions** object under the SAMPLE database and select **Run SQL files**.
   b. Specify the SQL file, `replsub.sql`, which you named in step 5.d, and click **OK**.
7. Right-click the **Replication Subscriptions** object under the SAMPLE database and select **Refresh**. The DEPTSUB subscription set appears as an object on the contents pane of the Control Center.

Step 4: Configure the Capture program

Tip: If your source server was on another machine, you would need to log on to the source server over the network. You would use a user ID that has DBADM or SYSADM authority for the source server. However, because the source server for this exercise is on your local machine, you don't need to log on again.

To configure the Capture program:

1. Right-click the **SAMPLE** database object and select **Configure**. The Configure Database — SAMPLE window opens.
2. Go to the Logs page, select the **Retain log files for roll-forward recovery** parameter from the list, and select the **Yes** radio button. By retaining the log, you ensure that DB2 won't overwrite log entries before the Capture program reads them.
3. Click **OK** to save the values.
4. Right-click the **SAMPLE** database object and select **Disconnect**. Click **No** so that you don't need to type your user ID to connect to the database again.
5. Right-click on the **SAMPLE** database object and select **Back-up -> Database**. Follow the instructions in the window to back up to a directory on your system using the default options.

   Tip: You must perform the back-up action to make the database accessible. The database was put in back-up pending mode when you specified that you want to retain log files for roll-forward recovery.

Step 5: Bind the Capture and Apply programs

Tip: For the purpose of this exercise, you will manually create and bind the Capture and Apply program packages. However, DB2 DataPropagator V6 for all supported UNIX, Windows, and OS/2 operating systems can automatically create and bind the packages for you. (Instructions for manually creating and binding the Capture and Apply program packages for each operating system are described in "Part 3. Operations" on page 133.)

To manually bind the Capture program

1. Select **Start -> Programs -> DB2 for Windows NT -> Command Window** to open a DB2 command window.
2. Check that you are still connected to the source server. If you are disconnected, type the following command before going to the next step:
   ```
   DB2 CONNECT TO SAMPLE
   ```
3. Go to `sqllib\bnd`. All the bind files are located in that directory.

4. Create and bind the Capture program package to the source server database by typing the following command:

   ```
   DB2 BIND @CAPTURE.LST ISOLATION UR BLOCKING ALL
   ```

   Tip: Most systems support UR (uncommitted read) format. If your system does not support it, substitute CS (cursor stability format) for UR.

   The CAPTURE.LST file contains a list of the packages created.

To manually bind the Apply program:

1. Check that you are still connected to the source server. If you disconnected after you configured the Capture program, type the following command before going to the next step:

   ```
   DB2 CONNECT TO SAMPLE
   ```
2. Create and bind the Apply program package to the source server by typing both of the following commands:

   ```
   DB2 BIND @APPLYUR.LST ISOLATION UR BLOCKING ALL
   DB2 BIND @APPLYCS.LST ISOLATION CS BLOCKING ALL
   ```

   The APPLYUR.LST and APPLYCS.LST files contain a list of the packages that were created.
3. Connect to the target server by typing this command:

   ```
   DB2 CONNECT TO COPYDB
   ```
4. Create and bind the Apply package to the target server database by typing both of the following commands:

   ```
   DB2 BIND @APPLYUR.LST ISOLATION UR BLOCKING ALL
   DB2 BIND @APPLYCS.LST ISOLATION CS BLOCKING ALL
   ```

   The APPLYUR.LST and APPLYCS.LST files contain a list of the packages that were created.

Step 6: Create a password file

For end-user authentication to occur at the source server, you must create a password file with an AUTH=SERVER scheme. The Apply program uses this file when connecting to the source server. Make sure that the user ID that will run the Apply program can read the password file.

To create a password file:

1. Go to the directory from which you will want to start the Apply program.

   Tip: You must put the password file in the same directory from which you will start the Apply program. If you try to start the Apply program in another directory, you will get an error message.

2. Open a file editing session for a new file.
3. Type the following records in the empty file:

   ```
   SERVER=SAMPLE USER=userid  PWD=password
   SERVER=COPYDB USER=userid  PWD=password
   ```

   Where:

   server
   : The name of the source, target, or control server, exactly as it appears in the subscription set table. (In this example, SAMPLE and COPYDB.)

   userid
   : The user ID that you plan to use to administer that particular server. This value is case-sensitive on Windows NT and UNIX operating systems.

   password
   : The password that is associated with that user ID. This value is case-sensitive on Windows NT and UNIX operating systems.

   Password file format: Do not put blank lines or comment lines in this file. Only add the server-name, user ID, and password information. This information enables you to use different passwords or the same password for each server.
4. Save the file as DEPTQUAL.PWD.

   Password file naming convention:

   The password file name is <*applyqual*>.PWD; where *applyqual* is a case-sensitive string that *must* match the case and value of the Apply qualifier (APPLY_QUAL) in the subscription set table. The file naming convention from the previous release of DB2 DataPropagator is also supported: <*applyqual*><*instance_name*><*control_server*>.PWD; which includes the case-sensitive Apply qualifier, the instance name that the Apply program runs under (the default name is DB2, in uppercase), and the name of the control server in uppercase (for example, COPYDB).

For more information about authentication and security, refer to the *IBM DB2 Administration Guide.*

Step 7: Replicate the scenario data

After defining the replication source and the subscription set, you can submit the copy request by starting the Capture and Apply programs.

To start the Capture program:

1. In a Windows NT window, go to the directory in the source server where you want to store the replication control files associated with the Capture program.

2. Type the following command to start the Capture program using the cold start option, and without automatic pruning:

   ```
   ASNCCP SAMPLE COLD NOPRUNE
   ```

   Tip: Usually you would not specify the cold start option; you would let the Capture program determine whether it should cold start or warm start. For this exercise, you are forcing the Capture program to cold start to clean up the records in the CD and UOW tables.

The Capture program starts running but no command prompt appears. This action creates a *.CCP file. The Capture program is initialized but it does not start capturing changes for the defined replication source until you start the Apply program and it completes its initial full-refresh copy.

To start the Apply program:

1. From another Windows NT window, go to the directory on the target server where you stored the password file.
2. Type the following command to start the Apply program and to call the ASNLOAD program:

   ```
   ASNAPPLY DEPTQUAL COPYDB LOADX
   ```

   Tip: The LOADX invocation parameter calls the ASNLOAD program. The default ASNLOAD program uses the EXPORT utility to export data from the source table and uses the LOAD utility to fully refresh the target table. You can modify ASNLOAD to call any IBM or vendor utility.

The Apply program starts running in the background. You can check the Apply trail table (ASN.IBMSNAP_APPLYTRAIL) for status information.

If you view the DEPTCOPY target table after one replication cycle, you should see results that match the data shown in Table 4.

Table 4. DEPTCOPY table

DEPTNO	DEPTNAME	MGRNO	ADMRDEPT	LOCATION
A00	Spiffy Computer Service	000010	A00	-
B01	Planning	000020	A00	-
C01	Information Center	000030	A00	-
D01	Development Center	-	A00	-

Table 4. DEPTCOPY table (continued)

DEPTNO	DEPTNAME	MGRNO	ADMRDEPT	LOCATION
D11	Manufacturing Systems	000060	D01	-
D21	Administration Systems	000070	D01	-
E01	Support Services	000050	A00	-
E11	Operations	000090	E01	-
E21	Software Support	000100	E01	-

Operating in a replication environment

After the replication environment is up and running, changes that are made to the replication source will be replicated to the target table. You must prune the control tables periodically to prevent them from growing too large. Although the Capture and Apply programs can run continuously, there are times when you will want to stop them (for example, to run utilities that use the table spaces that contain the control tables).

Step 1: Update the source table

Assume that two new departments were created at the Spiffy Computer Service: a technical writing department and a public relations department.

To update the source table:

1. Select **Start** -> **Programs** -> **DB2 for Windows NT** -> **Command Window** to open a DB2 command window.
2. Check that you are connected to the source server. If you are disconnected, type the following command before going to the next step:
   ```
   DB2 CONNECT TO SAMPLE
   ```
3. Add two new rows, one for each department, by typing both of the following commands:
   ```
   DB2 INSERT INTO DEPARTMENT VALUES ('F01','Technical Writing','000110','F01',NULL)
   DB2 INSERT INTO DEPARTMENT VALUES ('G01','Public Relations','000120','G01',NULL)
   ```
4. Connect to the target database by typing the following command:
   ```
   DB2 CONNECT TO COPYDB
   ```
5. Verify that the new rows are replicated to the target database by typing the following command:
   ```
   DB2 SELECT * FROM DEPTCOPY
   ```

Table 5 shows the results of the replication, with two new rows appended to the table.

Table 5. DEPTCOPY table after changes are replicated

DEPTNO	DEPTNAME	MGRNO	ADMRDEPT	LOCATION
A00	Spiffy Computer Service	000010	A00	-
B01	Planning	000020	A00	-
C01	Information Center	000030	A00	-
D01	Development Center	-	A00	-
D21	Administration Systems	000070	D01	-
E01	Support Services	000050	A00	-
E11	Operations	000090	E01	-
E21	Software Support	000100	E01	-
F01	Technical Writing	000110	F01	-
G01	Public Relations	000120	G01	-

Step 2: Prune the control tables

The following steps assume that the Capture program is running. If the Capture program is not running, the **prune** command will not work. Usually pruning should occur during offpeak hours.

To verify that there is something to prune:

1. From a DB2 command window, connect to the source database by typing the following command:
   ```
   DB2 CONNECT TO SAMPLE
   ```
2. Check that there are some rows in the unit-of-work table by typing the following command from the same window:
   ```
   DB2 SELECT COUNT(*) FROM ASN.IBMSNAP_UOW
   ```
   There should be two rows in the unit-of-work table from the previous replication.

To run the prune command:

1. Type the prune command; and include the name of the source server:
   ```
   ASNCMD SAMPLE PRUNE
   ```
2. To verify that the prune command worked, check that the unit-of-work table is empty by typing the following command:
   ```
   DB2 SELECT COUNT(*) FROM ASN.IBMSNAP_UOW
   ```
   There should be no rows in the table.

Step 3: Stop the Capture and Apply programs

An important part of maintaining your replication environment is regular database maintenance. Sometimes that maintenance will require you to stop the Capture and Apply programs. For example, you must stop the Capture and Apply programs before you run utilities that directly use the table spaces that are used by these programs.

To stop the Capture program:

1. Type the following command from the same DB2 command window:
   ```
   ASNCMD SAMPLE STOP
   ```

To stop the Apply program:

1. Type the following command from the same DB2 command window:
   ```
   ASNASTOP DEPTQUAL
   ```

You can run DB2 utilities on your database now that you stopped the Capture and Apply programs. (Running the utilities is beyond the scope of this exercise.)

Chapter 4. Data replication tasks

This chapter introduces the key replication tasks that you have to perform at various stages in the replication process. The tasks are grouped into these major stages:

1. Planning your replication requirements
2. Setting up your replication environment
3. Operating in your replication environment

After you read this chapter, go to “Part 2. Administration” on page 57 for detailed information about these tasks. Also, see “Part 3. Operations” on page 133 for specific information about using the Capture and Apply programs on particular operating systems.

Planning your replication requirements

An important step in coming up with the appropriate replication environment is determining the characteristics of your application data, who needs to access the data, and how frequently they need to access it.

You can use DB2 data replication to maintain data in more than one location and keep the various copies of it synchronized. You must determine where your source data will be coming from, whether you want all or some of the source information copied, whether you want only changes copied, and how many copies (or targets) you need. You also need to determine where the copies will be located.

Although you cannot update the source tables and target tables synchronously, you can schedule the updates to meet the needs of your applications and your replication environment. The frequency of replication depends on how much lag time is acceptable between the time that the source is updated and the time that the targets are updated. Therefore, you must decide how synchronized the copies must be with the source and with each other before you can come up with a replication model.

After you understand your application data requirements, you can design the replication model that will help you meet those requirements. There are many facts that you need to consider when you design your model. These are some of the more important decisions that you need to make:

The replication configuration
Based on your data needs, you must decide whether you need a

consolidation, distribution, update-anywhere, or occasionally connected configuration. You have the flexibility to design your environment so that it uses one of these configurations or some combination of them.

Where to locate the control tables
You will get slightly better performance if you place the control tables on the same server as the Apply program instead of placing them centrally. You can have your Apply programs share a single control server so that your control information is stored centrally. A central control server is popular because it simplifies the administration of large networks, but it has two drawbacks: the Apply program must access the control information over the network and, if the control server goes down, all of the Apply processes are affected.

The type of target tables to use
The type of target table that you use depends on your replication requirements. Each type is best suited for specific situations. For example, a replica is the only type of target table that you can use for update-anywhere replication; and a row-replica is the only type of target table you can use with DataPropagator for Microsoft Jet.

Whether to use existing target tables
You can let the administration interface create the target table for you or you can use an existing table as a target. If the existing tables are DB2 tables, the data types are supported by the DB2 data replication components. If your replication environment includes non-IBM databases, some of the data types might not map directly to the source tables that you are using.

Which columns to make available for replication
You can choose to capture only the after-image column values or both the before-image column values and the after-image column values. If you will be using the targets for auditing purposes, or if you have replica target tables, you must copy both the after-image and before-image column values.

How to capture SQL operations
You might want to capture all updates as two rows in the CD table: a DELETE of the before-image column values followed by an INSERT of the after-image column values. This includes updates of columns that will be the primary key of the target, columns that will be the partitioning key of the target, or columns that are part of the WHERE clause or predicate of the subscription set. You might need to adjust the size of the CD table to accommodate this increased overhead.

The level of constraints
If you have a read-only table, you do not need to set constraints at the target. You must use referential constraints to enforce referential

integrity *only* if you have target tables that are replica tables. The referential integrity of other types of target tables is ensured if you define your subscription sets appropriately.

Which joins to use
Joins are described in views, which in turn are defined in replication sources. For example, you might use a view to change the name of copied columns, to reference columns from related tables in the WHERE clause in your subscription predicate, to incrementally maintain copies that are inner joins of two or more tables, or to replicate information from one table when an update is made to another table.

When you are ready to plan your replication environment, see “Chapter 5. Planning for replication” on page 59 for detailed planning information.

Setting up your replication environment

After you design the replication model, you must set up your replication environment. These steps are involved in setting up your replication environment:

1. Setting up the system
2. Defining the replication criteria
3. Performing the initial replication

“Chapter 6. Setting up your replication environment” on page 83 contains detailed instructions on setting up your replication environment. The rest of this section introduces the steps involved in setting up your environment.

Setting up the system

To set up the system, you perform the following steps:

1. Migrate from previous releases of DataPropagator products.
2. Grant access to the proper user IDs.

Setting up the replication criteria

To set up the replication criteria, you perform the following steps:

1. Configure the administration tool. For example, if you are using DJRA, you need to associate passwords with databases.
2. Customize and create replication control tables.
3. Customize change data (CD) tables. This step is optional. You can change the default name and table space of your CD tables. If you are using the DB2 Control Center, you must customize your CD tables *before* you define

a replication source. If you are using the DJRA tool, you customize the CD tables when you define the replication source.

4. Define replication sources. This step includes identifying the table or view from which you want data copied and the types of changes that you want captured.
5. Define subscription sets and subscription-set members. This step includes associating the replication source with the target to which you want the changes replicated. You can define subscription sets at any time prior to starting the Apply program.
6. Configure the Capture program. This step includes enabling the source server for logging; it also includes creating and binding the Capture program package to the source server.
7. Configure the Apply program. This step includes creating and binding the Apply program package to the source server; it also includes creating and binding the Apply program to the target server.

Performing the initial replication

Important: When you set up your replication environment, you must start the Capture program and let it initialize fully before you start any Apply programs.

To perform the initial replication, you must perform the following steps in the exact order:

1. Make sure that at least one replication source is defined.
2. Start the Capture program. This step includes specifying invocation parameters (such as NOPRUNE, which prevents automatic pruning of the CD and UOW tables). After the Capture program is fully initialized, it will not capture any changes until the Apply program signals it to do so.
3. If you haven't already done so, define at least one subscription set.
4. Start one or more Apply programs. This step includes specifying invocation parameters (such as LOADX, which calls ASNLOAD—an exit routine to initialize target tables). Each Apply program will perform a full-refresh copy for all subscription-set members and the Capture program will begin capturing changes for the associated replication sources.
5. After the replication process starts, you can stop and start the Capture and Apply programs as often as you want. You can stop one or both of them, in any order, as long as you always restart the Capture program using the WARM or WARMNS option.

Tip: Use the WARMNS option if you want to be able to repair any problems (such as unavailable databases or table spaces) that might prevent a warm start from occurring.

Adding to your replication environment

You probably need to add replication sources and subscription sets to your replication environment from time to time.

To add to your replication environment, you must perform the following steps in the exact order:

1. Define the new replication source.
2. Run the Capture **reinit** command, or stop the Capture program and warm start it.
3. Define the new subscription sets.
4. If an Apply program is already running and it uses the Apply qualifier that is associated with the new subscription set, the Apply program will automatically recognize the new subscription set. Otherwise, you must start a new Apply program using the appropriate Apply qualifier before the Apply program can recognize the new subscription set.

Copying your replication environment

After you define your replication environment on one system (for example, a test system), you can copy the replication environment to another system (for example, a production system). You would use the promote functions to reverse-engineer your tables, replication sources, and subscription sets and to create a script file with the appropriate data definition language (DDL) and data manipulation language (DML). For more information about the promote functions, see “Copying your replication configuration to another system” on page 110 and the online help for the administration interface.

Operating in your replication environment

After your replication environment is up and running, and updates are replicated, you need to perform periodic maintenance tasks. These include the following tasks:

Configuring the pruning of control tables
The UOW and CD tables will grow too large if the contents are not pruned regularly. You can configure your system to prune automatically, or you can prune manually. You control how frequently obsolete information will be removed from these tables. If the tables aren’t pruned often enough, the table space that they’re in will run out of space, which will force the Capture program to stop. If they are pruned too often or during peak times, the pruning interferes with the change capture process. You can use the optimal pruning frequency for your replication environment.

Monitoring important criteria
There are many factors that determine how well your replication environment performs. You can use the Replication Monitor, which is part of DJRA, to generate a report that will help you monitor the activities of the Capture and Apply components, as well as the status of the subscription sets. For example, the report contains historical information to help you determine trends about subscription latencies.

Dealing with data modification conflicts
If you are using update-anywhere replication, and you did not design your configuration to prevent update conflicts, you must handle update conflicts and rejected transactions.

Performing regular database maintenance
If you want your replication environment to run smoothly, you must regularly perform database maintenance tasks. For example, use the RUNSTATS utility against the DB2 catalog tables to collect new statistics for tables and indexes. Also use the RUNSTATS utility once after the CD and UOW tables have sufficient data in them so that the DB2 Optimizer will use indexes on them. Periodically use the REORG utility (or the RGZPFM command in AS/400) for the change data tables, the unit-of-work table, and the target tables. You must also delete rows from the Apply trail table, which contains subscription set statistics and error information.

Coordinating with DB2 utility operations
If you want to run DB2 utilities (such as REORG, RUNSTATS, BIND PACKAGE, and REVOKE) that will use the table spaces that contain the replication control tables, you must stop the Capture and Apply programs before running the utilities.

Changing your replication configuration as your business needs change
You are likely to need to modify your replication environment from time to time. Whether you add a new column to an existing source table, or drop a source table, you will need to modify your replication criteria. Also, you will need to maintain password files.

Troubleshooting
If you find that your replication environment is not performing as you expected, or if you can’t replicate data, you can run the Replication Analyzer. The Replication Analyzer is a tool that is shipped as a sample. You can use the Replication Analyzer to analyze the behavior of the Capture program or the Apply program. It can answer such questions as: ″why is the Capture program not capturing?″ and ″why is the Apply program not applying?″ The Replication Analyzer can help diagnose problems, verify replication setup, and offer suggestions for performance tuning. You can also look in the Apply trail table for

status information about the Apply program, or in the Capture trace table for status information about the Capture program.

For general information about operating in a replication environment, see “Chapter 7. Operating DB2 DataPropagator” on page 117. For information about operating in a particular operating system, see the appropriate chapter in “Part 3. Operations” on page 133.

Part 2. Administration

This part of the book contains the following chapters:

"Chapter 5. Planning for replication" on page 59 explains the information you need to help you design your replication environment.

"Chapter 6. Setting up your replication environment" on page 83 describes the steps for setting up and starting replication.

"Chapter 7. Operating DB2 DataPropagator" on page 117 describes how to operate the Capture and Apply programs generally.

Chapter 5. Planning for replication

This chapter explains the information that you need to help you design your replication environment: capacity planning, storage requirements, network requirements, deciding what to replicate, auditing requirements, staging data, and planning for migration.

Capacity planning

The Capture program does not generally impact other applications and requires a minimum of central processing unit (CPU) or central processing complex (CPC) capacity. For example, you can schedule Capture for OS/390 at a lower priority than the application programs that update the source tables. In this case, the Capture program lags behind when CPU resource is constrained.

The Capture program does use CPU resources when it prunes the CD tables and UOW table, but you can defer this activity to reduce system impact.

The Apply program affects CPU usage depending on the frequency of replication, that is, on the currency requirement of the target database. Because the Apply program reads data from the source server and copies that data to the target server, it uses CPU resources on both systems.

In general, the DB2 Control Center and DJRA do not require much local CPU resources. However, when you generate the SQL for replication sources and subscription-set definitions, DB2 DataPropagator extensively searches the catalogs of the source server. And for large sites, these searches can have a noticeable CPU or database system impact.

Recommendations: Plan replication administration for times when impact to the source and target database systems is minimal. Use filtering to minimize the amount of data returned from the source server.

Storage planning

In addition to the storage required for DB2, replication requires storage for:

Database log and journal data
: The additional data logged to support the replication of data.

Active log file size for Capture for VSE and VM and current receiver size for Capture for AS/400
You need to ensure the data needed for replication remains on the active log, rather than on archived logs.

Target tables and control tables
The replicated user data and control tables (including change data tables).

Spill files
The Apply program requires temporary space to store data. Apply for OS/390 can use memory rather than disk space for the spill files; the Apply program for all other operating-system environments uses disk space for the spill files.

If there is insufficient disk space for the spill files, the Apply program terminates. If you specify that Apply for OS/390 should use memory, but there is not enough memory for the spill files, the Apply program abends; in this case, specify that the Apply program should use disk space and restart it.

All of the sizes given in the following sections are estimates only. To prepare and design a production-ready system, you must also account for such things as failure prevention. For example, the holding period of data (discussed in "Target tables and control tables" on page 61) might need to be increased to account for potential line outage.

If storage estimates seem unreasonably high, reexamine the frequency interval of the Apply program (how often your subscriptions run) and pruning. Trade-offs frequently must be considered between storage usage, capacity for failure tolerance, and CPU overhead.

Database log and journal data

Before you can replicate a table, you must create it (or alter it) with the DATA CAPTURE CHANGES keywords. One of the effects of these keywords is that DB2 logs full-row images for each UPDATE statement. For a replica table (in an update-anywhere scenario), DB2 also logs the before-images for each update to the table. Another increase to the log or journal volume comes from DB2's logging insertions to and deletions from the unit-of-work (UOW) and change data (CD) tables.

Although estimating the increase in the log or journal volume is not easy, in general you will need an additional three times the current log volume for all tables involved in replication.

To make a more accurate estimate, you must have detailed knowledge of the updating application and the replication requirements. For example, if an updating application typically updates 60% of the columns in a table, the replication requirements could cause the log records to grow by more than half compared to a similar table that is not replicated. One of the replication requirements that adds the most to the log is the capturing of before- and after-images (as in the case of update-anywhere replication scenarios). One way to reduce the log volume is to reduce the number of columns defined for the replication source.

In addition to logging for the source database, there is also logging for the target database, where the rows are applied. Because the Apply program does not issue interim checkpoints, you should estimate the maximum amount of data that the Apply program will process in one time interval and adjust the log space (or the space for the current receiver for AS/400) to accommodate that amount of data.

Active log file size for Capture for VSE and VM and current receiver size for Capture for AS/400

For VM and VSE, when the active log is full, DB2 archives its contents. For AS/400, when the current receiver is full, you need to switch to a new one, and optionally save and delete the oldest one. When a system handles a large number of transactions, the Capture program can occasionally lag behind. If the log is too small, some of the log records could be archived before they are captured. Capture for VSE and VM running with DB2 for VSE & VM cannot recover archived log records. [6]

For DB2 for VSE & VM, ensure that your log is large enough to handle at least 24 hours of transaction data. For DB2 for AS/400, ensure that the current receiver is large enough to handle at least 24 hours of data.

Target tables and control tables

The space required for a target table is usually no greater than that of the source table (or tables), but can be much larger if the target table is denormalized or includes before images (in addition to after-images) or history data. The following also affect the space required for a target table: the number of columns replicated, the data type of columns replicated, any row subsets defined for the subscription-set member, and data transformations performed during replication.

6. Capture for OS/390 running with DB2 for OS/390 V4 or higher and DB2 Universal Database V5 or higher can recover archived log records.

The CD tables and the UOW table also affect the disk space required for a target database. The space required for the replication control tables is generally small because each requires only a few rows.

The CD tables grow in size according to the amount of data replicated until the Capture program prunes them. To estimate the space required for the CD tables, first determine how long you want to keep the data before pruning it, then specify how often the Capture program should prune these tables or how often you issue the **prune** command. To determine the minimum size for the CD table, use the following formula:

```
minimum_CD_size =
  ( (21 bytes) + sum(length of all registered columns) ) *
  (number of inserts, updates, and deletes to source table) *
  (exception factor)
```

When calculating the number of bytes of data replicated, you need to include 21 bytes for overhead data added to the CD tables by the Capture program. In the formula, determine the number of inserts, updates, and deletes to the source table within the interval between capturing and pruning of data. The exception factor allows for such things as network failures or other failures that prevent the Apply program from replicating data. Use a value of 2 initially, then refine the value based on the performance of your replication environment.

Example: If the Capture program prunes applied rows from the CD table once daily, your interval is 24 hours. If the rows in the CD table are 100 bytes long (plus the 21 bytes for overhead), and 100,000 updates are applied during a 24-hour period, the storage required for the CD table is about 12 MB.

The UOW table grows and shrinks based on the number of rows inserted in a particular time interval (the number of commits issued within that interval by transactions that update source tables or by Capture for AS/400). You should initially overestimate the size required and monitor the space actually used to determine if any space can be recovered. The size of each row in the UOW table is fixed at 79 bytes (except for DB2 for AS/400, where it is 109 bytes). For a first approximation of the space needed for the UOW table, multiply 79 bytes (or 109 bytes) by the number of updates applied during a 2-hour period. Use a formula similar to the one given above for CD tables to obtain a better estimate for the space needed for the UOW table. For more information, see “Unit-of-work table” on page 324.

Spill files

The Apply program stores updates to target tables in temporary files called spill files. [7] These files hold the updates until the Apply program applies them to the target tables. The Apply program uses multiple spill files for subscription sets with multiple subscription-set members: one spill file for each target table. The Apply program stores the spill file on disk for every operating-system environment, but Apply for OS/390 can use virtual memory instead. Unless you have virtual memory constraints, store the spill files in virtual memory rather than disk.

The size of the spill file is equal to the size of the data selected for replication during each replication interval. You can estimate the size of the spill file by comparing the frequency interval (or data-blocking interval; see “Data blocking for large volumes of changes” on page 66) planned for the Apply program with the volume of changes in that same time period (or in a peak period of change). The spill file’s row size is the *target row* size, including any DB2 DataPropagator overhead columns. This row size is not in DB2 packed internal format, but is in expanded, interpreted character format (as fetched from the SELECT). The row also includes a row length and null terminators on individual column strings.

Example: If change volume peaks at 12,000 updates per hour and the Apply program frequency is planned for one-hour intervals, the spill file must hold one-hour’s worth of updates, or 12,000 updates. If each update represents 100 bytes of data, the spill file will be about 1.2 MB.

Network planning

This section describes connectivity requirements, discusses where to run the Apply program (using the push or pull configuration), and describes how data blocking can improve performance.

Connectivity

Because data replication usually involves physically separate databases, connectivity is important to consider during the planning stages. The workstation that runs the DB2 Control Center or DJRA must be able to connect to the control, source, and target server databases to perform their tasks. And the Apply program must be able to connect to the control, source, and target server databases.

7. If you are using the ASNLOAD utility, you have a load input file instead of a load spill file.

When the databases are connected to a network, connectivity varies according to the platforms being connected:

- For connections between DB2 Universal Database databases, your choices are TCP/IP, SNA, NetBIOS, or IPX/SPX.
- For connections between DB2 Universal Database databases and DB2 for OS/390, DB2 for VSE, or DB2 for VM databases, you need DB2 Connect Personal Edition (or DB2 Connect Enterprise Edition) on the same workstation as the database to which you are connecting, or you need DB2 Connect Enterprise Edition (or DB2 Universal Database Enterprise Edition) available through your network. You can use TCP/IP or SNA for any of the following: DB2 for OS/390 V5 or later, AS/400 V4R2 or later, or DB2 for VM V5 or later. For all other connections, you can only use SNA.

If you use password verification for DB2 for OS/390, use Data Communication Service by adding DCS to the CATALOG DB statement. If you connect using SNA, add SECURITY PGM to the CATALOG APPC NODE statement. However, if you connect using TCP/IP, there is no equivalent security keyword for the CATALOG TCPIP NODE statement.

If your replication design involves staging data at a server that is different from the source database, you must carefully consider the communications between the various servers. For example, in a mobile replication scenario between DB2 Universal Database running on Windows 95 on a laptop PC and DB2 for OS/390, a good connectivity scenario might be for the Windows 95 PC to dial a local server (for example, an AIX server with DB2 Universal Database Enterprise Edition) using TCP/IP over a modem. The AIX workstation then connects to DB2 for OS/390 to fulfill the request from the Windows 95 machine.

Be sure to limit the layers of emulation, LAN bridges, and router links required, because these can all affect replication performance.

Where to run the Apply program: push or pull configuration

You can run the Apply program at the source server or at the target server. When the Apply program runs at the source server, you have a *push configuration*: the Apply program pushes updates from the source server to the target server. When the Apply program runs at the target server, you have a *pull configuration*: the Apply program pulls updates from the source server to the target server.

The Apply program can run in either or both configurations at the same time: it can push updates for some subscription sets and pull updates for others.

If the target table is in a non-IBM database, the Apply program connects to a DB2 DataJoiner database (with DB2 DataJoiner connected to the non-IBM database) and applies the changes to the target table using DB2 DataJoiner nicknames. In this case, the Apply program pushes updates from the DB2 DataJoiner source server to the target server or pulls updates from the DB2 DataJoiner source server to the target server. The Apply program cannot push or pull directly from the non-IBM server.

Figure 13 shows the differences between the push and pull configurations.

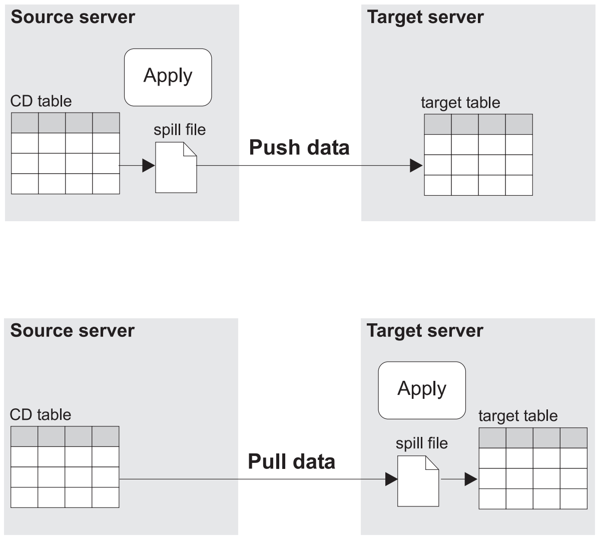


Figure 13. Push versus Pull Configuration

In the push configuration, the Apply program connects to the local source server (or to a DB2 DataJoiner source server for non-IBM sources) and retrieves the data. Then, it connects to the remote target server and pushes the updates to the target table. The Apply program pushes the updates row by row, and cannot use DB2's block-fetch capability to improve network efficiency.

In the pull configuration, the Apply program connects to the remote source server (or to a DB2 DataJoiner source server for non-IBM sources) to retrieve the data. DB2 can use block fetch to retrieve the data across the network efficiently. After all data is retrieved, the Apply program connects to the local target server and applies the changes to the target table.

Generally, a pull configuration performs better than a push configuration because it allows more efficient use of the network. However, under the following circumstances a push configuration is a better choice:

- When there is no Apply program for the target server platform, for example, as with VSE or VM.
- When you use remote journaling or relative record numbers (RRN) on your AS/400 source servers and the target server is a non-AS/400 system.
- The source table changes very infrequently, but when it does change it should be replicated as soon as possible.

To set up a push or pull configuration you need only to decide where to run the Apply program. DB2 DataPropagator, the DB2 Control Center, and DJRA recognize both configurations.

Data blocking for large volumes of changes

Replication subscriptions that replicate large blocks of changes in one Apply cycle can cause the spill files or log (for the target database) to overflow. For example, batch-Apply scenarios can produce a large backlog of enqueued transactions that need to be replicated. Or, an extended outage of the network can cause a large block of data to accumulate in the CD tables, which can cause spill-file overflows.

Use the Data Blocking page of the Subscription Timing notebook in the DB2 Control Center or the **Blocking factor** field of the Create Empty Subscription Sets window in DJRA to specify how many minutes worth of change data DB2 DataPropagator can replicate during a subscription cycle. The number of minutes that you specify determines the size of the data block. If the accumulation of change data is greater than the size of the data block, the Apply program converts a single subscription cycle into many mini-cycles, reducing the backlog to manageable pieces. It also retries any unsuccessful mini-cycles and will reduce the size of the data block to match available system resources. If replication fails during a mini-cycle, the Apply program retries the subscription set from the last successful mini-cycle. Figure 14 on page 67 shows how the changed data is broken down into subsets of changes.

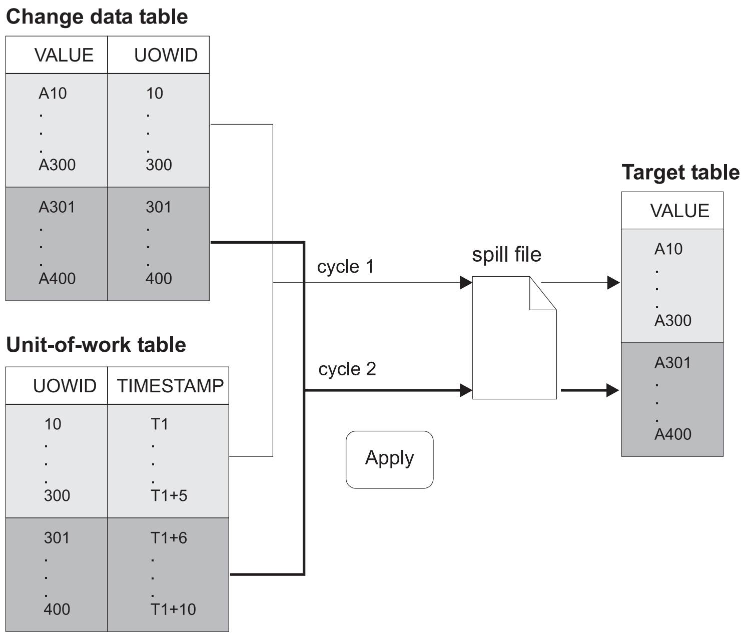


Figure 14. Data Blocking. You can reduce the amount of network traffic by specifying a data-blocking value.

By default, the Apply program uses no data blocking, that is, it copies all available committed data that has been captured. If you set a data-blocking value, the number of minutes that you set should be small enough so that all transactions for the subscription set that occur during the interval can be copied without causing the spill files or log to overflow. For AS/400, ensure that the total amount of data to be replicated during the interval does not exceed 4 MB.

Restrictions:

- You cannot split a unit of work.
- You cannot roll back previous mini-subscription cycles.
- You cannot use data blocking for full refreshes.

Deciding what to replicate

As part of planning for replication, you need to consider how the data will be used at the target site. Often, the source data will need to be subsetted, transformed, or enhanced for decision-support or data-warehousing applications. This section sorts these requirements into those that are easily fulfilled by using the DB2 Control Center and those that require direct manipulation of the control tables.

The Control Center and DJRA support the following data manipulations:

- Subsetting columns and rows
- Replicating joins using views
- Replicating before and after images
- Renaming columns
- Creating computed columns
- Using stored procedures for before and after run-time processing

The following sections describe the data manipulations that you can perform using the Control Center. This chapter also describes replicating large-object (LOB) data, limits for column names for before-image data, and data-type restrictions.

Subsetting columns and rows

IBM Replication supports both column (vertical) and row (horizontal) subsetting of the source table. This means that you can specify that only a subset of the source table columns and rows be replicated to the target table, rather than all of the columns and rows:

Column subsetting

In some replication scenarios, you might not want to replicate all columns to the target table, or the target table might not support all data types defined for the source table. You can define a column subset that has fewer columns than your source table. Column subsetting is available for all tables except replica tables.

You can define column subsetting at one of two times:

- When you define a replication source table for differential refresh.

  Select *only* those columns you want to make *available* for replication to a target table. Because CD tables must contain sufficient key data for point-in-time copies, you must include primary-key columns in your subset. The columns that you do not select will not be available for replication to any target table.
- When you define a subscription set.

  Use the advanced subscription options to select *only* those columns that you want to replicate to the target table. The columns that you do not select are still available for other subscription sets, but are not included for the current subscription set.

Recommendation: When you define a replication source, select *all* columns (that is, do not subset any of them). Create your column subsets when you define subscription sets. By defining your column subsets in the subscriptions rather than in the replication sources, you will not have to redefine your replication sources if your subscription requirements change.

Row subsetting

In some replication scenarios, you might want to replicate different data from a source table to several target tables. You can define a row subset that contains rows matching a certain condition (a WHERE clause), for example, all rows for department "J35".

Use the advanced subscription options to define a WHERE clause when you define the subscription. All target table types support row subsetting.

If the primary key values of the target table will be updated, or the table (or view) contains a logical partitioning column that will be updated, you must specify replication logical-partitioning-key support when you define the replication source. Replication logical-partitioning-key support performs an UPDATE as a DELETE followed by an INSERT. See "Enabling replication logical-partitioning-key support" on page 96 for more information.

Replicating joins using views

Join views fill many requirements, both for denormalizing (restructuring) copies in data-warehouse scenarios, which enables simpler queries of copied data, and also for addressing the routing problem, sometimes called the database partitioning problem in distributed computing scenarios.[8] Views are also useful when you need to specify predicates for a row subset that exceed 512 bytes (the capacity of the PREDICATES column of the subscription-targets-member control table). Thus, you can choose to manage your subset predicates using views rather than as part of the subscription-set definition.

To define a join view as a replication source, first define all tables that participate in the join as replication sources (you do not need to define subscriptions for them). Then, use the Control Center or DJRA to define the join. If the replication sources defined in the join have CD or CCD tables, the Control Center or DJRA creates a CD view from the replication sources' CD tables.

IBM Replication supports the following types of view definitions:

8. For example, knowing where to send a bank account update may require a join of the account table with the customer table to determine which branch of the bank the customer deals with. Typically, production databases are normalized so that the geographic details, such as branch-number, are not stored redundantly throughout the database.

- Simple views over a single table

  Only views of tables that reside within DB2 or DB2 DataJoiner databases are supported. Views of tables that are stored on Oracle, Microsoft SQL Server, Sybase, or Informix are not supported.
- Simple inner-joins over one or more defined replication sources
- Simple inner-joins over one or more CCD staging tables[9] that are defined as replication sources and maintained by an Apply program or an application other than an IBM Replication component and an external data source, such as DataPropagator NonRelational with IMS source data

Replicating before and after images

You can define both before and after images in your replication sources and subscriptions. A before-image column is a copy of a column before it is updated, and an after-image column is a copy of a column after it is updated. DB2 logs both the before-image and after-image columns of a table for each change to that table. Replicating before images can be useful for auditing purposes, and is required for update-anywhere scenarios.

The before and after images have different values for different actions, as shown below:

Action	**Column Value**
Full refresh	All before-image columns have a NULL value.
Insert	The before-image column has a NULL value.
Update	Column values before the change are captured in the before-image columns; values after the change are in the after-image columns. When you enable logical-partitioning key support, the before-image column appears in the deleted column and the after-image column appears in the inserted column. See “Enabling replication logical-partitioning-key support” on page 96 for more information.
Delete	Both the before-image and after-image columns contain the before-image value.

Before images do not make sense for base aggregate target-table types (there is no before image for computed columns). All other target-table types can make use of before-image columns.

9. The CCD tables for simple inner-joins must be complete and condensed. See “Staging data” on page 75.

Renaming columns

You can rename columns for point-in-time and user-copy target-table types. For other table types, you must define views in order to rename columns.

Creating computed columns

Using SQL, you can derive new columns from existing source columns. For aggregate target-table types, you can define new columns by using aggregate functions such as COUNT or SUM. For other table types, you can define new columns using SQL expressions.

You can also create computed columns by specifying user-defined functions when you create a table using the DB2 Control Center.

Using stored procedures for before and after run-time processing

You can define run-time processing statements using SQL statements or stored procedures that can run before or after the Apply program processes a subscription set. Such statements can be useful for pruning CCD tables and controlling the sequence in which subscription sets are processed. You can run the run-time processing statements at the source server before a subscription set is processed, or at both the source and target servers before or after a subscription set is processed. For example, you can execute SQL statements before retrieving the data, after replicating it to the target tables, or both.

Stored procedures use the SQL CALL statement without parameters. If the source table is in a non-IBM database, DB2 DataJoiner processes the SQL statements. The procedure name must be eight characters or less in length. The run-time procedures of each type are executed together as a single transaction. You can also define acceptable SQLSTATEs for each statement.

Depending on the DB2 platform, the SQL before and after processing statements can perform other processing, such as calling stored procedures.

Replicating large objects

DB2 Universal Database and DB2 for OS/390 V6 support large object (LOB) data types, which include: binary LOB (BLOB), character LOB (CLOB), and double-byte character LOB (DBCLOB). This section refers to all of these types as LOB data.

The Capture program reads the LOB descriptor to determine if any data in the LOB column has changed and thus should be replicated, but does not copy the LOB data to the CD tables. When a LOB column changes, the Capture program sets an indicator in the CD tables. When the Apply program reads this indicator, it then copies the entire LOB column (not just the changed portions of LOB columns) directly from the source table to the target table.

To allow the Capture program to detect changes to LOB data, you must include the DATA CAPTURE CHANGES keywords when you create (or alter) the source table.

Because a LOB column can contain up to two gigabytes of data, you must ensure that you have sufficient network bandwidth for the Apply program. Likewise, your target tables must have sufficient disk space to accommodate LOB data.

Restrictions:

- To copy LOB data between DB2 for OS/390 V6 and DB2 Universal Database (for any other operating system), you need DB2 Connect 5.2 or later.
- When the source data is in DB2 Universal Database (for any operating system other than OS/390) and the target data is in DB2 for OS/390 V6, you must use the push configuration for replicating LOB data. That is, the Apply program must run on the system with the source data.
- You can copy LOB data only to read-only tables. Thus, you cannot replicate LOB data to replica or row-replica tables.
- The primary key for the source table and the subscription-set definition must match. Code-page differences that affect key values can inhibit the Apply program's ability to locate the source-table row that contains LOB data.
- You cannot refer to LOB data using nicknames.
- For DB2 for OS/390, any table that contains LOB columns must also contain a ROWID column.

Limits on column names for capturing before-image data

DB2 DataPropagator limits column names to 17 characters. Because DB2 DataPropagator adds a before-image column identifier (usually *X*) to target tables and because you must ensure that each column name is unique, you cannot use longer column names for those tables you replicate. For tables you do not plan to replicate, you can use longer column names, but consider using 17-character names in case you might want to replicate these tables in the future. For tables in DB2 for OS/390, you can use 18-character column names, but DB2 DataPropagator will replace the 18th character with the before-image column identifier in target tables, so you must ensure that the first 17 characters of the name are unique.

Data restrictions

Currently, DB2 DataPropagator cannot replicate certain types of data. The major restrictions are:

- **Archive-log access restrictions with DB2 for VSE & VM**

  Allocate sufficient disk space for the active log because Capture for VSE and Capture for VM cannot read archived logs.
- **Data compression restrictions with DB2 for OS/390**

  DB2 DataPropagator can replicate data that is compressed through DB2 software or hardware compression on DB2 for OS/390 V4 (or higher) if the dictionary used to compress the data is available. Before issuing REORG for compressed replication sources, you must either:
  - Ensure that the Capture program completed capturing all existing changes.
  - Use the KEEPDICTIONARY option on the REORG command to preserve the existing compression dictionary.

  DB2 DataPropagator cannot replicate data that is compressed using EDITPROCs or FIELDPROCs.
- **Utility program restrictions**

  DB2 DataPropagator cannot capture updates made by any of the database utilities. DB2 DataPropagator also cannot capture updates to data loaded with the LOAD RESUME LOG YES options.
- **Data encryption restrictions**

  DB2 DataPropagator cannot replicate data that is encrypted.
- **Data type restrictions**

  DB2 DataPropagator *cannot* replicate the following data types under any circumstances:
  - Any column on which a VALIDPROC is defined
  - Binary data types with precision

  DB2 DataPropagator *can* replicate the following data types under certain circumstances:
  - Long variable graphic (LONG VARGRAPHIC) data requires that the source and target tables be in DB2 for OS/390, DB2 for VSE, or DB2 for VM.
  - Long variable character (LONG VARCHAR) data requires either that the source tables be in DB2 for OS/390 or both the source and target tables be in DB2 Universal Database Version 5.2 or later.
  - Binary large objects (BLOBs), character large objects (CLOBs), and double-byte character large objects (DBCLOBs) require DB2 for OS/390 V6 or DB2 Common Server V2 or higher. DB2 DataPropagator can

replicate only a full LOB; it cannot replicate parts of a LOB. See "Replicating large objects" on page 71.

– DB2 DataPropagator cannot capture changes to ASCII tables on DB2 for OS/390. DB2 DataPropagator can perform full refresh using ASCII tables.

User-defined data types (distinct data types in DB2 Universal Database) are converted to the base data type before replication.

- **DB2 DataJoiner restrictions**

  You must have one DB2 DataJoiner database for *each* non-IBM source server. Although you can use one DB2 DataJoiner database for replicating data to multiple non-IBM target servers, you need a unique DB2 DataJoiner database for *each* non-IBM source server. The reason for this restriction follows:

  For every replication scenario, there is a set of control tables, each with names that cannot be changed. When replicating to non-IBM target servers, none of these control tables needs to be located in the non-IBM database because a DB2 DataJoiner database is the target for the Apply program. The nicknames used here refer to the target table and not to any of the control tables.

  For non-IBM source servers, however, some of the control tables *must* be located in the non-IBM database so the Capture triggers can update them. Because of this location requirement, the DB2 DataJoiner nicknames associated with those control tables must be the actual control table names, and their schema must be ASN. Because a DB2 DataJoiner database cannot contain more than two identical nicknames with identical schemas, one DB2 DataJoiner database must be used for each non-IBM source server. You can, however, support multiple non-IBM source servers within one DB2 DataJoiner instance by creating multiple DB2 DataJoiner databases within that one DB2 DataJoiner instance.

  For DataPropagator for Microsoft Jet, row-replica tables are considered sources rather than targets, and so you do not need to define nicknames for Microsoft Jet or Microsoft Access tables in DB2 DataJoiner. See "Chapter 14. Mobile replication using DB2 DataPropagator for Microsoft Jet" on page 245.

Auditing data usage

Auditing is the need to track histories of data use, in terms of before and after comparisons of the data or identifying changes by time and by updating user ID.

IBM Replication supports auditing in the following ways:

Before and after images

When you define replication sources, you can include before-image columns of the updated rows in the target tables. A before-image copy is useful in some industries that require auditing or application rollback capability.

Maintenance of history

A noncondensed CCD table holds one row per UPDATE, INSERT, or DELETE operation, thus maintaining a history of the operations performed on the source table. If you capture UPDATEs as INSERTs and DELETEs (for partitioning key columns), the CCD table will have one row per DELETE and INSERT and two rows per UPDATE.

Transaction identification

Several columns in the CD tables and UOW table are available for audit use. You can find the approximate commit time of the changed row at the source server in the UOW table, and you can find the operation type (INSERT, UPDATE, and DELETE) in the CD table.

If you need more user-oriented identification, columns for the DB2 for OS/390 correlation ID and primary authorization ID or the AS/400 job name and user profile are available in the UOW table.

Staging data

During replication, DB2 DataPropagator *stages* changed data; that is, the Capture program captures changes to a source table only once and inserts changed rows into a change data (CD) table. The Apply program then retrieves the changes from the CD tables. The Capture program can also automatically prune changed rows from CD tables after they have been processed by all Apply programs and are no longer needed (if PRUNE is enabled). [10]

This section describes how you can stage data during replication and the role that consistent change data (CCD) tables play.

CD tables

A CD table receives an arbitrary number of changed data rows from the Capture program: one for each INSERT, UPDATE, and DELETE statement executed against the source table. The CD table does not know whether the transactions issuing the updates are committed, are incomplete, or are in flight. The Apply program joins the CD tables with the unit-of-work (UOW) table to determine which changes are committed and can therefore be

10. The Capture program does not, however, prune changed rows from consistent change data (CCD) tables. You must prune them manually.

replicated to target tables. Uncommitted changes are eventually pruned, depending on the retention limit that you define in the tuning parameters control table.

CCD tables

The Apply program joins the CD and UOW tables when determining what data is committed and can be replicated, but it does not save the results of the join; the Apply program can save the results of the join in consistent change data (CCD) target tables. The benefit of saving the join of the CD and UOW tables is that several subscriptions can refer to that join without having to perform it for each subscription cycle. CCD tables hold captured changes from INSERT, UPDATE, or DELETE operations against a source table.[11]

Because update-anywhere replication requires the most current change data for conflict detection, [12] CCD tables are not used in update-anywhere replication. For update-anywhere replication, every update should be replicated to the target table in the order in which it occurs; when you use CCD tables, the order in which the rows are replicated is not guaranteed. Also, the delays imposed on replication by using CCD tables can increase the likelihood of update conflicts' not being detected. Thus, if you define an internal CCD table, it is ignored by the Apply program when processing a subscription set with a replica as a target.

As described in "Chapter 1. Overview of data replication" on page 3, there are many types of CCD table, each of which has a different use. The following sections describe these types, how to define them, and what uses they have.

When using the DB2 Control Center to create your CCD tables, be sure to review the generated SQL statements to ensure the Control Center will create the type of CCD table, subscription, and registration you want.

Local and remote CCD tables

A CCD table is first classified by its *location*: local or remote. A local CCD table resides in the source database. A remote CCD table resides remote from the source database, that is, in any database in the network that the Apply program can access.

11. This statement is only true for noncondensed CCD tables; see "Condensed and noncondensed CCD tables" on page 77.

12. Also, referential constraints on the replica tables might not tolerate condensing that CCD tables allow.

When you specify the target table in the Subscription Definition window of the DB2 Control Center, you can choose whether the CCD table should be local or remote.

Complete and noncomplete CCD tables

In addition to its location, a CCD table can be classified by its *contents*. The first classification based on contents is whether the CCD table is complete or noncomplete. A complete CCD table contains *all* rows that satisfy the source view and subscription predicates from the source table or view. A noncomplete CCD table contains only modified rows from the source table. Thus, a noncomplete CCD table is initially empty and is populated as changes are made to the source table.

A complete CCD table is automatically registered as a replication source, so you will see it in the Replication Sources folder in the DB2 Control Center.

You specify that you want a CCD table to be complete by selecting the **Used as source for future copies** check box in the Advanced Subscription Definition window of the DB2 Control Center. If you do not select this check box, the CCD table is noncomplete.

Condensed and noncondensed CCD tables

The second classification of a CCD table based on its contents is whether it should be condensed or noncondensed. A condensed CCD table contains only the most current value for each row from the source table. A noncondensed CCD table contains all changes made to each row in the source table, that is, it represents the history of changes to each row.

For a condensed CCD table, the Apply program updates the values for a row; for a noncondensed CCD table, the Apply program inserts a new row to the CCD table for the updated row in the source table. For this reason, a condensed CCD table must have unique key values for each row, but a noncondensed CCD table can have multiple rows with the same key values. And because of the differences in key uniqueness, a condensed CCD table *must* have a unique index defined for it, whereas a noncondensed CCD table *must not* have a unique index.

If you select the **Used as source for future copies** check box in the Advanced Subscription Definition window of the DB2 Control Center, your CCD table will not only be complete, but also condensed. Likewise, if you do not select this check box, the CCD table will be both noncomplete and noncondensed.

If you select the **Include Unit-of-Work (UOW) table columns** check box in the Advanced Subscription Definition window of the DB2 Control Center, your CCD table will be noncomplete and condensed and it will include extra columns from the UOW table. If you do not select this check box, the CCD table will be both noncomplete and noncondensed and will not contain extra columns from the UOW table.

The DB2 Control Center provides no option for directly choosing whether a CCD table is condensed or noncondensed, but DJRA does. If you do not use DJRA, you must modify the register control table in the database where the CCD table resides and the subscription targets member table in the control database to specify that a CCD table should be condensed or noncondensed.

Internal and external CCD tables

The sequence in which you create CCD tables determines whether they are internal or external. The first local, noncomplete CCD table that you create is an internal CCD table; all other CCD tables are external. All remote CCD tables are external.

You cannot use an internal CCD table as an explicit source for a subscription set, but if an internal CCD table is present, the Apply program uses it to replicate changes instead of using the CD table.

If you perform a full refresh on an external CCD table, the Apply program performs a full refresh on all target tables that use this external CCD table as a replication source. This process is often referred to as a cascade full refresh.

Uses of CCD tables

In addition to minimizing the effects of joining the CD and UOW tables, you can use CCD tables to improve the efficiency and flexibility of your replication environment. The following examples show some of the uses for CCD tables:

- Maintaining complete histories of changes

  Use noncomplete, noncondensed CCD tables to keep a history of updates to a source table or to maintain an audit trail of database usage. For improved auditing capability, include the extra columns from the UOW table.
- Replicating data to multiple target tables

  Use a remote CCD table to reduce network traffic from the source server to the target servers. Changes to the source table are copied to the remote CCD, which acts as the source table for multiple target tables, thus potentially saving multiple network connections to the source server.

  Using remote CCD tables in this way is analogous to three-tier client/server configurations, where the source table acts as the first tier, the remote CCD table acts as the middle tier, and the target tables act as the third tier.

  Use a local, internal, noncomplete CCD table to control the time consistency of updates replicated to multiple sites. Then define one or more subscription sets using this CCD table as the replication source. This CCD table can shield its target tables from the volatility of the source tables if the replication to the CCD table is infrequent enough. Using a CCD table in this way is an example of a two-tier model.

- Condensing *hot-spot* updates

  Use a condensed CCD table to keep only the most current values for each row of the source table as it is continually updated. A hot spot develops when your application programs update a particular row many times in a short time interval. By keeping only the most current updates to each row in a condensed CCD table, you can reduce the network traffic because you do not have to replicate *all* of the updates to the target tables, instead you replicate only the most current update.

  In this case, ensure that the Apply program does not replicate too frequently (or as frequently as the hot spot develops) so you can benefit from using a condensed CCD table.

- Maintaining transaction-consistent replication

  Use condensed CCD tables to maintain transaction-consistent replication, that is to replicate only the net effect of all transactions that update the source table. To implement transaction-based replication, use noncondensed CCD tables to replicate every update from every transaction that updates the source table. Transaction-based replication is necessary for update-anywhere scenarios.

- Using CCD tables as replication sources

  A complete CCD target table is automatically registered as a replication source at the target server, and you can use this table when defining subscription sets. Using CCD tables as replication sources is useful, for example, for data warehousing and information repository scenarios.

- Using CCD tables as local caches for committed changes

  Use an internal CCD table as a local cache for committed changes to a source table. The Apply program replicates changes from an internal CCD table, rather than from CD tables, if one exists.

Using CCD tables for nonrelational data sources

Changes captured by application programs or other tools, such as DataPropagator NonRelational, can be defined as sources for subscription sets. The application program must create and maintain a complete CCD table. This CCD table must be external, but can be condensed or noncondensed. For example, DataPropagator NonRelational captures changes to IMS DB segments and updates its CCD table. You define the CCD table as a replication source using the DB2 Control Center or DJRA. You can then define subscription sets using this CCD table, regardless of where the original updates occur.

Pruning the CD and CCD tables

The Capture program can prune CD tables based on information inserted into the pruning control table by the Apply program. You control whether the Capture program prunes CD tables by using the PRUNE or NOPRUNE parameter. You can also control when the pruning takes place and how the prune interval is set by modifying the tuning parameters control table.

Several of the types of CCD table can continue to grow in size, especially noncondensed CCD tables. Pruning of these tables is not automatic; you must prune them manually or use an application program. For some types of CCD table, you may want to archive them and define new ones, rather than prune them.

When the source table is a non-IBM table, the Capture triggers prune the CCD table based on a synchpoint that the Apply program writes to the pruning control table.

Planning for migration

Beginning with DB2 Universal Database Version 5, the replication component installation is not optional. After you install DB2 UDB Version 6, you cannot:

- Continue to use Version 5 of the Capture and Apply programs.
- Install Version 6 of the Capture and Apply programs on a DB2 UDB Version 5 (or earlier) system.

Interoperability between Version 1 and Version 6 replication components is not supported. Therefore, you must complete all Version 1 to Version 5 migrations before you introduce DB2 UDB Version 6. For information about migration from Version 1 to Version 5, see the migration information on the DB2 DataPropagator Web pages: http://www.software.ibm.com/data/dpropr

DB2 UDB Version 6 includes significant new functions to support replication, including support for LOBs and DB2 Universal Database Satellite Edition. These functions involve changes to both the run-time (Capture and Apply program) and administration components. You can upgrade these components in any order.

The Version 5 Capture and Apply components can run alongside the Version 6 Capture and Apply components; you do not need to migrate all servers at the same time.

In addition:

- The Version 6 Capture and Apply programs are backward-compatible with Version 5 sources and subscription sets. The Version 6 components will continue to use the critical section table in the same way that the Version 5 Capture and Apply programs did if the new prune lock control table is not present.

- The Version 6 Capture and Apply programs can use the new invocation options introduced for DB2 Universal Database Satellite Edition, even if the administration component remains at the Version 5 level.
- Version 6 of the administration component supports defining sources and subscription sets that include LOB columns. If you use the Version 5 Capture and Apply programs, the presence of LOB columns will cause them to fail.

DB2 UDB Version 6 supports both the mobile replication enabler command **ASNCOPY** and the new DB2 Universal Database Satellite Edition enabler command **ASNSAT**. Both commands use the same run-time components and control tables, so you can use either one to control the Capture and Apply programs. However, you cannot use the DB2 Universal Database Satellite Edition **SYNCH** command in an existing replication environment because the **SYNCH** command relies on centralized administration controlled by a central control server. The central control server is not aware of any existing replication environment administered without use of the **SYNCH** command.

For more information about DB2 Universal Database Satellite Edition, see “Part 4. Occasionally connected environments” on page 237.

Chapter 6. Setting up your replication environment

This chapter describes the steps for setting up and starting replication. It does not include specifics about operating the Capture and Apply programs for each operating system. See “Part 3. Operations” on page 133 for such specifics.

You can use either the DB2 Control Center or the DB2 DataJoiner Replication Administration (DJRA) tool to define sources and targets for replication, to set the schedule for updating the targets, to specify the enhancements to the target data, and to define any triggers that initiate replication. You can use the DB2 Control Center to administer replication only when your source and target tables are in DB2 Universal Database databases (for any operating-system environment), but you can use DJRA to administer replication when your source and target tables are in DB2 Universal Database databases (for any operating-system environment) or in supported non-IBM databases.

The administration tasks described in this chapter set up the control information that both the Capture and Apply programs use to capture changed data and replicate it to the target tables, in the proper format, and at the appropriate interval.

Using the DB2 Control Center to set up replication

When setting up the replication environment, you can use the DB2 Control Center to manage the source and target table definitions and the control tables. Use the following high-level steps to administer your replication objects:

1. Check, and optionally update, the default settings in the Tools Settings notebook. See “Setting replication preferences in the DB2 Tools Settings notebook” on page 85 for more information.
2. Review the DPCNTL file for your platform to determine whether you need to customize the control tables for your site.
3. Optionally customize the DPCNTL file for your platform and site requirements. See “Defining replication control tables” on page 87 for more information.
4. Define and manage replication sources. See “Defining replication sources” on page 92 for more information.
5. Define and manage replication subscriptions. See “Defining replication subscriptions” on page 97 for more information.

After you create the control tables and define the replication sources and targets, you need to configure and run the Capture and Apply programs to begin replicating data.

You can access your replication sources and targets through the Control Center. There are three containers in the Control Center for organizing the objects that you use to set up and maintain your replication environment:

Tables folder
The folder containing DB2 tables.

Replication Sources folder
The folder containing tables that have been defined as replication sources: DB2 tables, views, or target tables redefined as sources for replication.

Replication Subscription folder
The folder containing subscription-set definitions for copying source data or source-data changes to target tables.

Each object also has a menu for the actions that can be performed with the object.

Configuring the Control Center for host RDBMSs

If you are connecting to a DB2 for OS/390, DB2 for VSE, DB2 for VM, or DB2 for AS/400 server from the Control Center, you must configure connectivity to the remote database, catalog the remote databases, and bind packages to the remote databases.

To bind the database:

1. Change to the directory where the Capture program bind files are located, which is usually the \SQLLIB\BND directory on the drive where you installed DB2 Universal Database or the client application enabler (CAE).
2. Create and bind the DB2 for OS/390, VSE, VM, or AS/400 package to the DB2 database by issuing the following commands:

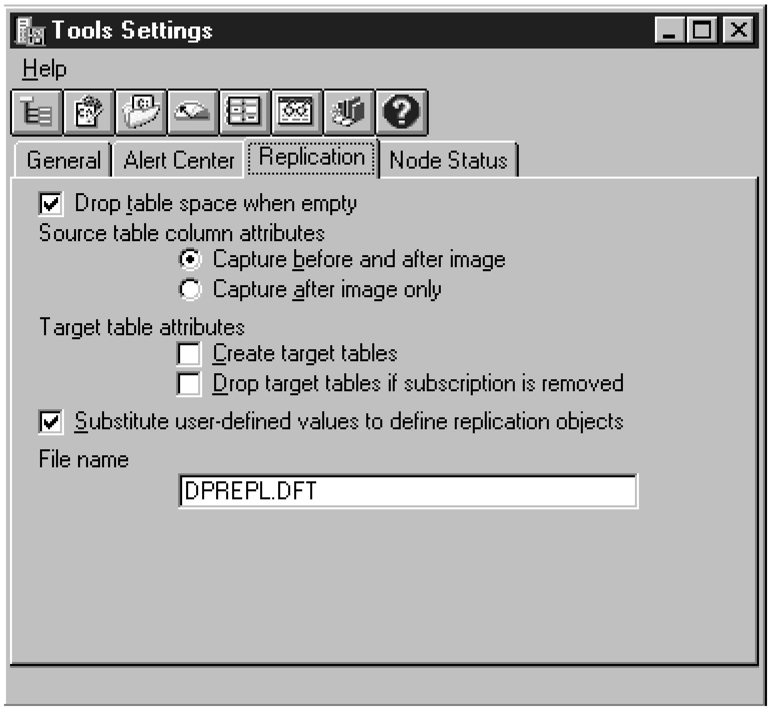
   ```
   DB2 CONNECT TO dbname USER userid USING password
   DB2 BIND @DDCSxxx.LST ISOLATION CS BLOCKING ALL SQLERROR CONTINUE
   ```

   Where CS specifies the cursor stability isolation level, and *xxx* specifies the platform name: MVS, VSE, VM, or AS/400.

   If the user ID and password are different than the local logon ID and password for the Control Center workstation, you must explicitly connect to the database server using the **Connect** menu choice from the pop-up menu for your remote database object.

Setting replication preferences in the DB2 Tools Settings notebook

The Tools Settings notebook contains default preferences for the DB2 Universal Database administration tools. You can set replication default values on the Replication page of the notebook, as shown in Figure 15. These default values are used for all replication activities administered by the Control Center.



[✓] Drop table space when empty

Source table column attributes

● Capture before and after image

○ Capture after image only

Target table attributes

[ ] Create target tables

[ ] Drop target tables if subscription is removed

[✓] Substitute user-defined values to define replication objects

File name: DPREPL.DFT

Figure 15. The Replication Page of the Tools Setting Notebook. Use this page to specify default preferences for replication.

Using the DB2 DataJoiner Replication Administration tool to set up replication

When you use the DB2 DataJoiner Replication Administration (DJRA) tool to perform replication administration tasks, DJRA connects to the source, target, or control server to create and update the control information and target tables on the server (depending on the operation performed). The client workstation where DJRA is located must be authorized and able to connect to all source, target, and control servers that are managed by DJRA.

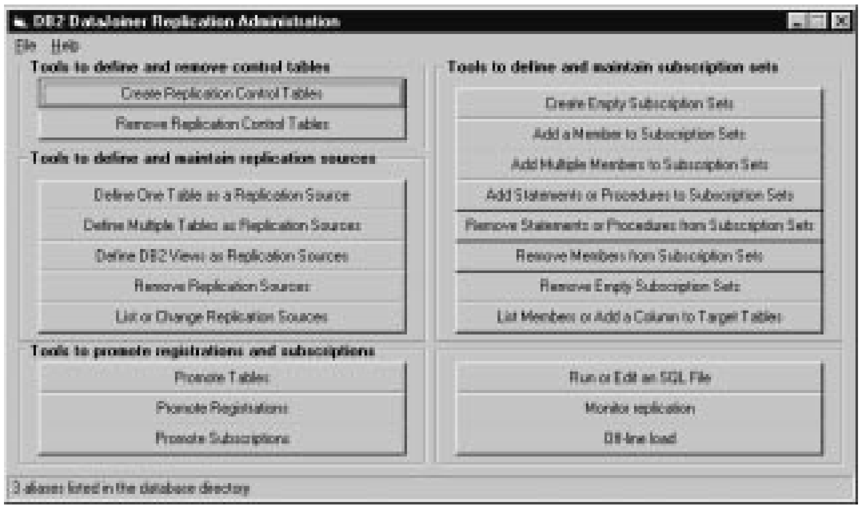
For DB2 source, target, or control servers, DB2 DataJoiner's distributed database connection services (DDCS) or the DB2 Connect product provides connectivity. For non-IBM sources and targets, DJRA uses DB2 DataJoiner to connect to the non-IBM servers. Non-IBM databases cannot act as control servers.

DJRA provides a user interface that is divided into areas that deal with control tables, sources, subscription sets, and the running or editing of SQL (see Figure 16 on page 87).

Using this interface, you can perform the following administration tasks:

- Create replication control tables and put them on your source, target, or control servers
- Define DB2 tables, non-IBM tables, and DB2 views as sources
- Remove replication sources
- Change the definitions for existing DB2 source tables to add new columns
- Promote table, registration, and subscription definitions
- Define subscription sets and subscription members
- Change existing subscription members for DB2 target tables to add new columns
- Remove subscription sets or subscription members that are no longer needed
- Add SQL statements or delete SQL statements or stored procedures that should run before or after the target tables are replicated
- Run or edit SQL that is generated by DJRA
- Monitor replication
- Perform an offline load of a table

You can also customize the logic for most of the administration tasks listed above.


DB2 DataJoiner Replication Administration
File
Help
Tools to define and remove control tables
Create Replication Control Tables
Remove Replication Control Tables
Tools to define and maintain replication sources
Define One Table as a Replication Source
Define Multiple Tables as Replication Sources
Define DB2 Views as Replication Sources
Remove Replication Sources
List or Change Replication Sources
Tools to promote registrations and subscriptions
Promote Tables
Promote Registrations
Promote Subscriptions
Tools to define and maintain subscription sets
Create Empty Subscription Sets
Add a Member to Subscription Sets
Add Multiple Members to Subscription Sets
Add Statements or Procedures to Subscription Sets
Remove Statements or Procedures from Subscription Sets
Remove Members from Subscription Sets
Remove Empty Subscription Sets
List Members or Add a Column to Target Tables
Run or Edit an SQL File
Monitor replication
Off-line load

3 aliases listed in the database directory

Figure 16. The DJRA Primary Window

For more information on using DJRA, see “Chapter 16. DJRA overview” on page 271 .

Defining replication control tables

Normally, the replication control tables are created in one of the following ways:

- By customizing the DPCNTL file for your operating system and running the file *before* initiating any actions from the DB2 Control Center.

  See comments within the file for tailoring the SQL for a specific database platform. You need to customize the DPCNTL file for the following definitions:

  – To define the location and size of DB2 for OS/390 table spaces and databases for the control tables. The Control Center creates the control tables in the default table space and database, unless you specify a different table space or database.

  – To define and size DB2 for VSE or VM dbspaces for the control tables. The Control Center creates the control tables in the default dbspace, unless you specify a different dbspace.

  – To tailor the control tables for specific operating system environments because not all definitions are supported for all operating systems.

  – To place control tables in specific DB2 Universal Database table spaces or in a DB2 EEE single node group.

- By using the Control Center to define replication sources and subscriptions; these actions create default versions of the control tables.

  If you use this option, you cannot customize the replication control tables without dropping the existing control tables and customizing them. If you are running OS/390, VSE/ESA, or VM/ESA, you *must* customize the replication control tables.
- By selecting **Create Replication Control Tables** in DJRA.

When you create customized control tables, you must customize the CREATE TABLE statements in the DPCNTL files. There is one DPCNTL file for each operating system environment; these files are located in the SQLLIB\SAMPLES\REPL\ directory. The file names are:

DPCNTL.UDB
Creates control tables for DB2 Universal Database.

DPCNTL.MVS
Creates control tables for DB2 for OS/390.

DPCNTL.VM
Creates control tables for DB2 for VSE & VM.

DPCNTL.400
Creates control tables for DB2 for AS/400.

DPCNTL.SAT
Creates and drops control tables for DB2 Universal Database Satellite Edition.

If, after creating customized control tables, you need to drop them, you must customize the DROP TABLE statements in the DPNCNTL files. There is a DPNCNTL file for each operating system environment located in the SQLLIB\SAMPLES\REPL\ directory. The files names are:

DPNCNTL.UDB
Drops control tables for DB2 Universal Database.

DPNCNTL.MVS
Drops control tables for DB2 for OS/390.

DPNCNTL.VM
Drops control tables for DB2 for VSE & VM.

DPNCNTL.400
Drops control tables for DB2 for AS/400.

To customize the SQL for creating or dropping control tables:

1. Open the appropriate file (SQLLIB\SAMPLES\REPL\DPCNTL.*platform_name* or

SQLLIB\SAMPLES\REPL\DPNCNTL.*platform_name*, where *platform_name* is UDB, MVS, or VM.) in a text editor.

2. Read the commented areas for each operating system and table.
3. Edit the file for your site or application.
4. Close the file.
5. Connect to the database in which the control tables will be created (use the **DB2 CONNECT TO** *database-name* command).
6. Run the file (DPCNTL or DPNCNTL) using one of the following commands from a command window:

```
db2 -tf dpcntl.platform_name

db2 -tf dpncntl.platform_name
```

Creating replication control tables using DJRA

You must create control tables at each DB2 (and DataJoiner) system involved in replication. For DB2 Universal Database systems, you can use the DB2 Control Center or DJRA to perform this task; for other systems, including DB2 for OS/390, DB2 for AS/400, and all non-IBM databases, you must use DJRA. When you complete this step, DJRA places a register table, a pruning control table, and a register synchronization control table at the database source (and for non-IBM sources, creates nicknames for these tables in DB2 DataJoiner).

From the DJRA primary window, click the **Create Replication Control Tables**. The fields you complete in order to create a control table are:

Source, control, or target server
When you click the down arrow, DJRA checks to see what type of server it is and then lists all databases and aliases that are catalogued on the workstation from which you are running DJRA. If you select a DataJoiner server from the list, the DataJoiner non-IBM source server pull-down list becomes active. If you do not choose a DataJoiner server, you will link directly to a DB2 database.

DataJoiner non-IBM source server
If you selected a DataJoiner alias from the **Source, control, or target server** pull-down list and you have performed server mappings in DataJoiner, then this list displays available remote server names.

Specify **(None)** if you want the control tables to be created in the DataJoiner database rather than in the remote server database.

Edit Tablespace Logic
Click this push button to customize table space names for control tables or for CREATE TABLESPACE options. The default table space names are:

- TS_UOW for the UOW table
- TS_CNTL for all other control tables

Generate SQL
Click this push button to generate SQL after you supply all the information on this panel. While the SQL is being generated, a window is displayed showing processing messages and error messages, if any.

When the procedure completes successfully, save the file by selecting **Save** from the **File** pull-down menu. You can now edit the generated SQL, according to the guidelines that are listed in "Editing DJRA-generated SQL" on page 289. When you are ready, run the SQL by selecting **Run** from the **File** pull down menu. You must save the generated SQL before you can run the SQL. You must run the SQL for generating control tables before you generate and run SQL to create replication sources or subscriptions.

Customizing and running replication SQL files

From the DB2 Control Center, you have the option to run a replication task immediately or save the generated SQL file to run at a later time. From DJRA, you can run or edit SQL files from the main window. The SQL files can be customized for large scale replication actions such as defining subscription sets, or customized for an application beyond implementations supported by either the Control Center or DJRA.

You might want to save and customize the SQL files to:

- Create multiple copies of the same replication action, customized for multiple servers.
- Customize CD table names.
- Define the location for CD tables (DB2 for OS/390 database, DB2 Universal Database table spaces, DB2 for VSE & VM dbspaces).
- Set the size of the table spaces, databases, or dbspaces of the CD tables.
- Define site-specific standards.
- Combine definitions together and run as a batch job.
- Defer the replication action until a specified time.
- Create libraries of SQL files for backup, site-specific customization, or to run stand-alone at distributed sites, such as for a mobile environment.

If you save the definitions of a large replication subscription set in an SQL file, you can rerun the definitions as necessary.

Setting up security for replication

Because DB2 DataPropagator is table driven, security for all replication objects depends on the database security. The database administrator who defines replication sources and subscriptions also defines security for them. Additionally, the Capture program must be authorized to access the source database and the Apply program must be authorized to access the control, source, and target databases.

Authorization requirements for administration

When you define replication sources and subscriptions, the DB2 Control Center and DJRA create many tables. Depending on the operating system, they might also create table spaces or dbspaces. Because all of these actions require a high level of database privilege, you should plan to have at least one user ID that acts as the replication administrator and has the authority to create objects, bind plans, and run generated SQL for each of the source databases.

Authorization requirements for running the Capture program

The user ID that runs the Capture program must be able to access the DB2 system catalog, be able to access and update all replication control tables, and have execute privileges on the Capture program packages. The user ID that runs the Capture program can be the same as the administrator user ID, but this is not a requirement.

For OS/390, the user ID that runs the Capture program should have either SYSADM authority or have the following authorizations:

- SELECT, UPDATE, INSERT, and DELETE privileges for all Capture-related tables created explicitly, and any Capture-related tables that the Control Center or DJRA implicitly creates. See "Chapter 21. Table structures" on page 299 for a list of these tables.
- SELECT privilege for the DB2 catalog (SYSIBM.SYSTABLES and SYSIBM.SYSCOLUMNS).
- TRACE privilege.
- MONITOR1 and MONITOR2 privilege.
- EXECUTE privilege for the Capture plan.

For VM and VSE, the user ID that runs the Capture program must have DBA authority. For all other operating systems, the user ID that runs the Capture program must have either DBADM or SYSADM authority.

Authorization requirements for running the Apply program

The user ID that runs the Apply program must be a valid logon ID for the source, control, and target servers, and for the workstation where the Control Center or DJRA is installed. The user ID that runs the Apply program must be able to access the source tables, access and update all replication control tables, and update the target tables. This user ID must also have execute privileges on the Apply program packages. The user ID that runs the Apply program can be the same as the administrator user ID, but this is not a requirement. With the proper authorization, any user ID can run any Apply program instance.

An Apply program might require a password file to connect to the source or target server. For more information about authorization requirements for the Apply program, see the Capture and Apply chapter for your operating system in “Part 3. Operations” on page 133.

Defining replication sources

To define a replication source using the DB2 Control Center, click the **Tables** folder for the source database to show all tables. Right-click on a table object to show the pop-up menu and select **Define as replication source**.

You can define replication sources using the **Quick** or **Custom** choices. **Quick** allows you to define a replication source using default values. **Custom** allows you to customize the defaults, such as specifying that certain columns should not be captured.

After you define the replication source, an object is created in the **Replication Sources** folder. The source table can now be defined in a subscription set.

To define a replication source using DJRA, click **Define One Table as a Replication Source** or **Define Multiple Tables as Replication Sources**, then fill in the required information, such as source server, source table names, and source columns.

For data restrictions when defining replication sources and subscriptions, see “Data restrictions” on page 73.

The Capture program does not recognize new DB2 replication sources until you issue either the **reinit** command or stop and restart the Capture program. The Capture program does not begin capturing changes for a replication source until a subscription is created for that replication source and the subscription set members have been fully refreshed.

Defining replication sources for update-anywhere replication

To define a replication source for update-anywhere replication using the DB2 Control Center, define a custom replication source and use the following selections:

1. Select the **Table will be used for update anywhere** check box.

   When you select this check box, the DB2 Control Center also selects the **Define as Source** and **Capture before image** check boxes for every column.
2. Select a conflict detection level:

   None No conflict detection.

   Attention: Conflicting updates between the source table and the replica will *not* be detected. This option is *not* recommended for update-anywhere replication.

   Standard
   Moderate conflict detection, in which the Apply program searches rows already captured in the replica's change data tables for conflicts. Standard detection is the default value.

   Enhanced
   Conflict detection that provides the best data integrity among all replicas and the source table. The Apply program locks all replicas in the subscription set against further transactions, and begins detection after all changes prior to the locking are captured.

   Even if you specify enhanced conflict detection, when the Apply program runs in a mobile environment (started with the **asncopy** or **asnsat** command, or with the COPYONCE keyword), the Apply program uses standard conflict detection.

To define a replication source for update-anywhere replication using DJRA, select the conflict detection level (described above) when you define a table as a replication source, and select the replica target structure when you add the member to the subscription set.

To reduce the risks of conflicts and costs of rejected conflicting transactions, use update-anywhere replication under the following conditions:

Fragmentation by key
Design your application so that the replication source is updated by replicas for key ranges at specific sites. For example, your New York

site can update sales records only for the Eastern United States (using ZIP codes[13] less than 49999 as the key range), but can read all sales records.

Fragmentation by time
Design your application so that the table can be updated only during specific time periods at specific sites. The time periods must be sufficiently separated to allow for the replication of any pending changes to be made to the site that is now becoming the master version. Remember to allow for time changes, such as Daylight Savings Time or Summer Time, and for time-zone differences.

Detecting conflicts

For update-anywhere replication, update conflicts can occur when:

- An update is made to a row in the source table and a different update is made to the same row in one or more replica tables.
- Constraints are violated.

The Apply program detects update conflicts, after they occur, during the subscription cycle. The source table is considered the primary table. That is, it can receive updates from replica tables, but if there is a conflict, the source table wins and the replica tables' conflicting transactions are rejected. The Apply program detects direct row conflicts by comparing the key values in the CD tables with the source and target tables. If it finds any that match, it marks the replica transaction as rejected in the UOW table and rolls back the replica transaction.

The Apply program cannot detect read dependencies. If, for example, an application reads information that is subsequently removed (by a DELETE statement or by a rolled back transaction), the Apply program cannot detect the dependency.

DB2 DataPropagator provides three levels of conflict detection: no detection, standard detection, and enhanced detection. Each level has a numerical value which is stored in the CONFLICT_LEVEL column of the register control table. You must decide, based on your tolerance for lost or rejected transactions and performance requirements, which type of detection to use. See "Defining replication sources for update-anywhere replication" on page 93 for more information about the levels of conflict detection and how to specify them.

Use the rejection codes provided in the UOW table to identify the before and after row values in the CD table for each rejected transaction. Because the ASNDONE exit routine runs at the end of each subscription cycle, you can

13. United States postal codes

add code to the routine to handle any rejected transactions. See "Using the ASNDONE exit routine" on page 115 for more information on the ASNDONE exit routine. Alternatively, because the change data rows and UOW control table rows for rejected transactions are exempt from normal pruning (they are, however, subject to RETENTION_LIMIT pruning), you could handle the rejected transactions as a batch by using a program that scans the UOW table.

Defining join replication sources

You can define replication sources that are views of other tables. After defining each replication source table included in the view, you can create a view replication source. The view replication source is then available for replication to a target table.

You cannot use the DB2 Control Center to define an existing view as a replication source; use DJRA instead.

To define a join using the DB2 Control Center:

1. Define the source tables to be used in the join as replication sources.
2. Click on the **Replication Sources** folder. Select the replication sources to be used in the join from the contents pane. Right-click the mouse button, then select **Define join** from the pop-up menu. The Define Join window opens.
3. In the **CREATE VIEW** field, type the SQL statement for the view. For example:
   ```
   USERID.VIEW_NAME AS SELECT A.COL1, A.COL2, B.COL6, B.COL5
   ```
   Do not type the words CREATE VIEW. This part of the statement is automatically supplied during processing.
4. In the **FROM** field, type table names that define the join. For example:
   ```
   TABLEA A, TABLEB B
   ```
   Do not type the word FROM. This part of the statement is automatically supplied during processing.
5. If you want to use a row predicate, type the WHERE clause SQL statement in the **WHERE** field. For example:
   ```
   A.COL1=B.COL1
   ```
   Do not type word WHERE. This part of the statement is automatically supplied during processing.
6. Select **OK** to save the values and close the window. After you run the SQL that defines the join view, it is available for replication subscriptions.

To define a view as a replication source using DJRA, click **Define DB2 Views as Replication Sources** and fill in the required information, such as source server, source view qualifier, and source view name.

Enabling replication logical-partitioning-key support

By default, the Capture program captures an update to the source table as an UPDATE statement. However, for the following conditions, you must instruct the Capture program to capture updates as DELETE and INSERT statements (that is, you must enable logical-partitioning-key support):

- Your source applications update one or more columns that are part of a target table's primary key.

  Because the values for the target-table primary key come from the changes captured on the source server, which reflect the new key values, these values cannot be used to find the existing target table row (it doesn't exist yet). Converting the UPDATE to a DELETE and INSERT pair ensures that the target table reflects the changes made at the source server.
- Your source applications update one or more columns referenced by a subscription-set predicate.

  In this case, the column included in the predicate need not be a primary-key column. If a subscription is defined with a predicate based on a specific column value (for example, WHERE DEPT = 'J35'), and you change that column (for example, to DEPT='FFK'), the captured change will not be selected for replication because it does not meet the predicate criteria. That is, your new FFK department will not be replicated because your subscription is based on department J35. Converting the UPDATE to a DELETE and INSERT pair ensures that the target-table row is deleted.
- Your source applications update one or more columns of a target-table partitioning key (either the target table is a partitioned database managed either by DB2 Extended Enterprise Edition (EEE) or DB2 for AS/400, or is a table in a DB2 for OS/390 partitioned table space).

  Enabling logical-partitioning-key support ensures that target rows are moved from one node to another when the source column for the logical-partitioning key is changed and replicated. The move is accomplished by a DELETE at the old node and an INSERT at the new node.

By default, when you update the primary keys of either the source or target tables, the Capture program captures the changed row for the update. The Apply program then attempts to update the row in the target table with the new key value. This new key value is not found in the target table, so the Apply program converts this update to an insert. In this case, the old row with the old key value remains in the table (and is unnecessary). When you enable replication logical-partitioning-key support, the Capture program captures the change as separate DELETE and INSERT statements: delete the old row and insert the new row.

Each captured UPDATE is converted to two rows in the CD table for all columns, non-key columns as well as key columns. You might need to adjust the space allocation for the CD table to accommodate this increase in captured data.

When you use the DB2 Control Center to define the source table, select the **Changed data for partitioned key columns captured as delete and insert** check box on the Define as Replication Source window to specify that the Capture program should capture updates as DELETE and INSERT statements.

When you use DJRA to define the source table, select the **Updates as delete/insert pairs** radio button from either the Define One Table as a Replication Source window or the Define Multiple Tables as Replication Sources window.

Defining replication subscriptions

To define a replication subscription using the DB2 Control Center:

1. Click the **Replication Sources** folder for the source database to show all tables and views defined as replication sources. The replication sources appear in the contents pane.
2. Select one or more tables or views that you want to define as sources for the subscription set, and right-click on one to show the pop-up menu and select **Define subscription**. The Define Subscription window opens.
3. Give the subscription a name, specify the target server, and specify the Apply qualifier for the subscription. You can also change the name of the target table and specify whether the Apply program should create the target table.
4. Click the **Advanced** push button to specify the target type and to specify specific columns and rows. See "Choosing a target-table type" on page 99 and "Defining the target-table structure: columns and rows" on page 100 for more information about these tasks.
5. Click the **Timing** push button to specify the frequency of replication and a data blocking value. See "Specifying a data-blocking value" on page 106 for more information.
6. Click the **SQL** push button to add SQL statements or stored procedures that you want to run before or after a subscription cycle. For example, you can add a DELETE statement to prune the Apply trail control table.

7. Click the **OK** push button to complete the subscription definition. The Subscription Information window opens. In that window, specify the control server name.

To define a replication subscription using DJRA:

1. From the main window, click **Create Empty Subscription Sets** to open the Create Empty Subscription Sets window.
2. In this window, specify the source server, control server, target servers, the Apply qualifier, the subscription set name, the subscription timing, and blocking factor.
3. Add subscription-set members to the subscription set.
   a. From the main window, click **Add a Member to Subscription Sets** or **Add Multiple Members to Subscription Sets** to display either the Add a Member to Subscription Sets window or the Add Multiple Members to Subscription Sets window.
   b. In this window, specify the subscription sets to which you want to add a member, the tables and views to add to the subscription set, whether the target table should be a column or row subset of the source table (see "Defining the target-table structure: columns and rows" on page 100), the target table type (see "Choosing a target-table type" on page 99), and how the index for the target table should be created.

If you defined an event to start the Apply program, you must populate the event table. See "Event timing" on page 107 for more information about this task. To begin replicating data to the target tables, start the Capture program at the source server, then start the Apply program using the name of the control server that you specified in the Subscription Information window.

Defining replication subscriptions for update-anywhere replication

To define a replication subscription for update-anywhere replication using the DB2 Control Center, define a subscription set and use the following selections:

1. Select the replication sources that you want to be in the subscription set. Include all sources affected by the replica tables being updated.
2. From the Subscription Definition window, select a target table to be defined as a replica table.
3. Click **Advanced** to open the Advanced Subscription notebook. The following selections are required on the Advanced Subscription notebook:
   a. From the Target Type page, click on **Target table is replica**.
   b. From the Target Columns page:
      1) Ensure that the **Subscribe** check boxes are selected for every column. *Do not* create new columns for the replica table.

        2) Specify a primary key for the replica table by clicking on the **Primary Key** check boxes next to the key column names.

           Make the primary key the same as the source-table primary key to prevent conflicts. Do not use before-image columns as primary-key columns for the target table.

           Important: For existing target tables, you must select the primary-key columns.

   c. If you want the replica to be a subset of the source table, type a row predicate in the **WHERE** field on the Rows page.
   d. Click **OK** to close the Advanced Subscription notebook.
4. Repeat step 3.b for each target table.
5. Click **Timing** to open the Subscription Timing notebook:
   a. On the Source to Target page, fill in the subscription set timing information for copying the source-table's changed data to the target tables.
   b. On the Replica to Source page, fill in the subscription-set timing information for copying the replica-table's changed data to the source tables.
   c. Click **OK** to close the notebook.
6. If you want to define SQL or CALL procedures to run before or after the subscription set is processed, click **SQL** and define the processing statements.

To define a replication subscription for update-anywhere replication using DJRA, select the replica target structure when you add the member to the subscription set.

Choosing a target-table type

You can specify a target-table type if you do not want to accept the default target type of user copy.

To specify a target-table type using the DB2 Control Center:

1. From the Define Subscription window, select a source and target table combination, and click **Advanced** to open the Advanced Subscription Definition notebook.
2. From the Target Type page, select one of the following table types:
   - For read-only target tables, you can select:

     User copy
       A target table that matches the source table exactly at the time of the copy.

Point-in-time
: A target table that matches the source table, with a timestamp column added.

Staging table
: Also known as a CCD table. See "CCD tables" on page 76 for more information about CCD tables. If you select this option without selecting either of the two options below, the DB2 Control Center creates a noncomplete, noncondensed CCD table.

  Used as source for future copies
  : Select this option if you want a complete, condensed CCD table.

  Include Unit-of-Work (UOW) table columns
  : Select this option if you want a noncomplete, condensed CCD table that includes extra columns from the UOW table.

Base aggregate
: A target table that contains aggregated data for a user table appended at specified intervals.

Change aggregate
: A target table that contains aggregated data based on changes recorded for a source table.

- For updateable target tables, select **Target table is replica** to create an updateable target table used for update-anywhere replication.

3. If you are finished using the Advanced Subscription Definition notebook, select **OK** to close the notebook. Otherwise, use the other pages of the notebook to define the target table columns and rows, as needed.

To specify a target-table type using DJRA, click **Add a Member to Subscription Sets** or **Add Multiple Members to Subscription Sets**. Fill in the required information for the subscription-set member. You can specify the target-table type from the **Table structure** drop-down list. The available types include the same as those described for the DB2 Control Center, plus choices for CCD table types.

Defining the target-table structure: columns and rows

For some applications, the target table does not need all of the rows or columns that exist in the source table. You can define the target table to be a column or row subset of the source table using the Advanced Subscription Definition notebook. For more information on subsetting, see "Subsetting columns and rows" on page 68.

Restriction: Replica target tables must contain the same columns as the source table: you cannot create subsets for them; you cannot add columns; and you cannot rename columns.

Defining the target-table columns

To define the target-table columns using the DB2 Control Center:

1. From the Define Subscription window, select a source and target table combination and click **Advanced** to open the Advanced Subscription Definition notebook.
2. From the Target Columns page, specify which columns should be the primary-key columns for the target table; you can rename columns and you can change column definitions.

   If you want to specify a column as a primary-key column for the target table, select the **Primary Key** check boxes next to the column names.

   Attention: For the following target-table types you must select one or more columns as part of a primary key: user copy, point-in-time, replica, and condensed staging tables. If you do not select columns for the primary key, DB2 uses the primary-key definition of the source table. However, if the source table does not have a primary-key definition, the Apply program issues an error message.

   If you want to rename a column, select the column name and type over the existing column name. A column name can have up to 17 characters and can be an ordinary or delimited identifier.

   If you want to change a column definition for the target table, click **Change** to open the Change Column window. From this window, you can:

   - Change the name of the column.
   - Change the definition of the column by entering an SQL expression. For example: `COUNT(*)` or `EMP_SALARY – EMP_COMM`.

     The expression can contain up to 254 characters and can be any valid SQL expression. This expression can contain ordinary or delimited identifiers. Columns used in the expression must be valid after-image columns from the source table. These column names are listed in the **Available columns** box.

     See the *DB2 SQL Reference* for information on valid SQL expressions. Invalid SQL expressions cause an SQL error when the Apply program processes the subscription.
   - See examples of valid SQL expressions; click **Examples**.

   If you want to remove a column from the target table, clear the **Subscribe** check box next to the column name.

If you want to create a new computed column or use aggregation for the target table:

a. Click **Create Column** to open the Create Column window.
b. Type the name of the column in the **Column name** field. The name can have up to 17 characters and can be an ordinary or delimited identifier.
c. Type the SQL expression defining the new column.
d. Click **OK** to close the window.

3. If you are finished using the Advanced Subscription Definition notebook, select **OK** to close the notebook. Otherwise, use the Rows page to define the target table rows, as needed.

To define the target-table columns using DJRA, click the **Selected columns** radio button from the Add a Member to Subscription Sets window. Then select the columns you want replicated to the target table.

Defining the target-table rows

To define the target-table rows using the DB2 Control Center:

1. From the Define Subscription window, select a source and target table combination and click **Advanced** to open the Advanced Subscription Definition notebook.
2. From the Rows page, specify a WHERE clause that defines the row subset.

   To specify which rows are copied to the target table, type an SQL predicate in the **WHERE** field. The predicate can contain ordinary or delimited identifiers. See the *DB2 SQL Reference* for more information about WHERE clauses.

   Row Predicate Restrictions:

   - Do not type `WHERE` in the clause; it is implied. Type `WHERE` in the clause only for subselect statements.
   - Do not end the clause with a semicolon (;).
   - You can use only those column names shown in the **Target columns** list on the Target Columns page; these names are also listed in the **Available columns** list.
   - Do not use before-image columns, computed columns, or IBMSNAP columns in the WHERE clause. Before-image columns are supported for CD tables but not for user tables, point-in-time tables, or replica tables.[14]
   - If you added computed columns (on the Target Columns page), you must provide a GROUP BY clause. Both base aggregate and change aggregate tables must have a GROUP BY clause.

14. If you use before-image columns or computed columns, for example, full refresh is no longer possible. You must also modify the register control table.

- If your WHERE clause contains the Boolean expression OR, enclose the predicate in parentheses; for example, `(COL1=X OR COL2=Y)`.
- If the target table is a change aggregate table and contains before-image columns, you must include the before-image columns in a GROUP BY clause, even though these columns are not displayed in the **Available columns** box on the Target Columns page.
- When both of the following conditions are true, you must provide a dummy WHERE clause:
  - You are creating an aggregate column that requires a GROUP BY clause.
  - You do not use any other predicate in the **WHERE** field.

  The Apply program issues error messages if you do not provide the dummy WHERE clause in this situation.

3. To see examples of SQL predicates, click the **Examples** button.

   WHERE clause examples:

   The following examples show WHERE clauses that you can use to filter rows of the target table. These examples are very general and designed for you to use as a model.
   - WHERE clause specifying rows with specific values

     To copy only the rows that contain a specific value, such as MGR for employees that are managers, use a WHERE clause like:
     ```
     EMPLOYEE = 'MGR'
     ```
   - WHERE clause specifying rows with a range of values

     To copy only the rows within a range, such as employee numbers between 5000 and 7000 to the target table, use a WHERE clause like:
     ```
     EMPID BETWEEN 5000 AND 7000
     ```
   - Dummy WHERE clause

     To support aggregation, use a WHERE clause like:
     ```
     1=1
     ```
4. If you are finished using the Advanced Subscription Definition notebook, click **OK** to close the notebook.

To define the target-table rows using DJRA, add a WHERE clause in the **Where clause** field of the Add a Member to Subscription Sets window.

Defining a subscription set with a user-defined table

DB2 DataPropagator allows you to use a previously-defined DB2 table as the target table in a subscription set. That is, you can define a subscription-set member to be a target table that is defined outside of the DB2 Control Center or DJRA. This type of target table is known as a user-defined target table.

Restrictions:

- The subscription-set definition must contain the same number of columns as exist in the user-defined target table.
- New columns in the subscription-set definition must allow nulls and must not have defined default values.
- The target table and unique index must exist when you run the Apply program.

***To define a subscription with a user-defined target table*:**

1. Refer to "Chapter 21. Table structures" on page 299 to determine the structure for the target-table type. For example, if you are defining a subscription for a base aggregate table, refer to the table structure definition for base aggregate tables.
2. Alter the target table to add any required columns, such as timestamp columns.
3. For point-in-time, user copy, replica, and condensed CCD tables, create a unique index.
4. Define the subscription-set member to match the user-defined target table structure, including new columns, subset columns, changed column names, and renamed before-image columns.

   From the DB2 Control Center, in the Define Subscription window:

   a. Clear the **Create table** check boxes next to the table names for which you are providing the target tables.
   b. Type the user-defined target-table name in the **Target table** field.
   c. If you want to subset columns or rows, enhance data, or specify a target-table type other than user copy, click **Advanced** to open the Advanced Subscription Definition notebook.

      If you want to select an alternate table type, see "Choosing a target-table type" on page 99. If you want to modify the target-table columns to match the user-defined target table, or subselect the rows or use an aggregate expression, see "Defining the target-table structure: columns and rows" on page 100.

DJRA tolerates existing target tables, and checks that the columns in the target table match those defined for the subscription-set member.

DB2 DataPropagator does not check for inconsistencies between the subscription definition and a user-defined target table. You must:

- Ensure that there is a target table that matches the subscription definition.
- Debug any inconsistencies that exist between the target table and the subscription definition.

- If you want to use a CCD table between the source table and the user-defined target table, define a subscription to match the attributes specified for the target table. Then define a subscription for the target table in which the CCD table is the source table.
- Ensure that there is a unique index for point-in-time, user copy, replica, and condensed CCD tables.

Defining SQL statements or stored procedures for the subscription set

You can define SQL statements or stored procedures to be run before or after the Apply program copies the data from the source to the target table. For example, you can prune the Apply trail control table each day to remove older entries.

To specify SQL statements or stored procedures for the subscription set using the DB2 Control Center:

1. From the Define Subscription window, click **SQL** to open the SQL window.

   Use the SQL window to add or remove SQL statements or stored procedures that run at the target or source server either before or after the replication subscription is processed. The statements are processed in the order that they appear in the list.
2. Click **Add** to open the Add SQL window.
3. Type the SQL statement or stored procedure name in the **SQL statement or Call procedure** field. The stored procedure name must begin with `CALL`. This field can contain ordinary or delimited identifiers.
4. If you know that the SQL statement or stored procedure will generate SQLSTATEs that would otherwise terminate execution, specify these SQLSTATEs so that the Apply program can bypass them and treat them as successful execution. For example, a DELETE statement will generate a SQLSTATE 02000 when attempting to delete nonexistent rows, but for new tables you might not care about this error.

   Enter valid 5-byte SQLSTATE values in the **SQLSTATE** field and click **Add**. The value is added to the **Acceptable SQLSTATE values** box. You can add up to 10 values.
5. Specify whether you want to run the SQL statement or stored procedure at the source or target server before the subscription set is processed, or at the target server after the subscription set is processed by clicking the appropriate radio button in the **Submit SQL statement** field.
6. Click **OK** to add the statement to the box in the SQL window and close the Add SQL window.

To specify SQL statements or stored procedures for the subscription using DJRA, click **Add Statements or Procedures to Subscriptions Sets**. Fill in the required information to specify the source server and subscription sets to which you are adding SQL statements or stored procedures, then specify the SQL statement or stored procedure, along with acceptable SQLSTATE values, and when it should run within the subscription-set cycle.

Data-sharing considerations

You can implement replication in an S/390 data-sharing environment. In a data-sharing environment, you run one Capture program for each source data-sharing group and one or more Apply programs for each target data-sharing group.

The Capture program can read the data-sharing logs for DB2 for MVS/ESA V4, DB2 for OS/390 V5, or DB2 for OS/390 V6. That is, you can run different versions of DB2 in a data-sharing environment, for example during version-to-version migration, and have one Capture program continue to capture transaction-consistent data. However, this mixed-version environment is not recommended for long-term use, either for replication or for DB2. See the *DB2 for OS/390 Administration Guide* for information about data sharing with mixed versions of DB2.

Specifying a data-blocking value

To specify how many minutes worth of change data DB2 DataPropagator can replicate during a subscription cycle, use the Data Blocking page of the Subscription Timing notebook in the DB2 Control Center, or set the **Blocking factor** in the Create Empty Subscription Sets window in DJRA. The number of minutes that you specify determines the size of the data block. See “Data blocking for large volumes of changes” on page 66 for more information about how to determine this value. DB2 DataPropagator saves this value in the MAX_SYNCH_MINUTES column of the subscription set control table.

Data currency requirements

How up to date do you want target tables to be? How out of date can they be without disrupting the application programs that use them? The answers to these questions reveal your data currency requirements. You can control how often the Apply program processes subscriptions and thereby control the currency of the data. You can set an interval (or relative timing) schedule for the Apply program, or define an event trigger that the Apply program uses to start processing a subscription set.

You define subscription timing with the Subscription Timing notebook in the DB2 Control Center or from the **Subscription set timing** field on the Create Empty Subscription Sets window in DJRA. You can control the timing using time-based or event-based scheduling, or you can use these timing options together. For example, you can set an interval of one day, and also specify an event that triggers the subscription cycle. For update-anywhere replication, you can also specify different timing for source-to-replica and replica-to-source replication.

Recommendation: When moving from a test environment to a production environment, set a mid-range timing value (such as 2 hours) and tune your system from there (to a more frequent or less frequent interval, as appropriate).

Interval timing (relative timing)

The simplest method of controlling subscription timing is to use interval timing. You determine a specific start time, date, and interval. The interval can be specific (from one minute to one year) or continuous, but time intervals are approximate. The Apply program begins processing a subscription set as soon as it can, based on its workload and the availability of resources. If you specify continuous timing, the Apply program replicates data as frequently as it can.

Choosing a timing interval does not guarantee that the frequency of replication will be exactly at that interval. Before specifying an interval, you should determine whether it is possible to refresh all tables in the subscription set within that interval: determine the amount of data that the Apply program is likely to select for each interval and estimate the time that it will take to copy the data.

You can set and change the interval using the DB2 Control Center, DJRA, or by executing SQL statements against the subscription-set control table.

Event timing

To replicate data using event timing, specify an event name when you define the subscription set in the DB2 Control Center or DJRA. You must also populate (using an application program or the DB2 Command Center) the subscription events table with a timestamp for the event name. When the Apply program detects the event, it begins replication.

The subscription events table has three columns, as shown in Table 6.

Table 6. The Subscription Events Table

EVENT_NAME	EVENT_TIME	END_OF_PERIOD
END_OF_DAY	1999-05-01-17.00.00.000000	1999-05-01-15.00.00.000000

EVENT_NAME is the name of the event that you specify while defining the subscription set. EVENT_TIME is the timestamp for when the Apply program begins processing the subscription set. END_OF_PERIOD is an optional value that indicates that updates that occur after the specified time should be deferred until a future time. Set EVENT_TIME using the clock at the control server, and set END_OF_PERIOD using the clock at the source server. This distinction is important if the two servers are in different time zones.

In Table 6 on page 107, for the event named END_OF_DAY, the timestamp value (1999-05-01-17.00.00.000000) is the time when the Apply program is to begin processing the replication subscription. The END_OF_PERIOD timestamp value (1999-05-01-15.00.00.000000) is the time after which updates are not replicated and will be replicated on the next day's cycle. That is, the event replicates all outstanding updates made before three o'clock, and defers all subsequent updates.

Your application programs must post events to the subscription events table to tie your application programs to subscription activity. When you post an entry using CURRENT TIMESTAMP for EVENT_TIME, you trigger the event named by EVENT_NAME. Any subscription set tied to this event becomes eligible to run. You can post events in advance, such as next week, next year, or every Saturday. If the Apply program is running, it starts at approximately the time that you specify. If the Apply program is stopped at the time that you specify, when it restarts, it checks the subscription events table and begins processing the subscription set for the posted event.

Data consistency requirements

When planning and defining a subscription set, you need to be aware of the following rules and constraints:

- If any member of the subscription set requires full-refresh copying for any reason, the entire set is refreshed. For update-anywhere replication, full-refresh copying occurs only from the replication source to the replica, not from replica to source.
- A single synchpoint is maintained for the subscription set to indicate the copy progress for the entire subscription set.
- You must refresh target tables that have referential-integrity constraints using the ASNLOAD exit routine to bypass referential-integrity checking.
- Do not define referential constraints for read-only target tables.
- The first occurrence of any referential-integrity violation terminates the current replication cycle. The subscription cycle is automatically retried, after the transaction is rejected and compensated.
- For update-anywhere replication, you must specify a collision-detection level when you define a replication source.

- For update-anywhere replication, each replica table must be of the same generation as all other replicas in the subscription set, and they must all have the same source table.
- For update-anywhere replication, because no single application program updates both source and target tables, referential-integrity violations cannot be detected in application logic, so you must use declarative referential-integrity constraints.
- The administration tools do not copy referential-constraint definitions from a source table to target tables. For update-anywhere replication, you must include all referential constraints that exist among the source tables in the replica tables to prevent referential-integrity violations. If you omit some referential constraints, an update made to a replica table could cause an referential-integrity violation when it is replicated to the source table.
- For a subscription that includes external CCDs, all the external CCDs must have a common original source database.

If you maintain your own CCD table, you must update three columns in the register control table: CCD_OLD_SYNCHPOINT, SYNCHPOINT, and SYNCHTIME:

CCD_OLD_SYNCHPOINT
: The synch point associated with the oldest row remaining in the CCD table.

  Before a full refresh of the CCD table, set CCD_OLD_SYNCHPOINT to NULL.

  After a full refresh of the CCD table, set the CCD_OLD_SYNCHPOINT to a value greater than the previous value of SYNCHPOINT. If SYNCHPOINT has no previous value (in the case of the initial load), set the CCD_OLD_SYNCHPOINT to X'00000000000000000000'.

SYNCHPOINT
: A sequence value useful for maintaining the state of CCD copies, subscription states, and for controlling pruning.

  Set the SYNCHPOINT for the CCD table to MAX(IBMSNAP_COMMITSEQ) whenever you commit new changes to the CCD table. Be sure also to set SYNCHTIME accordingly.

SYNCHTIME
: The timestamp-equivalent of SYNCHPOINT.

Loading target tables offline using DJRA

DJRA guides you through the process of creating an offline load of a table or database. The following procedure does not maintain the additional control information required for loading external CCD tables, so for these tables you must use the manual procedure.

To perform an offline load using DJRA:

1. Ensure that the Capture program is running.
2. Click **Off-line load** on the DJRA main window.
3. In the Off-line load — STEP 1 window, generate the SQL statements to disable current subscriptions for the selected subscription sets. After you load the target tables, you will re-enable these subscriptions.
4. Unload the source tables. Then click **Next Step** on the Off-line load — STEP 2 window.
5. Load the target tables. Then click **Next Step** on the Off-line load — STEP 3 window.
6. In the Off-line load — STEP 4 window, generate the SQL statements to re-enable current subscriptions for the selected subscription sets. Then click **Finished**.

Copying your replication configuration to another system

When you define tables, registrations, or subscription sets on one system (a test system, for example), and you need to copy the replication environment to another system (a production system, for example), you can use the DJRA promote functions. These functions reverse engineer your tables, registrations, or subscription sets, to create script files with appropriate data definition language (DDL) and data manipulation language (DML). Table 7 shows the three promote functions.

For example, use the promote functions to define subscription sets for remote DB2 Personal Edition target databases. After you define a model target system in your test environment, you can create subscription-set scripts (and modify which Apply qualifier is used and so on) for your DB2 Personal Edition systems, which are not otherwise supported from a central control point.

Table 7. Promote Functions

Promote table	This function promotes tables, table spaces, and indexes. It does not promote constraints defined for a table. This function is fully supported for DB2 UDB V5 and later, but for the IBM Common Server you can promote only tables, not table spaces.

Table 7. Promote Functions (continued)

Promote registration	This function promotes registrations and view registrations from a source server.
Promote subscription	This function promotes subscriptions: subscription sets, subscription-set members, subscription columns, subscription prune control, and subscription statements. It enables you to create a new subscription set from an existing one. From the Promote Subscriptions window in DJRA, you can change your subscriptions (before promoting them) by setting new values for any of the following fields: **Apply Qualifier**, **Set Name**, **Source server**, **Source alias**, **Target server**, **Target alias**, **Control server**, and **Control alias**.

Setting up the Capture program

This section describes setting the Capture program generally. See the appropriate chapter for your operating system environment (in "Part 3. Operations" on page 133) for specific information about setting up the Capture program.

Specifying tuning parameters for the Capture program

To control the performance of the Capture program, you can specify the following tuning parameters in the tuning parameters table:

Retention limit
The number of minutes to keep the CD table rows and the unit-of-work (UOW) table rows. The default value is 10 080, which is 7 days.

For AS/400, you can keep the size of tables smaller by reorganizing them using the RGZPFM command.

Lag limit
The number of minutes that the Capture program can be backlogged from the current local time before shutting itself down. The default value is 10 080, which is 7 days. If this value is exceeded, the Apply program will perform a full refresh of all target tables.

Recommendation: Set a high value for the lag limit to ensure the Capture program does not shut itself down unnecessarily.

If the database does not have or support an archive log, and the Capture program shuts itself down, you should perform a cold start of the Capture program.

Commit interval

The number of seconds that the Capture program waits before issuing a COMMIT statement. The default value is 30 seconds.

If the Apply program is *not* running at the same time as the Capture program, you can set the commit interval no higher than the DB2 timeout interval.

For AS/400, this value has a different meaning: it is the number of seconds between the time that an application program updates a source table and the time that the corresponding update in the CD table is written to disk. The commit interval can range from 30 to 600 seconds, and the default is 180. If the value is too small, overall system performance can be degraded.

Prune interval

The number of seconds that the Capture program waits before pruning the staging tables. The default value is ten times the commit value, or 300 seconds, whichever is larger. This parameter is ignored if you start the Capture program with the NOPRUNE option; however, you can override this option by using the **prune** command.

For AS/400, you can override this value by specifying a value for the wait-time subparameter of the CLNUPITV keyword of the **STRDPRCAP** command. If you specify *NO on the start-clean-up subparameter of the CLNUPITV keyword, the prune interval value is completely ignored.

For AS/400, if you start up the Capture program daily, *NO allows you to defer pruning (for example, to the weekend). During the week, you can use CLNUPITV (*DPRVSN *NO) on the **STDPRCAP** command. On weekends, you can use CLNUPITV (*DPRVSN *IMMED), which is the default.

To specify the tuning parameters, do one of the following tasks:

- Customize the DPCNTL.* file in the DB2 \sqllib\samples\repl directory before you define the first replication source for a database.[15]
- If you want to change the default values, update the tuning parameters table with the following SQL statement after you create it :

```
UPDATE TABLE ASN.IBMSNAP_CCPPARMS
SET RETENTION_LIMIT=number_of_minutes,
LAG_LIMIT=number_of_minutes,
COMMIT_INTERVAL=number_of_seconds,
PRUNE_INTERVAL=number_of_seconds
```

15. Customizing DPCNTL.400 is not necessary if you already installed DataPropagator Relational for AS/400.

If you need to change the values and refresh the tuning parameters while the Capture program is running, enter the **reinit** command after changing the table values. For AS/400, enter the **INZDPRCAP** command; for more information about the **INZDPRCAP** command, see “Reinitializing Capture for AS/400” on page 174.

For information on the structure of the tuning parameters table, see “Chapter 21. Table structures” on page 299.

- Run the AS/400 **CHGDPRCAPA** command. See “Changing Capture program attributes” on page 151 for more information about this command.

Restrictions when running the Capture program

The following actions cause the Capture program to terminate while it is running. Stop the Capture program if you want to perform any of the following tasks:

- Remove an existing replication source table. If you remove an existing replication source, the Capture program might attempt to insert data into a CD table that no longer exists.
- Make changes (other than ALTER ADD changes) that affect the structure of source tables or CD tables. This includes changes resulting from data definition language or utilities. Structural changes can compromise the data integrity of the copies.

Other Capture program restrictions include:

- If the Capture program shuts itself down, perform a cold start if the database does not have or support an archive log.
- Source tables must be created or altered using the DATA CAPTURE CHANGES option. The DB2 Control Center and DJRA automatically specify these keywords when you define replication sources.
- Capture does not capture any changes made by DB2 utilities, even if you specify LOG=YES.

Setting up the Apply program

This section describes setting the Apply program generally. See the appropriate chapter in “Part 3. Operations” on page 133 for your operating system environment for specific information about setting up the Apply program.

Refreshing target tables with the ASNLOAD exit routine

The Apply program can call the ASNLOAD exit routine whenever it performs a full refresh of a target table. Specify the LOADX parameter to cause the Apply program to call this routine.

You can use the ASNLOAD routine as shipped with the Apply program, or you can modify it. As shipped, the routine uses the DB2 EXPORT utility to export data from the source table and uses the DB2 LOAD utility to fully refresh the target table. You can modify the ASNLOAD routine to call any IBM or vendor utility. See the prolog section of the sample program (ASNLOAD.SMP) in the \sqllib\samples\repl directory for information about how to modify this exit routine.

You must use the ASNLOAD routine to fully refresh tables with referential integrity constraints in order to bypass referential integrity checking.

If your source servers are password protected, you must modify the ASNLOAD routine to provide the password file. However, if the password is administered by DB2 Universal Database Satellite Edition, the ASNLOAD routine does not require a password file, and you can use the IBM-supplied routine.

See "Refreshing target tables with the ASNLOAD exit routine for AS/400" on page 192 for information about using the ASNLOAD routine in an AS/400 environment.

Files generated on Windows and UNIX

When you run the ASNLOAD routine, it generates the following files:

- ASNA<*userid*><*database_instance_name*><*cntl_server*>.IXF

  This file contains the data exported from the source.
- ASNAEXPT<*userid*><*database_instance_name*><*cntl_server*>.MSG

  This file contains error, warning, or informational messages issued by the EXPORT APIs.
- ASNAIMPT<*userid*><*database_instance_name*><*cntl_server*>.MSG

  This file contains error, warning, or informational messages issued by the LOAD APIs.

Files generated on OS/2

When you run the ASNLOAD routine, it generates the following files:

- <*apply_qual*>.IXF

  This file contains the data exported from the source.

- *<apply_qual>*.EXP

  This file contains error, warning, or informational messages issued by the EXPORT APIs.
- *<apply_qual>*.LOA

  This file contains error, warning, or informational messages issued by the LOAD APIs.

Error handling

If an error occurs while the Apply program calls the ASNLOAD routine, or if the routine returns a nonzero return code, the Apply program issues a message, stops processing the current subscription set, and processes the next subscription set.

Using the ASNDONE exit routine

The Apply program can optionally call the ASNDONE exit routine after subscription processing completes, regardless of success or failure. You can modify this routine as necessary, for example, the routine can examine the UOW table to discover rejected transactions and initiate further actions, such as issuing a message or generating an alert. Another use for this exit routine is to deactivate a subscription set that fails (status = -1), and thus avoid retry by the Apply program until the failure is fixed.

See the prolog section of the sample program (ASNDONE.SMP) in the \sqllib\samples\repl directory for information about how to modify this exit routine. For AS/400, the following table indicates where you can find the source code for this routine:

Compiler language	Library name	Source file name	Member name
C	QDPR	QCSRC	ASNDONE
COBOL	QDPR	QCBLLESRC	ASNDONE
RPG	QDPR	QRPGLESRC	ASNDONE

See “Using the ASNDONE exit routine for AS/400” on page 191 for more information about using the ASNDONE exit routine in an AS/400 environment.

To use the ASNDONE exit routine:

1. Modify the ASNDONE routine to meet your site’s requirements.
2. Compile the program and place the executable in the appropriate directory.
3. Start the Apply program with the NOTIFY parameter to call the ASNDONE exit routine.

The parameters that the Apply program passes to the ASNDONE exit routine are:

- Set name
- Apply qualifier
- Value for the WHOS_ON_FIRST column in the subscription set control table
- Control server name
- Trace option
- Status value

Chapter 7. Operating DB2 DataPropagator

This chapter describes how to operate the Capture and Apply programs generally. For information about operating either of these programs (for example, starting, stopping, or scheduling) in a specific operating system environment, see “Part 3. Operations” on page 133. This chapter also describes regular database maintenance, monitoring replication, handling gaps, modifying your replication configuration, and troubleshooting.

Operating the Capture program

This section describes what you need to know before you start the Capture program, when you should perform a warm or cold start of the Capture program, and how to stop the Capture program with an event.

Before you start the Capture program

Before starting the Capture program, make sure that you complete the following post-installation tasks:

- Define one or more replication sources and subscriptions, as described in “Defining replication sources” on page 92 and “Defining replication subscriptions” on page 97. Defining replication sources and subscriptions creates the following control tables in the source server (where the Capture program runs):
  - ASN.IBMSNAP_REGISTER (register table)
  - ASN.IBMSNAP_PRUNCNTL (pruning control table)
  - ASN.IBMSNAP_CCPPARMS (tuning parameters table)
  - ASN.IBMSNAP_TRACE (trace table)
  - ASN.IBMSNAP_WARM_START (warm start table)
  - ASN.IBMSNAP_UOW (unit-of-work table)
  - ASN.IBMSNAP_CRITSEC (critical section table)
  - ASN.IBMSNAP_PRUNE_LOCK (prune lock table)

  Defining replication sources and subscriptions creates the following control tables in the control server:
  - ASN.IBMSNAP_SUBS_SET (subscription set table)
  - ASN.IBMSNAP_SUBS_MEMBR (subscription-targets-member table)
  - ASN.IBMSNAP_SUBS_STMTS (subscription statements table)
  - ASN.IBMSNAP_SUBS_COLS (subscription columns table)

  - ASN.IBMSNAP_SUBS_EVENT (subscription events table)
  - ASN.IBMSNAP_APPLYTRAIL (Apply trail table)
  - REG_EXT (replication source extension table, for AS/400 only)
  - AUTHTKN (Apply-qualifier-cross-reference table, for AS/400 only)
  - APPLY_JOB (Apply job table, for AS/400 only)

  You can also create these control tables manually by running the DPCNTL.* file from the RUN SQL Files window.
- Bind the Capture program to the source server from which the Capture program will capture changes. Capture for OS/390 includes bind JCL. Refer to the Capture for OS/390 program directory for information on running the bind program for Capture. For other operating systems, see the configuration section within the operating system-specific chapters (see "Part 3. Operations" on page 133).
- **For OS/390**: Ensure that you apply all relevant PTFs for your DB2 subsystem.
- Ensure that the register table has at least one entry in it by defining a replication source with the DATA CAPTURE CHANGES option. See "Defining replication sources" on page 92 for more information.
- **For Windows NT:** Decide whether you want to use the NT Service Control Manager (SCM) to automatically run the Capture program as an NT service. See "Starting Capture for Windows and OS/2" on page 219 for information about operating the Capture program as an NT service.
- **For VM:** A Capture program ASNPARMS file is provided that contains default values that the Capture program uses. Changing this file modifies the defaults. If you need different values for a specific database, copy the file to the Capture program virtual machine's A-disk. The following default values are contained in the Capture program ASNPARMS:
  - ENQ_NAME CAPTURE
  - LANGUAGE ASNLS001
- **For VSE:** The ASNS51CD job control member contains the name of the messages file used. The default is American English. Modify ASNS51CD to issue messages in a different language. See the Capture for VSE program directory for a list of supported languages.
- **For VM and VSE:** By default, the messages are issued in American English (ASNLS001). To issue messages in a different language, change the LANGUAGE parameter in the ASNPARMS file. See the program directory for the Capture program for a list of supported languages.

Starting or restarting the Capture program

When you start or restart the Capture program, you can include any of the following keywords: COLD, WARM, or WARMNS. If you are starting the Capture program for the first time, specify either COLD or WARM to cold start the Capture program. If you are restarting the Capture program after being shut down or after a failure, specify either WARM or WARMNS to warm start the Capture program. The following sections describe cold and warm starts, including how the Capture program handles warm starts, when it switches to an automatic cold start, and how to prevent an automatic cold start by forcing a warm start.

Cold start

When you cold start the Capture program, it deletes all rows from the CD tables and the UOW table and begins reading the end of the database log. Specify a cold start by including the COLD keyword when you start the Capture program. A warm start can also become a cold start in certain circumstances; see “Automatic cold start” on page 120.

After a cold start, the Apply program performs a full refresh of the target tables. You can specify the LOADX keyword when you start the Apply program to improve the performance of the full refresh, or you can use the technique described in “Loading target tables offline using DJRA” on page 110.

Warm start

When you stop the Capture program or if it fails, it writes information in the warm start control table to enable a warm start. There are cases when the Capture program cannot save warm start information. For example, an operator might cancel the Capture program or stop DB2. In this case, the Capture program uses information in the CD, UOW, or register tables to resynchronize to the time it stopped and thus allow a warm start.

When you restart the Capture program with the WARM or WARMNS keywords, it looks in the warm start table (or in the CD, UOW, or register tables) to determine if it can warm start or if it must cold start. If there is sufficient warm start information, the Capture program warm starts, otherwise it attempts a cold start; see “Automatic cold start” on page 120.

After a successful warm start, the Capture program deletes old rows in the warm start table.

Automatic cold start

If the Capture program cannot warm start, it attempts to perform a cold start. But, if you specify the WARMNS keyword, the Capture program does not cold start. The Capture program automatically switches to a cold start when:

- The warm start log sequence number lags behind the current log sequence by more than the LAG_LIMIT value (as specified in the tuning parameters table) or is not available from the database log.
- You start the Capture program for the first time.

  The first time you start the Capture program, you see message ASN0102W, indicating that warm start failed. The Capture program switches to a cold start. You can ignore this message when first starting the Capture program.

In each of these cases, the Capture program issues an informational message and performs a cold start. This cold start also causes a gap in the change data capture sequence because the Capture program jumps to a new position in the database log.

Preventing automatic cold start

To prevent the Capture program from attempting a cold start, specify the WARMNS keyword when starting the Capture program. If the warm start is not possible, instead of cold starting, the Capture program terminates. When the Capture program terminates in this way, the control tables remain intact. You must correct the problem that caused the Capture program to terminate before you attempted to restart it. If you do not correct the problem, the Capture program continues to terminate or performs a cold start every time that you restart it.

Stopping the Capture program with an event

In some situations, you might want to stop the Capture program when a particular event happens. For example, you might want to stop it at a specific time, or after all transactions for a particular application have committed. Create a table (or use an existing table), define it as a replication source, and identify a particular column whose update, when captured, causes a shutdown. You can also create an application program that periodically reads the SYNCHTIME value in the global row of the register control table.

Restriction: Your installation might not allow an application program to stop the Capture program.

Operating the Apply program

This section describes what you need to know before you start the Apply program and how to use the Apply program for forward recovery. For information about operating the Apply program (for example, starting, stopping, or scheduling) in a specific operating system environment, see "Part 3. Operations" on page 133.

Before you start the Apply program, ensure that:

- The control tables are defined.
- **For Windows and UNIX:** A password file has been created, if necessary, for end-user authentication at the source server. See "Providing end-user authentication at the source server" on page 215 for more information.
- At least one subscription is created and activated.
- The Apply program package is created.

  See the Apply for OS/390 program directory for information on BIND programs to create the Apply program packages. You must bind the Apply program to both the source and target databases.

  See "Configuring Apply for Windows and OS/2" on page 214 for information on bind programs to create the Apply packages for Windows and OS/2.

  See "Configuring the Apply program for UNIX platforms" on page 198 for information on bind programs to create Apply packages for UNIX platforms.

  See "Creating DPROPR/400 packages to use with remote systems" on page 178 for information about the CRTDPRPKG command, which creates the packages that are necessary for DataPropagator Relational for AS/400 to work with remote systems.
- The Capture program is started, and the ASN0100I initialization message was issued (if you are running a Capture program).
- You have proper authorization.
- **For OS/390:**
  - Ensure that all STEPLIB libraries of the Capture for MVS RUN JCL (if you are running Apply for OS/390) are APF-authorized.
  - Customize and execute the following JCL:
    - The link-edit sample JCL
    - The sample JCL to build the VSAM message file
    - The Apply bind sample JCL
    - The Apply Run/Invocation sample JCL

  - You must have SYSADM or DBADM privileges at the source, control, and target server. You must also have the proper authorization to run the Apply program, including EXECUTE privileges for Apply program packages.
  - You must have DBADM or CONTROL or SELECT authority that meets all requirements for defining a replication source and target.
- **For AS/400:** For Version 5, the Capture program is started on the source server before you start Apply for the first time. The Capture program updates the SYNCHTIME and SYNCHPOINT columns in the ASN/IBMSNAP_REGISTER table before the Apply program is started. The Apply program assumes that if a GLOBAL record is present in the ASN/IBMSNAP_REGISTER table, the SYNCHTIME and SYNCHPOINT columns are not null. For Version 1, the Apply program can be started before the Capture program.

Performing regular database maintenance

In addition to the regular maintenance that you perform for your databases, running replication requires you to perform the following maintenance:

- Run the REORG utility for CD tables and the unit of work table

  You should reorganize the CD tables and UOW table about once a week if they are heavily used. For DB2 for OS/390 Version 5 or higher, specify the PREFORMAT keyword. Preformatting the table space speeds up the Capture program's insert processing. If the table space is compressed, you must also specify the KEEPDICTIONARY keyword.
- Run REORG for target tables

  Because subscription predicates are very selective and can filter out a majority of the transaction updates, there is no general rule for how frequently you should reorganize target tables, but it should be at least as frequently as you reorganize the source tables.
- Delete old rows in the Apply trail control table (ASN.IBMSNAP_APPLYTRAIL)

  At the end of each subscription cycle, the Apply program inserts a row into the Apply trail control table. To prevent this table from growing too large, you need to delete these rows regularly. You can delete these rows whenever you like because although the Apply program writes to this table, it does not read from it. The subscription statistics and error diagnostics written to this table are for your benefit, and they are used by the Replication Monitor. An easy way to manage the growth of this table is to add an SQL statement to the subscription set; for example:

  ```
  DELETE FROM ASN.IBMSNAP_APPLYTRAIL
  WHERE LASTRUN < (CURRENT TIMESTAMP - 7 DAYS);
  ```
- Prune CCD tables

Neither the Capture program nor the Apply program automatically prunes CCD tables, and there is no command for pruning these tables. Condensed CCD tables are updated in place, so do not grow continually. Noncondensed CCD tables contain history, which you likely want to retain.

A condensed, noncomplete, internal CCD table grows, and with enough update activity, can approach the size of a complete CCD table. Because only the most recent changes are retrieved from it, there is no value in letting this table grow. To prune transactions that have already been replicated from this table, add an SQL statement to the internal CCD's subscription; for example:

```
DELETE FROM my.internal_ccd
WHERE IBMSNAP_COMMITSEQ <= (SELECT MIN(SYNCHPOINT) FROM ASN.IBMSNAP_PRUNCNTL);
```

This statement prunes the table based on the least active subscriptions, not just those subscriptions that refer to the source associated with the internal CCD, so you might want to modify the statement for more aggressive pruning.

The following operational procedures typically require exclusive use of DB2 table spaces or the catalog:

REORG

BIND PACKAGE

BIND PLAN

GRANT

REVOKE

Because these operational procedures do not coexist well with the Capture and Apply programs' issuing dynamic SQL (which implicitly locks catalog tables) or accessing table spaces, you should stop both the Capture and Apply programs when running utilities (and other similar operational procedures) to avoid any possible contention.

Monitoring the replication environment

You can use the Replication Monitor, included with the DJRA, to periodically generate a report that shows how your replication network is working:

- The color-coded report shows you whether your expectations for Capture latency and subscription latency are being met.
- The Replication Monitor keeps historical statistics, so you can determine whether end-to-end subscription latencies, for example, are staying consistent.
- The report file is in HTML format, viewable through your Web browser. You can publish the report on your company's intranet for the benefit of

users who need the information but might not have database privileges. After it is produced, the monitor report is completely independent of DB2, requiring no database accesses to read.

To start the Replication Monitor, click **Monitor replication** on the main DJRA window. From the Replication Administration Scheduler window, you can schedule the monitor to run periodically or you can run it once.

Resolving gaps between source and target tables

Occasionally, a gap can occur during the capturing of changed data for a source table. For example, if you shut down the Capture program and then cold start it, it deletes all rows from the CD table. In this case, updates might be made that the Capture program does not capture. Or, any updates that were in the CD table could have been deleted (by the cold start) *before* the Apply program could replicate them.

When a gap exists, the Apply program attempts a full refresh unless the target table is noncomplete. If the Apply program cannot perform a full refresh, data integrity could be lost.

Modifying your replication configuration

After you begin replication, you can change the configuration, including changing replication sources or subscriptions, removing sources or subscriptions, deactivating subscriptions, and cloning subscriptions.

Viewing or changing existing replication sources

Using either the DB2 Control Center or DJRA, you can view an existing replication source. With the Control Center, if you selected the **Table will be used for update anywhere** check box, you can change the conflict detection level defined for the replication source. All other fields and controls are unavailable for changes after you successfully define the replication source. With DJRA, you can change the set of columns available for replication.[16]

If you plan to change the replication source definition, stop or suspend the Capture program. To begin capturing changes for the changed replication source, start or reinitialize the Capture program. For information about the Capture program for your operating system environment, see “Part 3. Operations” on page 133.

16. You can change the set of columns available for replication only for DB2 sources, not for non-IBM sources.

Removing replication sources

When you no longer need a replication source, you can remove the object from the DB2 Control Center or DJRA and remove its control information from the control tables.

Attention:

1. Stop the Capture program before deleting a replication source. Do not merely suspend it. You can restart the Capture program after you have finished removing the replication source.
2. Although the Control Center removes dependent subscriptions, you should check whether a dependent subscription table is being used as a source for another subscription and cancel any such dependent subscription before you delete a replication source. DJRA does not remove dependent subscriptions, so you must delete any subscriptions that use the source for copying.

The Control Center and DJRA drop the table space for a DB2 replication source if it is empty. DJRA does not drop non-IBM database containers (table spaces, dbspaces, or segments). For the Control Center, you can ensure that the table space is never dropped by changing the settings on the Replication page of the Tools Settings notebook.

Activating and deactivating subscription sets

From the DB2 Control Center, you can control the active status of a subscription set. This feature is useful when you want to temporarily deactivate a subscription set without removing it. When you deactivate a subscription set, the Apply program completes its current processing cycle and then suspends operation for that subscription set. When you deactivate a subscription set, the Control Center greys out the icon for the subscription set.

Cloning a subscription set to another server

Using the DB2 Control Center, you can clone a subscription set to another server. Cloning creates a copy of an existing subscription set on a different target server, using a different Apply qualifier. This copy includes only subscription information; it does not include copy table, table space, or index definitions. You can clone one or more subscription sets at a time. The Control Center updates the control tables at the control server.

For information about copying your entire replication environment to another system, see “Copying your replication configuration to another system” on page 110.

Viewing or changing an existing subscription set

Using the DB2 Control Center, you can change a subset of the subscription-set values, primarily those that do not affect the structure of the target tables. You can change the following values in the Change Replication Subscription window and subwindows:

- The **Row predicate** in the Advanced Subscription notebook. The new predicate is not applied to existing rows in the target table. The Apply program uses the predicate starting with the next subscription cycle for the subscription set. See "Defining the target-table rows" on page 102 for more information and examples.
- The SQL or CALL procedure for before or after copying in the SQL window. See "Defining SQL statements or stored procedures for the subscription set" on page 105 for more information.
- The timing values in the Subscription Timing notebook.
- The data-blocking value in the Subscription Timing notebook. See "Specifying a data-blocking value" on page 106 for more information.

To view or change existing subscription-set members using DJRA, click the **List Members or Add a Column to Target Tables** button. Fill in the required information in the window, such as source-server name and source-table names, then optionally fill in the source-column name or SQL expression and the target-column name to add new columns or add computed columns to the target table.

Removing subscription sets

Removing a subscription-set definition deletes information about it from the control tables and deletes the target table from the target server. Using the DB2 Control Center, select one or more replication subscription objects from the contents pane and select **Remove** from the pop-up menu. Using DJRA, you must first remove all members from the subscription set, then you can remove the empty subscription set.

Troubleshooting

This section describes various problems that can occur when running the Capture and Apply programs and how to diagnose the cause of these problems. You should also use the Replication Analyzer to determine general and specific problems with the replication environment and see "Chapter 22. Problem determination facilities" on page 357.

Problem: *The Capture for OS/390 program does not start.*

Ensure that APF authorization was performed for all STEPLIB libraries as specified in the RUN JCL.

Problem: *Capture for VM or Capture for VSE does not start.*

Ensure that:

- Access was given for the database log and directory minidisks. Note that the Capture program issues internal links to these minidisks.
- Access was given to the C Run Time Library.
- *IDENT authorization was given to the Capture program virtual machine.
- The ASNLMAIN package file was loaded to the database.
- **For VM:** *IDENT authorization was given to the Capture program virtual machine.

Problem: *The Capture program is not capturing updates.*

Any of the following errors could prevent the Capture program from capturing updates:

- Proper authorization was not granted to the user ID running the Capture program.
- DATA CAPTURE CHANGES was not specified on the source tables to be captured. At startup, the Capture program checks that registered tables have DATA CAPTURE CHANGES specified. If tables are altered after the Capture program is running, the Capture program will not find the error situation.
- The proper order for starting the Capture and Apply programs was not used:
  1. Define replication sources and subscriptions before starting the Capture program.
  2. Start the Capture program and look for message number ASN0100I (initialization completed) in the system console or in the trace table.
  3. Start the Apply program.

  Check the trace table for possible error messages.

Problem: *I'm not sure if the Capture program is running successfully.*

The first time that you start the Capture and Apply programs, the Apply program performs a full refresh to populate the target tables. Then the Capture program writes message ASN0104I to the trace table, providing information related to table owner name, table name, and starting log sequence number value. This provides a point from which the Capture program starts to capture updates.

Updates captured from then on are placed in CD tables. They are eventually applied to target tables and pruned from the CD tables. After the Capture program runs for some time, you should see rows in the CD tables if changes are made to the sources. Periodically, check the trace table to see the progress made by the Capture program. If it encounters errors, it sends them to the console and also logs them in the trace table. Similarly, the Apply program logs its information in the Apply trail table.

Problem: *Capture for OS/390 issued message ASN0000E instead of the proper message number.*

Message ASN0000E is a generic message that is issued instead of a proper message if the specified VSAM message file in the RUN JCL was not found. See the Capture for OS/390 program directory for information on installing the VSAM message file.

Problem: *Capture for VM or Capture for VSE issued message ASN0000E instead of the proper message number.*

Message ASN0000E is a generic message that is issued instead of a proper message if either the default message file, ASNLS001 MSG, or the specified message file in the Capture startup JCL was not found. See the Capture for VM or Capture for VSE program directory for information on installing the message file.

Problem: *The Capture program terminates.*

The Capture program terminates either because of a severe error, or when you issue the **stop** command. The Capture program terminates with a return code that indicates successful or unsuccessful completion. Return codes are:

0	**stop** command issued
8	Error during initialization
12	Any other severe error

See "Capture program messages" on page 371 for an explanation of this message.

Problem: *Updates to the OS/390 source table are not being replicated to the target table.*

DB2 for OS/390 writes log information to the active log in multiples of 4 KB VSAM control intervals (CI). For the Capture program to obtain these log records, there must be at least one CI written to the log. If your application has source table updates less than 4 KB in a quiet DB2 (no other log updates), you do not see the update results in the CD table until the 4 KB limit is reached. To see your updates you can use -ARCHIVE LOG to flush out updates immediately.

Problem: *Capture for OS/390 failed while using the LE/370 environment.*

The Capture program can run in both C/370 and LE/370 environments. If running in the LE/370 environment, you must specify REGION=10M in the RUN step.

Problem: *Error message 0509 was issued.*

Error message 0509 occurs because multiple versions of DB2, or DB2 and DataJoiner, are installed on the same system:

- 0509-0306 Cannot load program asnccp for the following errors:
- 0509-0222 Cannot load the library libdb2.a(shr.o)
- 0509-0026 System error: a file or directory does not exist

Ensure that the LIBPATH environment variable is set to the same environment in which the Apply program starts.

Problem: *Apply component for DB2 Universal Database stops with an SQLCODE= -330, SQLSTATE=22517, "A string cannot be used, because its characters cannot be translated".*

When copying between DB2 for OS/390 and DB2 Universal Database, the CCSID translation can cause an INSERT to fail if a translated value is longer than the DB2 column in which it will be inserted. The Apply program can generate an SQLCODE -330 when it tries to insert a translated COPY_TABLE column value from the refresh control table on a DB2 Universal Database copy server into the pruning control table on a DB2 for OS/390 source server.

For example, if you use the Korean character set with mixed data at both a DB2 for OS/390 source server and a DB2 Universal Database target server, the INSERT fails because the original string is in mixed-data ASCII. When it is translated to EBCDIC mixed data with the Korean character set, if the resulting string length is greater than 18 characters (the maximum COPY_TABLE length), the INSERT fails with an SQLCODE of -330.

If you are running in a mixed environment, ensure that you have installed the latest maintenance for the CCSID support of your DB2 for OS/390 program.

For more information on character translation, see the character conversion appendix in the *DB2 for OS/390 Version 6: Installation Guide*.

Problem: *I received system error 1067 trying to start the Capture or Apply program as NT Service.*

Error code 1067 occurs under the following circumstances:

- You did not specify the user ID and password to run under and the Capture or Apply program is trying to run on the system account.
- You did not specify the ASNPATH environment variable correctly or you did not reboot the computer after updating the value of ASNPATH.
- There is no NTSERV.ASN file in the path specified by ASNPATH.
- The NTSERV.ASN file does not have the line:

  ```
  dbname pathname\asnccp.exe <parameters>
  ```

  followed by CRLF.

Problem: *The ASNSERV.LOG file in ASNPATH tells me that the Apply program was started correctly, but the Apply process terminated.*

To find out why the Apply program terminated, change the syntax of NTSERV.ASN to:

```
...ASNAPPLY USERQUAL TRCFLOW
```

The trace output will be written to the Apply trace file:

```
<ASNPATH pathname>APPLY<timestamp>.TRC
```

Problem: *I performed a successful bind, but when running the Apply program, I still get SQLCODE -805, SQLSTATE 51002.*

Make sure that the user ID has EXECUTE privilege on the Apply program packages, and make sure to bind both the Apply program packages to the control, source, and target server databases.

Problem: *The DB2 log filled to capacity because I copied a very large table.*

If the error occurred during a full refresh, you can use alternative methods to load large tables. You can either use the ASNLOAD exit routine, or you can perform your own load, as described in “Loading target tables offline using DJRA” on page 110.

If the error occurred while applying changed data, then you can change the data-blocking parameter to break down large blocks of changed data. See “Specifying a data-blocking value” on page 106.

Problem: *Capture was cold started, which caused the Apply program to perform a full refresh, but I don’t want a full refresh.*

If your target table is very large, and in cases where you decided to use only your own load mechanism, you might want to suppress any future full refreshes of the Apply program. Set the DISABLE_REFRESH flag to 1 in the register table at the source server for the source table. In this case, the Apply program issues message ASN1016E and does nothing.

Problem: *A gap was detected, so the Apply program won't perform a full refresh of my target table.*

Force a full refresh by resetting the LASTSUCCESS, SYNCHTIME, and SYNCHPOINT values in subscription set table to null.

Problem: *I unsuccessfully tried to start a second Apply program instance.*

You must run each instance with a unique Apply qualifier.

Problem: *I received a security violation message, and the Apply program is not authorized to connect to the database.*

The control server name, user ID, and password definitions are case sensitive and must match exactly those specified in the password file. Check your definitions again.

Apply for AS/400 does not use a password file, so it attempts to connect to the database using the user ID specified in the user parameter of the STRDPRAPY CL command. Ensure that you correctly set up your DRDA connectivity definitions.

Problem: *I received error ASN1003 with SQLCODE = -1032 and SQLSTATE = 57019.*

You must start the database manager before invoking the Apply program.

Part 3. Operations

This part of the book describes how to operate the Capture and Apply programs for a particular operating system:

"Chapter 8. Capture and Apply for OS/390" on page 135 describes how to operate the Capture and Apply programs on the OS/390 operating system.

"Chapter 9. Capture and Apply for AS/400" on page 149 describes how to operate the Capture and Apply programs on the AS/400 operating system.

"Chapter 10. Capture and Apply for UNIX platforms" on page 197 describes how to operate the Capture and Apply programs on UNIX operating systems.

"Chapter 11. Capture and Apply for Windows and OS/2" on page 213 describes how to operate the Capture and Apply programs on the Windows and OS/2 operating systems.

"Chapter 12. Capture for VM and Capture for VSE" on page 229 describes how to operate the Capture program on the VM and VSE operating systems.

Chapter 8. Capture and Apply for OS/390

This chapter describes how to set up and operate the Capture and Apply programs for OS/390. It also contains information specific to replicating DB2 for OS/390 data:

- “Rules for index types” on page 143
- “Using the DB2 ODBC Catalog” on page 143

Setting up the Capture and Apply programs

Setting up consists of installing the Capture and Apply programs and configuring the source, target, and control servers. Capture for OS/390 and Apply for OS/390 are packaged in SMP/E format. The installation sequence for each program consists of:

1. Customizing invocation JCL to suit your environment
2. Using SMP/E to install
3. Providing APF authorization
4. Creating and loading the VSAM messages file
5. Binding to the DB2 subsystem and target (control) subsystems that the DB2 subsystem will connect to

See the *Capture for OS/390 Program Directory* for Capture program installation instructions. See the *Apply for OS/390 Program Directory* for Apply program installation instructions.

Make sure to apply the correct DB2 maintenance before installing the Capture and Apply for OS/390 programs. The correct DB2 maintenance consists of:

- The DB2 maintenance instructions in the *Capture for OS/390 Program Directory* and the *Apply for OS/390 Program Directory*.
- The current DB2 maintenance recommended by IBM Service.

Operating Capture for OS/390

An administrator can use the commands in this section to perform the following Capture for OS/390 tasks:

- Starting
- Scheduling
- Stopping

- Suspending
- Resuming
- Reinitializing
- Pruning
- Displaying captured log progress

You can submit the commands from TSO or the MVS console.

This section also lists restrictions for running the Capture program.

Restrictions for running the Capture program

Capture for OS/390 cannot replicate certain types of data. See “Data restrictions” on page 73 for a list of restrictions.

Only one instance of the Capture program can run on a subsystem. In a data-sharing environment, only one of any member subsystems in a data-sharing group can run the Capture program. In a data-sharing environment, DB2 presents merged log records from all member subsystems.

Starting Capture for OS/390

After you start the Capture program, it runs continuously until you stop it or it detects an unrecoverable error.

To start the Capture for OS/390 program:

1. Prepare the JCL for OS/390 by specifying the appropriate optional invocation parameters in the PARM field of the ASNL2RN*x* DD statement. Customize the JCL to meet your site’s requirements.

   An example of this line in the invocation JCL is:

   ```
   //ASNL2RNx EXEC PGM=ASNLRPnn,PARM='DB2_subsystem_name NOTERM WARMNS SLEEP=2'
   ```

   where *x* and *nn* indicate the level of the Capture program, as follows:

   - For the Capture program running on DB2 for MVS Version 4 Release 1, *x* is 4 and *nn* is 64
   - For the Capture program running on DB2 for OS/390 Version 5 Release 1, *x* is 5 and *nn* is 65
   - For the Capture program running on DB2 for OS/390 Version 6, *x* is 6 and *nn* is 66
2. Submit the JCL from TSO or from the MVS console. Capture for OS/390 can run either as a batch job or a started task.

Table 8 defines the invocation parameters.

Table 8. ASNLRP Invocation Parameter Definitions for OS/390

Parameter	Definition
DB2_subsystem_name	Specifies the name of the DB2 subsystem that can connect to the control server. The default for the subsystem name is DSN. This parameter must be the first parameter. For data sharing, do not use the group attach name. Instead, specify a member subsystem name.
TERM (default)	Terminates the Capture program if DB2 is terminated.
NOTERM	Keeps the Capture program running if DB2 is terminated with MODE(QUIESCE). When DB2 initializes, the Capture program starts in WARM mode and begins capturing where it left off when DB2 terminated. If DB2 terminates via FORCE or due to abnormal termination, the Capture program terminates even if you selected this parameter. If you use the NOTERM option and start DB2 with restricted access (ACCESS MAINT), the Capture program will not be able to connect and will terminate.
WARM (default)	The Capture program resumes processing where it ended in its previous run if warm start information is available. If the Capture program cannot warm start, it switches to a cold start.
WARMNS	The Capture program resumes processing where it ended in its previous run if warm start information is available. Otherwise, it issues a message and terminates. With WARMNS, the Capture program does not automatically switch to a cold start. The Capture program leaves the trace, UOW, CD, and warm start tables intact. In case of errors, the Capture program terminates instead of switching to a cold start as when WARM is specified.
COLD	The Capture program starts up by deleting all rows in its CD table, UOW table, and trace table during initialization. All subscriptions to these replication sources will be fully refreshed during the next Apply program processing cycle.
PRUNE (default)	The Capture program automatically prunes the rows in the CD and UOW tables that the Apply program copied, at the interval specified in the tuning parameters table.

Table 8. ASNLRP Invocation Parameter Definitions for OS/390 (continued)

Parameter	Definition
NOPRUNE	Automatic pruning is disabled. The Capture program prunes the CD and UOW tables when you enter the **PRUNE** command.
NOTRACE (default)	No trace information is written.
TRACE	Writes trace messages to the standard output, SYSPRINT.
SLEEP=*n*	Where *n* is the number of seconds that the Capture program will wait when it finishes processing the active log. This parameter is available for the Capture program running on DB2 for MVS Version 4 Release 1 and later with data sharing. The default is SLEEP=0.
ALLCHG (default)	Specifies that an entry is made to the CD table whenever a source table row changes.
CHGONLY	Specifies that an entry is made to the CD table when a source table row changes only if the columns defined for replication (CD table columns) change values.

Scheduling Capture for OS/390

Use either the **$TA JES2** command or the **AT NetView** command to start Capture for OS/390 at a specific time. You must:

1. Create a procedure that calls Capture for OS/390 in the PROCLIB.
2. Modify the ICHRIN03 RACF module (or appropriate definitions for your MVS security package) to associate the procedure with a user ID.
3. Link-edit the module in SYS1.LPALIB.

Stopping Capture for OS/390

Use the **STOP** command to stop the Capture program gracefully and commit the log records that it processed up to that point.

```
►►──F──jobname──,STOP──────────────────────────────────────────────────►◄
```

Issue the **STOP** command before:

- Removing an existing replication source
- Opening and modifying an existing replication source
- Shutting down the database

Suspending Capture for OS/390

Use the **SUSPEND** command to relinquish OS/390 resources to operational transactions during peak periods without destroying the Capture program environment. This command suspends the Capture program until you issue the **RESUME** command.

```
►►──F──jobname──,SUSPEND──────────────────────────────────────────────►◄
```

Important: Do not use **SUSPEND** when canceling a replication source. Instead, stop the Capture program by entering the **STOP** command.

Resuming Capture for OS/390

Use the **RESUME** command to resume the suspended Capture program.

```
►►──F──jobname──,RESUME───────────────────────────────────────────────►◄
```

Reinitializing Capture for OS/390

Use the **REINIT** command to begin to capture changes from new source tables if you add a new replication source or ALTER ADD a column to a replication source and CD table while the Capture program is running. The **REINIT** command tells the Capture program to obtain newly added replication sources from the register table.

REINIT also rereads the tuning parameters table for any changes made to the tuning parameters.

```
►►──F──jobname──,REINIT───────────────────────────────────────────────►◄
```

Important: Do not use the **REINIT** command to reinitialize the Capture program after canceling a replication source or dropping a replication source table while the Capture program is running. Instead, stop the Capture program and start it again using the WARM or WARMNS option.

Pruning the change data and unit-of-work tables

Use the **PRUNE** command to start pruning the CD and UOW tables.

This command prunes tables once.

```
►►──F──jobname──,PRUNE──────────────────────────────────────────────────►◄
```

The Capture program issues the message ASN0124I when the command is successfully queued.

During pruning, if you stop or suspend the Capture program, pruning does not resume after you enter the **RESUME** command. You must enter the **PRUNE** command again to resume pruning.

Displaying captured log progress

Use the **GETLSEQ** command to provide the timestamp and current log sequence number. You can use this number to determine how far the Capture program has read the DB2 log.

```
►►──F──jobname──,GETLSEQ────────────────────────────────────────────────►◄
```

The Capture program issues the message ASN0125I indicating when the current log sequence number is successfully processed.

Operating Apply for OS/390

An administrator can use the commands in the following sections to perform the following Apply for OS/390 tasks:

- Starting
- Scheduling
- Stopping

You can submit the commands from TSO or an MVS console.

Starting Apply for OS/390

After you start the Apply program, it runs continuously until:

- You stop it in an orderly way.
- You cancel it.
- An unexpected error or failure occurs.

To start the Apply for OS/390 program:

Prepare the JCL for OS/390 by specifying the appropriate invocation parameters in the PARM field of the **ASNARUN** command. Customize the JCL to meet your site's requirements. Invocation JCL is included with the Apply for OS/390 product.

An example of this line in the invocation JCL is:

```
//ASNARUN EXEC PGM=ASNAPVnn,PARM='Apply_qual DB2_subsystem_name DISK'
```

Where *nn* is the level the Apply program, as follows:

- For the Apply program running on DB2 for MVS Version 4 Release 1, *nn* is 64
- For the Apply program running on DB2 for OS/390 Version 5 Release 1, *nn* is 65
- For the Apply program running on DB2 for OS/390 Version 6, *nn* is 66

Table 9 defines the invocation parameters.

Table 9. ASNARUN Invocation Parameter Definitions

Parameter	**Definition**
Apply_qual	Specifies the Apply program qualifier that the Apply program instance uses to identify the subscriptions to be served. The Apply qualifier is case sensitive and must match the value of APPLY_QUAL in the subscription set table. This must be the first parameter.
DB2_subsystem_name	Specifies the name of the DB2 subsystem that can connect to the control server. This parameter must be the second parameter. For data sharing, do not use the group attach name. Instead, specify a member subsystem name.
Control_server_name	Specifies the name of the server where the replication control tables will reside. If you do not specify this parameter, the default is the current server.
LOADXit	Specifies that the Apply program is to invoke ASNLOAD, an IBM-supplied exit routine that uses the export and load utilities to refresh target tables. Currently, no utility program is available for use by ASNLOAD on DB2 for OS/390.
NOLOADXit (default)	Specifies that the Apply program will not invoke ASNLOAD.
MEMory (default)	Specifies that a memory file stores the fetched answer set. The Apply program fails if there is insufficient memory for the answer set.
DISK	Specifies that a disk file stores the fetched answer set.

Table 9. ASNARUN Invocation Parameter Definitions (continued)

Parameter	Definition
INAMsg (default)	Specifies that the Apply program is to issue a message when the Apply program is inactive.
NOINAMsg	Specifies that the Apply program will not issue this message.
NOTRC (default)	Specifies that the Apply program does not generate a trace.
TRCERR	Specifies that the Apply program generates a trace that contains only error information.
TRCFLOW	Specifies that the Apply program generates a trace that contains error and execution flow information.
NOTIFY	Specifies that the Apply program is to invoke ASNDONE, an exit routine that returns control to the user when the Apply program processing ends.
NONOTIFY (default)	Specifies that the Apply program will not invoke ASNDONE.
SLEEP (default)	Specifies that the Apply program is to go to sleep if no new subscriptions are eligible for processing.
NOSLEEP	Specifies that the Apply program is to stop if no new subscriptions are eligible for processing.
DELAY(*n*)	Where *n*=0, 1, 2, 3, 4, 5, or 6. Specifies the delay time (in seconds) at the end of each Apply program cycle when continuous replication is used. The default delay time is 6 seconds.

Scheduling Apply for OS/390

Use either the **$TA JES2** command or the **AT NetView** command to start Apply for OS/390 at a specific time. You must:

1. Create a procedure that calls Apply for OS/390 in the PROCLIB.
2. Modify the ICHRIN03 RACF module (or appropriate definitions for your MVS security package) to associate the procedure with a user ID.
3. Link-edit the module in SYS1.LPALIB.

See *MVS/ESA JES2 Commands* for more information about using the **$TA JES2** command, and *NetView for MVS Command Reference* for more information about using the **AT NetView** command.

Stopping Apply for OS/390

Enter the following command to stop the Apply for OS/390 program:

```
►►──P──jobname──────────────────────────────────────────────────────────►◄
```

Rules for index types

Because TYPE 2 indexes do not lock index pages, you can avoid deadlock and timeout problems in your applications if you specify TYPE 2 indexes. TYPE 2 indexes also enable you to use other functions, such as parallel query central processor (CP) processing, improved partition independence, row locking, and the ability to read through locks. If you specify TYPE 2 indexes, all specifications of SUBPAGES are ignored, and an error message is issued.

If you do not specify an index type, the index type is determined as follows:

- If the LOCKSIZE is ROW, the default index type is TYPE 2, regardless of the type specified on the installation panel DSNTIPE.
- If the LOCKSIZE is not ROW, the default index type is the type specified in the field DEFAULT INDEX TYPE on the installation panel DSNTIPE. The default value for that field is TYPE 2.

Recommendations:

- Specify TYPE 2 as the index type for all table indexes. For best performance, bind the Capture program and the Apply packages using isolation UR. When you specify isolation UR, all indexes on the control tables are required to be created as TYPE 2 indexes. If source views are being used in replication subscriptions, and the Apply package is bound using isolation UR, only TYPE 2 indexes over the source tables involved in the source view can be used by the Apply program.
- For DB2 for OS/390 source servers, specify TYPE 2 on every generated CREATE INDEX statement and every CREATE INDEX statement in the DB2 Control Center file DPCNTL.MVS. (DRJA creates TYPE2 indexes automatically.)

Using the DB2 ODBC Catalog

The DB2 ODBC Catalog is designed to improve the performance of ODBC applications. The tables in the DB2 ODBC Catalog are prejoined and indexed to support faster catalog access for ODBC applications. IBM’s ODBC driver also supports multiple views of the DB2 ODBC Catalog.

Support for use of the DB2 ODBC Catalog is provided by DB2 DataPropagator Version 5 and later. For information about Version 5 level support, see the *IBM Replication Guide and Reference V5*. Enhancements to the DB2 ODBC Catalog for DB2 DataPropagator Version 6 include:

- Support for the SYSIBM.SYSROUTINES table
- Support for the SQLProcedureColumns ODBC function call

You can eliminate data currency problems by using the DB2 ODBC Catalog tables. DB2 DataPropagator for OS/390 Version 6 can keep data in the DB2 ODBC Catalog synchronized with the contents of the real DB2 catalog table. The Capture program identifies log records that represent changes to the DB2 catalog and records these changed data records in a staging table. The Apply program replicates the changed data records to the DB2 ODBC Catalog tables.

This section describes how to implement the DB2 ODBC Catalog using the automatic mode. The automatic mode automatically replicates any DB2 Catalog changes to the DB2 ODBC Catalog tables.

Setting up the DB2 ODBC Catalog

The following section provides setup instructions needed to prepare your client and server to run your ODBC queries.

Setting up the workstation client

To use the entire DB2 ODBC Catalog, add the entry `CLISCHEMA=CLISCHEM` to the DB2CLI.INI file. To use your own set of views rather than the entire DB2 ODBC Catalog, add the entry `CLISCHEMA=MYSCHEMA` to the DB2CLI.INI file. The following example contains both statements.

```
[tstcli1x]
uid=userid
pwd=password
autocommit=0
TableType="'TABLE','VIEW','SYSTEM TABLE'"

[tstcli2x]
  Assuming dbalias2 is a database in DB2 for MVS
SchemaList="'OWNER1','OWNER2','CURRENT SQLID'"

[MyVeryLongDBALIASName]
dbalias=dbalias3
SysSchema=MYSCHEMA

[RDBD2205]
AUTOCOMMIT=1
LOBMAXCOLUMNSIZE=33554431
LONGDATACOMPAT=1
PWD=USRT006
UID=USRT006
```

```
DBALIAS=RDBD2205
CLISCHEMA=CLISCHEM

[RDBD2206]
AUTOCOMMIT=1
LOBMAXCOLUMNSIZE=33554431
LONGDATACOMPAT=1
PWD=USRT006
UID=USRT006
DBALIAS=RDBD2206
CLISCHEMA=MYSCHEMA
```

You must define views for all the DB2 ODBC Catalog tables when you use your own schema. See Table 10 on page 146 for the list of the DB2 ODBC Catalog tables for which you must define a view. Use the following VIEW MYSCHEMA statement to define the DB2 ODBC Catalog views on CLISCHEM.*table_name* ODBC tables.

```
CREATE VIEW MYSCHEMA.table_name FROM CLISCHEM.table_name
 where TABLE_SCHEM=MYUSER
```

Where *table_name* is one of DB2 ODBC Catalog table names.

Setting up the server

To set up the server, define the following control information for replication:

1. Create the DB2 DataPropagator for OS/390 control tables, if they do not already exist.
   a. Review the header portion of the ASNL2CN6.SQL file and customize the table spaces according to your site requirements.
   b. Connect to the OS/390 RDB that contains the catalog from which you want to create a new DB2 ODBC Catalog.
   c. Run the ASNL2CN6.SQL file from either the client or the OS/390 server.
2. Create the source, subscription control, and table space information for the DB2 ODBC Catalog.
   a. Review the header portion of the ASNL2SY6.SQL, ASNL2RE6.SQL, and ASNL2SU6.SQL files and customize the table spaces according to your site requirements.
   b. Replace all occurrences of SRCE in the ASNL2SU6.SQL file with the OS/390 RDB name. You can also define additional views on the predefined subscriptions to further qualify the subscriptions.
   c. Connect to the OS/390 RDB that contains the catalog from which you want to create a new DB2 ODBC Catalog.
   d. Run the ASNL2SY6.SQL, ASNL2RE6.SQL, and ASNL2SU6.SQL files from either the client or the OS/390 server.

3. Start the Capture and Apply programs on OS/390. Starting the Capture and Apply programs populates the ODBC Catalog on OS/390.

DB2 ODBC Catalog tables

Table 10 lists the ODBC function calls that are supported by the DB2 ODBC Catalog and explains how the function calls are implemented by DB2 DataPropagator for OS/390 V6.

Table 10. ODBC Function Calls

ODBC Function Call	ODBC Catalog Tables
SQLColumns	The SELECT command is issued against preformatted data stored in CLISCHEM.COLUMNS. This call is implemented with the source table SYSIBM.SYSCOLUMNS.
SQLColumnPrivileges	The SELECT command is issued against prejoined data stored in CLISCHEM.COLUMNPRIVILEGES. This call is implemented with source tables SYSIBM.SYSCOLUMNS, SYSIBM.SYSTABAUTH, and SYSIBM.SYSCOLAUTH.
SQLForeignKeys	The SELECT command is issued against prejoined data stored in CLISCHEM.FOREIGNKEYS. This call is implemented with source tables SYSIBM.SYSRELS, SYSIBM.SYSFOREIGNKEYS, and SYSIBM.SYSCOLUMNS.
SQLPrimaryKeys	The SELECT command is issued against the primary keys stored in CLISCHEM.PRIMARYKEYS. This call is implemented with the source table SYSIBM.SYSCOLUMNS.
SQLProcedures	The SELECT command is issued against CLISCHEM.PROCEDURES, which contains only the columns required by the SQLProcedures function. This call is implemented with source table SYSIBM.SYSROUTINES.
SQLSpecialColumns	The SELECT command is issued against prejoined data stored in CLISCHEM.SPECIALCOLUMNS. This call is implemented with source tables SYSIBM.SYSCOLUMNS, SYSIBM.SYSKEYS, and SYSIBM.SYSINDEXES.
SQLTablesPrivileges	The SELECT command is issued against CLISCHEM.TABLEPRIVILEGES. This call is implemented with source table SYSIBM.SYSTABAUTH.
SQLTables	The SELECT command is issued against pre-joined data stored in CLISCHEM.TABLES. This call is implemented with source table SYSIBM.SYSTABLES.

Table 10. ODBC Function Calls (continued)

ODBC Function Call	ODBC Catalog Tables
SQLStatistics	The SELECT command is issued against pre-joined data stored in CLISCHEM.TSTATISTICS. This call is implemented with source tables SYSIBM.SYSTABLES, SYSIBM.SYSINDEXES, and SYSIBM.SYSKEYS.
SQLProcedureColumns	The SELECT command is issued against pre-joined data stored in CLISCHEM.PROCEDURECOLUMNS. This call is implemented with source tables SYSIBM.SYSROUTINES and SYSIBM.SYSPARMS.

Chapter 9. Capture and Apply for AS/400

This chapter describes how to set up and operate Version 5 of the DataPropagator Relational Capture and Apply programs for AS/400 (DPROPR/400).

You can run Version 1 and Version 5 of DPROPR/400 concurrently. However, this chapter does not contain information about DPROPR/400 Version 1. For information on using DPROPR/400 Version 1, see *DataPropagator Relational Capture and Apply for AS/400 V4R2, SC41–5346*.

Both the DB2 DataJoiner Replication Administration tool (DJRA) and the DB2 Control Center provide basic replication administration functions for defining replication sources and subscription sets. However, you should use DJRA for all replication administration tasks. Only DJRA provides support for the use of a relative record number (RRN) as a primary key.

Be sure to read the following sections before reading the sections on operating the Capture and Apply programs for AS/400:

- "Setting up the Capture and Apply programs"
- "Authorization requirements for running the Capture and Apply programs" on page 153
- "Restrictions for running the Capture program" on page 162
- "The Journal" on page 163
- "Using a relative record number (RRN) as a primary key" on page 169

Setting up the Capture and Apply programs

Setting up the Capture and Apply programs consists of installing DPROPR/400 and tuning the Capture program for optimum productivity. This section describes how to set up the Capture and Apply programs.

Installing DPROPR/400

You install DPROPR/400 in the same way that you install any other licensed program. Follow these steps for the regular installation of DPROPR/400:

1. Type `GO LICPGM` on the AS/400 command line.
2. Select option 11 (Install licensed programs).
3. Page down to locate **DataPropagator Relational (5769DP2)**.

If the window does not contain the product ID number (5769DP2) on the install screen, exit LICPGM and enter `RSTLICPGM` on the AS/400 command line, and then specify 5769DP2 for the product ID.

If the window does contain the ID number, type a 1 next to it and press the Enter key.

Verifying and customizing your DPROPR/400 installation

You should install DPROPR/400 before using the replication administration tools, because the installation process issues the **CRTDPRTBL** command to automatically create the DPROPR/400 replication control tables. These tables are created in the DataPropagator Relational collection (named ASN), if they do not already exist.

The installation program also creates an SQL journal and journal receiver for this library. It also creates work management objects. Table 11 lists the work management objects that are created.

Table 11. Work Management Objects

Description	Object type	Name
Subsystem description	*SBSD	QDPR/QZSNDPR
Job queue	*JOBQ	QDPR/QZSNDPR
Job description	*JOBD	QDPR/QZSNDPR

A note on work management: You can alter the default definitions or provide your own definitions. See *OS/400 Work Management V4R3, SC41–5306* for more information on changing these definitions.

Creating the replication control tables

If your replication control tables are accidentally deleted or corrupted, you can create them manually using the Create DPR Tables (**CRTDPRTBL**) command. You must have *ALLOBJ authority to run this command.

Important: The **CRTDPRTBL** command is the only command that you should use to create AS/400 control tables. Do not use DJRA to create the control tables.

```
►►──CRTDPRTBL──┬──────────────────────┬──────────────────────────►◄
               │          ┌─1─┐       │
               └─DPRVSN(──┴─5─┴────)──┘
```

Table 12. CRTDPRTBL Command Parameter Definitions for AS/400

Parameter	Definition and Prompts
DPRVSN	Specifies the version of the control tables to create. You can specify one or both of the version levels: **1** (default) Specifies Version 1 control tables. **5** Specifies Version 5 control tables. The system creates all of the control tables for replication sources and targets along with the default SQL journal.

Specifying tuning parameters for Capture for AS/400

To control the performance of the Capture program, you can adjust four tuning parameters on the server by changing the values of columns in the tuning parameters table.

To specify the tuning parameters, do one of the following tasks:

- Update the tuning parameters table manually. See “Specifying tuning parameters for the Capture program” on page 111 for more information.
- Run the **CHGDPRCAPA** command. See “Changing Capture program attributes” for more information about this command.
- Customize the DPCNTL.400 file in the DB2 Control Center \sqllib\samples\repl directory before you define the first replication source for a database. You do not need to customize the DPCNTL.400 file if you have already installed DPROPR/400.

Changing Capture program attributes

The Change DPR Capture Attributes (**CHGDPRCAPA**) command changes the global operating parameters in the tuning parameters table for the Capture program.

You can see the current values of the Capture program attributes if you issue the **CHGDPRCAPA** command with the F4 key to prompt on the command.

```
►►──CHGDPRCAPA──┬─────────────────────────────┬──┬──────────────────────────┬──────────►
                └─RETAIN(──┬─*SAME───┬──)──┘  └─LAG(──┬─*SAME───┬──)──┘
                           └─minutes─┘                └─minutes─┘
```

```
►─┬───────────────────────────────┬─┬──────────────────────────────┬──────────────────►◄
  └─FRCFRQ(──┬─*SAME────┬──)──────┘ └─CLNUPITV(──┬─*SAME─┬──)──────┘
             └─seconds──┘                        └─1-100─┘
```

Table 13. CHGDPRCAPA Command Parameter Definitions for AS/400

Parameter	Definition and Prompts
RETAIN	Specifies the number of minutes that data is retained in the CD tables before it is removed. The value of this parameter works with the CLNUPITV parameter. When the CLNUPITV value is reached, data in the CD and UOW tables is removed if the UOW table has a timestamp older than that of the earliest Apply cycle for the table. Ensure that the Apply intervals are set to copy changed information before the value on the RETAIN parameter is reached. This prevents your tables from becoming inconsistent. If they become inconsistent, the Apply program performs full refreshes. ***SAME** (default) Specifies that the value remains unchanged. *minutes* Specifies the number of minutes that the CD is retained. The maximum value is 35 000 000. The default value is 10 080 minutes (7 days).
LAG	Specifies the number of minutes that the Capture program can fall behind before clearing out the CD tables and starting over with change capture. When the lag limit is reached (that is, when the timestamp of the journal entry is older than the current time minus the lag limit), the Capture program assumes that it is too far behind to catch up. It then initiates a cold start for the tables that it is processing for that journal. The Apply program then performs a full refresh to provide the Capture program with a new starting point. Users typically set this value high so that it has no effect. ***SAME** (default) Specifies that the value remains unchanged. *minutes* Specifies the number of minutes that the CD entries are allowed to fall behind. The maximum value is 35 000 000. The default value is 10 080 minutes (7 days).

Table 13. CHGDPRCAPA Command Parameter Definitions for AS/400 (continued)

Parameter	Definition and Prompts
FRCFRQ	Specifies approximately how often the Capture program writes changes to the CD and UOW tables. The Capture program makes the changes available to the Apply program either when the buffers are filled or when this time has expired, whichever is sooner. Use this parameter to make source table changes more readily available for the Apply program on servers with a low rate of source table changes. This is a global value, and is used for all defined source tables. Setting this value at a lower number can affect processor usage. ***SAME** (default) Specifies that the value remains unchanged. *seconds* Specifies the number of seconds that the Capture program keeps CD table and UOW table changes in buffer space before making them available for use by the Apply program. This value can range from 30 to 600 seconds. The default value is 180 seconds.
CLNUPITV	Specifies the maximum length of time before the Capture program prunes old records from the CD tables and the UOW table, if it exists. This parameter works with the RETAIN parameter. The value of this parameter is converted from hours to seconds and stored in the PRUNE_INTERVAL column of the tuning parameters table. If the PRUNE_INTERVAL column is changed outside of the **CHGDPRCAPA** command, you might see changes due to rounding when you prompt using the F4 key. ***SAME** (default) Specifies that the value remains unchanged. *1–100* Specifies the maximum number of hours that you want the Capture program to wait before pruning. Valid values are 1–100.

Authorization requirements for running the Capture and Apply programs

This section describes the commands available for granting and revoking authority to the replication control tables: “Granting authority” on page 154 and “Revoking authority” on page 161.

Granting authority

The Grant DPR Authority (**GRTDPRAUT**) command authorizes a list of users to the replication control tables, so that the users can run the Capture and Apply programs. The **GRTDPRAUT** command compensates for the different authority requirements for DPROPR/400. For example, the authority requirements for the user who is running the Capture and Apply programs might differ from the authority requirements for the user who defines replication sources and targets.

You must have *ALLOBJ authority to grant authorities.

```
►►──GRTDPRAUT──USER(──┬─user-name─┬──)──AUT(──┬─*REGISTRAR──┬──)──────────────►
                      └─*PUBLIC───┘           ├─*SUBSCRIBER─┤
                                              ├─*CAPTURE────┤
                                              └─*APPLY──────┘

             ┌─1─┐                      ┌─*ALL────────────┐
►─DPRVSN(────┴─5─┴────)──APYQUAL(───────┼─*USER───────────┼───)──────────────►◄
                                        └─apply-qualifier─┘
```

Table 14. GRTDPRAUT Command Parameter Definitions for AS/400

Parameter	Definition and Prompts
USER	Specifies the users who have authority. *user-name* Specifies the names of up to 50 users who have authority. ***PUBLIC** Specifies that *PUBLIC authority is granted to the file, but (if insufficient for the task) is used only for those users who have no specific authority, who are not on the authorization list associated with the file, and whose group profile does not have any authority.

Table 14. GRTDPRAUT Command Parameter Definitions for AS/400 (continued)

Parameter	Definition and Prompts
AUT	Specifies the type of DPROPR/400 authority being granted. ***REGISTRAR** (default) Specifies that the users are granted the authorities to define, change, and remove subscription sets. For a complete list of authorities with AUT(*REGISTRAR), see Table 15 on page 157. ***SUBSCRIBER** Specifies that the users are granted authority to define, change, and remove subscription sets. For a complete list of authorities with AUT(*SUBSCRIBER), see Table 16 on page 158. ***CAPTURE** Specifies that the users are granted authority to run the Capture program. For a complete list of authorities granted with AUT(*CAPTURE), see Table 17 on page 159. ***APPLY** Specifies that the users are granted authority to run the Apply program. The command does not grant authority to any of the objects that reside on other databases accessed by the Apply program. When an Apply process is invoked, the user associated with the DRDA application server job must also be granted *APPLY authority. If the source is an AS/400 server, the **GRTDPRAUT** command should be run on the source server system, with the application server job user specified on the USER parameter and the Apply qualifier specified on the APYQUAL parameter. Authorities are not granted to the target tables unless the target server is the same as the control server and both reside on the system where the command is run. For a complete list of authorities granted with AUT(*APPLY), see Table 18 on page 160.
DPRVSN	Specifies the version of DPROPR/400. You can specify one or both of the version levels. **1** (default) Specifies Version 1 of DPROPR/400. **5** Specifies Version 5 of DPROPR/400.

Table 14. GRTDPRAUT Command Parameter Definitions for AS/400 (continued)

Parameter	Definition and Prompts
APYQUAL	Specifies the Apply qualifier to be used by the user specified with the USER parameter. This parameter is used only when AUT(*APPLY) or AUT(*SUBSCRIBER) is specified. ***ALL** (default) Specifies that the user is granted authority to run the Apply program or to define and remove subscriptions for *all* Apply qualifiers. ***USER** Specifies that the users specified on the USER parameter are granted authority to subscriptions with an Apply qualifier that is the same as the user name. *apply-qualifier* Specifies that the user is granted authority to run the Apply program or define and remove subscriptions for the Apply qualifiers associated with this Apply qualifier. • The user is granted authority to all replication sources, CD tables, and CCD tables associated with records in the pruning control table that have a value in the APPLY_QUAL column matching the value input with the APYQUAL parameter. • The user is granted authority to the subscriptions listed in the subscription-targets-member table that reside on this system.

You cannot use the **GRTDPRAUT** command while the Capture or Apply programs are running, or when applications that use the source tables are active because authorizations cannot be changed on files that are in use.

Examples

Example 1: To authorize user USER1 to define and modify replication sources:

```
GRTDPRAUT USER(USER1) AUT(*REGISTRAR) DPRVSN(5)
```

Example 2: To authorize user USER1 to define and modify subscriptions:

```
GRTDPRAUT USER(USER1) AUT(*SUBSCRIBER) DPRVSN(5)
```

Example 3: To authorize user USER1 to define and modify existing subscriptions associated with Apply qualifier A1:

```
GRTDPRAUT USER(USER1) AUT(*SUBSCRIBER) DPRVSN(5) APYQUAL(A1)
```

Example 4: To authorize a user to run the Apply program on the control server system for all subscriptions associated with Apply qualifier A1, where the target server is the same as the control server:

1. Run the following command on the system where the Apply program will run:
   ```
   GRTDPRAUT USER(USER1) AUT(*APPLY) DPRVSN(5) APYQUAL(A1)
   ```
2. If the application server job on the source server used by the Apply program runs under user profile USER1, run the following command on the source server systems:
   ```
   GRTDPRAUT USER(USER1) AUT(*APPLY) DPRVSN(5) APYQUAL(A1)
   ```

   If the application server job on the source server used by the Apply program runs under a different user profile; for example, QUSER, the command is:
   ```
   GRTDPRAUT USER(QUSER) AUT(*APPLY) DPRVSN(5) APYQUAL(A1)
   ```

The levels of authority

The following tables list the authorities granted when you specify:

- AUT(*REGISTRAR)
- AUT*(SUBSCRIBER)
- AUT(*CAPTURE)
- AUT(*APPLY)

on the **GRTDPRAUT** command.

The following table lists the authorities granted when you specify the AUT(*REGISTRAR) parameter on the **GRTDPRAUT** command:

Table 15. Authorities granted with GRTDPRAUT AUT(*REGISTRAR)

Library	Object	Type	Version	Authorizations
QSYS	ASN	*LIB	1 5	*USE, *ADD
ASN	QSQJRN	*JRN	1 5	*OBJOPR, *OBJMGT
ASN	IBMSNAP_REGISTER	*FILE	5	*OBJOPR, *READ, *ADD, *UPD, *DLT
ASN	IBMSNAP_REGISTERX	*FILE	5	*OBJOPR, *READ, *ADD, *UPD, *DLT

Table 15. Authorities granted with GRTDPRAUT AUT(*REGISTRAR) (continued)

Library	Object	Type	Version	Authorizations
ASN	IBMSNAP_REG_EXT	*FILE	1 5	*OBJOPR, *OBJMGT, *READ, *ADD, *UPD, *DLT
ASN	IBMSNAP_REG_EXTX	*FILE	1 5	*OBJOPR, *OBJMGT, *READ, *ADD, *UPD, *DLT
ASN	IBMSNAP_UOW	*FILE	1 5	*OBJOPR, *OBJMGT, *READ, *ADD *DLT
ASN	IBMSNAP_UOW_IDX	*FILE	1 5	*OBJOPR, *OBJMGT, *READ, *ADD, *DLT
ASN	IBMSNAP_PRUNCNTL	*FILE	5	*OBJOPR, *READ
ASN	IBMSNAP_CCPPARMS	*FILE	1 5	*OBJOPR, *READ, *UPD
ASN	QZSNCTLBLK	*USRSPC	1 5	*CHANGE
ASN	ASN4B*	*SQLPKG	5	*USE
ASN	ASN4C*	*SQLPKG	5	*USE
QSYS	Source library	*LIB	1 5	*USE
Source library	Source table	*FILE	1 5	*OBJOPR, *READ
QSYS	Control library	*LIB	1 5	*USE, *ADD
Control library	CDtimestamp - CD table	*FILE	5	*USE, *OBJMGT, *OBJEXIST

The following table lists the authorities granted when you specify the AUT(*SUBSCRIBER) parameter on the **GRTDPRAUT** command:

Table 16. Authorities granted with GRTDPRAUT AUT(*SUBSCRIBER)

Library	Object	Type	Version	Authorizations
QSYS	ASN	*LIB	5	*USE, *ADD
QSYS	IBMSNAP_SUBS_SET	*FILE	5	*CHANGE
ASN	IBMSNAP_APPLYTRAIL	*FILE	5	*CHANGE
ASN	IBMSNAP_SUBS_COL	*FILE	5	*CHANGE

Table 16. Authorities granted with GRTDPRAUT AUT(*SUBSCRIBER) (continued)

Library	Object	Type	Version	Authorizations
ASN	IBMSNAP_SUBS_EVENT	*FILE	5	*CHANGE
ASN	IBMSNAP_SUBS_STMTS	*FILE	5	*CHANGE
ASN	IBMSNAP_SUBS_MEMBR	*FILE	5	*CHANGE
ASN	IBMSNAP_REGISTER	*FILE	5	*USE, *UPD
ASN	IBMSNAP_REG_EXT	*FILE	1 5	*USE, *UPD
ASN	IBMSNAP_PRUNCNTL	*FILE	5	*USE, *ADD, *DLT
ASN	ASN4U*	*SQLPKG	5	*USE
ASN	ASN4A*	*SQLPKG	5	*USE
QSYS	Source library	*LIB	1 5	*USE
Source library	Source table	*FILE	1 5	*OBJOPR, *READ
QSYS	Control library	*LIB	5	*USE
Control library	ASNtimestampPC - pruning control table	*LIB	5	*USE
Control library	CD table	*FILE	1 5	*OBJOPR, *READ
Control library	Internal CCD table	*FILE	1 5	*OBJOPR, *READ
QSYS	Target library	*LIB	5	*USE, *ADD
Target library	Target table	*FILE	5	*USE, *OBJMGT, *OBJEXIST

The following table lists the authorities granted when you specify the AUT(*CAPTURE) parameter on the **GRTDPRAUT** command:

Table 17. Authorities granted with GRTDPRAUT AUT(*CAPTURE)

Library	Object	Type	Version	Authorizations
QSYS	ASN	*LIB	1 5	*USE, *OBJMGT
ASN	IBMSNAP_REGISTER	*FILE	1 5	*USE, *UPD
ASN	IBMSNAP_REG_EXT	*FILE	1 5	*USE, *UPD
QSYS	Control library	*LIB	1 5	*USE
Control library	CD table	*FILE	1 5	*OBJOPR, *OBJMGT, *READ, *UPD, *DLT, *ADD

Table 17. Authorities granted with GRTDPRAUT AUT(*CAPTURE) (continued)

Library	Object	Type	Version	Authorizations
Control library	CD table	*FILE	1 5	*OBJOPR, *OBJMGT, *READ, *UPD, *DLT, *ADD
ASN	IBMSNAP_PRUNCNTL	*FILE	5	*USE, *UPD
ASN	IBMSNAP_CRITSEC	*FILE	5	*USE
ASN	IBMSNAP_CCPPARMS	*FILE	1 5	*USE
ASN	IBMSNAP_UOW	*FILE	1 5	*CHANGE
ASN	IBMSNAP_TRACE	*FILE	5	*CHANGE
ASN	IBMSNAP_WARM_START	*FILE	5	*CHANGE
ASN	IBMSNAP_AUTHTKN	*FILE	5	*CHANGE
ASN	QZSBCTKBLK	*USRSPC	1 5	*CHANGE
ASN	ASNB*	SQLPKG	5	*USE
ASN	ASNC*	SQLPKG	5	*USE

The following table lists the authorities granted when you specify the AUT(*APPLY) parameter on the **GRTDPRAUT** command:

Table 18. Authorities granted with GRTDPRAUT AUT(*APPLY)

Library	Object	Type	Version	Authorizations
QSYS	ASN	*LIB	1 5	*USE
ASN	IBMSNAP_SUBS_SET	*FILE	5	*CHANGE
ASN	IBMSNAP_APPLYTRAIL	*FILE	5	*CHANGE
ASN	IBMSNAP_SUBS_COLS	*FILE	5	*USE
ASN	IBMSNAP_SUBS_EVENT	*FILE	5	*USE
ASN	IBMSNAP_SUBS_STMTS	*FILE	5	*USE
ASN	IBMSNAP_SUBS_MEMBR	*FILE	5	*USE
ASN	ASNA*	*SQLPKG	5	*USE
ASN	ASNU*	*SQLPKG	5	*USE
ASN	IBMSNAP_REGISTER	*FILE	5	*USE, *UPD
ASN	IBMSNAP_REG_EXT	*FILE	1 5	*USE, *UPD
ASN	IBMSNAP_UOW	*FILE	1 5	*USE, *UPD
ASN	IBMSNAP_PRUNCNTL	*FILE	5	*USE, *UPD, *ADD
ASN	IBMSNAP_CRITSEC	*FILE	5	*USE, *ADD

Table 18. Authorities granted with GRTDPRAUT AUT(*APPLY) (continued)

Library	Object	Type	Version	Authorizations
ASN	IBMSNAP_AUTHTKN	*FILE	5	*USE, *ADD
QSYS	Control library	*LIB	1 5	*USE
Control library	CD table	*FILE	1 5	*USE
QSYS	Target library	*LIB	5	*USE
Target library	Target table	*FILE	5	*CHANGE, *OBJMGT

Revoking authority

The Revoke DPR Authority (**RVKDPRAUT**) command revokes authority to the replication control tables so that users can no longer define or modify replication sources and subscriptions.

```
                          ┌──────────────────┐
                          │                  │              ┌1─┐
►►──RVKDPRAUT──USER(──┬───▼──user-name───────┴──┬──)──DPRVSN(──┴5─┴──)──────────►◄
                      └─*PUBLIC─────────────────┘
```

The command returns an error message if any of the following conditions occur:

- If a specified user does not exist.
- If the user running the command is not authorized to the specified user profiles.
- If the DPROPR/400 control tables do not exist.
- If the user running the command does not have permission to revoke authorities to the DPROPR/400 control tables.
- If the Capture or Apply programs are running.

Table 19. RVKDPRAUT Command Parameter Definitions for AS/400

Parameter	Definition and Prompts
USER	Specifies the users whose authority is revoked. *user-name* Specifies the names of up to 50 users whose authority is revoked. ***PUBLIC** Specifies that authority is revoked from all users without specific authority, who are not on the authorization list, and whose group profile does not have any authority.

Table 19. RVKDPRAUT Command Parameter Definitions for AS/400 (continued)

Parameter	Definition and Prompts
DPRVSN	Specifies the version of DPROPR/400. You can specify one or both of the version levels. **1** (default) Revoke authorities for Version 1 of DPROPR/400. **5** Revoke authorities for Version 5 of DPROPR/400.

Example

To revoke authorities to the control tables:

```
RVKDPRAUT USER(user-name) DPRVSN(5)
```

Restrictions for running the Capture program

If you perform any of the following actions while a Capture journal job is running, it terminates with message ASN2301 or ASN2201 in the job log:

- Delete the subscription row from the register table.
- Prevent the Capture program from gaining normal access by locking any of the control tables, user space, or user indexes needed for data capturing.
- Delete the user space ASN/QZSNCTLBLK.
- Delete the user index QDPR/QZSNINDEX5.
- Drop the index QDPR/IBMSNAP_UOW_IDX.
- Delete the message queue QDPR/QZSN5.
- Remove a message from the message queue QDPR/QZSN5 or send an extraneous message to the message queue QDPR/QZSN5.
- Attempt to add new rows to the CD tables or the UOW table when the user's storage limit is exceeded.
- Unsuccessfully attempt to allocate memory.

Keeping the Capture program running successfully

To keep the Capture program running, use the following guidelines:

- Use the default system-wide delete-journal-receiver support to prevent deletions of useful journal receivers. See "The delete journal receiver exit routine" on page 166 for more information.

  If you choose to manage journal receivers manually, delete journal receivers used for journaling source tables only if you are certain that all the entries on that receiver were processed by the Capture program.

When conditions make capturing data for a particular source table impossible, the Capture program changes the state of the source table from capturing changes to needing a full refresh. (See Table 23 on page 176 for a list of such conditions.) Other conditions that prevent data capturing for a source table are:

- An ALTER TABLE is performed on the source table or the CD table such that either:
  - A column in the CD table is no longer in the source table.
  - The column in the CD table has different attributes (data type, length) from its counterpart in the source table.

  When you need to perform an ALTER TABLE on the source table, ensure that you remove the subscription and define the source table again. Or you can use the DJRA alter register function to fix the changed data table. If you defined targets, use the DJRA alter register function to fix those target tables as well.
- A lock is placed on the source table or the CD table that prevents the Capture program from accessing the needed information.

The Journal

DPROPR/400 uses the information that it receives from the journals about changes to the data to populate the DPROPR/400 CD and UOW tables for replication.

DPROPR/400 runs under commitment control for most operations and therefore requires journaling on the control tables. (The QSQJRN journal is created when the **CRTDPRTBL** command creates a collection.)

Administrators must manually create the QSQJRN journal in the library that contains the replication source control tables and the library that contains the target tables. It is also the administrator's responsibility to ensure that all the source tables are journaled correctly.

Remote journal function

In previous versions of DPROPR/400, replication source definitions (including the control tables associated with a source) and the Capture program always resided on the same system. The remote journal function makes it possible to move the replication source definitions and the Capture program and its control tables away from the system on which the source tables reside, leaving more resources available on that system. With the remote journal function, processor usage can be lowered, DASD can be saved, and performance can be improved significantly.

Important: The intention of this type of setup is to have the replication source definitions on the same AS/400 system as the replication target.

A replication source definition that refers to a remote source table cannot be subscribed to by other platforms such as the Apply program for OS/390 or the Apply program for UNIX.

For more information about the remote journal function, see *AS/400 Remote Journal Function for High Availability and Data Replication, SG24-5189.*

Creating journals for source tables

To set up the source table journals, you must have the authority to create journals and journal receivers for the source tables to be defined.

Important: Use a different journal for the source tables than one of those created by DPROPR/400 (QSQJRN journals) in the ASN library, the source library, the control library, or the target library.

To create a source table journal:

1. Create a journal receiver in a library of your choice using the Create Journal Receiver (**CRTJRNRCV**) command. The following example uses a library named JRNLIB for journal receivers.

   ```
   CRTJRNRCV  JRNRCV(JRNLIB/RCV0001)
              THRESHOLD(50000)
              TEXT('DataPropagator Relational Journal Receiver')
   ```

   Be sure to:
   - Place the journal receiver in a library that is saved regularly.
   - Choose a journal receiver name that can be used to create a naming convention for future journal receivers, such as RCV0001. You can use the *GEN option to continue the naming convention when you change journal receivers. This type of naming convention is also useful if you choose to let the system manage the changing of your journal receivers.
2. Create the journal by using the Create Journal (**CRTJRN**) command:

   ```
   CRTJRN  JRN(JRNLIB/DJRN1)
           JRNRCV(JRNLIB/RCV0001)
           MNGRCV(*SYSTEM) DLTRCV(*YES)
           TEXT('DataPropagator Relational Journal')
   ```

   Be sure to:
   - Specify the name of the journal receiver that you created in the first step.
   - Use the Manage receiver (MNGRCV) parameter to have the system change the journal receiver and attach a new one when the attached

receiver becomes too large. If you choose this option, you do not need to use the **CRTJRN** command to detach receivers and create and attach new receivers manually.

- Specify DLTRCV(*NO) only if you have overriding reasons to do so (for example, if you need to save these journal receivers for recovery reasons). If you specify DLTRCV(*YES), these receivers might be deleted before you have a chance to save them.

You can use two values on the RCVSIZOPT parameter of the **CRTJRN** command (*RMVINTENT and *MINFIXLEN) to optimize your storage availability and system performance. See the *AS/400 Programming: Performance Tools Guide* for more information.

3. Start journaling the source table using the Start Journal Physical File (**STRJRNPF**) command, as in the following example:

```
STRJRNPF FILE(library/file)
         JRN(JRNLIB/DJRN1)
         OMTJRNE(*OPNCLO)
         IMAGES(*BOTH)
```

Specify the name of the journal that you created in step 2. The Capture program requires a value of *BOTH for the IMAGES parameter.

Managing journals and journal receivers

The Capture program uses the Receive Journal Entry (**RCVJRNE**) command to receive journals.

Specifying system management of journal receivers

It is recommended that you let the AS/400 system manage the changing of journal receivers. This is called *system change journal management*. Specify MNGRCV(*SYSTEM) when you create the journal, or change the journal to that value. If you use system change journal management support, you must create a journal receiver that specifies the threshold at which you want the system to change journal receivers. The threshold must be at least 5000 KB, and should be based on the number of transactions on your system. The system automatically detaches the receiver when it reaches the threshold size and creates and attaches a new journal receiver, if it can.

Specifying user management of journal receivers

If you specify MNGRCV(*USER) when you create the journal (meaning you want to manage changing your own journal receivers), a message is sent to the journal's message queue when the journal receiver reaches a storage threshold, if one was specified for the receiver.

Use the **CHGJRN** command to detach the old journal receiver and attach a new one. This command prevents `Entry not journaled` error conditions and limits the amount of storage space that the journal uses. To avoid affecting performance, do this at a time when the system is not at maximum use.

You can switch journal receiver management back to the system by specifying `CHGJRN MNGRCV(*SYSTEM)`.

You should regularly detach the current journal receiver and attach a new one for two reasons:

- Analyzing journal entries is easier if each journal receiver contains the entries for a specific, manageable time period.
- Large journal receivers can affect system performance and take up valuable space on auxiliary storage.

The default message queue for a journal is QSYSOPR. If you have a large volume of messages in the QSYSOPR message queue, you might want to associate a different message queue, such as DPRUSRMSG, with the journal. You can use a message handling program to monitor the DPRUSRMSG message queue. For an explanation of messages that can be sent to the journal message queue, see *OS/400 Backup and Recovery*.

The delete journal receiver exit routine

When you install DPROPR/400 on a V4R2 (or later) system, a *delete journal receiver* exit routine (DLTJRNRCV) is registered automatically. This exit routine is called any time a journal receiver is deleted, whether or not it is used for journaling the source tables. This exit routine determines whether or not a journal receiver can be deleted. (You no longer need to do this manually. Nor do you need to use the **ANZDPRJRN** command to delete old receivers.)

To take advantage of the delete journal receiver exit routine and leave journal management to the system, specify DLTRCV(*YES) and MNGRCV(*SYSTEM) on the **CHGJRN** or **CRTJRN** command.

If the journal that the receiver is associated with has no association with any of the source tables, this exit routine *approves* the deletion of the receiver.

If the journal receiver is used by one or more source tables, this exit routine makes sure that the receiver being deleted does not contain entries that have not been processed by the Capture program. The exit routine *disapproves* the deletion of the receiver if the Capture program still needs to process entries on that receiver.

If you must delete a journal receiver and the delete journal receiver exit routine does not approve the deletion, specify DLTJRNRCV DLTOPT(*IGNEXITPGM) to override the exit routine.

Removing the delete journal receiver exit routine: If you want to handle the deletion of journal receivers manually, you can remove the delete journal receiver exit routine by issuing the following command:

```
RMVEXITPGM EXITPNT (QIBM_QJO_DLT_JRNRCV)
           FORMAT(DRVC0100)
           PGMNBR(value)
```

To determine the PGMNBR value for the RMVEXITPGM command:

1. Issue the **WRKREGINF** command.
2. On the Work with Registration Information window, find the entry for exit point QIBM_QJO_DLT_JRNRCV. Enter 8 in the **Opt** field.
3. On the Work with Exit Programs window, find the entry for Exit Program QZSNDREP in library QDPR. The number that you need is under the Exit Program Number heading.

Registering the delete journal receiver exit routine for upgraded systems: If the 5769DP2 version of DPROPR/400 was installed on V4R1 and the operating system was upgraded to V4R2 or V4R3 without reinstalling the product, you must register the exit routine with this command:

```
ADDEXITPGM EXITPNT(QIBM_QJO_DLT_JRNRCV)
           FORMAT(DRCV0100)
           PGMNBR(value *LOW)
           CRTEXITPNT(*NO)
           PGM(QDPR/QZSNDREP)
```

Determining the progress of the Capture program

To determine the progress of the Capture program, you must either determine how much work remains between the last Capture process that was performed and the last Apply process, or use the DJRA Replication Monitor.

If the Capture program has ended, you can determine its progress by inspecting the warm start table. There is one row for each journal used by the source tables. The LOGMARKER column provides the timestamp of the last journal entry processed successfully. The SEQNBR column provides the journal entry sequence number of that entry.

If the Capture program is still running, you can determine its progress by completing the following tasks:

1. For each source table being captured, find its CD table.
2. In the last row of the CD table, note the hex value in the IBMSNAP_UOWID column.

3. Look in the UOW table for a row with the same IBMSNAP_UOWID value. If no matching IBMSNAP_UOWID exists in the UOW table, repeat the same process with the second-to-last row in the CD table. Proceed backward through the CD table until you find a match.
4. When you find a matching IBMSNAP_UOWID, note the value in the IBMSNAP_LOGMARKER column of the UOW row. This is the timestamp of the processed journal entry. All changes to the source table up to that time are ready to be applied.
5. Use the Display Journal (**DSPJRN**) command to determine how many journal entries remain to be processed by the Capture program. Direct the output to an output file (or to a printer for a printed report), as shown in the following example:

   ```
   DSPJRN FILE(JRNLIB/DJRN1)
          RCVRNG(*CURCHAIN)
          FROMTIME(timestamp)
          TOTIME(*LAST)
          JRNCDE(J F R C)
          OUTPUT(*OUTFILE)
          ENTDTALEN(1) OUTFILE(library/outfile)
   ```

   In the example, *timestamp* is the timestamp that you identified in step 4.

   The number of records in the output file is the approximate number of journal entries that remain to be processed by the Capture program.

Defining replication sources and subscription sets

Before you define source tables as replication sources with DPROPR/400, you must be authorized to the DPROPR/400 control tables.

There are no DPROPR/400 commands for defining replication sources and subscription sets. Use the DB2 DataJoiner Replication Administration tool (DJRA) to define replication sources and subscription sets. Before you define a table as a replication source, the source table must be journaled for both before-images and after-images, and the library where the CD table is created must have a QSQJRN journal.

When you define tables as replication sources, the CCSID attributes of CHAR, VARCHAR, GRAPHIC, and VARGRAPHIC columns in the CD table must be the same as the CCSID column attributes of the source table.

Using a relative record number (RRN) as a primary key

Many DB2 UDB for AS/400 source tables do not have a column that can be identified as a primary key column. DB2 DataPropagator requires primary key columns for the Apply program to track which updates are applied to which target table rows. To meet this requirement, DPROPR/400 supports the use of *relative record numbers (RRNs)* of source table rows as primary key columns. Both the CD table and the target table have an extra column, IBMQSQ_RRN, of type INTEGER. This column has the RRN of the source table row.

Because the RRN of a source table row does not change unless the source table is reorganized, the RRN value can be used as a primary key for the source table row if a source table is not reorganized. Any time that you reorganize a source table (to compress deleted rows, for example), DPROPR/400 performs a full refresh of all the target tables.

Important: Only the Apply program for AS/400 can be used to maintain copies that contain RRN columns, whether these copies are on an AS/400 or other target DB2 platforms.

Operating Capture for AS/400

The replication administrator user ID and users granted *CAPTURE authority can use the commands in this section to perform the following Capture for AS/400 tasks:

- Starting
- Scheduling
- Stopping
- Initializing
- Pruning

This section also describes how the Capture program handles warm and cold starts, in “Warm and cold starts” on page 175.

Starting Capture for AS/400

Use the Start DPR Capture (**STRDPRCAP**) command to start capturing changes to AS/400 database tables. Because this command processes all replication sources in the register table, make sure that the user running this command has the proper authority.

After you start the Capture program, it runs continuously until you stop it or it detects an unrecoverable error.

```
►►──STRDPRCAP──┬───────────────────────────┬──────────────────────────────────────►
               │            ┌─*YES─┐        │
               └─RESTART(───┴─*NO──┴───)───┘

►──┬──────────────────────────────────────────────────────┬──┬─────────────────────┬──►
   │       ┌─*LIBL/QZSNDPR─────────────────────────┐       │  │           ┌─1─┐      │
   └─JOBD(─┴─library-name/job-description-name─────┴──)───┘  └─DPRVSN(───┴─5─┴──)──┘

►──┬─────────────────────────┬──┬───────────────────────────────────────────────────┬──►
   │       ┌─120───┐         │  │             ┌─*DPRVSN───────┐  ┌─*IMMED────┐       │
   └─WAIT(─┴─value─┴───)───┘  └─CLNUPITV(──┼─*GLOBAL───────┼──┼─*DELAYED──┼──)────┘
                                              └─hours-to-wait─┘  └─*NO───────┘

►──┬──────────────────────────────────────────────┬─────────────────────────────────►◄
   │      ┌─*ALL──────────────────────────────┐    │
   │      │  ┌────────────────────────────┐   │    │
   │      │  ▼                            │   │    │
   └─JRN(─┴────library-name/journal-name──┴───┴──)─┘
```

Table 20. STRDPRCAP Command Parameter Definitions for AS/400

Parameter	Definition and Prompts
RESTART	Specifies how the Capture program handles warm and cold starts. ***YES** (default) The Capture program continues processing the changes from the point where it was when it ended previously. This is also known as a *warm start* and is the normal mode of operation. ***NO** The Capture program removes all information from the CD tables. The Capture program also removes all information from the UOW table when you specify JRN(*ALL). All subscriptions for affected source tables are full refreshed before change capturing resumes. This process is also known as a *cold start*. At times you might want to cold start a subset of sources. By specifying RESTART(*NO) and JRN(*library-name/journal-name*), you can cold start the Capture program for specified journals. When you cold start a subset of sources, the information in the UOW table is not removed. When you use the **STRDPRCAP** command to cold start a subset of sources, you can end the Capture program after about 15 minutes and warm start it again (this time, starting *all* the replication sources).

Table 20. STRDPRCAP Command Parameter Definitions for AS/400 (continued)

Parameter	Definition and Prompts
JOBD	Specifies the name of the job description to use when submitting the Capture program. ***LIBL/QZSNDPR** (default) Specifies the default job description provided with DPROPR/400. *library-name/job-description-name* Represents the name of the job description used for the Capture program.
DPRVSN	Specifies the version of the Capture program to start. You can specify one or both of the version levels. **1** (default) Start Version 1 of the Capture program. **5** Start Version 5 of the Capture program.
WAIT	Specifies the maximum number of seconds to wait before the Capture program checks its status. You can use this value to tune the responsiveness of the Capture program. A low value reduces the time that the Capture program takes before ending or initializing, but can have a negative effect on system performance. A higher value increases the time that the Capture program takes before ending or initializing, but can improve system performance. A value that is too high can result in decreased responsiveness while the Capture program is performing periodic processing. The amount of the decrease in responsiveness depends on the amount of change activity to source tables and the amount of other work occurring on the system. **120** (default) The Capture program waits 120 seconds. *value* The maximum number of seconds that the Capture program waits. You can specify from 60 to 6000 seconds.

Table 20. STRDPRCAP Command Parameter Definitions for AS/400 (continued)

Parameter	Definition and Prompts
CLNUPITV	Specifies the maximum amount of time before the Capture program prunes old records from the CD tables and the UOW table, if it exists. This parameter works in conjunction with the RETAIN parameter on the **CHGDPRCAPA** command. ***DPRVSN** (default) Specifies the interval. The value is ***GLOBAL**. ***GLOBAL** Specifies the interval as the same value as that of the PRUNE_INTERVAL column of the tuning parameters table. You can change this value by using the CLNUPITV parameter on the **CHGDPRCAPA** command. *hours-to-wait* Specifies the interval as a specific number of hours. ***IMMED** (default) Specifies to prune old records at the beginning of the specified interval (or immediately), and at each interval thereafter. ***DELAYED** Specifies that the Capture program prune old records at the end of the specified interval, and at each interval thereafter. ***NO** Specifies that the Capture program not prune records.
JRN	Specifies a subset of up to 50 journals that you want the Capture program to work with. The Capture program will start processing all the source tables that are currently journaled to this journal. ***ALL** Specifies that the Capture program will start working with all of the journals that have any source tables journaled to them. *library-name/journal-name* Represents the qualified name of the journal that you want the Capture program to work with.

You can run the **STRDPRCAP** command manually, or you can automatically run the command as a part of the initial program load (IPL startup program). For information on including the **STRDPRCAP** command in a startup program, see *OS/400 Work Management V4R3, SC41–5306*.

If the job description specified with the JOBD parameter uses job queue QDPR/QZSNDPR, and the DPROPR/400 subsystem is not active, the **STRDPRCAP** command starts the subsystem. If the job description is defined to use a different job queue and subsystem, you must start this subsystem manually with the Start Subsystem (**STRSBS**) command either before or after running the **STRDPRCAP** command:

```
STRSBS QDPR/QZSNDPR
```

You can set up the system to start the subsystem automatically by adding the **STRSBS** command to the program that is referred to in the QSTRUPPGM system value on your system.

Scheduling Capture for AS/400

Use the **SBMJOB** command to schedule the start of the Capture program on AS/400:

```
SBMJOB CMD('STRDPRCAP...') SCDDATE(...) SCDTIME(...)
```

Stopping Capture for AS/400

Use the End DPR Capture (**ENDDPRCAP**) command to end the Capture program.

Use this command to end the Capture program before shutting down the system. You might also want to end the program during periods of peak system use to increase the performance of other programs that run on the system.

```
►►──ENDDPRCAP──┬─────────────────────────────┬──┬─────────────────────────┬──────────►◄
               │          ┌─*CNTRLD─┐        │  │           ┌─1─┐         │
               └─OPTION(──┴─*IMMED──┴─────)──┘  └─DPRVSN(────┴─5─┴────)───┘
```

Table 21. ENDDPRCAP Command Parameter Definitions for AS/400

Parameter	Definition and Prompts
OPTION	Specifies how to end the Capture program. ***CNTRLD** (default) Specifies that the Capture program complete all tasks and then end normally. The **ENDDPRCAP** command might take longer when you specify the *CNTRLD option because the Capture program completes all of its subordinate processes before ending. ***IMMED** Specifies that the Capture program complete all tasks with the **ENDJOB OPTION(*IMMED)** command and end normally.

Table 21. ENDDPRCAP Command Parameter Definitions for AS/400 (continued)

Parameter	Definition and Prompts
DPRVSN	Specifies the version of the Capture program to end. You can specify one or both of the version levels. **1** (default) Specifies Version 1 of DPROPR/400. **5** Specifies Version 5 of DPROPR/400.

If you use the **ENDJOB** command, temporary objects might be left in the QDPR library. These objects have the types *DTAQ and *USRSPC, and are named QDPR*nnnnnn*, where *nnnnnn* is the job number of the job that used them. You can delete these objects when the job that used them (identified by the job number in the object name) is not active.

If the job QDPRCTL5 does not end long after issuing this command, use the **ENDJOB** command with *IMMED option to end this job and all the journal jobs running in the DPROPR/400 subsystem. Do not end Apply jobs running in the same subsystem if you want to end only the Capture program.

In rare cases when the job QDPRCTL5 ends abnormally, the journal jobs created by QDPRCTL5 might still be left running. The only way to end these jobs is to use the **ENDJOB** command with either the *IMMED or *CNTRLD option.

Reinitializing Capture for AS/400

The Initialize DPR Capture (**INZDPRCAP**) command initializes the Capture program by directing the Capture program to work with an updated list of source tables.

Source tables under the control of the program can change while the Capture program is running. Use the **INZDPRCAP** command to ensure that the Capture program processes the most up-to-date replication sources.

If you change the values of the tuning parameters while the Capture program is running, enter the **INZDPRCAP** command to reinitialize the program using these values.

The Capture program must be running before you run this command.

```
►►──INZDPRCAP──┬────────────────────────┬──────────────────────────────────►
               │         ┌─1─┐          │
               └─DPRVSN(─┴─5─┴─────)────┘
```

```
►─┬────────────────────────────────────────────────┬──────────────────────────►◄
  │         ┌─*ALL──────────────────────────────┐  │
  │         │ ┌─────────────────────────────┐   │  │
  │         │ ▼                             │   │  │
  └─JRN(────┴───library-name/journal-name───┴───┴──)┘
```

Table 22. INZDRCAP Command Parameter Definitions for AS/400

Parameter	Definition and Prompts
DPRVSN	Specifies the version of the Capture program to initialize. You can specify one or both of the version levels. **1** (default) Specifies Version 1 of DPROPR/400. **5** Specifies Version 5 of DPROPR/400.
JRN	Specifies a subset of up to 50 journals that you want the Capture program to work with. The Capture program will start processing all the source tables that are currently journaled to this journal. ***ALL** (default) Specifies that the Capture program works with all the journals. *library-name/journal-name* Specifies the qualified name of the journal that you want the Capture program to work with.

Pruning the change data and unit-of-work tables

The CLNUPITV parameter on the **STRDPRCAP** command specifies the maximum number of hours that the Capture program waits before pruning old records from the CD tables and the UOW table. For more information about the CLNUPITV parameter, see “Starting Capture for AS/400” on page 169.

Warm and cold starts

The value of the RESTART parameter on the **STRDPRCAP** command controls how the Capture program handles warm and cold starts.

Warm start process

Warm start information is saved in most cases. Occasionally, warm start information is not saved. In this case, the Capture program uses the CD tables, UOW table, or the pruning control table to resynchronize to the time that it was stopped.

Automatic cold starts

Sometimes the Capture program automatically switches to a cold start, even if you specified a warm start. On AS/400 systems, cold starts work on a journal-by-journal basis. So, for example, if a journal exceeds the lag limit, all replication sources using that journal are cold-started, whereas replication sources using a different journal are not cold started.

For more information about how the Capture program processes different journal entry types, see Table 23.

How the Capture program processes journal entry types

The following table describes how the Capture program processes different journal entry types.

Table 23. Capture program processing by journal entry

Journal Code[3]	Entry Type	Description	Processing
C	CM	Set of record changes committed	Insert a record in the UOW table.
C	RB	Rollback	No UOW row inserted.
F	AY	Journaled changes applied to physical file member	Issue an ASN2004 message and full refresh of file.
F	CE	Change end of data for physical file	Issue an ASN2004 message and full refresh of file.
F	CR	Physical file member cleared	Issue an ASN2004 message and full refresh of file.
F	EJ	Journaling for physical file member ended	Issue an ASN2004 message and full refresh of file.
F	IZ	Physical file member initialized	Issue an ASN2004 message and full refresh of file.
F	MD	Member removed from physical file (DLTLIB, DLTF, or RMVM)	Issue an ASN2004 message and attempt a full refresh.
F	MF	Storage for physical file member freed	Issue an ASN2004 message and full refresh of file.
F	MM	Physical file containing member moved (Rename Object (RNMOBJ) of library, Move Object (MOVOBJ) of file)	Issue an ASN200A message and attempt a full refresh.

Table 23. Capture program processing by journal entry (continued)

Journal Code[3]	Entry Type	Description	Processing
F	MN	Physical file containing member renamed (RNMOBJ of file, Rename Member (RNMM))	Issue an ASN200A message and attempt a full refresh.
F	MR	Physical file member restored	Issue an ASN2004 message and full refresh of file.
F	RC	Journaled changes removed from physical file member	Issue an ASN2004 message and full refresh of file.
F	RG	Physical file member reorganized	Issue an ASN2004 and full refresh of file only if RRN of source table is being used as propagation key.
J	NR	Identifier for next journal receivers	Reset the Capture program.
J	PR	Identifier for previous journal receivers	Increment the unique sequence number counter.
R	DL	Record deleted from physical file member	Insert a DLT record in the CD table.
R	DR	Record deleted for rollback	Insert a DLT record in the CD table.
R	PT	Record added to physical file member	Insert an ADD record in the CD table.
R	PX	Record added directly to physical file member	Insert an ADD record in the CD table.
R	UB	Before-image of record updated in physical file member	See note 1.
R	UP	After-image of record updated in physical file member	See note 1.
R	BR	Before-image of record updated for rollback	See note 2.
R	UR	After-image of record updated for rollback	See note 2.

Table 23. Capture program processing by journal entry (continued)

Journal Code[3]	Entry Type	Description	Processing

Notes:

1. The R-UP image and the R-UB image form a single UPD record in the CD table if the PARTITION_KEYS_CHG column in the register table is N. Otherwise, the R-UB image inserts a DLT record in the CD table and the R-UP image inserts an ADD record in the CD table.
2. The R-UR image and the R-BR image form a single UPD record in the CD table if the PARTITION_KEYS_CHG column in the register table is N. Otherwise, the R-BR image inserts a DLT record in the CD table and the R-UR image inserts an ADD record in the CD table.
3. The following values are used for the journal codes:

C	Commitment control operation
F	Database file operation
J	Journal or journal receiver operation
R	Operation on specific record

All other journal entry types are ignored by the Capture program.

Operating Apply for AS/400

A replication administrator user ID and users granted *APPLY authority can use the commands in this section to perform the following Apply for AS/400 tasks:

- Creating DPROPR/400 packages to use with remote systems
- Starting
- Scheduling
- Stopping

This section also describes two additional Apply program operations:

- “Using the ASNDONE exit routine for AS/400” on page 191
- “Refreshing target tables with the ASNLOAD exit routine for AS/400” on page 192

Creating DPROPR/400 packages to use with remote systems

You can use the Create DPR Packages (**CRTDPRPKG**) command to create the packages necessary to use the DPROPR/400 product with remote systems.

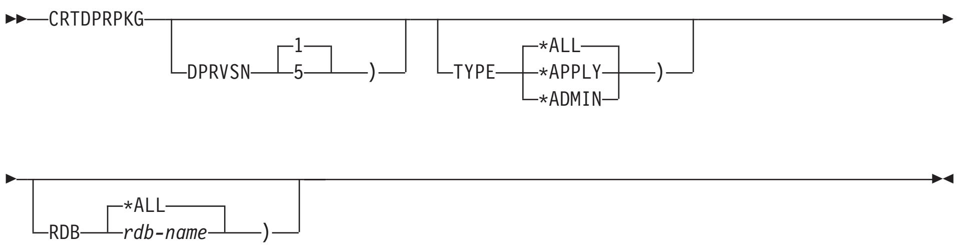


Table 24. CRTDPRPKG Command Parameter Definitions for AS/400

Parameter	Definition and Prompts
DPRVSN	Specifies the version of the DPROPR/400 package to use. You can specify one or both of the version levels: **1** (default) Specifies packages for Version 1 of DPROPR/400. **5** Specifies packages for Version 5 of DPROPR/400.
TYPE	Specifies which DPROPR/400 packages are created. ***ALL** (default) Specifies to create packages for all the DPROPR/400 programs that do remote SQL. ***APPLY** Specifies to create the packages for the programs used by the Apply program. ***ADMIN** Specifies to create the packages for the programs used by the CL commands.

Table 24. CRTDPRPKG Command Parameter Definitions for AS/400 (continued)

Parameter	Definition and Prompts
RDB	Specifies the relational database where the packages are created. If the RDB is on an AS/400 system and the ASN library does not exist on the remote system, the packages are not created. If the RDB is not on an AS/400 system and ASN is not defined as an authorization ID on that RDB, the packages are not created. ***ALL** (default) Specifies to create an SQL package on every RDB that is used as a source server or a target server by DPROPR/400. *rdb-name* Represents the name of the relational database. You can use the Work with RDB Directory Entries (**WRKRDBDIRE**) command to find this name. When prompting on the **CRTDPRPKG** command, you can press the F4 key to choose from the list of databases in the RDB directory.

The packages are created using the ASN qualifier. They are created in the ASN library for DB2 UDB for AS/400 platforms. For other platforms, the authorization ID ASN is used.

After creating the DPROPR/400 packages, this command grants *PUBLIC authority to the packages to allow them to be used by DPROPR/400 users.

The system also produces a spool file that contains the SQL messages associated with each attempt to create a package.

Before you start the Apply program

Before you start the Apply program, ensure that:

- The control tables have been created. If the tables do not exist, you can use the **CRTDPRTBL** command to create them. For information on the **CRTDPRTBL** command, see "Creating the replication control tables" on page 150.
- You have the proper authorization to run the Apply program. See "Granting authority" on page 154 for more information.
- At least one subscription set is created and activated.
- All target tables have a primary key index. Differential refresh performance is significantly affected if the primary key index is removed for a subscription.

  Important: The primary key index is built for you when you define a subscription set. Do not delete it accidentally.

- The Apply program package is created.
- The Capture program is started on the source server before you start the Apply program for the first time. The Capture program updates the SYNCHTIME and SYNCHPOINT columns of the GLOBAL record in the register table before the Apply program is started. The Apply program assumes that if a GLOBAL record is present in the register table, the SYNCHTIME and SYNCHPOINT columns are not null.

Starting Apply for AS/400

The Start DPR Apply (**STRDPRAPY**) command starts an instance of the Apply program on the local system. The Apply program continues running until you stop it or an unrecoverable error occurs.

```
►►──STRDPRAPY──┬─────────────────────────────┬──────────────────────────────────────►
               │          ┌─*CURRENT──┐      │
               └─USER(────┼─*JOBD─────┼────)─┘
                          └─user-name─┘

►──┬────────────────────────────────────────────────────────┬──┬─────────────────────┬──►
   │      ┌─*LIBL/QZSNDPR──────────────────────┐            │  │         ┌─1─┐       │
   └─JOBD(┼─library-name/job-description-name──┼───)────────┘  └─DPRVSN(─┴─5─┴────)──┘
          └─*LIBL/job-description-name─────────┘

►──┬─────────────────────────────────────┬──┬───────────────────────────┬──────────►
   │         ┌─*USER────────────┐        │  │        ┌─*LOCAL───┐        │
   └─APYQUAL(─┴─apply-qualifier──┴───)───┘  └─CTLSVR(─┴─rdb-name─┴───)───┘

►──┬──────────────────────────┬──┬──────────────────────────────────────────────────┬──►
   │       ┌─*NONE──┐         │  │             ┌─*NONE────────────────────────┐     │
   └─TRACE(─┼─*ERROR─┼────)───┘  └─FULLREFPGM(─┴─library-name/program-name────┴───)─┘
           ├─*ALL───┤
           └─*PRF───┘

►──┬─────────────────────────────────────────────────┬──┬─────────────────────────────┬──►
   │            ┌─*NONE────────────────────────┐     │  │           ┌─*YES─┐          │
   └─SUBNFYPGM(─┴─library-name/program-name────┴───)─┘  └─INACTMSG(─┴─*NO──┴──────)───┘

►──┬────────────────────────────┬──────────────────────────────────────────────────►◄
   │          ┌─*YES─┐          │
   └─ALWINACT(─┴─*NO──┴──────)───┘
```

Table 25. STRDPRAPY Command Parameter Definitions for AS/400

Parameter	Definition and Prompts
USER	Specifies the name of the user ID for which the Apply program is started. When you run this command, you must be authorized (have *USE rights) to the specified user profile. The Apply program runs under the specified user profile. The control tables (in ASN) are located on the relational database specified with the CTLSVR parameter. The same control tables are used regardless of the value specified on the USER parameter. ***CURRENT** (default) Specifies that the user ID associated with the current job is the user ID associated with this instance of the Apply program. ***JOBD** Represents the user ID specified in the job description associated with this instance of the Apply program. The job description cannot specify USER(*RQD). *user-name* (default) Specifies the user ID associated with this instance of the Apply program. The following IBM-supplied objects are *not* valid on this parameter: QDBSHR, QDFTOWN, QDOC, QLPAUTO, QLPINSTALL, QRJE, QSECOFR, QSPL, QSYS, or QTSTRQS. When prompting on the **STRDPRAPY** command, you can press the F4 key to see a list of users who defined subscriptions.
JOBD	Specifies the name of the job description to use when submitting the Apply program. ***LIBL/QZSNDPR** (default) Specifies the default job description provided with DPROPR/400. *library-name/job-description-name* Represents the name of the job description used for the Apply program.
DPRVSN	Specifies the version of the Apply program to start. You can specify one or both of the version levels. **1** (default) Start version 1 of the Apply program. **5** Start version 5 of the Apply program.

Table 25. STRDPRAPY Command Parameter Definitions for AS/400 (continued)

Parameter	Definition and Prompts
APYQUAL	Specifies that an Apply qualifier be used by an Apply program instance. All subscriptions that are grouped together with this Apply qualifier will be run by this Apply program instance. ***USER** (default) Specifies the user name on the USER parameter as the Apply qualifier. *apply_qualifier* Specifies the name used to group the subscriptions that are to be run by this Apply program instance. You can specify a maximum of 18 characters for the Apply qualifier name. This name follows the same naming conventions as an RDB name. The subscriptions to be run are identified by the records in the subscription set table with this value in the APPLY_QUAL column. When prompting on the **STRDPRAPY** command, you can press the F4 key to see a list of Apply qualifier names with existing subscriptions.
CTLSVR	Specifies the control server where the common control tables are located. ***LOCAL** (default) Specifies that the subscription control tables are located on the local relational database. *rdb-name* Represents the name of the relational database where the control tables are located. You can use the Work with RDB Directory Entries (**WRKRDBDIRE**) command to find this name. When prompting on the **STRDPRAPY** command, you can press the F4 key to see a list of available RDB names.

Table 25. STRDPRAPY Command Parameter Definitions for AS/400 (continued)

Parameter	**Definition and Prompts**
TRACE	Specifies whether the Apply program should generate a trace. If the Apply program generates a trace, the trace is output to a spool file called QPZSNATRC. ***NONE** (default) Specifies that no trace is generated. ***ERROR** Specifies that the trace should contain information for errors only. ***ALL** Specifies that the trace should contain information for errors, execution flow, and SQL statements issued by the Apply program. ***PRF** Specifies that the trace should contain information that can be used to analyze performance at different stages of the Apply program execution.
FULLREFPGM	Specifies whether the Apply program should invoke an exit routine to initialize a target table. When the Apply program determines that a target table needs to be full-refreshed, it invokes the specified exit routine rather than doing the full refresh itself. When a full-refresh exit routine is used by the Apply program, the value of the ASNLOAD column in the Apply trail table is Y. For examples and more information, see “Refreshing target tables with the ASNLOAD exit routine for AS/400” on page 192. ***NONE** (default) Specifies that a full-refresh exit routine is not used. *library-name/program-name* Represents the qualified name of the program that is called when the Apply program determines that it is necessary to do a full refresh of a target table. For example, to call program ASNLOAD in library DATAPROP, the qualified name is DATAPROP/ASNLOAD.

Table 25. STRDPRAPY Command Parameter Definitions for AS/400 (continued)

Parameter	Definition and Prompts
SUBNFYPGM	Specifies whether the Apply program is to invoke an exit routine when it finishes processing a subscription set. Input to the exit routine consists of the set name, Apply qualifier, completion status, and statistics including the number of rejects. The notify program allows you to examine the UOW table to determine the transactions that have been rejected and then allows you to take further actions such as issuing a message or generating an event. For more information, see "Using the ASNDONE exit routine for AS/400" on page 191. ***NONE** (default) Specifies that an exit routine is not used. *library-name/program-name* Represents the qualified name of the program to be called when the Apply program completes processing a subscription set. For example, to call program APPLYDONE in library DATAPROP, the qualified name is DATAPROP/APPLYDONE.
INACTMSG	Specifies whether the Apply program should generate a message whenever it completes its work and becomes inactive for a period of time. ***NO** (default) Specifies that no message is generated. ***YES** Specifies that the Apply program generate message ASN1044 before beginning a period of inactivity. Message ASN1044 indicates how long the Apply program will be inactive.
ALWINACT	Specifies whether the Apply program is able to run in an inactive state (sleep). ***YES** (default) Specifies that the Apply program should sleep if there is nothing to process. ***NO** Specifies that if the Apply program has nothing to process, the job started for the Apply program should end.

You can set up the system to automatically start the subsystem by adding the command that is referred to in the QSTRUPPGM value on your system. If you will use the QDPR/QZSNDPR subsystem, it will be started as part of the **STRDPRAPY** command processing.

Understanding subscription control table information

You can use the CTLSVR parameter to identify the control server where the subscription control tables for this Apply qualifier are found. The following control tables can be found on the control server:

- The subscription set and subscription-targets-member tables contain the information that the Apply program uses to determine which tables are copied, when they are copied, and the types of copies to make for each table.
- The subscription columns table contains the information about the columns that are to be copied to the copy table.
- The subscription statements table contains information about the SQL statements that are to be run before or after each subscription is processed.
- The subscription events table contains information about the events that might trigger a subscription to be processed.
- The Apply trail table contains information about the actions that were taken each time a subscription was processed.

If the relational database (RDB) specified with the CTLSVR parameter is a DB2 UDB for AS/400 database, the tables on the server are found in the ASN library. If the RDB is not a DB2 UDB for AS/400 database, you can access the tables using ASN as the qualifier.

Error conditions when starting the Apply program

The **STRDPRAPY** command issues an error message if any of the following conditions occur:

- If the user does not exist.
- If the user running the command is not authorized to the user profile specified on the command or the job description.
- If an instance of the Apply program is already active on the local system for this combination of Apply qualifier and control server.
- If the RDB name specified with the CTLSVR parameter is not in the relational database directory.
- If the control tables do not exist on the RDB specified with the CTLSVR parameter.
- If there are no subscriptions defined for the Apply qualifier specified with the APYQUAL parameter.

An Apply instance must be started for each unique Apply qualifier in every subscription set table. You can start multiple Apply processes by specifying a different Apply qualifier each time that you issue the **STRDPRAPY** command. These Apply processes will run under the same user profile.

Identifying Apply program jobs

Each Apply process is identified using both the Apply qualifier and the control server names. When run, the job started for the Apply process does not have sufficient external attributes to correctly identify which Apply process is associated with a particular Apply qualifier and control server combination. Therefore, the job is identified in the following way:

- The job is started under the user profile associated with the USER parameter.
- The first 10 characters of the Apply qualifier are truncated and become the job name.
- DPROPR/400 maintains an Apply job control table named IBMSNAP_APPLY_JOB in the ASN library on the local system. The table maps the Apply qualifier/control server values to the correct Apply program job.
- You can view the job log. The Apply qualifier and control server names are used in the call to the Apply program.

In general, you can identify the correct Apply program job by looking at the list of jobs running in the QZSNDPR subsystem if both:

- The first 10 characters of the Apply qualifier name are unique.
- The Apply program is started for the local control server only.

Scheduling Apply for AS/400

Use the **ADDJOBSCDE** command to start the Apply program at a specific time.

Stopping Apply for AS/400

The End DPR Apply (**ENDDPRAPY**) command ends an instance of the Apply program on the local system.

You should end the Apply program before any planned system down time. You might also want to end the Apply program during periods of peak system activity.

```
►►──ENDDPRAPY──┬───────────────────────────────┬──┬──────────────────────────────┬──────────────►
               │          ┌─*CURRENT──┐        │  │           ┌─*CNTRLD──┐       │
               └─USER(────┴─user-name─┴────)───┘  └─OPTION(───┴─*IMMED───┴────)──┘
```

```
►─┬──────────────────────┬─┬─────────────────────────────────────┬──────────────────►
  │          ┌─1─┐       │ │            ┌─*USER───────────┐      │
  └─DPRVSN(──┴─5─┴────)──┘ └─APYQUAL(───┴─apply-qualifier─┴───)──┘

►─┬───────────────────────────┬──────────────────────────────────────────────────►◄
  │          ┌─*LOCAL───┐     │
  └─CTLSVR(──┴─rdb-name─┴──)──┘
```

Table 26. ENDDPRAPY Command Parameter Definitions for AS/400

Parameter	Definition and Prompts
USER	This parameter is ignored unless the APYQUAL parameter has a value of *USER, in which case this is the Apply qualifier associated with the instance of Apply. ***CURRENT** (default) Specifies the Apply process of the user associated with the current job. *user-name* Specifies the Apply process of the specified user. When prompting on the **ENDDPRAPY** command you can press the F4 key to see a list of users who defined subscriptions.

Table 26. ENDDPRAPY Command Parameter Definitions for AS/400 (continued)

Parameter	Definition and Prompts
OPTION	Specifies how to end the Apply process. ***CNTRLD** (default) Specifies that the Apply process complete all of its tasks before ending. These tasks might take a considerable period of time if the Apply program is completing a subscription. ***IMMED** Specifies that the Apply program complete all of its tasks with the **ENDJOB OPTION(*IMMED)** command. The tasks end immediately, without any cleanup. Use this option only after a controlled end is unsuccessful, because it can cause undesirable results. (Unless the Apply program was asleep when you issued the **ENDDPRAPY** command, you should verify the target table contents.) If the Apply program was performing a full refresh to the target table, the target table might be empty as a result of ending the Apply program before the table was refreshed with the source table contents. If the target table is empty, you must force a full refresh for this replication target. You might find that a subscription is considered IN USE (the STATUS column in the subscription set table has a value of 1). If it is, reset the value to 0 or -1. This allows the subscription to be run again by the Apply program.
DPRVSN	Specifies the version of the Apply program to end. You can specify one or both of the version levels. **1** (default) Specifies version 1 of the Apply program. **5** Specifies version 5 of the Apply program.

Table 26. ENDDPRAPY Command Parameter Definitions for AS/400 (continued)

Parameter	Definition and Prompts
APYQUAL	Specifies the Apply qualifier used by an instance of the Apply program. All subscriptions that are grouped together with this Apply qualifier are run by the instance. ***USER** (default) Specifies that the user name specified on the USER parameter is the Apply qualifier. *apply_qualifier* Specifies the name used to group the subscriptions that this Apply instance runs. You can specify a maximum of 18 characters for the Apply qualifier name. This name follows the same naming conventions as an RDB name. You identify the subscriptions being run by the records in the subscription set table with this value in the APPLY_QUAL column. When prompting on the **ENDDPRAPY** command, you can press the F4 key to see a list of Apply qualifier names with existing subscriptions.
CTLSVR	Specifies the name of the relational database where the Version 5 control tables are located. ***LOCAL** (default) Specifies that the control tables are located on the local relational database. *rdb-name* Specifies that the subscription control tables are located on this relational database. You can use the Work with RDB Directory Entries (**WRKRDBDIRE**) command to find this name. When prompting on the **ENDDPRAPY** command, you can press the F4 key to choose from the list of databases in the RDB directory.

The **ENDDPRAPY** command uses the value of the APYQUAL and CTLSVR parameters to search the Apply job table for the job name, job number, and job user for the referenced Apply program, and ends that job.

ENDDPRAPY issues an error message if the following conditions occur:

- If the Apply job table does not exist or is corrupted.
- If there is no record in the Apply job table for the Apply qualifier and control server name.
- If the Apply job already ended.
- If the user ID running the command is not authorized to end the Apply job.

Additional Apply program operations

This section provides information about performing two additional Apply program functions: using the ASNDONE exit routine and refreshing target tables with the ASNLOAD exit routine.

Using the ASNDONE exit routine for AS/400

The ASNDONE exit routine is a program that the Apply program can optionally call after subscription processing completes, regardless of success or failure. A separate subscription notify program can be provided for each Apply qualifier. For general information about the ASNDONE exit routine, see “Using the ASNDONE exit routine” on page 115.

This section provides information about customizing the ASNDONE routine for an AS/400 environment.

When creating your subscription notify program, consider these activation group concerns:

If the program is created to run with a new activation group: the Apply program and the subscription notify program will not share SQL resources, such as RDB connections and open cursors. The activation handling code in the AS/400 operating system frees any resources allocated by the subscription notify program before control is returned to the Apply program. Additional resource is used every time that the Apply program calls the subscription notify program.

If the program is created to run in the caller’s activation group: it shares SQL resources with the Apply program. Design the program so that you minimize its impact on the Apply program. For example, the program might cause unexpected Apply program processing if it changes the current relational database (RDB) connection.

If the program is created to run in a named activation group: it does not share resources with the Apply program. Using a named activation group will avoid the activation group overhead every time the subscription notify program is called. Run time data structures and SQL resources can be shared between invocations. Application cleanup processing is not performed until the Apply program is ended, so the subscription notify program must be designed to ensure that it does not cause lock contention with the Apply program by leaving source tables, target tables, or control tables locked when control is returned to the Apply program.

When you start the Apply program, specify the name of the subscription notify program using the parameter SUBNFYPGM on the **STRDPRAPY** command. For example, if the program is named ASNDONE_1 and resides in library APPLIB, the parameter is coded:

```
SUBNFYPGM(APPLIB/ASNDONE_1).
```

Refreshing target tables with the ASNLOAD exit routine for AS/400

The ASNLOAD full-refresh exit routine is called by the Apply program:

- When it determines that a full refresh of a target table is necessary.
- If you specify the name of a full refresh program on the FULLREFPGM parameter when you start the Apply program.

When a full refresh of a subscription set is necessary, the Apply program calls the exit routine. The program then performs a full refresh of the target table (if necessary), or of each target table listed in the subscription set.

You can use an exit routine instead of the Apply program to perform a full refresh more efficiently. For example, if you are copying every row and every column from a source table to a target table, you can design a full-refresh exit routine that uses a Distributed Data Management (DDM) file and the Copy File (**CPYF**) CL command to copy the entire file from the source table to the target table.

If the exit routine returns a non-zero return code, the current subscription set being processed by the Apply program fails. Processing of the remainder of the subscription set is discontinued until the next iteration.

Guidelines for using ASNLOAD

The source for sample exit routines is included with DPROPR/400. The samples for the C, COBOL, and RPG languages are:

Compiler language	Library name	Source file name	Member name
C	QDPR	QCSRC	ASNLOAD
COBOL	QDPR	QCBLLESRC	ASNLOAD
RPG	QDPR	QRPGLESRC	ASNLOAD

You cannot direct the Apply program to use another program unless you end the Apply program and start it again with another **STRDPRAPY** command.

- To avoid interference with the Apply program, compile the exit routine so that it uses a new activation group (not the caller's activation group).
- The exit routine should perform a COMMIT operation.
- The system calls the exit routine to perform a full refresh of each target table associated with the subscription set. You can either:

  - Design the program to distinguish between the different target tables and subscription sets.
  - Associate a single subscription set with a single member to one Apply qualifier.
- You can compile the exit routine with a named activation group or with a new activation group. To get better performance, use a named activation group. With a named activation group, the exit routine must commit or roll back changes as needed. The Apply program will not cause changes to be committed or rolled back (unless it ends).

  The exit routine should either explicitly commit changes, or it should be compiled to implicitly commit changes when it completes. Any uncommitted changes when the exit routine completes are not committed until either:
  - The Apply program calls another exit routine with the same activation group.
  - The job started for the Apply program ends.

Required parameters for ASNLOAD

Return code
Specifies whether the exit routine was successful, indicated by a return code of 0. If the return code is not 0 the Apply program produces an error. If trace is on, the Apply program produces trace output.

Reason code
Specifies a value that can be used to further describe the exit routine failure. If the return code is not 0 and if trace is on, the Apply program includes the reason code information as part of the trace output. The values for the reason code should be specific to your user application.

Control server RDB name
Specifies the RDB name of the database where the subscription set tables are located. The name is padded with blanks.

Target server relational database (RDB) name
Specifies the name of the database where the target table is located. The name is padded with blanks.

Target table library
Specifies the name of the library that contains the target table. If the target server RDB name is not an AS/400 database, this parameter is the authorization ID of the target table, which is obtained from the TARGET_OWNER column of the row of the subscription member table that is currently processed by the Apply program. The name is padded with blanks.

Target table name
Specifies the name of the target table, which is obtained from the

TARGET_TABLE column of the row in the subscription-targets-member table that is currently processed by the Apply program. If the target server is a database for AS/400, the name can be either an SQL table name or an AS/400 system file name. The name is padded with blanks.

Control server RDB name
Specifies the RDB name of the database where the subscription tables are located. The name is padded with blanks.

Apply qualifier
Specifies the qualifier used to start this instance of the Apply program. This value is obtained from the APPLY_QUAL column of the row in the subscription set table that is currently processed by the Apply program. The name is padded with blanks.

Subscription set name
Specifies the name of the subscription set that the Apply program has just completed. This value is obtained from the SET_NAME column of the row in the subscription set table that is currently processed by the Apply program. The name is padded with blanks.

Source server RDB name
Specifies the RDB name of the database where the source table is located. The name is padded with blanks.

SQL SELECT statement
Specifies a variable length SQL statement that you can use to select the source table rows and columns to be copied to the target table. The following table shows the structure of the SQL SELECT statement.

Decimal Offset	Hex Offset	Type	Field
0	0	BINARY(4)	SQL statement length
4	4	Char(*)	SQL select statement

Trace indicator
Specifies whether the Apply program generates trace data. The exit routine can use the trace indicator to coordinate its internal trace with the Apply trace.

When the Apply program generates a trace, it prints to a spool file. If the exit routine is running in a separate activation group, the results print to a separate spool file. If the exit routine runs in the caller's activation group, the results print to the same spool file as the Apply trace.

The values for trace indicator are:

YES
Trace data is being produced.

NO
No trace data is being produced.

Other
No trace data is being produced.

Chapter 10. Capture and Apply for UNIX platforms

This chapter describes how to set up and operate the Capture and Apply programs on the following UNIX platforms:

- AIX
- HP-UX
- Solaris
- SCO UnixWare 7 (UnixWare 7)

Be sure to read the following sections before reading the sections on operating the Capture and Apply programs:

- “User ID requirements for running the Capture and Apply programs”
- “Setting up the Capture and Apply programs”

User ID requirements for running the Capture and Apply programs

Before you set up the Capture and Apply programs, you must set up a UNIX user account to run the programs and ensure that the user ID under which the Capture and Apply programs will run has the required privileges:

- Execute privilege on the Capture and Apply program packages
- DBADM or SYSADM authority for the source, control, and target servers

Setting up the Capture and Apply programs

Setting up consists of configuring the source, target, and control servers. The following sections provide instructions for configuring each server as well as information about providing end-user authentication at the source server.

Configuring the Capture program for UNIX platforms

Important: The Capture program is bound automatically during execution. Therefore, the following steps for binding the Capture program on UNIX are optional. If you want to specify options or check that all bind processes completed successfully, complete the following tasks:

1. Log on with the user ID that has sufficient privileges.
2. Connect to the source server database by entering:

   ```
   db2 connect to database
   ```

   where *database* is the source server database.

3. Prepare the source server database for roll-forward recovery by issuing the **update database configuration** command and the **backup database** command. For example:

   ```
   db2 update database configuration for database_alias using logretain on
   db2 backup database database_alias
   ```

   or:

   ```
   db2 update database configuration for database_alias using userexit on
   db2 backup database database_alias
   ```

   You might need to increase DBHEAP, APPLHEAPSZ, PCKCACHESZ, LOCKLIST, and LOGBUFSZ based on your installation requirements.
4. Change to the directory where the Capture program bind files are located, which is usually *$HOME*/sqllib/bnd.
5. Create and bind the Capture program package to the source server database by entering the following command:

   ```
   db2 bind @capture.lst isolation ur blocking all
   ```

   where UR specifies the list in uncommitted read format for greater performance.

   These commands create a list of packages, the names of which are in the file CAPTURE.LST.

Configuring the Apply program for UNIX platforms

Important: The Apply package is bound automatically during execution. Therefore, the following steps for binding the Apply package on UNIX are optional. If you want to specify options or check that all bind processes completed successfully, complete the following tasks:

1. Log on with the user ID that has sufficient privileges.
2. Change to the directory where the Apply program bind files are located, which is usually *$HOME*/sqllib/bnd.
3. Connect to the source server database by entering:

   ```
   db2 connect to database
   ```

   where *database* is the source server database.

   Note: If the source server database is catalogued as a remote database, you might need to specify a user ID and password on the **db2 connect to** command. For example:

   ```
   db2 connect to database user userid using password
   ```
4. Create and bind the Apply program package to the source server database by entering both of the following commands:

```
db2 bind @applycs.lst isolation cs blocking all
db2 bind @applyur.lst isolation ur blocking all
```

where `cs` specifies the list in cursor stability format, and `ur` specifies the list in uncommitted read format.

These commands create a list of packages, the names of which are in the files applycs.lst and applyur.lst.

5. Connect to the target server database by entering:

   ```
   db2 connect to database
   ```

   where *database* is the target server database.
6. Create and bind the Apply program package to the target server database by entering the following commands:

   ```
   db2 bind @applycs.lst isolation cs blocking all grant public
   db2 bind @applyur.lst isolation ur blocking all grant public
   ```

   Because the Apply program control tables use static SQL calls for the control tables, the Apply bind process searches for the control tables at each server that the Apply program is bound to, regardless of whether these control tables are used at a server.
7. Repeat the connect and bind steps for each server that the Apply program connects to. You must bind the Apply program to all source, target, and control servers.

Other configuration considerations for UNIX-based components

Ensure that the user ID from which the Capture and Apply programs are running has write privilege on the directories where you invoke the programs. Write privilege is necessary because both the Capture and Apply programs create files in the invocation directory.

The Capture program creates the following files in addition to the spill files:

- *instnameSRCSRVR*.ccp is a log file for the messages issued by the Capture program. These messages are also recorded in the trace table.
- *instnameSRCSRVR*.tmp contains the process ID of this invocation of the Capture program (to prevent multiple Capture programs from being started in the same instance to the same source server).

The Apply program creates the following files:

- *APPLYQUAL*.app is a log file for the messages issued by the Apply program. These messages are also recorded in the Apply trail table.

- ASNAPPLY*APPLYQUAL*.pid contains the process ID of this invocation of the Apply program (to prevent multiple Apply programs from being started with the same Apply qualifier).

For more information about configuration of UNIX-based components, see *IBM DB2 Universal Database for UNIX Quick Beginnings.*

Providing end-user authentication at the source server

In some cases you must provide a password file for end-user authentication to occur at the source server. The Apply program uses this file when connecting to the source server. Give read access only to the user ID that will run the Apply program. Following are environment-specific requirements:

- If you installed Apply for HP-UX, Apply for Solaris, or Apply for UnixWare 7, you must use an AUTH=SERVER scheme and provide a password file.
- If you installed Apply for AIX, you must provide a password file if you want to use an AUTHENTICATION=SERVER scheme at any server that the Apply program will connect to. If you use an AUTHENTICATION=CLIENT scheme for all servers, you do not need to provide a password file.

Creating a password file:

The password file must meet the following criteria:

- Be named as shown:

  applyqual.PWD

  Where *applyqual* is a case-sensitive string that must match the case and value of the Apply qualifier (APPLY_QUAL) in the subscription set table exactly.

  For example: DATADIR.PWD

  This naming convention is the same as the log file name (.app) and the spill file name (.*nnn*), but with a file extension of .pwd.
- Reside in the same directory from which you will start the Apply program.
- Do not put blank lines or comment lines in this file. Only add the server-name, user ID, and password information. This information enables you to use different passwords or the same password at each server.
- Have one or more records using the following format:

  SERVER=*server_name* USER=*userid* PWD=*password*

  Where:

server_name
: The name of the source, target, or control server, exactly as it appears in the subscription set table.

userid
: The user ID that you plan to use to administer that particular server. This value is case-sensitive.

password
: The password that is associated with the *userid*. This value is case-sensitive.

If you do not create a password file:

The Apply program for UNIX must be able to issue an SQL CONNECT statement without specifying the user ID and password. If the Apply program needs to connect to an OS/390 database with SNA connectivity, these settings are necessary:

- The DB2 for OS/390 database must be catalogued as AUTHENTICATION=CLIENT.
- The login ID must belong to PRIMARY GROUP=SYSTEM.

When you copy from DB2 for OS/390 sources, these settings are necessary:

- SECURITY=SAME for MVS CPI-C node.
- You specify the following values when you define the LU name by using the VTAM APPL:
  - VERIFY=NONE to indicate that any LU can request an LU-LU session
  - SECACPT=ALREADYV to indicate user ID and password checking at the requester

For more information about authentication and security, refer to the *IBM DB2 Universal Database Administration Guide.*

Operating Capture for UNIX platforms

An administrator can use the commands in this section to operate the Capture program for UNIX platforms. Enter the commands or a key combination from an AIX, HP-UX, Solaris, or UnixWare 7 window.

This section explains how to perform the following Capture program tasks:

- Starting
- Scheduling
- Stopping
- Suspending

- Resuming
- Reinitializing
- Pruning
- Displaying captured log progress

This section also lists restrictions for running the Capture program.

Restrictions for running the Capture program

Some actions cause the Capture program to terminate while it is running. Stop the Capture program before you take any of the following actions:

- Remove an existing replication source.
- Drop a replication source table.
- Make changes that affect the structure of source tables, such as changes resulting from data definition language or utilities. Structural changes can compromise the data integrity of the copies. (ALTER ADDs of new columns are an exception.)

The Capture program cannot capture any changes made by DB2 utilities, because the utilities do not log changes they make.

Starting Capture for UNIX platforms

After you start the Capture program, it runs continuously until you stop it or it detects an unrecoverable error.

To start the Capture program for UNIX:

1. Log in and make sure that the user ID under which the Capture program is running has write privilege on the directory.
2. Ensure that you set the DB2 instance name as shown:
   ```
   export DB2INSTANCE=db2_instance_name
   ```
   While the Capture program is running, a file with the name <*database_instance_name*><*database_name*>.ccp is created in the directory from which the Capture program is started. This file is a log file for the messages issued by the Capture program; these messages are also recorded in the trace table.
3. Set the LIBPATH environment variable or edit the .profile file in the same environment in which the Capture program starts.

   AIX example:
   ```
   export LIBPATH=db2instance_home_directory/sqllib/lib:/usr/lib:/lib
   ```
   HP-UX example:
   ```
   export SHLIB_PATH=db2instance_home_directory/sqllib/lib:/usr/lib:/lib
   ```

Solaris and UnixWare 7 example:

```
export LD_LIBRARY_PATH=db2instance_home_directory/sqllib/lib:/usr/lib:/lib
export NLS_PATH=/usr/lib/locale/%L/%N:/usr/lib/locale/prime/%N
```

where *db2instance_home_directory* is the name of the DB2 instance's home directory.

4. Enter the following command:

```
►►──asnccp──┬────────────┬──┬─────────┬──┬─────────┬──┬─────────┬──┬─────────┬──────────►
            └─src_server─┘  ├─warm────┤  ├─prune───┤  ├─notrace─┤  └─trcfile─┘
                            ├─warmns──┤  └─noprune─┘  └─trace───┘
                            └─cold────┘

►──┬──────────┬──┬──────────┬──┬──────────┬──┬───────────┬──┬─────────┬─────────────►◄
   └─notrctbl─┘  └─autostop─┘  └─logreuse─┘  └─logstdout─┘  ├─allchg──┤
                                                            └─chgonly─┘
```

Table 27 defines the invocation parameters.

Table 27. ASNCCP Command Invocation Parameter Definitions for UNIX Platforms

Parameter	Definition
src_server	Source server name must be the first parameter if entered. If not specified, the value from the DB2DBDFT environment variable is used.
warm (default)	The Capture program resumes processing where it ended in its previous run if warm start information is available. If the Capture program cannot warm start, it switches to a cold start.
warmns	The Capture program resumes processing where it ended in its previous run if warm start information is available. Otherwise, it issues a message and terminates. With warmns, the Capture program does not automatically switch to a cold start. The Capture program leaves the trace, UOW, CD, and warm start tables intact. In case of errors, the Capture program terminates instead of switching to a cold start as when warm is specified.
cold	The Capture program starts by deleting all rows in its CD table, UOW table, and trace table during initialization. All subscriptions to these replication sources will be fully refreshed during the next Apply program processing cycle.

Table 27. ASNCCP Command Invocation Parameter Definitions for UNIX Platforms (continued)

Parameter	Definition
prune (default)	The Capture program automatically prunes the rows in the CD and UOW tables that the Apply program has copied, at the interval specified in the tuning parameters table.
noprune	Automatic pruning is disabled. The Capture program prunes the CD and the UOW tables when you enter the **prune** command.
notrace (default)	No trace information is written.
trace	Writes trace messages to the standard output, stdout, unless trcfile is also specified.
trcfile	If both trcfile and trace are specified, the Capture program writes trace output to the trace file (*.trc). If you do not specify this option, the Capture program sends trace output to the standard output, stdout.
notrctbl	The Capture program messages are not logged in the trace table.
autostop	The Capture program terminates after it has captured all transactions logged before the Capture program was started.
logreuse	The Capture program reuses the log file (*.ccp) by first deleting it and then re-creating it when the Capture program is restarted. If you do not specify this option, the Capture program appends messages to the log file, even after the Capture program is restarted.
logstdout	The Capture program sends all messages to both the standard output (stdout) and the log file.
allchg (default)	Specifies that an entry is made to the CD table whenever a source table row changes.
chgonly	Specifies that an entry is made to the CD table when a source table row changes only if the columns defined for replication (CD table columns) change values.

Scheduling Capture for UNIX platforms

Use the **at** command to start the Capture program at a specific time. For example, the following command starts the Capture program at 3:00 p.m. on Friday:

```
at 3pm Friday asnccp warmns noprune
```

Stopping Capture for UNIX platforms

Use the **stop** command or a key combination to stop the Capture program in an orderly way and commit the log records that it processed up to that point.

Stop the Capture program before removing or modifying an existing replication source.

```
►►──asncmd──stop─────────────────────────────────────────────────────►◄
```

or

```
►►──asncmd──src_server──stop─────────────────────────────────────────►◄
```

To use the **stop** command, do the following from a window where the Capture program is not running:

1. Set environment variable DB2INSTANCE to the value set when the Capture program was started.
2. Set environment variable DB2DBDFT to the source server specified when the Capture program was started (or the DB2DBDFT value used when the Capture program was started).

   Note: You do not need to set the value of DB2DBDFT if you specify a value for *src_server* when you run the command.
3. Enter the command.

Attention: Follow the 3 steps listed above to enter all of the Capture program commands.

Suspending Capture for UNIX platforms

Use the **suspend** command to relinquish operating system resources to operational transactions during peak periods without destroying the Capture program environment. This command suspends the Capture program until you issue the **resume** command.

```
►►──asncmd──suspend──────────────────────────────────────────────────►◄
```

or

```
►►──asncmd──src_server──suspend──────────────────────────────────────────────►◄
```

Important: Do not use the **suspend** command when canceling a replication source. Instead, stop the Capture program.

Resuming Capture for UNIX platforms

Use the **resume** command to restart the Capture program if you suspended it using the **suspend** command.

```
►►──asncmd──resume───────────────────────────────────────────────────────────►◄
```

or

```
►►──asncmd──src_server──resume───────────────────────────────────────────────►◄
```

Reinitializing Capture for UNIX platforms

Use the **reinit** command to begin to capture changes from new source tables if you add a new replication source or ALTER ADD a column to a replication source and CD table while the Capture program is running. The **reinit** command tells the Capture program to obtain newly added replication sources from the register table.

reinit also rereads the tuning parameters table for any changes made to the tuning parameters.

```
►►──asncmd──reinit───────────────────────────────────────────────────────────►◄
```

or

```
►►──asncmd──src_server──reinit───────────────────────────────────────────────►◄
```

Important: Do not use the **reinit** command to reinitialize the Capture program after canceling a replication source or dropping a replication source table while the Capture program is running. Instead, stop the Capture program and restart it using the WARM or WARMNS option.

Pruning the change data and unit-of-work tables

Use the **prune** command to start pruning the CD and UOW tables.

This command prunes tables once.

```
►►──asncmd──prune────────────────────────────────────────────────────►◄
```

or

```
►►──asncmd──src_server──prune────────────────────────────────────────►◄
```

The Capture program issues the message ASN0124I when the command is successfully queued.

During pruning, if you stop or suspend the Capture program, pruning does not resume after you enter the **resume** command. You must enter the **prune** command again to resume pruning.

Displaying captured log progress

Use the **getlseq** command to provide the timestamp and current log sequence number. You can use this number to determine how far the Capture program has read the DB2 log.

```
►►──asncmd──getlseq──────────────────────────────────────────────────►◄
```

or

```
►►──asncmd──src_server──getlseq──────────────────────────────────────►◄
```

Tip: The DB2 UDB Find Log Sequence Number command (**db2lfsn**) enables you to identify the physical log file associated with the log sequence number. You can use this number to delete or archive log files no longer needed by the Capture program. For more information, see the *IBM DB2 Universal Database Command Reference*.

Operating Apply for UNIX platforms

An administrator can use the commands in the following sections to perform the following Apply program tasks:

- Starting
- Scheduling
- Stopping

Before you start the Apply program

Before you start the Apply program, ensure that:

- You have the proper authorization. See “Authorization requirements for running the Apply program” on page 92 for information about authorization for the Apply program.
- The control tables are defined.
- At least one subscription is created and activated.
- The Apply package is created.
- A password file has been created, if necessary, for end-user authentication at the source server. See “Providing end-user authentication at the source server” on page 200 for more information.
- The Capture program is started, and the ASN0100I initialization message was issued (if you are running a Capture program).

Starting Apply for UNIX platforms

After you start the Apply program, it runs continuously until:

- You stop it in an orderly way.
- You cancel it.
- An unexpected error or failure occurs.

To start the Apply program on UNIX:

1. Log on with the IBM Replication user ID.
2. Ensure that you set the DB2 instance name as shown:
   ```
   export DB2INSTANCE=db2_instance_name
   ```
3. Set the LIBPATH environment variable or edit the .profile file in the same environment in which the Apply program starts.

   AIX example:
   ```
   export LIBPATH=db2instance_home_directory/sqllib/lib:/usr/lib:/lib
   ```

   HP-UX example:
   ```
   export SHLIB_PATH=db2instance_home_directory/sqllib/lib:/usr/lib:/lib
   ```

Solaris and UnixWare 7 example:

```
export LD_LIBRARY_PATH=db2instance_home_directory/sqllib/lib:/usr/lib:/lib
export NLS_PATH=/usr/lib/locale/%L/%N:/usr/lib/locale/prime/%N
```

where *db2instance_home_directory* is the name of the DB2 instance's home directory.

4. Enter the **asnapply** command and options:

```
►►──asnapply──apl_qual──┬───────────┬──┬───────────┬──┬───────────┬──┬─────────┬──────────►
                        └─ctrl_serv─┘  ├─loadxit───┤  ├─inamsg────┤  ├─notrc───┤
                                       └─noloadxit─┘  └─noinamsg──┘  ├─trcerr──┤
                                                                     └─trcflow─┘

►──┬─────────┬──┬──────────┬──┬─────────┬──┬──────────┬──┬──────────┬──┬──────────┬──────►
   └─trcfile─┘  ├─notify───┤  ├─sleep───┤  └─delay(n)─┘  └─copyonce─┘  └─logreuse─┘
                └─nonotify─┘  └─nosleep─┘

►──┬───────────┬──┬──────────┬───────────────────────────────────────────────────────►◄
   └─logstdout─┘  └─trlreuse─┘
```

Table 28 defines the invocation parameters.

Table 28. ASNAPPLY Invocation Parameter Definitions for UNIX Platforms

Parameter	Definition
apl_qual	Specifies the Apply qualifier that the Apply program instance uses to identify the subscriptions to be served. The Apply qualifier is case sensitive and must match the value of APPLY_QUAL in the subscription set table. This must be the first parameter.
ctrl_serv	Specifies the name of the server where the replication control tables will reside. If you do not specify this parameter, the default is the default database or the value of DB2DBDFT.
loadxit	Specifies that the Apply program is to invoke ASNLOAD, an IBM-supplied exit routine that uses the export and load utilities to refresh target tables.
noloadxit (default)	Specifies that the Apply program will not invoke ASNLOAD.
inamsg (default)	Specifies that the Apply program is to issue a message when the Apply program is inactive.
noinamsg	Specifies that the Apply program will not issue this message.

Table 28. ASNAPPLY Invocation Parameter Definitions for UNIX Platforms (continued)

Parameter	Definition
notrc (default)	Specifies that the Apply program does not generate a trace.
trcerr	Specifies that the Apply program generates a trace that contains only error information.
trcflow	Specifies that the Apply program generates a trace that contains both error and execution flow information.
trcfile	If both trcfile and trace are specified, the Apply program writes trace output to the trace file (*.trc). If you do not specify this option, the Apply program sends trace output to the standard output, stdout.
notify	Specifies that the Apply program is to invoke ASNDONE, an exit routine that returns control to the user when the Apply program processing ends.
nonotify (default)	Specifies that the Apply program will not invoke ASNDONE.
sleep (default)	Specifies that the Apply program is to go to sleep if no new subscriptions are eligible for processing.
nosleep	Specifies that the Apply program is to stop if no new subscriptions are eligible for processing.
delay(*n*)	Where *n*=0, 1, 2, 3, 4, 5, or 6. Specifies the delay time (in seconds) at the end of each Apply program cycle when continuous replication is used. The default delay time is 6 seconds.
copyonce	The Apply program executes one copy cycle for each eligible subscription set and then terminates. An eligible subscription set is such that: • ACTIVATE > 0 • REFRESH_TIMING = R or B or REFRESH_TIMING = E and the specified event has occurred. MAX_SYNCH_MINUTES and END_OF_PERIOD are honored if specified.
logreuse	The Apply program reuses the log file (*.app) by first deleting it and then re-creating it when the Apply program is restarted. If you do not specify this option, the Apply program appends messages to the log file, even after the Apply program is restarted.
logstdout	The Apply program sends all messages to both the standard output (stdout) and the log file.
trlreuse	The Apply program empties the Apply trail table when the Apply program is started.

Scheduling Apply for UNIX platforms

Use the **at** command to start the Apply program at a specific time. For example, the following command starts the Apply program at 3:00 p.m. on Friday:

```
at 3pm Friday asnapply myqual
```

Stopping Apply for UNIX platforms

Use the **asnastop** command or a key combination to stop the Apply program in an orderly way.

```
►►──asnastop──apply_qualifier─────────────────────────────────────────────►◄
```

To use the command, do the following from a window where the Apply program is not running:

1. Set environment variable DB2INSTANCE to the value set when the Apply program was started.
2. Set environment variable DB2DBDFT to the source server specified when the Apply program was started (or the DB2DBDFT value used when the Apply program was started).
3. Enter the command.

Chapter 11. Capture and Apply for Windows and OS/2

This chapter describes how to set up and operate the Capture and Apply programs on the following platforms:

- Windows 95
- Windows 98
- Windows NT
- OS/2

Be sure to read the following sections before reading the sections on operating the Capture and Apply programs:

- “User ID requirements for running the Capture and Apply programs”
- “Setting up the Capture and Apply programs”

User ID requirements for running the Capture and Apply programs

The user ID under which the Capture and Apply programs will run must have the following privileges:

- Execute privilege on the Capture and Apply program packages
- DBADM or SYSADM authority for the source, control, and target servers

Setting up the Capture and Apply programs

Setting up consists of configuring the source, target, and control servers, and setting up NT services on Windows. The following sections provide instructions for configuring each server, providing end-user authentication at the source server, and setting up the NT Service Control Manager.

Configuring Capture for Windows and OS/2

Important: The Capture program is bound automatically during execution. Therefore, the following steps for binding the Capture program on Windows and OS/2 are optional. If you want to specify options or check that all bind processes completed successfully, complete the following tasks:

1. Log on with the user ID that has sufficient privileges.
2. Connect to the source server database by entering:
   ```
   DB2 CONNECT TO database
   ```
   where *database* is the source server database.

3. Prepare the source server database for roll-forward recovery by issuing the **UPDATE DATABASE CONFIGURATION** command and the **BACKUP DATABASE** command. For example:

   ```
   DB2 UPDATE DATABASE CONFIGURATION FOR database_alias USING LOGRETAIN ON
   DB2 BACKUP DATABASE database_alias
   ```

   or:

   ```
   DB2 UPDATE DATABASE CONFIGURATION FOR database_alias USING USEREXIT ON
   DB2 BACKUP DATABASE database_alias
   ```

   You might need to increase DBHEAP, APPLHEAPSZ, PCKCACHESZ, LOCKLIST, and LOGBUFSZ based on your installation requirements.
4. Change to the directory where the Capture program bind files are located, which is usually *drive*:\SQLLIB\BND.
5. Create and bind the Capture program package to the source server database by entering the following command:

   ```
   DB2 BIND @CAPTURE.LST ISOLATION UR BLOCKING ALL
   ```

   where UR specifies the list in uncommitted read format for greater performance.

   These commands create a list of packages, the names of which are in the file CAPTURE.LST.

Configuring Apply for Windows and OS/2

Important: The Apply package is bound automatically during execution. Therefore, the following steps for binding the Apply package on Windows and OS/2 are optional. If you want to specify options or check that all bind processes completed successfully, complete the following tasks:

1. Log on with the user ID that has sufficient privileges.
2. Change to the directory where the Apply program bind files are located, which is usually *drive*:\SQLLIB\BND.
3. Connect to the source server database by entering:

   ```
   DB2 CONNECT TO database
   ```

   where *database* is the source server database.

   Note: If the source server database is catalogued as a remote database, you might need to specify a user ID and password on the **DB2 CONNECT TO** command. For example:

   ```
   DB2 CONNECT TO database USER userid USING password
   ```
4. Create and bind the Apply program package to the source server database by entering the following commands:

```
DB2 BIND @APPLYCS.LST ISOLATION CS BLOCKING ALL
DB2 BIND @APPLYUR.LST ISOLATION UR BLOCKING ALL
```

where `cs` specifies the list in cursor stability format, and `ur` specifies the list in uncommitted read format.

These commands create a list of packages, the names of which are in the files APPLYCS.LST and APPLYUR.LST.

5. Connect to the target server database by entering:

   ```
   DB2 CONNECT TO database
   ```

   where *database* is the target server database.
6. Create and bind the Apply program package to the target server database by entering both of the following commands:

   ```
   DB2 BIND @APPLYCS.LST ISOLATION CS BLOCKING ALL GRANT PUBLIC
   DB2 BIND @APPLYUR.LST ISOLATION UR BLOCKING ALL GRANT PUBLIC
   ```

   Because the Apply program uses static SQL calls for the control tables, the Apply bind process searches for the control tables at each server that the Apply program is bound to, regardless of whether these control tables are used at a server.
7. Repeat the connect and bind steps for each server that the Apply program connects to. You must bind the Apply program to all source, target, and control servers.

Providing end-user authentication at the source server

For end-user authentication to occur at the source server, you must provide a password file with an AUTH=SERVER scheme. The Apply program uses this file when connecting to the source server. Give read access only to the user ID that will run the Apply program.

Creating a password file:

The password file must meet the following criteria:

- Be named as shown:

  ```
  applyqual.PWD
  ```

  Where *applyqual* is a case-sensitive string that must match the case and value of the Apply qualifier (APPLY_QUAL) in the subscription set table exactly.

  For example: `DATADIR.PWD`

This naming convention is the same as the log file name (.app) and the spill file name (.*nnn*), but with a file extension of .PWD.

- Reside in the directory from which you will start the Apply program.
- Do not put blank lines or comment lines in this file. Only add the server-name, user ID, and password information. This information enables you to use different passwords or the same password at each server.
- Have one or more records using the following format:

  ```
  SERVER=server_name USER=userid PWD=password
  ```

  Where:

  server_name
  : The source, target, or control server, exactly as it appears in the subscription set table.

  userid
  : The user ID that you plan to use to administer that particular server. On Windows, this value is case-sensitive.

  password
  : The password that is associated with the *userid*. On Windows, this value is case-sensitive.

For more information about authentication and security, refer to the *IBM DB2 Universal Database Administration Guide*.

Setting up the NT Service Control Manager

You can operate the Capture and Apply programs for Windows by using the DB2 command processor or by using the NT Service Control Manager (SCM). The SCM enables you to automatically start the Capture and Apply programs as services from the NT Control Panel.

You must install the replication service manually (installation is not automatic). The following steps explain how to install the replication service and set it up as an NT service.

Tip: In this section, *x:*\ refers to the drive and directory containing executable programs. These programs are usually located in the \sqllib\bin directory.

To install replication and set up the NT service:

1. Open a command window, and change to the directory containing the executable file ASNINST.EXE.
2. Install the replication service by typing the following command:

   ```
   ASNINST x:\ASNSERV.EXE
   ```
3. Set up the service from the NT Control Panel.

   a. Double-click the **Services** icon. The NT Services window opens.
   b. Select **Replication** and click **STARTUP**.
   c. Ensure that the startup type is automatic.
   d. Specify the local user ID and password and click **OK**. The user ID must be the one that runs the Capture and Apply programs and has the appropriate DB2 privileges.
4. Add the environment variable ASNPATH to specify the location of the Capture and Apply program files.
   a. Double-click the **System** icon on the NT Control Panel. The System Properties window opens.
   b. Click the **Environment** tab.
   c. Type the ASNPATH string in the **Variable** field as shown in the following example:

      `ASNPATH=`*`x:\`*
   d. Click **OK**.
   e. You must reboot the computer after updating the value of the ASNPATH environment variable.
5. Create an ASCII file called `ntserv.asn` to run the Capture and Apply programs.
   a. Enter the following records in the file:

      db_name x:`\ASNCCP` *parameters*

      db_name x:`\ASNAPPLY` *parameters*

      where *db_name* specifies the name of the source database for the Capture program and the name of the control database for the Apply program, *x:\* is the location of the programs, and *parameters* specifies one or more invocation parameters (such as APPLYQUAL).

      To use the Capture program and Apply program trace facilities, specify the invocation parameters in the file. For example:

      ```
      DBNAME1 C:\SQLLIB\BIN\ASNCCP COLD TRACE<CRLF>
      DBNAME2 C:\SQLLIB\BIN\ASNAPPLY APPLYQUAL DBNAME2 TRCFLOW TRCFILE<CRLF>
      ```

      The TRCFILE invocation parameter is necessary, in addition to the usual trace invocation parameter (such as TRCFLOW), to generate an Apply program trace.

      Do not specify an output file name for traces. These will be written to default locations, with default file names, as follows:
      - For the Capture program:

        `x:\instancenamedbname``.trc`
      - For the Apply program:

```
x:\APPLYtimestamp.trc
```

b. Save the file to the following location:

```
x:\ntserv.asn
```

The Replication Services program stores all messages in *x:*\asnserv.log. If you encounter any problems, check this log file for error messages.

To stop the Capture and Apply programs:

Important: After you start the service, the Capture and Apply programs run independently of ASNSERV. Therefore, stopping ASNSERV does not stop the Capture and Apply programs. Use the **ASNCMD STOP** command in a command window to stop the Capture program. Use the **ASNASTOP** command in a command window to stop the Apply program.

To remove replication from the NT service:

To remove Replication Services from the NT Control Panel, run the ASNREMV program.

Operating Capture for Windows and Capture for OS/2

An administrator can use the commands in this section to operate the Capture program on Windows and the Capture program on OS/2. Enter the commands or a key combination from an NT or an OS/2 window.

This section explains how to perform the following Capture program tasks:

- Starting
- Scheduling
- Stopping
- Suspending
- Resuming
- Reinitializing
- Pruning
- Displaying captured log progress

This section also lists restrictions for running the Capture program.

Restrictions for running the Capture program

Some actions cause the Capture program to terminate while it is running. Stop the Capture program before you take any of the following actions:

- Remove an existing replication source.

- Drop a replication source table.
- Make changes that affect the structure of source tables, such as changes resulting from data definition language or utilities. Structural changes can compromise the data integrity of the copies. (ALTER ADDs of new columns are an exception.)

The Capture program cannot capture any changes made by DB2 utilities, because the utilities do not log changes they make.

Starting Capture for Windows and OS/2

After you start the Capture program, it runs continuously until you stop it or it detects an unrecoverable error.

To start the Capture program using the NT Services:

1. Select **Replication** from the NT Services window.
2. Click the **START** push button. The Capture program starts according to the ASCII file information you provided.

You can also start the replication service by typing STRTSERV on the NT command line.

To start the Capture program using the DB2 command window:

1. If you created one or more DB2 for NT or DB2 for OS/2 instances, use the **SET** command to set the DB2INSTANCE environment variable to the DB2 for NT or DB2 for OS/2 instance with which you plan to run the Capture program:

   ```
   SET DB2INSTANCE=database_instance_name
   ```

   While the Capture program is running, a file with the name <*database_instance_name*><*database_name*>.CCP (Windows) or *database_name*.CCP (OS/2) is created in the directory from which the Capture program is started. This file is a log file for the messages issued by the Capture program; these messages are also recorded in the trace table.
2. Set environment variable DB2DBDFT to the source server specified when the Capture program was started (or the DB2DBDFT value used when the Capture program was started).
3. To start the Capture program, enter the **ASNCCP** command from the Windows or OS/2 window where you issued the **SET** command. The syntax is:

```
►►──ASNCCP──┬────────────┬──┬────────┬──┬─────────┬──┬──────────┬──┬─────────┬──────────►
            └─src_server─┘  ├─WARM───┤  ├─PRUNE───┤  ├─NOTRACE──┤  └─TRCFILE─┘
                            ├─WARMNS─┤  └─NOPRUNE─┘  └─TRACE────┘
                            └─COLD───┘

►──┬──────────┬──┬──────────┬──┬──────────┬──┬───────────┬──┬─────────┬──────────────────►◄
   └─NOTRCTBL─┘  └─AUTOSTOP─┘  └─LOGREUSE─┘  └─LOGSTDOUT─┘  ├─ALLCHG──┤
                                                            └─CHGONLY─┘
```

Table 29 defines the invocation parameters.

Table 29. ASNCCP Command Invocation Parameter Definitions for Windows and OS/2 Platforms

Parameter	Definition
src_server	Source server name must be the first parameter if entered. If not specified, the value from the DB2DBDFT environment variable is used.
WARM (default)	The Capture program resumes processing where it ended in its previous run if warm start information is available. If the Capture program cannot warm start, it switches to a cold start.
WARMNS	The Capture program resumes processing where it ended in its previous run if warm start information is available. Otherwise, it issues a message and terminates. With warmns, the Capture program does not automatically switch to a cold start. The Capture program leaves the trace, UOW, CD, and warm start tables intact. In case of errors, the Capture program terminates instead of switching to a cold start as when warm is specified.
COLD	The Capture program starts by deleting all rows in its CD table, UOW table, and trace table during initialization. All subscriptions to these replication sources will be fully refreshed during the next Apply program processing cycle.
PRUNE (default)	The Capture program automatically prunes the rows in the CD and UOW tables that the Apply program has copied, at the interval specified in the tuning parameters table.
NOPRUNE	Automatic pruning is disabled. The Capture program prunes the CD and the UOW tables when you enter the **PRUNE** command.
NOTRACE (default)	No trace information is written.

Table 29. ASNCCP Command Invocation Parameter Definitions for Windows and OS/2 Platforms (continued)

Parameter	Definition
TRACE	Writes trace messages to the standard output, stdout, unless TRCFILE is also specified.
TRCFILE	If both trcfile and trace are specified, the Capture program writes trace output to the trace file (*.trc). If you do not specify this option, the Capture program sends trace output to the standard output, stdout.
NOTRCTBL	The Capture program messages are not logged in the trace table.
AUTOSTOP	The Capture program terminates after it has captured all transactions logged before the Capture program was started.
LOGREUSE	The Capture program reuses the log file (*.ccp) by first deleting and then re-creating it when the Capture program is restarted. If you do not specify this option, the Capture program appends messages to the log file, even after the Capture program is restarted.
LOGSTDOUT	The Capture program sends all messages to both the standard output (stdout) and the log file.
ALLCHG (default)	Specifies that an entry is made to the CD table whenever a source table row changes.
CHGONLY	Specifies that an entry is made to the CD table when a source table row changes only if the columns defined for replication (CD table columns) change values.

Scheduling Capture for Windows and OS/2

For Windows: Use the **AT** command to start the Capture program at a specific time. For example, the following command string starts the Capture program for Windows at 15:00:

```
c:\>AT 15:00 /interactive "c:\SQLLIB\BIN\db2cmd.exe c:\CAPTURE\asnccp.exe warmns"
```

Before you enter the **AT** command, the Windows Schedule Service should already be started.

For OS/2: Use the Alarms program in the OS/2 Productivity set to start the Capture program for OS/2 at a specific time.

Stopping Capture for Windows and OS/2

Use the **STOP** command or a key combination to stop the Capture program in an orderly way and commit the log records that it processed up to that point.

Stop the Capture program before removing for modifying an existing replication source.

For Windows: If you started the Capture program as an NT service, stop the Capture program by selecting **Replication** from the NT Services window and clicking the **Stop** push button. After the stop message appears, the status field becomes blank.

For Windows and OS/2: If you started the Capture program from the DB2 command window, enter the following command:

```
►►──ASNCMD──STOP──────────────────────────────────────────────────────────►◄
```

or

```
►►──ASNCMD──src_server──STOP──────────────────────────────────────────────►◄
```

To use the command, do the following from a window where the Capture program is not running:

1. Set environment variable DB2INSTANCE to the value set when the Capture program was started.
2. Set environment variable DB2DBDFT to the source server specified when the Capture program was started (or the DB2DBDFT value used when the Capture program was started).

   Note: You do not need to set the value of DB2DBDFT if you specify a value for *src_server* when you run the command.
3. Enter the command.

Attention: Follow the same 3 steps listed above to enter all of the Capture program commands.

Suspending Capture for Windows and OS/2

Use the **SUSPEND** command to relinquish operating system resources to operational transactions during peak periods without destroying the Capture program environment. This command suspends the Capture program until you issue the **RESUME** command.

```
►►──ASNCMD──SUSPEND──────────────────────────────────────────────────►◄
```

or

```
►►──ASNCMD──src_server──SUSPEND──────────────────────────────────────►◄
```

Important: Do not use the **SUSPEND** command when canceling a replication source. Instead, stop the Capture program.

Resuming Capture for Windows and OS/2

Use the **RESUME** command to restart the Capture program if you suspended it using the **SUSPEND** command.

```
►►──ASNCMD──RESUME───────────────────────────────────────────────────►◄
```

or

```
►►──ASNCMD──src_server──RESUME───────────────────────────────────────►◄
```

Reinitializing Capture for Windows and OS/2

Use the **REINIT** command to begin to capture changes from new source tables if you add a new replication source or ALTER ADD a column to a replication source and CD table while the Capture program is running. The **REINIT** command tells the Capture program to obtain newly added replication sources from the register table.

REINIT also rereads the tuning parameters table for any changes made to the tuning parameters.

```
►►──ASNCMD──REINIT──────────────────────────────────────────────────►◄
```

or

```
►►──ASNCMD──src_server──REINIT──────────────────────────────────────►◄
```

Important: Do not use the **REINIT** command to reinitialize the Capture program after canceling a replication source or dropping a replication source table while the Capture program is running. Instead, stop the Capture program and restart it using the WARM or WARMNS option.

Pruning the change data and unit-of-work tables

Use the **PRUNE** command to start pruning the CD and UOW tables.

This command prunes tables once.

```
►►──ASNCMD──PRUNE───────────────────────────────────────────────────►◄
```

or

```
►►──ASNCMD──src_server──PRUNE───────────────────────────────────────►◄
```

The Capture program issues the message ASN0124I when the command is successfully queued.

During pruning, if you stop or suspend the Capture program, pruning does not resume after you enter the **RESUME** command. You must enter the **PRUNE** command again to resume pruning.

Displaying captured log progress

Use the **GETLSEQ** command to provide the timestamp and current log sequence number. You can use this number to determine how far the Capture program has read the DB2 log.

```
►►──ASNCMD──GETLSEQ─────────────────────────────────────────────────►◄
```

or

```
►►──ASNCMD──src_server──GETLSEQ──────────────────────────────────────►◄
```

Tip: The DB2 UDB Find Log Sequence Number command (**DB2LFSN**) enables you to identify the physical log file associated with the log sequence number. You can use this number to delete or archive log files no longer needed by the Capture program. For more information, see the *IBM DB2 Universal Database Command Reference*.

Operating Apply for Windows and OS/2

An administrator can use the commands in the following sections to perform the following Apply program tasks:

- Starting
- Scheduling
- Stopping

Before you start the Apply program

Before you start the Apply program, ensure that:

- You have the proper authorization. See "Authorization requirements for running the Apply program" on page 92 for information about authorization for the Apply program.
- The control tables are defined.
- At least one subscription is created and activated.
- The Apply package is created.
- **For Windows:** A password file has been created, for end-user authentication at the source server. See "Providing end-user authentication at the source server" on page 215 for more information.
- The Capture program is started, and the ASN0100I initialization message was issued (if you are running a Capture program).

Starting Apply for Windows and OS/2

After you start the Apply program, it runs continuously until:

- You stop it in an orderly way.
- You cancel it.
- An unexpected error or failure occurs.

To start the Apply program using the NT services:

1. Select **Replication** from the NT Services window.

2. Click the **Start** push button. The Apply program starts according to the ASCII file information you provided.

You can also start the replication service by typing STRTSERV on the Windows NT command line.

To start the Apply program using the DB2 command window:

Perform the following steps from a Windows or OS/2 window:

1. Log on with the IBM Replication user ID.
2. Ensure that you set the DB2 instance as shown:
   SET DB2INSTANCE=*db2_instance_name*
3. Enter the **ASNAPPLY** command from the Windows or OS/2 window where you issued the **SET** command:

```
►►──ASNAPPLY──Apl_qual──┬───────────┬──┬───────────┬──┬──────────┬──┬─────────┬──────►
                        └─Ctrl_serv─┘  ├─LOADXit───┤  ├─INAMsg───┤  ├─NOTRC───┤
                                       └─NOLOADXit─┘  └─NOINAMsg─┘  ├─TRCERR──┤
                                                                    └─TRCFLOW─┘

►──┬─────────┬──┬──────────┬──┬─────────┬──┬──────────┬──┬──────────┬──┬──────────┬──►
   └─TRCFILE─┘  ├─NOTIFY───┤  ├─SLEEP───┤  └─DELAY(n)─┘  └─COPYONCE─┘  └─LOGREUSE─┘
                └─NONOTIFY─┘  └─NOSLEEP─┘

►──┬───────────┬──┬──────────┬────────────────────────────────────────────────────►◄
   └─LOGSTDOUT─┘  └─TRLREUSE─┘
```

Table 30 defines the invocation parameters.

Table 30. ASNAPPLY Invocation Parameter Definitions on Windows and OS/2

Parameter	Definition
Apl_qual	Specifies the Apply qualifier that the Apply program instance uses to identify the subscriptions to be served. The Apply qualifier is case sensitive and must match the value of the APPLY_QUAL in the subscription set table. This must be the first parameter.
Ctrl_serv	Specifies the name of the server where the replication control tables will reside. If you do not specify this parameter, the default is the default database or the value of DB2DBDFT.
LOADXit	Specifies that the Apply program is to invoke ASNLOAD, an IBM-supplied exit routine that uses the export and load utilities to refresh target tables.

Table 30. ASNAPPLY Invocation Parameter Definitions on Windows and OS/2 (continued)

Parameter	Definition
NOLOADXit (default)	Specifies that the Apply program will not invoke ASNLOAD.
INAMsg (default)	Specifies that the Apply program is to issue a message when the Apply program is inactive.
NOINAMsg	Specifies that the Apply program will not issue this message.
NOTRC (default)	Specifies that the Apply program does not generate a trace.
TRCERR	Specifies that the Apply program generates a trace that contains only error information.
TRCFLOW	Specifies that the Apply program generates a trace that contains both error and execution flow information.
TRCFILE	If both trcfile and trace are specified, the Apply program writes trace output to the trace file (*.trc). If you do not specify this option, the Apply program sends trace output to the standard output, stdout.
NOTIFY	Specifies that the Apply program is to invoke ASNDONE, an exit routine that returns control to the user when the Apply program processing ends.
NONOTIFY (default)	Specifies that the Apply program will not invoke ASNDONE.
SLEEP (default)	Specifies that the Apply program is to go to sleep if no new subscriptions are eligible for processing.
NOSLEEP	Specifies that the Apply program is to stop if no new subscriptions are eligible for processing.
DELAY(*n*)	Where *n*=0, 1, 2, 3, 4, 5, or 6. Specifies the delay time (in seconds) at the end of each Apply program cycle when continuous replication is used. The default delay time is 6 seconds.
COPYONCE	The Apply program executes one copy cycle for each eligible subscription set and then terminates. An eligible subscription set is such that: • ACTIVATE > 0 • REFRESH_TIMING = R or B or REFRESH_TIMING = E and the specified event has occurred. MAX_SYNCH_MINUTES and END_OF_PERIOD are honored if specified.

Table 30. ASNAPPLY Invocation Parameter Definitions on Windows and OS/2 (continued)

Parameter	Definition
LOGREUSE	The Apply program reuses the log file (*.app) by first deleting it and then re-creating it when the Apply program is restarted. If you do not specify this option, the Apply program appends messages to the log file, even after the Apply program is restarted.
LOGSTDOUT	The Apply program sends all messages to both the standard output (stdout) and the log file.
TRLREUSE	The Apply program empties the Apply trail table when the Apply program is started.

Scheduling Apply for Windows and OS/2

For Windows: Use the Windows **AT** command to start the Apply program at a specific time. For example, the following command string starts the Apply program for Windows at 15:00.

```
c:\>AT 15:00 /interactive "c:\SQLLIB\BIN\db2cmd.exe
    c:\SQLLIB\BIN\asnapply.exe qualid1 cntldb"
```

Before you enter the **AT** command, the Windows Schedule Service should already be started.

For OS/2: Use the Alarms program in the OS/2 Productivity set to start the Apply program at a specific time.

Stopping Apply for Windows and OS/2

Use the **ASNASTOP** command or a key combination to stop the Apply program in an orderly way.

```
►►──ASNASTOP──Apply_qualifier──────────────────────────────────────────►◄
```

To use the command, do the following from a window where the Apply program is not running:

1. Set environment variable DB2INSTANCE to the value set when the Apply program was started.
2. Set environment variable DB2DBDFT to the source server specified when the Apply program was started (or the DB2DBDFT value used when the Apply program was started).
3. Enter the command.

Chapter 12. Capture for VM and Capture for VSE

This chapter describes how to set up and operate the Capture for VM and Capture for VSE programs.

Setting up the Capture program

Setting up consists of installing the Capture program and configuring the source servers.

See the *Capture for VM Program Directory* or the *Capture for VSE Program Directory* for Capture program installation instructions.

Operating Capture for VM and Capture for VSE

The administrator can use the commands in this section to operate Capture for VM and Capture for VSE.

This section explains how to perform the following Capture program tasks:

- Starting
- Stopping
- Suspending
- Resuming
- Reinitializing
- Pruning
- Displaying captured log progress

This section also lists restrictions for running the Capture program.

Restrictions for running the Capture program

Capture program restrictions are:

- Tables with field procedures for columns (FIELDPROC specified on CREATE or ALTER TABLE) are not supported by Capture for VM or Capture for VSE unless you create a new *one-way* FIELDPROC based on the existing FIELDPROC. The existing FIELDPROC doesn’t require any changes. If the corresponding column(s) on the CD table are defined with the new one-way FIELDPROC, and if the FIELDPROC does not change the length of the data, replication can be performed successfully.

- Because the Capture program identifies itself as an APPC/VM resource, you must specify appropriate IUCV VM/ESA System Directory control statements (such as IUCV *IDENT RESANY GLOBAL) for virtual machines that run the Capture program. For more information, see the *VM/ESA Planning and Administration Guide*.
- **For VM only:** There can be only one Capture program per database, and each Capture program runs in its own virtual machine. The Capture program identifies itself as an APPC/VM resource. By default, the resource ID value is CAPTURE. To use a different resource ID or to allow multiple Capture programs to run on the system for different DB2 databases, change the ENQ_NAME parameter in the ASNPARMS file.
- **For VM only:** The Capture program requires access to the appropriate level of the C Run Time Library. You must issue GLOBAL LOADLIB SCEERUN before starting the Capture program.
- **For VSE only:** Only one Capture program can be running per DB2 server for VSE database, and each Capture program runs in its own partition.

Starting Capture for VM and VSE

After you start the Capture program, it runs continuously until you stop it or it detects an unrecoverable error.

To start the Capture program for VM:

Invoke the ASNCCP module from a VM user ID. Keywords must be separated by one or more blanks:

```
►►──ASNCCP──┬─────────┬──┬─────────┬──┬────────┬──┬──────────┬──┬─────────┬──────────────►
            ├─WARM────┤  ├─PRUNE───┤  ├─TERM───┤  ├─NOTRACE──┤  ├─ALLCHG──┤
            ├─WARMNS──┤  └─NOPRUNE─┘  └─NOTERM─┘  └─TRACE────┘  └─CHGONLY─┘
            └─COLD────┘

►──┬────────────────────────────────┬────────────────────────────────────────────────►◄
   ├─NOUSERID───────────────────────┤
   └─USERID=──auth_name/password────┘
```

If conflicting invocation parameters are specified, the Capture program uses the value of the last parameter specified. For example, if ASNCCP is started using the COLD TRACE NOTRACE parameter string, no trace information is written (NOTRACE).

Table 31 on page 231 defines the invocation parameters.

To start the Capture program for VSE:

Sample job control member ASNS51BD provides an example of how to start the Capture program. Start the Capture program in a partition like a batch job. You can specify ASNCCP invocation parameters in the PARM= field, in the order shown, separated by one or more blanks:

```
►►──//──EXEC──PGM=ASNCCP──,PARM=──'──┬────────┬──┬─────────┬──┬────────┬──────────►
                                     ├─WARM───┤  ├─PRUNE───┤  ├─TERM───┤
                                     ├─WARMNS─┤  └─NOPRUNE─┘  └─NOTERM─┘
                                     └─COLD───┘

►──┬─────────┬──┬─────────┬──┤ Group 1 ├──'──────────────────────────────────────►◄
   ├─NOTRACE─┤  ├─ALLCHG──┤
   └─TRACE───┘  └─CHGONLY─┘
```

Group 1:

```
├──USERID=──auth_name/password──┬──────────────────────┬──────────────────────┤
                                └─Dbname=──databasename─┘
```

Table 31 defines the invocation parameters.

Table 31. ASNCCP Module Invocation Parameter Definitions for VM and VSE

Parameter	Definition
WARM (default)	The Capture program resumes processing where it ended in its previous run if warm start information is available. If the Capture program cannot warm start, it switches to a cold start.
WARMNS	The Capture program resumes processing where it ended in its previous run if warm start information is available. Otherwise, it issues a message and terminates. With WARMNS, the Capture program does not automatically switch to a cold start. The Capture program leaves the trace, UOW, CD, and warm start tables intact. In case of errors, the Capture program terminates instead of switching to a cold start as when WARM is specified.
COLD	The Capture program starts up by deleting all rows in its CD table, the UOW table, and the trace table during initialization. All subscriptions to these replication sources will be fully refreshed during the next Apply program processing cycle.
PRUNE (default)	The Capture program automatically prunes the rows in the CD and UOW tables that the Apply program copied, at the interval specified in the tuning parameters table.

Table 31. ASNCCP Module Invocation Parameter Definitions for VM and VSE (continued)

Parameter	Definition
NOPRUNE	Automatic pruning is disabled. The Capture program prunes the CD and UOW tables when you enter the **PRUNE** command.
TERM (default)	Terminates the Capture program if the DB2 server is terminated.
NOTERM	Keeps the Capture program running when the DB2 server is terminated. When the DB2 server initializes, the Capture program starts in WARM mode and begins capturing where it left off when DB2 terminated.
NOTRACE (default)	No trace information is written.
TRACE	Writes trace messages to the standard output, stdout.
ALLCHG (default)	Specifies that an entry is made to the CD table whenever a source table row changes.
CHGONLY	Specifies that an entry is made to the CD table when a source table row changes only if the columns defined for replication (CD table columns) change values.
USERID=*auth_name/password*	Specifies that the Capture program should connect to the database as user ID *auth_name*, with password *password*. The correct password must be provided or an error is returned. The *auth_name* and *password* are both from 1 to 8 characters in length. For VM/ESA, if you do not specify this parameter, the Capture program connects to the database as the user ID on which you issue ASNCCP.
Dbname=*databasename*	**For VSE only:** Identifies the name of the DB2 server for VSE database for which changes are to be captured. The name is from 1 to 18 characters in length. If not specified, the default is the database name as specified in the DBNAME directory or SQLDS if a DBNAME directory is not set up.

Stopping Capture for VM and VSE

Use the **STOP** command to stop the Capture program gracefully and commit the log records that it processed up to that point.

Issue the **STOP** command before:

- Removing an existing replication source
- Opening and modifying an existing replication source

- Shutting down the database

To stop the Capture program for VM:

```
►►──STOP──────────────────────────────────────────────────────────►◄
```

To stop the Capture program for VSE:

```
►►──MSG──partition──,DATA=STOP────────────────────────────────────►◄
```

where *partition* represents the partition that is running Capture for VSE.

If you stop the Capture program, it shuts itself down and issues an informational message. If it detects an error, the program shuts itself down after cleaning up the data in the affected tables so that the data will not be used. Staging tables are pruned when it is appropriate. In the case of abnormal termination, you must initiate a cold start because the warm start information could not be saved.

Suspending Capture for VM and VSE

Use the **SUSPEND** command to suspend the Capture program until you issue the **RESUME** command.

To suspend the Capture program for VM:

```
►►──SUSPEND───────────────────────────────────────────────────────►◄
```

To suspend the Capture program for VSE:

```
►►──MSG──partition──,DATA=SUSPEND─────────────────────────────────►◄
```

where *partition* represents the partition that is running Capture for VSE.

You can use this command to suspend the Capture program to improve performance for operational transactions during peak periods without destroying the Capture program run environment.

Important: Do not use the **SUSPEND** command when canceling a replication source. Instead, stop the Capture program.

Resuming Capture for VM and VSE

Use the **RESUME** command to resume the suspended Capture program.

To resume the Capture program for VM:

```
►►──RESUME─────────────────────────────────────────────────────────────►◄
```

To resume the Capture program for VSE:

```
►►──MSG──partition──,DATA=RESUME───────────────────────────────────────►◄
```

where *partition* represents the partition that is running Capture for VSE.

Reinitializing Capture for VM and VSE

Use the **REINIT** command to reinitialize the Capture program.

To reinitialize the Capture program for VM:

```
►►──REINIT─────────────────────────────────────────────────────────────►◄
```

To reinitialize the Capture program for VSE:

```
►►──MSG──partition──,DATA=REINIT───────────────────────────────────────►◄
```

where *partition* represents the partition that is running Capture for VSE.

Use the **REINIT** command to begin to capture changes from new source tables if you add a new replication source or ALTER ADD a column to a replication source and CD table while the Capture program is running. The **REINIT** command tells the Capture program to obtain newly added replication sources from the register table.

REINIT also rereads the tuning parameters table for any changes made to the tuning parameters.

Important: Do not use **REINIT** to reinitialize the Capture program after canceling or dropping a replication source table while the Capture program is running. Instead, stop the Capture program and start it again using the WARM or WARMNS option.

Pruning the change data and unit-of-work tables

Use the **PRUNE** command to start pruning the CD and UOW tables if you disabled pruning by using the **NOPRUNE** invocation parameter while starting the Capture program.

This command prunes tables once.

To prune the tables for VM:

```
►►──PRUNE──────────────────────────────────────────────────────────────►◄
```

To prune the tables for VSE:

```
►►──MSG──partition──,DATA=PRUNE────────────────────────────────────────►◄
```

where *partition* represents the partition that is running Capture for VSE.

During pruning, if you stop or suspend the Capture program, pruning does not resume after you enter the **RESUME** command. You must enter the **PRUNE** command again to resume pruning.

Displaying captured log progress

Use the **GETLSEQ** command to provide the timestamp and current log sequence number. You can use this number to determine how far the Capture program has read the DB2 log.

To display captured log progress for the Capture program for VM:

```
►►──GETLSEQ────────────────────────────────────────────────────────────►◄
```

To display captured log progress for the Capture program for VSE:

```
►►──MSG──partition──,DATA=GETLSEQ──────────────────────────────────────►◄
```

where *partition* represents the partition that is running Capture for VSE.

Part 4. Occasionally connected environments

This part of the book contains the following chapters:

Chapter 13. Satellite replication

This chapter describes how to set up, configure, and start DB2 data replication on satellites. A *satellite* is a DB2 server that synchronizes with a group at the satellite control server. A *group* is a collection of satellites that share characteristics such as database configuration and the application that runs on the satellite.

You can configure and maintain thousands of satellites from a central administration location, called the *satellite control server*. The satellite control server will automatically be updated when you use the Satellite Administration Center to administer satellites. Satellites can replicate data to and from all sources supported by the DB2 product family. For information about installing and administering satellites, see the *Administering Satellites Guide and Reference*.

Restrictions for satellites

Restrictions for satellites include:

- The enhanced level of conflict detection is not supported. If you specify the enhanced level, the Apply program assumes a standard level of conflict detection.
- The compiled ASNLOAD exit routine works only if your target definitions match your source definitions. Parameters such as number of columns, name of columns, type, and length must be exactly the same for both your target and source. If your target definitions do not match your source definitions, you need to customize and recompile the ASNLOAD sample source code or use the NOLOADX option. If your ASNLOAD exit routine requires a password, you must provide a password file.

Setting up replication for the satellite environment

You set up your satellite environment much like you set up replication for any other environment. The only difference with satellite is that you set it up in stages. Administrators need to do the following things to set up replication for the satellite environment:

1. Create the replication environment.
2. Set up the satellite environment.

This section discusses both of these steps.

Creating the replication environment

To create the replication environment:

1. Create your replication test environment:
   a. Create control tables.
   b. Define replication sources and subscription sets.

   Use the DB2 Control Center to create your control tables and define your replication sources and subscription sets. For more information on how to create control tables and define replication sources and subscription sets with the DB2 Control Center, see “Chapter 6. Setting up your replication environment” on page 83.
2. Promote replication sources and subscription sets from a test environment to a production environment.

   When you are sure that your replication test environment is running correctly, *promote* your replication sources and subscription sets. The promote function takes the parameters of defined replication sources and subscription sets from the replication test environment and re-creates the identical replication setup for the replication production environment. For more information about the promote function, see “Copying your replication configuration to another system” on page 110.

Setting up the satellite environment

After you create the replication environment, you must set up the satellite environment, and generalize the subscription sets for the satellite environment. For more information on setting up your satellite environment, see the *Administering Satellites Guide and Reference*.

To set up the satellite environment:

1. Define the group.

   To define a group, use the Satellite Administration Center. The *Satellite Administration Center* is a tool that provides centralized administrative support for satellites.
2. Create satellites for the group.

   You must create satellites to be members of the group that you just created. For more information about how to create satellites, see the *Administering Satellites Guide and Reference*.
3. Create an application version for the group.

   An application version is the set of setup, update, and cleanup batches that are associated with a specific version of an application that runs on the satellites of a group. If an application changes on the satellite that requires changes to the underlying database configuration, you must create a new application version to set up and maintain the new configuration. For more information about how to create an application version, see the *Administering Satellites Guide and Reference*.

4. Generalize the subscription set.

   You cannot perform data replication on the satellite environment until a subscription set is generalized. When you generalize a subscription set, parameters for the satellite environment are automatically defined in script files. These script files are based on a pre-existing replication configuration and are used to setup and invoke replication for satellites. For more information about generalizing a subscription set, see the *Administering Satellites Guide and Reference*.

   You can deploy the same replication configuration for thousands of satellites that execute the same application version. You can also deploy customized replication definitions for individual satellites.

5. Specify the type of database recovery that you want.

   Perform this step only if you did not specify the type of database recovery that you want on the satellite at installation time. If capturing data and database roll-forward recovery are required, set LOGRETAIN=RECOVERY. If capturing data is required without database roll-forward recovery, set LOGRETAIN=CAPTURE. When LOGRETAIN=CAPTURE, the Capture program will automatically call the **PRUNE LOGFILE** command when it completes its cycle. For more information about database recovery and considerations for using circular or archival logging, see the *DB2 Administration Guide*.

The Apply program obtains the user IDs and passwords required for replication as follows:

- If satellite users are using the satellite environment for administering their satellites, the administrator creates and maintains the user IDs and passwords required by replication using the Satellite Administration Center. When the satellites connect for synchronization, user IDs and their associated passwords are copied to the satellites. Passwords on the satellites are protected using encryption. For more information on creating and maintaining user IDs and passwords in the satellite environment, see the *Administering Satellites Guide and Reference*.
- If satellite users are not using the satellite environment for administering their satellites, the Apply program obtains the user IDs and passwords that it requires from the Apply program password file. For more information on creating these files, see “Providing end-user authentication at the source server” on page 215.

Starting replication on a satellite

Before starting replication: If one of the target table types is a replica table, the administrator must execute the **DPCNTL.SAT** command file in the satellite /sqllib/samples/repl directory before the satellite user can start replication on the satellite. The administrator can invoke the **DPCNTL.SAT** command from a script file in the setup batch. The script file with the **DPCNTL.SAT** command must follow immediately after the script files that configure replication. The **DPCNTL.SAT** command file creates source server control tables and sets default values in the tuning parameters table.

To start replication on a satellite: The satellite user clicks on the synchronization icon to synchronize his or her data. The satellite control server knows what each satellite needs. The first time that the satellite synchronizes, the satellite will download script files that contain the appropriate SQL statements and commands to set up replication on the satellite and to execute the Capture and Apply programs. When the satellite is configured appropriately, the satellite can replicate data between itself and its source server. When the administrator creates new scripts to update the configuration on the satellite, the satellite will download and execute them.

A new command, **ASNSAT**, is embedded within one of the script files that is used in the update batch when the subscription set is generalized. The **ASNSAT** command automatically starts the Capture and Apply programs with the optimal parameters for satellite. The **ASNSAT** command bypasses the Capture program if it is not necessary. The Capture and Apply programs will self-terminate after each program has completed a cycle.

The satellite users can invoke the **ASNSAT** command manually if they want to manipulate the Capture and Apply programs' parameters. To use the **ASNSAT** command, the satellite user types ASNSAT at the satellite command prompt. The syntax for the **ASNSAT** command is:

```
►►──ASNSAT──┬─────────────┬──┬───────────┬──┬───────────┬──────────────────────────►
            └─ -q─────────┤  └─ -n───────┤  └─ -t───────┤
             └apply_qual──┘   └cntl_srv──┘   └trgt_srv──┘

►──┬──────────────────────────────┬──┬──────────────────────────┬──────────────────►◄
   ├─ -c──────────────────────────┤  ├─ -a──────────────────────┤
   └Capture optional parameter────┘  └Apply optional parameter──┘
```

The Capture program writes messages to the .CCP file and trace table, and sends trace output to stdout. These .CCP files and the trace table grow indefinitely unless they are pruned manually. However, when the Capture program runs on a satellite, there is no administrator to prune these files. So to help manage the log and trace output of satellites without help from an administrator, a set of new invocation parameters is provided for the Capture program. For a list of the new Capture program invocation parameters, see Table 29 on page 220.

The -c option can override the following default Capture program options specified by the **ASNSAT** command: WARM, PRUNE, NOTRCTBL, LOGREUSE, LOGSTDOUT, NOTRCTBL, TRCFILE, and AUTOSTOP.

The Apply program provides new invocation parameters to manage the log and trace files without an administrator. For a list of the new Apply program invocation parameters, see Table 30 on page 226.

The -a option can override the following default Apply program options specified by the **ASNSAT** command: NOINAM, NOTRC, NONOTIFY, LOGREUSE, LOGSTDOUT, TRLREUSE, TRCFILE, COPYONCE, and LOADX.

Chapter 14. Mobile replication using DB2 DataPropagator for Microsoft Jet

This chapter describes DB2 DataPropagator for Microsoft Jet (DataPropagator for Microsoft Jet) and provides instructions for operating and monitoring the product.

What is DataPropagator for Microsoft Jet?

DataPropagator for Microsoft Jet extends the DB2 DataPropagator solution to support Microsoft Access and Microsoft Jet databases in LAN, occasionally connected, and mobile environments. Without any programming, you can replicate your server data into Microsoft Access tables for both browsing and updating.

DataPropagator for Microsoft Jet is a single executable that contains both the Capture and Apply capability and a portion of the administration facility. DataPropagator for Microsoft Jet runs on a client machine under Microsoft Windows NT or Windows 95, and reaches source databases via DB2 Client Application Enabler (CAE). DataPropagator for Microsoft Jet is packaged as part of DB2 DataJoiner Version 2 Release 2.1 (although you do not need to install a DB2 DataJoiner server to use this software) but also works with DB2 Universal Database (DB2 UDB), DB2 Common Server V2, and DB2 Connect. DataPropagator for Microsoft Jet requires the DataJoiner Replication Administration tool (DJRA) at the control point.

DataPropagator for Microsoft Jet replicates relational tables to and from Microsoft Jet databases, and detects and records any update conflicts (using the Microsoft Jet replication model). The *source server* can be DB2 or any non-DB2 replication sources defined through DB2 DataJoiner. The *control server* must be a DB2 or DB2 DataJoiner database. For more information about DB2 DataJoiner and the DataJoiner Replication Administration tool, see "Part 5. The DB2 DataJoiner Replication Administration tool" on page 269.

Figure 17 on page 246 illustrates how DataPropagator for Microsoft Jet supports replication of Microsoft Access and Microsoft Jet databases.

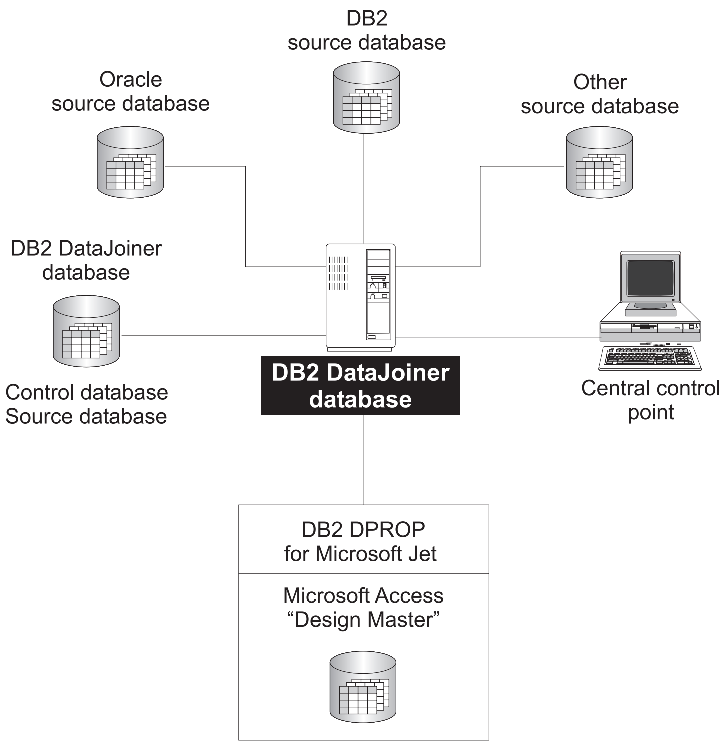


Figure 17. Microsoft Jet Database Replication. DataPropagator for Microsoft Jet extends IBM's data replication solution by supporting Microsoft Access and Microsoft Jet databases.

The advantages of mobile replication using DataPropagator for Microsoft Jet

A small DBMS with a replicated *subset* of a larger corporate database enables service employees and mobile professionals to run meaningful desktop applications while disconnected from a server network. These users connect to their corporate network only occasionally, and usually only long enough to synchronize their desktop database, e-mail, and messaging services with the corporate servers. For more information about subsets, see "Subsetting columns and rows" on page 68.

DataPropagator for Microsoft Jet administration doesn't require a direct connection to a Microsoft Jet database for administration. The DJRA tool maintains control information in the control server database. DataPropagator for Microsoft Jet running on a laptop is able to create Microsoft Jet databases, tables, and additional columns, and drop tables and old columns based on the current state of the control information in the server. To deploy a Microsoft Jet application, the application, database, and replication software must be installed before you distribute the laptop computers. However, the Microsoft Jet database does not need to be created in advance.

You can define or redefine replication source and subscription definitions for a Microsoft Jet database at any time, using DJRA, before or after you distribute the laptops for asynchronous processing by DataPropagator for Microsoft Jet.

If you have problems with your laptop, you can rebuild your Microsoft Jet database, tables, and contents by deleting the Jet database and resynchronizing using DataPropagator for Microsoft Jet. DataPropagator for Microsoft Jet can automatically rebuild your database.

For more information about usage scenarios involving mobile replication, see “Occasionally connected” on page 22.

Data integrity considerations

Within a network of DB2 databases, DB2 DataPropagator supports an *update-anywhere* model that is able to detect transaction conflicts. DataPropagator for Microsoft Jet supports an update-anywhere model, but with *weaker* row-conflict detection (similar to the standard Microsoft Jet model). If you choose to use DataPropagator for Microsoft Jet, you should be both familiar and comfortable with the standard Microsoft Jet replication model.

DataPropagator for Microsoft Jet reports synchronization conflicts in *conflict tables* in a very similar way to the built-in Microsoft Jet replication feature. This process can result in a loss of updates. If you use the single-user version of the DB2 Universal Database server on your laptop, for example, your application is assured of all-or-nothing transaction semantics when synchronizing with corporate servers. However, if you use Microsoft Jet as your mobile database, synchronization conflicts are handled on a row-by-row basis, so updates might be lost. Therefore, some updates might be flagged as conflicting while other updates propagate to the corporate database. If this situation is not acceptable, you need to program your own resolutions for all potential update conflicts. For more information about how DataPropagator for Microsoft Jet handles conflict errors, see “Error recovery” on page 253. For more information about programming your own resolutions, refer to the appropriate Microsoft documentation.

Terminology for DataPropagator for Microsoft Jet replication

The following terms represent replication concepts as they pertain to Microsoft Jet database replication. For definitions of general replication terms, see "Glossary" on page 425.

Client The Windows NT or Windows 95 machine on which DataPropagator for Microsoft Jet is installed.

Design Master
In Microsoft Jet database replication, the original database, which is saved as the master database. Each subsequent copy of the Microsoft Jet database maintained by Microsoft Jet replication on another server is called a Replica.

Row-replica
A type of update-anywhere replica maintained by DataPropagator for Microsoft Jet. Conflicts are detected row by row, not transaction by transaction, as they are for replicas. Row-replica is the only target table type supported by DataPropagator for Microsoft Jet. The source table type can be a DB2, Oracle, Sybase, Informix, or Microsoft SQL Server user table, or a DB2 replica. The source can also be a view of a DB2 user table or replica, including a join view with the restriction that all copied columns must come from only one of the tables referenced in the source view. The other columns in the source view can be referenced in the subscription predicates, but cannot be included in the row-replica.

Setting up DataPropagator for Microsoft Jet replication

To prepare the replication environment, you need to prepare the replication sources, control servers, and client environment. The following sections provide instructions for preparing your replication environment.

Preparing the replication source and control servers

You prepare the server to use DataPropagator for Microsoft Jet just as you would for DB2 DataPropagator. To prepare the server:

1. Create the necessary replication control tables at the control server by using DJRA.
2. Define replication sources by defining the source tables on each source server by using DJRA.
3. Define subscription sets by using DJRA. From the Create Empty Subscription Sets window, select the **Microsoft Jet** check box and enter the Microsoft Jet target server name. The target server name must be different from the control server name in this case.

4. Start the Capture program for each DB2 source server, if applicable.

Preparing the client environment

To prepare the client, install the following software (if it is not already installed):

1. Install DB2 Client Application Enabler (CAE) and configure DB2 connectivity to the source and control servers for the appropriate communication protocols.
2. Configure the DB2 ODBC driver by using the DB2 Client Configuration Assistant window.
3. Install either one of the following:
   - Microsoft Data Access Components (MDAC) (downloadable from http://www.microsoft.com/data/mdac15.htm)
   - Microsoft Access
4. Install the DAO component (downloadable from http://www.nesbitt.com/bctech.html or available on the Microsoft Visual C++ Version 5 CD-ROM).
5. Install DataPropagator for Microsoft Jet (during DB2 DataJoiner installation).
   - During installation you will be prompted to set the ASNJETPATH environment variable to specify the directory where DataPropagator for Microsoft Jet can create the log, trace, and password files. The file names are:
     – *Apply_qual*.LOG. Created by DataPropagator for Microsoft Jet.
     – *Apply_qual*.TRC. Created by DataPropagator for Microsoft Jet.
     – *Apply_qual*.PWD. Created by DataPropagator for Microsoft Jet.

     DataPropagator for Microsoft Jet also creates the target database in this directory if it does not already exist.
   - Define the Microsoft Jet database source in the ODBC Data Source Administration window, if it is not already defined.

Providing end-user authentication

If the source or control server requires authentication, create a password file.

To create a password file:

The password file must meet the following criteria:

- Be named `*APPLY_QUAL*.PWD`

  Where: *APPLY_QUAL* is the Apply qualifier, in uppercase. You specify a value for the Apply qualifier when you define a subscription set.

For example: `DATADIR.PWD`

Where: DATADIR is the Apply qualifier of the subscription set defined at the control server.

- Reside in the directory that is specified by ASNJETPATH.
- Contain all server-name / password pairs. This enables you to have a different (or the same) password at each server.
- Have one or more records that use the following format:

  ```
  SERVER=server_name PWD=password USER=userid
  ```

  The file cannot include blank lines or comment lines.

For more information about authentication and security, refer to the *DataJoiner Administration Supplement*.

Operating DataPropagator for Microsoft Jet

You can use the commands in this section to operate DataPropagator for Microsoft Jet.

This section explains how to perform the following tasks:

- Starting the Capture program at the source server
- Starting DataPropagator for Microsoft Jet at the client
- Stopping DataPropagator for Microsoft Jet at the client
- Troubleshooting DataPropagator for Microsoft Jet at the client

Starting the Capture program at the source server

Before you start DataPropagator for Microsoft Jet, you must start the Capture program on each DB2 source server, if applicable.

Starting DataPropagator for Microsoft Jet

Before you start DataPropagator for Microsoft Jet, you must establish any required line or LAN connection. DataPropagator for Microsoft Jet does not directly manage telephone connections, so you need to dial up the server manually or use any software that provides auto-dialing to establish a connection before you call DataPropagator for Microsoft Jet to perform database synchronization.

To start DataPropagator for Microsoft Jet, use the ASNJET command. Enter the ASNJET command from a command prompt.

```
►►──ASNJET──apply_qual──ctrl_srvr──┬──────────┬──┬─────────┬──┬──────────┬──────────────►
                                   ├─INAMSG───┤  ├─NOTRC───┤  ├─NOTIFY───┤
                                   └─NOINAMSG─┘  ├─TRCERR──┤  └─NONOTIFY─┘
                                                 └─TRCFLOW─┘

►──┬──────────┬──────────────────────────────────────────────────────────────────────►◄
   ├─MOBILE───┤
   └─NOMOBILE─┘
```

Table 32 defines the parameters.

Table 32. ASNJET Command Parameter Definitions for DataPropagator for Microsoft Jet

Parameter	Definition
apply_qual	Specifies the Apply qualifier that uniquely identifies this client.
ctrl_srvr	Specifies the control server alias.
INAMSG	Specifies that DataPropagator for Microsoft Jet issue an inactivity message to the log whenever DataPropagator for Microsoft Jet is going to sleep until the next copy cycle. This option is ignored if you specify the MOBILE option.
NOINAMSG (default)	Specifies that no inactivity message is issued.
NOTRC (default)	
TRCERR	Specifies that a trace file of minimal information is created.
TRCFLOW	Specifies that a trace file of extensive information is created.
NOTIFY	Specifies that DataPropagator for Microsoft Jet call the ASNJDONE exit routine at the completion of each subscription set, regardless of success or failure.
NONOTIFY (default)	Specifies that DataPropagator for Microsoft Jet does not call the ASNJDONE exit routine.
MOBILE	Specifies that DataPropagator for Microsoft Jet run in mobile mode (copy all active subscriptions only once, and then terminate).
NOMOBILE (default)	Specifies that DataPropagator for Microsoft Jet run continuously until it is stopped with the ASNJSTOP command.

Example 1: If you enter the following command from a command prompt, DataPropagator for Microsoft Jet is invoked with the Apply qualifier *MYQUAL*, the control server is *CNTLSRVR*, no inactivity message is generated, no trace is produced, the ASNJDONE exit routine is not called, and the active subscriptions are copied only once and then the program exits.

```
ASNJET MYQUAL CNTLSRVR MOBILE
```

Example 2: If you enter the following command from a command prompt, DataPropagator for Microsoft Jet is invoked with the Apply qualifier *AQ2*, the control server is *CNTLSRV*, an extensive trace is produced, and the program runs continuously until you stop it with the ASNJSTOP command.

```
ASNJET AQ2 CNTLSRV TRCFLOW NOMOBILE
```

Stopping DataPropagator for Microsoft Jet

When you start DataPropagator for Microsoft Jet using the MOBILE option, it runs until all active subscriptions are processed, and then terminates by itself. If you want to stop DataPropagator for Microsoft Jet, you can use the **ASNJSTOP** command to stop the program in an orderly way as soon as the current subscription set is copied and commit the log records processed up to that point.

Use the following command to stop DataPropagator for Microsoft Jet. Enter the ASNJSTOP command from a command prompt.

```
►►──ASNJSTOP──apply_qual──────────────────────────────────────────────►◄
```

Where *apply_qual* is the Apply qualifier that you used when you started DataPropagator for Microsoft Jet with the ASNJET command.

Example: If you enter the following command from a command prompt, DataPropagator for Microsoft Jet stops processing the Apply qualifier MQUAL as soon as the current subscription set is processed.

```
ASNJSTOP MYQUAL
```

You can also use one of the following key combinations from the window where the program is running to stop DataPropagator for Microsoft Jet:
- Ctrl+C
- Ctrl+Break

Troubleshooting DataPropagator for Microsoft Jet

If you encounter errors when you run ASNJET, ensure that:
- All replication sources and subscriptions are defined.
- The Capture program is started on the source server, if applicable.
- The control server and source server are defined as ODBC data sources.
- You supplied a password file in the ASNJETPATH directory.
- If you opened and updated a row-replica target table through Microsoft Access, you closed the table.

For error message information, see "Chapter 23. Capture and Apply messages" on page 371. For more information about troubleshooting, see "Troubleshooting" on page 126.

Returning control to users with the ASNJDONE exit routine

If you specify the NOTIFY parameter when you start DataPropagator for Microsoft Jet with the **ASNJET** command, DataPropagator for Microsoft Jet calls the exit routine ASNJDONE at the completion of each subscription set , regardless of success or failure. ASNJDONE.SMP is a sample program shipped with the product. You can modify it to meet the requirements of your installation. For example, the exit routine can examine the error table to discover rejected updates and initiate further actions, such as issuing a message or generating an alert.

See the prologue section in the sample exit routine ASNJDONE.SMP for instructions on how to modify the sample.

Parameters

The parameters that DataPropagator for Microsoft Jet passes to ASNJDONE are:

Control server
The control server alias.

Set name
The name of the set just processed.

Apply qualifier
The Apply qualifier of this DataPropagator for Microsoft Jet instance.

Trace option
The trace option specified when DataPropagator for Microsoft Jet was started.

Status value
Set to a value of 0 for success, and -1 for failure.

Error recovery

If the status value that DataPropagator for Microsoft Jet passes to ASNJDONE is -1, conflicts or errors might have been recorded. You can set the exit routine to examine the error codes and messages in the error message table. (There can be more than one row in the error message table.)

When DataPropagator for Microsoft Jet detects an update conflict between the RDBMS source and row-replica target table, it saves additional information for the ASNJDONE exit routine as follows:

- Inserts a row into the conflict table. (This is not the same conflict table that Microsoft Jet might detect between the Design Master and its Microsoft Jet Replicas.) The conflict table contains the row data that conflicted with the RDBMS update.
- Places the names of the conflict tables in the side information table. Each Microsoft Jet target table has its own conflict table. If a conflict is detected, the update to the row-replica loses while the source server update wins.

For other errors, such as referential integrity checks, DataPropagator for Microsoft Jet places additional information in the error information table, if applicable, to identify the row-replica table and the row that caused the error.

The exit routine can use this information to take remedial action. When the exit routine returns, the status is still -1 in the subscription set table. DataPropagator for Microsoft Jet does not expect any output or return codes from the exit routine.

DataPropagator for Microsoft Jet control tables

DataPropagator for Microsoft Jet requires the following new control tables, in addition to the existing DB2 DataPropagator control tables. For details about the column and index definitions for each of these new control tables, see "Chapter 21. Table structures" on page 299.

Control server tables

Row-replica-target-list table
Maintains the names of the row replica tables. This allows DataPropagator for Microsoft Jet to maintain a list of known row-replica tables in a stable DB2 or DB2 DataJoiner database. DataPropagator for Microsoft Jet uses this information during schema analysis to determine which, if any, row-replica tables should be deleted because the corresponding subscription member was dropped since the last synchronization.

Subscription-schema-changes table
Signals modifications to a subscription.

Target server tables

Conflict table
This table (one per target table, as needed at the target server) contains row data for DataPropagator for Microsoft Jet-detected

conflict losers. If there is a conflict between the same row in the Microsoft Jet database (target server) and the source server, the row in the Microsoft Jet database ″loses,″ so it is added to the conflict table and replaced by the row in the source.

Error information table
Contains additional information to identify the row-replica table and row that caused an error.

Error messages table
Contains error codes and error messages.

Error-side-information table
Contains the names of the conflict tables.

Key string table
Maps Microsoft Jet table identifiers and row identifiers to primary key values.

Synchronization generations table
Used to prevent cyclic updates from propagating back to the RDBMS from a Microsoft Jet database.

Chapter 15. Mobile replication for DB2

This chapter provides a brief overview of mobile replication, steps to plan and implement mobile replication, a description of mobile replication, and an explanation of how to run the mobile-replication-enabler program from the command line or from the mobile interface.

An overview of mobile replication

Mobile replication provides you with the flexibility that you need to efficiently manage your data on the go. Instead of trying to guess when you will be able to connect and synchronize your laptop computer with the home office, the mobile-replication-enabler allows you to transfer data on demand. The mobile-replication-enabler, also referred to as the ASNCOPY program, allows you to transfer data on demand because it controls the execution of the Capture and Apply programs.

You are not burdened with cumbersome manual procedures and high telecommunications costs. Instead the mobile interface, also referred to as the ASNMOBIL program, provides you with an interface to quickly and easily select and send data, as well as receive it. Typing `ASNMOBIL` on the command line will start the ASNMOBIL program. The mobile interface can then be used to invoke the ASNCOPY program. Mobile replication also minimizes the frequency and duration of communication line connections, thus reducing telecommunications costs.

The mobile-replication-enabler and its interface invocation program, ASNMOBIL, is supported on the following platforms:
- OS/2
- Windows NT
- Windows 95

The mobile-replication-enabler can replicate data to and from source servers on the following platforms:
- OS/2
- Windows NT
- AIX
- Sun Solaris
- HP
- SCO UnixWare 7

- MVS
- VM
- VSE

How mobile replication works

The mobile replication components are initiated with the mobile-replication-enabler, ASNCOPY. You can invoke the mobile-replication-enabler program from the Mobile Replication Enabler window, from the command line, or from an application program.

The mobile-replication-enabler replicates selected subscription sets as soon as possible and ignores relative timing or event timing options. It uses the same control tables that are used in nonmobile replication.

You can seamlessly make the switch between mobile and nonmobile modes, without needing to redefine subscriptions.

Mobile replication restrictions

Mobile replication has the following restrictions:

- The mobile-replication-enabler can support only one control server at a time. This control server is the default DB2 database located on your mobile client.
- The mobile-replication-enabler does not provide a dial-up or disconnect program. You can specify your own exit routines via the ASNDIAL and ASNHANGUP environment variables.
- Security for mobile replication must be provided by the underlying software products such as the DB2 Universal Database or the dial-up program for user authentication and DBMS access authorization.

Planning mobile replication

This section explains how to get your laptop up and running for mobile replication. This section discusses:

- Software and hardware requirements
- Communication program requirements
- Configuring the mobile client

Software and hardware requirements

In addition to the nonmobile IBM Replication hardware and software, mobile replication also requires communication hardware and software to transfer data. This section lists probable hardware and software needs. This list is not exhaustive, because it would be impossible to list all the hardware and software requirements for the unlimited number of possible configurations.

- Communication hardware
  – Modems
  – Phone lines
- Communication protocols
  – Netbios
  – TCP/IP
  – SNA
- Communication software
  – IBM LAN Distance

Communication program requirements

Your database administrator can develop a program to dial and disconnect your communications lines for you. Although mobile replication does not provide any connecting or disconnecting communication programs, it does provide a means to automate your user-defined connection/disconnection communication programs via two environment variables: ASNDIAL and ASNHANGUP.

ASNDIAL
Specifies the dial-up exit routine. When specified, the mobile replication component calls the dial-up program every time a physical connection is needed. Mobile replication does not pass any parameters and does not expect any return code from this exit routine.

ASNHANGUP
Specifies the disconnect exit routine. When specified, the mobile replication component calls the disconnect program as soon as the line is no longer needed. The only exception is when the disconnect program is disabled by the hold-line (-H) ASNCOPY invocation option. You can disable the automatic disconnect function for a variety of reasons. For example, you might want to issue DB2 commands against the source server after ASNCOPY is finished, or you might need a repeated copy to successfully copy a large answer set. Mobile replication does not pass any parameters and does not expect any return code from this exit routine.

Specifying ASNDIAL and ASNHANGUP environment variables in OS/2

The following section describes how to specify ASNDIAL and ASNHANGUP environment variables in OS/2.

To set the ASNDIAL and ASNHANGUP environment variables:

1. Declare the environment variables in your config.sys file. For example:
   - `SET ASNDIAL =` *C:\sqllib\bin\mydial.exe*
   - `SET ASNHANGUP =` *C:\sqllib\bin\myhangup.exe*

   Where:

   mydial.exe
   : The program that you use to connect your mobile client to the source server.

   myhangup.exe
   : The program that you use to disconnect your mobile client from the source server.
2. Reboot your system to have these settings take effect.

Specifying ASNDIAL and ASNHANGUP environment variables in Windows NT

The following section describes how to specify ASNDIAL and ASNHANGUP environment variables in Windows NT.

To set the ASNDIAL and ASNHANGUP environment variables:

1. From the **Control Panel** window, double-click the **System** icon. The System Properties notebook opens.
2. Select the **Environment** tab.
3. Set the ASNDIAL variable:
   a. In the **Value** field, type the path for your user-defined connect program. For example:
      ```
      C:\sqllib\bin\mydial.exe
      ```
   b. Click **Set**.
4. Set the ASNHANGUP variable:
   a. In the **Value** field, type the path for your user-defined disconnect program. For example:
      ```
      C:\sqllib\bin\myhangup.exe
      ```
   b. Click **Set**.
5. Click **OK**. The environment variables are set.

Specifying ASNDIAL and ASNHANGUP environment variables in Windows 95

The following section describes how to specify ASNDIAL and ASNHANGUP environment variables in Windows 95.

To set the ASNDIAL and ASNHANGUP environment variables:

1. Declare the environment variables in your autoexec.bat file. For example:

- SET ASNDIAL = *C:\sqllib\bin\mydial.exe*
- SET ASNHANGUP = *C:\sqllib\bin\myhangup.exe*

Where:

mydial.exe
: The program that you use to connect your mobile client to the source server.

myhangup.exe
: The program that you use to disconnect your mobile client from the source server.

2. Reboot your system to have these settings take effect.

Setting up the mobile client

You define replication sources and subscription sets in the same way that you define replication source and subscription sets in the nonmobile replication process. The unique situation created in the mobile environment is that the mobile client might not always be connected to the Control Center. When the control server is disconnected from the Control Center, you must find a way to get replication source and subscription set information to the mobile client.

There are two possible ways to distribute the replication source and subscription set information needed to run a mobile replication scenario. Each of these methods has restrictions. The examples below are for replication from a DB2 source. If you are replicating from a non-IBM source, use the DataJoiner Administration Tool instead of the DB2 Control Center.

- Install the DB2 Control Center on the laptop. While the mobile client is connected to the source server, define replication sources and subscription sets.
- Define replication sources and subscription sets using the Control Center on another server, save and customize the SQL files, download the files via diskette or FTP, and run the SQL files on the mobile client.
  1. Define replication sources and subscription sets at the Control Center by connecting to the source server.
  2. Save and edit the generated SQL files. Customize the values for the CONTROL_SERVER, CNTL_ALIAS, SOURCE_SERVER, and TARGET_SERVER columns of ASN.IBMSNAP_SUBS_SET table to match those on the mobile client.
  3. Transfer the customized SQL files and DPCNTL.UDB file to the mobile client via a diskette or by downloading the information.
  4. Run the DPCNTL.UDB files on the mobile client.
  5. Run the customized SQL files on the mobile client.

Configuring the mobile client for Windows NT and Windows 95

Configuring the mobile client is similar to configuring the target server in a nonmobile replication scenario. The only difference is that you need to bind the mobile client to the control server, in addition to binding the Capture and Apply programs to a target server.

See “Setting up the Capture and Apply programs” on page 213 for more information on how to configure the Capture and Apply programs at the target servers.

To bind the mobile client to the control server database:

1. Be sure that the user ID that you are using to run the Apply program has the required privileges:
   - EXECUTE privilege for the Apply packages
   - DBADM or SYSADM authority for the databases
2. Log on with the user ID that has sufficient privileges.
3. Go to the Apply program bind files directory, usually in *drive*:\SQLLIB\BND.
4. Connect to the control server database by entering:
   ```
   DB2 CONNECT TO control_server
   ```
5. Create and bind the Apply package to the control server database by entering both of the following commands:
   ```
   DB2 BIND ASNNY000.BND ISOLATION CS BLOCKING ALL
   DB2 BIND ASNNL000.BND ISOLATION UR BLOCKING ALL
   ```

   Where:

   CS The list in cursor stability format.

   UR
   The list in uncommitted read format.

Configuring the mobile client for OS/2

Configuring the mobile client is similar to configuring the target server in a nonmobile replication scenario. The only difference is that you need to bind the mobile client to a control server, in addition to binding the Capture and Apply programs to a target server.

To bind the mobile client to the control server database:

1. Be sure that the user ID that you are using to run an Apply program has the required privileges:

- EXECUTE privilege for the Apply packages
- DBADM or SYSADM authority for the databases

2. Log on with the user ID that has sufficient privileges.
3. Go to the Apply program bind files directory, usually in *drive*:\SQLLIB\BND.
4. Connect to the control server database by entering:
   ```
   DB2 CONNECT TO control_server
   ```
5. Create and bind the Apply package to the control server database by entering both of the following commands:
   ```
   DB2 BIND ASN2Y000.BND ISOLATION CS BLOCKING ALL
   DB2 BIND ASN2L000.BND ISOLATION UR BLOCKING ALL
   ```

   Where:

   CS The list in cursor stability format.

   UR
   The list in uncommitted read format.

Defining the control server for your mobile client

The mobile-replication-enabler uses the default database on the mobile client as its control server. You must specify the default database and instance name by setting the environment variables as follows:

- SET DB2DBDFT = *local target_server_name*
- SET DB2INSTANCE = *database_instance_name*

If your database environment does not match the default database environment, make the necessary changes so that the default database and instance is set properly.

Restriction: Do not use ″mobile″ as a database name. The Capture program uses the word ″mobile″ internally when running in the mobile mode; therefore, naming a database mobile will cause an error.

Mobile replication processing cycle

When the mobile-replication-enabler is invoked, the Capture and Apply programs replicate data as defined in the subscription set in the following manner:

1. ASNCOPY calls the Capture program. The Capture program stops when it captures all of the changes up to the time when it was called.

2. ASNCOPY sets the value of the ACTIVATE column in the ASN.IBMSNAP_SUBS_SET table to 2, which requests an immediate copy.
3. ASNCOPY calls the dial-up exit routine, if specified in the ASNDIAL environment variable.
4. ASNCOPY calls the Apply program.
5. The Apply program replicates all of the data sets that are marked for immediate replication. Replication is repeated if the sizes of the answer sets are regulated by MAX_SYNCH_MINUTES.
6. The Apply program calls the disconnect exit routine, if specified in the ASNHANGUP environment variable, at the earliest opportunity to minimized the phone cost.
7. ASNCOPY displays replication statistics from the information selected from the ASN.IBMSNAP_APPLYTRAIL table.
8. ASNCOPY stops.

Starting the mobile-replication-enabler using the ASNCOPY command

ASNCOPY, the mobile-replication-enabler, is the program that controls the execution of the Capture and Apply programs. You can specify parameters to run the mobile-replication-enabler when you invoke ASNCOPY from the command line. You can also specify mobile-replication-enabler parameters from the mobile interface. To execute the mobile interface, type ASNMOBIL from the command line. For more information about the mobile interface, see “Starting the mobile-replication-enabler using the mobile interface” on page 266.

Table 33 lists the parameters that you can specify for the ASNCOPY command. These parameters are not case sensitive.

Table 33. ASNCOPY Command Parameters

Parameter	Explanation
-T	Activates the trace option. You can set the trace option on from the mobile interface. When the trace is set on, information about all database activity will be written to a file called ASNCOPY.TRC. The Apply program trace is written in a file called *apply_qual*.TRC, where *apply_qual* is the Apply qualifier value for the OS/2 platform, and in a file called *APPLYYYYYMMDDHHMMSS*.TRC for the Windows NT and 95 platforms. The Capture program trace is written in the trace table called ASN.IBMSNAP_TRACE.

Table 33. ASNCOPY Command Parameters (continued)

Parameter	Explanation
-H	Holds the communication line connection even after all answer sets are fetched. When this parameter is not specified, the Apply program disconnects the link between the target and source servers by calling the disconnect exit routine as soon as all of the answer sets are fetched. This is done to minimize the line connection time. If you do not specify this parameter, you might not be able to repeat a copy, which might be needed if the target-server log file overflows. This problem is not detected until after the link is disconnected. In this case, retry the failed copy with this parameter specified. If overflow is detected before the link is disconnected, repeat copies are attempted regardless of whether this parameter is specified.
-L	Specifies that the Apply program call the ASNLOAD exit routine for a full refresh. In the ASNLOAD exit routine, you can invoke either the IBM-provided EXPORT/IMPORT or EXPORT/LOAD utilities, or any other programs for a faster full refresh. If referential constraints are defined between member tables of a subscription set, you must use this option to bypass RI constraint checking while the tables go through a full refresh.
-F	Specifies the fast path. This parameter tells ASNCOPY not to call the Capture program. If this parameter is not specified, the Capture program is always called regardless of the copy direction. Use this option only if either no changes have been made to the local tables since the last Capture program run or the subscription set to which all the changed tables belong is not to be copied. Otherwise, the changes will be lost.
-C	Specifies that ASNCOPY starts the Capture program with a cold start. If a cold start occurs, then replica target tables are refreshed from the source tables and all changes made locally that were not applied to the source server are lost. If this parameter is not specified, ASNCOPY uses the WARMNS start option. The -C parameter is ignored if the fast path (-F) parameter is specified or if the Capture program is not installed.
-R	Specifies the remote control server. If this parameter is not specified, the ASNCOPY program will use the local control server, named in the DB2DBDFT system variable, as the default control server.
-N	Calls the ASNDONE exit routine to notify completion of replication in the 'F' direction as well as the 'S' direction for each subscription set.
-Q *apply_qualifier*	You must specify this parameter when the Apply qualifier is not the same as the user ID.
-S *set_name_list*	Specifies a list of subscription set names that you want to copy.
-U *set_name_list*	Specifies the copy up (WHOS_ON_FIRST = F) direction of the replica sets. That is, copying up from the target server to the source server.
-D *set_name_list*	Specifies the copy down (WHOS_ON_FIRST = S) direction of the replica sets. That is, copying down from the source server to the target server.

Starting the mobile-replication-enabler using the mobile interface

The Mobile Replication Enabler window can be used to start the mobile-replication-enabler, ASNCOPY. To run the mobile interface, type `ASNMOBIL` from the command line. The Mobile Replication Enabler window opens, see Figure 18.

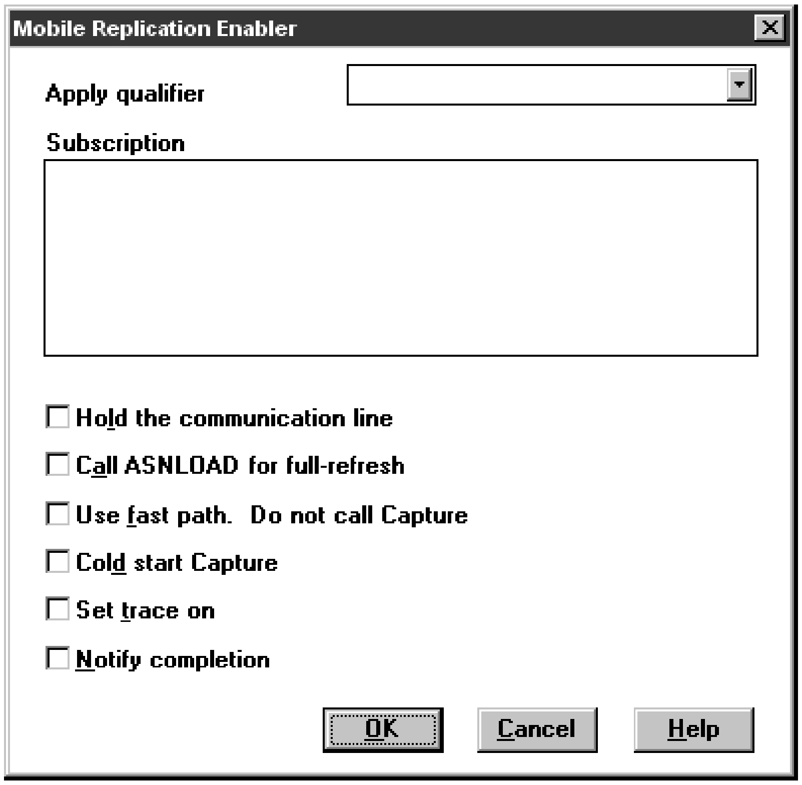

Mobile Replication Enabler
Apply qualifier
Subscription
Hold the communication line
Call ASNLOAD for full-refresh
Use fast path. Do not call Capture
Cold start Capture
Set trace on
Notify completion
OK
Cancel
Help

Figure 18. The Mobile Replication Enabler Window. You can set ASNCOPY parameters.

From the Mobile Replication Enabler window, you can:

- Hold the communication line
- Call ASNLOAD for full refresh copying
- Use the fast path
- Specify the cold start option for the Capture program
- Set the trace on
- Specify to be notified when replication is complete

The Mobile Replication Enabler window also allows you to select subscription sets and an Apply qualifier.

Selecting subscription sets

The mobile interface presents a list of active subscription sets from which you can select the subscription sets that you want to run. A subscription set contains the specification of the source and target tables, as well as the control information that governs a refresh or update.

The mobile-replication-enabler, ASNCOPY, will run with the subscription set names that you selected. A complete list of subscription sets can be found in the ASN.IBMSNAP_SUBS_SET table on the control server.

Selecting an Apply qualifier

An Apply qualifier is a unique name in the network to distinguish one instance of the Apply program from another. To populate a subscription set list with a particular subscription set, select the Apply qualifier that represents the Apply instance running on your client. ASNCOPY will now only fetch data according to the subscription set associated with the Apply qualifier selected.

Part 5. The DB2 DataJoiner Replication Administration tool

This part of the book contains the following chapters:

Chapter 16. DJRA overview

DJRA is one of two replication administration interfaces that automate many replication activities. From DJRA, you can select your source tables and prepare them for change capture, create your target tables, and create the control information for the Apply program. The control information causes the Apply program to copy data from the selected source tables to the selected target tables. You can also use DJRA to browse or change replication definitions.

DJRA, working with DB2 DataJoiner, the Capture program, Capture triggers, and the Apply program, replicates relational data from a variety of sources to a variety of targets. The databases that DJRA supports as sources or targets are:

- DB2 UDB V5 or later
- DB2 for MVS V3R1 or V4R1, and DB2 for OS/390 V5R1 or V6R1
- DB2 for common servers V2 and DB2 DataJoiner V2
- Oracle V7.0.13
- Informix V7.1 or later
- Sybase V4R6 or later (AIX) and Sybase V11 (Windows NT)
- Sybase SQL Anywhere Version 5.0 (Windows NT)
- Microsoft SQL Server V4.21 or later (AIX) and Microsoft SQL Server V6.0 or later (Windows NT)
- DataPropagator for Microsoft Jet (as target only)

You can use DB2 DataJoiner with IBM Replication to access data from non-IBM databases and replicate data from any database in your enterprise to any other, as shown in Figure 19 on page 272. You can also tailor or enhance data as it is copied, thus delivering detailed, divided, summarized, or derived data when and where it is needed.

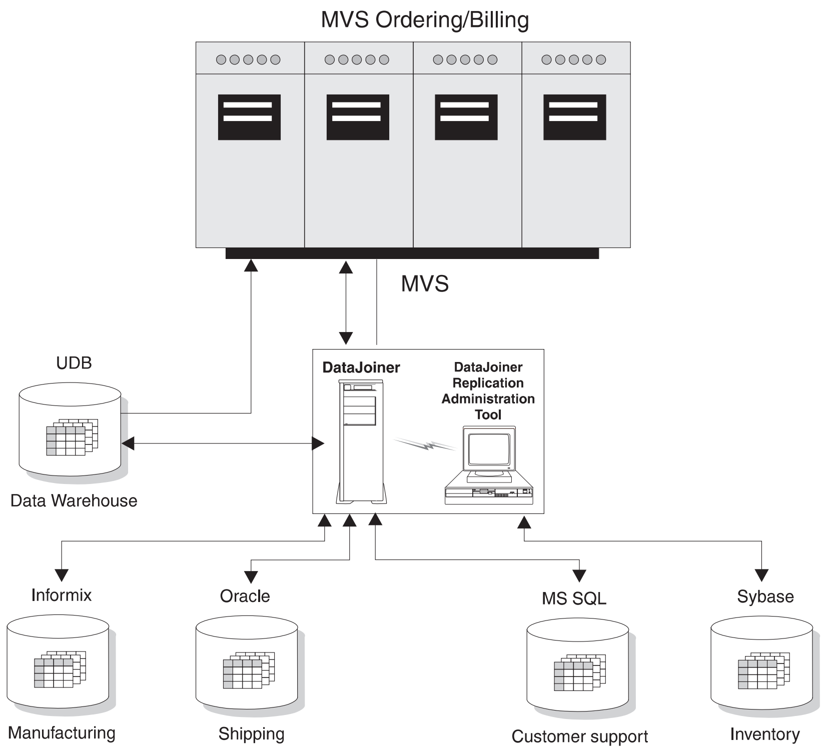


Figure 19. DB2 DataJoiner with IBM Replication Scenario. DB2 DataJoiner with IBM Replication enables you to copy data across your enterprise.

DJRA provides objects and actions that define and manage source and target table definitions. Working through DB2 DataJoiner, DJRA creates:

- Capture triggers on the non-IBM source servers
- Nicknames in the DB2 DataJoiner database for the remote tables where the changed data is to be captured
- Target tables (and their associated nicknames) in the non-IBM database for the remote target tables

The Apply program then reads from and writes to DB2 DataJoiner nicknames, eliminating the need to connect explicitly to non-IBM databases.

If the source database is a DB2 database, the Capture program for that database captures the changes, therefore, the Capture triggers and DB2 DataJoiner are not involved. If the target database is a DB2 database, the Apply program writes the changed data to the DB2 target database directly and DB2 DataJoiner is not involved.

DJRA and DB2 DataJoiner

DJRA accesses non-IBM databases through DB2 DataJoiner. DB2 DataJoiner presents a single view for various DB2 family and non-IBM databases, masking differences in data types, SQL dialects, and communications. You create nicknames for the non-IBM source or target tables in a DB2 DataJoiner database and then use DJRA to create a source definition for the nicknamed object. Within a single DB2 DataJoiner local database, you can define the nicknames for one or more source tables that reside on *one* remote, non-IBM source server.

After you create subscription sets for a non-IBM source server, the Apply program connects to the DB2 DataJoiner database that is associated with the non-IBM server and accesses (through nicknames) the information in the register control table and the staging table on the non-IBM source server (see Figure 20).

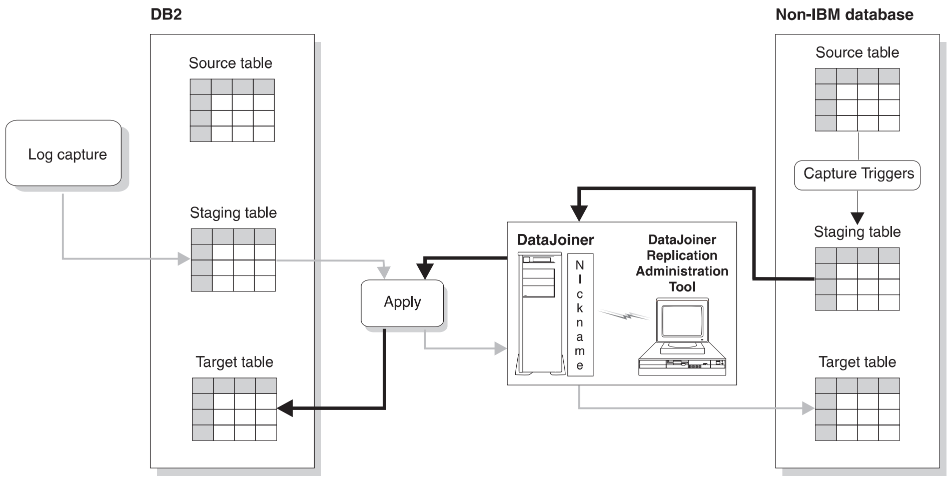


Figure 20. DB2 DataJoiner in Action. In a scenario where the source table is a non-IBM table (the dark arrows), DB2 DataJoiner nicknames give IBM Replication and the Apply program access to the non-IBM source server and to changes made to the non-IBM source table (through the staging table). In a scenario where the source table is a DB2 table (the light arrows), DB2 DataJoiner nicknames give the Apply program access to the non-IBM target tables.

Capture triggers for non-IBM sources

Capture triggers are used for replication from non-IBM databases. They capture changed data from a source table and make the changed data available for replication. Capture triggers perform the same task as the Capture program does for DB2, but in a different manner. DJRA generates the Capture triggers.

DJRA, working through DB2 DataJoiner, creates Capture triggers at the non-IBM source database when you define the source tables as replication sources. Capture triggers capture committed changes made to source data and place the captured changes into a staging table, called the *consistent change data* (CCD) table. The CCD table has a nickname in DB2 DataJoiner that programs that want to replicate the changes (for example, the Apply program) can access. See “Staging data” on page 75 for more information about CCD tables.

There are three triggers for each source table: DELETE, UPDATE, and INSERT.

How the Capture triggers capture the data changes

The Capture triggers work with the following objects: the CCD table, the register control table, the pruning control table, and the register synchronization control table.

DJRA generates SQL (when you define a table as a replication source) that, when run:

- Creates Capture triggers on the source table.
- Creates the CCD table on the source server. There is one CCD table for each source table.
- Inserts a row into the register control table (to represent the new source table).
- Creates a nickname for the CCD table in the DB2 DataJoiner database.

Whenever a delete, update, or insert operation occurs at the defined source, a Capture trigger records the change into the CCD table. When the Capture triggers retrieve changed information, they can also obtain before and after column data to put into the CCD table.

The Apply program then reads the CCD table (through DB2 DataJoiner nicknames), copies the changes to the target server, and applies the changes to the target table. Figure 21 on page 275 shows the relationship between the Capture triggers, the source table, the register control table, and the CCD table.

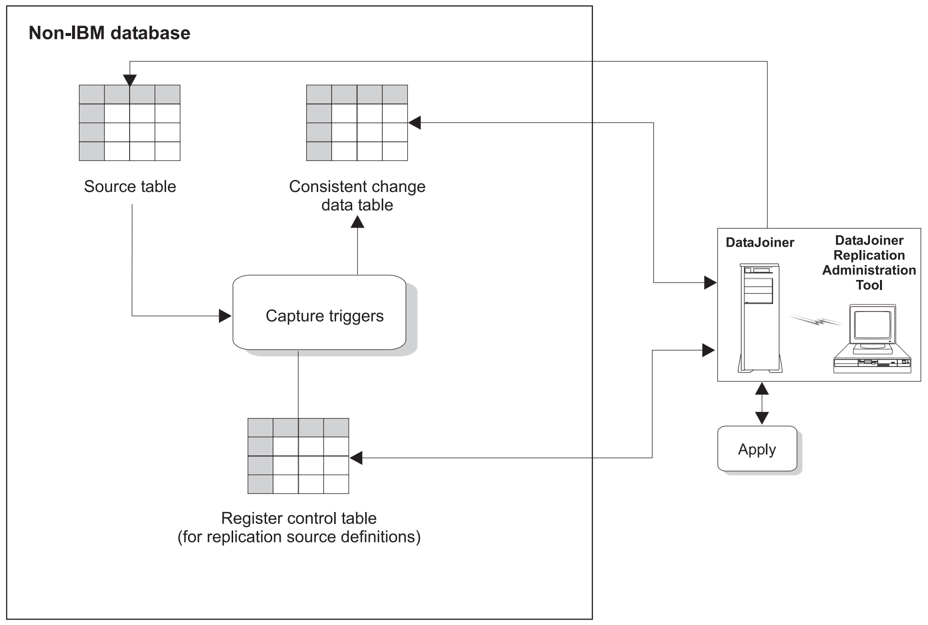

Non-IBM database
Source table
Consistent change data table
Capture triggers
Register control table (for replication source definitions)
DataJoiner
DataJoiner Replication Administration Tool
Apply

Figure 21. Capture Triggers at the Source Server. The Capture triggers monitor source changes, capture the changed data, and write the changed data to the CCD table.

Capture Triggers and pre-existing triggers

When DJRA creates and places Capture triggers on a non-IBM database, you might encounter the following situations:

- **On Oracle**: If an attempt is made to create a Capture trigger on a table where there is a pre-existing trigger with the same name, or where the pre-existing trigger performs an identical event (insert-before, insert-after, delete-before, delete-after, update-before, update-after), Oracle issues the following message: ORA-04081 (trigger name already exists). If this error is generated, the Capture trigger is not created.
- **On Informix**: If an attempt is made to create a Capture trigger on a table where there is a pre-existing trigger with the same name, or where the pre-existing trigger performs an identical event (insert, delete, update), Informix issues an *-741* error and will not create the Capture trigger.
- **On Microsoft SQL Server or Sybase**: If an attempt is made to create a Capture trigger on a table where there is a pre-existing trigger with the same name, or where the pre-existing trigger performs an identical event (insert, delete, update), Microsoft SQL Server and Sybase do not generate error or warning messages indicating a conflict. Microsoft SQL Server and Sybase replace the pre-existing trigger with the new Capture trigger.

However, DJRA does check to see if a trigger already exists. If a trigger with the same events exists, DJRA creates the new triggers but all lines within the trigger body are commented out. You must determine how you want to merge pre-existing triggers with the new triggers. Then you can uncomment the lines in the new triggers.

If you anticipate conflict between DJRA's Capture triggers and pre-existing triggers, put the content of both triggers into one trigger. For each table event, append the pre-existing business trigger to the end of the Capture trigger script that is generated by DJRA.

Improving Apply performance for Sybase or Microsoft SQL Server on AIX

If you are running the Apply program on AIX with a DBLIB connection for either Sybase or Microsoft SQL Server, you can significantly improve your overall performance. To take advantage of this improvement:

1. Retrieve the names of the packages for the Apply program. To find your package names, issue the following SQL statement:

   ```
   SELECT PKGNAME
   FROM SYSCAT.PACKAGES
   WHERE PKGNAME LIKE 'ASN%'
   ```

   The package names change with each release and with each service update, but this query retrieves names that are specific to your service level.
2. If you have an apply_name.ini file (in the sqllib directory), replace the package names with the ones that you retrieved in step 1. If you do not have an apply_name.ini file, create one and list the package names. The following lines show an example of an apply_name.ini file:

   ```
   ASN6A001+
   ASN6B001+
   ASN6C001+
   ASN6F001+
   ASN6I001+
   ASN6M001+
   ASN6P001
   ```
3. Create server options for the Apply packet and buffer sizes. Sample sever options for Sybase are:

   ```
   create server option apply_packet_size for server type sybase setting 16384;
   create server option apply_buffer_size for server type sybase setting 16384;
   ```

   Sample sever options for Microsoft SQL Server are:

   ```
   create server option apply_packet_size for server type mssqlserver setting 16384;
   create server option apply_buffer_size for server type mssqlserver setting 16384;
   ```

You can set the packet and buffer size to any appropriate value; set them to 16384 initially, and adjust as necessary.

4. Set the following environment variable:

   ```
   DJX_ASYNC_APPLY=TRUE
   ```

5. If you created or changed the apply_name.ini file, or if you changed the DJX_ASYNC_APPLY variable, you must stop and restart DataJoiner before these changes take effect. To stop and restart DataJoiner, issue the **db2stop** and **db2start** commands.

Chapter 17. Setting up the DB2 DataJoiner environment for replication

This chapter explains how to begin to set up your replication environment while installing and configuring DataJoiner. In this chapter, you:

1. Install DataJoiner.
2. Create a DataJoiner instance and, within the instance, a database for each non-IBM database that is a source for replication. Your target databases can use an existing database within the same DataJoiner instance, or you can create a new database for your targets.
3. Configure access to your data sources (your replication source, target, and control servers).
4. Connect clients to DataJoiner. If the Apply program[17] that you are using for replication is remote from DataJoiner, you must connect the Apply program as a client to DataJoiner.

After this chapter, proceed to "Chapter 18. Installing DJRA and connecting all databases" on page 283 to install DJRA and connect all databases that are involved in replication.

Seting up DataJoiner for AIX

This section describes how to prepare DataJoiner for AIX for replication.

Installing DataJoiner

Restore product files using installp or smitty by following the steps described in the *DB2 DataJoiner Planning, Installation, and Configuration Guide*. Installation of some components is optional. Later, as described in "Chapter 18. Installing DJRA and connecting all databases" on page 283, you will install DJRA on your NT workstation. Make sure that the workstation with DJRA can connect to the workstation with DB2 DataJoiner (if they are not the same).

Setting up an instance

As described in the *DB2 DataJoiner Planning, Installation, and Configuration Guide*, you:

17. The Apply program shipped with DB2 DataJoiner is the same as the Apply program shipped with DB2 Universal Database.

- Create an instance
- Set up environment variables
- Create DataJoiner databases
- Set up Apply for AIX as a local client

This section focuses on setting up the Apply program and creating DataJoiner databases as pertaining to replication.

Setting up Apply for AIX as a local client

Identify the Apply user ID as a local client on DataJoiner. See the *DB2 DataJoiner Planning, Installation, and Configuration Guide* for more information.

In addition to using Apply for AIX (provided with DataJoiner), you can also use other Apply programs on other platforms as discussed in "Connecting clients to DataJoiner" on page 281.

Creating DataJoiner databases

As described in the *DB2 DataJoiner Planning, Installation, and Configuration Guide*, you create databases that will be configured to access non-IBM databases as part of your replication system. You must create one DataJoiner database for *each* non-IBM replication *source* server. You can support many non-IBM replication *target* servers with one DataJoiner database. The DataJoiner databases that you set up reside on one DataJoiner instance.

For every non-IBM source, use the COLLATE USING parameter within the CREATE DATABASE command. Use IDENTITY.

Configuring database connections

The Apply program in DataJoiner uses the connections that you configure in this section to access source, target, and control servers.

Table 34. Configuring Database Connections in DataJoiner: AIX

If accessing	You need to
DB2 for OS/390	Catalog DB2 in DataJoiner
DB2 UDB V6 or DB2 CS V2	Catalog DB2 in DataJoiner
Informix, MS SQL Server, Sybase, Sybase SQL Anywhere, or Oracle	Create a server mapping and user mapping within DataJoiner

You must define server and user mappings for each DataJoiner database that requires access to a source or target. You must have one DataJoiner database for *each* non-IBM replication *source* server. You can support many *target* servers with one DataJoiner database. The DataJoiner databases that you set up reside on one DataJoiner instance.

Connecting clients to DataJoiner

For Apply for OS/390, refer to the *DB2 DataJoiner Planning, Installation, and Configuration Guide* for information on how to enable applications on MVS, such as Apply for OS/390, to access DataJoiner.

For Apply programs on DB2 UDB or DB2 CS, refer to the chapter applicable to your network’s communication protocol: TCP/IP, IPX/SPX, or APPC.

The steps that are described in the “Connecting Clients to DB2 DataJoiner” section in the *DB2 DataJoiner Planning, Installation, and Configuration Guide* must be performed for each DataJoiner database that is created to replicate with a remote Apply program. Later, you will connect DJRA as a client to DataJoiner.

What to do next

Before proceeding to “Chapter 18. Installing DJRA and connecting all databases” on page 283, make sure that you install all Capture and Apply programs on the systems of your choice.

Setting up DataJoiner for Windows NT

This section describes how to prepare DataJoiner for Windows NT for replication.

Installing DataJoiner

In the *DB2 DataJoiner Planning, Installation, and Configuration Guide*, you have the option of installing DJRA after you install DataJoiner. You do not have to select the Apply program. Apply for Windows NT is automatically installed when you install DataJoiner.

Setting up an instance

The DataJoiner databases that you set up for replication reside on one DataJoiner instance. You must create one DataJoiner database for *each* non-IBM replication *source* server. You can support many non-IBM replication *target* servers with one DataJoiner database. The DataJoiner databases that you set up reside on one DataJoiner instance. Use the DB2 **CREATE DATABASE** command to create each database. See the *DB2 DataJoiner Planning, Installation, and Configuration Guide* for more information.

You can configure Windows NT to automatically start your DataJoiner instance and DB2 security service:

1. Select **My Computer** from the desktop.
2. Select **Services**.
3. Select the DB2 instance to be used for replication and the DB2 security server.
4. Select **Startup**.

Configuring database connections

The Apply program in DataJoiner uses the connections that you configure in this section to access source, target, and control servers.

Table 35. Configuring Database Connections in DataJoiner: NT

If accessing	You need to
DB2 UDB V6 or DB2 CS V2	Catalog DB2 in DataJoiner
DB2 for MVS or DB2 for OS/390	Catalog the DB2 database
Informix, Sybase, Sybase SQL Anywhere, MS SQL Server, or Oracle	Create a server mapping and user mapping within DataJoiner

You must define server and user mappings, as directed in the *DB2 DataJoiner Planning, Installation, and Configuration Guide*, for each DataJoiner database that requires access to a source or target. You must have one DataJoiner database for *each* non-IBM replication *source* server. You can support many non-IBM replication *target* servers with one DataJoiner database. The DataJoiner databases that you set up reside on one DataJoiner instance.

Connecting clients to DataJoiner

For Apply programs on DB2 UDB or DB2 CS, refer to the applicable chapter of *DB2 DataJoiner Planning, Installation, and Configuration Guide* for information about your network's communication protocol: TCP/IP, IPX/SPX, or APPC.

The steps that are described in the "Connecting Clients to DB2 DataJoiner" section in the *DB2 DataJoiner Planning, Installation, and Configuration Guide* must be performed for each DataJoiner database that is created to replicate with a remote Apply program.

What to do next

Before proceeding to "Chapter 18. Installing DJRA and connecting all databases" on page 283, make sure that you install all Capture and Apply programs on the systems of your choice.

Chapter 18. Installing DJRA and connecting all databases

This chapter describes how to install DJRA and complete the setup of your replication environment. In this chapter you:

- Install DJRA on your Windows 95 or NT workstation.
- Configure access from the workstation on which DJRA is installed to each source, target, and control server.
- Set the DB2CODEPAGE environment variable for DataJoiner for AIX access.
- In DJRA, set replication-administrative preferences.
- From the workstation on which DJRA is installed, bind DB2 utilities and DB2 CLI to all source, target, and control servers.
- Using DJRA, create replication control tables at the source, target, and control servers.
- For the DB2 source server, bind the Capture program to the DB2 source server. Bind the Apply program to each source, target, and control server. Capture triggers, placed at non-IBM databases, need not be bound.
- From where the Apply program is located, set end-user authentication.

Installing DJRA

DJRA includes ObjectREXX as part of the installation. If you do not already have ObjectREXX installed DJRA will install it, otherwise DJRA will use your existing copy.

If you are installing DJRA on Windows NT and have not installed DJRA as part of the DataJoiner installation process, log on to Windows NT with a valid user name that has administrator authority. If you are installing DJRA on Windows 95, no administrator log on is required.

To install DJRA:

1. Insert the DB2 DataJoiner CD-ROM into the appropriate drive. Or download DJRA from the Web.[18]
2. Use the setup program:
   a. Open the Run window by using one of the following methods:
      - From the Program Manager, select **File** –>**Run**.
      - Select **Start**, then select **Run**.

18. http://www.software.ibm.com/data/dpropr

b. In the **Command Line** field, type the following command:

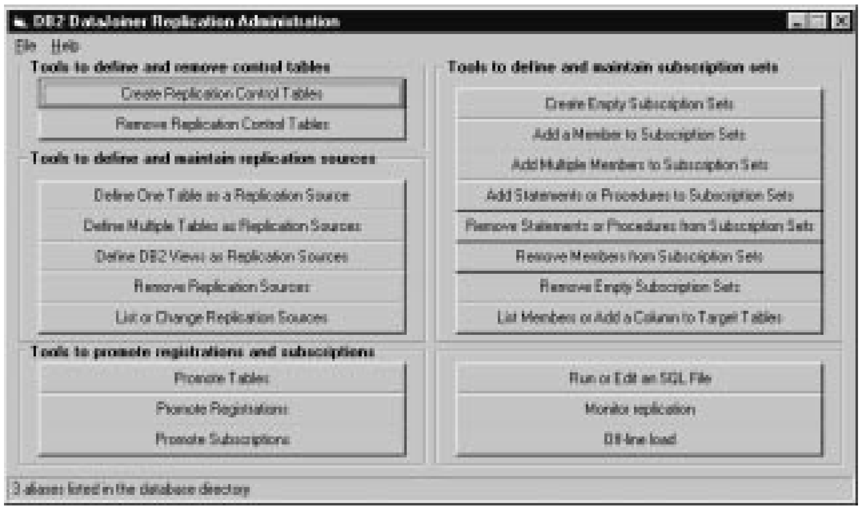
```
path\SETUP.exe
```

where *path* is the letter that designates your CD-ROM drive or the drive and directory to which you downloaded the DJRA file.

c. Click **OK** on the Run window.

The first Setup window opens.

3. Respond to the setup program's prompts. Online help is available to help you with the remaining steps. When you complete setup, DJRA appears in the Windows Program directory.
4. To start DJRA:
   a. Click the **Start** icon.
   b. Select the **Programs** menu.
   c. Select the **DataJoiner for Windows** menu.
   d. Select the **Replication** menu.
   e. Select **Replication Administration**. The DJRA primary window opens, as shown in Figure 22.


DB2 DataJoiner Replication Administration
File
Help
Tools to define and remove control tables
Create Replication Control Tables
Remove Replication Control Tables
Tools to define and maintain replication sources
Define One Table as a Replication Source
Define Multiple Tables as Replication Sources
Define DB2 Views as Replication Sources
Remove Replication Sources
List or Change Replication Sources
Tools to promote registrations and subscriptions
Promote Tables
Promote Registrations
Promote Subscriptions
Tools to define and maintain subscription sets
Create Empty Subscription Sets
Add a Member to Subscription Sets
Add Multiple Members to Subscription Sets
Add Statements or Procedures to Subscription Sets
Remove Statements or Procedures from Subscription Sets
Remove Members from Subscription Sets
Remove Empty Subscription Sets
List Members or Add a Column to Target Tables
Run or Edit an SQL File
Monitor replication
Off-line load

3 aliases listed in the database directory

Figure 22. DJRA Primary Window

Configuring access from DJRA to DataJoiner and DB2

Ensure that the machine on which you installed DJRA can access all source, target, and control servers directly or through DataJoiner. Refer to:

- *DB2 DataJoiner Planning, Installation, and Configuration Guide*, the chapter applicable to your network protocol: TCP/IP, IPX/SPX, APPC, or NetBIOS
- *Installing and Using DB2 Clients for Windows 95 and Windows NT: V2*
- *Installing and Configuring DB2 Clients*
- *DDCS User's Guide*, V2.3

Setting DB2CODEPAGE for DataJoiner for AIX access

If DJRA accesses DataJoiner for AIX, set the DB2CODEPAGE environment variable from your DJRA workstation. The value that you set is based on your country code. For example, if your country code is US, you would:

1. Select the **My Computer** icon.
2. Select the **Select System** icon.
3. From the System Properties folder, select **Environment**.
4. Type `DB2CODEPAGE` in **Variable**.
5. For US, type the value `437` in **User Variables**. For international English, type `850`.

Setting administrative preferences

You can specify your preferences for:

- DJRA working directory
- Location of console or file output
- User IDs and passwords
- Tracing SQL execution activity and Replication Monitor activity
- Converting table and qualifier names to uppercase or lowercase for Sybase or Microsoft SQL Server targets

To set preferences, select **File –> Preferences** from the menu on the DJRA primary window. You can change the preferences whenever you want to.

On the Connection page of the Preferences notebook, you see a list of databases that are currently cataloged on your system.

Restriction: If you are using Microsoft SQL Server in your replication environment, do not use an alias user ID. Microsoft SQL Server will reject the alias user ID.

Binding to source, target, and control servers

Bind DB2 utilities to each source, target, and control server, including DataJoiner. You must run the **bind** command separately for each DB2 or DataJoiner source, target, or control server. From the DJRA workstation, you must bind DB2 utilities and DB2 CLI; if you have a mixture of Windows NT and Windows 95 workstations, you must bind DB2 utilities to each DataJoiner and each DB2 database from at least one Window 95 workstation and at least one Window NT workstation. An example of what you enter is:

```
cd c:\sqllib\bnd
db2 connect to data-server-db
db2 bind @db2ubind.lst blocking all grant public
```

Binding the Capture and Apply programs in DB2 systems

Before you bind your Capture and Apply programs, make sure that you created control tables as described in "Creating replication control tables using DJRA" on page 89.

From the DB2 source server, you must bind the Capture program to the DB2 source server. Bind the Apply program to the logical source, target, and control servers.

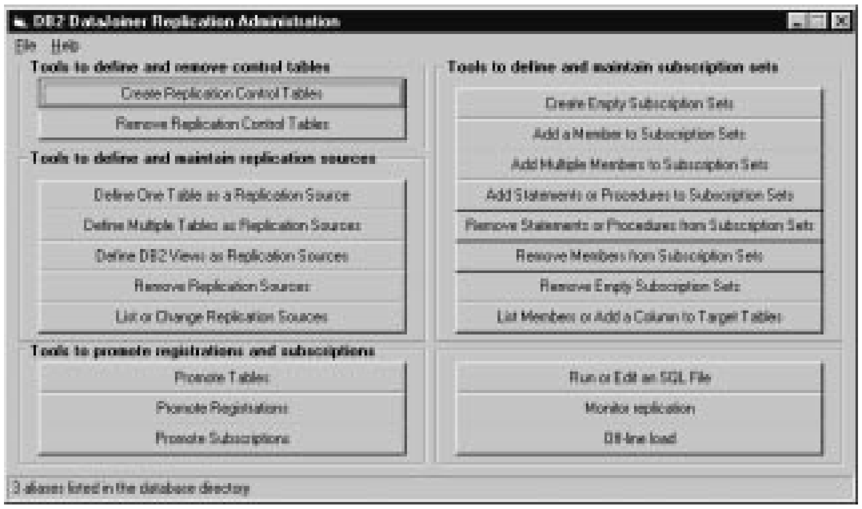
Chapter 19. Starting and using DJRA

This section describes how to begin running DJRA and describes the basic process of setting up replication.

Starting DJRA

To start DJRA:

1. Start all databases involved in replication.
2. Click the **Start** icon on the Windows 95 or NT desktop.
3. Select the **Programs** menu.
4. Select the **DataJoiner for Windows** menu.
5. Select the **Replication** menu.
6. Select **Replication Administration**. The DJRA primary window opens, as shown in Figure 23.


DB2 DataJoiner Replication Administration
File
Help
Tools to define and remove control tables
Create Replication Control Tables
Remove Replication Control Tables
Tools to define and maintain replication sources
Define One Table as a Replication Source
Define Multiple Tables as Replication Sources
Define DB2 Views as Replication Sources
Remove Replication Sources
List or Change Replication Sources
Tools to promote registrations and subscriptions
Promote Tables
Promote Registrations
Promote Subscriptions
Tools to define and maintain subscription sets
Create Empty Subscription Sets
Add a Member to Subscription Sets
Add Multiple Members to Subscription Sets
Add Statements or Procedures to Subscription Sets
Remove Statements or Procedures from Subscription Sets
Remove Members from Subscription Sets
Remove Empty Subscription Sets
List Members or Add a Column to Target Tables
Run or Edit an SQL File
Monitor replication
Off-line load

3 aliases listed in the database directory

Figure 23. DJRA Primary Window

General steps for setting up replication

Generally, each function in DJRA consists of five steps:

1. Select the function that you want (for example, **Define One Table as a Replication Source** or **Add a Member to Subscription Sets**). A window opens.
2. If applicable, edit DJRA logic to affect the statements that are created when you generate SQL by selecting the **Edit Logic**, **Edit Predicate Logic**, or **Edit Create Table Logic** push button from the DJRA function window. See "Editing DJRA logic" for more information.
3. Generate the SQL.
   a. Provide the required information, as prompted in the displayed window.
   b. Create the SQL by clicking **Generate SQL**. SQL is generated into a file but not run yet.
4. Review and edit, if needed, the generated SQL file from the console window or by clicking **Run or Edit an SQL file** from the primary window.
5. Run the SQL file by clicking **Run or Edit an SQL file** and then clicking **Run** from the primary window or by clicking **Run** from the console window.

   Make sure that you run or save your generated SQL files before generating another set of SQL files. If you do not run your SQL after it is generated, DJRA might generate duplicate names for objects. See "Running DJRA-generated SQL" on page 289 for more information.

Editing DJRA logic

You can edit the following files: SRCESVR.REX, TARGSVR.REX, CNTRLSVR.REX, and TBLSPACE.REX.

SRCESVR.REX
: Specifies the owner and name of the CD or CCD table and the table space in which the CD or CCD table is placed. Define one or multiple tables as replication sources and click **Edit Logic** to edit SRCESVR.REX before you generate SQL.

CNTRLSVR.REX
: Specifies the criteria by which rows from the source table can be replicated to the target. You can specify which columns in a source table are eligible for replication. You also can specify values to search individual source tables and replicate only the data that matches the value criteria. Add one or multiple members to subscription sets and click **Edit Predicate Logic** to edit the CNTLSVR.REX file.

TARGSVR.REX
: Specifies the table space or segment in which to create target tables. Check this file to make sure that the table spaces are being defined in the location of the target database where you want them. Add one or

multiple members to subscription sets and click **Edit Create Table Logic** to edit the TARGSVR.REX file.

TBLSPACE.REX
Specifies the table spaces for the control tables and the UOW table. Check this file to make sure that the table spaces are being defined in the location where you want them. Select **Create Replication Control Tables** and click **Edit Tablespace Logic** to edit the TBLSPACE.REX file.

You can specify where table spaces are created in the SRCESVR.REX and TARGSVR.REX files by changing the default directory (C:\) to the directory that you prefer. When you type in your directory, make sure that you add a backslash (\) after the directory. For example, if you are changing the directory from C:\ to F:\Test\, make sure that you type a backslash before the word Test and after the word Test. If you just type F:\Test, an error will occur when you attempt to generate SQL.

Editing DJRA-generated SQL

You can edit DJRA's SQL from the console window or by clicking **Run or Edit an SQL file** from the primary window. You can edit the SQL for several reasons. For example, you might want to:

- Edit create table and index statements to represent clusters and other database objects.
- For Oracle and other remote servers, ensure that tables are created in the existing table spaces that you want.
- For Microsoft SQL Server, create control tables on an existing segment.
- Review and edit subscription member predicates as a way of defining multiple subscriptions at one time. You can use substitute variables in your predicates and resolve the variables with programming logic.

When editing generated SQL, be careful not to change special markers that DJRA places within the SQL. For example, :ENDOFTRIGGER: or :ENDOFPROCEDURE: is part of a comment that is necessary for DJRA to run successfully. Altering create trigger blocks can result in incorrect SQL that ends in error when run.

Running DJRA-generated SQL

The **Run SQL** push button is intended to be used for SQL generated by DJRA. SQL that you generate outside DJRA might not run successfully if you use DJRA to start it. Likewise, you might not be able to run SQL generated by DJRA at a DB2 command line.

Recommendation: Run DJRA-generated SQL from DJRA.

Running the Capture and Apply programs

After you define replication sources and create subscription sets and members, you are ready to run the Capture and Apply programs. Capture triggers begin running automatically.

Chapter 20. DB2 DataJoiner with DJRA: data typing

When you are defining replication sources for non-IBM database tables, data typing often requires extra steps. When the source table is a DataJoiner nickname, DataJoiner handles any data-typing transformations when the nickname of the source table is created outside of DJRA. For more information, see the *IBM DB2 DataJoiner Application Programming and SQL Reference Supplement*.

During the subscription process, the final data mappings occur. When the target is a table that is accessed through a DataJoiner nickname, the DataJoiner nickname process does not always create the correct type, because:

- DataJoiner requires that the target table be created in the non-IBM database *before* a nickname is created on DataJoiner.
- When the nickname is created, DataJoiner chooses a data type known to DB2 common server that maps to the largest possible value in the target table.
- Sometimes the largest value from the source table (typically in DB2 for OS/390), is not the largest value that you can store in the target column.
- DJRA generates ALTER NICKNAME statements for the COLTYPE, SCALE, and LENGTH in the DataJoiner database for the target table nickname.

For example, when you replicate from DB2 to Microsoft SQL Server, and one of the DB2 columns has a type Date, the target SQL Server table has a type DateTime. When you create a nickname in DataJoiner for the SQL Server table, DataJoiner maps that column to Timestamp. DJRA, because it understands the data type of the source tables and the target tables, alters the nickname to use a type Date.

In general, you should let DJRA create the target table and ALTER NICKNAME statements. Otherwise, you must take care to choose the same data-type mappings that DataJoiner would generate if you want to create your own target tables on non-IBM databases, and you must ensure that these types are compatible with the source tables.

This section discusses how DJRA with DataJoiner handles data typing in six specific scenarios.

DB2-to-Oracle replication

The source is DB2 and the target is Oracle, the following restrictions and conversions are performed:

Table 36. Data Type Conversion: DB2 to Oracle

DB2 for OS/390 Source data type	Oracle Target data type	Changes after Create Nickname		
		COLTYPE	LENGTH	SCALE
FOR BIT DATA(n)	RAW(n)			
CHAR (<256)	CHAR(n)			
CHAR (>=256) VARCHAR2(n)	VARCHAR2(n)			
VARCHAR(n)	VARCHAR2(n)			
GRAPHIC(n)	CHAR(n)			
VARGRAPHIC(n)	VARCHAR(n)			
LONG VARCHAR	LONG			
DATE	DATE	DATE	4	
TIMESTAMP	DATE			
TIME	DATE	TIME	3	
SMALLINT	NUMBER(5)	DECIMAL	5	0
INTEGER	NUMBER(10)	DECIMAL	10	0
DECIMAL(n,m)	DECIMAL(n,m)	DECIMAL	n	m
FLOAT	FLOAT			

DB2-to-Informix replication

When the source is DB2 and the target is Informix, the following restrictions and conversions are performed:

Table 37. Data Type Conversion: DB2 to Informix

DB2 for OS/390 Source data type	Informix Target data type	Changes after Create Nickname		
		COLTYPE	LENGTH	SCALE
FOR BIT DATA(n)	CHAR(n) [1]			
CHAR(n)	CHAR(n)			
VARCHAR(<256)	VARCHAR(n)			
VARCHAR(>=256)	TEXT			
GRAPHIC(n)	BYTE			
VARGRAPHIC(n)	BYTE		No updates occur after create nickname	
LONG VARCHAR	TEXT			
DATE	DATE			
TIMESTAMP	DATETIME			

Table 37. Data Type Conversion: DB2 to Informix (continued)

DB2 for OS/390 Source data type	Informix Target data type	Changes after Create Nickname		
		COLTYPE	LENGTH	SCALE
	YEAR TO FRACTION(5)			
TIME	DATETIME HOUR TO SECOND			
SMALLINT	SMALLINT			
INTEGER	INT			
DECIMAL(n,m)	DECIMAL(n,m)			
FLOAT	FLOAT			

[1]For CHAR data in Informix, a string is terminated when the first non-printable character is encountered. In this example, the CHAR FOR BIT data from DB2 for OS/390 could be truncated when stored in Informix, making it appear as though X'00' is part of the string.

DB2 to Microsoft SQL Server, Sybase, or Sybase SQL Anywhere replication

When the source is DB2 and the target is MS SQL Server, Sybase, or Sybase SQL Anywhere, the following restrictions and conversions are performed:

Table 38. Data Type Conversion: DB2 to MS SQL Server, Sybase, or SQL Anywhere

DB2 for OS/390 SOURCE	MS SQL SERVER, SYBASE, OR SQL ANYWHERE TARGET	Changes after Create Nickname		
		COLTYPE	LENGTH	SCALE
CHAR (n) FOR BIT DATA	BINARY(n)	CHAR [1]		
VARCHAR(n) FOR BIT DATA(N)	VARBINARY(n) or BINARY(n)			
CHAR(n)	CHAR(n)			
VARCHAR(<256)	VARCHAR2(n) or VARCHAR(n)			
VARCHAR(<256)	TEXT			
GRAPHIC(n) [2]	?			
VARGRAPHIC(n) [2]	?			
LONG VARCHAR [2]	?			
DATE	DATETIME or DATE	DATE	4	
TIMESTAMP	DATETIME or DATESTAMP			
TIME	DATETIME or TIME	TIME	3	

Table 38. Data Type Conversion: DB2 to MS SQL Server, Sybase, or SQL Anywhere (continued)

DB2 for OS/390 SOURCE	MS SQL SERVER, SYBASE, OR SQL ANYWHERE TARGET	Changes after Create Nickname		
		COLTYPE	LENGTH	SCALE
SMALLINT	SMALLINT			
INTEGER	INT			
DECIMAL(n,m)	DECIMAL(n,m)			
FLOAT	FLOAT			

[1]DJRA creates the target table in MS SQL Server with data type ″binary.″ The DataJoiner nickname is created with COLTYPE of VARCHAR. DJRA updates the COLTYPE to CHAR.

[2]These DB2 for OS/390 data types have not been tested.

DB2 to Microsoft Jet replication

When the source is DB2 and the target is Microsoft Jet, the following restrictions and conversions are performed:

Table 39. Data Type Conversion: DB2 to Microsoft Jet

DB2 for OS/390 SOURCE	MS JET TARGET	Changes after Create Nickname		
		COLTYPE	LENGTH	SCALE
CHAR (n) FOR BIT DATA	Memo or OLE Object[1]			
VARCHAR(n) FOR BIT DATA(N)	Memo or OLE Object[1]			
CHAR(n)	Text or Memo[1]			
VARCHAR(<256)	Text or Memo[1]			
GRAPHIC(n) [2]	?			
VARGRAPHIC(n) [2]	?			
LONG VARCHAR [2]	Memo			
DATE	Datetime			
TIMESTAMP	Datetime			
TIME	Text			
SMALLINT	Number (Integer)			
INTEGER	Number (Long Integer)			
DECIMAL(n,m), NUMERIC	Number (Integer), Number (Long Integer), Number (Double), or Text[1]			
PRECISION, FLOAT, DOUBLE	Number (Double)			

[1]The value used depends on the precision set for the field.

[2]These DB2 for OS/390 data types have not been tested.

Part 6. Reference information

This part of the book contains the following chapters:

"Chapter 21. Table structures" on page 299 describes the source, control, and target table structures.

"Chapter 22. Problem determination facilities" on page 357 describes the available problem determination facilities.

"Chapter 23. Capture and Apply messages" on page 371 lists all of the messages issued by the Capture and Apply programs.

Chapter 21. Table structures

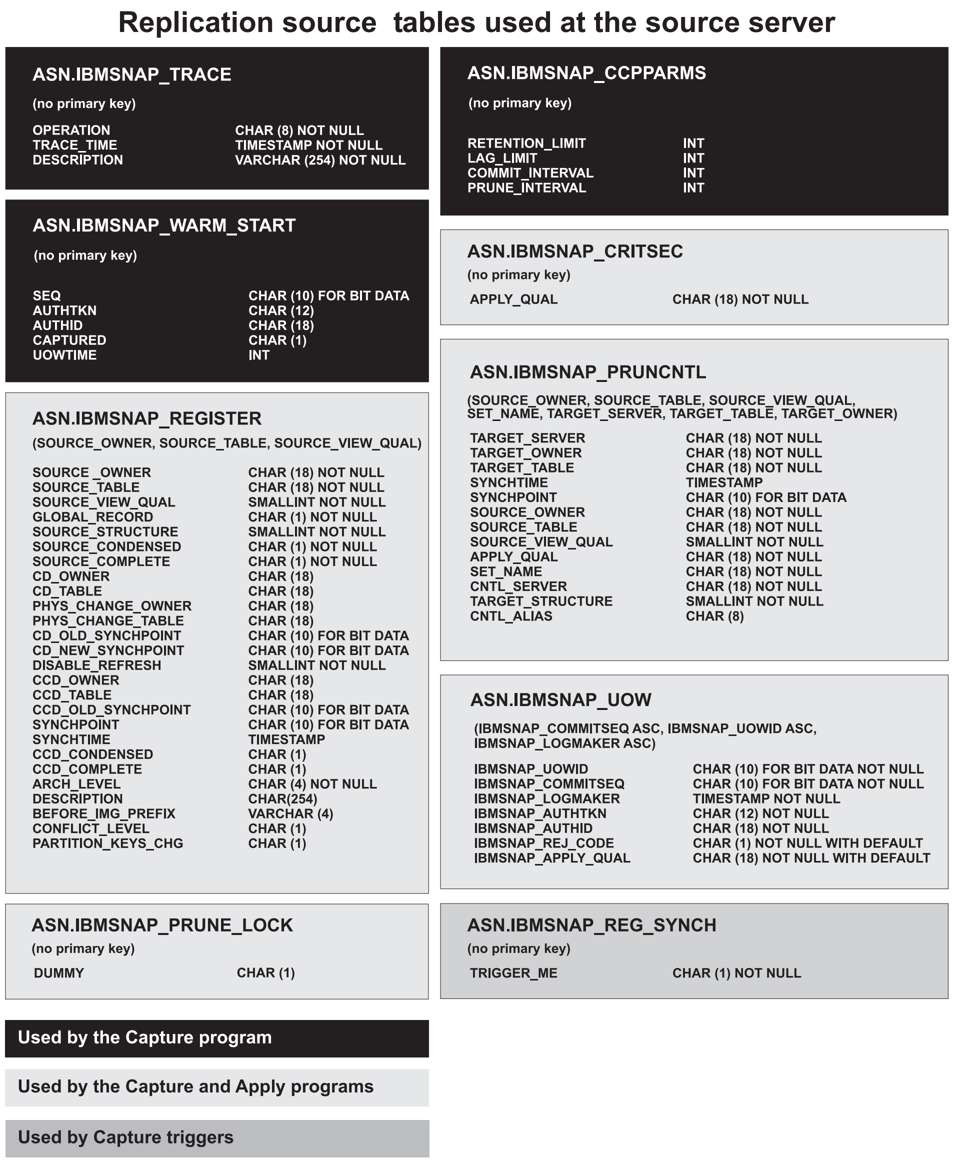
This chapter describes the relational database tables that DB2 DataPropagator uses during its replication process.

Table 40 on page 302, Table 41 on page 304, and Table 42 on page 305 provide directories to and brief descriptions of the tables listed in this chapter. When you become familiar with the tables, you can use the following charts as a quick reference of the source and control server tables, table keys, and their parameters.

Important: Do *not* use SQL to update any of the control tables other than the register, tuning parameter, subscription events, consistent-change-data (CCD), and replica tables. Altering control tables inappropriately can cause unexpected results and loss of data.

Tables at a glance

The following two diagrams show the source and control server tables, table keys, and their parameters.



Replication source tables used at the source server

ASN.IBMSNAP_TRACE

(no primary key)

OPERATION	CHAR (8) NOT NULL
TRACE_TIME	TIMESTAMP NOT NULL
DESCRIPTION	VARCHAR (254) NOT NULL

ASN.IBMSNAP_CCPPARMS

(no primary key)

RETENTION_LIMIT	INT
LAG_LIMIT	INT
COMMIT_INTERVAL	INT
PRUNE_INTERVAL	INT

ASN.IBMSNAP_WARM_START

(no primary key)

SEQ	CHAR (10) FOR BIT DATA
AUTHTKN	CHAR (12)
AUTHID	CHAR (18)
CAPTURED	CHAR (1)
UOWTIME	INT

ASN.IBMSNAP_CRITSEC

(no primary key)

APPLY_QUAL	CHAR (18) NOT NULL

ASN.IBMSNAP_PRUNCNTL

(SOURCE_OWNER, SOURCE_TABLE, SOURCE_VIEW_QUAL, SET_NAME, TARGET_SERVER, TARGET_TABLE, TARGET_OWNER)

TARGET_SERVER	CHAR (18) NOT NULL
TARGET_OWNER	CHAR (18) NOT NULL
TARGET_TABLE	CHAR (18) NOT NULL
SYNCHTIME	TIMESTAMP
SYNCHPOINT	CHAR (10) FOR BIT DATA
SOURCE_OWNER	CHAR (18) NOT NULL
SOURCE_TABLE	CHAR (18) NOT NULL
SOURCE_VIEW_QUAL	SMALLINT NOT NULL
APPLY_QUAL	CHAR (18) NOT NULL
SET_NAME	CHAR (18) NOT NULL
CNTL_SERVER	CHAR (18) NOT NULL
TARGET_STRUCTURE	SMALLINT NOT NULL
CNTL_ALIAS	CHAR (8)

ASN.IBMSNAP_REGISTER

(SOURCE_OWNER, SOURCE_TABLE, SOURCE_VIEW_QUAL)

SOURCE _OWNER	CHAR (18) NOT NULL
SOURCE_TABLE	CHAR (18) NOT NULL
SOURCE_VIEW_QUAL	SMALLINT NOT NULL
GLOBAL_RECORD	CHAR (1) NOT NULL
SOURCE_STRUCTURE	SMALLINT NOT NULL
SOURCE_CONDENSED	CHAR (1) NOT NULL
SOURCE_COMPLETE	CHAR (1) NOT NULL
CD_OWNER	CHAR (18)
CD_TABLE	CHAR (18)
PHYS_CHANGE_OWNER	CHAR (18)
PHYS_CHANGE_TABLE	CHAR (18)
CD_OLD_SYNCHPOINT	CHAR (10) FOR BIT DATA
CD_NEW_SYNCHPOINT	CHAR (10) FOR BIT DATA
DISABLE_REFRESH	SMALLINT NOT NULL
CCD_OWNER	CHAR (18)
CCD_TABLE	CHAR (18)
CCD_OLD_SYNCHPOINT	CHAR (10) FOR BIT DATA
SYNCHPOINT	CHAR (10) FOR BIT DATA
SYNCHTIME	TIMESTAMP
CCD_CONDENSED	CHAR (1)
CCD_COMPLETE	CHAR (1)
ARCH_LEVEL	CHAR (4) NOT NULL
DESCRIPTION	CHAR(254)
BEFORE_IMG_PREFIX	VARCHAR (4)
CONFLICT_LEVEL	CHAR (1)
PARTITION_KEYS_CHG	CHAR (1)

ASN.IBMSNAP_UOW

(IBMSNAP_COMMITSEQ ASC, IBMSNAP_UOWID ASC, IBMSNAP_LOGMAKER ASC)

IBMSNAP_UOWID	CHAR (10) FOR BIT DATA NOT NULL
IBMSNAP_COMMITSEQ	CHAR (10) FOR BIT DATA NOT NULL
IBMSNAP_LOGMAKER	TIMESTAMP NOT NULL
IBMSNAP_AUTHTKN	CHAR (12) NOT NULL
IBMSNAP_AUTHID	CHAR (18) NOT NULL
IBMSNAP_REJ_CODE	CHAR (1) NOT NULL WITH DEFAULT
IBMSNAP_APPLY_QUAL	CHAR (18) NOT NULL WITH DEFAULT

ASN.IBMSNAP_PRUNE_LOCK

(no primary key)

DUMMY	CHAR (1)

ASN.IBMSNAP_REG_SYNCH

(no primary key)

TRIGGER_ME	CHAR (1) NOT NULL

Used by the Capture program

Used by the Capture and Apply programs

Used by Capture triggers

Figure 24. Replication Source Tables Used at the Source Server. The replication source tables used by the Capture program, Apply program, and Capture triggers.

Subscription tables used at the control server

ASN.IBMSNAP_APPLYTRAIL

(no primary key)

APPLY_QUAL	CHAR (18) NOT NULL
SET_NAME	CHAR (18) NOT NULL
WHOS_ON_FIRST	CHAR (1) NOT NULL
ASNLOAD	CHAR (1)
MASS_DELETE	CHAR (1)
EFFECTIVE_MEMBERS	INT
SET_INSERTED	INT NOT NULL
SET_DELETED	INT NOT NULL
SET_UPDATED	INT NOT NULL
SET_REWORKED	INT NOT NULL
SET_REJECTED_TRXS	INT NOT NULL
STATUS	SMALLINT NOT NULL
LASTRUN	TIMESTAMP NOT NULL
LASTSUCCESS	TIMESTAMP
SYNCHPOINT	CHAR (10) FOR BIT DATA
SYNCHTIME	TIMESTAMP
SOURCE_SERVER	CHAR (18) NOT NULL
SOURCE_ALIAS	CHAR (8)
SOURCE_OWNER	CHAR (18)
SOURCE_TABLE	CHAR (18)
SOURCE_VIEW_QUAL	SMALLINT
TARGET_SERVER	CHAR (18) NOT NULL
TARGET_ALIAS	CHAR (8)
TARGET_OWNER	CHAR (18) NOT NULL
TARGET_TABLE	CHAR (18) NOT NULL
SQLSTATE	CHAR (5)
SQLCODE	INTEGER
SQLERRP	CHAR (8)
SQLERRM	VARCHAR (70)
APPERRM	VARCHAR (760)

ASN.IBMSNAP_SUBS_SET

(APPLY_QUAL, SET_NAME, WHOS_ON_FIRST)

APPLY_QUAL	CHAR (18) NOT NULL
SET_NAME	CHAR (18) NOT NULL
WHOS_ON_FIRST	CHAR (1) NOT NULL
ACTIVATE	SMALLINT NOT NULL
SOURCE_SERVER	CHAR (18) NOT NULL
SOURCE_ALIAS	CHAR (8)
TARGET_SERVER	CHAR (18) NOT NULL
TARGET_ALIAS	CHAR (8)
STATUS	SMALLINT NOT NULL
LASTRUN	TIMESTAMP NOT NULL
REFRESH_TIMING	CHAR (1) NOT NULL
SLEEP_MINUTES	INT
EVENT_NAME	CHAR (18)
LASTSUCCESS	TIMESTAMP
SYNCHPOINT	CHAR (10) FOR BIT DATA
SYNCHTIME	TIMESTAMP
MAX_SYNCH_MINUTES	INT
AUX_STMTS	SMALLINT NOT NULL
ARCH_LEVEL	CHAR (4) NOT NULL

ASN.IBMSNAP_SUBS_MEMBR

(APPLY_QUAL, SET_NAME, WHOS_ON_FIRST, SOURCE_OWNER, SOURCE_TABLE, SOURCE_VIEW_QUAL, TARGET_OWNER, TARGET_TABLE)

APPLY_QUAL	CHAR (18) NOT NULL
SET_NAME	CHAR (18) NOT NULL
WHOS_ON_FIRST	CHAR (1) NOT NULL
SOURCE_OWNER	CHAR (18) NOT NULL
SOURCE_TABLE	CHAR (18) NOT NULL
SOURCE_VIEW_QUAL	SMALLINT NOT NULL
TARGET_OWNER	CHAR (18) NOT NULL
TARGET_TABLE	CHAR (18) NOT NULL
TARGET_CONDENSED	CHAR (1) NOT NULL
TARGET_COMPLETE	CHAR (1) NOT NULL
TARGET_STRUCTURE	SMALLINT NOT NULL
PREDICATES	VARCHAR (512)

ASN.IBMSNAP_SUBS_EVENT

(EVENT_NAME, EVENT_TIME)

EVENT_NAME	CHAR (18) NOT NULL
EVENT_TIME	TIMESTAMP NOT NULL
END_OF_PERIOD	TIMESTAMP

ASN.IBMSNAP_SUBS_STMTS

(APPLY_QUAL, SET_NAME, WHOS_ON_FIRST, BEFORE_OR_AFTER, STMT_NUMBER)

APPLY_QUAL	CHAR (18) NOT NULL
SET_NAME	CHAR (18) NOT NULL
WHOS_ON_FIRST	CHAR (1) NOT NULL
BEFORE_OR_AFTER	CHAR (1) NOT NULL
STMT_NUMBER	SMALLINT NOT NULL
EI_OR_CALL	CHAR (1) NOT NULL
SQL_STMT	VARCHAR (1O24)
ACCEPT_SQLSTATES	VARCHAR (50)

ASN.IBMSNAP_SUBS_COLS

(APPLY_QUAL, SET_NAME, WHOS_ON_FIRST, TARGET_OWNER, TARGET_TABLE, TARGET_NAME)

APPLY_QUAL	CHAR (18) NOT NULL
SET_NAME	CHAR (18) NOT NULL
WHOS_ON_FIRST	CHAR (1) NOT NULL
TARGET_OWNER	CHAR (18) NOT NULL
TARGET_TABLE	CHAR (18) NOT NULL
COL_TYPE	CHAR (1) NOT NULL
TARGET_NAME	CHAR (18) NOT NULL
IS_KEY	CHAR (1) NOT NULL
COLNO	SMALLINT NOT NULL
EXPRESSION	VARCHAR (254) NOT NULL

Used by the Apply program

Figure 25. The Control Server Subscription Tables. The control server subscription tables used by the Apply program.

List of tables used at the source server

The following table provides a list of tables used at the source server.

Table 40. Quick Reference for Tables Used at the Source Server during DB2 DataPropagator Processes

Table name	Internal name and Description	See page
Apply-qualifier-cross-reference table (AS/400 specific)	**ASN.IBMSNAP_AUTHTKN** Contains information to support update-anywhere.	322
Capture enqueue table (VM and VSE specific)	**ASN.IBMSNAP_CCPENQ** Used to ensure that only one Capture program is running per database.	318
Change data table	**CD** A staging table that contains changed data information. Created when a replication source is defined.	326
Critical section table	**ASN.IBMSNAP_CRITSEC** Used to prevent circular replication for update-anywhere subscriptions.	320
Pruning control table	**ASN.IBMSNAP_PRUNCNTL** Coordinates synchpoint updates by allowing the Apply program to communicate with the Capture program and coordinates pruning of tables. There is one pruning control table at each source server and one row per source-to-target copy.	314
Prune lock table	**ASN.IBMSNAP_PRUNE_LOCK** Used to serialize the access of staging tables during a cold start or retention limit pruning.	321
Register table	**ASN.IBMSNAP_REGISTER** Contains information about replication sources, such as the names of the replication source tables, their attributes, and their corresponding CD and CCD table names.	307

Table 40. Quick Reference for Tables Used at the Source Server during DB2 DataPropagator Processes (continued)

Table name	Internal name and Description	See page
Register extension table (AS/400 specific)	**ASN.IBMSNAP_REG_EXT** An extension of the replication source table, ASN.IBMSNAP_REGISTER. Contains additional information about replication sources, such as the journal name and the remote source table's RDB entry name.	313
Register synchronization table	**ASN.IBMSNAP_REG_SYNCH** When replicating from a non-IBM data source, an update trigger on this table initiates an update of the SYNCHPOINT value for all the rows in the register table before the Apply program reads the information from the register table.	323
Trace table	**ASN.IBMSNAP_TRACE** Contains Capture program audit trail information.	321
Tuning parameters table	**ASN.IBMSNAP_CCPPARMS** Contains parameters that you can modify to control the performance of the Capture program.	317
Unit-of-work table	**ASN.IBMSNAP_UOW** Contains information about committed transactions. Used to maintain transaction consistency.	324
Warm start table	**ASN.IBMSNAP_WARM_START** Contains information that enables the Capture program to resume capturing from the point in the log or journal where it last stopped. For AS/400 platforms, this table is used to determine the starting time of the **RCVJRNE** (Receive Journal Entry) command.	318

List of tables used at the control server

The following table provides a list of tables used at the control server.

Table 41. Quick Reference for Tables Created at the Control Server during DB2 DataPropagator Processes

Table name	Internal name and Description	See page
Apply trail table	**ASN.IBMSNAP_APPLYTRAIL** Contains Apply program audit trail and problem diagnostic information.	340
Subscription columns table	**ASN.IBMSNAP_SUBS_COLS** Maps target table or view columns to the corresponding source table or view column or user-defined expression.	333
Subscription events table	**ASN.IBMSNAP_SUBS_EVENT** Contains user-defined event names that control the execution of a subscription set. This table can be updated by using SQL.	339
Subscription set table	**ASN.IBMSNAP_SUBS_SET** Contains processing information for a set of subscription members, which are processed by the Apply program as a group.	327
Subscription statements table	**ASN.IBMSNAP_SUBS_STMTS** Contains SQL statements or stored procedure calls that are embodied by a subscription set.	335
Subscription-targets-member table	**ASN.IBMSNAP_SUBS_MEMBR** Identifies a source and target table (or view) pair and specifies processing information for that pair.	331
Row-replica-target-list table (Microsoft Jet specific)	**ASN.IBMSNAP_SUBS_TGTS** Maintains the names of the row-replica tables. Row-replica tables are a type of target table used specifically with the Microsoft Jet database.	337
Subscription-schema-changes table (Microsoft Jet specific)	**ASN.IBMSNAP_SCHEMA_CHG** Used to signal add or delete modifications to a subscription.	338

List of tables used at the target server

The following table provides a list of tables used at the target server.

Table 42. Quick Reference for Target Tables

Table name	Internal name and Description	See page
Base aggregate table	*userid.target_table*.target_table Contains data aggregated from a source table.	350
Change aggregate table	*userid.target_table* Contains data aggregations based on changes from a source table.	351
Consistent-change-data table	*userid.target_table* Contains additional columns to help identify transactions. Individual operations, transactions, and the approximate time of those transactions are ordered in this table.	347
Point-in-time table	*userid.target_table* Identical to a user copy table, except that the IBMSNAP_LOGMARKER column is included to record a specific commit time from the source server.	346
Replica table	*userid.target_table* A type of update-anywhere target table.	349
Row-replica table	*userid.target_table* A type of Microsoft Jet target table that can be updated.	352
User copy table	*userid.target_table* A copy of the user table.	346
Conflict table (Microsoft Jet specific)	**IBMSNAP_***target_name***_CONFLICT** Contains row data for DataPropagator for Microsoft Jet-detected conflict losers.	352
Error information table (Microsoft Jet specific)	**IBMSNAP_ERROR_INFO** Contains additional information to identify the row-replica table and row that caused an error.	353

Table 42. Quick Reference for Target Tables (continued)

Table name	Internal name and Description	See page
Error messages table (Microsoft Jet specific)	**IBMSNAP_ERROR_MESSAGE** Contains error codes and error messages. There can be more than one row in this table. Depending on the error code, additional information will be available in the error information, error-side-information, and conflict tables.	353
Error-side-information table (Microsoft Jet specific)	**IBMSNAP_SIDE_INFO** Contains the names of the conflict tables.	354
Key string table (Microsoft Jet specific)	**IBMSNAP_GUID_KEY** Maps the Microsoft Jet table identifiers and row identifiers to primary key values when the following actions occur: • Rows are deleted from Microsoft Jet database tables. • Deletes are recorded in MSysTombstone with s_Generation, TableGUID and s_GUID (row) identifiers, but without primary key details. • The primary key values are needed to propagate Microsoft Jet database deletes to an RDBMS.	354
Synchronization generations table (Microsoft Jet specific)	**IBMSNAP_S_GENERATION** Prevents cyclic updates from propagating back to the RDBMS from a Microsoft Jet database.	355

Tables used at the source server

The following section provides a brief description of the tables used at the source server and the columns in each table. These tables are created automatically the first time that you define a replication source if you use the Control Center and they do not already exist on the source server.

The administration tools use the information in the register, register extension, and pruning control tables to define your source and target tables for replication. After you define your replication sources, the Capture program uses the tuning parameters, capture enqueue, warm start, critical section, trace, and Apply-qualifier-cross-reference tables to control and audit your data. In addition to all the Capture program control and audit tables mentioned, the Capture triggers also use the register synchronization table to control data. The UOW and CD tables track data that has not been replicated.

Register table

This table contains information that you can update by using SQL.

ASN.IBMSNAP_REGISTER

The register table contains information about replication sources, such as the names of the replication source tables, their attributes, and their staging table names. A row is automatically inserted into this table every time a new replication source is defined at this server. You must update this table to maintain an external CCD table.

The register table is the place to look if you need to know how you defined your replication sources.

Table 43 provides a brief description of the register table columns.

Table 43. Register Table Columns

Column name	Description
SOURCE_OWNER	The owner of the source table or view.
SOURCE_TABLE	The source from which data is being captured.
SOURCE_VIEW_QUAL	This value is set to 0 for physical tables that are defined as sources and is greater than 0 for views that are defined as sources. This column is used to support multiple subscriptions for different source views with identical SOURCE_OWNER and SOURCE_TABLE column values.
GLOBAL_RECORD	A flag that indicates whether this row is the global record. In the global record, only the SYNCHPOINT and SYNCHTIME columns are set by the Capture program to reflect its progress. If the Capture program has not been running, then there is no global record. **Y** This row is the global record. **N** This row is not the global record.

Table 43. Register Table Columns (continued)

Column name	Description
SOURCE_STRUCTURE	A value that identifies the structure of the source table or view: **1** User table **3** CCD table **4** Point-in-time table **5** Base aggregate table **6** Change aggregate table **7** Replica table **8** User copy table **9** Row-replica table
SOURCE_CONDENSED	A flag that indicates: **Y** For any given primary key, the CCD, replica, and user tables show only one row. **N** All changes must remain, retaining a complete update history. **A** Valid only for base aggregate or change aggregate tables.
SOURCE_COMPLETE	A flag that indicates: **Y** The source table contains a row for every primary key value of interest. **N** The source table contains some subset of rows of primary key values.
CD_OWNER	The owner of the change data table or a view.
CD_TABLE	The name of the change data table or view for captured updates to the source table (set when you define the replication source). This value is used by the Apply program and can be the name of a table or a view. The Capture program inserts one row into the CD table for every committed and uncommitted change to this replication source. The Apply program then joins this table with the UOW table so that only committed changes are replicated.

Table 43. Register Table Columns (continued)

Column name	Description
PHYS_CHANGE_OWNER	The owner of the PHYS_CHANGE_TABLE. For a view defined as a source, this value equals the value of a CD or CCD table referenced in the change data view definition. For non-view replication sources, the value equals the CD_OWNER or the CCD_OWNER column. The Capture program uses this value to properly maintain CD_OLD_SYNCHPOINT and CD_NEW_SYNCHPOINT for view replication sources. The Apply program uses this value to properly maintain CCD_OLD_SYNCHPOINT and SYNCHPOINT for view replication sources that are based on CCD tables that the Apply program maintains.
PHYS_CHANGE_TABLE	The name of the physical CD or CCD table. For a view replication source, this value equals the value of the CD or CCD table replication source definition referenced in the change data view definition. For non-view replication sources, the value equals the CD_TABLE or the CCD_TABLE column. The Capture program uses this value to properly maintain CD_OLD_SYNCHPOINT and CD_NEW_SYNCHPOINT for view replication sources. The Apply program uses this value to properly maintain CCD_OLD_SYNCHPOINT and SYNCHPOINT for view replication sources that are based on the CCD tables that the Apply program maintains.
CD_OLD_SYNCHPOINT	The approximate SYNCHPOINT value when the Capture program begins to capture changes from the source table. The Capture program sets this value to NULL during a cold start. The Apply program sets this value to NULL for a target replica when cascading a gap condition. If the value is null when the synchpoint column of the pruning control table is set to x’00000000000000000000’, the Capture program sets an initial value, and the same sequence number is reflected back into the SYNCHPOINT column of the pruning control table; this is the sequence number associated with the pruning control table update. Subsequent values are set by the Capture program when old rows are pruned from the table.
CD_NEW_SYNCHPOINT	The Capture program advances this column as it inserts new rows into the CD table. If the Capture program did not insert into the change data table recently, then the value does not advance. The Apply program uses this column to see if there are new changes to be replicated.

Table 43. Register Table Columns (continued)

Column name	Description
DISABLE_REFRESH	When this column is created, it contains the ’0’ flag. If you set the flag to ’1’, the Apply program is not allowed to perform a full refresh of the source server until the flag is set back to ’0’. This column is used to defer, not eliminate, a full refresh for a subscription. For example, you might want to defer a full refresh when the Capture program starts up cold or a gap in the log is detected. The Apply program will not process subscriptions to this replication source until control table values have been updated. This flag prevents full refresh activity from overloading the source database during peak periods. This column is initialized to ’0’. You can use a program at the source database site to set this flag. **0** Full refreshes are allowed. **1** Full refreshes are prevented.
CCD_OWNER	The owner of the local consistent-change-data table.
CCD_TABLE	The name of the staging table that contains committed-only captured updates.
CCD_OLD_SYNCHPOINT	The SYNCHPOINT value of the oldest row in the external CCD table. This value can be much older than any row remaining in the CCD table. This value is set in one of the following ways: • By the administration tool when the consistent-change-data table is automatically defined as a source. CCD_OLD_SYNCHPOINT is set to NULL. • By the Control Center when a consistent-change-data table is defined as an external replication source table. CCD_OLD_SYNCHPOINT is set to MIN(IBMSNAP_COMMITSEQ) of the consistent-change-data table. • By the Apply program or another external application. • Manually, for CCD replication sources that are not created and maintained by the Apply program. This is the case for CCD tables that contain IMS changes generated by DataPropagator NonRelational.

Table 43. Register Table Columns (continued)

Column name	Description
SYNCHPOINT	In the global row, where the GLOBAL_RECORD column = ’Y’, this is the log or journal identifier (synchpoint) of the last log or journal record processed by the Capture program. The Apply program compares this value to the last synchpoint that it processed to see if there are new changes available for replication. For CCD source definitions, this is the equivalent to CD_NEW_SYNCHPOINT and is updated by the Apply program that maintains the CCD table. This column must be set manually for a CCD replication source that is not created and maintained by the Apply program. An example is a CCD table of IMS changes generated by DataPropagator NonRelational.
SYNCHTIME	A source server timestamp. The Capture program or an external program, such as DataPropagator NonRelational, updates this timestamp whether there are changes to be processed or not. The Apply program uses this value when advanced conflict detection is selected for update-anywhere replication to ensure that the Capture program captured all outstanding changes for a replication source table.
CCD_CONDENSED	A flag that indicates: **Y** This CCD replication source has only the last captured change for a source table row. **N** This CCD replication source has one row for each source table row change.
CCD_COMPLETE	A flag that indicates: **Y** The CCD table contains a row for every primary key value of interest. **N** The CCD table is initially empty and then is populated with some subsets of primary key value rows.
ARCH_LEVEL	The architectural level of the definition in the row. This level is defined by IBM, and for Version 6 is ’0201’.
DESCRIPTION	A field for comments that you enter when defining replication sources.

Table 43. Register Table Columns (continued)

Column name	Description
BEFORE_IMG_PREFIX	Represents the default character identifying before-image column names in the CD table. The value can be NULL, but must not match any leading character identifying after-image user data column names in the CD table. The length of BEFORE_IMG_PREFIX is: **1** For an ASCII or an EBCDIC single-byte character system prefix character. **2** For an ASCII double-byte character system prefix character. **4** For an EBCDIC double-byte character system prefix character. This length allows for shift-in and shift-out characters.
CONFLICT_LEVEL	This column is assumed to never change and to be the same for all descendents of the user table. A flag that indicates: **0** The Apply program does not check for conflicts. Data consistency must be enforced by your application design to avoid potential conflicting updates. **1** Standard detection with cascading transaction rejection. The Apply program checks for conflicts based on the changes captured to this point. The Apply program will reverse any conflicting transaction at the replica, as well as any transactions with dependencies on the conflicting transaction. Changes captured after the Apply program begins conflict detection will not be checked during this Apply cycle. **2** Enhanced detection with cascading transaction rejection. The Apply program waits until the Capture program captures all changes from the log or journal (see description of the SYNCHTIME column) and then does a standard conflict detection (CONFLICT_LEVEL = 1). During the wait time, the Apply program holds a LOCK on the source tables to ensure that no changes are made during the conflict detection process.

Table 43. Register Table Columns (continued)

Column name	Description
PARTITION_KEYS_CHG	This value is assumed to be the same for all the user table's dependent replicas. A flag indicating: **N** Updates to the source table are staged by the Capture program as an update and processed by the Apply program as an update statement to the target table. **Y** Updates to the source table are staged by the Capture program as a delete and insert pair. The Apply program processes the delete first and the insert second. When this flag is set, every update to a replication source is stored in the CD table as two rows: a delete row and an insert row. This flag ensures that an update to a key or partitioning column is always processed correctly. Use this flag when: • The source columns for target table primary keys can be updated at the source table. • The source columns for target table partitioning columns that were defined in subscription predicates or in partitioned target databases can be updated at the source table. • The target table is stored in a DB2 for OS/390 partitioned table space and user transactions update all or part of the target table space's partitioning key. • The target table is a DB2 Extended Enterprise Edition table stored in a multi-node node group. **NULL** If this is the global control row.

Register extension table for AS/400

Important: Do *not* use SQL to update this table. Altering this table inappropriately can cause unexpected results and loss of data.

ASN.IBMSNAP_REG_EXT

This table is an AS/400-specific table that provides supplemental information for the register table, ASN.IBMSNAP_REGISTER. For every register table row, there is a matching register extension table row containing a few additional AS/400-specific columns.

Register extension table

Use this table to complete the information from the register table to track where and how you defined your replication sources on an AS/400 server.

Table 44 provides a brief description of the register extension table columns.

Table 44. Register Extension Table Columns

Column name	Description
SOURCE_OWNER	The owner of the source table or view.
SOURCE_TABLE	The source from which data is being captured.
SOURCE_NAME	A 10-character source table or view system name, used to issue commands.
SOURCE_MBR	The name of the source table member being captured. Used for issuing Receive Journal Entry (**RCVJRNE**) commands and ALIAS support.
SOURCE_TABLE_RDB	For remote-journal cases, this column contains the Relational Database (RDB) name of the system where the source table actually resides. For non-remote-journal cases, this column is NULL.
JRN_LIB	The library name of the journal that the source table uses.
JRN_NAME	The name of the journal used by a source table. An asterisk followed by nine blanks in this column means that the source table is not currently journaled in a journal. Therefore, it is not possible to capture data for this source table.
FR_START_TIME	Full refresh start time. This column is updated by Capture for AS/400, not the Control Center, during operations.
SOURCE_VIEW_QUAL	Supports the view of subscriptions by matching the similar column in the register table. This value is set to equal 0 for physical tables that are defined as a source and is greater than 0 for views that are defined as sources. You must have this column to support multiple subscriptions for different source views with identical SOURCE_OWNER and SOURCE_TABLE column values.

Pruning control table

Important: Do *not* use SQL to update this table. Altering this table inappropriately can cause unexpected results and loss of data.

ASN.IBMSNAP_PRUNCNTL

The pruning control table coordinates the pruning of the change data (CD) tables, which have the potential for unlimited growth. For each new subscription, the Apply program first updates the pruning control table and then it begins a full refresh for the new subscription. After the full refresh, the Capture program begins capturing changes from the replication source. When the Capture program begins to capture data, it updates the pruning control table to notify the Apply program. During each Apply cycle, the Apply program updates the pruning control table to indicate the last change applied. The Capture program then uses the information to prune the CD and UOW tables.

The rows in the pruning control table are not deleted during a cold start of the Capture program. The administration tools use the values from the pruning control table to provide a list of copies defined as source tables and views.

There is one pruning control table at each source server and one row in the pruning control table for each subscription-set member.

You can manually prune your table by issuing the **prune** command or have it done automatically by updating the PRUNE_INTERVAL column in the tuning parameters table. See "Tuning parameters table" on page 317 for more information about using the tuning parameters table.

Use this table to monitor the pruning status of your CD and UOW tables.

Table 45 provides a brief description of each of the pruning control table columns.

Table 45. Pruning Control Table Columns

Column name	Description
TARGET_SERVER	The remote database (RDB) name of the server where target tables or views are stored.
TARGET_OWNER	A qualifier for a target table or view.
TARGET_TABLE	The target to which data is being applied.
SYNCHTIME	A source server timestamp. The SYNCHTIME value equals the SYNCHTIME field value in the subscription set table. The Capture program or an external program, such as DataPropagator NonRelational, updates this timestamp whether there are changes to be processed or not. The Apply program uses this value when advanced conflict detection is selected for update-anywhere replication to ensure that the Capture program captures all outstanding changes for a replication source table.

Table 45. Pruning Control Table Columns (continued)

Column name	Description
SYNCHPOINT	The SYNCHPOINT value equals the SYNCHPOINT field value in the subscription set table. This value is used to coordinate the pruning of CD tables. The Apply program sets this initial value to hex 0s, indicating refresh. If the Apply program sets a nonzero value, the CD table can be eligible for pruning.
SOURCE_OWNER	The owner of the source table or view.
SOURCE_TABLE	The source from which data is being captured.
SOURCE_VIEW_QUAL	Supports the view of physical tables by matching the similar column in the register table. This value is set to 0 for physical tables that are defined as sources and is greater than 0 for views that are defined as sources. This column is used to support multiple subscriptions for different source views with identical SOURCE_OWNER and SOURCE_TABLE column values.
APPLY_QUAL	A unique identifier for a group of subscription sets. This case sensitive value is supplied by the user when defining a subscription set. This column is part of the foreign key from the subscription set table. See "Subscription set table" on page 327 for more details.
SET_NAME	An identifier for a group of subscription-set members. This value is supplied by the user when defining a subscription set. This column is part of the foreign key from the subscription set table. See "Subscription set table" on page 327 for more details.
CNTL_SERVER	The RDB name of the control server for the Apply program updating this row.
TARGET_STRUCTURE	A value that identifies the type of target table or view: **1** Source table **2** Not available **3** CCD table **4** Point-in-time table **5** Base aggregate table **6** Change aggregate table **7** Replica table **8** User copy table **9** Row-replica table

Table 45. Pruning Control Table Columns (continued)

Column name	Description
CNTL_ALIAS	The DB2 Universal Database alias corresponding to the control server named in the CNTL_SERVER column.

Tuning parameters table

This table contains information that you can update by using SQL.

ASN.IBMSNAP_CCPPARMS

This table contains parameters that you can modify to control the performance of the Capture program. You can set these parameters to modify the length of time that you retain data in the CD table, the amount of time that the Capture program is allowed to lag in processing log records, how often data will be committed, and how often your CD and UOW tables are pruned. These modifications must be done manually because there are no DB2 DataPropagator processes that update this table after it is created. The Capture program can only read your modifications during its start processing; therefore, you should stop and start the Capture program if you want your modifications to take effect.

Table 46 provides a brief description of the tuning parameters table columns.

Table 46. Tuning Parameters Table Columns

Column name	Description
RETENTION_LIMIT	The age limit, in minutes, for keeping CD table rows. This value is used with the SYNCHPOINT column of the pruning control table to determine the pruning limit. Any change data rows older than this value are pruned, even if they have not been copied to all targets. Transactions rejected after update conflict detection will have their changes pruned by RETENTION_LIMIT aging, not by normal pruning. The default value is 10 080.
LAG_LIMIT	The amount of time, in minutes, that the Capture program is allowed to lag in processing log records before it shuts itself down. During periods of high update frequency, full refreshes can be more economical than updates. The default value is 10 080.

Table 46. Tuning Parameters Table Columns (continued)

Column name	Description
COMMIT_INTERVAL	The Capture program commit threshold, in seconds, for any inserts, updates, or deletes to the global UOW table and any pruning control tables. The default value is 30. On systems that do not support ISOLATION (UR), this value should be less than the DB2 lock timeout value to prevent Apply program instances from timing out due to contention with the Capture program.
PRUNE_INTERVAL	The Capture program commit threshold, in seconds, for automatic or manual pruning of CD and UOW rows that are no longer needed. The default value is 300. Values set lower save space, but increase processing costs. Values set higher require more CD and UOW table space, but decrease processing costs. There is no effect on table space or processing cost when the NOPRUNE option is selected.

Capture enqueue table (VM and VSE specific)

Important: Do *not* use SQL to update this table. Altering this table inappropriately can cause unexpected results and loss of data.

ASN.IBMSNAP_CCPENQ

The Capture enqueue table is used in the VM and VSE environments only. This table is used to ensure that there is only one Capture program running per database.

Table 47 provides a list and a brief description of the Capture enqueue table column.

Table 47. Capture Enqueue Table Column

Column name	Description
LOCKNAME	Unique name of the resource for this database.

Warm start table

Important: Do *not* use SQL to update this table. Altering this table inappropriately can cause unexpected results and loss of data.

ASN.IBMSNAP_WARM_START

This table is created in the same database as the register table and contains information that enables the Capture program to restart from the last log or journal record read. Use the information in this table to avoid a full refresh of your system.

The following three tables show platform-specific layouts of the warm start table. The first table shows the layout for all platforms other than VM/VSE and AS/400, the second table shows the VM/VSE layout, and the last table shows the AS/400 layout.

Table 48. Warm Start Table Columns

Column name	Description
SEQ	The last captured sequence number from the log or journal record. Used for quickly restarting following a shutdown or failure.
AUTHTKN	The DB2 token for the unit of work associated with the SEQ log or journal record.
AUTHID	The DB2 authorization ID for the unit of work associated with the SEQ log or journal record.
CAPTURED	A flag indicating whether or not this unit of work was captured. **Y** This unit of work was captured. **N** This unit of work was not captured.
UOWTIME	The MVS time of day, or Windows NT, HP-UX, Sun Solaris, OS/2, and AIX Coordinated Universal Time (UTC) clock indicating when the unit of work associated with the SEQ position was captured (source server timestamp).

Table 49. Warm Start Table Columns for VM and VSE Platforms

Column name	Description
SEQ	The last captured sequence number from the log or journal record. Used for quickly restarting following a shutdown or failure.
UOWID	The unit-of-recovery ID from the log record header for this unit of work.
AUTHID	The DB2 authorization ID for the unit of work associated with the SEQ log or journal record.

Table 49. Warm Start Table Columns for VM and VSE Platforms (continued)

Column name	Description
CAPTURED	A flag indicating whether or not this unit of work was captured. **Y** This unit of work was captured. **N** This unit of work was not captured.
UOWTIME	The VSE and VM time-of-day clock indicating when the unit of work associated with the SEQ log or journal record was captured (source server timestamp).

This AS/400 table is used to determine the starting time of the **RCVJRNE** (Receive Journal Entry) command. A row is inserted into the warm start table for each journal that is used by a replication source or a group of replication sources.

Table 50 provides a brief description of the warm start table columns for the AS/400 platform.

Table 50. Warm Start Table Columns for AS/400 Platform

Column name	Description
JRN_LIB	The library name of the journal.
JRN_NAME	The name of the journal used by a source table. An asterisk followed by nine blanks in this column means that the source table is not currently journaled in a journal. Therefore, it is not possible to capture data for this source table.
JRN_JOB_NUMBER	The job number of the current job for a particular journal. If the journal is not active, this column contains the job number of the last job that was processed.
LOGMARKER	The timestamp of the last processed journal entry.
UID	A unique number that is used as a prefix for the contents of the IBMSNAP_UOWID column located in the ASN.IBMSNAP_UOW tables.
SEQNBR	The sequence number of the last processed journal entry.

Critical section table

Important: Do *not* use SQL to update this table. Altering this table inappropriately can cause unexpected results and loss of data.

ASN.IBMSNAP_CRITSEC

This table is used to prevent circular replication in an update-anywhere scenario.

Table 51 provides a brief description of the critical section table column.

Table 51. Critical Section Table Column

Column name	Description
APPLY_QUAL	A unique identifier for a group of subscription sets. This value is supplied by the user when defining a subscription set. Each Apply process is started with an APPLY_QUAL. This value is used during update-anywhere replication to prevent circular propagation of the changes made by the Apply program. See "Subscription set table" on page 327 for more details.

Prune lock table

Important: Do *not* use SQL to update this table. Altering this table inappropriately can cause unexpected results and loss of data.

ASN.IBMSNAP_PRUNE_LOCK

The prune lock table is used to serialize the access of staging tables during a cold start or retention limit pruning. There are no rows in this table. The Capture and Apply programs use this table as a logical lock to serialize their operations during these critical phases.

Trace table

Important: Do *not* use SQL to update this table. Altering this table inappropriately can cause unexpected results and loss of data.

ASN.IBMSNAP_TRACE

This table contains audit trail information for the Capture program. Everything that is done by the Capture program is recorded in this table, which makes it one of the best places to look if a problem with the Capture program occurs. If you issue a cold start, all of the trace table's entries are deleted, so you might want to save a copy of this table before you issue a cold start command.

The following two tables show platform-specific layouts of the trace table. Table 52 on page 322 shows the layout for all platforms other than AS/400, and Table 53 on page 322 shows the AS/400 layout.

Table 52. Trace Table Columns

Column name	Description
OPERATION	The type of Capture program operation, for example, initialization, capture, or error condition.
TRACE_TIME	The time that a row is inserted into the trace table.
DESCRIPTION	The message ID followed by the message text. The message can be informational or error. This column contains English-only text. See “Chapter 23. Capture and Apply messages” on page 371 for a detailed description of the correlating message ID in the DESCRIPTION column.

Table 53. Trace Table Columns for AS/400

Column name	Description
OPERATION	The type of Capture program operation, for example, initialization, capture, or error condition.
TRACE_TIME	The time that a row is inserted into the trace table.
JOB_NAME	The fully qualified name of the job that wrote this trace entry. • position 1-10: ’QDPRCTL5’ or the journal job name • position 11-20: The ID of the user who started the Capture program • position 21-26: The job number
JOB_STR_TIME	The starting time of the job named in the JOB_NAME column.
DESCRIPTION	The message ID followed by the message text. The message ID is the first 7 characters of the DESCRIPTION column. The message text starts at the 9th position of the DESCRIPTION column.

Apply-qualifier-cross-reference table (AS/400 specific)

Important: Do *not* use SQL to update this table. Altering this table inappropriately can cause unexpected results and loss of data.

ASN.IBMSNAP_AUTHTKN

The Apply-qualifier-cross-reference table is used in the AS/400 environment only. This table is used during update-anywhere replication to keep track of the jobs run by a particular Apply qualifier.

Table 54 on page 323 provides a brief description of the apply-qualifier-cross-reference table columns.

Table 54. Apply-Qualifier-Cross-Reference Table Columns

Column name	Description
APPLY_QUAL	A unique identifier for a group of subscription sets. This value is supplied by the user when defining a subscription set. Each Apply process is started with an APPLY_QUAL. This value is used during update-anywhere replication to prevent circular propagation of the changes made by the Apply program. See "Subscription set table" on page 327 for more details.
IBMSNAP_AUTHTKN	The job name associated with a transaction. Capture for AS/400 matches this column with the name of the job that issued the transaction to determine if a transaction is issued by either the Apply program or a user application. If the names match, then Capture for AS/400 copies the APPLY_QUAL column to the UOW row. If the names do not match, then Capture for AS/400 sets the APPLY_QUAL column of the UOW row blank. This column is not automatically copied to other tables; you must select it and copy it as a user data column.
IBMSNAP_LOGMARKER	The approximate commit time at the source server.

Register synchronization table

Important: Do *not* use SQL to update this table. Altering this table inappropriately can cause unexpected results and loss of data.

ASN.IBMSNAP_REG_SYNCH

There is an update trigger on this table that initiates an update of the SYNCHPOINT value for all the rows in the register table when the Apply program fetches data from a non-IBM data source.

Table 55 provides a brief description of the register synchronization table column.

Table 55. Register Synchronization Table Column

Column name	Description
TRIGGER_ME	**Y** A trigger was initiated to update the SYNCHPOINT value for all rows in the register table.

Unit-of-work table

Important: Do *not* use SQL to update this table. Altering this table inappropriately can cause unexpected results and loss of data.

ASN.IBMSNAP_UOW

The unit-of-work (UOW) table ensures data integrity by recording transactions that were committed at the source server. The Apply program joins the UOW and CD table based on matching IBMSNAP_UOWID values to ensure that only committed changes are being copied. The results are ordered by the log or journal record sequence number of the change in the CD table within the committed units of work. If you issue a cold start, all of this table's entries are deleted, so you might want to save a copy of this table before you issue a cold start command.

The Capture program requires that there is one UOW table for each source server. The Capture program inserts one new row into this table for every log or journal record that commits changes to replication sources. The Capture program also prunes the UOW table based on information inserted into the pruning control table by the Apply program.

Table 56 provides a brief description of the UOW table columns.

Table 56. UOW Table Columns

Column name	Description
IBMSNAP_UOWID	The unit-of-work identifier from the log record header for this unit of work.
IBMSNAP_COMMITSEQ	The log record sequence number of the captured commit statement.
IBMSNAP_LOGMARKER	The approximate commit time at the source server.
IBMSNAP_AUTHTKN	The authorization token associated with the transaction. This ID is useful for database auditing. For DB2 for OS/390, this column is the correlation ID. For DB2 for AS/400, this column is the job name of the job that caused a transaction. This column is not automatically copied to other tables; you must select it and copy it as a user data column. This user data column should be a CCD target type.

Table 56. UOW Table Columns (continued)

Column name	**Description**
IBMSNAP_AUTHID	The authorization ID associated with the transaction. It is useful for database auditing. For DB2 for OS/390, this column is the primary authorization ID. For DB2 for AS/400, this column has the name of the user profile ID under which the application that caused the transaction ran. This column holds a 10-character ID padded with blanks. This column is not automatically copied to other tables; you must select it and copy it as a user data column. This user data column should be a CCD target type.
IBMSNAP_REJ_CODE	This value is set only during update-anywhere replication if conflict detection is specified as standard or advanced when you define your replication source. **0** A transaction with no known conflict. **1** A transaction that contains a conflict where the same row in the source and replica tables have a change that was not propagated. When a conflict occurs, the transaction will be reversed at the replica table. **2** A cascade-rejection of a transaction dependent on a prior transaction having at least one same-row conflict. When a conflict occurs, the transaction will be reversed at the replica table. **3** A transaction that contains at least one referential-integrity constraint violation. Because this transaction violates the referential constraints defined on the source table, the Apply program will mark this subscription set as failed. Updates cannot be copied until the referential integrity definitions are corrected. **4** For a cascade-rejection of a transaction dependent on a prior transaction having at least one constraint conflict.
IBMSNAP_APPLY_QUAL	This column prevents circular replication during update-anywhere processing. It remains blank for local updates, but contains the name of the associated Apply program for updates that are made by the Apply program for an update-anywhere subscription set. The Capture program derives this value from the critical section table.

Change data table

Important: Do *not* use SQL to update this table. Altering this table inappropriately can cause unexpected results and loss of data.

CD

Change data (CD) tables record all changes made to a replication source. Committed, uncommitted, and incomplete changes are inserted as rows into the CD table. The CD table works with the UOW table to provide commit information. (See "Consistent-change-data (CCD) tables" on page 16 for more information.) Pruning of the CD table rows is coordinated by the pruning control table. (See "Pruning control table" on page 314 for more information.)

CD tables are automatically created when you define a replication source. For each replication source that is enabled for data capture, there is one CD table. If you issue a cold start, all of the CD table's entries are deleted.

Manually changing the CD table is not recommended. However, the CD table can be a useful resource for problem determination. Knowing exactly what changes were committed or not committed can help you understand where the Capture program failed.

Recommendation: Although the Control Center automatically creates an index, a unique ascending index is strongly recommended for the IBMSNAP_UOWID and IBMSNAP_INTENTSEQ columns.

Table 57 provides a list and a brief description of each of the CD table columns.

Table 57. CD Table Columns

Column name	Description
IBMSNAP_UOWID	Unit-of-work ID for an update. The Apply program uses this column to join the CD table with the UOW table so that only committed changes are replicated.
IBMSNAP_INTENTSEQ	Log or journal record sequence number that uniquely identifies a change. This value is globally ascending.
IBMSNAP_OPERATION	Character value of I, U, or D, indicating an insert, update, or delete record.
DATA1	User column from source table specified by the user when defining replication sources.

Table 57. CD Table Columns (continued)

Column name	Description
AFTER-IMAGE	User column from source table selected by the user when defining a replication source. This column will have the same name, data type, and null attributes as the source column. The after-image column also contains the equivalent source table column value after the change has been made.
BEFORE-IMAGE	User column from source table selected by the user when defining a replication source. This column will have the same name, data type, and null attributes as the source column. The name is the source column prefixed with the BEFORE_IMG_PREFIX value from the register table. This column contains the equivalent source table column value before the change was made.

Tables used at the control server

The control server is the DB2 system that you chose to hold your subscription definitions. The following section provides a brief description of the tables used at the control server and the columns in each table. If you are using the Control Center, these tables, which contain information about your subscription definitions, are automatically created when you define a subscription if they do not already exist.

The subscription set, subscription-targets-member, subscription column, subscription statements, row-replica, and subscription-schema-changes tables contain information about subscriptions. When a new subscription set is defined, the administration tools simultaneously update rows in the subscription set, subscription columns, subscription-targets-member, and subscription statements tables.

The subscription events and Apply trail tables are used by the Apply program to control and audit your data.

Subscription set table

Important: Do *not* use SQL to update this table. Altering this table inappropriately can cause unexpected results and loss of data.

ASN.IBMSNAP_SUBS_SET

Subscription set table

The subscription set table lists all of the subscription sets defined at the control server and identifies the source and target server pairs that are processed as a group. Rows are inserted into this table when you create your subscription definition.

Use this table to identify subscription sets that have been defined.

Table 58 provides a brief description of the subscription set table columns.

Table 58. Subscription Set Table Columns

Column name	**Description**
APPLY_QUAL	Uniquely identifies a group of subscription sets that are processed by the same Apply program process. This user-specified value must be unique for the control server where the subscription set table is located. For update-anywhere, this value must be unique at the control server and at the source server. This value is case-sensitive. You must specify this value when you define a subscription set. See "Part 3. Operations" on page 133 for more details.
SET_NAME	Identifies a group of target tables (subscription-set members) that are processed by the Apply program as a group. This user-specified value must be unique within an Apply qualifier. Changes for subscription members in a set are processed in a single transaction during the Apply program processing cycle.
WHOS_ON_FIRST	The following values are used to control the order of processing in update-anywhere replication scenarios. **F** (first) The target table is the user table or parent replica. The source table is the dependent row-replica that is lower in the hierarchy of replication. F is not used for read-only subscriptions. **S** (second) The source table is the user table, parent replica, or other source. The target table is the dependent row-replica or other copy that is lower in the hierarchy of replication. S is used for all read-only subscriptions.

Table 58. Subscription Set Table Columns (continued)

Column name	Description
ACTIVATE	The following values are flags set by either the Control Center (0 and 1) or by the Apply program (2). **0** The subscription set is deactivated. **1** The subscription set is active indefinitely. **2** The subscription set is used for a one-time-only subscription execution.
SOURCE_SERVER	The RDB name of the source server where the source tables and views are defined.
SOURCE_ALIAS	The DB2 Universal Database alias corresponding to the source server named in the SOURCE_SERVER column.
TARGET_SERVER	The RDB name of the server where the target tables and views are defined.
TARGET_ALIAS	The DB2 Universal Database alias corresponding to the target server named in the TARGET_SERVER column.
STATUS	A value that represents in-progress and completed work status for the Apply program. **-1** A failed execution. **0** A stable definition row. **1** A pending or in-progress execution. Do not modify this definition or any rows related to this subscription set in other control tables. **2** A continuing execution of a single logical subscription set that was divided according to the MAX_SYNCH_MINUTES control column and is being serviced by multiple subscription cycles. Do not modify this row or any row related to this subscription set in other control tables.
LASTRUN	The estimated time that the last subscription set began. The Apply program sets the LASTRUN value each time a subscription set is processed. It is the approximate time at the control server that the Apply program begins processing the subscription set.

Table 58. Subscription Set Table Columns (continued)

Column name	Description
REFRESH_TIMING	Sets the timing between statement executions. **R** The Apply program uses the value in SLEEP_MINUTES to determine replication timing. **E** The Apply program checks the time value in the SUBS_EVENT table to determine replication timing. **B** Indicates that a subscription set has both relative and event timing specifications. Therefore, this subscription set can be eligible for a refresh based on either the timer or event timing criteria.
SLEEP_MINUTES	Specifies the time of inactivity between subscription set processing when REFRESH_TIMING = 'R' or 'B'.
EVENT_NAME	A unique character string used to represent an event. Use this identifier to update the subscription events table when you want to trigger replication for a subscription set.
LASTSUCCESS	The control server timestamp for the beginning of the last successful processing of a subscription set.
SYNCHPOINT	The Apply program uses the SYNCHPOINT value from the global row of the register table at the source server if GLOBAL_RECORD = 'Y'. If data blocking is specified in the subscription set definition, then the SYNCHPOINT value is the log or journal record sequence number of the last change applied during the Apply process.
SYNCHTIME	The Capture program or an external program, such as DataPropagator NonRelational, updates this timestamp whether there are changes to be processed or not. The Apply program uses this value when advanced conflict detection is selected for update-anywhere replication to ensure that the Capture program has captured all outstanding changes for a replication source table.

Table 58. Subscription Set Table Columns (continued)

Column name	Description
MAX_SYNCH_MINUTES	A time-threshold limit to regulate the amount of change data to fetch and apply during a subscription cycle. The Apply program breaks the subscription set processing into mini-cycles based on the IBMSNAP_LOGMARKER column in the UOW or CCD table at the source server and issues a COMMIT at the target server after each successful mini-cycle. The limit is automatically recalculated if the Apply program encounters a resource constraint that makes the set limit unfeasible. MAX_SYNCH_MINUTES values that are less than 1 will be treated the same as a MAX_SYNCH_MINUTES value equal to NULL.
AUX_STMTS	The number of SQL BEFORE and AFTER statements that you define in the subscriptions statements table.
ARCH_LEVEL	The architectural level of the definition contained in the row. This field identifies the rules under which a row was created.

Subscription-targets-member table

Important: Do *not* use SQL to update this table. Altering this table inappropriately can cause unexpected results and loss of data.

ASN.IBMSNAP_SUBS_MEMBR

This table or view contains information about the individual source and target table pairs defined for a subscription set. Rows are automatically inserted into this table when you create a subscription member.

Use this table or view to identify a specific source and target table pair within a subscription set.

Table 59 on page 332 provides a brief description of the subscription-targets-member table columns.

Table 59. Subscription-Targets-Member Table Columns

Column name	Description
APPLY_QUAL	Uniquely identifies a group of subscription sets that are processed by the same Apply program process. This user-specified value must be unique for the control server where the subscription set table is located. For update-anywhere, this value must be unique at the control server and at the source server. This value is case-sensitive. You must specify this value when you define a subscription set.
SET_NAME	Identifies a group of target tables (subscription-set members) that are processed by the Apply program as a group. This user-specified value must be unique within an Apply qualifier. Changes for subscription members in a set are processed in a single transaction during the Apply program processing cycle.
WHOS_ON_FIRST	The following values are used to control the order of processing in update-anywhere replication scenarios. **F** (first) The target table is the user table or parent replica. The source table is the dependent row-replica that is lower in the hierarchy of replication. F is not used for read-only subscriptions. **S** (second) The source table is the user table, parent replica, or other source. The target table is the dependent row-replica or other copy that is lower in the hierarchy of replication. S is used for all read-only subscriptions.
SOURCE_OWNER	The owner of the source table or view.
SOURCE_TABLE	The source from which data is being captured.
SOURCE_VIEW_QUAL	Supports the view of physical tables by matching the similar column in the register table. This value is set to 0 for physical tables that are defined as sources and is greater than 0 for views that are defined as sources. This column is used to support multiple subscriptions for different source views with identical SOURCE_OWNER and SOURCE_TABLE column values.
TARGET_OWNER	A qualifier for a target table or view.
TARGET_TABLE	The target to which data is being applied.

Table 59. Subscription-Targets-Member Table Columns (continued)

Column name	Description
TARGET_CONDENSED	A flag indicating: **Y** For any given primary key, the target table shows only one row. **N** All changes must remain, retaining a complete update history. **A** Valid only for base aggregate or change aggregate tables.
TARGET_COMPLETE	A flag indicating: **Y** The target table contains a row for every primary key value of interest. **N** The target table contains some subset of rows of primary key values.
TARGET_STRUCTURE	The structure of the target table: **1** User table **3** CCD table **4** Point-in-time table **5** Base aggregate table **6** Change aggregate table **7** Replica **8** User copy **9** Row-replica (Microsoft Jet specific)
PREDICATES	Lists the predicates to be placed in a WHERE clause to subset the horizontal fragment maintained in the TARGET_TABLE column. The letter 'A' is a predefined correlation-name for the physical source table used in a correlated subquery. Do not specify an ORDER BY clause because the Apply program can not generate an ORDER BY clause. Aggregate tables require a dummy predicate followed by a GROUP BY clause as a predicate.

Subscription columns table

Important: Do *not* use SQL to update this table. Altering this table inappropriately can cause unexpected results and loss of data.

ASN.IBMSNAP_SUBS_COLS

This table contains information about the columns of the subscription-set members being copied in a subscription set. The subscription columns table contains supplemental information to the subscription-targets-member table.

Rows are automatically inserted or deleted from this table when information at the column level of a source and target table pair is changed.

Use this table if you need information about specific columns in a subscription-set member.

Table 60 provides a brief description of the subscription columns table columns.

Table 60. Subscription Columns Table Columns

Column name	Description
APPLY_QUAL	Identifies the Apply program for the platform instance that will run this subscription set. The value must be unique among all Apply program processes maintaining dependent replicas of a user table or parent replica, and unique among all Apply program processes sharing a common set of control tables. This value is case sensitive. You must specify this value when you define a subscription set.
SET_NAME	Names a subscription set. This value is unique within an Apply qualifier.
WHOS_ON_FIRST	The following values are used to control the order of processing in update-anywhere replication scenarios. **F** (first) The target table is the user table or parent replica. The source table is the dependent row-replica that is lower in the hierarchy of replication. F is not used for read-only subscriptions. **S** (second) The source table is the user table, parent replica, or other source. The target table is the dependent row-replica or other copy that is lower in the hierarchy of replication. S is used for all read-only subscriptions.
TARGET_OWNER	A qualifier for a target table or view.
TARGET_TABLE	The target to which data is being applied.

Table 60. Subscription Columns Table Columns (continued)

Column name	Description
COL_TYPE	A flag indicating: **A** For an after-image column. **B** For a before-image column. **C** For a computed column without a SQL column function reference. **F** For a computed column with a SQL column function reference. **L** A large object (LOB) column. **R** Signifies a relative record number column, provided by the system and used as a primary key column. Used only by DPROPR for AS/400.
TARGET_NAME	The name of the target table or view column. It does not need to match the source column name. Internal CCD column names cannot be renamed. They must match the CD table column names.
IS_KEY	**Y** The column is all or part of the primary key of the target (all condensed copies must have primary keys). **N** The column is not part of a key of the target.
COLNO	The numeric location of the column in the original source, to be preserved relative to other user columns in displays and subscriptions.
EXPRESSION	The source column name or an SQL expression representing the target column.

Subscription statements table

Important: Do *not* use SQL to update this table. Altering this table inappropriately can cause unexpected results and loss of data.

ASN.IBMSNAP_SUBS_STMTS

This table contains the user-defined SQL statements or stored procedure calls that will be executed before or after each subscription-set processing cycle. Execute immediately (EI) statements or stored procedures can be executed at the source or target server only. This table is populated when you define a subscription that uses SQL statements or stored procedure calls.

Subscription statements table

Table 61 provides a brief description of the subscription statements table columns.

Table 61. Subscription Statements Table Columns

Column name	Description
APPLY_QUAL	Uniquely identifies a group of subscription sets that are processed by the same Apply program process. This user-specified value must be unique for the control server where the subscription set table is located. For update-anywhere, this value must be unique at the control server and at the source server. This value is case-sensitive. You must specify this value when you define a subscription set.
SET_NAME	Identifies a group of target tables (subscription-set members) that are processed by the Apply program as a group. This user-specified value must be unique within an Apply qualifier. Changes for subscription-set members in a set are processed in a single transaction during the Apply program processing cycle.
WHOS_ON_FIRST	The following values are used to control the order of processing in update-anywhere replication scenarios. **F** (first) The target table is the user table or parent replica. The source table is the dependent row-replica that is lower in the hierarchy of replication. F is not used for read-only subscriptions. **S** (second) The source table is the user table, parent replica, or other source. The target table is the dependent row-replica or other copy that is lower in the hierarchy of replication. S is used for all read-only subscriptions.
BEFORE_OR_AFTER	A value indicating: **A** The statement is executed at the target server after all of the answer set rows are applied. **B** The statement is executed at the target server before any of the answer set rows are applied. **S** The statement is executed at the source server before opening the answer set cursors. **G** The statement is executed at the source server before opening any cursors to either fetch answer set rows or fetch registration details.

Table 61. Subscription Statements Table Columns (continued)

Column name	Description
STMT_NUMBER	Defines the relative order of execution within the scope of BEFORE_OR_AFTER.
EI_OR_CALL	A value indicating: **E** The SQL statement should be run as an EXEC SQL EXECUTE IMMEDIATE. **C** The SQL statement contains a stored procedure name to run as an EXEC SQL CALL.
SQL_STMT	One of the following values: **Statement** The SQL statement should run as an EXEC SQL EXECUTE IMMEDIATE statement, if EI_OR_CALL = ’E’. **Procedure** The 8-byte name of an SQL-stored procedure, without parameters or the CALL keyword, that runs as an EXEC SQL CALL statement if EI_OR_CALL = ’C’.
ACCEPT_SQLSTATES	One to ten 5-byte SQLSTATE values that you specified when you defined the subscription. These non-zero values are accepted by the Apply program as a successful execution. Any other value will cause a failed execution.

Row-replica-target-list table (Microsoft Jet specific)

Important: Do *not* use SQL to update this table. Altering this table inappropriately can cause unexpected results and loss of data.

ASN.IBMSNAP_SUBS_TGTS

This table is necessary to identify when a member has been deleted from a subscription set for a Microsoft Jet database target, so that the row-replica table can be deleted from the Microsoft Jet database. The row-replica-target-list table allows DataPropagator for Microsoft Jet to maintain a list of known row-replica tables in a stable DB2 or DataJoiner database. DataPropagator for Microsoft Jet will use this information during schema analysis to determine if any row-replica tables should be deleted because the corresponding subscription-set member was dropped since the last synchronization.

Row-replica-target-list table

Table 62 provides a brief description of the row-replica-target-list table columns.

Table 62. Row-Replica-Target-List Table Columns

Column name	Description
APPLY_QUAL	Uniquely identifies a group of subscription sets that are processed by the same Apply program process. This user-specified value must be unique for the control server where the subscription set table is located. For update-anywhere, this value must be unique at the control server and at the source server. This value is case-sensitive. You must specify this value when you define a subscription-set.
SET_NAME	Identifies a group of target tables (subscription-set members) that are processed by the Apply program as a group. This user-specified value must be unique within an Apply qualifier.
WHOS_ON_FIRST	The following values are used to control the order of processing in update-anywhere replication scenarios. **F** (first) The target table is the user table or parent replica. The source table is the dependent row-replica that is lower in the hierarchy of replication. F is not used for read-only subscriptions. **S** (second) The source table is the user table, parent replica, or other source. The target table is the dependent row-replica or other copy that is lower in the hierarchy of replication. S is used for all read-only subscriptions.
TARGET_OWNER	A qualifier for a target table or view.
TARGET_TABLE	The target to which data is being applied.
LAST_POSTED	This column is the timestamp of when this row was inserted into the table. This column is for informational purposes only.

Subscription-schema-changes table (Microsoft Jet specific)

Important: Do *not* use SQL to update this table. Altering this table inappropriately can cause unexpected results and loss of data.

ASN.IBMSNAP_SCHEMA_CHG

This table allows DataPropagator for Microsoft Jet to quickly determine if some relevant schema change has occurred since its last synchronization. If a modification is made, DataPropagator for Microsoft Jet will drive a thorough analysis of the replication control information. DataPropagator for Microsoft Jet will then create or drop row-replica tables, or columns in row-replica tables, to automatically converge the Microsoft Jet database schema with the schema described by the replication control information. This schema convergence occurs before data synchronization, so that new columns and new tables are copied.

Table 63 provides a brief description of the subscription-schema-changes table columns.

Table 63. Subscription-Schema-Changes Table Columns

Column name	Description
APPLY_QUAL	Uniquely identifies a group of subscription sets that are processed by the same Apply program process. This user-specified value must be unique for the control server where the subscription set table is located. For update-anywhere, this value must be unique at the control server and at the source server. This value is case-sensitive. You must specify this value when you define a subscription set.
SET_NAME	Identifies a group of target tables (subscription-set members) that are processed by the Apply program as a group. This user-specified value must be unique within an Apply qualifier. Changes for subscription members in a set are processed in a single transaction at the target site during the Apply program processing cycle.
LAST_CHANGED	This column is the timestamp of when this row was last changed in this table. This column is for informational purposes only.

Subscription events table

This table contains information that you can update by using SQL.

ASN.IBMSNAP_SUBS_EVENT

This table contains information about the event triggers that are copied in a subscription set. The subscription events table contains event names and timestamps associated with the event names. You insert a row into this table when you create a new event to execute the start of an Apply process. See “Event timing” on page 107.

Subscription events table

Table 64 provides a brief description of the subscription events table columns.

Table 64. Subscription Events Table Columns

Column name	Description
EVENT_NAME	When you replicate events between systems, this column contains a globally unique character string in a global name space configuration. Otherwise, this column contains a control server unique character string.
EVENT_TIME	A control server timestamp of a current or future posting time. User applications signalling replication events provide the values in this column.
END_OF_PERIOD	A source server timestamp value that acts like an upper boundary. Any transactions that are committed after this period are not replicated, until a later event is posted. The only way to prevent eligible change data from replicating during a subscription cycle is to make sure that the value in this column is less than the CURRENT TIMESTAMP value at the source server.

A unique index on EVENT_NAME and EVENT_TIME is created automatically by either the Control Center or through DPCNTL.

Apply trail table

Important: Do *not* use SQL to update this table. Altering this table inappropriately can cause unexpected results and loss of data.

ASN.IBMSNAP_APPLYTRAIL

The Apply trail table contains audit trail information for the Apply program. This table records a history of updates performed against subscriptions. It is a repository of diagnostic and performance statistics. These facts make the apply trail table one of the best places to look if a problem occurs with the Apply program. Because this table is not automatically pruned, it is up to you to do so.

Table 65 provides a brief description of the Apply trail table columns.

Table 65. Apply Trail Table Columns

Column name	Description
APPLY_QUAL	Uniquely identifies a group of subscription sets that are processed by the same Apply program process. This user-specified value must be unique for the control server where the subscription set table is located. For update-anywhere, this value must be unique at the control server and at the source server. This value is case-sensitive. You must specify this value when you define a subscription set.
SET_NAME	Identifies a group of target tables (subscription-set members) that are processed by the Apply program as a group. This user-specified value must be unique within an Apply qualifier. Changes for subscription members in a set are processed in a single transaction at the target site during the Apply program processing cycle.
WHOS_ON_FIRST	The following values are used to control the order of processing in update-anywhere replication scenarios. **F** (first) The target table is the user table or parent replica. The source table is the dependent row-replica that is lower in the hierarchy of replication. F is not used for read-only subscriptions. **S** (second) The source table is the user table, parent replica, or other source. The target table is the dependent row-replica or other copy that is lower in the hierarchy of replication. S is used for all read-only subscriptions.

Table 65. Apply Trail Table Columns (continued)

Column name	Description
ASNLOAD	Contains one of the following values: **Y** Indicates that the Apply program was started and that the LOADXit parameter and the ASNLOAD exit routine were called to perform a full refresh on a subscription set. **N** Indicates that the ASNLOAD exit routine was not called because either a full refresh was not needed or the Apply program was not started with the LOADXit parameter. **NULL** Indicates that an Apply program error occurred before the Apply program could determine whether the ASNLOAD exit routine should be called.
MASS_DELETE	A mass delete is always triggered during a full refresh. The following are values for this column: **Y** Indicates that a full refresh was done for a subscription set. **N** Indicates that a full refresh was not done for a subscription set. **NULL** Indicates that an error occurred before the Apply program could determine whether or not a full refresh was needed.
EFFECTIVE_MEMBERS	The number of subscription-set members that are changed during an Apply cycle, either by a full refresh or by the replication of inserts, updates, and deletes. This number ranges between zero and the number of defined subscription-set members.
SET_INSERTED	The total number of rows inserted into subscription-set members during the subscription cycle.
SET_DELETED	The total number of rows deleted from subscription-set members during the subscription cycle.
SET_UPDATED	The total number of rows updated in subscription-set members during the subscription cycle.

Table 65. Apply Trail Table Columns (continued)

Column name	Description
SET_REWORKED	The Apply program will rework changes under the following conditions: • If an insert fails because the row already exists in the target table, the Apply program converts the insert to an update of the existing row. • If the update fails because the row does not exist in the target table, the Apply program converts the update to an insert.
SET_REJECTED_TRXS	The total number of transactions rejected due to an update-anywhere conflict. This column is used only for update-anywhere subscription sets where conflict detection has been defined as standard or advanced.
STATUS	A value that represents in-progress and completed work status for the Apply program. **-1** A failed execution. **0** A stable definition row. **1** A pending or in-progress execution. Do not modify this definition or any rows related to this subscription set in other control tables. **2** A continuing execution of a single logical subscription that was divided according to the MAX_SYNCH_MINUTES control column and is being serviced by multiple subscription cycles. Do not modify this row or any row related to this subscription set in other control tables.
LASTRUN	The estimated time that the last subscription began. The Apply program sets the LASTRUN value each time a subscription set is processed. It is the approximate time at the control server that the Apply program begins processing the subscription set.
LASTSUCCESS	The control server timestamp for the beginning of the last successful processing of a subscription set.
SYNCHPOINT	The Apply program uses the SYNCHPOINT value from the global row of the register table at the source server if GLOBAL_RECORD = ’Y’. If data blocking is specified in the subscription definition, then the SYNCHPOINT value is the log or journal record sequence number of the last change applied during the Apply process.

Table 65. Apply Trail Table Columns (continued)

Column name	Description
SYNCHTIME	The Capture program or an external program, such as DataPropagator NonRelational, updates this timestamp whether there are changes to be processed or not. The Apply program uses this value when advanced conflict detection is selected for update-anywhere replication to ensure that the Capture program has captured all outstanding changes for a replication source table.
SOURCE_SERVER	The RDB name of DB2 for OS/390, DB2 for VSE, and DB2 for VM where the source tables and views are defined.
SOURCE_ALIAS	The DB2 Universal Database alias corresponding to the source server named in the SOURCE_SERVER column.
SOURCE_OWNER	The owner of the source table or view.
SOURCE_TABLE	The source from which data is being captured.
SOURCE_VIEW_QUAL	Supports the view of physical tables by matching the similar column in the register table. This value is set to 0 for physical tables that are defined as sources and is greater than 0 for views that are defined as source. This column is used to support multiple subscriptions for different source views with identical SOURCE_OWNER and SOURCE_TABLE column values.
TARGET_SERVER	The RDB name of the target server where the target tables and views are defined.
TARGET_ALIAS	The DB2 Universal Database alias corresponding to the target server named in the TARGET_SERVER column.
TARGET_OWNER	A qualifier for a target table or view.
TARGET_TABLE	The target to which data is being applied.
SQLSTATE	The SQL state code for a failed execution. Otherwise, NULL.
SQLCODE	The SQL error code for a failed execution. Otherwise, NULL.
SQLERRP	The database product identifier of the server where an SQL error occurred that caused a failed execution. Otherwise, NULL.
SQLERRM	The SQL error information for a failed execution. Otherwise, NULL.
APPERRM	The Apply error message ID and text for a failed execution. Refer to “Chapter 23. Capture and Apply messages” on page 371 for detailed message information. Otherwise, NULL.

Apply job table (AS/400 specific)

Important: Do *not* use SQL to update this table. Altering this table inappropriately can cause unexpected results and loss of data.

IBMSNAP_APPLY_JOB

This AS/400-specific table is used to guarantee a unique APPLY_QUAL value for all instances of the Apply program running at the control server. A row will be added to this table every time an instance of the Apply program is started. If you start a new instance of the Apply program with an APPLY_QUAL value that already exists, your start command will fail.

Table 66 provides a brief description of the Apply job table columns.

Table 66. Apply Job Table Columns

Column name	**Description**
APPLY_QUAL	A unique identifier for a group of subscription sets. This value is supplied by the user when defining a subscription set. Each instance of the Apply program is started with an APPLY_QUAL. This value is used during update-anywhere replication to prevent circular replication of the changes made by the Apply program. See the subscription set table on page “Subscription set table” on page 327 for more details.
CONTROL_SERVER	Name of the RDB where the control tables and view are defined.
USER_NAME	Name of the user who started a new instance of the Apply program
JOB_NAME	The fully qualified name of the job that wrote this trace entry: • position 1-10: ’QDPRCTL5’ or the journal job name • position 11-20: The ID of the user who started the Capture program • position 21-26: The job number
JOB_NUMBER	The job number of the current job for a particular journal. If the journal is not active, this column contains the job number of the last job that was processed.

Tables used at the target server

The following section provides a brief description of the Apply program target tables used at the target server and the columns in each table.

User copy table

Important: Do *not* use SQL to update this table. Altering this table inappropriately can cause unexpected results and loss of data.

userid.target_table

The user copy table is identical to the point-in-time target table with the exception of the IBMSNAP_LOGMARKER column, which is not included in the user copy table.

Except for subsetting and data enhancement, a user copy table reflects a valid state of the source table, but not necessarily the most current state. References to user copy tables (or any other type of target table) reduce the risk of contention problems that results from a high volume of direct access to the source tables. Accessing local user copy tables is much faster than using the network to access remote source tables for each query.

Table 67 provides a brief description of the user copy table columns.

Table 67. User Copy Table Columns

Column name	Description
user key columns	The primary key of the target table, although it is not necessarily a component of the primary key of the source table. You can use predicates to prevent a NULL value from being assigned to the key fields of any copies.
user nonkey columns	The nonkey data columns from the source table or view. The columns from the source table do not need to match these columns, but the data types must match.
user computed columns	User-defined columns that are derived from SQL expressions. You can use computed columns with SQL functions to convert source data types to different target data types.

Point-in-time table

Important: Do *not* use SQL to update this table. Altering this table inappropriately can cause unexpected results and loss of data.

userid.target_table

The point-in-time table is similar to the user copy table, but contains an additional system column (IBMSNAP_LOGMARKER) containing the approximate timestamp of when the particular row was inserted or updated at the source system. Otherwise, a point-in-time table is much like a past image of the source table. Point-in-time copies reflect a valid state of the source table, but not necessarily the most current state.

Refer to the IBMSNAP_LOGMARKER column for a commit time, so you will know the point in time that your copy of the source table resembles.

Table 68 provides a brief description of the point-in-time table columns.

Table 68. Point-in-Time Table Columns

Column name	**Description**
user key columns	The primary key of the target table, although it is not necessarily a component of the primary key of the source table. You can use predicates to prevent a NULL value from being assigned to the key fields of any copies.
user nonkey columns	The nonkey data columns from the source table or view. The columns from the source table do not need to match these columns, but the data types must match.
user computed columns	User-defined columns that are derived from SQL expressions. You can use computed columns with SQL functions to convert source data types to different target data types.
IBMSNAP_LOGMARKER	The approximate commit time at the source server. This column is NULL following a full refresh.

Consistent-change-data table

This table contains information that you can update by using SQL.

userid.target_table

CCD tables are staging tables that contain committed change data. Maintaining CCD tables requires updating the CCD_OLD_SYNCHPOINT and SYNCHPOINT columns of the register table.

The CCD table can be:

- A staging table maintained by one Apply program

  The result of a join between the CD and UOW tables can be stored here, so that you perform the join step only once for replicating changes to multiple targets. The CCD table can be maintained on a remote system. The advantage of maintaining your CCD table remotely is that you reduce the work load on your source. You replicate a set of changes from the original source to the CCD table only once. The CCD table then acts as the source and maintains all changes. Later, the changes to the CCD table will be updated to the original source.

- External source table for nonrelational and multivendor data

  External programs can create CCD tables to be used by DB2 DataPropagator as replication sources. An example is DataPropagator NonRelational, which captures IMS changes and maintains a CCD table so that the copies of IMS data can be re-created in a relational database.

For CCD tables:

- The Capture program does not insert data into CCD tables and does not prune them. Instead, your application requirements should determine the history retention period for CCD tables (described in “Staging data” on page 75). Therefore, pruning of CCD tables is not automatic by default, but can be easily automated using an SQL statement to be processed after the subscription cycle.
- For condensed CCD tables (CCD_CONDENSED = ’Y’), a unique index is required for user-data primary-key columns to maintain the CCD table.
- Noncondensed CCD tables can contain duplicate rows if changes are reapplied after a failure during the previous copy operation.
- An external CCD table is an alternate source for the original user table. The user table does not include computed columns; therefore, computed columns should not be included in the CCD subscriptions.
- If an external program, other than the Apply program, maintains the external CCD table, the external program must initialize, maintain, and supply the correct values for the control columns.
- Before-image user data columns must be nullable and therefore cannot be part of a primary key for a condensed CCD table.
- Null attributes of the after-image user data columns should match the null attributes of the source.
- Views of change data tables can be included in view replication sources.
- Views that are defined as replication sources can refer only to CCD tables that are complete and condensed.

The originally captured operation code in the IBMSNAP_OPERATION column and the sequence numbers IBMSNAP_INTENTSEQ and IBMSNAP_COMMITSEQ are included in CCD tables. For condensed CCD tables, only the latest values are kept for each row. The copy operation in IBMSNAP_OPERATION is an insert, update, or delete. The codes are:

I	Insert
U	Update

D Delete

Special cases for condensed CCD tables:

- Because condensed CCD tables have a unique index constraint, inserts to a row with a key that already exist will fail. The insert becomes an update.
- Updates to rows that do not exist with a key will fail. The update becomes an insert.
- A delete is always handled as an update and the row remains in the CCD table.

Table 69 provides a brief description of the CCD table columns.

Table 69. CCD Table Columns

Column name	**Description**
IBMSNAP_INTENTSEQ	Log or journal record sequence number that uniquely identifies a change. This value is globally ascending.
IBMSNAP_OPERATION	Character value of I, U, or D, indicating an insert, update, or delete record.
IBMSNAP_COMMITSEQ	The log record sequence number of the captured commit statement. This value groups inserts, updates, and deletes by original source transactions.
IBMSNAP_LOGMARKER	The approximate commit time at the source server.
user key columns	The primary key of the target table, although it is not necessarily a component of the primary key of the source table. You can use predicates to prevent a NULL value from being assigned to the key fields of any copies.
user nonkey columns	The nonkey data columns from the source table. The columns from the source table do not need to match these columns, but the data types must match.
user computed columns	User-defined columns that are derived from SQL expressions. You can use computed columns with SQL functions to convert source data types to different target data types.

Replica table

This table contains information that you can update by using SQL.

userid.target_table

The replica must have the same primary key as the source table. Because of these similarities, the replica table can be used as a source table for further subscription sets, making the target server a source server as well. Converting a target table into a source table is done automatically when you define a replica target type and specify the CHANGE DATA CAPTURE attribute. See "Defining replication subscriptions for update-anywhere replication" on page 98 for more information.

Table 70 provides a brief description of the replica table columns.

Table 70. Replica Table Columns

Column name	Description
user key columns	The primary key of the target table, although it is not necessarily a component of the primary key of the source table. You can use predicates to prevent a NULL value from being assigned to the key fields of any copies.
user nonkey columns	The nonkey data columns from the source table. The columns from the source table do not need to match these columns, but the data types must match.

Base aggregate table

Important: Do *not* use SQL to update this table. Altering this table inappropriately can cause unexpected results and loss of data.

userid.target_table

Base aggregate tables are target tables that contain data aggregated from a source table. Functions are performed on data located at the source table, and the result of the function is inserted as a row in the base aggregate table.

For base aggregate tables:
- Before-image user data columns must be nullable.
- If the computation will never generate a null value, then the computed columns should be NOT NULL.
- Null attributes of the after-image user data columns should match null attributes of the source, except for primary key columns, which should always be NOT NULL.

Table 71 provides a brief description of the base aggregate table columns.

Table 71. Base Aggregate Table Columns

Column name	Description
user columns	Columns computed from the source table.
IBMSNAP_LLOGMARKER	The current source server timestamp at the time of refresh.

Table 71. Base Aggregate Table Columns (continued)

Column name	Description
IBMSNAP_HLOGMARKER	The current source server timestamp at the time of refresh.

Change aggregate table

Important: Do *not* use SQL to update this table. Altering this table inappropriately can cause unexpected results and loss of data.

userid.target_table

A change aggregate table is a target table that contains data aggregations based on changes from a source table. This table is similar to the base aggregate table, except that the functions being performed at the source table are done only for changes that occur during a specific time interval. The results of these functions are inserted as rows into the change aggregate table. Before-image user data columns must be nullable in change aggregate tables.

Table 72 provides a brief description of the change aggregate table columns.

Table 72. Change Aggregate Table Columns

Column name	Description
user key columns	The primary key of the target table, although it is not necessarily a component of the primary key of the source table. You can use predicates to prevent a NULL value from being assigned to the key fields of any copies.
user nonkey columns	The nonkey data columns from the source table. The columns from the source table do not need to match these columns, but the data types must match.
user computed columns	User-defined columns that are derived from SQL expressions. You can use computed columns with SQL functions to covert source data types to different target data types.
IBMSNAP_LLOGMARKER	The oldest IBMSNAP_LOGMARKER or IBMSNAP_LLOGMARKER value present in the (CD+UOW) or CCD table rows being aggregated.
IBMSNAP_HLOGMARKER	The youngest IBMSNAP_LOGMARKER or IBMSNAP_HLOGMARKER value present in the (CD+UOW) or CCD table rows being aggregated.

Row-replica table (Microsoft Jet specific)

This table contains information that you can update by using SQL.

userid.target_table

This table is an update-anywhere replica table maintained by DataPropagator for Microsoft Jet. Conflicts are detected row by row, not transaction by transaction, as they are for replica tables. Row-replica is the only type of target table supported by DataPropagator for Microsoft Jet. The source table can be a DB2, Oracle, Sybase, Informix, or Microsoft SQL Server user table, or a DB2 replica. The source can also be a view of a DB2 user table or replica table, including a join view.

Table 73 provides a brief description of the row-replica table columns.

Table 73. Row-Replica Table Columns

Column name	Description
user key columns	The primary key of the target table, although it is not necessarily a component of the primary key of the source table. You can use predicates to prevent a NULL value from being assigned to the key fields of any copies.
user nonkey columns	The nonkey data columns from the source table. The columns from the source table do not need to match these columns, but the data types must match.

Conflict table (Microsoft Jet specific)

Do *not* use SQL to update this table. Altering this table inappropriately can cause unexpected results and loss of data.

IBMSNAP_<*target name*>**_CONFLICT**

This table is a conflict table for tracking synchronization conflicts and errors. This Microsoft Jet database control table mimics Microsoft's conflict tables. This table contains the conflict loser's row data. The columns are the same as the corresponding row-replica table. This table can have more than one row. The conflict table is created along with the row-replica table in the Microsoft Jet database and dropped when the row-replica table is dropped.

Table 74 provides a brief description of the conflict table columns.

Table 74. Conflict Table Columns

Column name	Description
target name	The corresponding row-replica table's name.
column names of row-replica	A list of column names found in the corresponding row-replica table.

Error information table (Microsoft Jet specific)

Do *not* use SQL to update this table. Altering this table inappropriately can cause unexpected results and loss of data.

IBMSNAP_ERROR_INFO

This table identifies the row-replica table and row that caused the error. This table can have more than one row. The error information table is created along with the Microsoft Jet database and never dropped.

Table 75 provides a brief description of the error information table columns.

Table 75. Error Information Table Columns

Column name	Description
TableName	The name of the row-replica table that is the source of the row that caused the error.
RowGuid	The GUID of the row that caused the error.
Operation	One of the following commands to identify the operation that caused the error: INSERT, UPDATE, or DELETE.
Reason	The DB2 DataPropagator error message number.

Error messages table (Microsoft Jet specific)

Do *not* use SQL to update this table. Altering this table inappropriately can cause unexpected results and loss of data.

IBMSNAP_ERROR_MESSAGE

This table identifies the nature of an error. It contains the error code and error message. This table can have more than one row. The error messages table is created along with the Microsoft Jet database and never dropped.

Table 76 on page 354 provides a brief description of the error messages table columns.

Table 76. Error Messages Table Columns

Column name	Description
Reason	The DB2 DataPropagator error message number.
ReasonText	The DB2 DataPropagator error message text.

Error-side-information table (Microsoft Jet specific)

Do *not* use SQL to update this table. Altering this table inappropriately can cause unexpected results and loss of data.

IBMSNAP_SIDE_INFO

This table is a conflict table for tracking synchronization conflicts and errors. This Microsoft Jet database control table mimics Microsoft's conflict tables. This table contains the names of the conflict tables created by DataPropagator for Microsoft Jet.

Table 77 provides a brief description of the error-side-information table column.

Table 77. Error-Side-Information Table Column

Column name	Description
ConflictTableName	The conflict table name created by DataPropagator for Microsoft Jet.

Key string table (Microsoft Jet specific)

Do *not* use SQL to update this table. Altering this table inappropriately can cause unexpected results and loss of data.

IBMSNAP_GUID_KEY

This table maps the Microsoft Jet table names and row identifiers to primary key values when the following changes occur:

- Rows are deleted from Microsoft Jet database tables.
- Deletes are recorded in MSysTombstone with s_Generation, TableGUID, and s_GUID (row) identifiers, but without primary key details.
- The primary key values are needed to replicate Microsoft Jet database deletes to an RDBMS.

When DataPropagator for Microsoft Jet replicates deletes to another Microsoft Jet database, only the internal row identifier is sent. To replicate deletes outside of the Microsoft Jet environment, DataPropagator for Microsoft Jet needs to replicate a searched delete, with predicates referencing primary key values. The key string table allows DataPropagator for Microsoft Jet to maintain the key values needed to replicate a delete to an RDBMS, even after the row has been physically deleted from the row-replica table.

Table 78 provides a brief description of the key string table columns.

Table 78. Key String Table Columns

Column name	Description
RowReplicaname	Identifies the row-replica table where the row was inserted.
s_GUID	Identifies the row in the specific row-replica table.
key_string	The string of ″and-ed″ DB2 SQL predicates identifying the key columns and their row values, with character constants delimited by single ' '. The column names are taken from the row-replica definition and can contain uppercase letters, lowercase letters or both. The constant values are taken from the rows themselves and the string values can contain uppercase letters, lowercase letters, numeric characters, or any combination of the three. Microsoft Jet database supports ASCII, so the string constants can contain single- or double-byte characters. For example: `COL1=(character) AND COL2=(character)`

Synchronization generations table (Microsoft Jet specific)

Do *not* use SQL to update this table. Altering this table inappropriately can cause unexpected results and loss of data.

IBMSNAP_S_GENERATION

This table is used to prevent cyclic updates from replicating back to the RDBMS from a Microsoft Jet database. When DB2 is the target, this function is accomplished in a different way, using the APPLY_QUAL column of the critical section table, which results in a posting to the APPLY_QUAL column of the UOW table by the Capture program.

The s_GENERATION column is maintained by Microsoft Jet and set to the same generation number as any other updates made since the last synchronization. If synchronization is successful the synchronization generations table will contain one row whose Update_Type value is F.

Due to the risk of partial failures during a DataPropagator for Microsoft Jet synchronization cycle, and because the WHOS_ON_FIRST = S flow is handled before the WHOS_ON_FIRST = F flow, multiple RDBMS-to-Jet generations can be posted before any Microsoft Jet database changes replicate to the RDBMS. In such a case, there is the possibility that a list of s_GENERATION values will need to be skipped over when determining which s_GENERATION of changes needs to be replicated to the RDBMS.

Table 79 provides a brief description of the synchronization generations table columns.

Table 79. Synchronization Generations Table

Column name	Description
Update_Type	A value that indicates whether a generation of changes is: **'L'** Local to the Microsoft Jet database **'F'** Foreign
JetSynchtime	This is a dummy column, set to the time of a forced Microsoft Jet database synchronization.

Chapter 22. Problem determination facilities

This chapter provides basic information for using IBM Replication problem determination facilities, such as message explanation, trace and log records, and control table information. It also provides a high-level example of how to use the information to trace errors and a list of questions often asked by IBM service personnel that can help you to prepare to work with IBM Software Support. Use the information in this chapter to gather information about usage and operational errors so that you can work with IBM Software Support to resolve software problems.

IBM Replication provides the following facilities for problem determination:

- Error messages and SQL states for the DB2 Control Center, the Capture program, and the Apply program
- The Apply program trail table, log file, and trace file
- The Capture program trace table, log file, and trace file
- IBM Replication control tables and files

When using the problem determination facilities to test or debug your replication scenarios, we recommend that you:

- Attempt any new replications in a test environment.
- Stop other replication activity while gathering information about a problem to reduce the volume of data to sift through.

See "Problem source identification questions" on page 368 for a list of questions that can help you to research the error condition.

Replication diagnosis resources

The following section describes resources for determining how to diagnose replication errors.

Errors encountered during replication administration

The DB2 Control Center or DJRA can encounter errors either when it is gathering information from source servers, target servers, or control servers to create the SQL statements for administration or when it is actually running the SQL to set up the replication sources and subscriptions. The primary indicators are SQL messages and SQL states that accompany the error. The SQL messages and states are issued in error message windows in the DB2 Control Center.

The IBM Replication tools are primarily relational database applications. The replication control tables are created using DDL issued by the DB2 Control Center, and data is replicated primarily by SQL SELECT, INSERT, UPDATE, and DELETE statements issued by the Capture and Apply programs. When an error occurs during replication administration tasks, the error messages are normally relational database error messages, such as SQL messages and SQL states. See the DB2 message reference for your platform for more information about DB2 error messages and SQL states.

Errors encountered while running the Capture and Apply programs

The Capture and Apply programs can encounter a problem while capturing and replicating changed data, even though the SQL that the DB2 Control Center generated for defining replication sources and subscriptions ran without error. You can determine the cause of the errors with information in the following locations:

- SQL messages and SQL states found in the Apply trail table
- The Apply program trace file
- The Capture program trace table

The Capture and Apply programs issue their own messages. The messages for the Capture and Apply programs begin with the letters ASN. Explanations and user response information are provided in this book in “Chapter 23. Capture and Apply messages” on page 371, in the *DB2 Universal Database Messages Reference*, and in DB2 Universal Database online help.

The messages for the Capture and Apply programs are issued or recorded in the following locations:

- At the command line processor window or console from which the Capture and Apply programs are started
- In the Apply trail table and the Capture program trace table
- In the trace files for the Capture and Apply programs
- In the log files for the Capture and Apply programs

Apply program problem determination facilities

SQL and Apply program error messages can be found in the Apply trail table and the Apply program trace file. There is one trace file at the server associated with the Apply program. The Apply program log file tracks the activities of the Apply program and can be a useful diagnosis tool.

The Apply trail table

There is an Apply trail table (ASN.IBMSNAP_APPLYTRAIL) located at each control server with the subscription control tables, such as ASN.IBMSNAP_SUBS_SET. The Apply program inserts a new row in the Apply trail table every time it attempts to replicate a subscription. There is a row for all successful and unsuccessful subscription cycles of each replication subscription. For a description of the ASN.IBMSNAP_APPLYTRAIL table, see "Chapter 21. Table structures" on page 299.

The Apply trail table records one SQL code and one SQL state for a replication subscription that does not get replicated successfully. Additional SQL codes and states associated with the problem can be found in the Apply program trace file.

You can query the Apply trail table for information about successful and unsuccessful replications. Some key fields in the table that have problem indicators are:

STATUS
Contains -1 to indicate a failed execution.

SQLSTATE
Contains the error SQLSTATE for a failed execution.

SQLCODE
Contains the error SQLCODE for a failed execution.

SQLERRM
Contains the text of the SQL error message.

APPERRM
Contains the text of the Apply program error message.

Within the error message text, determine which database the Apply program was connected to when the error occurred; for example, did the error occur while the Apply program was connected to the source server or the copy server?

The Apply trail table has fields that identify the source and target databases and tables so that you can locate the Apply trail control rows that are causing replication errors. To reduce the number of rows that you examine, you can:

- Delete all rows from the Apply trail table, to clean out rows from past replications, before starting the Apply program.
- Temporarily disable replications that are successful in order to capture rows in the Apply trail table only for replications that have problems before starting the Apply program.

An example query for the ASN.IBMSNAP_APPLYTRAIL is:

```
SELECT * FROM ASN.IBMSNAP_APPLYTRAIL
SELECT TARGET_TABLE, STATUS, SQLSTATE, SQLCODE, SQLERRM, APPERRM
FROM ASN.IBMSNAP_APPLYTRAIL
```

Apply program trace file

The Apply program creates a trace file when the Apply program trace invocation parameter is used. See the Capture and Apply chapter for your platform in this book for the Apply program invocation command.

If you specify a trace option, specify the name of a trace output file and, for workstation systems, precede the output file name with a pipe symbol (>). For example, to start Apply for Windows with trace, issue the following command from the command line processor window:

```
\APPLY>asnapply myapply mydbnt2 trcflow > apply.trc
```

Where:

myapply
is the Apply qualifier.

mydbnt2
is the control server where the Apply program finds the control tables.

trcflow
indicates that the Apply program traces all error and flow information.

apply.trc
is the file to which the output is directed.

The trace file is located in the same directory from which the Apply program is started.

After the Apply program is stopped, you can view the trace file with any editor. You can also transmit the file to other systems, such as by FTP, or print it.

You have two trace options:

TRCFLOW
Provides very detailed information and is oriented toward helping IBM Service diagnose errors. When using TRCFLOW, we suggest isolating the subscription error, such as by running it in a test environment or disabling other error-free replication subscriptions to reduce the volume of information.

TRCERR
Provides less detail and is a better choice when you are new to the replication tools.

Within the trace, particularly with the TRCFLOW option, entries are made into the trace file for the Apply program's activities. The following are examples of the recorded information:

- Connecting to the control server to obtain information on replication subscriptions to be processed
- Connecting to source servers to fetch rows to be replicated from the CD table to the target table
- Connecting to the target servers to insert, update, and delete rows into and from the target tables

The Apply program inserts error messages and indicators in the trace file at points when it encounters an error.

The Apply program log file

The Apply program also has a log file containing messages with a summary of the Apply program's activities. The log file is in the same directory from which the Apply program is started and where the *.SPL file and any trace files are located.

The name of the Apply program log file is the Apply qualifier associated with the control server with the extension of *.APP. For example, for an Apply program operating with Apply qualifier MYAPPLY, the log file name is MYAPPLY.APP.

Because the Apply program log file information is high level, it typically directs you to the ASN.IBMSNAP_APPLYTRAIL table for more detailed information.

Capture program problem determination facilities

The Capture program has a trace table, a log file, and trace file generated when the trace invocation parameter is used to start the Capture program.

Note for AS/400: The Capture program on AS/400 does not support tracing facilities. To trace problems, view the job logs of the control and journal jobs. See "Capture for AS/400 problem determination facilities" on page 364 for more information.

Capture program trace table

The trace table is located in source server databases where the Capture program maintains change tables for replication sources. The trace table, ASN.IBMSNAP_TRACE, contains basic information about the activities of the Capture program instance. For a description of the ASN.IBMSNAP_TRACE table, see “Chapter 21. Table structures” on page 299.

You can query the ASN.IBMSNAP_TRACE table with normal SQL (such as SELECT) or query tools. For example:

```
SELECT * FROM ASN.IBMSNAP_TRACE
```

Capture program trace file

The Capture program can be started with the trace invocation parameter. This parameter specifies that problem determination records be recorded in standard output to the screen or to a file, which contains the Capture program internal logic flows. When starting the Capture program, on workstation systems, precede the output file name with a pipe symbol (>).

To start the Capture program with trace on a workstation system:

```
\CAPTURE>asnccp mysrcdb x x trace > cap.trc
```

Where:

mysrcdb
: is the source server database for this Capture program instance.

x x
: indicates that no parameters are provided for the type of start (WARM, WARMNS, COLD) and pruning (PRUNE or NOPRUNE) so the defaults are assumed.

trace
: indicates that the Capture program traces all error and flow information.

cap.trc
: is the file to which the output is directed.

The trace file is located in the same directory from which the Capture program is started.

Capture program log

The Capture program creates a log file, named by the source database that is specified when the Capture program is started; the file extension is *.CCP. If the Capture program is started with asnccp mysrcdb, the log file is named mysrcdb.ccp. The Capture Program log file is located in the same directory from which the Capture program is started.

Capture for OS/390 problem determination facilities

The Capture program for OS/390 provides you with the following tools to assist you if a severe error occurs:

- Alert generation
- Trace buffer
- Trace output
- Storage dump

This section provides information on these tools.

Alert generation

If a severe error occurs, the Capture program for OS/390 alerts NetView if NetView is active. The alert uses the NMVT format for a generic alert defined by the SNA generic alert architecture. If NetView is unavailable, diagnostic information is still available because the Capture for OS/390 program error messages are sent to the MVS console.

Trace buffer

The Capture for OS/390 program puts a small amount of critical diagnostic data in a wraparound trace buffer during processing. Each trace buffer entry describes current data capture status. If a severe error occurs, the Capture program prints the trace buffer before terminating. The printing of the trace buffer supplements the Capture program error message.

Trace output

When an error occurs, you can run Capture for OS/390 with the TRACE option. When you use this option, the Capture program writes trace information logic flow to SYSPRINT. This information can be used by IBM Service to diagnose operational problems.

Storage dump

When the Capture for OS/390 program terminates with a severe error, it saves critical diagnostic data in the CEEDUMP data set. This information is more detailed than the information in the Capture program trace buffer and can be used by IBM Service to diagnose operational problems.

Capture for VM and VSE problem determination facilities

The Capture program provides you with the following tools to assist you if a severe error occurs:

- Trace buffer

- Trace output
- Storage dump

This section provides information on these tools.

Trace buffer

The Capture program puts a small amount of critical diagnostic data in a wraparound trace buffer during processing. Each trace buffer entry describes current data capture status. If a severe error occurs, the Capture program prints the trace buffer before terminating. The printing of the trace buffer supplements the Capture program error message.

Trace output

When an error occurs, you can run the Capture program with the TRACE option. When you use this option, the Capture program writes trace information logic flow to the VM console (for VM) or STDOUT (for VSE). This information can be used by IBM Service to diagnose operational problems.

Storage dump

When the Capture program terminates with a severe error, it saves critical diagnostic data in the CEEDUMP data set. This information is more detailed than the information in the Capture program trace buffer.

Capture for AS/400 problem determination facilities

Capture for AS/400 has unique problem determination facilities because of its dependency on journals and journal receivers as its primary source of input. The following section describes Capture program for AS/400 problem determination facilities and problem recovery methods.

Where to start looking for information

The first place to start looking for currently active jobs is the jobs running in the DPROPR/400 subsystem. Issue the work with submitted jobs command **WRKSBSJOB QZSNDPR** to receive a list of all the active jobs in that subsystem. If you fail to find the job, use the **WRKSBMJOB** command to locate and view the job log for the job. The name of the Capture program control job is QZSNCTL5. The name of the journal job is the same as the journal name (either QSQJRN (the default journal name for SQL collections) or a name that you specified for your journal).

As you go along, record the 6-digit job numbers. They might be needed in the following steps.

Journal job does not start

You might start the Capture program only to find out that only one job, the job QDPRCTL5, is running. If this persists longer than 5 minutes, you can check the following conditions:

- If you are using **WRKSBSJOB** or **WRKACTJOB** command to display active jobs, enter option 7 on the command line for job QZSNCTL5. Message ASN2017 might be waiting for a reply.
- Run the command **WRKJOB QSQJRN** (or the name of the journal job on your system). The journal job might have started, but ended right away for some reason (for example, because the lag limit was exceeded). If you find a recent journal job, display its job log to first confirm it is the right job and then to determine why it ended.
- The QDPRCTL5 control job might not have found any replication sources that are eligible for replication. Two conditions make a replication source eligible for replication:
  - The replication source must have at least one replication subscription defined.
  - The Apply program must start a full refresh to copy the original contents of the source table to the target table, even if the source is empty. The full refresh brings the target table in synchronization with the source table. At the time the Apply job starts the full refresh, the pruning control table row for this replication pair has its SYNCHPOINT column set to hex zeroes.

  When you first issue the **STRDPRCAP** command, there may be replication pairs that meet the first condition but not the second one. No journal job will be started at that time if none of the replication sources meet both conditions.

  Every two minutes thereafter (or at the frequency you specify on the WAIT parameter of the **STRDPRCAP** command), the QZSNCTL5 job wakes up to determine if any replication source exists that meet both conditions listed above. If it finds one, it starts the journal job.

  To see if a replication pair's pruning control table row has a SYNCHPOINT set to hex zeroes, issue the following statement on the source server:

  ```
  SELECT HEX(SYNCHPOINT) FROM ASN/IBMSNAP_PRUNCNTL
                   WHERE SOURCE_TABLE='xxx' AND SOURCE_OWNER='yyy'
                   AND SOURCE_VIEW_QUAL=z
  ```

  where *xxx* is the library name, *yyy* is the table name, and *z* is the source view qualifier of the replication source in question. The contents of *xxx* and *yyy* are case-sensitive.

Collecting data for Capture program problem determination

Following is a list of items that are needed for Capture program problem determination, in order of importance:

1. The job log of the Capture control job QDPRCTL5.
2. The job log of the Capture journal job in question.
3. Keeping the journal receivers is very important because they contain important time sequence information. Save the journal receivers in a file or make sure the receivers are kept if a remote sign-on by IBM service is likely to occur. The journal receivers of the following tables are useful in problem determination:

   The control tables
   : The register table, the register extension table, the pruning control table, the unit-of-work table, the critical section table, and the warm start table. The journal for these tables is ASN.QSQJRN.

   The source table in question
   : These journal receivers are kept as long as the Capture program needs them. Make sure they are not deleted automatically by the system if problem determination is to take place.

   The change data table of the replication source in question
   : Usually the journal is named QSQJRN and is in the same library as the source table. If it is not, find the library and the name of the journal by issuing a **DSPFD** *xxx/yyy* command (where *xxx* is the library, and *yyy* is the system name of the change data table in question). To find the system name of the change data table, issue the following statement:

   ```
   SELECT DBXFIL FROM QSYS/QADBXREF
                   WHERE DBXLFI = 'sqlname' AND DBXLIB = 'xxx'
   ```

   where *sqlname* is the SQL name of the subject change data table. The contents of *sqlname* and *xxx* are case sensitive.
4. The formatted dump of the user index QDPR/QZSNINDEX5. Issue the following command before the Capture program ends:

   ```
   DMPOBJ QDPR/QZSNINDEX5 *USRIDX
   ```
5. If you have questions about the behavior of a certain replication sources, it is important to collect the contents of the corresponding rows in the register table and the register extension table. You can collect them by issuing the following statements:

   ```
   SELECT A.*, HEX(CD_OLD_SYNCHPOINT), HEX(CD_NEW_SYNCHPOINT)
                   FROM ASN/IBMSNAP_REGISTER A
                   WHERE SOURCE_OWNER='xxx' AND SOURCE_TABLE='yyy'

   SELECT * FROM ASN.IBMSNAP_REG_EXT
                   WHERE SOURCE_OWNER='xxx' AND SOURCE_TABLE='yyy'
   ```

where *xxx* is the library name and *yyy* is the table name of the replication source in question. The contents of *xxx* and *yyy* are case sensitive.

6. At times, you need to retrieve the contents of the global row in the register table with the following statement:

```
SELECT A.*, HEX(SYNCHPOINT)
                FROM ASN/IBMSNAP_REGISTER A
                WHERE GLOBAL_RECORD = 'Y'
```

7. The Capture program trace table has entries that record the significant job events. To find entries for the Capture program control job or the Capture program journal job, enter the following statement (assuming the job number of the job you want to investigate is 011932):

```
SELECT * FROM ASN/IBMSNAP_TRACE
                WHERE SUBSTR(JOB_NAME, 21, 6) = '011932'
                ORDER BY TRACE_TIME
```

   The DESCRIPTION column provides important information about the job.

   The following statement can be used to gather trace table entries after a certain time (at 7 a.m., March 31, 1999, for example):

```
SELECT * FROM ASN/IBMSNAP_TRACE
                WHERE TRACE_TIME > '1999-03-31-07.00.00.000000'
                ORDER BY TRACE_TIME
```

   The following statement retrieves all of the ASN0303 (data capturing is interrupted ...) trace entries after 7 a.m., March 31, 1999:

```
SELECT * FROM ASN/IBMSNAP_TRACE
                WHERE TRACE_TIME > '1999-03-31-07.00.00.000000' AND
                      SUBSTR(DESCRIPTION, 1, 7) = 'ASN0303'
                ORDER BY TRACE_TIME
```

8. The warm start table has one entry for every journal that is used by one or more replication sources. You can retrieve the row for the journal job by entering the following statement:

```
SELECT * FROM ASN/IBMSNAP_WARM_START
                WHERE JRN_LIB='xxx' AND JRN_NAME='yyy'
```

   where *xxx* is the library name and *yyy* is the table name of the journal. The contents of *xxx* and *yyy* are case sensitive.

9. In some cases, it is necessary to collect data from the target server to determine how and why the Apply program fails to replicate to the target table:

   - The audit trail table.
   - The job log of the Apply jobs.
   - The subscription set table and its journal receivers (since it may be necessary to find out how some columns changed along the time line).

10. The formatted dump of the user spaces for the journal jobs. They are in library QDPR, and they have the name of QDPR*xxxxxx*, where *xxxxxx* is the job number of the journal job. Issue the following command before Capture ends:

    ```
    DMPOBJ QDPR/QDPRxxxxxx *USRSPC
    ```

Problem determination scenario

The following steps are a high-level description of how you might trace a replication error, using the facilities discussed in this chapter.

In this scenario, you used the DB2 Control Center to define replication sources and subscriptions. The SQL for your replication requests completed satisfactorily, but the Apply program can’t execute the replication successfully. To determine the error, you could:

1. Examine any error messages returned directly to the terminal for the Apply program job or process.
2. Examine the Apply trail table (ASN.IBMSNAP_APPLYTRAIL) for any indicators of the problem.
3. Examine the Capture program trace table (ASN.IBMSNAP_TRACE) for indicators from the Capture program’s activity.
4. Examine the log files for the Capture and Apply programs for indicators from the activities of the Capture and Apply programs.
5. Rerun the Capture and Apply programs with the trace option and examine the trace file for indicators of the problem.

Problem source identification questions

If you call IBM Software Support Services, you will be asked the following types of questions by Level 2 Service Support. You can save time and perhaps diagnose the error yourself by researching the answers to these questions.

1. What was occurring at the time of the problem?
2. What has changed recently in the environment?
3. Describe the environment.
4. What is the CSD level of DB2 Universal Database V6 where the Control Center is installed?
5. On what platform is the Capture program running?
6. At what maintenance level is the Capture program?
7. What is the maintenance level of DB2 where the Capture program is running?
8. On what platform is the Apply program running?

9. At what maintenance level is the Apply program?
10. On what release of DB2 does the Apply program run?
11. What is the maintenance level of DB2 where the Apply program is running?
12. Is this a mobile user?
13. What are the ASN messages that are issued?
14. Are there other messages either in SYSLOG (for AS/400, the QSYSOPR message queue) or on the screen?
15. What is the complete message text for all messages? Be sure to note all message numbers, database names, table names, user IDs, and file names that appear in messages.
16. Where does the failure occur?
    a. The DB2 Control Center
       - Is the problem with a replication source or subscription?
       - What messages appear?
       - Can the user successfully connect to the source or target database from a command line or command prompt window?
    b. The Apply program
       - Is the Apply program running?
       - If not, what occurs when the Apply program starts?
       - What messages appear?
       - Is there error information in the ASN.IBMSNAP_APPLYTRAIL table?
       - Is there error information in the Apply program log file (for AS/400, in the Apply job log)?
       - Are data changes being successfully replicated to the target table?
       - Do all tables in a replication subscription have the same problem?
       - What table types (for example, user copy, point-in-time, CCD) are involved in the failure?
       - Did you run the Apply program with trace?
       - Are CALL procedures being used?
       - Is a CCD being used?
    c. The Capture program
       - Is the Capture program running?
       - If the Capture program is not running, what happens when a warm start of the Capture program is tried?
       - Is there error information in the ASN.IBMSNAP_TRACE table?
       - Is there error information in the Capture program log file?
       - What is the DB2 configuration?

- Are data changes being successfully inserted into the CD tables?
- Does the user ID running the Capture program have sufficient privileges to run the Capture program?

d. DJRA

- What level of DJRA are you using? (Click **About** from the main window for this information.)
- If accessing non-IBM data sources, what level of DB2 DataJoiner are you using, and on which platform (AIX or NT)?
- Save the Generated Script file and the output file.

Chapter 23. Capture and Apply messages

The following is a list of messages issued by DB2 replication for the Capture and Apply programs. A brief explanation of the status is provided.

Unless otherwise stated, all error codes described here are internal error codes used by IBM Service and IBM development. Also, unless otherwise stated, error messages are issued with a return code of 8.

The replication messages have the following prefixes:

ASN0 The Capture program

ASN1 The Apply program

ASN2 Capture for AS/400

In addition to using the information here, you can obtain explanations for messages by typing the following at a DB2 command prompt:

```
db2 message number
```

Note: The Replication Administration messages (DBA6001-DBA6110) are listed in the DB2 *Messages Reference*.

Capture program messages

Note: For soft SQL errors, see the DB2 *Messages Reference* for your platform.

ASN0000S An internal error occurred for message number "<number>". The error code is "<error_code>". The return code is "<return_code>".

Explanation: The message file for Capture was installed incorrectly.

User Response: Refer to the installation and configuration information in this book pertaining to your platform. Make sure the message file is installed in the correct directory. If it is, contact your IBM Service representative.

ASN0001E The Capture program encountered an SQL error.

Parameters:

- **Routine name is "<name>"**
- **SQL request is "<request>"**
- **table name is "<table_name>"**
- **SQLCODE is "<sqlcode>"**
- **SQLERRML is "<sqlerrml>"**
- **SQLERRMC is "<sqlerrmc>"**

Explanation: A nonzero SQLCODE was returned when the Capture program issued an EXEC SQL statement.

User Response: See the messages and codes publication of the DB2 database manager on your platform for information about SQL return codes that use SQLERRML and SQLERRMC as substitution fields. Contact your DBA for more information.

ASN0002E The Capture program could not connect to DB2.

Parameters:

- **Routine name is "<routine>"**
- **SQLCODE is "<sqlcode>"**

Explanation: An error occurred when the Capture program issued either

- a CONNECT function to DB2 for VSE & VM
- a CONNECT function to DB2 Call Attachment Facility (CAF)
- an implicit connect to DB2 for common services

User Response: See DB2 codes in the messages and codes publication of the DB2 database manager on your platform for the appropriate reason code.

For DB2 for OS/390 errors, see the section in the administration guide that describes the Call Attachment Facility. Contact your DBA for questions and diagnosis.

If you are running Capture under DB2 UDB for UNIX or under DataJoiner for UNIX, ensure that the LIBPATH environment variable is set to the same environment in which the Capture program starts. See "Chapter 10. Capture and Apply for UNIX platforms" on page 197 for more information.

ASN0003E The Capture program could not open the plan.

Parameters:

- **Routine name is "<routine>"**
- **Return code is "<return_code>"**
- **Reason code is "<reason_code>"**
- **Subsystem is "<subsystem>"**
- **Plan name is "<ASNLPLAN>"**

Explanation: An error occurred when the Capture program tried to open the plan, ASNLPLAN.

User Response: See the DB2 Codes section in the messages and codes publication of the DB2 database manager on your platform to find the appropriate reason code. See the appropriate section in the administration guide publication of the DB2 database manager on your platform: "Call Attachment Facility".

ASN0004E The Capture program could not start the trace.

Parameters:

- **Routine name is "<routine>"**
- **Return code is "<return_code>"**
- **Reason code is "<reason_code>"**

Explanation: An error occurred when the START TRACE DB2 command was issued, or when Capture programread the DB2 log.

User Response: See the DB2 Codes section of in the messages and codes publication of the DB2 database manager on your platform to find the appropriate reason code. For more information, see either of the following sections in the administration guide publication of the DB2 database manager on your platform: "Call Attachment Facility" (CAF) for START TRACE DB2 errors, or the Instrumentation Facility Interface (IFI) for DB2 log read errors, or contact your DBA. If CAF or the IFI returned a message, it is also printed on the system display console.

ASN0005E The Capture program encountered an error while reading the DB2 log.

Parameters:

- **Routine name is "<routine>"**
- **LSN is "<log_sequence_number>"**
- **Return code is "<return_code>"**
- **Reason code is "<reason_code>"**

Explanation: An error occurred when the Capture program read the DB2 log. There might be an SQL error.

For Capture program for OS/390, a dump has been generated for this message. The output appears in the data set whose name is specified by the CEEDUMP DDNAME on your Capture program for OS/390 invocation JCL.

For DB2 DataPropagator, the "<return_code>" value is for the Asynchronous Read Log. For UNIX, the log file might not be in the path.

For Capture for VSE, the "<return code>" is for the VSE/VSAM GET macro.

For Capture for VM, the "<return code>" is for Diagnose X'A4'.

User Response: See the DB2 Codes section in the messages and codes publication of the DB2 database manager on your platform for the appropriate reason code.

For Capture program for OS/390, see the Instrumentation Facility Interface (IFI) section in the administration guide publication of the DB2 database manager on your platform or contact your DBA.

For Capture for VSE, see the *VSE/VSAM Commands and Macros*, *VSE/ESA System Macro Reference*, and *VSE/ESA V2R3 Messages and Codes* manuals for more information.

For VM/ESA, see the VM/ESA Programming Services for more information.

For the IBM DPROPR Capture of Universal Database, see the active and archived database logs section in the administration guide for common servers or contact your IBM Service Representative.

ASN0006E The Capture program encountered an unexpected log error of unknown log variation. The routine name is "<routine>".

Explanation: An unexpected log error not reported by either:

- the Instrumentation Facility Interface (IFI) for Capture program for OS/390, or
- the Asynchronous Read Log API for IBM DPROPR Capture of Universal Database

occurred while the Capture program was processing the DB2 log records. The Capture program could not determine the type of SQL update associated with the log record.

For Capture program for OS/390, a dump has been generated for this message. The output appears in the dataset whose name is specified by the CEEDUMP DDNAME on your Capture program for OS/390 invocation JCL.

User Response: Contact your IBM Service Representative.

ASN0007E The Capture program encountered an unexpected log error of unimplemented data type. The routine name is "<routine>".

Explanation: An unexpected log error not reported by either:

- the Instrumentation Facility Interface (IFI) for Capture program for OS/390, or
- the Asynchronous Read Log API for IBM DPROPR Capture of Universal Database

occurred while the Capture program was processing the DB2 log records. The Capture program could not determine the type of SQL update associated with the log record.

For Capture program for OS/390, a dump has been generated for this message. The output appears in the dataset whose name is specified by the CEEDUMP DDNAME on your Capture program for OS/390 invocation JCL.

User Response: Contact your IBM Service representative.

ASN0008I The Capture program was stopped.

Explanation: The IBM Replication administrator stopped the Capture program using one of the valid methods.

Explanation: This message is for your information only.

User Response: No action is required.

ASN0009E The table was created without the DATA CAPTURE CHANGES (DCC) attribute.

Parameters:

- **Routine name is "<routine>"**
- **Table name is "<table_name>"**

Explanation: The source table was defined without the DCC attribute and the Capture program tried to capture changes for the replication source.

User Response:

1. Stop the Capture program.
2. Delete the replication source.
3. Define the replication source again; if you do not have the **Data capture is full-refresh only** check box selected, the DB2 Control Center will alter the source table with the DCC attribute.
4. Start the Capture program.

ASN0010E The Capture program cannot obtain enough storage.

Parameters:

- **Routine name is "<routine>"**
- **Storage required is "<amount>"**

Explanation: The Capture program cannot continue processing because not enough free storage is available.

User Response: For Capture program for OS/390, ensure that the REGION parameter has enough storage allocated to run your job. If necessary, contact your OS/390 system programmer to determine the method for requesting sufficient storage.

For Capture for VM, a request to obtain virtual storage could not be satisfied. You might need to increase the size of the virtual machine in which Capture program runs.

For Capture for VSE, all available GETVIS storage has been exhausted. You might need to restart the Capture program after allocating a larger partition.

ASN0011E The DB2 compression dictionary is not available or the IFCID 306 buffer is invalid.

Parameters:

- **Routine code is "<routine_code>"**
- **Reason code is "<reason_code>"**

Explanation: In the case of DB2 compression dictionary is not available error, the Capture program attempted to read log records for an old compression dictionary. DB2 for OS/390 only retains one version of the compression dictionary in memory. DB2 can only decompress log records for a compressed table if the compression dictionary used to compress the log records is still the current compression dictionary.

In the case of the IFCID 306 buffer being invalid, the control information is missing from the buffer.

For both cases, a dump has been generated for this message. The output appears in the dataset whose name is specified by the CEEDUMP DDNAME on your Capture program for OS/390 invocation JCL.

User Response: For the DB2 compression dictionary error, to avoid an unwanted cold start of the Capture program, you must capture all log records for a compressed table before creating a new version of the compression dictionary. Use the KEEPDICTIONARY option to retain the current version of the compression dictionary during routine REORG processing.

When you want a new compression dictionary for the table, you must synchronize running the REORG utility with running your updated applications and the Capture program as follows:

1. Quiesce your updated applications.
2. Let the Capture program capture all logged updates for the compressed table.
3. Use the REORG utility on the compressed table, creating a new compression dictionary.
4. Release your updated applications.

For the IFCID 306 buffer error, ensure all DB2 maintenance is current.

ASN0013E The Capture program required a column that was not defined in the change data (CD) table.

Parameters:

- **Routine name is "<routine>"**
- **Table name is "<table_name>"**

Explanation: The user did not define an IBMSNAP required column in the change data table.

User Response: Ensure that the change data table definition is correct. Refer to "Chapter 21. Table structures" on page 299 for more information.

ASN0014E The processing of the Capture program has fallen below a minimum level. The log record lags current time by "<number>" seconds. The routine name is "<routine>".

Explanation: The Capture program terminated because a high DB2 transaction rate caused the Capture program to run slower than the defined minimum level.

User Response: Refer to the Capture and Apply chapter for your platform for more information on the lag limit. Perform a cold start.

ASN0015E The Capture program encountered a storage allocation error.

Parameters:

- **Routine name is "<routine>"**
- **Storage required is "<amount>"**

Explanation: A storage allocation error was detected; sufficient storage is not available. The Capture program might have been installed improperly.

For the Capture program on AIX, you might not have set the soft links for the component files to the shared directory.

User Response: Determine why memory could not be allocated by looking at the operating system and application task status. Contact your system programmer to determine the method of requesting the storage listed in the error message.

For Capture for AIX, determine whether you have set the soft links for the component files.

For Capture for VM, a request to obtain virtual storage could not be satisfied. You might need to increase the size of the virtual machine in which Capture program runs.

For Capture for VSE, all available GETVIS storage has been exhausted. You might need to restart the Capture program after allocating a larger partition.

ASN0016E The Capture program could not begin capturing changes because there was no eligible replication source.

Parameters:

- **Routine name is "<routine>"**
- **Table name is "<table_name>"**

Explanation: The replication source information in the register table has not been defined.

The Capture program started but could not find source tables that were:

- Enabled with the DATA CAPTURE CHANGES option of the CREATE or ALTER TABLE statement.
- Defined as replication sources with the **Data capture is full-refresh only** check box cleared on the Define as Source window.

User Response: Ensure that the register table is defined properly. For more information about the register table, see "Chapter 21. Table structures" on page 299. Verify that replication sources have been defined.

ASN0017E The Capture program encountered a severe internal error and could not issue the correct error message. The routine name is "<routine>"; the return code is "<return_code>"; the error message number is "<error_message_num>".

Explanation: The Capture program could not retrieve the message from the Capture program messages file.

User Response: Edit the Capture program error message file. Locate the ASNnnnn error message number to determine which error message should have been issued. See the information about the error message in this listing to determine how to resolve the error.

ASN0018W The Capture program did not process updates made to the register table rows. The routine name is "<routine>"; the table name is "<table_name>".

Explanation: The user changed a replication source definition while the Capture program was running and then issued a REINIT command. The register table, which contains a row for each replication source, might not match the other replication source control tables.

User Response:

1. Stop Capture.
2. Delete the replication source.
3. Redefine the replication source.
4. Start Capture.

ASN0019E The Capture program libraries are not authorized for the Authorized Program Facility (APF).

Explanation: The Capture program cannot process the STOP, SUSPEND, RESUME, or REINIT commands because the STEPLIB libraries are not authorized for APF.

User Response: Authorize the Capture link library for APF.

ASN0020I Netview Generic Alerts Interface failure. The Netview return code is "<return_code>".

Explanation: The Network Major Vector Transport (NMVT) could not be sent to Netview by the program because the program interface failed. This is a secondary informational message.

User Response: See the Netview programming documentation for a description of the return code to determine the interface error. The Capture program alerts will not be received by the System Services Control Point (SSCP) until the error is corrected.

ASN0021I Netview Program to Program Interface unavailable. The Netview return code is "<return_code>".

Explanation: Netview is unavailable. This is a secondary informational message.

User Response: See the Netview programming documentation for a description of the return code to determine the Netview problem. For example, the subsystem might not have been started.

ASN0022E DB2 release "<release>" is not supported. The routine name is "<routine>".

Explanation: The Capture program does not support this release of DB2.

User Response: Run the Capture program with the appropriate release of DB2.

ASN0023I The Capture program successfully reinitialized the register table. The table name is "<table_name>"; the routine name is "<routine_name>".

Explanation: A REINIT command was issued and the updates were successfully made to the Capture programinternal control information. This message is for your information only.

User Response: No action is required.

ASN0024I The Capture program did not need to reinitialize the register table. Table “<table_name>”did not change.

Explanation: The REINIT command was issued. No updates were made to the register table since initialization or the last REINIT. This message is for your information only.

User Response: No action is required.

ASN0025I The Capture program reinitialized the register table. Table “<table_name>” has “<number>” potentially bad row(s).

Explanation: This message accompanies ASN0018W. Reinitialization was performed as requested despite potential problems reported in ASN0018W.

User Response: See ASN0018W.

ASN0026W The Capture program could not allocate the trace buffer. The routine name is “<routine>”; the storage required is <required_storage>”.

Explanation: A storage allocation error was detected; not enough storage is allocated for the trace buffer. The trace buffer is an information-only feature of the Capture program and the allocated storage is not required for the Capture program to run.

User Response: Contact your system programmer to determine the method of requesting the storage listed in the error message.

For Capture for VM, a request to obtain virtual storage could not be satisfied. You might need to increase the size of the virtual machine in which the Capture program runs.

For Capture for VSE, all available GETVIS storage has been exhausted. You might need to restart the Capture program after allocating a larger partition.

ASN0027W The Capture program is already active.

Explanation: You tried to start more than one the Capture program per DB2 subsystem or database.

For VSE/ESA, Capture for VSE generates a unique lock name for each database. This lock name is already in use, indicating that Capture for VSE is already active for the database.

For VM/ESA, Capture for VM has determined that the resource ID used as a lock is already in use. The resource ID is specified on the ENQ_NAME parameter of the CAPTURE ASNPARMS file.

User Response: For DB2 for OS/390 subsystems, either run only one instance of the Capture program for all subsystems that are members of a data-sharing group, or run only one instance of the Capture program on any stand-alone system.

For other DB2 database platforms, run only one Capture program per database.

For Capture for VM, you can change the ENQ_NAME parameter in the CAPTURE ASNPARMS to ensure unique values for each Capture program if you want to run Capture for VM for more than one DB2 database on a system.

ASN0028I The Capture program is suspended by operator command.

Explanation: The IBM Replication administrator suspended the Capture program and has entered a wait state. This message is for your information only.

User Response: No action is required.

ASN0029I The Capture program is resumed by operator command.

Explanation: The IBM Replication administrator resumed the Capture program from a suspended state and the Capture program has continued running. This message is for your information only.

User Response: No action is required.

ASN0030I The Capture program command entered by the operator was unrecognized.

Explanation: The IBM Replication administrator entered a command not recognized by the Capture program. The only valid commands are:

- STOP (*Ctrl+C* for DB2 DataPropagator)
- SUSPEND
- RESUME
- REINIT
- PRUNE
- GETLSEQ

There are no parameters allowed for these commands.

User Response: Use only valid Capture program commands.

ASN0031E The Capture program tuning parameters table can have only one row. The routine name is "<routine>"; the table name is "<table_name>".

Explanation: The tuning parameters table was not defined correctly or has been updated with invalid rows.

User Response: Refer to "Chapter 21. Table structures" on page 299 to determine the correct format of this table. Remove any invalid rows.

ASN0033E The Capture program could not reinitialize the register table. The table name is "<table_name>".

Explanation: The IBM Replication administrator tried to reinitialize the Capture program, but there was an error in the register table. A user might have tried to update a replication source while the Capture program was running or suspended, and the register table might not match the other control tables.

User Response: This is a secondary message. See any preceding messages for more information about the error. See the Capture and Apply section for your platform for more information about reinitializing the Capture program and "Chapter 21. Table structures" on page 299 for information about the register table.

ASN0034E An incorrect value was supplied for column "<column>"of the Capture program tuning parameters table. The routine name is "<routine>"; the table name is "<table_name>".

Explanation: The tuning parameters table does not have the correct values. Values might be out of range.

User Response: Refer to the Capture and Apply section for your platform for more information. Check the lag limit, retention period and commit frequency.

ASN0035W Some rows were found in the register table with an unsupported architectural level. The routine name is "<routine>"; the table name is "<table_name>".

Explanation: The register table version does not match the current version of the Capture program. The current version of the DB2 Control Center is not compatible with the version of the Capture program that you are running.

User Response: Refer to "Chapter 21. Table structures" on page 299 to check the required value for the ARCH_LEVEL column in the register table. Verify that the value in the register table at the source server is correct. If not, use compatible versions of the Control Center and the Capture program.

ASN0036E DB2 was terminated abnormally. The routine name is "<routine>".

Explanation: DB2 was terminated while the Capture program was still active.

For OS/390, VSE/ESA or VM/ESA, DB2 was terminated while Capture program was active and the user did not specify the NOTERM invocation parameter.

User Response: Start DB2 and start the Capture program.

ASN0037W DB2 was terminated in QUIESCE mode. The routine name is "<routine>".

Explanation: DB2 was terminated while the Capture program was still active.

User Response: Start DB2 and start the Capture program.

ASN0038E The disconnect to DB2 failed. The routine name is "<routine>"; the return code is "<return_code>"; the reason code is "<reason_code>".

Explanation: DB2 was stopped in QUIESCE mode, but user wanted to leave the Capture program running. While terminating the connection to DB2, Capture program received an error returned code from the Call Attachment Facility (CAF).

User Response: Restart Capture program.

ASN0040E An error was returned from the FORK function of "<platform>". The error is "<error_text>".

Explanation: An AIX FORK function returned a negative value. "<Error_text>"describes the error.

User Response: See *AIX Calls and Subroutines Reference* for information about FORK functions, use the provided error text to determine the error, or contact your IBM Service Representative.

ASN0041E An error was returned while getting the instance name. The reason code is "<reason_code>".

Explanation: The SQLEGINS API of DB2 Universal Database returned an error.

User Response: See the *DB2 for common servers API Reference* for information about the SQLEGINS API to determine the error or contact your IBM Service Representative.

ASN0042E An error was returned from the EXECLP function. The error is "<error_text>".

Explanation: The AIX EXECLP function returned a negative value. "<Error_text>" describes the error.

User Response: See the *AIX Calls and Subroutines Reference* for information about the EXECLP function or contact your IBM Service Representative.

ASN0043E A child process of ASNLMAIN died.

Explanation: The child process created by ASNLMAIN terminated. Possible causes include:

- A user stopped the child process.
- There is an AIX system problem.

User Response: Check the system processes for conflicts or contact your AIX system programmer.

ASN0044E The child process has not called the dummy process after an extended wait.

Explanation: The child process was unable to call the dummy routine ASNLPVRF. The installation softlinks might not have been set.

User Response: Verify whether the installation softlinks have been set, check the system for problems, or contact your IBM Service Representative.

ASN0045E An error was returned from the MSGRCV function. The error is "<error_text>".

Explanation: The function MSGRCV returned an error. "<Error_text>" describes the error.

User Response: Use the provided error text to determine the error, or contact your IBM Service Representative.

ASN0046E An error was returned from the MSGGET function. The error is "<error_text>".

Explanation: The function MSGGET returned an error. "<Error_text>" describes the error. This error occurs during message handling.

User Response: Use the provided error text to determine the error, or contact your IBM Service Representative.

ASN0047E An error was returned from the FTOK function of "<platform>". The error is "<error_text>".

Explanation: The AIX function FTOK returned an error. "<Error_text>" describes the error.

User Response: See *AIX Calls and Subroutines Reference* for information about the FTOK function, use the provided error text to determine the error, or contact your IBM Service Representative.

ASN0048E The Capture program could not open the log file. The error is "<error_text>. The error code is "<error_code>".

Explanation: The Capture program could not open the log file. Some possible reasons are:

- The Capture program log file was deleted.
- The user does not have the correct authorization for the Capture program directory.

User Response: Contact your system programmer to determine the error or contact your IBM Service Representative.

ASN0050E The Capture program encountered an error while writing to the error message file.

Explanation: An I/O error occurred while writing to the Capture program log file.

User Response: Check the trace table for error messages.

ASN0053E An error was returned by the Asynchronous Read Log API (SQLURLOG).

Parameters:

Initial LSN is
"<log_sequence_number>"

FIRSTRead LSN is
"<first_read_LSN>"

lastRead LSN is
"<last_read_LSN>"

CurActive LSN is
"<currently_active_LSN>"

log Recswritten is
"<log_records_written>"

log Byteswritten is
"<log_bytes_written>"

Explanation: The Asynchronous Read Log API returned an SQLCODE in the SQL error message that preceded this message. The information in this message provides additional information about the SQL error.

User Response: See ASN0001E for information about SQLCODEs.

ASN0054E The Capture program did not recognize the invocation parameter.

Explanation: An invalid invocation parameter was entered with the ASNCCP command.

User Response: Enter a valid invocation parameter.

See the Capture and Apply section for your platform for information about valid parameters.

ASN0055E The Capture program encountered an SQLTYPE that is not supported in the origin table.

Parameters:

- **Routine Name is "<routine>"**
- **Column Number is "<column_num>"**

Explanation: The Capture program encountered an invalid SQL type. A table might have been defined as a replication source outside the DB2 Control Center and contains unsupported SQL types (for example, LONG VARGRAPHIC).

User Response: Delete the replication source and use the DB2 Control Center to define replication sources to ensure only valid types are defined. Or, when manually defining the replication source, ensure that the table has supported SQL types. See the messages and codes publication of the DB2 database manager on your platform to determine the invalid SQLTYPE.

ASN0056E ASN.IBMSNAP_UOW table does not exist.

Explanation: The unit-of-work (UOW) table might have been dropped, or the source server database might have been dropped.

User Response: Contact your IBM Service representative.

ASN0100I The Capture program initialization is successful.

Explanation: This message is for your information only.

User Response: No action is required.

ASN0101W The Capture program warm start failed because existing data is too old; a cold start will be attempted.

Explanation: The data in the change data tables is older than the value "<current_timestamp_lag_limit>". A cold start will be performed.

User Response: See "Chapter 21. Table structures" on page 299 for more information about warm and cold starts to determine why Capture program could not warm start.

ASN0102W The Capture program will switch to cold start because the warm start information is insufficient.

Explanation: A problem occurred during the retrieval of the warm start information. The warm start table data was invalid. A cold start will be performed.

For DB2 Universal Database, an Asynchronous Read Log API error occurred while reading the log during warm start. For OS/390, an Instrumentation Facility Information (IFI) error occurred while reading the log during warm start.

User Response: See "Chapter 21. Table structures" on page 299 for more information about warm and cold starts to determine why Capture program could not warm start.

ASN0103I The Capture program started with: "<server_name>".

Parameters:

- **SERVER_NAME is "<server_name>"**
- **ENQ_NAME is "<enq_name>"**
- **START_TYPE is "<start_type>"**
- **TERM_TYPE is "<term_type>"**
- **PRUNE_TYPE is "<prune_type>"**

Explanation: This is an informational message that displays the DB2 server name and the Capture program start up option.

For Capture for VSE and VM, the ENQ_NAME shows the name on which Capture program locks to make sure that there is only one Capture program running for any DB2 database. The lock name can be specified for VM/ESA by changing the ENQ_NAME parameter value in the CAPTURE ASNPARMS file.

User Response: No action is required.

ASN0104I Change capture started for owner "<owner>", the table name is "<copy_table>" at log sequence number (LSN) "<log_sequence_number>".

Explanation: The Capture program was started for the table owner and table name at the specified log sequence number (LSN). This message is issued for each origin table for which the Capture program captures changes. This message is for your information only.

User Response: No action is required.

ASN0105I Data that has been copied was pruned from the change data table and the unit-of-work table.

Explanation: This message is for your information only.

User Response: No action is required.

ASN0106I The Capture program is waiting for DB2 to come up.

Explanation: When the Capture program is initially brought up, if DB2 is not up at that time, the Capture program waits until DB2 is up. After DB2 is up, the Capture program makes the connection and begins to capture changes.

If the NOTERM option is specified in the Capture invocation parameters, and DB2 comes down smoothly, the Capture program waits for it to come back up.

User Response: No action is required.

ASN0110E Capture for OS/390 Storage Dump. The Control Address is "<address>".

Explanation: This is an informational message printed at the top of storage dumps for severe errors. When a dump is generated for a message, the dump output appears in the dataset whose name is specified by the CEEDUMP DDNAME on your Capture for OS/390 invocation JCL.

User Response: No action is required.

ASN0115I The warm start control information was not supplied. The routine name is "<routine>"; the table name is "<table_name>".

Explanation: The warm start table is missing or corrupted. This table provides a faster warm start. The Capture program will warm start.

User Response: No action is required.

ASN0116I The Capture program did not reinitialize the tuning parameters table. The routine name is "<routine>"; the table name is "<table_name>".

Explanation: The REINIT command was issued, but tuning parameter information from the tuning parameters table was not available. The previous tuning parameter values were retained.

User Response: No action is required.

ASN0117W Warm start control information was not saved. The routine name is "<routine>"; the table name is "<table_name>".

Explanation: An error occurred that prevented warm start information from being saved in the IBMSNAP_WARM_START table. Warm start can be attempted and can take longer because backup sources will be used.

User Response: No action is required.

ASN0121E The Capture program warm start failed because existing data is too old. The Capture program will terminate.

Explanation: The time of the warm start information exceeded LAG_LIMIT.

User Response: No response required; the Capture program will terminate because WARMNS was specified.

ASN0122E An error occurred while reading the warm start information or DB2 log. The Capture program will terminate.

Explanation: A problem occurred while retrieving the warm start information. The warm start table data was invalid or for OS/390, an Instrumentation Facility Interface (IFI) error occurred while reading the log during warm start.

User Response: No response required; the Capture program is terminating because WARMNS was specified.

ASN0123I The highest log sequence number of a successfully captured log record is "<log_sequence_number>".

Explanation: The Capture program saved the highest log sequence number (LSN) in the warm start table. This is the point at which the Capture program finished successfully processing the log data.

User Response: No response required; this message accompanies termination.

ASN0124I The prune command was accepted; the pruning action is queued.

Explanation: The IBM Replication administrator entered the prune command and the Capture program has queued the request. The Capture program will prune the change data (CD) table and the unit-of-work (UOW) table.

User Response: No response required.

ASN0125I The current log sequence number of successfully processed log records is "<log_sequence_number>". The log timestamp is "<timestamp>".

Explanation: Capture program is processing the DB2 log at the log sequence number provided.

User Response: No action is required.

ASN0126E The Capture program encountered a syntax error. The Capture program will terminate.

Explanation: The Capture program encountered the wrong combination of invocation parameters.

User Response: Check the Capture and Apply section for your platform for more information about the START command syntax.

ASN0130I The user requested that the Capture program start reading from the end of the DB2 log.

Explanation: The user specified the WRMSKPM parameter when invoking the Capture program.

User Response: No action is required.

ASN0132I The Capture program was invoked by asncopy with the mobile option.

Explanation: This message is for your information only.

User Response: No action is required.

ASN0133I The Capture program reached the end of the mobile transactions.

Explanation: This message is for your information only.

User Response: No action is required.

ASN0134E The Capture program could not obtain the start of log information when it was invoked by asncopy with the mobile option.

Explanation: The Capture program was unable to locate the point in the log where it needed to start reading information.

User Response: Wait for subsequent messages which will provide more detailed information.

ASN0135E The trial period for the Capture program has expired.

Explanation: The trial period for the DB2 DataPropagator product has ended. You can no longer use this product until you order and install the DataPropagator licensed feature of DB2 for OS/390.

User Response: Contact the person responsible for ordering the DB2 DataPropagator product.

ASN0136I The trial version of Capture will end in *nn* days.

Explanation: You are using the trial version of DB2 DataPropagator. After *nn* days have passed, you will no longer be able to use DB2 DataPropagator unless you install the DataPropagator licensed feature of DB2 for OS/390.

User Response: None; however you might want to contact the person responsible for ordering the DB2 DataPropagator product.

ASN0137E The product registration module has unexpected content.

Explanation: The content of the registration module (ASNLPR61) for the DB2 DataPropagator feature is not as expected for this version of the DB2 DataPropagator product. No further use of the product is possible until you provide the correct registration module.

User Response: Verify that the DB2 DataPropagator feature was installed without errors. If errors occurred, correct them and try again.

If the DB2 DataPropagator feature installed without error and you are correctly accessing the feature-registration module (ASNLPR61), contact IBM customer service for assistance.

ASN0138E The product trial module has unexpected content.

Explanation: The content of the DB2 DataPropagator trial module is not as expected for this version of the DB2 DataPropagator product. No further use of the product is possible until you provide the correct trial module.

User Response: Verify that the DB2 DataPropagator feature was installed without errors. If errors occurred, correct them and try again.

If the DB2 DataPropagator feature installed without error and you are correctly accessing it, contact IBM customer service for assistance.

ASN0139E The Capture program could not open the trace file. The error is "<error_code>".

Explanation: The user specified the TRCFILE option, but the Capture program could not open the trace file. Possible reasons are:

- The directory specified in the ASNPATH environment variable is incorrect.
- The user does not have the correct authorization for the directory.

User Response: Contact your system programmer or your IBM Service Representative.

ASN0200E An incorrect parameter "<parameter>" was passed to the Capture program.

Explanation: For VM/ESA, one of the following situations caused an error:

- An incorrect parameter was specified on the ASNCCP invocation command.
- The CAPTURE ASNPARMS file contained an invalid parameter.
- An invalid parameter was specified on the :RESID tag in the RESID NAMES file for the :DBNAME. For example, the RESID could be too long.

For VSE/ESA, an invalid parameter was specified on the ASNCCP invocation command.

User Response: Verify that the parameters supplied are valid. See the Capture and Apply section for your platform for more information about the ASNCCP command.

ASN0201E The Capture program encountered a "<platform>" error. The routine name is "<routine>"; the function name is "<function>"; the return code is "<return_code>".

Explanation: On VM:

- For the LINK function, Capture program encountered an error while attempting to LINK the minidisks identified in the *database* SQLFDEF file.

database is the database identified with the SQLINIT or SQLGLOB commands, the default of SQLDBA.

- For the FSREAD, FSPOINT, or FSTATE function errors, the Capture program encountered an error while trying to read CAPTURE ASNPARMS or the *database* SQLFDEF file.
- For the XCIDRM function, Capture program was unable to obtain the resource ID it uses as a lock to ensure that only one Capture program is active for a DB2 database. The error may have occurred for the following reasons:
  - The virtual machine in which the application is running does not have authority to connect to *IDENT.
  - The virtual machine in which the application is running does not have the authority to declare the resource.

On VSE:

- For the GENCB, MODCB, OPEN, GET, CLOSE, or ENDREQ function errors, Capture programencountered an error while trying to set up or read the database log or directory.
- For the GETVIS, FREEVIS, or XPCC function errors, Capture program encountered an error while trying to perform one of these functions.

User Response: Correct the error as described in the platform documentation. On VM:

- For the LINK function, see *VM/ESA CP Command and Utility Reference* for more information about the return code.
- For the FSREAD, FSPOINT, or FSTATE function errors, see *VM/ESA CMS Application Reference - Assembler*.
- For the XCIDRM function, see *VM/ESA CPI Communications User Guide* for more information the return code.
- For other functions, refer to the platform product application development and command documentation.

On VSE:

- For the GENCB, MODCM, OPEN, GET, CLOSE, or ENDREQ function errors, see *VSE/ESA Messages and Codes Reference* for more information about the IBM VSE/VSAM macros.
- For the GETVIS, FREEVIS, or XPCC function errors, see *VSE/ESA Systems Macro Reference*.

ASN0202E The USERID parameter was not specified.

Explanation: The USERID parameter is required in the PARM= field on the EXEC job control statement that is passed to the Capture program.

User Response: Add the USERID= parameter, specifying the user ID and password, in the PARM= field and resubmit the job.

ASN0203I Linking to "<diskname>" minidisk"<diskowner>"as "<vdev>".

Explanation: The Capture program is about to issue an internal CP link command to the specified database minidisk.

User Response: If prompted, enter the minidisk password.

Apply program messages

ASN1000S An internal error occurred for message number "<number>". Its substitution fields are "<substitution_field_1>", "<substitution_field_2>", "<substitution_field_3>", "<substitution_field_4>", "<substitution_field_5>", "<substitution_field_6>", and "<substitution_field_7>". The error code is "<error_code>". The return code is "<return_code>".

Explanation: The message file for Apply was installed incorrectly.

User Response: Refer to the installation and configuration information in this book pertaining to your platform. Make sure the message file is installed in the correct directory. If it is, contact your IBM Service representative.

ASN1001E The Apply program encountered an SQL error.

Parameters:

- **ERRCODE is "<error_code>"**
- **SQLSTATE is "<sqlstate>"**
- **SQLCODE is "<sqlcode>"**
- **SQLERRM is "<sqlerrm>"**
- **SQLERRP is "<sqlerrp>"**
- **server name is "<server_name>"**
- **table name is "<table_name>"**

Explanation: An error occurred during the execution of an SQL statement.

User Response: Refer to your database messages reference for SQL.

ASN1002E The "<table_name>" could not be locked. ERRCODE is "<error_code>", SQLSTATE is "<sqlstate>", SQLCODE is "<sqlcode>", SQLERRM is "<sqlerrm>", SQLERRP is "<sqlerrp>", server name is "<server_name>", table name is "<table_name>"

Explanation: The Apply program could not lock the table.

User Response: Refer to your database messages reference.

ASN1003E The Apply program could not connect to the server "<server>".

Parameters:

- **error code is "<error_code>"**
- **SQLSTATE is "<sqlstate>"**
- **SQLCODE is "<sqlcode>"**
- **SQLERRM is "<sqlerrm>"**
- **SQLERRP is "<sqlerrp>"**

Explanation: The Apply program attempted to connect to the database and received a failing return code because either the database was not up or too many users were accessing it.

User Response: If you are running Apply under DB2 UDB for UNIX or under DataJoiner for UNIX, ensure that the LIBPATH environment variable is set to the same environment in which the Apply program starts. See "Chapter 10. Capture and Apply for UNIX platforms" on page 197 for more information.

Refer to your database messages reference for SQL.

ASN1004I The trial version of Apply will end in *nn* days.

Explanation: You are using the trial version of DB2 DataPropagator. After *nn* days have passed, you will no longer be able to use DB2 DataPropagator unless you install the

DataPropagator licensed feature of DB2 for OS/390.

User Response: None; however you might want to contact the person responsible for ordering the DB2 DataPropagator product.

ASN1005E The trial period for the Apply program has expired.

Explanation: The trial period for the DB2 DataPropagator product has ended. You can no longer use this product until you order and install the DataPropagator licensed feature of DB2 for OS/390.

User Response: Contact the person responsible for ordering the DB2 DataPropagator product.

ASN1006E The product registration module has unexpected content.

Explanation: The content of the registration module (ASNAPR61) for the DB2 DataPropagator feature is not as expected for this version of the DB2 DataPropagator product. No further use of the product is possible until you provide the correct registration module.

User Response: Verify that the DB2 DataPropagator feature was installed without errors. If errors occurred, correct them and try again.

If the DB2 DataPropagator feature installed without error and you are correctly accessing the feature-registration module (ASNAPR61), contact IBM customer service for assistance.

ASN1007E The product trial module has unexpected content.

Explanation: The content of the DB2 DataPropagator trial module is not as expected for this version of the DB2 DataPropagator product. No further use of the product is possible until you provide the correct trial module.

User Response: Verify that the DB2 DataPropagator feature was installed without errors. If errors occurred, correct them and try again.

If the DB2 DataPropagator feature installed without error and you are correctly accessing it, contact IBM customer service for assistance.

ASN1008E The subscription set with Apply qualifier "<qualifier>" and set name "<set_name>" is not defined correctly. ERRCODE is %3.

Explanation: The subscription set is not defined correctly.

User Response: Make sure that the WHOS_ON_FIRST column in ASN.IBMSNAP_SUBS_SET is specified correctly.

ASN1009E There is no subscription set defined for Apply qualifier "<qualifier>".

Explanation: There is no subscription set defined for Apply qualifier "<qualifier>".

User Response: Define at least one subscription set for Apply qualifier "<qualifier>".

ASN1010E The Apply program could not insert row "<row>" into the audit trail table due to the following error: "<error_code>".

Explanation: This is an SQL return code indicating that the audit trail table was not set up with the same structure as the table in "Chapter 21. Table structures" on page 299.

User Response: Refer to "Chapter 21. Table structures" on page 299 and your database SQL manual.

ASN1011E The copy request has incompatible source and target attributes. The error code is "<error_code>".

Explanation: This is an SQL error code indicating that the attributes of the target table must be compatible with the attributes of the source table.

User Response: Refer to the BASE_STRUCTURE column in the register table for the compatibility of the source and target attributes.

ASN1012E The source table structure is invalid. The error code is "<error_code>".

Explanation: This is an SQL return code indicating that the source table structure in the register table was not set up according to the SOURCE_STRUCTURE column in the register table.

User Response: Refer to "Chapter 21. Table structures" on page 299, the SOURCE_STRUCTURE column in the register table for valid source table structures.

ASN1013E The target table structure is invalid. The error code is "<error_code>".

Explanation: The target table structure in the subscriptions target member table (ASN.IBMSNAP_SUBS_MEMBR) was not valid.

User Response: Refer to "Chapter 21. Table structures" on page 299 for valid target table structures.

ASN1014E The Apply program could not find a source for the copy request because it could not find the change data table. The error code is "<error_code>".

Explanation: The change data table was not defined in the register table because either the Apply program did not find the change data table name in the register table or the source table was not registered correctly.

User Response: Refer to "Chapter 21. Table structures" on page 299 and verify that the change data table is correctly defined in the register table (ASN.IBMSNAP_REGISTER CD_OWNER, CD_TABLE).

ASN1015I The Apply program is waiting for the Capture program at server "<server_name>" to advance the global SYNCHTIME. Verify that the Capture program is running.

Explanation: This message is for your information only.

User Response: No action is required.

ASN1016I Refresh copying has been disabled. The error code is "<error_code>".

Explanation: While attempting to perform a full refresh, the Apply program encountered a DISABLE_REFRESH column in the register table which was set on.

User Response: Either turn off the DISABLE_REFRESH column or bypass the Apply program and perform a manual refresh.

ASN1017E Apply could not find any target column names. The error code is "<error_code>".

Explanation: Apply could not find any columns in the ASN.IBMSNAP_SUBS_COLS subscription columns table.

User Response: Redefine the subscription set and subscription-set members (see "Chapter 6. Setting up your replication environment" on page 83 for instructions).

ASN1018I The Apply program is processing subscription set "<set_name>"("<whos_on_first>"). ("<set_number>" of "<total_sets>").

Explanation: This message is for your information only.

User Response: No action is required.

ASN1019E The target table does not have any key columns. The error code is "<error_code>".

Explanation: The Apply program could not find key column names in one of the columns requiring a primary key.

User Response: Redefine the subscription set and the subscription-set members (see "Chapter 6. Setting up your replication environment" on page 83 for instructions).

ASN1020S The Apply program could not reserve a storage block. The error code is "<error_code>".

Explanation: The Apply program could not obtain the required (memory) storage.

User Response: Contact your IBM Service representative.

ASN1021S The Apply program could not read the work file. The error code is "<error_code>".

Explanation: The Apply program could not read the work file due to a system error.

User Response: Determine if the problem is caused by lack of space and contact your system administrator to obtain what is needed.

ASN1022S The Apply program could not write into the work file. The error code is "<error_code>".

Explanation: Either the user does not have the proper access authority for one or all of the files or not enough space is left after writing to the target file.

User Response: Determine whether the problem is caused by a lack of access authority or a lack of space and contact your system administrator to obtain what is needed.

ASN1023S The Apply program could not open the work file. The error code is "<error_code>".

Explanation: The Apply program could not open the work file.

User Response: Contact your IBM Service representative.

ASN1024S The Apply program could not close the work file. The error code is "<error_code>".

Explanation: The Apply program could not close the work file.

User Response: Contact your IBM Service representative.

ASN1025I The Apply program completed processing for subscription set "<set_name>"("<whos_on_first>"). The return code is "<return_code>".

Explanation: This message is for your information only.

User Response: No action is required.

ASN1026I The Apply program encountered an error while trying to bind. SQLSTATE is "<sqlstate>", SQLCODE is "<sqlcode>".

Explanation: An error occurred during the execution of bind.

User Response: Refer to your database messages reference.

ASN1029E The SQL statement could not execute. The error code is "<error_code>".

Explanation: The execution of the SQL statement specified by the user was not successful.

User Response: Refer to the SQLSTATE, SQLCODE, SQLERRO, and SQLERRM in the apply trail table and your database SQL manual for detailed information.

ASN1030S The Apply program encountered an OS/2 error. The error code is "<error_code>"; the return code is "<return_code>".

Explanation: The execution of an OS/2 API failed.

User Response: For more information on the return code, refer to the *OS/2 WARP Control Program Programming Reference*.

ASN1031E The SQL statement is empty. The error code is "<error_code>".

Explanation: The SQL statement is an empty string.

User Response: Specify the SQL statement to be executed.

ASN1032S The Apply program log file could not be opened. The error code is "<error_code>"; the return code is "<return_code>".

Explanation: The Apply program could not open the log file.

User Response: For more information on the return code, either refer to the *OS/2 WARP Control Program Programming Reference* or to the system library information for your particular platform.

ASN1033E The Apply program could not write to the Apply log file. The error code is "<error_code>"; the return code is "<return_code>".

Explanation: The Apply program could not write to the log file.

User Response: For more information on the return code, either refer to the *OS/2 WARP Control Program Programming Reference* or to the system library information for your particular platform.

ASN1034E Stored procedures are not supported in DB2 for MVS/ESA V3. The error code is "<error_code>".

Explanation: DB2, Version 3 does not support the stored procedure call.

User Response: Remove the stored procedure CALL statement from the statement table (ASN.IBMSNAP_SUBS_STMT).

ASN1035E The Apply program could not access the subscription columns table.

Parameters:

- **error code is "<error_code>"**
- **SQLSTATE is "<sqlstate>"**
- **SQLCODE is "<sqlcode>"**
- **SQLERRM is "<sqlerrm>"**
- **SQLERRP is "<sqlerrp>"**
- **server name is "<server_name>"**
- **table name is "<table_name>"**

Explanation: An error occurred during the execution of an SQL statement.

User Response: Refer to your database messages reference for SQL.

ASN1036E The column type "<col_type>" for expression "<expression>" is invalid. The error code is "<error_code>".

Explanation: The value for the COL_TYPE column in the subscription columns table is invalid.

User Response: Change the value to A, B, C, F, or R.

ASN1037E The Apply program could not obtain the date and time. The error code is "<error_code>"; the return code is "<return_code>".

Explanation: The OS/2 API DosGetDateTime failed.

User Response: For more information on the return code, refer to the *OS/2 WARP Control Program Programming Reference*.

ASN1038E No column names or expressions were specified in the subscription columns table.

Explanation: Column names or expressions for a copy statement must be specified.

User Response: Refer to "Chapter 6. Setting up your replication environment" on page 83 for more information about requirements for subscription definitions.

ASN1039S The Apply program plan, "<plan_name>", could not be opened.

Parameters:

- **error code is "<error_code>"**
- **return code is "<return_code>"**
- **reason code is "<reason_code>"**

Explanation: The Apply program plan could not be opened.

User Response: Refer to the *Apply for OS/390 Program Directory*.

ASN1040S The Apply program encountered an OS/390 error. The error code is "<error_code>"; the return code is "<return_code>".

Explanation: Execution of an OS/390 system operation failed.

User Response: Refer to your OS/390 system library information.

ASN1041I The Apply program was started using subsystem name: "<subsystem>".

Explanation: This is not an error message, however, you should make sure that the displayed subsystem name is valid.

User Response: Verify that the subsystem name is valid.

ASN1042W There are too many invocation parameters.

Explanation: The number of parameters you specified when you invoked the Apply program exceeds the maximum allowed.

User Response: Refer to the Capture and Apply section for your platform for information on the appropriate number of invocation parameters.

ASN1043E There is already one Apply instance running with this Apply program qualifier "<qualifier>". The error code is "<error_code>"; the reason code is "<reason_code>".

Explanation: Verification attempt failed.

User Response: Make sure that only one instance of the Apply program is running under this user ID on this subsystem or database.

ASN1044I The Apply program will become inactive for "<number>" minutes and "<number>" seconds.

Explanation: This message is for your information only.

User Response: No action is required.

ASN1045I The Apply program was started using database "<database>".

Explanation: This message is for your information only.

User Response: No action is required unless this is not the intended database.

ASN1046S The Apply program libraries are not authorized for the Authorized Program Facility (APF).

Explanation: The Apply program libraries must be APF authorized.

User Response: Authorize the Apply libraries.

ASN1048E The execution of a copy statement failed. See the Apply trail table for full details: "<text>"

Explanation: A copy statement could not execute. In the message, "<text>"identifies the "<copy_server>", "<copy_owner, copy_table, stmt_number>", and "<cntl_server>".

User Response: Check the APPERRM fields in the audit trail table to determine why the copy statement failed.

ASN1049S The Apply program encountered a system error. The error code is "<error_code>". The return code is "<return_code>".

Explanation: Execution of a system operation failed.

User Response: Refer to the system library information for your particular platform.

ASN1050E The Apply program encountered an invalid operation while updating the target table. The error code is "<error_code>". The invalid operation to be applied is "<operation>".

Explanation: The operation field of a row fetched from the source table is not valid.

User Response: Contact your IBM Service representative.

ASN1051E The Apply program detected a gap between the source "<source>" table and the target table. The error code is "<error_code>".

Explanation: The Apply program has detected that the Capture program had lost change data before the Apply program could copy it. For example, the Capture program may have been cold started.

User Response: Check the control tables to determine why the gap is present. Take proper action to preserve data integrity before you reset the control table information to execute the definition again.

ASN1052E The Apply program could not find the ASNLOAD program.

Explanation: The Apply program cannot find the ASNLOAD program in the current directory.

User Response: Make sure that ASNLOAD is in the directory from which you are invoking the Apply program.

ASN1053E The execution of the ASNLOAD program failed. The return code is "<return_code>".

Explanation: The ASNLOAD program detected an error.

User Response: Refer to the messages files generated by the EXPORT and IMPORT utilities. Note that these files names are different for Apply for OS/2 and Apply for AIX.

ASN1054S The Apply program could not find the registration information for source owner "<src_ownr>", source table "<src_tbl>", and source view qualifier "<src_view_qual>".

Explanation: The source table registration is incorrect or incomplete.

User Response: Drop the registration and redo it. Also make sure that the registration information is in both the register table and the pruning control table.

ASN1055S The Apply program could not find the prune control information for source owner "<src_ownr>", source table "<src_tbl>", source view qualifier "<src_view_qual>", target owner "<tgt_ownr>", and target table "<tgt_tbl>".

Explanation: The source table registration is incorrect.

User Response: Drop the subscription and redo it.

ASN1056E The Apply program could not connect to the server due to lack of user ID/password. The error code is "<error_code>".

Explanation: The Apply program could not find the password and user ID to connect to the server.

User Response: Make sure that the Apply program password exists. If you are using DB2 Universal Database Satellite Edition, make sure that the password and user ID are defined to the client systems.

ASN1057E The Apply program could not read the password in the Apply password file. The error code is "<error_code>".

Explanation: The Apply program found no password.

User Response: If you want to use the AUTHENTICATION=SERVER scheme, you must provide a password as described in the Apply program section in the Capture and Apply chapter for your platform.

ASN1058E The Apply program could not close the password file. The error code is "<error_code>".

Explanation: The Apply program could not close the password file.

User Response: Contact your IBM Service representative.

ASN1059E The Apply program detected invalid syntax for line "<line>" in the password file. The error code is "<error_code>".

Explanation: The Apply program could not recognize a line in the password file.

User Response: Correct the syntax error in the password file. See the Apply program section in the Capture and Apply chapter for your platform.

ASN1060E The dynamic allocation for the temporary work file failed. The error code is "<error_code>".

Explanation: A system error was encountered during dynamic allocation.

User Response: Contact your IBM Service representative.

ASN1061E An invalid keyword parameter was specified. The error code is "<error_code>".

Explanation: An invalid invocation parameter was specified and ignored by the Apply program.

User Response: Correct the invocation parameter. See the Apply program section in the Capture and Apply chapter for your platform.

ASN1063E A subscription set cannot have more than 200 members. The error code is "<error_code>".

Explanation: The number of subscriptions has exceeded the maximum allowed number of 200.

User Response: Remove excess members from the subscription.

ASN1066S An internal Apply program error occurred. The error code is "<error_code>".

Explanation: An internal Apply error occurred.

User Response: Contact your IBM Service representative.

ASN1067E The Apply program has detected update conflicts and compensated rejected transactions. See the unit-of-work table for details. The error code is "<error_code>".

Explanation: More than one application updated the same row in a table from different locations. Some transactions have been rejected and compensated.

User Response: See the ASN.IBMSNAP_UOW table for details.

ASN1068E The Apply program has deactivated the subscription due to an RI violation. The error code is "<error_code>".

Explanation: A referential integrity violation was detected when copying data from the source table to a replica. The Apply program has terminated and the subscription has been deactivated.

User Response: Correct the referential integrity error and reactivate the subscription.

ASN1069E The Apply program has deactivated the subscription due to an RI violation. All the affected units-of-works have been marked in the unit-of-work table and compensated. The error code is "<error_code>".

Explanation: A referential integrity violation was detected when replicating data from the replica to the user table. The Apply program has terminated and the subscription has been deactivated.

User Response: Correct the referential integrity error and reactivate the subscription.

ASN1070E The Apply program could not lock the target table.

Parameters:

- **ERRCODE is "<error_code>"**
- **SQLSTATE is "<sqlstate>"**
- **SQLCODE is "<sqlcode>"**
- **SQLERRM is "<sqlerrm>"**
- **SQLERRP is "<sqlerrp>"**
- **server name is "<server_name>"**
- **table name is "<table_name>"**

Explanation: The Apply program could not lock the target tables before it was to check update conflicts.

User Response: Verify that all the target tables are available before rerunning Apply.

ASN1071E The Apply program has detected an error while reading the temporary work file. The error code is "<error_code>".

Explanation: The Apply program has detected an error while reading the temporary work file.

User Response: Contact your IBM Service representative.

ASN1072E The Apply program could not find the ASNDONE program.

Explanation: The Apply program could not find the user exit program, ASNDONE.

User Response: Verify that the ASNDONE program is located in the correct directory.

ASN1073E The execution of the ASNDONE program failed. The return code is "<return_code>".

Explanation: An error occurred while calling the user exit, ASNDONE.

User Response: Contact your IBM Service representative.

ASN1097I The Apply program stopped due to the above error.

Explanation: The error reported previously caused the Apply program to stop.

User Response: Fix the error reported before this message.

ASN1100I A user has stopped the Apply program.

Explanation: A user issued the STOP command to stop the Apply program.

User Response: No action is required.

ASN1109I Not all of the Jet database changes are applied due to an RI violation.

Explanation: There was at least one change in the row-replica target list table that violates the referential integrity (RI) of the source table.

User Response: Refer to the IBMSNAP_ERROR_INFO and IBMSNAP_ERROR_MESSAGE tables for more details.

ASN1110I The Apply program created Jet database "<db_name>".

Explanation: The target database <db_name> was created.

User Response: No action is required.

ASN1111I The Apply program converted Jet Database "<db_name>" to a Design Master.

Explanation: The database that you specified is now a Design Master from which all Microsoft Jet Replicas will be created.

User Response: No action is required.

ASN1115I ODBC call was successful with sqlcode "<sqlcode>", sqlstate "<sqlstate>", and message "<message>".

Explanation: The ODBC call was successful, but a message was issued. This message is for your information only.

User Response: No action is required.

ASN1116E ODBC call failed. sqlcode "<sqlcode>", sqlstate "<sqlstate>", and message "<message>".

Explanation: An error occurred during the execution of an ODBC operation against either the DB2 ODBC driver or the MS Jet ODBC driver.

User Response: Refer to the appropriate ODBC reference for more information.

ASN1130E Execution of DAO call failed. ERRCODE "<error_code>", DAO error number "<error_number>", and DAO error message "<error_message>".

Explanation: An error occurred during a Microsoft Data Access Object (DAO) execution.

User Response: Refer to the Microsoft DAO reference for more information.

ASN1135E File operation failed. File name is "<file_name>", error code is "<error_code>".

Explanation: Open, close, read, or write operations failed.

User Response: Verify that the user has authority for the file operation. Also verify that there is enough space in the system.

ASN1200I The asncopy program completed.

Explanation: This message is for your information only.

User Response: No action is required.

ASN1201S Place holder for generic message - internal error

Explanation: The asncopy program encountered an SQL error.

Parameters:

- ERRCODE is "<error_code>"
- SQLSTATE is "<sqlstate>"
- SQLCODE is "<sqlcode>"
- SQLERRM is "<sqlerrm>"
- SQLERRP is "<sqlerrp>"
- server name is "<server_name>"
- table name is "<table_name>"

User Response: Refer to your database messages reference for SQL.

ASN1202E The asncopy program encountered an SQL error. ERRCODE is "<error code>", SQLSTATE is "<sqlstate>", SQLCODE is "<sqlcode>", SQLERRM is "<sqlerrm>", SQLERRP is "<sqlerrp>", table name is "<table name>".

Explanation: This message is for your information only.

User Response: No action is required.

ASN1203I The asncopy program was stopped by the user.

Explanation: This message is for your information only.

User Response: No action is required.

ASN1204E The asncopy program encountered an incorrect keyword. The keyword is "<keyword>".

Explanation: A keyword was entered incorrectly.

User Response: Execute the command again, using the correct keyword.

ASN1205E The asncopy program terminated due to a Capture program error.

Explanation: An inconsistency in Capture program executions has caused the asncopy program to end.

User Response: Refer to the trace produced by the Capture program (ASN.IBMSNAP_TRACE) or the asncopy program error log to determine the cause of the error.

ASN1206E The asncopy program terminated due to an Apply program error.

Explanation: An inconsistency in Apply program executions has caused the asncopy program to end.

User Response: Refer to the apply trail table or the asncopy program error log to determine the cause of the error.

ASN1207E The subscription for "<subscription>" was not activated.

Explanation: The selected subscription is inactive.

User Response: Either activate the subscription or select another one.

ASN1208E The asncopy program could not find the subscription for set "<set>".

Explanation: The selected subscription does not exist.

User Response: Enter the correct subscription.

ASN1209E The asncopy program could not find any eligible subscription.

Explanation: Either no subscription name was specified or the names specified are invalid.

User Response: Check the subscription names and be sure to enter the correct ones.

ASN1210E An Apply qualifier must be specified following the keyword -q.

Explanation: You must specify an Apply qualifier following the keyword -q.

User Response: Specify an Apply qualifier following the keyword -q.

ASN1211E Set names must be specified following the keyword "<keyword>".

Explanation: You must specify the set names following the keyword (O, U, D, or S).

User Response: Reinitiate the asncopy program, specifying the keyword and then the set names.

ASN1212E A read-only set name "<set_name>" is found following the keyword "<keyword>".

Explanation: A read-only set name was specified following the keyword U or D.

User Response: Specify only replica for the keywords U and D.

ASN1214E The set name "<set_name>" is specified more than once.

Explanation: The same set name cannot be specified in more than one list.

User Response: Reinitiate the asncopy program, being sure to specify each set name only once among all lists.

ASN1221I Set "<set_name>" has been successfully refreshed with "<number>"rows at "<time>".

Explanation: This message is for your information only.

User Response: No action is required.

ASN1222I Set ″<set_name>″ has successfully inserted ″<number>″ rows, deleted ″<number>″ rows, updated ″<number>″ rows at ″<time>″.

Explanation: This message is for your information only.

User Response: No action is required.

ASN1223E The Apply program could not copy for set "<set_name>".

Explanation: The Apply program encountered a problem while copying.

User Response: Refer to the apply trail table or the asncopy program error log to determine the cause of the error.

ASN1230S The asncopy program encountered a system error. The error code is "<error_code>"and the return code is "<return_code>".

Explanation: The asncopy program encountered an error in the database.

User Response: Trace the error and call your IBM Service representative.

ASN1240E A system error has been detected. The error code is "<error_code>", return code is "<return_code>".

Explanation: The asncopy program encountered an error in the database.

User Response: Trace the error and call your IBM Service representative.

ASN1242E A SQL error occurred. ERRCODE is "<error_code>", SQLSTATE is ″<sqlstate>″, SQLCODE is ″<sqlcode>″, SQLERRM is ″<sqlerrm>″, SQLERRP is ″<sqlerrp>″, table name is ″<table_name>″.

Explanation: This message is for your information only.

User Response: No action is required.

ASN1243E There is no eligible subscription in the ASN.IBMSNAP_SUBS_SET table.

Explanation: Either a subscription has not been selected or the apply qualifier is invalid.

User Response: Verify the subscription names and apply qualifier.

ASN1244E User has not selected any set.

Explanation: A subscription set has not been selected from the ASNMOBIL dialog.

User Response: Select at least one set from ASNMOBIL dialog.

ASN1303E The ASNSAT program encountered an incorrect invocation keyword. The keyword is "<keyword>".

Explanation: An unknown keyword parameter was specified.

User Response: Specify the correct keyword parameter.

ASN1304E The ASNSAT program terminated due to a Capture error.

Explanation: The Capture program returned an error.

User Response: Determine the error from the Capture log file.

ASN1305E The ASNSAT program terminated due to an Apply error.

Explanation: The Apply program returned an error.

User Response: Determine the error from the Apply log file.

ASN1309E Cannot find default control database name from the directory.

Explanation: The user did not specify the control server name, and the Apply program could not find the default control server name from the directory.

User Response: Specify the control server name following the -n keyword.

ASN1310E The ASNSAT program encountered a system error while attempting to invoke the Capture program. Return code is "<return_code>".

Explanation: An operating system error occurred while calling asnccp.

User Response: Make sure that the Capture program is in the execution path.

ASN1311E The ASNSAT program encountered a system error while attempting to invoke the Apply program. Return code is "<return_code>".

Explanation: An operating system error occurred while calling asnapply.

User Response: Make sure that the Apply program is in the execution path.

ASN1312E The default target server, DB2DBDFT, is not set.

Explanation: The user did not specify the target server name, and the ASNSAT program could not determine the default database name from DB2DBDFT.

User Response: Specify the target server name following the -t keyword.

ASN1314E An SQL error occurred while ASNSAT was getting the default Apply qualifier. SQLSTATE is “<sqlstate>”, SQLCODE is “<sqlcode>”.

Explanation: The user did not specify the Apply qualifier. The ASNSAT program encountered an error while retrieving the USER special register.

User Response: Specify the Apply qualifier following the -q keyword.

ASN1315E Cannot connect to database server. SQLSTATE is “<sqlstate>”, SQLCODE is “<sqlcode>”.

Explanation: An error occurred while attempting to connect to the target database.

User Response: Refer to your database messages reference.

ASN1316E ASNSAT encountered an error while trying to bind. The SQLSTATE is “<sqlstate>”, SQLCODE is “<sqlcode>”.

Explanation: An error occurred while attempting to auto bind.

User Response: Make sure that the bind file exists in the sqllib\bnd directory.

ASN1317E An SQL error occurred while ASNSAT was getting the CD_TABLE value from ASN.IBMSNAP_REGISTER table. SQLSTATE is “<sqlstate>”, SQLCODE is “<sqlcode>”.

Explanation: An SQL error occurred while selecting from the register table.

User Response: Refer to your database messages reference.

ASN1318E An SQL error occurred while ASNSAT attempted to get the DB2 node type. SQLSTATE is “<sqlstate>”, SQLCODE is “<sqlcode>”.

Explanation: An error occurred while retrieving the node type configuration parameter.

User Response: Refer to your database messages reference.

Capture for AS/400 messages

ASN200A User table &1 registration not satisfied. The registration probably should be removed.

Explanation: There are many reasons that data capturing for a replication source cannot continue. Depending on the severity, you can receive either an ASN2004 or an ASN200A message.

Because a journal job is usually responsible for capturing data from several replication sources, the journal job is not affected by these messages. These messages were generated due to a specific replication source. After sending the ASN2004 or ASN200A message, the Capture program continues processing other replication sources. The program ends only if the last remaining replication source it is processing results in an ASN2004 or ASN200A message.

User Response: Use the DSPMSGD command to determine the conditions that caused this message. For example:

```
DSPMSGD ASN200A QDPR/QDPRMSG
```

ASN2004 User table &1 registration not satisfied.

Explanation: See ASN200A.

User Response: Check the job log of the job that sent the message. Correct the problem and try the request again.

ASN2017 Starting point not found for journal &1 (C I G R).

Explanation: One of the first tasks in the Capture control job is to establish the starting point with which to resume journal entry processing. The journal receiver containing entries corresponding to that starting point must be online. If that receiver is deleted prematurely, the control job sends message ASN2017 to ask you how you want to proceed.

Response **I** tells the control job to ignore the fact that a receiver is missing. Data capturing resumes using the current receiver chain. If you response **I**, you are responsible for the integrity of the replication.

Response **C** cancels the Capture control job.

Response **R** retries establishing the starting point.

User Response: In most cases, a response of **G** is appropriate. (Cold start all the replication sources using this journal.)

ASN2028 Internal error in Capture program.

Explanation: The Capture program detected an internal error and posted a reason code.

User Response: Record the reason code and contact your system administrator.

ASN2039 Lag limit exceeded.

Explanation: Exceeding the lag limit causes the Capture component to send an ASN2039 message to the job log and to the system operator message queue.

User Response: By gathering system performance data, you can better determine what action might be necessary. Some possible solutions are:

- To increase the lag limit.
- To prioritize (by using a lower numerical value) the Capture jobs to a level identical to or higher than that of the interactive jobs.
- To reschedule the workload to a time period when the demand on the system is less, or to move some of the workload to a different system.
- To add more resources to the system (a system upgrade).

ASN2201 Internal error in Capture program.

Explanation: An error occurred in the Capture program.

User Response: Check the job log to determine the cause of the problem. Record the reason code and contact your system administrator.

ASN2301 Internal error in Capture program.

Explanation: This message is sent by the journal job as an escape message before the journal job ends. A condition exists that makes it impossible to continue capturing data.

User Response: Check the job log to determine the cause of the problem. To aid in problem determination, use the DSPMSGD command to determine the conditions that caused this message to be displayed. For example:

```
DSPMSGD ASN2301 QDPR/QDPRMSG
```

Record the reason code and contact your system administrator.

Part 7. Appendixes

Appendix A. How the DB2 library is structured

The DB2 Universal Database library consists of SmartGuides, online help, books and sample programs in HTML format. This section describes the information that is provided, and how to access it.

To access product information online, you can use the Information Center. You can view task information, DB2 books, troubleshooting information, sample programs, and DB2 information on the Web. See "Accessing information with the Information Center" on page 414 for details.

Completing tasks with SmartGuides

SmartGuides help you complete some administration tasks by taking you through each task one step at a time. SmartGuides are available through the Control Center and the Client Configuration Assistant. The following table lists the SmartGuides.

Note: Create Database, Index, and Configure Multisite Update SmartGuide are available for the partitioned database environment.

SmartGuide	Helps You to...	How to Access...
Add Database	Catalog a database on a client workstation.	From the Client Configuration Assistant, click **Add**.
Back up Database	Determine, create, and schedule a backup plan.	From the Control Center, click with the right mouse button on the database you want to back up and select **Backup**->**Database using SmartGuide**.
Configure Multisite Update SmartGuide	Perform a multi-site update, a distributed transaction, or a two-phase commit.	From the Control Center, click with the right mouse button on the **Database** icon and select **Multisite Update**.
Create Database	Create a database, and perform some basic configuration tasks.	From the Control Center, click with the right mouse button on the **Databases** icon and select **Create**->**Database using SmartGuide**.

SmartGuide	Helps You to...	How to Access...
Create Table	Select basic data types, and create a primary key for the table.	From the Control Center, click with the right mouse button on the **Tables** icon and select **Create**->**Table using SmartGuide**.
Create Table Space	Create a new table space.	From the Control Center, click with the right mouse button on the **Table spaces** icon and select **Create**->**Table space using SmartGuide**.
Index	Advise which indexes to create and drop for all your queries.	From the Control Center, click with the right mouse button on the **Index** icon and select **Create**->**Index using SmartGuide**.
Performance Configuration	Tune the performance of a database by updating configuration parameters to match your business requirements.	From the Control Center, click with the right mouse button on the database you want to tune and select **Configure using SmartGuide**.
Restore Database	Recover a database after a failure. It helps you understand which backup to use, and which logs to replay.	From the Control Center, click with the right mouse button on the database you want to restore and select **Restore**->**Database using SmartGuide**.

Accessing online help

Online help is available with all DB2 components. The following table describes the various types of help. You can also access DB2 information through the Information Center. For information see "Accessing information with the Information Center" on page 414.

Type of Help	Contents	How to Access...
Command Help	Explains the syntax of commands in the command line processor.	From the command line processor in interactive mode, enter: ? *command* where *command* is a keyword or the entire command. For example, `? catalog` displays help for all the CATALOG commands, while `? catalog database` displays help for the CATALOG DATABASE command.

Type of Help	Contents	How to Access...
Control Center Help *Client Configuration Assistant Help* *Event Analyzer Help* *Command Center Help*	Explains the tasks you can perform in a window or notebook. The help includes prerequisite information you need to know, and describes how to use the window or notebook controls.	From a window or notebook, click the **Help** push button or press the F1 key.
Message Help	Describes the cause of a message, and any action you should take.	From the command line processor in interactive mode, enter: ? *XXXnnnnn* where *XXXnnnnn* is a valid message identifier. For example, `? SQL30081` displays help about the SQL30081 message. To view message help one screen at a time, enter: ? *XXXnnnnn* | `more` To save message help in a file, enter: ? *XXXnnnnn* > *filename.ext* where *filename.ext* is the file where you want to save the message help.
SQL Help	Explains the syntax of SQL statements.	From the command line processor in interactive mode, enter: `help` *statement* where *statement* is an SQL statement. For example, **help** SELECT displays help about the SELECT statement. **Note:** SQL help is not available on UNIX-based platforms.
SQLSTATE Help	Explains SQL states and class codes.	From the command line processor in interactive mode, enter: **?** *sqlstate* or **?** *class-code* where *sqlstate* is a valid five-digit SQL state and *class-code* is the first two digits of the SQL state. For example, `? 08003` displays help for the 08003 SQL state, while `? 08` displays help for the 08 class code.

DB2 information – hardcopy and online

The table in this section lists the DB2 books. They are divided into two groups:

Cross-platform books
These books contain the common DB2 information for all platforms.

Platform-specific books
These books are for DB2 on a specific platform. For example, there are separate *Quick Beginnings* books for DB2 on OS/2, on Windows NT, and on the UNIX-based platforms.

Cross-platform sample programs in HTML
These samples are the HTML version of the sample programs that are installed with the SDK. They are for informational purposes and do not replace the actual programs.

Most books are available in HTML and PostScript format, or you can choose to order a hardcopy from IBM. The exceptions are noted in the table.

On OS/2 and Windows platforms, HTML documentation files can be installed under the doc\html subdirectory. Depending on the language of your system, some files may be in that language, and the remainder are in English.

On UNIX platforms, you can install multiple language versions of the HTML documentation files under the doc/%L/html subdirectories. Any documentation that is not available in a national language is shown in English.

You can obtain DB2 books and access information in a variety of different ways:

View	See “Viewing online information” on page 413.
Search	See “Searching online information” on page 416.
Print	See “Printing the PostScript books” on page 416.
Order	See “Ordering the printed books” on page 417.

Name	Description	Form Number File Name for Online Book	HTML Directory
	Cross-Platform Books		

Name	Description	Form Number File Name for Online Book	HTML Directory
Administration Guide	*Administration Guide, Design and Implementation* contains information required to design, implement, and maintain a database. It also describes database access using the Control Center(whether local or in a client/server environment), auditing, database recovery, distributed database support, and high availability. *Administration Guide, Performance* contains information that focuses on the database environment, such as application performance evaluation and tuning. You can order both volumes of the *Administration Guide* in the English language in North America using the form number SBOF-8922.	Volume 1 SC09-2839 db2d1x60 Volume 2 SC09-2840 db2d2x60	db2d0
Administrative API Reference	Describes the DB2 application programming interfaces (APIs) and data structures you can use to manage your databases. Explains how to call APIs from your applications.	SC09-2841 db2b0x60	db2b0
Application Building Guide	Provides environment setup information and step-by-step instructions about how to compile, link, and run DB2 applications on Windows, OS/2, and UNIX-based platforms. This book combines the *Building Applications* books for the OS/2, Windows, and UNIX-based environments.	SC09-2842 db2axx60	db2ax
APPC, CPI-C and SNA Sense Codes	Provides general information about APPC, CPI-C, and SNA sense codes that you may encounter when using DB2 Universal Database products. **Note:** Available in HTML format only.	No form number db2apx60	db2ap

Name	Description	Form Number File Name for Online Book	HTML Directory
Application Development Guide	Explains how to develop applications that access DB2 databases using embedded SQL or JDBC, how to write stored procedures, user-defined types, user-defined functions, and how to use triggers. It also discusses programming techniques and performance considerations. This book was formerly known as the *Embedded SQL Programming Guide*.	SC09-2845 db2a0x60	db2a0
CLI Guide and Reference	Explains how to develop applications that access DB2 databases using the DB2 Call Level Interface, a callable SQL interface that is compatible with the Microsoft ODBC specification.	SC09-2843 db2l0x60	db2l0
Command Reference	Explains how to use the command line processor, and describes the DB2 commands you can use to manage your database.	SC09-2844 db2n0x60	db2n0
Data Movement Utilities Guide and Reference	Explains how to use the Load, Import, Export, Autoloader, and Data Propogation utilities to work with the data in the database.	SC09-2858 db2dmx60	db2dm
DB2 Connect Personal Edition Quick Beginnings	Provides planning, installing, and configuring information for DB2 Connect Personal Edition.	GC09-2830 db2c1x60	db2c1
DB2 Connect User's Guide	Provides concepts, programming and general usage information about the DB2 Connect products.	SC09-2838 db2c0x60	db2c0
Connectivity Supplement	Provides setup and reference information on how to use DB2 for AS/400, DB2 for OS/390, DB2 for MVS, or DB2 for VM as DRDA application requesters with DB2 Universal Database servers, and on how to use DRDA application servers with DB2 Connect application requesters. **Note:** Available in HTML and PostScript formats only.	No form number db2h1x60	db2h1
Glossary	Provides a comprehensive list of all DB2 terms and definitions. **Note:** Available in HTML format only.	No form number db2t0x50	db2t0

Name	Description	Form Number File Name for Online Book	HTML Directory
Installation and Configuration Supplement	Guides you through the planning, installation, and set up of platform-specific DB2 clients. This supplement contains information on binding, setting up client and server communications, DB2 GUI tools, DRDA AS, distributed installation, and the configuration of distributed requests and access methods to heterogeneous data sources.	GC09-2857 db2iyx60	db2iy
Message Reference	Lists messages and codes issued by DB2, and describes the actions you should take.	GC09-2846 db2m0x60	db2m0
DB2 Replication Guide and Reference	Provides planning, configuration, administeration, and usage information for the IBM Replication tools supplied with DB2.	SC26-9642-00 db2e0x60	db2e0
SQL Getting Started	Introduces SQL concepts, and provides examples for many constructs and tasks.	SC09-2856 db2y0x60	db2y0
SQL Reference, Volume 1 and Volume 2	Describes SQL syntax, semantics, and the rules of the language. Also includes information about release-to-release incompatibilities, product limits, and catalog views. You can order both volumes of the *SQL Reference* in the English language in North America with the form number SBOF-8923.	SBOF-8923 Volume 1 db2s1x60 Volume 2 db2s2x60	db2s0
System Monitor Guide and Reference	Describes how to collect different kinds of information about databases and the database manager. Explains how to use the information to understand database activity, improve performance, and determine the cause of problems.	SC09-2849 db2f0x60	db2f0
Troubleshooting Guide	Helps you determine the source of errors, recover from problems, and use diagnostic tools in consultation with DB2 Customer Service.	S10J-8169	db2p0

Name	Description	Form Number File Name for Online Book	HTML Directory
What's New	Describes the new features, functions, and enhancements in DB2 Universal Database, Version 6.0, including information about Java-based tools.	SC09-2851 db2q0x60	db2q0
	Platform-Specific Books		
Administering Satellites Guide and Reference	Provides planning, configuration, administeration, and usage information for satellites.	GC09-2821 db2dsx60	db2ds
DB2 Personal Edition Quick Beginnings	Provides planning, installation, migration, and configuration information for DB2 Universal Database Personal Edition on the OS/2, Windows 95, and Windows NT operating systems.	GC09-2831 db2i1x60	db2i1
DB2 for OS/2 Quick Beginnings	Provides planning, installation, migration, and configuration information for DB2 Universal Database on the OS/2 operating system. Also contains installing and setup information for many supported clients.	GC09-2834 db2i2x60	db2i2
DB2 for UNIX Quick Beginnings	Provides planning, installation, migration, and configuration information for DB2 Universal Database on UNIX-based platforms. Also contains installing and setup information for many supported clients.	GC09-2836 db2ixx60	db2ix
DB2 for Windows NT Quick Beginnings	Provides planning, installation, migration, and configuration information for DB2 Universal Database on the Windows NT operating system. Also contains installing and setup information for many supported clients.	GC09-2835 db2i6x60	db2i6
DB2 Enterprise - Extended Edition for UNIX Quick Beginnings	Provides planning, installation, and configuration information for DB2 Extended Enterprise Edition for UNIX. Also contains installing and setup information for many supported clients.	GC09-2832 db2v3x60	db2v3

Name	**Description**	**Form Number** **File Name for Online Book**	**HTML Directory**
DB2 Enterprise - Extended Edition for Windows NT Quick Beginnings	Provides planning, installation, and configuration information for DB2 Extended Enterprise Edition for Windows NT. Also contains installing and setup information for many supported clients.	GC09-2833 db2v6x60	db2v6
DB2 Connect Enterprise Edition for OS/2 and Windows NT Quick Beginnings	Provides planning, migration, installation, and configuration information for DB2 Connect Enterprise Edition on the OS/2 and Windows NT operating systems. Also contains installation and setup information for many supported clients. This book was formerly part of the *DB2 Connect Enterprise Edition Quick Beginnings.*	GC09-2828 db2c6x60	db2c6
DB2 Connect Enterprise Edition for UNIX Quick Beginnings	Provides planning, migration, installation, configuration, and usage information for DB2 Connect Enterprise Edition in UNIX-based platforms. Also contains installation and setup information for many supported clients. This book was formerly part of the *DB2 Connect Enterprise Edition Quick Beginnings.*	GC09-2829 db2cyx60	db2cy
DB2 Data Links Manager for AIX Quick Beginnings	Provides planning, installation, configuration, and task information for DB2 Data Links Manager for AIX.	GC09-2837 db2z0x60	db2z0
DB2 Data Links Manager for Windows NT Quick Beginnings	Provides planning, installation, configuration, and task information for DB2 Data Links Manager for Windows NT.	GC09-2827 db2z6x60	db2z6
DB2 Query Patrol Administration Guide	Provides administration information on DB2 Query Patrol.	SC09-2859 db2dwx60	db2dw
DB2 Query Patrol Installation Guide	Provides installation information on DB2 Query Patrol.	SC09-2860 db2iwx60	db2iw
DB2 Query Patrol User's Guide	Describes how to use the tools and functions of the DB2 Query Patrol.	SC09-2861 db2wwx60	db2ww

Name	Description	Form Number File Name for Online Book	HTML Directory
	Cross-Platform Sample Programs in HTML		
Sample programs in HTML	Provides the sample programs in HTML format for the programming languages on all platforms supported by DB2 for informational purposes (not all samples are available in all languages). Only available when the SDK is installed. See *Application Building Guide* for more information on the actual programs. **Note:** Available in HTML format only.	No form number	db2hs/c db2hs/cli db2hs/clp db2hs/cpp db2hs/cobol db2hs/cobol_mf db2hs/fortran db2hs/java db2hs/rexx

Notes:

1. The character in the sixth position of the file name indicates the language of a book. For example, the file name `db2d0e60` indicates that the *Administration Guide* is in English. The following letters are used in the file names to indicate the language of a book:

Language	Identifier
Brazilian Portuguese	b
Bulgarian	u
Czech	x
Danish	d
Dutch	q
English	e
Finnish	y
French	f
German	g
Greek	a
Hungarian	h
Italian	i
Japanese	j
Korean	k
Norwegian	n
Polish	p
Portuguese	v
Russian	r
Simp. Chinese	c
Slovenian	l
Spanish	z

Swedish	s
Trad. Chinese	t
Turkish	m

2. For late breaking information that could not be included in the DB2 books:
   - On UNIX-based platforms, see the Release.Notes file. This file is located in the DB2DIR/Readme/%L directory, where %L is the locale name and DB2DIR is:
     - /usr/lpp/db2_06_00 on AIX
     - /opt/IBMdb2/V6.0 on HP-UX, Solaris, SCO UnixWare 7, and Silicon Graphics IRIX
     - /usr/IBMdb2/V6.0 on Linux.
   - On other platforms, see the RELEASE.TXT file. This file is located in the directory where the product is installed.
   - Under Windows Start menu

Viewing online information

The manuals included with this product are in Hypertext Markup Language (HTML) softcopy format. Softcopy format enables you to search or browse the information, and provides hypertext links to related information. It also makes it easier to share the library across your site.

You can view the online books or sample programs with any browser that conforms to HTML Version 3.2 specifications.

To view online books or sample programs on all platforms other than SCO UnixWare 7:

- If you are running DB2 administration tools, use the Information Center. See “Accessing information with the Information Center” on page 414 for details.
- Select the Open Page menu item of your Web browser. The page you open contains descriptions of and links to DB2 information:
  - On UNIX-based platforms, open the following page:
    ```
    file:/INSTHOME/sqllib/doc/%L/html/index.htm
    ```
    where *%L* is the locale name.
  - On other platforms, open the following page:
    ```
    sqllib\doc\html\index.htm
    ```
    The path is located on the drive where DB2 is installed.

If you have not installed the Information Center, you can open the page by double-clicking on the **DB2 Online Books** icon. Depending on the system you are using, the icon is in the main product folder or the Windows Start menu.

To view online books or sample programs on the SCO UnixWare 7:

- DB2 Universal Database for SCO UnixWare 7 uses the native SCOhelp utility to search the DB2 information. You can access SCOhelp by the following methods:
  - entering the "scohelp" command on the command line,
  - selecting the `Help` menu in the Control Panel of the CDE desktop or
  - selecting `Help` in the Root menu of the Panorama desktop

  For more information on SCOhelp, refer to the *Installation and Configuration Supplement*.

Accessing information with the Information Center

The Information Center provides quick access to DB2 product information. The Information Center is available on all platforms on which the DB2 administration tools are available.

Depending on your system, you can access the Information Center from the:

- Main product folder
- Toolbar in the Control Center
- Windows Start menu
- Help menu of the Control Center

The Information Center provides the following kinds of information. Click the appropriate tab to look at the information:

Tasks	Lists tasks you can perform using DB2.
Reference	Lists DB2 reference information, such as keywords, commands, and APIs.
Books	Lists DB2 books.
Troubleshooting	Lists categories of error messages and their recovery actions.
Sample Programs	Lists sample programs that come with the DB2 Software Developer’s Kit. If the Software Developer’s Kit is not installed, this tab is not displayed.
Web	Lists DB2 information on the World Wide

Web. To access this information, you must have a connection to the Web from your system.

When you select an item in one of the lists, the Information Center launches a viewer to display the information. The viewer might be the system help viewer, an editor, or a Web browser, depending on the kind of information you select.

The Information Center provides some search capabilities, so you can look for specific topics, and filter capabilities to limit the scope of your searches.

For a full text search, click the `Search` button of the Information Center follow the *Search DB2 Books* link in each HTML file.

The HTML search server is usually started automatically. If a search in the HTML information does not work, you may have to start the search server by double-clicking its icon on the Windows or OS/2 desktop.

Refer to the release notes if you experience any other problems when searching the HTML information.

Note: Search function is not available in the Linux and Silicon Graphics environments.

Setting up a document server

By default, the DB2 information is installed on your local system. This means that each person who needs access to the DB2 information must install the same files. To have the DB2 information stored in a single location, use the following instructions:

1. Copy all files and subdirectories from \sqllib\doc\html on your local system to a Web server. Each book has its own subdirectory containing all the necessary HTML and GIF files that make up the book. Ensure that the directory structure remains the same.
2. Configure the Web server to look for the files in the new location. For information, see the NetQuestion Appendix in *Installation and Configuration Supplement*.
3. If you are using the Java version of the Information Center, you can specify a base URL for all HTML files. You should use the URL for the list of books.
4. Once you are able to view the book files, you should bookmark commonly viewed topics. Among those, you will probably want to bookmark the following pages:

- List of books
- Tables of contents of frequently used books
- Frequently referenced articles, such as the *ALTER TABLE* topic
- The Search form

For information about setting up a search, see the NetQuestion Appendix in *Installation and Configuration Supplement* book.

Searching online information

To search for information in the HTML books, you can do the following:

- Click on **Search the DB2 Books** at the bottom of any page in the HTML books. Use the search form to find a specific topic. This function is not available in the Linux or Silicon Graphics IRIX environments.
- Click on **Index** at the bottom of any page in an HTML book. Use the index to find a specific topic in the book.
- Display the table of contents or index of the HTML book, and then use the find function of the Web browser to find a specific topic in the book.
- Use the bookmark function of the Web browser to quickly return to a specific topic.
- Use the search function of the Information Center to find specific topics. See “Accessing information with the Information Center” on page 414 for details.

Printing the PostScript books

If you prefer to have printed copies of the manuals, you can decompress and print PostScript versions. For the file name of each book in the library, see the table in “DB2 information – hardcopy and online” on page 406. Specify the full path name for the file you intend to print.

On OS/2 and Windows platforms:

1. Copy the compressed PostScript files to a hard drive on your system. The files have a file extension of .exe and are located in the x:\doc\language\books\ps directory, where x: is the letter representing the CD-ROM drive and *language* is the two-character country code that represents your language (for example, EN for English).
2. Decompress the file that corresponds to the book that you want. Each compressed book is a self-extracting executable file. To decompress the

book, simply run it as you would run any other executable program. The result from this step is a printable PostScript file with a file extension of .ps.
3. Ensure that your default printer is a PostScript printer capable of printing Level 1 (or equivalent) files.
4. Enter the following command from a command line:
   ```
   print filename.ps
   ```

On UNIX-based platforms:
1. Mount the CD-ROM. Refer to your *Quick Beginnings* manual for the procedures to mount the CD-ROM.
2. Change to /cdrom/doc/%L/ps directory on the CD-ROM, where */cdrom* is the mount point of the CD-ROM and *%L* is the name of the desired locale. The manuals will be installed in the previously-mentioned directory with file names ending with .ps.Z.
3. Decompress and print the manual you require using the following command:
   - For AIX:
     ```
     zcat filename | qprt -P PSPrinter_queue
     ```
   - For HP-UX, Solaris, or SCO UnixWare 7:
     ```
     zcat filename | lp -d PSPrinter_queue
     ```
   - For Silicon Graphics IRIX:
     ```
     zcat < filename | lp -d PSPrinter_queue
     ```

   where *filename* is the full path name and extension of the compressed PostScript file and *PSprinter_queue* is the name of the PostScript printer queue.

   For example, to print the English version of *DB2 for UNIX Quick Beginnings* on AIX, you can use the following command:
   ```
   zcat /cdrom/doc/en/ps/db2ixe60.ps.Z || qprt -P ps1
   ```

Ordering the printed books

You can order the printed DB2 manuals either as a set or individually. There are three sets of books available. The form number for the entire set of DB2 books is SBOF-8926-00. The form number for the books listed under the heading ″Cross-Platform Books″ is SBOF-8924-00.

Note: These form numbers only apply if you are ordering books that are printed in the English language in North America.

You can also order books individually by the form number listed in "DB2 information – hardcopy and online" on page 406. To order printed versions, contact your IBM authorized dealer or marketing representative, or phone `1-800-879-2755` in the United States or `1-800-IBM-4YOU` in Canada.

Appendix B. Education and services for DB2 data replication

This appendix describes the services and education available for DB2 data replication.

Services

IBM and IBM Business Partners offer consulting and services supporting the DB2 data replication solution. Customized services are available in addition to service offerings that help you:

- Plan and design your application.
- Install, configure, and integrate the products.
- Evaluate operational and tuning considerations.
- Evaluate application and data migration.
- Educate and train staff.

For additional information on IBM products and services, contact your IBM software provider, or, in the U.S. and Canada, call 1-800-IBM-3333.

Education

The following classes are provided by IBM Education and Training:

- Data Replication: Basic Usage (DW140)
- Data Replication: Advanced Usage (DW150)

Details about these courses can be found at the following site on the World Wide Web: http://www.software.ibm.com/data/dpropr/education.html

General Education Information on the Web
IBM Education and Training information is available on the Web. You can access the entire curriculum of courses directly from the IBM Global Campus Web site: http://www.training.ibm.com/ibmedu

Custom Classes
Replication courses can be tailored to address your unique environment and needs. To find out more information, call 1-800-IBM-TEACH, Ext. CUSTOM (800-426-8322, Ext. CUSTOM).

IBM Employees: For complete course descriptions, see the EDUCATION application on HONE or MSE.

Appendix C. Notices

Any reference to an IBM licensed program in this publication is not intended to state or imply that only IBM's licensed program may be used. Any functionally equivalent product, program or service that does not infringe any of IBM's intellectual property rights may be used instead of the IBM product, program, or service. Evaluation and verification of operation in conjunction with other products, except those expressly designated by IBM, is the user's responsibility.

IBM may have patents or pending patent applications covering subject matter in this document. The furnishing of this document does not give you any license to these patents. You can send license inquiries, in writing, to the

IBM Director of Licensing
IBM Corporation
North Castle Drive
Armonk, NY 10504-1785
U.S.A.

Licensees of this program who wish to have information about it for the purpose of enabling: (i) the exchange of information between independently created programs and other programs (including this one) and (ii) the mutual use of the information which has been exchanged, should contact:

IBM Canada Limited
Office of the Lab Director
1150 Eglinton Ave. East
North York, Ontario
M3C 1H7
CANADA

Such information may be available, subject to appropriate terms and conditions, including in some cases, payment of a fee.

This publication may contain 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.

Programming interface information

This book describes intended programming interfaces that allow the customer to write programs to obtain the services of IBM Replication. This information is identified where it occurs by an introductory statement to a chapter or section.

This book also documents information that is **NOT** intended to be used as programming interfaces of IBM Replication. Do **NOT** use diagnosis, modification, or tuning information as a programming interface. This information is identified where it occurs by an introductory statement to a chapter or section.

Trademarks

The following terms are trademarks or registered trademarks of the IBM Corporation in the United States and/or other countries:

ACF/VTAM
ADSTAR
AISPO
AIX
AIXwindows
AnyNet
APPN
AS/400
CICS
C Set++
C/370
DATABASE 2
DataHub
DataJoiner
DataPropagator
DataRefresher
DB2
DB2 Connect
DB2 Universal Database
Distributed Relational Database Architecture
Extended Services
FFST
First Failure Support Technology
IBM
IMS
Lan Distance
MVS/ESA
MVS/XA
OS/400
OS/390
OS/2
PowerPC
QMF
RACF
RISC System/6000
SP
SQL/DS
SQL/400
S/370
System/370
System/390
SystemView
VisualAge
VM/ESA
VSE/ESA
VTAM
WIN-OS/2

Trademarks of other companies

Glossary

A

after-image. The updated content of a source table column that is recorded in a change data table, or in a database log or journal. Contrast with *before-image*.

application version. The set of setup, update, and cleanup batches that are associated with a specific version of an application that runs on the satellites of a group.

Apply job table. An AS/400 specific replication control table at the control server used to guarantee a unique APPLY_QUAL value for all instances of the Apply program running at the control server.

Apply program. A program that is used to refresh or update a target table, depending on the applicable source-to-target rules. Contrast with *Capture program* and *Capture trigger*.

Apply qualifier. A case-sensitive character string that identifies subscription sets that are unique to each instance of the Apply program.

Apply-qualifier-cross-reference table. An AS/400 specific replication control table at the source server that contains information to support update-anywhere replication.

Apply trail table. A replication control table at the control server that records a history of the updates performed against target tables.

archive log. The set of log files that are closed and are no longer needed for normal processing. These files are retained for use in roll-forward recovery. Contrast with *active log*.

audit trail. Data, in the form of a logical path linking a sequence of events, used for tracing the transactions that affected the contents of a record.

B

base aggregate table. A type of target table that contains data aggregated from a source table or a point-in-time table at intervals.

before-image. The content of a source table column prior to a refresh or update, as recorded in a change data table, or in a database log or journal. Contrast with *after-image*.

binary large object (BLOB). A sequence of bytes, where the size of the sequence ranges from 0 to 2 gigabytes. This string does not have an associated code page and character set. Image, audio, and video objects are stored in BLOBs.

BLOB. Binary large object.

C

Capture enqueue table. This VM and VSE specific control table at the source server is used to ensure that only one Capture program is running per database.

Capture program. A program that reads database log or journal records to capture data about changes made to DB2 source tables. Contrast with *Apply program* and *capture trigger*.

Capture trigger. A mechanism that captures delete, update, and insert operations performed on non-IBM source tables. Contrast with *Capture program* and *Apply program*.

cascade rejection. The process of rejecting a replication transaction because it is associated with a transaction that had a conflict detected and was itself rejected.

CCD table. Consistent-change-data table.

CD table. Change data table.

change aggregate table. A type of target table that contains data aggregations based on changes recorded for a source table.

change data (CD) table. A replication control table at the source server that contains changed data for a replication source table.

character large object (CLOB). A sequence of characters (single-byte, multi-byte, or both) where the length can be up to 2 gigabytes. This data type can be used to store large text objects. Also called character large object string.

client. Any program (or workstation that it is running on) that communicates with and accesses a database server.

CLOB. Character large object.

cold start. The process of starting the Capture program, using an initial program load procedure. Contrast with *warm start*.

conflict detection. In update-anywhere replication configurations, the process of detecting if the same row was updated in the source and target tables during the same replication cycle. When a conflict is detected, the transaction that caused the conflict is rejected. See also *enhanced conflict detection*, *standard conflict detection*, and *row-replica conflict detection*.

conflict table. A Microsoft Jet specific table at the target server that contains row data for DataPropagator for Microsoft Jet-detected conflict losers.

consistent-change-data (CCD) table. A replication table that is used for staging data, with four replication control columns. See also *internal CCD table* and *external CCD table*.

consolidation replication. A replication model in which the data from multiple source tables is replicated to a single target table. Contrast with *fan-out replication*.

Control Center. A graphical user interface that shows database objects (such as databases and tables) and their relationship to each other. From the Control Center you can perform the tasks on database objects.

control server. The database location of the applicable subscription definitions and Apply program control tables.

control table. A table in which replication source and subscription definitions or other replication control information is stored.

critical section table. A replication control table at the source server that is used to prevent circular replication for update-anywhere subscriptions.

D

database log. A set of primary and secondary log files consisting of log records that record all changes to a database. The database log is used to roll back changes for transactions that are not committed and to recover a database to a consistent state.

database management system (DBMS). Synonym for database manager.

database manager. A computer program that manages data by providing the services of centralized control, data independence, and complex physical structures for efficient access, integrity, recovery, data currency control, privacy, and security.

database server. A functional unit that provides database services for databases.

DBCLOB. Double-byte character large object.

DBMS. Database management system.

delimited identifier. A sequence of characters enclosed within quotation marks (″). The sequence must consist of a letter followed by zero or more characters, each of which is a letter, a digit, or the underscore character.

Design Master. The original copy of a Microsoft Jet database. Only the Design Master supports changes to the database structure (table, query, and form design).

differential refresh. A process in which only changed data is copied to the target table, replacing existing data.

distinct type. A user-defined data type that is internally represented as an existing type (its source type), but is considered to be a separate and incompatible type for semantic purposes. See also *user-defined type (UDT).*

double-byte character large object (DBCLOB). A sequence of double-byte characters, where the size can be up to 2 gigabytes. This data type can be used to store large double-byte text objects. Also called *double-byte character large object string*. Such a string always has an associated code page.

E

enhanced conflict detection. Conflict detection that guarantees data integrity among all replicas and the source table. The Apply program locks all replicas in the subscription set against further transactions, and begins detection after all changes made prior to locking have been captured. See also *standard conflict detection*, *conflict detection*, and *row-replica conflict detection*.

error information table. A Microsoft Jet specific table at the target server that contains additional information to identify the row-replica table and row that caused an error.

error messages table. A Microsoft Jet specific table at the target server that contains error codes and error messages.

error-side-information table. A Microsoft Jet specific table at the target server that contains the names of the conflict tables.

external source table. A non-DB2 table that is manually updated to match the consistent-change-data table structure and defined as a replication source. Also called an external CCD table. See also *consistent-change-data (CCD) table*.

F

fan-out replication. A replication model in which data from one source table is copied to multiple target tables, thereby distributing the data to multiple locations. Contrast with *consolidation replication*.

full refresh. A process in which all of the data of interest in a user table is copied to the target table, replacing existing data. Contrast with *differential refresh.*

G

gap. A situation in which the Capture program is not able to read a range of log or journal records, so there is potential loss of change data.

group. A collection of satellites that share characteristics such as database configuration and the application that runs on the satellite.

I

internal CCD table. A consistent-change-data table that is a join of the change data table and the unit-of-work table at the source server.

J

join. A relational operation that allows for retrieval of data from two or more tables based on matching column values.

K

key. A column or an ordered collection of columns that are identified in the description of a table, index, or referential constraint.

key string table. A Microsoft Jet specific table at the target server that maps the Microsoft Jet table identifiers and row identifiers to primary key values when the following actions occur:

- Rows are deleted from Microsoft Jet database tables.
- Deleted rows are recorded in MSysTombstone with s_Generation, TableGUID, and s_GUID (row) identifiers, but without primary key details.
- The primary key values are needed to replicate Microsoft Jet database deletes to an RDBMS.

L

large object (LOB). A sequence of bytes, where the length can be up to 2 gigabytes. It can be any of three types: BLOB (binary), CLOB (single-byte character or mixed) or DBCLOB (double-byte character).

LOB. Large object.

local database. A database that is physically located on the workstation in use. Contrast with *remote database*.

lock.
1. A means of serializing events or access to data
2. A means of preventing uncommitted changes made by one application process from being perceived by another application process and for preventing one application process from updating data that is being accessed by another process

locking. The mechanism used by the database manager to ensure the integrity of data. Locking prevents concurrent users from accessing inconsistent data.

M

member. See *subscription-set member*.

mobile client. The node, usually a laptop computer, where the mobile replication enabler and replication source and target tables used in a mobile environment are located. The mobile replication mode is started from the mobile client.

mobile replication enabler. A replication program that starts the mobile replication mode at the mobile client.

mobile replication mode. A mode of replication in which the Capture and Apply programs operate as needed rather than continuously. This mode is started from the mobile client and allows data to be replicated when the mobile client is connected to the source or target server.

N

nullable. The condition where a value for a column, function parameter, or result can have an absence of a value. For example, a field for a person's middle initial does not require a value.

null value. A parameter for which no value is specified.

O

object.
1. Anything that can be created or manipulated with SQL—for example, tables, view, indexes, or packages.
2. In object-oriented design or programming, an abstraction consisting of data and operations associated with that data.

ODBC. Open Database Connectivity.

ODBC driver. A driver that implements ODBC function calls and interacts with a data source.

Open Database Connectivity (ODBC). An API that allows access to database management systems using callable SQL, which does not require the use of an SQL preprocessor. The ODBC architecture allows users to add modules, called database drivers, that link the application to their choice of database management systems at run time. Applications do not need to be linked directly to the modules of all the supported database management systems.

ordinary identifier. In SQL, a letter, which might be followed by zero or more characters, each of which is a letter (a-z and A-Z), a symbol, a number, or the underscore character, used to form a name.

P

package. A control structure produced during program preparation that is used to execute SQL statements.

partitioning key.

1. An ordered set of one or more columns in a table. For each row in the table, the values in the partitioning key columns are used to determine on which database partition the row belongs.
2. In replication, an ordered set of one or more columns in a table. For each row in the source table, the values in the partitioning key columns are used to determine in which target table the row belongs.

point-in-time table. A type of target table whose content matches all or part of a source table, with an added system column that identifies the approximate time when the particular row was inserted or updated at the source system.

predicate. An element of a search condition that expresses or implies a comparison operation.

primary key. A unique key that is part of the definition of a table. A primary key is the default parent key of a referential constraint definition.

pruning control table. A replication control table at the source server that coordinates the pruning of the change data and unit-of-work control tables. The values in this table indicate how much data has been replicated by the Apply program and can be safely pruned by the Capture mechanisms.

prune lock table. A replication control table at the source server that is used to serialize the access of staging tables during cold start and retention limit pruning.

R

RDBMS. Relational database management system.

referential integrity. The state of a database in which all values of all foreign keys are valid.

registration. The process of identifying a source table to DB2 DPROP to make the table available for subscription. See also *replication source*.

register table. A replication control table at the source server that relates each source table or view to an associated change data table and consistent-change-data table, if applicable.

register extension table. An AS/400 specific replication control table at the source server that is an extension of the register table. This table contains additional information about replication sources, such as the journal name and the remote source table's RDB entry name.

register synchronization table. A replication control table at the source server that contains an update trigger that initiates an update of the SYNCHPOINT value for all the rows in the register table before the Apply program reads the information from the register table. The register synchronization table is used when replicating from a non-IBM data source.

rejected transaction. A transaction containing one or more updates from replica tables that are out of date in comparison to the source table.

replica target table. A replication table at the target server that is a type of update-anywhere target table.

replication. The process of maintaining a defined set of data in more than one location. It involves copying designated changes for one location (a source) to another (a target), and synchronizing the data in both locations.

replication source. A database table or view that is defined as a source for replication. This type of table can accept copy requests and is the source table in a replication subscription set. See also *subscription set* and *registration*.

remote database. A database that is physically located on a workstation other than the one in use. Contrast with *local database*.

row-replica. A type of update-anywhere replica maintained by DataPropagator for Microsoft Jet without transaction semantics.

row-replica conflict detection. Conflict detection done row by row, not transaction by transaction, as is done for DB2 replicas.

row-replica table. A type of target table used specifically with the Microsoft Jet database.

row-replica-target-list table. A Microsoft Jet specific table at the control server that maintains the names of the row-replica tables.

S

satellite. An occasionally connected client machine that has a DB2 server that synchronizes with its group at the satellite control database.

Satellite Administration Center. A graphical user interface that provides centralized administrative support for satellites.

satellite control server. A DB2 Universal Database system that contains the satellite control database, SATCTLDB.

serialization.

- The consecutive ordering of items.
- In DB2 for AS/400, the process of controlling access to a resource to protect the integrity of the resource.

source server. The database location of the replication source and the Capture program.

source table. A table that contains the data that is to be copied to a target table. The source table can be a replication source table, a change data table, or a consistent-change-data table. Contrast with *target table*.

spill file. A temporary file created by the Apply program that is used as the source for updating data to multiple target tables.

staging table. A consistent-change-data table that can be used as the source for updating data to multiple target tables.

standard conflict detection. Conflict detection in which the Apply program searches for conflicts in rows that are already captured in the replica's change data tables. See also *conflict detection*, *enhanced conflict detection*, and *row-replica conflict detection*.

subscription. See *subscription set*.

subscription columns table. A replication control table that contains column details of target tables.

subscription events table. A replication control table that defines the events that trigger replication, including the event name and time.

subscription-schema-changes table. A Microsoft Jet specific table that is used to signal when add or delete modifications are made to a subscription.

subscription set. The specification of a group of source tables, target tables, and the control information that governs the replication of changed data.

subscription-set member. A member of a subscription set. There is one member for each source-target pair. Each member defines the structure of the target table and which rows and columns will be replicated from the source table.

subscription set table. A replication control table that defines the members of a subscription set.

subscription statements table. A replication control table used to store the optional SQL statements that can be run at the beginning or end of the subscription set cycle.

subscription-targets-member table. A replication control table that maps the source and target table relationships within a subscription set.

synchronization generations table. A Microsoft Jet specific table at the target server that prevents cyclic updates from replicating back to the RDBMS from a Microsoft Jet database.

T

target server. The database location of the target table. Normally this is also the location of the Apply program.

target table. The table on the target server to which data is copied. It can be a user copy table, a point-in-time table, a base aggregate table, a change aggregate table, a consistent-change-data table, or a replica table.

temporary table. A table created during the processing of an SQL statement to hold intermediate results.

trace table. A table that contains a high-level record of the execution of the Capture program.

transaction. An exchange between a workstation and a program, two workstations, or two programs that accomplishes a particular action or result. Some examples are the entry of a customer's deposit and the updating of the customer's balance.

trigger. In DB2, an object in a database that is invoked indirectly by the database manager when a particular SQL statement is run.

tuning parameters table. A table at the source server that contains timing information used by the Capture program. The information includes:

- How long to keep rows in the change data table.
- How much time can elapse before changes are stored in a database log or journal.
- How often to commit changed data to the unit-of-work tables.

two-phase commit. A two-step process by which recoverable resources and an external subsystem are committed. During the first step, the database manager subsystems are polled to ensure that they are ready to commit. If all subsystems respond positively, the database manager instructs them to commit.

U

UDT. User-defined type.

uncommitted read (UR). An isolation level that allows an application to access uncommitted changes of other transactions. The application does not lock other applications out of the row that it is reading unless the other application attempts to drop or alter the table.

unit-of-work table. A replication control table at the source server that contains commit records read from the database log or journal. The records include a unit-of-recovery ID that can be used to join the unit-of-work table and the change data table to produce transaction-consistent change data. For DB2, the unit-of-work table optionally includes the correlation ID, which can be useful for auditing purposes.

update. A process in which the changes to data in a source table are used to refresh a target table. This process is also known as *differential refresh*.

UR. Uncommitted read.

user copy table. A target table whose content matches all or part of a source table and contains only user data columns.

user-defined type (UDT). A data type that is not native to the database manager and was created by a user. See also *distinct type*.

user table. A table created for and used by an application before it is defined as a replication source. It is used as the source for updates to read-only target tables such as the consistent-change-data, replicas, and row-replica tables.

V

view. A logical table that consists of data that is generated by a query.

W

warm start. A start of the Capture program that allows reuse of previously initialized input and output work queues. Contrast with *cold start*.

warm start table. A table used by the Capture program to save the position in a DBMS log for later reference during a warm start.

work file. A temporary file used by the Apply program when processing a subscription set.

Index

Contacting IBM

This section lists ways you can get more information from IBM.

If you have a technical problem, please take the time to review and carry out the actions suggested by the *Troubleshooting Guide* before contacting DB2 Customer Support. Depending on the nature of your problem or concern, this guide will suggest information you can gather to help us to serve you better.

For information or to order any of the DB2 Universal Database products, contact an IBM representative at a local branch office or contact any authorized IBM software remarketer.

Telephone

If you live in the U.S.A., call one of the following numbers:
- 1-800-237-5511 to learn about available service options.
- 1-800-IBM-CALL (1-800-426-2255) or 1-800-3IBM-OS2 (1-800-342-6672) to order products or get general information.
- 1-800-879-2755 to order publications.

For information on how to contact IBM outside of the United States, see Appendix A of the IBM Software Support Handbook. You can access this document by accessing the following page:

```
http://www.ibm.com/support/
```

then performing a search using the keyword “handbook”.

Note that in some countries, IBM-authorized dealers should contact their dealer support structure instead of the IBM Support Center.

World Wide Web

http://www.software.ibm.com/data/
http://www.software.ibm.com/data/db2/library/

The DB2 World Wide Web pages provide current DB2 information about news, product descriptions, education schedules, and more. The DB2 Product and Service Technical Library provides access to frequently asked questions, fixes, books, and up-to-date DB2 technical information. (Note that this information may be in English only.)

Anonymous FTP Sites

ftp.software.ibm.com

Log on as anonymous. In the directory /ps/products/db2, you can find demos, fixes, information, and tools concerning DB2 and many related products.

Internet Newsgroups

comp.databases.ibm-db2, bit.listserv.db2-l

These newsgroups are available for users to discuss their experiences with DB2 products.

CompuServe

GO IBMDB2 to access the IBM DB2 Family forums

All DB2 products are supported through these forums.

To find out about the IBM Professional Certification Program for DB2 Universal Database, go to http://www.software.ibm.com/data/db2/db2tech/db2cert.html

Printed in the United States of America
on recycled paper containing 10%
recovered post-consumer fiber.

SC26-9642-00