



DB2 Replication Guide and Reference

Version 5 Release 2



DB2 Replication Guide and Reference

Version 5 Release 2

Before using this information and the product it supports, be sure to read the general information under Appendix D, "Notices" on page 451.

This document contains proprietary information of IBM. It is provided under a license agreement and is protected by copyright law. The information contained in this publication does not include any product warranties and any statements provided in this manual should not be interpreted as such.

Order publications through your IBM representative or the IBM branch office serving your locality or by calling 1-800-879-2755 in U.S. or 1-800-IBM-4YOU in Canada.

When you send information to IBM, you grant IBM a nonexclusive right to use or distribute the information in any way it believes appropriate without incurring any obligation to you.

© **Copyright International Business Machines Corporation 1994, 1998. All rights reserved.**

Note to U.S. Government Users — Documentation related to restricted rights — Use, duplication or disclosure is subject to restrictions set forth in GSA ADP Schedule Contract with IBM Corp.

Contents

Welcome to Replication	xvii
Conventions	xvii
Terminology	xvii
How to Read the Syntax Diagrams	xviii
Road Map	xix

Part 1. Introduction to IBM Replication 1

Chapter 1. The Complete IBM Data Replication Solution	3
How the Tools Work Together	4
Legacy Data Sources	6
Multivendor Data Sources and Targets	6
Interoperability with Lotus Notes and ODBC Accessible Data Stores	6
Services	7
IBM Replication Education	7

Chapter 2. Introduction to the Replication Tools	9
The Control Center	9
Tasks	9
Objects	10
Types of Copies	11
Auto-Registration	12
The Replication Control Tables	12
The Capture Program	13
How the Capture Program Captures Changes	13
Control Tables Used by the Capture Program	14
The Apply Program	15
How the Apply Program Replicates Data	16
The Apply Processing Cycle	16
How the Apply Program Selects a Source Table	17
The Apply Qualifier	17
The Control Server	17
Control Tables Used by the Apply Program	18
Full Refresh and Differential Refresh Replication	18

Chapter 3. Getting Started	21
Performing a Replication Scenario for Windows NT	21
Defining a Replication Source	23
Defining a Subscription	24
Configuring the Capture and Apply Programs	25
Configuring the Capture Program	26
Configuring the Apply Program	26
Providing End-User Authentication at the Source Server	27
Starting the Capture and Apply Programs	28

Stopping the Capture and Apply Programs	28
---	----

Part 2. Planning for Replication	31
---	-----------

Chapter 4. Replication Usage Scenarios	33
---	-----------

Recommended Usage Scenarios	33
Operational to Decision Support System Data Replication	33
Distributed Database Systems	34
Improved Network Load Balancing	34
Data Consolidation and Distribution	34
Improved Application Availability	35
Data Archive	35
Building Audit Trails	35
Mobile Replication	35
Mobile Replication with MS Jet Client	36
Potential Usage Scenarios	36
Update-Anywhere Replication	36
Logical Recovery	37
Extending IBM Replication	37
Usage Scenarios Not Recommended for Replication	37
Synchronous Replication	37
Hot-Site Recovery	38

Chapter 5. System Planning	39
---	-----------

Replication Products	39
Hardware and Software Requirements	40
The Capture and Apply Programs on the DB2 Universal Database	40
The Capture and Apply Programs for DB2 for MVS, DB2 for VSE, or DB2 for VM	40
Hardware Requirements	40
Software Requirements	41
Storage Requirements	42
DBMS Logging	42
New Tables	43
The Apply Spill File	43
Data Blocking for Large Volumes of Changes	44
The Active Log File Size for Capture for MVS on DB2 for MVS 3.1 and Capture for VSE and VM 5.1	45
General Storage Considerations	46
Capacity Requirements Analysis	46
The Capture Program	46
The Apply Program	46
Administration	46
Network Requirements	47
Connectivity Possibilities	47
Bandwidth Impact Analysis	48
Pull versus Push Apply Design	48
Throughput Capacity	50
Security and Authorization Requirements	51

Authorization Requirements for Administration	51
Authorization Requirements for the Capture Program	51
Authorization Requirements for the Apply Program	51
Customization Requirements	52
Chapter 6. Application Planning	53
Data Transformation Requirements	53
Basic Data Enhancement	53
Advanced Data Enhancement	57
Auditing Requirements	58
Mobile Replication Requirements	58
Staging Changed Data	59
CD and CCD Tables	59
Staging Tables	61
Benefits of Staging Data	61
Transaction-Based versus Transaction-Consistent Replication: Using Internal CCD Tables to Reduce Network Load	65
How External CCD Tables Are Refreshed (Cascade CCD Full Refresh)	65
Developing a Data Warehouse with CCD Tables	66
Replication Logical Partitioning Key Considerations	66
Data Sharing Considerations	67
Extended Enterprise Edition Considerations	67
Data Restrictions	67
Limits on Column Names for Capturing Before-Image Data	68
Data Currency Requirements	69
Interval Timing (Relative Timing)	69
Event Timing	69
Data Consistency Requirements	70
Update-Anywhere Replication	72
Conflict Detection	74
Recovering from Conflicts	74
CCD Tables	75
Recommended Usage	75
Restrictions	75
Minimizing Contention between Capture and Apply for MVS 3.1 Sources and Targets	76
How Locking Affects Contention on Platforms without ISOLATION (UR) Support	76
Resolving Gaps between Source and Target Tables	77
Using Your Own Full Refresh Technique for External CCD Tables	78

Part 3. Administering Your Replication System 81

Chapter 7. Administration Overview	83
Overview of Replication Administration Steps	83
Navigating to Your Replication Objects with the Control Center	84
Configuring the Control Center for Non-DB2 Universal Database RDBMSs	85
Setting Replication Preferences in the DB2 Tools Settings Notebook	85
Working with Customized Replication Control Tables	88

Customizing and Running Replication SQL Files	90
Ordinary and Delimited Identifiers	92
Chapter 8. Working with Replication Sources	93
What Is a Replication Source?	93
Defining Replication Sources	93
Defining a Replication Source with the Default Values (Quick)	94
Defining a Custom Replication Source	95
Defining an External CCD Table as a Replication Source	97
Defining Join Replication Sources	98
Viewing or Changing Existing Replication Sources	100
Removing Replication Sources	101
Defining Replication Subscriptions for Update Anywhere	102
Chapter 9. Working with Replication Targets	103
What Are Replication Subscriptions?	103
Defining Replication Subscriptions	103
Defining Replication Sources: Advanced Tasks	107
Choosing a Target Table Type	107
Defining the Target Table Structure: Columns and Rows	108
Setting the Copying Schedule: Time or Event Based	114
Specifying Mini-Cycles for the Apply Program to Copy Committed Data	116
Defining SQL Statements or CALL Procedures for the Replication Subscription	117
Defining a Replication Subscription with a User-Defined Target Table	120
Restrictions	121
Maintaining a User-Defined Target Table	121
Activating and Deactivating Replication Subscriptions	122
Cloning a Replication Subscription to Another Server	123
Viewing or Changing an Existing Replication Subscription	123
Removing Replication Subscriptions	124
Defining Replication Sources and Replication Subscriptions for Update Anywhere	125

Part 4. IBM Replication Capture and Apply 127

Chapter 10. Capture and Apply for MVS	129
Setting Up the Capture and Apply Programs	129
Specifying Tuning Parameters for the Capture Program	130
DB2 for MVS Rules Regarding Indexes	131
Restrictions When Running the Capture Program	131
Authorization for Running the Capture Program	132
Recovering from Severe Errors	132
Alert Generation	133
Trace Buffer	133
Trace Output	133
Storage Dump	133
Operating Capture for MVS	133
Before You Start the Capture Program	133
Starting Capture for MVS	134

Scheduling Capture for MVS	135
Stopping Capture for MVS	136
Suspending Capture for MVS	136
Resuming Capture for MVS	137
Reinitializing Capture for MVS	137
Pruning the Change Data and Unit-of-Work Tables	137
Providing the Current Log Sequence Number	138
Warm and Cold Starts	138
Warm Start Process	138
Automatic Cold Starts	139
Forcing a Warm Start	139
Operating Apply for MVS	139
Before You Start the Apply Program	140
Starting Apply for MVS	140
Scheduling Apply for MVS	142
Stopping Apply for MVS	142
Additional Apply Program Operations	142
Scheduling Subscriptions with the Event Table	142
Returning Control to Users with ASNDONE	143
Initiating a Forward Recovery with the Apply Program	143
Loading the Tables within a Subscription Set	144
Troubleshooting	145
Problems Using the Capture Program	145
Problems Using the Apply Program	147
Chapter 11. Capture and Apply for Windows NT and Windows 95	151
Setting Up the Capture and Apply Programs	151
Configuring the Capture Program for Windows NT and Windows 95	151
Configuring the Apply Program for Windows NT and Windows 95	152
Providing End-User Authentication at the Source Server	153
Setting Up the NT Service Control Manager	153
Specifying Tuning Parameters for the Capture Program	155
Restrictions When Running the Capture Program	156
Authorization for Running the Capture Program	156
Operating Capture for Windows NT and Windows 95	156
Before You Start the Capture Program	156
Starting Capture for Windows NT and Windows 95	157
Scheduling Capture for Windows NT and Windows 95	158
Stopping Capture for Windows NT and Windows 95	158
Suspending Capture for Windows NT and Windows 95	160
Resuming Capture for Windows NT and Windows 95	160
Reinitializing Capture for Windows NT and Windows 95	161
Pruning the Change Data and Unit-of-Work Tables	161
Providing the Current Log Sequence Number	162
Warm and Cold Starts	162
Warm Start Process	162
Automatic Cold Starts	163
Forcing a Warm Start	163

Operating Apply for Windows NT and Windows 95	163
Before You Start the Apply Program	163
Starting Apply for Windows NT and Windows 95	164
Scheduling Apply for Windows NT and Windows 95	166
Stopping Apply for Windows NT and Windows 95	166
Additional Apply Program Operations	167
Scheduling Subscriptions with the Event Table	167
Refreshing Target Tables with ASNLOAD	167
Returning Control to Users with ASNDONE	168
Initiating a Forward Recovery with the Apply Program	168
Loading the Tables within a Subscription Set	169
Troubleshooting	170
Problems Using the Capture Program	170
Problems Using the Apply Program	171
Chapter 12. Capture and Apply for OS/2	175
Setting Up the Capture and Apply Programs	175
Configuring the Capture Program for OS/2	175
Configuring the Apply Program for OS/2	176
Specifying Tuning Parameters for the Capture Program	177
Restrictions When Running the Capture Program	178
Authorization for Running the Capture Program	178
Operating Capture for OS/2	178
Before You Start the Capture Program	178
Starting Capture for OS/2	179
Scheduling Capture for OS/2	180
Stopping Capture for OS/2	180
Suspending Capture for OS/2	181
Resuming Capture for OS/2	182
Reinitializing Capture for OS/2	182
Pruning the Change Data and Unit-of-Work Tables	183
Providing the Current Log Sequence Number	184
Warm and Cold Starts	184
Warm Start Process	184
Automatic Cold Starts	185
Forcing a Warm Start	185
Operating Apply for OS/2	185
Before You Start the Apply Program	185
Starting Apply for OS/2	186
Scheduling Apply for OS/2	188
Stopping Apply for OS/2	188
Additional Apply Program Operations	188
Scheduling Subscriptions with the Event Table	189
Refreshing Target Tables with ASNLOAD	189
Returning Control to Users with ASNDONE	190
Initiating a Forward Recovery with the Apply Program	190
Loading the Tables within a Subscription Set	190
Troubleshooting	192

Problems Using the Capture Program	192
Problems Using the Apply Program	193
Chapter 13. Capture and Apply for UNIX Platforms	197
Setting up a UNIX User Account	197
Setting Up the Capture and Apply Programs	197
Configuring the Capture Program for UNIX Platforms	197
Configuring the Apply Program for UNIX Platforms	198
Other Configuration Considerations for UNIX-Based Components	199
Providing End-User Authentication at the Source Server	199
Specifying Tuning Parameters for the Capture Program	201
Restrictions When Running the Capture Program	202
Authorization for Running the Capture Program	202
Operating Capture for AIX, Capture for HP-UX, Capture for Solaris, and Capture for UnixWare 7	202
Before You Start the Capture Program	202
Starting Capture for AIX, Capture for HP-UX, Capture for Solaris, and Capture for UnixWare 7	203
Scheduling Capture for AIX, Capture for HP-UX, Capture for Solaris, and Capture for UnixWare 7	204
Stopping Capture for AIX, Capture for HP-UX, Capture for Solaris, and Capture for UnixWare 7	205
Suspending Capture for AIX, Capture for HP-UX, Capture for Solaris, and Capture for UnixWare 7	206
Resuming Capture for AIX, Capture for HP-UX, Capture for Solaris, and Capture for UnixWare 7	206
Reinitializing Capture for AIX, Capture for HP-UX, Capture for Solaris, and Capture for UnixWare 7	207
Pruning the Change Data and Unit-of-Work Tables	208
Providing the Current Log Sequence Number	208
Warm and Cold Starts	209
Warm Start Process	209
Automatic Cold Starts	209
Forcing a Warm Start	209
Operating Apply for AIX, Apply for HP-UX, Apply for Solaris, and Apply for UnixWare 7	210
Before You Start the Apply Program	210
Starting Apply for AIX, Apply for HP-UX, Apply for Solaris, and Apply for UnixWare 7	210
Scheduling Apply for AIX, Apply for HP-UX, Apply for Solaris, and Apply for UnixWare 7	213
Stopping Apply for AIX, Apply for HP-UX, Apply for Solaris, and Apply for UnixWare 7	213
Additional Apply Program Operations	214
Scheduling Subscriptions with the Event Table	214
Refreshing Target Tables with ASNLOAD	214
Returning Control to Users with ASNDONE	215
Initiating a Forward Recovery with the Apply Program	215

Loading the Tables within a Subscription Set	216
Troubleshooting	217
Problems Using the Capture Program	217
Problems Using the Apply Program	219
Chapter 14. Capture for VSE	223
Setting Up the Capture Program	223
Specifying Tuning Parameters for the Capture Program	223
Restrictions When Running the Capture Program	224
Authorization for Running the Capture Program	225
Recovering from Severe Errors	225
Trace Buffer	225
Trace Output	225
Storage Dump	225
Operating Capture for VSE	225
Before You Start the Capture Program	226
Starting Capture for VSE	226
Stopping Capture for VSE	228
Suspending Capture for VSE	229
Resuming Capture for VSE	230
Reinitializing Capture for VSE	230
Pruning the Change Data and Unit-of-Work Tables	231
Providing the Current Log Sequence Number	231
Warm and Cold Starts	231
Warm Start Process	231
Automatic Cold Starts	232
Forcing a Warm Start	232
Troubleshooting: Problems Using the Capture Program	232
Chapter 15. Capture for VM	235
Setting Up the Capture Program	235
Specifying Tuning Parameters for the Capture Program	235
Restrictions When Running the Capture Program	236
Authorization for Running the Capture Program	237
Recovering from Severe Errors	237
Trace Buffer	237
Trace Output	237
Storage Dump	237
Operating Capture for VM	238
Before You Start the Capture Program	238
Starting Capture for VM	239
Stopping Capture for VM	240
Suspending Capture for VM	241
Resuming Capture for VM	242
Reinitializing Capture for VM	242
Pruning the Change Data and Unit-of-Work Tables	242
Providing the Current Log Sequence Number	243
Warm and Cold Starts	243

Warm Start Process	243
Automatic Cold Starts	243
Forcing a Warm Start	244
Troubleshooting: Problems Using the Capture Program	244

Part 5. Mobile Replication 247

Chapter 16. Mobile Replication for DB2	249
An Overview of Mobile Replication	249
Highlights	249
How Mobile Replication Works	250
Mobile Replication Restrictions	250
Planning Mobile Replication	250
Software and Hardware Requirements	250
Communication Program Requirements	251
Setting Up the Mobile Client	253
Configuring the Mobile Client for Windows NT and Windows 95	253
Configuring the Mobile Client for OS/2	254
Defining the Control Server for Your Mobile Client	255
Mobile Replication Processing Cycle	255
Starting the Mobile Replication Enabler Using the ASNCOPY Command	255
Starting the Mobile Replication Enabler Using the Mobile Graphical Interface	256
Selecting Replication Subscriptions	257
Selecting an Apply Qualifier	258

Chapter 17. Mobile Replication Using IBM DB2 DataPropagator for Microsoft Jet	259
What Is DataPropagator for Microsoft Jet?	259
The Advantages of Mobile Replication Using DataPropagator for Microsoft Jet	260
Data Integrity Considerations	261
Terminology for DataPropagator for Microsoft Jet Replication	261
Operating DataPropagator for Microsoft Jet	262
Starting Capture at the Source Server	263
Starting DataPropagator for Microsoft Jet	263
Stopping DataPropagator for Microsoft Jet	264
Troubleshooting DataPropagator for Microsoft Jet	265
Returning Control to Users with the ASNJDONE Exit	265
Parameters	266
Error Handling	266
DataPropagator for Microsoft Jet Control Tables	266
Control Server Tables	267
Target Server Tables	267

Part 6. Reference Information 269

Chapter 18. Migrating from DataPropagator Relational Version 1 to IBM Replication Version 5	271
--	------------

Migration Process Overview	271
Collection	271
Analysis	272
Migration	272
Migration Requirements	272
Migration Precautions	273
Installation of the Capture and Apply Programs on DB2 Version 2	273
Before You Begin Migration	274
Invoking the Migration Program and Actions	274
Collecting Data with BUILDDDB	275
Analyzing Data with PREPARE	276
Migrating Data with MIGRATE	277
Migrating a Capture Program Process	277
Handling Migration Process Failures for the Capture Program	280
Migrating an Apply Program Process	280
Handling Migration Process Failures for the Apply Program	282
Reverting to Version 1 with FALLBACK	282
Falling Back from an Apply Program Process	283
Falling Back from a Capture Program Process	284
Dropping Version 1 Control Tables and Migration Control Tables with CLEANUP	285
Dropping Apply Program Views and Tables	285
Dropping Capture Program Views and Tables	286
Tables in the Migration Database	287
Version Control Table	287
Global Change Data Control Table	287
Global View Dependencies Table	288
Global View Text Table	289
Global Refresh Control Table	289
Global Refresh Columns Table	291
Registration Migration Units (RMU) Table	292
Subscription Migration Units (SMU) Table	292
RMU Dependencies on SMU Table	293
SMU Dependencies on RMU Table	293
Chapter 19. Table Structures	295
List of Tables Used at the Source Server	295
List of Tables Used at the Control Server	296
List of Tables Used at the Target Server	297
Control Tables Used at the Source Server	299
Capture Enqueue Table	299
Change Data Table	299
Critical Section Table	301
Pruning Control Table	302
Register Table	306
Trace Table	312
Tuning Parameters Table	313
Unit-of-Work Table	314
Warm Start Table	317

Warm Start Table for Capture for VSE and VM	318
Control Tables Used at the Control Server	319
Apply Trail Table	319
Subscription Columns Table	322
Subscription Events Table	324
Subscription Set Table	326
Subscription Statements Table	330
Subscription Targets Member Table	332
Subscription Schema Changes Table (Microsoft Jet Specific)	335
Row-Replica Target List Table (Microsoft Jet Specific)	336
Control Tables Used at the Target Server	337
Base Aggregate Target Table	338
Change Aggregate Target Table	339
Consistent Change Data Table	340
Point-in-Time Target Table	343
Replica Target Table	344
User Copy Target Table	345
Conflict Table (Microsoft Jet Specific)	347
Error Information Table (Microsoft Jet Specific)	347
Error Messages Table (Microsoft Jet Specific)	348
Error Side Information Table (Microsoft Jet Specific)	349
Key String Table (Microsoft Jet Specific)	349
Synchronization Generations Table (Microsoft Jet Specific)	350
Chapter 20. Problem Determination Facilities	353
Replication Diagnosis Resources	353
Errors Encountered during Replication Administration	353
Errors Encountered While Running the Capture and Apply Programs	354
The Apply Program Problem Determination Facilities	354
The Apply Trail Control Table (ASN.IBMSNAP_APPLYTRAIL)	354
Apply Program Trace File	355
The Apply Program Log File	356
Capture Program Problem Determination Facilities	356
Capture Program Trace Control Table (ASN.IBMSNAP_TRACE)	357
Capture Program Trace File	357
Capture Program Log	357
Problem Determination Scenario	357
Problem Source Identification Questions	358
Chapter 21. IBM Replication Messages	361
Capture Program Messages	361
Apply Program Messages	377
Migration Messages	391
Chapter 22. Sample Invocation JCL	401
Link-Edit JCL for Capture (MVS for DB2 Version 3)	401
Link-Edit JCL for Capture (MVS for DB2 Version 4)	402
Link-Edit JCL for Capture (MVS for DB2 Version 5)	404

Bind Package JCL for Capture (MVS for DB2 Version 3)	405
Bind Package JCL for Capture (MVS for DB2 Version 4)	406
Bind Package JCL for Capture (MVS for DB2 Version 5)	407
Invocation JCL for Capture (MVS for DB2 Version 3)	408
Invocation JCL for Capture (MVS for DB2 Version 4)	410
Invocation JCL for Capture (MVS for DB2 Version 5)	411
Link-Edit JCL for Apply (MVS for DB2 Version 3)	412
Link-Edit JCL for Apply (MVS for DB2 Version 4)	413
Link-Edit JCL for Apply (MVS for DB2 Version 5)	414
Bind Package JCL for Apply (MVS for DB2 Version 3)	416
Bind Package JCL for Apply (MVS for DB2 Version 4)	419
Bind Package JCL for Apply (MVS for DB2 Version 5)	421
Invocation JCL for Apply (MVS for DB2 Version 3)	423
Invocation JCL for Apply (MVS for DB2 Version 4)	425
Invocation JCL for Apply (MVS for DB2 Version 5)	426

Part 7. Appendixes 429

Appendix A. How the DB2 Library Is Structured	431
SmartGuides	431
Online Help	432
DB2 Books	433
Viewing Online Books	437
Searching Online Books	438
Printing the PostScript Books	438
Ordering the Printed DB2 Books	439
Information Center	439

Appendix B. Tables at a Glance 441

Appendix C. What's New in IBM Replication	445
Packaging Integration	445
Improved Administration	445
Batch Application Stream Integration	446
New Target Table Types	446
New Database Data Types	446
Data Consistency	446
Improved Reliability	447
Improved Performance for the Capture and Apply Programs	447
Operational and Maintenance Enhancements	447
Expanded Support for Run-Time Processing Statements	448
New Apply Program Invocation Parameters	448
Simplified Authorization and Security	448
New Update-Anywhere Capability	448
Support for Occasionally Connected Systems	449
Support for Microsoft Databases	449

Appendix D. Notices 451

	Programming Interface Information	451
	Trademarks	452
	Trademarks of Other Companies	452
	Appendix E. Contacting IBM	453
	Glossary	455
	Index	463

Welcome to Replication

This book describes how to plan, configure, administer, and operate the IBM Replication tools available within DB2 Universal Database Version 5 Release 2 (V5.2) and as DataPropagator Relational Version 5 Release 1 (V5.1) program products.

This book is for database administrators, LAN administrators, and any other knowledge workers who have responsibility for administering and operating the Capture and Apply programs, including planning, configuring, customizing control tables, defining replication sources and targets, and maintaining replication objects.

Part 1 gives an overview of IBM Replication and provides a getting started scenario.

Part 2 describes planning considerations important for designing and implementing replication.

Part 3 provides administration task information for defining and managing replication sources and targets.

Part 4 describes how to operate the Capture and Apply programs on each platform.

Part 5 provides information about mobile replication.

Part 6 provides reference information for migration, table structures, problem determination facilities, messages, and sample JCL.

Conventions

This book uses these highlighting conventions:

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

Terminology

This book uses industry standard terminology for database, copying, and LAN concepts. Unless otherwise noted, the following components are referred to by the following abbreviated names:

Formal name	In this book it is called ...
IBM DB2 Server for OS/390 Version 5 Release 1	DB2 for MVS V5.1
IBM DataPropagator Relational Capture for MVS, Version 5 Release 1	Capture for MVS
IBM DataPropagator Relational Apply for MVS, Version 5 Release 1	Apply for MVS
IBM DB2 Server Capture for VSE, Version 5 Release 1	Capture for VSE
IBM DB2 Server Capture for VM, Version 5 Release 1	Capture for VM
IBM Capture for the DB2 Universal Database, Version 5 Release 1	Capture for DB2 Universal Database (includes all DB2 Universal Database platforms)
IBM Apply for the DB2 Universal Database, Version 5 Release 1	Apply for DB2 Universal Database (includes all DB2 Universal Database platforms)
The replication Capture tool	The Capture program for <i>any DB2 Universal Database platform</i>
The replication Apply tool	The Apply program for <i>any DB2 Universal Database platform</i>
The Solaris Operating Environment	Solaris
SCO UnixWare 7	UnixWare 7
DB2 for OS/2, DB2 for Windows NT, DB2 for AIX, DB2 for HP-UX, DB2 for Solaris, and DB2 for SCO Unixware 7	DB2 Universal Database
Capture for AIX, Capture for HP-UX, Capture for Solaris, and Capture for UnixWare 7	Capture for the UNIX** OS-based platforms
Apply for AIX, Apply for HP-UX, Apply for Solaris, and Apply for UnixWare 7	Apply for the UNIX** OS-based platforms
IBM DB2 DataPropagator for Microsoft Jet	DataPropagator for Microsoft Jet

How to Read the 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.

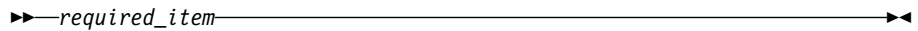
Diagrams of syntactical units other than complete statements start with the ►— symbol and end with the —> symbol.

- Keywords, their allowable synonyms, and reserved parameters, are either shown in uppercase or lowercase, depending on whether the platform is for MVS, OS/2, or

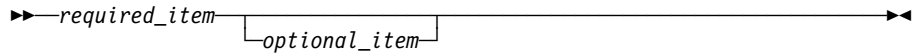
UNIX products. These items must be entered exactly as shown. Variables appear in lowercase italics (for example, *column-name*). They represent user-defined parameters or suboptions.

When entering commands, parameters and keywords must be separated by at least one blank if there is no intervening punctuation.

- Enter punctuation marks (slashes, commas, periods, parentheses, quotation marks, equal signs) 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).

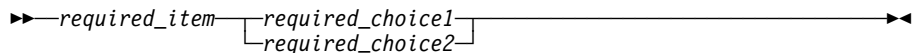


- Optional items appear below the main path.

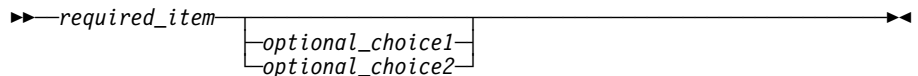


- If you can choose from two or more items, they appear vertically, in a stack.

If you *must* choose one of the items, one item of the stack appears on the main path.



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



Road Map

If you want to ...	Refer to ...
Learn about the IBM Data Replication Solution	Chapter 1, "The Complete IBM Data Replication Solution" on page 3 or see the Web site at http://www.software.ibm.com/data/dbtools/datarepl.html
Learn what's new in IBM Replication	Appendix C, "What's New in IBM Replication" on page 445.
Learn about replication concepts	Chapter 2, "Introduction to the Replication Tools" on page 9.
Learn how other customers have implemented replication	Chapter 4, "Replication Usage Scenarios" on page 33 or see the customer case studies at Web site http://www.software.ibm.com/data/dpropr/casestudy.html
Plan your replication environment at the system level	Chapter 5, "System Planning" on page 39.

If you want to ...	Refer to ...
Plan your replication environment at the application level	Chapter 6, "Application Planning" on page 53.
Migrate from DPROPR V1 to IBM Replication	Chapter 18, "Migrating from DataPropagator Relational Version 1 to IBM Replication Version 5" on page 271 and see the online migration guide at Web site http://www.software.ibm.com/data/dpropr/mig.htm
Learn about mobile replication	Part 5, "Mobile Replication" on page 247.
Get started with a sample replication task	Chapter 3, "Getting Started" on page 21.
Set replication preferences in the Tools Settings notebook	"Setting Replication Preferences in the DB2 Tools Settings Notebook" on page 85.
Customize the replication control tables	"Working with Customized Replication Control Tables" on page 88.
Define and manage replication sources	Chapter 8, "Working with Replication Sources" on page 93.
Define and manage replication targets	Chapter 9, "Working with Replication Targets" on page 103.
Configure the Capture and Apply programs	Part 4, "IBM Replication Capture and Apply" on page 127; see the chapter for your platform.
Operate the Capture and Apply programs	Part 4, "IBM Replication Capture and Apply" on page 127; see the chapter for your platform.
Learn about and run IBM DB2 DataPropagator for Microsoft Jet	Chapter 17, "Mobile Replication Using IBM DB2 DataPropagator for Microsoft Jet" on page 259
Troubleshoot errors for the Capture and Apply programs	Part 4, "IBM Replication Capture and Apply" on page 127; see the chapter for your platform.
Learn about the target table structures	Chapter 19, "Table Structures" on page 295.
Debug error messages	Chapter 21, "IBM Replication Messages" on page 361 and Chapter 20, "Problem Determination Facilities" on page 353.
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.

To locate information on other topics, see Appendix A, "How the DB2 Library Is Structured" on page 431 for a complete description of the DB2 library.

Part 1. Introduction to IBM Replication

This part introduces IBM Replication: the tools, the concepts, and how to get started using replication.

Chapter 1. The Complete IBM Data Replication Solution

Data replication consists of two basic processes:

- Capturing changed data from a table or view in a *source* database
- Copying (or “applying”) changed data from a *source* table to one or more *target* tables in the same or different databases

The IBM replication solution lets you determine how the changed data is to be captured and how (and under what conditions) this data is to be copied to the targets. *Replication administration* is the process of creating and managing these two basic processes. Replication administration includes:

- Defining a table, database, or view as a source table for change capture and replication
- Creating a replication target table
- Defining the conditions under which the data from the defined source is copied to the defined target

IBM's complete data replication solution delivers comprehensive data replication tools that function together in a single solution. These tools include:

- IBM Replication:
 - DB2 Universal Database V5.2 replication tools: the Control Center replication administration features and the Capture and Apply programs
 - IBM Capture and Apply for MVS V5.1
 - Capture for VSE and VM V5.1: integrated with IBM DB2 Server for VSE and VM V5.1
- DPROPR V1 for support of DB2 DataJoiner Version 2
- DataPropagator Relational Capture and Apply for OS/400 V3.1
- IBM DataPropagator NonRelational (DPROP NR)
- IBM DataRefresher
- IBM DB2 DataJoiner and the DataJoiner Replication Administration Tool
- Lotus NotesPump

This chapter provides overview information about how these products support the complete data replication solution. This book describes IBM Replication features only. For more information about IBM's Data Replication Solution, see the Web site <http://www.software.ibm.com/data/dbtools/datarepl.html>

How the Tools Work Together

IBM Replication is a set of easy-to-use, automated copy tools that replicate data from sources to targets. Data can be copied:

- Among DB2 for MVS, DB2 for VSE & VM, DB2 Universal Database, and DB2 for OS/400 database servers
- Across platforms for MVS, VM, VSE, OS/400, AIX, HP-UX, SCO UnixWare 7, OS/2, Windows NT, Windows 95, and the Solaris operating environment
- On non-DB2 sources and targets through DB2 DataJoiner; such as Oracle, Sybase, Informix, and Microsoft SQLServer

You can tailor or enhance data as it is copied and deliver detailed, subset, summarized, or derived data when and where it is needed. You can use IBM Replication to define, synchronize, automate, and manage copy operations from a single control point for data across your enterprise. Figure 1 shows an example of data replication between various servers.

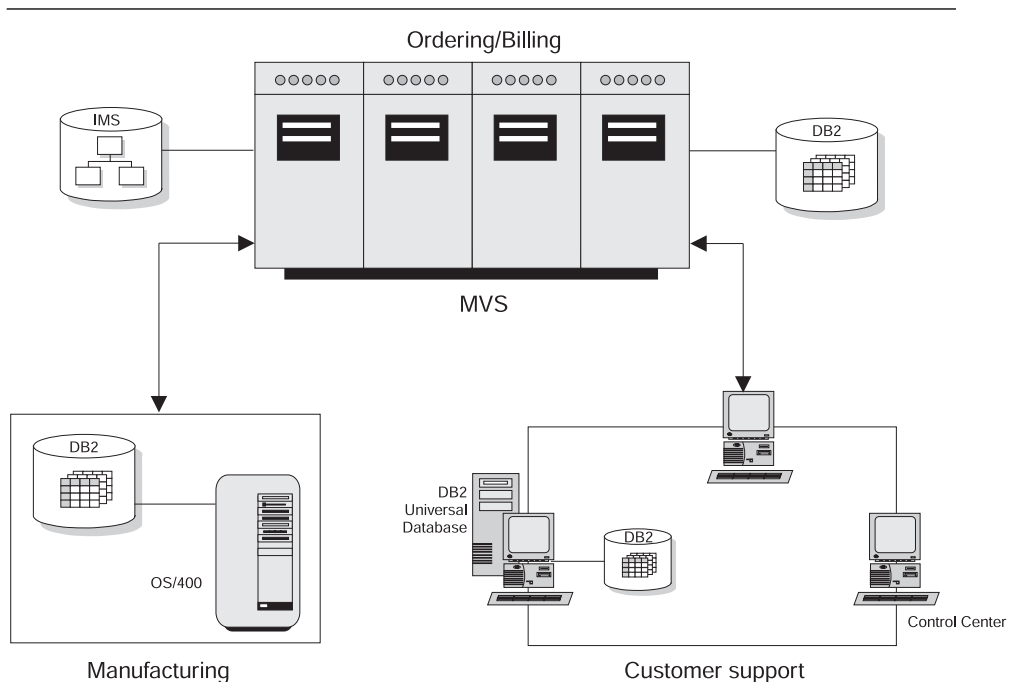


Figure 1. IBM Replication Tools. The IBM Replication tools copy data across an enterprise.

By supporting sources and targets that include the DB2 family, IMS, VSAM, Oracle, Sybase, Microsoft, and Lotus Notes, IBM's solution ensures that you have timely, reliable, and consistent data across your enterprise.

IBM's architecture is built on standard SQL to leverage the database engine capabilities for data enhancement, network connectivity, and data security. IBM database support for distributed relational database architecture (DRDA) enables replication to other DRDA application servers.

An architected data staging area supports interoperability among multivendor databases, between heterogeneous data models, and among products from independent software vendors.

Figure 2 illustrates how IBM's replication solution supports heterogeneity in replication environments.

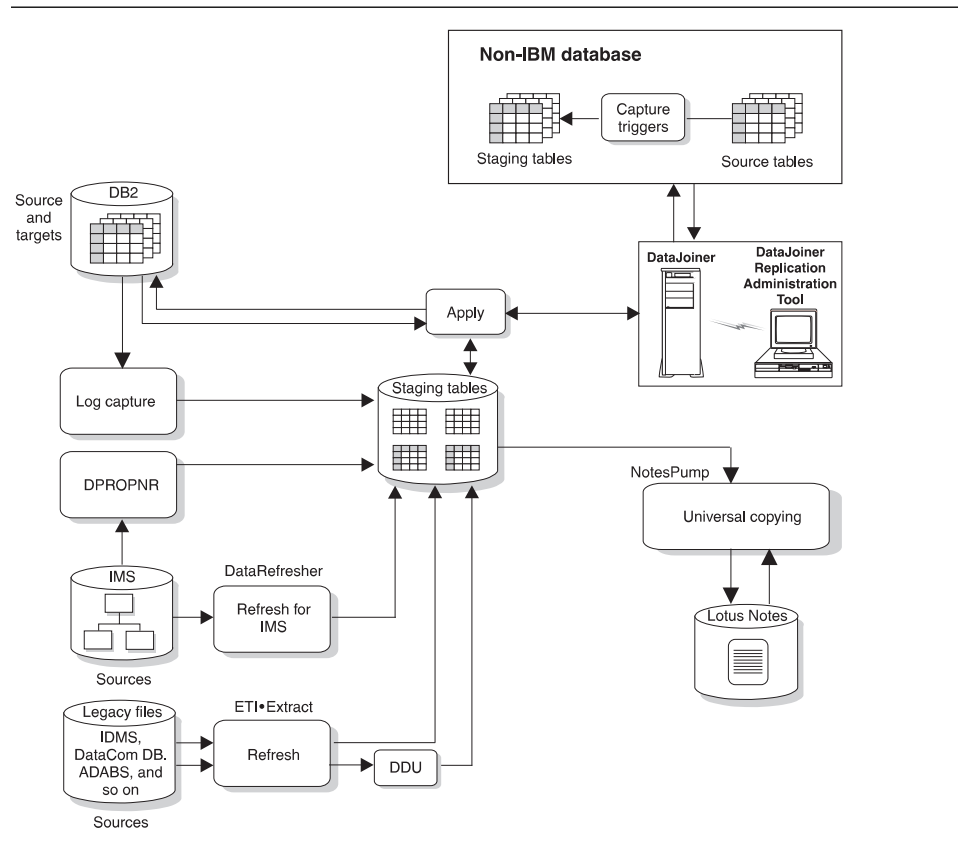


Figure 2. IBM's Data Replication Solution and Product Environment

The IBM Replication tools capture source data from DB2 and IMS. Captured data for copying is placed into staging tables before the Apply program reads the tables and copies the changes. Updates need to be captured only once. You can stage the updates at any number of intermediate sites, which allows for flexible data distribution.

The Apply program filters, enhances, and replicates data to the target databases. DB2 DataJoiner enables the Apply program to read from or write to multivendor sources and

targets. NotesPump acts as an alternative Apply program for multivendor, ODBC, and Notes targets.

Legacy Data Sources

DataPropagator NonRelational and DataRefresher provide a coordinated replication solution for IMS data sources, by bringing IMS, VSAM, and flat file data into the staging area, making it available for replication to client/server targets. You can use DataRefresher to copy data (full refresh copy) from IMS to DB2. You can use DPRPNR to capture changes from the IMS log and replicate them to a DB2 staging table. Replicating from IMS to DB2 makes it possible to implement decision-support applications that use relational technology. You can replicate to a staging table on DB2 for OS/390, which allows IBM Replication to further filter, enhance, distribute, and apply the data to relational databases across your enterprise.

In addition to IMS sources, DataRefresher can also copy data from VSAM and flat file sources. DataRefresher produces data extracts in load-ready format for any DB2 database and for both user tables or data staging area targets. The Data Difference Utility (DDU) can then detect recent changes in these DB2 load files and propagate these changes efficiently.

You can use DataRefresher's GUI to specify data definitions and data extract requests. You can then share the definitions and requests with DataPropagator NonRelational to replicate changes from source to target databases.

Multivendor Data Sources and Targets

IBM's DB2 DataJoiner extends replication to multivendor sources and targets including Oracle, Sybase, Microsoft SQLServer, and others. DB2 DataJoiner is a multidatabase server that provides a new level of transparent data access. It can help businesses meet the challenge of efficiently accessing distributed data by enabling users to develop a virtual, enterprise-wide relational database.

DB2 DataJoiner provides a single-site image of all of your data, relational and nonrelational, local and remote, from IBM and non-IBM platforms, as though the data were local. DB2 DataJoiner masks differences in SQL dialects, data access methods, networking protocols, operating systems, and so on. For many businesses, replication is the mechanism of choice to improve access to data in heterogeneous environments. DB2 DataJoiner's advanced replication capabilities extend the benefits of homogeneous replication—reduced network traffic, enhanced data availability, better performance, and more data sharing—to heterogeneous environments. IBM Replication technology is integrated in DB2 DataJoiner to provide both synchronous and asynchronous access to data. DB2 DataJoiner provides continuous update replication, point-in-time replication, and integrated replication administration that streamlines the replication process.

Interoperability with Lotus Notes and ODBC Accessible Data Stores

Lotus NotesPump provides data replication between relational database management systems, ODBC accessible data stores, and Lotus Notes databases. NotesPump can copy data bidirectionally between supported data sources.

When used alone, NotesPump refreshes the target data from a direct query of the source data. NotesPump Release 2 understands the IBM Replication staging area architecture, and can act as an alternative Apply program to apply changes to multi-vendor, ODBC, and Notes data stores.

For example, to replicate DB2 data to Notes, you can use IBM Replication to capture the changes and create a staging table. Then, NotesPump can read the changes from the staging table and apply the changes to a Notes database.

Services

IBM and IBM Business Partners offer consulting and services supporting the IBM 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.

Contact your IBM software provider, or call in the U.S. and Canada, 1-800-IBM-3333, for additional information on IBM products and services.

IBM Replication Education

The following IBM Replication classes are provided by IBM Education and Training:

- Data Replication: DataPropagator Relational Basic Usage (DW140)
- Data Replication: DataPropagator Relational 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.

Chapter 2. Introduction to the Replication Tools

IBM Replication consists of the replication administration features of the Control Center and two tools, the Capture and Apply programs. The Control Center provides administration support for the replication environment with objects and actions that define and manage source and target table definitions. The Capture and Apply programs are responsible for capturing the updates to the source tables and applying the changes to the target tables.

The following sections describe how the IBM Replication tools work and interact with each other.

The Control Center

The Control Center is the database administration tool for the DB2 Universal Database and is the tool that you use for replication administration. The Control Center automates many initialization functions, such as creating target tables and control tables when you specify target copy information.

This section briefly explains the main replication tasks that you perform from the Control Center, the objects that you use in the Control Center, the types of target table copies that you can create, auto-registration, and the control tables that are created when you use the Control Center. See Part 3, “Administering Your Replication System” on page 81 for more detailed information about Control Center replication tasks. See *Administration Getting Started* for your platform for more information about the Control Center. See the DB2 DataJoiner V2 documentation for details about heterogeneous replication administration.

Tasks

You must be a replication administrator to perform the replication administration tasks from the Control Center.

The main Control Center administration tasks for replication are:

Defining tables as sources, called replication sources

Replication sources are DB2 tables, or views, used as sources for copying data to a target table. When you define a table as a replication source, you specify how that table will be used in your replication scenario. For example, you specify which columns in the table are available for replication, whether the data existing in a column before a change should be captured in addition to the change to the column. When you define a replication source, you set the information that the Capture program will use. The Capture program uses the information when it looks at the DB2 log to find changes for the table. The Capture program then copies the changes it finds in the log to a change data table. A change data table is created for every replication source that you define.

Defining replication targets, called replication subscriptions

When you define a replication subscription, you are specifying target tables, their structure, the sources for the target tables, and the schedule for the updates to the target tables. You can create new columns in the target tables with the exact information from existing columns in the source table. You can also create new columns in the target table based on:

- The computation of one or more existing columns in the source table
- Other SQL functions that summarize or average all of the rows from a source table into a single summary row

You can also specify SQL statements that you want to execute before or after the subscription is executed.

Other tasks include removing replication sources that are no longer sources for copies, cloning replication sources and subscriptions to other servers, and removing replication subscriptions.

You do *not* use the Control Center to operate the Capture and Apply programs.

Objects

You can access your replication source and target definitions through the Control Center. There are three containers that organize the objects that you use to set up and maintain your replication environment:

Tables folder

Contains user-defined DB2 relational tables, as well as system catalog tables. You select the tables in the folder to define replication sources.

Replication Sources folder

Contains tables that have been defined as replication sources. These can be DB2 tables, system tables, views, or target tables redefined as sources for replication.

Replication Subscription folder

Contains replication subscription definitions for copying updated source data to target tables.

You work with three GUI objects: **Tables**, **Replication Sources**, and **Replication Subscriptions**, to manage replication sources and targets.

Figure 3 on page 11 shows the Control Center window.

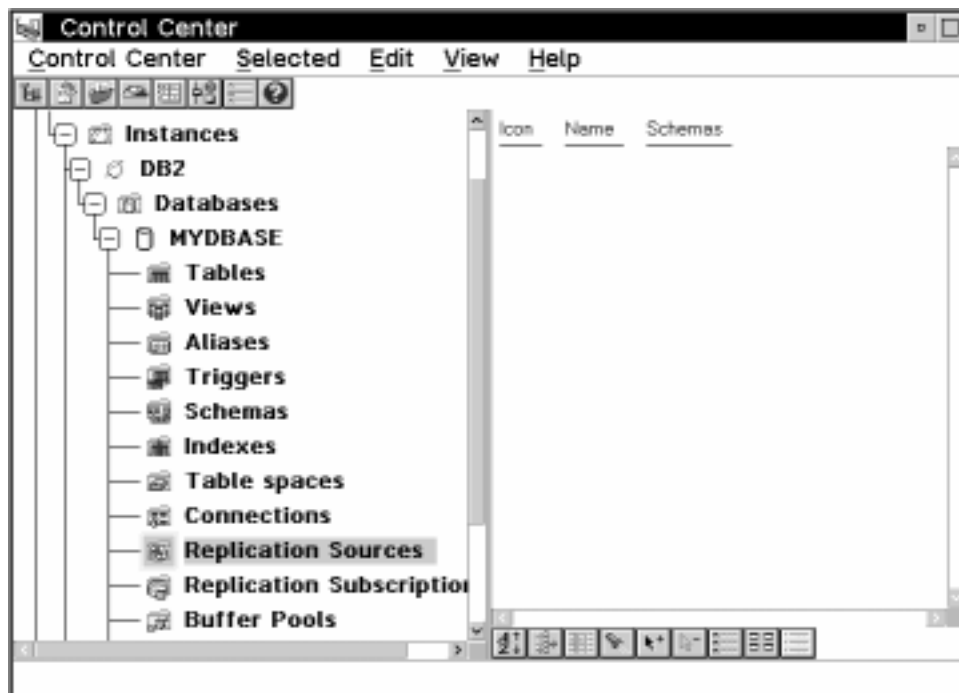


Figure 3. The Control Center and the Replication Folders. The Control Center and the replication folders provide administration tools for replication.

Types of Copies

IBM Replication offers you many choices for target table structures. Depending on the kind of data you want in your application system, you can choose how changes are copied to the target table. Target tables function as historical or trend information, sources for update-anywhere replication, or simply an identical copy of the source table. The different types of target tables are:

User copy

A complete, condensed copy of the replication source table that must have a primary key.

Point-in-time

A complete, condensed copy of the replication source table at a certain point in time that must have a primary key. This table contains timestamp columns to indicate when a transaction occurred.

Base aggregate

A history table in which new rows are appended for each subscription cycle using the result of an SQL column function calculation against the replication source table data.

Change aggregate

A history table in which a new row is appended for each changed row in the replication source table using the result of an SQL column function calculation against only recently changed data.

Condensed, noncomplete consistent change data (CCD)

A small staging table that, when defined locally to the source, is useful for netting out “hot spot” updates before replication to other sites.

Condensed, complete CCD

A full-sized staging table useful for efficient remote staging, which allows for remote copies to be both initialized and maintained without needing to access the original source each time it is updated.

Noncondensed, noncomplete CCD

A table that is initially empty and is appended by each insert, update, and delete action; useful for auditing purposes.

Noncondensed, complete CCD

A table that starts out as a copy of the full-sized source table and is appended by each insert, update, and delete action. It retains all information about the source table and supports “as of” historical queries. For example, it can return an answer to a query as if you had run the query against the replication source table last Tuesday, or a month ago, or yesterday.

Replica

A target table that can be updated. Changes to this table are replicated back to the replication source table. This table is used in update-anywhere scenarios.

Auto-Registration

Some target table types are designed to become sources for additional replication. The CCD table and the replica table are almost always used in multiple tier or circular replication scenarios and are automatically defined as replication sources when you define them as part of a replication subscription.

The Replication Control Tables

IBM Replication tools use control tables to communicate with each other and to manage replication requests such as defining and managing sources and targets, capturing changes, and replicating changes. These control tables are located at the source, control, and target servers, which are defined in “Terminology” on page xvii.

When you use the Control Center to perform replication administration tasks, it creates and maintains the replication control tables used by the Capture and Apply programs. These tables are created and stored at the source, control, and target servers. The workstation where the Control Center is located must be connected to all the databases where source tables exist and the Capture program runs (the source server), all the databases where the target tables will be created and the Apply program usually runs (the target server), and all the databases where the subscription control tables are stored (the control server). The exception is the “occasionally connected,” or mobile environment, in which you can create and save as SQL a replication subscription definition, transfer it to another machine, and run it from there.

Most administrators need to customize the control tables for their site requirements or DB2 platforms. To customize the control tables, you must perform some steps *before* you perform replication requests (that is, *before* you define sources or subscriptions with the Control Center). For more information on customizing control tables, see “Working with Customized Replication Control Tables” on page 88.

The Capture Program

The Capture program is the replication tool that captures the changed data and makes the changed data available for replication. The Capture program runs at the source server database.

The Capture program captures changes made to data in replication source tables by reading the database log or journal. It places the captured changes into change data tables. There is only one Capture program at each domain of the database directory, that is, one Capture program in each of the following domains:

- DB2 Universal Database database
- DB2 for OS/390 subsystem in a non-data-sharing environment
- DB2 for OS/390 data-sharing group¹
- DB2 for VM or DB2 for VSE database

The Capture program usually runs continuously, but you can stop it while running utilities or modifying replication sources. The Capture program runs independently of the Control Center, but uses control information created by the Control Center.

Tasks that you can perform with the Capture program include:

- Starting the Capture program
- Scheduling the Capture program
- Stopping the Capture program
- Suspending the Capture program temporarily
- Resuming the Capture program
- Reinitializing the Capture program
- Monitoring the Capture program
- Pruning the control tables

See the Capture and Apply chapter for your Capture program platform for instructions about performing these tasks.

How the Capture Program Captures Changes

DB2 records every transaction in a log file for diagnostic and recovery purposes. The Capture program monitors the DB2 log to detect change records from source tables that are defined as replication sources. The Capture program retrieves change and commit information from the active and archive logs on DB2 for MVS 4.1 or higher and

¹ For DB2 for MVS/ESA Version 4.1 or higher, only one instance of the Capture program can run for all subsystems that are members of a data-sharing group because tables in a data-sharing group are shared among multiple subsystems.

DB2 Universal Database. ² These records contain a before-image and after-image of the table row. The Capture program captures these changes in the change data (CD) tables. The Capture program also maintains information about committed units of work in the unit-of-work (UOW) table. This table is joined with the CD table to identify and replicate committed updates. The Apply program can then copy the committed updates to the target site and apply them to the copy of the source table (the target table). Figure 4 shows the relationship between the Capture program, the DB2 log, the source table, and the control tables.

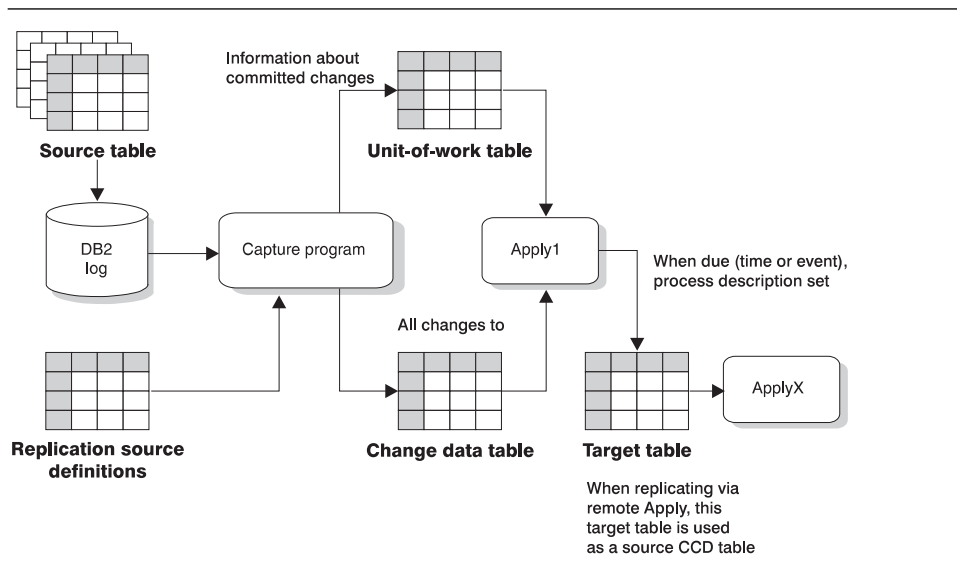


Figure 4. The Capture Program at the Source Server. The Capture program at the source server reads the DB2 log and captures changed data in the change data (CD) table.

Control Tables Used by the Capture Program

The Capture program uses control tables to manage capturing changes from multiple source tables. These tables are created when you set up administration with the Control Center. The control tables used by the Capture program are located at the source server. The following list describes the source server control tables:

Register

Contains the name of the source table and the name of the change data table associated with the source table.

Critical section

Contains the Apply qualifier and is used for concurrency control between the Capture and Apply programs.

² Capture for MVS can retrieve only the active log on DB2 for MVS 3.1. Capture for VSE and VM 5.1 can read only the active log on DB2 for VSE & VM.

Tuning parameters

Contains performance tuning information specific to the Capture program.

Trace

Contains trace information for the Capture program.

Unit-of-work (UOW)

Contains committed unit-of-work information used by the Apply program.

Pruning control

Contains information about how far through the CD table the Apply program has progressed in replicating changes to the target tables. The Capture program uses this information to prune the CD table.

Warm start

Contains information about the units-of-work in progress and starting sequence information necessary for restarting the Capture program without performing a full refresh.

Change data (CD)

Contains the changed data read by the Capture program from the database log or journal, and is a source for update copies. There is one CD table for each replication source table.

These tables are described in more detail in Chapter 19, “Table Structures” on page 295.

The Apply Program

The Apply program is the replication tool that replicates changes of the source table data to the target table.

The Apply program reads the changed data previously captured and stored in a CD table and applies the changes to target tables. The Apply program also reads data directly from source tables when copying the entire source table to the target table (called a *full refresh copy*).

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).

The Apply program runs independently of the Control Center, but uses control information that the Control Center creates. The control information that the Apply program uses is stored in tables at the control server.

Tasks that you can perform with the Apply program include:

- Starting the Apply program
- Scheduling the Apply program

- Stopping the Apply program

See the Capture and Apply chapter in this book for your Apply program platform for instructions about performing these tasks.

How the Apply Program Replicates Data

When the Apply program reads the changed data stored in CD tables, it applies it to target tables at either local or remote servers. It can also perform column functions, such as SUM and AVG, on the data from the source table or CD table and append the result to the target tables. The Apply program can run at any server that can connect through the SQL CONNECT statement to each database server where source and target tables reside. Figure 5 shows the Apply program's relationship with the source server control tables, the subscription definition control tables, and the target table.

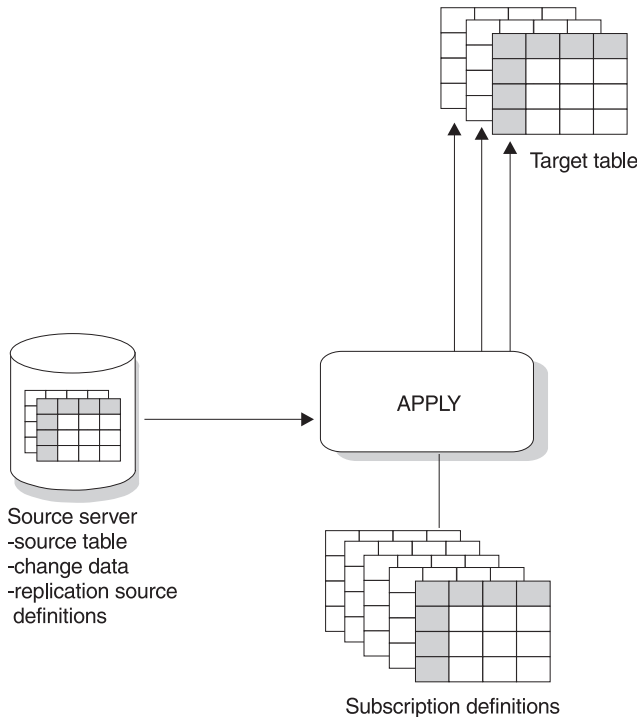


Figure 5. The Apply Program. The Apply program reads the source server tables, manages the subscription control tables, and updates the target tables.

The Apply Processing Cycle

In the more common "pull" mode, the Apply program runs at the target server and connects to the source and control server to pick up changed data and to read the control tables. Table 1 on page 17 describes how the Apply program completes a replication cycle from the target server.

Table 1. The Apply Program Cycle: A high-level overview of how a replication subscription is processed.

Step	Server
1. Look for work; check the subscription control tables.	Control server
2. Pick up recent change data to be applied to the target table.	Source server
3. Write the answer set into a local "spill" file (possibly an in-memory file).	Target server
4. Apply the change data in the spill file to the target table.	Target server
5. Update subscription status.	Control server
6. Report subscription progress in the pruning control table.	Source server

How the Apply Program Selects a Source Table

When the Apply program refreshes or updates a target, it chooses from a list of potential source tables in the following order:

1. The CCD table associated with the defined replication source table
2. The CD table associated with the replication source table
3. The replication source table associated with the replication subscription

In some cases, the source table considered first might not be the one that the Apply program reads from. For example, if the target table is a point-in-time table but has not yet been initialized, the Apply program must use a source table that is complete. If the CCD table is not complete (that is, `CCD_COMPLETE=N` in the register control table), the replication source table is selected as the refresh source. (The complete attributes are set when the CCD table is created.)

The Apply Qualifier

When the Apply program is started, it is supplied with a job qualifier, called an *Apply qualifier*, independent from a logon user ID. The Apply qualifier (`APPLY_QUAL`) is associated with individual replication subscriptions and is responsible for replicating only those replication subscriptions. You specify a value for the Apply qualifier when you define a subscription with the Control Center.

The Apply qualifier allows a user ID to run more than one instance of the Apply program, using different Apply qualifiers.

The Control Server

The control server is the logical server that contains the subscription control tables. Each Apply program is associated with a control server, which you specify when you invoke 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 tables in the control server. However, locating the control server at the source server, if it is a server in a secure environment, can provide improved security.

Each control server is associated with one or more Apply qualifiers. The qualifier associates a control server with an Apply program and a replication subscription. This association allows a replication subscription to be serviced by one Apply program and identifies where the control tables for that subscription are located. You cannot easily change the Apply qualifier, so it is important to plan carefully.

Control Tables Used by the Apply Program

The Apply program relies on the following control tables to control the replication of source table changes to the target tables:

Source server control tables

The control tables used by the Capture program are: the register table, the unit-of-work table, the pruning control table, the critical section table, and the change data tables.

Control server control tables

Apply trail

Records statistics about full refresh and differential refresh (update) copies that are performed during each subscription set cycle.

Subscription set

Defines the characteristics of each subscription set, such as the subscription set name, the Apply qualifier that is associated with the subscription set, and the timing of the subscription set.

Subscription statements

Contains the SQL statements or names of stored procedures to be run before or after the subscription member (or subscription set) is processed.

Subscription events

Contains the timing information for copying the subscription set based on event triggering. This table is maintained by a user application or by other subscription sets. This table itself can be replicated if you want to distribute event notifications.

Subscription targets member

Contains information for an individual subscription set that maps a specific source table to a specific target table. Subscription sets of DB2 views usually have many subscription members for the same target table.

Subscription columns

Contains supplemental information about each copied column, such as renamed columns or computed columns.

These tables are described in more detail in Chapter 19, “Table Structures” on page 295.

Full Refresh and Differential Refresh Replication

The Apply program copies data from the source to the target either by full refresh or by differential refresh. For full refresh copying, the Apply program copies the entire source table and copies it to the target table. The Capture program does not capture changes, and there are no CD or CCD tables involved. You can specify full refresh copying when

you define the replication source table. If you have large tables, you might want to use a fastload program to simulate an initial full refresh copy. See the Capture and Apply chapter for your platform in this book for more information about fastload programs such as ASNLOAD.

For differential refresh, the Apply program copies only the changed data from the CD table to the target table. The first time the Apply program copies data to the target table or after a cold start of the Capture program, the Apply program uses full refresh copying to populate the target table. (See the Capture and Apply chapter for your platform in this book for more information about cold start.) After the target table is populated, differential refresh is used. This type of copying is the default and is used unless you specify otherwise when you define the replication source.

Chapter 3. Getting Started

This chapter presents a high-level overview and practical application of the following steps required to set up a replication environment:

1. Plan for a replication scenario at the system level and application level as described in Chapter 5, “System Planning” on page 39 and Chapter 6, “Application Planning” on page 53.
2. Define a replication source as described in Chapter 8, “Working with Replication Sources” on page 93.
3. Define a replication subscription as described in Chapter 9, “Working with Replication Targets” on page 103.
4. Configure the Capture and Apply programs. See the Capture and Apply chapter for your particular platform for configuration information.
5. Start the Capture program as described in the Capture and Apply chapter for your particular platform.
6. Start the Apply program as described in the Capture and Apply chapter for your particular platform.

You do not need to refer to the detailed information referenced in these steps to perform the scenario in this chapter. The terms you will encounter, such as Apply qualifier, are explained in Chapter 2, “Introduction to the Replication Tools” on page 9 and in the glossary.

Performing a Replication Scenario for Windows NT

The hypothetical scenario in this section and the following sections enable you to acquire preliminary experience using the Control Center and the Capture and Apply programs. This scenario walks you through the steps above using hypothetical data to copy a simple DB2 source table to a target DB2 database for Windows NT. This type of copy is known as a *user copy*, which is a complete, condensed copy of the replication source table without the timestamp columns.

This scenario uses the sample source table shown in Table 2 on page 22. This is the DEPARTMENT table in the SAMPLE database that is contained in the DB2 Universal Database. You can create the SAMPLE database from the First Steps icon in the DB2 Universal Database.

Table 2. Sample Source Table Rows

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	—

Nomenclature for the scenario is as follows:

DEPARTMENT	Source table name
DEPARTCOPY	Target table name
SAMPLE	Source server database
COPYDB	Target server database
COPYDB	Control server database
DEPTQUAL	Apply qualifier
DB2	Instance name

This scenario accomplishes the following objectives:

- Prevents capture changes for the LOCATION column
- Prevents before-image changes for ADMRDEPT
- Captures changes for all other columns
- Sends copies to the target table at a rate of 1 copy each minute, ending only when Apply for NT is stopped

To set up the environment to run this scenario for Windows NT, do the following:

1. Create the SAMPLE database to be used as the source server database.
2. Create a database called COPYDB to be used as the target and control server database.
3. Verify that these databases appear in the Control Center.

The following sections explain the steps required to define a user copy for DEPARTMENT:

- “Defining a Replication Source” on page 23
- “Defining a Subscription” on page 24
- “Configuring the Capture and Apply Programs” on page 25
- “Starting the Capture and Apply Programs” on page 28

Defining a Replication Source

To define a replication source:

1. Open the Define as Replication Source window:
 - a. From the Control Center, expand the object tree under the Systems icon until you see the SAMPLE database.
 - b. Click on SAMPLE, then click on the Tables folder. Any existing tables are displayed in the pane on the right side of the Control Center.
 - c. Click mouse button 2 on the source table, DEPARTMENT, and select **Define as Replication Source - Custom** from the pop-up menu.
2. From the **Available columns** box, select LOCATION and clear the **Define as source** check box. This specifies that this column is unavailable as a source column during subscription and cannot be replicated. This action also clears the **Capture before image** check box.
3. From the **Available columns** box, select ADMRDEPT and clear the **Capture before image** check box. This action prevents before image replication of the source column to the target table.
4. Click on **OK** to save the values and close the window. A confirmation window opens.
 - a. The **Save to SQL file and run later** radio button is activated as the default. Click on **OK**. The Control Center will generate the necessary SQL.
 - b. Type the file name where you want to save the file in the Run SQL files window and click on **OK**.
 - c. Click on **OK** in the DB2 message window to save the values and automatically close the Define as Replication Source window.
5. Click mouse button 2 on the **Replication Sources** folder and select **Run SQL files**.
6. Run the SQL file you named above.
7. Click on **OK** in the DB2 message window when the message appears that the script file has run successfully.
8. Click mouse button 1 on the **Replication Sources** folder. The replication source, DEPARTMENT, appears in the contents panel of the Control Center.

Now the source table DEPARTMENT has been defined as a replication sources table. Additionally, the change data table for DEPARTMENT and all the Capture control tables are created in the default table space for the SAMPLE database. If you want to create the tables in other table spaces, see "Working with Customized Replication Control Tables" on page 88 and "Customizing and Running Replication SQL Files" on page 90 for more information.

Defining a Subscription

To define a subscription to copy the changes from DEPARTMENT, the source table, to the target table named DEPARTCOPY:

1. From the Replication Sources list displayed in 8 on page 23 above, click mouse button 2 on the DEPARTMENT object on the right pane of the Control Center, then select **Define subscription**. The Define Subscription window opens.
2. Set up the target table and subscription definition.
 - a. Type DEPTSUB in the **Subscription name** field.
 - b. Type COPYDB in the **Target server** field. This database is where the target table will reside.
 - c. Type DEPTQUAL in the **Apply qualifier** field. The case-sensitive character string identifies the subscription definitions unique to each instance of the Apply program that will run this subscription.
 - d. Type DEPARTCOPY over the default name of the target table. The Control Center adds a table qualifier prefix to the target table name.
 - e. Specify that you want to create the target table by selecting the **Create table** check box for the DEPARTCOPY target table.
3. Click on the **Advanced** push button. The Target Type page of the Advanced Subscription Definition notebook opens.
 - a. Leave the **User Copy** radio button selected as the default.
 - b. Click on the **Target Columns** tab.
 - 1) Select the check box under **Subscribe** for XDEPTNO.
 - 2) Check the **Primary key** check box next to DEPTNO.
 - c. Click on the **Rows** tab and type the following WHERE clause:
DEPTNO >= 'A00'
 - d. Click on **OK** to save these settings and return to the Define Subscription window.
4. Define SQL statements to be processed during subscription run time:
 - a. Click on the **SQL** push button. The SQL window opens.
 - b. Click on the **Add** push button. The Add SQL window opens.
 - c. Type the following processing statement in the **SQL statement** field:

```
DELETE FROM ASN.IBMSNAP_APPLYTRAIL
WHERE LASTRUN < (CURRENT_TIMESTAMP - 1 DAY)
```
 - d. Type the SQLSTATE value 02000 in the **SQLSTATE** field and click on **Add**. This value is added to the **Acceptable SQLSTATE values** list box.
 - e. Click on the radio button that indicates you want to submit the statement before the subscription is processed.

- f. Click on **OK**. The statement is added to the list box in the SQL window and the Add SQL window closes.
 - g. Click on **OK** to return to the Define Subscription window.
5. Specify when and how often to replicate the subscription by clicking on the **Timing** push button. The Source to Target page of the Subscription Timing notebook opens.
 - a. Keep the current date in the **Start date** field.
 - b. Keep the current time in the **Start time** field.
 - c. Select the **Time-based** check box and the **Using relative timing** radio button.
 - d. Use the spin buttons on the **Minutes** field to select 1-minute intervals.
 - e. Use the spin buttons on the **Hours** field to change the default number to 0.
 - f. Click on the **Data Blocking** tab.
 - g. Click on the spin buttons and select 5 as the number of minutes at a time that Apply will copy committed data.
 - h. Click on **OK** to save these values and return to the Define Subscription window.
6. Click on **OK** to submit the subscription. The Subscription Information window opens.
 - a. Type in COPYDB.
 - b. Select the **Save to SQL file and run later** radio button and click on **OK**.
 - c. Type the file name where you want to save the file in the Run SQL files window and click on **OK**.
 - d. Click on **OK** in the DB2 message window to save the values and close the Define Subscription window.
7. Click mouse button 2 on the **Replication Subscriptions** folder under the SAMPLE database and select **Run SQL files**.
 - a. Run the SQL file you named above.
 - b. Click on **OK** in the DB2 message window when the message appears that this file has run successfully.
8. Click mouse button 1 on the **Replication Subscriptions** folder under the SAMPLE database. The DEPTSUB subscription appears as an instance on the contents pane of the Control Center.

Configuring the Capture and Apply Programs

The following sections provide instructions for configuring the Capture and Apply programs for Windows NT, and for providing end-user authentication at the source server.

Configuring the Capture Program

To configure the Capture program:

1. From the source server system, ensure that the user ID under which the Capture program is to run has the required privileges:

- Execute privilege on the Capture packages
- DBADM or SYSADM privileges for the database

2. Log on with the user ID.

3. Update the source database with LOGRETAIN on by entering:

```
DB2 UPDATE DATABASE CONFIGURATION FOR SAMPLE USING LOGRETAIN ON
```

4. Enable database logging by entering:

```
DB2 BACKUP DATABASE SAMPLE
```

5. From the DB2 command line processor or the Command Center, connect to the source server database by entering:

```
DB2 CONNECT TO SAMPLE
```

6. Change to the \SQLLIB\BND directory, which is where the bind files are located.

7. 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
```

If your system does not support uncommitted read format, substitute CS (cursor stability format) for UR instead.

These commands create a list of packages, the names of which can be found in the CAPTURE.LST file.

Configuring the Apply Program

To configure the Apply program:

1. From the source, control, and target server system, ensure that the user ID under which the Apply program is to run has the required privileges:

- Execute privilege on ASNAPPLY
- DBADM or SYSADM privileges for the database

2. Log on with the user ID.

3. Connect to the source server database by entering:

```
DB2 CONNECT TO SAMPLE
```

4. Create and bind the Apply package to the source server database by entering both of the following commands:

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

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

These commands create a list of packages, the names of which can be found in the APPLYUR.LST and APPLYCS.LST files.

5. Connect to the target server database by entering:

```
DB2 CONNECT TO COPYDB
```

6. Create and bind the Apply package to the target server database by entering both of the following commands:

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

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

These commands create a list of packages, the names of which can be found in the APPLYUR.LST and APPLYCS.LST files.

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 for the Apply program to use when connecting to the source server. It is advisable to limit read access of this file to the user ID that will run the Apply program.

The password file must meet the following criteria:

- Be named as shown:

```
<APPLYQUAL><instname><CNTLSRVR>.PWD
```

Where:

APPLYQUAL

The Apply qualifier in upper case. The Apply qualifier is case sensitive and must match the value of APPLY_QUAL in the subscription set table.

instname

The instance name that the Apply program runs under (value of DB2INSTANCE).

CNTLSRVR

The name of the control server in upper case.

- Reside in the directory from which the Apply program starts.
- Contain all server-name/password pairs for the file. The pairs enable you to have a different (or the same) password at each control server. The Apply user ID is used for all connections.
- Have one or more records using the following format:

```
SERVER=server_name PWD=password USER=userid
```

Where *server_name* is the source, target, or control database. The file cannot include blank lines or comment lines.

To create a password file:

1. Type the following records in your editor. Do not include blank lines or comment lines in the file.

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

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

2. Save the file as DEPTQUALdb2COPYDB.PWD.

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

Starting the Capture and Apply Programs

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

To start the Capture program from the SAMPLE source server:

1. Enter the following command from the Windows NT window:

```
ASNCCP SAMPLE
```

No confirmation message appears. The Capture program has successfully started if it is running and no command prompt appears. The Capture program has initialized but does not start capturing changes for the registered source tables until you start the Apply program and it completes its initial full refresh. See the troubleshooting section in your particular platform chapter for problem determination.

To start the Apply program:

1. Enter the following command from another Windows NT window:

```
ASNAPPLY DEPTQUAL COPYDB
```

No confirmation message appears. The Apply program has successfully started if the program is running and no command prompt appears. You can also check the ASN.IBMSNAP_APPLYTRAIL table for status information. See the troubleshooting section in your particular platform chapter for problem determination.

Stopping the Capture and Apply Programs

To stop the Capture and Apply programs after the user copy replication is complete:

1. Stop the Capture program.
 - a. From a DB2 command window, use the SET command to set the environment variable DB2DBDFT to the source server name:

```
SET DB2DBDFT=SAMPLE
```

- b. Enter the following command from the same DB2 command window:

```
ASNCMD STOP
```

You can alternatively use one of the following key combinations from the window where the Capture program is running:

- Ctrl+C
- Ctrl+Break

2. Stop the Apply program.

- a. From a DB2 command window, use the SET command to set the environment variable DB2DBDFT to the target server name:

```
SET DB2DBDFT=COPYDB
```

- b. Enter the following command from the same DB2 command window:

```
ASNASTOP DEPTQUAL
```

You can alternatively use one of the following key combinations from the window where the Apply program is running:

- Ctrl+C
- Ctrl+Break

Sample data for the target table results after one iteration are shown in Table 3. Note that the order of the target table columns has changed, but the data remains the same as for the sample source table.

Table 3. DEPARTCOPY Table Results

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

Part 2. Planning for Replication

This part describes the planning issues you must consider before setting up your replication environment. It discusses recommended replication scenarios that have been implemented by other customers and planning issues at the system and application level.

Chapter 4. Replication Usage Scenarios

This chapter describes customer environments where IBM Replication has been proven to work successfully, some rather more complex environments where the product can be used with some caution, and environments that are not supported. The scenarios covered here are not exhaustive because customers are always coming up with new and creative ways of using IBM Replication. The key is to know in advance what your chances of success are!

Recommended Usage Scenarios

Unless otherwise indicated, each of the scenarios in this section is a tried and tested implementation based on actual customer installations.³

Operational to Decision Support System Data Replication

One of the common uses of IBM Replication is to replicate data changes from an operational system to a decision support system. Generally it is not advisable to run decision support and online systems alongside each other. A process is therefore required to maintain the decision support system in line with the operational system.

IBM Replication solved this problem for one financial institution that had to replicate updates from its customer information database held on DB2 for MVS to a decision support system that was also to be on DB2 for MVS. The main requirements were that the operational system not require any code modifications and the performance of the operational system not be impacted in any way.

IBM Replication was implemented to capture updates from the key operational tables and, on an hourly basis, replicate them to consistent change data (CCD) tables in the decision support system (DSS) DB2 subsystem. CCD tables were used because history information was required in the DSS, but none was maintained in the operational system. The IBM Replication tasks were given an MVS dispatching priority such that no CPU resource was ever taken away from the operational system. If there was a shortfall in capacity during peak periods, the Capture program task simply lagged behind.

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 at some time in the future if required.

³ In many cases some of the details have been changed to conceal the real identity of the customer. Sometimes two customer scenarios have been combined.

Distributed Database Systems

A large European retail chain has almost 500 stores around the country, each of which gathers purchase details through an electronic point of sale (EPOS) system. The data is transferred nightly to a central DB2 for MVS site using a preexisting file transfer process from the EPOS terminals. The company wants to leverage the value of the data by enhancing it at the central site and then distributing it to each of the branches and to some regional offices.

IBM Replication solved the problem by providing a weekly, controlled distribution of multilevel summarized results. Apply for AIX is used to summarize the details from the MVS site before transporting them and applying them to the local DB2 for AIX databases. Each store is able to see the local trends and their performance against other outlets in the region. Regional managers are able to more effectively plan their marketing and distribution strategies by using the information to help set objectives for the individual stores.

Improved Network Load Balancing

A small bank rolled out several new OS/2-based client/server applications to its 85 branches. A major source of data for the new applications is the customer and financial reference data, which is derived and held in two operational systems, one on DB2 for MVS and the other on DB2 for AIX. It was not possible to directly access the host site for all data requirements, so a data replication solution that uses IBM Replication was devised.

Changes arising from the two operational systems throughout the day are captured and staged on the AIX platform. Several CCD tables are built, with Apply for AIX taking feeds from both DB2 for MVS and the local DB2 for AIX system. Overnight, the changes are Replicated to the OS/2 applications in the branches, ready for use the next morning.

This approach helps to minimize the network traffic by maintaining a local copy of the database in each branch. Furthermore, IBM Replication replicates *only* the changed customer and financial data and ensures that only the most recent update for each key is replicated, thus reducing network requirements.

Data Consolidation and Distribution

A large financial institution wanted to improve the flow of information from two legacy operational systems to its OS/2-based branch organization. The intent was to provide more accurate and timely data to help loan application research and to detect credit card fraud. The data to drive the loan application work was in DB2 for MVS, and the credit card details were captured in an older IMS-based system. Previous attempts to copy the legacy data had been an unworkable mixture of ad hoc reports and file transfer techniques.

Today, using IBM Replication, the branch analysts enjoy a much improved service, with daily feeds from the IMS system and twice weekly downloads of the loan history details. The DB2 for MVS-based data is captured daily but staged in a CCD table for the two weekly replication jobs. Only data that has changed is replicated, and the changes are

replicated to provide a historical perspective. The credit card information is captured from IMS with DataPropagator NonRelational. The resulting staging tables are then replicated to the branches.

Improved Application Availability

An international bank has a very sensitive online environment. The online system is used 23 hours 45 minutes a day. Every day the bank must stop the system to quiesce it for a batch application. The batch application requires exactly one day's worth of data. This data is needed for a day account. During the 15 minutes when the system is down, the bank extracts the tables required. When the bank finishes the extraction, the online system is made available for the next financial day.

To meet its goal of using the online system 24 hours a day, but not lose its application quiesce point, the bank decided to use IBM Replication. Data changes made during the online day were captured and then replicated to CCD staging tables. The batch application was modified to take input from CCD tables rather than extract files. The quiesce point can be identified from the data captured.

The source server and copy server are the same DB2 subsystem, so no distributed access is required except for the Control Center.

Data Archive

The archiving of aged or redundant DB2 data has long been a challenge for many large organizations. With IBM Replication, aged or redundant data can be safely deleted from the main tables without worrying about the archive database. The tables that require archiving are defined as replication sources. In doing this, whenever data is deleted, IBM Replication captures the deleted data and can “resurrect” the deleted row in a new target CCD table. The CCD table becomes the archive database.

Building Audit Trails

IBM Replication can be used to track changes to relational tables for auditing purposes, to determine which users made particular changes to the data.

One customer installation, for example, generated audit data by writing audit information to the IMS log, which was not a problem while all database access was through IMS. However, new applications were developed that accessed DB2 through DRDA, bypassing IMS completely. In this instance, IBM Replication was implemented to capture the updates. CCD tables were used, requesting both before and after image columns. The IBMSNAP_AUTHTKN column was replicated from the unit-of-work (UOW) control table to identify who performed the update.

Mobile Replication

Because of increasing competition from telephone-based direct insurance providers, an established insurance company wants to be more focused and aggressive in its sales campaigns. It wants to equip its sales representatives and agents, who rarely visit the company's office, with a set of offers to attract both new and existing

customers—special introductory offers, personalized packages, special “today-only” offers, and tailored cross-selling offers.

The sales force is supplied with laptop computers running OS/2 and DB2 for OS/2. As a sales campaign is launched, custom programming helps to target customers, and IBM Replication is used to download the targeted customer profiles and history, as well as the latest product offers. IBM Replication also solves the problem of keeping the information up to date with regular dial-in downloads. Of course, only new and changed data rows are copied across the network.

See “Mobile Replication Requirements” on page 58 for more planning information.

Mobile Replication with MS Jet Client

Wanting to improve its check-in and check-out process to better serve its customers, a large hotel chain has equipped each of its cleaning carts with a mobile computer. Using wireless communications, every mobile computer is connected to the front desk. The application, database, and replication software must be installed before the database administrator distributes the cleaning cart computers. The administrator can create or change the replication definitions at any time, before or after the cleaning cart computers are distributed.

Each mobile computer is running DataPropagator for MS Jet and Windows 95. Hotel guests no longer need to wait until 3:00 PM to check in. Now room availability information is reported immediately after the room is cleaned. Even if a room attendant forgets to send the updated room status, DataPropagator for MS Jet will initiate regular dial-in downloads to synchronize room availability information at the front desk. The front desk can also estimate the time that a room will be cleaned. If a guest arrives early, the front desk can locate the room attendant closest to a room and send a status change to have the room cleaned immediately.

See Chapter 17, “Mobile Replication Using IBM DB2 DataPropagator for Microsoft Jet” on page 259 for more planning information.

Potential Usage Scenarios

This section is an unusual but important one: it tells you where IBM Replication can be used, but where it is recommended that care be taken in designing the system. The scenarios described in this section are complicated and difficult to implement as first replication projects.

Update-Anywhere Replication

IBM Replication supports update-anywhere replication, in which multiple copies (or replicas) of the same source table are synchronized and each replica can be a source of updates to the source table and the other replicas. This scenario allows you to set up circular subscriptions, where Table A updates Table B and Table B updates Table A. Customers can have a global database that updates and is updated by multiple replica target tables at multiple sites, such as bank branches or a mobile sales force.

Update-anywhere replication scenarios work best when transaction conflicts between the global database and the replicas can be avoided, such as when replicas can update only key ranges at specific sites, or when sites can make updates during certain time periods.

Conflict detection is provided at three levels and can be a data integrity and performance issue that should be well planned. See “Update-Anywhere Replication” on page 72 for more planning information.

Logical Recovery

IBM Replication can capture both before and after images of any update captured and the correlation ID (if using Capture for MVS) or the authorization ID of the user that performed the update. Using this information, it is possible to write some code that can “undo” updates that have been made to a given table. This approach could prove useful in the event of a rogue batch application incorrectly updating data. You may want to undo its updates but not the updates of any other application. However, care is required as the integrity of the database could be jeopardized if the application processing is not fully understood.

Extending IBM Replication

There are occasions when the data transformation required between source and target tables is so complex that IBM Replication cannot provide an end-to-end solution. In general, if the manipulation can be achieved with SQL, then IBM Replication can perform the translation for you. The use of SQL extends to procedural logic on some database platforms. DB2 Universal Database triggers and user-defined functions and triggers can be a part of the SQL invoked to perform the transformations.

Alternatively, it is reasonably easy to develop user code to read CCD tables populated by IBM Replication, perform any transformations required, and then apply the changes to the target tables. See “Defining SQL Statements or CALL Procedures for the Replication Subscription” on page 117 for more information about using stored procedures before or after the changes are applied to the target tables.

Usage Scenarios Not Recommended for Replication

This section describes scenarios that IBM Replication was not designed to support. In these cases, you must look for other solutions. Hints are given as to which products or techniques may be helpful.

Synchronous Replication

IBM Replication does not support synchronous replication. However, if the network connection from the base application to the target is unavailable, the base system will still be available if you use IBM Replication, and data changes are not lost.

In many cases, organizations that start out with an initial requirement for immediate availability of copy data find that the benefits of staged delivery (such as, better network use, less database contention, and the opportunity to enhance the data) are more

attractive, so they reexamine the original requirements. IBM Replication supports continuous timing of subscriptions.

If synchronous data delivery is essential to the application, the recommended approach is to use DRDA two-phase commit within the application.

Hot-Site Recovery

Many planners are tempted to consider IBM Replication for hot site recovery because of its ability to replicate tables to other sites. However, IBM Replication was not designed for such recovery because the underlying architecture is asynchronous. There is no guarantee of how much data has been captured or applied at any one instant.

Hot site recovery is a complex and specialized task and is supported by several IBM and vendor features and tools. For example, in the System/390 area, the peer-to-peer remote copy (PPRC) hardware feature and the extended recovery component (XRC) are available.

Chapter 5. System Planning

This chapter helps to establish the feasibility of the replication implementation by considering the system requirements: which replication products to use, software and hardware requirements, storage requirements, CPU capacity, network arrangement, security authorizations, and user IDs.

Even if your requirements are not fully defined at this stage, this chapter will start you thinking about the kinds of questions that you should be asking.

Replication Products

IBM Replication products are packaged either in product solutions or are available separately. Table 4 describes how the administration tools and Capture and Apply programs are packaged.

Table 4 (Page 1 of 2). The Replication products packaging solution

Product Name	Platforms	Replication Features	Includes host and OS/400 connectivity?	Includes DB2?	Includes Client Application Enabler (CAE)?
DB2 Enterprise Edition	OS/2, Windows NT, AIX, HP/UX, Solaris, SCO UnixWare 7	Capture program, Apply program, and DPROPR V1 Migration program on OS/2, Windows NT, Windows 95	Yes	Yes	Yes
DB2 Workgroup Edition	OS/2, Windows NT, AIX, HP/UX, Solaris, SCO UnixWare 7	Capture program, Apply program, and DPROPR V1 Migration program on OS/2, Windows NT, Windows 95	No	Yes	Yes
DB2 Personal Edition(1)	OS/2, Windows NT, Windows 95	Capture program, Apply program, and DPROPR V1 Migration program on OS/2, Windows NT, Windows 95	No	Yes: single user	No
DB2 Connect Personal Edition(2)	OS/2, Windows NT, Windows 95, Windows 3.1	DPROPR V1 Migration program on OS/2, Windows NT, Windows 95	Yes, single-user	No	No

Table 4 (Page 2 of 2). The Replication products packaging solution

Product Name	Platforms	Replication Features	Includes host and OS/400 connectivity?	Includes DB2?	Includes Client Application Enabler (CAE)?
DB2 Connect Enterprise Edition(1)	OS/2, Windows NT, AIX, HP/UX, Solaris, SCO UnixWare 7	DPROPR V1 Migration program on OS/2, Windows NT, Windows 95	Yes	No	Yes
DPROPR Capture for MVS, V5.1	MVS	Capture program	Yes, packaged with DB2 Connect Personal Edition	No	No
DPROPR Apply for MVS, V5.1	MVS	Apply program	Yes, packaged with DB2 Connect Personal Edition	No	No
DB2 Server for VSE, 5.1	VSE	Capture Program	Yes, packaged with DB2 Connect Personal Edition	Yes	No
DB2 Server for VM, 5.1	VM	Capture program	Yes, packaged with DB2 Connect Personal Edition	Yes	No

(1) Both products are required at a minimum for the mobile environment.

(2) Included in the Capture and Apply program packages for MVS, and Capture program packages for VSE and VM.

Hardware and Software Requirements

The following section describes the hardware and software requirements for the replication tools.

The Capture and Apply Programs on the DB2 Universal Database

See the “Server Product Requirements” under “Hardware Requirements” and “Software Requirements” in *DB2 Universal Database Quick Beginnings* for your platform.

The Capture and Apply Programs for DB2 for MVS, DB2 for VSE, or DB2 for VM

The following sections describe the hardware and software requirements for the Capture and Apply Programs for DB2 for MVS, DB2 for VSE, or DB2 for VM.

Hardware Requirements

Table 5 on page 41 lists the hardware requirements for the Capture and Apply Programs for DB2 for MVS, DB2 for VSE, or DB2 for VM.

Table 5. Summary of Hardware Requirements for the Capture and Apply Programs for DB2 for MVS, DB2 for VSE, or DB2 for VM

Product	Requirements
Capture for MVS, 5.1	Any hardware that supports MVS/ESA Version 5 Release 1 (5655–068) or higher (for data sharing), or MVS/ESA Version 3 Release 1 Modification 3 (5695–047) or higher (without data sharing) System/390 Parallel Sysplex is required for data sharing
Apply for MVS, 5.1	Any hardware that supports MVS/ESA Version 5 Release 1 (5655–068) or higher (for data sharing), or MVS/ESA Version 3 Release 1 Modification 3 (5695–047) or higher (without data sharing) System/390 Parallel Sysplex is required for data sharing
Capture for VSE, 5.1	Any hardware that supports VSE/ESA Version 2 Release 1 Modification 2 or higher
Capture for VM, 5.1	Any hardware that supports VM/ESA Version 2 Release 1 or higher

Software Requirements

Table 6 lists the software requirements for the Capture and Apply Programs for DB2 for MVS, DB2 for VSE, or DB2 for VM.

Table 6 (Page 1 of 2). Summary of Software Requirements for the Capture and Apply Programs for DB2 for MVS, DB2 for VSE, or DB2 for VM

Product	Requirements
Capture for MVS, 5.1	<ul style="list-style-type: none"> • IBM MVS/ESA Version 3 Release 1 Modification 3 (5695-047) or higher (data sharing requires MVS/ESA Version 5 Release 1 (5655–068) or higher) • DFSMS Version 1 Release 1 (5695-DF1) or MVS/DFP Version 3 Release 3 (5665-XA3) or higher • IBM C/370 Library Version 2 or IBM SAA AD/Cycle Language Environment/370 Version 1 Release 3 or higher, including any PTFs • IBM DATABASE 2 Server for OS/390, Version 5, Release 1 (5648-158), or IBM DATABASE 2 for MVS Version 3 Release 1 or higher, including any PTFs • IBM System Modification Program/Extended Version 1 Release 8 (5668-949)
Apply for MVS, 5.1	<ul style="list-style-type: none"> • IBM MVS/ESA Version 3 Release 1 Modification 3 (5695-047) or higher (data sharing requires MVS/ESA Version 5 Release 1 (5655–068) or higher) • DFSMS Version 1 Release 1 (5695-DF1) or MVS/DFP Version 3 Release 3 (5665-XA3) or higher • IBM C/370 Library Version 2 or IBM SAA AD/Cycle Language Environment/370 Version 1 Release 3 or higher, including any PTFs • IBM DATABASE 2 Server for OS/390, Version 5, Release 1 (5648-158), or IBM DATABASE 2 for MVS Version 3 Release 1 or higher, including any PTFs • IBM System Modification Program/Extended Version 1 Release 8 (5668-949)
Capture for VSE, 5.1	<ul style="list-style-type: none"> • IBM DATABASE 2 Server for VSE & VM Version 5, Release 1 (5648-158) • IBM Language Environment for VSE Version 1 Release 4 (5685-094) or the equivalent C & CEL run-time support provided in the base of VSE/ESA Version 2 Release 2.

Table 6 (Page 2 of 2). Summary of Software Requirements for the Capture and Apply Programs for DB2 for MVS, DB2 for VSE, or DB2 for VM

Product	Requirements
Capture for VM, 5.1	<ul style="list-style-type: none"> • IBM DATABASE 2 Server for VSE & VM Version 5, Release 1 (5648-158) • IBM Language Environment Version 1 Release 5 (5688-198) or higher, or the equivalent C & CEL run-time support provided in the base of VM/ESA Version 2

Storage Requirements

The additional storage requirements associated with the introduction of replication into your system will arise in three areas:

1. **Database management system log and journal data** — the additional data logged to support the capturing and applying of data.
2. **New tables** — all of the copied user data and any IBM Replication control tables, including staging tables.
3. **The Apply spill file** — the Apply program requires temporary space to store refresh answer sets. (With Apply for MVS, the spill file function can use memory rather than disk.)

DBMS Logging

If update copying is required, the database management system (DBMS) must log full row images for any update made to a table that has been selected for propagation. Additional log data is also generated when data is inserted or deleted from the unit-of-work (UOW) and change data (CD) tables in DB2.

It is not a straightforward calculation to estimate the increase that will be experienced in the log or journal volume. As a rule of thumb, it is recommended that the estimated increase be sized at three times the current log volume for the tables subject to replication.

A more detailed estimate requires detailed knowledge of the updating application and the replication requirements. If an updating application typically updates 60% of the columns in a table, the DATA CAPTURE CHANGES requirement will cause these log records to grow by more than half. Log records that would typically hold more or less of the row data would grow less or more, respectively. Several customers report a 60%-80% increase in the size of the update log records for the application tables that will be captured.

If only a small number of the columns are defined in the replication source, the inserts and deletes that are made to the CD table will cause small log records to be generated for these updates, whereas a replication source with all before and after images of the columns of a table will generate a larger log volume, as in the case of update-anywhere scenarios.

There will also be logging on the target database, where the rows are applied. Because the Apply program does not issue interim checkpoints, an estimate should be made for the maximum answer set of changed rows that will be handled by the Apply program, and the space for the logs should be allocated accordingly.

New Tables

The estimates for this storage are also based on the source updating applications and the replication requirements. The tables that account for the most storage use in replication are the UOW table, the CD tables, and the target tables. Space requirements can be estimated for the target tables based on the source tables.

Space requirements for the CD tables can be estimated by first determining the “holding” period of the data. For example, if the Apply program is run once daily, with a “pruning” deletion of the applied rows from the CD table occurring once a day just after the Apply program, the base “holding” period would be 24 hours worth of updates. If the CD rows are 100 bytes long, and 100,000 updates are captured on an average day, the storage required for the CD table is about 10 MB. This is a sufficient approximation for initial storage requirement estimates. The CD table rows include 21 bytes of overhead data, in addition to the set of columns defined in the replication source.

The UOW table size is harder to estimate. The number of rows inserted in a day would equal the number of commits issued on transactions that update tables subject to replication. You should initially overestimate the size required and monitor the space actually used to see whether any space can be recovered. UOW table rows are fixed at 79 bytes long.

The space requirements for the IBM Replication control tables other than the UOW and CD tables are generally quite small. Most control tables require only a few rows, and default sizings are usually adequate.

The target table can be estimated by starting with the size of the source table, and then adding or subtracting changes to the target table from data transformation (adding columns), aggregating columns, or subsetting columns or rows.

The Apply Spill File

The spill file used by the Apply program is equal to the size of the data selected for replication at each interval, including full refresh.⁴ The spill file uses disk storage on every platform, or it can use virtual memory with Apply for MVS. The size of the answer set to be selected by the Apply program can be estimated by looking at the frequency interval planned for the Apply program, as compared to the volume of changes in that same time period, or in the peak period of change. For example, if change volume peaks at 12,000 updates per hour, and the Apply program frequency is planned at one-hour intervals, the one-hour interval will determine the average size of the spill file. In this case, the spill file must hold one-hour's worth of updates, or, 12,000 updates. The row size is the *target row* size, including any IBM Replication overhead columns. This

⁴ If you are using ASNLOAD, instead of a load spill file there will be a load input file.

row size will not be in DB2 packed internal format, but in expanded, interpreted character format, as fetched from the select. The row also includes a row length and null terminators on the individual column strings. Multiple spill files are used for replication subscriptions with multiple target tables; one spill file for each target table.

Data Blocking for Large Volumes of Changes

You may have a replication subscription with a large block of changes from one or more CD or CCD tables to be replicated in one Apply cycle. These large blocks can cause a backlog of queued transactions. They can also cause the spill file or log to overflow. Batch-Apply scenarios, for example, can produce a large backlog of queued transactions that need to be propagated (as is typical when the subscription runs only once a day or less). If a large block of change data accumulated in the CD tables due to an extensive Apply program outage, for example, breaking down the block of data prevents spill file overflows.

You can use the Data Blocking page of the Subscription Timing notebook to specify how many minutes worth of change data can move in a subscription cycle. The number of minutes you specify is used to determine 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 automatically retries three times, adapting to the work load level to copy in blocks that won't cause spill file or log overflow. To adapt the work load, the Apply program cuts the value in half to get the answer set size small enough to replicate successfully. If there is still a problem, it uses one-fourth of the value to align the changed data backlog with the available system resources. This reduces the stress on the network and DBMS resources and reduces the risk of failure. If a failure occurs during one of the mini-cycles, then only the failed mini-cycle needs to be redone. When the Apply program later retries the replication subscription, it will resume from the point following the last successful mini-cycle. Figure 6 on page 45 shows how the changed data is broken down into subsets of changes.

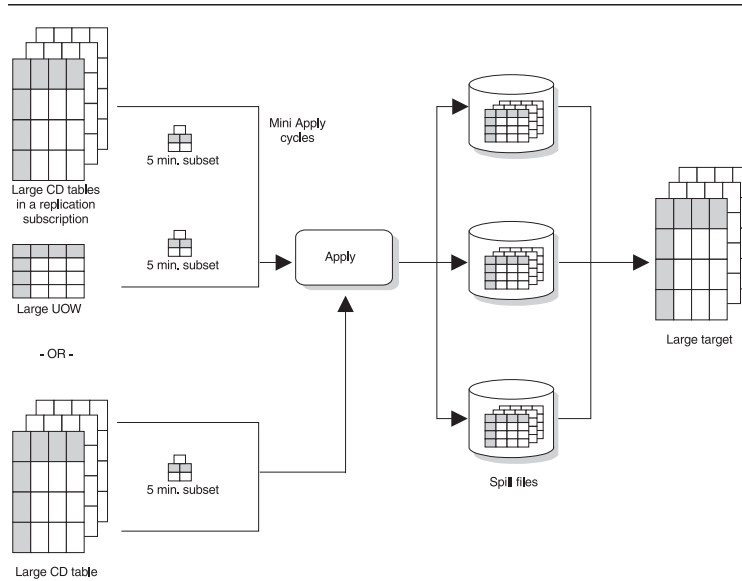


Figure 6. Data Blocking Value. You can reduce the amount of data being replicated at a time by specifying a blocking value.

The data blocking minute default is NULL. This NULL default value will copy all committed data available, no matter how big or small. The number of minutes that you set should be small enough so that all the transactions for the replication subscription that occurred during the interval can be copied without causing the spill file to overflow or a log full condition. The space required for the spill files is the sum of the space required for each answer set in the replication subscription. The column MAX_SYNCH_MINUTES in the subscription set control table contains the value that you specify.

Restrictions:

- You can't split a unit of work.
- You can't roll back previous mini-subscription cycles.
- You use data blocking in update mode only, not full refresh.

The Active Log File Size for Capture for MVS on DB2 for MVS 3.1 and Capture for VSE and VM 5.1

The Capture program captures updates from the log file. When the log is full, its contents are archived on an archive log file. If the 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 can be archived before they are captured. Archived log records cannot be recovered by Capture for MVS, VSE, and VM running with DB2 for MVS 3.1 or DB2 for VSE & VM 5.1. However, archived log records can be recovered by Capture for MVS running with DB2 for MVS 4.1 or higher, or the DB2 Universal Database V5.

If you are running on DB2 for MVS 3.1 or DB2 for VSE & VM 5.1, the size of your log should be large enough to handle at least 24 hours of transaction data.

General Storage Considerations

All of these sizes are estimates only. To prepare and design a production-ready system, factors such as failure prevention must also be taken into account. For example, the holding period of data might need to be increased to account for potential line outage.

If storage estimates seem to be unreasonably high, the frequency interval of the Apply program and pruning should be reexamined. Trade-offs frequently must be considered between storage utilization, capacity for failure tolerance, and CPU overhead consumed.

Capacity Requirements Analysis

IBM Replication requires processor resources in support of the Capture and Apply programs and the Control Center for replication administration.

The Capture Program

The CPU requirements for the Capture program are generally low. In a well-tuned environment, customers report that the Capture program does not impact the updating applications and requires a minimum of CPU capacity. Capture for MVS can be scheduled at a lower priority than the source applications. The Capture program will simply lag behind during periods where CPU resource is constrained.

The Capture program needs CPU resource during pruning, and this action can be deferred to times when system impact is low.

The Apply Program

The Apply program CPU requirements can vary greatly. The main factor that affects CPU is the currency requirement of the target system. The more frequent the propagation, the higher the overhead per propagated row and the greater the CPU consumption.

The Apply program requires CPU on both the source and the target system, where they are different. For example, Apply for OS/2 could be a Distributed Database Facility (DDF) thread performing mainly SQL select, but some SQL update activity, on a DB2 for MVS system. Estimating the cost of remote threads in a planned large distribution scenario might lead to a design involving an interim staging platform, as described in “Staging Changed Data” on page 59.

Administration

Replication administration can be planned for a time when impact to the source and target database systems will be minimal.

In general, the impact of the Control Center is not significant in terms of the local CPU. However, during times of intensive replication source and subscription definitions, the catalogs of the data server site are extensively searched. For large MVS sites, this can have a noticeable CPU or database system impact.

See the *Hardware Requirements* section in the *DB2 Universal Database Quick Beginnings* book for your platform to determine memory and disk requirements.

Network Requirements

Anticipating the basic networking requirements for replication is easy—connect all of the sources to the targets, and connect the control point to all sources and targets for which it will perform administrative tasks.

Deciding among the various possible connectivity scenarios, estimating how much capacity will be required, and determining what level of data currency will be possible given the current available bandwidth can be very difficult chores, however. This section describes connectivity possibilities, bandwidth impact analysis, pull versus push Apply design, and throughput capacity factors that you must consider.

There can be definitive trade-offs between storage required, CPU consumed, network bandwidth consumed, and achieved replication throughput. In the planning stages it is a good idea to consider these aspects and understand both the available threshold capacity as well the relative priorities of each aspect.

Connectivity Possibilities

The Capture program is always self-contained to a database, subsystem, or data sharing group, and must be able to connect to the source server database. The Control Center workstation must be able to connect to source and target server databases to perform its tasks, and the Apply program component must communicate with both the source and target server databases, when these are different.

Where communications are used, the connectivity is always through DRDA or the DB2 Universal Database equivalent. The actual communications software that can be used to support the DRDA connectivity varies according to the platforms being connected. Between DB2 Universal Database databases, the choices are TCP/IP, SNA, NetBIOS, and IPX/SPX. DB2 Personal Connect Edition (PCE) is required for connections between DB2 Universal Database databases and DB2 for MVS, DB2 for VSE, or DB2 for VM. TCP/IP or SNA can be used with DB2 for MVS 5.1 and PCE 5.1. All other connections use SNA only.

The more layers of emulation used, LAN bridges added, or router linkups required, the more restricted the replication performance will be. Planning for both current and future needs is essential.

Communications resources can be a major factor in a replication design that involves staging data at a server different from the source database. For example, in a mobile replication scenario between DB2 for OS/390 Version 5 Release 1 and DB2 for OS/2, the best connectivity scenario might be to run TCP/IP over a modem link between the

remote OS/2 and an AIX staging platform running Apply for AIX. The AIX database would be connected to DB2 for OS/390 through PCE and TCP/IP.

Bandwidth Impact Analysis

IBM Replication is designed to allow for low impact to the network. For example, it allows for the replication of changed data only, supports a data staging arrangement, provides for summarization at the source, can be scheduled to run at off-peak times, and uses DRDA, which enables high-speed, secure data delivery. However, replicating data is not free, and one of the key costs is in bandwidth. So, what is IBM Replication going to do to your network?

The Control Center requires a small amount of capacity. However, the Control Center impact is limited to set up and maintenance of the replication objects.

The Apply program task requires network capacity if the target server and the source server are not the same database or subsystem. In general, the capacity required depends on the volume of data to be applied, the timing window available in which to apply the data, the desired currency of the target data, and the bandwidth installed or to be installed. For example, if a batch program generates many megabytes of change data and the data must be applied to the target system within 30 minutes, the bandwidth requirements will be higher than if the target can be up to 24 hours out of date. The Apply program could then be scheduled to use surplus capacity during periods when network traffic is lighter. For more efficient use of the network capacity, the Apply program is usually installed at the target server so that it can pull the data from the source server.

Remember that the Apply program is an SQL application and is therefore subject to all of the influences with which any SQL application must handle. Given these factors, the best indicators of likely performance are often found outside the IBM Replication area, in general distributed relational database studies.⁵

Pull versus Push Apply Design

The pull versus push configuration is a question of where the Apply program is running: at the source server or the target server. In the push method, the Apply program runs at the source server. In the pull method, the Apply program runs at the target server. The level of granularity is at the replication subscription level; one Apply program could be pushing for some replication subscriptions and pulling for others.

When the Apply program processes a replication subscription, it first connects to the source server to fetch the current changed data. This data is fetched into a spill file that is local to the Apply program. After the data is retrieved, the Apply program connects to

⁵ See the following sources for detailed performance measurements:

- *DDCS/2 to DB2 Performance Benchmarks*
- *Getting Started with DB2 Stored Procedures*
- Web site: <http://www.csc.ibm.com/advisor/library/>

the target server and applies the changes, one row at a time, as an INSERT, UPDATE, or DELETE to each target table.

Figure 7 shows the difference between push and pull modes.

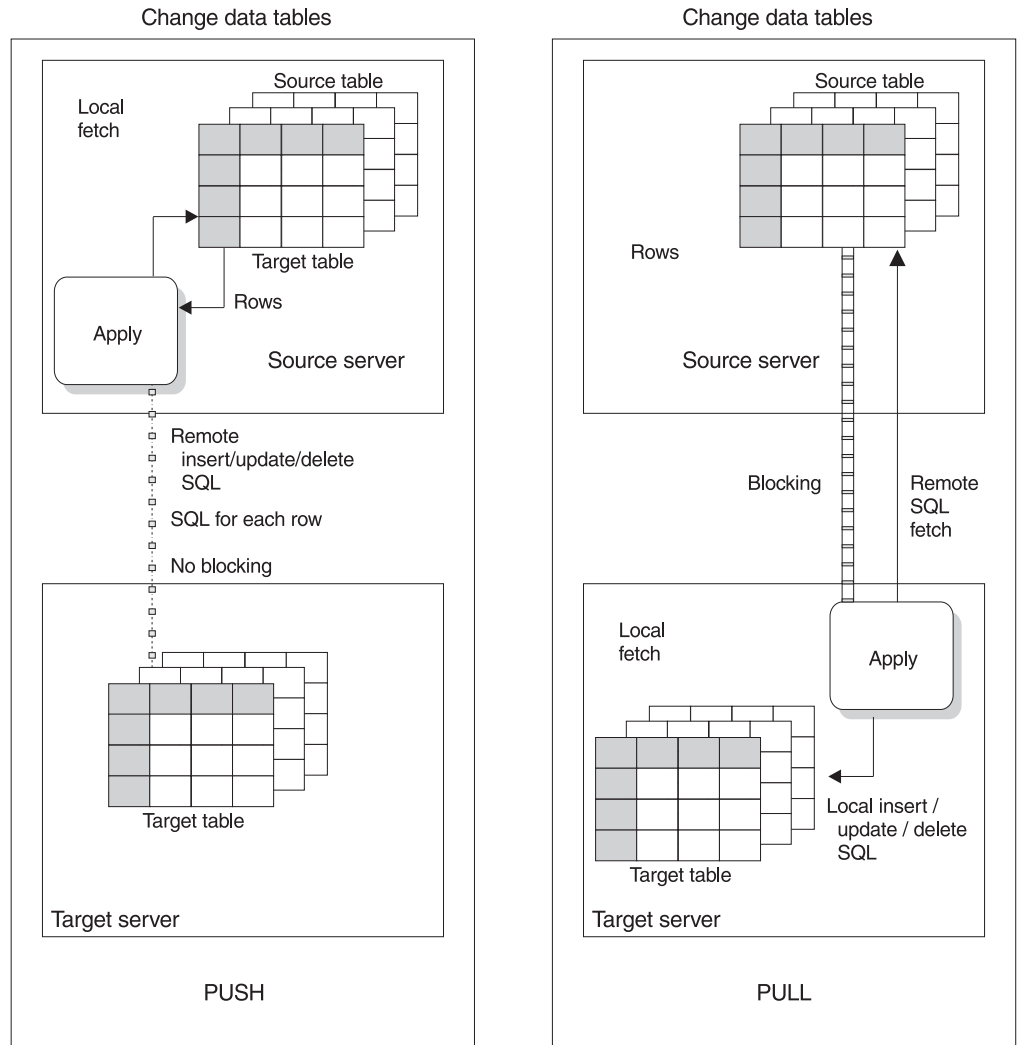


Figure 7. Push Versus Pull Mode

In pull mode, the Apply program connects to the remote source server to retrieve the data. DB2 can then use block fetch to pass the data across the network efficiently. When all data has been retrieved, the Apply program connects locally to the target server and applies the changes to the target table. The row-by-row process occurs as a local operation.

In push mode, the Apply program connects to the local source server and retrieves the data. Then it connects to the remote target server and pushes the updates to the target table. The row-by-row process occurs as a remote operation, with no blocking for network efficiency.

You do not have to do any special configuration to set up a push or pull configuration, except decide where to run the Apply program. The replication components and the Control Center recognize both configurations. The Control Center automatically sets up the replication control tables so the Apply program can push or pull data.

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.
- The source table changes very infrequently, but when it changes it should be replicated as soon as possible.

Throughput Capacity

Many individual factors influence the throughput possible with IBM Replication. The most important factors include:

- Network bandwidth and tuning
- Source and target machine CPU capacity available
- Database tuning and contention factors
- Change volume and work load mix
- Frequency of the Apply program and number of subscriptions
- The number of Apply program instances used
- Apply program pull versus push approach

Given the complex set of variables involved, you cannot accurately predict the throughput that might be achievable in a given system. At the same time, a feasibility study would normally need to include some estimation of the potential throughput that is possible.

One way of looking at throughput estimation is to break it down into two parts (assumes a remote pull configuration):

- Remote block fetch of x number of rows — stored into a file (or in MVS, into memory, which of course is the fastest)
- Insert/update/delete of x number of rows in the copy server database

A feasibility study normally includes some estimation of the potential throughput that is possible, and developing a prototype is recommended to verify the throughput in an environment that reflects production conditions.

Security and Authorization Requirements

Security for IBM Replication is a matter of database security. The entire system is table driven, and therefore security of all IBM Replication objects involves database security mechanisms.

Each database has an administrator, who requires sufficient privileges to define replication sources and targets. Additionally, the Apply program uses a qualifier that must be coordinated, but the same user ID can be used to run multiple Apply program instances.

Authorization Requirements for Administration

During the initial defining of replication sources and subscriptions, many tables are created. Depending on the platform, table spaces or dbspaces might also be created. All of these actions require a fairly high level of database privilege, and therefore you should plan on having at least one user ID that acts as the replication administrator and has the authority to create objects and bind plans on each of the replication databases.

The administrator user ID must be a valid logon ID at both the workstation where the Control Center is installed and the source and target sites. The administrator user ID can be used as the user ID running the Capture program or the Apply program, but this is not a requirement.

Authorization Requirements for the Capture Program

The user ID that runs the Capture program must be able to access the system catalog tables, be able to access and update all IBM Replication control tables that are built at the source database, subsystem, or data sharing group, and have execute privileges on the Capture program plan.

For more information about authorization requirements for the Capture program, see the Capture and Apply chapter for your platform in this book.

Authorization Requirements for the Apply Program

The Apply program user ID must be a valid logon ID on the source, control, and target servers, and the workstation where the Control Center is installed. The user ID that runs the Apply program must be able to access the replication source tables; access and update all IBM Replication control tables that are built at the source and target database, subsystem, or data sharing group; and update the replication target tables. This user ID must also have execute privileges on the Apply program plan. With the proper authorization, any user ID can run any Apply program instance.

An Apply program running on DB2 Universal Database might require a password file to connect to the source or target server. For an explanation of configuring security when the Apply program is running on DB2 Universal Database, and more information about authorization requirements for the Apply program, see the Capture and Apply chapter for your platform in this book.

Customization Requirements

The Control Center creates and populates the replication control tables through SQL files. You can customize these files outside of the Control Center before using the Control Center for replication administration. Most customers need to customize the replication control tables for their site.

The Control Center also creates and populates the replication target tables using SQL generated by replication requests, such as defining a replication source. You can store the SQL in a file and then edit and run it later. Most customers need to customize the control information for replication sources and subscriptions.

See “Working with Customized Replication Control Tables” on page 88 and “Customizing and Running Replication SQL Files” on page 90 to learn how to customize SQL files for your site.

Chapter 6. Application Planning

This chapter helps you evaluate your application level requirements against product capabilities. The term *application level requirements* refers to the requirements of the target application, specifically the type of data required in the target tables (including any subsetting, transformations, or enhancements to the source data), the currency of the data copies, and the consistency of the data copies.

This chapter also helps you further refine your requirements through an evaluation of your source-to-target mapping needs. After reading this chapter, you will be able to plan the time required to implement your replication solution. You will understand which requirements are met by basic techniques, and which require more advanced techniques.

Data Transformation Requirements

Source data most likely has to be subsetted, transformed, or enhanced as part of the replication process, particularly in the support of decision support or data warehousing. This section sorts these requirements into those that are easily fulfilled by using the Control Center and those that require direct manipulation of the control tables.

Basic Data Enhancement

This section explains the following basic capabilities that the Control Center supports.

- Choices of target table types
- Subsetting targets
- Source views
- Before and after images
- Column renaming
- Computed columns
- Before and after run-time processing

Choices of Target Table Types

You can use the following types of target tables:

User copy target tables

These are the most common target tables, and are copies of the replication source without an overhead timestamp column. These tables require a primary key.

Point-in-time target tables

These are plain copies of source tables with timestamp columns added to specify when updates were made. These tables require a primary key.

Aggregate tables

These tables are used to track data and changes to data. These tables are built when rows are added or appended to the table over time.

Queries with column functions (AVG, MAX, MIN, SUM, COUNT) can range over many thousands of qualifying rows, yet they can return very compact, easy-to-

understand results. By adding a timestamp and aggregating the results of these calculations, you can track broad trends, while still retaining the benefits of data reduction and compact storage.

There are two kinds of aggregate tables:

Base aggregate tables

In these tables, calculations are made against the source table. This type of history is useful for tracking broad indicators that have relatively low volatility. Base aggregate tables can include calculations against stable base data; calculations are made according to the subscription frequency criteria. Use base aggregate tables if you need to summarize your source table contents on a regular basis. For example, you can use a base aggregate table to track a daily inventory of items from a source table.

Change aggregate tables

In these tables, calculations are made against change data, not base data. Each calculation ranges over the recent changes since the time of the last calculation. You can track insert, update, and delete operations collectively or individually, determined by the filtering predicates that you specify. If there are no recent updates, calculation is deferred, creating very compact storage. Use these tables if you need to summarize the results of the changes made between each Apply program refresh operation. You can use change aggregate tables on their own, where daily summaries of change events are required, or as input to an application that must apply the summaries to a cumulative table.

Noncondensed CCD target tables

These tables are used for history tables or audit trail tables. They can contain changes only, or they can include the initial state of the source table.

Condensed CCD target tables

These target tables can be used as an alternative to a point-in-time copy when it is important to keep the last state of any record, (that is, no records are to be deleted). A delete is processed as an update, with the IBMSNAP_OPERATION column acting as the logical delete flag.

Updateable replica target tables

These target tables can be used as sources for copying data back to the replication source table and are used in update-anywhere scenarios

Subsetting Targets

IBM Replication supports both vertical and horizontal subsetting of the source table. This means that you can specify that only a subset of the replication source table columns and rows are 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 into the CD table. With vertical subsetting, you can define the CD table to have fewer columns than your base table. However, CD tables must contain sufficient key data for point-in-time copies, which are maintained with key-qualified predicates. Omitting columns

from the target is appropriate if the column data types are not supported. Vertical subsetting is available for all tables except replica tables.

You can define vertical subsetting either:

- When you define a replication source table for differential refresh: Deselect the columns that you do not want to make *available* for replication to a target table.
- When you define a replication subscription: Use the advanced subscription options to deselect source table columns that you do not want to replicate to the target.

Row subsetting

You can split a single source table into various target tables based on the source column contents (for example, department number or region code) or you can specify that only certain source rows be replicated. Use the advanced subscription options to define a WHERE clause when you define the subscription. All target table types support row subsetting. You might need to define a view replication source to bring together all the columns needed to define the row subset.

Target table primary keys are assumed to be invariable. If one or more of the columns defined in the predicate are updated, you must specify replication logical partitioning key support when you define the replication source. Replication logical partitioning key support defines updates as DELETE and INSERT statements.

Source Views

Using the Control Center, you can define a join as a replication source. The joins can include only tables defined as replication sources. If the replication sources defined in the join have CD (or CCD) tables, a CD view is created from the replication sources' CD tables. The Capture program (or the Apply program, for CCD sources) maintains the control information for the joined replication sources and the CD views in the source server control tables.

Join views fill many requirements, both for denormalizing (restructuring) copies in data warehouse scenarios, enabling easier querying of copied data, and also for addressing the routing problem, sometimes called the database partitioning problem in distributed computing scenarios. For example, knowing where to send a bank account update may require a join of the account table with the customer table, in order to know 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 production database. Views are also useful when you need to specify predicates that exceed the capacity of the ASN.IBMSNAP_SUBS_MEMBR.PREDICATES control table column. You can choose to manage your subsetting predicates through views rather than in the subscription meta data.

IBM Replication supports the following types of view definitions:

- Simple inner-joins over one or more defined replication sources
- Simple inner-joins over CCD staging tables that are defined as replication sources and maintained by an Apply program or an application other than an IBM Repli-

cation component and an external data source, such as DataPropagator NonRelational and IMS source data

Before and After Images

Both before and after images can be defined in replication sources and subscriptions. A before-image column is a copy of a column before it is updated. DB2 logs both the before-image and after-image columns of the table for each change to the table. Before images do not make sense with a base aggregate target table type. All other target table types can make use of before-image columns.

Column Renaming

Source site column names can be renamed for point-in-time and user-copy target table types.

Computed Columns

New columns can be derived from the existing source columns on the basis of SQL expressions. New columns can be aggregate functions such as COUNT or SUM in the case of aggregate target table types, or simple derivations in the case of all other target table types.

User-defined functions for DB2 Universal Database can be specified by using the computed column facility in the Control Center.

Before and After Run-Time Processing

You can define run-time processing statements using SQL statements and stored procedures before or after the Apply program processes the replication subscription. This feature is useful for pruning CCD tables and controlling the sequence in which replication subscriptions are processed. The run-time processing statements can be run at the source server before the replication subscription is processed, and at both the source and target servers after the replication subscription is processed. For example, you can transform source data before replicating the data or target data after it is copied.

The stored procedures use the SQL CALL statement, newly supported by IBM Replication, without parameters. The run-time procedures are executed together in a single unit-of-work. You can also define acceptable SQLSTATEs for each processing statement.

Depending on the DB2 platform, logic can be invoked through the SQL before and after processing:

- Calls to stored procedures can be made through a before or after processing statement, or triggers can be used to do more complex transformations.
- On DB2 Universal Database, before or after SQL statements can invoke user-defined functions through the use of triggers.

Advanced Data Enhancement

The transformations listed below require manipulation of the IBM Replication control tables outside of the Control Center. The techniques required to implement these transformations are not discussed in this book. For planning purposes, when you find that you will use many of these techniques in the replication scenario, you should allow additional implementation time.

- **Data consolidation** - Union scenarios can be used to consolidate identically structured data into a common consolidated target table.
- **Outer join of source tables** - Join scenarios can be used when one or more of the sources are subject to change. An outer join can be implemented with multiple subscriptions to a single table.
- **Target views** - You use the target views only when consolidating data from multiple sources, such as multisite union scenarios. Multiple source tables are updated and consolidated into a join at the target server. These types of views are not supported in the Control Center.

When the target table is maintained by different servers or different replication subscriptions, a subset view ensures that the DELETE statement generated before a full-refresh is copied to the appropriate horizontal fragment. The Apply program is almost always driven by SQL operations at the source server and in addition, generates an unqualified DELETE statement at the target server before applying a full-refresh answer set. By defining a subset view over the target table, and defining the replication subscription target to be the view, you restrict the DELETE statement to the appropriate horizontal fragment. That way, a full refresh of data from the source does not affect the information from the other sources.

- **Poorly structured data** - Many older applications often have poorly structured data; for example, conflicting sources for common data items, no primary keys defined, poorly normalized tables, and heavily encoded data values.

Although IBM Replication might be able to handle these individually, there are some extreme situations where it is not useful to attempt replication before some manual data cleansing is done.

General rule: If the data transformation can be expressed by using SQL, IBM Replication will almost certainly be able to execute it. The SQL is introduced either as part of the SELECT statement that does the copying or as part of the user-specified SQL before and after statements. This SQL might include procedural logic by including database triggers, user-defined functions, and stored procedure calls as described in “Basic Data Enhancement” on page 53.

IBM Replication can also be supplemented with procedural code in user-written programs. For example, a program could be scheduled to manipulate data in the CD or CCD tables before execution of the Apply program. Or, if the changes are so radical that the data cannot be changed in place, you could go as far as writing a program to read the data changes from the CD or CCD tables and then applying them to the target tables yourself. (See “Defining SQL Statements or CALL Procedures for the Replication Subscription” on page 117 to learn about running SQL and Call procedures with replication subscriptions.)

Auditing Requirements

Auditing requirements relate to the need to track histories in terms of before and after comparisons and identification of changes by time and updating user ID.

IBM Replication supports auditing in the following ways:

- **Before and after images** - When you define replication sources, you declare whether or not before or after image columns of the updated rows are available to subscribers. 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 building a history.
- **Transaction identification** - Each captured change row is automatically stamped with control columns, which are available for audit usage. Two of these columns are the approximate commit time of the changed row at the source server and the operation code (I=Insert, U=Update and D=Delete).

If more user-oriented identification is required, columns for the DB2 correlation ID (if DB2 for MVS is the source database) and the primary authorization ID can be requested in the copy table.

Mobile Replication Requirements

Mobile replication requirements center around having copies of centrally located tables on mobile clients. Special considerations arise with this approach, primarily from the need for the target site to initiate and control the replication operation. IBM Replication supports mobile replication requirements in the following ways:

- **On-demand execution of the Capture and Apply programs** - In an unpredictable mobile environment, the mobile client must initiate and control the connections. To reduce communication costs, the duration of phone connections must be minimized and the mobile client is not continuously in operation. The Capture and Apply programs must operate only for the duration within which all the accumulated transactions are captured and copied to or from the centrally located tables. When the replication sources and subscriptions are defined on the mobile client, the Capture and Apply programs are independent of all other IBM Replication components. They can be run in an on-demand mode whenever required to dynamically connect to the target site.
- **Push and pull options** - If you must upload new or changed data from the mobile clients to the replication source site, the Apply program on the mobile client automatically pushes the data to the replication source site.
- **Changed data retention** - Because of unpredictable connection arrangements, you must ensure that all changed data is retained until each of the mobile clients has replicated its data. The best way to configure IBM Replication for optimized data retention is to use a CCD table, and possibly locate it on a staging server. Two Apply programs are needed. One Apply program moves the data from the CD

and UOW tables to a CCD table so that the Capture program can prune the changed data on a regular basis. Another Apply program, on an intermediate server, moves the data from the CCD table to the mobile client. This configuration is not available for update-anywhere replication.

Staging Changed Data

One of the advantages of IBM Replication is that it allows you to *stage* changed data; that is, the Capture program captures changes to a source table only once and inserts change data rows into a CD table. The Apply program then pulls the changes from the CD tables. The Capture program also automatically prunes change data rows from change data (CD) tables when they are no longer needed if PRUNE is enabled; it does not, however, prune change data from consistent change data (CCD) tables.

CD and CCD Tables

A CD table receives an arbitrary number of change data rows from the Capture program that are not condensed. The CD table has no knowledge of transaction boundaries, or 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 the committed changes to apply to copies. Uncommitted changes are eventually pruned, depending on the retention limit that you define in the Capture program tuning parameters control table.

Rows in a CD table reflect changes that are equivalent, if not identical, to the original operational updates. Uncommitted and incomplete changes can appear in rows in a CD table.

CCD staging tables are copies, defined in much the same way as point-in-time copies. They are the join of the CD and UOW tables and contain only committed change data. Figure 8 on page 60 shows which columns of the UOW table are included in the CCD table.

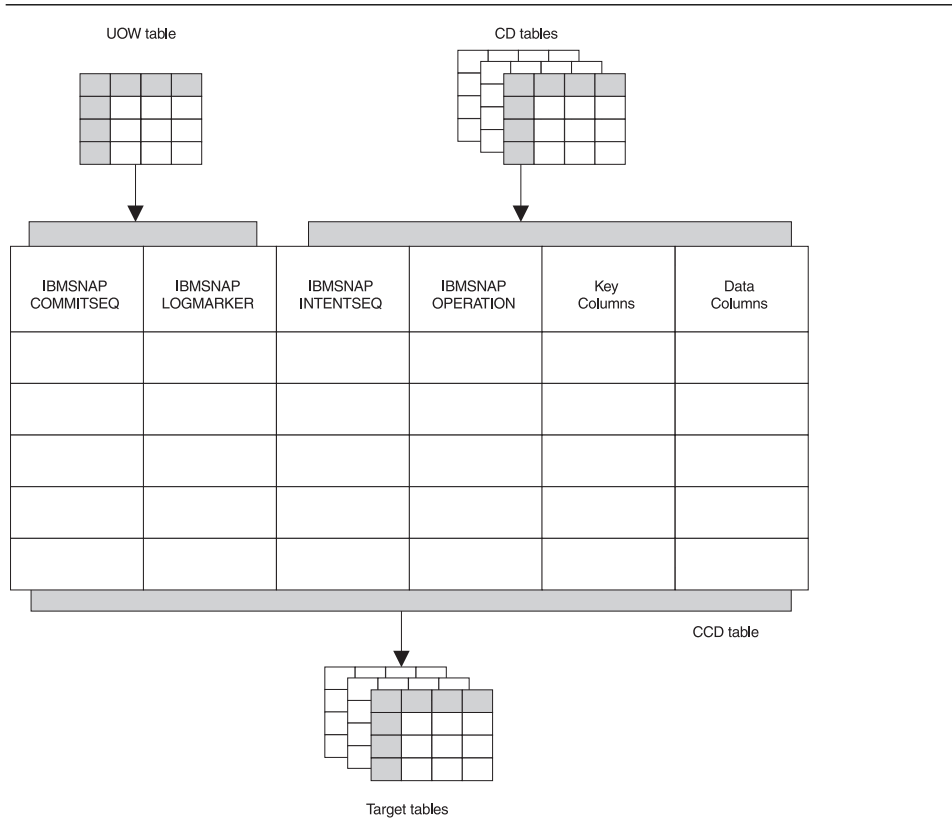


Figure 8. The CCD Table. This table is the join of the UOW table at the source server and the CD table for the replication source table.

Table 7 on page 61 shows the options that you have when you define these staging tables, and which options are the default selected by the Control Center.

Table 7. CCD Staging Table Attributes

CCD Is Local to:	Is CCD Complete?	Is CCD Condensed?	How Can CCD be Used in This Configuration?
Source table	Y ¹	Y	The CCD table is redundant with the local user table, and is therefore not usable as a history.
Remote copies	Y ²	Y	As a remote staging table, where history retention is not required.
Source table	N ²	Y	As a local staging table, which provides a stable source for synchronizing "fan out" copies.
Remote copies	N ¹	Y	By advanced users, who need to write their own apply programs to maintain indexed files or foreign DBMS tables. This configuration cannot support initialization of new point-in-time copies.
Source table	Y ¹	N	As local history table.
Remote copies	Y ²	N	As remote history table.
Source table	N ²	N	As a general-purpose, local staging table. When used with the original user table, this configuration can support copies that are complete histories. This table contains all change data for a given interval, but is not a complete history without the user table.
Remote copies	N ¹	N	By advanced users, whose application requirements are for change data only. This configuration cannot support initialization of point-in-time copies.

¹Table and definitions must be set up outside of the Control Center.

²The Control Center creates the table and sets up definitions.

Staging Tables

CCD staging tables hold captured changes from insert, update, or delete operations against a base table. CCD staging tables can be local to, or remote from, the original source.

A *complete* staging table contains every row of interest from the original source. A *condensed* staging table contains only the most current value for the row. Because condensed CCD tables do not have the same potential for unlimited growth as CD tables, they do not usually require pruning. Noncondensed CCD tables contain complete histories.

The Apply program inserts rows into noncondensed CCD tables; for condensed CCD tables, the Apply program updates rows already in the table. The Apply program uses CD and CCD tables as sources when maintaining replication subscriptions for point-in-time, change aggregate, user copy, and CCD target tables.

Benefits of Staging Data

You can use staging tables to:

- Maintain complete histories of data changes.
- Design generalized information and data warehouses.
- Support a variety of data delivery configurations.

- Minimize change data and unit-of-work join processing.
- Condense “hot spot” updates before transmitting data.

For example, you can use staging tables as part of a replication scenario that includes data from IMS and other sources. IBM's DataPropagator NonRelational can deliver IMS change data into a CCD staging table. You can then define the CCD table as a replication source.

With staging tables, you can set up sophisticated distribution networks and balance your work load across multiple DBMSs. You can copy DB2 for MVS changes to a DB2 Universal Database database, and then have, for example, 50 other replication subscriptions referring only to the staging tables. In this way, the DB2 for MVS database is required to maintain only one set of copies directly, although dozens of sets of copies are maintained indirectly.

Using Internal, Local, and Remote CCD Tables

When you create the first CCD table in a replication subscription of a source table and store the target table at the same source server, the target table is called an *internal CCD table*. Internal CCD tables for DB2 sources are created by joining the unit-of-work and CD tables locally. They serve as a local cache for committed changed data.

If you create a second CCD table replication subscription against that source table, and store the target table at the same source server, the target table is called a *local* CCD table; only the first CCD at the source server is the internal CCD. Figure 9 on page 63 shows an example of an internal or local CCD table that replicates changes to two target tables.

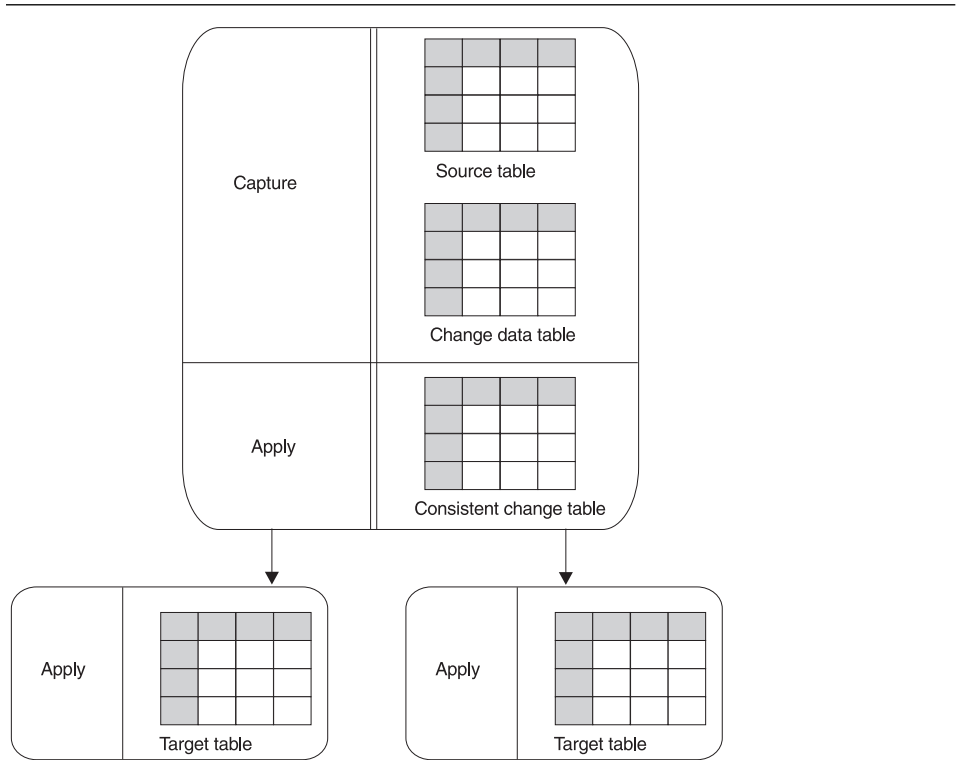


Figure 9. An Internal or Local CCD Table. The internal or local CCD table is located at the source server.

A CCD table at a server which is not the source server is called a *remote* CCD table. The remote CCD table can be located at the target server or an intermediate staging server as shown in Figure 10 on page 64.

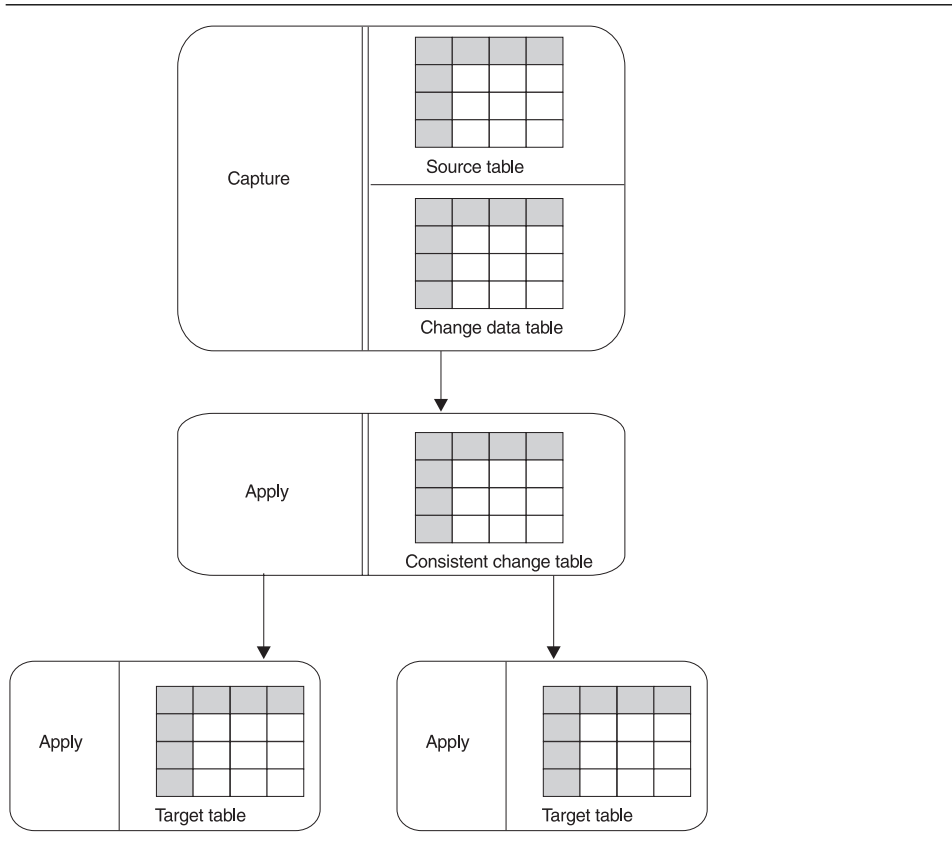


Figure 10. A Remote CCD Table. The remote CCD table is located at an intermediate staging server.

When you subscribe to a source table, the Control Center automatically defines certain target tables as a replication source for further copying, known as auto-registration. However, internal CCD tables are auto-registered differently than CCD tables. When internal CCD tables are auto-registered, replication source information is stored in the `CCD_OWNER` and `CCD_TABLE` columns of the source table row in the register control table. Staging tables created in subsequent replication subscriptions are defined as replication source tables.

This difference in auto-registration means that you cannot create replication subscriptions against internal CCD tables. Instead, you create replication subscriptions against the replication source table. The Apply program follows the hierarchy described in "How the Apply Program Selects a Source Table" on page 17 to select a source table from which to copy. In most cases, the Apply program chooses the CCD table associated with the source table.

External Data Sources

Changes captured within applications or other system tools, such as DataPropagator NonRelational, can also be defined as sources for replication subscription. The external data source must provide a complete CCD table and the CCD table must be updated by the application. For example, if an IMS segment is the source, DataPropagator NonRelational updates the DB2 CCD table. If the source table is not from IMS, you will need another program to update the CCD table. You can then define the CCD table as a surrogate replication source table with the Control Center. The CCD table can be stored and defined as a replication source in any supported database. You can then define replication subscriptions, regardless of whether the original transaction updates occurred in an IMS or DB2 database.

Transaction-Based versus Transaction-Consistent Replication: Using Internal CCD Tables to Reduce Network Load

IBM Replication supports both transaction-based replication (replication of every update used by every transaction) and nontransaction-based replication (replication of just the net results of the recent activity).

The following example illustrates the difference between the two types:

```
Transaction 1: Update table1 set col1 = 'X' where key1 = 425
               Update table2 set col2 = 'B' where key2 = 425
Transaction 2: Update table1 set col1 = 'Y' where key1 = 425
Transaction 3: Update table1 set col1 = 'Z' where key1 = 425
```

In transaction-based replication, all four transactions are captured and replicated. In transaction-consistent replication, only the second update in Transaction 1 and Transaction 3 are replicated.

Transaction-based replication is necessary in update-anywhere scenarios.

Transaction-consistent replication is superior to transaction-based replication because it produces the same change data results with fewer updates actually replicated. This type of replication reduces network load and can increase the availability of the target table.

You can implement transaction-consistent replication by using the internal CCD table model. In this model, outbound queues are condensed before replication, keeping only the latest captured value for each row. The Apply program condenses the queues by copying noncondensed change data already in the CD table into an internal (local) condensed CCD table. This CCD is then used as the source for replicating changes to the target table. Figure 9 on page 63 shows the configuration of an internal or local CCD table.

How External CCD Tables Are Refreshed (Cascade CCD Full Refresh)

When an Apply program refreshes an external CCD table, it deletes all of the rows from the pruning control table associated with the CCD table. The missing CCD table rows in the pruning control table indicate a full refresh and alert the Apply program that replication subscriptions based on the external CCD table must also be refreshed.

The Apply program keeps track of replication subscriptions based on the external CCD table in the following way:

- If the refreshed CCD table contains rows, the Apply program updates the value in the CCD_OLD_SYNCHPOINT column of the corresponding CCD row in the CD control table. The value is set to the minimum commit sequence value of the refresh rows applied to the CCD table.
- If the CCD table is empty as a result of the refresh, the CCD_OLD_SYNCHPOINT value is set to binary zeroes. You maintain the CCD_OLD_SYNCHPOINT value after pruning the CCD table because you are responsible for pruning the CCD table.

Developing a Data Warehouse with CCD Tables

You can define any number of replication subscriptions for CCD tables that refer to a source table. You can also define replication subscriptions referring to CCD tables that are remotely located from the original source table. In this sense, remote CCD tables become surrogate source tables. This introduces distribution databases or *warehouses* that serve as sources for all other copies. Changes captured on your operational systems need only be replicated once to the warehouse database to be applied to CCD tables in the warehouse database. These CCD staging tables can then serve as surrogate source tables for all other replication subscription definitions.

Replication Logical Partitioning Key Considerations

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

- Your source applications update one or more columns of a target table primary key.
- Your source applications update one or more columns of a target table partitioning key (either the target table is a partitioned database managed by DB2 Extended Enterprise Edition (EEE) table or a table in a DB2 for MVS partitioned table space).
- Your source applications update one or more columns referenced in a subscription predicate.

Without replication logical partitioning key support, when the primary keys of either the source or target tables are being updated, the Capture program captures the changed row for the update. The Apply program then attempts to make an update to a row on 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. The new row is fine, but 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 in the CD table.

For example, the primary key value for a table is the employee last name "Wood" and the employee gets married and changes her last name to "Smith". Without replication

logical partitioning key support, when the Apply program updates the employee's last name with a new primary key value, it does not find the value "Smith" and inserts a new row. The row with the primary key value "Wood" is not deleted.

With replication logical partitioning key support, the row with the primary key value "Wood" is first deleted, and a new row with the primary key value "Smith" is inserted.

Use the Control Center to specify that the Capture program should support replication logical partitioning key support while defining the replication source. See "Defining a Custom Replication Source" on page 95 to learn how to specify that updates are captured as DELETE and INSERT statements.

Data Sharing Considerations

You can implement replication in a data-sharing environment. In a data-sharing environment, you can run one Capture program per source data sharing group and one or more Apply programs per target data sharing group.

Extended Enterprise Edition Considerations

The Apply program and the replication administration function of the Control Center support replication to EEE target servers. Replication support for logical partitioning keys provides the ability to replicate data to partitioned tables with the Capture program option to capture updates as separate DELETE and INSERT operations.

To replicate to a EEE target server, you must select the **Changed data for partitioned key columns captured as delete and insert** option while defining the replication source from the Control Center.

Although the replication logical partitioning key support allows you to replicate to EEE servers, it is not directly related to the DB2 EEE logical partitioning key.

Data Restrictions

Currently, IBM Replication cannot replicate certain types of data. Here is a list of the major restrictions of which you should be aware:

- **Archive log access restrictions with DB2 for MVS V3.1, and DB2 for VSE & VM**

IBM Replication cannot replicate data that is held only in a DB2 archive log unless you are using DB2 for MVS V4.1 or higher or DB2 Universal Database.

- **Data compression restrictions with DB2 for MVS**

IBM Replication can replicate data that is compressed through DB2 software or hardware compression on DB2 for MVS V4.1 or higher if the compression dictionary that was used to compress the row is available. Before issuing REORG for compressed replication sources, perform one of the following tasks:

- Ensure that the Capture program has completed capturing all of the existing changes.
- Use the KEEPDICTIONARY option on the REORG command to preserve the existing compression dictionary.

IBM Replication cannot replicate data that is compressed through EDITPROCs or FIELDPROCs.

- **Utility program restrictions**

IBM Replication cannot capture the data updates made by any of the database utilities. This restriction applies also to data loaded with an option of LOAD RESUME LOG YES.

- **Data encryption restrictions**

IBM Replication cannot replicate data that is encrypted.

- **Data type restrictions**

Based on the current level of IBM Replication described in this book, here is a full list of the data types that cannot be replicated under any circumstances:

- Binary large objects (BLOBs)
- Character large objects (CLOBs)
- Double-byte character large objects (DBCLOBs)
- Any column on which an EDITPROC has been defined
- Any column on which a FIELDPROC has been defined
- Any column on which a VALIDPROC has been defined
- Binary data types with precision

Here is a list of the data types that can be replicated only under certain circumstances:

- Long variable graphic (LONG VARGRAPHIC) requires that the source and target tables are in DB2 for MVS, DB2 for VSE, or DB2 for VM.
- Any user-defined type (distinct data type in DB2 Universal Database) will be converted to the base data type before replication.

Limits on Column Names for Capturing Before-Image Data

When the change data table is created, the last character of the column name is truncated when the before-image column identifier (usually *x*) is added to it. The column name is truncated because DB2 limits column names to 18 characters. Therefore, before image data cannot be captured and replicated except with the following restrictions:

- On MVS, you can use 18-character column names but the first 17 characters of the column name must be unique within a table. (The 18th character will be replaced by the before image column identifier.)
- On all other platforms, column names can be a maximum of 17 characters and must be unique within a table.

Data Currency Requirements

Data currency reflects how up to date you want the copy tables to be. IBM Replication makes it very easy to control the Apply program processing interval and therefore the currency of the data. It is sometimes very tempting to start out wanting your copies to be as current as possible. In a test environment, you might be setting all frequency intervals to the minimum. When moving to production, it is advisable to start with a mid-range frequency value and “tune” your system from that point.

You can set an interval or relative timing schedule, or trigger the Apply program with events to start processing a replication subscription. Customers have devised many creative ways of controlling the replication subscription timing mechanism, but the most typical ways are interval and event-based timing.

You define subscription timing with the Subscription Timing notebook from the Control Center, and control the timing using time-based and event-based scheduling. These timing options can be used together or independently of each other. For example, you can set an interval of one day, but specify that an event can also trigger the subscription cycle.

Interval Timing (Relative Timing)

Interval timing, copying on a specified interval, is the simplest method of controlling subscription timing. You determine a specific start time, date, and interval, which can be a set time interval (from one minute to one year) or continuous copying. The entire subscription set is replicated on the same frequency. Time intervals are approximate. The Apply program will begin processing the replication subscription as soon as it can, based on its work load and the availability of resources. You can set and change the interval with the Control Center or by using standard SQL against the subscription set control table.

Continuous timing replicates the subscription set as frequently as the Apply program can replicate the data.

Choosing a timing interval does not guarantee that the frequency will be exactly at that interval. When you decide on the refresh interval for the replication subscription, you should determine whether it is possible to refresh all tables in the replication subscription within the interval. To check this, 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.

Event Timing

Event timing is copying that is triggered by an event. You specify an event name when you define the subscription in the Control Center, and then you have an application or a user populate the subscription events table with a timestamp for the event name. When the Apply program detects the event value, it begins replicating the replication subscription. You can use event-based timing together with interval timing for the same subscription.

The subscription events table, ASN.IBMSNAP_EVENT has three columns as shown in Table 8 on page 70.

Table 8. The Subscription Events Table

EVENT_NAME	EVENT_TIME	END_OF_PERIOD
END_OF_DAY	1997-12-01-17:00:00.000000	1997-12-01-15:00:00.000000

EVENT_NAME is the event that you specify while defining the replication subscription. The Control Center populates this column in the subscription set table after the replication subscription is defined. EVENT_TIME is the timestamp for the time when the Apply program begins processing the replication subscription. END_OF_PERIOD is an optional value that indicates that transactions after this time should be deferred until a future date. EVENT_TIME is set by the clock at the control server, while END_OF_PERIOD is set by the clock at the source server. These two databases may be on servers in different time zones, so this distinction is important.

In Table 8, END_OF_DAY is the event name that you specified while defining the replication subscription. The timestamp value, 1997-12-01-17:00:00.000000, is the time when the Apply program is to begin processing the replication subscription. The timestamp value, 1997-12-01-15:00:00.000000, is the transaction time when changes are no longer replicated; these transactions will be replicated on the next day's business cycle.

Your applications post events; the Control Center does not post events. You tie your applications to subscription activity through named events. If you post an entry using CURRENT_TIMESTAMP for EVENT_TIME, then you trigger the event named by EVENT_NAME. Any replication subscription tied to this event is now eligible to run. You can post events in advance, such as next week, next year, or every Saturday. If the Apply program is running, the Apply program will start at approximately the time that you specify. If the Apply program is stopped at the time you specify, and then restarted later, it checks the subscription events table, notices the posted event, and begins processing the replication subscription.

Data Consistency Requirements

IBM Replication always guarantees table consistency, that is, all data in the target table is actually committed in the base table. IBM Replication also provides consistency across sets of tables with the replication subscription, which can have multiple, related members. These members are the source and target pairs; and the members are usually bound by transaction relationships.

Because the replication members have something in common, the changed data needs to be copied in the same unit of work. Each replication subscription has a name and is identified by a row in the subscription set control table. A replication subscription with replica tables has two rows, one for each direction of copying (from source to replica and replica to source).

Figure 11 on page 71 shows the relationship of the replication subscription with the Apply program.

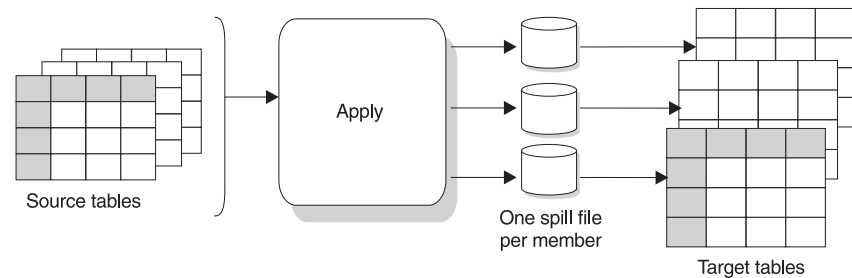


Figure 11. Replication Subscriptions and the Apply Program

The Apply program reads changes from the replication source or CD table and creates one spill file created for each table. The Apply program replicates the changes to the target tables.

The replication subscription helps maintain referential constraints that exist among a set of replication source and replica tables. The replication subscription also limits the boundary for cascade rejection due to RI violation or update collision. Finally, replication subscriptions keep the source-to-target definitions for all the components of a view target table together. The replication subscription should contain all target tables that are related, such as with RI constraints.

Replication Subscription Rules

The following section describes the rules and constraints you need to be aware of in planning and defining a replication subscription.

- If any member of the replication subscription requires full-refresh copying for any reason, the entire set is full refreshed. Full-refresh copying occurs only from the replication source to the replica, not the other way around.
- Each replica target table must be of the same generation as all the other replicas in the replication subscription and come from the same replication source table.
- A single synchpoint is maintained for the replication subscription to indicate the copy progress for the entire replication subscription.
- Collision detection levels must be specified while defining a replication source and affects the processing performed on a replica table in the replication subscription.
- Target tables with RI constraints must be full-refreshed by an outside means using the ASNLOAD exit to bypass RI checking.
- RI violations cannot be detected by application logic in an update-anywhere environment. Declarative RI constraints must be used.
- Referential constraints should not be defined for read-only target tables.

- All referential constraints that exist among the source tables should be included in the replication subscription replicas to prevent RI violations.
- The first occurrence of any RI violation terminates the current replication cycle. The subscription cycle will be retried automatically, after the transaction is rejected and compensated.
- All sources referenced by a given subscription set that are external CCDs must come from a single, original source.
- Each SOURCE_TABLE reference, whether a direct table reference or an indirect reference using a view, must be of the same generation as all other SOURCE_TABLE references in the set.

Rules for external programs that maintain CCD tables

If you want to maintain your own CCD table, you must update three columns: CCD_OLD_SYNCHPOINT, SYNCHPOINT, and SYNCHTIME. CCD_OLD_SYNCHPOINT is the SYNCHPOINT associated with the oldest row remaining in the CCD_TABLE, though this sequence number might be older if the CCD table is condensed and the oldest row was overwritten by a more recent update. SYNCHPOINT is a sequence value useful for maintaining the state of CCD copies, subscription states, and for controlling pruning. SYNCHTIME is the timestamp equivalent of SYNCHPOINT.

If the CCDs are maintained by a program external to IBM Replication, there is no source server or target server distinction (because the context is just one database).

- Before a full refresh of the CCD, set CCD_OLD_SYNCHPOINT in the register table to NULL.
- After a full refresh of the CCD, set the CCD_OLD_SYNCHPOINT to a value greater than the previous value of SYNCHPOINT. If SYNCHPOINT has no previous value (because this is the initial load), set the CCD_OLD_SYNCHPOINT to *x'00000000000000000000'*
- Set the SYNCHPOINT in the register table with MAX(IBM SNAP_COMMITSEQ) in the CCD whenever committing new changes to the CCD. (SYNCHTIME should also be set accordingly, since it is the timestamp equivalent of SYNCHPOINT.)

Update-Anywhere Replication

Update-anywhere replication is the replication of changed data from either a replication source table or a replica target table defined in a replication subscription. Unlike one-way replication, the target table (the replica) can be updated and the Capture program at the target server captures the changes and the Apply program pushes the changes back to the replication source at the source server. Updates to all of the replicas in a replication subscription are replicated together.

With the Capture program running at both the source and target servers, changes against the original user table (the replication source object) and its updateable copy (replica) can both be captured, and the Apply program then replicates these changes

from one server to the other to maintain data currency. This is also true for a one-to-many configuration (one source table to many replicas); changes to the replication source table can be copied to all of its replicas, and changes to any of the replicas can be copied to the replication source table. The changes are then copied to all other replicas.

Figure 12 shows a basic update-anywhere replication model. Changes are originated at both the source and replica tables.

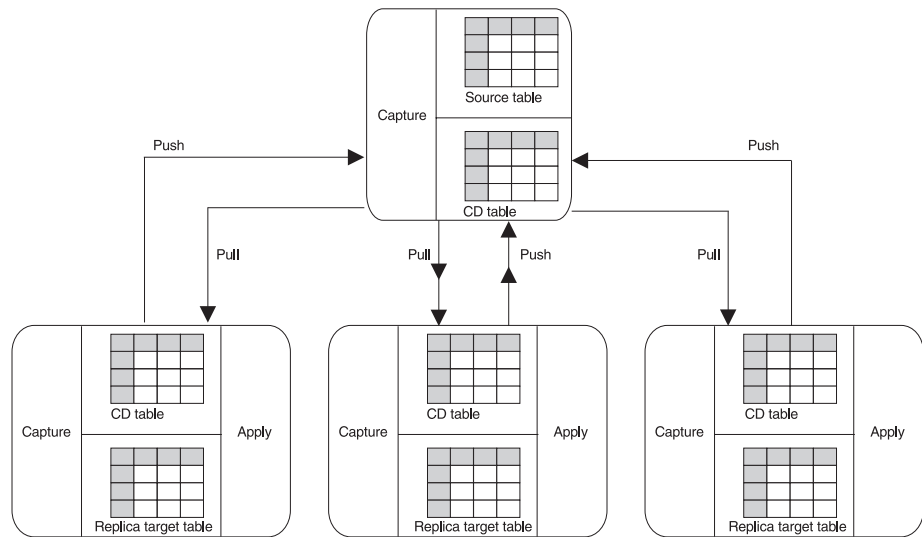


Figure 12. Update-Anywhere Replication. Update-anywhere replication is used to replicate changes from both the source and target servers.

Assessing the suitability of update-anywhere replication for your application is not so much a matter of looking for replication features as it is deciding what sort of anomalies are tolerable to your applications. Is it acceptable that sometimes updates are lost? If so, then you can turn off conflict detection and save processing cycles. Is it acceptable that occasionally a transaction is rejected and automatically compensated? These are the questions you need to ask, and include in your application design process.

The same-row conflict check done by the Apply program is only as good as the information provided. Updates that have not been captured are unknown to the Apply program, escaping the conflict check. Because the Capture program is asynchronous from the user transactions, there is a cost/risk decision that must be made. The cost is the availability of the table for update. The only way for the Capture program to synchronize with your updating applications is to block your updating applications until it has captured all outstanding updates. Taking table level locks on production tables has a negative impact on your production system.

The alternative to synchronizing with the application is to look at the risk of a conflicting update occurring during the Capture program latency window, and at the risk of losing

the occasional update. Some customers say that losing some updates is not a problem (For example, one application was a hotel reservations system where rooms are over booked as routine practice), and others design their update cycle so that there is no possibility of conflict.

Conflict Detection

The Apply program detects update conflicts, after they occur, during the subscription cycle. When transactions are rejected, the Apply program compensates the transactions at the replica.

The replication source is considered the primary, or global table. It can receive updates from replicas, but if there is a conflict, the replication source wins and the replicas' conflicting transactions are rejected. An update conflict occurs when an application and one or more replicas try to update the same version of a row in a table at different locations or when constraints are violated. Direct row conflicts are detected by comparing the key values in the CD tables with the source and target tables. If any match is found, the replica unit of work is marked as rejected in the UOW table and the transaction is compensated.

Conflict detection cannot detect read-dependencies. If, for example, an application reads information that is subsequently removed by compensation, the dependency cannot be detected.

Conflict detection is provided at three levels: no detection, standard detection, and enhanced detection. Each level has a numerical value and 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.

None No detection. The value is 0.

Standard The Apply program checks for conflicts within the currently captured information, but with the risk of not seeing updates still held in the log. The value is 1.

Enhanced The Apply program takes a shared table lock on the application table during the conflict check and updating period. The value is 2.

Recovering from Conflicts

By using rejection codes provided in the UOW table, you can identify the before and after row values in the change data table for each rejected transaction. When the Apply program compensates the rejected transactions, you must decide how to handle the rejected transactions. The ASNDONE user exit notifies you when the subscription cycle is completed, regardless of whether there are conflicts or rejected transactions. You can have your application handle the rejected transactions and have the Apply program run your application logic in the ASNDONE user exit. See the Capture and Apply chapter in this book for your platform for more information on the Apply program ASNDONE exit. You can choose to handle the rejections in batch; the change data rows and UOW control table rows for rejected transactions are exempt from normal pruning. They are subject only to RETENTION_LIMIT pruning.

The following suggestions for handling the rejected transactions have not been tested in user environments and should be used as suggestions:

- In a mobile environment, develop part of the application to run at the end of the Apply program's subscription cycle, immediately notifying the user of any conflicts that might be replicated to other servers.
- In a server-to-server configuration, write a batch application that periodically scans the UOW control tables at the replica sites for evidence of rejected transactions.
- Automate the re-running of any rejected transaction, putting them into a proper transaction sequence.

CCD Tables

Because update-anywhere replication requires referential integrity, CCD tables are not used in update-anywhere replication. CCD tables stage data and transactions are condensed. In update-anywhere replication, every transaction update should be replicated to the target server in the original order it occurred, which does not occur with CCD staging tables. Additionally, there may be triggers defined which rely on reading each update. Because you want to synchronize the source and target servers as up-to-the-minute as possible to detect update conflicts, you want to prevent the delays imposed by CCD staging. If you defined an internal CCD at the source server, it is ignored by the Apply program when processing a subscription with a replica as a target.

Recommended Usage

To reduce the risks of conflicts and cost 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.

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.

Restrictions

The following list describes restrictions for update-anywhere replication:

- Enhanced conflict detection requires simultaneous availability of the Capture and Apply programs.
- RI violations cannot be detected by application logic in an update-anywhere environment. Use declarative RI constraints.

If referential constraints that exist among the source tables are omitted from the corresponding set of replicas, an update made to a replica could result in an RI violation when replicating the update to a source table.

- Update-anywhere is not supported for tables with 18-character column names except with restrictions. See “Limits on Column Names for Capturing Before-Image Data” on page 68 for more information.

Minimizing Contention between Capture and Apply for MVS 3.1 Sources and Targets

Contention can occur when the Capture and Apply programs try to access the same table on DB2 platforms that do not support ISOLATION (UR).

To minimize contention:

- Use the NOPRUNE Capture program option to defer pruning so that pruning activity does not block the Apply program.
- Minimize the number of Apply program instances copying from the same source server. The more Apply program instances that copy from the same source server, the higher the probability that the Capture and Apply programs encounter contention difficulties.
- Minimize the number of Apply program instances copying from the same source table. The more Apply program instances that copy from the same source table, the higher the probability that the Apply program instances encounter contention over access to the pruning control tables.
- Schedule the Capture and Apply programs to run at different times of the day.
- Shorten the Capture program commit intervals (in the tuning parameters table), and lengthen the frequency of the Apply interval (on the Frequency page of the Subscription Definition notebook).

How Locking Affects Contention on Platforms without ISOLATION (UR) Support

Change data tables and UOW tables can be held for a long time; they can be locked in two ways: global locks or a single LOCK TABLE. The first is a global lock, which is a lock against the global critical section table. A single LOCK TABLE, obtained before the table is accessed, is issued.

The Capture program gets the critical section lock in *exclusive* mode before it starts capturing changes. During this time, the Apply program cannot access those tables; the Apply program must get the critical section lock in *shared* mode. After this sequence of change captures (that is, after a scheduled COMMIT), the Capture program becomes inactive, allowing one or more Apply program processes to get the lock in shared mode.

You can decrease the COMMIT_INTERVAL in the tuning parameters table; this forces the Capture program to give up the lock more often, giving the Apply program the opportunity to access the change data table. If you decrease the COMMIT_INTERVAL value to less than 20 seconds while using automatic pruning, you also increase the frequency of pruning while slowing performance.

The benefit of trying to make change data available to the Apply program immediately is outweighed by the increased chance of contention between the Capture and Apply programs.

Resolving Gaps between Source and Target Tables

Occasionally, a gap can occur between the capturing of the changed data for a source table and the replication of the changed data to the target table. When this happens, a user with sufficient privileges might need to reset the control table information before executing the definition again. However, to preserve data integrity, check the control tables and take any necessary action.

For example, if you shut down the Capture program and then cold start it, it deletes all rows from the CD table. (See the Capture and Apply chapter for your platform in this book to learn more about cold start.) Between the time you shut down the Capture program and cold started it, updates might have been made that the Capture program did not capture. Additionally, any updates that were in the CD table were deleted at the cold start before the Apply program could copy them. A gap now exists between the target table and the CD table.

When a gap is present, the Apply program attempts to refresh complete copies unless the target table is not complete (COMPLETE=N). If the target table is not complete and the Apply program cannot perform a refresh, data integrity could be lost. In this case, before you reset the copy definition and resume copying, you need to check:

- The control information in the subscription set control table of the target table (ASN.IBMSNAP_SUBS_SET).
- The target table itself
- The base table

The Capture and Apply programs use the synchpoint to coordinate their work. The synchpoint is the log sequence number and indicates the progress of the replication subscription through a subscription cycle. The Capture and Apply programs maintain and use this value to prevent the pruning of data that the Apply program did not copy. The synchpoint value is maintained in the SYNCHPOINT column of the subscription set control table.

To verify whether a gap exists:

Compare the SYNCHPOINT column in the subscription set control table with the CD_OLD_SYNCHPOINT column in the register control table. If SYNCHPOINT is lower than CD_OLD_SYNCHPOINT, a gap exists.

To resolve the gap problem:

Determine whether continued performance or data integrity is more important.

- If data integrity is important, manually insert the missing rows into the target table.

- If continued performance of the Apply program is more important, set CD_OLD_SYNCHPOINT to a lower value so that the Apply program continues regardless of the data loss.

Using Your Own Full Refresh Technique for External CCD Tables

This section uses an example to describe the manual load procedure to use when the source table is a CCD table and there is a Capture program at the source site. If there is no Capture program at the source site, you will need to modify the procedure.

In this example:

- The source table has been defined as a replication source.
- A subscription has been defined with the following attributes:
 - Subscription name: CCDSUB
 - Subscription qualifier: CCDQUAL
 - Source Table name: SOURCE_TABLE
 - Target Table name: SOURCE_CCD
 - Target Type: Staging table used as a source for copies
- A second subscription has been defined with the following attributes:
 - Subscription name: UCSUB
 - Subscription qualifier: UCQUAL
 - Source Table name: SOURCE_CCD
 - Target Table name: SOURCE_USER_COPY
 - Target Type: User copy

The following steps describe the manual load procedure:

1. Disable full refresh

Issue the following SQL on the server where the SOURCE_TABLE is located:

```
UPDATE ASN.IBMSNAP_REGISTER SET DISABLE_REFRESH=1
WHERE SOURCE_OWNER = 'source owner' AND
      SOURCE_TABLE = 'SOURCE_TABLE'
```

Issue the following SQL on the server where the SOURCE_CCD table is located:

```
UPDATE ASN.IBMSNAP_REGISTER SET DISABLE_REFRESH=1
WHERE SOURCE_OWNER = 'source owner' AND
      SOURCE_TABLE = 'SOURCE_CCD'
```

2. Deactivate the subscriptions (or stop the Apply processes)

Issue the following SQL on the control server for the CCD subscription:

```
UPDATE ASN.IBMSNAP_SUBS_SET SET ACTIVATE = 0
WHERE SET_NAME = 'CCDSUB' AND APPLY_QUAL = 'CCDQUAL'
```

Issue the following SQL on the control server for the user copy subscription:

```
UPDATE ASN.IBMSNAP_SUBS_SET ACTIVATE = 0
WHERE SET_NAME = 'UCSUB' AND APPLY_QUAL = 'UCQUAL'
```

3. If it is not already started, start the Capture program on the server where the SOURCE_TABLE is located. It is very important that you start the Capture program before proceeding to the next step.

4. Update the ASN.IBMSNAP_PRUNCNTL row for the CCD subscription

Issue the following SQL on the server where the SOURCE_TABLE is located:

```
UPDATE ASN.IBMSNAP_PRUNCNTL
      SET SYNCHPOINT = x'000000000000000000000000',
          SYNCHTIME = CURRENT_TIMESTAMP
WHERE SET_NAME = 'CCDSUB' AND APPLY_QUAL = 'CCDQUAL'
```

5. Verify that the Capture program has processed the PRUNCNTL update

Issue the following SQL on the server where the SOURCE_TABLE is located:

```
SELECT MIN(SYNCHPOINT), SYNCHTIME
      FROM ASN.IBMSNAP_PRUNCNTL
WHERE SET_NAME = 'CCDSUB' AND APPLY_QUAL = 'CCDQUAL'
```

Make a note of these values since they will be used later (in this example as ccdsynchp and ccdsyncht).

6. Unload the SOURCE_TABLE and load the SOURCE_USER_COPY using DB2 or vendor utility programs

Note: If you want to bypass the full refresh, you can skip this step. However, source updates committed between the last change processed by the Apply program and the update of ASN.IBMSNAP_PRUNCNTL will not be replicated.

7. Update the ASN.IBMSNAP_REGISTER row for the SOURCE_CCD

Issue the following SQL on the server where the SOURCE_CCD is located:

```
UPDATE ASN.IBMSNAP_REGISTER
      SET CCD_OLD_SYNCHPOINT = ccdsynchp,
          SYNCHPOINT = ccdsynchp,
          SYNCHTIME = ccdsyncht
WHERE SOURCE_OWNER = 'source owner' AND
      SOURCE_TABLE = 'SOURCE_CCD'
```

Note that ccdsynchp and ccdsyncht are the values from Step 5.

8. Update the ASN.IBMSNAP_SUBS_SET row for the SOURCE_CCD

Issue the following SQL on the control server for the CCD subscription:

```
UPDATE ASN.IBMSNAP_SUBS_SET
      SET LASTRUN = CURRENT_TIMESTAMP
          LASTSUCCESS = CURRENT_TIMESTAMP
          SYNCHTIME = CURRENT_TIMESTAMP
          SYNCHPOINT = NULL
WHERE SET_NAME = 'CCDSUB' AND APPLY_QUAL = 'CCDQUAL'
```

- |
- | 9. Update the ASN.IBMSNAP_PRUNCNTL row for the user copy subscription

| Issue the following SQL on the server where the SOURCE_CCD is located:

| UPDATE ASN.IBMSNAP_PRUNCNTL
| SET SYNCHPOINT = ccdsynchp,
| SYNCHTIME = ccdsyncht
| WHERE SET_NAME = 'UCSUB' AND APPLY_QUAL = 'UCQUAL'

| Note that ccdsynchp and ccdsyncht are the values from Step 5.

- | 10. Update the ASN.IBMSNAP_SUBS_SET row for the SOURCE_USER_COPY

| Issue the following SQL on the control server for the user copy subscription:

| UPDATE ASN.IBMSNAP_SUBS_SET
| SET LASTRUN = ccdsyncht,
| LASTSUCCESS = ccdsyncht
| SYNCHTIME = ccdsynchp
| WHERE SET_NAME = 'UCSUB' AND APPLY_QUAL = 'UCQUAL'

- | 11. Start the Apply program or set ACTIVATE = 1 for the applicable subscription set row
| in ASN.IBMSNAP_SUBS_SET

| Issue the following SQL on the control server for the CCD subscription:

| UPDATE ASN.IBMSNAP_SUBS_SET SET ACTIVATE = 1
| WHERE SET_NAME = 'CCDSUB' AND APPLY_QUAL = 'CCDQUAL'

| Issue the following SQL on the control server for the user copy subscription:

| UPDATE ASN.IBMSNAP_SUBS_SET SET ACTIVATE = 1
| WHERE SET_NAME = 'UCSUB' AND APPLY_QUAL = 'UCQUAL'

| **Note:** Since this example uses two Apply qualifiers, you would need to start an
| Apply process for each qualifier.

Part 3. Administering Your Replication System

Administering a replication system involves identifying source tables and views, and defining the location, structure, and subscription cycle of the target tables. This part describes administration concepts and tasks that can be performed using the DB2 Universal Database Control Center.

Chapter 7. Administration Overview

You use the Control Center 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 kick off replication. The administration tasks described in this chapter set up the control information that both the Capture and Apply programs use to capture updated data and then replicate it to the target tables, in the proper format, and at the appropriate interval.

Replication administration has two major tasks: defining replication sources and defining replication targets (subscriptions). Replication sources are DB2 tables or views used as sources for copying data to one or more target tables. Replication subscriptions are the specifications for one or more target tables and their location, structure, and timing schedule, as well as any SQL enhancements that are necessary. Other replication tasks include maintaining the replication sources and subscriptions, and cloning them to other servers.

Overview of Replication Administration Steps

When setting up the replication environment, you can use the Control Center to manage the source and target table definitions and the control tables. See “Replication Products” on page 39 for information about how the Control Center is packaged. The following list describes the high-level steps you use to administer your replication objects. See Chapter 3, “Getting Started” on page 21 for a sample scenario for most of these tasks.

1. Check and optionally update the default settings in the Tools Settings notebook.
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 “Working with Customized Replication Control Tables” on page 88 for more information.
4. Define and manage replication sources. See Chapter 8, “Working with Replication Sources” on page 93 for more information.
5. Define and manage replication subscriptions. See Chapter 9, “Working with Replication Targets” on page 103 for more information.

Once you have created the control tables and defined the replication sources and targets, you need to configure and run the Capture and Apply programs to begin replicating data.

This chapter describes the following tasks:

- How to navigate through the Control Center to your replication objects
- How to set replication preferences (default settings) in the Tools Settings notebook
- How to customize the DPCNTL file and create the control tables

- How to customize and run SQL files generated by replication requests
- How to specify ordinary and delimited identifiers

Navigating to Your Replication Objects with the Control Center

You can access your replication sources and targets through the Control Center interface. 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 relational 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 definitions for copying updated source data to target tables.

Figure 13 shows the Control Center. The folders you use for replication tasks are in the tree structure on the left side of the window, called the object tree.

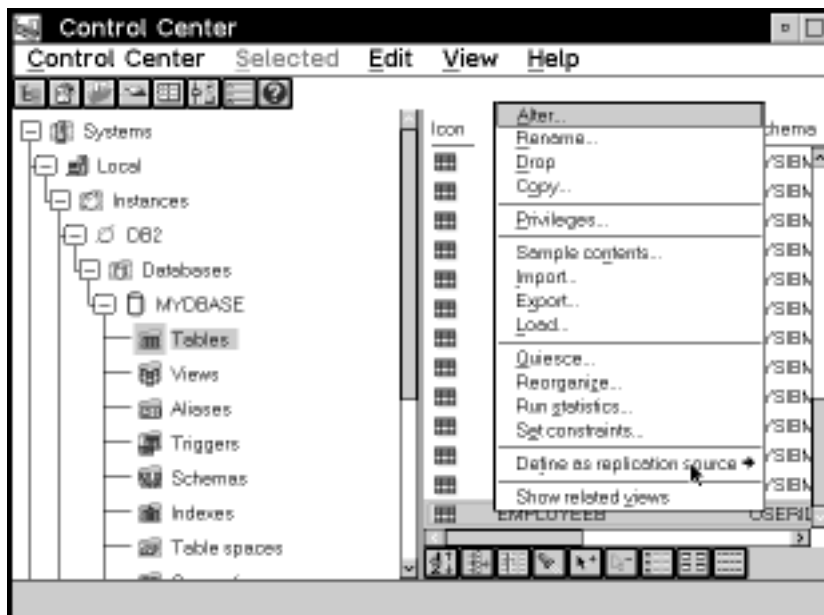


Figure 13. The Replication Folder Objects in the Control Center Window. The Replication folder objects are used to administer your replication environment.

Each object also has a menu with the actions that can be performed with the object. In Figure 13, a table object has been selected and the pop-up menu is displayed with the **Define as replication source** action selected.

To access the replication folders and objects:

1. From the Control Center window, expand the object tree until you find the replication folder that you need to use: **Tables**, **Replication Sources**, or **Replication Subscriptions**.
2. Click on the folder. All objects contained in the folder are displayed in the pane on the right side of the window (the contents pane).
3. Click mouse button 2 on the object you want in the contents pane and select an action from the pop-up menu. A window related to the action opens or a process is launched.

In addition, some actions can be performed against an entire folder. For example, you can refresh the **Replication Sources** folder.

Configuring the Control Center for Non-DB2 Universal Database RDBMSs

If you are connecting to a DB2 for MVS, DB2 for VSE, or DB2 for VM source or target database server from the Control Center, you must bind the database to the Control Center.

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 installation drive of the Capture program.
2. Create and bind the DB2 for MVS, VSE, or VM package to the DB2 database by entering the following command:

```
DB2 BIND @DDCSxxx.LST ISOLATION UR BLOCKING ALL SQLERROR CONTINUE
```

Where *UR* specifies the list in uncommitted read format for greater performance and *xxx* specifies the platform name: MVS, VSE, or VM.

Additionally, you might need to perform an explicit connect from the Control Center. If the password is different than the local logon ID and password at the Control Center workstation, it is recommended that you explicitly connect to the database server using the **Connect** menu choice from the pop-up menu for the icon of your DB2 for MVS database.

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 14 on page 86. These default values are used for all replication activities administered by the Control Center.

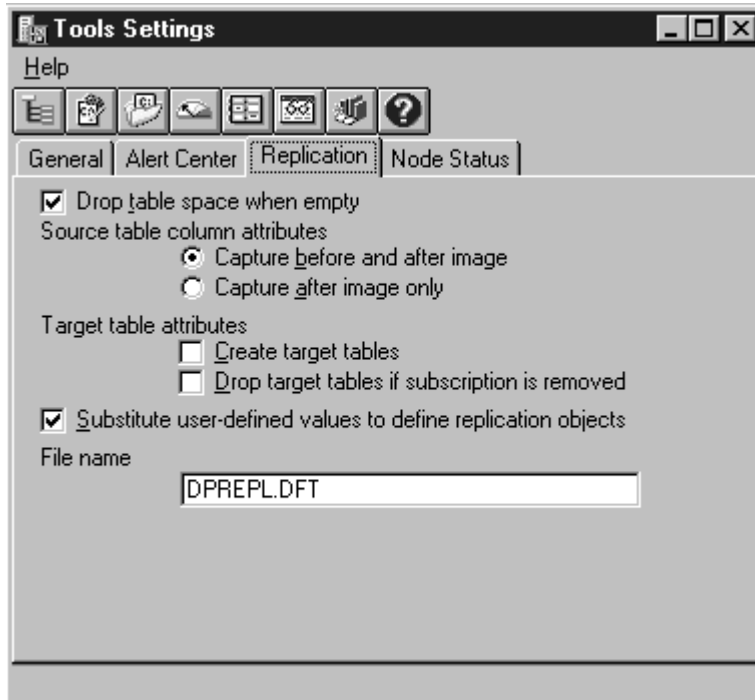


Figure 14. The Replication Page of the Tools Setting Notebook. The Replication page of the Tools Settings notebook contains default preferences for replication.

The default values for replication sources and subscriptions are:

Table spaces

Change data table and target table spaces are dropped when empty.

Replication sources

Before- and after-image columns are captured.

Replication targets

- Target tables must be created by the user.
- Target tables are not dropped if the subscription is removed.

To set replication preferences in the Tools Settings notebook:

1. From the Control Center, select the **Tools Settings** icon in the tool bar as shown in Figure 15 on page 87.

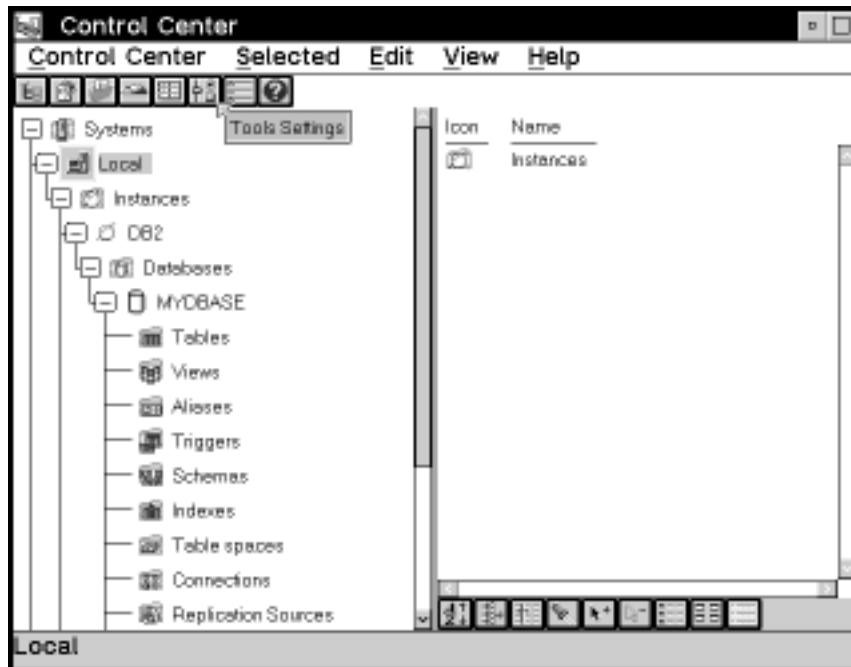


Figure 15. The Tools Settings icon. The Tools Setting icon opens the Tools Setting window where you can set replication preferences.

The Tools Settings notebook opens.

2. Click on the **Replication** tab to open the Replication page.
3. If you want both the table space for the change data table (at the source server) and the table space for the target table (at the target server) to be dropped when the table space is empty, select the **Table spaces are dropped when empty** check box.

When selected, table spaces are dropped for the specified table types on MVS, OS/2, Windows NT or 95, AIX, HP-UX, Solaris, UnixWare 7, VSE, and VM.

4. For capturing before- and after-image columns on the replication source, select a radio button:

Capture before and after image

The before-image and after-image of the source table column are captured and can be replicated to a target table. The before- and after-image columns will have the following values in the target table depending on the type of change made to the source column:

Insert

The before-image column has a NULL value.

Delete

Both the before- and after-image columns contain the before-image 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 the target table is initialized (or full refreshed due to a Capture program cold start), before-image columns have NULL values.

Capture after image only

Only the after image of the source table column is captured and can be replicated to a target table.

5. If you want the Control Center to create the target table automatically while defining a replication subscription, select the **Create target tables** check box. If the target table is not selected, you need to create your own target table. See “Defining a Replication Subscription with a User-Defined Target Table” on page 120 for more information about defining and maintaining target tables.
6. If you want to drop affected target table and related control tables when the replication subscription set is removed, select the **Drop target tables if subscription is removed** check box.
7. Close the Tool Settings notebook to save your changes.

Working with Customized Replication Control Tables

The replication control tables are created in two ways:

- By customizing the DPCNTL file for your platform and then running the file before running any actions from the Control Center.
- By running the first replication request at a server from the Control Center to create the default version of the control tables.

If you use the second option, you cannot customize the replication control tables for your site without dropping the existing control tables and customizing the tables as discussed below. The default custom tables are designed for the DB2 Universal Database platforms. If you are running on MVS/ESA, VSE/ESA, or VM/ESA, you *must* customize the replication control tables.

You need to customize the file for your platform before performing any replication action at the server. See comments within the file for tailoring the SQL to run on a specific database platform. You will need to customize the DPCNTL file for the following definitions:

- To define and size DB2 for MVS table spaces and databases for the control tables. The Control Center automatically creates the control tables in the default table space and database, unless you specify a different table space or database.
- To define DB2 for MVS V3.1 padding to support row-level locking.
- To define and size DB2 for VSE or VM dbspaces for the control tables. The Control Center will automatically create the control tables in the default dbspace, unless you specify a different dbspace.

- To tailor the control tables to run on specific platforms because not all definitions are supported on all platforms.
- To define additional indexes for better performance on a specific platform.
- To place control tables in specific DB2 Universal Database table spaces or in a DB2 EEE single node group.

When you create customized control tables, you need to customize SQL files for the CREATE TABLE statements. The DPCNTL file contains the SQL for creating control tables. There is a DPCNTL file for each platform located in the SQLLIB\SAMPLES\REPL\ directory. The files names are:

DPCNTL.UDB

Creates control tables for the DB2 Universal Database platform.

DPCNTL.MVS

Creates control tables for the DB2 for MVS platform.

DPCNTL.VM

Creates control tables for the DB2 for VSE & VM platforms.

To customize the SQL for creating control tables:

1. Open the SQLLIB\SAMPLES\REPL\DPCNTL.*platform_name* file as ASCII in an editor, where *platform_name* is the platform name in the previous list.
2. Read the commented areas for each platform and table.
3. Edit the file for your site or application.
4. Close the file.
5. After you have connected to the database in which the control tables will be created (via **DB2 CONNECT TO** *database-name*), run the file by entering the following command from a command line window:

```
db2 -tf dpcntl.platform_name
```

When you drop customized control tables, you need to customize SQL files for the DROP TABLE statements. The DPNCNTL file contains the SQL for dropping control tables. There is a DPNCNTL file for each platform located in the SQLLIB\SAMPLES\REPL\ directory. The files names are:

DPNCNTL.UDB

Drops control tables for the DB2 Universal Database platform.

DPNCNTL.MVS

Drops control tables for the DB2 for MVS platform.

DPNCNTL.VM

Drops control tables for the DB2 for VSE & VM platforms.

To customize SQL for dropping the control tables:

1. Open the SQLLIB\SAMPLES\REPL\DPNCNTL.*platform_name* file as ASCII in an editor, where *platform_name* is the platform name in the previous list.

2. Read the commented areas for each platform and table.
3. Edit the file for your site or application.
4. Close the file.
5. After you have connected to the database in which the control tables will be created (via **DB2 CONNECT TO** *database-name*), run the file by entering the following command from the command line window:

```
db2 -tf dpcnt1.platform_name
```

Customizing and Running Replication SQL Files

You have the option to run a replication task immediately or save the SQL generated by a replication action in an SQL file to be updated and run at a later time. The SQL files can be customized for large scale replication actions such as defining subscriptions, or customized for the application beyond implementations supported by the Control Center.

SQL files allow you to customize the replication tasks for your site or application and give you flexibility as to when and how you run the files. You can:

- Batch the definitions together.
- Defer execution of 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 in a mobile environment.

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

Types of customizing that you might do include:

- Creating multiple copies of the same replication action, customized for multiple servers, and run a large scale action.
- Customizing CD table names.
- Defining the location for CD tables (DB2 for MVS database, DB2 Universal Database table spaces, DB2 for VSE & VM dbspaces).
- Creating and sizing the table spaces, databases, or dbspaces of the CD tables.
- Defining site-specific standards.

After completing a replication task, such as defining a replication source, the Run Now or Save SQL window opens, giving you a choice to run the SQL for the task now, or save it so you can edit it and run it later.

To defer the replication action SQL files:

1. Select the **Save SQL to file and run later** check box in the Run Now or Save SQL window.
2. Click on **OK**. The Save SQL File window opens.

3. Type the name and file path for the file.
4. Click on **OK** to close the window and save the SQL file. A confirmation window indicates that the file has been saved.

After you defer the SQL file, you can customize the file for your application.

To customize a replication action SQL file:

1. Open the SQL file that you saved as ASCII in an editor.
2. Edit the file for your site or application.
3. Click on **OK** to close and save the file.

You can run deferred files from the Control Center as described below.

To run deferred replication action SQL files:

1. In the Control Center, select the replication folder for the object type that you are running the SQL file for. For example, if you saved and customized an SQL file to define a replication source, select the **Replication Sources** folder.
2. Click mouse button 2 to display the pop-up menu for the folder and select the **Run SQL** menu choice. The Run SQL File window opens (Figure 16).



Figure 16. The Run SQL files menu choice. The Run SQL files menu choice is used to run deferred SQL files for the replication object you select.

3. Select the directory path and file name of the SQL file.

4. Click on **OK** to run the file. A confirmation window opens indicating when the SQL file has run successfully.

You can also run the SQL file from the DB2 Universal Database Command Line Processor (CLP) or the DB2 Universal Database Command Center.

Ordinary and Delimited Identifiers

Many of the fields in the Control Center allow two types of SQL identifiers: ordinary identifiers and delimited identifiers. Many replication values, such as table names, require that the case of the value match the name of the object. The online help and the tasks in this book specify which fields accept both ordinary or delimited identifiers.

An ordinary identifier:

- Must start with a letter
- Can include uppercase letters, numbers, and underscores (`_`)
- Cannot be a reserved word

Note: If you type a lowercase letter as part of an ordinary identifier in a field, the lowercase letter is stored as an uppercase letter.

A delimited identifier:

- Must be enclosed within quotation marks ("")
- Can include uppercase and lowercase letters, numbers, underscores (`_`), and spaces
- Can contain a quotation mark, represented by two consecutive quotation marks

Note: Leading spaces are stored as part of a delimited identifier, but trailing spaces are ignored.

To name a target table WKLYSAL, you can type: WKLYSAL or wklysa1 or Wklysa1 in the **Target table** field. The lowercase letters are changed to uppercase and the table name is stored as WKLYSAL. If you want the name of the table to be Wkly Sa1, type: "Wkly Sa1" in the **Target table** field. The quotation marks are recognized as the delimiters of the identifier, and the table name is stored as Wkly Sa1.

If you want the name of a column to be "Nickname", type: ""Nickname"" in the **Column name** field. The outermost quotation marks are recognized as the delimiters of the identifier and each pair of consecutive quotation marks represents one quotation mark. The column name is stored as "Nickname".

Chapter 8. Working with Replication Sources

This section describes replication sources and how to work with them in your replication environment.

What Is a Replication Source?

A replication source is a table used as a source for copying data to a target table. Replication sources can be the following types of tables:

User table

A table created and used (read and updated by an application) outside of replication. Updates to the table can be captured and copied to a replication target table.

Replication target table

Read-only target tables that have been redefined as sources for replication.

Consistent change data (CCD) tables

A staging table, a target table type that has been redefined as a source table for staged replication.

External CCD table

A consistent change data table that has been populated from an external source such as an IMS segment.

Replica table

A target table that is used to replicate data back to the source table.

When you define a table as a replication source, a replication source object is created and a row is inserted into the register control table, making it available as a source table for replication subscription. Unless you specify full-refresh copying, the source table is enabled for change data capture: the table is checked for the Data Capture Changes option of the CREATE TABLE SQL statement. If it is not already enabled, the source table is altered to enable the option.

Defining Replication Sources

To define a replication source, use the Control Center Object Tree to navigate to the **Tables** folder. The **Define as replication source** pop-up menu choice allows you to define a table object as a replication source. Figure 17 on page 94 shows the pop-up menu for the table object and the **Define as replication source** menu choice.

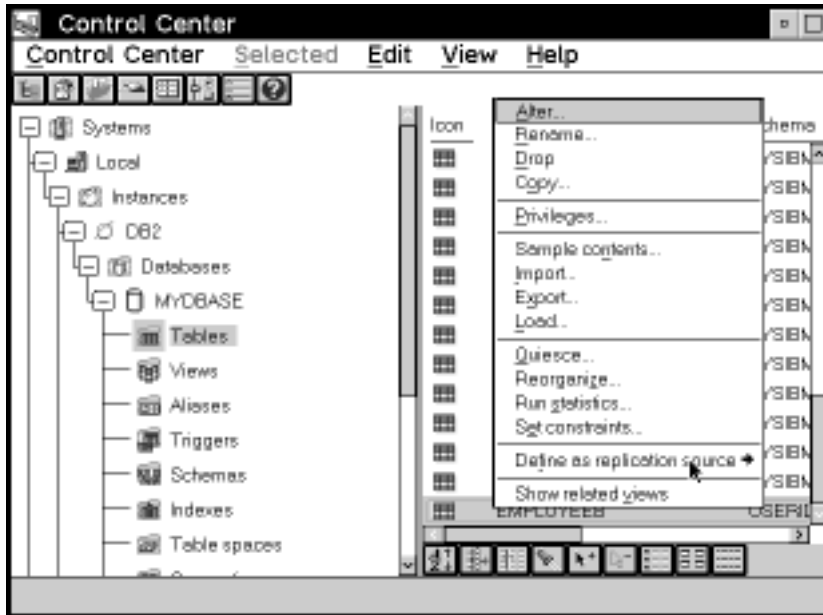


Figure 17. The Table object pop-up menu. The Table object pop-up menu contains the Define as Replication Source menu action.

You can define replication sources using the **Quick** or **Custom** pop-up menu choices. **Quick** allows you to define a replication source using the defaults set in place. **Custom** allows you to customize the defaults, such as specifying any columns that should not be captured for update replication.

When you finish defining the source, you can select whether to run the SQL that defines the source, or defer and run the file later. When you run the SQL file to define the source, an object is created in the **Replication Sources** folder. The table can now be defined in a subscription set.

For data restrictions when defining replication sources and targets, see “Data Restrictions” on page 67.

Defining a Replication Source with the Default Values (Quick)

Use the **Define as Replication Source -> Quick** menu choice, available from a table object on the right side (contents pane) of the Control Center, to define tables as sources for replication using the default values.

To define a default replication source with the Quick menu choice:

1. Check the default settings in the Tools Settings notebook on the Replication page. See “Setting Replication Preferences in the DB2 Tools Settings Notebook” on page 85 for a list of the setting defaults and instructions for opening the notebook and changing the settings.

Additional default values can be changed only by using the **Custom** menu choice and selecting choices on the Define as Replication Source window.

- All source columns are available for data capture.
- Partition key columns are captured as UPDATE statements by the Capture program.
- If the source table will be used in update-anywhere replication, conflict detection is STANDARD.

If these default values are not acceptable for your application, go to “Defining a Custom Replication Source.”

2. In the Control Center, navigate to the **Tables** folder in the source database and select it. The tables in the source database are displayed on the right side of the window contents pane.
3. From the contents pane, click mouse button 2 on the source table icon and select the **Define as Replication Source -> Quick** menu choice to launch the definition of the source. The Run Now or Save SQL window opens. Specify whether to submit the replication request now or save the SQL statements generated by the action in a file to be run later. See “Customizing and Running Replication SQL Files” on page 90 to learn how to submit the request now or later.
4. After you submit the request, verify that the replication source was created by selecting the **Replication Sources** folder in the object tree and selecting the **Refresh** menu choice.
5. To begin capturing changes to the source table, start or reinitialize the Capture program. See the Capture and Apply chapter for your platform in this book for more information about the Capture program.

Defining a Custom Replication Source

Use the Define as Replication Source window to identify tables that are available as sources for replication.

To define a custom replication source with the Custom menu choice:

1. Check the default settings in the Tools Settings notebook on the Replication page. See “Setting Replication Preferences in the DB2 Tools Settings Notebook” on page 85 for a list of the settings defaults and to learn how to open the notebook and change the settings.
2. In the Control Center, navigate to the **Tables** folder in the source database and select it. The tables in the source database are displayed on the right side of the window contents pane.
3. From the contents panel, click mouse button 2 on the source table icon and select **Define as Replication Source -> Custom** to open the Define as Replication Source window, as shown in Figure 18 on page 96.

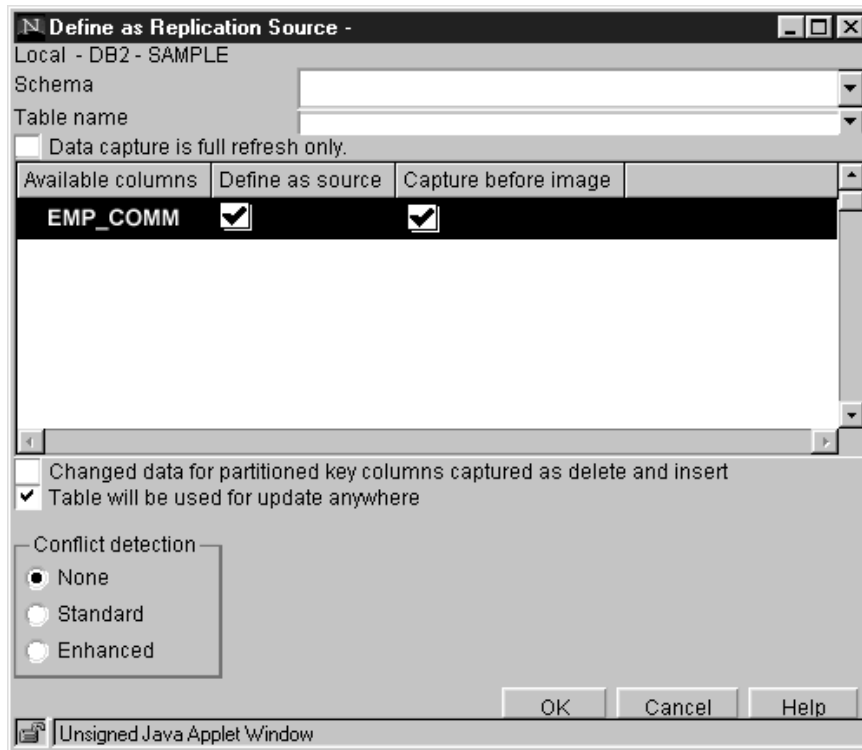


Figure 18. The Define as Replication Source window. The Define as Replication Source window is used to define replication sources.

4. If you want to specify that the Capture program will not capture changes to the source table and that the source data will be available for refresh copying, select the **Data capture is full refresh only** check box.
5. If you want to make a column unavailable for replication, clear the **Define as source** check box for that column. When cleared, the column name becomes unavailable as a source column while defining a replication subscription and cannot be replicated.
Define as source is automatically selected if the **Table will be used for update anywhere** check box is selected.
6. If you do not want to capture the before image of a column, clear the **Capture before image** check box for that column. The after image is always replicated.
This check box is selected automatically when the **Table will be used for update anywhere** check box is selected.
7. If you want support for logical partition keys, select the **Changed data for partitioned key columns captured as delete and insert** check box. The Capture program treats updates to primary key columns as INSERT and DELETE statements, rather than UPDATE statements.

For before- and after-image columns, updates are turned into deletes of the before-image column values, followed in sequence by inserts of the after-image values. Within the context of the generated DELETE change row, all before-image and after-image columns have the before-image values of the update. Within the context of the generated INSERT change row, all after-image columns have the after-image values of the update.

8. If the source table might be used for update-anywhere replication:

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

When you select this check box, the **Conflict detection** choices are enabled. If you do not select the **Table will be used for update anywhere** check box, the conflict detection level default is None.

- b. Select a level of conflict detection.

None Select **None** if you do not want any conflict detection.

Attention: Conflicting updates between the source table and the replica will *not* be detected.

Standard Select **Standard** if you want 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 Select **Enhanced** for 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 have been captured.

See “Conflict Detection” on page 74 for more information about conflict detection.

9. Select **OK** to save the values and close the window. The Run Now or Save SQL window opens. Specify whether to submit the replication request now or save the SQL statements generated by the action in a file to be run later. See “Customizing and Running Replication SQL Files” on page 90 to learn how to submit the request now or later.
10. After you submit the request, verify that the replication source was created by selecting the **Replication Sources** folder in the object tree and selecting the **Refresh** menu choice.
11. To begin capturing changes to the source table, start or reinitialize the Capture program. See the Capture and Apply chapter for your platform in this book for more information about the Capture program.

Defining an External CCD Table as a Replication Source

Changes captured within applications or other system tools, such as DataPropagator NonRelational, can also be defined as sources for replication subscription. The external data source must provide a complete CCD table, which you can define as a replication

source with the Control Center and use as a surrogate source table. You can create the CCD table outside of the replication tools, have your application populate it, and then define the table as a replication source. See “External Data Sources” on page 65 for more information about external data sources.

To define a CCD table as an external replication source:

1. Create and populate the CCD table with the user data columns in CCD table format, to hold changes that originated from the external data source. See Chapter 19, “Table Structures” on page 295 for more information about CCD table format.
2. Define the CCD table as a replication source. See “Defining Replication Sources” on page 93.

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 to be copied to a target table.

You cannot define an existing view as a replication source.

To define a join:

1. Define as replication sources the source tables to be used in the join. See “Defining Replication Sources” on page 93.
2. From the Control Center object tree, click on the **Replication Sources** folder and select the replication sources to be used in the join from the contents pane.
3. Click mouse button 1 and select **Define join** from the pop-up menu. The Define Join window opens, as shown in Figure 19 on page 99.

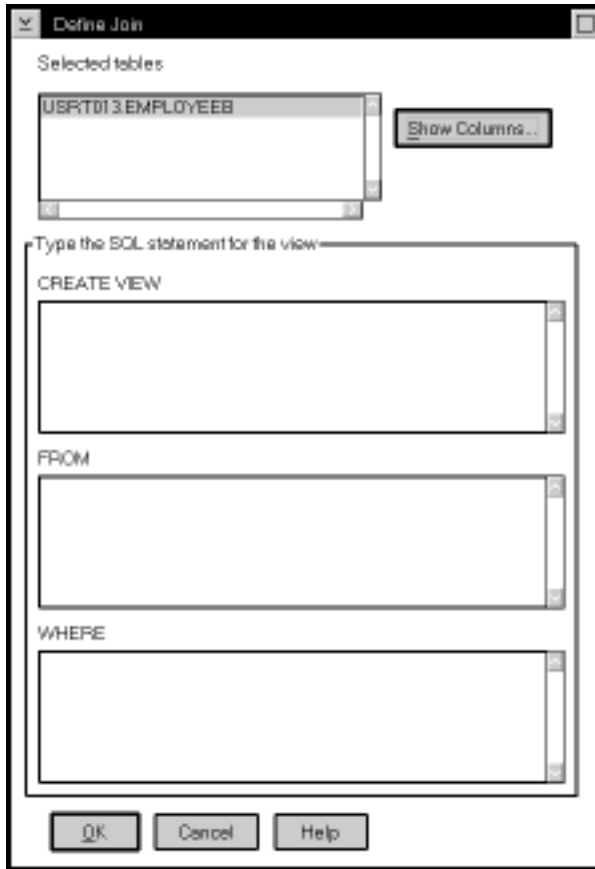


Figure 19. The Define Join Window. You can define joins as replication sources.

4. To view a list of columns available for the join:
 - a. Select a table name from the **Selected tables** box.
 - b. Select **Show Columns**. The Show Columns window opens and shows all of the columns in the selected table.
 - c. Select **OK** to close the window and continue defining the join.

5. 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
```

This field can contain ordinary or delimited identifiers. See “Ordinary and Delimited Identifiers” on page 92 for more information.

Do not type the words **CREATE VIEW**. This part of the statement is automatically supplied during processing.

6. In the **FROM** field, type the SQL statement for the view. For example:

```
TABLEA A, TABLEB B
```

This field can contain ordinary or delimited identifiers. See “Ordinary and Delimited Identifiers” on page 92 for more information.

Do not type the word FROM. This part of the statement is automatically supplied during processing.

7. If you want to use a row predicate, type the WHERE clause SQL statement in the **WHERE** field. For example:

```
A.COL1=B.COL1
```

This field can contain ordinary or delimited identifiers. See “Ordinary and Delimited Identifiers” on page 92 for more information.

Do not type word WHERE. This part of the statement is automatically supplied during processing.

8. Select **OK** to save the values and close the window. The Run Now or Save SQL window opens. Specify whether to submit the replication request now or save the SQL statements generated by the action in a file to be run later. See “Customizing and Running Replication SQL Files” on page 90 to learn how to submit the request now or later.
9. After you submit the request, verify that the replication source was created by selecting the **Replication Sources** folder in the object tree and selecting the **Refresh** menu choice.

Viewing or Changing Existing Replication Sources

You can view an existing replication source or, 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.

To view or change the replication source:

1. If you plan to change the replication source definition, stop or suspend the Capture program. See the Capture and Apply chapter for your platform in this book to learn more about the Capture program.
2. Select the **Replication Sources** folder. The replication sources appear in the contents pane.
3. Select the replication source object you want to view or change from the contents pane and select the **Change** menu choice from the pop-up menu. The Change Replication Source window opens.
4. If the table was made available for update-anywhere replication and you want to change the **Conflict detection** choices, select a radio button for the change.
5. Select **Cancel** if you have not changed the replication source definition. Select **OK** to save the values and close the window. The Run Now or Save SQL window opens. Specify whether to submit the replication request now or save the SQL statements generated by the action in a file to be run later. See “Customizing and

Running Replication SQL Files” on page 90 to learn how to submit the request now or later.

6. To begin capturing changes for the changed replication source, start or reinitialize the Capture program. See the Capture and Apply chapter for your platform in this book to learn more about the Capture program.

Notes for changing the conflict level detection:

If you want to be able to change the conflict detection level, you must have done the following while defining the replication source:

- Selected the **Define as source** check boxes selected for all columns
- Selected the **Capture before image** check boxes for all columns
- Optionally selected the **Table will be used for update anywhere replication** check boxes

Removing Replication Sources

When you no longer need a replication source, you can remove the object from the Control Center and its control information from the control tables.

Attention:

1. Do not delete a replication source while the Capture program is running.
2. Although the Control Center removes dependent subscriptions, it is recommended that you check whether a dependent subscription table is being used as a source for another subscription and that you cancel any such dependent subscription before you delete a replication source.

The Control Center drops the replication source table space if it is empty. You can change this default behavior on the Replication page of the Tools Settings notebook so that the table space is not dropped.

To remove a replication source:

1. Stop the Capture program. See the Capture and Apply chapter for your platform in this book to learn how to stop the Capture program.

Attention: Do not use the SUSPEND command.

2. Select the **Replication Sources** folder. The replication sources appear in the contents pane.
3. Select one or more replication source objects that you want to delete and select **Remove** from the pop-up menu.

A confirmation window prompts you to confirm that you want to remove the replication sources.

4. Select **Yes** in the confirmation window to delete the replication sources. The Run Now or Save SQL window opens. Specify whether to submit the replication request

now or save the SQL statements generated by the action in a file to be run later. See “Customizing and Running Replication SQL Files” on page 90 to learn how to submit the request now or later.

Attention: Before running the deferred remove request, first stop the Capture program.

5. After you submit the request, verify that the replication sources were deleted by selecting the **Replication Sources** folder in the object tree and selecting the **Refresh** menu choice.
6. Restart the Capture program. See the Capture and Apply chapter for your platform in this book to learn how to restart the Capture program.

Defining Replication Subscriptions for Update Anywhere

See “Defining Replication Sources and Replication Subscriptions for Update Anywhere” on page 125 for information about how to define replication subscriptions for update-anywhere replication.

Chapter 9. Working with Replication Targets

The following section describes replication subscriptions and how to work with them in your replication environment.

What Are Replication Subscriptions?

A replication subscription defines the relationship between one or more source and target tables, the structure of a target table, and the way the Apply program replicates data to the target table: when, how often, and which enhancements. Each definition is called a member and also defines the type of target table to be created, such as a user copy table or a replica. Data is replicated from all replication source tables to all target tables in the same processing cycle, and all data is replicated from all replica target tables to the replication source table in the same processing cycle.

The replication subscription logically groups one or more definitions that are somehow related; for example, source tables that have RI constraints. For the related tables, the changed data needs to be copied to the target tables at the same time, or in the same subscription interval. See “Data Consistency Requirements” on page 70 for more information about replication subscriptions and RI constraints.

Defining Replication Subscriptions

Use the Define Subscription window to set up sources and targets to define a basic replication subscription. See “Navigating to Your Replication Objects with the Control Center” on page 84 to learn how to navigate to the **Replication Sources** folder. Many of the fields in the Define Subscription window accept both ordinary and delimited identifiers. These fields are identified in the step. See “Ordinary and Delimited Identifiers” on page 92 for more information.

To define a replication subscription:

1. Select the **Replication Sources** folder from the Control Center as shown in Figure 20 on page 104. The replication sources appear in the contents pane.



Figure 20. The Replication Sources Folder with Objects and Actions

2. Select one or more replication sources you want to define as sources for the replication subscription, and select **Define subscription** from the pop-up menu. The Define Subscription window opens, as shown in Figure 21 on page 105.

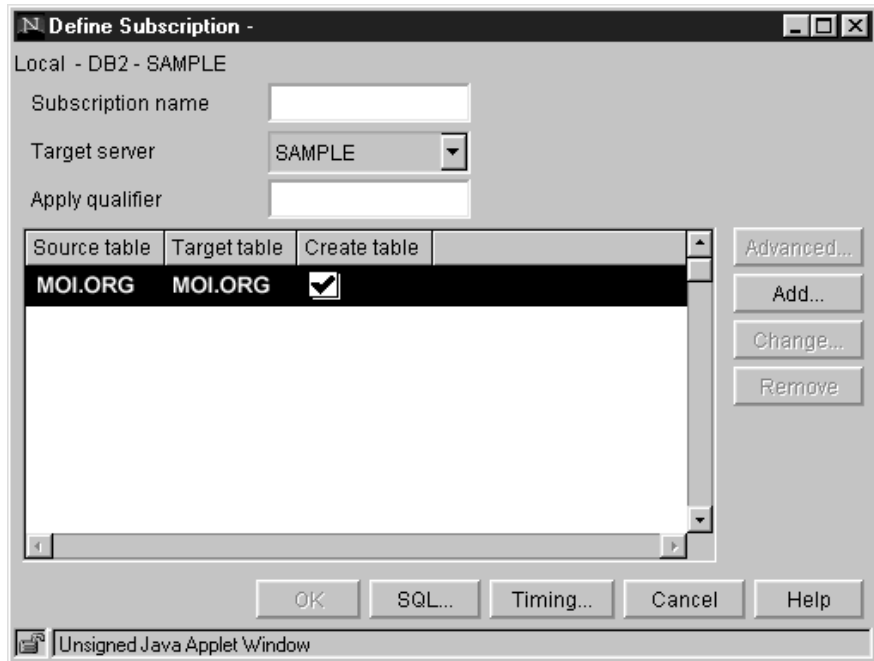


Figure 21. The Define Subscription Window

3. Type a set name for the replication subscription in the **Subscription name** field. The set name can be up to 18 characters and can be an ordinary or delimited identifier; for example, MYSUBSCRIPTION.

Note: The name of a subscription set consists of three parts: the control server (CONTROL_SERVER), the apply qualifier (APPLY_QUAL), and the set name (SET_NAME). The **Subscription name** field is used to assign the set name.

4. Specify the database where the target tables will reside in the **Target server** field. The displayed default is the alias of the source server. This value can be an ordinary or delimited identifier; for example, MYDBASE.
5. Specify a character string associating this replication subscription with a unique instance of the Apply program in the **Apply qualifier** field. This value can be an ordinary or delimited identifier and must be in uppercase; for example, MYAPPLY.

Note: The value entered in the **Apply qualifier** field will need to be used when the Apply program is invoked.

6. If you want to change the name of the target table, click on the name in the **Target table** field and then click on **Change**. In the Change dialog window, enter the target table name. You can repeat this step for each target table in the list box. This value can be an ordinary or delimited identifier.
7. Specify whether to create the target table by selecting the **Create table** check box for every target table that you want to create when you subscribe.

See “Defining a Replication Subscription with a User-Defined Target Table” on page 120 for more information about this option.

To have the target tables created automatically, select the **View-> Tools Setting** menu choice from the Control Center. In the **Tools Setting** notebook, select the **Replication** page, and select the **Create target tables** check box.

8. If you want to customize the target table structure, enhance any data, or specify the timing for the replication subscription, go to “Defining Replication Sources: Advanced Tasks” on page 107.
9. Complete the subscription definition.
 - a. Click on **OK** to save the values and close the window. The Subscription Information window opens as shown in Figure 22.

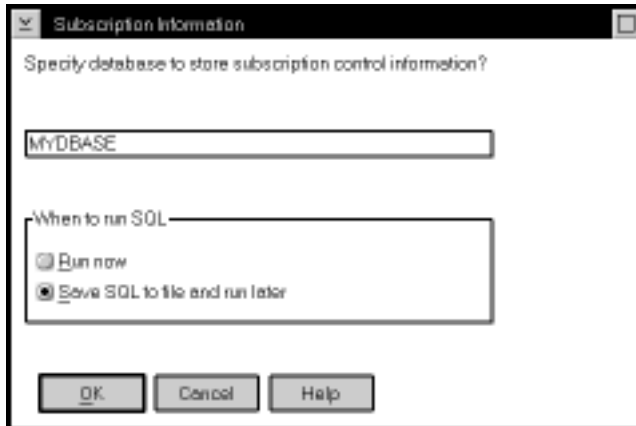


Figure 22. The Subscription Information Window. Use this window to enter the control server name and specify whether to run now or save the SQL to a file

- b. To specify the control server database, enter the database name in the **Specify database to store subscription control information** field.

Attention: You can specify an Apply qualifier for the control server while defining the first replication subscription that uses the control server. This Apply qualifier will remain associated with the control server until the last subscription with the Apply qualifier is removed. See “The Control Server” on page 17 for more information about the Apply qualifier and the control server and how to identify them.

- c. Specify whether to submit the replication request now or save the SQL statements generated by the action in a file to be run later.
10. After you submit the request, verify that the replication subscription was created by selecting the **Replication Subscription** folder in the object tree and selecting the **Refresh** menu choice.
11. If you defined an event to start the Apply program, populate the event table. See the “Scheduling Subscriptions with the Event Table” section in the Capture and Apply chapter for your platform in this book.

- To begin replicating data to the target tables, start the Apply program using the name of the control server you specified on the Subscription Information window. See the Capture and Apply chapter for your platform to learn more about the Apply program.

Defining Replication Sources: Advanced Tasks

The following sections describe optional, advanced tasks for defining the replication subscription.

Choosing a Target Table Type

You can specify a specific target table type if you do not want the default target type of user copy. Your application might require a target table that provides a history of changes to the data or an aggregation of columns in the source table. See “Data Transformation Requirements” on page 53 for more information about determining what type of table you should select based on your application's requirements.

Use the Target Type page of the Advanced Subscription Definition notebook to select a target type. Figure 23 shows the Target Type page.

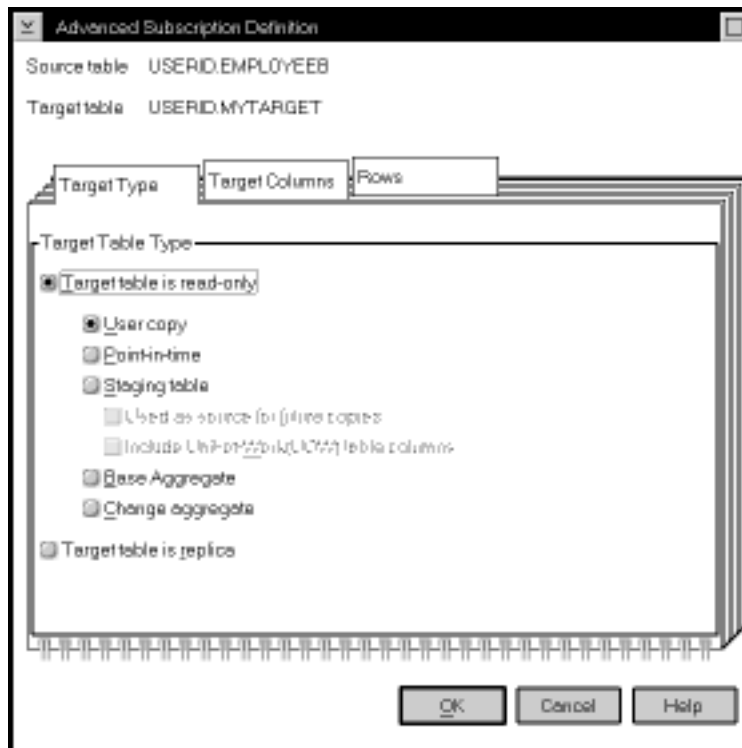


Figure 23. The Target Type Page of the Advanced Subscription Definition Notebook

To specify a target table type:

1. From the Subscription Definition window, select a source and target table combination.
2. Click on **Advanced**. The Advanced Subscription Definition notebook opens.
3. 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 data exactly at the time of the copy.

Point in time

A target table that matches the source table, with a timestamp column added.

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 base table.

Staging

Also known as a CCD table. Usually, a CCD table is a target table that is a join of the source table's change data table and unit-of-work tables. However, it may be created directly by an application program or by another product. A staging table can be an internal CCD table or a local or remote CCD table.

- For updateable target tables, select **Target table is replica**, an updateable target table that is used to replicate data back to the source table or to other target tables.

4. 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.
5. Complete the subscription. See step 9 on page 106.

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. Or, you might need to rename or create new columns for the target table. For data enhancement, performance, security, or other reasons, you can subset the target table vertically (delete, create new, or rename columns) or horizontally (enter an SQL predicate) using the advanced features of the Define Subscription window.

Defining the Target Table Columns

The following section describes how to define the target table columns that are copied to the target table. For data restrictions when defining replication targets, see “Data Restrictions” on page 67.

Attention: Replica target tables must contain the same columns as the source table: they cannot be subsetted, have columns added, or have columns renamed.

To define the target table columns:

1. From the Define Subscription window, select a source and target combination.
2. Click on **Advanced** to open the Advanced Subscription Definition notebook.
3. Open the Target Columns page (Figure 24).

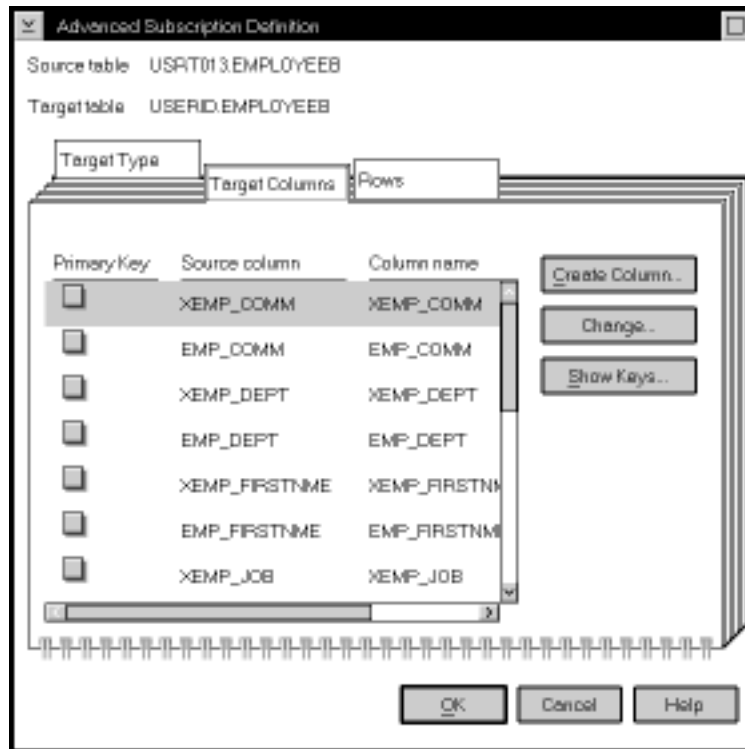


Figure 24. The Target Columns Page of the Advanced Subscription Definition Notebook. Use this page to subset, create new, and rename columns for the target table.

4. If you want to specify a column as a primary key column for the target table, click on the **Primary Key** check boxes next to the column name.

Warning: You are required to select one or more columns as part of a primary key if you use one of the following target table types: user copy, point-in-time, replica, or condensed staging tables. If you do not select columns for the primary key, then the primary key definition is carried over from the primary key definition of the source table. However, if the source table does not have a primary key definition, an Apply program runtime error will occur.

5. If you want to rename a column, select the column name that you want to edit, and type over the existing column name. This value can be up to 18 characters and can be an ordinary or delimited identifier.
6. If you want to change a column definition for the target table:
 - a. Select **Change**. The Change Column window opens. It has the same fields as the Create Column window, shown in Figure 25 on page 111.
 - b. Optional: Change the name of the column in the **Column name** field. The name can be up to 18 characters. This value can be an ordinary or delimited identifier; for example, CUST_COUNT.
 - c. Type the SQL expression to change the definition of the column. For example: COUNT(*).

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 your database SQL reference for information on valid SQL expressions. Invalid SQL expressions cause an SQL error when the subscription is processed by the Apply program.
 - d. Optional: To see examples of SQL expressions, click on **Examples**.
 - e. Click on **OK** to close the window.
7. If you want to remove a column from the target table, clear the **Subscribe** check box next to the column name.
8. If you want to create a new computed column or use aggregation for the target table:
 - a. Click on **Create Column**. The Create Column window opens, as shown in Figure 25 on page 111.

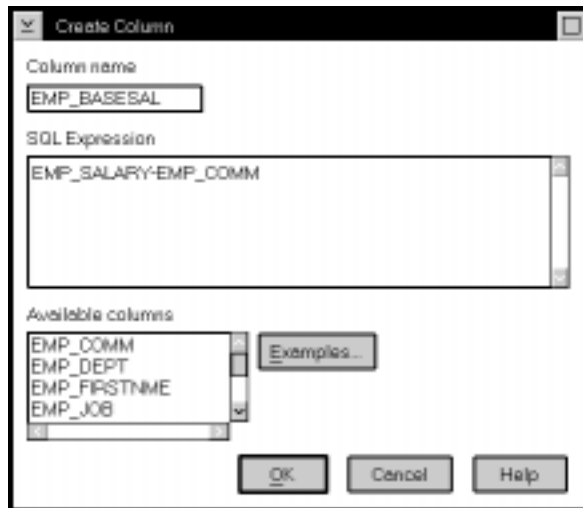


Figure 25. The Create Column Window. You can define new columns for the target table.

- b. Type the name of the column in the **Column name** field. The name can be up to 18 characters and can be an ordinary or delimited identifier.
- c. Type the SQL expression defining the new column.

The expression can contain up to 254 characters, can be any valid SQL expression, and 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 your database SQL reference for information on valid SQL expressions. Invalid SQL expressions cause an SQL error when the subscription is processed by the Apply program.

- d. Optional: To see examples of SQL expressions, click on **Examples**.
- e. Click on **OK** to close the window.

9. If you are finished using the Advanced Subscription Definition notebook, click on **OK** to close the notebook. Otherwise, use the Rows page to define the target rows as needed.
10. Complete the subscription. See step 9 on page 106.

Defining the Target Table Rows

The following section describes how to define the target table rows that are copied to the target table and the restrictions for doing so.

Row Predicate Restrictions:

- You can use only column names from the **Target columns** list on the Target Columns page; these names are also listed in the **Available columns** list box.
- Do not use before-image columns or computed columns.

Before-image columns are supported in change data tables but not user tables. Do not use these columns and predicates without detailed knowledge of the register control table values. (If you use these columns, full refresh is no longer available.)

- If you created computed columns on the Target Columns page, you must provide a GROUP BY clause. A computed column is an SQL expression which includes a function that summarizes data. Examples are SUM(WITHDRAWALS), COUNT(*), and AVG(BALANCE). Both base and change aggregate target tables must have a GROUP BY clause.
- Do not type WHERE in the clause; it is implied. Type WHERE in the clause for subselect statements.
- Do not end the clause with a semicolon (;).
- 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 the GROUP BY clause on the Rows page, even though the before-image columns are not displayed in the **Source columns** box.
- You must provide a dummy WHERE clause when both of the following conditions are true:
 - You are creating an aggregate column that requires a GROUP BY clause.
 - You do not use any other predicate in the **WHERE** field.

You can receive Apply program run-time errors if you do not provide the dummy WHERE clause in this situation.

To define the target table rows:

1. From the Define Subscription window, select a source and target combination.
2. Click on **Advanced** to open the Advanced Subscription Definition notebook.
3. Open the Rows page (Figure 26 on page 113).



Figure 26. The Rows Page of the Advanced Subscription Definition Notebook. You can subset the rows for the target table by typing in a WHERE clause.

4. To specify which rows are copied to the target table, enter an SQL predicate in the **WHERE** field. The predicate can contain ordinary or delimited identifiers. See your database's SQL reference for more information about WHERE clauses.

Attention: Do not type the word WHERE. It is automatically supplied during processing.
5. To see examples of SQL predicates, click on **Examples**.
6. If you are finished using the Advanced Subscription Definition notebook, click on **OK** to close the notebook.
7. Complete the subscription. See step 9 on page 106.

WHERE clause examples:

The following examples contain 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. You can also click on **Examples** on the Rows page to link to additional examples in the online help. See your database's SQL reference for more information about WHERE clauses.

- 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, assuming that the EMPID column is defined as NOT NULL, use a WHERE clause like the following:

```
EMPID IS NULL
```

Setting the Copying Schedule: Time or Event Based

You can set the copying schedule when you define a replication subscription. You can use relative or event timing. See “Data Currency Requirements” on page 69 to determine what kind of timing to use.

Set the timing of the replication subscription from the Subscription Timing notebook, accessed by clicking on **Timing** from the Subscription Definition window. The following figure shows the Subscription Timing notebook. There are two timing pages. Both pages have same fields.

Source to Target

The timing information for replicating changed data from the replication source tables to the target tables.

Replica to Source

The timing information for replicating changed data from replica tables to the replication source tables. Complete this page only for replication subscriptions with replica tables.

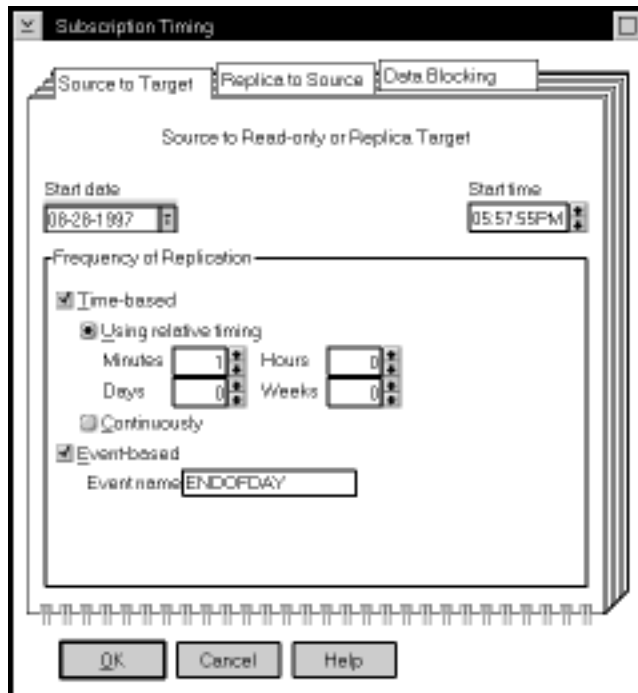


Figure 27. The Subscription Timing Notebook. You can specify relative or event timing for the replication subscription.

To specify a time-based frequency:

1. From the Define Subscription window, click on **Timing**. The Source to Target page of the Subscription Timing notebook opens.
2. Specify the date on which you want the Apply program to start replicating the replication subscription in the **Start date** field.
3. Specify the time that you want the replication to start in the **Start time** field by typing over the existing characters or using the spin buttons.
4. Select the **Time-based** check box.
5. Choose the type of interval that you want by selecting one of the following radio buttons:
 - Using relative timing** Specifies a specific interval, in minutes, hours, days, or weeks. Use the radio dials to specify the interval that you want.
 - Continuously** Specifies to replicate continuously.
6. If you are finished with this notebook, click on **OK**. Otherwise, click on one of the other notebook tabs to specify replica to source timing or data blocking for the replication subscription.
7. Complete the subscription. See step 9 on page 106.

To specify an event-based frequency:

1. From the Define Subscription window, click on **Timing**. The Source to Target page of the Subscription Timing notebook opens.
2. Click on the **Event-based** check box.
3. In the **Event** field, type the event name with which you will populate the event table to trigger replication. This value can be an ordinary or delimited identifier.
4. If you are finished with this notebook, click on **OK** to close the notebook. Otherwise, click on one of the other notebook tabs to specify replica to source timing or data blocking information for the replication subscription.
5. Complete the subscription. See step 9 on page 106.

See the "Scheduling Subscriptions with the Event Table" section in the Capture and Apply chapter for your platform in this book.

Specifying Mini-Cycles for the Apply Program to Copy Committed Data

You can specify the number of minutes of data that the Apply program copies at a time. This value helps the Apply program break down a large block of changed data into smaller subscription cycles, preventing spill file or log overflows. See "Data Blocking for Large Volumes of Changes" on page 44 for information on how to determine this value.

To specify a data blocking value:

1. From the Define Subscription window, click on **Timing**. The Subscription Timing notebook opens.
2. Open the Data Blocking page as shown in Figure 28 on page 117.

You can specify the number of minutes of data copied at a time.

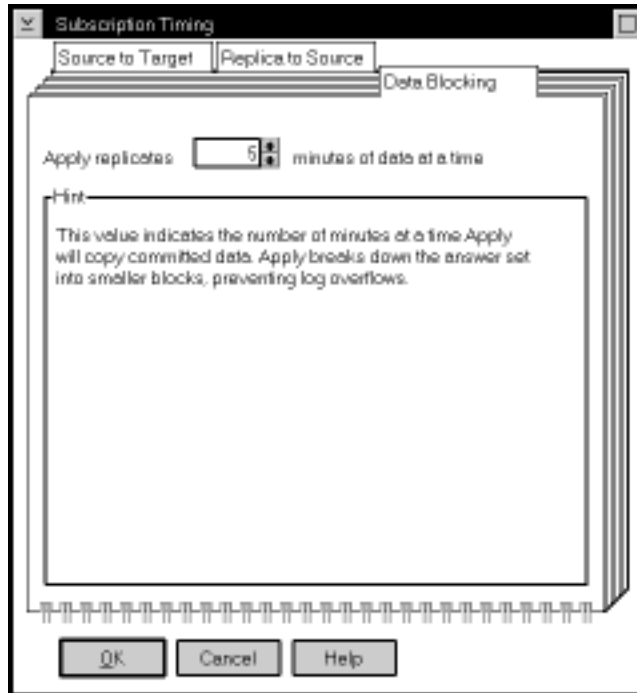


Figure 28. The Data Blocking Page of the Subscription Timing Notebook

3. Enter the interval value of the number of minutes that you want the Apply program to copy data at a time.

Note: An interval value of 0 means that data blocking is disabled. The complete answer set will be fetched and applied.

4. If you are finished using the Advanced Subscription notebook, click on **OK** to close the notebook.
5. Complete the subscription. See step 9 on page 106.

Defining SQL Statements or CALL Procedures for the Replication Subscription

You can define SQL statements or CALL procedures to be run before or after the Apply program copies the data from the source to the target table.

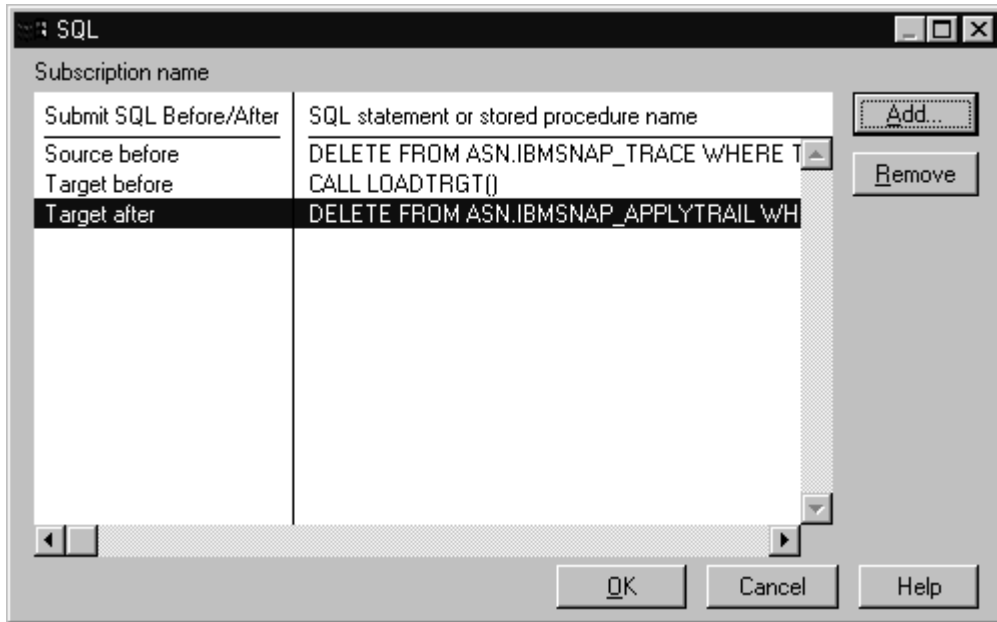


Figure 29. The SQL Window. You can add or remove SQL or CALL procedures for processing at the source server before or at the target server before or after the replication subscription is processed.

To specify SQL statements or CALL procedures statements for the replication subscription:

1. From the Define Subscription window, click on **SQL**. The SQL window opens (Figure 29).

Use the SQL window to add or remove SQL statements or CALL procedures that are submitted at the target or source server either before or after the replication subscription is processed. The statements are processed in the order they appear in the list.

2. Click on **Add**. The Add SQL window opens, as shown in Figure 30 on page 119.



Figure 30. The Add SQL Window. You can add SQL statements or CALL procedures.

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 need to enter a valid SQLSTATE that the Apply program needs to pass, type a valid 5-byte SQLSTATE value in the **SQLSTATE** field and click on **Add**. The value is added to the **Acceptable SQLSTATE values** box. You can enter up to 10 values. The Apply program interprets these values as successful execution.
5. Specify whether you want to submit the SQL statement or CALL procedure at the target or source server before the replication subscription is processed, or at the target server after the replication subscription is processed by clicking on the appropriate radio button in the **Submit SQL statement** field.
6. Click on **OK**. The statement is added to the box in the SQL window and the Add SQL window closes.

7. Repeat for each SQL statement or CALL procedure.
8. Click on **OK** to return to the Define Subscription window.
9. Complete the subscription. See 9 on page 106.

To remove an SQL statement for CALL procedure:

1. From the Define Subscription window, click on **SQL**. The SQL window opens.
2. Select the statement you want to remove.
3. Click on **Remove**.
4. Click on **OK** to close the window.
5. Complete the subscription. See step 9 on page 106.

Defining a Replication Subscription with a User-Defined Target Table

The Control Center allows you to use a previously defined DB2 table as the target table in a replication subscription. This type of target table is known as a user-defined target table. With this feature, you can define a replication subscription with a target table that is defined outside of replication administration in the Control Center.

The following sections explain how to define a user-defined target table, the restrictions, and maintaining the target table.

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

1. Refer to Chapter 19, “Table Structures” on page 295 to determine the structure for the target table type. For example, if you are defining a subscription for a base aggregate target table, refer to the table structure definition for base aggregate target tables.
2. Alter the target table to add any required columns, such as timestamp columns.
3. Create a unique index for condensed tables (point-in-time or condensed CCD tables). Define the index using the SQL statements provided in Chapter 19, “Table Structures” on page 295.
4. Define the subscription to match the user-defined target table structure. This task includes creating new columns, subsetting columns, changing column names, or renaming before-image columns to include the before-image column prefix.

On the Define Subscription window:

- a. Clear the **Create table** check boxes for 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 on **Advanced** to open the Advanced Subscription Definition notebook.

- 1) If you want to select an alternate table type, see “Choosing a Target Table Type” on page 107.
- 2) If you want to modify the target table columns to match the user-defined target table, see “Defining the Target Table Structure: Columns and Rows” on page 108.
- 3) If you want to subselect the rows or use an aggregate expression, see “Defining the Target Table Structure: Columns and Rows” on page 108.

Restrictions

The following conditions cause run-time errors:

- The subscription definition contains fewer columns than exist in the user-defined target table.
- The new columns in the subscription definition do not allow nulls or have defined default values.
- The target table and unique index do not exist when you run the Apply program.

Maintaining a User-Defined Target Table

When you use a user-defined target table during subscription, you assume the responsibilities listed below. The Control Center does not check for inconsistencies between the subscription definition and a user-defined target table. You must complete the following activities:

- When running the Apply program:
 - Ensure that there is a target table that matches the subscription definition.
 - Ensure that there is a unique index for point-in-time, condensed CCD, user copy, and replica target tables.
- Debug any inconsistencies that exist between the target table and the subscription definition.
- For a user-defined CCD target table, define the subscription to match the attributes specified for the target table by specifying:
 - The target table type as a staging table
 - Whether the table has one or multiple changes per key

For CCD target tables: When a subscription is to be defined on a base table having a user-defined internal CCD, ensure that the internal CCD table was created to match the subscription definition.

Because an internal CCD table replaces the join of the CD and UOW tables as the source for updates to the target table, ensure that the CCD table exists before attempting to subscribe to the base table again. If the internal CCD target table is not created, the source table will not be available for another subscription to the base table.

Activating and Deactivating Replication Subscriptions

You can control the active status of a replication subscription. This feature is useful when you want to temporarily deactivate a replication subscription without removing it. When you deactivate a replication subscription, the Apply program completes its current processing cycle and then stops.

To deactivate a replication subscription:

1. Select the **Replication Subscriptions** folder from the object tree. The replication subscriptions appear in the contents pane.
2. Verify that the replication subscription is active by looking at the status icon to the left of the replication subscription object that you want to deactivate. Figure 31 shows activated and deactivated replication subscriptions.

Icon	Name	Apply Identifier	Control Server
	MYSUB1	MYAPPLY	MYDBASE
	MYSUB2	MYAPPLY	MYDBASE
	MYSUB3	MYAPPLY	MYDBASE
	MYSUB4	MYAPPLY	MYDBASE

Figure 31. The active and inactive icons. The active and inactive icons show which replication subscriptions have been activated and deactivated.

The first two are active, the second two are inactive.

If the icon for the replication subscription that you are working with shows that the replication subscription is active, continue to the next step. If the icon is inactive, do not continue with this task because the replication subscription is already inactive.

3. In the contents pane, select the replication subscription object that you want to deactivate and select **Activate/Deactivate** from the pop-up menu.

The replication subscription is deactivated, and the status icon is updated to reflect the change.

To reactivate a deactivated replication subscription:

1. Select the **Replication Subscriptions** folder from the object tree. The replication subscriptions appear in the contents pane.
2. Verify that the replication subscription is inactive by looking at the status icon to the left of the replication subscription object that you want to activate. If the icon shows that the replication subscription is inactive, continue to the next step. If the icon is active, do not continue with this task because the replication subscription is already active.
3. In the contents pane, select the replication subscription object that you want to reactivate and select **Activate/Deactivate** from the pop-up menu.

The replication subscription is reactivated and the status icon is updated to reflect the change.

Cloning a Replication Subscription to Another Server

You can clone a replication subscription to another server. Cloning creates an exact copy of an existing replication subscription on a different target server, using a different Apply qualifier. You can clone one or more replication subscriptions at a time. The Control Center updates the control tables at the control server.

To clone a replication subscription:

1. Select the **Replication Subscriptions** folder from the object tree. The replication subscriptions appear in the contents pane.
2. In the contents pane, select the replication subscription object that you want to modify and select **Clone** from the pop-up menu. The Clone window opens.
3. Specify a new Apply qualifier in the **Apply qualifier** field. This value can be an ordinary or delimited identifier.
4. Specify the new target server in the **Target server** field. This value can be an ordinary or delimited identifier.
5. Click on **OK** to save the values and close the window. The Subscription Information window opens.
6. To specify the control server database, enter the database name in the **Specify database to store subscription control information** field.
7. Specify whether to submit the replication request now or save the SQL statements generated by the action in a file to be run later. See “Customizing and Running Replication SQL Files” on page 90 to learn how to submit the request now or later.
8. After you submit the request, verify that the replication subscription was cloned by selecting the **Replication Subscription** folder in the object tree and selecting the **Refresh** menu choice. Check the Apply qualifier and target server name for the replication subscription.

Viewing or Changing an Existing Replication Subscription

You can change a subset of the replication subscription values, primarily those that do not affect the structure of the target tables.

To change the replication subscription:

1. Select the **Replication Subscriptions** folder from the object tree. The replication subscriptions appear in the contents pane.
2. In the contents pane, select the replication subscription object that you want to modify and select **Change** from the pop-up menu. The Change Subscription Definition window opens.

3. You can change the following values in the Change 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 predicate is used starting with the next subscription cycle for the replication subscription. See “Defining the Target Table Rows” on page 111 for more information and examples.
 - The SQL or CALL procedure for before or after copying in the SQL window. See “Defining SQL Statements or CALL Procedures for the Replication Subscription” on page 117 for more information.
 - The timing values in the Subscription Timing notebook. See “Setting the Copying Schedule: Time or Event Based” on page 114 for more information.
 - The data blocking value in the Subscription Timing notebook. See “Specifying Mini-Cycles for the Apply Program to Copy Committed Data” on page 116 for more information.
4. Click on **OK** to save the values and close the window. The Run Now or Save SQL window opens. Specify whether to submit the replication request now or save the SQL statements generated by the action in a file to be run later. See “Customizing and Running Replication SQL Files” on page 90 to learn how to submit the request now or later.

Removing Replication Subscriptions

When you no longer need a replication subscription definition, or the requirements have changed enough that you need to redefine it, you can remove it. Removing a replication subscription definition deletes information about it from the control tables and deletes the target table from the target server. If you have selected to drop table spaces in the Tools Settings notebook, the table space is also dropped.

You can remove replication subscription definitions that are dependent on replication sources that you are removing. (If you remove a replication source which has dependent replication subscriptions, the dependent subscriptions will be removed automatically.)

To remove a replication subscription:

1. Select one or more replication subscription objects from the contents pane and select **Remove** from the pop-up menu. The Remove Subscription Confirmation window opens.
2. Click on **Yes** to save the values and close the window. The Run Now or Save SQL window opens. Specify whether to submit the replication request now or save the SQL statements generated by the action in a file to be run later. See “Customizing and Running Replication SQL Files” on page 90 to learn how to submit the request now or later.
3. After you submit the request, verify that the replication subscription was removed by selecting the **Replication Subscriptions** folder in the object tree and selecting the **Refresh** menu choice.

Defining Replication Sources and Replication Subscriptions for Update Anywhere

The following steps describe how to set up the sources and targets for update-anywhere replication. They do not list all the possible steps for defining the source and subscription; just those of particular interest when administering an update-anywhere scenario.

To define a replication source and subscription for update anywhere:

1. Define a custom replication source, using the following selections. See “Defining Replication Sources” on page 93 to learn how to define a replication source.
 - a. Ensure that the **Define as Source** check boxes is selected for every column.
 - b. Ensure that the **Capture before image** check boxes is selected for every column.
 - c. Click on the **Table will be used for update anywhere** check box.
 - d. Select a conflict detection level.
2. Define a replication subscription set using the following selections. See “Defining Replication Subscriptions” on page 103 for information about the steps for this task.
 - a. Select the replication sources to be in the subscription set. Include all sources affected by the replica tables being updated.
 - b. From the Subscription Definition window, select a target table to be defined as a replica, an updateable target table.
 - c. Click on **Advanced** to open the Advanced Subscription notebook. The following sections are required on the Advanced Subscription notebook:
 - 1) From the Target Table page, click on **Target table is replica**.
 - 2) From the Target Columns page:
 - a) Ensure that the **Subscribe** check boxes are selected for every column. *Do not* create new columns for the replica table.
 - b) Specify a primary key for the replica table by clicking on the **Primary Key** check boxes next to the key column names.

Note: Make the primary key the same as the source table primary key to prevent conflicts.
 - 3) If you want the replica to be a subset of the source table, from the Rows page, type a row predicate in the **WHERE** field.
 - 4) Click on **OK** to close the Advanced Subscription notebook.
 - d. Repeat the Advanced Subscription notebook steps for each target table.
 - e. Click on **Timing** to open the Subscription Timing notebook.
 - 1) On the Source to Target page, fill in the subscription set timing information for copying the source tables changed data to the target tables.

- | 2) On the Replica to Target page, fill in the subscription set timing information for copying the replica tables changed data to the source tables.
- | 3) Click on **OK** to close the notebook.
- | f. If you want to define SQL or CALL procedures to run before or after the subscription set is processed, click on **SQL** and define the processing statements.
- |

Part 4. IBM Replication Capture and Apply

This part describes how to configure, operate, and troubleshoot the Capture and Apply programs. Each chapter describes the Capture and Apply programs for a particular platform.

Chapter 10. Capture and Apply for MVS

This chapter describes how to set up, operate, and troubleshoot the Capture and Apply programs for MVS.

Read the following sections before reading the sections on operating Capture for MVS:

- “Setting Up the Capture and Apply Programs”
- “Specifying Tuning Parameters for the Capture Program” on page 130
- “Restrictions When Running the Capture Program” on page 131
- “Authorization for Running the Capture Program” on page 132
- “Recovering from Severe Errors” on page 132

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 MVS and Apply for MVS are packaged in SMP/E format. The installation sequence for each program consists of:

- Customizing invocation JCL to suit your environment
- Using SMP/E to install
- Providing APF authorization
- Creating and loading the VSAM messages file
- Binding to the DB2 subsystem
- Invoking

Capture for MVS uses the DB2 for MVS/ESA Instrumentation Facility Interface (IFI) to retrieve log records. Make sure to apply the correct DB2 maintenance before installing the Capture program. The correct DB2 maintenance consists of the set of DB2 maintenance instructions in the program directory of the Capture program, and the current DB2 maintenance recommended by IBM Service.

See “Installation Steps for Capture for MVS” in the Capture for MVS program directory for instructions about installing the Capture for MVS program. See “Installation Steps for Apply for MVS” in the Apply for MVS program directory for instructions about installing the Apply for MVS program. System programmers should read all of the steps in the program directory through step 9, “Installation Checklist.” System administrators should read step 10, “Update and Execute the Bind Job,” and step 11, “Update and execute the Run Job” in the program directory. Sample invocation JCL is listed in Chapter 22, “Sample Invocation JCL” on page 401.

Make sure to apply the correct DB2 maintenance before installing the Apply for MVS program. The correct DB2 maintenance consists of the set of DB2 maintenance instructions in the program directory of the Apply program, and the current DB2 maintenance recommended by IBM Service.

Specifying Tuning Parameters for the Capture Program

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

Retention limit

The number of minutes to keep the change data (CD) table rows and the unit-of-work (UOW) table rows. The default value is 10,800, which is 7 days. The rows are deleted up to where the changes have been applied.

Lag limit

The number of minutes the Capture program can be backlogged from the current local time before shutting itself down. The default value is 10,800 (which is 7 days). This value is higher for a busy system; therefore, a lower lag limit shuts down the Capture program.

Commit interval

The number of seconds to wait before issuing a COMMIT statement. The default value is 30 seconds. Set the interval smaller than the DB2 timeout interval if the Capture and Apply programs are running at the same time. This precaution helps to avoid locking overhead for Capture running on DB2 for MVS 3.1 where uncommitted read is not supported. 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. To avoid unnecessary lock contentions, issue a COMMIT statement regularly. This should be close to the DB2 timeout value since COMMIT has associated resource costs. See the DB2 Administration Guide for additional details.

Prune interval

The number of seconds to wait 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 with the PRUNE command.

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

- Modify DPCNTL.* in the Control Center /sqlib/samples/repl directory before you define the first replication source for a database.
- If you want to change the default values, update the table with the following SQL statement after you create the tuning parameters table:

```
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
```

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 information on the structure of the tuning parameters table, see Chapter 19, “Table Structures” on page 295.

DB2 for MVS Rules Regarding Indexes

Do not specify TYPE 1 indexes if the table space containing the identified table has a LOCKSIZE or ROW. If you specify TYPE 1 in a data sharing environment, a warning message is issued if you specify a value greater than 1 for SUBPAGES. A TYPE 1 index with more than one subpage cannot be accessed when there is inter-DB2 read/write interest in the index.

You cannot use an isolation UR option with an access path that uses a TYPE 1 index.

TYPE 2 indexes are a new type of index in MVS Version 4. Because TYPE 2 indexes do not lock index pages, your applications can avoid deadlock and timeout problems on the index. The key advantage of TYPE 2 indexes is that they allow you to use other Version 4 enhancements, 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, all specifications of SUBPAGES are ignored, and an error message is issued.

If you do not specify a type of index, the 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.

Recommendation

- Defer all MVS 4.1 table indexes as TYPE 2. For best IBM Replication performance, you should bind the Capture and Apply programs using isolation UR. When using this option, all indexes on the control tables are required to be created as TYPE 2 indexes. In addition, if source views are being used in replication subscriptions, and the Apply program is bound using isolation UR, then only TYPE 2 indexes over the registered source tables involved in the source view are usable by the Apply program.
- For a DB2 for MVS or DB2 for OS/390 source servers specify the TYPE 2 clause on every generated CREATE INDEX statement and every create index statement in the DB2 UDB Control Center file DPCNTL.MVS.

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.
- Make changes that affect the structure of source tables. This includes changes resulting from data definition language or utilities. Structural changes can compromise the data integrity of the copies.

Capture for MVS does not support certain programmatic statements for the source table. See “Data Restrictions” on page 67 for a list of these statements.

Other Capture program restrictions are:

- Capture for MVS does not support compressed data for a source table if you use DB2 for MVS Version 3.1.
- If the Capture program shuts itself down, you should perform a cold start if the database does not have or support the archive log. DB2 for MVS Version 3.1 does not support the archive log.
- User tables must be created or altered using the DATA CAPTURE CHANGES option.
- Capture does not capture any changes made by DB2 utilities, even if you specify LOG=YES.
- If you are using DB2 for MVS/ESA Version 4.1 or higher, only one instance of the Capture program can run on any subsystem that is a member of a data-sharing group. The DB2 log interface is serial, not parallel. The Capture program must read merged log records from all data sharing members.

Authorization for Running the Capture Program

The user ID that operates the Capture program should either be SYSADM or have the following authorizations:

- SELECT, UPDATE, INSERT, and DELETE privilege for all Capture-related tables created explicitly, and any Capture-related tables the Control Center implicitly creates. See Chapter 19, “Table Structures” on page 295 for a list of these tables.
- SELECT privilege for the DB2 catalog table, SYSIBM.SYSTABLES
- TRACE privilege
- MONITOR1 and MONITOR2 privilege
- EXECUTE privilege for the Capture plan

Recovering from Severe Errors

Capture for MVS provides you with the following tools to assist you if a severe error occurs:

- Alert generation
- Trace buffer
- Trace output
- Storage dump

The following sections provide information on these tools.

Alert Generation

If a severe error occurs, Capture for MVS 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 Capture for MVS error messages are sent to the MVS console.

Trace Buffer

Capture for MVS 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, IBM replication writes trace information logic flow to SYSPRINT. This information may be used by IBM Service to diagnose operational problems.

Storage Dump

When Capture for MVS 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 may be used by IBM Service to diagnose operational problems.

Operating Capture for MVS

The administrator can use the commands in this section to perform the following Capture for MVS tasks:

- Starting
- Scheduling
- Stopping
- Suspending
- Resuming
- Reinitializing
- Pruning
- Monitoring

You can submit the commands from TSO or the MVS console.

Before You Start the Capture Program

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

- Define one or more replication sources and subscriptions as described in Chapter 8, “Working with Replication Sources” on page 93 and Chapter 9, “Working with Replication Targets” on page 103. This creates the following control tables that must be defined in the database where Capture is to run:

- ASN.IBMSNAP_REGISTER
- ASN.IBMSNAP_PRUNCNTL
- ASN.IBMSNAP_CCPPARMS
- ASN.IBMSNAP_TRACE
- ASN.IBMSNAP_WARM_START
- ASN.IBMSNAP_UOW
- ASN.IBMSNAP_CRITSEC
- ASN.IBMSNAP_SUBS_SET
- ASN.IBMSNAP_SUBS_MEMBR
- ASN.IBMSNAP_SUBS_STMTS
- ASN.IBMSNAP_SUBS_COLS
- ASN.IBMSNAP_SUBS_EVENT
- ASN.IBMSNAP_APPLYTRAIL

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. Bind JCL is included in the Capture for MVS program product JCL. Refer to the Capture for MVS program directory for information on running the bind program for Capture.
- Ensure that ASN.IBMSNAP_REGISTER has at least one entry in it by defining a replication source with the DATA CAPTURE CHANGES option. See Chapter 8, “Working with Replication Sources” on page 93 for more information.

Starting Capture for MVS

After you start the Capture program, it runs continuously until you stop it or it detects an error.

To start the Capture program:

1. Prepare the JCL for MVS by specifying the appropriate optional invocation parameters in the PARM field of the ASNLRN nn DD statement. Tailor the JCL to meet your site's requirements. Invocation JCL is included with the Capture for MVS product and is listed in Chapter 22, “Sample Invocation JCL” on page 401.

An example of this line in the invocation JCL is:

```
//ASNLRN $nn$  EXEC PGM=ASNLRP $nn$ ,PARM='DB2_subsys_name NOTERM WARMNS NOPRUNE'
```

where nn is the level of the Capture program, as follows:

- For the Capture program that runs on DB2 for MVS Version 3 Release 1, nn is 23
 - For the Capture program that runs on DB2 for MVS Version 4 Release 1, nn is 24
 - For the Capture program that runs on DB2 for OS/390 Version 5 Release 1, nn is 25
2. Submit the JCL from TSO or from the MVS console. Capture for MVS can run either as a batch job or a started task.

Table 9 on page 135 defines the parameters.

Table 9. ASNLRP Invocation Parameter Definitions for MVS

Parameter	Definition
DB2_subsys_name	The default is DSN for the subsystem name. This must be the first parameter.
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 comes up, 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. Refer to "Warm and Cold Starts" on page 138 for more information.
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, Capture terminates instead of switching to a cold start as when WARM is specified. Refer to "Warm and Cold Starts" on page 138 for more information.
COLD	The Capture program starts up by deleting all rows in its CD tables, UOW table, and trace table during initialization. Refer to "Warm and Cold Starts" on page 138 for more information.
PRUNE (default)	The Capture program automatically prunes the CD and the UOW tables that the Apply program has copied since the last pruning.
NOPRUNE	Automatic pruning is disabled. The Capture program prunes the CD and UOW tables when you enter the PRUNE command. See the PRUNE command for more information.
NOTRACE (default)	No trace information is written.
TRACE	Writes debug trace messages to the standard output, <i>SYSPRINT</i> .

Scheduling Capture for MVS

Use either the \$TA JES2 command or the AT NetView command to start Capture for MVS at a specific time. You must:

1. Create a procedure that calls Capture for MVS 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 MVS

```
▶▶—F—jobname,STOP—▶▶
```

Purpose

Use the STOP command to stop the Capture program gracefully and commit the log records that it processed up to that point.

Usage

Enter this command from TSO or the MVS console before:

- Removing an existing replication source
- Modifying an existing replication source

Stopping the Capture Program with a Trigger

In some situations, you might want to trigger the stop of the Capture program. For example, you might want to stop it at a specific time, or after all transactions for a particular application have been committed. The simplest way to set up this trigger is to create a table (or use an existing table), define it as a replication source, and identify a particular column whose update, when captured, triggers shutdown.

To stop the Capture program with a trigger:

Use the following example as a guideline.

1. Create a table with a column named STOPCAP or another unique identifier.
2. Define the table as a replication source.
3. Create an application that periodically reads the CD table associated with this replication source table to search for STOPCAP.
4. Schedule the application to start before your Apply programs.

When your application encounters STOPCAP in the CD table, it should issue the command to stop the Capture program.

Suspending Capture for MVS

```
▶▶—F—jobname,SUSPEND—▶▶
```

Purpose

Use the SUSPEND command to relinquish MVS 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.

Usage

Enter the SUSPEND command from TSO or the MVS console.

Attention: Do not use SUSPEND when canceling a replication source. Instead, stop the Capture program by entering the STOP command.

Resuming Capture for MVS

```
▶▶—F—jobname,RESUME————▶▶
```

Purpose

Use the RESUME command to restart the suspended Capture program.

Usage

Enter the RESUME command from TSO or the MVS console.

Reinitializing Capture for MVS

```
▶▶—F—jobname,REINIT————▶▶
```

Purpose

Use the REINIT command to have the Capture program reread the ASN.IBMSNAP_REGISTER tables or the ASN.IBMSNAP_CCPPARMS table while it is running. Use this command when you add a new replication source with Control Center while the Capture program is running.

This command ensures that the Capture program recognizes the new replication source. It also rereads the ASN.IBMSNAP_CCPPARMS table for any changes made to the tuning parameters.

Attention: Do not use REINIT 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 use the warm start (WARM or WARMNS) option.

Usage

Enter the REINIT command from TSO or the MVS console.

Pruning the Change Data and Unit-of-Work Tables

```
▶▶—F—jobname,PRUNE————▶▶
```

Purpose

Use the PRUNE command to start pruning the change data (CD) table and unit-of-work (UOW) tables if you used the NOPRUNE invocation parameter to disable pruning when you started the Capture program. This command prunes the CD and UOW tables once.

Usage

Enter the PRUNE command from TSO or the MVS console. 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.

Providing the Current Log Sequence Number

```
►►—F—jobname,GETLSEQ—◄◄
```

Purpose

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.

Usage

Enter the GETLSEQ command from TSO or the MVS console. The Capture program issues the message ASN0125I indicating when the current log sequence number is successfully processed.

Warm and Cold Starts

This section explains how the Capture program handles warm starts, how it switches to an automatic cold start, and when you might want to force a warm start.

Warm Start Process

When you start the Capture program with the WARM or WARMNS parameter, it searches for the warm start table, ASN.IBMSNAP_WARM_START, which is created during the first definition of a registration source or when DPCNTL.* is executed. This table contains information that enables the Capture program to quickly resynchronize to the time when it stopped. If the warm start table is empty, the Capture program can resynchronize using either the register table ASN.IBMSNAP_REGISTER, UOW table, or CD tables.

Warm start information is saved in most cases. In extreme cases, warm start information might not be saved. For example, an operator might cancel the Capture program or stop DB2. In this case, the Capture program uses the CD, UOW, or register tables to resynchronize to the time it was stopped.

After a successful warm start, the old rows in the warm start table are deleted.

The Capture program switches to a cold start if you did not specify WARMNS and the warm start log sequence number is not available in the DB2 3.1 active log, or if it is not available in the DB2 for OS/390 active or archived logs. The Apply program performs a full refresh after a cold start. You can use the technique described in “Loading the Tables within a Subscription Set” on page 144 instead of having the Apply program perform the LOADX of the target table. For information about handling gap messages, see “Problems Using the Apply Program” on page 147 and read the section about forcing the Apply program to perform a full refresh.

Automatic Cold Starts

Sometimes the Capture program automatically switches to a cold start, even when you specify a warm start. The switch is made when:

- The warm start log sequence lags behind the current log sequence by more than the LAG_LIMIT value, as specified in the tuning parameters table ASN.IBMSNAP_CCPPARMS.
- The warm start log sequence is not available on the DB2 active log. This situation applies only for DB2 for MVS 3.1.
- You invoke the Capture program for the first time.

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

In each of these cases, the Capture program issues an informational message and performs a cold start.

Forcing a Warm Start

You might want to prevent the Capture program from cold starting in some situations. For instance, the Capture program cold starts if DB2 goes down, or if someone brings down the DB2 tablespace containing the CD table. Forcing a warm start with the WARMNS parameter ensures that the control tables remain intact. You must correct the problem that caused the Capture program to terminate. If you do not correct the problem, the Capture program continues to terminate or perform a cold start every time you start it.

Operating Apply for MVS

An administrator can use the commands in the following sections to perform the following Apply for MVS tasks:

- Starting
- Scheduling
- Stopping

You can submit the commands from TSO or an MVS console.

Before You Start the Apply Program

Before you start the Apply program, ensure that:

- The control tables are defined.
- At least one subscription is created and activated.
- The Apply program package is created.

See the Apply for MVS program directory and Chapter 22, “Sample Invocation JCL” on page 401 for information on BIND programs to create the Apply program packages. You must bind the Apply program to both the source and target databases.

- You must 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
- For DB2 for MVS/ESA, you need to 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.

For DB2 Universal Database, you must have DBADM or CONTROL or SELECT authority that meets all requirements for defining a replication source and target.

Starting Apply for MVS

After you start the Apply program, it runs continuously until:

- You stop it in an orderly way.
- An unexpected error or failure occurs.

To start the Apply for MVS program:

1. Prepare the JCL for MVS by specifying the appropriate optional invocation parameters in the PARM field of the ASNARUN nn command. Tailor the JCL to meet your site's requirements. Invocation JCL is included with the Apply for MVS product.

An example of this line in the invocation JCL is:

```
//ASNARUN EXEC PGM=ASNAPVnn,PARM='Apply_qual DB2_subsystem_name DISK'
```

Where nn indicates the version of Apply required for a given version of DB2 for MVS:

- For the Apply program running on DB2 for MVS Version 3 Release 1, nn is 23
- For the Apply program running on DB2 for MVS Version 4 Release 1, nn is 24
- For the Apply program running on DB2 for OS/390 Version 5 Release 1, nn is 25

Table 10 on page 141 defines the required and optional positional parameters.

Table 10. ASNARUN Invocation Parameter Definitions

Parameter	Definition
<i>Apply_qual</i>	Specifies the Apply program qualifier 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 is a positional parameter that is required as the first parameter.
<i>DB2_subsystem_name</i>	This is a positional parameter that is required as the second parameter.
<i>Control_server_name</i>	Specifies the name of the server where the replication control tables will reside. If you do not specify this optional positional parameter, the default is the current server.
LOADXit	Specifies that the Apply program is to invoke ASNLOAD, an exit program that can call an IBM or multivendor utility, to initialize a target table. This keyword parameter is optional.
NOLOADXit (default)	Specifies that the Apply program will not invoke ASNLOAD. Currently, no utility program is available for use by ASNLOAD on DB2 for MVS.
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. This keyword parameter is optional.
DISK	Specifies that a disk file stores the fetched answer set. This keyword parameter is optional.
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. This keyword parameter is optional.
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. This keyword parameter is optional.
TRCFLOW	Specifies that the Apply program generates a trace that contains error and execution flow information. This keyword parameter is optional.
NOTIFY	Specifies that the Apply program is to invoke ASNDONE, an exit program that returns control to the user when the Apply program processing ends. This keyword parameter is optional.
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. This keyword parameter is optional.
DELAY (<i>n</i>)	Where <i>n</i> =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. This keyword parameter is optional.
DELAY (6) (default)	Specifies a delay time of 6 seconds at the end of each Apply program cycle when continuous replication is used.

Scheduling Apply for MVS

Use either the \$TA JES2 command or the AT NetView command to start Apply for MVS at a specific time.

To start Apply for MVS at a specific time:

1. Create a procedure that calls Apply for MVS 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 See *NetView for MVS Command Reference* for more information about using the AT NetView command.

Stopping Apply for MVS

Enter the following command from TSO or an MVS console to stop the Apply for MVS program:

```
▶▶—P—jobname————▶▶
```

Additional Apply Program Operations

The following sections provide information about performing the additional Apply program functions of scheduling subscriptions, returning control to users, initiating a forward recovery, and loading tables within a subscription set.

Scheduling Subscriptions with the Event Table

You can include an event name in your replication subscription. Your subscription cycles can run according to a timer, an event occurrence, or both. See “Setting the Copying Schedule: Time or Event Based” on page 114 for more information about the purpose and usage of event scheduling.

You can schedule a subscription by manipulating the event table. The DDL statement for the event table appears as follows:

```
CREATE TABLE ASN.IBMSNAP_SUBS_EVENT (  
    EVENT_NAME CHAR(18) NOT NULL,  
    EVENT_TIME TIMESTAMP NOT NULL,  
    END_OF_PERIOD TIMESTAMP);  
CREATE UNIQUE INDEX ASN.IBMSNAP_SUBS_EVENT  
    ON ASN.IBMSNAP_SUBS_EVENT (EVENT_NAME ASC, EVENT_TIME ASC);
```

For example, to start the Apply program at 6:00 p.m. on 9/24/97 and copy changes from the source server up to 4:00 p.m. on the same date, you would insert the following values in ASN.IBMSNAP_SUBS_EVENT:

```
('EVENTA', '1997-09-24-18.00.00.00000', '1997-09-24-16.00.00.00000'),
```

A subscription becomes eligible for processing when EVENT_TIME falls between LASTSUCCESS and the control server's CURRENT_TIMESTAMP. The END_OF_PERIOD column enables you to specify a timestamp upper limit for change data.

Note: You must still ensure that the Capture program has processed through the log up to the end-of-period timestamp to ensure that all data up until that time has been captured.

Returning Control to Users with ASNDONE

ASNDONE is a user exit program that the Apply program invokes after set subscription processing completes, regardless of success or failure. You can modify ASNDONE to meet the requirements of your installation. For example, the user exit can examine the UOW table to discover rejected transactions and initiate further actions, such as issuing a message or generating an alert.

To use ASNDONE:

1. Modify the ASNDONE file 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 NONOTIFY invocation parameter to call the ASNLOAD program.

The parameters passed to ASNDONE are as follows:

- Control server name
- Set name
- Apply qualifier
- WHOS_ON_FIRST value
- Trace option
- Status value

Initiating a Forward Recovery with the Apply Program

In cases of an incomplete rollback or partially restored backup, you might need to copy a “window” of changes for completion.

To use the Apply program to limit the range of changes to those excluded from the rollback:

1. Determine how much data is lost.

Check the IBMSNAP_LOGMARKER value in the target table. The highest value is the most recently committed change.

2. Use SQL to identify the range of changes. Either:

- Use the WHERE clause on the Rows page of the Advanced Subscription Definition notebook.

or

- Manually update the PREDICATES column of the control server subscription table (ASN.IBMSNAP_SUBS_MEMBR).

For example, if your timestamp shows the last committed change to be 941106174322, your SQL should reflect changes that occur after that time.

Loading the Tables within a Subscription Set

The following section describes how to load the tables of a subscription set as an alternative to using the LOADX option when running the Apply program. This technique involves performing your own full refresh on behalf of the Apply program. By doing your own full refresh, the Apply program assumes that *it* has initialized your point-in-time copies. You may want to use this technique in the following situations:

- To automate the loading of many, large copies of tables.
- To fully-refresh tables with referential integrity constraints, in order to bypass referential integrity checking.

Note: Before you begin, ensure that the Capture program is running.

The following example assumes that you want to perform a load of the tables within the replication subscription SET001 with an Apply program qualifier of APPLY001.

1. Optional: It is recommended that you disable the full refresh capability for the applicable source tables as shown in the following SQL statements; the Apply program then issues an error message, rather than performing a full refresh if the following procedure is not performed correctly.

```
UPDATE ASN.IBMSNAP_REGISTER SET DISABLE_REFRESH = 1
WHERE SOURCE_OWNER = 'source owner' AND
SOURCE_TABLE = 'source table'
```

2. Ensure that the Apply program is inactive, or deactivate the applicable replication subscription as shown in the following SQL statements:

```
UPDATE ASN.IBMSNAP_SUBS_SET SET ACTIVATE = 0
WHERE SET_NAME = 'SET001' AND APPLY_QUAL = 'APPLY001'
```

3. Update SYNCHPOINT and SYNCHTIME in ASN.IBMSNAP_PRUNCNTL for each row that corresponds to a member of the replication subscription as shown in the following SQL statements:

```
UPDATE ASN.IBMSNAP_PRUNCNTL
SET SYNCHPOINT = x'00000000000000000000',
    SYNCHTIME = CURRENT_TIMESTAMP
WHERE SET_NAME = 'SET001' AND APPLY_QUAL = 'APPLY001'
```

4. Verify that the Capture program has processed the updates in the step above.
 - If the Capture program has not started capturing changes for these tables, this update initiates the process. ASN.IBMSNAP_TRACE will contain GOCAPT messages for the source tables.
 - If the Capture program has previously started capturing changes for these tables (other subscriptions exist for the tables and these have already been activated), you can verify that this pruning control table update was successful

by querying the same SYNCHPOINT values that were updated to hex zeroes. When these values no longer appear as zeroes, the Capture program has performed the necessary processing for you to continue.

5. Unload the source table and load the target table, using DB2 or vendor utility programs.

6. Update ASN.IBMSNAP_SUBS_SET as shown in the following SQL statements:

```
UPDATE ASN.IBMSNAP_SUBS_SET
SET LASTRUN = CURRENT_TIMESTAMP,
LASTSUCCESS = CURRENT_TIMESTAMP,
SYNCHTIME = CURRENT_TIMESTAMP,
SYNCHPOINT = NULL
WHERE SET_NAME = 'SET001' AND APPLY_QUAL = 'APPLY001'
```

7. Start the Apply program, or set ACTIVATE = 1 for the applicable replication subscription row in ASN.IBMSNAP_SUBS_SET as shown in the following SQL statements:

```
UPDATE ASN.IBMSNAP_SUBS_SET SET ACTIVATE = 1
WHERE SET_NAME = 'SET001' AND APPLY_QUAL = 'APPLY001'
```

Note: In cases where you are trying to identify problems by manually resetting values in the PRUNCNTL table, note that the PRUNCNTL table's SYNCHPOINT values need to match the CD_OLD_SYNCHPOINT values in the REGISTER table. If the values do not match, then the Apply program will want to perform a full refresh.

Troubleshooting

The following sections describe commonly known issues with Capture and Apply for MVS programs.

Problems Using the Capture Program

Before running Capture for MVS, ensure that you have applied all PTFs for your level of DB2 and for the IFI installed on your DB2 subsystem. Specific Capture program problems are described below.

Problem

The Capture for MVS program does not start.

Ensure that APF authorization has been performed for all STEPLIB libraries as specified in the RUN JCL.

Problem

The Capture for MVS program is not capturing updates.

Any of the following could prevent the Capture program from capturing updates:

- Proper authorization was not granted to the administrator. The Capture program requires that the administrator have TRACE, MONITOR1, and MONITOR2 privileges.
- DATA CAPTURE CHANGES was not specified on the base tables to be captured.
- 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 Capture for MVS and look for message number ASN0100I (initialization completed) in the system console or in the ASN.IBMSNAP_TRACE table.
 3. Start the Apply program.

Check the ASN.IBMSNAP_TRACE table for possible error messages.

Problem

Updates to the base table are not being replicated to the target table.

DB2 writes log information to the active log in multiples of 4KB VSAM control intervals (CI). In order for Capture to obtain these log records, there must be at least one CI written to the log. If your application has base table updates less than 4KB in a quiet DB2 (no other log updates), you do not see the update results in the CD table until the 4KB limit is reached. To see your updates you can either:

- Use -ARCHIVE LOG to flush out updates immediately.
- Use a filler table (applicable only for the Capture program running on DB2 for MVS Version 3 Release 1).

Following is an example of a filler table :

```
CREATE TABLE ASN.FILLER (
  GARBAGE0 CHAR(200),
  GARBAGE1 CHAR(200),
  GARBAGE2 CHAR(200),
  GARBAGE3 CHAR(200),
  GARBAGE4 CHAR(200),
  GARBAGE5 CHAR(200),
  GARBAGE6 CHAR(200),
  GARBAGE8 CHAR(200),
  GARBAGE9 CHAR(200));
```

You can insert dummy rows into this sample filler table to reach the 4KB limit, so the Capture program can see other updates immediately.

Problem

I'm not sure if the Capture program is running successfully.

The first time 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 ASN.IBMSNAP_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 ASN.IBMSNAP_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 ASN.IBMSNAP_APPLYTRAIL table.

Problem

Capture for MVS failed while using LE/370 environment.

Capture can run under both C/370 and LE/370 environments. If running under LE/370 environment, you must specify REGION=3M in the RUN step.

Problem

Capture for MVS issued message ASN0000E instead of the proper message number.

If you receive message ASN0000E, it implies that a generic message has been issued instead of a proper message. This happens if the specified VSAM message file in RUN JCL was not found. See the Capture for MVS program directory for information on installing the VSAM message file.

Problem

Capture for MVS terminates.

The Capture for MVS 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 361 for an explanation of this message.

Problems Using the Apply Program

Before running the Apply program, ensure that:

- Replication sources and subscriptions have been defined.
- The Capture program has been started, and the ASN0100I initialization message has been issued (if you are running a Capture program).
- APF-authorization has been performed for all STEPLIB libraries of the Apply for MVSRUN JCL (if you are running Apply for MVS).

- Bind has been performed for all control, source, and target servers that the Apply program accesses.

When an error occurs while running the Apply program, the status field in ASN.IBMSNAP_SUBS_SET is set to -1 for that copy definition.

Errors as well as successful executions are logged in the ASN.IBMSNAP_APPLYTRAIL table. APPERRM contains the message text. Any SQL errors are also logged in this table. Error messages are also issued on the console.

Specific Apply program problems are described below.

Problem

I have 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 has filled to capacity because I copied a very large table.

If the error occurred during a full refresh, you can use an alternative method to load large tables, described in “Loading the Tables within a Subscription Set” on page 144.

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 Mini-Cycles for the Apply Program to Copy Committed Data” on page 116.

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 have 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 ASN.IBMSNAP_REGISTER at the source server for the source table. In this case, the Apply program issues message ASN1016E.

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 ASN.IBMSNAP_SUBS_SET to null.

I unsuccessfully tried to start a second Apply program instance.

You must run each instance with a unique Apply program qualifier.

Problem

I received error ASN1003 with SQLCODE = -1032 and SQLSTATE = 57019.

You must start the database manager before invoking the Apply program.

Problem

Apply components 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 MVS 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 MVS source server.

For example, if you use the Korean character set with mixed data at both DB2 for MVS source server and 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.

Important: If you are running in a mixed environment, ensure you have installed the latest maintenance for the CCSID support of your DB2 for MVS program.

For more information on character translation, see the Character Conversion for Distributed Data chapter in *DB2 Version 4 Administration Guide, Volumes 1,2*.

Problem

I received a security violation message, and the Apply program is not authorized to connect to the database.

The control server name, userid, and password definitions must match exactly those specified in the password file, and are case sensitive. Check your definitions again.

Chapter 11. Capture and Apply for Windows NT and Windows 95

This chapter describes how to set up, operate, and troubleshoot Capture and Apply for Windows NT and Windows 95.

Read the following sections before reading the sections on operating the Capture program:

- “Setting Up the Capture and Apply Programs”
- “Specifying Tuning Parameters for the Capture Program” on page 155
- “Restrictions When Running the Capture Program” on page 156
- “Authorization for Running the Capture Program” on page 156

Setting Up the Capture and Apply Programs

Setting up consists of configuring the source, target, and control servers, and setting up NT services. 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 the Capture Program for Windows NT and Windows 95

1. Ensure that the user ID under which the Capture program is running has the required privileges:

- Execute privilege on the Capture packages
- DBADM or SYSADM privileges for the database

2. Log on with the user ID that has sufficient privileges.

3. Connect to the source server database by entering:

```
DB2 CONNECT TO database
```

where *database* is the source server database.

4. Prepare the source server database (source 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 UPDATE DATABASE CONFIGURATION FOR database_alias USING USEREXIT ON
DB2 BACKUP DATABASE database_alias
```

Note: You might need to increase DBHEAP, APPLHEAPSZ, and LOGBUFSZ based on your installation requirements.

5. Change to the directory where the Capture program bind files are located, which is usually *drive*:\SQLLIB\BND.

6. 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 can be found in the CAPTURE.LST file.

Configuring the Apply Program for Windows NT and Windows 95

1. Ensure that the user ID under which the Apply program is running has the required privileges:

- Execute privilege for the Apply packages
- DBADM or SYSADM privileges for the database

2. Log on with the user ID that has sufficient privileges.
3. Change to the directory where the Apply program bind files are located, which is usually in *drive:\SQLLIB\BND*.
4. Connect to the source server database by entering:

```
DB2 CONNECT TO database
```

where *database* is the source server database.

5. Create and bind the Apply 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 The list in cursor stability format.

UR The list in uncommitted read format.

These commands create a list of packages, the names of which can be found in the APPLYCS.LST and APPLYUR.LST files.

6. Connect to the target server database by entering:

```
DB2 CONNECT TO database
```

where *database* is the target server database.

7. Create and bind the Apply 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
```

Note: Because the Apply program control tables use static SQL calls, the Apply bind process searches for nearly all of the control tables at each server that it is bound to, regardless of whether these control tables have any use or meaning at each server.

8. Repeat the connect and bind steps for each server that the Apply program connects to. You must bind the Apply program to the 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 the AUTH=SERVER scheme for the Apply program to use when connecting to the source server. It is advisable to limit read access of this file 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><instname><CNTLSRVR>.PWD`

Where:

APPLYQUAL

The Apply qualifier in upper case. The Apply qualifier is case sensitive and must match the value of APPLY_QUAL in the subscription set table.

instname

The instance name that the Apply program runs under (the value of DB2INSTANCE).

CNTLSRVR

The name of the control server in upper case.

For example: DATADIRapp1y1REPRDDB.PWD

Note that this naming convention is the same as the .LOG file name and the .SPL file name, but with a file extension of PWD.

- Reside in the directory from which the Apply program starts
- Contain all server-name/password pairs for the file. The pairs enable you to have a different (or the same) password at each control server. The Apply user ID is used for all connections.
- Have one or more records using the following format:

`SERVER=server_name USER=userid PWD=password USER=userid`

Where *server_name* is the source, target, or control database. The file cannot include blank lines or comment lines.

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 NT and Windows 95 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.

To set up NT services:

1. Install the replication service by entering the fully-qualified path name of the ASNSERV executable as shown in the following example:
ASNINST D:\SQLLIB\BIN\ASNSERV.EXE
2. Set up the service from the NT Control Panel.
 - a. Click on the Services icon. The Services window appears.
 - b. Select Replication and click on the STARTUP push button.
 - c. Ensure that the startup type radio button indicates automatic.
 - d. Specify the local user ID and password, then click on OK. The user ID must be the one that runs the Capture and Apply programs with the appropriate DB2 privileges.
3. Add the environment variable ASNPATH to specify the location of the Capture and Apply files.
 - a. Select System from the NT Control Panel. The System Properties window appears.
 - b. Click on the Environment tab.
 - c. Type the ASNPATH string in the System Variable field as shown in the following example:
ASNPATH=D:
 - d. Click on OK.
4. Create an ASCII file to execute 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* is the name of your source database for the Capture program and control database for the Apply program, *x* is the drive, and *parameters* is one or more parameters as in the following example:
DBNAME1 C:\CAPTURE\ASNCCP COLD TRACE<CRLF>
DBNAME2 C:\APPLY\ASNAPPLY APPLYQUAL DBNAME2<CRLF>
 - b. Save the file to the following name:
D:\NTSERV.ASN

After starting the service, the Capture and Apply programs run independently of ASNSERV. Consequently, stopping ASNSERV does not stop the Capture and Apply programs.

To remove ASNSERV from the NT Control Panel, use the ASNREMV program. After removing ASNSERV, Replication no longer appears on the Services window.

Specifying Tuning Parameters for the Capture Program

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

Retention limit

The number of minutes to keep the change data table rows and the unit-of-work (UOW) table rows. The default value is 10,800, which is 7 days. The rows are deleted up to where the changes have been applied.

Lag limit

The number of minutes the Capture program can be backlogged from the current local time before shutting itself down. The default value is 10,800 (which is 7 days). This value is higher for a busy system; therefore, a lower lag limit shuts down the Capture program. If the Capture program shuts itself down, you should perform a cold start if the database does not have or support an archive log.

Commit interval

The number of seconds to wait before issuing a COMMIT statement. The default value is 30 seconds. Set the interval smaller than the DB2 timeout interval if the Capture and Apply programs are running at the same time. This precaution helps to avoid locking overhead. 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.

Prune interval

The number of seconds to wait 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.

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

- Modify DPCNTL.* in the Control Center /sqlib/samples/repl directory before you define the first replication source for a database.
- Update the table with the following SQL statement after you create the tuning parameters table:

```
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
```

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 information on the structure of the tuning parameters table, see Chapter 19, “Table Structures” on page 295.

Restrictions When Running the Capture Program

The Capture program does not capture any changes made by DB2 utilities.

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:

- Cancel an existing replication source.
- Drop a replication source table.
- Make changes that affect the structure of source tables. This includes changes resulting from data definition language or utilities. Structural changes can compromise the data integrity of the copies.

Authorization for Running the Capture Program

The user ID that operates the Capture program should either be the replication administrator or have DBADM or SYSADM authority.

Operating Capture for Windows NT and Windows 95

The IBM Replication administrator user ID can use the commands in this section to operate Capture for Windows NT and Windows 95. Enter the commands or key combinations from an NT window.

This section explains how to perform the following Capture program tasks:

- Starting
- Scheduling
- Stopping
- Suspending
- Resuming
- Reinitializing
- Pruning
- Monitoring

Before You Start the Capture Program

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

- Define one or more replication sources and subscriptions as described in Chapter 8, “Working with Replication Sources” on page 93 and Chapter 9, “Working with Replication Targets” on page 103. This creates the following control tables that must be defined in the database where the Capture program is to run:
 - ASN.IBMSNAP_REGISTER
 - ASN.IBMSNAP_PRUNCNTL
 - ASN.IBMSNAP_CCPPARMS
 - ASN.IBMSNAP_TRACE
 - ASN.IBMSNAP_WARM_START

- ASN.IBMSNAP_UOW
- ASN.IBMSNAP_CRITSEC
- ASN.IBMSNAP_SUBS_SET
- ASN.IBMSNAP_SUBS_MEMBR
- ASN.IBMSNAP_SUBS_STMTS
- ASN.IBMSNAP_SUBS_COLS
- ASN.IBMSNAP_SUBS_EVENT
- ASN.IBMSNAP_APPLYTRAIL

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 servers from which the Capture program will capture changes. See “Configuring the Capture Program for Windows NT and Windows 95” on page 151 for more information.
- Enable the DATA CAPTURE CHANGES attribute for your source table using either the CREATE TABLE or ALTER TABLE statement. See Chapter 8, “Working with Replication Sources” on page 93 for more information.
- Decide whether you want to use the NT Service Control Manager (SCM) to automatically run the Capture program as an NT service. The following section provides information about operating the Capture program as an NT service.

Starting Capture for Windows NT and Windows 95

After you start the Capture program, it runs continuously until you stop it or it detects an error.

To start the Capture program using the NT services:

1. Select Replication from the NT Services window.
2. Click on 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 instances, from the System window, set the DB2INSTANCE environment variable to the DB2 for NT 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` is created in the directory from which the Capture program is invoked. This file is a log file for the messages issued by the Capture program; these messages are also recorded in the ASN.IBMSNAP_TRACE table.

2. Enter the ASNCCP command from the Capture for Windows NT or Windows 95 window where you issued the SET command. The syntax is:

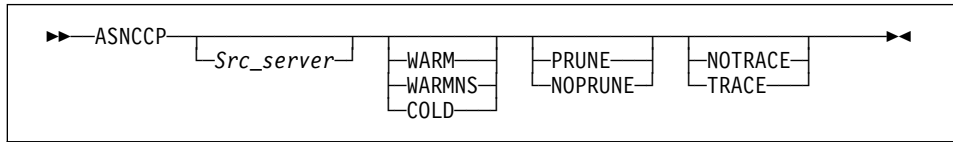


Table 11 defines the optional parameters.

Table 11. ASNCCP Command Parameter Definitions for Windows NT or Windows 95

Parameter	Definition
Src_server	Source server name positional parameter that must be the first parameter if entered. If not specified, the value from the DB2DBDFT environment variable will be 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. See "Warm and Cold Starts" on page 162 for more information.
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, change data, 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. See "Warm and Cold Starts" on page 162 for more information.
COLD	The Capture program starts by deleting all rows in its change data, UOW, and trace tables during initialization. See "Warm and Cold Starts" on page 162 for more information.
PRUNE (default)	The Capture program automatically prunes the change data and UOW tables.
NOPRUNE	Automatic pruning is disabled. The Capture program prunes the change data and UOW tables when you enter the PRUNE command. See the PRUNE command for more information.
NOTRACE (default)	No trace information is written.
TRACE	Writes debug trace messages to the standard output, <i>stdout</i> .

Scheduling Capture for Windows NT and Windows 95

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 NT at 15:00.

```
c:\>AT 15:00 /interactive "c:\SQLLIB\BIN\db2cmd.exe c:\CAPTURE\asnccp.exe cold"
```

Note: The Windows NT or Windows 95 Schedule Service must be started before using the AT command.

Stopping Capture for Windows NT and Windows 95

If you started the Capture program as an NT service, stop ASNCCP by selecting Replication from the NT Services window and clicking the STOP push button. After the stop message appears, the status field becomes blank.

If you started the Capture program from the DB2 command window, enter the following command:



▶▶—ASNCMD—STOP—◀◀

or one of the following:

- Ctrl+C
- Ctrl+Break

Purpose

Use the STOP command or the 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 an existing replication source
- Opening an existing replication source and modifying the **Enable Refresh and Update** field

Usage

- To use the command, do the following from a different window than where the Capture program is 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).
 3. Enter the command.
- If you want to use one of the key combinations, press it from the same NT window in which the Capture program is running.

Stopping the Capture Program with a Trigger

In some situations, you might want to trigger the stop of the Capture program. For example, you might want to stop it at a specific time, or after all transactions for a particular application have been committed. The simplest way to set up this trigger is to create a table (or use an existing table), define it as a replication source, and identify a particular column whose update, when captured, triggers shutdown.

To stop the Capture program with a trigger:

Use the following example as a guideline.

1. Create a table with a column named STOPCAP or another unique identifier.
2. Define the table as a replication source.
3. Create an application that periodically reads the change data table associated with this replication source table to search for STOPCAP.
4. Schedule the application to start before your Apply programs.

When your application encounters STOPCAP in the change data table, it should issue the command to stop the Capture program.

Suspending Capture for Windows NT and Windows 95

▶▶—ASNCMD—SUSPEND————▶▶

Purpose

Use the SUSPEND command to relinquish Windows NT or Windows 95 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.

Usage

To use the command, do the following from a different window than where the Capture program is 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).
3. Enter the command.

Attention: Do not use SUSPEND when canceling a replication source. Instead, stop the Capture program using Ctrl+C.

Resuming Capture for Windows NT and Windows 95

▶▶—ASNCMD—RESUME————▶▶

Purpose

Use the RESUME command to resume the suspended Capture program.

Usage

To use the command, do the following from a different window than where the Capture program is 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).
3. Enter the command.

Reinitializing Capture for Windows NT and Windows 95

▶▶—ASNCMD—REINIT————▶▶

Purpose

Use the REINIT command to make the Capture program reread the ASN.IBMSNAP_REGISTER tables or the ASN.IBMSNAP_CCPPARMS table while it is running. Use this command when you add a new replication source with the Control Center while the Capture program is running.

This command ensures that the Capture program recognizes the new replication source. The Capture program also rereads the ASN.IBMSNAP_CCPPARMS table for any changes made to the tuning parameters.

Attention: Do not use REINIT 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 use warm start (WARM or WARMNS).

Usage

To use the command, do the following from a different window than where the Capture program is running:

1. Set environment variable DB2INSTANCE to the value that was 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 that was used when the Capture program was started).
3. Enter the command.

Pruning the Change Data and Unit-of-Work Tables

▶▶—ASNCMD—PRUNE————▶▶

Purpose

Use the PRUNE command to start pruning the change data (CD) table and the unit-of-work (UOW) tables if you used the NOPRUNE invocation parameter to disable pruning when you started the Capture program. This command prunes the CD and UOW tables once. The Capture program issues the message ASN0124I when the command is successfully queued.

Usage

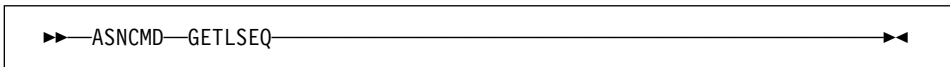
To use the command, do the following from a different window than where the Capture program is running:

1. Set environment variable DB2INSTANCE to the value that was 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 that was used when the Capture program was started).
3. Enter the command.

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.

Providing the Current Log Sequence Number



Purpose

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.

Usage

To use the command, do the following from a different window than where the Capture program is running:

1. Set environment variable DB2INSTANCE to the value that was 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 that was used when the Capture program was started).
3. Enter the command.

Warm and Cold Starts

This section explains how the Capture program handles warm starts, how it switches to an automatic cold start, and when you might want to force a warm start.

Warm Start Process

When you start the Capture program with the WARM or WARMNS parameter, it searches for the warm start table, ASN.IBMSNAP_WARM_START, which is created during the first definition of a registration source or when DPCNTL.* is executed. This table contains information that enables the Capture program to quickly resynchronize to the time when it stopped. If this table is not available, the Capture program can resynchronize using the register table, UOW table, or change data tables.

Warm start information is saved in most cases. In extreme cases, warm start information might not be saved. For example, an operator might cancel the Capture program or stop DB2. In this case, the Capture program uses the change data, UOW, or register tables to resynchronize to the time it was stopped.

After a successful warm start, the old rows in the warm start table are deleted.

The Capture program switches to a cold start if you did not specify WARMNS and the warm start log sequence number is not available in the log. The Apply program performs a full refresh after a cold start. For information about handling gap messages, see “Problems Using the Apply Program” on page 171 and read the section about forcing the Apply program to perform a full refresh.

Automatic Cold Starts

Sometimes the Capture program automatically switches to a cold start, even when you specify a warm start. The switch is made when:

- The warm start log sequence lags behind the current log sequence by more than the LAG_LIMIT value, as specified in the tuning parameters table ASN.IBMSNAP_CCPPARMS.
- You invoke the Capture program for the first time.

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

For each of these cases, the Capture program issues an informational message and performs a cold start.

Forcing a Warm Start

You might want to prevent the Capture program from cold starting in some situations. For instance, the Capture program cold starts if DB2 goes down, or if someone brings down the DB2 table space containing the change data tables. Forcing a warm start with the WARMNS parameter ensures that the control tables remain intact. You must correct the problem that caused the Capture program to terminate. If you do not correct the problem, the Capture program continues to terminate every time you start it.

Operating Apply for Windows NT and Windows 95

An administrator can use the commands in the following sections to perform the following Apply for Windows NT tasks:

- Starting
- Scheduling
- Stopping

Before You Start the Apply Program

Before you start the Apply program, ensure that:

- The control tables are defined.
- 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 153 for more information.

- At least one subscription is created and activated.
- The Apply for Windows NT package is created. See “Configuring the Apply Program for Windows NT and Windows 95” on page 152 for information on bind programs to create the Apply packages.
- You have the proper authorization. See “Authorization Requirements for the Apply Program” on page 51 for information about authorization for the Apply program.

Starting Apply for Windows NT and Windows 95

After you start the Apply program, it runs continuously until:

- You stop it in an orderly way.
- 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 on 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 NT command line.

To start the Apply program using the DB2 command window:

1. Ensure that you set the DB2 instance as shown:
SET DB2INSTANCE=*db2_instance_name*
2. Enter the ASNAPPLY command and one or more options from the Windows NT or Windows 95 window.

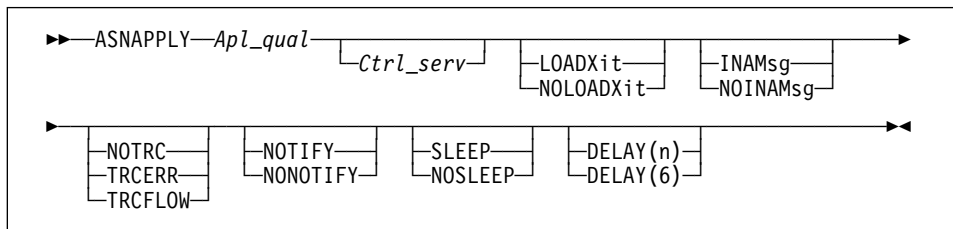


Table 12 on page 165 defines the optional positional parameters.

Table 12. ASNAPPLY Parameter Definitions for NT

Parameter	Definition
<i>ApI_qual</i>	Specifies the Apply program qualifier 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 is a positional parameter that is required as the first parameter.
<i>Ctrl_serv</i>	Specifies the name of the server where the replication control tables will reside. If you do not specify this optional positional parameter, the default is the default database or the value of DB2DBDFT for DB2 for Windows NT or Windows 95.
LOADXit	Specifies that the Apply program is to invoke ASNLOAD, an exit program that can call an IBM or multivendor utility, to initialize a target table. This keyword parameter is optional.
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. This keyword parameter is optional.
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. This keyword parameter is optional.
TRCFLOW	Specifies that the Apply program generates a trace that contains error and execution flow information. This keyword parameter is optional.
NOTIFY	Specifies that the Apply program is to invoke ASNDONE, an exit program that returns control to the user when the Apply program processing ends. This keyword parameter is optional.
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. This keyword parameter is optional.
DELAY (<i>n</i>)	Where <i>n</i> =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. This keyword parameter is optional.
DELAY (6) (default)	Specifies a delay time of 6 seconds at the end of each Apply program cycle when continuous replication is used.

When Apply for Windows NT or Windows 95 executes, it generates two files:

- `<userid><database_instance_name><database_name>.LOG`

This file is created in the directory where Apply for Windows NT or Windows 95 is invoked. It contains messages issued by Apply for Windows NT or Windows 95.

- `<userid><database_instance_name><database_name>.SPL`

This file is created in the directory where Apply for Windows NT or Windows 95 is invoked. It is an internal file used by Apply for Windows NT or Windows 95. Do not erase it.

Scheduling Apply for Windows NT and Windows 95

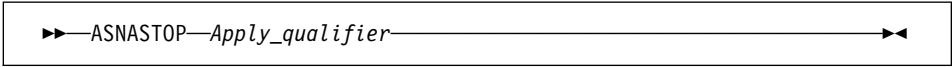
Use the Windows NT or Windows 95 AT command to start the Apply program at a specific time. For example, the following command string starts Apply for Windows NT at 15:00.

```
c:\>AT 15:00 /interactive "c:\SQLLIB\BIN\db2cmd.exe  
c:\SQLLIB\BIN\asnapply.exe qualid1 cnt1db"
```

Note: Before entering the AT command, the Schedule service should already be started.

Stopping Apply for Windows NT and Windows 95

Use the following command to stop Apply for Windows NT or Windows 95.



```
▶▶—ASNASTOP—Apply_qualifier—◀◀
```

You can alternatively use one of the following key combinations from the window where the Apply program is running:

- Ctrl+C
- Ctrl+Break

Purpose

Use the ASNASTOP command or the key combination to stop the Apply program in an orderly way and commit the log records that it processed up to that point.

Usage

To use the command, do the following from a different window than where the Apply program is running:

1. Set environment variable DB2INSTANCE to the value that was 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 that was used when the Apply program was started).
3. Enter the command.

To use one of the key combinations, press it from the same window in which the Apply program is running.

Additional Apply Program Operations

The following section provides information about performing the additional Apply program functions of scheduling subscriptions, refreshing target tables, returning control to users, initiating a forward recovery, and loading tables within a subscription set.

Scheduling Subscriptions with the Event Table

You can include an event name in your replication subscription. Your subscription cycles can run according to a timer, an event occurrence, or both. See “Setting the Copying Schedule: Time or Event Based” on page 114 for more information about the purpose and usage of event scheduling.

You can schedule a subscription by manipulating the event table. The DDL statement for the event table appears as follows:

```
CREATE TABLE
  ASN.IBMSNAP_SUBS_EVENT (
    EVENT_NAME CHAR(18) NOT NULL,
    EVENT_TIME TIMESTAMP NOT NULL,
    END_OF_PERIOD TIMESTAMP);
CREATE UNIQUE INDEX ASN.IBMSNAP_SUBS_EVENT
  ON ASN.IBMSNAP_SUBS_EVENT (EVENT_NAME ASC, EVENT_TIME ASC);
```

For example, to start the Apply program at 6:00 p.m. on 9/24/97 and copy changes from the source server up to 4:00 p.m. on the same date, you would insert the following values in ASN.IBMSNAP_SUBS_EVENT:

```
('EVENTA', '1997-09-24-18.00.00.000000', '1997-09-24-16.00.00.000000'),
```

A subscription becomes eligible for processing when EVENT_TIME falls between LASTSUCCESS and the control server's CURRENT_TIMESTAMP. The END_OF_PERIOD column enables you to specify a timestamp upper limit for change data.

Refreshing Target Tables with ASNLOAD

ASNLOAD is a user exit program that the Apply program invokes when you specify the LOADX invocation parameter. The Apply program calls the ASNLOAD program whenever it performs a full refresh of a target table.

You can use the ASNLOAD program as shipped with the Apply program, or you can modify it. As shipped, ASNLOAD 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. See the prolog section in the sample program (ASNLOAD.SMP) for instructions on how to modify the program to meet the requirements of your installation.

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

Files Generated

If you invoke ASNLOAD, the following files are generated:

- ASNA<userid><database_instance_name><cntl_server>.IXF
- ASNAEXPT<userid><database_instance_name><cntl_server>.MSG

This file contains the data exported from the source.

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.

Error Handling

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

Returning Control to Users with ASNDONE

If you specify the NOTIFY parameter when starting the Apply program, the user exit program ASNDONE is called after subscription processing completes, regardless of success or failure. You can modify ASNDONE to meet the requirements of your installation. For example, the user exit can examine the UOW table to discover rejected transactions and initiate further actions, such as issuing a message or generating an alert.

See the prolog section in the sample program (ASNDONE.SMP) for instructions on how to modify the program.

Initiating a Forward Recovery with the Apply Program

In cases of an incomplete rollback or partially restored backup, you might need to copy a “window” of changes for completion.

To use the Apply program to limit the range of changes to those excluded from the rollback:

1. Determine how much data is lost.
Check the IBMSNAP_LOGMARKER value in the point-in-time table. The highest value is the most recently committed change.
2. Use SQL to identify the range of changes. Either:
 - Use the WHERE clause on the Rows page of the Advanced Subscription Definition notebook.or

- Manually update the PREDICATES column of the control server subscription table ASN.IBMSNAP_SUBS_MEMBR.

For example, if your timestamp shows the last committed change to be 941106174322, your SQL should reflect changes that occur after that time.

Loading the Tables within a Subscription Set

The following section describes how to load the tables of a subscription set as an alternative to using the LOADX option when running the Apply program. This technique involves performing your own full refresh on behalf of the Apply program. By doing your own full refresh, the Apply program assumes that *it* has initialized your point-in-time copies. You may want to use this technique in the following situations:

- To automate the loading of many, large copies of tables.
- To fully-refresh tables with referential integrity constraints, in order to bypass referential integrity checking.

Note: Before you begin, ensure that the Capture program is running.

The following example assumes that you want to perform a load of the tables within the replication subscription SET001 with an Apply program qualifier of APPLY001.

1. Optional: It is recommended that you disable the full refresh capability for the applicable source tables as shown in the following SQL statements; the Apply program then issues an error message, rather than performing a full refresh if the following procedure is not performed correctly.

```
UPDATE ASN.IBMSNAP_REGISTER SET DISABLE_REFRESH = 1
WHERE SOURCE_OWNER = 'source owner' AND
SOURCE_TABLE = 'source table'
```

2. Ensure that the Apply program is inactive, or deactivate the applicable replication subscription as shown in the following SQL statements:

```
UPDATE ASN.IBMSNAP_SUBS_SET SET ACTIVATE = 0
WHERE SET_NAME = 'SET001' AND APPLY_QUAL = 'APPLY001'
```

3. Update SYNCHPOINT and SYNCHTIME in ASN.IBMSNAP_PRUNCNTL for each row that corresponds to a member of the replication subscription as shown in the following SQL statements:

```
UPDATE ASN.IBMSNAP_PRUNCNTL
SET SYNCHPOINT = x'00000000000000000000',
SYNCHTIME = CURRENT_TIMESTAMP
WHERE SET_NAME = 'SET001' AND APPLY_QUAL = 'APPLY001'
```

4. Verify that the Capture program has processed the updates in the step above.
 - If the Capture program has not started capturing changes for these tables, this update initiates the process. ASN.IBMSNAP_TRACE will contain GOCAPT messages for the source tables.
 - If the Capture program has previously started capturing changes for these tables (other subscriptions exist for the tables and these have already been activated), you can verify that this pruning control table update was successful

by querying the same SYNCHPOINT values that were updated to hex zeroes. When these values no longer appear as zeroes, the Capture program has performed the necessary processing for you to continue.

5. Unload the source table and load the target table, using DB2 or vendor utility programs.

6. Update ASN.IBMSNAP_SUBS_SET as shown in the following SQL statements:

```
UPDATE ASN.IBMSNAP_SUBS_SET
SET LASTRUN = CURRENT_TIMESTAMP,
LASTSUCCESS = CURRENT_TIMESTAMP,
SYNCHTIME = CURRENT_TIMESTAMP,
SYNCHPOINT = NULL
WHERE SET_NAME = 'SET001' AND APPLY_QUAL = 'APPLY001'
```

7. Start the Apply program, or set ACTIVATE = 1 for the applicable replication subscription row in ASN.IBMSNAP_SUBS_SET as shown in the following SQL statements:

```
UPDATE ASN.IBMSNAP_SUBS_SET SET ACTIVATE = 1
WHERE SET_NAME = 'SET001' AND APPLY_QUAL = 'APPLY001'
```

Note: In cases where you are trying to identify problems by manually resetting values in the PRUNCNTL table, note that the PRUNCNTL table's SYNCPPOINT values need to match the CD_OLD_SYNCHPOINT values in the REGISTER table. If the values do not match, then the Apply program will want to perform a full refresh.

Troubleshooting

The following sections describe commonly known issues with Capture and Apply for Windows NT and Windows 95 programs.

Problems Using the Capture Program

Problem

The Capture for Windows NT program is not capturing updates.

Any of the following could prevent the Capture program from capturing updates:

- Proper authorization was not granted to the user ID running the Capture program. The Capture program requires DBADM or SYSADM privileges.
- DATA CAPTURE CHANGES was not specified on the base tables to be captured.
- 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 ASN.IBMSNAP_TRACE table.
 3. Start the Apply program.

Check the ASN.IBMSNAP_TRACE table for possible error messages.

Problem

I'm not sure if the Capture program is running successfully.

The first time 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 ASN.IBMSNAP_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 change data tables. They are eventually applied to target tables and pruned from the change data tables. After the Capture program runs for some time, you should see rows in the change data tables if changes are made to the sources. Periodically, check the ASN.IBMSNAP_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 ASN.IBMSNAP_APPLYTRAIL table.

Problem

Capture for Windows NT terminates.

The Capture for Windows NT 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 361 for an explanation of this message.

Problems Using the Apply Program

Before running the Apply program, ensure that:

- Replication sources and subscriptions have been defined.
- The Capture program has been started, and the ASN0100I initialization message has been issued (if you are running a Capture program).
- Bind has been performed for all control, source, and target servers that the Apply program accesses.

When an error occurs while running the Apply program, the status field in ASN.IBMSNAP_SUBS_SET is set to -1 for that copy definition.

Errors as well as successful executions are logged in the ASN.IBMSNAP_APPLYTRAIL table. APPERRM contains the message text. Any SQL errors are also logged in this table. Error messages are also issued on the console.

Specific Apply program problems are described below.

Problem

I have 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 packages, and make sure to bind both the Apply program packages to the control, source, and target server databases.

Problem

The DB2 log has 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, described in Table 12 on page 165, or you can perform your own load, described in “Loading the Tables within a Subscription Set” on page 169.

If the error occurred while applying changed data, you can change the data blocking parameter to break down large blocks of changed data. See “Specifying Mini-Cycles for the Apply Program to Copy Committed Data” on page 116.

Problem

The Capture program 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 have 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 ASN.IBMSNAP_REGISTER at the source server for the source table. In this case, the Apply program issues message ASN1016E.

Problem

I received system error 1067 trying to start the Capture or Apply as NT Service.

Error code 1067 occurs under the following circumstances:

- You did not specify the userid and password to run under and replication is trying to run on the system account.
- You did not specify the ASNPATH environment variable correctly.
- 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 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 Apply trace file:

```
<ASNPATH pathname>APPLY<timestamp>.TRC
```

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 ASN.IBMSNAP_SUBS_SET to null.

Problem

I unsuccessfully tried to start a second Apply program instance.

You must run each instance with a unique Apply program qualifier.

Problem

I received error ASN1003 with SQLCODE = -1032 and SQLSTATE = 57019.

You must start the database manager before invoking the Apply program.

Problem

The Apply components for DB2 Universal Database stop with an SQLCODE= -330, SQLSTATE=22517, "A string cannot be used, because its characters cannot be translated".

When copying between DB2 for MVS 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 target server into the pruning control table on a DB2 for MVS source server.

For example, if you use the Korean character set with mixed data at both DB2 for MVS source server and 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.

Important: If you are running in a mixed environment, ensure you have installed the latest maintenance for the CCSID support of your DB2 for MVS program.

For more information on character translation, see the Character Conversion for Distributed Data chapter in *DB2 Version 4 Administration Guide, Volumes 1,2*.

| ***Problem***

| *I received a security violation message, and the Apply program is not authorized to*
| *connect to the database.*

| The control server name, userid, and password definitions must match exactly those
| specified in the password file, and are case sensitive. Check your definitions again.

Chapter 12. Capture and Apply for OS/2

This chapter describes how to set up, operate, and troubleshoot Capture and Apply for OS/2.

Read the following sections before reading the sections on operating the Capture and Apply programs.

- “Setting Up the Capture and Apply Programs”
- “Specifying Tuning Parameters for the Capture Program” on page 177
- “Restrictions When Running the Capture Program” on page 178
- “Authorization for Running the Capture Program” on page 178

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.

Configuring the Capture Program for OS/2

1. Ensure that the user ID under which the Capture program is running has the required privileges:
 - Execute privilege on the Capture packages
 - DBADM or SYSADM privileges for the source, control, and target servers

2. Log on with the user ID that has sufficient privileges.

3. Connect to the source server database by entering:

```
DB2 CONNECT TO database
```

where *database* is the source server database.

4. Prepare the source server database (source 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 UPDATE DATABASE CONFIGURATION FOR database_alias USING USEREXIT ON
DB2 BACKUP DATABASE database_alias
```

Note: You might need to increase DBHEAP, APPLHEAPSZ, and LOGBUFSZ based on your installation requirements.

5. Change to the directory where the Capture program bind files are located, which is usually *drive*:\SQLLIB\BND.

6. 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 can be found in the CAPTURE.LST file.

Configuring the Apply Program for OS/2

1. Ensure that the user ID under which Apply is running has the required privileges:

- Execute privilege for Apply packages
- DBADM or SYSADM privileges for the database(s)

2. Log on with the user ID that has sufficient privileges.

3. Change to the directory where the Apply program bind files are located, which is usually in *drive:\SQLLIB\BND*.

4. Connect to the source server database by entering:

```
DB2 CONNECT TO database
```

where *database* is the source server database.

5. Create and bind the Apply 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 The list in cursor stability format.

UR The list in uncommitted read format.

These commands create a list of packages, the names of which can be found in the APPLYCS.LST and APPLYUR.LST files.

6. Connect to the target server database by entering:

```
DB2 CONNECT TO database
```

where *database* is the target server database.

7. Create and bind the Apply 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
```

Note: Because the Apply control tables use static SQL calls, the Apply bind process searches for nearly all of the control tables at each server that it is bound to, regardless of whether these control tables have any use or meaning at each server.

8. Repeat the connect and bind steps for each server that the Apply program connects to. You must bind the Apply program to the source, target, and control servers.

Specifying Tuning Parameters for the Capture Program

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

Retention limit

The number of minutes to keep the change data table rows and the UOW table rows. The default value is 10,800, which is 7 days. The rows are deleted up to where the changes have been applied.

Lag limit

The number of minutes the Capture program can be backlogged from the current local time before shutting itself down. The default value is 10,800 (which is 7 days). This value is higher for a busy system; therefore, a lower lag limit shuts down the Capture program. If the Capture program shuts itself down, you should perform a cold start if the database does not have or support an archive log.

Commit interval

The number of seconds to wait before issuing a COMMIT statement. The default value is 30 seconds. Set the interval smaller than the DB2 timeout interval if the Capture and Apply programs are running at the same time. This precaution helps to avoid locking overhead. 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.

Prune interval

The number of seconds to wait 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.

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

- Modify DPCNTL.* in the Control Center /sqlib/samples/repl directory before you define the first replication source for a database.
- Update the table with the following SQL statement after you create the tuning parameters table:

```
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
```

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 information on the structure of the tuning parameters table, see Chapter 19, “Table Structures” on page 295.

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:

- Cancel an existing replication source.
- Drop a replication source table.
- Make changes that affect the structure of source 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 are:

- The Capture program does not capture any changes made by DB2 utilities.
- Capture for OS/2 does not support large object.

Authorization for Running the Capture Program

The user ID that operates the Capture program should have DBADM or SYSADM authority.

Operating Capture for OS/2

The administrator user ID can use the commands in this section to operate Capture for OS/2. Enter the commands or key combinations from an OS/2 window.

This section explains how to perform the following Capture for OS/2 tasks:

- Starting
- Scheduling
- Stopping
- Suspending
- Resuming
- Reinitializing
- Pruning
- Monitoring

Before You Start the Capture Program

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

- Define one or more replication sources and subscriptions as described in Chapter 8, “Working with Replication Sources” on page 93 and Chapter 9, “Working with Replication Targets” on page 103. This creates the following control tables that must be defined in the database where the Capture program is to run:
 - ASN.IBMSNAP_REGISTER
 - ASN.IBMSNAP_PRUNCNTL
 - ASN.IBMSNAP_CCPPARMS

- ASN.IBMSNAP_TRACE
- ASN.IBMSNAP_WARM_START
- ASN.IBMSNAP_UOW
- ASN.IBMSNAP_CRITSEC
- ASN.IBMSNAP_SUBS_SET
- ASN.IBMSNAP_SUBS_MEMBR
- ASN.IBMSNAP_SUBS_STMTS
- ASN.IBMSNAP_SUBS_COLS
- ASN.IBMSNAP_SUBS_EVENT
- ASN.IBMSNAP_APPLYTRAIL

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. “Configuring the Capture Program for OS/2” on page 175 for more information.
- Enable the DATA CAPTURE CHANGES attribute for your source table using either the CREATE TABLE or ALTER TABLE statement. See Chapter 8, “Working with Replication Sources” on page 93 for more information.

Starting Capture for OS/2

After you start the Capture program, it runs continuously until you stop it or it detects an error.

To start the Capture program:

1. If you created one or more DB2 for OS/2 instances, use the SET command to set the DB2INSTANCE environment variable to the 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_name.CCP* is created in the directory from which the Capture program is invoked. This file is a log file for the messages issued by the Capture program; these messages are also recorded in the ASN.IBMSNAP_TRACE table.

2. To invoke the Capture program, enter the ASNCCP command from the OS/2 window where you issued the SET command. The syntax is:

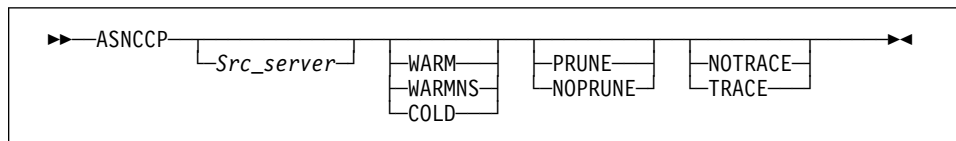


Table 13 on page 180 defines the optional parameters.

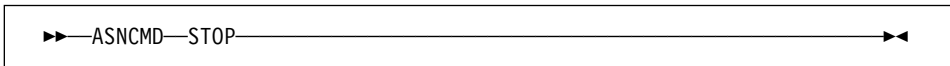
Table 13. ASNCCP Command Parameter Definitions for OS/2

Parameter	Definition
Src_server	Source server name positional parameter that must be the first parameter if entered. If not specified, the value from the DB2DBDFT environment variable will be 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. See “Warm and Cold Starts” on page 184 for more information.
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, change data, 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. See “Warm and Cold Starts” on page 184 for more information.
COLD	The Capture program starts by deleting all rows in its change data table, UOW table, and trace table during initialization. Refer to “Warm and Cold Starts” on page 184 for more information.
PRUNE (default)	The Capture program automatically prunes the change data and UOW tables.
NOPRUNE	Automatic pruning is disabled. The Capture program prunes the change data and UOW tables when you enter the PRUNE command. See the PRUNE command for more information.
NOTRACE (default)	No trace information is written.
TRACE	Writes debug trace messages to the standard output, <i>stdout</i> .

Scheduling Capture for OS/2

Use the Alarms program in the OS/2 Productivity set to start Capture for OS/2 at a specific time.

Stopping Capture for OS/2



or one of the following:

- Ctrl+C
- Ctrl+Break

Purpose

Use the STOP command or the 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 an existing replication source
- Opening an existing replication source and modifying the **Enable Refresh and Update** field

Usage

- To use the command, do the following from a different OS/2 window than where the Capture program is running:
 1. Set the environment variable DB2INSTANCE to the value set when the Capture program was started.
 2. Set the 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. Enter the command.
- If you want to use one of the key combinations, press it from the same OS/2 window in which the Capture program is running.

Stopping the Capture Program with a Trigger

In some situations, you might want to trigger the stop of the Capture program. For example, you might want to stop it at a specific time, or after all transactions for a particular application have been committed. The simplest way to set up this trigger is to create a table (or use an existing table), define it as a replication source, and identify a particular column whose update, when captured, triggers shutdown.

To stop the Capture program with a trigger:

Use the following example as a guideline.

1. Create a table with a column named STOPCAP or another unique identifier.
2. Define the table as a replication source.
3. Create an application that periodically reads the change data table associated with this replication source table to search for STOPCAP.
4. Schedule the application to start before your Apply programs.

When your application encounters STOPCAP in the change data table, it should issue the command to stop the Capture program.

Suspending Capture for OS/2



```
▶▶—ASNCMD—SUSPEND—◀◀
```

Purpose

Use the SUSPEND command to relinquish OS/2 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.

Usage

To use the command, do the following from a different window than where the Capture program is 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).
3. Enter the command.

Attention: Do not use SUSPEND when canceling a replication source. Instead, stop the Capture program using Ctrl+C.

Resuming Capture for OS/2

```
▶▶—ASN CMD—RESUME—————▶▶
```

Purpose

Use the RESUME command to restart the Capture program if you suspended it using the SUSPEND command.

Usage

To use the command, do the following from a different window than where the Capture program is 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).
3. Enter the command.

Reinitializing Capture for OS/2

```
▶▶—ASN CMD—REINIT—————▶▶
```

Purpose

Use the REINIT command to make the Capture program reread the ASN.IBMSNAP_REGISTER table or the ASN.IBMSNAP_CCPPARMS table while it is running. Use this command when you add a new replication source with the Control Center or when the values in the ASN.IBMSNAP_CCPPARMS table are changed while the Capture program is running.

This command ensures that the Capture program recognizes the new replication source. The Capture program also rereads the ASN.IBMSNAP_CCPPARMS table for any changes made to the tuning parameters.

Attention:

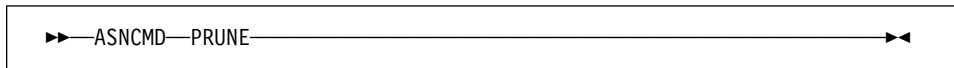
Do not use REINIT to reinitialize the Capture program after canceling a replication source or dropping a replication source table while the Capture program is running. Instead, stop the Capture program and restart it using the WARM or WARMNS option.

Usage

To use the command, do the following from a different window than where the Capture program is 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).
3. Enter the command.

Pruning the Change Data and Unit-of-Work Tables



Purpose

Use the PRUNE command to prune the CD and UOW tables if you used the NOPRUNE invocation parameter to disable pruning when you started the Capture program. This command prunes the change data (CD) and unit-of-work (UOW) tables once. The Capture program issues the message ASN0124I when the command is successfully queued.

Usage

To use the command, do the following from a different window than where the Capture program is 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).
3. Enter the command.

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.

Providing the Current Log Sequence Number

▶▶—ASNCMD—GETLSEQ————▶▶

Purpose

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.

Usage

To use the command, do the following from a different window than where the Capture program is 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).
3. Enter the command.

Warm and Cold Starts

This section explains how the Capture program handles warm starts, how it switches to an automatic cold start, and when you might want to force a warm start.

Warm Start Process

When you start the Capture program with the WARM or WARMNS parameter, it searches for the ASN.IBMSNAP_WARM_START table, which is created either during the first definition of a registration source or when DPCNTL.* is executed. This table contains information that enables the Capture program to quickly resynchronize to the time when it stopped. If this table is not available, the Capture program can resynchronize using either the register table, UOW table, or change data tables.

Warm start information is saved in most cases. In extreme cases, warm start information might not be saved. For example, an operator might cancel the Capture program or stop DB2. In this case, the Capture program uses the change data, UOW, or register table to resynchronize to the time it was stopped.

After a successful warm start, the old rows in the warm start table are deleted.

The Capture program switches to a cold start if you did not specify WARMNS and the warm start log sequence number is not available in the log. The Apply program performs a full refresh after a cold start. For information about handling gap messages, see “Problems Using the Apply Program” on page 193 and read the section about forcing the Apply program to perform a full refresh.)

Automatic Cold Starts

Sometimes the Capture program automatically switches to a cold start, even when you specify a warm start. The switch is made when:

- The warm start log sequence lags behind the current log sequence by more than the LAG_LIMIT value as specified in the tuning parameters table ASN.IBMSNAP_CCPPARMS.
- You invoke the Capture program for the first time.

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

For each of these cases, the Capture program issues an informational message and performs a cold start.

Forcing a Warm Start

You might want to prevent the Capture program from cold starting in some situations. For instance, the Capture program cold starts if DB2 goes down, or if someone brings down the DB2 table space containing the change data table. Forcing a warm start with the WARMNS parameter ensures that the control tables remain intact. You must correct the problem that caused the Capture program to terminate. If you do not correct the problem, the Capture program continues to terminate every time you start it.

Operating Apply for OS/2

An administrator can use the commands in the following sections to perform the following Apply for OS/2 tasks:

- Starting
- Scheduling
- Stopping

Before You Start the Apply Program

Before you start the Apply program, ensure that:

- The control tables are defined.
- At least one subscription is created and activated.
- The Apply for OS/2 package is created. See “Configuring the Apply Program for OS/2” on page 176 for information on bind programs to create the Apply for OS/2 package.
- You have the proper authorization. See “Authorization Requirements for the Apply Program” on page 51 for information about authorization for the Apply program.

Starting Apply for OS/2

When you start the Apply program, it runs continuously until:

- You stop it in an orderly way.
- An unexpected error or failure occurs.

To start the Apply program:

Perform the following steps from an OS/2 window:

1. Log on with the IBM Replication user ID.
2. Ensure that you set the DB2 instance name as shown:
SET DB2INSTANCE=*db2_instance_name*
3. Enter the ASNAPPLY command and one or more options, if desired, from the OS/2 window.

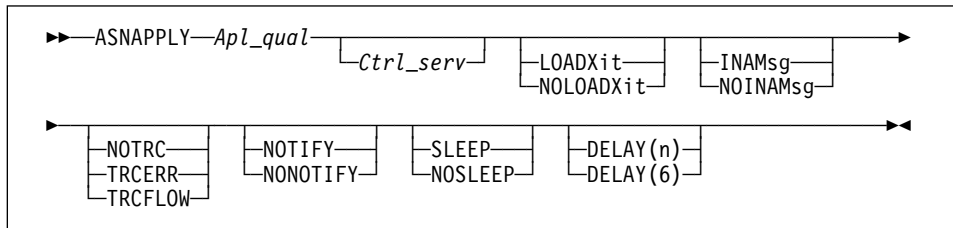


Table 14 on page 187 defines the optional positional parameters.

Table 14. ASNAPPLY Parameter Definitions for OS/2

Parameter	Definition
<i>Apl_qual</i>	Specifies the Apply program qualifier 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 is a positional parameter that is required as the first parameter.
<i>Ctrl_serv</i>	Specifies the name of the server where the replication control tables will reside. If you specify this optional positional parameter, it must be the second parameter. If you do not specify this parameter, the default is the default database or the value of DB2DBDFT for DB2 for OS/2.
LOADXit	Specifies that the Apply program is to invoke ASNLOAD, a program that can call an IBM or multivendor utility, to initialize a target table. This keyword parameter is optional.
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. This keyword parameter is optional.
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. This keyword parameter is optional.
TRCFLOW	Specifies that the Apply program generates a trace that contains error and execution flow information. This keyword parameter is optional.
NOTIFY	Specifies that the Apply program is to invoke ASNDONE, an exit program that returns control to the user when the Apply program processing ends. This is an optional keyword parameter.
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. This keyword parameter is optional.
DELAY (<i>n</i>)	Where <i>n</i> =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. This keyword parameter is optional.
DELAY (6) (default)	Specifies a delay time of 6 seconds at the end of each Apply program cycle when continuous replication is used.

When Apply for OS/2 executes, it generates two files:

- *userid.LOG*

This file is created in the directory where Apply for OS/2 is invoked. It contains messages issued by Apply for OS/2.

- *userid.SPL*

This file is created in the directory where Apply for OS/2 is invoked. It is an internal file used by Apply for OS/2. Do not erase it.

Scheduling Apply for OS/2

Use the Alarms program in the OS/2 Productivity set to schedule the Apply program to start at a specific time.

Stopping Apply for OS/2

Use the following command to stop Apply for OS/2.

```
▶▶—ASNASTOP—Apply_qualifier————▶▶
```

You can alternatively use one of the following key combinations from the window where the Apply program is running:

- Ctrl+C
- Ctrl+Break

Purpose

Use the ASNASTOP command or the key combination to stop the Apply program in an orderly way and commit the log records that it processed up to that point.

Usage

To use the command, do the following from a different window than where the Apply program is 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.

If you want to use one of the key combinations, press it from the same window in which the Apply program is running.

Additional Apply Program Operations

The following section provides information about performing the additional Apply program functions of scheduling subscriptions, refreshing target tables, returning control to users, initiating a forward recovery, and loading tables within a subscription set.

Scheduling Subscriptions with the Event Table

You can include an event name in your replication subscription. Your subscription cycles can run according to a timer, an event occurrence, or both. See “Setting the Copying Schedule: Time or Event Based” on page 114 for more information about the purpose and usage of event scheduling.

You can schedule a subscription by manipulating the event table. The DDL statement for the event table appears as follows:

```
CREATE TABLE
  ASN.IBMSNAP_SUBS_EVENT (
    EVENT_NAME CHAR(18) NOT NULL,
    EVENT_TIME TIMESTAMP NOT NULL,
    END_OF_PERIOD TIMESTAMP);
CREATE UNIQUE INDEX ASN.IBMSNAP_SUBS_EVENT
  ON ASN.IBMSNAP_SUBS_EVENT (EVENT_NAME ASC, EVENT_TIME ASC);
```

For example, to start the Apply program at 6:00 p.m. on 9/24/97 and copy changes from the source server up to 4:00 p.m. on the same date, you would insert the following values in ASN.IBMSNAP_SUBS_EVENT:

```
('EVENTA', '1997-09-24-18.00.00.000000', '1997-09-24-16.00.00.000000'),
```

A subscription becomes eligible for processing when EVENT_TIME falls between LASTSUCCESS and the control server's CURRENT_TIMESTAMP. The END_OF_PERIOD column enables you to specify a timestamp upper limit for change data.

Refreshing Target Tables with ASNLOAD

ASNLOAD is a user exit program that the Apply program invokes when you specify the LOADX invocation parameter. The Apply program calls the ASNLOAD program whenever it performs a full refresh of a target table.

You can use the ASNLOAD program as shipped with the Apply program, or you can modify it. As shipped, ASNLOAD 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. See the prolog section in the sample program (ASNLOAD.SMP) for instructions on how to modify the program to meet the requirements of your installation.

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

Files Generated

If you invoke ASNLOAD, the following files are generated:

- *<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 ASNLOAD, or if ASNLOAD returns a nonzero return code, the Apply program issues a message, stops processing that subscription, and processes the next subscription.

Returning Control to Users with ASNDONE

If you specify the NOTIFY parameter when starting the Apply program, the user exit program ASNDONE is called after subscription processing completes, regardless of success or failure. You can modify ASNDONE to meet the requirements of your installation. For example, the user exit can examine the UOW table to discover rejected transactions and initiate further actions, such as issuing a message or generating an alert.

See the prolog section in the sample program (ASNDONE.SMP) for instructions on how to modify the program.

Initiating a Forward Recovery with the Apply Program

In cases of an incomplete rollback or partially restored backup, you might need to copy a “window” of changes for completion.

To use the Apply program to limit the range of changes to those excluded from the rollback:

1. Determine how much data is lost.
Check the IBMSNAP_LOGMARKER value in the point-in-time table. The highest value is the most recently committed change.
2. Use SQL to identify the range of changes. Either:
 - Use the WHERE clause on the Rows page of the Advanced Subscription Definition notebook.
 - or
 - Manually update the PREDICATES column of the control server subscriptions table (ASN.IBMSNAP_SUBS_MEMBR).

For example, if your timestamp shows the last committed change to be 941106174322, your SQL should reflect changes that occur after that time.

Loading the Tables within a Subscription Set

The following section describes how to load the tables of a subscription set as an alternative to using the LOADX option when running the Apply program. This technique involves performing your own full refresh on behalf of the Apply program. By doing your

own full refresh, the Apply program assumes that *it* has initialized your point-in-time copies. You may want to use this technique in the following situations:

- To automate the loading of many, large copies of tables.
- To fully-refresh tables with referential integrity constraints, in order to bypass referential integrity checking.

Note: Before you begin, ensure that the Capture program is running.

The following example assumes that you want to perform a load of the tables within the replication subscription SET001 with an Apply program qualifier of APPLY001.

1. Optional: It is recommended that you disable the full refresh capability for the applicable source tables as shown in the following SQL statements; the Apply program then issues an error message, rather than performing a full refresh if the following procedure is not performed correctly.

```
UPDATE ASN.IBMSNAP_REGISTER SET DISABLE_REFRESH = 1
WHERE SOURCE_OWNER = 'source owner' AND
SOURCE_TABLE = 'source table'
```

2. Ensure that the Apply program is inactive, or deactivate the applicable replication subscription as shown in the following SQL statements:

```
UPDATE ASN.IBMSNAP_SUBS_SET SET ACTIVATE = 0
WHERE SET_NAME = 'SET001' AND APPLY_QUAL = 'APPLY001'
```

3. Update SYNCHPOINT and SYNCHTIME in ASN.IBMSNAP_PRUNCNTL for each row that corresponds to a member of the replication subscription as shown in the following SQL statements:

```
UPDATE ASN.IBMSNAP_PRUNCNTL
SET SYNCHPOINT = x'00000000000000000000',
SYNCHTIME = CURRENT_TIMESTAMP
WHERE SET_NAME = 'SET001' AND APPLY_QUAL = 'APPLY001'
```

4. Verify that the Capture program has processed the updates in the step above.
 - If the Capture program has not started capturing changes for these tables, this update initiates the process. ASN.IBMSNAP_TRACE will contain GOCAPT messages for the source tables.
 - If the Capture program has previously started capturing changes for these tables (other subscriptions exist for the tables and these have already been activated), you can verify that this pruning control table update was successful by querying the same SYNCHPOINT values that were updated to hex zeroes. When these values no longer appear as zeroes, the Capture program has performed the necessary processing for you to continue.
5. Unload the source table and load the target table, using DB2 or vendor utility programs.
6. Update ASN.IBMSNAP_SUBS_SET as shown in the following SQL statements:

```

UPDATE ASN.IBMSNAP_SUBS_SET
SET LASTRUN = CURRENT_TIMESTAMP,
LASTSUCCESS = CURRENT_TIMESTAMP,
SYNCHTIME = CURRENT_TIMESTAMP,
SYNCHPOINT = NULL
WHERE SET_NAME = 'SET001' AND APPLY_QUAL = 'APPLY001'

```

7. Start the Apply program, or set ACTIVATE = 1 for the applicable replication subscription row in ASN.IBMSNAP_SUBS_SET as shown in the following SQL statements:

```

UPDATE ASN.IBMSNAP_SUBS_SET SET ACTIVATE = 1
WHERE SET_NAME = 'SET001' AND APPLY_QUAL = 'APPLY001'

```

Note: In cases where you are trying to identify problems by manually resetting values in the PRUNCNTL table, note that the PRUNCNTL table's SYNCPOINT values need to match the CD_OLD_SYNCPOINT values in the REGISTER table. If the values do not match, then the Apply program will want to perform a full refresh.

Troubleshooting

The following sections describe commonly known issues with the Capture and Apply for OS/2 programs.

Problems Using the Capture Program

Problem

Capture for OS/2 is not capturing updates.

Any of the following could prevent the Capture program from capturing updates:

- Proper authorization was not granted to the user ID running the Capture program. The Capture program requires DBADM or SYSADM privileges.
- DATA CAPTURE CHANGES was not specified on the base tables to be captured.
- 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 a message number ASN01001 (initialization completed) in the system console or in the ASN.IBMSNAP_TRACE table.
 3. Start the Apply program.

Check the ASN.IBMSNAP_TRACE table for possible error messages.

Problem

I'm not sure if the Capture program is running successfully.

The first time 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 ASN.IBMSNAP_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 change data tables. They are eventually applied to target tables and pruned from the change data tables. After the Capture program runs for some time, you should see rows in the change data tables if changes are made to the source tables. Periodically, check the ASN.IBMSNAP_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 ASN.IBMSNAP_APPLYTRAIL table.

Problem

Capture for OS/2 terminates.

The Capture for OS/2 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 361 for an explanation of this message.

Problems Using the Apply Program

Before running the Apply program, ensure that:

- Replication sources and subscriptions have been defined.
- The Capture program has been started, and the ASN0100I initialization message has been issued (if you are running a Capture program).
- Bind has been performed for all control, source, and target servers that the Apply program accesses.

When an error occurs while running the Apply program, the status field in ASN.IBMSNAP_SUBS_SET is set to -1 for that copy definition.

Errors as well as successful executions are logged in the ASN.IBMSNAP_APPLYTRAIL table. APPERRM contains the message text. Any SQL errors are also logged in this table. Error messages are also issued on the console.

Specific Apply program problems are described below.

Problem

I have 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 Apply program packages to the control, source, and target server databases.

Problem

The DB2 log has 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, described in Table 5, ASNARUN Invocation Parameter Definitions, or you can perform your own load.

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 Mini-Cycles for the Apply Program to Copy Committed Data” on page 116 in chapter 7.

Problem

The Capture program 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 have 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 ASN.IBMSNAP_REGISTER at the source server for the source table. In this case, the Apply program issues message ASN1016E.

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 ASN.IBMSNAP_SUBS_SET to null.

I unsuccessfully tried to start a second Apply program instance.

You must run each instance with a unique Apply program qualifier.

Problem

I received error ASN1003 with SQLCODE = -1032 and SQLSTATE = 57019.

You must start the database manager before invoking the Apply program.

Problem

Apply components 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 MVS 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. Apply 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 target server into the pruning control table on a DB2 for MVS source server.

For example, if you use the Korean character set with mixed data at both DB2 for MVS source server and 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.

Important: If you are running in a mixed environment, ensure you have installed the latest maintenance for the CCSID support of your DB2 for MVS program.

For more information on character translation, see the Character Conversion for Distributed Data chapter in *DB2 Version 4 Administration Guide, Volumes 1,2*.

| **Problem**

| *I received a security violation message, and the Apply program is not authorized to connect to the database.*

| The control server name, userid, and password definitions must match exactly those specified in the password file, and are case sensitive. Check your definitions again.

Chapter 13. Capture and Apply for UNIX Platforms

This chapter describes how to set up, operate, and troubleshoot Capture and Apply for the following UNIX platforms:

- AIX
- HP-UX
- Solaris
- SCO UnixWare 7 (UnixWare 7)

Read the following sections before reading the sections on operating Capture and Apply.

- “Setting up a UNIX User Account”
- “Setting Up the Capture and Apply Programs”
- “Specifying Tuning Parameters for the Capture Program” on page 201
- “Restrictions When Running the Capture Program” on page 202
- “Authorization for Running the Capture Program” on page 202

Setting up a UNIX User Account

Before you set up the Capture and Apply programs, you must set up a UNIX user account to run the programs. To set up a user account, add the following two statements to your `.profile`:

```
export SHLIB=$SHLIB:/db2dir
export LD_LIBRARY_PATH=$LD_LIBRARY_PATH:/db2dir
```

Where `db2dir` is the library directory where DB2 is installed. For example, if you're working in a Solaris environment, add the following statements to your `.profile`:

```
export SHLIB=$SHLIB:/opt/ibmdb2/v5.0/lib
export LD_LIBRARY_PATH=$LD_LIBRARY_PATH:/opt/IBDb2/v5.0/lib
```

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

1. Ensure that the user ID under which the Capture program is running has the required privileges:
 - Execute privilege on the Capture packages
 - DBADM or SYSADM privileges for the source, control, and target servers
2. Log on with the user ID that has sufficient privileges.
3. Connect to the source server database by entering:
`db2 connect to database`

where *database* is the source server database.

4. Prepare the source server database (source 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 update database configuration for database_alias using userexit on
db2 backup database database_alias
```

Note: You might need to increase DBHEAP, APPLHEAPSZ, and LOGBUFSZ based on your installation requirements.

5. Change to the directory where the Capture program bind files are located, which is usually *\$HOME/sqllib/bnd*.
6. 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.

Configuring the Apply Program for UNIX Platforms

1. Ensure that the user ID under which Apply is running has the required privileges:
 - Execute privilege for Apply packages
 - DBADM or SYSADM privileges for the database

2. Log on with the user ID that has sufficient privileges.

3. Change to the directory where the Apply program bind files are located, which is usually *\$HOME/sqllib/bnd*.

4. Connect to the source server database by entering:

```
DB2 CONNECT TO database
```

where *database* is the source server database.

5. Create and bind the Apply 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 The list in cursor stability format.

UR The list in uncommitted read format.

These commands create a list of packages, the names of which can be found in the *applycs.lst* and *applyur.lst* files.

6. Connect to the target server database by entering:

```
db2 connect to database
```

where *database* is the target server database.

7. Create and bind the Apply 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
```

Note: Because the Apply control tables use static SQL calls, the Apply bind process searches for nearly all of the control tables at each server that it is bound to, regardless of whether these control tables have any use or meaning at each server.

8. Repeat the connect and bind steps for each server that the Apply program connects to. You must bind the Apply program to the 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:

- `<instname><SRCSRVR>.CCP`- A log file for the messages issued by the Capture program; these messages are also recorded in the trace table (`asn.ibmnap_trace`).
- `<instname><SRCSRVR>.TMP`- Contains process ID of this invocation of Capture (to prevent multiple Capture programs from being invoked in the same instance to the same source server).

The Apply program creates the following files:

- `<APPLYQUAL><instname><CNTLSRVR>.LOG`- A log file for the messages issued by the Apply program; contains the same information as that in the apply trail table (`asn.ibmnap_applytrail`).
- `ASNAPPLY<APPLYQUAL>.PID`- Contains process ID of this invocation of Apply (to prevent multiple Apply programs from being invoked with the same Apply qualifier).

For all other issues pertaining to configuration of UNIX-based components, see *IBM DB2 Universal Database for UNIX Quick Beginnings*.

Providing End-User Authentication at the Source Server

For end-user authentication to occur at the source server, in some cases you need to provide a password file for Apply to use when connecting to the source server. Following are environment-specific changes:

- If you installed Apply for HP-UX, Apply for Solaris, or Apply for UnixWare 7, you must use the `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 the `AUTHENTICATION=SERVER` scheme. If you use the `AUTHENTICATION=CLIENT` scheme, you do not need to provide a password file.

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

If you create a password file:

The password file must meet the following criteria:

- Be named as shown:

`<APPLYQUAL><instname><CNTLSRVR>.PWD`

Where:

APPLYQUAL

The Apply qualifier in upper case. The Apply qualifier is case sensitive and must match the value of APPLY_QUAL in the subscription set table.

instname

The instance name in lower case that Apply runs under. The value of DB2INSTANCE.

CNTLSRVR

The name of the control server in upper case.

For example: DATADIRApply1REPRTDB.PWD

Note that this naming convention is the same as the log file name (.LOG) and the spill file name (.SPL), but with a file extension of PWD.

- Reside in the directory from which the Apply program starts.
- Contain all server-name/password pairs for the file. This enables you to use a different (or the same) password at each server. The Apply user ID is used for all connections.
- Have one or more records using the following format:

`SERVER=server_name PWD=password USER=userid`

Where *server_name* is the source, target, or control database. The file cannot include blank lines or comment lines.

It is advisable to limit read access of this file to the user ID that will run Apply.

If you do not create a password file:

Apply for UNIX-based platforms must be able to issue an SQL CONNECT statement without specifying the user ID and password. To do so, ensure that:

- The DB2 for MVS database is catalogued as AUTHENTICATION=CLIENT.
- The login ID belongs to PRIMARY GROUP=SYSTEM.

When copying from DB2 for MVS sources, ensure that:

- SECURITY=SAME for MVS CPI-C node.

- You specify the following values when you define the LU name via 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.

Specifying Tuning Parameters for the Capture Program

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

Retention limit

The number of minutes to keep the change data table rows and the unit-of-work (UOW) table rows. The default value is 10,800, which is 7 days. The rows are deleted up to where the changes have been applied.

Lag limit

The number of minutes the Capture program can be backlogged from the current local time before shutting itself down. The default value is 10,800 (which is 7 days). This value is higher for a busy system; therefore, a lower lag limit shuts down the Capture program. If Capture shuts itself down, you should perform a cold start if the database does not have or support an archive log.

Commit interval

The number of seconds to wait before issuing a COMMIT statement. The default value is 30 seconds. Set the interval smaller than the DB2 timeout interval if Capture and Apply are running at the same time. This precaution helps to avoid locking overhead. If Apply is *not* running at the same time as the Capture program, you can set the commit interval no higher than the DB2 timeout interval.

Prune interval

The number of seconds to wait 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 Capture with the NOPRUNE option.

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

- Modify DPCNTL.* in the Control Center /sqlib/samples/repl directory before you define the first replication source for a database.
- Update the table with the following SQL statement after you create the tuning parameters table:

```
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
```

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 information on the structure of the tuning parameters table, see Chapter 19, “Table Structures” on page 295.

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:

- Cancel an existing replication source.
- Drop a replication source table.
- Make changes that affect the structure of source 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 are:

- Capture cannot capture any changes made by DB2 utilities, since the utilities don't log changes they make.
- Capture for UNIX platforms does not support large objects.

Authorization for Running the Capture Program

The user ID that operates the Capture program should either have DBADM or SYSADM authority.

Operating Capture for AIX, Capture for HP-UX, Capture for Solaris, and Capture for UnixWare 7

The administrator user ID can use the commands in this section to operate Capture for UNIX platforms. Enter the commands or key combinations from an AIX, HP-UX, Solaris, or UnixWare 7 window.

This section explains how to perform the following Capture tasks:

- Starting
- Scheduling
- Stopping
- Suspending
- Resuming
- Reinitializing
- Pruning
- Monitoring

Before You Start the Capture Program

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

- Define one or more replication sources and subscriptions as described in Chapter 8, “Working with Replication Sources” on page 93 and Chapter 9,

“Working with Replication Targets” on page 103. This creates the following control tables that must be defined in the database where Capture is to run:

- ASN.IBMSNAP_REGISTER
- ASN.IBMSNAP_PRUNCNTL
- ASN.IBMSNAP_CCPPARMS
- ASN.IBMSNAP_TRACE
- ASN.IBMSNAP_WARM_START
- ASN.IBMSNAP_UOW
- ASN.IBMSNAP_CRITSEC
- ASN.IBMSNAP_SUBS_SET
- ASN.IBMSNAP_SUBS_MEMBR
- ASN.IBMSNAP_SUBS_STMTS
- ASN.IBMSNAP_SUBS_COLS
- ASN.IBMSNAP_SUBS_EVENT
- ASN.IBMSNAP_APPLYTRAIL

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 servers from which the Capture program will capture changes. See “Configuring the Capture Program for UNIX Platforms” on page 197 for more information.
- Enable the DATA CAPTURE CHANGES attribute for your source table using either the CREATE TABLE or ALTER TABLE statement. See Chapter 8, “Working with Replication Sources” on page 93 for more information.

Starting Capture for AIX, Capture for HP-UX, Capture for Solaris, and Capture for UnixWare 7

After you start Capture, it runs continuously until you stop it or it detects an error.

To start Capture for AIX, Capture for HP-UX, Capture for Solaris, and Capture for UnixWare 7:

1. From AIX, HP-UX, Solaris, or UnixWare 7, log in and make sure that the user ID under which Capture is running has write privilege on the directory.
2. Ensure that you set the DB2 instance 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 invoked. This file is a log file for the messages issued by the Capture program; these messages are also recorded in the ASN.IBMSNAP_TRACE table.

3. Set the LIBPATH environment variable in the same environment in which the Capture program starts.

AIX example:

```
export LIBPATH=~/.sqllib/lib:/usr/lib:/lib
```

HP-UX example:

```
export SHLIB_PATH=~/sqllib/lib:/usr/lib:/lib
```

Solaris and UnixWare 7 example:

```
export LD_LIBRARY_PATH=~/sqllib/lib:/usr/lib:/lib
export NLS_PATH=~/usr/lib/locale/%L/%N:/usr/lib/locale/prime/%N
```

4. Enter the following command:

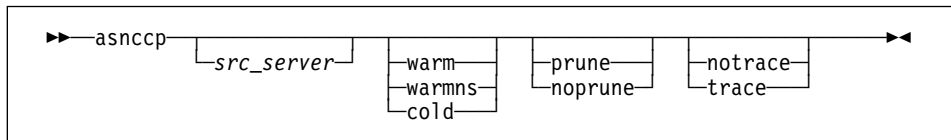


Table 15 defines the optional parameters.

Table 15. ASNCCP Command Parameter Definitions for UNIX Platforms

Parameter	Definition
src_server	Source server name positional parameter that must be the first parameter if entered. If not specified, the value from the DB2DBDFT environment variable will be 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. See “Warm and Cold Starts” on page 209 for more information.
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, Capture terminates instead of switching to a cold start as when warm is specified. See “Warm and Cold Starts” on page 209 for more information.
cold	The Capture program starts by deleting all rows in its CD table, UOW table, and trace table during initialization. Refer to “Warm and Cold Starts” on page 209 for more information.
prune (default)	The Capture program automatically prunes the CD tables and the UOW table.
noprune	Automatic pruning is disabled. The Capture program prunes the CD tables and the UOW table when you enter the prune command. See the PRUNE command for more information.
notrace (default)	No trace information is written.
trace	Writes debug trace messages to the standard output, <i>stdout</i> .

Scheduling Capture for AIX, Capture for HP-UX, Capture for Solaris, and Capture for UnixWare 7

Use the **at** command to start Capture at a specific time. For example, the following command starts Capture at 3:00 p.m. on Friday:

at 3pm Friday asncpp warmns noprun

Stopping Capture for AIX, Capture for HP-UX, Capture for Solaris, and Capture for UnixWare 7

```
▶▶—asncmd—stop—————▶▶
```

or

```
▶▶—kill—-INT—pid—————▶▶
```

or one of the following:

- Ctrl+C
- Ctrl+D

Purpose

Use the **stop** command, the **kill** command, or the key combinations to stop the Capture program in an orderly way and commit the log records that it processed up to that point.

Stop Capture before:

- Removing an existing replication source
- Opening an existing replication source and modifying the **Enable Refresh and Update** field

Usage

- If you want to use the **stop** command, do the following from a different window than where Capture is running:
 1. Export DB2INSTANCE to the value set when the Capture program was started.
 2. Export DB2DBDFT to the source server specified when the Capture program was started (or the DB2DBDFT value used when the Capture program was started).
 3. Enter **asncmd stop**.
- If you want to use the **kill** command, identify the process id of the Capture program and enter **kill -int pid**. You do not need to set the environment variables.

Stopping the Capture Program with a Trigger

In some situations, you might want to trigger the stop of the Capture program. For example, you might want to stop it at a specific time, or after all transactions for a particular application have been committed. The simplest way to set up this trigger is to create a table (or use an existing table), define it as a replication source, and identify a particular column whose update, when captured, triggers shutdown.

To stop the Capture program with a trigger:

Use the following example as a guideline.

1. Create a table with a column named STOPCAP or another unique identifier.
2. Define the table as a replication source.
3. Create an application that periodically reads the CD table associated with this replication source table to search for STOPCAP.
4. Schedule the application to start before your Apply programs.

When your application encounters STOPCAP in the CD table, it should issue the command to stop the Capture program.

Suspending Capture for AIX, Capture for HP-UX, Capture for Solaris, and Capture for UnixWare 7

```
▶▶—asnrcmd—suspend—————▶▶
```

Purpose

Use the **suspend** command to relinquish AIX, HP-UX, Solaris, or UnixWare 7 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.

Attention:

Do not use the **suspend** command when canceling a replication source. Instead, stop the Capture program.

Usage

To use the command, do the following from a different window than where the Capture program is 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).
3. Enter the command.

Resuming Capture for AIX, Capture for HP-UX, Capture for Solaris, and Capture for UnixWare 7

```
▶▶—asnrcmd—resume—————▶▶
```

Purpose

Use the **resume** command to restart the Capture program if you suspended it using the **suspend** command.

Usage

To use the command, do the following from a different window than where the Capture program is 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).
3. Enter the command.

Reinitializing Capture for AIX, Capture for HP-UX, Capture for Solaris, and Capture for UnixWare 7

```
▶▶—asnrcmd—reinit—————▶▶
```

Purpose

Use the **reinit** command to have the Capture program reread the ASN.IBMSNAP_REGISTER tables or the ASN.IBMSNAP_CCPPARMS table while it is running. Use this command when you add a new replication source with Control Center or when the values in the ASN.IBMSNAP_CCPPARMS table are changed while the Capture program is running.

This command ensures that the Capture program recognizes the new replication source. The Capture program also rereads the ASN.IBMSNAP_CCPPARMS table for any changes made to the tuning parameters.

Attention:

Do not use the **reinit** command to reinitialize the Capture program after canceling a replication source or dropping a replication source table while Capture is running. Instead, stop the Capture program and restart it using the WARM or WARMNS option.

Usage

To use the command, do the following from a different window than where the Capture program is 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).

3. Enter the command.

Pruning the Change Data and Unit-of-Work Tables

```
▶▶—asnrcmd—prune————▶▶
```

Purpose

Use the **prune** command to perform pruning of the change data (CD) table and unit-of-work (UOW) table if you disabled pruning by using the **no prune** invocation parameter while starting the Capture program.

This command prunes the CD and UOW tables once. The Capture program issues the message ASN0124I when the command is successfully queued.

Usage

To use the command, do the following from a different window than where the Capture program is 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).
3. Enter the command.

During pruning, if you stop or suspend Capture, pruning does not resume after you enter the **resume** command. You must enter the **prune** command again to resume pruning.

Providing the Current Log Sequence Number

```
▶▶—asnrcmd—getlseq————▶▶
```

Purpose

Use the **getlseq** command to provide the timestamp and current log sequence number. You can use this number to determine how far Capture has read the DB2 log.

Usage

To use the command, do the following from a different window than where the Capture program is 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).

3. Enter the command.

Warm and Cold Starts

This section explains how the Capture program handles warm starts, how it switches to an automatic cold start, and when you might want to force a warm start.

Warm Start Process

When you start the Capture program with the WARM or WARMNS parameter, it searches for the warm start table, ASN.IBMSNAP_WARM_START, which is created during the first definition of a registration source or when DPCNTL.* is executed. This table contains information that enables the Capture program to quickly resynchronize to the time when it stopped. If this table is unavailable, the Capture program can resynchronize using either the register table, UOW table, or CD tables.

Warm start information is saved in most cases. In extreme cases, warm start information might not be saved. For example, an operator might cancel the Capture program or stop DB2. In this case, the Capture program uses the CD, UOW, or register tables to resynchronize to the time it was stopped.

After a successful warm start, the old rows in the warm start table are deleted.

The Capture program switches to a cold start if you did not specify WARMNS and the warm start log sequence number is not available in the log. Apply performs a full refresh after a cold start. For information about handling gap messages, see “Problems Using the Apply Program” on page 219 and read the section about forcing the Apply program to perform a full refresh.

Automatic Cold Starts

Sometimes the Capture program automatically switches to a cold start, even when you specify a warm start. The switch is made when:

- The warm start log sequence lags behind the current log sequence by more than the LAG_LIMIT value, as specified in the tuning parameters table ASN.IBMSNAP_CCPPARMS.
- You invoke the Capture program for the first time.

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

For each of these cases, the Capture program issues an informational message and performs a cold start.

Forcing a Warm Start

You might want to prevent the Capture program from cold starting in some situations. For instance, the Capture program cold starts if DB2 goes down, or if someone brings down the DB2 table space containing the CD table. Forcing a warm start with the

WARMNS parameter ensures that the control tables remain intact. You must correct the problem that caused the Capture program to terminate. If you do not correct the problem, the Capture program continues to terminate every time you start it.

Operating Apply for AIX, Apply for HP-UX, Apply for Solaris, and Apply for UnixWare 7

An administrator can use the commands in the following sections to perform the following Apply tasks:

- Starting
- Scheduling
- Stopping

Before You Start the Apply Program

Before you start the Apply program, ensure that:

- The control tables are defined.
- 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 199 for more information.
- At least one subscription is created and activated.
- The Apply package is created. See “Configuring the Apply Program for UNIX Platforms” on page 198 for information on bind programs to create Apply packages for UNIX platforms.
- You have the proper authorization. See “Authorization Requirements for the Apply Program” on page 51 for information about authorization for the Apply program.

Starting Apply for AIX, Apply for HP-UX, Apply for Solaris, and Apply for UnixWare 7

After you start the Apply program, it runs continuously until:

- You stop it in an orderly way.
- An unexpected error or failure occurs.

To start Apply for AIX, Apply for HP-UX, Apply for Solaris, and Apply for UnixWare 7:

1. Ensure that you set the DB2 instance name as shown:

```
export DB2INSTANCE=db2_instance_name
```
2. Set the LIBPATH environment variable in the same environment in which the Apply program starts. For example, you can set LIBPATH with the following statement:

```
export LIBPATH= /sql1lib/lib:/usr/lib:/lib
```
3. From AIX, HP-UX, Solaris, or UnixWare 7, log on with the IBM Replication user ID. Make sure that the user ID under which Apply is running has write privilege on the directory.

Table 16. ASNAPPLY Parameter Definitions for UNIX Platforms

Parameter	Definition
<i>apl_qual</i>	Specifies the Apply program qualifier 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 is a positional parameter that is required as the first parameter.
<i>ctrl_serv</i>	Specifies the name of the server where the replication control tables will reside. If you do not specify this optional positional parameter, the default is the default database or the value in DB2DBDFT.
loadxit	Specifies that the Apply program is to invoke ASNLOAD, an exit program that can call an IBM or multivendor utility, to initialize a target table. This keyword parameter is optional.
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. This keyword parameter is optional.
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. This keyword parameter is optional.
trcflow	Specifies that the Apply program generates a trace that contains error and execution flow information. This keyword parameter is optional.
notify	Specifies that the Apply program is to invoke ASNDONE, an exit program that returns control to the user when the Apply program processing ends. This keyword parameter is optional.
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. This keyword parameter is optional.
delay (<i>n</i>)	Where <i>n</i> =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. This keyword parameter is optional.
delay (6) (default)	Specifies a delay time of 6 seconds at the end of each Apply program cycle when continuous replication is used.

When Apply for AIX, Apply for HP-UX, Apply for Solaris, or Apply for UnixWare 7 executes, it generates two files:

- `<userid><database_instance_name><database_name>.LOG`

This file contains the log messages issued by Apply during execution. Apply also logs messages to the ASN.IBMSNAP_APPLYTRAIL table. Check the columns APPERRM, SQLSTATE, SQLCODE, SQLERRP, and SQLERRM for information.

- `<userid><database_instance_name><database_name>.SPL`

This is an internal file used by Apply.

Scheduling Apply for AIX, Apply for HP-UX, Apply for Solaris, and Apply for UnixWare 7

Use the **at** command to start Apply at a specific time. For example, the following command starts Apply at 3:00 p.m. on Friday:

```
at 3pm Friday asnapply 1 1 r
```

Stopping Apply for AIX, Apply for HP-UX, Apply for Solaris, and Apply for UnixWare 7

Use the following command to stop Apply for AIX, Apply for HP-UX, Apply for Solaris, and Apply for UnixWare 7:

```
▶▶—asnastop—apply_qualifier—▶▶
```

or

```
▶▶—kill—-INT—pid—▶▶
```

You can alternatively use one of the following key combinations from the window where the Apply program is running:

- Ctrl+C
- Ctrl+Break

Purpose

Use the command or the key combination to stop the Apply program in an orderly way and commit the log records that it processed up to that point.

Usage

To use the command, do the following from a different window than where the Apply program is 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.

To use one of the key combinations, press it from the same NT window in which the Apply program is running.

Additional Apply Program Operations

The following section provides information about performing the additional Apply program functions of scheduling subscriptions, refreshing target tables, returning control to users, and initiating a forward recovery.

Scheduling Subscriptions with the Event Table

You can include an event name in your replication subscription. Your subscription cycles can run according to a timer, an event occurrence, or both. See “Setting the Copying Schedule: Time or Event Based” on page 114 for more information about the purpose and usage of event scheduling.

You can schedule a subscription by manipulating the event table. The DDL statement for the event table appears as follows:

```
CREATE TABLE
  ASN.IBMSNAP_SUBS_EVENT (
    EVENT_NAME CHAR(18) NOT NULL,
    EVENT_TIME TIMESTAMP NOT NULL,
    END_OF_PERIOD TIMESTAMP);
CREATE UNIQUE INDEX ASN.IBMSNAP_SUBS_EVENT
  ON ASN.IBMSNAP_SUBS_EVENT (EVENT_NAME ASC, EVENT_TIME ASC);
```

For example, to start the Apply program at 6:00 p.m. on 9/24/97 and copy changes from the source server up to 4:00 p.m. on the same date, you would insert the following values in ASN.IBMSNAP_SUBS_EVENT:

```
('EVENTA', '1997-09-24-18.00.00.000000', '1997-09-24-16.00.00.000000');
```

A subscription becomes eligible for processing when EVENT_TIME falls between LASTSUCCESS and the control server's CURRENT_TIMESTAMP. The END_OF_PERIOD column enables you to specify a timestamp upper limit for change data.

Refreshing Target Tables with ASNLOAD

ASNLOAD is a user exit program that the Apply program invokes when you specify the LOADX invocation parameter. The Apply program calls the ASNLOAD program whenever it performs a full refresh of a target table.

You can use the ASNLOAD program as shipped with the Apply program, or you can modify it. As shipped, ASNLOAD 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. See the prolog section in the sample program (ASNLOAD.smp) for instructions on how to modify the program to meet the requirements of your installation.

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

Files Generated

If you invoke ASNLOAD, the following files are generated:

- ASNA<userid><database_instance_name><database_name>.IXF

This file contains the data exported from the source.

- ASNAEXPT<userid><database_instance_name><database_name>.MSG

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

- ASNAIMPT<userid><database_instance_name><database_name>.MSG

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

Error Handling

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

Returning Control to Users with ASNDONE

If you specify the NOTIFY parameter when starting the Apply program, the user exit program ASNDONE is called after subscription processing completes, regardless of success or failure. You can modify ASNDONE to meet the requirements of your installation. For example, the user exit can examine the UOW table to discover rejected transactions and initiate further actions, such as issuing a message or generating an alert.

See the prolog section in the sample program (ASNDONE.smp) for instructions on how to modify the program.

Initiating a Forward Recovery with the Apply Program

In cases of an incomplete rollback or partially restored backup, you might need to copy a “window” of changes for completion.

To use the Apply program to limit the range of changes to those excluded from the rollback:

1. Determine how much data is lost.

Check the IBMSNAP_LOGMARKER value in the point-in-time table. The highest value is the most recently committed change.

2. Use SQL to identify the range of changes. Either:

- Use the WHERE clause on the Rows page of the Advanced Subscription Definition notebook.

or

- Manually update the PREDICATES column of the control server subscription table (ASN.IBMSNAP_SUBS_MEMBR).

For example, if your timestamp shows the last committed change to be 941106174322, your SQL should reflect changes that occur after that time.

Loading the Tables within a Subscription Set

The following section describes how to load the tables of a subscription set as an alternative to using the LOADX option when running the Apply program. This technique involves performing your own full refresh on behalf of the Apply program. By doing your own full refresh, the Apply program assumes that *it* has initialized your point-in-time copies. You may want to use this technique in the following situations:

- To automate the loading of many, large copies of tables.
- To fully-refresh tables with referential integrity constraints, in order to bypass referential integrity checking.

Note: Before you begin, ensure that the Capture program is running.

The following example assumes that you want to perform a load of the tables within the replication subscription SET001 with an Apply program qualifier of APPLY001.

1. Optional: It is recommended that you disable the full refresh capability for the applicable source tables as shown in the following SQL statements; the Apply program then issues an error message, rather than performing a full refresh if the following procedure is not performed correctly.

```
UPDATE ASN.IBMSNAP_REGISTER SET DISABLE_REFRESH = 1
WHERE SOURCE_OWNER = 'source owner' AND
SOURCE_TABLE = 'source table'
```

2. Ensure that the Apply program is inactive, or deactivate the applicable replication subscription as shown in the following SQL statements:

```
UPDATE ASN.IBMSNAP_SUBS_SET SET ACTIVATE = 0
WHERE SET_NAME = 'SET001' AND APPLY_QUAL = 'APPLY001'
```

3. Update SYNCHPOINT and SYNCHTIME in ASN.IBMSNAP_PRUNCNTL for each row that corresponds to a member of the replication subscription as shown in the following SQL statements:

```
UPDATE ASN.IBMSNAP_PRUNCNTL
SET SYNCHPOINT = x'00000000000000000000',
    SYNCHTIME = CURRENT_TIMESTAMP
WHERE SET_NAME = 'SET001' AND APPLY_QUAL = 'APPLY001'
```

4. Verify that the Capture program has processed the updates in the step above.
 - If the Capture program has not started capturing changes for these tables, this update initiates the process. ASN.IBMSNAP_TRACE will contain GOCAPT messages for the source tables.
 - If the Capture program has previously started capturing changes for these tables (other subscriptions exist for the tables and these have already been activated), you can verify that this pruning control table update was successful

by querying the same SYNCHPOINT values that were updated to hex zeroes. When these values no longer appear as zeroes, the Capture program has performed the necessary processing for you to continue.

5. Unload the source table and load the target table, using DB2 or vendor utility programs.

6. Update ASN.IBMSNAP_SUBS_SET as shown in the following SQL statements:

```
UPDATE ASN.IBMSNAP_SUBS_SET
SET LASTRUN = CURRENT_TIMESTAMP,
LASTSUCCESS = CURRENT_TIMESTAMP,
SYNCHTIME = CURRENT_TIMESTAMP,
SYNCHPOINT = NULL
WHERE SET_NAME = 'SET001' AND APPLY_QUAL = 'APPLY001'
```

7. Start the Apply program, or set ACTIVATE = 1 for the applicable replication subscription row in ASN.IBMSNAP_SUBS_SET as shown in the following SQL statements:

```
UPDATE ASN.IBMSNAP_SUBS_SET SET ACTIVATE = 1
WHERE SET_NAME = 'SET001' AND APPLY_QUAL = 'APPLY001'
```

Note: In cases where you are trying to identify problems by manually resetting values in the PRUNCNTL table, note that the PRUNCNTL table's SYNCPOINT values need to match the CD_OLD_SYNCHPOINT values in the REGISTER table. If the values do not match, then the Apply program will want to perform a full refresh.

Troubleshooting

The following sections describe commonly known issues with Capture and Apply for UNIX platforms.

Problems Using the Capture Program

Problem

The Capture program is not capturing updates.

Any of the following could prevent the Capture program from capturing updates:

- Proper authorization was not granted to the user ID running the Capture program. The Capture program requires DBADM or SYSADM privileges.
- DATA CAPTURE CHANGES was not specified on the base tables to be captured.
- 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 ASN.IBMSNAP_TRACE table.
 3. Start the Apply program.

Check the ASN.IBMSNAP_TRACE table for possible error messages.

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 asncpp 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 Capture program starts.

Problem

I'm not sure if the Capture program is running successfully.

The first time you start the Capture and Apply programs, the Apply program performs a full refresh to populate the target tables. Then Capture writes message ASN0104I to the ASN.IBMSNAP_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 Capture runs for some time, you should see rows in the CD tables if changes are made to the sources. Periodically, check the ASN.IBMSNAP_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 ASN.IBMSNAP_APPLYTRAIL table.

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 361 for an explanation of this message

Problems Using the Apply Program

Before running the Apply program, ensure that:

- Replication sources and subscriptions have been defined.
- The Capture program has been started, and the ASN0100I initialization message has been issued (if you are running a Capture program).
- Bind has been performed for all control, source, and target servers that the Apply program accesses.

When an error occurs while running the Apply program, the status field in ASN.IBMSNAP_SUBS_SET is set to -1 for that copy definition.

Errors as well as successful executions are logged in the ASN.IBMSNAP_APPLYTRAIL table. APPERRM contains the message text. Any SQL errors are also logged in this table. Error messages are also issued on the console.

Specific Apply program problems are described below.

Problem

I have 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 has 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, described in Table 5, ASNARUN Invocation Parameter Definitions, or you can perform your own load.

If the error occurred while applying changed data, you can change the data blocking parameter to break down large blocks of changed data. See “Specifying Mini-Cycles for the Apply Program to Copy Committed Data” on page 116.

Problem

The Capture program 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 have 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 ASN.IBMSNAP_REGISTER at the source server for the source table. In this case, the Apply program issues message ASN1016E.

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 ASN.IBMSNAP_SUBS_SET to null.

I unsuccessfully tried to start a second Apply program instance.

You must run each instance with a unique Apply program qualifier.

Problem

I received error ASN1003 with SQLCODE = -1032 and SQLSTATE = 57019.

You must start the database manager before invoking the Apply program.

Problem

Apply components 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 MVS 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 TARGET_TABLE column value from the refresh control table on a DB2 Universal Database copy server into the pruning control table on a DB2 for MVS source server.

For example, if you use the Korean character set with mixed data at both DB2 for MVS source server and 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 TARGET_TABLE length), the INSERT fails with an SQLCODE of -330.

Important: If you are running in a mixed environment, ensure you have installed the latest maintenance for the CCSID support of your DB2 for MVS program.

For more information on character translation, see the Character Conversion for Distributed Data chapter in *DB2 Version 4 Administration Guide, Volumes 1,2*.

Problem

I received a security violation message, and the Apply program is not authorized to connect to the database.

The control server name, userid, and password definitions must match exactly those specified in the password file, and are case sensitive. Check your definitions again.

| **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 asncpp 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.

Chapter 14. Capture for VSE

This chapter describes how to set up, operate, and troubleshoot Capture for VSE.

Read the following sections before reading the sections on operating the Capture program:

- “Setting Up the Capture Program”
- “Specifying Tuning Parameters for the Capture Program”
- “Restrictions When Running the Capture Program” on page 224
- “Authorization for Running the Capture Program” on page 225
- “Recovering from Severe Errors” on page 225

Setting Up the Capture Program

Setting up consists of installing the Capture for VSE program and configuring the source servers. All Capture for VSE program required tables must be defined prior to invoking the Capture program. These are created by the Control Center when a replication source is defined. You can also create these control tables manually by running the DPCNTL.* file from the RUN SQL Files window.

See the Capture for VSE program directory for instructions about installing 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 ASN.IBMSNAP_CCPPARMS tuning parameters table:

Retention limit

The number of minutes to keep the change data (CD) table rows and the unit-of-work (UOW) table rows. The default value is 10,800, which is 7 days. The rows are deleted up to where the changes have been applied.

Lag limit

The number of minutes the Capture program can be backlogged from the current local time before shutting itself down. The default value is 10,800, which is 7 days. This value is higher for a busy system; therefore, a lower lag limit shuts down the Capture program. If the Capture program shuts itself down, you should perform a cold start if the database log has been archived and the log has wrapped around.

Commit interval

The number of seconds to wait before issuing a COMMIT statement. The default value is 30 seconds. Set the interval smaller than the DB2 timeout interval if the Capture and Apply programs are running at the same time. If the Apply program is not running at the same time as Capture program, you can set the commit interval no higher than the DB2 timeout interval.

Prune interval

The number of seconds to wait 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 Capture program with the NOPRUNE option; however, you can override this option with the PRUNE command.

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

- Modify DPCNTL.* in the Control Center /sqlib/samples/repl directory before you define the first replication source for a database.
- Update the table with the following SQL statement after you create the tuning parameters table:

```
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
```

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 information on the structure of the tuning parameters table, see Chapter 19, “Table Structures” on page 295.

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:

- Cancel an existing replication source.
- Drop a replication source table.
- Make changes that affect the structure of source 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 are:

- Changes are not captured when the database is running with LOGMODE=N.
- One CD table exists per user table defined to the Capture program.
- You must specify DATA CAPTURE CHANGES on the CREATE or ALTER TABLE statement for the user tables.
- Tables with field procedures for columns (FIELDPROC specified on CREATE or ALTER TABLE) are not supported by Capture for VSE.
- There can be only one Capture program running per DB2 server for VSE database, and each Capture program runs in its own partition.

Authorization for Running the Capture Program

The user ID that operates the Capture program should have DBA authority.

When running Capture for VSE with the DB2 server for VSE, the Capture program job also dynamically creates a table (ASN.IBMSNAP_CCPENQ) to determine whether or not a Capture program is already running.

Recovering from Severe Errors

Capture for VSE provides you with the following tools to assist you if a severe error occurs:

- Trace buffer
- Trace output
- Storage dump

The following sections provide information on these tools.

Trace Buffer

Capture for VSE 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 termination. 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, IBM Replication writes trace information logic flow to STDOUT.

Storage Dump

When Capture for VSE 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.

Operating Capture for VSE

The administrator can use the commands in this section to operate Capture for VSE.

This section explains how to perform the following Capture for VSE tasks:

- Starting
- Stopping
- Suspending
- Resuming
- Reinitializing
- Pruning
- Monitoring

Before You Start the Capture Program

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

- Define one or more replication sources and subscriptions as described in Chapter 8, “Working with Replication Sources” on page 93 and Chapter 9, “Working with Replication Targets” on page 103. This creates the following control tables that must be defined in the database where the Capture program is to run:
 - ASN.IBMSNAP_REGISTER
 - ASN.IBMSNAP_PRUNCNTL
 - ASN.IBMSNAP_CCPPARMS
 - ASN.IBMSNAP_TRACE
 - ASN.IBMSNAP_WARM_START
 - ASN.IBMSNAP_UOW
 - ASN.IBMSNAP_CRITSEC
 - ASN.IBMSNAP_SUBS_SET
 - ASN.IBMSNAP_SUBS_MEMBR
 - ASN.IBMSNAP_SUBS_STMTS
 - ASN.IBMSNAP_SUBS_COLS
 - ASN.IBMSNAP_SUBS_EVENT
 - ASN.IBMSNAP_APPLYTRAIL
- Ensure that the administrator has DBA privilege.
- Ensure that ASN.IBMSNAP_REGISTER has at least one entry in it by defining a replication source with the DATA CAPTURE CHANGES option.
- Enable the DATA CAPTURE CHANGES attribute for your source tables using either the CREATE TABLE or ALTER TABLE statement. See Chapter 8, “Working with Replication Sources” on page 93 for more information.

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 program directory for a list of supported languages.

Starting Capture for VSE

After you start the Capture program, it runs continuously until you stop it or it detects an error.

To start the Capture program:

Start it in a partition like a batch job. Sample job control member ASNS51BD provides an example of how to start the Capture program. You can specify ASNCCP invocation parameters in the PARM field, in the order shown, separated by one or more blanks:

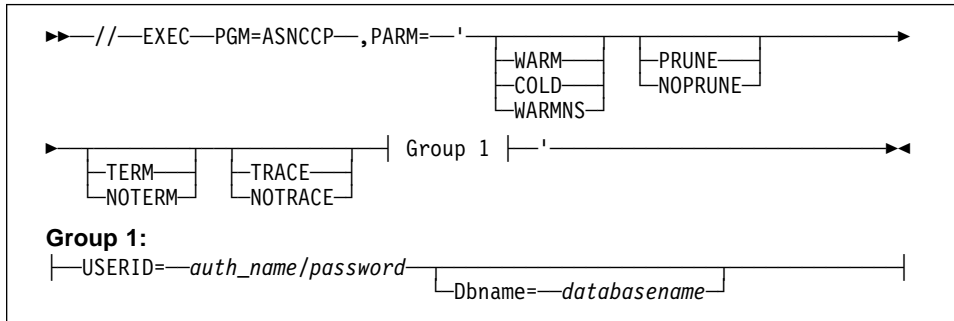


Table 17 on page 228 defines the optional parameters.

Table 17. ASNCCP Command Parameter Definitions for 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. See “Warm and Cold Starts” on page 162 for more information.
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. See “Warm and Cold Starts” on page 162 for more information.
COLD	The Capture program starts by deleting all rows in its CD, UOW, and trace tables during initialization. See “Warm and Cold Starts” on page 162 for more information.
PRUNE (default)	The Capture program automatically prunes the CD and UOW tables.
NOPRUNE	Automatic pruning is disabled. The Capture program prunes the CD and UOW tables when you enter the PRUNE command. See the PRUNE command for more information.
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 comes up, 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 debug trace messages to the standard output, <i>stdout</i> .
USERID=auth_name/password	Specifies that the Capture program should connect to the database as user ID <i>auth_name</i> . The correct password must be provided or an error is returned. The <i>auth_name</i> and <i>password</i> are both from 1 to 8 characters in length.
Dbname=database_name	Identifies the name of the DB2 server for VSE database for which changes are to be captured. It 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 has not been set up.

Stopping Capture for VSE

```
▶▶—MSG—partition,DATA=STOP—◀◀
```

where *partition* represents the partition that is running Capture for VSE.

Purpose

Use the STOP command to stop the Capture program gracefully and commit the log records that it processed up to that point.

STOP should be issued before:

- Removing an existing replication source
- Opening and modifying an existing replication source
- Shutting down the database

Usage

If you stop the Capture program, the Capture program 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.

Capture for VSE issues informational or error messages to the console.

Stopping the Capture Program with a Trigger

In some situations, you might want to trigger the stop of the Capture program. For example, you might want to stop it at a specific time, or after all transactions for a particular application have been committed. The simplest way to set up this trigger is to create a table (or use an existing table), define it as a replication source, and identify a particular column whose update, when captured, triggers shutdown.

To stop the Capture program with a trigger:

Use the following example as a guideline.

1. Create a table with a column named STOPCAP or another unique identifier.
2. Define the table as a replication source.
3. Create an application that periodically reads the CD table associated with this replication source table to search for STOPCAP.
4. Schedule the application to start before your Apply programs.

When your application encounters STOPCAP in the CD table, it should issue the command to stop the Capture program.

Suspending Capture for VSE

```
▶▶—MSG—partition,DATA=SUSPEND————▶▶
```

where *partition* represents the partition that is running Capture for VSE.

Purpose

Use the SUSPEND command to suspend the Capture program until you issue the RESUME command.

Usage

You can use this command to suspend the Capture program to improve performance for operational transactions during peak periods without destroying the Capture for VSE program run environment.

Resuming Capture for VSE

```
▶▶—MSG—partition,DATA=RESUME————▶▶
```

where *partition* represents the partition that is running Capture for VSE.

Purpose

Use the RESUME command to resume the suspended Capture program.

Usage

You can use this command to resume the Capture program if you suspended it with the SUSPEND command.

Reinitializing Capture for VSE

```
▶▶—MSG—partition,DATA=REINIT————▶▶
```

where *partition* represents the partition that is running Capture for VSE.

Purpose

Use the REINIT command to reinitialize the Capture program.

Usage

Use the REINIT command to begin to capture changes from new source tables if you add a new replication source with the Control Center while the Capture program is running. This command ensures that the Capture program recognizes the new replication sources in the register table. To allow the new replication source to be visible to the Capture program, you must issue a REINIT command. The REINIT command tells the Capture program to obtain newly added replication sources from the register table (ASN.IBMSNAP_REGISTER).

REINIT also rereads the tuning parameters table (ASN.IBMSNAP_CCPARMS) for any changes made to the tuning parameters.

Attention: Do not use REINIT 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 use warm start (WARM or WARMNS).

Pruning the Change Data and Unit-of-Work Tables

```
▶▶—MSG—partition,DATA=PRUNE————▶▶
```

where *partition* represents the partition that is running Capture for VSE.

Purpose

Use the PRUNE command to initiate the pruning of the CD and UOW (UOW) tables, if you used the NOPRUNE invocation parameter to disable pruning when you started the Capture program.

Usage

This command prunes tables once. 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.

Providing the Current Log Sequence Number

```
▶▶—MSG—partition,DATA=GETLSEQ————▶▶
```

where *partition* represents the partition that is running Capture for VSE.

Purpose

Use the GETLSEQ command to provide the timestamp and current log sequence number.

Usage

Use the GETLSEQ command to determine how far the Capture program has read the DB2 log.

Warm and Cold Starts

This section explains how the Capture program handles warm starts, how it switches to an automatic cold start, and when you might want to force a warm start.

Warm Start Process

When you start the Capture program with the WARM or WARMNS parameter, it searches for the warm start table, ASN.IBMSNAP_WARM_START, which is created during the first definition of a registration source or when DPCNTL.* is executed. This table contains information that enables the Capture program to quickly resynchronize to the time when it stopped. If this table is not available, the Capture program can resynchronize using either the register table, UOW table, or CD tables.

The Capture program switches to a cold start if you did not specify WARMNS and the warm start log sequence number is not available in the DB2 log. (The Apply program performs a full refresh after a cold start.)

Warm start information is saved in most cases. In extreme cases, warm start information might not be saved. For example, you might cancel the Capture program or stop DB2 with SQLEND QUICK in a VSE environment. In this case, the Capture program uses the CD, UOW, or register tables to resynchronize to the time it was stopped.

After a successful warm start, the old rows in the warm start table are deleted.

Automatic Cold Starts

Sometimes the Capture program automatically switches to cold start, even when you specify a warm start. (However, the switch to a cold start is not made when WARMNS is specified.) The switch is made when:

- The warm start log sequence lags behind the current log sequence by more than the LAG_TIME value as specified in the tuning parameters table, ASN.IBMSNAP_CCPPARMS.
- No warm start information is available, because it could not be saved.
- The warm start log sequence is not available on the DB2 server for VSE active log. For example, this could happen if a COLDLOG was performed or a restore from an older archive was performed.
- The Capture program cannot keep up with the amount of log activity, and the log wraps.
- You invoke the Capture program for the first time.

In each of these cases, the Capture program issues an informational message and performs a cold start.

Forcing a Warm Start

You might want to prevent the Capture program from cold starting in some situations. For instance, the Capture program cold starts if DB2 goes down, and the NOTERM parameter was not specified. Forcing a warm start with the WARMNS parameter ensures that the control tables remain intact. You must correct the problem that caused the Capture program to terminate. If you do not correct the problem, the Capture program continues to terminate every time you start it.

Troubleshooting: Problems Using the Capture Program

This section describes commonly known issues with the Capture for VSE program. Before running the Capture program, ensure that you have applied all PTFs for your level of DB2.

Problem

Capture for VSE does not start.

Ensure that:

- Access has been given for the database log and directory minidisks by specifying the DLBLs in the Capture program startup JCL.
- Access has been given to the C Run Time Library.
- The ASNLMAIN package file has been loaded to the database.

Problem

The Capture program is not capturing updates.

Any of the following could prevent the Capture for VSE program from capturing updates:

- DATA CAPTURE CHANGES was not specified on the base tables to be captured.
- 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) on the virtual machine console or in the ASN.IBMSNAP_TRACE table.
 3. Start the Apply program.

Check the ASN.IBMSNAP_TRACE table for possible error messages.

Problem

I'm not sure if the Capture program is running successfully.

The first time 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 ASN.IBMSNAP_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 ASN.IBMSNAP_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 ASN.IBMSNAP_TRAIL table.

Problem

Capture for VSE issued message ASN0000E instead of the proper message number.

Capture for VSE issued a generic message instead of a proper message. This happens if the specified VSAM message file in the Capture startup JCL was not found. See the Capture for VSE program directory for information on installing the VSAM message file.

Problem

Capture for VSE terminates.

The Capture for VSE 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 361 for an explanation of this message.

Chapter 15. Capture for VM

This chapter describes how to set up, operate, and troubleshoot the Capture for VM program.

Read the following sections before reading the sections on operating the Capture program:

- “Setting Up the Capture Program”
- “Specifying Tuning Parameters for the Capture Program”
- “Restrictions When Running the Capture Program” on page 236
- “Authorization for Running the Capture Program” on page 237
- “Recovering from Severe Errors” on page 237

Setting Up the Capture Program

Setting up consists of installing the Capture for VM program and configuring the source servers. All required Capture for VM program tables must be defined prior to invoking the Capture program. These are created by the DB2 Version 5 Control Center when a replication source is defined. You can also create these control tables manually by running the DPCNTL.* file from the Run SQL File window.

See the Capture for VM program directory for instructions about installing 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 ASN.IBMSNAP_CCPPARMS tuning parameters table:

Retention limit

The number of minutes to keep the change data (CD) table rows and the unit-of-work (UOW) table rows. The default value is 10,800, which is 7 days. The rows are deleted up to where the changes have been applied.

Lag limit

The number of minutes the Capture program can be backlogged from the current local time before shutting itself down. The default value is 10,800, which is 7 days. This value is higher for a busy system; therefore, a lower lag limit shuts down the Capture program. If the Capture program shuts itself down, you should perform a cold start if the database log has been archived and the log has wrapped around.

Commit interval

The number of seconds to wait before issuing a COMMIT statement. The default value is 30 seconds. Set the interval smaller than the DB2 timeout interval if the Capture and Apply programs are running at the same time. If the Apply program is not running at the same time as the Capture program, you can set the commit no higher than the DB2 timeout interval.

Prune interval

The number of seconds to wait 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 with the PRUNE command.

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

- Modify DPCNTL.* in the Control Center /sqlib/samples/repl directory before you define the first replication source for a database.
- Update the table with the following SQL statement after you create the tuning parameters table:

```
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
```

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 information on the structure of the tuning parameters table, see Chapter 19, “Table Structures” on page 295.

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:

- Cancel an existing replication source.
- Drop a replication source table.
- Make changes that affect the structure of source 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 are:

- Changes are not captured when the database is running with LOGMODE=N.
- One CD table exists per user table defined to the Capture program.
- You must specify DATA CAPTURE CHANGES on the CREATE or ALTER TABLE statement for the user tables.
- Tables with field procedures for columns (FIELDPROC specified on CREATE or ALTER TABLE) are not supported by Capture for VM.
- 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.

- 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*.
- The Capture program requires access to the appropriate level of the C Run Time Library. On VM, you must issue GLOBAL LOADLIB SCEERUN before invoking ASNCCP.

Authorization for Running the Capture Program

The user ID that operates the Capture program should have DBA authority.

When running the Capture for VM with the DB2 server for VM, the Capture program job also dynamically creates a table (ASN.IBMSNAP_CCPENQ) to determine whether or not a Capture program is already running.

Recovering from Severe Errors

Capture for VM provides you with the following tools to assist you if a severe error occurs:

- Trace buffer
- Trace output
- Storage dump

The following sections provide information on these tools.

Trace Buffer

Capture for VM 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 termination. 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, IBM Replication writes trace information logic flow to VM console.

Storage Dump

When Capture for VM 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.

Operating Capture for VM

The administrator can use the commands in this section to operate Capture for VM.

This section explains how to perform the following Capture for VM tasks:

- Starting
- Stopping
- Suspending temporarily
- Resuming
- Reinitializing
- Pruning
- Monitoring

Before You Start the Capture Program

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

- Define one or more replication sources and subscriptions as described in Chapter 8, “Working with Replication Sources” on page 93 and Chapter 9, “Working with Replication Targets” on page 103. This creates the following control tables that must be defined in the database where the Capture program is to run:
 - ASN.IBMSNAP_REGISTER
 - ASN.IBMSNAP_PRUNCNTL
 - ASN.IBMSNAP_CCPPARMS
 - ASN.IBMSNAP_TRACE
 - ASN.IBMSNAP_WARM_START
 - ASN.IBMSNAP_UOW
 - ASN.IBMSNAP_CRITSEC
 - ASN.IBMSNAP_SUBS_SET
 - ASN.IBMSNAP_SUBS_MEMBR
 - ASN.IBMSNAP_SUBS_STMTS
 - ASN.IBMSNAP_SUBS_COLS
 - ASN.IBMSNAP_SUBS_EVENT
 - ASN.IBMSNAP_APPLYTRAIL
- Ensure that the administrator has DBA privilege.
- Ensure that ASN.IBMSNAP_REGISTER has at least one entry in it by defining a replication source with the DATA CAPTURE CHANGES option.
- Enable the DATA CAPTURE CHANGES attribute for your source tables using either the CREATE TABLE or ALTER TABLE statement. See Chapter 8, “Working with Replication Sources” on page 93 for more information.

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

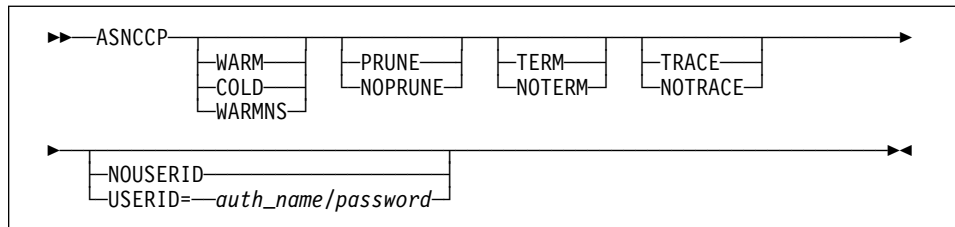
By default, the messages are issued in American English (ASNLS001). To issue messages in a different language, change the LANGUAGE parameter in the ASNPARMs file. See the program directory for the Capture program for a list of supported languages.

Starting Capture for VM

After you start the Capture program, it runs continuously until you stop it or it detects an error.

To start the Capture program:

- Invoke the ASNCCP module from a VM user ID. Keywords must be separated by one or more blanks. The syntax is:



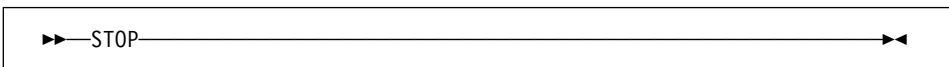
If conflicting invocation parameters are specified, Capture for VM uses the value of the last parameter specified. For example, if ASNCCP is started using the COLD TRACE NOTRACE parameter string, no informational event trace will be written (NOTRACE).

Table 18 on page 240 defines the optional parameters.

Table 18. ASNCCP Command Parameter Definitions for VM

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. See “Warm and Cold Starts” on page 162 for more information.
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. See “Warm and Cold Starts” on page 162 for more information.
COLD	The Capture program starts by deleting all rows in its CD, UOW, and trace tables during initialization. See “Warm and Cold Starts” on page 162 for more information.
PRUNE (default)	The Capture program automatically prunes the CD tables and the UOW table.
NOPRUNE	Automatic pruning is disabled. The Capture program prunes the CD and UOW tables when you enter the PRUNE command. See the PRUNE command for more information.
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 comes up, 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 debug trace messages to the standard output, <i>stdout</i> .
USERID = <i>auth_name/password</i>	Specifies that the Capture program should connect to the database as user ID <i>auth_name</i> . The correct password must be provided or an error is returned. The <i>auth_name</i> 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.

Stopping Capture for VM



Purpose

Use the STOP command to stop the Capture program gracefully and commit the log records that it processed up to that point.

STOP should be issued before:

- Removing an existing replication source

- Opening and modifying an existing replication source
- Shutting down the database

Usage

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.

Capture for VM issues informational or error messages to the console.

Stopping the Capture Program with a Trigger

In some situations, you might want to trigger the stop of the Capture program. For example, you might want to stop it at a specific time, or after all transactions for a particular application have been committed. The simplest way to set up this trigger is to create a table (or use an existing table), define it as a replication source, and identify a particular column whose update, when captured, triggers shutdown.

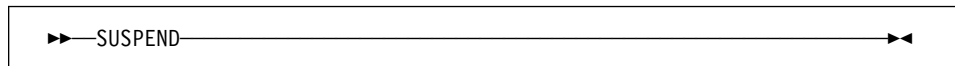
To stop the Capture program with a trigger:

Use the following example as a guideline.

1. Create a table with a column named STOPCAP or another unique identifier.
2. Define the table as a replication source.
3. Create an application that periodically reads the CD table associated with this replication source table to search for STOPCAP.
4. Schedule the application to start before your Apply programs.

When your application encounters STOPCAP in the CD table, it should issue the command to stop the Capture program.

Suspending Capture for VM



Purpose

Use the SUSPEND command to suspend the Capture program until you issue the RESUME command.

Usage

You can use this command to suspend the Capture program to improve performance for operational transactions during peak periods without destroying the Capture for VM program run environment.

Resuming Capture for VM

```
▶▶—RESUME—◀◀
```

Purpose

Use the RESUME command to resume the suspended Capture program.

Usage

You can use this command to resume the Capture program if you suspended it with the SUSPEND command.

Reinitializing Capture for VM

```
▶▶—REINIT—◀◀
```

Purpose

Use the REINIT command to reinitialize the Capture program.

Usage

Use the REINIT command to begin to capture changes from new source tables if you add a new replication source with the Control Center while the Capture program is running. This command ensures that the Capture program recognizes the new replication sources in the register table. To allow the new replication source to be visible to the Capture program, you must issue a REINIT command. The REINIT command tells the Capture program to obtain newly added replication sources from the register table (ASN.IBMSNAP_REGISTER).

REINIT also rereads the tuning parameters table (ASN.IBMSNAP_CCPARMS) for any changes made to the tuning parameters.

Attention: Do not use REINIT 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 use warm start (WARM or WARMNS).

Pruning the Change Data and Unit-of-Work Tables

```
▶▶—PRUNE—◀◀
```

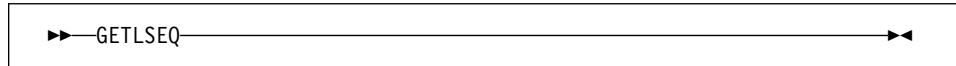
Purpose

Use the PRUNE command to start pruning the change data (CD) and unit-of-work (UOW) tables, if you used the NOPRUNE invocation parameter to disable pruning when you start the Capture program.

Usage

This command prunes tables once. 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.

Providing the Current Log Sequence Number



Purpose

Use the GETLSEQ command to provide the timestamp and current log sequence number.

Usage

Use the GETLSEQ command to determine how far the Capture program has read the DB2 log.

Warm and Cold Starts

This section explains how the Capture program handles warm starts, how it switches to an automatic cold start, and when you might want to force a warm start.

Warm Start Process

When you start the Capture program with the WARM or WARMNS parameter, it searches for the warm start table, ASN.IBMSNAP_WARM_START, which is created during the first definition of a registration source or when DPCNTL.* is executed. This table contains information that enables the Capture program to quickly resynchronize to the time when it stopped. If this table is not available, the Capture program can resynchronize using either the register table, UOW table, or CD tables.

The Capture program switches to a cold start if you did not specify WARMNS and the warm start log sequence number is not available in the log. The Apply program performs a full refresh after a cold start.

Warm start information is saved in most cases. In extreme cases, warm start information might not be saved. For example, an operator might cancel the Capture program or stop DB2 with SQLEND QUICK in a VM environment. In this case, the Capture program uses the CD, unit-of-work, or register tables to resynchronize to the time it was stopped.

After a successful warm start, the old rows in the warm start table are deleted.

Automatic Cold Starts

Sometimes the Capture program automatically switches to cold start, even when you specify a warm start. However, the switch to a cold start is not made when WARMNS is specified. The switch is made when:

- The warm start log sequence lags behind the current log sequence by more than the LAG_TIME value as specified in the tuning parameters table, ASN.IBMSNAP_CCPPARMS.
- No warm start information is available, because it could not be saved.
- The warm start log sequence is not available on the DB2 active log. For example, this could happen if a COLDLOG was performed or a restore from an older archive was performed.
- The Capture program cannot keep up with the amount of log activity, and the log wraps.
- You invoke the Capture program for the first time.

In each of these cases, the Capture program issues an informational message and performs a cold start.

Forcing a Warm Start

You might want to prevent the Capture program from cold starting in some situations. For instance, the Capture program cold starts if DB2 goes down, and the NOTERM parameter was not specified. Forcing a warm start with the WARMNS parameter ensures that the control tables remain intact. You must correct the problem that caused the Capture program to terminate. If you do not correct the problem, the Capture program continues to terminate every time you start it.

Troubleshooting: Problems Using the Capture Program

This section describes commonly known issues with the Capture for VM program. Before running Capture for VM, ensure that you have applied all PTFs for your level of DB2.

Problem

Capture for VM does not start.

Ensure that:

- Access has been given for the database log and directory minidisks. Note that the Capture program issues internal links to these minidisks.
- Access has been given to the C Run Time Library.
- *IDENT authorization has been given to the Capture program virtual machine.
- The ASNLMAIN package file has been loaded to the database.

Problem

The Capture program is not capturing updates.

Any of the following could prevent Capture for VM from capturing updates:

- DATA CAPTURE CHANGES was not specified on the base tables to be captured.

- The proper order for starting the Capture and Apply programs was not used:
 1. Replication sources and subscriptions must be defined before starting Capture for VM.
 2. Start Capture for VM and look for message number ASN0100I (initialization completed) on the virtual machine console or in the ASN.IBMSNAP_TRACE table.
 3. Start the Apply program.

Check the ASN.IBMSNAP_TRACE table for possible error messages.

Problem

I'm not sure if the Capture program is running successfully.

After 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 ASN.IBMSNAP_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.

Subsequent captured updates are placed in CD tables and 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 ASN.IBMSNAP_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 ASN.IBMSNAP_APPLYTRAIL table.

Problem

Capture for VM issued message ASN0000E instead of the proper message number.

Capture for VM issued a generic message instead of a proper message. This happens if either the default message file, ASNLS001 MSG, or the specified message file in CAPTURE ASNPARMS was not found. See the Capture for VM program directory for information on installing the message file.

Problem

Capture for VM terminates.

The Capture for VM 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 361 for an explanation of this message.

Part 5. Mobile Replication

IBM offers two mobile solutions: one for mobile replication of DB2 databases, and one for mobile replication of Microsoft Jet databases using IBM DB2 DataPropagator for Microsoft Jet. This part contains a chapter about each of these solutions.

Chapter 16. 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 graphical 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 new mobile graphical 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 graphical 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 graphical 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

Highlights

Mobile replication highlights include the following features:

- Automatic connect and disconnect of your mobile client via user exit programs
- Automatic start and stop of the Capture and Apply programs

- Copy on demand
- The ability to select replication subscriptions
- Telecommunication cost optimization

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 replication subscriptions as soon as possible and ignores relative timing or event timing options. It uses the same control tables that are used in non-mobile replication.

You can seamlessly make the switch between mobile and non-mobile 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 user exit programs 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 non-mobile 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 IBM 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 user exit program. When specified, the mobile replication component calls the dial-up program every time a physical connection is needed. IBM Mobile Replication does not pass any parameters and does not expect any return code from this exit program.

ASNHANGUP Specifies the disconnect user exit program. 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 has been 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. IBM Mobile Replication does not pass any parameters and does not expect any return code from this exit program.

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 you use to connect your mobile client to the source server.

myhangup.exe

The program 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 on 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:\sql11ib\bin\mydial.exe`
 - b. Click on **Set**.
4. Set the ASNHANGUP variable:
 - a. In the **Value** field, type the path for your user-defined disconnect program. For example:
`C:\sql11ib\bin\myhangup.exe`
 - b. Click on **Set**.
5. Click on **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:\sql11ib\bin\mydial.exe*
 - SET ASNHANGUP = *C:\sql11ib\bin\myhangup.exe*

Where:

mydial.exe

The program you use to connect your mobile client to the source server.

| *myhangup.exe*

| The program 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 registrations and subscriptions in the same way that you define registrations and subscriptions in the non-mobile 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 registration and subscription information to the mobile client.

There are two possible ways to distribute the registration and replication subscription information needed to run a mobile replication scenario. Each of these methods has restrictions.

- Install the DB2 Control Center on the laptop. While the mobile client is connected to the source server, define registrations and subscriptions.
- Define registrations and subscriptions 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 registrations and replication subscriptions 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 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 non-mobile 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 151 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 privileges 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 non-mobile 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.

See “Setting Up the Capture and Apply Programs” on page 175 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 an Apply program has the required privileges:
 - Execute privilege for the Apply packages
 - DBADM or SYSADM privileges 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 = *control_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. For more information about how to define replication sources and subscriptions see Chapter 8, "Working with Replication Sources."

Mobile Replication Processing Cycle

When the mobile replication enabler is invoked, the Capture and Apply programs replicate data as defined in the replication subscription in the following manner:

1. ASNCOPY calls the Capture program. The Capture program stops when it has captured all 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 user exit program, 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 user exit program, if specified in the ASNHANGUP environment variable, at the earliest opportunity to minimize the phone cost.
7. ASNCOPY displays replication statistics from the information selected from the ASN.IBMSNAP_APPLYTRAIL table.
8. ASNCOPY then 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 graphical interface. To execute the mobile graphical interface, type ASNMOBIL from the command line. For more information about the mobile graphical interface see "Starting the Mobile Replication Enabler Using the Mobile Graphical Interface" on page 256.

Table 19 on page 256 lists the parameters that you can specify for the ASNCOPY command. These parameters are not case sensitive.

Table 19. ASNCOPY Command Parameters

Parameter	Explanation
-T	Activates the trace option. You can set the trace option on from the Mobile GUI. 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.
-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 user exit program 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 user exit program for a full refresh. In the ASNLOAD user exit program, 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 replication subscription, 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 replication subscription 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 all 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.
-Q <i>apply_qualifier</i>	You must specify this parameter when the Apply qualifier is not the same as the user id.
-S <i>set_name_list</i>	Specifies a list of replication subscription names that you want to copy.
-U <i>set_name_list</i>	Specifies the copy up (WHOS_ON_FIRST = F) legs of replica sets.
-D <i>set_name_list</i>	Specifies the copy down (WHOS_ON_FIRST = S) legs of replica sets.

Starting the Mobile Replication Enabler Using the Mobile Graphical Interface

The mobile graphical interface, Mobile Replication Enabler window, can be used to start the mobile replication enabler, ASNCOPY. To initiate the mobile graphical interface, type ASNMobil from the command line. The Mobile Replication Enabler window opens, see Figure 32 on page 257.

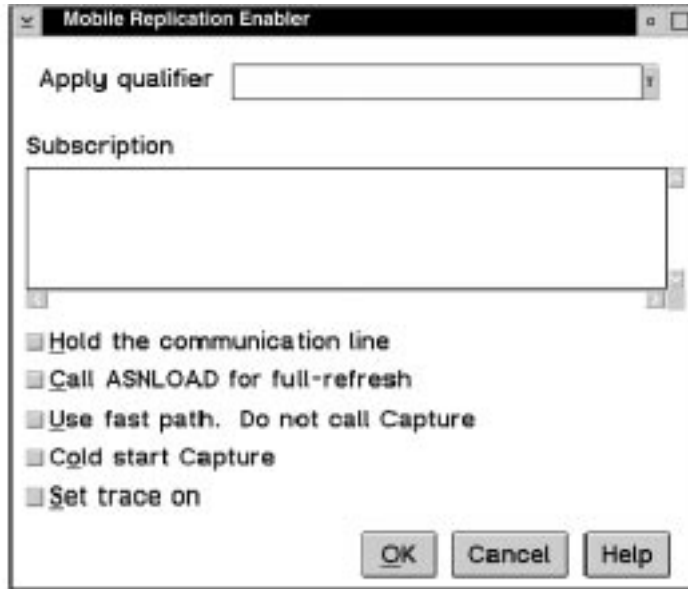


Figure 32. The Mobile Replication Enabler Window. You can set ASNCOPY parameters.

From the Mobile Replication Enabler window, you can:

- Set the trace on
- Hold the communications line
- Call ASNLOAD for full refresh copying
- Use the fast path
- Specify the cold start option for the Capture program

The Mobile Replication Enabler window also allows you to select replication subscriptions and an Apply qualifier.

Selecting Replication Subscriptions

The mobile graphical interface presents a list of active replication subscriptions from which you can select the replication subscriptions you wish to run. A replication subscription 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 replication subscription names you selected. A complete list of replication subscriptions 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 replication subscription list with a particular replication subscription, simply select the Apply qualifier that represents the Apply instance running on your client. ASNCOPY will now only fetch data according to the replication subscription associated with the Apply qualifier selected.

Chapter 17. Mobile Replication Using IBM DB2 DataPropagator for Microsoft Jet

This chapter describes IBM 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 IBM's enterprise data replication 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 (UDB), DB2 Common Server V2, DB2 Connect, and DDCS. DataPropagator for Microsoft Jet requires the DataJoiner Replication Administration tool 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 the Web site <http://www.software.ibm.com/data/datajoiner>. For more information about DataPropagator Relational, see the Web site <http://www.software.ibm.com/data/dpropr>.

Figure 33 on page 260 illustrates how DataPropagator for Microsoft Jet supports replication of Microsoft Access and Microsoft Jet databases.

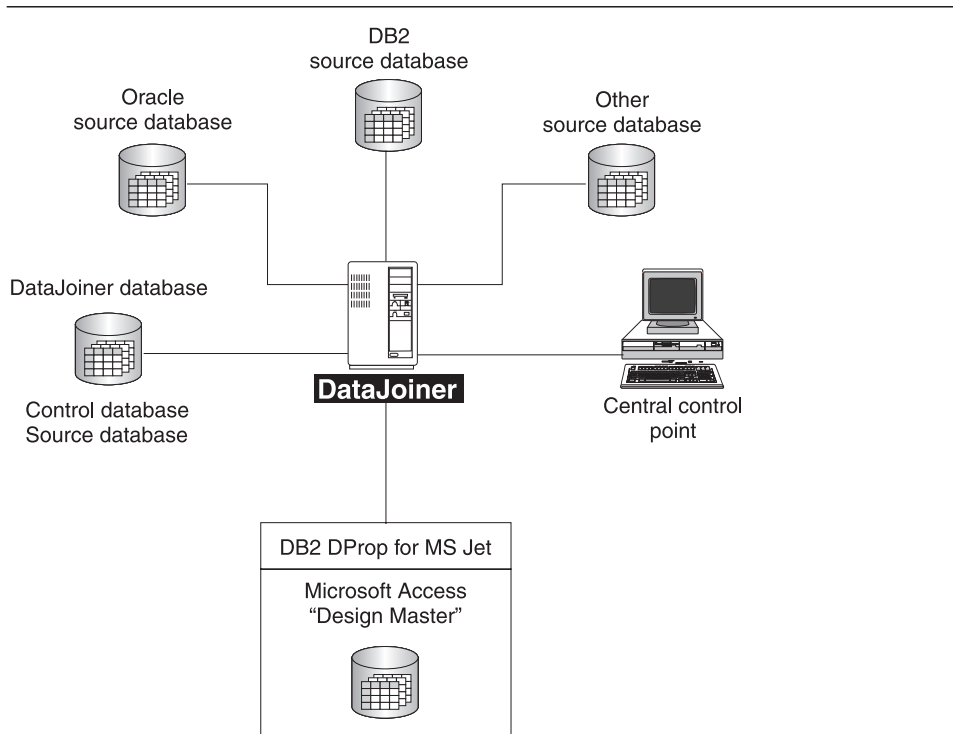


Figure 33. Microsoft Jet Database Replication. DataPropagator for Microsoft Jet extends IBM's Data Replication Solution by supporting Microsoft Access and Microsoft Jet databases.

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 Chapter 6, "Application Planning" on page 53.

DataPropagator for Microsoft Jet administration doesn't require a direct connection to a Microsoft Jet database for administration. The DataJoiner Replication Administration 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.

You can define or redefine replication source and subscription definitions for a Microsoft Jet database at any time, using the DataJoiner Replication Administration tool, 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 simply 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 “Mobile Replication with MS Jet Client” on page 36.

Data Integrity Considerations

Within a network of DB2 databases, DataPropagator Relational supports an *update-anywhere* model which 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 Handling” on page 266. 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 455.

Apply qualifier

A name that identifies an Apply instance of DataPropagator for Microsoft Jet. This name is a case-sensitive character string of up to 18 bytes, and must be unique within all replication network servers accessible from the client. You specify a value for the Apply qualifier when you define a subscription.

Client

The Windows NT or Windows 95 machine on which DataPropagator for Microsoft Jet is installed.

Control server

The database where the subscription control tables are located. For DataPropagator for Microsoft Jet replication, the control server must be a DB2 or DB2 DataJoiner database.

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.

Registration

The process of identifying a source table to make the table available for subscription. You register source tables using the DataJoiner Replication Administration tool, which inserts one row into the register table and the pruning control tables at the source server. You can also use the DB2 Universal Database Control Center to register your source tables, but you must use the DataJoiner Replication Administration tool to create your subscriptions.

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.

Source server

The database where the source tables are located. For DataPropagator for Microsoft Jet replication, the source server can be DB2 or any non-DB2 replication sources defined through DB2 DataJoiner. The control server can be collocated with a UDB or DB2 DataJoiner source server.

Subscription

A specification of source tables and columns to replicate, targets for the replication, and the timing for the replication. You create subscriptions using the DataJoiner Replication Administration tool, which inserts two rows into the subscription set table, and one or more rows into the subscription targets member table and subscription columns table at the control server.

Target server

The Microsoft Jet database where the target tables are located. The target server is also a Design Master.

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 Capture 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 Capture 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.

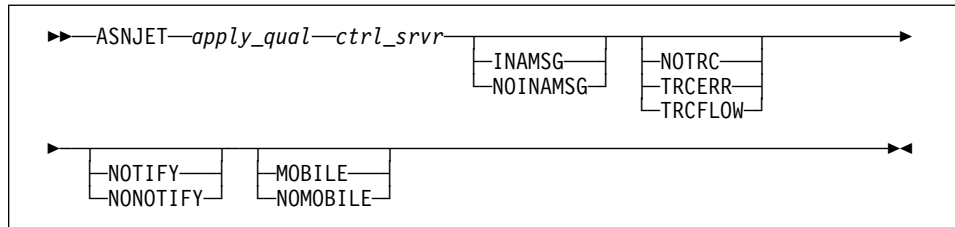


Table 20 on page 264 defines the parameters.

Table 20. ASNJET Command Parameter Definitions for DataPropagator for Microsoft Jet

Parameter	Definition
<i>apply_qual</i>	Specifies the Apply qualifier that uniquely identifies this client.
<i>ctrl_srvr</i>	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)	Specifies that no trace file is created.
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 at the completion of each set subscription, regardless of success or failure.
NONOTIFY (default)	Specifies that DataPropagator for Microsoft Jet does not call the ASNJDONE exit.
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:

```
ASNJET MYQUAL CNTLSRVR MOBILE
```

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 user exit is not called, and the active subscriptions are copied only once and then the program exits.

Example 2: If you enter the following command from a command prompt:

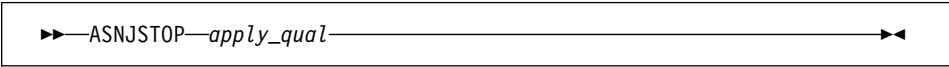
```
ASNJET AQ2 CNTLSRV TRCFLOW NOMOBILE
```

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.

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:

```
ASNJSTOP MYQUAL
```

DataPropagator for Microsoft Jet stops processing the Apply qualifier MQUAL as soon as the current subscription set is processed.

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 registered 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 21, “IBM Replication Messages” on page 361. For more information about troubleshooting, see “Troubleshooting” on page 170.

Returning Control to Users with the ASNJDONE Exit

If you specify the NOTIFY parameter when you start DataPropagator for Microsoft Jet with the ASNJET command, DataPropagator for Microsoft Jet calls the user exit program ASNJDONE at the completion of each set subscription, 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 user exit can examine the error table to discover rejected updates and initiate further actions, such as issuing a message or generating an alert.

See the prolog section in the sample user exit program 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 Handling

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 as follows:

- Inserts a row into the conflict table. (This is not the same conflict 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.

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 program can use this information to take remedial action. When the exit program 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 user exit program.

DataPropagator for Microsoft Jet Control Tables

DataPropagator for Microsoft Jet requires the following new control tables, in addition to the existing DataPropagator relational control tables. For details about the column and index definitions for each of these new control tables, see Chapter 19, "Table Structures" on page 295.

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

Used to signal 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 an Microsoft Jet database.

Part 6. Reference Information

| This part provides reference information for the replication tools. It describes how to
| migrate from the previous version, describes the control and target table structures,
| describes the available problem determination facilities, lists all of the messages issued
| by the Capture and Apply programs, and provides sample invocation JCL.

Chapter 18. Migrating from DataPropagator Relational Version 1 to IBM Replication Version 5

This chapter describes the steps involved in migration, the restrictions and assumptions, and how to use the migration program to migrate from the prior version of IBM Replication known as DataPropagator Relational. A detailed migration guide is available at Web site: <http://www.software.ibm.com/data/dpropr/mig.htm>

If you are a Version 1 customer, you will be able to migrate your replication definitions by running the migration program. However, if you used certain advanced techniques, you might have definitions that are no longer valid for IBM Replication Version 5, thereby requiring manual intervention. For these situations, the migration program enables you to analyze first, and then migrate your unique replication definitions, if desired.

Migration Process Overview

From the perspective of the migration program, the migration process consists of the following steps:

- Collection
- Analysis
- Migration

This section provides a brief overview of each step. Detailed information about the steps and command syntax are provided later in this chapter.

Collection

The collection step, also known as the BUILDDB action, consists of gathering the existing Version 1 replication definitions into a DB2 database, the migration database, which will be used to store information during the migration process.

You can repeat the collection step, each time generating a new version of the registration and subscription details for analysis.

The Migration Database

The migration database contains several global control tables, two of which store registration and subscription information from all specified Version 1 servers. Information contained in the Version 1 servers is added to the migration database during the collection phase. The tables in the database are created only at the beginning of the first collection step for each migration database, and contain information used in the later migration steps.

For more information about the migration database tables, see “Tables in the Migration Database” on page 287.

Analysis

The analysis step, also known as the PREPARE action, consists of analyzing the contents of the global control tables in the migration database to detect potential migration exceptions and make necessary adjustments.

The migration analysis step does the following:

- Updates the migration database with modified information that will be used later in the migration step.
- Writes a migration analysis report that explains the automated modifications and recommendations for migrating the Version 1 system.

After reading the migration report, you can decide whether the automated migration choices are appropriate. If not, you can override the program's results by using standard SQL to update the data in the global control tables. Then you can rerun the analysis step to validate the current migration instructions.

Migration

The migration step consists of three actions: MIGRATE, FALLBACK, and CLEANUP. The MIGRATE action reads the replication definitions, original or modified at the analysis step, from Version 1 control tables and migration database, and migrates to Version 5 control table formats. This action can migrate one source server, one target server, or one of each (source and target) at a time. When this action is performed on a source server, views will be created for allowing the coexistence of the Version 5 capture program and the Version 1 Apply program.

Source server and target server are Version 5 terms that replace the DataPropagator Relational Version 1 terms of data server and copy server. The Version 5 terminology is used throughout this chapter.

You can back out of Version 5 migration and return to Version 1 using the FALLBACK action. If you choose this action, the program extracts the timely information such as timestamps and log sequence numbers from the current Version 5 control tables. It then updates Version 1 control tables with the extracted information based on the original or modified replication definitions in Version 1 control tables and migration database.

After successful migration, you can drop unused Version 1 tables and views using the CLEANUP action.

Migration Requirements

Consider the following points before migrating.

- Migration requires DBADM authority for the databases that contain the Version 1 and Version 5 control tables, and the migration global tables.
- Only one Capture instance per DB2 subsystem/database can exist at a time.

- You must migrate the Capture program to Version 5 before migrating any associated Apply program processes.
- The Version 5 Capture program can coexist with the Version 1 Apply program. However, if the Version 1 Apply program has subscriptions to CCD sources, a PTF for one of the following APARs must be incorporated into that Apply program. Table 21 lists the APAR numbers for common server platforms:

Table 21. Apply Program APAR Numbers for Common Server Platforms

Platform	APAR Number
AIX 6000	IX76243
MVS	PQ08520
OS/2	GG03705
Windows 95 and Windows NT	GG03706

- To use the migration program for Windows 95 or Windows NT, you must create a password file.

The password file must meet the following criteria:

- Be named:
userid.PWD

Where:

userid

The user ID of the person who runs the migration program.

- The first record must contain only the password.

Migration Precautions

Only transaction consistent is supported; convergent consistent is not supported. Consequently, when you migrate Version 1 registrations with CD_CONSISTENT=C, they may lose some compensation records. The subscriptions will continue to run without errors, however.

It is recommended that you not use the Version 5 Control Center when Version 5 Capture and Version 1 Apply coexist. However, if you need new Version 5 replication source and subscription definitions, you can use the Version 5 Control Center.

Installation of the Capture and Apply Programs on DB2 Version 2

You can independently install the Version 5 Capture and Apply programs without having to migrate your database to Version 5 of DB2.

Before You Begin Migration

Before you proceed with the migration steps in the following sections, be aware of the following points.

- The migration program can execute at a Windows NT, Windows 95, or OS/2 control point, which must be able to connect to every source, target, and control server.
- The initial value for the Version 5 APPLY_QUAL is the Version 1 Apply ID. This Apply ID is the subscriber's ID supplied with the migration program.

The term *Apply qualifier* (APPLY_QUAL) is a new concept in Version 5 subscription as well as a column in many subscription control tables. The value in APPLY_QUAL identifies the Apply instance that runs a given subscription. If you want to use an APPLY_QUAL different than the Version 1 Apply ID, you must update the migration database with a new APPLY_QUAL.

- The initial value for the Version 5 SET_NAME is the Version 1 COPY_TABLE.
The term *replication subscription* (SET_NAME) is a new concept in Version 5. It groups many individual subscriptions to source tables on the same source server, where updates to all copies in the group are committed in a single target server transaction, and each copy reflects all source server updates up to a common source server synchronization point.
- The ASN.IBMSNAP_CCPPARMS, ASN.IBMSNAP_CRITSEC, and ASN.IBMSNAP_UOW tables remain in their altered format if fallback occurs.
- The data in the Version 1 userid.IBMSNAP_TRAIL table is not copied to or from the Version 5 ASN.IBMSNAP_APPLYTRAIL table during the MIGRATE or the FALLBACK actions.
- You must use the migration report to determine the order of registration and subscription migration.

Invoking the Migration Program and Actions

You can invoke the migration program by entering the following command followed by an action and two or more required parameters. If you enter the command without entering an action, syntax help is provided.

ASNMIG *action parameters*

Where *action* is one of the following choices:

- BUILDDB** Required action that gathers Version 1 replication definitions and saves them in an ASCII file named BUILDDB.SQL (default name). After completing this action, you must execute this file using the DB2 -TF command to create the migration global control tables and insert rows to these tables. The SQL statements for creation of these tables appear in the BUILDDB.SQL file only if one or more of the tables does not exist.

- PREPARE** Required action that either transforms incompatible Version 1 column values into compatible Version 5 values, or informs you about non-transformable Version 1 values.
- MIGRATE** Required action that migrates from Version 1 to Version 5.
- FALLBACK** Optional action that reverts from Version 5 to Version 1.
- CLEANUP** Optional action that drops unused Version 1 tables and views after successful migration.

The following sections explain how to use each action and its associated parameters.

Collecting Data with BUILDDB

Enter the following commands to collect Version 1 registration and subscription information from the servers in your replication environment:

1.

```
Set db2dbdft=mig_db
```

2.

```
ASN MIG BUILDDB mig_db serv_list
```

Where:

mig_db Name of the migration database where you want to store the global control tables. The database must already exist before you can enter a value for this parameter. This is a required parameter.

serv_list List of databases from which you want to gather Version 1 replication information. You can use an asterisk (*), in addition to one required database alias name, as a wildcard if you want BUILDDB to add to the list of servers as it runs. This is a required parameter.

BUILDDB puts a collection trace into the BUILDDB.TRC file, and puts SQL statements that must be executed in DB2 into the BUILDDB.SQL file. After completing this action, you must execute this file using the DB2 -TF command to create the migration global control tables and insert rows to these tables. The SQL statements for creation of these tables appear in the BUILDDB.SQL file only if one or more of the tables does not exist.

When you specify the wildcard character, *, in your replication network, BUILDDB examines all of the replication definitions for the servers provided and found in the *userid*.IBMSNAP_ROUTING, *userid*.IBMSNAP_REF_CNTL, *userid*.IBMSNAP_TRAIL, and *userid*.ASN. *timestamp*PC tables. The migration database contains global control tables that store registration and subscription details for these servers. When you run BUILDDB, it is recommended that you specify that all servers (source, target, control) be processed together. This will ensure that the prepare and analysis steps will include all information.

BUILDDDB.SQL creates a new version of migration analysis data in your migration database every time you run it. The ASN.VERSION_CONTROL table contains the version and timestamp history of when you last executed BUILDDDB.

Analyzing Data with PREPARE

Enter the following commands to prepare the analysis of the migration database after you run the BUILDDDB.SQL file:

1.

```
set db2dbdft=mig_db
```

2.

```
ASNMMIG PREPARE mig_db ver x
```

Where:

mig_db Name of the migration database where the global control tables are stored. This is a required parameter.

Note: If you want to create your global migration control tables in a DB2 for MVS subsystem, DSNDB04 is the default database.

ver Version of the migration data that you want to analyze. You can use L to select the latest version, or you can select a specific version as noted in your migration database. This is a required parameter.

x Use either V or O in the position of this character. V validates only the replication definitions in the global control tables before proceeding with Version 5 migration. O updates the replication definitions in the global control tables based on the Version 5 control table data. This is an optional parameter. The default option is “O” for the first run of the PREPARE action, and “V” for subsequent runs.

Note: Each time the PREPARE action is run for a given migration version, it puts the transformed or new values in the columns (other than the columns that contain the original replication definitions copied from Version 1 servers) in the ASN.GLOBAL_CD_CNTL or ASN.GLOBAL_REF_CNTL table. Therefore, the default option is “O” for the first run of the PREPARE action. When the migration program knows there are values in the columns prepared by the PREPARE action, it defaults this option to “V” to verify values you might have provided.

PREPARE puts an analysis trace into the PREPARE.TRC file, and puts an analysis summary into the PREPARE.RPT file. The trace file is not translated, and should be handled like the Apply trace file. You should always save the output of each run, and carefully analyze the PREPARE.RPT file for any migration issues.

Migration issues that the PREPARE action checks for are:

- Subscription predicates that limit the scope of data applied

- Subscriptions released by EI SQL statements to the IBMSNAP.REF_CNTL table
- Subscriptions referencing Version 1 control tables
- Subscriptions using the FA run mode. All other run modes are converted to type AO
- Subscriptions scheduled with ENABLE = -n
- SQL_STMTs directed to run elsewhere from the source or target server
- Configurations requiring simultaneous Version 5 migration
- Grouping subscriptions in the replication subscription
- DataJoiner servers
- Existence of convergent consistent tables
- Join registrations through views

Migrating Data with MIGRATE

The migration process includes:

- Run the DPCNTL.UDB or DPCNTL.MVS file using the DB2 -TF command to create Version 5 replication control tables.
- The MIGRATE action migrates one source server, one target server, or one of each (source and target) at a time.
- The output of the MIGRATE action puts a migration trace into the MIGRATE.TRC file, and a report of the success or failure of the migration execution in the MIGRATE.RPT file.

See “Handling Migration Process Failures for the Capture Program” on page 280 for more information about migration failures.

- The MIGRATE action reads the run-time related information, such as the timestamp, log sequence number, or status from source, target, and control servers during migration or fallback.
- When you migrate the Capture program, all registrations at the source server are migrated to Version 5. When you migrate the Apply program, all subscriptions at all the control servers in the subscriber ID's routing table are migrated to Version 5 subscriptions. The MIGRATE action migrates a Version 1 multiple control server environment to one Version 5 control server. When you run ASNMIG, you must specify the location for the consolidated Version 5 control server.

The following sections explain how to migrate the Version 1 registrations and Version 1 subscriptions.

Migrating a Capture Program Process

To migrate the Capture program from Version 1 to Version 5:

1. Stop the Capture program.

2. Connect to the source server database by entering:

```
DB2 CONNECT TO database
```

where *database* is the source server database.

3. Run the DPCNTL.MVS or DPCNTL.UDB file using the DB2 -TF command. This does the following:

- Drops and creates ASN.IBMSNAP_CRITSEC table.
- Creates ASN.IBMSNAP_WARM_START table if it does not exist.
- Creates ASN.IBMSNAP_REGISTER and ASN.IBMSNAP_PRUNCNTL tables.

4. Bind and create the migration package at the source server by entering the following command:

```
DB2 BIND ASNMIG.BND ISOLATION CS BLOCKING ALL
```

where *CS* specifies cursor stability as the isolation level.

5. Repeat the connect and bind steps for the migration database and every server that will be involved in the migration process.

6. Enter the following commands:

- a.

```
set db2dbdft=mig_db
```

- b.

```
ASNMIG MIGRATE D=mig_db S=v1_data_serv V=ver
```

Where:

mig_db Name of the migration database where the global control tables are stored. This is a required parameter.

v1_data_serv Name of the Version 1 data server, known as the source server in Version 5. This is a required parameter.

ver Version of the migration data that you want to migrate. If you do not specify this parameter, ASNMIG selects the latest version number. This is an optional parameter.

You can specify only one migration database, source server, and version.

Note: For the Windows 95 and Windows NT platforms, when you specify parameters with the ASNMIG command for the MIGRATE action, you must surround those parameters with double quotes.

7. Examine the migration report (MIGRATE.RPT) to determine if the program executed successfully.
8. Make sure the Version 5 Capture program is installed on the source server. To install the program, see *Quick Beginnings* for your particular platform.
9. Bind the Version 5 Capture program as described in "Configuring the Capture Program for Windows NT and Windows 95" on page 151 for Windows NT and

Windows 95, and in “Configuring the Capture Program for OS/2” on page 175 for OS/2.

10. Warm start the Version 5 Capture program as described in “Starting Capture for Windows NT and Windows 95” on page 157 for Windows NT and Windows 95, and in “Starting Capture for OS/2” on page 179 for OS/2.

When you run ASN.MIGRATE for a source server where the Capture program runs, MIGRATE performs the following actions:

1. Connects to the source server that you specify.
2. Checks if the APPLY_QUAL column is in the ASN.IBMSNAP_CRITSEC table. If so, proceed. Otherwise, exit.
3. Check if the ASN.IBMSNAP_WARM_START table exists. If not, exit. Otherwise, the rows in all warm start tables are deleted. That is, all Version 1 and Version 5 warm start tables are empty after migration.
4. Alters the ASN.IBMSNAP_CCPPARMS table to add a new PRUNE_INTERVAL column.
5. Alters the ASN.IBMSNAP_UOW table to add two new columns: IBMSNAP_REJ_CODE and IBMSNAP_APPLY_QUAL.
6. Creates the Version 1 ASN.IBMSNAP_CD_CNTL1 table for saving Version 1 registrations.
7. Constructs the rows from the Version 1 ASN.IBMSNAP_CD_CNTL table and ASN.GLOBAL_CD_CNTL table in the migration database, and puts them in the Version 5 ASN.IBMSNAP_REGISTER table.
8. Copies from the Version 1 ASN.IBMSNAP_CD_CNTL table to the ASN.IBMSNAP_CD_CNTL1 table.
9. Drops the Version 1 ASN.IBMSNAP_CD_CNTL table, and then creates the Version 1 ASN.IBMSNAP_CD_CNTL view based on the join of entries in the Version 1 ASN.IBMSNAP_CD_CNTL1 and Version 5 ASN.IBMSNAP_REGISTER tables. If the Version 1 Apply program runs with the Version 5 Capture program concurrently, the Apply program gets the names of the Version 1 pruning control and critical section tables from the ASN.IBMSNAP_CD_CNTL1 table.
10. For each registered source:
 - a. Determines the backup Version 1 pruning control (PC) table name and creates the backup PC table. The backup PC table name is constructed by appending a “1” to the original name (if the table name length is less than 18 characters), or replacing the first character with “M” if the table name length is equal to 18 characters. If the original PC table name already starts with “M,” you are responsible for changing the original Version 1 PC table name before starting the MIGRATE action.
 - b. Copies from the original PC table to the backup PC table.
 - c. Constructs the rows from the PC table and ASN.GLOBAL_CD_CNTL table and puts them in the Version 5 ASN.IBMSNAP_PRUNCNTL table. The values

of the SOURCE_OWNER, SOURCE_TABLE, and SOURCE_VIEW_QUAL columns are resolved when the PREPARE action is run, and are stored in the ASN.GLOBAL_CD_CNTL table. At this point, the values of the APPLY_QUAL and SET_NAME columns are set to "?". The CNTL_ALIAS column has a null value and the CNTL_SERVER column still contains the Version 1 control server name. The values are not known until the Version 1 Apply is migrated later.

- d. Drops the Version 1 pruning control tables.
- e. Creates the Version 1 pruning control view based on the Version 5 ASN.IBMSNAP_PRUNCNTL table.

Handling Migration Process Failures for the Capture Program

Migration of control tables at the data server is committed based on the successful migration of each table. For example, it is possible that ASN.IBMSNAP_CD_CNTL table is migrated and committed, but one of the pruning control tables failed to migrate. In such a case, you must use the FALLBACK action to restore to Version 1. After the cause of the failure is fixed, you can migrate the data server again.

Migrating an Apply Program Process

To migrate Version 1 subscriptions to Version 5:

1. Stop the Version 1 Apply instances currently executing on the target servers to be migrated.
2. Connect to the Version 5 control server database by entering:

```
DB2 CONNECT TO database
```

where *database* is the control server database.
3. Run the DPCNTL.UDB or DPCNTL.MVS file using the DB2 -TF command to create the following control tables:
 - ASN.IBMSNAP_SUBS_SET
 - ASN.IBMSNAP_SUBS_MEMBR
 - ASN.IBMSNAP_SUBS_COLS
 - ASN.IBMSNAP_SUBS_STMTS
 - ASN.IBMSNAP_SUBS_EVENT
 - ASN.IBMSNAP_APPLYTRAIL
4. Bind and create the migration package at the control server by entering the following command:

```
DB2 BIND ASNMIG.BND ISOLATION CS BLOCKING ALL
```

where *CS* specifies cursor stability as the isolation level.
5. Repeat the connect and bind steps for the migration database and all associated source, target, and Version 1 control servers.
6. Enter the following commands:

a.
set db2dbdft=mig_db

b.
ASN MIGRATE D=mig_db T=rout_serv C=v5_ctrl_serv U=sub_id V=ver

Where:

<i>mig_db</i>	Name of the migration database where the global control tables are stored. This is a required parameter.
<i>rout_serv</i>	Server where the Version 1 routing table resides. This is a required parameter.
<i>v5_ctrl_serv</i>	Version 5 control server, which contains the control tables for the Apply program process. This is a required parameter.
<i>sub_id</i>	Subscriber ID of an Apply program process. This is a required parameter.
<i>ver</i>	Version of the migration data that you want to migrate. If you do not specify this parameter, ASN MIG selects the latest version number. This is an optional parameter.

You can specify only one subscriber ID, the server where the subscriber's routing table resides, one migration database, one version of the migration data, and one Version 5 control server. Specifically, this MIGRATE action will migrate the subscriptions of a subscriber all at once.

Note: For the Windows 95 and Windows NT platforms, when you specify parameters with the ASN MIG command for the MIGRATE action, you must surround those parameters with double quotes.

7. Examine the migration report (MIGRATE.RPT) to determine if the program executed successfully.
8. Make sure the Version 5 Apply program is installed on the correct server. To install the program, see *Quick Beginnings* for your particular platform.
9. Bind the Version 5 Apply program as described in "Configuring the Apply Program for Windows NT and Windows 95" on page 152 for Windows NT and Windows 95, and in "Configuring the Apply Program for OS/2" on page 176 for OS/2.
10. Start the Version 5 Apply program as described in "Starting Apply for Windows NT and Windows 95" on page 164 for Windows NT and Windows 95, and in "Before You Start the Apply Program" on page 185 for OS/2.

When you run ASN MIGRATE for a subscriber whose Apply program might run on some target server, MIGRATE performs the following actions:

1. Connects to the server where the routing table resides, and retrieves all Version 1 control server names.
2. For each Version 1 control server:

- a. Constructs the rows from the Version 1 *subscriber*.IBMSNAP_REF_CNTL table and the ASN.GLOBAL_REF_CNTL table and puts them in the Version 5 ASN.IBMSNAP_SUBS_SET, ASN.IBMSNAP_SUBS_MEMBR, and ASN.IBMSNAP_SUBS_STMTS tables at the IBM Replication control server.
 - b. Connects to the source server and updates the APPLY_QUAL, SET_NAME, CNTL_SERVER, and CNTL_ALIAS columns in the ASN.IBMSNAP_PRUNCNTL table.
 - c. Copies the Version 1 *subscriber*.IBMSNAP_REF_COLS table at the Version 1 control server to the Version 5 ASN.IBMSNAP_SUBS_COLS table at the Version 5 control server.
3. If more than one Version 1 control server exists, the MIGRATE action collapses multiple Version 1 control servers into a single Version 5 control server. The subscription definitions from each Version 1 control server are copied to the single IBM Replication control server.
 4. If auto-registration is detected at the target server, MIGRATE instructs you to enter the following command to migrate the ASN.IBMSNAP_CD_CNTL and the Version 1 pruning control tables before the migration of this Apply program process is complete:


```
ASNmig MIGRATE D=mig_db S=v1_copy_serv V=ver
```

 Auto-registration occurs when the subscriber's COPY_OWNER and COPY_TABLE in the *subscriber*.IBMSNAP_REF_CNTL table are the same as the BASE_OWNER and BASE_TABLE in the ASN.IBMSNAP_CD_CNTL table at the target server.
 5. Creates *subscriber*.IBMSNAP_ROUTING1, copies data from *userid*.IBMSNAP_ROUTING to it, then drops *subscriber*.IBMSNAP_ROUTING. This prevents you from inadvertently running the Version 1 Apply program process after you migrate to Version 5.

Handling Migration Process Failures for the Apply Program

The migration of the subscriber's control tables is committed when the subscriber's IBMSNAP_ROUTING table is dropped and the IBMSNAP_ROUTING1 table is created. If the MIGRATE action fails before this occurs, you do not need to run the FALLBACK action because the subscriber's Version 1 control tables are still intact.

Reverting to Version 1 with FALLBACK

You can revert to Version 1 by running the ASNmig command with the FALLBACK action. You need to fall back from all Version 5 Apply program processes before you fall back from associated Version 5 Capture program processes. You can fall back from Apply program processes in whichever order you choose.

After you fall back to Version 1, you can use the same migration process as described in "Migrating Data with MIGRATE" on page 277. You cannot fall back to Version 1 after executing the CLEANUP action, because all unused Version 1 control tables and views are dropped.

When fallback occurs at the source server, ASN.IBMSNAP_CRITSEC, ASN.IBMSNAP_UOW, and ASN.IBMSNAP_CCPPARMS remain in their altered format with the additional or changed columns, which do not cause any conflicts with the Version 1 Capture and Apply programs.

If you successfully migrate both Capture and Apply and you subsequently used the Control Center to define new replication sources and subscriptions before falling back, all new Version 5 replication sources and subscriptions are lost. When this happens, the migration program issues an informational message.

Falling Back from an Apply Program Process

To fall back from Version 5 to Version 1 for an Apply program process:

1. Stop the Version 5 Apply program that you are migrating.
2. Enter the following commands:

a.

```
set db2dbdft=mig_db
```

b.

```
ASNMIIG FALLBACK D=mig_db T=rout_serv C=v5_ctrl_serv U=sub_id V=ver
```

Where:

<i>mig_db</i>	Name of the migration database where the global control tables are stored. This is a required parameter.
<i>rout_serv</i>	Server where the routing table resides. This is a required parameter.
<i>v5_ctrl_serv</i>	Version 5 control server, which contains the control tables for the Apply program process. This is a required parameter.
<i>sub_id</i>	Subscriber ID of an Apply program process. This is a required parameter.
<i>ver</i>	Version of the migration data that you want to fall back to. If you do not specify this parameter, ASNMIIG selects the latest version number. This is an optional parameter.

Note: For the Windows 95 and Windows NT platforms, when you specify parameters with the ASNMIIG command for the FALLBACK action, you must surround those parameters with double quotes.

3. Examine the fallback report (FALLBACK.RPT) to determine if the program executed successfully.
4. Make sure the Version 1 Apply program still exists on the target server where it will execute. To install the program, see *DataPropagator Relational Guide (SC26-3399-05)* for your particular platform.
5. Bind the Version 1 Apply program as described in *DataPropagator Relational Guide (SC26-3399-05)* for your particular platform.

6. Start the Version 1 Apply program as described in *DataPropagator Relational Guide (SC26-3399-05)* for your particular platform after you successfully execute the FALLBACK action.

When you start ASN MIG FALLBACK for a subscriber, the program performs the following actions:

1. Creates *subscriber*.IBMSNAP_ROUTING, copies data from *subscriber*.IBMSNAP_ROUTING1 to it, then drops *subscriber*.IBMSNAP_ROUTING1.
2. Copies back the information in the Version 5 common subscription tables to the *subscriber*.IBMSNAP_REF_CNTL table at the Version 1 control server. It matches the Version 1 DATA_SERVER, BASE_OWNER, BASE_TABLE, COPY_SERVER, COPY_OWNER, and COPY_TABLE to the Version 5 SOURCE_SERVER, SOURCE_OWNER, SOURCE_TABLE, TARGET_SERVER, TARGET_OWNER, and TARGET_TABLE, respectively.

If auto-registration is detected at the target server, FALLBACK instructs you to enter the following command to fall back the ASN.IBMSNAP_REGISTER and ASN.IBMSNAP_PRUNCNTL after the fallback of this Apply program process is complete:

```
ASN MIG FALLBACK D=mig_db S=v5_targ_serv V=ver
```

Auto-registration occurs when the TARGET_SERVER, TARGET_OWNER, and TARGET_TABLE columns in the ASN.IBMSNAP.SUBS_SET and ASN.IBMSNAP.SUBS_MEMBR tables are the same as the SOURCE_SERVER, SOURCE_OWNER and SOURCE_TABLE columns in the ASN.IBMSNAP_REGISTER table at the target server.

You can fall back the ASN.IBMSNAP_REGISTER and ASN.IBMSNAP_PRUNCNTL tables after the fallback of this Apply program process is finished.

Falling Back from a Capture Program Process

To fall back from Version 5 to Version 1 for a Capture program process:

1. Stop the Capture program.
2. Enter the following commands:

a.

```
set db2dbdft=mig_db
```

b.

```
ASN MIG FALLBACK D=mig_db S=source_serv V=ver
```

Where:

mig_db Name of the migration database where the global control tables are stored. This is a required parameter.

source_serv Source server of Version 5. This is a required parameter.

`ver` Version of the migration data that you want to fall back to. If you do not specify this parameter, ASNMIG selects the latest version number. This is an optional parameter.

Note: For the Windows 95 and Windows NT platforms, when you specify parameters with the ASNMIG command for the FALLBACK action, you must surround those parameters with double quotes.

3. Examine the fallback report (FALLBACK.RPT) to determine if the program executed successfully.
4. Make sure the Version 1 Capture program is installed on the source server. To install the program, see *DataPropagator Relational Guide (SC26-3399-05)* for your particular platform.
5. Bind the Version 1 Capture program as described in *DataPropagator Relational Guide (SC26-3399-05)* for your particular platform.
6. Warm start the Version 1 Capture program as described in *DataPropagator Relational Guide (SC26-3399-05)* for your particular platform after you successfully execute the FALLBACK action.

When you start ASNMIG FALLBACK for a source server, the program performs the following actions:

1. Drops the ASN.IBMSNAP_CD_CNTL view.
2. Creates the ASN.IBMSNAP_CD_CNTL table, then copies data to it from the Version 1 ASN.IBMSNAP_CD_CNTL1 and Version 5 ASN.IBMSNAP_REGISTER tables based on the information in the ASN.GLOBAL_CD_CNTL table.
3. Drops the ASN.IBMSNAP_CD_CNTL1 table.
4. Drops the Version 1 pruning control views and creates their counterpart tables.
5. Copies data from the Version 1 backup pruning control tables and the Version 5 ASN.IBMSNAP_PRUNCNTL table to the Version 1 pruning control tables.

Dropping Version 1 Control Tables and Migration Control Tables with CLEANUP

If all replication components in a network are migrated and perform satisfactorily, you can run the ASNMIG command with the CLEANUP action to drop Version 1 views and tables at the Version 1 control, source, and target servers. Before you enter the command, be aware that you cannot fall back after entering the command.

Dropping Apply Program Views and Tables

Enter the following commands to drop the Version 1 Apply program tables:

1.
`set db2dbdft=mig_db`
2.
`ASNMIG CLEANUP D=mig_db C=v5_ctrl_serv T=rout_serv U=sub_id V=ver`

Where:

<i>mig_db</i>	Name of the migration database where the global control tables are stored. This is a required parameter.
<i>v5_ctrl_serv</i>	Version 5 control server. This is a required parameter.
<i>rout_serv</i>	Server where the routing table resides. This is a required parameter.
<i>sub_id</i>	Subscriber ID of an Apply program process. This is a required parameter.
<i>ver</i>	Version of the migration data that you want to clean up. If you do not specify this parameter, ASNMIG CLEANUP selects the latest version number. This is an optional parameter.

Note: For the Windows 95 and Windows NT platforms, when you specify parameters with the ASNMIG command for the CLEANUP action, you must surround those parameters with double quotes.

After you enter this command, the following tables or views are dropped:

- At the Version 1 control server:
 - *subscriber*.IBMSNAP_REF_CNTL tables
 - *subscriber*.IBMSNAP_REF_COLS tables
 - *subscriber*.IBMSNAP_TRAIL tables
- At the server where the subscriber's routing table resides:
 - *subscriber*.IBMSNAP_ROUTING1 tables

Dropping Capture Program Views and Tables

Enter the following commands to drop Version 1 Capture program views and tables:

1.

```
set db2dbdft=mig_db
```

2.

```
ASNMIG CLEANUP D=mig_db S=data_serv V=ver
```

Where:

<i>mig_db</i>	Name of the migration database where the global control tables are stored. This is a required parameter.
<i>data_serv</i>	Version 1 source server. This is a required parameter.
<i>ver</i>	Version of the migration data that you want to clean up. If you do not specify this parameter, ASNMIG CLEANUP selects the latest version number. This is an optional parameter.

Note: For the Windows 95 and Windows NT platforms, when you specify parameters with the ASNMIG command for the CLEANUP action, you must surround those parameters with double quotes.

- After you enter this command, the following tables or views are dropped at the Version 1 source server, or at the copy server if auto-registration occurs:
 - ASN.IBMSNAP_CD_CNTL1 table
 - ASN.IBMSNAP_CD_CNTL view
 - Version 1 pruning control views and backup pruning control tables
 - ASN.IBMSNAP_AUTH table

Tables in the Migration Database

This section describes the relational database tables in the migration database.

Version Control Table

This table contains the timestamp history of when BUILDDDB data was collected, and when it was last updated by the PREPARE action.

Table 22. Version Control Table (VERSION_CONTROL)

Column Name	Type	Length	Nulls?	Description
VERSION	SMALLINT	2	No	Version of the migration data.
COLLECT_TIME	TIMESTAMP	10	No	The COLLECT_TIME timestamp is assigned by the BUILDDDB action when it is gathering the data.
PREPARE_TIME	TIMESTAMP	10	Yes	The PREPARE_TIME timestamp indicates the last time that a particular version of the BUILDDDB action data was analyzed by the PREPARE action. If the PREPARE action with the '0' option is used a second time against the same copy of the BUILDDDB data, you will be warned that the previous analysis and any overrides will be lost.

Global Change Data Control Table

This table contains all Version 1 registration information.

Table 23 (Page 1 of 2). Global Change Data Control Table (GLOBAL_CD_CNTL)

Column Name	Type	Length	Nulls?	Description
VERSION	SMALLINT	2	No	Version of the migration data.
DATA_ALIAS	CHAR	8	No	Column data is from the current BUILDDDB action.
DATA_SERVER	CHAR	18	No	Column data is from the current BUILDDDB action.
BASE_OWNER	CHAR	18	No	Column data is from ASN.IBMSNAP_CD_CNTL.

Table 23 (Page 2 of 2). Global Change Data Control Table (GLOBAL_CD_CNTL)

Column Name	Type	Length	Nulls?	Description
BASE_TABLE	CHAR	18	No	Column data is from ASN.IBMSNAP_CD_CNTL.
CONTAINS_COMMITSEQ	CHAR	1	No	Column data is from ASN.IBMSNAP_CD_CNTL.
CD_OWNER	CHAR	18	Yes	Column data is from ASN.IBMSNAP_CD_CNTL.
CD_TABLE	CHAR	18	Yes	Column data is from ASN.IBMSNAP_CD_CNTL.
PRUNE_CNTL_OWNER	CHAR	18	No	Column data is from ASN.IBMSNAP_CD_CNTL.
PRUNE_CNTL_TABLE	CHAR	18	No	Column data is from ASN.IBMSNAP_CD_CNTL.
FULLREFRESHENABLE	SMALLINT	2	No	Column data is from ASN.IBMSNAP_CD_CNTL.
CCD_OWNER	CHAR	18	Yes	Column data is from ASN.IBMSNAP_CD_CNTL.
CCD_TABLE	CHAR	18	Yes	Column data is from ASN.IBMSNAP_CD_CNTL.
CD_CONDENSED	CHAR	1	Yes	Column data is from ASN.IBMSNAP_CD_CNTL.
CD_CONSISTENT	CHAR	1	Yes	Column data is from ASN.IBMSNAP_CD_CNTL.
CD_COMPLETE	CHAR	1	Yes	Column data is from ASN.IBMSNAP_CD_CNTL.
CCD_CONDENSED	CHAR	1	Yes	Column data is from ASN.IBMSNAP_CD_CNTL.
CCD_CONSISTENT	CHAR	1	Yes	Column data is from ASN.IBMSNAP_CD_CNTL.
CCD_COMPLETE	CHAR	1	Yes	Column data is from ASN.IBMSNAP_CD_CNTL.
BASE_STRUCTURE	SMALLINT	2	No	Column data is from ASN.IBMSNAP_CD_CNTL.
BASE_CONDENSED	CHAR	1	No	Column data is from ASN.IBMSNAP_CD_CNTL.
BASE_CONSISTENT	CHAR	1	No	Column data is from ASN.IBMSNAP_CD_CNTL.
BASE_COMPLETE	CHAR	1	No	Column data is from ASN.IBMSNAP_CD_CNTL.
REGISTRAR	CHAR	18	No	Column data is from ASN.IBMSNAP_CD_CNTL.
ARCH_LEVEL	CHAR	4	No	Column data is from ASN.IBMSNAP_CD_CNTL.
DESCRIPTION	CHAR	254	Yes	Column data is from ASN.IBMSNAP_CD_CNTL.
PREFIX_CHAR	VARCHAR	4	Yes	Column data is from ASN.IBMSNAP_CD_CNTL.
BASE_TYPE	CHAR	1	Yes	Indicates the type of the BASE_TABLE.
REMOTE_SERVER	CHAR	18	Yes	Used for the DataJoiner server name.
CD_CNTL_TYPE	CHAR	1	No	Indicates the type of the ASN.IBMSNAP_CD_CNTL table. The data value can be either 'T' or 'V'.
SOURCE_OWNER	CHAR	18	Yes	Source owner in Version 5.
SOURCE_TABLE	CHAR	18	Yes	Source table in Version 5.
SOURCE_VIEW_QUAL	SMALLINT	2	Yes	Source view qualifier in Version 5.
NEW_BASE_STRUCTURE	SMALLINT	2	Yes	Source structure in Version 5.
OMIT	CHAR	1	Yes	Indicates whether a registration is to be migrated to Version 5. If the data value is 'Y', the registration is not migrated.

Global View Dependencies Table

This table contains all view dependencies in a server. For every registered source table that is a view, the dependent tables will be fetched from SYSVIEWDEP.

Table 24. Global View Dependencies Table (GLOBAL_VIEWDEP)

Column Name	Type	Length	Nulls?	Description
VERSION	SMALLINT	2	No	Version of the migration data.
DATA_ALIAS	CHAR	8	No	Column data to be provided by the BUILDDB action; data server alias in Version 1.
DATA_SERVER	CHAR	18	No	Column data to be provided by the BUILDDB action; data server in Version 1.
BASE_OWNER	CHAR	18	No	Column data to be provided by the BUILDDB action; owner of the base table.
BASE_TABLE	CHAR	18	No	Column data to be provided by the BUILDDB action; name of the base table.
PHYS_TABLE_OWNER	CHAR	18	No	Column data to be provided by the BUILDDB action; owner of the physical table if the BASE_TABLE is a view.
PHYS_TABLE	CHAR	18	No	Column data to be provided by the BUILDDB action; name of the physical table if the BASE_TABLE is a view.

Global View Text Table

This table contains all the CREATE VIEW statements for registered copy table views on a copy server.

Table 25. Global View Text Table (GLOBAL_VIEW_TEXT)

Column Name	Type	Length	Nulls?	Description
VERSION	SMALLINT	2	No	Version of the migration data.
COPY_ALIAS	CHAR	8	No	Column data to be provided by the BUILDDB action; copy server alias in Version 1.
COPY_SERVER	CHAR	18	No	Column data to be provided by the BUILDDB action; copy server in Version 1.
COPY_OWNER	CHAR	18	No	Column data to be provided by the BUILDDB action; owner of the copy table in Version 1.
COPY_TABLE	CHAR	18	No	Column data to be provided by the BUILDDB action; name of the copy table in Version 1.
SEQNO	SMALLINT	2	No	Column data is copied from SYSIBM.SYSVIEWS.
TEXT	VARCHAR	3900	No	Column data is copied from SYSIBM.SYSVIEWS.

Global Refresh Control Table

This table contains all Version 1 subscription information.

Table 26 (Page 1 of 3). Global Refresh Control Table (GLOBAL_REF_CNTL)

Column Name	Type	Length	Nulls?	Description
VERSION	SMALLINT	2	No	Version of the migration data.

Table 26 (Page 2 of 3). Global Refresh Control Table (GLOBAL_REF_CNTL)

Column Name	Type	Length	Nulls?	Description
LOCAL_ALIAS	CHAR	8	Yes	Column from the current BUILDDDB action; local server alias in Version 1.
LOCAL_SERVER	CHAR	18	Yes	Column from the current BUILDDDB action; server which has the subscriber's routing table.
CNTL_ALIAS	CHAR	8	No	Column from the current BUILDDDB action; control server alias in Version 1.
CNTL_SERVER	CHAR	18	No	Column from the current BUILDDDB action; control server in Version 1.
DATA_ALIAS	CHAR	8	Yes	Column from the current BUILDDDB action; data server alias in Version 1.
COPY_ALIAS	CHAR	8	Yes	Column from the current BUILDDDB action; copy server alias in Version 1.
SUBSCRIBER	CHAR	18	No	Column from the current BUILDDDB action; the Apply ID of the subscriber.
COPY_TYPE	CHAR	1	No	Column from the current BUILDDDB action; the data indicates the type of COPY_TABLE.
ENABLE	SMALLINT	2	No	Column from ASN.IBMSNAP_REF_CNTL.
DATA_SERVER	CHAR	18	No	Column from ASN.IBMSNAP_REF_CNTL.
BASE_OWNER	CHAR	18	No	Column from ASN.IBMSNAP_REF_CNTL.
BASE_TABLE	CHAR	18	No	Column from ASN.IBMSNAP_REF_CNTL.
STMT_NUMBER	SMALLINT	2	No	Column from ASN.IBMSNAP_REF_CNTL.
PRIORITY	SMALLINT	2	No	Column from ASN.IBMSNAP_REF_CNTL.
COPY_OWNER	CHAR	18	No	Column from ASN.IBMSNAP_REF_CNTL.
COPY_TABLE	CHAR	18	No	Column from ASN.IBMSNAP_REF_CNTL.
RUNMODE	CHAR	2	No	Column from ASN.IBMSNAP_REF_CNTL.
STATUS	SMALLINT	2	No	Column from ASN.IBMSNAP_REF_CNTL.
INTERVAL_MINUTES	INTEGER	4	No	Column from ASN.IBMSNAP_REF_CNTL.
COPY_CONDENSED	CHAR	1	No	Column from ASN.IBMSNAP_REF_CNTL.
COPY_COMPLETE	CHAR	1	No	Column from ASN.IBMSNAP_REF_CNTL.
COPY_CONSISTENT	CHAR	1	No	Column from ASN.IBMSNAP_REF_CNTL.
COPY_STRUCTURE	SMALLINT	2	No	Column from ASN.IBMSNAP_REF_CNTL.
ARCH_LEVEL	CHAR	4	No	Column from ASN.IBMSNAP_REF_CNTL.
COPY_SERVER	CHAR	18	No	Column from ASN.IBMSNAP_REF_CNTL.
PREDICATES	VARCHAR	512	Yes	Column from ASN.IBMSNAP_REF_CNTL.
SQL_STMT	VARCHAR	1024	Yes	Column from ASN.IBMSNAP_REF_CNTL.
PHYS_COPY_OWNER	CHAR	18	No	Column from the current BUILDDDB action; owner of the physical table if COPY_TABLE is a view.
PHYS_COPY_TABLE	CHAR	18	No	Column from the current BUILDDDB action; name of the physical table if COPY_TABLE is a view.
COUNT_NEG_ENABLES	INTEGER	4	No	The number of negative ENABLEs in the <i>subscriber.IBMSNAP_TRAIL</i> table.
DEFAULT_SET_NAME	CHAR	18	Yes	The default subscription set name assignment.

Table 26 (Page 3 of 3). Global Refresh Control Table (GLOBAL_REF_CNTL)

Column Name	Type	Length	Nulls?	Description
BEFORE_OR_AFTER	CHAR	1	Yes	Used to indicate if an SQL statement is to be removed (when the value is 'R' or 'X') or assigned (when the value is 'A', 'B', or 'S').
DATA_LIMITING_PRED	CHAR	1	Yes	Used to indicate if any predicates would limit the scope of data fetched.
RELEASE_SQL_STMT	CHAR	1	Yes	Used to indicate if any SQL statements are used to release other substitutions.
OTHER_V1_PRED_REF	CHAR	1	Yes	Used to indicate any other access to a Version 1 control table in a predicate.
OTHER_V1_STMT_REF	CHAR	1	Yes	Used to indicate any other access to a Version 1 control table in an SQL statement.
ACTION_ON_PRED	CHAR	1	Yes	Used to indicate that a predicate is to be removed ('R') or substituted ('S').
SUBSTITUTE_PRED	VARCHAR	512	Yes	Used to store the substituted predicate.
ACTION_ON_SQLSTMT	CHAR	1	Yes	Used to indicate that an SQL statement is to be removed ('R') or substituted ('S').
SUBSTITUTE_SQLSTMT	VARCHAR	1024	Yes	Used to store the substituted SQL statement.
JOINVIEW_SET_NAME	CHAR	18	Yes	The set name for the join view subscriptions.
DLP_PRED_SET_NAME	CHAR	18	Yes	The set name for the subscriptions that have data limiting predicates.
SAME_DSCS_SET_NAME	CHAR	18	Yes	The set name for subscriptions having the same data server or copy server; used for candidate subscription sets.
FINAL_SET_NAME	CHAR	18	Yes	Used to indicate the final assignment of the subscription sets.
OMIT	CHAR	1	Yes	Used to indicate whether a subscription or EI statement is to be migrated to Version 5. If the data value is 'Y', it is not migrated.
NEW_APPLY_QUAL	CHAR	18	Yes	Option to supply a new Apply ID for execution of the Apply program, allowing exploitation of secondary AUTHIDs.
REFRESH_TIMING	CHAR	1	Yes	Option to specify event refresh timing in Version 5.
EVENT_NAME	CHAR	18	Yes	Option to specify the event name in Version 5.
SOURCE_OWNER	CHAR	18	Yes	The source owner in Version 5.
SOURCE_TABLE	CHAR	18	Yes	The source table name in Version 5.
SOURCE_VIEW_QUAL	SMALLINT	2	Yes	The view qualifier in Version 5.

Global Refresh Columns Table

This table contains the contents of the Version 1 REF_COLS tables.

Table 27 (Page 1 of 2). Global Refresh Columns Table (GLOBAL_REF_COLS)

Column Name	Type	Length	Nulls?	Description
VERSION	SMALLINT	2	No	Version of the migration data.

Table 27 (Page 2 of 2). Global Refresh Columns Table (GLOBAL_REF_COLS)

Column Name	Type	Length	Nulls?	Description
CNTL_ALIAS	CHAR	8	No	Column from the current BUILDDB action; the control server alias in Version 1.
CNTL_SERVER	CHAR	18	No	Column from the current BUILDDB action; the control server in Version 1.
SUBSCRIBER	CHAR	18	No	Column from the current BUILDDB action; the Apply ID of the subscriber.
COPY_SERVER	CHAR	18	No	Column from ASN.IBMSNAP_REF_CNTL.
COPY_OWNER	CHAR	18	No	Column from ASN.IBMSNAP_REF_CNTL.
COPY_TABLE	CHAR	18	No	Column from ASN.IBMSNAP_REF_CNTL.
STMT_NUMBER	SMALLINT	2	No	Column from ASN.IBMSNAP_REF_CNTL.
COL_TYPE	CHAR	1	No	Column from ASN.IBMSNAP_REF_CNTL.
TARGET_NAME	CHAR	18	No	Column from ASN.IBMSNAP_REF_CNTL.
IS_KEY	CHAR	1	No	Column from ASN.IBMSNAP_REF_CNTL.
COLNO	SMALLINT	2	No	Column from ASN.IBMSNAP_REF_CNTL.
EXPRESSION	VARCHAR	254	No	Column from ASN.IBMSNAP_REF_CNTL.

Registration Migration Units (RMU) Table

This table is not initialized by BUILDDB. It is used by the PREPARE action to identify Registration Migration Units (RMUs) and to assess and store RMU migration dependencies.

Table 28. Registration Migration Units Table (RMU_LIST)

Column Name	Type	Length	Nulls?
VERSION	SMALLINT	2	No
RMU_NUMBER	SMALLINT	2	Yes
DATA_ALIAS	CHAR	8	No
DATA_SERVER	CHAR	18	No
INDEPENDENT	CHAR	1	Yes
SEQUENCE_NUMBER	SMALLINT	2	Yes

Subscription Migration Units (SMU) Table

This table is not initialized by BUILDDB. It is used by the PREPARE action to identify Subscription Migration Units (SMUs) and to assess and store SMU migration dependencies.

Table 29 (Page 1 of 2). Subscription Migration Units Table (SMU_LIST)

Column Name	Type	Length	Nulls?
VERSION	SMALLINT	2	No
SMU_NUMBER	SMALLINT	2	Yes

Table 29 (Page 2 of 2). Subscription Migration Units Table (SMU_LIST)

Column Name	Type	Length	Nulls?
SUBSCRIBER	CHAR	18	No
CNTL_ALIAS	CHAR	8	No
CNTL_SERVER	CHAR	18	No
INDEPENDENT	CHAR	1	Yes
SEQUENCE_NUMBER	SMALLINT	2	Yes
SUB_FROM_BASE_VIEW	CHAR	1	Yes
LOCAL_ALIAS	CHAR	8	No

RMU Dependencies on SMU Table

This table lists all occurrences of Apply programs maintaining a CCD on a copy server. In the case of dependencies, the Apply programs must be migrated at the same time as the registrations.

Table 30. RMU Dependencies on SMU Table (RMU_DEPENDS_ON_SMU)

Column Name	Type	Length	Nulls?
VERSION	SMALLINT	2	No
RMU_NUMBER	SMALLINT	2	Yes
MIGRATE_RMU_ALIAS	CHAR	8	No
MIGRATE_RMU	CHAR	18	No
SMU_NUMBER	SMALLINT	2	Yes
DEPENDS_SMU_SUBSCR	CHAR	18	No
DEPENDS_SMU_CTLALS	CHAR	8	No
DEPENDS_SMU_CTLSVR	CHAR	18	No
DEPENDS_SMU_LCLALS	CHAR	8	No

SMU Dependencies on RMU Table

This table lists all occurrences of Apply programs maintaining a CCD on a copy server. In the case of dependencies, the Apply programs must be migrated at the same time as the registrations.

Table 31 (Page 1 of 2). SMU Dependencies on RMU Table (SMU_DEPENDS_ON_RMU)

Column Name	Type	Length	Nulls?
VERSION	SMALLINT	2	No
SMU_NUMBER	SMALLINT	2	Yes
MIGRATE_SMU_SUBSCR	CHAR	18	No
MIGRATE_SMU_CTLALS	CHAR	8	No
MIGRATE_SMU_CTLSVR	CHAR	18	No
MIGRATE_SMU_CPYSVR	CHAR	18	No

Table 31 (Page 2 of 2). SMU Dependencies on RMU Table (SMU_DEPENDS_ON_RMU)

Column Name	Type	Length	Nulls?
RMU_NUMBER	SMALLINT	2	Yes
DEPENDS_RMU_ALIAS	CHAR	8	No
DEPENDS_RMU	CHAR	18	No
MIGRATE_SMU_LCLALS	CHAR	8	No

Chapter 19. Table Structures

This chapter describes the relational database tables that IBM Replication uses in its copy operations.

Sample create table and create index statements are provided for all tables.

Table 32 on page 296, Table 33 on page 297, and Table 34 on page 298 provide directories to and brief descriptions of the tables listed in this chapter. For a view-at-a-glance chart of the control and source server tables, table keys, and their parameters see Appendix B, "Tables at a Glance" on page 441.

List of Tables Used at the Source Server

The following table provides a list of tables used at the source server.

Table 32. Quick Reference for Tables Used at the Source Server during IBM Replication Processes

Table name	Internal name and Description	See page
Capture enqueue table	ASN.IBMSNAP_CCPENQ Used in the VM and VSE environments to ensure that only one Capture program is running per database.	299
Change data table	ASN.CD <i>timestamp</i> Created while defining replication sources. It contains changed data information. The timestamp in the table name is from the source server; for auto-registered tables, the timestamp is from the target server.	299
Critical section table	ASN.IBMSNAP_CRITSEC Used as a logical lock between the Capture program and the Apply program for serialization of internal logic.	301
Pruning control table	ASN.IBMSNAP_PRUNCNTL Pruning control information for the current source server. There is one pruning control table at each source server and one row per source-to-target copy. This table associates timestamp with log address, allows the Apply program to trigger the Capture program, and coordinates pruning.	302
Register table	ASN.IBMSNAP_REGISTER Contains information about replication sources, such as the names of the replication source tables, their attributes, and their staging table names.	306
Trace table	ASN.IBMSNAP_TRACE Contains Capture program audit trail information. Required if the Capture program is installed.	312
Tuning parameters table	ASN.IBMSNAP_CCPPARMS Contains parameters that you can modify to control the performance of the Capture program.	313
Unit-of-work table	ASN.IBMSNAP_UOW Used to maintain transaction consistency. It is a staging table that contains commit information. The information from this staging table is joined with change information from the change data table to produce consistent, committed changes.	314
Warm start table	ASN.IBMSNAP_WARM_START Contains information that enables the Capture program to resynchronize.	317
Warm start table for Capture for VSE and VM	ASN.IBMSNAP_WARM_START Contains information that enables the Capture program to resynchronize in the VM and VSE environments.	318

List of Tables Used at the Control Server

The following table provides a list of tables used at the control server.

Table 33. Quick Reference for Tables Created at the Control Server during IBM Replication Processes

Table name	Internal name and Description	See page
Apply trail table	ASN.IBMSNAP_APPLYTRAIL Records a history of updates performed against subscriptions. This table is a repository of diagnostics and performance statistics that can be used to audit update activity.	319
Subscription columns table	ASN.IBMSNAP_SUBS_COLS Contains information on the common subscription columns being copied in a subscription.	322
Subscription events table	ASN.IBMSNAP_SUBS_EVENT Contains information on the events being copied in a subscription.	324
Subscription set table	ASN.IBMSNAP_SUBS_SET Ensures that each set name is used only once for every Apply qualifier.	326
Subscription statements table	ASN.IBMSNAP_SUBS_STMTS Contains information on the statements being copied in a subscription.	330
Subscription targets member table	ASN.IBMSNAP_SUBS_MEMBR Contains information on the target members being copied in a subscription.	332
Row-replica target list table (Microsoft Jet specific)	ASN.IBMSNAP_SUBS_TGTS Maintains the names of the row-replica tables.	336
Subscription schema changes table (Microsoft Jet specific)	ASN.IBMSNAP_SCHEMA_CHG Used to signal one of the following modifications to a subscription: <ul style="list-style-type: none"> • Adding a member to a set (ADDMEMBR) • Deleting a member from a set (DELMEMBR) • Adding a column to a set (ALTERMEM) 	335

List of Tables Used at the Target Server

The following table provides a list of tables used at the target server.

Table 34. Quick Reference for Target Tables

Table name	Internal name and Description	See page
Base aggregate target table	<i>userid.target_table</i> A target table that contains data aggregated from a source table.	338
Change aggregate target table	<i>userid.target_table</i> A target table that contains data aggregations based on changes from a source table.	339
Consistent change data target table	<i>userid.target_table</i> A join of the change data table and the unit-of-work table. The join step needs to be performed only once for the case of a one-many fan-out snapshot, if the results of a join are stored in the consistent change data target table.	340
Point-in-time target table	<i>userid.target_table</i> Indicates a specific commit time from the source server.	343
Replica target table	<i>userid.target_table</i> Contains a primary key identical to the primary key of the user table.	344
User copy target table	<i>userid.target_table</i> Identical to a point-in-time table except that the IBMSNAP_LOGMARKER column is not included here.	345
Conflict table (Microsoft Jet specific)	IBMSNAP_<target_name>_CONFLICT Contains row data for DataPropagator for Microsoft Jet-detected conflict losers.	347
Error information table (Microsoft Jet specific)	IBMSNAP_ERROR_INFO Contains additional information to identify the row-replica table and row that caused an error.	347
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.	348
Error side information table (Microsoft Jet specific)	IBMSNAP_SIDE_INFO Contains the names of the conflict tables.	349
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: <ul style="list-style-type: none"> • Deletes rows 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. 	349
Synchronization generations table (Microsoft Jet specific)	IBMSNAP_S_GENERATION Prevents cyclic updates from propagating back to the RDBMS from a Microsoft Jet database.	350

Control Tables Used at the Source Server

The following section provides a brief description of the control tables used at the source server, the columns in each control table, and the create table and create index statements for each table. The structure of all control tables is subject to change.

Capture Enqueue Table

ASN.IBMSNAP_CCPENQ

The Capture enqueue table is used in the VM and VSE environments only.

Table 35 provides a list and a brief description of the Capture enqueue table column.

Table 35. CD Table Column

Column name	Description
LOCKNAME	Unique name of the resource for this database.

Figure 34 shows the create table statement for Capture enqueue tables.

For DB2 for VM/VSE

```
CREATE TABLE ASN.IBMSNAP_CCPENQ (
  LOCKNAME CHAR (9) NOT NULL);
```

Figure 34. Create Table Statement for DB2 for VM/VSE Capture Enqueue Tables

Change Data Table

This section contains programming interface information.

ASN.CDtimestamp

Change data (CD) tables contain changed data assigned to source table rows, in time series. The changed data is used to update other target tables. There is one CD table for each source table. The CD table is created when you define a replication source that is enabled for data capture.

CD tables can contain committed changes and uncommitted changes, and possibly incomplete changes in rows. Normally, the Capture program inserts rows into this table for a particular platform. It also prunes rows periodically. On cold start, all of the CD table's entries will be deleted.

Commit sequencing, which is not known when records are inserted into the CD table, is provided through the unit-of-work table.

Change Data Table

The Control Center automatically creates an index. A unique ascending index is required in the IBMSNAP_UOWID and IBMSNAP_INTENTSEQ columns.

Qualities of CD tables:

- Before-image user data columns and computed user data columns must be nullable.
- Null attributes of the after-image user data columns must match null attributes of the source.
- Views of change data tables can be included in view replication sources.
- If standard or enhanced conflict detection is used for replicas, all before-image and after-image column values must be captured.
- DPROPR V1 supported an IBMSNAP_LOGMARKER column in the change data table. For compatibility with existing installations, the presence of the once optional IBMSNAP_LOGMARKER column is tolerated. This column is no longer used.

The Capture program prunes the CD tables based on information inserted into the pruning control table by the Apply program. The Apply program maintains the log sequence number in the SYNCHPOINT column of the pruning control table. Initially, the Apply program sets this sequence number to zero when it performs a full refresh. A zero value signals the Capture program to start capturing. When the Apply program copies changes from the CD table to the target table, it updates the SYNCHPOINT column. The Capture program can then prune changes in the CD table up through the row with the highest log sequence number.

Pruning occurs depending on whether you start the Capture program with the PRUNE or NOPRUNE invocation parameter and how the prune interval is set in the tuning parameters table. If you don't specify a parameter, PRUNE is the default. See the Capture and Apply chapter for your platform in this book to learn how to set pruning.

Table 36 provides a list and a brief description of each of the CD table columns.

Column name	Description
IBMSNAP_UOWID	Unit-of-work ID. It is also the foreign key into the unit-of-work table.
IBMSNAP_INTENTSEQ	Unique identifier that describes the sequence of a change within a transaction.
IBMSNAP_OPERATION	Character value of 'I', 'U', or 'D', indicating an insert, update, or delete record.
KEY1	User column from source table specified by the user when defining replication sources.
DATA1	User column from source table specified by the user when defining replication sources.

Figure 35 on page 301 and Figure 36 on page 301 show the create table and create index statements for CD tables.

For DB2 for OS/390 Version 4 or Higher

```
CREATE TABLE userid.CDtimestamp (
  IBMSNAP_UOWID CHAR(10) FOR BIT DATA NOT NULL,
  IBMSNAP_INTENTSEQ CHAR(10) FOR BIT DATA NOT NULL,
  IBMSNAP_OPERATION CHAR(1) NOT NULL,
  <user data columns>);
```

```
CREATE TYPE 2 INDEX userid.IXtimestamp
  timestamp ON userid.CDtimestamp (
  IBMSNAP_UOWID ASC,
  IBMSNAP_INTENTSEQ ASC);
```

Figure 35. Create Table and Create Index Statements for DB2 for OS/390 Version 4 or Higher CD Tables

For All Other Platforms

```
CREATE TABLE userid.CDtimestamp (
  IBMSNAP_UOWID CHAR(10) FOR BIT DATA NOT NULL,
  IBMSNAP_INTENTSEQ CHAR(10) FOR BIT DATA NOT NULL,
  IBMSNAP_OPERATION CHAR(1) NOT NULL,
  <user data columns>);
```

```
CREATE INDEX userid.IXtimestamp ON
  userid.CDtimestamp (IBMSNAP_UOWID ASC, IBMSNAP_INTENTSEQ ASC);
```

Figure 36. Create Table and Create Index Statements for CD Tables on All Other Platforms

Critical Section Table

ASN.IBMSNAP_CRITSEC

The critical section table is used for concurrency control purposes between the Capture and Apply programs for servers without isolation UR (uncommitted read).

Table 37 provides a brief description of the critical section table column.

Table 37. Critical Section Table Column

Column name	Description
APPLY_QUAL	Identifies the Apply program for the platform instance that will run this subscription. 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. When you use a user-written Apply program, you must specify this value when you define a subscription. This value is used to populate the IBMSNAP_APPLY_QUAL column of the unit-of-work table.

Pruning Control Table

Figure 37 on page 302 and Figure 38 on page 302 show the create table and create index statements for critical section tables.

For DB2 for OS/390 Version 4 or Higher

```
CREATE TABLE ASN.IBMSNAP_CRITSEC(  
  APPLY_QUAL CHAR(18) NOT NULL,  
  DATA CAPTURE CHANGES;  
  
CREATE TYPE 2 INDEX ASN.IBMSNAP_CRITSECX ON ASN.IBMSNAP_CRITSEC  
  (APPLY_QUAL ASC);
```

Figure 37. Create Table and Create Index Statements for DB2 for OS/390 Version 4 or Higher Critical Section Tables

For All Other Platforms

```
CREATE TABLE ASN.IBMSNAP_CRITSEC(  
  APPLY_QUAL CHAR(18) NOT NULL,  
  <additional padding columns for DB2 for OS/390 version 3>  
  DATA CAPTURE CHANGES;  
  
CREATE UNIQUE INDEX ASN.IBMSNAP_CRITSECX ON ASN.IBMSNAP_CRITSEC  
  (APPLY_QUAL ASC);
```

Figure 38. Create Table and Create Index Statements for Critical Section Tables on All Other Platforms

Pruning Control Table

This section contains programming interface information.

ASN.IBMSNAP_PRUNCNTL

The pruning control table provides a timestamp and log-address translation in case the SYNCHPOINT column for Capture for MVS is a relative byte address value.

The pruning control table also coordinates the pruning of the change data tables, which have the potential for unlimited growth. There is one pruning control table at each source server and one row for each subscription member.

This table is located at the source server, but it is also automatically created at the target server when replica and CCD target tables are defined. Replica and CCD target tables created at the target server can be used as source tables for further subscriptions, making the target server a source server as well. This is performed as part of auto-registration, described in Part 3, “Administering Your Replication System” on page 81.

Pruning Control Table

The rows in the pruning control table are not deleted during a cold start of the Capture program. The Control Center uses the values from the pruning control table to list direct copies of a particular source table.

Table 38 provides a brief description of each of the pruning control table columns.

Table 38 (Page 1 of 2). Pruning Control Table Columns

Column name	Description
TARGET_SERVER	Matches the column in the subscription set table of the same name. The name that appears in this column is also the name of the database (DRDA application server) where the target is stored. For DB2 for OS/2, there is no DRDA application server, so this column is informational.
TARGET_OWNER	The middle qualifier of the target table name. Default is the user ID of the user defining the subscription. Matches the column of the same name in the subscription targets member table.
TARGET_TABLE	The third qualifier of the target table name. Matches the column of the same name in the subscription targets member table.
SYNCHTIME	A source server timestamp that can be added to any captured log records for any source table. This timestamp indicates that a change did not occur before this time. If the log records are individually timestamped, use those timestamps; otherwise, these values are approximate and are set by the Apply program at the start of a subscription cycle and after each subsequent cycle. These values are eventually appended into each change data table row.
SYNCHPOINT	The SYNCHPOINT value equals the SYNCHPOINT field value in the common subscription set table. This value is used to coordinate the pruning of change data tables. The Apply program sets this initial value to 0, indicating refresh. If the Apply program sets a nonzero value, the change data table can be eligible for pruning.
SOURCE_OWNER	The middle qualifier of the source table, which names the owner of the source table whose updates are being captured.
SOURCE_TABLE	The third qualifier of the source table, which names the source table whose updates are being captured.
SOURCE_VIEW_QUAL	Supports join subscriptions by matching the similar columns in the register table. You must have this column to support multiple subscriptions for the different source views with identical SOURCE_OWNER, SOURCE_TABLE column values.
APPLY_QUAL	Identifies the Apply program for the platform instance that will run this subscription. 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. This column is part of the foreign key from the subscription set table.
SET_NAME	Names a subscription set. This value is unique within an Apply qualifier value. This column is part of the foreign key from the subscription set table.
CNTL_SERVER	The RDB name of the control server for the Apply program updating this row; either Apply for MVS, Apply for VSE, or Apply for VM.

Pruning Control Table

Table 38 (Page 2 of 2). Pruning Control Table Columns

Column name	Description
TARGET_STRUCTURE	A value that identifies the type of target table: <ol style="list-style-type: none">1 Source table2 N/A3 CCD table4 Point-in-time table5 Base aggregate table6 Change aggregate table7 Replica8 User copy
CNTL_ALIAS	The name of the control server used by the Apply program on the DB2 Universal Database. This alias name identifies the location of the subscription definition and is later used to perform administrative actions on a subscription. This alias name does not necessarily match the alias of the control server used by the Apply program for the DB2 Universal Database.

Figure 39, Figure 40 on page 305, and Figure 41 on page 306 show the create table and create index statements for pruning control tables.

For DB2 for OS/390 Version 4 or Higher

```
CREATE TABLE ASN.IBMSNAP_PRUNCNTL (  
  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))  
DATA CAPTURE CHANGES;
```

```
CREATE TYPE 2 UNIQUE INDEX ASN.IBMSNAP_PRUNCNTLX ON ASN.IBMSNAP_PRUNCNTL  
  (SOURCE_OWNER ASC, SOURCE_TABLE ASC, SOURCE_VIEW_QUAL ASC, SET_NAME ASC,  
  TARGET_SERVER ASC, TARGET_TABLE ASC, TARGET_OWNER ASC);
```

Figure 39. Create Table and Create Index Statements for DB2 for OS/390 Version 4 or Higher Pruning Control Tables

For DB2 for VM/VSE

```
CREATE TABLE ASN.IBMSNAP_PRUNCNTL (  
  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))  
  DATA CAPTURE CHANGES;  
  
CREATE UNIQUE INDEX ASN.IBMSNAP_PRUNCNTLX ON ASN.IBMSNAP_PRUNCNTL  
  (SOURCE_OWNER ASC, SOURCE_TABLE ASC, SOURCE_VIEW_QUAL ASC,  
  SET_NAME ASC, TARGET_SERVER ASC, TARGET_TABLE ASC, TARGET_OWNER ASC);
```

Figure 40. Create Table and Create Index Statements for DB2 for VM/VSE Pruning Control Tables

Register Table

For All Other Platforms

```
CREATE TABLE ASN.IBMSNAP_PRUNCNTL (  
  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)  
  <additional padding columns for DB2 for OS/390 Version 3>  
  DATA CAPTURE CHANGES;  
  
CREATE UNIQUE INDEX ASN.IBMSNAP_PRUNCNTLX ON ASN.IBMSNAP_PRUNCNTL  
  (SOURCE_OWNER ASC, SOURCE_TABLE ASC, SOURCE_VIEW_QUAL ASC,  
  SET_NAME ASC, TARGET_SERVER ASC, TARGET_TABLE ASC, TARGET_OWNER ASC);
```

Figure 41. Create Table and Create Index Statements for Pruning Control Tables on All Other Platforms

To avoid clone subscriptions, the APPLY_QUAL column is not included in the index. Instead, you must post each entry during subscription definition.

Register Table

This section contains programming interface information.

ASN.IBMSNAP_REGISTER

The register table holds information about replication source objects at the source server.

Table 39 provides a brief description of the register table columns.

Table 39 (Page 1 of 5). Register Table Columns

Column name	Description
SOURCE_OWNER	The middle qualifier of the source table, which names the owner of the source table whose updates are being captured.
SOURCE_TABLE	The third qualifier of the source table, which names the source table whose updates are being captured.
SOURCE_VIEW_QUAL	This value is set to equal 0 for registered physical tables and is greater than 0 for view registration.

Register Table

Table 39 (Page 2 of 5). Register Table Columns

Column name	Description
GLOBAL_RECORD	<p>A flag that indicates whether this row is the global record. This record is generated by the Capture program. If the Capture program is not installed, then there is no global record.</p> <p>Y This row is the global record. N This row is not the global record.</p>
SOURCE_STRUCTURE	<p>A value that identifies structure of the base table:</p> <p>1 Source table 3 CCD table 4 Point-in-time copy 5 Base aggregate copy 6 Change aggregate copy 7 Replica 8 User copy</p>
SOURCE_CONDENSED	<p>A flag that indicates:</p> <p>Y Changes can be netted out, with at most one row in the base table for every original table primary key value. N All changes must remain, retaining a complete update history. A Valid only for base aggregate or change aggregate tables.</p>
SOURCE_COMPLETE	<p>A flag indicating:</p> <p>Y The base table contains a row for every primary key value of interest. N The base table contains some subset of rows of primary key values.</p>
CD_OWNER	The owner of the change data table.
CD_TABLE	The name of the staging table 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.
PHYS_CHANGE_OWNER	The owner of the PHYS_CHANGE_TABLE. For a view registration, the value equals the value of the change data replication source referenced in the change data view definition. The Capture program uses this value to properly maintain CD_OLD_SYNCPOINT and CD_NEW_SYNCPOINT for view replication sources. The Apply program uses this value to properly maintain CCD_OLD_SYNCPOINT and SYNCPOINT for view registrations based on CCD target table registrations that the Apply program maintains.
PHYS_CHANGE_TABLE	The name of the physical CD or CCD table. For a view replication source, the value equals the value of the change data table replication source definition referenced in the change data view definition. The Capture program uses this value to properly maintain CD_OLD_SYNCPOINT and CD_NEW_SYNCPOINT for view replication sources. The Apply program uses this value to properly maintain CCD_OLD_SYNCPOINT and SYNCPOINT for view registrations based on the CCD target table registrations that the Apply program maintains.

Register Table

Table 39 (Page 3 of 5). Register Table Columns

Column name	Description
CD_OLD_SYNCHPOINT	The SYNCHPOINT value of the last cold start of the Capture program. The Capture program sets this value to NULL when the CD_TABLE is emptied. 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, which 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	<p>The SYNCHPOINT value associated with the most recent change inserted into the change data table. If the Capture program has not inserted into the change data table recently, then the value does not advance.</p> <p>This value reduces the need to prepare and open cursors on change data tables. CD_NEW_SYNCHPOINT can reliably indicate a <i>no-change</i> condition only when CD_NEW_SYNCHPOINT is less than or equal to the SYNCHPOINT column of the subscription set table. Change conditions are not reported as reliably because uncommitted work will cause this value to increase, even when there are no committed updates to replicate.</p>
DISABLE_REFRESH	<p>A flag that indicates whether full refresh queries are issued against the source table if a change data table becomes invalid, if a gap is detected, or if the Capture program cold starts. This flag prevents full refresh activity from overloading the source database when the Capture program is restarted. You can set this flag manually, use a program at the source database site to set it, or have the Capture program set it automatically while executing an UPDATE operation.</p> <p>0 Full refreshes are enabled. 1 Full refreshes are prevented.</p> <p>This column is initialized to 0.</p>
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 as copied from a join of the local change data table (defined by a subscription definition, which names the source table) and the unit-of-work table.
CCD_OLD_SYNCHPOINT	<p>The SYNCHPOINT value of the oldest row in the external consistent change data table. This value is set in one of the following ways:</p> <ul style="list-style-type: none"> • By the Control Center when the consistent change data table is being auto-registered. 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.
SYNCHPOINT	The SYNCHPOINT value equals the SYNCHPOINT field value in the register table. This value is used to coordinate the pruning of change data tables. The Apply program sets this initial value to 0, indicating refresh. If the Apply program sets a nonzero value, the change data table can be eligible for pruning.

Register Table

Table 39 (Page 4 of 5). Register Table Columns

Column name	Description
SYNCHTIME	A source server timestamp that can be added to any captured log records for any source table. This timestamp indicates that a change did not occur before this time. If the log records are individually timestamped, use those timestamps; otherwise, these values are approximate and are set by the Apply program at the start of a subscription cycle and after each subsequent cycle. These values are eventually appended into each change data table row.
CCD_CONDENSED	A flag indicating: <ul style="list-style-type: none"> Y Changes can be netted out, with at most one row in the CCD table for every base table primary key value. N All changes must remain, retaining a complete update history.
CCD_COMPLETE	A flag indicating that: <ul style="list-style-type: none"> Y The CCD table contains a row for every primary key value of interest. N The CCD table contains some subset of rows of primary key values.
ARCH_LEVEL	The architectural level of the definition in the row. This level is defined by IBM, and for Version 5 is '0201'.
DESCRIPTION	A field for comments that you enter when defining replication sources.
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: <ul style="list-style-type: none"> 1 For an ASCII single-byte character system prefix character. 1 For 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	A flag that indicates: <ul style="list-style-type: none"> 2 Enhanced detection with cascading transaction rejection. 1 Standard detection (no log flush to CD) with cascading transaction rejection. 0 No conflict detection. <p>CONFLICT_LEVEL is assumed to never change and to be the same for all descendents of the user table.</p>

Register Table

Table 39 (Page 5 of 5). Register Table Columns

Column name	Description
PARTITION_KEYS_CHG	<p>A flag indicating:</p> <ul style="list-style-type: none">N The source table columns that are included in the partitioning keys are not updated by applications, only deleted or inserted. The Capture program will capture updates in a single row with IBMSNAP_OPERATION = 'U'.Y The source table columns that are included in the partitioning keys are not updated by applications. Updates are captured as two rows in the CD table with ascending IBMSNAP_INTENTSEQ values. The first row contains the IBMSNAP_OPERATION value, where 'D' = all other deletions, meaning the before-image of all columns. The second row contains the IBMSNAP_OPERATION value, where 'I' = all other insertions, meaning the after-image of all columns. The delete must have a lower IBMSNAP_INTENTSEQ value than the insert, so that the updated partitioning key will not allow out-migration from a particular partition. In such cases, the insert must come last so that the row will remain in the partition. <p>NULL If this is the global control row.</p> <p>This value is assumed to be the same for all the user table's dependent replicas.</p>

Figure 42 on page 311 and Figure 43 on page 312 show the create table and create index statements for register tables.

DB2 for OS/390 Version 4 or Higher

```
CREATE TABLE ASN.IBMSNAP_REGISTER (  
  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));  
  
CREATE TYPE 2 UNIQUE INDEX ASN.IBMSNAP_REGISTERX ON  
  ASN.IBMSNAP_REGISTER (SOURCE_OWNER ASC, SOURCE_TABLE ASC,  
  SOURCE_VIEW_QUAL ASC);
```

Figure 42. Create Table and Create Index Statements for DB2 for OS/390 Version 4 or Higher Register Tables

Trace Table

For All Other Platforms

```
CREATE TABLE ASN.IBMSNAP_REGISTER (  
  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);  
<additional padding columns for DB2 for OS/390 Version 3>;  
  
CREATE UNIQUE INDEX ASN.IBMSNAP_REGISTERX ON  
  ASN.IBMSNAP_REGISTER (SOURCE_OWNER ASC, SOURCE_TABLE ASC,  
  SOURCE_VIEW_QUAL ASC);
```

Figure 43. Create Table and Create Index Statements for Register Tables on All Other Platforms

Trace Table

ASN.IBMSNAP_TRACE

This table contains trace information for Capture for MVS, Capture for VM, Capture for VSE, and the DB2 Universal Database.

On cold start, all of the trace table's entries will be deleted.

The message numbers ASN0100I, ASN0101W, ASN0102W are issued for warnings and initialization information.

Table 40 on page 313 provides a brief description of the trace table columns.

Tuning Parameters Table

Table 40. Trace Table Columns

Column name	Description
OPERATION	The type of Capture program operation, for example, initialization, capture, or error condition.
TRACE_TIME	The time 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 and is used by IBM service personnel.

Figure 44 show the create table statements for trace tables.

For All Platforms

```
CREATE TABLE ASN.IBMSNAP_TRACE (  
  OPERATION CHAR(8) NOT NULL,  
  TRACE_TIME TIMESTAMP NOT NULL,  
  DESCRIPTION VARCHAR(254) NOT NULL);
```

Figure 44. Create Table Statement for Trace Tables on All Platforms

Tuning Parameters Table

ASN.IBMSNAP_CCPPARMS

This table contains parameters that you can modify to control the performance of the Capture program.

Table 41 provides a brief description of the tuning parameters table columns.

Table 41 (Page 1 of 2). 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 by all clients. Transactions rejected after update conflict detection will have their changes pruned by RETENTION_LIMIT aging, not by normal pruning. The default value is 10,800.
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 10,800.

Unit-of-Work Table

Table 41 (Page 2 of 2). Tuning Parameters Table Columns

Column name	Description
COMMIT_INTERVAL	The Capture program commit threshold, in seconds, for any inserts, updates, or deletes to any CD tables, pruning control tables, and the global ASN.IBMSNAP_UOW table. 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 will save space, but will increase processing costs. Values set higher require more CD and UOW table space but decrease processing costs.

Figure 45 shows the create table statement for the tuning parameter table.

For All Platforms

```
CREATE TABLE ASN.IBMSNAP_CCPPARMS (  
  RETENTION_LIMIT INT,  
  LAG_LIMIT INT,  
  COMMIT_INTERVAL INT,  
  PRUNE_INTERVAL INT);
```

Figure 45. Create Table Statement for the Tuning Parameter Table for All Platforms

Unit-of-Work Table

This section contains programming interface information.

ASN.IBMSNAP_UOW

The UOW table ensures data integrity by recording transactions that were committed at the source server. By performing a join of this table and the changes logged by the Capture program, the Apply program ensures that only committed changes are being copied. On cold start, all of this table's entries will be deleted.

This table is indexed into the CD table, ordered by the commit sequence. This order is necessary because:

- The CD table is assumed to contain uncommitted changes.
- The CD table cannot be ordered in commit sequence without updating records after insertion, because the relative timing of row updates might not be reflected in the relative timing of commits.

This table is created automatically when you define a replication source. The existence and use of this table are required if the Capture program is installed.

Unit-of-Work Table

The Capture program prunes the UOW table based on information inserted into the pruning control table by the Apply program. The Apply program maintains the log sequence number in the SYNCHPOINT column of the pruning control table. Initially, the Apply program sets this sequence number to zero when it performs a full refresh. A zero value signals the Capture program to start capturing. When the Apply program copies changes from the CD table to the target table, it updates the SYNCHPOINT column.

Pruning occurs depending on whether you start the Capture program with the PRUNE or NOPRUNE invocation parameter and how the prune interval is set in the tuning parameters table. If you don't specify a parameter, PRUNE is the default. See the Capture and Apply chapter for your platform in this book to learn how to set pruning.

Table 42 provides a brief description of the UOW table columns.

Table 42. UOW Table Columns

Column name	Description
IBMSNAP_UOWID	The unit-of-recovery ID from the log record header for this unit of work.
IBMSNAP_COMMITSEQ	The transaction commit sequencing value.
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 MVS, this field is the correlation ID. This column is not automatically copied to other tables; you must select it and copy it as a user data column.
IBMSNAP_AUTHID	The authorization ID associated with the transaction. It is useful for database auditing. For DB2 for MVS, this field is the primary authorization ID. This column is not automatically copied to other tables; you must select it and copy it as a user data column.
IBMSNAP_REJ_CODE	A value indicating: 0 For a transaction with no known conflict 1 For a transaction containing at least one same-row intra-table update conflict 2 For a cascade-rejection of a transaction dependent on a prior transaction having at least one same-row conflict 3 For a transaction containing at least one constraint conflict, such as an inter-table referential constraint conflict 4 For a cascade-rejection of a transaction dependent on a prior transaction having at least one constraint conflict
IBMSNAP_APPLY_QUAL	This column is blank for local updates and the name of the associated Apply program for foreign updates. The Capture program derives this value from the critical section table.

Figure 46 on page 316, Figure 47 on page 316, and Figure 48 on page 317 show the create table and create index statements for UOW tables.

Unit-of-Work Table

For DB2 for OS/390 Version 4 or Higher

```
CREATE TABLE ASN.IBMSNAP_UOW (  
  IBMSNAP_UOWID CHAR(10) FOR BIT DATA NOT NULL,  
  IBMSNAP_COMMITSEQ CHAR(10) FOR BIT DATA NOT NULL,  
  IBMSNAP_LOGMARKER 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);  
  
CREATE TYPE 2 UNIQUE INDEX ASN.IBMSNAP_UOW_IDX ON ASN.IBMSNAP_UOW (  
  IBMSNAP_COMMITSEQ ASC, IBMSNAP_UOWID ASC, IBMSNAP_LOGMARKER ASC);
```

Figure 46. Create Table and Create Index Statements for DB2 for OS/390 Version 4 or Higher Unit-of-Work Tables

For DB2 for VM/VSE

```
CREATE TABLE ASN.IBMSNAP_UOW (  
  IBMSNAP_UOWID CHAR(10) FOR BIT DATA NOT NULL,  
  IBMSNAP_COMMITSEQ CHAR(10) FOR BIT DATA NOT NULL,  
  IBMSNAP_LOGMARKER TIMESTAMP NOT NULL,  
  IBMSNAP_AUTHTKN CHAR(12) NOT NULL,  
  IBMSNAP_AUTHID CHAR(18) NOT NULL),  
  IBMSNAP_REJ_CODE CHAR(1) NOT NULL,  
  IBMSNAP_APPLY_QUAL CHAR(18) NOT NULL);  
  
CREATE UNIQUE INDEX ASN.IBMSNAP_UOW_IDX ON ASN.IBMSNAP_UOW (  
  IBMSNAP_COMMITSEQ ASC, IBMSNAP_UOWID ASC, IBMSNAP_LOGMARKER ASC);
```

Figure 47. Create Table and Create Index Statements for DB2 for VM/VSE Unit-of-Work Tables

For All Other Platforms

```
CREATE TABLE ASN.IBMSNAP_UOW (
  IBMSNAP_UOWID CHAR(10) FOR BIT DATA NOT NULL,
  IBMSNAP_COMMITSEQ CHAR(10) FOR BIT DATA NOT NULL,
  IBMSNAP_LOGMARKER 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);

CREATE UNIQUE INDEX ASN.IBMSNAP_UOW_IDX ON ASN.IBMSNAP_UOW (
  IBMSNAP_COMMITSEQ ASC, IBMSNAP_UOWID ASC, IBMSNAP_LOGMARKER ASC);
```

Figure 48. Create Table and Create Index Statements for Unit-of-Work Tables on All Other Platforms

Warm Start Table

ASN.IBMSNAP_WARM_START

This table contains information that enables the Capture program to resynchronize (see the warm start section of the Capture and Apply programs chapter for your platform). This table is created in the same database as the register table.

Table 43 provides a brief description of the warm start table columns.

Table 43. Warm Start Table Columns

Column name	Description
SEQ	The log RBA for this unit of work. Records the Capture program's position in the DBMS log. Used for quickly restarting following a shutdown or failure.
AUTHTKN	The DB2 token for the unit of work associated with the SEQ position.
AUTHID	The DB2 authorization ID for the unit of work associated with the SEQ position.
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 (TOD), or Windows NT, HP-UX, Sun Solaris, OS/2 and AIX coordinated universal time (CUT) clock indicating when the unit of work associated with the SEQ position was captured (source server timestamp).

Figure 49 on page 318 show the create table statements for the warm start table.

Warm Start Table for Capture for VSE and VM

For All Platforms

```
CREATE TABLE ASN.IBMSNAP_WARM_START (  
  SEQ CHAR(10) FOR BIT DATA,  
  AUTHTKN CHAR(12),  
  AUTHID CHAR(18),  
  CAPTURED CHAR(1),  
  UOWTIME INT);
```

Figure 49. Create Table Statement for the Warm Start Table on All Platforms

Warm Start Table for Capture for VSE and VM

ASN.IBMSNAP_WARM_START

This table contains information that enables the Capture program to resynchronize for VSE and VM (see the warm start section for the Capture and Apply programs for VSE and VM). This table is created in the same database as the register table.

Table 44 provides a brief description of the columns in the warm start table for Capture for VSE and VM.

Table 44. Warm Start for Capture for VSE and VM Table Columns

Column name	Description
SEQ	The log RBA for this unit of work. Records the Capture program's position in the DBMS log. 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 position.
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 or VM time of day (TOD) clock indicating when the unit of work associated with the SEQ position was captured (source server timestamp).

Figure 50 on page 319 shows the create table statement for warm start tables for Capture for VSE and VM.

For DB2 for VM/VSE

```
CREATE TABLE ASN.IBMSNAP_WARM_START (
  SEQ CHAR(10) FOR BIT DATA,
  UOWID CHAR(10) FOR BIT DATA
  AUTHID CHAR(18),
  CAPTURED CHAR(1),
  UOWTIME INT);
```

Figure 50. Create Table Statement for Warm Start Tables for Capture for VSE and VM

Control Tables Used at the Control Server

The following section provides a brief description of the control tables used at the control server, the columns in each control table, and the create table and create index statements for each table. These tables are automatically created when you define subscriptions. The structure of all control tables is subject to change.

Apply Trail Table

ASN.IBMSNAP_APPLYTRAIL

The Apply trail table records a history of updates performed against subscriptions. The subscription statistics can be used to audit update activity. This table is a repository of diagnostics and performance statistics. Because this table is not automatically pruned, it is up to you to do so.

Table 45 provides a brief description of the Apply trail table columns.

Table 45 (Page 1 of 3). Apply Trail Table Columns

Column name	Description
APPLY_QUAL	Identifies the Apply program for the platform instance that will run this subscription. 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_NAME	Names a subscription set. This value is unique within an Apply qualifier.
WHOS_ON_FIRST	Allows both the up and down replication subscriptions to be contained in the same set without the potential for multiple SOURCE_SERVER values for each APPLY_QUAL, SET_NAME pairing. When the value of this column is: <ul style="list-style-type: none"> F (first) The target table is the user table or parent replica and the source table is the dependent replica-propagation is UP the hierarchy. 'F' is not used for read-only subscriptions. S (second) The source table is the user table, parent replica or other source and the target table is the dependent replica or other copy which is lower in the hierarchy-propagation is DOWN the hierarchy. 'S' is also used for read-only subscriptions.

Apply Trail Table

Table 45 (Page 2 of 3). Apply Trail Table Columns

Column name	Description
ASNLOAD	<p>One of the following values:</p> <p>Y or N Indicates whether ASNLOAD was called during the subscription set process.</p> <p>NULL Indicates that an error occurred before the decision to call ASNLOAD was made.</p>
MASS_DELETE	<p>One of the following values:</p> <p>Y or N Indicates whether MASS_DELETE was triggered during the full refresh of a subscription set.</p> <p>NULL Indicates that an error occurred before the decision to issue a mass delete was made.</p>
EFFECTIVE_MEMBERS	The number of members associated with calls to ASNLOAD, or the number of members for which rows are fetched and inserted, updated, or deleted.
SET_INSERTED	The total number of rows individually inserted into any set members during the subscription cycle.
SET_DELETED	The total number of rows individually deleted from any set members during the subscription cycle.
SET_UPDATED	The total number of rows individually updated in any set members during the subscription cycle.
SET_REWORKED	The total number of inserts reworked as updates and updates reworked as inserts for any set members during the subscription cycle.
SET_REJECTED_TRXS	The total number of rejected replica transactions due to a direct or cascading update conflict. Always zero if either source or target is a replica.
STATUS	<p>A value that represents in-progress and completed work status for the Apply program.</p> <p>-1 A known failed execution. You can modify the definition.</p> <p>0 A stable definition row that can be modified. This value is the default.</p> <p>1 A pending or in-progress execution. Do not modify this definition or any rows related to this subscription in other control tables.</p> <p>2 A continuing execution of a single logical subscription that was divided according to the MAX_SYNC_MINUTES control column and is being serviced by multiple subscription cycles. Do not modify this row or any row related to this subscription in other control tables.</p> <p>You cannot set the completion state at the control server until the copy is refreshed.</p>
LASTRUN	The estimated time that the last subscription definition began. This value is calculated by adding the LASTRUN value with the INTERVAL_MINUTES value.
LASTSUCCESS	The control server wall clock time of the beginning of a successful subscription cycle, recorded at the end of the cycle if STATUS=0 or STATUS=2. If not, existing value is unchanged.

Apply Trail Table

Table 45 (Page 3 of 3). Apply Trail Table Columns

Column name	Description
SYNCHPOINT	The SYNCHPOINT value equals the SYNCHPOINT field value in the register table. This value is used to coordinate the pruning of change data tables. The Apply program sets this initial value to 0, indicating refresh. If the Apply program sets a nonzero value, the change data table can be eligible for pruning.
SYNCHTIME	A source server timestamp that can be added to any captured log records for any source table. This timestamp indicates that a change did not occur before this time. If the log records are individually timestamped, use those timestamps; otherwise, these values are approximate and are set by the Apply program at the start of a subscription cycle and after each subsequent cycle. These values are eventually appended into each change data table row.
SOURCE_SERVER	The RDB name of DB2 for MVS, DB2 for VSE, and DB2 for VM where the source tables and views are defined.
SOURCE_ALIAS	The name of the source server used by the Apply program on the DB2 Universal Database client interface. This alias identifies the location of the subscription definition and is later used to perform administrative actions on a subscription. This alias does not necessarily match the alias of the source server used by the Apply program for the DB2 Universal Database and can be null if the database has no DB2 Universal Database name.
SOURCE_OWNER	The middle qualifier of the source table, which names the owner of the source table whose updates are being captured.
SOURCE_TABLE	The third qualifier of the source table, which names the source table whose updates are being captured.
SOURCE_VIEW_QUAL	Supports join subscriptions by matching the similar column in the register table.
TARGET_SERVER	The RDB name of the MVS, VSE, or VM server where the target table and views are defined.
TARGET_ALIAS	The name of the target server used by the Apply program on the DB2 Universal Database client interface. This alias identifies the location of the subscription definition and is later used to perform administrative actions on a subscription. This alias does not necessarily match the alias of the target server used by the Apply program for the DB2 Universal Database.
TARGET_OWNER	The middle qualifier of the target name. Use the value in the SOURCE_OWNER column as the default.
TARGET_TABLE	The third qualifier of the target table name. Use the value in the SOURCE_TABLE column as the default.
SQLSTATE	If an error, the SQL error code. Otherwise, NULL.
SQLCODE	If an error, the database-specific SQL error code. Otherwise, NULL.
SQLERRP	If an error, the database product identifier. Otherwise, NULL.
SQLERRM	If an error, the string containing the SQL error. Otherwise, NULL.
APPERRM	The Apply error message text from SQLCA if there was an error, otherwise NULL. This value is constant for each copy derived, directly or indirectly, from the original source table.

Figure 51 on page 322 shows the create table statement for the Apply trail table.

Subscription Columns Table

For All Platforms

```
CREATE TABLE ASN.IBMSNAP_APPLYTRAIL (  
  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 INT,  
  SQLERRP CHAR(8),  
  SQLERRM VARCHAR(70),  
  APPERRM VARCHAR(760) );
```

Figure 51. Create Table Statement for the Apply Trail Table

Subscription Columns Table

This section contains programming interface information.

ASN.IBMSNAP_SUBS_COLS

This table contains information on the subscription columns being copied in a replication subscription. The subscription columns table contains supplemental information to the subscription targets member table. For example, it contains the replication subscription column name, target owner column, and target tables where the target members exists; whether the column is part of the primary key or not; and the SQL expression necessary to generate an aggregate column.

Subscription Columns Table

Table 46 on page 323 provides a brief description of the subscription columns table columns.

Table 46. Subscription Columns Table Columns

Column name	Description
APPLY_QUAL	Identifies the Apply program for the platform instance that will run this subscription. 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_NAME	Names a subscription set. This value is unique within an Apply qualifier.
WHOS_ON_FIRST	Allows both the up and down replication subscriptions to be contained in the same set without the potential for multiple SOURCE_SERVER values for each APPLY_QUAL, SET_NAME pairing. When the value of this column is: F (first) The target table is the user table or parent replica and the source table is the dependent replica-propagation is UP the hierarchy. 'F' is not used for read-only subscriptions. S (second) The source table is the user table, parent replica or other source and the target table is the dependent replica or other copy which is lower in the hierarchy-propagation is DOWN the hierarchy. 'S' is also used for read-only subscriptions.
TARGET_OWNER	The middle qualifier of the target name. Use the value in the SOURCE_OWNER column as the default.
TARGET_TABLE	The third qualifier of the target table name. Use the value in the SOURCE_TABLE column as the default.
COL_TYPE	A flag indicating: A For after-image column. B For before-image column. C For computed column without SQL column function reference. F For computed column with SQL column function reference. R Signifies a relative record number column, provided by the system and used as a primary key column. Used by only DPROPR for OS/400.
TARGET_NAME	The name of the target table column. It does not need to match the source column name. CCD column names cannot be renamed columns. 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 identifier or source column expression.

Figure 52 on page 324 and Figure 53 on page 324 show the create table and create index statements for the subscription columns table.

Subscription Events Table

For DB2 for OS/390 Version 4 or Higher

```
CREATE TABLE ASN.IBMSNAP_SUBS_COLS (  
  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);  
  
CREATE TYPE 2 UNIQUE INDEX ASN.IBMSNAP_SUBS_COLSX ON ASN.IBMSNAP_SUBS_COLS  
  (APPLY_QUAL ASC, SET_NAME ASC, WHOS_ON_FIRST ASC, TARGET_OWNER  
  ASC, TARGET_TABLE ASC, TARGET_NAME ASC);
```

Figure 52. Create Table and Create Index Statements for DB2 for OS/390 Version 4 or Higher Subscription Columns Tables

For All Other Platforms

```
CREATE TABLE ASN.IBMSNAP_SUBS_COLS (  
  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);  
  
CREATE UNIQUE INDEX ASN.IBMSNAP_SUBS_COLSX ON ASN.IBMSNAP_SUBS_COLS  
  (APPLY_QUAL ASC, SET_NAME ASC, WHOS_ON_FIRST ASC, TARGET_OWNER  
  ASC, TARGET_TABLE ASC, TARGET_NAME ASC);
```

Figure 53. Create Table and Create Index Statements for Subscription Columns Tables on All Other Platforms

Subscription Events Table

This section contains programming interface information.

ASN.IBMSNAP_SUBS_EVENT

Subscription Events Table

This table contains information on the events being copied in a replication subscription. The common subscription events table contains event names and timestamps associated with the event names.

Table 47 provides a brief description of the subscription events table columns.

Table 47. Subscription Events Table Columns

Column name	Description
EVENT_NAME	A globally unique character string in a global name space configuration or 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. An upper-bound function that blocks replication 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.

Figure 54 and Figure 55 on page 326 show the create table and create index statements for the subscription events table columns.

For DB2 for OS/390 Version 4 or Higher

```
CREATE TABLE ASN.IBMSNAP_SUBS_EVENT (  
    EVENT_NAME CHAR(18) NOT NULL,  
    EVENT_TIME TIMESTAMP NOT NULL,  
    END_OF_PERIOD TIMESTAMP );
```

```
CREATE TYPE 2 UNIQUE INDEX ASN.IBMSNAP_SUBS_EVENTX ON  
ASN.IBMSNAP_SUBS_EVENT (EVENT_NAME ASC, EVENT_TIME ASC);
```

Figure 54. Create Table and Create Index Statements for DB2 for OS/390 Version 4 or Higher Subscription Events Tables

Subscription Set Table

For All Other Platforms

```
CREATE TABLE ASN.IBMSNAP_SUBS_EVENT (  
    EVENT_NAME CHAR(18) NOT NULL,  
    EVENT_TIME TIMESTAMP NOT NULL,  
    END_OF_PERIOD TIMESTAMP );  
  
CREATE UNIQUE INDEX ASN.IBMSNAP_SUBS_EVENTX ON  
    ASN.IBMSNAP_SUBS_EVENT (EVENT_NAME ASC, EVENT_TIME ASC);
```

Figure 55. Create Table and Create Index Statements for Subscription Events Tables on All Other Platforms

Subscription Set Table

This section contains programming interface information.

ASN.IBMSNAP_SUBS_SET

The subscription set table lists all the sets defined at the Control Center and ensures that each set name is used only once for every Apply qualifier.

Table 48 provides a brief description of the subscription set table columns.

Table 48 (Page 1 of 3). Subscription Set Table Columns

Column name	Description
APPLY_QUAL	Identifies the Apply program for the platform instance that will run this subscription. 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_NAME	Names a subscription set. This value must be unique within an Apply qualifier.
WHOS_ON_FIRST	Allows both the up and down replication subscriptions to be contained in the same set without the potential for multiple SOURCE_SERVER values for each APPLY_QUAL, SET_NAME pairing. When the value of this column is: F (first) The target table is the user table or parent replica and the source table is the dependent replica-propagation is UP the hierarchy. 'F' is not used for read-only subscriptions. S (second) The source table is the user table, parent replica or other source and the target table is the dependent replica or other copy which is lower in the hierarchy-propagation is DOWN the hierarchy. 'S' is also used for read-only subscriptions.
ACTIVATE	Indicates: 0 The subscription set is deactivated. 1 The request is active indefinitely. 2 The set can be copied immediately.

Subscription Set Table

Table 48 (Page 2 of 3). Subscription Set Table Columns

Column name	Description
SOURCE_SERVER	The RDB name of the source server where the source tables and views are defined.
SOURCE_ALIAS	The name of the source server used by the Apply program on the DB2 Universal Database client interface. This alias identifies the location of the subscription definition and is later used to perform administrative actions on a subscription. This alias does not necessarily match the alias of the source server used by the Apply program for the DB2 Universal Database and can be NULL if the database has no DB2 Universal Database name.
TARGET_SERVER	The RDB name of the server where the target table and views are defined.
TARGET_ALIAS	The name of the target server used by the Apply program on the DB2 Universal Database client interface. This alias identifies the location of the subscription definition and is later used to perform administrative actions on a subscription. This alias does not necessarily match the alias of the target server used by the Apply program for the DB2 Universal Database.
STATUS	<p>A value that represents in-progress and completed work status for the Apply program.</p> <ul style="list-style-type: none"> -1 A known failed execution. You can modify this definition. 0 A stable definition row that can be modified. This value is the default. 1 A pending or in-progress execution. Do not modify this definition or any rows related to this subscription 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 in other control tables. <p>You cannot set the completion state at the control server until the copy is refreshed.</p>
LASTRUN	The estimated time that the last subscription definition began. This value is calculated by adding the LASTRUN value with the INTERVAL_MINUTES value.
REFRESH_TIMING	<p>Sets the timing between statement executions.</p> <ul style="list-style-type: none"> 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 a subscription that has both relative and event timing specifications. <p>In all other instances, use SLEEP_MINUTES.</p>
SLEEP_MINUTES	Specifies the time of inactivity between statement executions.
EVENT_NAME	The optional foreign key into the subscription event table located at the control server.
LASTSUCCESS	The control server wall clock time of the beginning of a successful subscription cycle, recorded at the end of the cycle if STATUS=0 or STATUS=2. If not, the existing value is unchanged.

Subscription Set Table

Table 48 (Page 3 of 3). Subscription Set Table Columns

Column name	Description
SYNCHPOINT	The SYNCHPOINT value equals the SYNCHPOINT field value in the common subscription set table. This value is used to coordinate the pruning of change data tables. The Apply program sets this initial value to 0, indicating refresh. If the Apply program sets a nonzero value, the change data table can be eligible for pruning.
SYNCHTIME	A source server timestamp that can be added to any captured log records for any source table. This timestamp indicates that a change did not occur before this time. If the log records are individually timestamped, use those timestamps; otherwise, these values are approximate and are set by the Apply program at the start of a subscription cycle and after each subsequent cycle. These values are eventually appended into each change data table row.
MAX_SYNCH_MINUTES	A time-threshold limit to regulate the amount of change data to fetch and apply during a subscription cycle. The limit is automatically recalculated if the Apply program encounters a resource constraint making 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	A 0 value represents the absence of any subscription statements table rows for the subscription. Otherwise, this value represents the total number of subscription statement table rows.
ARCH_LEVEL	The architectural level of the definition contained in the row. This field identifies the rules under which a row was created.

Figure 56 on page 329 and Figure 57 on page 330 show the create table and create index statements for subscription set tables.

For DB2 for OS/390 Version 4 or Higher

```
CREATE TABLE ASN.IBMSNAP_SUBS_SET (  
  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,  
  
CREATE TYPE 2 UNIQUE INDEX ASN.IBMSNAP_SUBS_SETX ON ASN.IBMSNAP_SUBS_SET  
  (APPLY_QUAL ASC, SET_NAME ASC, WHOS_ON_FIRST ASC);
```

Figure 56. Create Table and Create Index Statements for DB2 for OS/390 Version 4 or Higher Subscription Set Tables

Subscription Statements Table

For All Other Platforms

```
CREATE TABLE ASN.IBMSNAP_SUBS_SET (  
  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,  
  <additional padding columns for DB2 for OS/390 version 3>);  
  
CREATE UNIQUE INDEX ASN.IBMSNAP_SUBS_SETX ON ASN.IBMSNAP_SUBS_SET  
  (APPLY_QUAL ASC, SET_NAME ASC, WHOS_ON_FIRST ASC);
```

Figure 57. Create Table and Create Index Statements for Subscription Set Tables on All Other Platforms

Subscription Statements Table

This section contains programming interface information.

ASN.IBMSNAP_SUBS_STMTS

This table contains information on the statements being executed in a replication subscription set. Execute immediately (EI) statements or stored procedures can be executed at the source or target server only. The subscription statements table contains zero rows if you did not specify any statements or stored procedures.

A unique index on the APPLY_QUAL, SET_NAME, WHOS_ON_FIRST, BEFORE_OR_AFTER, and STMT_NUMBER columns is created automatically by the Control Center.

Table 49 on page 331 provides a brief description of the subscription statements table columns.

Subscription Statements Table

Table 49. Subscription Statements Table Columns

Column name	Description
APPLY_QUAL	Identifies the Apply program for the platform instance that will run this subscription. 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_NAME	Names a subscription set. This value must be unique within an Apply qualifier.
WHOS_ON_FIRST	Allows both the up and down replication subscriptions to be contained in the same set without the potential for multiple SOURCE_SERVER values for each APPLY_QUAL, SET_NAME pairing. When the value of this column is: <ul style="list-style-type: none"> F (first) The target table is the user table or parent replica and the source table is the dependent replica-propagation is UP the hierarchy. 'F' is not used for read-only subscriptions. S (second) The source table is the user table, parent replica or other source and the target table is the dependent replica or other copy which is lower in the hierarchy-propagation is DOWN the hierarchy. 'S' is also used for read-only subscriptions.
BEFORE_OR_AFTER	A value indicating: <ul style="list-style-type: none"> 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 the opening of the answer set cursors.
STMT_NUMBER	Defines the relative order of execution within the scope of BEFORE_OR_AFTER.
EI_OR_CALL	A value indicating: <ul style="list-style-type: none"> E The SQL statement should be run as an EXEC SQL EXECUTE IMMEDIATE at the target server. C The SQL statement contains a stored procedure name to run as an EXEC SQL CALL at the target server.
SQL_STMT	One of the following values: <ul style="list-style-type: none"> 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 values are acceptable SQLSTATES. The values would otherwise cause the subscription to stop executing.

Figure 58 on page 332 and Figure 59 on page 332 show the create table and create index statements for the subscription statements table.

Subscription Targets Member Table

For DB2 for OS/390 Version 4 or Higher

```
CREATE TABLE ASN.IBMSNAP_SUBS_STMTS (  
  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(1024),  
  ACCEPT_SQLSTATES VARCHAR (50));  
  
CREATE TYPE 2 UNIQUE INDEX ASN.IBMSNAP_SUBS_STMTX ON ASN.IBMSNAP_SUBS_STMTS  
(APPLY_QUAL ASC, SET_NAME ASC, WHOS_ON_FIRST ASC, BEFORE_OR_AFTER ASC,  
  STMT_NUMBER ASC);
```

Figure 58. Create Table and Create Index Statements for DB2 for OS/390 Version 4 or Higher Subscription Statements Tables

For All Other Platforms

```
CREATE TABLE ASN.IBMSNAP_SUBS_STMTS (  
  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(1024),  
  ACCEPT_SQLSTATES VARCHAR (50));  
  
CREATE UNIQUE INDEX ASN.IBMSNAP_SUBS_STMTX ON ASN.IBMSNAP_SUBS_STMTS  
(APPLY_QUAL ASC, SET_NAME ASC, WHOS_ON_FIRST ASC, BEFORE_OR_AFTER ASC,  
  STMT_NUMBER ASC);
```

Figure 59. Create Table and Create Index Statements for Subscription Statements Tables on All Other Platforms

Subscription Targets Member Table

This section contains programming interface information.

ASN.IBMSNAP_SUBS_MEMBR

This table contains information about the individual member and target tables defined for a subscription set.

Subscription Targets Member Table

A unique index on APPLY_QUAL, SET_NAME, WHOS_ON_FIRST, SOURCE_OWNER, SOURCE_TABLE, SOURCE_VIEW_QUAL, TARGET_OWNER, and TARGET_TABLE is created automatically by the Control Center.

Table 50 provides a brief description of the subscription targets member table columns.

Table 50 (Page 1 of 2). Subscription Targets Member Table Columns

Column name	Description
APPLY_QUAL	Identifies the Apply program for the platform instance that will run this subscription. 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_NAME	Names a subscription set. This value must be unique within an Apply qualifier. For replicas, both the to-replica and from-replica subscriptions share the same APPLY_QUAL and SET_NAME values determined by their WHOS_ON_FIRST values.
WHOS_ON_FIRST	Allows both the up and down replication subscriptions to be contained in the same set without the potential for multiple SOURCE_SERVER values for each APPLY_QUAL, SET_NAME pairing. When the value of this column is: <ul style="list-style-type: none"> F (first) The target table is the user table or parent replica and the source table is the dependent replica-propagation is UP the hierarchy. 'F' is not used for read-only subscriptions. S (second) The source table is the user table, parent replica or other source and the target table is the dependent replica or other copy which is lower in the hierarchy-propagation is DOWN the hierarchy. 'S' is also used for read-only subscriptions.
SOURCE_OWNER	The middle qualifier of the source table, which names the owner of the source table whose updates are being captured.
SOURCE_TABLE	The third qualifier of the source table, which names the source table whose updates are being captured.
SOURCE_VIEW_QUAL	Supports the join of subscriptions by matching the similar column in the register table.
TARGET_OWNER	The middle qualifier of the target name. Use the value in the SOURCE_OWNER column as the default.
TARGET_TABLE	The third qualifier of the target table name. Use the value in the SOURCE_TABLE column as the default.
TARGET_CONDENSED	A flag indicating: <ul style="list-style-type: none"> Y Changes can be netted out, with at most one row in the base table for every original table primary key value. N All changes must remain, retaining a complete update history. A Valid only for base aggregate or change aggregate tables.

Subscription Targets Member Table

Table 50 (Page 2 of 2). Subscription Targets Member Table Columns

Column name	Description
TARGET_COMPLETE	A flag indicating: Y The base table contains a row for every primary key value of interest. N The base table contains some subset of rows of primary key values.
TARGET_STRUCTURE	The structure of the target table: 1 Source table 3 CCD table 4 Point-in-time table 5 Base aggregate table 6 Change aggregate table 7 Replica 8 User copy
PREDICATES	Lists the predicates to be placed in a WHERE clause to subset the view (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 as the Apply program can generate an ORDER BY clause. The PREDICATES column can be updated directly with a predicate in situations where an out-of-range insert is performed by a local application.

Figure 60 and Figure 61 on page 335 show the create table and create index statements for the subscription targets member table.

For DB2 for OS/390 Version 4 or Higher

```
CREATE TABLE ASN.IBMSNAP_SUBS_MEMBR (
  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) );
```

```
CREATE TYPE 2 UNIQUE INDEX ASN.IBMSNAP_SUBS_MEMIX ON ASN.IBMSNAP_SUBS_MEMBR
(APPLY_QUAL ASC, SET_NAME ASC, WHOS_ON_FIRST ASC, SOURCE_OWNER
ASC, SOURCE_TABLE ASC, SOURCE_VIEW_QUAL ASC, TARGET_OWNER ASC,
TARGET_TABLE ASC);
```

Figure 60. Create Table and Create Index Statements for DB2 for OS/390 Version 4 or Higher Subscription Targets Member Tables

Subscription Schema Changes Table

For All Other Platforms

```
CREATE TABLE ASN.IBMSNAP_SUBS_MEMBR (  
    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) );
```

```
CREATE UNIQUE INDEX ASN.IBMSNAP_SUBS_MEMIX ON ASN.IBMSNAP_SUBS_MEMBR  
(APPLY_QUAL ASC, SET_NAME ASC, WHOS_ON_FIRST ASC, SOURCE_OWNER  
ASC, SOURCE_TABLE ASC, SOURCE_VIEW_QUAL ASC, TARGET_OWNER ASC,  
TARGET_TABLE ASC);
```

Figure 61. Create Table and Create Index Statements for Subscription Targets Member Tables on All Other Platforms

To provide the necessary source/target member name uniqueness, you can introduce either source or target view names when there are multiple members with the same fully qualified source and target names.

In a cloned subscription set, the SOURCE_SERVER, SOURCE_OWNER, SOURCE_TABLE, SOURCE_VIEW_QUAL, TARGET_OWNER, TARGET_TABLE, and SET_NAME values are identical to the original subscription set. The cloned subscription is distinguished from the original by differing APPLY_QUAL values.

Subscription Schema Changes Table (Microsoft Jet Specific)

This section contains programming interface information.

ASN.IBMSNAP_SCHEMA_CHG

This table is used to signal that one or more of the following modifications has been made to a subscription:

- A new member has been added (ADDMEMBR)
- An existing member has been deleted (DELMEMBR)
- An existing member had an additional column added (ALTERMEM)

This table allows DataPropagator for Microsoft Jet to quickly determine that 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 51 provides a brief description of the subscription schema changes table columns.

Table 51. Subscription Schema Changes Table

Column name	Description
APPLY_QUAL	Identifies the Apply program for the platform instance that will run this subscription. 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_NAME	Names a subscription set. This value must be unique within an Apply qualifier.
LAST_CHANGED	This column is the timestamp of when this row was last changed in this table. This column is for informational purposes only.

Figure 62 shows the create table statement for the subscription schema changes table.

For Microsoft Jet

```
CREATE TABLE ASN.IBMSNAP_SCHEMA_CHG (  
  APPLY_QUAL CHAR (18) NOT NULL,  
  SET_NAME CHAR (18) NOT NULL,  
  LAST_CHANGED TIMESTAMP NOT NULL);
```

Figure 62. Create Table Subscription Schema Changes Table

Row-Replica Target List Table (Microsoft Jet Specific)

This section contains programming interface information.

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 member has been dropped since the last synchronization.

Table 52 on page 337 provides a brief description of the row-replica target list table columns.

Table 52. Row-Replica Target List Table Columns

Column name	Description
APPLY_QUAL	Identifies the Apply program for the platform instance that will run this subscription. 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_NAME	Names a subscription set. This value must be unique within an Apply qualifier.
WHOS_ON_FIRST	Allows both the up and down replication subscriptions to be contained in the same set without the potential for multiple SOURCE_SERVER values for each APPLY_QUAL, SET_NAME pairing.
TARGET_OWNER	The middle qualifier of the target table name; default is the user ID of the user defining the subscription. Matches the column of the same name in the subscription targets member table.
TARGET_TABLE	The third qualifier of the target table name. Matches the column of the same name in the subscription targets member table.
LAST_POSTED	This column is the timestamp of when this row was inserted into the table. This column is for informational purposes only.

Figure 63 shows the create table and create index statements for the row-replica target list table.

For Microsoft Jet

```
CREATE TABLE ASN.IBMSNAP_SUBS_TGTS (
  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,
  LAST_POSTED TIMESTAMP );
```

```
CREATE UNIQUE INDEX ASN.IBMSNAP_SUBS_TGTSX ON ASN.IBMSNAP_SUBS_TGTS
  (APPLY_QUAL, SET_NAME, WHOS_ON_FIRST, TARGET_OWNER, TARGET_TABLE);
```

Figure 63. Create Table and Create Index Statements for Row-Replica Target List Tables

Control Tables Used at the Target Server

The following section provides a brief description of the control tables used at the target server, the columns in each control table, and the create table and create index statements for each table. The structure of all target tables is application-dependent.

Base Aggregate Target Table

Base Aggregate Target Table

This section contains programming interface information.

userid.target_table

Base aggregate tables are target tables that contain data aggregated from a source table.

For base aggregate tables:

- A descending non-unique index on the IBMSNAP_LLOGMARKER table is created by the Control Center.
- 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 53 provides a brief description of the base aggregate target table columns.

Table 53. Base Aggregate Target Table Columns

Column name	Description
<i>user columns</i>	Columns computed from the base table.
IBMSNAP_LLOGMARKER	The oldest (lowest) IBMSNAP_LOGMARKER or IBMSNAP_LLOGMARKER value in the CD or CCD table rows being aggregated.
IBMSNAP_HLOGMARKER	The most recent (highest) IBMSNAP_LOGMARKER or IBMSNAP_LLOGMARKER value in the CD or CCD table rows being aggregated.

Figure 64 and Figure 65 on page 339 show the create table and create index statements for the base aggregate target table.

For DB2 for OS/390 Version 4 or Higher

```
CREATE TABLE userid.target_table(  
  IBMSNAP_LLOGMARKER TIMESTAMP,  
  IBMSNAP_HLOGMARKER TIMESTAMP,  
  <user data columns>,  
  <aggregate columns>);
```

```
CREATE TYPE 2 UNIQUE INDEX userid.IXtimestamp ON  
  userid.target_table (IBMSNAP_LLOGMARKER DESC);
```

Figure 64. Create Table and Create Index Statements for DB2 for OS/390 Version 4 or Higher Base Aggregate Target Tables

Change Aggregate Target Table

For All Other Platforms

```
CREATE TABLE userid.target_table(
  IBMSNAP_LLOGMARKER TIMESTAMP,
  IBMSNAP_HLOGMARKER TIMESTAMP,
  <user data columns>,
  <aggregate columns>);

CREATE UNIQUE INDEX userid.IXtimestamp ON
  userid.target_table (IBMSNAP_LLOGMARKER DESC);
```

Figure 65. Create Table and Create Index Statements for Base Aggregate Target Tables on All Other Platforms

Change Aggregate Target Table

This section contains programming interface information.

userid.target_table

A change aggregate table is a target table that contains data aggregations based on changes from a source table.

For change aggregate tables:

- A non-unique descending index on the IBMSNAP_LLOGMARKER column is created by the Control Center.
- Before-image user data columns must be nullable.
- Computed user data columns should be NOT NULL if your design can determine that the computation will never generate a null value, for example in a concatenation scenario.
- Because CCD tables can contain duplicate rows, the DISTINCT clause should be used when a CCD table is the replication source table of a change aggregate target table.

Table 54 provides a brief description of the change aggregate target table columns.

Table 54. Change Aggregate Target Table Columns

Column name	Description
<i>User columns, including computed columns</i>	These columns are computed from change data related to the base table.
IBMSNAP_LLOGMARKER	The oldest (lowest) IBMSNAP_LOGMARKER or IBMSNAP_LLOGMARKER value in the CD or CCD table rows being aggregated.
IBMSNAP_HLOGMARKER	The most recent (highest) IBMSNAP_LOGMARKER or IBMSNAP_LLOGMARKER value in the CD or CCD table rows being aggregated.

Consistent Change Data Table

Figure 66 on page 340 and Figure 67 on page 340 show the create table and create index statements for the change aggregate target table.

For DB2 for OS/390 Version 4 or Higher

```
CREATE TABLE userid.target_table (  
    IBMSNAP_LLOGMARKER TIMESTAMP NOT NULL,  
    IBMSNAP_HLOGMARKER TIMESTAMP NOT NULL);  
<user data columns>,  
<aggregate columns>);  
  
CREATE TYPE 2 INDEX userid.IXtimestamp ON  
    userid.target_table (IBMSNAP_LLOGMARKER DESC);
```

Figure 66. Create Table and Create Index Statements for DB2 for OS/390 Version 4 or Higher Change Aggregate Target Tables

For All Other Platforms

```
CREATE TABLE userid.target_table (  
    IBMSNAP_LLOGMARKER TIMESTAMP NOT NULL,  
    IBMSNAP_HLOGMARKER TIMESTAMP NOT NULL);  
<user data columns>,  
<aggregate columns>);  
  
CREATE INDEX userid.IXtimestamp ON  
    userid.target_table (IBMSNAP_LLOGMARKER DESC);
```

Figure 67. Create Table and Create Index Statements for Change Aggregate Target Tables on All Other Platforms

Consistent Change Data Table

This section contains programming interface information.

userid.target_table

CCD tables contain committed change data.

The CCD table can be:

- A table of data.
- A staging table maintained by one Apply program.
The result of a join between the CD table and the UOW table can be stored here, so that you perform the join step only once for fan-out copying.
- A secondary staging table.
- An external source table for nonrelational and multivendor data.

Consistent Change Data Table

The external source table allows IBM Replication to act as a “loading dock” and deliver transaction-consistent data from nonrelational sources.

For CCD tables:

- A non-unique ascending index on IBMSNAP_COMMITSEQ is created. This index improves the performance of updating CCD table copies.
- If the CCD table is condensed (CCD_CONDENSED = 'Y'), a unique index is required for user data primary key columns to maintain the CCD table. CCD copies with TARGET_CONDENSED = 'N' can contain duplicates if changes are reapplied following a failure of the Apply program.
- Before-image user data columns must be nullable and therefore cannot be part of a primary key for a condensed CCD table.
- 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.
- 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.
- 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.
- Noncondensed CCD tables can contain duplicate rows if changes are reapplied after a failure during the previous copy operation.
- 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 Changed Data” on page 59). 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.

CCD tables and point-in-time (PIT) tables are changed, rather than appended. The originally captured operation code in the IBMSNAP_OPERATION column and the sequence numbers IBMSNAP_INTENTSEQ and IBMSNAP_COMMITSEQ are copied into and out of CCD staging 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

Consistent Change Data Table

Exceptions:

- The operation code is I, but a unique index constraint causes the insert to fail because a row with the key already exists. The insert becomes an update.
- The operation code is U, but the update fails because no row exists with the key. The update becomes an insert.
- A delete fails with a “row not found” condition. The condition is ignored by the Apply program.
- For a condensed CCD, a delete is always handled as an update.

Table 55 provides a brief description of the CCD table columns.

Table 55. CCD Table Columns

Column name	Description
IBMSNAP_INTENTSEQ	Unique identifier for this change; it describes the sequence of a change within a transaction.
IBMSNAP_OPERATION	Character value of I, U, or D, indicating an insert, update, or delete record.
IBMSNAP_COMMITSEQ	The transaction commit sequencing value.
IBMSNAP_LOGMARKER	The approximate commit time at the source server.
<user data columns>	Columns from source tables specified while defining replication sources.

Figure 68 and Figure 69 on page 343 show the create table and create index statements for the CCD table.

For DB2 for OS/390 Version 4 or Higher

```
CREATE TABLE userid.target_table (  
  IBMSNAP_INTENTSEQ CHAR(10) FOR BIT DATA NOT NULL,  
  IBMSNAP_OPERATION CHAR(1) NOT NULL,  
  IBMSNAP_COMMITSEQ CHAR(10) FOR BIT DATA NOT NULL,  
  IBMSNAP_LOGMARKER TIMESTAMP NOT NULL,  
  <user data columns>);
```

```
CREATE TYPE 2 INDEX userid.IXtimestamp ON  
  userid.target_table(IBMSNAP_COMMITSEQ ASC);
```

For Condensed CCD Tables

```
CREATE TYPE 2 UNIQUE INDEX userid.IXtimestamp ON  
  userid.target_table (<primary key columns>);
```

Figure 68. Create Table and Create Index Statements for DB2 for OS/390 Version 4 or Higher CCD Tables

For All Other Platforms

```
CREATE TABLE userid.target_table (  
  IBMSNAP_INTENTSEQ CHAR(10) FOR BIT DATA NOT NULL,  
  IBMSNAP_OPERATION CHAR(1) NOT NULL,  
  IBMSNAP_COMMITSEQ CHAR(10) FOR BIT DATA NOT NULL,  
  IBMSNAP_LOGMARKER TIMESTAMP NOT NULL,  
  KEY1 CHAR(1),  
  DATA1 CHAR(1));
```

```
CREATE INDEX userid.IXtimestamp ON  
  userid.target_table(IBMSNAP_COMMITSEQ ASC);
```

For Condensed CCD Tables

```
CREATE UNIQUE INDEX userid.IXtimestamp ON  
  userid.target_table (<primary key columns>);
```

Figure 69. Create Table and Create Index Statements for CCD Tables on All Other Platforms

Point-in-Time Target Table

This section contains programming interface information.

userid.target_table

A point-in-time table contains an added 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 an image of the source table, but at some time in the past. Point-in-time copies reflect a valid state of the source table, but not necessarily the most current state.

For point-in-time tables:

- A unique index is automatically created and required for the primary key.
- Before-image user data columns must be nullable.
- Before-image columns should not be included in the primary key.
- 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.

CCD tables and point-in-time tables are changed, rather than appended.

Table 56 on page 344 provides a brief description of the point-in-time target table columns.

Replica Target Table

Table 56. Point-in-Time Target Table Columns

Column name	Description
<i>user key columns</i>	The primary key of the target table, although it is not necessarily a component of the primary key of the base table. You can use predicates to prevent a NULL value from being assigned to the key fields of any copies.
<i>user nonkey columns</i>	The nonkey data column from the base table.
IBMSNAP_LOGMARKER	The approximate commit time at the source server. This column is NULL following a full refresh.

Figure 70 and Figure 71 show the create table and create index statements for the point-in-time table.

For DB2 for OS/390 Version 4 or Higher

```
CREATE TABLE userid.target_table (  
  <user data columns>,  
  IBMSNAP_LOGMARKER TIMESTAMP NOT NULL);  
  
CREATE TYPE 2 UNIQUE INDEX userid.IXtimestamp ON  
  userid.target_table (<primary key columns>);
```

Figure 70. Create Table and Create Index Statements for DB2 for OS/390 Version 4 or Higher Point-In-Time Tables

For All Other Platforms

```
CREATE TABLE userid.target_table (  
  <user data columns>,  
  IBMSNAP_LOGMARKER TIMESTAMP NOT NULL);  
  
CREATE UNIQUE INDEX IXtimestamp ON userid.target_table (<primary key columns>);
```

Figure 71. Create Table and Create Index Statements for Point-In-Time Tables on All Other Platforms

Note: The primary key requires a unique index.

Replica Target Table

This section contains programming interface information.

userid.target_table

This table contains a primary key identical to the primary key of the user table. The replica must have the same primary key as the source table.

Table 57 on page 345 provides a brief description of the replica target table columns.

User Copy Target Table

Table 57. Replica Target Table Columns

Column name	Description
<i>user key columns</i>	The primary key of the target table, although it is not necessarily a component of the primary key of the base table. You can use predicates to prevent a NULL value from being assigned to the key fields of any copies.
<i>user nonkey columns</i>	The nonkey data columns from the base table.

Figure 72 and Figure 73 show the create table and create index statements for the replica target table

For DB2 for OS/390 Version 4 or Higher

```
CREATE TABLE userid.target_table (  
  <user data columns>  
  DATA CAPTURE CHANGES;  
  
CREATE TYPE 2 UNIQUE INDEX userid.IXtimestamp ON  
  userid.target_table (<primary key columns>);
```

Figure 72. Create Table and Create Index Statements for DB2 for OS/2 Version 4 or Higher Replica Target Tables

For All Other Platforms

```
CREATE TABLE userid.target_table (  
  <user data columns>  
  DATA CAPTURE CHANGES;  
  
CREATE UNIQUE INDEX userid.IXtimestamp ON userid.target_table (<primary key columns>);
```

Figure 73. Create Table and Create Index Statements for Replica Target Tables on All Other Platforms

User Copy Target Table

This section contains programming interface information.

userid.target_table

This table is the default target type and indicates a specific commit time from the source server. The user copy target 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 target table.

A user copy table reflects a valid state of the source table, except for subsetting and data enhancement, but not necessarily the most current state. Because the tables are

User Copy Target Table

physical objects, references to user copy target tables (or any other target table type) reduce the contention that results from too much 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.

For user copy target tables:

- A unique index is automatically created and required for the primary key.
- Before-image user data columns must be nullable.
- Before-image columns should not be included in the primary key.
- 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 58 provides a brief description of the user copy target table columns.

Table 58. User Copy Target Table Columns

Column name	Description
<i>user key columns</i>	The primary key of the target table, although it is not necessarily a component of the primary key of the base table. You can use predicates to prevent a NULL value from being assigned to the key fields of any copies.
<i>user nonkey columns</i>	The nonkey data columns from the base table.

Figure 74 and Figure 75 on page 347 shows the create table and create index statements for the user copy target table.

For DB2 for OS/390 Version 4 or Higher

```
CREATE TABLE userid.target_table (  
  <user data columns>);  
  
CREATE TYPE 2 UNIQUE INDEX userid.IXtimestamp ON  
  userid.target_table (<primary key columns>);
```

Figure 74. Create Table and Create Index Statements for DB2 for OS/390 Version 4 or Higher User Copy Target Tables

For All Other Platforms

```
CREATE TABLE userid.target_table (  
    <user data columns>);  
  
CREATE UNIQUE INDEX userid.IXtimestamp ON userid.target_table (<primary key columns>);
```

Figure 75. Create Table and Create Index Statements for User Copy Target Tables on All Other Platforms

The following table is a conflict table for tracking synchronization conflicts and errors. This Microsoft Jet database control table mimics Microsoft's conflict tables.

Conflict Table (Microsoft Jet Specific)

This section contains programming interface information.

IBMSNAP_<target name>_CONFLICT

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 59 provides a brief description of the conflict table columns.

Table 59. Conflict Table Columns

Column name	Description
<i>target name</i>	The corresponding row-replica's table name.
<i>column names of row-replica</i>	A list of column names found in the corresponding row-replica table.

Error Information Table (Microsoft Jet Specific)

This section contains programming interface information.

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 60 on page 348 provides a brief description of the error information table columns.

Error Messages Table

Table 60. 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 DPROPR error message number.

Figure 76 shows the create table statement for the error information table.

For Microsoft Jet

```
CREATE TABLE IBMSNAP_ERROR_INFO (  
  TableName Text 37, RowGuid ReplicationID,  
  Operation Text 6, Reason Long Integer);
```

Figure 76. Create Table for the Error Information Table

Error Messages Table (Microsoft Jet Specific)

This section contains programming interface information.

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 61 provides a brief description of the error messages table columns.

Table 61. Error Messages Table Columns

Column name	Description
Reason	The DPROPR error message number.
ReasonText	The DPROPR error message text.

Figure 77 shows the create table statement for the error messages table.

For Microsoft Jet

```
CREATE TABLE IBMSNAP_ERROR_MESSAGE (
    Reason Long Integer, ReasonText Memo);
```

Figure 77. Create Table Statement for the Error Messages Table

The following table is a conflict table for tracking synchronization conflicts and errors. This Microsoft Jet database control table mimics Microsoft's conflict tables.

Error Side Information Table (Microsoft Jet Specific)

This section contains programming interface information.

IBMSNAP_SIDE_INFO

This table contains the names of the conflict tables created by DataPropagator for Microsoft Jet.

Table 62 provides a brief description of the error side information table columns.

Table 62. Error Side Information Table Columns

Column name	Description
ConflictTableName	The conflict table name created by DataPropagator for Microsoft Jet.

Figure 78 shows the create table statement for the error side information table.

For Microsoft Jet

```
CREATE TABLE IBMSNAP_SIDE_INFO (
    ConflictTableName Text 37);
```

Figure 78. Create Table Statement for the Error Side Information Table

Key String Table (Microsoft Jet Specific)

This section contains programming interface information.

IBMSNAP_GUID_KEY

This table maps the Microsoft Jet table names and row identifiers to primary key values when the following actions occur:

- Deletes rows from Microsoft Jet database tables.
- Deletes are recorded in MSysTombstone with s_Generation, TableGUID and s_GUID (row) identifiers, but without primary key details.

Synchronization Generations Table

- The primary key values are needed to propagate Microsoft Jet database deletes to an RDBMS.

When DataPropagator for Microsoft Jet propagates deletes to another Microsoft Jet database, only the internal row identifier is sent. To propagate deletes outside of the Microsoft Jet environment, DataPropagator for Microsoft Jet needs to propagate 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 propagate a delete to an RDBMS, even after the row has been physically deleted from the row-replica table.

Table 63 provides a brief description of the key string table columns.

Table 63. 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 quotes. The column names are taken from the row-replica definition and can contain upper case letters, lower case letters or both. The constant values are taken from the rows themselves and the string values can contain upper case letters, lower case letters, numeric characters, or any combination of the three. Microsoft Jet database supports both ASCII and UNICODE, so the string constants can contain single or double byte characters. For example: COL1=(character) AND COL2=(character)

Figure 79 shows the create table and create index statements for the key string table.

For Microsoft Jet

```
CREATE TABLE IBMSNAP_GUID_KEY (  
    RowReplicaname Text 37 required,  
    s_GUID ReplicationID required,  
    key_string Memo required);  
  
CREATE UNIQUE INDEX IBMSNAP_GUID_KEYIX ON IBMSNAP_GUID_KEY  
(RowReplicaname, s_GUID);
```

Figure 79. Create Table and Create Index Statements for Key String Tables

Synchronization Generations Table (Microsoft Jet Specific)

This section contains programming interface information.

IBMSNAP_S_GENERATION

Synchronization Generations Table

This table is used to prevent cyclic updates from propagating 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 ASN.IBMSNAP_CRITSEC table, which results in a posting to ASN.IBMSNAP_UOW.APPLY_QUAL by the Capture program.

The s_GENERATION column will be 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'.

Multiple RDBMS-to-Jet generations can be posted before any Microsoft Jet database changes propagate to the RDBMS, 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. 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 need to be propagated to the RDBMS.

Table 64 provides a brief description of the synchronization generations table columns.

Table 64. Synchronization Generations Table

Column name	Description
Update_Type	A value that indicates whether a generation of changes are: 'L' Local to the Microsoft Jet database 'F' Foreign
JetSynctime	This is a dummy column, set to the time of a forced Microsoft Jet database synchronization.

Figure 80 shows the create table and create index statements for the synchronization generations table.

For Microsoft Jet

```
CREATE TABLE IBMSNAP_S_GENERATION (  
    Update_Type Text 1,  
    JetSynctime DateTime );  
  
CREATE UNIQUE INDEX IBMSNAP_S_GENERATION_IX ON IBMSNAP_S_GENERATION (JetSynctime);
```

Figure 80. Create Table and Create Index Statements for Synchronization Generations Tables

Synchronization Generations Table

Chapter 20. 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 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 358 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 Control Center 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 Control Center.

The IBM Replication tools are primarily relational database applications. The replication control tables are created using DDL issued by the 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 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 control table
- The Apply program trace file
- The Capture program trace control 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 21, “IBM Replication Messages” on page 361, 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 control table (ASN.IBMSNAP_APPLYTRAIL) and the Capture program trace table (ASN.IBMSNAP_TRACE)
- In the trace files for the Capture and Apply programs
- In the log files for the Capture and Apply programs

The Apply Program Problem Determination Facilities

SQL and Apply program error messages can be found in the ASN.IBMSNAP_APPLYTRAIL 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 Control Table (ASN.IBMSNAP_APPLYTRAIL)

There is an Apply trail control 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 control 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 19, “Table Structures” on page 295.

The Apply trail control 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 control table, ASN.IBMSNAP_APPLYTRAIL 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 control 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 ASN.IBMSNAP_APPLYTRAIL 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 ASN.IBMSNAP_APPLYTRAIL 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 NT with trace, issue the following command from the command line processor window:

```
\APPLY>asnapply myapply mydbnt2 trcfow > apply.trc
```

Where:

myapply	is the Apply qualifier.
mydbnt2	is the control server where the Apply program finds the control tables.
trcfow	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 *.LOG. For example, for an Apply program operating with a Apply qualifier of MYAPPLY, the log file name is MYAPPLY.LOG.

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.

Capture Program Trace Control Table (ASN.IBMSNAP_TRACE)

The trace control table is located in source server databases where the Capture program maintains change tables for replication sources. The trace control 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 19, “Table Structures” on page 295.

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 mysrddb x x trace > cap.trc
```

Where:

mysrddb	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 mysrddb`, the log file is named `mysrddb.ccp`. The Capture Program log file is located in the same directory from which the Capture program is started.

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 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 control 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 V5 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 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 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?
 - 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?

Chapter 21. IBM Replication Messages

The following is a list of messages issued by IBM Replication for the Capture and Apply programs and by migration. A brief explanation of the status is provided.

IBM Replication messages are intended to aid the user, as well as IBM Development and Support Center personnel.

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.

IBM Replication messages are prefixed as follows:

ASN0 The Capture program

ASN1 The Apply program

Besides using the information here, you can obtain explanations for messages by typing the following:

db2 "<message number>".

Note: The Replication Administration messages (DBA6001 — DBA6110) are listed in the UDB messages book.

Capture Program Messages

Note: For soft SQL errors, see the DB2 messages manual 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.

ASN0002E • ASN0004E

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 MVS 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 Version 5 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 13, "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 program read 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 for MVS, 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 for MVS invocation JCL.

For IBM DPROPR Capture of the Universal Database, 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 for MVS, 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 Return and Error Codes" manual for more information.

For VM/ESA, see the VM/ESA Programming Services for more information.

For the IBM DPROPR Capture of the 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 for MVS, or
- the Asynchronous Read Log API for IBM DPROPR Capture of the 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 for MVS, 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 for MVS invocation JCL.

User Response: Contact your IBM Service Representative.

ASN0007E • ASN0010E

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 for MVS, or
- the Asynchronous Read Log API for IBM DPROPR Capture of the 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 for MVS, 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 for MVS 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 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 for MVS, ensure that the REGION parameter has enough storage allocated to run your job. If necessary, contact your MVS 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 MVS 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 for MVS 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 19, "Table Structures" on page 295 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 • ASN0017E

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 19, "Table Structures" on page 295. 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 program internal control information. This message is for your information only.

User Response: No action is required.

ASN0024I •ASN0028I

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 MVS 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 IBM DPROPR Capture of the Universal Database)
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 parameter table can have only one row. The routine name is "<routine>"; the table name is "<table_name>".

Explanation: The tuning parameter table was not defined correctly or has been updated with invalid rows.

User Response: Refer to Chapter 19, "Table Structures" on page 295 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 more information about reinitializing the Capture program and Chapter 19, "Table Structures" on page 295 for information about the register table.

ASN0034E An incorrect value was supplied for column "<column>" of the Capture program tuning parameter 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 Control Center is not compatible with the version of the Capture program that you are running.

User Response: Refer to Chapter 19, “Table Structures” on page 295 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 MVS/ESA, VSE/ESA or VM/ESA, DB2 was terminated while Capture program was active and the user did not specify the NOTERM start up operand.

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 SQLEGENS API of DB2 Universal Database returned an error.

User Response: See the *DB2 for common servers API Reference* for information about the SQLEGENS 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 • ASN0055E

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 ASN.IBMSNAP_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 Control Center and contains unsupported SQL types (e.g. LONG or large object types).

User Response: Delete the replication source and use the 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 19, "Table Structures" on page 295 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 MVS/ESA, an Instrumentation Facility Information (IFI) error occurred while reading the log during warm start.

User Response: See Chapter 19, "Table Structures" on page 295 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 MVS 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 MVS 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 MVS, 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 •ASN0201E

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.

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 program encountered 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 •ASN1003E

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 critical section table could not be locked.

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 critical table, probably because of simultaneous activity on the table.

User Response: None required. The Apply program will attempt to get a lock on the critical section table during the next interval or event on the next cycle. If this condition persists, check data base activity for the critical section table.

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 Version 5 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 13, "Capture and Apply for UNIX Platforms" on page 197 for more information.

Refer to your database messages reference for SQL.

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 19, “Table Structures” on page 295.

User Response: Refer to Chapter 19, “Table Structures” on page 295 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 19, “Table Structures” on page 295, 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 19, “Table Structures” on page 295 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 19, “Table Structures” on page 295 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 •ASN1023S

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: Refer to “Defining Replication Subscriptions” to redefine the replication subscription.

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: Refer to “<Defining Replication Descriptions>” to redefine the replication description.

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 spill file.

User Response: Contact your IBM Service representative.

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 • ASN1040S

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 the "Defining Replication Sources" section of the "Administering Your Replication System" chapter 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 MVS Program Directory; make sure the plan name in the BIND JCL is ASNAP210.

ASN1040S The Apply program encountered an MVS error. The error code is "<error_code>"; the return code is "<return_code>".

Explanation: Execution of an MVS system operation failed.

User Response: Refer to your MVS 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 theApply 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 • ASN1055S

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 password file could not be opened. The error code is “<error_code>”.

Explanation: The user-created password file does not exist.

User Response: If you want to use the AUTHENTICATION=SERVER scheme, you must provide a password file as described in the Apply program section in the Capture and Apply chapter for your platform.

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.

ASN1065E •ASN1070E

ASN1065E The Apply program detected a referential integrity violation for UOWID “<ID_number>”. The error code is “<error_code>”.

Explanation: The Apply program has detected an error in the source data.

User Response: Correct the referential integrity problem and rerun the Apply program.

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 a 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 a 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 • ASN1203I

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 • ASN1244E

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.

Migration Messages

ASN2700E The server name “<server_name>” is invalid.

Explanation: The first character of the database server name is invalid.

User Response: Refer to the section on naming conventions for information on valid server names. Then provide the correct server name.

ASN2701E The database “<database_name>” or subsystem “<subsystem_name>” must be an 8-bytes-or-less alias name.

Explanation: The server name consists of more than eight characters.

User Response: Enter the correct alias name. It must consist of eight characters or less.

ASN2702I SQL CONNECT to “<server_type>” “<server_name>” succeeded.

Explanation: This message is for your information only.

User Response: No action is required.

ASN2703E SQL CONNECT to “<server_type>” “<server_name>” failed. Sqlstate is “<sqlstate>”, sqlcode is “<sqlcode>”, sqlerrm is “<sqlerrm>”, sqlerrp is “<sqlerrp>”.

Explanation: The migration tool cannot connect to the server.

User Response: Refer to the SQLSTATE values section of your database messages and codes manual.

ASN2704I Action “<action>” for “<table_name>” table starts.

Explanation: This message is for your information only.

User Response: No action is required.

ASN2705I Action “<action>” for “<view_name>” view starts.

Explanation: This message is for your information only.

User Response: No action is required.

ASN2706I Action “<action>” for “<table_name>” table succeeded.

Explanation: This message is for your information only.

User Response: No action is required.

ASN2707I Action “<action>” for “<view_name>” view succeeded.

Explanation: This message is for your information only.

User Response: No action is required.

ASN2708E • ASN2715I

ASN2708E Action “<action>” for “<table_name>” table failed.

Explanation: The migration fallback or the cleanup for the table failed.

User Response: Refer to the previous SQL statement execution to find the SQLSTATE value for the error message.

ASN2709E Action “<action>” for “<view_name>” view failed.

Explanation: The action for the view, MIGRATE, FALLBACK or CLEANUP, failed.

User Response: Refer to the previous SQL statement execution for the SQLSTATE value for the error message.

ASN2710E The BUILDDDB action was not done for subscriber “<subscriber_userid>” at “<server_type>” “<server_name>”.

Explanation: The BUILDDDB step was not executed for the subscriber.

User Response: Perform the BUILDDDB and the PREPARE step for the subscriber.

ASN2711E The BUILDDDB action was not done for “<server_type>” “<server_name>”.

Explanation: The BUILDDDB step was not executed for the server.

User Response: Perform the BUILDDDB and PREPARE steps for the server.

ASN2712E The requested version “<version>” is invalid.

Explanation: The control table version value entered is not valid.

User Response: Correct the value and reenter it or enter L to run the latest version.

ASN2713E The PREPARE action must be executed before the MIGRATE or FALLBACK action.

Explanation: The PREPARE step was not executed before the MIGRATE step.

User Response: Perform the PREPARE step before performing the MIGRATE step.

ASN2714I Action “<action>” for “<server_type>” “<server_name>” started or action “<action>” for “<subscriber>” “<userid>”.

Explanation: This message is for your information only.

User Response: No action is required.

ASN2715I Action “<action>” for “<server_type>” “<server_name>” succeeded or action “<action>” for “<subscriber>” “<userid>”.

Explanation: This message is for your information only.

User Response: No response is required.

ASN2716E Action "<action>" for "<server_type>" "<server_name>" failed or action "<action>" for "<subscriber>" "<userid>".

Explanation: The migration, fallback, or cleanup for the server failed.

User Response: Refer to the previous SQL statement execution for the SQLSTATE value and then refer to the listing of that value in the message information for your platform to determine the cause of the error.

ASN2717E Data server "<data_server_name>" has not been migrated yet.

Explanation: The data server with dependent subscriptions has not been migrated yet.

User Response: Migrate the data server.

ASN2718I Data server "<data_server_name>" has been migrated.

Explanation: This message is for your information only.

User Response: No action is required.

ASN2719E Action "<action>" for table "<table_name>" was not performed.

Explanation: The action for the table was not executed.

User Response: Repeat the action.

ASN2720E Action "<action>" for view "<view_name>" was not performed.

Explanation: The action for the view was not executed.

User Response: Repeat the action.

ASN2721I Current userid is "<user_id>" and "<server_type>" alias is "<alias_name>".

Explanation: This message is for your information only.

User Response: No action is required.

ASN2722I No registration information was collected at version "<version>".

Explanation: This message is for your information only.

User Response: No action is required.

ASN2723I No subscription information was collected at version "<version>".

Explanation: This message is for your information only.

User Response: No action is required.

ASN2724I Table "<table_name>" already exists.

Explanation: This message is for your information only.

User Response: No action is required.

ASN2725I •ASN2734E

ASN2725I View “<view_name>” already exists.

Explanation: This message is for your information only.

User Response: No action is required.

ASN2726I Table “<table_name>” does not exist.

Explanation: This message is for your information only.

User Response: No action is required.

ASN2727I View “<view_name>” does not exist.

Explanation: This message is for your information only.

User Response: No action is required.

ASN2728I More than one table “<table_name>” exists.

Explanation: This message is for your information only.

User Response: No action is required.

ASN2729E SQL message is “<sqlmsg>”, sqlcode is “<sqlcode>”, sqlerrm is “<sqlerrm>”, sqlerrp is “<sqlerrp>”.

Explanation: The SQL statement execution failed.

User Response: Refer to the SQL message and the SQLCODE values section of your database messages and codes manual.

ASN2730E No alias name was found for the server “<server_type>” “<server_name>”.

Explanation: No alias name was found for the “<server_type>” “<server_name>” was not provided at the BUILDDB step.

User Response: Provide the complete list of server alias names to the BUILDDB program and run the program again.

ASN2733I The global control information of version “<version>” has been prepared at “<timestamp>”.

Explanation: The SQL statement execution failed.

User Response: Refer to the SQLSTATE values section of your database messages and codes manual.

ASN2734E The file “<file_name>” cannot be opened.

Explanation: This file is either in error or does not exist.

User Response: Verify that the file has been installed correctly. If it has but it still cannot be opened, contact your IBM Service representative. If it has not been installed correctly, reinstall it.

ASN2735E The requested version “<version>” was not found.

Explanation: The information for the requested version cannot be found.

User Response: See the ASN.VERSION_CONTROL table for the available version number specified.

ASN2736E The action parameter was not provided, or was not specified before the keyword parameters.

Explanation: The value for one of the actions (MIGRATE, FALLBACK, or CLEANUP) was not entered before the keyword parameters.

User Response: Enter the action parameter before the keyword parameters.

ASN2737E Too many parameters were entered.

Explanation: The number of parameters entered exceeded the number allowed for the action.

User Response: Verify the necessary parameters and reenter the command.

ASN2738E This parameter is not a keyword parameter: “<parameter_name>”.

Explanation: The parameter entered is not a keyword parameter.

User Response: Enter ASNMIG to see the help for the command syntax. Then enter the correct keyword parameter.

ASN2739E Invalid action parameter was entered. Valid actions are BUILDDB, PREPARE, MIGRATE, FALLBACK, and CLEANUP.

Explanation: The action parameter entered is not a valid value.

User Response: Enter ASNMIG to see a listing of the valid action parameters. Then enter the correct action parameter.

ASN2740E Invalid parameter: “<keyword_parameter>”.

Explanation: The parameter segment entered was not one of the following: D=, S=, C=, T=, U=, or V=.

User Response: Enter the correct parameter.

ASN2741E Invalid combination of keyword parameters.

Explanation: The keyword parameters specified for the action were incorrectly entered.

User Response: See "Migrating DPROPR from Version 1 to Version 5" for the keywords needed for migration.

ASN2742E SQL SELECT failed. Sqlstate is “<sqlstate>”, sqlcode is “<sqlcode>”, sqlerrm is “<sqlerrm>”, sqlerrp is “<sqlerrp>”.

Explanation: The SQL statement execution failed.

User Response: Refer to the SQL message and the SQLCODE values section of your database messages and codes manual.

ASN2743I •ASN2751E

ASN2743I The column being added already exists.

Explanation: This message is for your information only.

User Response: No action is required.

ASN2744I The row being inserted already exists.

Explanation: This message is for your information only.

User Response: No action is required.

ASN2745I The row being updated already exists.

Explanation: This message is for your information only.

User Response: No action is required.

ASN2746I The V5 registrations defined after the last migration were lost.

Explanation: The Version 5 registrations defined after the last migration from Version 1 to Version 5 were lost because of fallback.

User Response: Register Version 5 after the next migration.

ASN2747I The action "<action>" ended successfully.

Explanation: This message is for your information only.

User Response: No action is required.

ASN2748E The action "<action>" ended unsuccessfully.

Explanation: An error occurred during the BUILDDB or PREPARE action.

User Response: Find the error in the action trace file and rerun the action.

ASN2749E The "<action>" partially succeeded because it could not connect to some servers.

Explanation: A wrong server name was provided.

User Response: Rerun the action using the correct server name.

ASN2750E No V1 control server name exists in "<userid>" ROUTING1 table. Fallback from V5 to V1 cannot be performed.

Explanation: The subscriber's routing table contains no rows so fallback cannot be performed.

User Response: Determine why the ROUTING1 table is empty and rebuild it.

ASN2751E SQL PREPARE failed. Sqlstate is "<sqlstate>", sqlcode is "<sqlcode>", sqlerrm is "<sqlerrm>", sqlerrp is "<sqlerrp>".

Explanation: The subscriber's routing table contains no rows so fallback cannot be performed.

User Response: Determine why the routing table is empty and rebuild it.

ASN2752E SQL OPEN CURSOR failed. Sqlstate is “<sqlstate>”, sqlcode is “<sqlcode>”, sqlerrm is “<sqlerrm>”, sqlerrp is “<sqlerrp>”.

Explanation: The SQL statement execution failed.

User Response: Refer to the SQL message and the SQLCODE values section of your database messages and codes manual.

ASN2753E SQL FETCH CURSOR failed. Sqlstate is “<sqlstate>”, sqlcode is “<sqlcode>”, sqlerrm is “<sqlerrm>”, sqlerrp is “<sqlerrp>”.

Explanation: The SQL statement execution failed.

User Response: Refer to the SQL message and the SQLCODE values section of your database messages and codes manual.

ASN2754I SQL CLOSE CURSOR failed. Sqlstate is “<sqlstate>”, sqlcode is “<sqlcode>”, sqlerrm is “<sqlerrm>”, sqlerrp is “<sqlerrp>”.

Explanation: The SQL statement execution failed.

User Response: Refer to the SQL message and the SQLCODE values section of your database messages and codes manual.

ASN2755I This PREPARE run is for validation only.

Explanation: This message is for your information only.

User Response: No action is required.

ASN2756I Action “<action>” trace file is “<action>”.TRC file.

Explanation: The program trace containing records of the action is in the “<action>”.TRC file BUILDDB, PREPARE, MIGRATE, FALLBACK, or CLEANUP).

User Response: Check the trace file to see if there were any errors.

ASN2757I Action “<action>” report file is “<action>”.RPT file

Explanation: This file is created during the migration (PREPARE, MIGRATE, FALLBACK, or CLEANUP).

User Response: Check the report file to see the summary of migration activity and resolve any errors before continuing with migration.

ASN2758E The migration tool cannot execute on this platform “<platform_name>”.

Explanation: Migration could not be performed because of incompatibility between the migration tool and the database it is running on. The problem is probably caused by an error in the installation of the program.

User Response: Refer to the installation section and reinstall the program. Then run the migration tool.

ASN2759E • ASN2765I

ASN2759E Invalid optional parameter “<parameter_value>”. Accepted parameters are 'V' for Validate and 'O' for Override.

Explanation: The only acceptable validation parameter is V.

User Response: Specify V for the validation parameter and reenter it.

ASN2760I Checking if auto-registration for subscriber “<userid>” exists.

Explanation: This message is for your information only.

User Response: No action is required.

ASN2761W Auto-registration has been detected at the copy server “<server_name>”. The ASN.IBMSNAP_CD_CNTL and pruning control tables need to be migrated first.

Explanation: The ASN.IBMSNAP_CD_CNTL and pruning control tables may not have been migrated at the copy server.

User Response: Migrate the ASN.IBMSNAP_CD_CNTL and pruning control tables at the copy server.

ASN2762I Please issue the following command: ASNMIG “<action>” D=“<migration db>” S=“<V1 data server>” V=“<version>”.

Explanation: This message follows ASN2761W. The ASN.IBMSNAP_CD_CNTL and pruning control tables may not have been migrated at the copy server.

User Response: Issue the command recommended in the message.

ASN2763I No auto-registration for subscriber “<userid>” has been detected.

Explanation: This message is for your information only.

User Response: No action is required.

ASN2764I The ASN.IBMSNAP_CD_CNTL and pruning control tables created from auto-registration task for subscriber “<subscriber>”’s subscription have been migrated.

Explanation: This message is for your information only.

User Response: No action is required.

ASN2765I The action “<action>” SQL file is “<action>”.SQL file.

Explanation: The SQL statement generated by the BUILDDB action is written to the “<action>”.SQL file (a DB2 statement file).

User Response: Use the DB2 Command Line Processing (CLP) facility to execute the SQL statement in the file.

ASN2766W Auto-registration has been detected at the target server “<server_name>”. The ASN.IBMSNAP_REGISTER and ASN.IBMSNAP_PRUNCNTL tables can be fallen back, but not necessary.

Explanation: The ASN.IBMSNAP_REGISTER and the ASN.IBMSNAP_PRUNCNTL tables may not have fallen back at the copy server.

User Response: Fall back the ASN.IBMSNAP_REGISTER and the pruning control tables at the copy server, if desired.

ASN2767E The list of servers must include at least one valid database alias preceding the wildcard character (*).

Explanation: Either all the server aliases entered were invalid or no alias was entered with the wildcard character (*). The wildcard character alone is not sufficient for the program to run.

User Response: Reenter the wildcard character with the proper server alias.

ASN2768W The wildcard character (*) was not specified, so ensure that you did include the complete list of servers for collection.

Explanation: No wildcard character (*) was entered during the BUILDDB action.

User Response: You can continue with the provided server list if this is your intention. If not, reenter the server list with the wildcard character.

ASN2769E SQL EXECUTE/EXECUTE IMMEDIATE failed. Sqlstate is “<sqlstate>”, sqlcode is “<sqlcode>”, sqlerrm is “<sqlerrm>”, sqlerrp is “<sqlerrp>”.

Explanation: The SQL statement execution failed.

User Response: Refer to the SQL message and the SQLCODE values section of your database messages and codes manual.

ASN2770E All Applys having subscriptions from data server “<server_name>” must fall back first.

Explanation: At least one Apply subscribed from this data server has not fallen back yet.

User Response: Perform the FALLBACK for all Applys before the Capture.

ASN2771I Subscriber “<userid>” does not have rows in the subscription tables.

Explanation: The IBM Replication V5 tables contain no information that would allow a fallback to DPROPR V1 for that subscriber.

User Response: Verify that the subscriber userid is correct. If it is, review the executed actions.

ASN2772E SQL COMMIT failed. Sqlstate is “<sqlstate>”, sqlcode is “<sqlcode>”, sqlerrm is “<sqlerrm>”, sqlerrp is “<sqlerrp>”.

Explanation: The SQL statement execution failed.

User Response: Refer to the SQL message and the SQLCODE values section of your database messages and codes manual.

ASN2773I •ASN2773I

ASN2773I To change to override option, reissue the command with the parameter: "O".

Explanation: The PREPARE action has been performed at the specified version.

User Response: To restart the program, specify the additional "O" to override the values in the DPROP V1 global control table in the migration database.

Chapter 22. Sample Invocation JCL

Link-Edit JCL for Capture (MVS for DB2 Version 3)

```

//*****
//* JOB NAME = ASNL2LK3                               */
//*                                                    */
//* DESCRIPTIVE NAME = LINK-EDIT JCL FOR THE IBM      */
//*                DATAPROPAGATOR RELATIONAL CAPTURE FOR */
//*                MVS FOR DB2 VERSION 3.1           */
//*                                                    */
//* STATUS = VERSION 05  RELEASE 01  MODIFICATION LEVEL 00 */
//*                                                    */
//* FUNCTION = CREATE THE DATAPROPAGATOR RELATIONAL  */
//*                CAPTURE FOR MVS LOAD MODULE       */
//*                                                    */
//*                LICENSED MATERIALS - PROPERTY OF IBM */
//*                5655-A23 (C) COPYRIGHT IBM CORP 1993, 1997 */
//*                ALL RIGHTS RESERVED.              */
//*                US GOVERNMENT USERS RESTRICTED RIGHTS - */
//*                USE, DUPLICATION OR DISCLOSURE RESTRICTED */
//*                BY GSA ADP SCHEDULE CONTRACT WITH IBM CORP. */
//*                                                    */
//* NOTES =                                           */
//* 1) REVIEW ALL STATEMENTS.                        */
//* 2) CHANGE THE HLQUAL FIELD(S) TO A VALID HIGH LEVEL */
//*    QUALIFIER TO COMPLY WITH YOUR SITE'S NAMING STANDARDS. */
//*    ASNL.V5R1M0 IS RECOMMENDED AS THE HIGH LEVEL QUALIFIER. */
//* 3) BE SURE TO REMOVE THE TEXT THAT BEGINS WITH <== AT THE */
//*    END OF SOME LINES                               */
//*****
//*
//ASNL2LK3 JOB USER=USERID,PASSWORD=PASSWORD <== SPECIFY JOB CARD PARM
//*****
//*                FOR EXECUTION OF CAPTURE FOR MVS LINK/EDIT PROCEDURE*/
//*****
//ASNL2LK3 EXEC PGM=IEWL,PARM='MAP,AMODE=31,RMODE=ANY'
//SYSLIB DD DISP=SHR,DSN=EDC.V2R1M0.SEDCBASE <== VERIFY C/370
// DD DISP=SHR,DSN=PLI.V2R3M0.SIBMBASE <== LIBRARY NAMES AND
// DD DISP=SHR,DSN=DB2.V3R1.SDSNLOAD <== DB2 LIBRARY NAME
```

```

/*      NETVIEW LIBRARIES (REMOVE COMMENT, IF NETVIEW IS INSTALLED)
/*      DD DSN=SYS1.CNMLINK,DISP=SHR
/*      DD DSN=SYS1.CNMINST,DISP=SHR
/*      DD DSN=SYS1.CNMSAMP,DISP=SHR
/*      DD DSN=SYS1.BNJMISC,DISP=SHR
/*      DD DSN=SYS1.BNJPNL1,DISP=SHR
/*      DD DSN=SYS1.BNJPNL2,DISP=SHR
//OBJ   DD DISP=SHR,DSN=HLQUAL.SASNLOBJ          <== VERIFY FILE NAME
//SYSLMOD DD DISP=(OLD,KEEP,KEEP),
//      DSN=HLQUAL.SASNLLNK                    <== VERIFY FILE NAME
//SYSPRINT DD SYSOUT=*
//SYSUT1 DD DSN=&&SYSUT1,UNIT=SYSDA,DISP=(NEW,DELETE),
//      SPACE=(32000,(30,30))
//SYSLIN DD DDNAME=SYSIN
//SYSIN  DD *
        INCLUDE SYSLIB(DSNALI)
        INCLUDE OBJ(ASNLTD31)
        INCLUDE OBJ(ASNLTM31)
        INCLUDE OBJ(ASNLTC31)
        INCLUDE OBJ(ASNL2WTO)
        INCLUDE OBJ(ASNL2MSG)
        SETCODE AC(1)
        NAME ASNLRP23(R)
        INCLUDE OBJ(ASNL2CPY)
        INCLUDE OBJ(ASNL2U00)
        NAME ASNL2U00(R)
/*
//

```

Link-Edit JCL for Capture (MVS for DB2 Version 4)

```

/*****/
/* JOB NAME = ASNL2LK4                               */
/*                                                    */
/* DESCRIPTIVE NAME = LINK-EDIT JCL FOR THE IBM      */
/*                   DATAPROPAGATOR RELATIONAL CAPTURE FOR */
/*                   MVS FOR DB2 VERSION 4.1         */
/*                                                    */
/* STATUS = VERSION 05  RELEASE 01  MODIFICATION LEVEL 00 */
/*                                                    */
/* FUNCTION = CREATE THE DATAPROPAGATOR RELATIONAL   */
/*                   CAPTURE FOR MVS LOAD MODULE     */
/*                                                    */

```



```

/**                                                                    */
/**      LICENSED MATERIALS - PROPERTY OF IBM                          */
/**      5655-A23 (C) COPYRIGHT IBM CORP 1993, 1997                    */
/**      ALL RIGHTS RESERVED.                                          */
/**      US GOVERNMENT USERS RESTRICTED RIGHTS -                       */
/**      USE, DUPLICATION OR DISCLOSURE RESTRICTED                    */
/**      BY GSA ADP SCHEDULE CONTRACT WITH IBM CORP.                   */
/**                                                                    */
/** NOTES =                                                            */
/** 1) REVIEW ALL STATEMENTS.                                          */
/** 2) CHANGE THE HLQUAL FIELD(S) TO A VALID HIGH LEVEL               */
/**     QUALIFIER TO COMPLY WITH YOUR SITE'S NAMING STANDARDS.        */
/**     ASNL.V5R1M0 IS RECOMMENDED AS THE HIGH LEVEL QUALIFIER.       */
/** 3) BE SURE TO REMOVE THE TEXT THAT BEGINS WITH <== AT THE        */
/**     END OF SOME LINES                                              */
/*******                                                              */
/**                                                                    */
/**ASNL2LK4 JOB USER=USERID,PASSWORD=PASSWORD <== SPECIFY JOB CARD PARM
/*******                                                              */
/**      FOR EXECUTION OF CAPTURE FOR MVS LINK/EDIT PROCEDURE*/
/*******                                                              */
/**ASNL2LK4 EXEC PGM=IEWL,PARM='MAP,AMODE=31,RMODE=ANY'
/**SYSLIB  DD DISP=SHR,DSN=EDC.V2R1M0.SEDCBASE <== VERIFY C/370
/**        DD DISP=SHR,DSN=PLI.V2R3M0.SIBMBASE <== LIBRARY NAMES AND
/**        DD DISP=SHR,DSN=DSN410.SDSNLOAD <== DB2 LIBRARY NAME
/**      NETVIEW LIBRARIES (REMOVE COMMENT, IF NETVIEW IS INSTALLED)
/**        DD DSN=SYS1.CNMLINK,DISP=SHR
/**        DD DSN=SYS1.CNMINST,DISP=SHR
/**        DD DSN=SYS1.CNMSAMP,DISP=SHR
/**        DD DSN=SYS1.BNJMISC,DISP=SHR
/**        DD DSN=SYS1.BNJPNL1,DISP=SHR
/**        DD DSN=SYS1.BNJPNL2,DISP=SHR
/**OBJ     DD DISP=SHR,DSN=HLQUAL.SASNLOBJ <== VERIFY FILE NAME
/**SYSLMOD DD DISP=(OLD,KEEP,KEEP),
/**        DSN=HLQUAL.SASNLLNK <== VERIFY FILE NAME

```

```

//SYSPRINT DD SYSOUT=*
//SYSUT1 DD DSN=&&SYSUT1,UNIT=SYSDA,DISP=(NEW,DELETE),
//          SPACE=(32000,(30,30))
//SYSLIN DD DDNAME=SYSIN
//SYSIN DD *
INCLUDE SYSLIB(DSNALI)
INCLUDE OBJ(ASNLTD41)
INCLUDE OBJ(ASNLTM41)
INCLUDE OBJ(ASNLTC41)
INCLUDE OBJ(ASNL2WTO)
INCLUDE OBJ(ASNL2MSG)
SETCODE AC(1)
NAME ASNLRP24(R)
INCLUDE OBJ(ASNL2CPY)
INCLUDE OBJ(ASNLTU00)
NAME ASNLTU00(R)
/*
//

```

Link-Edit JCL for Capture (MVS for DB2 Version 5)

```

/******//
/* JOB NAME = ASNL2LK5 */
/* */
/* DESCRIPTIVE NAME = LINK-EDIT JCL FOR THE IBM */
/*          DATAPROPAGATOR RELATIONAL CAPTURE FOR */
/*          MVS FOR DB2 VERSION 5.1 */
/* */
/* STATUS = VERSION 05  RELEASE 01  MODIFICATION LEVEL 00 */
/* */
/* FUNCTION = CREATE THE DATAPROPAGATOR RELATIONAL */
/*          CAPTURE FOR MVS LOAD MODULE */
/* */
/*          LICENSED MATERIALS - PROPERTY OF IBM */
/*          5655-A23 (C) COPYRIGHT IBM CORP 1993, 1997 */
/*          ALL RIGHTS RESERVED. */
/*          US GOVERNMENT USERS RESTRICTED RIGHTS - */
/*          USE, DUPLICATION OR DISCLOSURE RESTRICTED */
/*          BY GSA ADP SCHEDULE CONTRACT WITH IBM CORP. */
/* */

```

```

/* NOTES = */
/* 1) REVIEW ALL STATEMENTS. */
/* 2) CHANGE THE HLQUAL FIELD(S) TO A VALID HIGH LEVEL QUALIFIER TO COMPLY WITH YOUR SITE'S NAMING STANDARDS. */
/* ASNL.V5R1M0 IS RECOMMENDED AS THE HIGH LEVEL QUALIFIER. */
/* 3) BE SURE TO REMOVE THE TEXT THAT BEGINS WITH <== AT THE END OF SOME LINES */
/* ***** */
/*
//ASNL2LK5 JOB USER=USERID,PASSWORD=PASSWORD <== SPECIFY JOB CARD PARM
//*****
/* FOR EXECUTION OF CAPTURE FOR MVS LINK/EDIT PROCEDURE*/
//*****
//ASNL2LK5 EXEC PGM=IEWL,PARM='MAP,AMODE=31,RMODE=ANY'
//SYSLIB DD DISP=SHR,DSN=EDC.V2R1M0.SEDCBASE <== VERIFY C/370
// DD DISP=SHR,DSN=PLI.V2R3M0.SIBMBASE <== LIBRARY NAMES AND
// DD DISP=SHR,DSN=DSN510.SDSNLOAD <== DB2 LIBRARY NAME
/* NETVIEW LIBRARIES (REMOVE COMMENT, IF NETVIEW IS INSTALLED)
/* DD DSN=SYS1.CNMLINK,DISP=SHR
/* DD DSN=SYS1.CNMINST,DISP=SHR
/* DD DSN=SYS1.CNMSAMP,DISP=SHR
/* DD DSN=SYS1.BNJMISC,DISP=SHR
/* DD DSN=SYS1.BNJPNL1,DISP=SHR
/* DD DSN=SYS1.BNJPNL2,DISP=SHR
//OBJ DD DISP=SHR,DSN=HLQUAL.SASNLOBJ <== VERIFY FILE NAME
//SYSLMOD DD DISP=(OLD,KEEP,KEEP),
// DSN=HLQUAL.SASNLLNK <== VERIFY FILE NAME
//SYSPRINT DD SYSOUT=*
//SYSUT1 DD DSN=&&SYSUT1,UNIT=SYSDA,DISP=(NEW,DELETE),
// SPACE=(32000,(30,30))
//SYSLIN DD DDNAME=SYSIN
//SYSIN DD *
INCLUDE SYSLIB(DSNALI)
INCLUDE OBJ(ASNLTD51)
INCLUDE OBJ(ASNLTM51)
INCLUDE OBJ(ASNLTC51)
INCLUDE OBJ(ASNL2WTO)
INCLUDE OBJ(ASNL2MSG)
SETCODE AC(1)
NAME ASNLRP25(R)
INCLUDE OBJ(ASNL2CPY)
INCLUDE OBJ(ASNLTU00)
NAME ASNLTU00(R)
/*
//

```

Bind Package JCL for Capture (MVS for DB2 Version 3)

```

/*****/
/* JOB NAME = ASNL2BD3 */
/* */
/* DESCRIPTIVE NAME = BIND JCL FOR THE IBM */
/* DATAPROPAGATOR RELATIONAL CAPTURE FOR */
/* MVS FOR DB2 VERSION 3.1 */
/* */
/* STATUS = VERSION 05 RELEASE 01 MODIFICATION LEVEL 00 */
/* */
/* FUNCTION = BIND THE DATAPROPAGATOR RELATIONAL */
/* CAPTURE FOR MVS PLAN */
/* */
/* LICENSED MATERIALS - PROPERTY OF IBM */
/* 5655-A23 (C) COPYRIGHT IBM CORP 1993, 1997 */
/* ALL RIGHTS RESERVED. */
/* US GOVERNMENT USERS RESTRICTED RIGHTS - */
/* USE, DUPLICATION OR DISCLOSURE RESTRICTED */
/* BY GSA ADP SCHEDULE CONTRACT WITH IBM CORP. */
/* */
/* NOTES = */
/* 1) REVIEW ALL STATEMENTS. */
/* 2) CHANGE THE HLQUAL FIELD(S) TO A VALID HIGH LEVEL */
/* QUALIFIER TO COMPLY WITH YOUR SITE'S NAMING STANDARDS. */
/* ASNL.V5RIM0 IS RECOMMENDED AS THE HIGH LEVEL QUALIFIER. */
/* 3) BE SURE TO REMOVE THE TEXT THAT BEGINS WITH <== AT THE */
/* END OF SOME LINES */
/* */
/*****/
/* */
//ASNL2BD3 JOB USER=USERID,PASSWORD=PASSWORD <== SPECIFY JOB CARD PARM
/*****/
/* FOR EXECUTION OF CAPTURE FOR MVS BIND PROCEDURE */
/*****/
//ASNL2BD3 EXEC PGM=IKJEFT01
//SYSPRINT DD SYSOUT=*
//SYSPRINT DD SYSOUT=*
//SYSUDUMP DD SYSOUT=*
//STEPLIB DD DISP=SHR,DSN=DB2.V3R1.SDSNLOAD <== VERIFY DB2 LIB NAME
//DBRMLIB DD DISP=SHR,DSN=HLQUAL.SASNLDBM <== VERIFY FILE NAME
//SYSTSIN DD *

DSN S(DB31) <== REPLACE DB31 WITH YOUR SUBSYSTEM NAME
BIND PLAN(ASNLDP510) MEMBER(ASNLDM31) ACTION(REP) ACQUIRE(USE) -
RELEASE(COMMIT) ISOLATION(CS)
/*
//

```

Bind Package JCL for Capture (MVS for DB2 Version 4)

```

//*****
//* JOB NAME = ASNL2BD4 */
//* */
//* DESCRIPTIVE NAME = BIND JCL FOR THE IBM */
//* DATAPROPAGATOR RELATIONAL CAPTURE FOR */
//* MVS FOR DB2 VERSION 4.1 */
//* */
//* STATUS = VERSION 05 RELEASE 01 MODIFICATION LEVEL 00 */
//* */
//* FUNCTION = BIND THE DATAPROPAGATOR RELATIONAL */
//* CAPTURE FOR MVS PLAN */
//* */
//* LICENSED MATERIALS - PROPERTY OF IBM */
//* 5655-A23 (C) COPYRIGHT IBM CORP 1993, 1997 */
//* ALL RIGHTS RESERVED. */
//* US GOVERNMENT USERS RESTRICTED RIGHTS - */
//* USE, DUPLICATION OR DISCLOSURE RESTRICTED */
//* BY GSA ADP SCHEDULE CONTRACT WITH IBM CORP. */
//* */
//* NOTES = */
//* 1) REVIEW ALL STATEMENTS. */
//* 2) CHANGE THE HLQUAL FIELD(S) TO A VALID HIGH LEVEL */
//* QUALIFIER TO COMPLY WITH YOUR SITE'S NAMING STANDARDS. */
//* ASNL.V5R1M0 IS RECOMMENDED AS THE HIGH LEVEL QUALIFIER. */
//* 3) BE SURE TO REMOVE THE TEXT THAT BEGINS WITH <== AT THE */
//* END OF SOME LINES */
//* */
//*****
//ASNL2BD4 JOB USER=USERID,PASSWORD=PASSWORD <== SPECIFY JOB CARD PARM
//*****
//* FOR EXECUTION OF CAPTURE FOR MVS BIND PROCEDURE */
//*****
//ASNL2BD4 EXEC PGM=IKJEFT01
//SYSPRINT DD SYSOUT=*
//SYSTSPRT DD SYSOUT=*
//SYSUDUMP DD SYSOUT=*
//STEPLIB DD DISP=SHR,DSN=DSN410.SDSNLOAD <== VERIFY DB2 LIB NAME
//DBRMLIB DD DISP=SHR,DSN=HLQUAL.SASNLDBM <== VERIFY FILE NAME
//SYSTSIN DD *

DSN S(DB24) <== REPLACE DB24 WITH YOUR SUBSYSTEM NAME
BIND PLAN(ASNLP510) MEMBER(ASNLDM41) ACTION(REP) ACQUIRE(USE) -
RELEASE(COMMIT) ISOLATION(UR)
/*
//

```

Bind Package JCL for Capture (MVS for DB2 Version 5)

```

/*****/
/* JOB NAME = ASNL2BD5 */
/* */
/* DESCRIPTIVE NAME = BIND JCL FOR THE IBM */
/* DATAPROPAGATOR RELATIONAL CAPTURE FOR */
/* MVS FOR DB2 VERSION 5.1 */
/* */
/* STATUS = VERSION 05 RELEASE 01 MODIFICATION LEVEL 00 */
/* */
/* FUNCTION = BIND THE DATAPROPAGATOR RELATIONAL */
/* CAPTURE FOR MVS PLAN */
/* */
/* LICENSED MATERIALS - PROPERTY OF IBM */
/* 5655-A23 (C) COPYRIGHT IBM CORP 1993, 1997 */
/* ALL RIGHTS RESERVED. */
/* US GOVERNMENT USERS RESTRICTED RIGHTS - */
/* USE, DUPLICATION OR DISCLOSURE RESTRICTED */
/* BY GSA ADP SCHEDULE CONTRACT WITH IBM CORP. */
/* */
/* NOTES = */
/* 1) REVIEW ALL STATEMENTS. */
/* 2) CHANGE THE HLQUAL FIELD(S) TO A VALID HIGH LEVEL */
/* QUALIFIER TO COMPLY WITH YOUR SITE'S NAMING STANDARDS. */
/* ASNL.V5RIM0 IS RECOMMENDED AS THE HIGH LEVEL QUALIFIER. */
/* 3) BE SURE TO REMOVE THE TEXT THAT BEGINS WITH <== AT THE */
/* END OF SOME LINES */
/* */
/*****/
/*
//ASNL2BD5 JOB USER=USERID,PASSWORD=PASSWORD <== SPECIFY JOB CARD PARM
/*****/
/* FOR EXECUTION OF CAPTURE FOR MVS BIND PROCEDURE */
/*****/
//ASNL2BD5 EXEC PGM=IKJEFT01
//SYSPRINT DD SYSOUT=*
//SYSPRINT DD SYSOUT=*
//SYSUDUMP DD SYSOUT=*
//STEPLIB DD DISP=SHR,DSN=DSN510.SDSNLOAD <== VERIFY DB2 LIB NAME
//DBRMLIB DD DISP=SHR,DSN=HLQUAL.SASNLDBM <== VERIFY FILE NAME
//SYSTSIN DD *

DSN S(D51A) <== REPLACE D51A WITH YOUR SUBSYSTEM NAME
BIND PLAN(ASNLDP510) MEMBER(ASNLDM51) ACTION(REP) ACQUIRE(USE) -
RELEASE(COMMIT) ISOLATION(UR)
/*
//

```

Invocation JCL for Capture (MVS for DB2 Version 3)

```

//*****
//* JOB NAME = ASNL2RN3 */
//* */
//* DESCRIPTIVE NAME = INVOCATION JCL FOR THE IBM */
//* DATAPROPAGATOR RELATIONAL CAPTURE FOR */
//* MVS FOR DB2 VERSION 3.1 */
//* */
//* STATUS = VERSION 05 RELEASE 01 MODIFICATION LEVEL 00 */
//* */
//* FUNCTION = INVOKE THE DATAPROPAGATOR RELATIONAL */
//* CAPTURE FOR MVS PROGRAM */
//* */
//* LICENSED MATERIALS - PROPERTY OF IBM */
//* 5655-A23 (C) COPYRIGHT IBM CORP 1993, 1997 */
//* ALL RIGHTS RESERVED. */
//* US GOVERNMENT USERS RESTRICTED RIGHTS - */
//* USE, DUPLICATION OR DISCLOSURE RESTRICTED */
//* BY GSA ADP SCHEDULE CONTRACT WITH IBM CORP. */
//* */
//* NOTES = */
//* 1) REVIEW ALL STATEMENTS. */
//* 2) CHANGE THE HLQUAL FIELD(S) TO A VALID HIGH LEVEL */
//* QUALIFIER TO COMPLY WITH YOUR SITE'S NAMING STANDARDS. */
//* ASNL.V5R1M0 IS RECOMMENDED AS THE HIGH LEVEL QUALIFIER. */
//* 3) IF YOU ARE RUNNING UNDER LE/370 ENVIRONMENT, ADD */
//* REGION=3M IN THE EXEC STATEMENT. */
//* 4) REPLACE DB31 WITH YOUR SUBSYSTEM NAME */
//* 5) BE SURE TO REMOVE THE TEXT THAT BEGINS WITH <== AT THE */
//* END OF SOME LINES */
//* */
//*****
//ASNL2RN3 JOB USER=USERID,PASSWORD=PASSWORD <== SPECIFY JOB CARD PARM
//*****
// FOR EXECUTION OF THE CAPTURE FOR MVS PROGRAM */
//*****
//ASNL2RN3 EXEC PGM=ASNLRP23,PARM='DB31 COLD' <== VERIFY DB2 SUBSYSTEM
//STEPLIB DD DISP=SHR,DSN=HLQUAL.SASNLLNK <== VERIFY LIB NAME
// DD DISP=SHR,DSN=DB2.V3R1.SDSNLOAD <== VERIFY DB2 AND
// DD DISP=SHR,DSN=EDC.V2R1M0.SEDCLINK <== C/370 LIBRARY
// DD DISP=SHR,DSN=PLI.V2R3M0.SIBMLINK <== NAMES
//MSGSDD DISP=SHR,DSN=HLQUAL.MSGS <== VERIFY FILE NAME
//CEEDUMP DD SYSOUT=*
//SYSTEM DD SYSOUT=*
//SYSUDUMP DD SYSOUT=*
//SYSSPRINT DD SYSOUT=*
//

```

Invocation JCL for Capture (MVS for DB2 Version 4)

```

//*****
//* JOB NAME = ASNL2RN4
//*
//* DESCRIPTIVE NAME = INVOCATION JCL FOR THE IBM
//* DATAPROPAGATOR RELATIONAL CAPTURE FOR
//* MVS FOR DB2 VERSION 4.1
//*
//* STATUS = VERSION 05 RELEASE 01 MODIFICATION LEVEL 00
//*
//* FUNCTION = INVOKE THE DATAPROPAGATOR RELATIONAL
//* CAPTURE FOR MVS PROGRAM
//*
//* LICENSED MATERIALS - PROPERTY OF IBM
//* 5655-A23 (C) COPYRIGHT IBM CORP 1993, 1997
//* ALL RIGHTS RESERVED.
//* US GOVERNMENT USERS RESTRICTED RIGHTS -
//* USE, DUPLICATION OR DISCLOSURE RESTRICTED
//* BY GSA ADP SCHEDULE CONTRACT WITH IBM CORP.
//*
//* NOTES =
//* 1) REVIEW ALL STATEMENTS.
//* 2) CHANGE THE HLQUAL FIELD(S) TO A VALID HIGH LEVEL
//* QUALIFIER TO COMPLY WITH YOUR SITE'S NAMING STANDARDS.
//* ASNL.V5R1M0 IS RECOMMENDED AS THE HIGH LEVEL QUALIFIER.
//* 3) IF YOU ARE RUNNING UNDER LE/370 ENVIRONMENT, ADD
//* REGION=3M IN THE EXEC STATEMENT.
//* 4) REPLACE DB24 WITH YOUR SUBSYSTEM NAME
//* 5) BE SURE TO REMOVE THE TEXT THAT BEGINS WITH <== AT THE
//* END OF SOME LINES
//*****
//ASNL2RN4 JOB USER=USERID,PASSWORD=PASSWORD <== SPECIFY JOB CARD PARM
//*****
//* FOR EXECUTION OF THE CAPTURE FOR MVS PROGRAM
//*****
//ASNL2RN4 EXEC PGM=ASNLRP24,PARM='DB24 COLD' <== VERIFY DB2 SUBSYSTEM
//STEPLIB DD DISP=SHR,DSN=HLQUAL.SASNLLNK <== VERIFY LIB NAME
// DD DISP=SHR,DSN=DSN410.SDSNLOAD <== VERIFY DB2 AND
// DD DISP=SHR,DSN=EDC.V2R1M0.SEDCLINK <== C/370 LIBRARY
// DD DISP=SHR,DSN=PLI.V2R3M0.SIBMLINK <== NAMES
//MSGS DD DISP=SHR,DSN=HLQUAL.MSGS <== VERIFY FILE NAME
//CEEDUMP DD SYSOUT=*
//SYSTEM DD SYSOUT=*
//SYSUDUMP DD SYSOUT=*
//SYSPRINT DD SYSOUT=*
//

```


Invocation JCL for Capture (MVS for DB2 Version 5)

```
//*****  
//* JOB NAME = ASNL2RN5 */  
//* */  
//* DESCRIPTIVE NAME = INVOCATION JCL FOR THE IBM */  
//* DATAPROPAGATOR RELATIONAL CAPTURE FOR */  
//* MVS FOR DB2 VERSION 5.1 */  
//* */  
//* STATUS = VERSION 05 RELEASE 01 MODIFICATION LEVEL 00 */  
//* */  
//* FUNCTION = INVOKE THE DATAPROPAGATOR RELATIONAL */  
//* CAPTURE FOR MVS PROGRAM */  
  
//* */  
//* LICENSED MATERIALS - PROPERTY OF IBM */  
//* 5655-A23 (C) COPYRIGHT IBM CORP 1993, 1997 */  
//* ALL RIGHTS RESERVED. */  
//* US GOVERNMENT USERS RESTRICTED RIGHTS - */  
//* USE, DUPLICATION OR DISCLOSURE RESTRICTED */  
//* BY GSA ADP SCHEDULE CONTRACT WITH IBM CORP. */  
//* */  
//* NOTES = */  
//* 1) REVIEW ALL STATEMENTS. */  
//* 2) CHANGE THE HLQUAL FIELD(S) TO A VALID HIGH LEVEL */  
//* QUALIFIER TO COMPLY WITH YOUR SITE'S NAMING STANDARDS. */  
//* ASNL.V5R1M0 IS RECOMMENDED AS THE HIGH LEVEL QUALIFIER. */  
//* 3) IF YOU ARE RUNNING UNDER LE/370 ENVIRONMENT, ADD */  
//* REGION=3M IN THE EXEC STATEMENT. */  
//* 4) REPLACE D51A WITH YOUR SUBSYSTEM NAME */  
//* 5) BE SURE TO REMOVE THE TEXT THAT BEGINS WITH <== AT THE */  
//* END OF SOME LINES */  
//* */  
//*****  
//*  
//ASNL2RN5 JOB USER=USERID,PASSWORD=PASSWORD <== SPECIFY JOB CARD PARM  
//*****  
//* FOR EXECUTION OF THE CAPTURE FOR MVS PROGRAM */  
//*****  
//ASNL2RN5 EXEC PGM=ASNLRP25,PARM='D51A COLD' <== VERIFY DB2 SUBSYSTEM  
//STEPLIB DD DISP=SHR,DSN=HLQUAL.SASNLLNK <== VERIFY LIB NAME  
// DD DISP=SHR,DSN=DSN510.SDSNLOAD <== VERIFY DB2 AND  
// DD DISP=SHR,DSN=EDC.V2R1M0.SEDCLINK <== C/370 LIBRARY  
// DD DISP=SHR,DSN=PLI.V2R3M0.SIBMLINK <== NAMES  
//MSGSDD DD DISP=SHR,DSN=HLQUAL.MSGS <== VERIFY FILE NAME  
//CEEDUMP DD SYSOUT=*  
//SYSTEM DD SYSOUT=*  
//SYSUDUMP DD SYSOUT=*  
//SYSPRINT DD SYSOUT=*  
//
```

Link-Edit JCL for Apply (MVS for DB2 Version 3)

```

//*****
//* JOB NAME = ASNA2LK3
//*
//* DESCRIPTIVE NAME = LINK-EDIT JCL FOR THE IBM
//* DATAPROPAGATOR RELATIONAL APPLY FOR MVS
//* FOR DB2 V3
//*
//* STATUS = VERSION 05 RELEASE 01 MODIFICATION LEVEL 00
//*
//* FUNCTION = CREATE THE DATAPROPAGATOR RELATIONAL
//* APPLY FOR MVS LOAD MODULE
//* DB2 V3
//*
//* LICENSED MATERIALS - PROPERTY OF IBM
//* 5655-A22 (C) COPYRIGHT IBM CORP 1993, 1997
//* ALL RIGHTS RESERVED.
//* US GOVERNMENT USERS RESTRICTED RIGHTS -
//* USE, DUPLICATION OR DISCLOSURE RESTRICTED
//* BY GSA ADP SCHEDULE CONTRACT WITH IBM CORP.
//*
//* NOTES =
//* 1) REVIEW ALL STATEMENTS.
//* 2) CHANGE THE HLQUAL FIELD(S) TO A VALID HIGH LEVEL
//* QUALIFIER TO COMPLY WITH YOUR SITE'S NAMING STANDARDS.
//* ASNA.V5R1M0 IS RECOMMENDED AS THE HIGH LEVEL QUALIFIER.
//* 3) DELETE THE TEXT THAT BEGINS WITH <== AT THE
//* END OF SOME LINES
//*
//*****
//ASNA2LK3 JOB USER=USERID,PASSWORD=PASSWORD <== SPECIFY JOB CARD PARM
//*****
//* FOR EXECUTION OF APPLY FOR MVS LINK/EDIT PROCEDURE
//*****
//ASNA2LK3 EXEC PGM=IEWL,PARM='MAP,AMODE=31,RMODE=ANY'
//SYSPRINT DD SYSOUT=*
//SYSLIB DD DISP=SHR,DSN=EDC.V2R1M0.SEDCBASE <== VERIFY C/370
// DD DISP=SHR,DSN=PLI.V2R3M0.SIBMBASE <== LIBRARY NAMES AND
// DD DISP=SHR,DSN=DB2.V3R1.SDSNLOAD <== DB2 LIBRARY NAME
//OBJ DD DISP=SHR,DSN=HLQUAL.SASNAOBJ <== VERIFY FILE NAME
//SYSLMOD DD DISP=SHR,DSN=HLQUAL.SASNALNK <== VERIFY FILE NAME
//SYSUT1 DD DSN=&&SYSUT1,UNIT=SYSDA,
// SPACE=(32000,(30,30))
//SYSLIN DD DDNAME=SYSIN
//SYSIN DD *
```

```

INCLUDE SYSLIB(DSNALI)
INCLUDE OBJ(ASNTS000)
INCLUDE OBJ(ASNT9000)
INCLUDE OBJ(ASNTM000)
INCLUDE OBJ(ASNT3000)
INCLUDE OBJ(ASNTB000)
INCLUDE OBJ(ASNTA000)
INCLUDE OBJ(ASNTF000)
INCLUDE OBJ(ASNTC000)
INCLUDE OBJ(ASNTI000)
INCLUDE OBJ(ASNTW000)
INCLUDE OBJ(ASNTX000)
SETCODE AC(1)
NAME ASNAPV23(R)
INCLUDE OBJ(ASNA2CPY)
INCLUDE OBJ(ASNTU000)
NAME ASNTU000(R)
/*
//

```

Link-Edit JCL for Apply (MVS for DB2 Version 4)

```

//*****/
/* JOB NAME = ASNA2LK4 */
/* */ */
/* DESCRIPTIVE NAME = LINK-EDIT JCL FOR THE IBM */
/* DATAPROPAGATOR RELATIONAL APPLY FOR MVS */
/* FOR DB2 V4 */ */
/* */ */
/* STATUS = VERSION 05 RELEASE 01 MODIFICATION LEVEL 00 */
/* */ */
/* FUNCTION = CREATE THE DATAPROPAGATOR RELATIONAL */
/* APPLY FOR MVS LOAD MODULE */
/* DB2 V4 */ */
/* */ */
/* LICENSED MATERIALS - PROPERTY OF IBM */
/* 5655-A22 (C) COPYRIGHT IBM CORP 1993, 1997 */
/* ALL RIGHTS RESERVED. */
/* US GOVERNMENT USERS RESTRICTED RIGHTS - */
/* USE, DUPLICATION OR DISCLOSURE RESTRICTED */
/* BY GSA ADP SCHEDULE CONTRACT WITH IBM CORP. */
/* */ */
/* */ */

```

```

/* NOTES = */
/* 1) REVIEW ALL STATEMENTS. */
/* 2) CHANGE THE HLQUAL FIELD(S) TO A VALID HIGH LEVEL QUALIFIER TO COMPLY WITH YOUR SITE'S NAMING STANDARDS. */
/* ASNA.V5R1M0 IS RECOMMENDED AS THE HIGH LEVEL QUALIFIER. */
/* 3) DELETE THE TEXT THAT BEGINS WITH <== AT THE END OF SOME LINES */
/* */
/******
/*
//ASNA2LK4 JOB USER=USERID,PASSWORD=PASSWORD <== SPECIFY JOB CARD PARM
/******
/* FOR EXECUTION OF APPLY FOR MVS LINK/EDIT PROCEDURE */
/******
//ASNA2LK4 EXEC PGM=IEWL,PARM='MAP,AMODE=31,RMODE=ANY'
//SYSPRINT DD SYSOUT=*
//SYSLIB DD DSN=EDC.V2R1M0.SEDCBASE,DISP=SHR <== VERIFY C/370
// DD DSN=PLI.V2R3M0.SIBMBASE,DISP=SHR <== LIBRARY NAMES AND
// DD DSN=DSN410.SDSNLOAD,DISP=SHR <== DB2 LIBRARY NAME
//OBJ DD DSN=HLQUAL.SASNAOBJ,DISP=SHR <== VERIFY FILE NAME
//SYSLMOD DD DSN=HLQUAL.SASNALNK,DISP=SHR <== VERIFY FILE NAME
//SYSUT1 DD DSN=&&SYSUT1,UNIT=SYSDA,
// SPACE=(32000,(30,30))
//SYSLIN DD DDNAME=SYSIN
//SYSIN DD *
INCLUDE SYSLIB(DSNALI)
INCLUDE OBJ(ASNTS000)
INCLUDE OBJ(ASNT9000)
INCLUDE OBJ(ASNT4000)
INCLUDE OBJ(ASNTP000)
INCLUDE OBJ(ASNTB000)
INCLUDE OBJ(ASNTA000)
INCLUDE OBJ(ASNTF000)
INCLUDE OBJ(ASNTC000)
INCLUDE OBJ(ASNTI000)
INCLUDE OBJ(ASNTW000)
INCLUDE OBJ(ASNTX000)
SETCODE AC(1)
NAME ASNAPV24(R)
INCLUDE OBJ(ASNA2CPY)
INCLUDE OBJ(ASNTU000)
NAME ASNTU000(R)
/*
//

```

Link-Edit JCL for Apply (MVS for DB2 Version 5)

```

//*****
//* JOB NAME = ASNA2LK5 */
//* */
//* DESCRIPTIVE NAME = LINK-EDIT JCL FOR THE IBM */
//* DATAPROPAGATOR RELATIONAL APPLY FOR MVS */
//* FOR DB2 V5 */
//* */
//* STATUS = VERSION 05 RELEASE 01 MODIFICATION LEVEL 00 */
//* */
//* FUNCTION = CREATE THE DATAPROPAGATOR RELATIONAL */
//* APPLY FOR MVS LOAD MODULE */
//* DB2 V5 */
//* */
//* LICENSED MATERIALS - PROPERTY OF IBM */
//* 5655-A22 (C) COPYRIGHT IBM CORP 1993, 1997 */
//* ALL RIGHTS RESERVED. */
//* US GOVERNMENT USERS RESTRICTED RIGHTS - */
//* USE, DUPLICATION OR DISCLOSURE RESTRICTED */
//* BY GSA ADP SCHEDULE CONTRACT WITH IBM CORP. */
//* */
//* NOTES = */
//* 1) REVIEW ALL STATEMENTS. */
//* 2) CHANGE THE HLQUAL FIELD(S) TO A VALID HIGH LEVEL */
//* QUALIFIER TO COMPLY WITH YOUR SITE'S NAMING STANDARDS. */
//* ASNA.V5R1M0 IS RECOMMENDED AS THE HIGH LEVEL QUALIFIER. */
//* 3) DELETE THE TEXT THAT BEGINS WITH <== AT THE */
//* END OF SOME LINES */
//* */
//*****
//ASNA2LK5 JOB USER=USERID,PASSWORD=PASSWORD <== SPECIFY JOB CARD PARM
//*****
//* FOR EXECUTION OF APPLY FOR MVS LINK/EDIT PROCEDURE */
//*****
//ASNA2LK5 EXEC PGM=IEWL,PARM='MAP,AMODE=31,RMODE=ANY'
//SYSPRINT DD SYSOUT=*
//SYSLIB DD DSN=EDC.V2R1M0.SEDCBASE,DISP=SHR <== VERIFY C/370
// DD DSN=PLI.V2R3M0.SIBMBASE,DISP=SHR <== LIBRARY NAMES AND
// DD DSN=DSN510.SDSNLOAD,DISP=SHR <== DB2 LIBRARY NAME
//OBJ DD DSN=HLQUAL.SASNAOBJ,DISP=SHR <== VERIFY FILE NAME

```

```

//SYSLMOD DD DSN=HLQUAL.SASNALNK,DISP=SHR      <== VERIFY FILE NAME
//SYSUT1  DD DSN=&&SYSUT1,UNIT=SYSDA,
//          SPACE=(32000,(30,30))
//SYSLIN  DD DDNAME=SYSIN
//SYSIN   DD *
          INCLUDE SYSLIB(DSNALI)
          INCLUDE OBJ(ASNTS000)
          INCLUDE OBJ(ASNT9000)
          INCLUDE OBJ(ASNT4000)
          INCLUDE OBJ(ASNTP000)
          INCLUDE OBJ(ASNTB000)
          INCLUDE OBJ(ASNTA000)
          INCLUDE OBJ(ASNTF000)
          INCLUDE OBJ(ASNTC000)
          INCLUDE OBJ(ASNTI000)
          INCLUDE OBJ(ASNTW000)
          INCLUDE OBJ(ASNTX000)
          SETCODE AC(1)
          NAME ASNAPV25(R)
          INCLUDE OBJ(ASNA2CPY)
          INCLUDE OBJ(ASNTU000)
          NAME ASNTU000(R)
/*
//

```

Bind Package JCL for Apply (MVS for DB2 Version 3)

```

//*****/
/* JOB NAME = ASNA2BD3 */
/* */ */
/* DESCRIPTIVE NAME = BIND PACKAGE JCL FOR THE IBM */
/*          DATAPROPAGATOR RELATIONAL APPLY FOR MVS */
/*          FOR DB2 V3 */ */
/* */ */
/* STATUS = VERSION 05  RELEASE 01  MODIFICATION LEVEL 00 */
/* */ */
/* FUNCTION = BIND THE DATAPROPAGATOR RELATIONAL */
/*          APPLY FOR MVS PACKAGE FOR DB2 V3 */
/* */ */

```

```

/*      LICENSED MATERIALS - PROPERTY OF IBM      */
/*      5655-A22 (C) COPYRIGHT IBM CORP 1993, 1997      */
/*      ALL RIGHTS RESERVED.      */
/*      US GOVERNMENT USERS RESTRICTED RIGHTS -      */
/*      USE, DUPLICATION OR DISCLOSURE RESTRICTED      */
/*      BY GSA ADP SCHEDULE CONTRACT WITH IBM CORP.      */
/*      */
/* NOTES =      */
/* 1) REVIEW ALL STATEMENTS.      */
/* 2) CHANGE THE HLQUAL FIELD(S) TO A VALID HIGH LEVEL      */
/*    QUALIFIER TO COMPLY WITH YOUR SITE'S NAMING STANDARDS.      */
/*    ASNA.V5R1M0 IS RECOMMENDED AS THE HIGH LEVEL QUALIFIER.      */
/* 3) DELETE THE TEXT THAT BEGINS WITH <== AT THE      */
/*    END OF SOME LINES      */
/*      */
/******
/*
//ASNA2BD3 JOB USER=USERID,PASSWORD=PASSWORD <== SPECIFY JOB CARD PARM
//*****
/*      FOR EXECUTION OF APPLY FOR MVS BIND PROCEDURE      */
//*****
//ASNA2BD3 EXEC PGM=IKJEFT01
//SYSPRINT DD  SYSOUT=*
//SYSPRINT DD  SYSOUT=*
//SYSUDUMP DD  SYSOUT=*
//STEPLIB DD  DISP=SHR,DSN=DB2.V3R1.SDSNLOAD
//DBRMLIB DD  DISP=SHR,DSN=HLQUAL.SASNADBM      <== VERIFY FILE NAME
//SYSTSIN DD  *
      DSN S(DB31)          <== REPLACE DB31 WITH YOUR SUBSYSTEM NAME
                          <== VERIFY DB2 LOCATION NAME
                          <== AND COLLECTION-ID;
                          <== INCLUDE THE 7 BIND PACKAGE SUBCOMMANDS
                          <== FOR EACH DB2 TO BE ACCESSED BY
                          <== THE APPLY PROGRAM

      BIND PACKAGE(LOCATION-NAME1.COL1) -
      MEMBER(ASNDI000) -
      ACTION(REP) -
      RELEASE(COMMIT) ISOLATION(CS)

```

```
|
|      BIND PACKAGE(LOCATION-NAME1.COL1) -
|      MEMBER(ASNDM000) -
|      ACTION(REP) -
|      RELEASE(COMMIT) ISOLATION(CS)
```

```
|
|      BIND PACKAGE(LOCATION-NAME1.COL1) -
|      MEMBER(ASND3000) -
|      ACTION(REP) -
|      RELEASE(COMMIT) ISOLATION(CS)
```

```
|
|      BIND PACKAGE(LOCATION-NAME1.COL1) -
|      MEMBER(ASNDB000) -
|      ACTION(REP) -
|      RELEASE(COMMIT) ISOLATION(CS)
```

```
|
|      BIND PACKAGE(LOCATION-NAME1.COL1) -
|      MEMBER(ASNDF000) -
|      ACTION(REP) -
|      RELEASE(COMMIT) ISOLATION(CS)
```

```
|
|      BIND PACKAGE(LOCATION-NAME1.COL1) -
|      MEMBER(ASNDA000) -
|      ACTION(REP) -
|      RELEASE(COMMIT) ISOLATION(CS)
```

```
|
|      BIND PACKAGE(LOCATION-NAME1.COL1) -
|      MEMBER(ASNDC000) -
|      ACTION(REP) -
|      RELEASE(COMMIT) ISOLATION(CS)
```

```
|
|      BIND PACKAGE(LOCATION-NAME2.COL1) -
|      MEMBER(ASNDI000) -
|      ACTION(REP) -
|      RELEASE(COMMIT) ISOLATION(CS)
```

```
|
|      .
|      .
|      .
```

```
|
|      BIND PACKAGE(LOCATION-NAME2.COL1) -
|      MEMBER(ASNDC000) -
|      ACTION(REP) -
|      RELEASE(COMMIT) ISOLATION(CS)
```

```
|
|      BIND PLAN(ASNAP510) -
|      PKLIST(LOCATION-NAME1.COL1.*,LOCATION-NAME2.COL1.*) -
|      ACTION(REP) ACQUIRE(USE) -
|      RELEASE(COMMIT)
```

```
|
|      /*
|      //
```

Bind Package JCL for Apply (MVS for DB2 Version 4)

```
//*****  
//* JOB NAME = ASNA2BD4 */  
//* */  
//* DESCRIPTIVE NAME = BIND PACKAGE JCL FOR THE IBM */  
//* DATAPROPAGATOR RELATIONAL APPLY FOR MVS */  
//* FOR DB2 V4 */  
//* */  
//* STATUS = VERSION 05 RELEASE 01 MODIFICATION LEVEL 00 */  
//* */  
//* FUNCTION = BIND THE DATAPROPAGATOR RELATIONAL */  
//* APPLY FOR MVS PACKAGE FOR DB2 V4 */  
//* */  
//* LICENSED MATERIALS - PROPERTY OF IBM */  
//* 5655-A22 (C) COPYRIGHT IBM CORP 1993, 1997 */  
//* ALL RIGHTS RESERVED. */  
//* US GOVERNMENT USERS RESTRICTED RIGHTS - */  
//* USE, DUPLICATION OR DISCLOSURE RESTRICTED */  
//* BY GSA ADP SCHEDULE CONTRACT WITH IBM CORP. */  
//* */  
//* NOTES = */  
//* 1) REVIEW ALL STATEMENTS. */  
//* 2) CHANGE THE HLQUAL FIELD(S) TO A VALID HIGH LEVEL */  
//* QUALIFIER TO COMPLY WITH YOUR SITE'S NAMING STANDARDS. */  
//* ASNA.V5R1M0 IS RECOMMENDED AS THE HIGH LEVEL QUALIFIER. */  
//* 3) DELETE THE TEXT THAT BEGINS WITH <== AT THE */  
//* END OF SOME LINES */  
//* */  
//*****  
//*  
//ASNA2BD4 JOB USER=USERID,PASSWORD=PASSWORD <== SPECIFY JOB CARD PARM  
//*****  
//* FOR EXECUTION OF APPLY FOR MVS BIND PROCEDURE */  
//*****  
//ASNA2BD4 EXEC PGM=IKJEFT01  
//SYSPRINT DD SYSOUT=*  
//SYSPRINT DD SYSOUT=*  
//SYSUDUMP DD SYSOUT=*  
//STEPLIB DD DISP=SHR,DSN=DSN410.SDSNLOAD  
//DBRMLIB DD DISP=SHR,DSN=HLQUAL.SASNADBMS <== VERIFY FILE NAME  
//SYSTSIN DD *
```

```
DSN S(DB24)          <== REPLACE DB24 WITH YOUR SUBSYSTEM NAME
                    <== VERIFY DB2 LOCATION NAME
                    <== AND COLLECTION-ID;
                    <== INCLUDE THE 7 BIND PACKAGE SUBCOMMANDS
                    <== FOR EACH DB2 TO BE ACCESSED BY
                    <== THE APPLY PROGRAM
```

```
BIND PACKAGE(LOCATION-NAME1.COL1) -
MEMBER(ASNDI000) -
ACTION(REP) -
RELEASE(COMMIT) ISOLATION(UR)
```

```
BIND PACKAGE(LOCATION-NAME1.COL1) -
MEMBER(ASND4000) -
ACTION(REP) -
RELEASE(COMMIT) ISOLATION(UR)
```

```
BIND PACKAGE(LOCATION-NAME1.COL1) -
MEMBER(ASNDP000) -
ACTION(REP) -
RELEASE(COMMIT) ISOLATION(UR)
```

```
BIND PACKAGE(LOCATION-NAME1.COL1) -
MEMBER(ASNDB000) -
ACTION(REP) -
RELEASE(COMMIT) ISOLATION(UR)
```

```
BIND PACKAGE(LOCATION-NAME1.COL1) -
MEMBER(ASNDF000) -
ACTION(REP) -
RELEASE(COMMIT) ISOLATION(CS)
```

```
BIND PACKAGE(LOCATION-NAME1.COL1) -
MEMBER(ASNDA000) -
ACTION(REP) -
RELEASE(COMMIT) ISOLATION(UR)
```

```
BIND PACKAGE(LOCATION-NAME1.COL1) -
MEMBER(ASNDC000) -
ACTION(REP) -
RELEASE(COMMIT) ISOLATION(UR)
```

```

BIND PACKAGE(LOCATION-NAME2.COL1) -
MEMBER(ASNDI000) -
ACTION(REP) -
RELEASE(COMMIT) ISOLATION(UR)
.
.
.
.

BIND PACKAGE(LOCATION-NAME2.COL1) -
MEMBER(ASNDC000) -
ACTION(REP) -
RELEASE(COMMIT) ISOLATION(UR)

BIND PLAN(ASNA510) -
PKLIST(LOCATION-NAME1.COL1.*,LOCATION-NAME2.COL1.*) -
ACTION(REP) ACQUIRE(USE) -
RELEASE(COMMIT)
/*
//

```

Bind Package JCL for Apply (MVS for DB2 Version 5)

```

//*****
/* JOB NAME = ASNA2BD5 */
/* */
/* DESCRIPTIVE NAME = BIND PACKAGE JCL FOR THE IBM */
/* DATAPROPAGATOR RELATIONAL APPLY FOR MVS */
/* FOR DB2 V5 */
/* */
/* STATUS = VERSION 05 RELEASE 01 MODIFICATION LEVEL 00 */
/* */
/* FUNCTION = BIND THE DATAPROPAGATOR RELATIONAL */
/* APPLY FOR MVS PACKAGE FOR DB2 V5 */
/* */
/* LICENSED MATERIALS - PROPERTY OF IBM */
/* 5655-A22 (C) COPYRIGHT IBM CORP 1993, 1997 */
/* ALL RIGHTS RESERVED. */
/* US GOVERNMENT USERS RESTRICTED RIGHTS - */
/* USE, DUPLICATION OR DISCLOSURE RESTRICTED */
/* BY GSA ADP SCHEDULE CONTRACT WITH IBM CORP. */

```

```

/**                                                    */
/** NOTES =                                           */
/** 1) REVIEW ALL STATEMENTS.                         */
/** 2) CHANGE THE HLQUAL FIELD(S) TO A VALID HIGH LEVEL */
/**    QUALIFIER TO COMPLY WITH YOUR SITE'S NAMING STANDARDS. */
/**    ASNA.V5R1M0 IS RECOMMENDED AS THE HIGH LEVEL QUALIFIER. */
/** 3) DELETE THE TEXT THAT BEGINS WITH <== AT THE   */
/**    END OF SOME LINES                               */
/**                                                    */
/*******/
/**
//ASNA2BD5 JOB USER=USERID,PASSWORD=PASSWORD <== SPECIFY JOB CARD PARM
/*******/
/**    FOR EXECUTION OF APPLY FOR MVS BIND PROCEDURE */
/*******/
//ASNA2BD5 EXEC PGM=IKJEFT01
//SYSPRINT DD  SYSOUT=*
//SYSPRINT DD  SYSOUT=*
//SYSUDUMP DD  SYSOUT=*
//STEPLIB DD  DISP=SHR,DSN=DSN510.SDSNLOAD
//DBRMLIB DD  DISP=SHR,DSN=HLQUAL.SASNADB M <== VERIFY FILE NAME
//SYSTSIN DD  *
DSN S(D51A)                <== REPLACE D51A WITH YOUR SUBSYSTEM NAME
                           <== VERIFY DB2 LOCATION NAME
                           <== AND COLLECTION-ID;
                           <== INCLUDE THE 7 BIND PACKAGE SUBCOMMANDS
                           <== FOR EACH DB2 TO BE ACCESSED BY
                           <== THE APPLY PROGRAM

BIND PACKAGE(LOCATION-NAME1.COL1) -
MEMBER(ASNDI000) -
ACTION(REP) -
RELEASE(COMMIT) ISOLATION(UR)

BIND PACKAGE(LOCATION-NAME1.COL1) -
MEMBER(ASND4000) -
ACTION(REP) -
RELEASE(COMMIT) ISOLATION(UR)

```

```

|         BIND PACKAGE(LOCATION-NAME1.COL1) -
|         MEMBER(ASNDP000) -
|         ACTION(REP) -
|         RELEASE(COMMIT) ISOLATION(UR)
|
|         BIND PACKAGE(LOCATION-NAME1.COL1) -
|         MEMBER(ASNDB000) -
|         ACTION(REP) -
|         RELEASE(COMMIT) ISOLATION(UR)
|
|         BIND PACKAGE(LOCATION-NAME1.COL1) -
|         MEMBER(ASNDF000) -
|         ACTION(REP) -
|         RELEASE(COMMIT) ISOLATION(CS)
|
|         BIND PACKAGE(LOCATION-NAME1.COL1) -
|         MEMBER(ASNDA000) -
|         ACTION(REP) -
|         RELEASE(COMMIT) ISOLATION(UR)
|
|         BIND PACKAGE(LOCATION-NAME1.COL1) -
|         MEMBER(ASNDC000) -
|         ACTION(REP) -
|         RELEASE(COMMIT) ISOLATION(UR)
|         BIND PACKAGE(LOCATION-NAME2.COL1) -
|         MEMBER(ASNDI000) -
|         ACTION(REP) -
|         RELEASE(COMMIT) ISOLATION(UR)
|         .
|         .
|         .
|
|         BIND PACKAGE(LOCATION-NAME2.COL1) -
|         MEMBER(ASNDC000) -
|         ACTION(REP) -
|         RELEASE(COMMIT) ISOLATION(UR)
|
|         BIND PLAN(ASNA510) -
|         PKLIST(LOCATION-NAME1.COL1.*,LOCATION-NAME2.COL1.*) -
|         ACTION(REP) ACQUIRE(USE) -
|         RELEASE(COMMIT)
|         /*
|         //

```

Invocation JCL for Apply (MVS for DB2 Version 3)

```

/*******
/** JOB NAME = ASNA2RN3 */
/** */
/** DESCRIPTIVE NAME = INVOCATION JCL FOR THE IBM */
/** DATAPROPAGATOR RELATIONAL APPLY FOR MVS */
/** */
/** STATUS = VERSION 05 RELEASE 01 MODIFICATION LEVEL 00 */
/** */
/** FUNCTION = INVOKE THE DATAPROPAGATOR RELATIONAL */
/** APPLY FOR MVS PROGRAM */
/** */
/** LICENSED MATERIALS - PROPERTY OF IBM */
/** 5655-A22 (C) COPYRIGHT IBM CORP 1993, 1997 */
/** ALL RIGHTS RESERVED. */
/** US GOVERNMENT USERS RESTRICTED RIGHTS - */
/** USE, DUPLICATION OR DISCLOSURE RESTRICTED */
/** BY GSA ADP SCHEDULE CONTRACT WITH IBM CORP. */
/** */
/** NOTES = */
/** 1) REVIEW ALL STATEMENTS. */
/** 2) CHANGE THE HLQUAL FIELD(S) TO A VALID HIGH LEVEL */
/** QUALIFIER TO COMPLY WITH YOUR SITE'S NAMING STANDARDS. */
/** ASNA.V5R1M0 IS RECOMMENDED AS THE HIGH LEVEL QUALIFIER. */
/** 3) MODIFY THE BLKSIZE IN THE ASNASPL DD STATEMENT TO THE */
/** ONE THAT IS MOST APPROPRIATE AT YOUR INSTALLATION */
/** 4) IF YOU ARE RUNNING UNDER LE/370 ENVIRONMENT, ADD */
/** REGION=3M IN THE EXEC STATEMENT. */
/** 5) DELETE THE TEXT THAT BEGINS WITH <== AT THE */
/** END OF SOME LINES */
/** 6) REPLACE DB31 WITH YOUR SUBSYSTEM NAME */
/** */
/*******
/**
/**ASNARUN JOB USER=USERID,PASSWORD=PASSWORD <== SPECIFY JOB CARD PARM
/** FOR EXECUTION OF THE APPLY FOR MVS PROGRAM

```

```

/*
/*****
//ASNARUN EXEC PGM=ASNAPV23,
//          PARM='APPLY_QUALIFIER DB31 CONTROL_SERVER DISK TRCFLOW'
/*          <== VERIFY APPLY_QUALIFIER
/*          <== VERIFY DB2 SUBSYSTEM ID
/*          <== VERIFY CONTROL_SERVER
//SYSTEM DD SYSOUT=*
//SYSPRT DD SYSOUT=*
//SYSUDUMP DD SYSOUT=*
//SYSPRINT DD SYSOUT=*
//STEPLIB DD DSN=HLQUAL.SASNALNK,DISP=SHR <== VERIFY FILE NAME
//          DD DSN=DB2.V3R1.SDSNLOAD,DISP=SHR <== VERIFY DB2 AND
//          DD DSN=EDC.V2R1M0.SEDCLINK,DISP=SHR <== C/370 LIBRARY
//          DD DSN=PLI.V2R3M0.SIBMLINK,DISP=SHR <== NAMES
//MSGSD DD DSN=HLQUAL.MSGS,DISP=SHR <== VERIFY FILE NAME
//ASNASPL DD DSN=&&ASNASPL,
//          DISP=(NEW,DELETE,DELETE),
//          UNIT=SYSDA,SPACE=(CYL,(5,5)), <== VERIFY UNIT
//          DCB=(RECFM=VB,BLKSIZE=6404) <== VERIFY BLKSIZE
//

```

Invocation JCL for Apply (MVS for DB2 Version 4)

```

/*****
/* JOB NAME = ASNA2RN4 */
/* */
/* DESCRIPTIVE NAME = INVOCATION JCL FOR THE IBM */
/*          DATAPROPAGATOR RELATIONAL APPLY FOR MVS */
/* */
/* STATUS = VERSION 05 RELEASE 01 MODIFICATION LEVEL 00 */
/* */
/* FUNCTION = INVOKE THE DATAPROPAGATOR RELATIONAL */
/*          APPLY FOR MVS PROGRAM */

```

```

/**                                                    */
/**      LICENSED MATERIALS - PROPERTY OF IBM          */
/**      5655-A22 (C) COPYRIGHT IBM CORP 1993, 1997  */
/**      ALL RIGHTS RESERVED.                          */
/**      US GOVERNMENT USERS RESTRICTED RIGHTS -     */
/**      USE, DUPLICATION OR DISCLOSURE RESTRICTED   */
/**      BY GSA ADP SCHEDULE CONTRACT WITH IBM CORP.  */
/**                                                    */
/** NOTES =                                           */
/** 1) REVIEW ALL STATEMENTS.                         */
/** 2) CHANGE THE HLQUAL FIELD(S) TO A VALID HIGH LEVEL */
/**    QUALIFIER TO COMPLY WITH YOUR SITE'S NAMING STANDARDS. */
/**    ASNA.V5R120M1 IS RECOMMENDED AS HIGH LEVEL QUALIFIER. */
/** 3) MODIFY THE BLKSIZE IN THE ASNASPL DD STATEMENT TO THE */
/**    ONE THAT IS MOST APPROPRIATE AT YOUR INSTALLATION */
/** 4) IF YOU ARE RUNNING UNDER LE/370 ENVIRONMENT, ADD */
/**    REGION=3M IN THE EXEC STATEMENT.                */
/** 5) DELETE THE TEXT THAT BEGINS WITH <== AT THE   */
/**    END OF SOME LINES                               */
/** 6) REPLACE DB24 WITH YOUR SUBSYSTEM NAME          */
/**                                                    */
/*******/
/**
//ASNARUN JOB USER=USERID,PASSWORD=PASSWORD <== SPECIFY JOB CARD PARM
//                FOR EXECUTION OF THE APPLY FOR MVS PROGRAM
/**
/*******/
//ASNARUN EXEC PGM=ASNAPV24,
//          PARM='APPLY_QUALIFIER DB41 CONTROL_SERVER DISK TRCFLOW'
/**          <== VERIFY APPLY_QUALIFIER
/**          <== VERIFY DB2 SUBSYSTEM ID
/**          <== VERIFY CONTROL_SERVER
//SYSTEM DD SYSOUT=*
//SYSTSPRT DD SYSOUT=*
//SYSUDUMP DD SYSOUT=*
//SYSPRINT DD SYSOUT=*
//STEPLIB DD DSN=HLQUAL.SASNALNK,DISP=SHR           <== VERIFY FILE NAME
//          DD DSN=DSN410.SDSNLOAD,DISP=SHR         <== VERIFY DB2 AND
//          DD DSN=EDC.V2R1M0.SEDCLINK,DISP=SHR     <== C/370 LIBRARY
//          DD DSN=PLI.V2R3M0.SIBMLINK,DISP=SHR     <== NAMES
//MSGSD   DD DSN=HLQUAL.MSGS,DISP=SHR              <== VERIFY FILE NAME
//ASNASPL DD DSN=&&ASNASPL,
//          DISP=(NEW,DELETE,DELETE),
//          UNIT=SYSDA,SPACE=(CYL,(5,5)),          <== VERIFY UNIT
//          DCB=(RECFM=VB,BLKSIZE=6404)            <== VERIFY BLKSIZE
//

```

Invocation JCL for Apply (MVS for DB2 Version 5)


```

//*****
//* JOB NAME = ASNA2RN5 */
//* */
//* DESCRIPTIVE NAME = INVOCATION JCL FOR THE IBM */
//* DATAPROPAGATOR RELATIONAL APPLY FOR MVS */
//* */
//* STATUS = VERSION 05 RELEASE 01 MODIFICATION LEVEL 00 */
//* */
//* FUNCTION = INVOKE THE DATAPROPAGATOR RELATIONAL */
//* APPLY FOR MVS PROGRAM */
//* */
//* LICENSED MATERIALS - PROPERTY OF IBM */
//* 5655-A22 (C) COPYRIGHT IBM CORP 1993, 1997 */
//* ALL RIGHTS RESERVED. */
//* US GOVERNMENT USERS RESTRICTED RIGHTS - */
//* USE, DUPLICATION OR DISCLOSURE RESTRICTED */
//* BY GSA ADP SCHEDULE CONTRACT WITH IBM CORP. */
//* */
//* NOTES = */
//* 1) REVIEW ALL STATEMENTS. */
//* 2) CHANGE THE HLQUAL FIELD(S) TO A VALID HIGH LEVEL */
//* QUALIFIER TO COMPLY WITH YOUR SITE'S NAMING STANDARDS. */
//* ASNA.V5R1M0 IS RECOMMENDED AS THE HIGH LEVEL QUALIFIER. */
//* 3) MODIFY THE BLKSIZE IN THE ASNASPL DD STATEMENT TO THE */
//* ONE THAT IS MOST APPROPRIATE AT YOUR INSTALLATION */
//* 4) IF YOU ARE RUNNING UNDER LE/370 ENVIRONMENT, ADD */
//* REGION=3M IN THE EXEC STATEMENT. */
//* 5) DELETE THE TEXT THAT BEGINS WITH <== AT THE */
//* END OF SOME LINES */
//* 6) REPLACE D51A WITH YOUR SUBSYSTEM NAME */
//* */
//*****
//*
//ASNARUN JOB USER=USERID,PASSWORD=PASSWORD <== SPECIFY JOB CARD PARM
//* FOR EXECUTION OF THE APPLY FOR MVS PROGRAM
//*
//*****

```

```

//ASNARUN EXEC PGM=ASNAPV25,
//          PARM='APPLY_QUALIFIER DB51 CONTROL_SERVER DISK TRCFLOW'
//*          <== VERIFY APPLY_QUALIFIER
//*          <== VERIFY DB2 SUBSYSTEM ID
//*          <== VERIFY CONTROL_SERVER
//SYSTEM DD SYSOUT=*
//SYSTSPRT DD SYSOUT=*
//SYSUDUMP DD SYSOUT=*
//SYSPRINT DD SYSOUT=*
//STEPLIB DD DISP=SHR,DSN=HLQUAL.SASNALNK <== VERIFY FILE NAME
//          DD DISP=SHR,DSN=DSN510.SDSNLOAD <== VERIFY DB2 AND
//          DD DISP=SHR,DSN=EDC.V2R1M0.SEDCLINK <== C/370 LIBRARY
//          DD DISP=SHR,DSN=PLI.V2R3M0.SIBMLINK <== NAMES
//MSGSDD DISP=SHR,DSN=HLQUAL.MSGS <== VERIFY FILE NAME
//ASNASPL DD DSN=&&ASNASPL,
//          DISP=(NEW,DELETE,DELETE),
//          UNIT=SYSDA,SPACE=(CYL,(5,5)), <== VERIFY UNIT
//          DCB=(RECFM=VB,BLKSIZE=6404) <== VERIFY BLKSIZE
//

```

Part 7. Appendixes

Appendix A. How the DB2 Library Is Structured

The DB2 Universal Database library consists of SmartGuides, online help, and books. 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 "Information Center" on page 439 for details.

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. The following table lists the SmartGuides.

Note: Not all SmartGuides are available for the partitioned database environment.

SmartGuide	Helps you to...	How to Access...
<i>Add Database</i>	Catalog a database on a client workstation.	From the Client Configuration Assistant, click on Add .
<i>Create Database</i>	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->New .
<i>Performance Configuration</i>	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 performance .
<i>Backup Database</i>	Determine, create, and schedule a backup plan.	From the Control Center, click with the right mouse button on the database you want to backup and select Backup->Database using SmartGuide .
<i>Restore Database</i>	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 .
<i>Create Table</i>	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 .
<i>Create Table Space</i>	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 .

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 “Information Center” on page 439.

Type of Help	Contents	How to Access...
<i>Command Help</i>	Explains the syntax of commands in the command line processor.	From the command line processor in interactive mode, enter: <i>? command</i> where <i>command</i> 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.
<i>Control Center Help</i>	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 on the Help push button or press the F1 key.
<i>Message Help</i>	Describes the cause of a message, and any action you should take.	From the command line processor in interactive mode, enter: <i>? XXXnnnnn</i> where <i>XXXnnnnn</i> is a valid message identifier. For example, ? SQL30081 displays help about the SQL30081 message. To view message help one screen at a time, enter: <i>? XXXnnnnn more</i> To save message help in a file, enter: <i>? XXXnnnnn > filename.ext</i> where <i>filename.ext</i> is the file where you want to save the message help.
<i>SQL Help</i>	Explains the syntax of SQL statements.	From the command line processor in interactive mode, enter: help statement where <i>statement</i> is an SQL statement. For example, help SELECT displays help about the SELECT statement.

Type of Help	Contents	How to Access...
<i>SQLSTATE Help</i>	Explains SQL states and class codes.	From the command line processor in interactive mode, enter: <i>? sqlstate</i> or <i>? class-code</i> where <i>sqlstate</i> is a valid five-digit SQL state and the <i>class-code</i> is 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 Books

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 UNIX-based and Intel-based platforms.

Platform-specific books These books are for DB2 on a specific platform. For example, for DB2 on OS/2, on Windows NT, and on the UNIX-based platforms, there are separate *Quick Beginnings* books.

Most books are available in HTML and PostScript format, and in hardcopy that you can order from IBM. The exceptions are noted in the table.

If you want to read the English version of the books, they are always provided in the directory that contains the English documentation.

You can obtain DB2 books and access information in a variety of different ways:

View	See "Viewing Online Books" on page 437.
Search	See "Searching Online Books" on page 438.
Print	See "Printing the PostScript Books" on page 438.
Order	See "Ordering the Printed DB2 Books" on page 439.

Book Name	Book Description	Form Number File Name
Cross-Platform Books		
<i>Administration Getting Started</i>	Introduces basic DB2 database administration concepts and tasks, and walks you through the primary administrative tasks.	S10J-8154 db2k0x50
<i>Administration Guide</i>	Contains information required to design, implement, and maintain a database to be accessed either locally or in a client/server environment.	S10J-8157 db2d0x51
<i>API Reference</i>	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.	S10J-8167 db2b0x51

Book Name	Book Description	Form Number File Name
<i>CLI Guide and Reference</i>	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.	S10J-8159 db2l0x50
<i>Command Reference</i>	Explains how to use the command line processor, and describes the DB2 commands you can use to manage your database.	S10J-8166 db2n0x51
<i>DB2 Connect Enterprise Edition Quick Beginnings</i>	Provides planning, migrating, installing, configuring, and using information for DB2 Connect Enterprise Edition. Also contains installation and setup information for all supported clients.	S10J-7888 db2cyx51
<i>DB2 Connect Personal Edition for Windows and OS/2 Enviroments Quick Beginnings</i>	Provides planning, installing, configuring, and using information for DB2 Connect Personal Edition.	S10J-8162 db2c1x51
<i>DB2 Connect User's Guide</i>	Provides concepts, programming and general using information about the DB2 Connect products.	S10J-8163 db2c0x51
<i>DB2 Connectivity Supplement</i>	Provides setup and reference information for customers who want 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 customers who want to use DRDA Application Servers with DB2 Connect (formerly DDCS) application requesters. Note: Available in HTML and PostScript formats only.	No form number db2h1x51
<i>Embedded SQL Programming Guide</i>	Explains how to develop applications that access DB2 databases using embedded SQL, and includes discussions about programming techniques and performance considerations.	S10J-8158 db2a0x50
<i>Glossary</i>	Provides a comprehensive list of all DB2 terms and definitions. Note: Available in HTML format only.	No form number db2t0x50
<i>Installing and Configuring DB2 Clients</i>	Provides installation and setup information for all DB2 Client Application Enablers and DB2 Software Developer's Kits. Note: Available in HTML and PostScript formats only.	No form number db2iyx51
<i>Master Index</i>	Contains a cross reference to the major topics covered in the DB2 library. Note: Available in PostScript format and hardcopy only.	S10J-8170 db2w0x50
<i>Message Reference</i>	Lists messages and codes issued by DB2, and describes the actions you should take.	S10J-8168 db2m0x51

Book Name	Book Description	Form Number File Name
<i>DB2 Replication Guide and Reference</i>	Provides planning, configuring, administering, and using information for the IBM Replication tools supplied with DB2.	S95H-0999 db2e0x52
<i>Road Map to DB2 Programming</i>	Introduces the different ways your applications can access DB2, describes key DB2 features you can use in your applications, and points to detailed sources of information for DB2 programming.	S10J-8155 db2u0x50
<i>SQL Getting Started</i>	Introduces SQL concepts, and provides examples for many constructs and tasks.	S10J-8156 db2y0x50
<i>SQL Reference</i>	Describes SQL syntax, semantics, and the rules of the language. Also includes information about release-to-release incompatibilities, product limits, and catalog views.	S10J-8165 db2s0x51
<i>System Monitor Guide and Reference</i>	Describes how to collect different kinds of information about your database and the database manager. Explains how you can use the information to understand database activity, improve performance, and determine the cause of problems.	S10J-8164 db2f0x50
<i>Troubleshooting Guide</i>	Helps you determine the source of errors, recover from problems, and use diagnostic tools in consultation with DB2 Customer Service.	S10J-8169 db2p0x50
<i>What's New</i>	Describes the new features, functions, and enhancements in DB2 Universal Database, Version 5, including information about Java-based tools.	S04L-6230 db2q0x51
Platform-Specific Books		
<i>Building Applications for UNIX Environments</i>	Provides environment setup information and step-by-step instructions to compile, link, and run DB2 applications on a UNIX system.	S10J-8161 db2axx51
<i>Building Applications for Windows and OS/2 Environments</i>	Provides environment setup information and step-by-step instructions to compile, link, and run DB2 applications on a Windows or OS/2 system.	S10J-8160 db2a1x50
<i>DB2 Personal Edition Quick Beginnings</i>	Provides planning, installing, migrating, configuring, and using information for DB2 Universal Database Personal Edition on OS/2, Windows 95, and the Windows NT operating systems.	S10J-8150 db2i1x50
<i>DB2 SDK for Macintosh Building Your Applications</i>	Provides environment setup information and step-by-step instructions to compile, link, and run DB2 applications on a Macintosh system. Note: Available in PostScript format and hardcopy for DB2 Version 2.1.2 only.	S50H-0528 sqla7x02
<i>DB2 SDK for SCO OpenServer Building Your Applications</i>	Provides environment setup information and step-by-step instructions to compile, link, and run DB2 applications on a SCO OpenServer system. Note: Available for DB2 Version 2.1.2 only.	S89H-3242 sqla9x02

Book Name	Book Description	Form Number File Name
<i>DB2 SDK for SINIX Building Your Applications</i>	Provides environment setup information and step-by-step instructions to compile, link, and run DB2 applications on a SINIX system. Note: Available in PostScript format and hardcopy for DB2 Version 2.1.2 only.	S50H-0530 sqla8x00
<i>Quick Beginnings for OS/2</i>	Provides planning, installing, migrating, configuring, and using information for DB2 Universal Database on OS/2. Also contains installing and setup information for all supported clients.	S10J-8147 db2i2x50
<i>Quick Beginnings for UNIX</i>	Provides planning, installing, configuring, migrating, and using information for DB2 Universal Database on UNIX-based platforms. Also contains installing and setup information for all supported clients.	S10J-8148 db2ixx51
<i>Quick Beginnings for Windows NT</i>	Provides planning, installing, configuring, migrating, and using information for DB2 Universal Database on the Windows NT operating system. Also contains installing and setup information for all supported clients.	S10J-8149 db2i6x50
<i>DB2 Extended Edition for UNIX Quick Beginnings</i>	Provides planning, installing, configuring, and using information for DB2 Universal Database Extended Enterprise Edition for UNIX. This book supercedes the <i>DB2 Extended Enterprise Edition Quick Beginnings for AIX</i> book, and is suitable for use with all versions of DB2 Extended Enterprise Edition that run on UNIX-based platforms.	S99H-8314 db2v3x51
<i>DB2 Extended Edition for Windows NT Quick Beginnings</i>	Provides planning, installing, configuring, and using information for DB2 Universal Database Extended Enterprise Edition for Windows NT.	S09L-6713 db2v6x51

Notes:

1. The character in the sixth position of the file name indicates the language of a book. For example, the file name db2d0e50 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	Language	Identifier
Brazilian Portuguese	B	Japanese	J
Bulgarian	U	Korean	K
Czech	X	Norwegian	N
Danish	D	Polish	P
English	E	Russian	R
Finnish	Y	Simp. Chinese	C
French	F	Slovenia	L
German	G	Spanish	Z
Greek	A	Swedish	S
Hungarian	H	Trad. Chinese	T

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_05_00 on AIX
 - /opt/IBMcdb2/V5.0 on HP-UX, Solaris, Gemini, and SGI.
 - On other platforms, see the RELEASE.TXT file. This file is located in the directory where the product is installed.

Viewing Online Books

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 use any HTML Version 3.2-compliant browser to view the online books.

To view online books:

- If you are running DB2 administration tools, use the Information Center. See “Information Center” on page 439 for details.
- Use the open file function of your Web browser. The page you open contains descriptions of and links to DB2 books:
 - On UNIX-based platforms, open the following page:
`file:/INSTHOME/sql1lib/doc/%L/html/index.htm`
where %L is the locale name.
 - On other platforms, open the following page:
`sql1lib\doc\html\index.htm`

The path is located on the drive where DB2 is installed.

You can also 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.

Note: The **DB2 Online Books** icon is only available if you do not install the Information Center.

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 sub-directories from \sql1lib\doc\html on your local system to a web server. Each book has its own sub-directory 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 *Setting up DB2 Online Documentation on a Web Server* at:

<http://www.software.ibm.com/data/pubs/papers/db2html.html>

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 such as:
 - List of books
 - Tables of contents of frequently used books
 - Frequently referenced articles like the *ALTER TABLE* topic
 - Search form.

For information about setting up a search, see the *What's New* book.

Searching Online Books

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.
- 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 "Information Center" on page 439 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 Books" on page 433.

Note: 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. The result from this step is a printable PostScript file with a file extension of .psz.

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.psz
```

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:

```
zcat filename | lp -d PSprinter_queue
```

- For Silicon Graphics IRIX and SINIX:

```
zcat < filename | lp -d PSprinter_queue
```

where *filename* is the name of 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 *Quick Beginnings for UNIX* on AIX, you can use the following command:

```
zcat /cdrom/doc/en/ps/db2ixe50.ps.Z | qprt -P ps1
```

Ordering the Printed DB2 Books

You can order the printed DB2 manuals either as a set, or individually. There are two sets of books available. The form number for the entire set of DB2 books is SB0F-8915-00. The form number for the books listed under the heading "Cross-Platform Books" is SB0F-8914-00.

Note: These form numbers only apply if you are ordering books that are printed in the English language.

You can also order books individually by the form number listed in "DB2 Books" on page 433. 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.

Information Center

The Information Center provides quick access to DB2 product information. You must install the DB2 administration tools to obtain the Information Center.

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
- **db2ic** command.

The Information Center provides the following kinds of information. Click on 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, follow the *Search DB2 Books* link in each HTML file, or use the search feature of the help viewer.

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 via 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.

Appendix B. Tables at a Glance

The following two diagrams provide a view-at-a-glance look at the control and source subscription tables, table keys, and their parameters.

Control Server Subscription Tables

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_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
EL_OR_CALL	CHAR (1) NOT NULL
SQL_STMT	VARCHAR (1024)
ACCEPT_SQLSTATES	VARCHAR (50)

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_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

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)

Used by APPLY

Figure 81. The Control Server Subscription Tables. The control server subscription tables used by the Apply and Capture program.

Source Server Registration Tables

<p>ASN.IBMSNAP_TRACE</p> <p>(no primary key)</p> <table> <tr> <td>OPERATION</td> <td>CHAR (8) NOT NULL</td> </tr> <tr> <td>TRACE_TIME</td> <td>TIMESTAMP NOT NULL</td> </tr> <tr> <td>DESCRIPTION</td> <td>VARCHAR (254) NOT NULL</td> </tr> </table>	OPERATION	CHAR (8) NOT NULL	TRACE_TIME	TIMESTAMP NOT NULL	DESCRIPTION	VARCHAR (254) NOT NULL	<p>ASN.IBMSNAP_CCPPARMS</p> <p>(no primary key)</p> <table> <tr> <td>RETENTION_LIMIT</td> <td>INT</td> </tr> <tr> <td>LAG_LIMIT</td> <td>INT</td> </tr> <tr> <td>COMMIT_INTERVAL</td> <td>INT</td> </tr> <tr> <td>PRUNE_INTERVAL</td> <td>INT</td> </tr> </table>	RETENTION_LIMIT	INT	LAG_LIMIT	INT	COMMIT_INTERVAL	INT	PRUNE_INTERVAL	INT																																																																
OPERATION	CHAR (8) NOT NULL																																																																														
TRACE_TIME	TIMESTAMP NOT NULL																																																																														
DESCRIPTION	VARCHAR (254) NOT NULL																																																																														
RETENTION_LIMIT	INT																																																																														
LAG_LIMIT	INT																																																																														
COMMIT_INTERVAL	INT																																																																														
PRUNE_INTERVAL	INT																																																																														
<p>ASN.IBMSNAP_WARM_START</p> <p>(no primary key)</p> <table> <tr> <td>SEQ</td> <td>CHAR (10) FOR BIT DATA</td> </tr> <tr> <td>AUTHTKN</td> <td>CHAR (12)</td> </tr> <tr> <td>AUTHID</td> <td>CHAR (18)</td> </tr> <tr> <td>CAPTURED</td> <td>CHAR (1)</td> </tr> <tr> <td>UOWTIME</td> <td>INT</td> </tr> </table>	SEQ	CHAR (10) FOR BIT DATA	AUTHTKN	CHAR (12)	AUTHID	CHAR (18)	CAPTURED	CHAR (1)	UOWTIME	INT	<p>ASN.IBMSNAP_CRITSEC</p> <table> <tr> <td>APPLY_QUAL</td> <td>CHAR (18) NOT NULL</td> </tr> </table>	APPLY_QUAL	CHAR (18) NOT NULL																																																																		
SEQ	CHAR (10) FOR BIT DATA																																																																														
AUTHTKN	CHAR (12)																																																																														
AUTHID	CHAR (18)																																																																														
CAPTURED	CHAR (1)																																																																														
UOWTIME	INT																																																																														
APPLY_QUAL	CHAR (18) NOT NULL																																																																														
<p>ASN.IBMSNAP_REGISTER</p> <p>SOURCE_OWNER, SOURCE_TABLE, SOURCE_VIEW_QUAL</p> <table> <tr> <td>SOURCE_OWNER</td> <td>CHAR (18) NOT NULL</td> </tr> <tr> <td>SOURCE_TABLE</td> <td>CHAR (18) NOT NULL</td> </tr> <tr> <td>SOURCE_VIEW_QUAL</td> <td>SMALLINT NOT NULL</td> </tr> <tr> <td>GLOBAL_RECORD</td> <td>CHAR (1) NOT NULL</td> </tr> <tr> <td>SOURCE_STRUCTURE</td> <td>SMALLINT NOT NULL</td> </tr> <tr> <td>SOURCE_CONDENSED</td> <td>CHAR (1) NOT NULL</td> </tr> <tr> <td>SOURCE_COMPLETE</td> <td>CHAR (1) NOT NULL</td> </tr> <tr> <td>CD_OWNER</td> <td>CHAR (18)</td> </tr> <tr> <td>CD_TABLE</td> <td>CHAR (18)</td> </tr> <tr> <td>PHYS_CHANGE_OWNER</td> <td>CHAR (18)</td> </tr> <tr> <td>PHYS_CHANGE_TABLE</td> <td>CHAR (18)</td> </tr> <tr> <td>CD_OLD_SYNCHPOINT</td> <td>CHAR (10) FOR BIT DATA</td> </tr> <tr> <td>CD_NEW_SYNCHPOINT</td> <td>CHAR (10) FOR BIT DATA</td> </tr> <tr> <td>DISABLE_REGRESH</td> <td>SMALLINT NOT NULL</td> </tr> <tr> <td>CCD_OWNER</td> <td>CHAR (18)</td> </tr> <tr> <td>CCD_TABLE</td> <td>CHAR (18)</td> </tr> <tr> <td>CCD_OLD_SYNCHPOINT</td> <td>CHAR (10) FOR BIT DATA</td> </tr> <tr> <td>SYNCHPOINT</td> <td>CHAR (10) FOR BIT DATA</td> </tr> <tr> <td>SYNCHTIME</td> <td>TIMESTAMP</td> </tr> <tr> <td>CCD_CONDENSED</td> <td>CHAR (1)</td> </tr> <tr> <td>CCD_COMPLETE</td> <td>CHAR (1)</td> </tr> <tr> <td>ARCH_LEVEL</td> <td>CHAR (4) NOT NULL</td> </tr> <tr> <td>DESCRIPTION</td> <td>CHAR(254)</td> </tr> <tr> <td>BEFORE_IMG_PREFIX</td> <td>VARCGAR (4)</td> </tr> <tr> <td>CONFLICT_LEVEL</td> <td>CHAR (1)</td> </tr> <tr> <td>PARTITION_KEYS_CHG</td> <td>CHAR (1)</td> </tr> </table>	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_REGRESH	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	VARCGAR (4)	CONFLICT_LEVEL	CHAR (1)	PARTITION_KEYS_CHG	CHAR (1)	<p>ASN.IBMSNAP_PRUNCNTL</p> <p>SOURCE_OWNER, SOURCE_TABLE, SOURCE_VIEW_QUAL, SET_NAME, TARGET_SERVER, TARGET_TABLE, TARGET_OWNER</p> <table> <tr> <td>TARGET_SERVER</td> <td>CHAR (18) NOT NULL</td> </tr> <tr> <td>TARGET_OWNER</td> <td>CHAR (18) NOT NULL</td> </tr> <tr> <td>TARGET_TABLE</td> <td>CHAR (18) NOT NULL</td> </tr> <tr> <td>SYNCHTIME</td> <td>TIMESTAMP</td> </tr> <tr> <td>SYNCHPOINT</td> <td>CHAR (10) FOR BIT DATA</td> </tr> <tr> <td>SOURCE_OWNER</td> <td>CHAR (18) NOT NULL</td> </tr> <tr> <td>SOURCE_TABLE</td> <td>CHAR (18) NOT NULL</td> </tr> <tr> <td>SOURCE_VIEW_QUAL</td> <td>SMALLINT NOT NULL</td> </tr> <tr> <td>APPLY_QUAL</td> <td>CHAR (18) NOT NULL</td> </tr> <tr> <td>SET_NAME</td> <td>CHAR (18) NOT NULL</td> </tr> <tr> <td>CNTL_SERVER</td> <td>CHAR (18) NOT NULL</td> </tr> <tr> <td>TARGET_STRUCTURE</td> <td>SMALLINT NOT NULL</td> </tr> <tr> <td>CNTL_ALIAS</td> <td>CHAR (8)</td> </tr> </table>	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)
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_REGRESH	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	VARCGAR (4)																																																																														
CONFLICT_LEVEL	CHAR (1)																																																																														
PARTITION_KEYS_CHG	CHAR (1)																																																																														
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)																																																																														
	<p>ASN.IBMSNAP_UOW</p> <p>IBMSNAP_COMMITSEQ ASC, IBMSNAP_UOWID ASC, IBMSNAP_LOGMAKER ASC</p> <table> <tr> <td>IBMSNAP_UOWID</td> <td>CHAR (10) FOR BIT DATA NOT NULL</td> </tr> <tr> <td>IBMSNAP_COMMITSEQ</td> <td>CHAR (10) FOR BIT DATA NOT NULL</td> </tr> <tr> <td>IBMSNAP_LOGMAKER</td> <td>TIMESTAMP NOT NULL</td> </tr> <tr> <td>IBMSNAP_AUTHTKN</td> <td>CHAR (12) NOT NULL</td> </tr> <tr> <td>IBMSNAP_AUTHID</td> <td>CHAR (18) NOT NULL</td> </tr> <tr> <td>IBMSNAP_REJ_CODE</td> <td>CHAR (1) NOT NULL WITH DEFAULT</td> </tr> <tr> <td>IBMSNAP_APPLY_QUAL</td> <td>CHAR (18) NOT NULL WITH DEFAULT</td> </tr> </table>	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																																																																
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																																																																														

Used by CAPTURE

Used by CAPTURE and APPLY

Figure 82. The Source Server Registration Tables. The source server registration tables used by the Apply and Capture program.

Appendix C. What's New in IBM Replication

The DataPropagator Relational Version 1 (DPROPR V1) products have been updated for Version 5 (V5) and are now known collectively as IBM Replication. IBM Replication V5 provides the replication capabilities you learned to expect from DPROPR V1 plus many new features and integration with DB2 Universal Database V5.2.

Packaging Integration

Most Capture and Apply programs are now packaged with the database.

- IBM Replication for Windows NT, Windows 95, OS/2, AIX, Solaris, and HP-UX are packaged with DB2 Universal Database V5. DB2 Universal Database V5.2 features the addition of SCO UnixWare 7 to this list of platforms.
- DPROPR Capture for VSE and VM V5.1 are packaged with DB2 Server for VSE and VM which includes DB2 Universal Database for administration.
- DPROPR Capture and Apply for MVS V5.1 are separately available.

Improved Administration

IBM Replication provides a higher level of integration with DB2 on your client administration platforms.

- A lower cost of administration:
 - Administration for IBM Replication is installed with the DB2 Universal Database on the Windows NT, Windows 95, and OS/2 platforms and is handled through the Control Center, the administration tool of the DB2 Universal Database.
 - Administration through the Control Center provides features for power users to define multiple registration, now known as "defining replication sources," or multiple subscriptions, in a single user action.
- Replication administration now supports both "push" and "pull" configurations; these configurations allow the Apply program to push data to or pull data from the target table:
 - Push configurations are used for low latency requirements
 - Pull configurations provide better performance
 - Push and pull configurations enable update-anywhere and mobile replication
- IBM Replication administration now supports view replication sources and subscriptions:
 - You can define an n-way join view as a replication source in a single action.
 - You can subscribe to an n-way join replication source in a single action.
- The Control Center provides flexible customization of the replication environment for your site's requirements.

Batch Application Stream Integration

IBM Replication V5 provides better integration with your batch application stream.

- Distributed event management:

Your applications can drive subscriptions locally, or across an entire network.

- End-of-period controls:

You can specify not just *when* replication should occur, but *which* changes should be replicated, enabling you to coordinate replication with your business practices, such as daily accounting periods.

New Target Table Types

New target table types have been added to fit your application needs.

- User copy tables:

Read-only copies, without any additional timestamp or control columns

- Replica tables:

Horizontal fragments of your existing application tables, which are all updateable anywhere and anytime.

New Database Data Types

Support for two new database data types has been added to expand your application capabilities.

- BIGINT database data type:

BIGINT is a new database data type (SQLTYPE=492/493) that has the same characteristics as INTEGER but takes 8 bytes to store.

- LONG VARCHAR data type:

The Capture program now has the capability to support LONG VARCHAR data types.

Data Consistency

IBM Replication supports group subscriptions and replication for logical partitioning keys.

- Replication subscriptions have multiple members that can be grouped for referential integrity. All the members in the replication subscription are replicated in the same transaction.
- Support for logical partitioning keys provides independence from key updates. Subscription predicates are now as easy for updated columns as for non-updated columns. The partitioning of copy table spaces is independent of source table partitioning.

Improved Reliability

IBM Replication has improved reliability:

- Better resource handling for batch and recovery scenario
- Planned replication thresholds, achieving commit thresholds
- Automatic thresholds when secondary resource-limited conditions arise

Improved Performance for the Capture and Apply Programs

The Capture and Apply programs capture and replicate data faster and more efficiently.

- Reduced Apply program connect processing:

Grouped subscriptions reduce the number of times the Apply program connects to the source, control, and target servers. For example, when maintaining 40 copies, a group replication subscription replaces 40 individual subscriptions. Connect processing is reduced 40:1.

- Apply program notification:

The Capture program now notifies the Apply program when there are changes to be applied to avoid unnecessary dynamic SQL and joins at the source site.

- Non-blocking protocol:

- The Apply program uses a new non-blocking protocol with the Capture program, taking advantage of *uncommitted read* isolation to avoid contention and allows for a 100% Capture duty cycle, regardless of the Apply program's activity.
- The protocol with the Apply program is nonlocking, allowing you to choose the Capture program commit interval independently of the DB2 lock timeout value. The Apply program processes do not timeout if the Capture program commit interval is increased, which reduces the PREPARE and COMMIT costs of change capture.

- Static SQL performance for the Capture program:

The Capture program benefits from changes in DB2 for MVS Version 5.1, which allow dynamic SQL statements to be cached in the EDM pool, just as static SQL statements are cached. These changes effectively let the Capture program, a dynamic SQL application, run with static SQL performance.

Operational and Maintenance Enhancements

IBM Replication has made it easier for you to operate the replication tools and maintain the control information.

- Maintaining the Apply program is simpler:
 - Common subscription control tables reduce the effort for back up and recovery of definitions.

- A single database can be defined as the location of all subscription control information for the entire replication network.
- You can start Capture and Apply for Windows NT in the background using Windows NT services.
- You can now specify the database on the start commands for the Capture and Apply programs, allowing automated startup of these tools for multiple databases.

Expanded Support for Run-Time Processing Statements

You can run SQL statements and stored procedures at the beginning or end of each subscription cycle to manipulate data applied to your copies at both the source and target servers.

New Apply Program Invocation Parameters

IBM Replication has two new invocation parameters for starting the Apply program:

- The NOSLEEP parameter gives you the option of starting Apply as a one-time process so that the program stops if no new subscriptions are eligible for processing.
- The DELAY(n) parameter enables you to specify a specific time delay (from 0 to 6 seconds) at the end of an Apply cycle when you use continuous replication.

Simplified Authorization and Security

IBM Replication has a simpler administration structure than DPROPR V1, providing lower authorization costs and improved end-user authentication.

- An Apply program instance can now be run by more than one user ID, and DB2 for MVS RACF groups are supported.
- Password management for end-user authentication is more flexible; an Apply program instance can use different passwords when connecting to more than one server.

New Update-Anywhere Capability

IBM Replication now provides update-anywhere replication, enabling you to synchronize distributed databases. Update-anywhere replication provides the following support:

- Rigorous conflict detection mechanisms for declarative constraint conflicts and trigger-detected conflicts.
- Automatic compensation of conflicting transactions.
- Asynchronous transaction coordination:

Conflicting transactions are compensated much like the familiar DB2 rollback processing. Any detected dependent transactions are also rejected and compensated.

Conflicting transactions are marked in the Capture program control information so you can easily determine which transactions have been rejected.

Support for Occasionally Connected Systems

IBM Replication has enhanced the Capture and Apply programs to support mobile and occasionally connected systems.

- IBM Replication provides dial and disconnect exits for telephone connectivity.
- The Apply program can defer reporting the subscription progress to the pruning control table. Deferred reporting reduces remote SQL connect processing for a replication subscription to a single connect call and optimizes remote thread allocation and password verification.
- With enhanced support for push and pull configurations, mobile users need one Apply program on their laptop computer to replicate to and from the central computer and to connect as a mobile client for update-anywhere replication. The single Apply program process on the mobile client handles posting of updates replicating to or from the laptop. While disconnected, the mobile client imposes no burden on the network because central servers are not kept busy continuously looking for the mobile client. Rather, the laptop computer “finds” the central server when it dials in.

Support for Microsoft Databases

IBM's enterprise data replication solution has been extended 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.

Appendix D. 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,
500 Columbus Avenue,
Thornwood, NY, 10594
USA.

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
Department 071
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

The IBM Replication Guide and Reference primarily documents information that is NOT intended to be used as Programming Interfaces of IBM Replication.

The IBM Replication Guide and Reference also documents 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.

Trademarks

The following terms are trademarks or registered trademarks of the IBM Corporation in the United States and/or other countries:

ACF/VTAM	IBM
ADSTAR	IMS
AISPO	Lan Distance
AIX	MVS/ESA
AIXwindows	MVS/XA
AnyNet	NetView
APPN	OS/400
AS/400	OS/390
CICS	OS/2
C Set++	PowerPC
C/370	QMF
DATABASE 2	RACF
DatagLANce	RISC System/6000
DataHub	SAA
DataJoiner	SQL/DS
DataPropagator	SQL/400
DataRefresher	S/370
DB2	System/370
Distributed Relational Database Architecture	System/390
DRDA	SystemView
Extended Services	VisualAge
FFST	VM/ESA
First Failure Support Technology	VSE/ESA
	VTAM
	WIN-OS/2

Trademarks of Other Companies

Microsoft, Windows, Microsoft Access, and the Windows 95 logo are trademarks or registered trademarks of Microsoft Corporation.

UNIX is a registered trademark in the United States and other countries licensed exclusively through X/Open Company Limited.

Java HotJava and Solaris are trademarks of Sun Microsystems, Inc.

HP-UX is a trademark of Hewlett-Packard.

SINIX is a trademark of Siemens Nixdorf.

SCO UnixWare 7 is a trademark of Santa Cruz Operation, Inc.

Other company, product, or service names, which may be denoted by a double asterisk (**), may be trademarks or service marks of others.

Appendix E. 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>

Glossary

A

after-image. The updated content of a source table element that is recorded in a change data table or in a database log or journal. Contrast with *before-image*.

Apply program. A replication program that is used to refresh or update a target table, depending on the applicable source-to-target rules. Contrast with *Capture program*.

Apply qualifier. A character string that identifies subscription definitions that are unique to each instance of the Apply program.

Apply trail table. A replication source table at the control server that records a history of the full refreshes and differential refreshes 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*.

asynchronous batched update. A process in which all changes to the source are recorded and applied to existing target data at specified intervals. Contrast with *asynchronous continuous update*.

asynchronous continuous update. A process in which all changes to the source are recorded and applied to existing target data after being committed in the base table. Contrast with *asynchronous batched update*.

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. (T)

auto-registration. A process in which replica and consistent change data tables are automatically defined as replication sources at the target server.

B

base aggregate table. A target table type that contains data aggregated from a source table or a point-in-time table at intervals.

base table. A table created with the CREATE TABLE statement. Such a table has both its description and data physically stored in the database. Contrast with *view*.

before-image. The content of a source table element prior to a full refresh or differential refresh, 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.

bind. In SQL, the process by which the output from the SQL precompiler is converted to a usable structure called an access plan. During this process, access paths to the data are selected and some authorization checking is performed.

bind file. A file produced by the precompiler when the **bind** command or API is used with the BINDFILE option. This file includes information on all SQL statements in the application program.

BLOB. Binary large object.

C

Capture program. A replication program that reads database log or journal records to capture data about changes made to source tables. Contrast with *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.

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. The Capture program populates the CD table by copying the changes from the database log or journal. The contents of the CD table are then copied by the Apply program to the target table.

client. Any program (or workstation that it is running on) that communicates with and accesses a database server.

CLOB. Character large object.

cold start. A system start, using an initial program load procedure. Contrast with *warm start*.

common critical section table. A replication control table at the source server that is used to establish concurrency control between the Capture and Apply programs and to prevent an update replication cycle.

common 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 program.

common registrations 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.

common subscription columns table. A replication control table that contains column details of target tables.

common subscription events table. A replication control table that defines the events that trigger replication, including the event name and time.

common subscription set table. A replication control table that defines the members of a subscription set including the set name, Apply qualifier, source server, target server, and status.

common 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 set subscription cycle.

common subscription targets member table. A replication control table that maps the source and target table relationships within a subscription set.

complete. A table attribute indicating that the table contains a row for every primary key value of interest. As a result, a complete source table can be used to perform a full refresh of a target table.

condensed. A table attribute indicating that the table contains current data rather than a history of changes to the data. A condensed table includes no more than one row for each primary key value in the table. As a result, a condensed table can be used to supply current information for a full refresh .

conflict detection. The process of detecting an out-of-date row in a replica that was updated by a user application. 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*.

consistent change data (CCD) table. A replication table that is used for staging data, with four replication control columns. It can be one of the following types:

- An internal CCD table that is a join of the change data table and the unit-of-work table at the source server.
- A regular CCD table that is a copy of the internal CCD table on a remote server.
- An external source table that is not a DB2 origin table; it is manually updated with four replication columns and defined as a replication source 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 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 provided by the DBA Utility, Visual Explain, and Performance Monitor tools.

control server. The database location of the applicable subscription definitions and Apply trail table.

control table. A table in which replication source and subscription definitions or other replication control information is stored.

copy table. See target table.

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 units of work that are not committed and to recover a database to a consistent state. See also *primary log* and *secondary log*.

DAO. Data Access Object

database server. A functional unit that provides database services for databases.

DBA Utility. A tool that lets DB2 users configure databases and database manager instances, manage the directories necessary for accessing local and remote databases, back up and recover databases or table spaces, and manage media on a system using a graphical interface. The tasks provided by this tool can be accessed from the Control Center.

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, digit, or the underscore character.

Design Master. The original copy of a 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).

DMS table space. Database managed space table space.

E

enhanced conflict detection. Conflict detection that guarantees data integrity among all replicas and the origin 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* and *conflict detection*.

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. 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*.

foreign update. An update that was applied to a target table and replicated to the local table.

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 and there is potential loss of change data.

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.

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).

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.

long table space. A table space that can store only long string or large object (LOB) data.

M

member. See *subscription set member*.

migration.

1. The process of moving data from one computer system to another without converting the data.
2. Installation of a new version or release of a program to replace an earlier version or release.

mobile client. The node, usually a laptop computer, where the mobile enabler and replication source and target tables used in a mobile environment are located. The mobile replication mode is invoked 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 autonomously and continuously. This mode is invoked from the mobile client and allows data to be replicated when the mobile client is available for a connection 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. See 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.

performance monitor. A tool that lets database administrators use a graphical interface to monitor the performance of a DB2 system for tuning purposes. The tasks provided by this tool can be accessed from the Control Center.

point-in-time. 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.

primary log. A set of one or more log files used to record changes to a database. Storage for these files is allocated in advance. Contrast with *secondary log*.

Q

query. A request for information from the database based on specific conditions; for example, a request for a list of all customers in a customer table whose balance is greater than \$1000.

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 DPROPR to make the table available for subscription. This is done by using the DataJoiner replication administration tool. The result is a row inserted into the ASN.IBMSNAP_REGISTER TABLE, which is located at the source server. See also *replication source*.

regular table space. A table space that can store any nontemporary data.

rejected transaction. A transaction containing one or more updates from replica tables that are out of date in comparison to the origin table.

replica. A type of target table that can be updated locally and receives updates from a user table through a subscription. It can be a source for updating the user table or read-only target tables.

replication. The process of taking changes that are stored in the database log or journal at the source server and applying them to the target server.

replication administrator. The user responsible for defining replication sources and subscriptions. This user can also run the Capture and Apply programs.

replication source. A database table 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*.

replication subscription. (1) A specification for copying changed data from replication sources to target tables at a specified time and frequency, with the option of enhancing data. It defines all of the information that is required by the Apply program to copy data. (2) (For MS Jet) The process of requesting an automatic refresh or update of a set of one or more members. This is done by using the DataJoiner replication administration tool. The result is two rows into the ASN.IBMSNAP_SUBS_SET table, one or more rows into the ASN.IBMSNAP_SUBS_MEMBR and ASN_IBMSNAP_SUBS_COLS tables. These tables are located at the control server.

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.

S

secondary log. A set of one or more log files used to record changes to a database. Storage for these files is allocated as needed when the primary log is full.

SMS table space. System managed space table space.

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 target table that is used as the source for updating data to multiple target tables during a full refresh.

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* and *enhance conflict detection*.

subscription. See replication subscription.

subscription set. The specification of a group of source tables, target tables, and the control information that governs the replication of changed data. Updates are committed in the same transaction.

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.

T

table space. An abstraction of a collection of containers into which database objects are stored. A table space provides a level of indirection between a database and the tables stored within the database. A table space:

- Has space on media storage devices assigned to it.
- Has tables created within it. These tables will consume space in the containers that belong to the table space. The data, index, long field, and LOB portions of a table can be stored in the same table space, or can be individually broken out into separate table spaces.

See also system managed space (SMS) table space, database managed space (DMS) table space, long table space, regular table space, temporary table space.

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.

temporary table space. A table space that can store only temporary tables.

trace table. A table that contains a high-level record of the execution of the Capture program.

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

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 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*.

update replication cycle.

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, consistent change data tables, and replicas.

V

view. A logical table that consists of data that is generated by a query. Contrast with *base table*.

Visual Explain. A tool that lets database administrators and application programmers use a graphical interface to display and analyze detailed information on the access plan of a given SQL statement. The tasks provided by this tool can be accessed from the Control Center.

W

warm start. A restart 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 position in a DBMS log for later reference during warm start.

Index

A

activating and deactivating subscriptions 122
active log size, storage requirements 45
administration
 authorization requirements 51
 high-level steps 83
 overview 9, 83
after-image columns
 description 87
 partitioned key support values 97
aggregate tables, description 53
alert generation (MVS) 133
Apply program
 Apply qualifier
 defining 105
 description 17
 authorization requirements 51
 bandwidth impact 48
 capacity requirements 46
 configuring, overview 26
 connectivity requirements 47
 control tables 18
 data blocking 44
 dropping V1 tables 285
 error recovery 354
 for AIX
 binding the package 198
 configuring 198
 files generated 212
 forcing a full refresh 209
 gap messages 209
 operating 197, 210
 parameter definitions 211
 scheduling when to start 213
 setting up 197
 starting 210
 stopping 213
 troubleshooting information 219
 using 197
 for HP-UX
 binding the package 198
 configuring 198
 files generated 212
 forcing a full refresh 209
 gap messages 209
 operating 197, 210
 parameter definitions 211

Apply program (*continued*)
 for HP-UX (*continued*)
 scheduling when to start 213
 setting up 197
 starting 210
 stopping 213
 troubleshooting information 219
 using 197
 for MVS
 forcing a full refresh 139
 gap messages 139
 operating 129, 139
 parameter definitions 140
 scheduling when to start 142
 setting up 129
 starting 140
 stopping 142
 troubleshooting information 147
 using 129
 for OS/2
 binding the package 176
 configuring 176
 files generated 187
 forcing a full refresh 184
 gap messages 184
 operating 175, 185
 parameter definitions 186
 scheduling when to start 188
 setting up 175
 starting 186
 stopping 188
 troubleshooting information 193
 using 175
 for Solaris
 binding the package 198
 configuring 198
 files generated 212
 forcing a full refresh 209
 gap messages 209
 operating 197, 210
 parameter definitions 211
 scheduling when to start 213
 setting up 197
 starting 210
 stopping 213
 troubleshooting information 219
 using 197

- Apply program (*continued*)
 - for UnixWare 7
 - binding the package 198
 - configuring 198
 - files generated 212
 - forcing a full refresh 209
 - gap messages 209
 - operating 197, 210
 - parameter definitions 211
 - scheduling when to start 213
 - setting up 197
 - starting 210
 - stopping 213
 - troubleshooting information 219
 - using 197
 - for Windows NT and Windows 95
 - binding the package 27, 152
 - configuring 152
 - files generated 165
 - forcing a full refresh 163
 - gap messages 163
 - operating 151, 163
 - parameter definitions 164
 - scheduling when to start 166
 - setting up 151
 - starting 28, 164
 - stopping 28, 166
 - troubleshooting information 171
 - using 151
 - forward recovery 143, 168, 190, 215
 - full versus differential refresh 18
 - gap detection 77
 - influence on throughput capacity 50
 - introduction 15
 - loading large copies 144, 169, 190, 216
 - loading user copies 144, 169, 190, 216
 - log file 356
 - messages 377
 - migrating 280
 - mini-cycles 44
 - minimizing contention 76
 - naming conventions xvii
 - problem determination 354
 - processing cycle 16, 69
 - processor requirements 46
 - push vs. pull configuration 48
 - replicating changes 16
 - run-time processing statements 56
 - sample JCL 401
 - scheduling events 142, 167, 189, 214
- Apply program (*continued*)
 - scheduling processing cycles 69
 - selection order of source table 17
 - SQL application, as an 48
 - subsetting target columns and rows 54
 - trace file 355
 - user ID 51
- Apply qualifier
 - description 17
 - Microsoft Jet 261
 - specifying 105
- Apply spill file
 - data blocking 44
 - storage requirements 43
- Apply trail tables
 - CREATE TABLE statement 322
 - description 18, 319
 - problem determination 354
- applying changes to target tables, overview 16
- ARCHIVE LOG command 146
- archived data restrictions 67
- ASNCOPY command 255
- ASNDIAL environment variable 251
- ASNDONE
 - for Apply for MVS 143
 - for Apply for OS/2 190
 - for Apply for UNIX 215
 - for Apply for Windows NT and Windows 95 168
 - update-anywhere replication 74
- ASNHANGUP environment variable 251
- ASNJDONE user exit 265
- ASNJET command 263
- ASNJSTOP command 264
- ASNLOAD
 - and Apply for UNIX 211
 - error handling 168, 190, 215
 - files generated
 - for Apply for OS/2 189
 - for Apply for UNIX 215
 - for Apply for Windows for NT and Windows 95 168
 - refreshing point-in-time tables 167, 189, 214
- ASNLMIG command 274
- ASNMOBIL command 256
- audit trails, scenario for building 35
- auditing techniques 58
- authentication
 - end-user for UNIX 199
 - end-user for Windows NT and Windows 95 27, 153

- authorization and security requirements 51
- auto-registration
 - description 12
 - internal CCD tables 64

B

- bandwidth impact, network requirements 48
- base aggregate target tables
 - CREATE TABLE statement 339
 - defining 108
 - description 11, 54, 338
- before-image columns
 - and change aggregate tables 112
 - data transformation 56, 58
 - description 87
 - NULL value 88
 - partitioned key support values 97
 - restrictions 68
 - specifying 96
 - usage 87
- binary data types 68
- binding non-DB2 Universal Database RDBMs 85
- binding the package
 - Apply for OS/2 176, 254
 - Apply for UNIX 198
 - Apply for Windows NT and Windows 95 152
 - Capture for OS/2 176
 - Capture for UNIX 198
 - Capture for Windows NT and Windows 95 26, 152
 - Windows NT and Windows 95 for the mobile client 254
- blobs (binary large objects) 68
- blocking factor for subscription cycle, defining 116
- BUILDDDB command 275
- building audit trails scenario 35

C

- CALL procedures
 - defining for the replication subscription 117
 - run-time processing statements 56
- capacity requirements
 - Apply program 46
 - Capture program 46
 - Control Center 46
 - planning 46
- Capture enqueue tables
 - CREATE TABLE statement 299
 - description 299

- Capture program
 - alert generation (MVS) 133
 - authorization requirements 51
 - bandwidth impact 48
 - capacity requirements 46
 - Capture user ID 51
 - capturing changes 13
 - configuring, overview 26
 - connectivity requirements 47
 - control tables 14
 - dropping V1 tables 286
 - error recovery 354
 - Capture for MVS 132
 - Capture for VM 237
 - Capture for VSE 225
 - errors 356
 - for AIX
 - authorization for running 202
 - binding the package 198
 - cold start parameter 204
 - cold start, automatic 209
 - cold start, preventing 209
 - configuring 197, 202
 - log sequence number, providing 208
 - operating 197, 202
 - performance options 201
 - post-installation tasks 202
 - preparing for roll-forward recovery 197
 - pruning 208
 - reinitializing 207
 - restrictions 202
 - resuming 206
 - scheduling when to start 204
 - setting up 197
 - starting 203
 - stopping 205
 - suspending 206
 - troubleshooting information 217
 - tuning parameters 201
 - using 197
 - warm start parameter 204, 209
 - warm start, forcing 209
 - for HP-UX
 - authorization for running 202
 - binding the package 198
 - cold start parameter 204
 - cold start, automatic 209
 - cold start, preventing 209
 - configuring 197, 202
 - log sequence number, providing 208
 - operating 197, 202

- Capture program (*continued*)
 - for HP-UX (*continued*)
 - performance options 201
 - post-installation tasks 202
 - preparing for roll-forward recovery 197
 - pruning 208
 - reinitializing 207
 - restrictions 202
 - resuming 206
 - scheduling when to start 204
 - setting up 197
 - starting 203
 - stopping 205
 - suspending 206
 - troubleshooting information 217
 - tuning parameters 201
 - using 197
 - warm start parameter 204, 209
 - warm start, forcing 209
 - for MVS
 - authorization for running 132
 - cold start parameter 134
 - cold start, automatic 139
 - cold start, preventing 139
 - configuring 133
 - DB2 for MVS 3.1, active log size 45
 - error recovery 132
 - in a data-sharing environment 13
 - log sequence number, providing 138
 - operating 129, 133
 - performance options 130
 - post-installation tasks 133
 - pruning 137
 - reinitializing 137
 - restrictions 131
 - resuming 137
 - scheduling when to start 135
 - setting up 129
 - starting 134
 - stopping 136
 - suspending 136
 - troubleshooting information 145
 - tuning parameters 130
 - unsupported statements 132
 - using 129
 - warm start parameter 134, 138
 - warm start, forcing 139
 - for OS/2
 - authorization for running 178
 - binding the package 176
 - cold start parameter 179

- Capture program (*continued*)
 - for OS/2 (*continued*)
 - cold start, automatic 185
 - cold start, preventing 185
 - configuring 175, 178
 - log sequence number, providing 184
 - operating 175, 178
 - performance options 177
 - post-installation tasks 178
 - preparing for roll-forward recovery 175
 - pruning 183
 - reinitializing 182
 - restrictions 178
 - resuming 182
 - scheduling when to start 180
 - setting up 175
 - starting 179
 - stopping 180
 - suspending 181
 - troubleshooting information 192
 - tuning parameters 177
 - using 175
 - warm start parameter 179, 184
 - warm start, forcing 185
 - for Solaris
 - authorization for running 202
 - binding the package 198
 - cold start parameter 204
 - cold start, automatic 209
 - cold start, preventing 209
 - configuring 197, 202
 - log sequence number, providing 208
 - operating 197, 202
 - performance options 201
 - post-installation tasks 202
 - preparing for roll-forward recovery 197
 - pruning 208
 - reinitializing 207
 - restrictions 202
 - resuming 206
 - scheduling when to start 204
 - setting up 197
 - starting 203
 - stopping 205
 - suspending 206
 - troubleshooting information 217
 - tuning parameters 201
 - using 197
 - warm start parameter 204, 209
 - warm start, forcing 209

- Capture program (*continued*)
 - for UnixWare 7
 - authorization for running 202
 - binding the package 198
 - cold start parameter 204
 - cold start, automatic 209
 - cold start, preventing 209
 - configuring 197, 202
 - log sequence number, providing 208
 - operating 197, 202
 - performance options 201
 - post-installation tasks 202
 - preparing for roll-forward recovery 197
 - pruning 208
 - reinitializing 207
 - restrictions 202
 - resuming 206
 - scheduling when to start 204
 - setting up 197
 - starting 203
 - stopping 205
 - suspending 206
 - troubleshooting information 217
 - tuning parameters 201
 - using 197
 - warm start parameter 204, 209
 - warm start, forcing 209
 - for VM
 - active log size 45
 - authorization for running 237
 - cold start parameter 239
 - cold start, automatic 243
 - cold start, preventing 244
 - configuring 238
 - error recovery 237
 - log sequence number, providing 243
 - operating 235, 238
 - performance options 235
 - post-installation tasks 238
 - pruning 242
 - reinitializing 242
 - restrictions 236
 - resuming 242
 - setting up 235
 - starting 239
 - stopping 240
 - suspending 241
 - troubleshooting information 244
 - tuning parameters 235
 - using 235
 - warm start parameter 239, 243

- Capture program (*continued*)
 - for VM (*continued*)
 - warm start, forcing 244
 - for VSE
 - active log size 45
 - authorization for running 225
 - cold start parameter 227
 - cold start, automatic 232
 - cold start, preventing 232
 - configuring 226
 - error recovery 225
 - log sequence number, providing 231
 - operating 223, 225
 - performance options 223
 - post-installation tasks 226
 - pruning 231
 - reinitializing 230
 - restrictions 224
 - resuming 230
 - setting up 223
 - starting 226
 - stopping 228
 - suspending 229
 - troubleshooting information 232
 - tuning parameters 223
 - using 223
 - warm start 231
 - warm start parameter 227
 - for Windows NT and Windows 95
 - authorization for running 156
 - binding the package 152
 - cold start parameter 158
 - cold start, automatic 163
 - cold start, preventing 163
 - configuring 151, 156
 - log sequence number, providing 162
 - operating 151, 156
 - performance options 155
 - post-installation tasks 156
 - preparing for roll-forward recovery 151
 - pruning 161
 - reinitializing 161
 - restrictions 156
 - resuming 160
 - scheduling when to start 158
 - setting up 151
 - starting 28, 157
 - stopping 28, 158
 - suspending 160
 - troubleshooting information 170
 - tuning parameters 155

- Capture program (*continued*)
 - for Windows NT and Windows 95 (*continued*)
 - using 151
 - warm start parameter 158, 162
 - warm start, forcing 163
 - gap detection 77
 - identifying external data sources 65
 - introduction 13
 - log file 357
 - messages 361
 - migrating 277
 - minimizing contention 76
 - naming conventions xvii
 - problem determination 356
 - processor requirements 46
 - pruning CCD tables 59
 - sample JCL 401
 - Service Control Manager 153
 - staging data 59
 - stopping with a trigger (MVS) 136
 - stopping with a trigger (OS/2) 181
 - stopping with a trigger (UNIX) 205
 - stopping with a trigger (VM) 241
 - stopping with a trigger (VSE) 229
 - stopping with a trigger (Windows NT and Windows 95) 159
 - storage dump (MVS) 133
 - storage dump (VM) 237
 - storage dump (VSE) 225
 - trace buffer (MVS) 133
 - trace buffer (VM) 237
 - trace buffer (VSE) 225
 - trace file 357
 - trace output (MVS) 133
 - trace output (VM) 237
 - trace output (VSE) 225
 - usage 13
- capturing changes, overview 13
- CCD (consistent change data) tables
 - auto-registration 12, 64
 - condensed, complete 12
 - condensed, noncomplete 12
 - CREATE TABLE statement 343
 - data warehouse 66
 - defining 108
 - description 12, 54, 59, 340
 - external data sources 65, 97
 - external, refreshing 65
 - internal 62, 65
 - local 62
- CCD (consistent change data) tables (*continued*)
 - noncondensed, complete 12
 - noncondensed, noncomplete 12
 - remote 62
 - restrictions 75
 - staging changed data 59
 - updating condensed copies 340
- CD (change data) tables
 - and consistent change data 59
 - and the Capture program 13
 - CREATE TABLE statement 301
 - description 15, 59, 299
 - pruning 300
- change aggregate target tables
 - CREATE TABLE statement 340
 - defining 108
 - description 12, 54, 339
- changed data retention, planning 58
- changing
 - replication sources 100
 - replication subscriptions 123
- classes, education 7
- CLEANUP command 285
- CLOBs (character large objects) 68
- cloning subscriptions 123
- cold start
 - Capture for MVS 134
 - Capture for OS/2 179
 - Capture for UNIX 204
 - Capture for VM 239
 - Capture for VSE 227
 - Capture for Windows NT and Windows 95 158
 - gaps 77
 - low lag limit
 - Capture for MVS 130
 - Capture for OS/2 177
 - Capture for UNIX 201
 - Capture for VM 235
 - Capture for VSE 223
 - Capture for Windows NT and Windows 95 155
- preventing
 - Capture for MVS 139
 - Capture for OS/2 185
 - Capture for UNIX 209
 - Capture for VM 244
 - Capture for VSE 232
 - Capture for Windows NT and Windows 95 163
- switching to automatically
 - Capture for MVS 139
 - Capture for OS/2 185
 - Capture for UNIX 209

- cold start (*continued*)
 - switching to automatically (*continued*)
 - Capture for VM 243
 - Capture for VSE 232
 - Capture for Windows NT and Windows 95 163
- columns
 - before-image and after-image 56, 87
 - changing the definition 110
 - computed
 - data transformation 56
 - defining 110
 - creating new in target table 110
 - defining in target table 108
 - fragmenting 54
 - removing from target table 110
 - renaming
 - data transformation 56
 - defining 110
 - restrictions on names 68
 - specifying the primary key 109
 - subsetting 54, 108
- commit interval
 - Capture for MVS 130
 - Capture for OS/2 177
 - Capture for UNIX 201
 - Capture for VM 235
 - Capture for VSE 223
 - Capture for Windows NT and Windows 95 155
- complete
 - CCD tables 17
 - data attribute 17
- condensed
 - CCD tables 61, 340
 - copies, updating 340
 - staging tables 61
- configuring
 - Apply program 25
 - Capture program 25
 - client for mobile replication 253
 - connectivity requirements 47
 - Control Center for non-DB2 Universal Database RDBMs 85
- conflict detection
 - data integrity 72
 - description 74
 - levels 74
 - lost or rejected transactions 74
 - planning 72
 - specifying 97
- conflict tables, description 267, 347
- conflicts
 - lost or rejected transactions 74
 - recovering from 74
- consulting and services 7
- contention between the Capture and Apply programs
 - locking 76
 - no ISOLATION (UR) support 76
- continuous timing
 - planning 69
 - setting for the subscription 115
- Control Center
 - authorization requirements 51
 - bandwidth impact 48
 - capacity requirements 46
 - configuring for non-DB2 Universal Database RDBMs 85
 - connectivity requirements 47
 - control tables created by 12
 - introduction 9
 - objects 10
 - processor requirements 46
 - replication objects 84
 - setting preferences 85
- control server
 - choosing location 17
 - defining for mobile replication 255
 - Microsoft Jet 262
 - specifying 106
- control tables
 - Apply trail 18, 319
 - at the control server 319
 - at the source server 299
 - at the target server 337
 - base aggregate target 338
 - Capture enqueue 299
 - CCD target 340
 - CD 15, 299
 - change aggregate 339
 - conflict 267, 347
 - created by Control Center 12
 - critical section 14, 301
 - customizing control file 88
 - DataPropagator for Microsoft Jet 266
 - error information 267, 347
 - error messages 267, 348
 - error side information 267, 349
 - estimating storage requirements 43
 - key string 267, 349
 - migration 287

- control tables (*continued*)
 - planning for customization 52
 - point-in-time target 343
 - pruning control 15, 302
 - quick reference
 - at a glance 441
 - control server 296
 - source server 295
 - target server 297
 - register 14, 306
 - replica target 344
 - row-replica target list 267, 336
 - subscription columns 18, 322
 - subscription events 18, 324
 - subscription schema changes 267, 335
 - subscription set 18, 326
 - subscription statements 18, 330
 - subscription targets member 18, 332
 - synchronization generations 267, 350
 - trace 15, 312
 - tuning parameters 15, 313
 - UOW 15, 314
 - used by Apply program 18
 - used by Capture program 14
 - user copy target 345
 - warm start 15, 317
 - warm start for VSE and VM 318
- conventions
 - highlighting xvii
 - naming xvii
- copies
 - loading large 144, 169, 190, 216
 - point in time 144, 169, 190, 216
 - types of refresh 18
- copy table types 11
- correcting a gap 77
- CREATE TABLE statement
 - Apply trail tables 322
 - base aggregate target tables 339
 - Capture enqueue tables 299
 - CCD tables 343
 - CD tables 301
 - change aggregate target tables 340
 - critical section tables 302
 - error information tables 348
 - error messages tables 349
 - error side information tables 349
 - key string tables 350
 - point-in-time target tables 344
 - pruning control tables 306

- CREATE TABLE statement (*continued*)
 - register tables 312
 - replica target tables 345
 - row-replica target list tables 337
 - subscription columns tables 324
 - subscription events tables 326
 - subscription schema changes tables 336
 - subscription set tables 330
 - subscription statements tables 332
 - subscription targets member tables 335
 - synchronization generations tables 351
 - trace tables 313
 - tuning parameters tables 314
 - UOW tables 315
 - user copy target tables 347
 - warm start tables 318
 - warm start tables for Capture for VSE and VM 319
- critical section tables
 - CREATE TABLE statement 302
 - description 14, 301
- custom, defining replication sources 95
- customizing
 - control tables 88
 - planning requirements 52
 - SQL files 90

D

- data archive scenario 35
- data blocking 44
- DATA CAPTURE CHANGES 42
- data capture, specifying 96
- data compression restriction 67
- data consistency
 - planning 70
 - requirements 70
- data consolidation
 - and distribution scenario 34
 - data transformation 57
- data currency
 - effect on CPU usage 46
 - planning 69
 - requirements 69
- data encryption restrictions 68
- data enhancements
 - before-image columns 56
 - column renaming 56
 - column subsetting 54
 - computed columns 56
 - data consolidation 57

- data enhancements *(continued)*
 - for replication targets 53
 - noncondensed CCD tables 54
 - outer join of source tables 57
 - point-in-time target tables 53
 - poorly structured data 57
 - row subsetting 55
 - run-time processing 56
 - triggers 56
 - updateable replica target tables 54
 - user copy target tables 53
 - data integrity
 - DataPropagator for Microsoft Jet 261
 - resolving gaps 77
 - data replication solution 3
 - data sharing
 - and Capture for MVS 13
 - data transformation requirements 67
 - data transformation
 - advanced transformation 57
 - auditing requirements 58
 - basic enhancements 53
 - condensed CCD tables 54
 - data restrictions 67
 - data sharing considerations 67
 - Extended Enterprise Edition considerations 67
 - mobile replication 58
 - planning 53
 - replication logical partitioning key 66
 - data types, restrictions 68
 - data warehouse 66
 - DataPropagator for Microsoft Jet
 - control tables 266
 - operating 262
 - overview 259
 - starting 263
 - stopping 264
 - troubleshooting 265
 - DataPropagator NonRelational 6
 - DB2
 - ARCHIVE LOG command 146
 - control intervals 146
 - log 13, 146
 - DB2 DataJoiner
 - for legacy data 6
 - for multivendor replication 6
 - DB2 library
 - books 433
 - Information Center 439
 - language identifier for books 436
 - DB2 library *(continued)*
 - late breaking information 437
 - online help 432
 - ordering printed books 439
 - printing PostScript books 438
 - searching online books 438
 - setting up document server 437
 - SmartGuides 431
 - structure of 431
 - viewing online books 437
 - DB2 Tools Settings notebook 85
 - DB2INSTANCE
 - when starting Capture for OS/2 179
 - when starting Capture for UNIX 203
 - when starting Capture for Windows NT and Windows 95 157
 - DBCLOBs (double-byte character large objects) 68
 - deactivating and activating subscriptions 122
 - decision support system scenario 33
 - deferring replication requests 97, 106
 - defining
 - external CCD tables as sources 97
 - replication source joins 98
 - replication sources
 - example 23
 - steps 94, 95
 - subscriptions
 - advanced steps 107
 - basic steps 103
 - example 24
 - Design Master 262
 - detecting a gap 77
 - diagnosing errors 353
 - differential refresh 18
 - distinct data type 68
 - distributed database system scenario 34
 - dropping V1 tables 285
 - dummy WHERE clause 112
- ## E
- EDITPROCs 68
 - education
 - custom classes 7
 - IBM Global Campus URL 7
 - enhanced conflict detection 74
 - enhancements, IBM Replication 445
 - enhancing data, planning 53
 - environment variables
 - ASNDIAL
 - description 251

- environment variables (*continued*)
 - ASNDIAL (*continued*)
 - specifying for OS/2 251
 - specifying for Windows 95 252
 - specifying for Windows NT 252
 - ASNHANGUP
 - description 251
 - specifying for OS/2 251
 - specifying for Windows 95 252
 - specifying for Windows NT 252
- error information tables
 - CREATE TABLE statement 348
 - description 267, 347
- error messages tables
 - CREATE TABLE statement 349
 - description 267, 348
- error recovery
 - Capture for MVS 132
 - Capture for VM 237
 - Capture for VSE 225
- error side information tables
 - CREATE TABLE statement 349
 - description 267, 349
- errors, diagnosing 353
- event table
 - Apply for MVS 142
 - Apply for OS/2 189
 - Apply for UNIX 214
 - Apply for Windows NT and Windows 95 167
 - description 324
 - populating 69, 106
- event-based timing
 - description 69
 - planning 69
 - setting for the subscription 116
- examples
 - sample JCL 401
 - SQL for columns 110
 - WHERE clauses 113
- Extended Enterprise Edition, data transformation requirements 67
- extending IBM Replication scenario 37
- external CCD tables
 - defining as sources 97
 - description 65
 - maintaining 72
 - refreshing 65
- external data sources 65

F

- FALLBACK command 283, 284
- fast path xix
- FIELDPROCs 68
- flat file data 6
- forward recovery
 - Apply for MVS 143
 - Apply for OS/2 190
 - Apply for UNIX 215
 - Apply for Windows NT and Windows 95 168
- fragmentation
 - horizontal 55
 - vertical 54
- fragmenting target tables 54
- full refresh
 - description 18
 - forcing
 - Apply for MVS 148
 - Apply for OS/2 194
 - Apply for UNIX 220
 - Apply for Windows NT and Windows 95 173
 - specifying 96
 - suppressing
 - Apply for MVS 148
 - Apply for OS/2 194
 - Apply for UNIX 219
 - Apply for Windows NT and Windows 95 172

G

- gap detection
 - description 77
 - resolving 77
 - verifying 77
- gap messages
 - for MVS 139
 - for OS/2 184
 - for UNIX 209
 - for Windows NT and Windows 95 163
- global lock 76
- GROUP BY clause 112

H

- hardware requirements 40, 250
- highlighting conventions xvii
- history data 58
- horizontal subsetting
 - description 55

horizontal subsetting (*continued*)
steps 111
hot-site recovery, description 38

I

IBM Education resources
custom classes 7
IBM Global Campus URL 7
IBM Replication
education 7
enhancements 445
introduction 4
messages 361
planning 31
services and consulting 7
tools 3, 9
identifiers, ordinary and delimited 92
identifying external data sources 65
improved application availability scenario 35
improved network load scenario 34
IMS and VSAM data 6
installation
Capture and Apply for MVS 129
Capture for VM 235
Capture for VSE 223
considerations for UNIX components 199
independent migration 273
services and consulting 7
internal CCD tables
auto-registration 64
description 62
reducing network load 65
interval timing, planning 69
introduction to IBM Replication
Apply program 15
Capture program 13
Control Center 9
overview 4
ISOLATION (UR)
contention problems 76
platforms that don't support 76

J

JCL, sample 401
joins
defining as sources 98
description 55, 57

K

key string tables
CREATE TABLE statement 350
description 267, 349

L

lag limit
Capture for MVS 130
Capture for OS/2 177
Capture for UNIX 201
Capture for VM 235
Capture for VSE 223
Capture for Windows NT and Windows 95 155
large copies, loading 144, 169, 190, 216
large replication jobs 44
legacy data sources 6
loading large copies 144, 169, 190, 216
local CCD tables 62
locking, how it affects contention 76
log file
Apply program 356
Capture program 357
log records
archived before captured 45
determining the size 45
log sequence number
Capture for MVS 138
Capture for OS/2 184
Capture for UNIX 208
Capture for VM 243
Capture for VSE 231
Capture for Windows NT and Windows 95 162
logging requirements, DBMS logging 42
logical partitioning key support
description 66
specifying 96
logical recovery scenario 37
LONG VARGRAPHIC data type 68
lost transactions 74
Lotus Notes Pump 6
LU sessions 201

M

MAX_SYNCH_MINUTES, description 44
messages
Apply program 377
Capture program 361, 377

- messages (*continued*)
 - migration 391
- Microsoft Jet 259
- MIGRATE command 278, 280
- migration
 - analysis 272
 - analyzing data 276
 - building 275
 - collection 271, 275
 - control tables 287
 - deleting unused tables and views 285
 - falling back to Version 1
 - Apply program 283
 - Capture program 284
 - messages 391
 - migrating data
 - Apply program 280
 - Capture program 277
 - prerequisites 272
 - process 271
 - services and consulting 7
 - starting 274
- mini-cycles
 - Apply program 44
 - defining for the subscription 116
- minimizing contention 76
- mobile replication
 - configuring
 - client 253
 - control server 255
 - data transformation requirements 58
 - environment variables 251
 - for OS/2
 - binding the package 254
 - configuring 254
 - for Windows NT and Windows 95
 - binding the package 254
 - configuring 253
 - Microsoft Jet 259
 - overview 249
 - planning 250
 - processing cycle 255
 - restrictions 250
 - scenario 35
- mobile replication enabler
 - starting from the command line 255
 - starting from the graphical interface 256
- MS Jet
 - mobile replication scenario 36
 - replication 259

- multivendor data 6

N

- network requirements
 - bandwidth impact 48
 - connectivity 47
 - planning 47
 - throughput capacity 50
- no conflict detection 74
- non-DB2 data 6
- non-DB2 Universal Database RDBMs, configuring the Control Center 85

O

- ODBC-accessible data 6
- on-demand replication, data transformation 58
- ordinary and delimited identifiers 92
- outer join of source tables 57

P

- parameter definitions
 - Apply for MVS 140
 - Apply for OS/2 186
 - Apply for UNIX 211
 - Apply for Windows NT and Windows 95 164
- password files, authorization requirements 51
- performance options
 - Capture for MVS 130
 - Capture for OS/2 177
 - Capture for UNIX 201
 - Capture for VM 235
 - Capture for VSE 223
 - Capture for Windows NT and Windows 95 155
- planning
 - active log size, storage requirements 45
 - application level 53
 - capacity requirements 46
 - contention 76
 - customization requirements 52
 - data consistency 70
 - data currency 69
 - data transformation 53
 - hardware requirements 40
 - mobile replication 250
 - network requirements 47
 - processor requirements 46
 - replication products 39

- planning (*continued*)
 - security and authorization requirements 51
 - services and consulting 7
 - software requirements 40, 42
 - storage requirements 42
 - system level 39
 - update anywhere
 - CCD tables 75
 - conflict detection 74
 - overview 72
 - recommended usage 75
 - recovering from conflicts 74
 - restrictions 75
- point-in-time target tables
 - CREATE TABLE statement 344
 - defining 108
 - description 11, 53, 343
- poorly structured data, data transformation 57
- potential replication scenarios 36
- predicates, defining for target tables 113
- preferences, setting 85
- PREPARE command 276
- primary key
 - errors when updating 55
 - logical partitioning 66
- problem determination 353
- processor requirements, planning 46
- products, replication 39
- programming interface information 451
- PRTCT 201
- prune interval
 - Capture for MVS 130
 - Capture for OS/2 177
 - Capture for UNIX 201
 - Capture for VM 236
 - Capture for VSE 224
 - Capture for Windows NT and Windows 95 155
- pruning
 - Capture for MVS 137
 - Capture for OS/2 183
 - Capture for UNIX 208
 - Capture for VM 242
 - Capture for VSE 231
 - Capture for Windows NT and Windows 95 161
 - CCD tables 59
 - CD tables 300
 - condensed CCD tables 61
 - UOW tables 315
- pruning control tables
 - CREATE TABLE statement 306

- pruning control tables (*continued*)
 - description 15, 302
- push and pull Apply program configuration
 - choosing a configuration 50
 - data transformation 58
 - description 48

Q

- questions, problem determination 358
- quick, defining replication sources 94

R

- recommended replication scenarios 33
- referential integrity 70
- refresh copying
 - description 18
 - external CCD tables 65
 - specifying 96
- refresh, suppressing
 - Apply for MVS 148
 - Apply for OS/2 194
 - Apply for UNIX 219
 - Apply for Windows NT and Windows 95 172
- register tables
 - CREATE TABLE statement 312
 - description 14, 306
- reinitializing
 - Capture for MVS 137
 - Capture for OS/2 182
 - Capture for UNIX 207
 - Capture for VM 242
 - Capture for VSE 230
 - Capture for Windows NT and Windows 95 161
- rejected transactions 74
- relative timing
 - description 69
 - planning 69
 - setting for the subscription 115
- remote CCD tables 62
- removing
 - replication sources 101
 - replication subscriptions 124
- replica target tables
 - auto-registration 12
 - CREATE TABLE statement 345
 - defining 108
 - description 12, 54, 344
 - update-anywhere replication 72

- replication
 - administration 83
 - before-image columns, specifying 96
 - conflict detection, specifying 97
 - DataPropagator NonRelational 6
 - DB2 DataJoiner 6
 - deferring 97, 106
 - IMS and VSAM 6
 - introduction
 - Apply program 15
 - Capture program 13
 - Control Center 9
 - legacy data 6
 - logical partitioning key support, specifying 96
 - Lotus Notes 6
 - messages 361
 - mobile 249
 - multivendor 6
 - non-DB2 6
 - objects, description 10
 - potential usage scenarios 36
 - products, description 39
 - recommended usage scenarios 33
 - scenarios not recommended 37
 - solution 3
 - sources 9
 - subscriptions 10
 - supported environments 4
 - transaction based 65
 - transaction consistent 65
 - update anywhere
 - ASNDONE user exit 74
 - conflict detection 74
 - description 72
 - specifying 97
- replication sources
 - changing 100
 - conflict detection 97
 - defining
 - custom 95
 - example 23
 - external CCD 97
 - for update anywhere 125
 - joins 98
 - overview 93
 - with default values 94
 - description 9
 - example of creating 21
 - outer join 57
 - overview 93
- replication sources (*continued*)
 - removing 101
 - selecting 17
 - specifying before-image columns 96
 - specifying data capture 96
 - specifying refresh copying 96
 - specifying replication logical partitioning key support 96
 - specifying update-anywhere replication 97
 - types 93
 - views 55
- replication subscriptions
 - activating and deactivating 122
 - and the Apply qualifier 17
 - changing 123
 - cloning 123
 - completing the definition 106
 - continuous 69
 - data consistency 70
 - defining
 - advanced steps 107
 - basic steps 103
 - columns 108
 - control server 106
 - example 24
 - for update anywhere 125
 - mini-cycles 116
 - overview 103
 - rows 111
 - run-time processing 117
 - target table structure 106, 108
 - target table type 107
 - description 10
 - event-based timing 69
 - example of creating 21
 - interval timing 69
 - mobile replication 257
 - naming 105
 - overview 103
 - referential integrity 70
 - relative timing 69
 - removing 124
 - run-time processing statements 56
 - selecting user-defined target tables 105
 - selection order for source 17
 - setting the timing 114
 - specifying the Apply qualifier 105
 - specifying the target server 105
 - specifying the target table name 105
 - timing 69

- replication subscriptions (*continued*)
 - triggering events 106
 - views 57
- resolving a gap 77
- restrictions
 - data types 68
 - limits on column names 68
 - mobile replication 250
 - update anywhere 75
- resuming
 - Capture for MVS 137
 - Capture for OS/2 182
 - Capture for UNIX 206
 - Capture for VM 242
 - Capture for VSE 230
 - Capture for Windows NT and Windows 95 160
- retention limit
 - Capture for MVS 130
 - Capture for OS/2 177
 - Capture for UNIX 201
 - Capture for VM 235
 - Capture for VSE 223
 - Capture for Windows NT and Windows 95 155
- road map of this book xix
- row-replica target list tables
 - CREATE TABLE statement 337
 - description 262, 267, 336
- rows
 - defining in target table 111
 - defining the predicate 113
 - restrictions for subsetting 111
 - subsetting 55, 111
- run-time processing
 - data transformation 56
 - defining for the subscriptions 117

S

- sample JCL 401
- scenarios
 - building audit trails 35
 - creating a user copy target table 21
 - data archive 35
 - data consolidation and distribution 34
 - decision support system 33
 - distributed database system 34
 - extending IBM Replication 37
 - hot-site recovery, not recommended 38
 - improved application availability 35
 - improved network load 34

- scenarios (*continued*)
 - logical recovery 37
 - mobile replication 35
 - mobile replication with MS Jet 36
 - not recommended 37
 - potential customer implementations 36
 - problem determination 357
 - recommended customer implementations 33
 - synchronous replication, not recommended 37
 - update-anywhere replication 36
- scheduling
 - Apply for MVS 142
 - Apply for OS/2 188
 - Apply for UNIX 213
 - Apply for Windows NT and Windows 95 166
 - Capture for MVS 135
 - Capture for OS/2 180
 - Capture for UNIX 204
 - Capture for Windows NT and Windows 95 158
 - event table
 - for Apply for MVS 142
 - for Apply for OS/2 189
 - for Apply for UNIX 214
 - for Apply for Windows NT and Windows 95 167
 - event timing 69
 - relative timing 69
 - replication subscriptions 69
- SECACPT 201
- security and authorization requirements, planning 51
- selection order for source table 17
- Service Control Manager
 - Apply for Windows NT 153
 - Capture for Windows NT 153
- services and consulting 7
- setting up a user account for UNIX 197
- setting up document server 437
- setting up the Capture and Apply programs
 - for MVS 129
 - for OS/2 175
 - for UNIX 197
 - for Windows NT and Windows 95 151
- setting up the Capture program
 - for VM 235
 - for VSE 223
- soft errors 361
- software requirements 40, 42, 250
- solution, data replication 3
- source server
 - end-user authentication 27
 - Microsoft Jet 262

- sources
 - multivendor 6
 - selection order 17
 - views 55
- SQL
 - before and after 56
 - files, customizing 90
 - statements
 - defining for the replication subscription 117
 - run-time processing 56
- staging data
 - benefits of 61
 - data warehouse 66
 - description 59
- staging table
 - attributes 60
 - complete 61
 - condensed 61
 - defining 108
 - ways to use 60
- standard conflict detection 74
- starting
 - Apply for MVS 140
 - Apply for OS/2 186
 - Apply for UNIX 210
 - Apply for Windows NT and Windows 95 28, 164
 - Capture for MVS 134
 - Capture for OS/2 179
 - Capture for UNIX 203
 - Capture for VM 239
 - Capture for VSE 226
 - Capture for Windows NT and Windows 95 28, 157
 - DataPropagator for Microsoft Jet 263
 - migration program 274
- stopping
 - Apply for MVS 142
 - Apply for OS/2 188
 - Apply for UNIX 213
 - Apply for Windows NT and Windows 95 28, 166
 - Capture for MVS 136
 - Capture for OS/2 180
 - Capture for UNIX 205
 - Capture for VM 240
 - Capture for VSE 228
 - Capture for Windows NT and Windows 95 28, 158
 - DataPropagator for Microsoft Jet 264
- storage dump (MVS) 133
- storage dump (VM) 237
- storage dump (VSE) 225
- storage requirements
 - active log size 45
 - Apply spill file 43
 - data blocking for large volumes 44
 - DBMS logging 42
 - general considerations 46
 - new tables 43
 - planning 42
 - UOW table size 43
 - virtual memory on MVS 43
- subscription columns tables
 - CREATE TABLE statement 324
 - description 18, 322
- subscription events tables
 - CREATE TABLE statement 326
 - description 18, 324
- subscription schema changes tables
 - CREATE TABLE statement 336
 - description 267, 335
- subscription set tables
 - CREATE TABLE statement 330
 - description 18, 326
 - user load 144, 169, 190, 216
- subscription statements tables
 - CREATE TABLE statement 332
 - description 18, 330
- subscription targets member tables
 - CREATE TABLE statement 335
 - description 18, 332
- subscriptions
 - Microsoft Jet replication 262
 - replication 10
- subsetting
 - columns 54
 - horizontal 55
 - rows 55
 - source tables 54
 - vertical 54
- suppressing full refresh
 - Apply for MVS 148
 - Apply for OS/2 194
 - Apply for UNIX 219
 - Apply for Windows NT and Windows 95 172
- suspending
 - Capture for MVS 136
 - Capture for OS/2 181
 - Capture for UNIX 206
 - Capture for VM 241
 - Capture for VSE 229
 - Capture for Windows NT and Windows 95 160

- synchronization generations tables
 - CREATE TABLE statement 351
 - description 267, 350
- synchronous replication, description 37
- syntax diagrams, how to read xviii
- system planning 39

T

- table structures 295
- tables
 - structures 295
 - tuning parameters
 - Capture for MVS performance 130
 - Capture for OS/2 performance 177
 - Capture for UNIX performance 201
 - Capture for VM performance 235
 - Capture for VSE performance 223
 - Capture for Windows NT and Windows 95 performance 155
- target server
 - Microsoft Jet 262
 - specifying 105
- target tables
 - auto-registration 12
 - base aggregate 11, 54, 338
 - CCD
 - condensed, complete 12
 - condensed, noncomplete 12
 - description 59, 340
 - noncondensed, complete 12
 - noncondensed, noncomplete 12
 - change aggregate 12, 54, 339
 - columns, defining 108
 - defining for update anywhere 125
 - fragmenting 54
 - names, specifying 105
 - point-in-time 11, 343
 - replica 12, 344
 - rows, defining 111
 - specifying a type 107
 - storage requirements 43
 - structure, specifying 106, 108
 - subsetting columns 54
 - subsetting rows 55
 - table structures, quick reference 297
 - types 11
 - user copy 11, 345
 - user-defined 120
- targets
 - description 10
 - views 57
- terminology xvii, 261
- throughput capacity 50
- timing
 - continuous 69
 - defining for the subscription 114
 - event-based 69
 - interval 69
 - relative 69
 - replication subscriptions 69
- tools, IBM Replication 3, 9
- trace buffer (MVS) 133
- trace buffer (VM) 237
- trace buffer (VSE) 225
- trace file
 - Apply program 355
 - Capture program 357
- trace output (MVS) 133
- trace output (VM) 237
- trace output (VSE) 225
- trace tables
 - CREATE TABLE statement 313
 - description 15, 312
 - problem determination 357
- transaction identification 58
- transaction-based replication 65
- transaction-consistent replication 65
- transactions, lost or rejected 74
- triggers
 - data transformation 56
 - stopping the Capture program (MVS) 136
 - stopping the Capture program (OS/2) 181
 - stopping the Capture program (UNIX) 205
 - stopping the Capture program (VM) 241
 - stopping the Capture program (VSE) 229
 - stopping the Capture program (Windows NT and Windows 95) 159
- troubleshooting information
 - Apply for MVS 147
 - Apply for OS/2 193
 - Apply for UNIX 219
 - Apply for Windows NT and Windows 95 171
 - Capture for MVS 145
 - Capture for OS/2 192
 - Capture for UNIX 217
 - Capture for VM 244
 - Capture for VSE 232
 - Capture for Windows NT and Windows 95 170

- troubleshooting information (*continued*)
 - DataPropagator for Microsoft Jet 265
- tuning parameters
 - Capture for MVS 130
 - Capture for OS/2 177
 - Capture for UNIX 201
 - Capture for VM 235
 - Capture for VSE 223
 - Capture for Windows NT and Windows 95 155
- tuning parameters tables
 - CREATE TABLE statement 314
 - description 15, 313

U

- unit-of-work (UOW) tables
 - and the Capture program 13
 - CREATE TABLE statement 315
 - description 15, 314
 - pruning 315
- update-anywhere replication
 - ASNDONE user exit 74
 - CCD tables 75
 - conflict detection 74
 - defining sources and targets 125
 - planning 72
 - recommended usage
 - fragmentation by key 75
 - fragmentation by time 75
 - recovering from conflicts 74
 - restrictions 75
 - scenario 36
 - specifying for sources 97
- updated primary key columns 66
- updating condensed copies, with changed data 340
- user copy target tables
 - CREATE TABLE statement 347
 - defining 108
 - description 11, 53, 345
 - scenario for creating 21
- user exit program
 - ASNDIAL 251
 - ASNHANGUP 251
 - ASNJDONE 265
- user load of tables 144, 169, 190, 216
- user-defined target tables
 - CCD table 121
 - maintaining 121
 - specifying 105
 - subscriptions 120

- utility capture restrictions 68

V

- VALIDPROCs 68
- VERIFY 201
- verifying a gap 77
- verifying the replication request 97, 106
- vertical subsetting
 - description 54
 - steps 108
- views
 - defining as sources 98
 - description 55, 57
- VSAM data 6

W

- warehouse database 66
- warm start
 - Capture for MVS 134
 - Capture for OS/2 179
 - Capture for UNIX 204
 - Capture for VM 239
 - Capture for VSE 227
 - Capture for Windows NT and Windows 95 158
- forcing
 - Capture for MVS 139
 - Capture for OS/2 185
 - Capture for UNIX 209
 - Capture for VM 244
 - Capture for VSE 232
 - Capture for Windows NT and Windows 95 163
- switching to cold start
 - Capture for MVS 138
 - Capture for OS/2 184
 - Capture for UNIX 209
 - Capture for VM 243
 - Capture for VSE 231
 - Capture for Windows NT and Windows 95 162
- warm start tables
 - CREATE TABLE statement 318
 - description 15, 317
 - for Capture for VSE and VM
 - CREATE TABLE statement 319
 - description 318
- WHERE clause
 - dummy 112
 - examples 113
 - filtering rows 113

WHERE clause (*continued*)
restrictions 112



Printed in the United States of America
on recycled paper containing 10%
recovered post-consumer fiber.

S95H-0999-02

