DB2 Data Management Software

IBM

Draft Document for Review September 16, 2002 4:29 pm

SG24-6828-00

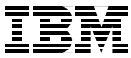
The Practical Guide to DB2 UDB Data Replication V8

- Covers iSeries, UNIX/AIX, Windows, and zSeries
- Practical examples
- Advanced scenarios

Lijun (June) Gu
Lloyd Budd
Aysegul Cayci
Colin Hendricks
Micks Purnell
Carol Rigdon

Redbooks

ibm.com/redbooks



International Technical Support Organization

IBM Data Replication V8

October 2002

Note: Before using this information and the product it supports, read the information in “Notices” on page xxiii.

First Edition (October 2002)

This edition applies to Version ???, Release ???, Modification ??? of ???insert-product-name??? (product number ????-???).

Contents

Figures xi

Tables xvii

Examples xix

Notices xxiii
Trademarks xxiv

Preface xxv
The team that wrote this redbook xxv
Become a published author xxvii
Comments welcome xxvii

Chapter 1. Introduction to DB2 Replication V8 xxix
1.1 Overview of the IBM replication solution xxx
1.2 Why use replication? xxxi
1.2.1 Distribution of data to other locations xxxi
1.2.2 Consolidation of data from remote systems xxxii
1.2.3 Bidirectional exchange of data xxxiii
1.2.4 Other requirements xxxv
1.3 DB2 V8 replication from 30,000 feet xxxvi
1.3.1 Administration xxxvi
1.3.2 Capture xxxvii
1.3.3 Apply xli
1.3.4 Alert Monitor xlvi
1.4 Putting the pieces together xlviii
1.4.1 Administration for all scenarios xlviii
1.4.2 Data distribution and data consolidation xlix
1.4.3 Bidirectional with a master (update anywhere) li
1.4.4 Bidirectional with no master (peer to peer) li
1.4.5 Alert Monitor configuration li
1.5 DB2 Replication V8 close up liii
1.5.1 Administration — defining a replication scenario liii
1.5.2 Operations — DB2 Capture and Apply lviii
1.5.3 Operations — Informix Capture and Apply lxiv
1.5.4 Administration and operations — Alert Monitor lxvii
1.6 What's new in DB2 Replication V8 lxix
1.6.1 Administration lxix

1.6.2 Capture lxix
1.6.3 Apply lxx
1.6.4 Monitor lxxi
1.6.5 Troubleshooting lxxii
1.7 The Redbook environment lxxiii

Chapter 2. Getting started with Replication Center lxxv
2.1 DB2 Replication Center's architecture lxxvii
2.2 Technical requirements for DB2 Replication Center lxxxvi
2.2.1 Hardware requirements. lxxxvi
2.2.2 Software Requirements. lxxxvi
2.2.3 Networking requirements lxxxviii
2.2.4 Requirements at replication servers lxxxix
2.2.5 Requirements for replication to/from non-DB2 servers xc
2.3 DB2 products needed to use Replication Center xciii
2.4 How to Get Replication Center xciv
2.4.1 How to get DB2 Connect Personal Edition xcv
2.5 Installing DB2 Replication Center xcvii
2.5.1 System kernel parameters on Solaris, HP-UX, and Linux xcvii
2.5.2 Installing DB2 Administration Client with Replication Center xcvii
2.6 Configuring DB2 Connectivity for Replication Center c
2.7 Replication Center and file directories. cii
2.8 Desktop environment for Replication Center. ciii
2.9 Opening DB2 Replication Center cv
2.10 A Quick Tour of DB2 Replication Center. cix
2.11 Managing your DB2 Replication Center Profile. cxiii
2.12 Replication Center dialogue windows cxvii
2.13 Run Now or Save SQL cxix
2.13.1 Running Saved SQL Files Later cxxiii
2.14 Creating Control Tables - Quick or Custom cxxix
2.15 Adding Capture and Apply Control Servers cxxx
2.16 Replication Center objects for non-DB2 servers cxxxi
2.17 Creating registrations and subscriptions cxxxii
2.18 Replication Center Launchpad cxxxv
2.19 Trying replication with the DB2 SAMPLE database cxxxvi
2.20 More Replication Center tips cxxxviii

Chapter 3. Replication control tables cxli
3.1 Introduction to replication control tables cxlii
3.2 Setting up capture control tables. cxliii
3.2.1 Create capture control tables cxlv
3.2.2 Platform specific issues, capture control tables cl
3.3 Setting up apply control tables cliii

3.3.1 Creating apply control tables clv
3.3.2 Platform specific issues, apply control tables clvii
3.4 Advanced considerations clix
3.4.1 Creating control tables at a command prompt clix
3.4.2 Capture control tables, advanced considerations clix
3.4.3 Apply control tables, advanced considerations clxi
3.4.4 Sizing tablespaces for control tables clxii
3.4.5 Control tables described clxv

Chapter 4. Replication Sources clxix
4.1 What is a replication source? clxx
4.2 Define a replication source from Replication Center clxx
4.2.1 Registering the replication sources clxx
4.2.2 Selecting the replication sources clxxi
4.2.3 Defining the registration options clxxiii
4.2.4 iSeries replication sources clxxxvi
4.2.5 CD Table cxciii
4.2.6 non-DB2 sources - CCD tables cxcvi
4.2.7 non-DB2 sources - Capture Triggers and Procedures cxcvi
4.3 Views as replication sources cxcviii
4.3.1 Views over one table cxcix
4.3.2 Views over multiple tables cxcix
4.3.3 Restrictions on views cci

Chapter 5. Subscription Set cciii
5.1 Subscription Set and Subscription Set Members cciv
5.1.1 Subscription attributes cciv
5.2 Subscription set and member planning ccvi
5.2.1 Member definitions to non-DB2 targets servers ccvii
5.2.2 Subscription set and apply qualifiers planning ccviii
5.3 Define Subscriptions using the Replication Center ccix
5.3.1 Create Subscription Sets with members ccix
5.3.2 Create Subscription Set without members ccx
5.3.3 Subscription Sets from non-DB2 servers ccx
5.3.4 Subscription Sets to non-DB2 servers ccx
5.3.5 Adding subscription members to existing subscription sets ccx
5.3.6 Subscription Sets and Member Notebook ccxii
5.4 Target Types Descriptions ccxli
5.4.1 User Copy ccxli
5.4.2 Point-in-time ccxli
5.4.3 Aggregate tables ccxlii
5.4.4 CCD (consistent change data) ccxlii
5.4.5 Replica ccxlix

5.5 Data Blocking ccli
5.6 Scheduling Replication ccliii
5.7 SQL script description cclv
5.8 Create subscriptions using iSeries CL commands cclix
5.8.1 Add Subscription set - ADDDPRSUB cclix
5.8.2 Add subscription members - ADDDPRSUBM. cclxiii

Chapter 6. Operating Capture and Apply cclxv
6.1 Basic operations on Capture and Apply cclxvi
6.1.1 Basic operations from the Replication Center. cclxvi
6.1.2 Basic operations from the command prompt cclxxxviii
6.1.3 Considerations for DB2 UDB for UNIX and Windows ccci
6.1.4 Considerations for DB2 UDB for z/OS cccvi
6.1.5 Considerations for DB2 UDB for iSeries cccxvi
6.1.6 Troubleshooting the operations. cccxvi
6.2 Capture and Apply Parameters cccxxv
6.2.1 Change Capture parameters. cccxxvi
6.2.2 Capture parameters cccxxvi
6.2.3 Apply Parameters cccxxvii
6.3 Other operations cccxxviii
6.3.1 Pruning control tables cccxxviii
6.3.2 Reinitializing Capture cccxxx
6.3.3 Suspend and resume Capture cccxxxi
6.4 Using ASNLOAD for the initial load. cccxxxi
6.4.1 Using ASNLOAD on DB2 UDB for UNIX and Windows cccxxxii
6.4.2 Using ASNLOAD on DB2 UDB for z/OS cccxxxv

Chapter 7. Troubleshooting and monitoring. cccxxxvii
7.1 Troubleshooting and monitoring introduced cccxxxviii
7.1.1 DB2 Administration Client cccxxxviii
7.2 Basic replication troubleshooting. cccxxxix
7.2.1 Getting help. cccxxxix
7.2.2 Status of Capture and Apply cccxlvi
7.2.3 Platform specific troubleshooting cccl
7.3 Replication alert monitoring. cccl
7.3.1 Creating monitoring control tables cccli
7.3.2 Create contacts ccclii
7.3.3 Alert Conditions. ccccliii
7.3.4 The replication monitor program ccclv
7.3.5 Receiving an alert ccclv
7.3.6 Replication monitoring example ccclvi
7.3.7 Using JCL to start monitoring on z/OS ccclxiii
7.4 Other monitoring ccclxiv

7.4.1 Examining historic data ccclxiv
7.4.2 Health center ccclxiv
7.4.3 System Monitoring ccclxv
7.5 Advanced troubleshooting ccclxv
7.5.1 asnanalyze and ANZDPR ccclxv
7.5.2 DB2 Replication trace ccclxix
7.5.3 asntrc example for Apply ccclxx
7.5.4 asntrc example for Capture ccclxxi

Chapter 8. Maintaining Your Replication Environment ccclxxiii
8.1 Maintaining Registrations ccclxxiv
8.1.1 Adding new registrations ccclxxiv
8.1.2 Deactivating and activating registration table ccclxxiv
8.1.3 Removing registrations. ccclxxviii
8.1.4 Changing capture schemas ccclxxix
8.1.5 Changing registration attributes for registered tables ccclxxix
8.2 Maintaining Subscriptions ccclxxxi
8.2.1 Adding new Subscriptions Sets. ccclxxxi
8.2.2 Deactivating and activating subscriptions ccclxxxi
8.2.3 Changing Subscription sets ccclxxxiii
8.2.4 Removing Subscription sets ccclxxxvi
8.2.5 Adding members to existing subscription sets ccclxxxvii
8.2.6 Changing attributes of subscription sets ccclxxxix
8.2.7 Adding a new column to a source and target table cccxc
8.3 Promoting a replication configuration to another system cccxcii
8.3.1 Promoting registered tables cccxcii
8.3.2 Promoting registered views cccxciii
8.3.3 Promoting subscription sets cccxciv
8.4 Maintaining capture and apply control servers cccxcv
8.4.1 Manually pruning replication control tables cccxcvi
8.4.2 RUNSTATS for replication tables cccxcvii
8.4.3 REORG for replication tables cccxcviii
8.4.4 Rebinding replication packages and plans cccxcviii
8.4.5 Recovering source tables, replication tables, or target tables. . cccxcix
8.4.6 Managing DB2 logs and journals used by Capture . . . . . cd
8.5 Full refresh procedures . . . . . cdi
8.5.1 Automatic full refresh . . . . . cdi
8.5.2 Manual full refresh. . . . . cdii
8.5.3 Bypassing the full refresh . . . . . cdii

Chapter 9. Advanced Replication Topics . . . . . cdiii
9.1 Replication filtering . . . . . cdiv
9.1.1 Replicating column subsets . . . . . cdiv

9.1.2 Replicating row subsets . . . . . cdvi
9.2 Replication transformations . . . . . cdx
9.2.1 Capture transformations . . . . . cdx
9.2.2 Source table views . . . . . cdxi
9.2.3 Apply Transformations . . . . . cdxiv
9.2.4 Before and after SQL statements . . . . . cdxiv
9.3 Replication of large objects . . . . . cdxvi
9.3.1 DB2 LOB replication . . . . . cdxvi
9.3.2 Informix LOB replication . . . . . cdxvii
9.4 Replication of DB2 Spatial Extender data . . . . . cdxix
9.5 Update anywhere replication . . . . . cdxxii
9.5.1 Administration — defining update anywhere replication . . . . . cdxxii
9.5.2 Operations — Capture and Apply . . . . . cdxxiv
9.6 DB2 Peer to peer replication . . . . . cdxxix
9.6.1 Administration and operations for peer to peer replication . . . . . cdxxix
9.6.2 Adding another peer . . . . . cdxxxiii
9.6.3 Conflict detection using triggers . . . . . cdxxxvi

Chapter 10. Performance . . . . . cdxli
10.1 End-to-end system design for replication . . . . . cdxlii
10.1.1 Pull replication system design . . . . . cdxlii
10.1.2 Push replication system design . . . . . cdxliii
10.1.3 iSeries-to-iSeries replication with remote journalling . . . . . cdxliv
10.1.4 Replicating to non-DB2 servers . . . . . cdxlvi
10.1.5 Replicating from non-DB2 . . . . . cdxlvii
10.2 Capture Performance . . . . . cdxlix
10.2.1 Reading the DB2 Log . . . . . cdxlix
10.2.2 Reading iSeries Journals . . . . . cdli
10.2.3 Collecting transaction information in memory . . . . . cdli
10.2.4 Capture Insert into CD and UOW tables . . . . . cdlv
10.2.5 Capture Pruning . . . . . cdlix
10.2.6 Capture's latency . . . . . cdlxi
10.2.7 Capture's throughput . . . . . cdlxiii
10.3 CD tables and the IBMSNAP_UOW table . . . . . cdlxv
10.4 Apply Performance . . . . . cdlxviii
10.4.1 Fetching changes from the source server . . . . . cdlxviii
10.4.2 Caching Apply's dynamic SQL at source servers . . . . . cdlxx
10.4.3 Specifying more memory for caching dynamic SQL packages . . cdlxxi
10.4.4 Transporting changes across the network . . . . . cdlxxii
10.4.5 Apply spill files . . . . . cdlxxii
10.4.6 Apply updating the target tables . . . . . cdlxxiii
10.4.7 Target table and DB2 log characteristics at the target server . . cdlxxiv
10.4.8 Caching Apply's dynamic SQL at the target server . . . . . cdlxxv

10.4.9 Querying target tables while Apply is replicating . . . . . cdlxxv
10.4.10 Apply operations and Subscription set parameters . . . . . .cdlxxvi
10.4.11 End-to-end latency of replication . . . . . cdlxxx
10.4.12 Apply Throughput . . . . . cdlxxxii
10.4.13 Time spent on each of Apply's sub-operations . . . . . cdlxxxiii
10.5 Configuring for low-latency replication . . . . . cdxci
10.6 Develoment Benchmarks . . . . . cdxciv
10.6.1 Benchmark system configuration . . . . . cdxciv
10.6.2 Benchmark results . . . . . cdxcvi

Part 1. Appendixes . . . . . cdxcix

Appendix A. DB2 Admin Client and DB2 Connect PE install . . . . . .di

Appendix B. Configuring Connections for Replication Center . . . . . dvii
B.1 Connecting Directly to DB2 for z/OS and OS/390 . . . . . .dix
B.2 Connecting to DB2 for z/OS via DB2 Connect . . . . . dxiii
B.3 Connecting directly to iSeries (AS/400) . . . . . dxvii
B.4 Connecting to iSeries via DB2 Connect server . . . . . .dxxi
B.5 Connecting to DB2 UNIX or Linux server . . . . . dxxiii
B.6 Connecting to DB2 on Windows . . . . . .dxxvii
B.7 Testing Connections and Binding Packages . . . . . .dxxxii

Appendix C. Configuring federated access to Informix . . . . . .dxxxvii
C.1 Technical requirements for federated access . . . . . dxxxviii
C.2 Information from the Informix server . . . . . dxxxviii
C.3 Software installation on the DB2 system . . . . . .dxlii
C.4 Configuring Informix sqlhosts . . . . . .dxliii
C.5 Preparing the DB2 instance for federated . . . . . dxliv
C.6 Configuring federated objects in a DB2 database . . . . . dxlvii
C.6.1 Creating federated objects using SQL statements . . . . . .dxlviii
C.6.2 Creating federated objects using the Control Center . . . . . dlii

Appendix D. Additional material . . . . . .dlxiii
Locating the Web material . . . . . .dlxiii
Using the Web material . . . . . .dlxiii
System requirements for downloading the Web material . . . . . dlxiv
How to use the Web material . . . . . dlxiv

Glossary . . . . . .dlxv

Related publications . . . . . dlxvii
IBM Redbooks . . . . . dlxvii
Other resources . . . . . dlxvii

Referenced Web sites . . . . . dlxviii
How to get IBM Redbooks . . . . . dlxviii
IBM Redbooks collections . . . . . dlxviii

Index . . . . . dlxix

Figures

1-1 Data distribution xxxii
1-2 Data consolidation xxxiii
1-3 Bidirectional replication xxxiv
1-4 Peer to peer replication xxxv
1-5 Using the Replication Center xxxvii
1-6 Capture for DB2 xxxviii
1-7 Capture with iSeries remote journaling xxxix
1-8 Capture for Informix xl
1-9 Apply for DB2 source and target xlii
1-10 Replication from DB2 to Informix xliv
1-11 Replication from Informix to DB2 xlv
1-12 Alert Monitor. xlvi
1-13 The Redbook Environment lxxiii
2-1 Graphical user interface. lxxviii
2-2 Use of API's, JDBC, and DB2 Connectivity lxxix
2-3 Use of DB2 Administration Server (DASe) lxxxi
2-4 Use of Java Toolbox/400 API's to iSeries servers lxxxii
2-5 Access to profiles, passwords, and SQL files lxxxiii
2-6 Non-DB2 sources and targets lxxxiv
2-7 Federated Server configuration example to Informix xci
2-8 Opening Replication Center from DB2 Admin Tool menu bar cvi
2-9 Replication Center icon cvi
2-10 Replication Center and Replication Center Launchpad cvii
2-11 Replication Center when opened the first time cix
2-12 Expanded tree cx
2-13 Replication Center icon tool bar. cxi
2-14 Expanded tree with objects and options cxii
2-15 Manage Passwords window cxiv
2-16 Create Apply Control Tables - Custom cxvii
2-17 Generated SQL message box cxviii
2-18 Run Now or Save SQL window cxix
2-19 File Browser for specifying file name and path cxxii
2-20 Specifying file path and filename cxxiii
2-21 Adding a Capture Control Server. cxxx
2-22 Add Capture Control Servers window cxxxi
2-23 Capture Control Server icon for Informix cxxxii
2-24 Opening Register Tables dialog window cxxxiii
2-25 Filter example cxxxiv

3-1 Launch Pad: Create capture control tables cxliii
3-2 Use an existing table space cxlviii
3-3 Launchpad: Create apply control tables cliii
3-4 Enter a new, unique capture schema clxi
4-1 Add registerable tables clxxii
4-2 Register tables clxxiv
4-3 How Apply replicates the updates to the target key clxxvii
4-4 Update-anywhere replication clxxix
4-5 Full refresh only clxxx
4-6 Capture updates as pairs of inserts and deletes clxxxiii
4-7 Re-capture changes. clxxxiv
4-8 iSeries registered sources cxc
4-9 ADDDPRREG CL command screen prompt - 1st Screen cxcii
4-10 ADDDPRREG CL command screen prompt - Last Screen cxcii
4-11 Replication source and CD table example. cxciii
5-1 Add members to subscription set. ccxii
5-2 Create Subscription Set notebook ccxiii
5-3 Source to Target mapping- Source and target tables ccxvii
5-4 Member Properties -- CCD Properties. ccxx
5-5 Member properties - Column Selection ccxxi
5-6 Member Properties - Column mapping ccxxii
5-7 SQL Expression Builder Window ccxxiv
5-8 Member Properties - Group by in Column Mapping ccxxv
5-9 Member Properties - Target Table Index ccxxvi
5-10 Member property - Use existing target index option ccxxix
5-11 Member properties - Create another index ccxxx
5-12 Show existing target indexes ccxxxi
5-13 Member properties - Row selection ccxxxii
5-14 Member Properties - Create target-table table space ccxxxiv
5-15 Member properties - Replica definitions ccxxxv
5-16 Subscription Set - Schedule replication ccxxxvi
5-17 Subscription set - SQL before and after statement ccxxxviii
5-18 Add SQL Statement or Procedure ccxxxix
5-19 SQL Assistance ccxl
5-20 CCD table multi tier replication ccxlvii
5-21 Update anywhere configuration with multi replicas ccl
5-22 Data blocking cclii
5-23 Subscription SQL Script - Update Pruning controls tables cclv
5-24 Subscription SQL Script - Subscription Set control table. cclv
5-25 Subscription SQL Script - Subscription member control table cclvi
5-26 Subscription SQL Script - Subscription column control table. cclvi
5-27 Subscription SQL Script - Subscription statement control table cclvii
5-28 Subscription SQL Script - Create target table and index cclvii

5-29 ADDDPRSUB CL command screen prompt - 1st screen cclix
5-30 ADDDPRSUB CL command screen prompt - 2nd screen cclx
5-31 ADDDPRSUB CL command screen prompt - 3rd screen cclx
5-32 ADDDPRSUB CL command screen prompt - 4th screen cclxi
5-33 ADDDPRSUB CL command screen prompt - 5th screen cclxi
5-34 ADDDPRSUB CL command screen prompt - 6th screen cclxii
5-35 ADDDPRSUB CL command screen prompt - Last screen cclxii
5-36 ADDDPRSUBM CL command screen prompt - 1st screen cclxiii
5-37 ADDDPRSUBM CL command screen prompt - 2nd screen cclxiv
5-38 ADDDPRSUBM CL command screen prompt - Last screen cclxiv
6-1 Start Capture parameter specification from Replication Center. . . . cclxviii
6-2 Start Capture cclxx
6-3 Start iSeries Capture program parameters cclxxi
6-4 Start Capture command on the iSeries cclxxii
6-5 Sample replication configuration on Windows and AIX cclxxiii
6-6 Start Apply parameter specification from Replication Center cclxxiv
6-7 Start iSeries Apply. cclxxix
6-8 Stop Capture cclxxx
6-9 Query status from Replication Center cclxxxi
6-10 WRKSBSJOB - Work with subsystem display for Capture cclxxxiii
6-11 Successful Capture program job log for controlling job cclxxxiv
6-12 Successful Capture job log for the journal job cclxxxiv
6-13 WRKSBSJOB - Work with sub system for Apply cclxxxv
6-14 Successfull Apply program joblog cclxxxvi
6-15 Unsuccessfull Apply program joblog cclxxxvii
6-16 Apply report cclxxxvii
6-17 Start Capture from the command prompt cclxxxix
6-18 Start Capture and query status from USS shell. ccxcii
6-19 Start Apply and query status from the command prompt. ccxciv
6-20 Sample replication configuration for z/OS to z/OS. ccxcvi
6-21 STRDPRCAP - Start capture program. First screen ccxcviii
6-22 STRDPRCAP - Start Capture program. Last screen ccxcviii
6-23 ENDDPRCAP - End Capture program. ccxcix
6-24 STRDPRAPY - Start Apply program first screen ccc
6-25 STRDPRAPY - Start Apply program last screen ccc
6-26 ENDDPRAPY - End Apply program. ccci
6-27 Enablement for replication cccii
7-1 DB2 messages. cccxl
7-2 Capture Messages. cccxlvii
7-3 Apply Report cccxlviii
7-4 Select alert condition for capture ccccliv
7-5 Capture and Apply alert entries in Replication Center. ccclvi
7-6 Alert conditions for Apply ccclvii

7-7 Monitor conditions displayed in Command Center ccclviii
7-8 Start Monitor window example in Replication Center ccclix
7-9 Monitor Alert identifier ccclxi
7-10 Alerts detail display in Replication Center ccclxii
7-11 Replication alert records in the ASN.IBMSNAP_ALERTS table . . . ccclxiii
7-12 Health Center. ccclxv
8-1 Deactivate Subscription Set. ccclxxv
8-2 Deactivating registration tables ccclxxvi
8-3 SQL to activate registration after unexpected capture errors . . . ccclxxvii
8-4 Removing registrations ccclxxviii
8-5 iSeries CL command to remove registrations ccclxxix
8-6 Registered table properties ccclxxx
8-7 SQL to reset pruning for deactivated subscription sets ccclxxxii
8-8 SQL to change Apply subscription sets name ccclxxxiii
8-9 Change subs set name in subs set statement table ccclxxxiii
8-10 Change subscription set name in the pruning control tables ccclxxxiv
8-11 SQL to change apply qualifier at the apply control server ccclxxxiv
8-12 Change subs set name in subs set statement table ccclxxxv
8-13 SQL to change the apply qualifier in the pruning control tables. . ccclxxxv
8-14 Remove subcription sets ccclxxxvi
8-15 iSeries CL command to remove subscription sets. ccclxxxvii
8-16 SQL to update pruning control tables to resume apply processingccclxxx-vii
8-17 Locate subscription member map ID ccclxxxviii
8-18 Insert the CAPSTART signal ccclxxxviii
8-19 SQL to check CAPSTART signal was started ccclxxxviii
8-20 SQL for Apply to start subscription set and prevent full refresh. . ccclxxxix
8-21 SQL to activate registration after unexpected capture errors cccxcix
10-1 Replication - ‘pull’ system design. . . . . .cdxliii
10-2 Replication - ‘push’ system design. . . . . .cdxliv
10-3 Replication iSeries-to-iSeries using remote journalling . . . . . cdxlv
10-4 Replication to Informix . . . . . .cdxlvi
10-5 Replicating from Informix - pull system design . . . . . .cdxlviii
10-6 Capture Latency window . . . . . .cdlxiii
10-7 Replication Center - Apply End-to-End Latency . . . . . .cdlxxxi
A-1 DB2 Connect PE - Setup Launchpad . . . . . dii
A-2 DB2 Connect PE - features selection window . . . . . .diii
B-1 Configuration Assistant - Add new database . . . . . dx
B-2 Configuration Assistant - DB2 for z/OS entry. . . . . dxii
B-3 Configuration Assistant - DB2 for z/OS via DB2 Connect Gateway . . dxv
B-4 iSeries Relational Database name in WRKRDBDIRE . . . . . dxvii
B-5 Configuration Assistant - iSeries connection entry . . . . . dxx
B-6 Configuration Assistant - DB2 on Linux or UNIX connection entry . . dxxvi

B-7 Configuration Assistant - DB2 on Windows connection entry . . . . . dxxxi
B-8 Configuration Assistant - Selected - Test connection . . . . . .dxxxii
B-9 Configuration Assistant - Test Connection window . . . . . dxxxiii
B-10 Configuration Assistant - Connection Test successful. . . . . dxxxiv
B-11 Configuration Assistant - Bind dialog window . . . . . dxxxv
C-1 Control Center - accessing federated database objects . . . . . dliii
C-2 Control Center - Create Wrapper window . . . . . dliv
C-3 Control Center - Create Server window. . . . . .dlv
C-4 Control Center - Server Options window . . . . . dlvi
C-5 Control Center - Create User Mapping window . . . . . .dlvii
C-6 Control Center - server connection test with nickname filter . . . . . .dlviii
C-7 Control Center - Create Nicknames - defaults. . . . . .dlx
C-8 Control Center - Create Nicknames with custom settings . . . . . dlxi
C-9 Control Center with nicknames . . . . . .dlxii

Tables

1-1 Products required for capturing changes xlix
1-2 Recommended Apply products for applying changes l
1-3 Alert Monitor Server requirements lii
1-4 Monitored servers requirements lii
1-5 Apply rework rules lxiii
3-1 List of the Capture Control Tables cxlv
3-2 z/OS default table spaces cl
3-3 Listing of capture control tables for non-DB2 relational sources cli
3-4 List of apply control tables clv
3-5 Tablespace sizing for capture control tables clxii
3-6 Sizing worksheet for IBMSNAP_UOW clxiii
3-7 Tablespace sizing for apply control tables clxiv
3-8 IBMSNAP_APPPARMS Columns Described clxvi
4-1 Sizing worksheet for CD tables cxcv
4-2 Change capture or full refresh replication of a view cc
5-1 CCD table column description ccxlii
5-2 Type of CCD tables ccxlv
5-3 IBMSNAP_EVENT Column description ccliv
6-1 DB2 DataPropagator V8.1 plans and packages cccviii
6-2 Old and new Capture parameters cccxiii
6-3 Old and new Apply parameters cccxiv
7-1 Common Message Identify Prefixes cccxli
7-2 Message Identify Endings cccxlii
7-3 Monitor control tables. cccli
7-4 Tablespace sizing for monitor control tables ccclii
9-1 Control tables for update anywhere replication . . . . . cdxxii
9-2 Starting points for PEERn subscription sets . . . . . cdxxxiv
9-3 Changes processed at PEER1 . . . . . cdxxxv
10-1 Description of Apply ‘S’ Performance Trace Records: . . . . . cdlxxxvi
10-2 Description of Apply Trace ‘M’ Performance Trace Records . . . . cdlxxxviii
10-3 IRWW Tables’ Row Lengths . . . . . cdxciv
10-4 IRWW workload characteristics . . . . . cdxcv
10-5 Benchmark soure server Database Configuration parameters . . . . cdxcv
10-6 Benchmark target table latency . . . . . cdxcvii

Examples

Note to Author: This file needs to be removed from your .book before final publication. Delete the LOE file from your book using:
Open .book > select the file > File > Rearrange Files > Delete > Done

2-1 First few lines of generated SQL for Capture Control tables cxxiv
2-2 Connection Information cxxvii
3-1 Size of row in UOW cxlvii
3-2 Create SMS cxlvii
3-3 IBMSNAP_APPPARMS DDL. clxv
3-4 Initialize and modify the parameters for Apply clxvii
3-5 IBMSNAP_COMPENSATE DDL clxviii
4-1 Standard conflict detection. clxxxv
4-2 Create Journal receiver clxxxvi
4-3 Create journal clxxxvii
4-4 Start Journal. clxxxvii
4-5 Create the remote journal clxxxviii
4-6 A view on one table cxcix
4-7 View on two tables, one differential refresh other not registered cc
4-8 View on two tables, both of them differential refresh cci
6-1 Configure connection cclxxvii
6-2 Create Apply SQL package cclxxviii
6-3 Grant authority to SQL packages. cclxxviii
6-4 STRDPRAPY command generated from the Replication Center . . cclxxx
6-5 Sample USS profile Data Propagator cclxxxviii
6-6 Capture log ccxc
6-7 Apply log ccxciv
6-8 Sample JCL for starting Capture cccxii
6-9 Sample JCL for starting Apply cccxii
6-10 List applications cccxx
6-11 Display Capture threads on z/OS cccxxi
6-12 Display Apply threads on z/OS cccxxii
6-13 IPC queue of Capture on z/OS cccxxiii
6-14 IPC queue of Apply on AIX cccxxiii
6-15 Sample configuration file for ASNLOAD cccxxxiii
7-1 Detailed output of a DB2 Message cccxli
7-2 Taking a db2trc cccxlv

7-3 CAP.log cccxlix
7-4 Email notification ccclv
7-5 Query of monitor conditions table ccclvii
7-6 Query of Alerts ccclxii
7-7 Sample JCL to start monitor on z/OS ccclxiii
7-8 asnanalyze command ccclxvi
7-9 Analyzer output to terminal ccclxvii
7-10 Analyser output table of contents ccclxviii
8-1 Manually deactivating a registration ccclxxvi
8-2 SQL statement to prune APPLYTRAIL table cccxcvi
8-3 SQL statement to prune ASN.IBMSNAP_APPLYTRACE cccxcvii
9-1 Subsetting columns using a source table view . . . . . cdv
9-2 CD table trigger to exclude captured deletes . . . . . cdvi
9-3 CD table trigger to exclude changes based on a column value . . . . . cdvii
9-4 Informix row filter . . . . . cdviii
9-5 Row filter to include only rows with a certain value . . . . . cdviii
9-6 CD table trigger to set default values for a null column . . . . . cdx
9-7 set values in one column based on another column . . . . . cdxi
9-8 set values in one column based on another table . . . . . cdxi
9-9 Source table view over one table . . . . . cdxii
9-10 Source table view over two tables . . . . . cdxii
9-11 CD table views . . . . . cdxiii
9-12 CD table view modification for double delete problem . . . . . cdxiii
9-13 CD table with a LOB column . . . . . cdxvi
9-14 DB2 table with a spatial data column . . . . . cdxix
9-15 Source table view for spatial replication . . . . . cdxix
9-16 Target table for spatial replication . . . . . cdxx
9-17 Target table trigger to convert text to spatial . . . . . cdxx
9-18 Adding conflict columns to application tables . . . . . cdxxxvii
9-19 Triggers to populate conflict columns . . . . . cdxxxvii
9-20 Trigger with modifications to prevent conflicts . . . . . cdxxxviii
9-21 File with conflict SQLSTATEs used by Apply . . . . . cdxxxix
10-1 Performance Trace Records in formatted asntrc for Apply . . . . . cdlxxxv
10-2 Member info in formatted anstrc output for Apply . . . . . cdlxxxvi
10-3 Peformance Trace Records used in our calculations . . . . . cdlxxxix
B-1 DB2 CLP commands for connection direct to DB2 for z/OS . . . . . dxii
B-2 DB2 z/OS Alias name at DB2 Connect server . . . . . dxiv
B-3 DB2 CLP commands for B2 for z/OS via DB2 Connect server . . . . . dxv
B-4 DB2 CLP commands for connection direct to iSeries . . . . . dxx
B-5 DB2 CLP commands for connection to iSeries via DB2 Connect server . . dxxii
B-6 Obtaining DB2 UNIX Alias from DB directory . . . . . dxxiv
B-7 DB2 CLP commands for connection to DB2 on UNIX . . . . . dxxv

B-8 Obtaining the DB2 Windows Alias name . . . . . dxxviii
B-9 DB2 CLP Commands for connection to DB2 on Windows . . . . . dxxx
B-10 Testing connectivity using DB2 CLP . . . . . dxxxiv

Notices

This information was developed for products and services offered in the U.S.A.

IBM may not offer the products, services, or features discussed in this document in other countries. Consult your local IBM representative for information on the products and services currently available in your area. Any reference to an IBM product, program, or service is not intended to state or imply that only that IBM product, program, or service may be used. Any functionally equivalent product, program, or service that does not infringe any IBM intellectual property right may be used instead. However, it is the user's responsibility to evaluate and verify the operation of any non-IBM product, program, or service.

IBM may have patents or pending patent applications covering subject matter described in this document. The furnishing of this document does not give you any license to these patents. You can send license inquiries, in writing, to:
IBM Director of Licensing, IBM Corporation, North Castle Drive Armonk, NY 10504-1785 U.S.A.

The following paragraph does not apply to the United Kingdom or any other country where such provisions are inconsistent with local law: INTERNATIONAL BUSINESS MACHINES CORPORATION PROVIDES THIS PUBLICATION "AS IS" WITHOUT WARRANTY OF ANY KIND, EITHER EXPRESS OR IMPLIED, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF NON-INFRINGEMENT, MERCHANTABILITY OR FITNESS FOR A PARTICULAR PURPOSE. Some states do not allow disclaimer of express or implied warranties in certain transactions, therefore, this statement may not apply to you.

This information could include technical inaccuracies or typographical errors. Changes are periodically made to the information herein; these changes will be incorporated in new editions of the publication. IBM may make improvements and/or changes in the product(s) and/or the program(s) described in this publication at any time without notice.

Any references in this information to non-IBM Web sites are provided for convenience only and do not in any manner serve as an endorsement of those Web sites. The materials at those Web sites are not part of the materials for this IBM product and use of those Web sites is at your own risk.

IBM may use or distribute any of the information you supply in any way it believes appropriate without incurring any obligation to you.

Information concerning non-IBM products was obtained from the suppliers of those products, their published announcements or other publicly available sources. IBM has not tested those products and cannot confirm the accuracy of performance, compatibility or any other claims related to non-IBM products. Questions on the capabilities of non-IBM products should be addressed to the suppliers of those products.

This information contains examples of data and reports used in daily business operations. To illustrate them as completely as possible, the examples include the names of individuals, companies, brands, and products. All of these names are fictitious and any similarity to the names and addresses used by an actual business enterprise is entirely coincidental.

COPYRIGHT LICENSE:
This information contains sample application programs in source language, which illustrates programming techniques on various operating platforms. You may copy, modify, and distribute these sample programs in any form without payment to IBM, for the purposes of developing, using, marketing or distributing application programs conforming to the application programming interface for the operating platform for which the sample programs are written. These examples have not been thoroughly tested under all conditions. IBM, therefore, cannot guarantee or imply reliability, serviceability, or function of these programs. You may copy, modify, and distribute these sample programs in any form without payment to IBM for the purposes of developing, using, marketing, or distributing application programs conforming to IBM's application programming interfaces.

Trademarks

The following terms are trademarks of the International Business Machines Corporation in the United States, other countries, or both:

Redbooks(logo)™ 

Trademark Search: Open all files to be trademark searched except this file. Use the **Toolkit>Trademark Search** and copy/paste the resulting FrameMaker console IBM trademarks to the list above using a CellBody tag. Copy/paste the Lotus Development Corporation trademarks, if any, from the bottom of the FM console to the Lotus trademark list below.
Sort Trademark lists: Sort the lists, if needed, by converting to a table, sorting table cells and converting back to a list. Use the following three steps:
1. Select all marks to be sorted and **Table>Convert to Table>Tab_1x1>Convert**
2. Select all table cells and **Table>Sort>Column 1>Sort**
3. Select all table cells, **Table>Convert to Paragraphs>Row by Row** and delete extra blank lines from list.

Delete this note box when done.
Delete “Other company trademarks” attribution lines if their trademarks are not used in your book/paper.

The following terms are trademarks of International Business Machines Corporation and Lotus Development Corporation in the United States, other countries, or both:

Lotus® Word Pro®

The following terms are trademarks of other companies:

ActionMedia, LANDesk, MMX, Pentium and ProShare are trademarks of Intel Corporation in the United States, other countries, or both.

Microsoft, Windows, Windows NT, and the Windows logo are trademarks of Microsoft Corporation in the United States, other countries, or both.

Java and all Java-based trademarks and logos are trademarks or registered trademarks of Sun Microsystems, Inc. in the United States, other countries, or both.

C-bus is a trademark of Corollary, Inc. in the United States, other countries, or both.

UNIX is a registered trademark of The Open Group in the United States and other countries.

SET, SET Secure Electronic Transaction, and the SET Logo are trademarks owned by SET Secure Electronic Transaction LLC.

Other company, product, and service names may be trademarks or service marks of others.

Preface

IBM DB2 Replication (called DataPropagator on some platforms) is a poweful, flexible facility for copying DB2 and/or Informix data from one place to another. The IBM replication solution includes transformation, joining, and filtering of data. You can move data between different platforms. You can distribute data to many different places or consolidate data in one place from many different places. You can exchange data between systems.

The objective of the redbook is to provide you with detailed information you can use to install, configure, and implement replication among the IBM database family -- DB2 and Informix. The redbook is organized so that each chapter builds upon information contained in the chapters before it.

- Chapter 1 is an overview of IBM DB2 Replication at a high and then a low level. This chapter introduces you to the replication components, product requirements, sample configurations, and the new functions introduced in Version 8.
- Chapters 2 - 5 give you details on installing, configuring, and using the new DB2 V8 Replication Center to define replication scenarios.
- Chapters 6 - 8 cover the details of operating the replication programs to capture and apply changes and include monitoring, troubleshooting, and maintaining the replication environment.
- Chapter 9 includes advanced topics, such as bi-directional replication, replication of large objects, and replication of spatial data.
- Chapter 10 describes the DB2 V8 replication performance benchmarks run at the IBM Silicon Valley Lab.

The team that wrote this redbook

This redbook was produced by a team of specialists from around the world working at the International Technical Support Organization, San Jose Center.

Lijun (June) Gu is is a Project Leader at the International Technical Support Organization (ITSO), San Jose Center, California (USA), where she conducts projects on all aspects of DB2 Universal Database (DB2 UDB). She is an IBM-Certified Solution Expert of DB2 UDB Database Administrator and an IBM-Certified Specialist DB2 UDB User. She has extensive experience in DB2 UDB, UNIX/AIX/LINUX, and ADSM administration as well as database design and modeling. She holds three master degrees: MS in Computer Science, MS in Analytical Chemistry and MS in Soil Science.

Lloyd Budd is a DB2 Support Analyst at the IBM Toronto Lab in Markham, Canada. He is an IBM Certified Solutions Expert - DB2 UDB Version 5, 6, & 7 Database Administration. He has 2 years of experience with DB2. He holds a bachlors degree in Computer Science specializing in Software Engineering from the University of Victoria, Canada. His areas of expertise include DB2 Multi-platform logging, backup, and recovery. He is an active member of the DB2 user, documentation, and development communities. In his personal time, he is involved in the open source community.

Aysegul Cayci is a DB2 instructor and consultant on DB2 for z/OS and UNIX and Windows platforms in Turkey. She has 13 years of experience in DB2. She has been working at Polaris Computer Systems for 7 years. She has BSc in Computer Science from Middle East Technical University, Ankara and MSc in Computer Engineering from Bosphorus University, Istanbul. She is an IBM Certified Solutions Expert - DB2 UDB Version 7.1 Database Administration, DB2 UDB Version 7.1 Family Application Development, DB2 UDB V7.1 for OS/390 Database Administration. Her areas of expertise include DB2 performance management and database administration.

Colin Hendricks is an Certified IBM I/T Specialist in the United, States. He has over 20 years of experience in IT field, 13 years with IBM. Currently providing pre-sales support on BI/CRM Solutions and Data Management products on the iSeries platform. Specializing in iSeries data replication technical support and implementation. Through out his I/T career he has worked primarily on IBM midrange platform, performing various technical support roles, IT operations management, project management, application designing and development.

Micks Purnell has been with IBM over 20 years. He was a networking technical specialist with expertise in SNA software and hardware, then became a product manager for IBM's TCP/IP and OSI networking software products. Micks moved on to become a client/server technical specialist focused on distributed database and on client/server system development issues and then became the project manager for a successful customer proof-of-concept using IBM's DB2 Replication Version 1 and DataJoiner Version 1. Micks has continued to support DataJoiner and DB2 replication in various positions since that time. His current position in IBM's Advance Technical Support as a technical specialist for these products and for the DB2 federated server.

Carol Rigdon is an IBM I/T specialist in the United States, providing pre-sales technical support for DB2 in the Americas. She has 20 years of experience in data management and has been working with replication for the last 7 years. She holds a degree in computer science from the University of North Florida, US. Her areas of expertise include homogeneous and heterogeneous database replication, cross-platform database connectivity, and database federation.

Thanks to the following people for their contributions to this project:

Emma Jacobs
Journel Saniel
International Technical Support Organization, San Jose Center

Beth Hamel
Patrick See
Jaime Anaya
Jayanti Mahapatra
Ken Chia
Ravichandran Subramaniam
Chuck Vanhavermaet
Kathy Kwong
Michael Morrison
David Yang
Jing-Song Jang
Benedikt Berger
Karan Karu
Lakshmi Palaniappan
Miranda Kwong
Kris Tachibana
Jason Chen
IBM Silicon Valley Lab, San Jose, CA, USA

Become a published author

Join us for a two- to six-week residency program! Help write an IBM Redbook dealing with specific products or solutions, while getting hands-on experience with leading-edge technologies. You'll team with IBM technical professionals, Business Partners and/or customers.

Your efforts will help increase product acceptance and customer satisfaction. As a bonus, you'll develop a network of contacts in IBM development labs, and increase your productivity and marketability.

Find out more about the residency program, browse the residency index, and apply online at:

ibm.com/redbooks/residencies.html

Comments welcome

Your comments are important to us!

We want our Redbooks to be as helpful as possible. Send us your comments about this or other Redbooks in one of the following ways:

- Use the online **Contact us** review redbook form found at:

  ibm.com/redbooks

- Send your comments in an Internet note to:

  redbook@us.ibm.com

- Mail your comments to:

  IBM Corporation, International Technical Support Organization
  Dept. QXXE Building 80-E2
  650 Harry Road
  San Jose, California 95120-6099

1

Introduction to DB2 Replication V8

This chapter discusses the following topics

- Overview of the IBM replication solution
- Why use replication?
- DB2 V8 replication from 30,000 feet
- Putting the pieces together
- DB2 Replication V8 close up
- What's new in DB2 Replication V8
- The Redbook environment

1.1 Overview of the IBM replication solution

Replication is the copying of data from one place to another. Data can be extracted by programs, transported to some other location, and then loaded at the receiving location. A more efficient alternative is to extract only the changes since the last processing cycle and transport/apply those to the receiving location.

Data may be filtered and transformed during replication. There may be other requirements for replication, such as time constraints. In most cases, replication must not interfere with existing applications and have minimal impact on production systems. The replication processes need to be managed and monitored.

This book covers the IBM DB2 replication solution. DB2 replication provides change-based replication for the IBM database family, including:

- DB2 Universal Database for z/OS and OS/390 Versions 6, 7, and 8

  DB2 DataPropagator for Z/OS and OS/390 V8
- DB2 Universal Database for iSeries Version 5 Release 2

  DataPropagator for iSeries V5R2
- DB2 Universal Database Version 8 for Windows, UNIX, and Linux

  Included with the DB2 product, including 64bit versions and Linux OS/390
- Informix Dynamic Server Versions 7, 8, and 9

  DB2 Universal Database Enterprise Server Edition Version 8 or DB2 Connect Version 8 is required.

  Informix Dynamic Server replication is not covered in this book. If the source and target systems for replication are both Informix, then Informix Enterprise Replication or Informix High Availability Data Replication should be used. Refer to the Informix web site, `http://www.software.ibm.com/data/informix` for information about Informix replication solutions.

  Use this book if you are copying from Informix to DB2 or to Informix from DB2

Note: Support for heterogeneous replication is included in the DB2 V8 replication solution. This type of replication requires federated access to the non-IBM database. Federated access to non-IBM databases will be provided in a future IBM product that will work with DB2 V8 replication.

1.2 Why use replication?

Businesses use replication for many reasons. The business requirements can be categorized:

- Distribution of data to other locations
- Consolidation of data from other locations
- bidirectional exchange of data with other locations
- Some variation or combination of the above

1.2.1 Distribution of data to other locations

Distribution of data involves moving all or a subset of the data to one or more locations.

Data can be copied to a central data warehouse or decision support system. Often, this involves data transformation and denormalization. Subsets of the data can be copied to data marts to provide groups of users with local access. This allows you to leverage your enterprise data with business intelligence tools, while maintaining the security and performance of your production applications.

Distribution of data is also used to provide data to applications in the same or different environments. This can be as simple as maintaining a copy of the production data on another similar system. There may be complex data transformation needed to fit new application requirements. The new application may a web application, a purchased package, or an application distributed on multiple laptop computers. The data that is copied may need to be filtered and/or transformed for the target application.

Distribution of data can also be used to provide application co-existence when migrating from one environment to another. Legacy data can be copied to the new environment for reference by the new applications until such time as the legacy applications are migrated to the new environment.

Figure 1-1 shows two target servers replicating from a single source server:

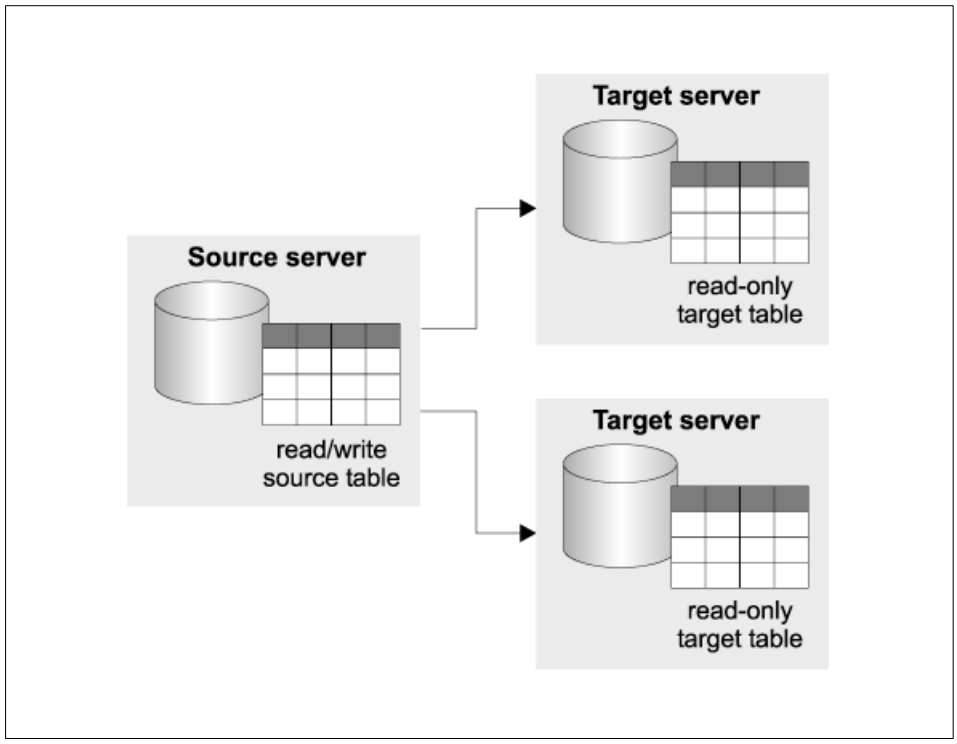


Figure 1-1 Data distribution

The two target servers may be copying different subsets or transformations of the data on different schedules.

1.2.2 Consolidation of data from remote systems

An enterprise may have data on many different distributed systems. Retail companies have data at each store. Manufacturing companies have data at each plant. Insurance companies have data at each branch office or on each salesperson's laptop computer. Replication can copy changes from each of the distributed sites to a central site for analysis, reporting, and for enterprise application processing.

Consolidation of data can be very useful for business intelligence applications such as OLAP or Data Mining.

Figure 1-2 shows a target server replicating from two source servers.

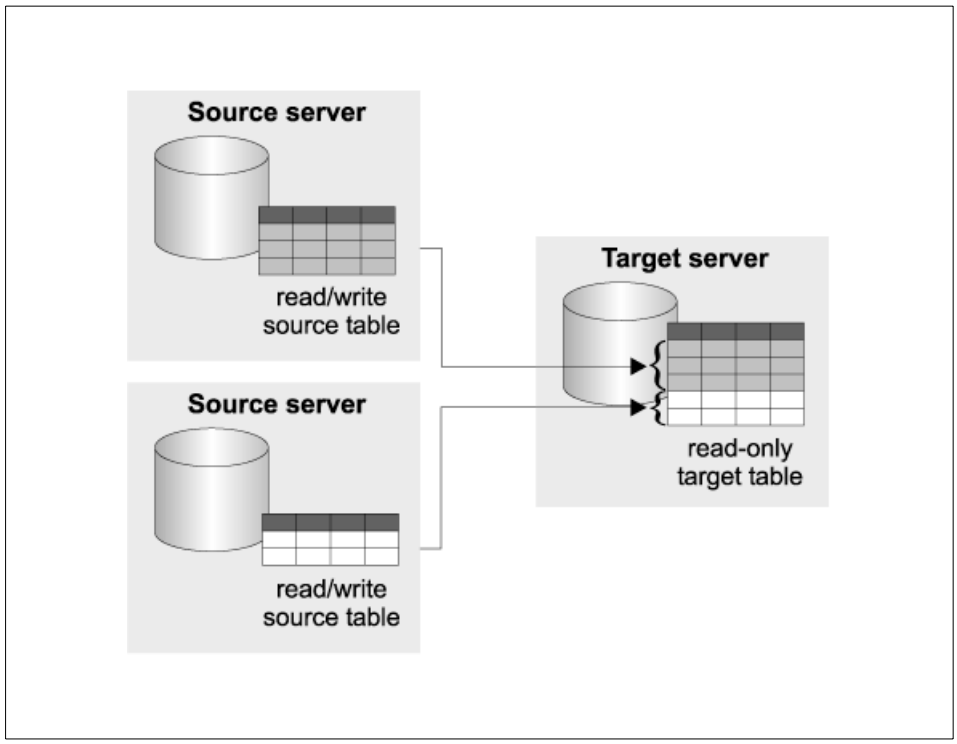


Figure 1-2 Data consolidation

The data at the source servers has the same structure. In SQL terms, the target is a UNION of the sources.

1.2.3 Bidirectional exchange of data

If the data can be updated at multiple locations, then replication must process changes made at any of the sites in a coordinated fashion. One location serves as the master location and distributes changes to the target locations. Changes made at the targets flow to other target sites through the master.

Bidirectional replication is used for mobile applications where the target may be a computer in a branch office or in a delivery truck. Often, there are many targets and they are only occasionally connected to the source system. The connection may be via phone lines, so efficiency is important.

This is sometimes called “master-slave” replication.

Figure 1-3 illustrates bidirectional replication with a designated master:

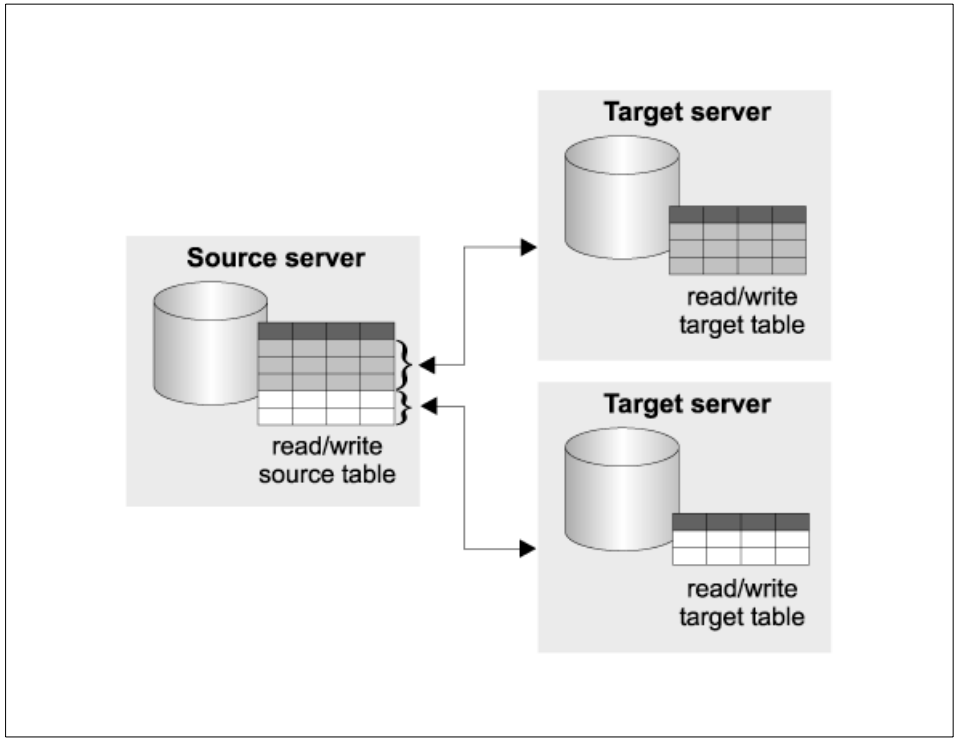


Figure 1-3 Bidirectional replication

Another type of bidirectional replication does not have a designated master location. Each location copies changes from all other locations directly. This is often called “multi-master” or “peer-to-peer” replication. Peer-to-peer replication can be used to maintain disaster recovery sites, to provide fail-over systems for high availability, and to balance query workload across multiple locations.

Figure 1-4 shows a peer-to-peer configuration:

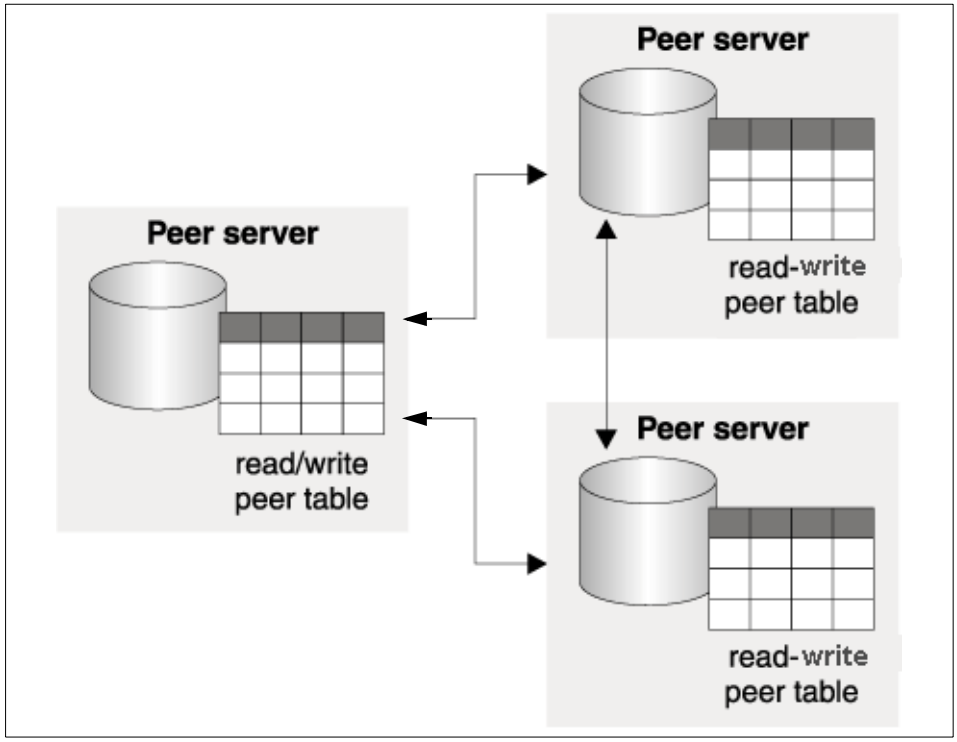


Figure 1-4 Peer to peer replication

1.2.4 Other requirements

Some applications may not require a complete copy of the data, but need a record of all the changes that have occurred. This is useful for maintaining an audit trail and also can be used to provide the changes for special processing. For instance, the changes from a source system might be staged to another table where they could be cleansed and validated before being processed at the desired target location. This is a variation on the data distribution technique, with the target table receiving only changes.

Changes can also be used in a multi-tier replication scenario, where changes are copied from a central source to a staging area on another system and then copied from the staging area to the desired target locations. This is useful when there are many target locations because it minimizes the impact on the source system.

1.3 DB2 V8 replication from 30,000 feet

The IBM replication solution has four components:

- Administration
- Capture
- Apply
- Alert Monitor

The four components communicate via relational tables, called *control tables*. The control tables are created and populated using the *Replication Center*. The Capture, Apply, and Alert Monitor programs read and update information in the control tables.

1.3.1 Administration

The Replication Center is a graphical user interface used to define replication sources and map sources to targets. It is also used to manage and monitor the Capture and Apply processes on local and remote systems. The Replication Center runs on Windows and UNIX systems and must have connectivity to both the source and target servers. The DB2 V8 Administration Client for Windows and UNIX includes the Replication Center.

Figure 1-5 shows a replication administrator using the Replication Center to manage the other three replication components:

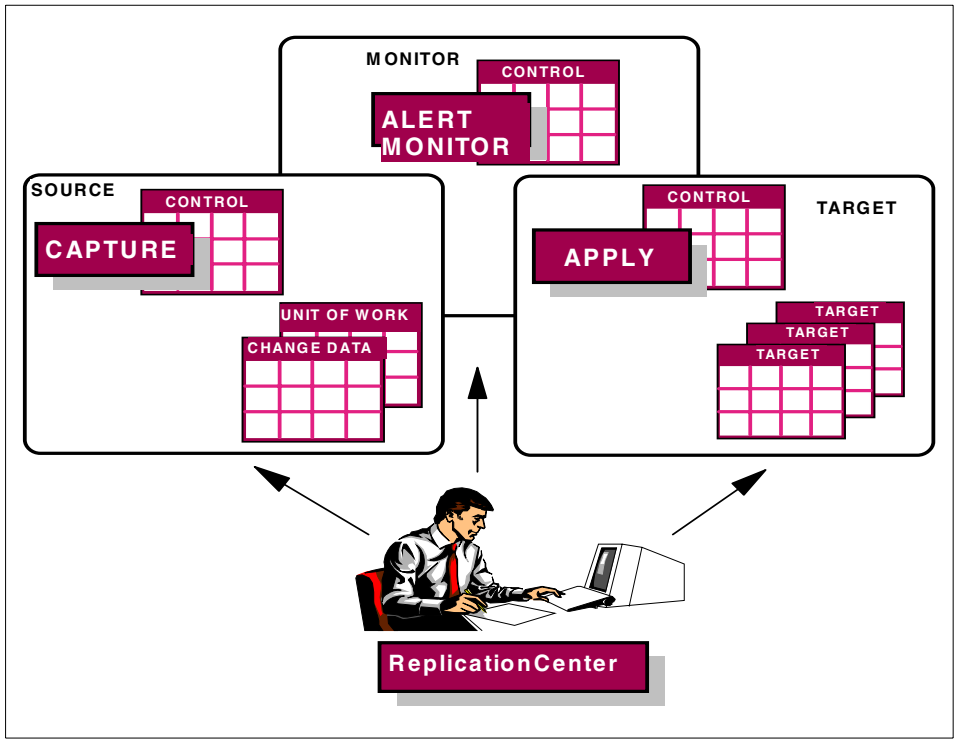


Figure 1-5 Using the Replication Center

1.3.2 Capture

Changes to DB2 source tables are captured by a Capture program running at the source server. The DB2 source server can be DB2 for z/OS and OS/390 Versions 6,7 and 8, DB2 for iSeries Version 5 Release 2, or DB2 for Windows and UNIX Version 8. Changes to Informix source tables are captured by triggers created automatically when the replication source is defined. Data can be filtered by column during the Capture process.The captured changes are stored in a table local to the source table and are automatically deleted after they have been applied.

DB2 Capture

Figure 1-6 is the data flow for DB2 Capture:

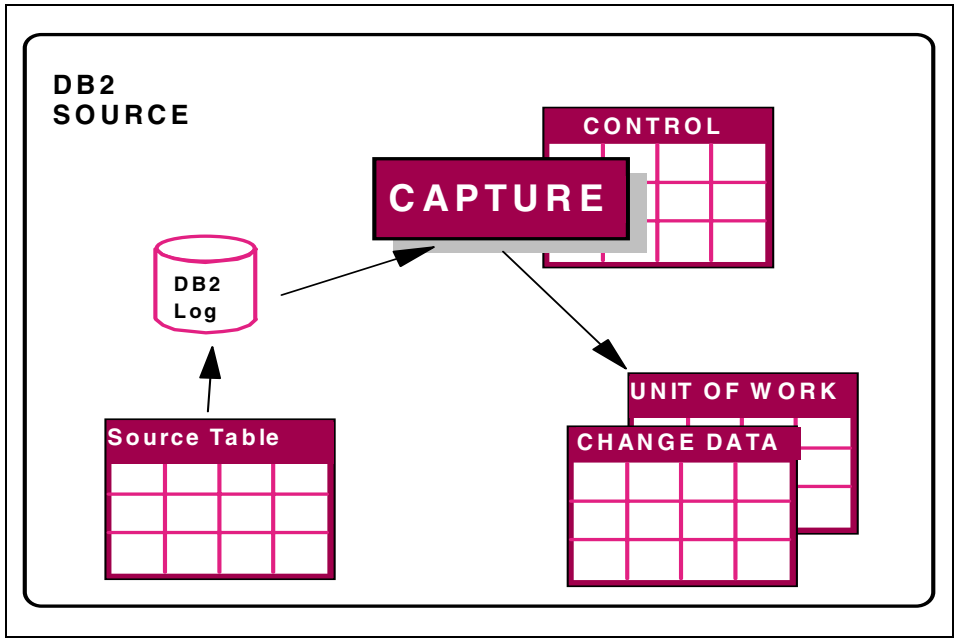


Figure 1-6 Capture for DB2

When changes are made to the source table, DB2 writes log records. These log records are used for database recovery and for replication. The Capture program uses database interfaces to access log records:

- DB2 z/OS and OS/390 IFI 306
- DB2 Windows and UNIX asynchronous log read API db2ReadLog
- iSeries RTVJRNE command

Each source table has a corresponding *Change Data* (CD) table where the captured changes are stored. The CD table is created by the Replication Center when you define a replication source table. You can choose to capture a subset of the source table columns. You can also capture the values before the change is made (called *before-image* columns) and/or the values after the change is made (called *after-image* columns). The log record sequence number (*LSN*) of a change is used to uniquely identify that change.

DB2 Capture holds the changes in memory until a COMMIT is issued for those changes. When a COMMIT is issued for a transaction that involves replication source tables, Capture inserts the captured changes into the appropriate CD tables and stores the COMMIT information in the Unit of Work (UOW) control

table. When Capture detects a ROLLBACK, it removes the associated changes from memory.

The CD tables and the UOW table are always located on the server where the source table is located. The only exception is when iSeries remote journaling is used.

Figure 1-7 shows iSeries replication with remote journaling:

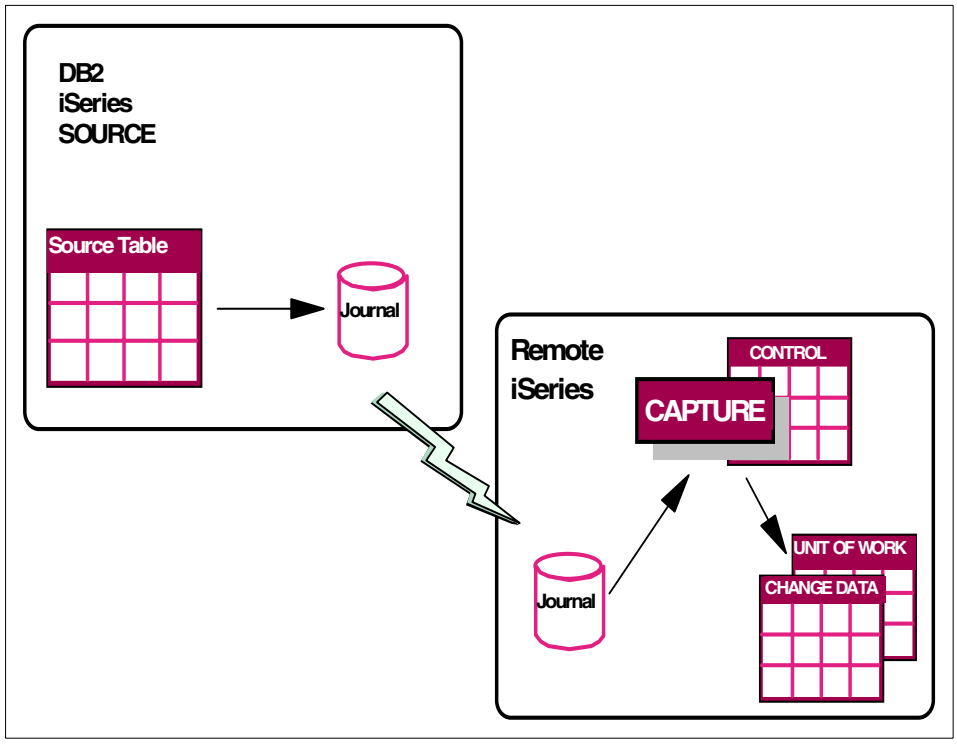


Figure 1-7 Capture with iSeries remote journaling

You can set up remote journaling on the iSeries with the **ADDRMTJRN** command. Changes to the source table are written to the local log (journal) and also sent to another iSeries system synchronously or asynchronously. Capture can run on the remote iSeries system and replicate changes from the journal on that system.

You can run multiple Capture programs on a source server to improve throughput. Each Capture has its own schema for control tables and its own set of CD tables. The schema is defined using the Replication Center when you create the capture control tables. You specify the schema when you define your replication sources and targets and when you start the Capture program. The default *capture schema* is ASN.

DB2 capture for bidirectional replication

For bidirectional replication, called *Update Anywhere*, there will be a Capture program running on both servers with one Apply program running at the target server to process changes in both directions. *Peer-to-peer* replication has a Capture and an Apply program running at each server.

Informix Capture

Figure 1-8 is the data flow for Informix capture triggers:

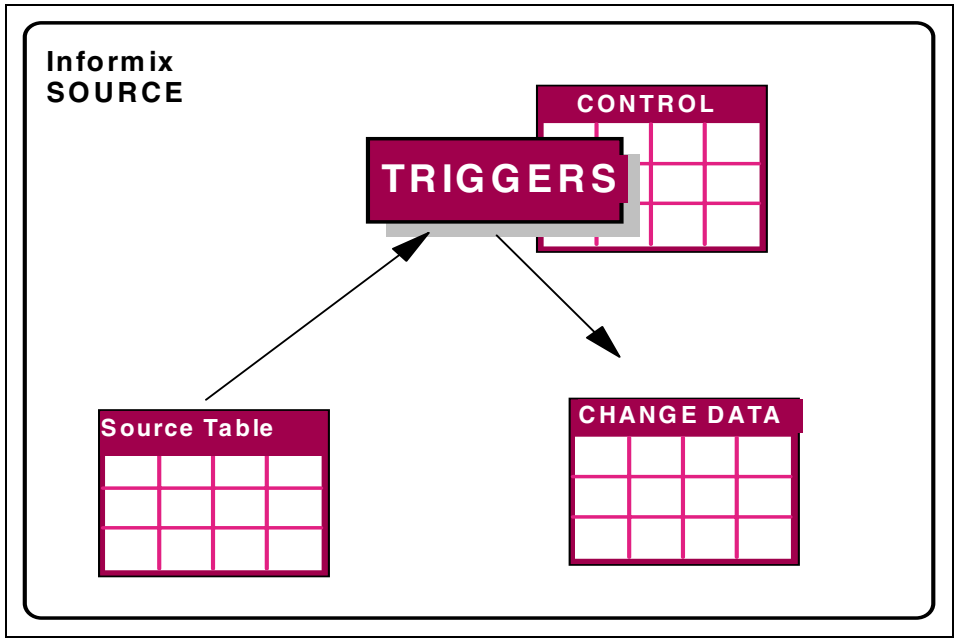


Figure 1-8 Capture for Informix

Remember that DB2 V8 for Windows and UNIX or DB2 Connect V8 is required when defining replication from or to an Informix database. DB2 Replication is written using DB2 syntax and data types, so the DB2 V8 federated capability is required to translate from DB2 to Informix. If you use one DB2 V8 federated database to access multiple Informix source databases, then each Informix source database must have its own unique capture schema. There will be one capture control table defined in the DB2 V8 federated database, ASN.IBMSNAP_CAPSCHEMAS.

When you define an Informix replication source table, the Replication Center creates insert, update, and delete triggers for the source table. Informix triggers are limited to 256 characters in length. This is not enough for the replication logic, so three Informix stored procedures are also created. The Replication Center

also creates a consistent change data (CCD) table to hold the changes captured by the triggers. There is one CCD table for each Informix replication source. All these objects are located in the same Informix database as the source.

When a change is made to the Informix source table, the appropriate trigger is executed with the before- and after-images of the changed row. The trigger calls the associated stored procedure and that procedure inserts a row into the CCD table, along with control information. Triggers do not have access to transaction information, so there is no UOW table.

1.3.3 Apply

Captured changes are applied to target tables by Apply programs. The Apply program can run on any server and must have connectivity to both the source and the target servers. Data can be filtered by column, filtered by row, joined with other data (using views), and transformed with SQL expressions during the Apply process. bidirectional replication, including peer-to-peer is supported only for the DB2 family.

DB2 source and target

Figure 1-9 is the data flow for a DB2 source and target:

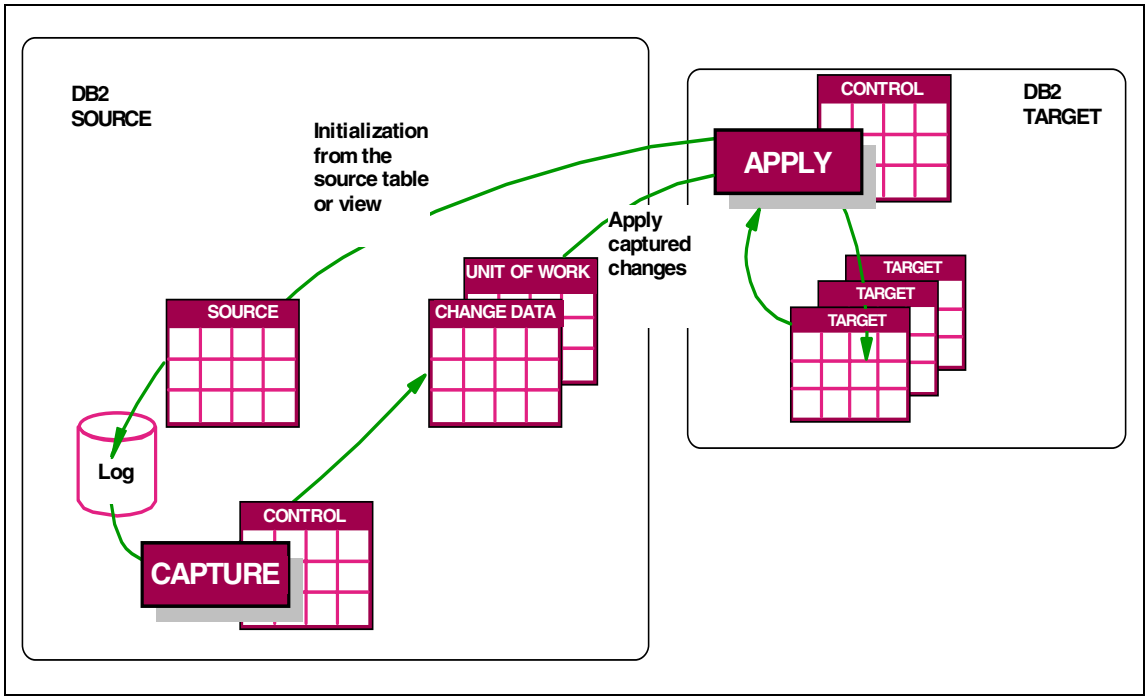


Figure 1-9 Apply for DB2 source and target

The Replication Center is used to map a source table or view to a target table. This is called a *subscription*. You define a *subscription set* which is a group of one or more target tables (called *subscription members*) that will be processed as a unit by Apply. The changes from the CD tables are applied for each table separately, in the same order they occurred at the source. A single COMMIT is issued after the last member in the set is processed. You can specify transactional replication for subscription members that are user copies (read only) or replicas and changes will be applied in the same order that they occurred at the source across all the members in the set. If your target tables have DB2 referential constraints or other relationships that must be preserved, then you should choose transactional replication when defining the subscription set.

Apply selects from the source tables for the first initialization of the target tables, using any transformations or filtering you defined. This is called *full refresh*. Optionally, you can do this step yourself. The Replication Center provides Manual Refresh to update the apply control tables when you do the initial population of the target tables outside of replication.

After the full refresh, Apply selects changes from the CD tables and applies those changes to the target tables. Some target table types may require a join of the CD and the UOW table during the Apply process. The join is not required for read-only copies, used for data distribution and data consolidation. The join is required for bidirectional copies.

Apply can be run as a batch process or as a task that runs all the time. You specify the schedule for replication when you define the subscription set. The schedule can be time-based, on an interval from zero seconds to one year. You can also schedule Apply using an event. You name the event and then insert a row into the Apply events control table whenever you want Apply to start copying.

DB2 source and Informix target

Figure 1-10 is the data flow for a DB2 source copying to an Informix target:

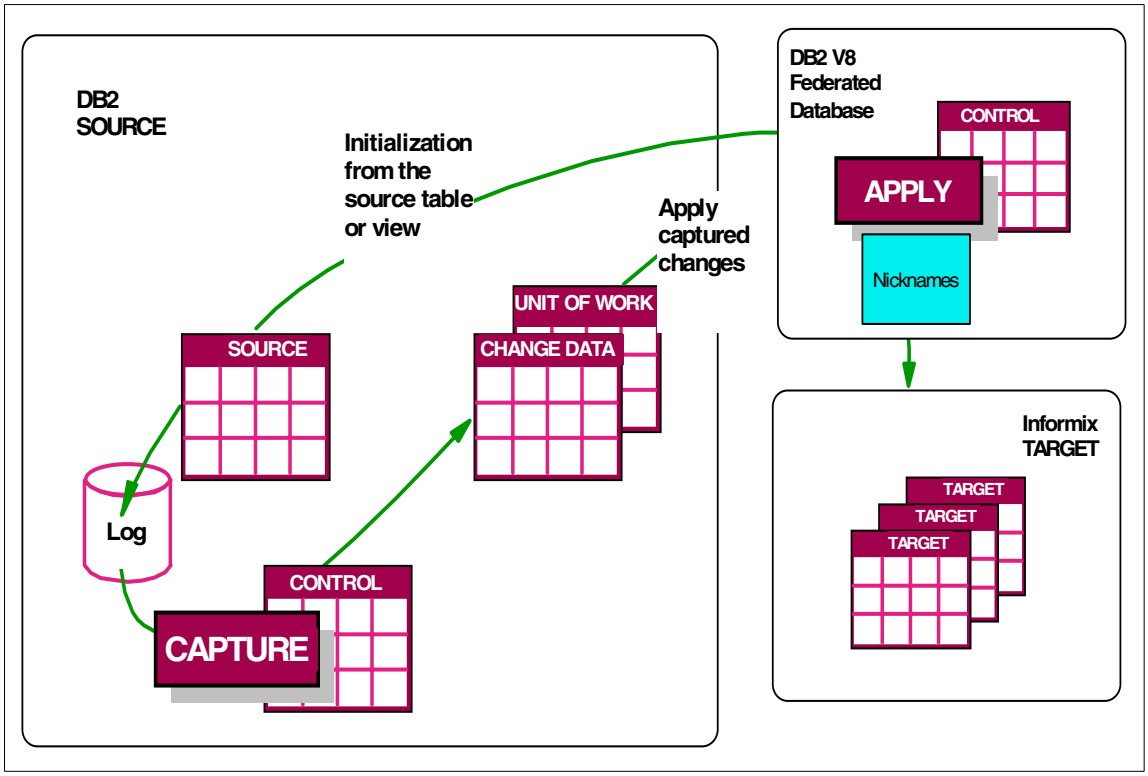


Figure 1-10 Replication from DB2 to Informix

This is the same process that is used for DB2 to DB2 replication. The only difference is that there must be a DB2 V8 federated database with nicknames for the Informix target tables. The Apply programs connects to the DB2 federated database and issues DB2 inserts, updates, deletes, and commits against the nicknames. DB2 V8 translates the DB2 SQL syntax and data types to Informix syntax and data types. The DB2 V8 federated server can be installed on the same system as Informix or on a different server. The Informix SDK Client must be installed on the federated server. If your source server is DB2 V8 for Windows or UNIX, then your source server can also act as the federated server.

Informix source and DB2 target

Figure 1-11 is the data flow for replicating from Informix to DB2:

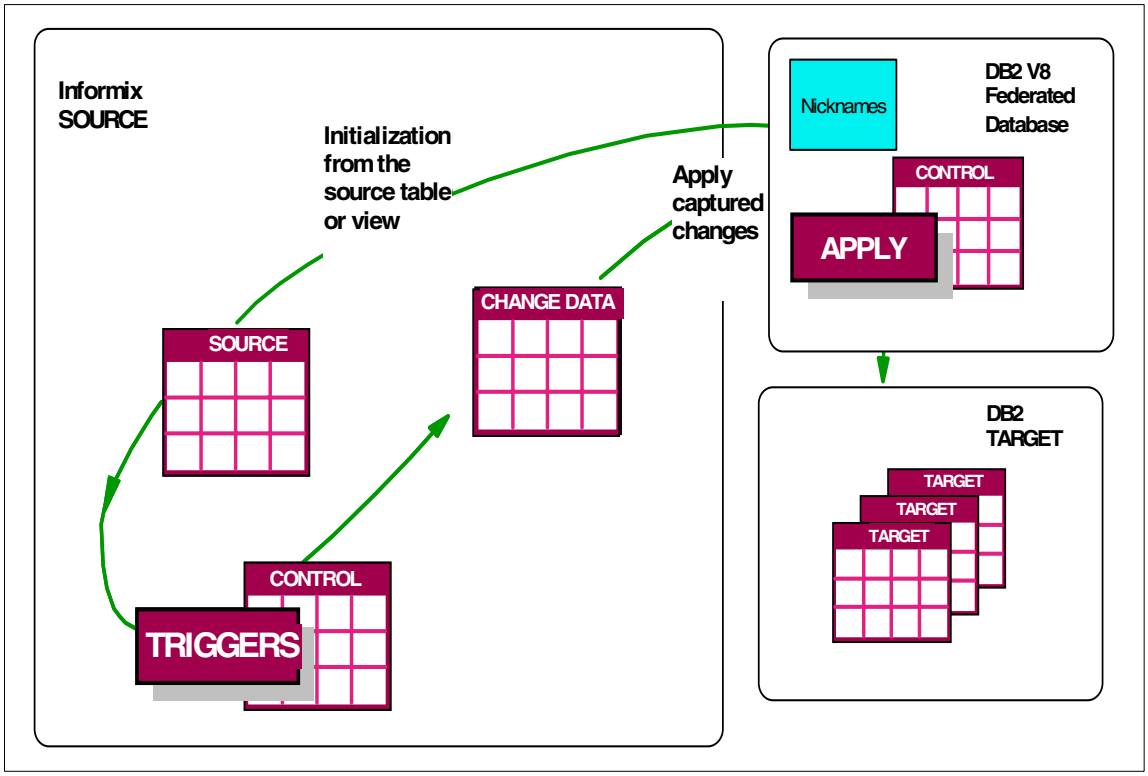


Figure 1-11 Replication from Informix to DB2

This is the same process that is used for DB2 to DB2 replication. The difference is that there must be a DB2 V8 federated database with nicknames for the Informix source, control, and change data tables. The Apply programs connects to the DB2 federated database and issues DB2 selects against the nicknames. DB2 V8 translates the DB2 SQL syntax and data types to Informix syntax and data types so that Apply can process the data against the DB2 target. The DB2 V8 federated server can be installed on the same system as Informix or on a different server. The Informix SDK Client must be installed on the federated server. If your target server is DB2 V8 for Windows or UNIX, then your target server can also act as the federated server.

1.3.4 Alert Monitor

The Replication Alert Monitor is included with DB2 V8 for Windows and UNIX, and with DB2 DataPropagator for z/OS and OS/390. It can be used to monitor replication on those platforms as well as replication on iSeries. The Alert Monitor has its own set of control tables, defined with the Replication Center. Replication administrators define thresholds and events through the Replication Center for Capture and Apply servers. You can also define users or groups of users to receive e-mail notification when an alert occurs. The server where the monitor runs is called the Monitor Server. A monitor server can monitor a single local server or many different servers. The Alert Monitor does not monitor the Capture triggers on Informix source servers.

Figure 1-12 shows an Alert Monitor monitoring two different Capture and Apply servers:

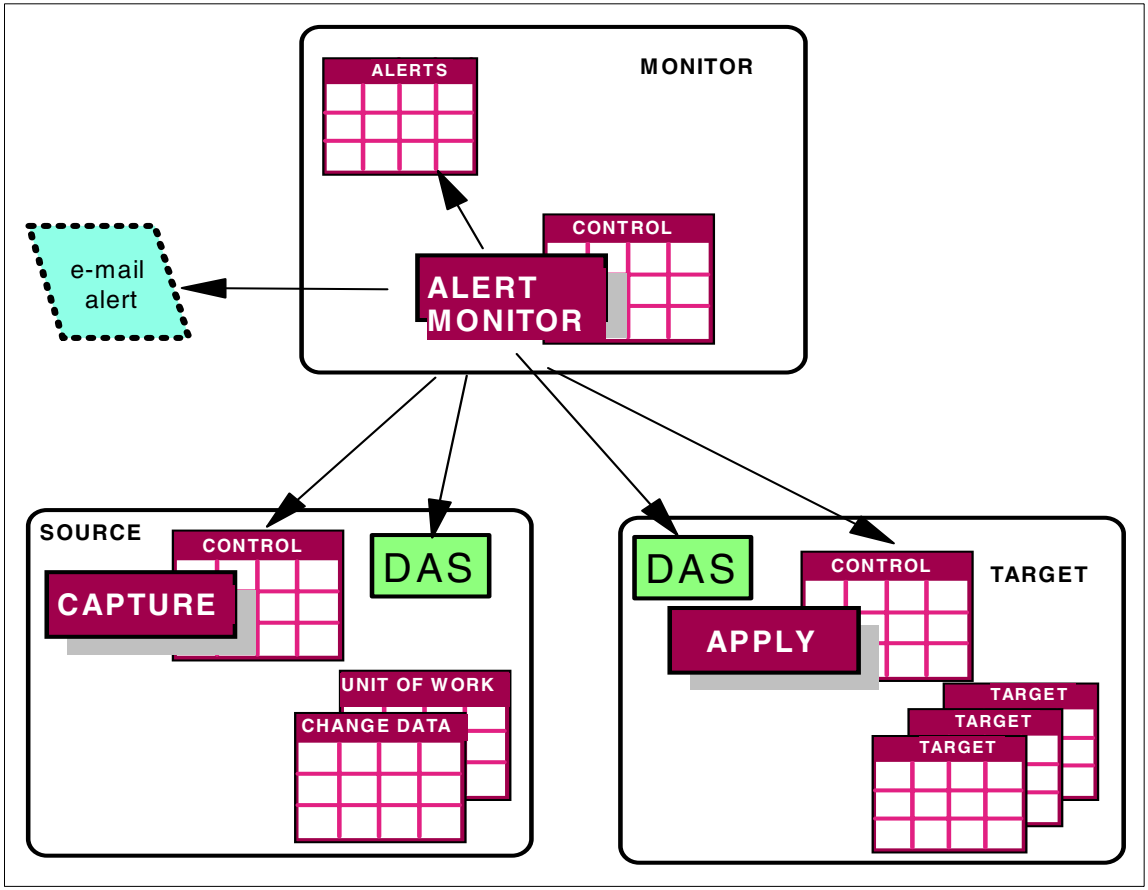


Figure 1-12 Alert Monitor

The Alert Monitor program collects information from the capture and apply control tables. It also uses the Database Administration Server (DAS) installed on the Capture and Apply servers to receive remote commands and supply system information.

DAS for Windows and UNIX is installed when you install DB2 for Windows and UNIX.

The DB2 Administration Server for z/OS will be included with DB2 for z/OS and OS/390 V8. DAS is installed in the Unix System Services shell (USS). DAS V8 will support DB2 for z/OS and OS/390 Versions 7 and 8. DAS is required on the Alert Monitor server and on any DB2 for z/OS and OS/390 capture and apply control server that will be monitored.

DAS is not required on any iSeries Capture and Apply servers that will be monitored.

The monitor interval is set from the Replication Center and controls how often the Alert Monitor checks events and thresholds.

When a monitored event such as an error message occurs or a monitor threshold is exceeded, the Alert Monitor inserts an alert into the ASN.IBMSNAP_ALERTS table and sends e-mail notification to the contacts you have defined.

1.4 Putting the pieces together

DB2 replication is a very flexible solution which supports many different configurations of both products and servers. This section describes some of the most common configurations.

1.4.1 Administration for all scenarios

The Replication Center requires connectivity to source, target, and monitor control servers. The products you need for the Replication Center are:

- DB2 Administration Client for Windows and UNIX V8
- DB2 Connect V8 if any one of your servers is located on z/OS, OS/390, or iSeries

  The DB2 Connect function is included in the DB2 Connect products and in DB2 Enterprise Server Edition V8.
- DB2 Enterprise Server Edition for Windows and UNIX V8 or DB2 Connect V8 if any one of your servers is an Informix server.

  DB2 ESE and DB2 Connect provide federated access to DB2 and Informix data. This federated capability is required for defining Informix sources and Informix targets.
- DB2 Administration Server for z/OS V8

  DAS must be installed on any Capture or Apply z/OS control server if you use the Replication Center for operations on those control servers. Operations include starting and stopping processes and monitoring process.

1.4.2 Data distribution and data consolidation

The data distribution and data consolidation scenarios assume that the target tables are updated only by replication, not by any other users or applications.

Products to capture changes

The type of source server determines what products are needed to capture changes. Remember that Capture always runs on the source server, except when iSeries remote journaling is used. Each source server (or iSeries where changes are remotely journaled) must have a Capture program installed. The source servers with the required products are listed in Table 1-1:

Table 1-1 Products required for capturing changes

Source Server	Required Product to capture changes
DB2 for z/OS and OS/390	DB2 DataPropagator for z/OS and OS/390 installed on the source server
DB2 for iSeries	DataPropagator for iSeries installed on the source server or on the iSeries server with the remote journals
DB2 for Windows and UNIX	Capture program is installed during DB2 installation
Informix Dynamic Server	DB2 for Windows and UNIX V8 or DB2 Connect V8 for federated access to Informix installed on the Informix server or on a separate Windows/UNIX server

Products to apply changes

The Apply program can run on any system in your enterprise, as long as that system can connect to the source, target, and control servers. For the best performance, we recommend that Apply run on the target server so that it can block fetch changes over the network and apply those changes locally. Table 1-2 lists the best choices for source-target combinations:

Table 1-2 Recommended Apply products for applying changes

SOURCE	TARGET	PRODUCT TO APPLY CHANGES
DB2 for z/OS and OS/390 DB2 for iSeries DB2 for Windows & UNIX Informix Dynamic Server	DB2 for z/OS and OS/390	DB2 DataPropagator for z/OS and OS/390 installed on the target server. If the source is Informix, then DB2 for Windows and UNIX V8 or DB2 Connect V8 must be installed on the Informix server or on some other server.
DB2 for z/OS and OS/390 DB2 for iSeries DB2 for Windows & UNIX Informix Dynamic Server	DB2 for iSeries	DataPropagator for iSeries installed on the target server. If the source is Informix, then DB2 for Windows and UNIX V8 or DB2 Connect V8 must be installed on the Informix server or on some other server.
DB2 for z/OS and OS/390 DB2 for iSeries DB2 for Windows & UNIX Informix Dynamic Server	DB2 for Windows and UNIX V8	The Apply program is installed when DB2 for Windows and UNIX is installed. If the source is Informix, then the DB2 for Windows and UNIX V8 target server should also be configured as a federated server.
DB2 for z/OS and OS/390 DB2 for iSeries DB2 for Windows & UNIX Informix Dynamic Server	Informix Dynamic Server	DB2 for Windows and UNIX V8 or DB2 Connect V8 must be installed on the Informix server or on some other server.

1.4.3 Bidirectional with a master (update anywhere)

Update anywhere replication requires that both the master site and each target site (called replicas) have a Capture program running to capture changes. Refer to Table 1-1 to determine which products you need in your environment to provide the Capture program.

In an update anywhere scenario, Apply should be running at each replica to process the changes that are coming from the master and the changes that need to be sent to the master. UseTable 1-1 to determine what products are needed to apply changes in an update anywhere scenario. For example, assume that the master site is DB2 for z/Os and OS/390. There are 20 replicas, all running DB2 for Windows and UNIX V8. You would need to install DB2 DataPropagator for z/OS and OS/390 on the master site. The replicas already have Capture and Apply programs, since they are included in DB2 for Windows and UNIX.

Remember that update anywhere is only valid for the DB2 database family. It is not supported for Informix sources or targets.

1.4.4 Bidirectional with no master (peer to peer)

Peer-to-peer replication requires that all peer sites run a Capture program. Look in Table 1-1 to determine which product you need to provide the Capture program. Each peer site must also have at least one Apply program to copy changes from all the other peers. Table 1-1 and Table 1-2 show the products needed for each type of source server. An example of peer-to-peer is two DB2 for z/Os and OS/390 servers. You would need to install DB2 DataPropagator for z/OS and OS/390 on each server. This product includes both Capture and Apply (and the Alert Monitor), so it would provide all the needed function. If the two servers were DB2 for Windows and UNIX, then no additional products would be needed.

Again, this is only valid for the DB2 database family. You cannot use this scenario with a non-DB2 database, including Informix.

1.4.5 Alert Monitor configuration

The Alert Monitor requires DB2 UDB V8 for Windows and UNIX or DB2 for z/OS and OS/390. The Alert Monitor program uses two methods to collect replication information:

- DB2 SQL against capture and apply control tables — DRDA communication support is included in the database products required by the Alert Monitor and in all the database products where Capture and Apply run.

- Database Administration Server (DAS) — DAS must be running on each server that will be monitored. DAS is included with DB2 V8 for Windows and UNIX. DASe for z/OS and OS/390 will be available in a future version of DB2 for z/OS and OS/390. Once this version is available, you can use the DASe to monitor replication on both the new version of DB2 and the previous version of DB2 (DB2 for z/OS and OS/390 V7).

The products required for the Alert Monitor Server are listed in Table 1-3:

Table 1-3 Alert Monitor Server requirements

Alert Monitor Server Location	Required Product
Windows or UNIX	DB2 Universal Database for Windows and UNIX V8
z/OS or OS/390 (running in the UNIX System Services shell)	DB2 DataPropagator for z/OS and OS/390 V8 and DB2 Administration Server for z/OS V8

The products required on a Capture or Apply server to be monitored for alerts are listed in Table 1-4:

Table 1-4 Monitored servers requirements

Capture or Apply Server Location	Required Products
DB2 Universal Database for Windows and UNIX V8	None, DAS is included with the product
DB2 Universal Database for z/OS or OS/390 V7 or higher	DB2 DataPropagator for z/OS and OS/390 V8 and DB2 Administration Server for z/OS V8
DB2 for iSeries V5R2	None, DAS is not used for iSeries monitoring

1.5 DB2 Replication V8 close up

This section is a closer look at how the different components work together. The chapters that follow will give you detailed information on each step in the replication process.

Replication tasks are:

1. Defining database servers that will be replication sources and targets
2. Defining replication source tables
3. Defining replication subscriptions
4. Operating Capture and Apply
5. Monitoring Capture and Apply

1.5.1 Administration — defining a replication scenario

For detailed information on these tasks, refer to:

- Chapter 2, “Getting started with Replication Center” on page lxxv
- Chapter 3, “Replication control tables” on page cxli
- Chapter 4, “Replication Sources” on page clxix
- Chapter 5, “Subscription Set” on page cciii

Defining database servers as replication sources and targets

A database server is a DB2 for z/OS subsystem or data sharing group, a DB2 for iSeries database, a DB2 for Windows and Unix database, or an Informix Dynamic Server database. In this section, all of these will be referred to as *databases* or *servers*. A server is a *federated* database if it contains *nicknames* which point to non-DB2 tables. Replication which includes Informix requires a federated database.

A database that will act as a replication source is referred to as a *capture control server*. You use the Replication Center to define a capture control server by creating the capture control tables in the database. There can be multiple sets of capture control tables, each with a different *capture schema* that you specify when creating the control tables. Each replication source and subscription and each Capture program is associated with a specific capture schema. There is one global table, *ASN.IBMSNAP_CAPSCHEMAS*, which has an entry for each schema you define within a database. There are twelve capture control tables for DB2 capture control servers.

If the source server is an iSeries database, then there are two additional capture control tables.

If the source server is an Informix database, then there are 6 capture control tables created in the Informix database, with corresponding nicknames in the DB2 V8 federated database. The ASN.IBMSNAP_CAPSCHEMAS table is created in the DB2 V8 federated database.

A database that will act as a replication target is called a *target server*. The control tables for Apply can be located on the capture control server, on the target server, or any other server in your enterprise that can be accessed by the Apply program. The database where the apply control tables are located is called the *apply control server*. Apply control tables always have the schema *ASN*. There are 10 apply control tables.

A database that will be used to monitor replication is a *monitor server*. The monitor control tables are defined in the monitor server database using the replication center. The monitor server can be located anywhere in your enterprise and must have access to the capture and apply control servers that you wish to monitor. There are 9 monitor control tables and they always have the schema *ASN*.

All the control tables are described 3 of this book and in detail in the *IBM DB2 Universal Database Replication Guide and Reference Version 8, SC27-1121-00.*

Defining replication source tables

This task is called *registering* a source table. The capture control tables must already exist in the database where the source table is located (or, if using remote journaling, in the iSeries where the remote journal is located). You select a source table from a list of tables in the source database and a capture schema. You can choose to register a source table for full refresh only or for change capture. Other options are described in Chapter 4, “Replication Sources” on page clxix of this book.

A DB2 iSeries source table must be *journaled* before it can be registered. DB2 for iSeries does not have a global DB2 log. Each table can optionally have a journal to log changes to the table. The journal is required for DB2 iSeries tables that will be registered as replication source tables.

These actions occur when you register a DB2 source table:

- If the source table is not already defined with DATA CAPTURE CHANGES, the source table is altered to have that attribute. This enables full-row logging for the source table and allows Capture to capture the before and after images of changes to that table.
- A change data table, called a *CD table*, is created to hold the captured changes. The CD table has the columns you selected when registering the table. The CD table columns for after-image values have the same names

and data types as the source table. If you choose to capture the before-image values, the CD table columns have the column names of the source prefixed with a character you choose. The default before-image prefix is X. There are also control columns in the CD table to identify the type of change and the log record sequence number of the change.

- A row is inserted into the capture control table *capschema*.IBMSNAP_REGISTER with information about the registration, including the name of the CD table. The STATE column in this row is set to I. New registrations are always inactive until a subscription is processed against them.
- If the source table is in an iSeries database then the name of the journal for that source table is inserted into the *capschema*.IBMSNAP_REG_EXT table at the capture control center.

These actions occur when you register an Informix source table:

- A consistent change data table, called a *CCD table*, is created to hold the captured changes. The CCD table has the columns you selected when registering the table. The CCD table columns for after-image values have the same names and data types as the source table. If you choose to capture the before-image values, the CCD table columns have the column names of the source prefixed with a character you choose. The default before-image prefix is X. The CCD table has an operation column to identify the type of change and a sequence column.
- Three triggers and three associated stored procedures are created to capture the changes. There is one trigger/procedure pair each for inserts, updates, and deletes to the source table. The stored procedures generate a unique sequence number for each change and insert changed values into the CCD table.
- A trigger and stored procedure are created on the capture control table, cap_schema.IBMSNAP_PRUNCNTL. If the trigger and stored procedure already exist, they are modified to include this new registration. For DB2 sources, the Capture program deletes (*prunes*) CD table rows after they have been applied. For non-DB2 sources, pruning is done via these triggers. Informix triggers are limited in size to 256 characters, so stored procedures are defined and called by the triggers.

Defining replication subscriptions

This task creates a *subscription set*, which is one or more source table to target table mappings. The source to target mappings are called *subscription members*. A subscription set has only one source server and one target server. Each subscription set has an *apply qualifier* associated with it. An Apply program is started with an apply qualifier and that Apply program will process all the sets

with a matching apply qualifier. An apply qualifier can have many sets. Each set has zero or more members. More detail is available in Chapter 5, “Subscription Set” on page cciii of this book.

Defining a subscription set

When you create a subscription set, you specify the apply control server, the apply qualifier, the set name, the capture control server and schema, the target server and scheduling information. The set can be scheduled on a time interval and/or when an *event* occurs. If you choose event-based replication, then you specify an event name for the subscription set. Rows in the ASN.IBMSNAP_SUBS_EVENTS table determine the start and end point for an event. You insert into this table the event-name, start time for Apply processing and optionally the end time for the changes processed.

You can create an empty subscription set and add members later or create the set and members at the same time.

This is what takes place when you define a subscription set:

- A row is inserted into the apply control table ASN.IBMSNAP_SUBS_SET.
- If the source server is an Informix database, then special SQL statements are inserted into the ASN.IBMSNAP_SUBS_STMTS table to control Apply processing.

Defining subscription members

When you add a member to a subscription set, you specify the source and target table, the mappings from source columns to target columns, the unique target table columns for the target table index, and *row filters* (predicates).

There are 6 target table types:

- User copy — a complete or partial copy of a source table, optionally with transformed columns or new columns
- Point-in-time — a user copy with an IBMSNAP_LOGMARKER column that has the timestamp when each row was changed at the source server
- Consistent change data (CCD) — a complete or partial copy of the changes to a source table

  A CCD target table includes columns for the type of change (insert, update, delete), the log record sequence number of the change, the log record sequence number of the commit for that change, and the timestamp of that commit. A CCD can be:

  – Complete — initialized with a CCD row for each row in the source table
  – Non-complete — no initialization, only a CCD row for each change

  - Condensed — each CCD row represents the last change to a source table row
  - Non-condensed — one CCD row for each change to a source table row

  You can also choose to store control information, such as the authorization id that made the change, in the CCD table.

- Replica — an updateable copy of all or a portion of a source table

  Replicas are used for update anywhere replication. Only DB2 tables can be replicas; this target type cannot be chosen for Informix target tables.

- Base aggregate — an aggregate of a source table based on SQL column functions and GROUP BY filters that you define

  Apply issues a select against the source table each time it processes a base aggregate target table and inserts new rows in the target table.

- Change aggregate — an aggregate of the changes to a source table based on SQL column functions and GROUP BY filters that you define

  Apply issues a select against the CD table each time it processes a change aggregate table and inserts new rows in the target table.

This is what takes place when you define a subscription member:

- If this is the first member added to a set, a row is inserted into the ASN.IBMSNAP_PRUNE_SET table at the capture control server with set information. This information is used when pruning change data after it has been applied.
- One row is inserted into the ASN.IBMSNAP_PRUNCNTL table at the capture control server for each subscription member. Each member is assigned a MAP_ID number which uniquely identifies that subscription member. The MAP_ID starts at 0 and increases by 1 for every subscription to this capture control server. It is stored in the MAP_ID column of ASN.IBMSNAP_PRUNCTL.
- One row is inserted into the ASN.IBMSNAP_SUBS_MEMBR table at the apply control server. This row has the apply qualifier, set name, and source and target table names. This is the source table to target table mapping. The MEMBER_STATE for each member is N for new. The PREDICATES value is the row filter that you specified for this mapping.

Note: The PREDICATES row filter is used when Apply initializes the target table (select from the source table) and when Apply processes changes (select from the CD table). You can specify a predicate to be used only when Apply processes changes by updating the UOW_CD_PREDICATES column of ASN.IBMSNAP_SUBS_MEMBR. You would do this if you wished to filter changes based on a value in the CD or UOW table. For instance, if you wished to block the replication of deletes, you would specify IBMSNAP_OPERATION <> 'D' for the UOW_CD_PREDICATES.

UOW_CD_PREDICATES cannot be set from the Replication Center. You must update the ASN.IBMSNAP_SUBS_MEMBR table manually while connected to the apply control center.

- One row for each target table column is inserted into the ASN.IBMSNAP_SUBS_COLS table at the apply control server. Each row is a mapping from a source column or SQL expression to a target table column. Also included is a flag to identify target table columns that are part of the target table primary key or unique index. This is the source table column to target table column mapping.
- If the source server is an Informix database, then special SQL statements are inserted into the ASN.IBMSNAP_SUBS_STMTS table to control Apply processing. If you defined SQL statements that you want to run before or after each Apply processing cycle, they are inserted into ASN.IBMSNAP_SUBS_STMTS.
- If the target table does not already exist, it is created, along with a unique index. If the target server is an Informix database, a nickname is created for the target table.
- If the target table is a *replica* (Update Anywhere), then there is extra processing to define the replication from the replica to the source server:
  - A second set is created in ASN.IBMSNAP_SUBS_SET, and rows inserted into ASN.IBMSNAP_SUBS_MEMBR and ASN.IBMSNAP_SUBS_COLS.
  - The replica is registered as a replication source at the apply control server.

1.5.2 Operations — DB2 Capture and Apply

Chapter 6, "Operating Capture and Apply" on page cclxv provides detailed information by platform about operating Capture and Apply. In this section, we will discuss what Capture and Apply do when processing changes for a read only target table when the source is DB2.

Both Capture and Apply are **multi-threaded** applications.

Capture has five threads:

- INIT thread sets the environment and starts the other threads.
- ADMIN thread manages administrative tasks, including error messages and traces, and stores monitoring statistics at the capture control server.
- HOLDL thread holds an exclusive lock on *capschema*.IBMSNAP_CAPENQ to prevent a second instance of Capture with the same capture schema and capture control server.
- WORKER thread requests log records from DB2, manages log information and inserts rows in CD tables and the UOW table. This thread also updates the capture control tables.
- PRUNE thread deletes (prunes) rows from the CD tables and UOW tables after they have been applied. This thread also prunes *capschema*.IBMSNAP_TRACE, *capschema*.IBMSNAP_CAPMON, and *capschema*.IBMSNAP_SIGNAL, based on limits you set for Capture startup.

Apply has three threads:

- ADMIN thread manages administrative tasks.
- HOLDL thread holds exclusive lock on the ASN.IBMSNAP_APPENQ row with a matching apply qualifier to prevent a second instance of Apply with the same apply control server and apply qualifier.
- WORKER thread processes subscription sets based on the apply control server and apply qualifier specified when Apply is started. This thread accesses capture control tables, source tables, CD tables, UOW tables, apply control tables, and target tables.

You can use the Query Status option on the Replication Center to show the status of each of the Capture and Apply threads.

The cornerstone of DB2 replication is the log record sequence number, the *LSN*. The LSN for each change is stored in the CD table row for that change, along with the LSN of the commit statement for that change's unit of work. These values are used to ensure that changes are replicated in the order they occurred at the source. Capture and apply control tables have columns named *SYNCHPOINT*, which are used as progress indicators to control replication and restart processing.

Recovery logging must be enabled for a DB2 for Windows and UNIX capture control server before Capture is started the first time.

Capture initializes global information

You start Capture with a capture schema and capture control server name. The first time Capture runs for a capture schema/capture control server combination, it inserts a row into *capschema*.IBMSNAP_REGISTER with the GLOBAL_RECORD column set to Y and the SYNCHPOINT/SYNCHTIME columns set to the current point in the DB2 log. Capture reads the *capschema*.IBMSNAP_REGISTER table to find registrations. A registration is not active until Apply has signalled that a full refresh has been done.

Note: You must start Capture at least once to initialize the global record values in the ASN.IBMSNAP_REGISTER table. Once that is done, all Capture and Apply communications are done through the ASN.IBMSNAP_SIGNAL table. Capture does not need to be running when a subscription is defined or when Apply is started.

Also, you can add new registrations without re-initializing Capture. When Apply runs for the first time using the new registration, Capture will dynamically load the registration information into memory.

Apply full refreshes the target table

Apply is started with an apply qualifier and the name of the apply control server. Apply connects to the apply control server and selects rows from ASN.IBMSNAP_SUBS_SET which have a matching apply qualifier. Apply ignores any sets which are not active (ACTIVATE column is 0). A new set has SYNCHPOINT and SYNCHTIME and LASTSUCCESS values of NULL in ASN.IBMSNAP_SUBS_SET. This tells Apply that a full refresh is needed.

Apply selects the member information from ASN.IBMSNAP_SUBS_MEMBR at the apply control server and updates the *capschema*.IBMSNAP_PRUNCNTL at the capture control server with the SYNCHPOINT column = x'00000000000000000000' and the SYNCHTIME column = CURRENT TIMESTAMP for each member.

Then, Apply inserts one row into *capschema*.IBMSNAP_SIGNAL at the capture control server for each member in the set. The value for SIGNAL_TYPE is CMD, for SIGNAL_SUBTYPE is CAPSTART and for SIGNAL_INPUT_IN is the MAP_ID of the subscription member from the *capschema*.IBMSNAP_PRUNCNTL table at the capture control server.

A user exit program, ASNLOAD is supplied with Apply. This program, which can be modified if you wish, uses DB2 export and import (or load) commands to do the initial full refresh for better performance. The Apply parameter LOADXIT controls whether or not the ASNLOAD program is used.

If you start Apply with the parameter LOADXIT set to N, then Apply selects rows from the source table or view. This select includes any transformations you have defined with SQL expressions and any row filters you specified for each member. Apply stores the result set in a *spill file* at the server where Apply was started. There is one spill file for each member in the subscription set. Apply issues a DELETE against the target table to delete existing rows and then INSERTs to the target table using the data from the spill file.

If you start Apply with the parameter LOADXIT set to Y, then Apply calls the ASNLOAD user exit. Apply passes control information to the exit, including the select with the transformations and row filters. There is a sample ASNLOAD exit shipped with Apply. ASNLOAD can call native unload and load utilities to improve the performance of the full refresh.

Another alternative is to do the full refresh yourself, but you must use the Replication Center “Full Refresh > Manual” option to ensure that the capture and apply control tables are updated correctly.

Apply then updates the LASTSUCCESS and SYNCHTIME columns for this set in ASN.IBMSNAP_SUBS_SET and changes the MEMBER_STATE for each member in ASN.IBMSNAP_SUBS_MEMBR to L to indicated that the target tables have been loaded.

Capture starts capturing changes

Capture requests log records from DB2. When Capture detects the insert that Apply executed for the *capschema*.IBMSNAP_SIGNAL table, it begins to capture changes:

Capture takes the log information for the *capschema*.IBMSNAP_SIGNAL insert and:

- Matches the SIGNAL_INPUT_IN value from the log record with a MAP_ID in *capschema*.IBMSNAP_PRUNCNTL.
- Updates the matched row in PRUNCNTL with SYNCHPOINT set to the LSN of the SIGNAL insert and SYNCHTIME set to the timestamp of the SIGNAL insert.
- Retrieves the registration information from the *capschema*.IBMSNAP_REGISTER if it is not already loaded in memory.
- Updates the CD_OLD_SYNCHPOINT with the LSN of the SIGNAL insert and sets the STATE to A (active) for this registration.

Capture uses the LSN of the SIGNAL insert as the starting point for capturing changes for this source table. Note that Capture starts from a point in the DB2

log BEFORE Apply does the select for a full refresh. This ensures that no changes are missed during the initialization of the target tables.

When Capture receives a log record with a change for a registered table, it stores the log record in memory. Each log record is associated with a DB2 unit of work and carries the unit of work identifier. When Capture receives a log record for a commit of a unit of work involving a registered table, it inserts the change information into the appropriate CD tables and inserts the commit information into the UOW table. If a rollback is received, Capture removes the associated changes from memory.

Capture also updates capture control tables to indicate progress and to save restart information. Capture issues commits of this information based on the COMMIT_INTERVAL parameter specified when Capture is started.These tables are updated:

- *capschema*.IBMSNAP_RESTART holds the LSNs for a Capture restart point in the DB2 log.
- *capschema*.IBMSNAP_REGISTER rows are updated with the LSN of the most recently inserted UOW table row (CD_NEW_SYNCHPOINT) for each source table that had change activity.
- *capschema*.IBMSNAP_REGISTER global_record row is updated with the LSN and timestamp of the most recently inserted UOW table row.

Apply processes changes

Apply checks for active sets (ACTIVATE =1) in ASN.IBMSNAP_SUBS_SET at the apply control server. Apply then reviews the active sets to see if any are eligible for processing. A set is eligible if the sleep_minutes plus the timestamp value of LASTRUN is greater than the current time or if the EVENT_NAME has been posted in the ASN.IBMSNAP_SUBS_EVENT table.

If a set is eligible, then Apply does the following:

- Reads the ASN.IBMSNAP_SUBS_STMTS table and issues any SQL statements that were defined for execution BEFORE Apply runs.
- Reads the member and column mapping information from the ASN.IBMSNAP_SUBS_MEMBR and ASN.IBMSNAP_SUBS_COLS tables.
- Sets the *lower bound* for changes equal to the SYNCHPOINT in the ASN.IBMSNAP_SUBS_SET table.
- Connects to the capture control server and checks the global SYNCHPOINT in the *capschema*.IBMSNAP_REGISTER. This LSN is the *upper bound* for changes. If the lower and upper bounds are equal, then there are no changes to process.

- Selects changes for each member from the CD tables between the lower and upper bounds. The select includes the SQL transformations and row filters defined for the member.

  Note: If the target table type is not user copy, then this select is a join of the CD table and the UOW table. You can force the join for a user copy by setting the JOIN_UOW_CD column in ASN.IBMSNAP_SUBS_MEMBR to Y. You do this if you wish to copy one or more of the values (authid is an example) from the UOW table to your target table. You would also do this if you have a row filter which refers to a column in the UOW table.

  JOIN_UOW_CD cannot be set using the Replication Center. You must issue the SQL directly while connected to the apply control server.

- Stores the result set in a spill file on the server where Apply was started. There is one spill file for each subscription member.
- Reads the spill files and issues inserts, updates, deletes against the target tables
  - If the target table type is a replica or a user copy and you chose transactional processing, then Apply reads all of the spill files and applies each unit of work in the same order it occurred at the source server. Apply will issue commits after processing each x units of work, where x is the number you specified when defining the subscription set.
  - If the target table type is not replica or user copy, or if the target table type is user copy and you did not choose transactional processing when defining the subscription set, then Apply will process each spill file separately and issue one commit at the end of all processing.
  - Changes may need to be reworked when issuing inserts, updates, and deletes. This is sometimes called upsert. Rework modifies the operation of a change to ensure that it can be processed. Table 1-5 is a list of the rework rules:

Table 1-5 Apply rework rules

Source change	Target table row	Change reworked to
insert	already exists	update
update	does not exist	insert
delete	does not exist	null — change is ignored

- Executes any SQL statements from the ASN.IBMSNAP_SUBS_STMTS which are marked to be run AFTER Apply processing.

- Updates the ASN.IBMSNAP_SUBS_SET SYNCHPOINT and SYNCHTIME columns for this set at the apply control server with the LSN of the upper bound and the timestamp of the upper bound. The MEMBER_STATEs in ASN.IBMSNAP_MEMBR for all members of the set are set to S.
- Updates the *capschema*.IBMSNAP_PRUNE_SET SYNCHPOINT column for this set at the capture control server with the upper bound LSN.
- Inserts an audit row into ASN.IBMSNAP_APPLYTRAIL at the apply control server.

Capture prunes applied changes

The Capture PRUNE thread is controlled by the PRUNE_INTERVAL and AUTOPRUNE parameters specified when Capture is started. If AUTOPRUNE is NO, then no pruning takes place unless you issue the prune command through the Replication Center or with the **asnccmd** command. If AUTOPRUNE is YES, then pruning will occur every x seconds, where x is the value assigned to PRUNE_INTERVAL. Pruning is independent of the Capture worker thread.

Apply updates the *capschema*.IBMSNAP_PRUNE_SET table SYNCHPOINT with the LSN of the point in the DB2 log where changes were last selected and applied.

The PRUNE thread deletes from the CD tables and UOW tables based on these values. If there are multiple subscriptions to the same source table, then the lowest value is used to ensure that no change is deleted before it has been applied. Pruning is cursor-based, with interim commits and does not require a join of the CD and UOW tables. The *capschema*.IBMSNAP_REGISTER CD_OLD_SYNCHPOINT value is updated after pruning.

1.5.3 Operations — Informix Capture and Apply

Chapter 6, “Operating Capture and Apply” on page cclxv provides detailed information by platform about operating Capture and Apply. In this section, we will discuss what Capture and Apply do when processing changes for a read only target table when the source is Informix.

When you create Capture control tables for an Informix server, you specify a capture schema which is stored in the ASN.IBMSNAP_CAPSCHEMAS table at the DB2 V8 federated database. All other control tables are created in Informix using the remote schema that you specify. Nicknames are created for these control table in the DB2 V8 federated database so that Apply can connect to the federated database and access the Informix source server as though it were DB2.

The Informix capture control tables are:

- *remoteschema*.IBMSNAP_REGISTER
- *remoteschema*.IBMSNAP_PRUNCNTL
- *remoteschema*.IBMSNAP_PRUNESET
- *remoteschema*.IBMSNAP_SIGNAL
- *remoteschema*.IBMSNAP_REG_SYNCH
- *remoteschema*.IBMSNAP_SEQTABLE

Triggers and associated stored procedures are created on the Informix server to simulate DB2 Capture updates to capture control tables.

Changes to Informix source tables are captured by triggers on the source tables. There is an insert, update, and delete trigger on each source table and each trigger has a corresponding stored procedure.

Apply uses the same threads to replicate from Informix source tables that are used for replicating from DB2 tables.

You can use the Query Status option on the Replication Center to show the status of each of Apply threads. but there is no equivalent for Informix capture triggers and stored procedures.

The Informix capture stored procedures generate a unique sequence number for each change, which is used in place of the DB2 LSN. These values are used to ensure that changes are replicated in the order they occurred at the source. Capture and apply control tables have columns named *SYNCHPOINT*, which are used as progress indicators to control replication and restart processing.

Capture triggers begin capturing

The Informix capture triggers are active as soon as they are defined, so changes are captured immediately after you register an Informix source table. Part of the registration process is the insertion of a row into the *capschema*.IBMSNAP_REGISTER table. If the source is Informix, then this is a nickname which points to an Informix table named *remoteschema*.IBMSNAP_REGISTER. You specify the *remoteschema* when you register the table. Unlike the registration row for a DB2 table, the row for an Informix source table has values for the Informix CCD — CCD_OWNER, CCD_TABLE, and CCD_OLD_SYNCHPOINT.

Apply full refreshes the target table

Apply processing of the target table full refresh is the same, but there are differences in the capture control server processing. Apply does these steps when copying from an Informix source server

- Inserts a signal into the *remoteschema*.IBMSNAP_SIGNAL table. A trigger/stored procedure on the *remoteschema*.IBMSNAP_SIGNAL table sets

the SYNCHPOINT in *remoteschema*.IBMSNAP_PRUNCNTL to hex zeroes to indicate a full refresh is being started.

- Executes the SQL statement from ASN.IBMSNAP_SUBS_STMTS that was inserted when defining the subscription. The statement is an update to the *remoteschema*.IBMSNAP_REG_SYNCH table. This table has a trigger/stored procedure which selects from the most current sequence value from the *remoteschema*.IBMSNAP_SEQTABLE and updates the *remoteschema*.IBMSNAP_REGISTER SYNCHPOINT and SYNCHTIME columns. This is the beginning sequence number for Apply.
- Selects data from the source table based on your source to target mapping.
- Deletes all rows from the target table and inserts the new rows.
- Updates ASN.IBMSNAP_SUBS_SET.

Apply processes changes

Apply checks for active sets (ACTIVATE =1) in ASN.IBMSNAP_SUBS_SET at the apply control server. Apply then reviews the active sets to see if any are eligible for processing. A set is eligible if the sleep_minutes plus the timestamp value of LASTRUN is greater than the current time or if the EVENT_NAME has been posted in the ASN.IBMSNAP_SUBS_EVENT table.

If a set is eligible, then Apply does the following:

- Reads the ASN.IBMSNAP_SUBS_STMTS table and issues an SQL statement at the Informix source server. The following takes place:
  - Apply connects to the DB2 V8 federated database and issues UPDATE *remote_schema*.IBMSNAP_REG_SYNCH SET TRIGGER_ME = 'Y'.
  - The remote_schema.IBMSNAP_REG_SYNCH table has a trigger which calls a stored procedure which updates remote_schema.IBMSNAP_REGISTER SYNCHPOINT and SYNCHTIME values with the current sequence number. This will be the *upper bound* for Apply change processing.
- Reads the member and column mapping information from the ASN.IBMSNAP_SUBS_MEMBR and ASN.IBMSNAP_SUBS_COLS tables.
- Sets the *lower bound* for changes equal to the SYNCHPOINT in the ASN.IBMSNAP_SUBS_SET table.
- Selects changes for each member from the CCD tables between the lower and upper bounds. The select includes the SQL transformations and row filters defined for the member.
- Stores the result set in a spill file on the server where Apply was started. There is one spill file for each subscription member.

- Reads the spill files and issues inserts, updates, deletes against the target tables Changes may need to be reworked when issuing inserts, updates, and deletes.
- Executes any SQL statements from the ASN.IBMSNAP_SUBS_STMTS which are marked to be run AFTER Apply processing.
- Updates the ASN.IBMSNAP_SUBS_SET SYNCHPOINT and SYNCHTIME columns for this set at the apply control server with the LSN of the upper bound and the timestamp of the upper bound. The MEMBER_STATEs in ASN.IBMSNAP_MEMBR for all members of the set are set to S.
- Updates the *capschema*.IBMSNAP_PRUNE_SET SYNCHPOINT column for this set at the capture control server with the upper bound LSN.
- Inserts an audit row into ASN.IBMSNAP_APPLYTRAIL at the apply control server.

Capture prunes applied changes

Informix prune triggers on the *remoteschema*.IBMSNAP_PRUNCNTL handle the deletion of rows from the CCD tables after they are applied.

Apply updates the *remoteschema*.IBMSNAP_PRUNE_SET and *remoteschema*.IBMSNAP_PRUNCNTL tables SYNCHPOINT columns with the sequence number of the last change successfully applied.

The prune trigger deletes from the CCD tables based on these values. If there are multiple subscriptions to the same source table, then the lowest value is used to ensure that no change is deleted before it has been applied.

1.5.4 Administration and operations — Alert Monitor

The Alert Monitor architecture has similarities with Capture and Apply. It has its own set of control tables and start-up parameters. A monitor server is a DB2 for Windows or Unix database or a DB2 for z/os and OS/390 subsystem or data sharing group which contains the monitor control tables.

Chapter 7, “Troubleshooting and monitoring” on page cccxxxvii in this book covers monitoring and troubleshooting.

Administration

You identify a monitor server by creating the monitor control tables in that server using the Replication Center. Each instance of the Alert Monitor program is started with a *monitor qualifier* that you define. Under the monitor qualifier, you define *alert conditions* for the Capture and apply control servers you wish to monitor. Alert conditions are defined for capture schemas, apply qualifiers, and

subscription. For each alert condition, you specify a *contact* or *contact group* that should be notified if the condition is met. A contact is an e-mail address. A contact group is just that, a set of contacts. The monitor qualifier, alert conditions, and contact information are created using the Replication Center and stored in monitor control tables.

The Alert Monitor can check for:

- Status of Capture and Apply programs
- Error and warning messages
- Latency thresholds
- Memory usage
- Subscription set failures or full refreshes
- Transactions rejected due to update anywhere conflicts
- Transactions reworked by Apply

Operations

You can have several Alert Monitor programs running on the same or different systems, each monitoring a different set of Capture and Apply servers. The Alert Monitor program must be able to connect to the monitor control server and all monitored capture/apply control servers.

The monitor_interval is the number of seconds in a monitor cycle and is specified when the Alert Monitor is started. The Alert Monitor checks for alert conditions by selecting values from capture and apply control tables and issuing system commands to the DAS running on the capture or apply server.

If the Alert Monitor detects an alert condition, then an e-mail message describing the condition is sent to the contact or contact group defined in the monitor control tables and the alert is inserted into the ASN.IBMSNAP_ALERTS table at the monitor control server.

The MAX_NOTIFICATIONS_PER_ALERT startup parameter can be used to prevent flooding the contacts with alerts for the same problem. The MAX_NOTIFICATIONS_MINUTES startup parameter is used to control the number of minutes between notifications for the same alert.

The ASN.IBMSNAP_ALERTS table is pruned by the Alert Monitor at startup, based on the ALERT_PRUNE_LIMIT startup parameter.

1.6 What's new in DB2 Replication V8

This section is a summary of the DB2 Replication V8 enhancements. A more detailed discussion is available in the *IBM DB2 Universal Database Replication Guide and Reference Version 8, SC27-1121-00*.:

1.6.1 Administration

The improvements in administration are:

- The new Replication Center combines and extends the administrative functions found in the previous tools — DataJoiner Replication Administration (DJRA) and replication functions in the Control Center. It is a graphical user interface for defining replication scenarios and managing and monitoring replication processes. Appendix A has comparisons between the Replication Center and DJRA and between the Replication Center and the Control Center.
- Source and target table names can be up to 128 characters and column names up to 30 characters, subject to any limits imposed by the database source, target, or control server.
- Replication definitions can be added or changed without re initializing the Capture program.
- New columns can be added to replication sources while Capture is running
- A migration utility is included to convert existing DB2 Replication V5, V6, and V7 environments to DB2 Replication V8.
- All stored passwords are encrypted.

1.6.2 Capture

Capture enhancements are:

- The new IBMSNAP_SIGNAL table is used for improved communications between Capture and Apply. Once Capture has been started successfully the first time, there is no longer a requirement to always start Capture before starting Apply, or that Capture must be running before Apply processes a new subscription.
- Changes are not inserted in change data and UOW tables until they have been committed. A join of the CD and UOW table is no longer required for user copies. Changes which are rolled back are no longer put in the CD table.
- Greater control over the changes captured has been added:

  - If you choose only a subset of source columns when defining a replication source, you can specify that no changes should be captured for that source unless the change affects your selected columns.
  - When defining a replica, you can specify that the changes processed by Apply from the master site should not be recaptured at the replica site.
- Multiple Capture programs can run on the same DB2 database, subsystem, or data sharing group:
  - Each Capture has its own schema, control tables and change data tables. A Capture program is uniquely defined by the combination of source server and capture schema. You can use multiple Captures to improve throughput.
  - This also allows you to define multiple non-DB2 replication sources in a single federated database.
- Capture prunes changes concurrently with the capture of changes on DB2 for Windows, UNIX, and z/OS so that pruning no longer affects replication latency. Pruning no longer requires a join of each Change Data table with the Unit of work table and is cursor-based with interim commits.
- The new IBMSNAP_SIGNAL table, created with data capture changes, provides a way to communicate with Capture through log records. Apply inserts records into this table to signal that capturing should start for a table. It is also used to signal update anywhere replication and to provide precise end points for Apply events.
- Capture start up parameters can be modified while Capture is running. New options have been added for warm start.
- One Windows NT or 2000 service for each Capture program.

  The services are defined through the Replication Center or with commands. You can stop and start the services from the Windows Services window.
- Capture is supported on 64-bit platforms — Windows, UNIX, z/OS.
- Capture is enabled for the MVS Automatic Restart Manager (ARM) on z/OS. If Capture is registered to ARM, then ARM will automatically restart Capture in the event of a Capture or system failure.

1.6.3 Apply

Apply enhancements include:

- Joins of the Change Data table and the Unit of Work table are no longer required when applying changes to user copies. All the information needed for such copies is contained in the Change Data table. If information from the

Unit of Work table is needed for a subscription predicate or target table column, then you can set a flag to tell Apply to do the join.

- Changes to target table primary key values can now be handled without converting all captured updates to delete/insert pairs. You must capture the before-image of the source table columns used for the target table primary key. Apply can use the before-image values to locate the target table row for an update. The previous alternative, where Capture converts all updates to delete/insert pairs, is still available and should be used if partitioning key values are subject to change.
- Faster full refreshes of target tables can be done using the load improvements in DB2 for Windows and UNIX V8 and DB2 for z/OS and OS/390 V7 or later. The Apply exit ASNLOAD sample program has been updated to take advantage of utility options offered on each platform.
- Apply password files are now encrypted. The new **asnpwd** command is used to create and maintain the password file. No passwords are stored in readable text.
- One Windows NT or 2000 service for each Apply program.

  The services are defined through the Replication Center or with commands. You can stop and start the services from the Windows Services window.
- Apply is supported on 64-bit platforms — Windows, UNIX, z/OS.
- Apply is enabled for the MVS Automatic Restart Manager (ARM) on z/OS. If Apply is registered to ARM, then ARM will automatically restart Apply in the event of an Apply or system failure.
- The ASNDLCOPYD daemon is no longer required to replicate files stored in DB2 Data Links Manager Version 8.1. Data Links provides a replication daemon for retrieving and storing external files managed by Data Links. If the Data Links reference to an external file is defined with RECOVERY YES, then DB2 replication can ensure that the reference and external file are consistent when replicated.

1.6.4 Monitor

Extensive monitoring functions have been added to DB2 Replication:

- Replication Center Monitoring

  You can check the status of replication processes and display historical information including:

  – Capture messages
  – Capture throughput analysis
  – Capture latency
  – Apply messages

  - Apply throughput analysis
  - End-to-end latency

- New commands with parameters to show the status of Capture and Apply processes:
  - Capture **asnccmd** with the **status** parameter
  - Apply **asnacmd** with the **status** parameter
- Replication Alert Monitor

  The Replication Alert Monitor is a separate program which can monitor one or many Capture/Apply processes. You use the Replication Center to define alert conditions, such as error messages, program termination, and exceeding limits on memory or latency. The Alert Monitor stores alerts in a control table that can be viewed from the Replication Center. You define users or groups of users that will receive e-mail notification when an alert occurs.

1.6.5 Troubleshooting

Serviceability improvements include:

- New trace facilities based on the **db2trc** model have been added for Capture and Apply. The **asntrc** command starts and stops a trace while the Capture and Apply programs are running. The new trace is available on Windows, UNIX and z/OS.
- New trace points have been added to DataPropagator for iSeries to provide more debugging information.
- The Replication Analyzer Program has been updated to work with DB2 replication V8 control tables.

1.7 The Redbook environment

This book was developed using the beta versions of DB2 Universal Database for Windows and Unix V8, DataPropagator for z/OS and OS/390 V8, and DataPropagator for iSeries V8. Figure 1-13 shows the testing environment:

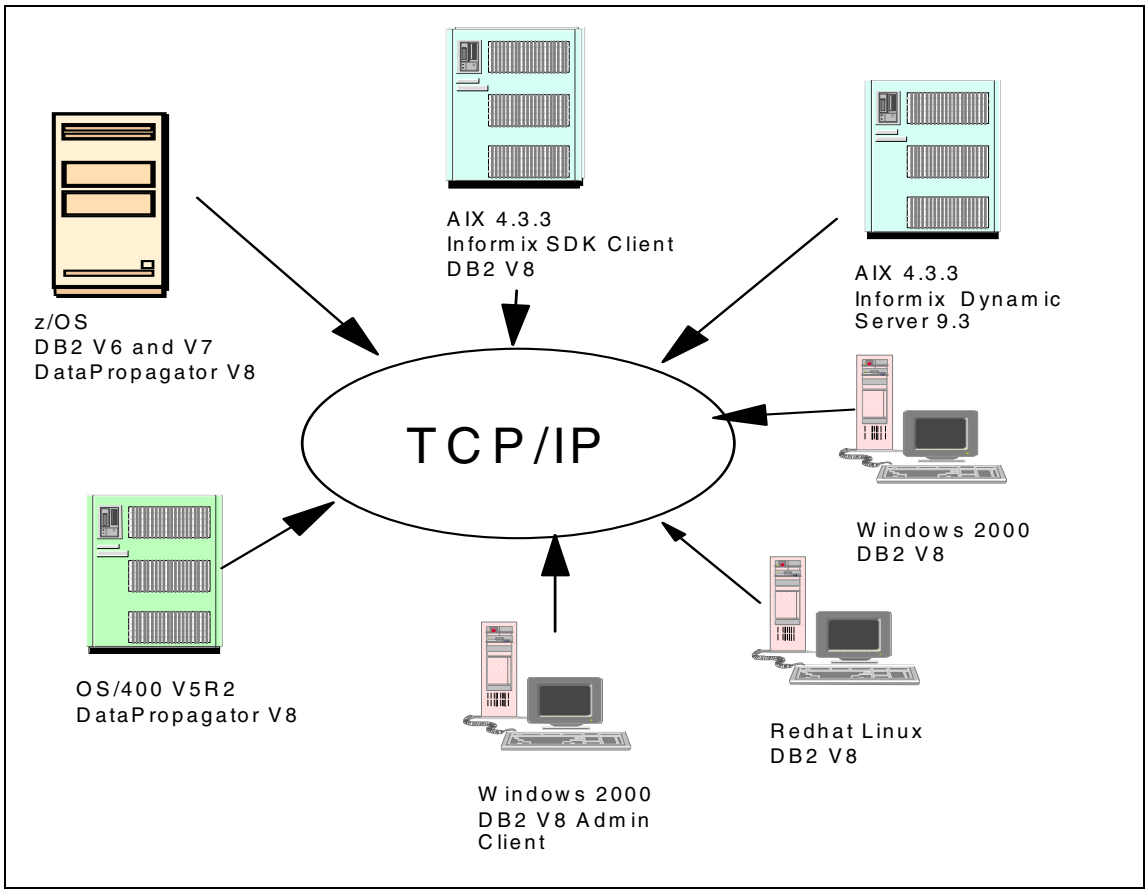


Figure 1-13 The Redbook Environment

We used the SAMPLE database on DB2 for Windows and Unix, the SALES demo database on Informix, the SAMPLE tables on DB2 for z/OS and OS/390, and created equivalent sample tables on DB2 for iSeries.

2

Getting started with Replication Center

In this chapter we will learn about the graphical user interface for administering, operating, and monitoring DB2 Replication Version 8. We will cover:

- architecture of the DB2 Version 8 Replication Center
- technical requirements for installing and operating the Replication Center
- where to get DB2 Replication Center
- connectivity requirements for DB2 Replication Center
- how to open DB2 Replication Center
- navigating DB2 Replication Center's graphical user interface
- how to save, run, or save-and-run-later DB2 Replication Center tasks

The DB2 Replication Center Version 8 can be used to define and operate DB2 Replication between any DB2 data sources, and between DB2 and Informix data sources.

DB2 Replication Center Version 8 can be used to set up and operate replication from:

- DB2 for OS/390 Version 6, DB2 for z/OS Version 7, and (future) DB2 for z/OS Version 8
- DB2 for iSeries Version 5 Release 2
- DB2 Universal Database Version 8 for Linux, UNIX, Windows
- Informix IDS

Replication Center can set up and operate replication to:

- DB2 for OS/390 Version 6, DB2 for z/OS Version 7, and (future) DB2 for z/OS Version 8
- DB2 for iSeries Version 5 Release 2
- DB2 Universal Database Version 8 for Linux, UNIX, Windows
- Informix IDS and XPS

DataJoiner Replication Administration (DJRA) and the DB2 Control Center that is with DB2 Administration Client Version 5, 6, or 7 cannot be used to administer DB2 Replication Version 8.

Also, DB2 UDB Version 8 Replication Center cannot be used to administer prior versions of DB2 Replication.

2.1 DB2 Replication Center's architecture

DB2 Replication Center is a graphical user interface and program written in Java.

Replicating Center's look and feel is similar to that of other administrative tools delivered with the DB2 Administration Client, particularly the DB2 Control Center. Replication Center uses DB2 client-to-server connectivity to access DB2 for z/OS and OS/390, iSeries, Linux, UNIX, and Windows. For performing definitions and monitoring, Replication Center uses regular DB2 connectivity to submit SQL. For operations, Replication Center uses DB2 Administration Server (DAS) communications to send commands to z/OS, OS/390, Linux ,UNIX, and Windows. Replication Center also uses API's from the JavaToolkit/400 to send commands to iSeries. To work with Informix, Replication Center depends on the federated server capabilities of a DB2 ESE Version 8 or DB2 Connect EE Version 8. We will discuss all of this in greater detail below.

User interface

Replication Center, though it is delivered with the DB2 Administration Client, has a separate user interface. Replication Center is opened separately on the desktop from the other tools, such as the DB2 Control Center, that come with DB2 Administration Client. Replication Center can be opened from within these other tools, and these other tools can be opened from within Replication Center. Also, there are some functions that are shared among the tools; for instance, the dialog windows for filtering the names of a tables, views, or nicknames that should be displayed. Figure 2-1 below shows an example of Replication Center's user interface.

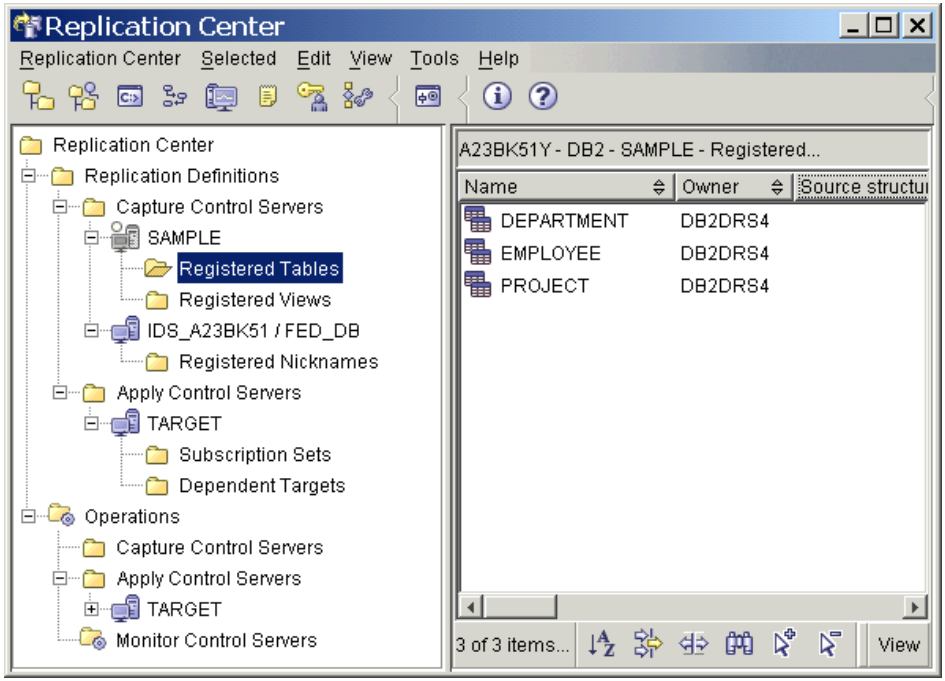


Figure 2-1 Graphical user interface

User-interaction methodology

Replication Center's dialogs for defining and operating replication interact between you and the various replication control servers to obtain the information needed to generate the SQL or commands to perform a task, such as to create the Capture Control tables at a replication source. When generating SQL, Replication Center also gets input from its own customizable profiles for replication control tables, replication source objects, and replication target objects; Replication Center has default assumptions for these profiles, but you can modify these profiles to change the assumptions used when Replication Center generates the SQL for a particular task.

Before the generated SQL or command for a particular task is executed, it is displayed to you. You can edit the SQL or command in Replication Center's *Run now or Save SQL* dialog window. You can then run the SQL or command immediately, or saving it to a file to be run later.

Replication Center's connectivity for defining replication

When defining or displaying information about registered sources, targets, and subscription sets, DB2 Replication Center uses JDBC and depends on DB2 client-to-server connectivity with the DB2 servers. See Figure 2-2.

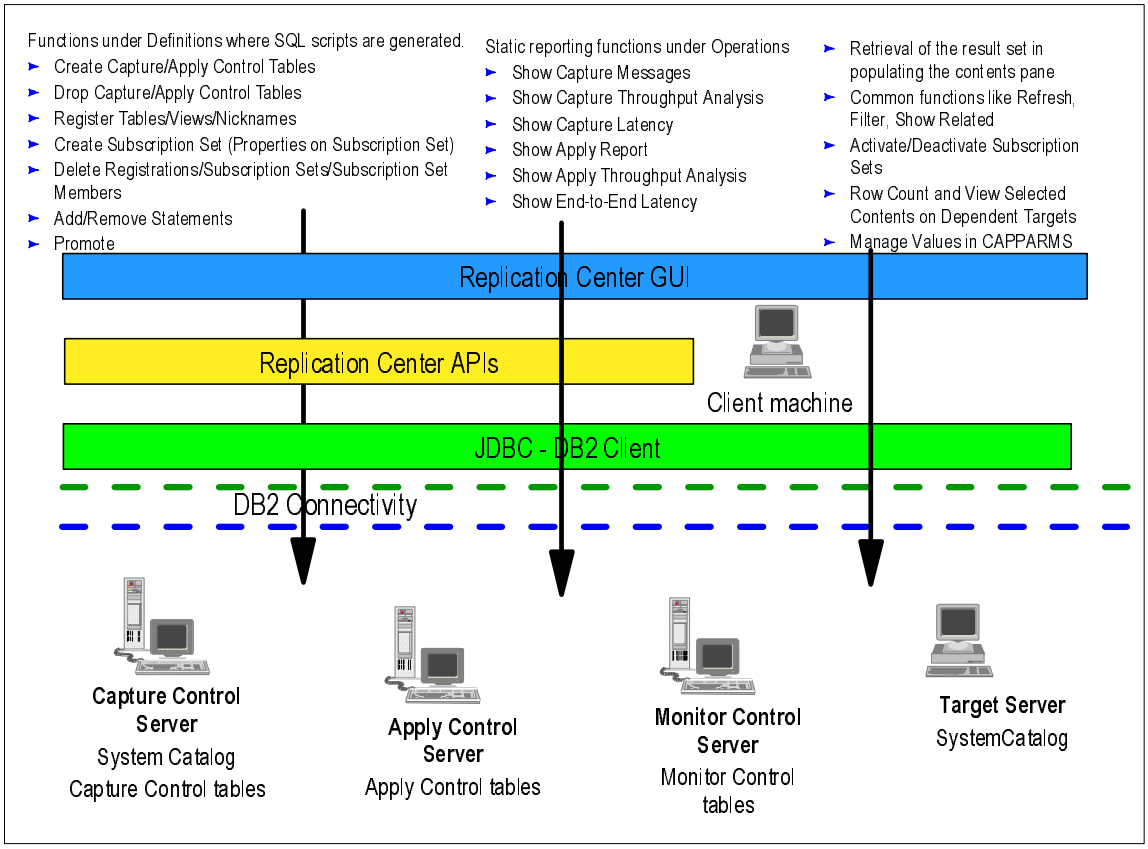


Figure 2-2 Use of API's, JDBC, and DB2 Connectivity

To obtain information to fill in the graphical user interface, and for simple tasks such as updating the status of a subscription, Replication Center uses SELECT and UPDATE statements delivered directly from the Replication Center GUI to the server via JDBC and DB2 connectivity.

For more complex replication definition tasks, Replication Center uses its own API's for these tasks. Each API is a separate ASNCLP command. These API's are within the replication center software. The API's in turn generate the SQL that executes the replication definition task. Replication Center will use JDBC and DB2 Connectivity to execute the SQL.

Replication Center's use of its own API's is 'seamless' and not apparent to the user. That is, the user will only see the SQL that is generated by the Replication Center and Replication Center API's working together and will not see the format and content of the Replication API request made by the Replication Center GUI.

Sorry, but we will not be covering the Replication Center API's (ASNCLP) in this redbook. And while these API's are not documented in the *DB2 UDB Version 8 Replication Guide and Reference SC27-1121* , DB2 Replication product development may make information about ASNCLP via the product website

DB2 client-to-server connectivity to DB2 Universal Database servers on Linux, UNIX, and Windows depends on the DB2 Runtime Client that comes with DB2 Administration Client being configured to communicate to DB2 on the server. Communications can be over TCP/IP, NetBIOS, Named Pipes, or APPC.

DB2 client-to-server connectivity to DB2 for z/OS and OS/390 servers depends on either DB2 Connect Personal Edition on the replication center workstation, or on the DB2 Runtime Client in DB2 Administration Client being configured to connect to DB2 for z/OS or OS/390 via a DB2 Connect Enterprise Edition or DB2 ESE server.

Replication Center's connectivity for replication operations

For Replication Center to start, stop, or check the status of Capture, Apply, and Replication Monitor, Replication Center uses DB2 Administrative Server (DAS) client-to-server communications. See Figure 2-3.

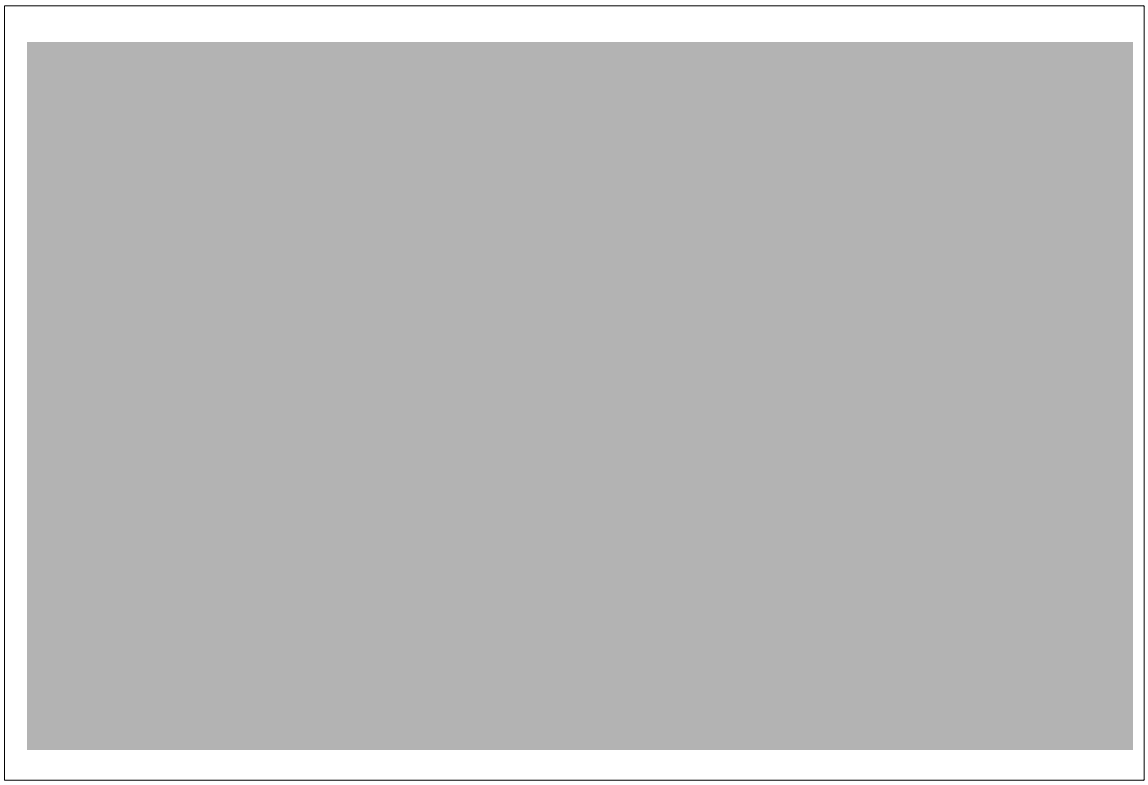


Figure 2-3 Use of DB2 Administration Server (DASe)

DB2 Administration Server server capability comes with DB2 for Linux, UNIX, and Windows. DB2 Administration Server server for z/OS will be made available for DB2 for z/OS Version 6 and Version 7 via 390 Enablement packages JDB661D and JDB771D respectively; there will be DB2 for z/OS PTF's required to work with DAS on z/OS and OS/390.

DB2 Administration Server client capability comes with DB2 Administration Client on Linux, UNIX, and Windows.

DB2 Administration Server only supports communications from DB2 Admin Server client over TCP/IP.

If DB2 Administration Server is not running on a Capture, Apply, or Monitor server, you can still stop, start, and check their status by accessing that system directly via telnet, 3270-terminal emulator, 5520-terminal emulator, or windows remote workstation software. Capture, Apply, and Monitor can then be started or stopped using the Capture, Apply, and Monitor the commands and methods

described in the *DB2 UDB Version 8 Replication Guide* (SC27-1121) operations chapters.

Replication Center connectivity to iSeries

To create replication definitions at iSeries Capture and Apply Control Servers, and to start and stop Capture and Apply on iSeries systems, Replication Center uses API's of JavaToolbox/400 on the Replication Center system. See Figure 2-4.

Figure 2-4 Use of Java Toolbox/400 API's to iSeries servers

A subset of the JavaToolbox/400 API's are included within Replication Center. The API's that are included with Replication Center permit Replication Center to send to iSeries requests for the CL commands needed for replication such as ADDDPRREG to define a replication source and STRDPRAPY to start Apply.

Replication Center and local profiles and files

For creating and managing Replication Center profile, and SQL files, Replication Center interacts in various ways directly with the system on which it is running. See Figure 2-5.

Figure 2-5 Access to profiles, passwords, and SQL files

Replication Center connectivity with non-DB2 servers

For working with Informix, and, in the future, non-DB2 replication sources and targets supported today by DataJoiner Version 2, Replication Center, as well as Apply and Replication Monitor, depend on the federated server capabilities of DB2 UDB ESE Version 8, DB2 Connect EE Version 8, or other (future) IBM product with exactly the same DB2 federated server capability as DB2 UDB ESE Version 8. See Figure 2-6.

Figure 2-6 Non-DB2 sources and targets

For replication from Informix, Replication Center will need to:

- create Capture Control tables in Informix
- create staging tables in Informix
- create triggers and procedures in Informix
- create nicknames and tables in the DB2 federated database that contains the federated server definition for the Informix data source

To create the Capture Control tables, staging tables, triggers, and procedures in the Informix replication source, Replication Center will use DB2 Federated Server Set Passthru capability to the Informix server.

For replication to non-DB2 targets, Replication Center can create the target tables if they don't exist. Replication Center uses DB2 Federated Server Set Passthru capability to Informix to create the target table. If target tables already

exists in Informix, Replication Center will need to be able to connect to a DB2 federated database that contains nicknames for the target tables.

2.2 Technical requirements for DB2 Replication Center

DB2 Replication Center Version 8 is delivered with the DB2 Administration Client Version 8 and has the same hardware, software, and connectivity requirements.

Check *Quick Beginnings for DB2 Version 8 Clients* and *DB2 UDB Version 8 Release Notes* for the final word on supported hardware and software required for DB2 Administration Client. The following is from the *Quick Beginnings for DB2 Version 8 Clients*.

2.2.1 Hardware requirements

DB2 Replication Center will run on hardware platforms that are able to run the operating systems supported by DB2 Administration Client. The supported hardware systems are:

- Personal Computers capable of running the Windows and Linux operating systems supported by the DB2 Administration Client.
- IBM RISC System/6000
- HP 9000 Series 700 or 800 system
- Sun Solaris SPARC-based computer
- IBM z/Series systems capable of running the Linux operating systems supported by the DB2 Administration Client.

For memory, minimum supported memory is 256 Megabytes. 512Megabytes of memory or greater is highly recommended.

For CPU, systems with only one processor are adequate.

For disk, minimum recommended disk space for DB2 Administration Client is 110MB on Windows systems and 150MB on UNIX.

For network attachments, the system running Replication Center will need a network adapter that will permit the workstation to access all systems that will be involved in replication. See networking requirements below.

2.2.2 Software Requirements

As stated above, Replication Center comes with the DB2 Administration Client and can run on systems supported by DB2 Administration Client. Also, we recommend you check *Quick Beginnings for DB2 UDB Servers on UNIX* and *Quick Beginnings for DB2 UDB Servers on Windows*, and DB2 UDB Release

Notes for Windows and for UNIX for details on supported operating systems and their levels.

The operating systems supported by DB2 Administration Client include:

- Microsoft Windows 2000
- Microsoft Windows/NT Version 4 with Service Pack 6a or later
- Microsoft Windows 98
- Microsoft Windows ME
- Microsoft Windows XP (32-bit or 64-bit)
- Microsoft Windows.NET servers (32-bit or 64-bit)
- AIX 4.3.3.78 or later.
- Sun Solaris 2.7 (32-bit or 64-bit) or later
  - Check *DB2 Client Quick Beginnings* for required patch levels
- Hewlett Packard HP-UX 11.0 32-bit or 64-bit or HP-UX 11i 32-bit or 64-bit. Check *DB2 Client Quick Beginnings* for release bundle details.
- Linux for Intel
  - For 32-bit Intel systems:
    - kernel level 2.4.9 or higher
    - glibc 2.2.4
    - RPM 3
  - For 64-bit Intel systems
    - Red Hat Linux 7.2, or
    - SuSE Linux SLES-7
- Linux for z/Series (390)
  - Red Hat Linux 7.2, or
  - SuSE Linux SUES-7

Other software requirements

- On Windows
  - Java Runtime Environment 1.3.1. JRE is included with DB2 Clients for Windows.
  - TCP/IP, Named Pipes or NetBIOS, which are included with the Windows operating system. Note: DB2 Admin Server client-to-server communications only supports TCP/IP.
  - For LDAP support, Microsoft LDAP client or IBM SecureWay LDAP Client V3.1.1. Microsoft LDAP client is included with Windows ME, XP, 2000, and .NET.

- On AIX
  - Java Runtime Environment 1.3.1 or later. Included with DB2 Clients for AIX.
  - TCP/IP - included with AIX
  - For LDAP support, IBM SecureWay Directory Client V3.1.1 is required
- On HP-UX
  - Java Runtime Environment 1.3.1 or later. Included with DB2 Clients for HP-UX.
  - TCP/IP - included with HP-UX.
- On Solaris
  - Java Runtime Environment 1.3.0 is required for Solaris 32-bit and Java Runtime Environment 1.4.0 is required for Solaris 64-bit. JRE is not included with DB2 Clients for Solaris.
  - TCP/IP - included with Solaris.
- On Linux systems
  - Java Runtime Environment 1.3.1. JRE is not provided with DB2 Clients for Linux.
  - TCP/IP - included with Linux

Additional pre-installation tasks on Solaris, HP-UX, and Linux:

- System kernel parameters in /etc/system must be set for DB2 by the Linux/UNIX administrator (root) and the system re-booted before the DB2 Client is installed. See ‘Modifying kernel parameters’ sections under the Solaris, HP-UX, and Linux installation instructions in the Quick Beginnings for DB2 Servers.

2.2.3 Networking requirements

A system running the Replication Center must be able to access:

- All DB2 Capture Control Servers, Apply Control Servers, Monitor Servers, and DB2 target servers whether they are:
  - Windows, UNIX systems, or Linux Systems that will run DB2 Universal Database Version 8
  - IBM z/Series mainframes running DB2 for OS/390 Version 6, DB2 for z/OS Version 7 or (future) DB2 for z/OS Version 8.
  - IBM iSeries Version 5 Release 2

- If there are Informix replication sources or targets, the Replication Center workstation will need to be able to access a system running either DB2 ESE or DB2 Connect EE Version 8. That system will in turn need to be able to access the Informix server. DB2 Connect or DB2 ESE Version 8 could be installed on the Informix server itself.

Replication Control Tables can be created and Replication Source definitions, and Replication Subscriptions can be defined over various networking protocols as supported by the DB2 Server. DB2 UDB servers on Windows support TCP/IP, NetBIOS, and Named Pipes for communications from clients.

But for managing replication operations (such as starting/stopping Capture Apply, and Monitor), which depends on DB2 Administration Server client-to-server communications, Replication Center will need TCP/IP connectivity to the DB2 servers.

It is best to test networking connectivity from the Replication Center system to each of the DB2 or DB2 federated servers that Replication Center will need to access. For instance, if TCP/IP will be used, ping each of the servers by their hostnames or IP addresses.

2.2.4 Requirements at replication servers

In a nutshell, DB2 on the systems with the Capture Control tables, Apply Control Tables, Monitor Control Tables, and the target tables needs to be able to accept connections from the DB2 Client on the Replication Center workstation and you will need a userid on those systems that can read the DB2 system catalog, create Capture, Apply, and Monitor Control tables, staging tables and target tables. In other words, the DB2 listener for inbound communications needs to be configured and running. For instance:

- on z/OS or OS/390:
  - the DB2 for z/OS and OS/390 Distributed Data Facility (DDF)
- on iSeries:
  - DDM TCP/IP listener
- on Linux, UNIX, and Windows, for TCP/IP communications:
  - the TCP/IP Service Name in the DB2 Database Manager Configuration and the DB2COMM in the DB2 Profile Registry (db2set).

The Replication Center user will also need a userid on these systems that can read the DB2 system catalog, can create the Capture, Apply, Monitor, staging and target tables, and can bind packages.

When replicating to or from Informix:

- The Informix server will need to be able to accept connections from the Informix Client SDK on a DB2 federated server. Typically this means that the Informix server's onconfig and sqlhosts information must indicate that the servers listener is running for Informix protocol onsoctcp.
- You will need a userid at the Informix data source:
  - If replicating to Informix and the target tables do not already exist, the userid will need to be able to read the Informix system catalog and create the target table.
  - If replicating to Informix and the target tables already exist, the userid will need to be able to read the Informix system catalog and insert/update/delete into the target table
  - If replicating from Informix, the userid will need to be able to read the Informix system catalog, create tables, create triggers, and create procedures on the source tables.
- Informix Client SDK will need to be installed on the DB2 federated server system and configured to connect to the Informix server.
- The DB2 ESE Version 8 system that is providing federated server access to Informix will need to be configured to receive connections from the DB2 Client on the Replication Center workstation.
- The Replication Center user will need a userid on the federated server system that can create Capture, Apply, Monitor Control tables in the federated database and can also create nicknames for the source or target tables that are in the Informix server.

2.2.5 Requirements for replication to/from non-DB2 servers

Before Replication Center can define replication either to or from a non-DB2 data source, the non-DB2 data source needs to be accessible by DB2 ESE Version 8 federated database capabilities. The federated database capabilities required for DB2 Replication Version 8 are available in DB2 ESE Version 8 and DB2 Connect Enterprise Edition Version 8 for use with in Informix. In the future, IBM will make the same capability available for use with Oracle, Sybase, Microsoft SQL Server, and Teradata. Neither DataJoiner Version 2 nor DB2 Relational Connect Version 7 can be used by DB2 Replication Version 8. Figure 2-7 shows an example of the configuration details for DB2 federated access to an Informix replication source or target.

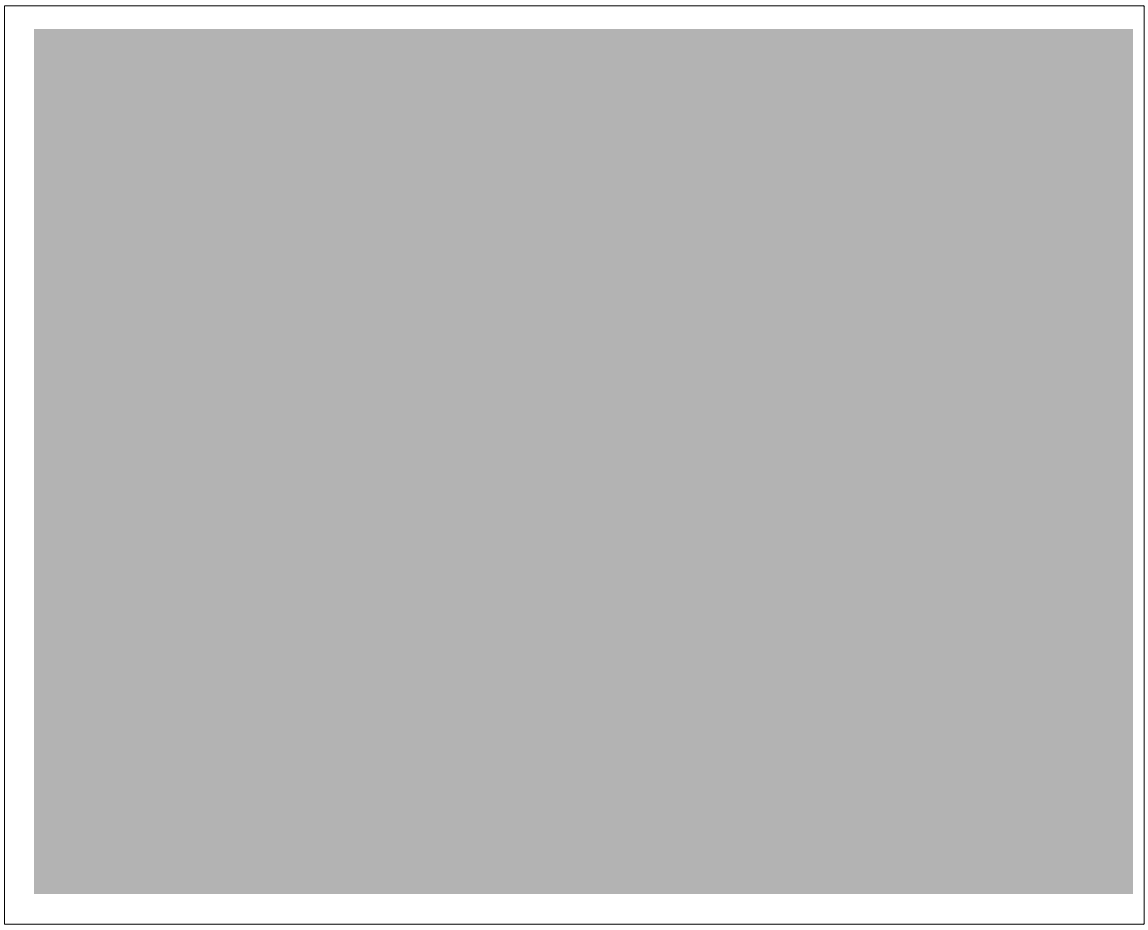


Figure 2-7 Federated Server configuration example to Informix

Here is a summary of the steps required to set up federated access from DB2 ESE Version 8 to an Informix data source.

- Informix Client SDK must installed and available on the DB2 ESE Version 8 system, and it must be configured to connect to the non-DB2 data source. Typically this means an entry for the Informix server in the sqlhosts file on UNIX or in the SQLHOSTS information in the Windows Registry on windows.
- DB2 ESE Version 8 or DB2 Connect EE Version 8 needs to be installed and latest fixpack applied. If DB2 ESE Version 8 is a source or target, a separate installation of DB2 ESE Version 8 or DB2 Connect Version 8 is not required for the federated access to Informix.

  - on Linux and UNIX system, djxlink needs to be performed to create a wrapper library that is linked with the data source client software. This step is not required on Windows systems.

- An INFORMIX Wrapper definition must be created in the DB2 ESE V8 database.
- Server and User Mapping definition for the Informix server must be created in the DB2 ESE V8 database. It is recommended that Set Passthru and then Create Nickname be used to test the Server/User Mapping definitions.
    - The Server Option IUD_APP_SVPT_ENFORCE should be specified with a setting of 'N'. If this option is not specified, insert/update/delete operations, which are required for replication both to and from Informix, won't be enabled.

More detail on the technical requirements and steps to configure federated access from DB2 ESE or DB2 Connect Version 8 to Informix are covered in Appendix C, "Configuring federated access to Informix" on page dxxxvii.

2.3 DB2 products needed to use Replication Center

To determine what DB2 products you'll need with Replication Center you should list your replication sources and replication targets and indicate what 'type' of DB2 products and whether you have Informix sources and/or targets. By types of DB2 products, we mean:

- DB2 for z/OS and DB2 for OS/390
- DB2/400 for iSeries
- DB2 Universal Database for Linux, UNIX, Windows

For Replication Center to connect to DB2 UDB (or to DB2 Connect) on Linux, UNIX, or Windows, the *minimum* DB2 product you require on the Replication Center workstation is the DB2 Administration Client. DB2 Administration Client includes within it the DB2 Runtime client which actually provides the basic DB2 connectivity to DB2 UDB on Linux, UNIX or Windows.

For Replication Center to connect to DB2 for z/OS or DB2/400 on iSeries, the *minimum* DB2 product you require on the Replication Center workstation is DB2 Connect Personal Edition**.** DB2 Connect Personal Edition includes DB2 Administration Client with it. DB2 Connect Personal Edition also includes DB2 Runtime Client and so if your source/target system is DB2 for z/OS or iSeries but the other system (source or target) replicated with is DB2 UDB on Linux, UNIX, or Windows, then DB2 Connect Personal Edition has all you need to connect to both source and target system.

An *alternative* to installing DB2 Connect Personal Edition on the Replication Center workstation, is to install only the DB2 Administration Client (with the DB2 Runtime Client included) on the Replication Center workstation and to connect to the DB2 for z/OS or DB2/400 source/target system via DB2 Connect Enterprise Edition or DB2 ESE on a Linux, UNIX, or Windows server.

For Replication Center to connect to a DB2 ESE or DB2 Connect that has federated access to Informix, then the *minimum* DB2 product you require on the Replication Center workstation is the DB2 Administration Client.

Note: Replication Center *cannot* use AS/400 Client Access to connect to an iSeries system.

2.4 How to Get Replication Center

DB2 Replication Center is included with the DB2 Version 8 Administration Client.

DB2 Administration Client in turn includes the DB2 UDB Version 8 Runtime Client which is the minimum DB2 presence required on the Replication Center system to be able to access DB2 source, control, and target servers and DB2 federated servers.

The DB2 Administration Client and DB2 Runtime Client capabilities are also included with the installation media for the DB2 Connect and DB2 UDB Server products, such as:

- DB2 Connect Personal Edition Version 8
- DB2 Connect Enterprise Edition Version 8
- DB2 UDB Personal Edition Version 8
- DB2 UDB Developers Edition Version 8
- DB2 UDB Enterprise Server Edition Version 8

The DB2 Administration Client can be included when any of the above products are installed. When fixpacks are available for DB2 UDB Version 8, which will be a month or two after general availability of DB2 UDB Version 8, it is best to download and install the latest DB2 UDB fixpack which can apply maintenance to all the DB2 components that were included in the initial DB2 installation. When applying the fixpack, be sure to indicate that you want the updates to the DB2 Administration Client to be included.

For access to DB2 for z/OS or OS/390 and iSeries, DB2 Administration Client will depend on the DB2 Connect capabilities of a DB2 Connect product or of DB2 ESE Version 8 on the same or a different system from the DB2 Administration Client system.

The DB2 Administration Client can be installed from the Client Installation CD's included with a DB2 Connect or DB2 Server product, or it can be downloaded from IBM. Once the first DB2 UDB Version 8 fixpack becomes available, a month or two after availability of DB2 UDB Version 8, it is recommended to download the DB2 Administrative Client from the IBM, rather than to install from the DB2 Client installation CD's. At the download site, IBM provides complete DB2 Client installation packages which include all the updates of the latest available fixpack. Thus once the first DB2 UDB Version 8 fixpack becomes available, the DB2 Client, with latest fixpack included, can be installed in one step by downloading it from the IBM download site, thus avoiding the two step process of first installing the client from the product CD's and then applying the fixpack.

To get the latest DB2 UDB Version 8 Administratin Client you can go to the DB2 Universal Database website with a browser and select Downloads and answer

the questions appropriately to navigate to the download server, or go directly to the IBM download ftp server itself from a command prompt or a browser.

The DB2 Universal Database website, as of this writing, is:

http://www.ibm.com/software/data/db2/udb/

Or go direct to the download server. It can be reached from a browser with

ftp://ftp.software.ibm.com

The DB2 UDB fixpacks and most current client software can be found under:

ftp://ftp.software.ibm.com/ps/products/db2/fixes/*language/platform-release*/client/admin

For example, for the DB2 UDB Version 7 Administrative Client for Windows 2000, the full URL is:

ftp://ftp.software.ibm.com/ps/products/db2/fixes/english-us/db2ntv7/client/admin/

We use this example because as of the writing of this redbook, DB2 UDB Version 8 wasn't available and so there were no DB2 UDB Version 8 Client downloads available, with or without fixpack included, at the IBM download site. When DB2 UDB Version 8 does become generally available, we might expect the URL to the DB2 UDB Version 8 Administration Client for Windows NT and 2000 to look something like this:

ftp://ftp.software.ibm.com/ps/products/db2/fixes/english-us/db2ntv8/client/admin/

The DB2 Client download file is usually in the form of a zip file for Windows platforms and a compressed tar file (.tar.Z) for UNIX and LInux platforms.

2.4.1 How to get DB2 Connect Personal Edition

Unlike DB2 Administrative Client, DB2 Connect Personal Edition is not free from IBM and so the installation media is not available from external IBM websites available to all customers.

Since prior versions of DB2 Replication on z/OS, OS/390, and iSeries also required the use of an administration tool that ran on workstations (Control Center and DataJoiner Replication Administration), prior versions of DB2 Replication on z/OS and iSeries included, in their shipment, a copy of DB2 Connect Personal Edition. We expect the same to be true for DB2 Replication Version 8 on z/OS and on iSeries.

Also, IBM customers with Software Passport Advantage contracts with IBM that include DB2 Connect Personal Edition may be able to download DB2 Connect Personal Edition from an IBM server that has been designated for them.

2.5 Installing DB2 Replication Center

Before covering the details of installing Replication Center, let's cover a pre-requisite for completing a successful installation of any DB2 product on Solairs, HP-UX, or Linux.

2.5.1 System kernel parameters on Solaris, HP-UX, and Linux

Remember, if you are installing DB2 Administration Client (including Replication Center), DB2 Connect, or DB2 UDB on Solaris, HP-UX, or Linux, you should first check the operating system kernel parameters, and if they are not set appropriately for DB2, then a person with systems administrator authority (root) will need to set them as needed for DB2 and the system will need to be rebooted before DB2 Administrative Client, DB2 Connect, or DB2 Server is installed. This is mentioned in the *DB2 UDB Version 8 Quick Beginnings for DB2 Clients,* in the *DB2 UDB Version 8 Quick Beginnings for DB2 Servers,* and the *DB2 Connect Enterprise Edition Version 8 Quick Beginnings* books.

If you do not have these books, either in hard copy or online, they can be found at the IBM DB2 Library website:

```
http://www.ibm.com/software/data/db2/library
   Select 'DB2 Universal Database'
```

The recommended settings for the Solaris and HP-UX kernel parameters are listed in *DB2 UDB Version 8 Quick Beginnings for DB2 Servers GC09-4836* in the Part 7. Reference in the chapter 'Additional Reference Topics.' In this chapter there are tables:

- Recommended HP-UX kernel configuration parameters
- Recommended Solaris kernel configuration parameters.

Instructions on how to set the HP-UX and Solaris kernel parameters are in *DB2 UDB Version 8 Quick Beginnings for DB2 Servers GC09-4836* chapter Installing DB2 Servers on UNIX in the sections 'Modifying kernel parameters (HP-UX)' and Modifying kernel parameters (Solaris).'

The recommended settings for the Linux kernel parameters are in *DB2 UDB Version 8 Quick Beginnings for DB2 Servers GC09-4836* in the chapter Installing DB2 Servers on UNIX in the section Preparing for Installation (Linux) - Modifying kernel parameters.

2.5.2 Installing DB2 Administration Client with Replication Center

The final authority on installing DB2 Administrative Client is the *DB2 UDB Version 8 Quick Beginnings for DB2 Clients GC09-4832*. If you are installing DB2

Connect Personal Edition and including DB2 Administration Client in the installation, please also consult Quick Beginnings for *DB2 Connect Personal Edition Version 8 GC06-4834.*

If you don't have these books in hardcopy or online, they can be found online at IBM at

```
http://www.ibm.com/software/data/db2/library
   Select 'DB2 Universal Database'
```

If you have the installation media for a DB2 Server or DB2 Connect product, it is OK to install that on your workstation, being sure to indicate that the Administrative Client be included among the components to be installed. In fact, if installing DB2 ESE Version 8 is an option for you, it is recommended that you do this as this will allow you create a couple DB2 databases on your workstation for becoming familiar with DB2 Replication without involving DB2 on a server, mainframe, or iSeries. Other options for having a replication 'sandbox' on your own workstation are:

- DB2 Personal Edition Version 8 if you will be administering replication only between DB2 on Linux, UNIX, and Windows systems. DB2 Personal Edition also includes the DB2 Administration Client and DB2 Runtime Client.
- DB2 Personal Edition Version 8 plus DB2 Connect Personal Edition Version 8 if you will be administering replication on z/OS, OS/390, iSeries as well as on Linux, UNIX, and WIndows systems.

Installing on UNIX

See the *DB2 UDB Version 8 Quick Beginnings for DB2 Clients GC09-4832* chapter 'Installing a DB2 Client' section on 'Installing DB2 Clients on UNIX.'

root needs to do the install.

If you have the installation CD for DB2 Clients, or for a DB2 Server, or DB2 Connect product, you can place it in a CD-ROM drive accessible from the system, find 'db2setup' on the CD, and run it (./db2setup) to bring up the DB2 Setup Wizard.

If you downloaded the DB2 Administration Client from IBM, root will need to first uncompress and untar the file and then find 'db2setup' and run it (./db2setup) to bring up the DB2 Setup Wizard.

Once in the DB2 Setup Wizard, choose 'Install Products' and be sure to include the DB2 Administration Client or DB2 Administration Tools among the components installed. This will installed the DB2 Control Center, DB2 Replication Center, and other DB2 administration tools on the workstation.

The DB2 software will be installed in:

- /usr/opt/db2_08_01 on AIX
- /opt/IBM/db2/V81. on other UNIX and Linux systems

Installing on Windows

See the *DB2 UDB Version 8 Quick Beginnings for DB2 Clients GC09-4832* chapter 'Installing a DB2 Client' section on 'Installing DB2 Clients on Windows.' If you are installing a DB2 Server or DB2 Connect product, then see the appropriate chapters in *DB2 UDB Version 8 Quick Beginnings for DB2 Servers GC09-4836* or the applicable *DB2 Connect Version 8 Quick Beginnings* guide (GC06-4834 and GC09-4833).

You will need an account on the workstation that has the appropriate authority. If you are only installing the DB2 Administrative Client the minimum is:

- On Windows 98 and ME:
  - Any valid Windows 98 user account
- On Windows 2000, Windows NT, Windows.NET, and Windows XP:
  - A user account with more authority than the Guests group, such as an account in the Users or Administrators group.
    - If in the Users group and on Windows 2000 or Windows.NET, then the registry permissions have to be modified to allow **Users** write access to the Windows Registry branch HKEY_LOCAL_MACHINE/Software

The above rules are true for installing the DB2 Administrative Client. If you are installing a DB2 Server or DB2 Connect product, it should be with an account in the Administrators group.

In our project that worked with DB2 Replication Version 8 to create this redbook, we installed DB2 ESE Version 8, including Administrative Client, using a local user account that was in the local Administrators group.

Example of DB2 Connect Personal / DB2 Admin Client install

Appendix A, "DB2 Admin Client and DB2 Connect PE install" on page dil contains an example of the steps to install DB2 Administration Client or DB2 Connect Personal Edition Version 8 on Windows.

2.6 Configuring DB2 Connectivity for Replication Center

DB2 connectivity from workstation that has Replication Center to DB2 servers are source servers, Capture Control Servers, Apply Control Servers, target servers, or federated servers can be performed either using DB2 Configuration Assistant or DB2 Command Line Processor.

Open the Configuration Assistant with:

Start-->Programs-->DB2-->Setup Tools-->Configuration Assistant

Open DB2 Command Line Processor commands

Start-->Programs-->DB2-->Command Line Tools-->Command Line Processor

Detailed examples of how to configure connections both ways are given in Appendix B, “Configuring Connections for Replication Center” on page dvii. The examples there are:

- Direct connection to DB2 for z/OS or OS/390
- Connection to DB2 for z/OS or OS/390 via a DB2 Connect server
- Direct connection to DB2 for iSeries.
- Connection to DB2 for z/OS or OS/390 via a DB2 Connect server
- Connection to DB2 on Linux or on UNIX
- Connection to DB2 on Windows

When configuring the connection, whether in Configuration Assistant or via DB2 CLP commands, include the operating system type of the server; Replication Center uses this information to determine if the server is DB2 UDB for Linux, UNIX, Windows, or DB2 for z/OS and OS/3390, or iSeries. The operating system type is specified:

- In *Configuration Assistant*, on the *Node Options* tab
- In DB2 CLP Commands, on the **`catalog tcpip node`** command.

See Appendix B for examples.

After configuring a connection to a DB2 data source, you can test the connection either in DB2 Configuration Assistant or using the DB2 Command Line Processor.

Appendix B also includes example of how to test your connection.

Though Replication Center, Capture, Apply and Monitor appear to bind any packages they need at any DB2 server they need to access, there may be occasions when you need to explicitly bind packages needed by Replication

Center, Capture, Apply or Monitor. The appendix also covers how to bind packages.

2.7 Replication Center and file directories

Replication Center definition and operations dialog windows, when you click ‘OK’, will generate SQL to perform the definition task but not run the SQL immediately. You will be presented the ‘Run now or Save SQL’ dialog window. If you elect to save the SQL to file before running it, you will probably want a directory already created to save the SQL/Command file into. For instance, on our workstations we created the folder *DB2Repl* and in that folder a sub-folder *ReplCtr* to contain files that would be created by Replication Center when we elected to save SQL generated by Replication Center definition tasks.

Likewise for Replication Center Operations task windows if you start and stop Capture and Apply using Replication Center. When you click ‘OK’, Replication Center will generate the command in a *Run or Save* window and you can save the command with its parameters in a file.

We’ll discuss using the Replication Center’s *Save and Run* dialogue in “Run Now or Save SQL” on page cxix.

2.8 Desktop environment for Replication Center

You can do many things related to Replication definitions and operations from within Replication Center, but still you will probably find it useful to have other tools available as you are creating or modifying your replication definitions and when you start Capture or Apply or activate a Subscription Set. Here are some suggestions.

For checking on table names and characteristics:

- in DB2 for z/OS or OS/390:
  - a 3270 terminal emulator (IBM Personal Communications) using
    - DB2 Administration Tools
    - SPUFI
  - a DB2 Command Line Processor session from your workstation
- in iSeries (AS/400)
  - a Client Access session from your workstation using
    - Operational Navigator
  - a 5250 terminal emulator (IBM Personal Communications) using
    - CL commands such as DSPFD and DSPFFD
  - a DB2 Command Line Processor session from your workstation
- in DB2 for Linux, UNIX, and Windows
  - DB2 Control Center on your workstation, using ‘Alter Table’ and ‘Show Related’ options
  - DB2 Command Line Processor on your workstation

For checking the contents of tables:

- in DB2 for z/OS or OS/390:
  - a 3270 terminal emulation session using
    - SPUFI
  - a DB2 Command Line Processor session from your workstation
- in iSeries (AS/400)
  - a Client Access session from your workstation using
    - Operational Navigator
  - a 5250 terminal emulation session using
    - CL commands such as DSPPFM (Display Physical Master) and RUNQRY (Display Contents with column names)
  - a DB2 Command Line Processor session from your workstation
- in DB2 for Linux, UNIX, and Windows
  - DB2 Command Center on your workstation
  - DB2 Control Center using **Sample Contents** option
  - DB2 Command Line Processor on your workstation
- Replication Center also has an option for seeing the contents of source and target tables. Under **Replication Definitions --> Apply Control Servers**, select a particular server, and then **Dependent Targets** under that server. In

the dependent targets displayed in the right window, highlight a particular target and either right mouse click or pick **Selected** from the menu bar and then select **View Selected Contents** from among the options. The *View Selected Contents* window will let you

- limit the number of rows displayed,
- select the target or the source table. To see the contents of the source table, use the check box in the middle of the screen.
- specify a predicate (WHERE clause) to limit the rows displayed.

For checking and managing files created by Replication Center:

- On UNIX and Windows
  - a command prompt or GUI, such as Windows Explorer, for viewing and managing files may be helpful. Replication Center can not write over existing files so it may be useful to have Windows Explorer, a DOS-prompt (Windows) or terminal session (UNIX/Linux) for seeing, renaming, removing files.

2.9 Opening DB2 Replication Center

There are a number of ways to open the DB2 Replication Center on your workstation:

- Start-Programs (on Windows):
  - **Start-->Programs-->DB2-->General Administration Tools-->Replication Center**
- At command prompt
  - **db2rc**

    On Windows, this will work in a DOS Prompt or in DB2 Command Window Prompt

    On UNIX and Linux, if you are not the DB2 instance owner, then before you enter **db2rc,** your PATH variable must have the DB2 directories added to it and the DB2INSTANCE variable must be set. An easy way to do this is to add the DB2 instance owner's '*db2profile* to your *.profile*.

    For instance, if the DB2 instance owner on your system is *db2inst1* with home directory */home/db2inst1*, then full path to his db2profile is

    ```
    /home/db2inst1/sqllib/db2profile
    ```

    You can edit your .profile, and add the line

    ```
    . /home/db2inst1/sqllib/db2profile
    ```

    For your PATH to be updated and DB2INSTANCE added to your environment, you will need to either exit and login again to the workstation or execute your .profile.
- From within DB2 Control Center, Command Center, or other DB2 Administrative Tool
  - From the Menu bar, select **Tools > Replication Center**

    See Figure 2-8.

Figure 2-8 Opening Replication Center from DB2 Admin Tool menu bar

From within DB2 Control Center, Command Center or other Administration Tool, select the Replication Center icon. See Figure 2-9.

Figure 2-9 Replication Center icon

The Replication Center should open on your desktop. The first time you open the Replication Center, two windows should open. The Replication Center Launchpad will be on top, and behind it will be the Replication Center itself. See Figure 2-10.

Figure 2-10 Replication Center and Replication Center Launchpad

If you do not want the Launchpad to open every time you start the Replication Center, check the box ‘Do not show the Launchpad again when Replication Center opens’; this check-box is in the lower left corner of the Launchpad. Your Replication Center profile will be updated so that Launchpad won’t open with the Replication Center. You will still be able to open the Launchpad if you need it by going to the Replication Center’s menu bar and selecting ‘**Replication Center --> Launchpad.**’

If you are not familiar with DB2 Replication, the Launchpad may be useful to you since it guides you the major steps of setting up and starting replication.

But if this is your first time opening Replication Center on your workstation, we recommend you close the Launchpad to get to the Replication Center itself so you can start updating your Replication Center profiles.

2.10 A Quick Tour of DB2 Replication Center

When you open Replication Center for the first time, it will probably look like in Figure 2-11.

Figure 2-11 Replication Center when opened the first time

The Replication Center has a look and feel common to the DB2 UDB Administration Tools:

- Left window, or *tree view,* contains a hierarchical tree of major object types and their names. The sub-structure of objects under a particular icon can be seen by selecting the '+' sign next to the object and retracted by pressing the '-' sign next to the object.

  To see actions that can be performed on an object in the tree, highlight the object and right mouse click to see the options.

- Right window, or *content pane,* contains a list of objects under an object type or name that was highlighted in the tree in the left window.

  Selecting a particular object in the right window will expand the tree in the left window and show its sub-objects in the right window.

  To see actions that can be performed on an object in the right window, highlight the object and right mouse click to see the options.

  Among the options on many objects will be 'Show Related' which will display a list of various types of other objects that could be related.

  In Figure 2-12, we have expanded the hierarchical tree to show Capture Control servers. There are no objects in the right window because we have not created the Capture Control tables in any servers yet.

Figure 2-12 Expanded tree

- The Replication Center's top menu bar contains various categories of options:
  - Replication Center
    - options for the whole Replication Center, including managing system userid's and passwords Replication Center will use when accessing various servers
  - Selected
    - options available for a specific object highlighted in the left or right windows below.
  - Edit
    - Find object or string, select object, and other similar options
  - View
    - Create or change the filter for objects to be displayed in the windows below.
    - Sort the contents of the windows or customize the column headings
    - Refresh the view, causing Replication Center to check for recent additions, deletions to the objects displayed.
  - Tools
    - open another one of the DB2 Administration Tools (Control Center, Command Center, etc.)or change over all Tools Settings. If the Administrative Tool you want is already open on your workstation, you can select its window or use **Alt-->Tab** to get to it.
  - Help
    - DB2 Information Center online.
    - DB2 Tutorials
    - 'About' which displays details of the version, release, and maintenance level of the DB2 product code on the system.
    - Other helpful resources.
- The Replication Center's tool bar also provides a row icons for opening other DB2 tools. See Figure 2-13.

Figure 2-13 Replication Center icon tool bar

The icons have ‘info pops’; hover your pointer over the icon for two seconds and an info pop will appear indicating what the icon is for. For most users of Replication Center, the most interesting will likely be:

- Control Center is the 1st icon on the left
- Command Center is the 3rd icon from the left
- Tools Settings is the 3rd icon from the right
- DB2 Information Center is the ‘i’ icon.

After putting a few replication definitions in place, the Replication Center might look as in Figure 2-14. In the tree in the left window, under the Capture Control Server ‘SAMPLE, Registered Source icon has been selected. In the right window only one registered table DB2DRS4.DEPARTMENT is shown; the small number of objects shown in the right window could either because there in fact only a small number of tables registered at SAMPLE, or because a filter was used to restrict the list of objects displayed in the right window. In the example, the object has been highlighted and right-mouse button clicked to show a list of options. The same list could have been displayed by highlighting the object and clicking **Selected** in the menu bar at the top of the Replication Center.

Figure 2-14 Expanded tree with objects and options

Selecting **Properties** would open the *Registered Table Properities* window, which looks like the *Register Tables* window, but is populated with the attributes of the registration.

Selecting **Show Related** will show other types of items that are related, for instance if there are any subscriptions defined from this registered table.

2.11 Managing your DB2 Replication Center Profile

Our first step after opening Replication Center the first time was to put information into the *Replication Center Profile*. When Replication Center accesses any DB2 servers, creates Capture, Apply, or Monitor control tables at servers, Registers tables, or Creates Subscriptions, the Replication Center Profile provides input to the definition process that can reduce the amount of keying that you will need to do in the definition window or the amount of editing you will need to do to the generated SQL before it is run.

After you provide input to the Replication Center Pprofile, you can change it anytime, even right before you perform a specific definition task that would use input from the profile.

The Replication Center Profile itself is a file - *db2repl.prf* - but its contents is managed through Replication Center and it cannot be read or updated through a text editor. You will notice that the size of the file grows as you do more updates to Replication Center Password information, Control Tables profiles, Source Object Profiles, and Target Object Profiles.

db2repl.prf is located in the DB2 instance directory.

- On windows, if the DB2 instance is 'DB2', which is typical, the *db2repl.prf* is in *c:\Program Files\IBM\SQLLIB\DB2*.
- On UNIX, look for it in the home directory, or a sub-directory, of the DB2 instance owner.

Replication Center maintains a backup - *dbrepl.prf.bkp* **-** in the same directory as the current db2repl.prf. If for some reason you lose both db2repl.prf and db2repl.prf.bkp, Replication Center will create a new one when you open and save from any of the Replication Center's profile windows.

Though there is in fact only one Replication Center Profile (*db2repl.prf)*, there are several dialog windows within Replication Center that can enter and change data in the profile:

- *Manage Passwords for Replication Center*

  When Replication Center needs to connect to DB2 servers to obtain information or to execute SQL or commands that Replication Center has generated in one of its definition or operations dialogs, Replication Center looks in the Replication Profile to see if the profile contains a userid and password to use in the connection. If the Replication Profile does not contain a password entry for the DB2 server or system, then Replication Center will prompt the user for the userid and password to use.

– opened from top menu tool bar **Replication Center** or the icon **Replication Center** in the tree in the left window. The *Manage Passwords for Replication Center* with no entries appears as in Figure 2-15.

Figure 2-15 Manage Passwords window

– In *Database* tab, **Add** any DB2 servers you can access from the DB2 Client or DB2 Connect Personal Edition that will either have replication sources, replication targets, or control tables for Capture, Apply or Monitor. If you will replicating to or from Informix servers, 'Add' the DB2 for Windows, Linux, or Windows database that will contain the federated Server definition for the Informix server. For 'Userid' and 'Password', put the userid and password by which Replication Center will access the DB2 server. The Userid should have adequate authority and privileges to read the catalog at the DB2 server and to create Capture, Apply, or Monitor control tables, staging tables, and/or target tables.

– In *System* tab, **Add** systems running DB2 Administrative Server (DAS) that Replication Center will access to start Capture, Apply, or Replication Monitor. This could include Linux, UNIX, or Windows systems running DB2 UDB. It could include z/OS systems running DB2 for z/OS and the DB2 Administration Server (DAS) running. DB2 Administration Server server for z/OS will be made available for DB2 for z/OS Version 6 and Version 7 via 390 Enablement packages JDB661D and JDB771D respectively; there will be DB2 for z/OS PTF's required to work with DAS on z/OS and OS/390.

  If a server doesn't have DAS, you will still be able to start Capture, Apply, or Monitor by logging directly into the system (i.e. not via Replication Center) and use the Capture, Apply, and Monitor commands to start/stop Capture, Apply, and Monitor.

– After you have added a Database or System you can later return to *Manage Replication Center Passwords and* **Change**' the password.

- The userid/passwords entered via the Manage Replication Center Passwords is stored in encrypted form in *db2repl.prf.*

- *Manager Control Table Profiles*
  - The Replication Center Profile can contain a different entry for each type of replication server. See the list in the discussion of platforms below. Before creating the control tables at a new server, you may want to return to the Control Tables Profile to change the settings for the new replication server's platform type.
  - Opened from **Replication Definitions** icon with right mouse click for from top menu bar **Selected**
  - When opened, at the top, select the type of platform for which you would currently like to update the profile:
    - DB2 Universal Database for Windows and UNIX (and Linux)
    - DB2 Universal Database for z/OS and OS/390
    - non-DB2 Capture Servers Informix, Oracle, Sybase and Microsoft SQL Server. Note, only Informix is supported for heterogeneous replication with DB2 UDB Version 8. Oracle, Sybase, Microsoft SQL Server will be supported in the future with a product that replaces DataJoiner Version 2.
    - DB2 UDB for iSeries (AS/400) is not included because the Capture and Apply Control Tables on iSeries are created using the CRTDPRTBLS CL command on the iSeries.

    Once you have selected the type of platform, you will notice the options in the dialogue window change as appropriate for the platform. For instance, for DB2 on UNIX and Windows you can specify containers for tablespaces. For DB2 on z/OS and OS/390 you can specify databases and storage groups for tablespaces.

    You should note that the 'default' behavior is to create one tablepspace to hold all the Capture Control tables except IBMSNAP_UOW, create a separate tablespace for IBMSNAP_UOW, and to create one tablespace to hold all the Apply Control tables.

    After filling in the window, click **Apply** in the lower-right corner to update the profile, and then **Close** to close the window.

    You can come back and update the Control Table profile later, such as right before creating the control tables at a particular server.

- *Manage Source Object Profiles*
  - The Replication Center Profile can contain a different entry for each Capture Control Server. Before creating a new registration, you may want

to open the Manage Source Objects Profile and change the settings for the erver where the table to be registered is located.

- opened from **Replication Definitions** icon, or from a specific Capture Control Server icon with right mouse click for from top menu bar **Selected**
- if opened from the **Replication Definitions** icon, you will first be presented a window for selecting a specific Capture Control Server
  - The Source Object Profiles can't be updated until at least one Capture Control Server has been defined.
- the available fields will be specific to the type of platform of the Capture Control server. For instance, for UNIX and Windows there will be fields for tablespace containers and for z/OS and OS/390 there will be fields for database and storage groups.
- there are fields for specifying an algorithm for the names of tablespaces, CD tables, and indexes on CD tables that may be created in the table Registration process.
- when filled in, or updated, click 'OK' to store the information into the Replication Center profile.

- *Manage Target Object Profiles*
  - The Replication Center Profile can contain a different entry for each Target Server. Before creating a new subscription set to a specific server, or before adding a new member to an existing set to specific server, you may want to open the Manage Target Objects Profile and change the settings for the server where the new table will be located.
  - opened from **Replication Definitions** icon with right mouse click for from top menu bar **Selected**
  - you will first be presented a window for selecting a particular Target Server
  - if target tables for a Subscription member don't already exist, Replication Center will use information in this profile for the definition of the target tables.
  - contains fields for specifying the algorithm that Replication Center will use for target table names, indexes on target tables, and name and characteristics of tablespaces for target tables.
  - when filled in, or changed, click 'OK' to store the information into the Replication Center profile.

2.12 Replication Center dialogue windows

Create Control Tables - Custom dialog window, shown in Figure 2-16, is offered as an example of the Replication Center dialogue windows to define and operate replication.

This dialogue was opened by highlighting **Replication Center -->Replication Definitions --> Apply Control Center** and either using the right mouse button to see the available options or by going to **Selected** in the menu bar at the top of the main Replication Center window. From among the options **Create Apply Control Tables-->Custom** was selected.

If there are multiple DB2 servers known to Replication Center, you will be presented a window displaying a list and asked to pick the one where you want to create the control tables.

Figure 2-16 Create Apply Control Tables - Custom

In this window, the tablespace information and index-naming information (back tab), was filled in from the Control Tables Profile. If the Control Tables Profile has not yet been set up on this workstation, then the tablespace/index naming information would be based on the default settings that come with Replication Center. You will be able to either change the values in the dialogue, ‘Cancel’ and go back and update the Control Tables Profile to change the value that will be in the dialogue the next time you open it.

Clicking ‘OK’ will cause Replication Center to generate the SQL that will execute this definition or operations step, but the SQL will not be executed. It will be displayed in a *Run now or Save SQL* window that will follow.

Actually, in the foreground, will be message box which will have any messages associated with generating the SQL. The message box will look like the one in Figure 2-17.

Figure 2-17 Generated SQL message box

You can read the messages and press **Close** to get to the *Run now or Save SQL* window. The first *Run now or Save SQL* window you see may look like that in Figure 2-33 which was for creating the Capture Control Tables.

Required fields (red outline)

In the Replication Center dialog windows, if there are fields where an entry is required, the field is outlined in red until an entry is made in the field. You will notice that the button to close the window or go to the next activity is not available to click until the fields with the red outlines have been filled in.

2.13 Run Now or Save SQL

For an example of the Replication Center *Run Now or Save SQL* dialogue window, see Figure 2-18.

Figure 2-18 Run Now or Save SQL window

We'd like to point out some things about this window and how to use it:

- The SQL (or Command) that was generated is in the *SQL Script* or *Command* window in the lower part of the window. To see more of the SQL at once, you can resize the window so the display field is larger. You canedit the text in the window to change what will be run or saved, or you can hit **Cancel** and go back to the dialogue that generated the SQL, change the values in the input fields so that different SQL will be generated. You could also go back and make changes to the Control Table Profile (or Source Object Profile or Target Object Profile) to change the input to the definition dialogue that generates

the SQL and go through the definition dialogue again to generate different SQL.

- ‘**Run now**, in the upper left corner of the window, is the default selection. If you press ‘OK’ (in the lower rigt corner) the generated SQL will be executed and, if it runs successfully, the *Run now or save SQL* window will close. If you press **Apply**, the generated SQL will be executed, and the *Run now or save SQL* window will remain open. You could then press **Cancel** to close the window. If you were to press OK, instead of Apply, you will get a message box indicating success or failure. If the SQL was successful, the *Run now or Save SQL* window itself should close.
- If you select‘**Save to file** in the upper left corner, then pressing Apply will save the SQL to a file (which file and where we’ll discuss below) and leave the window open. You should see a message box indicating if the file was successfully created. You could then select **Run now** in the upper left corner and **OK** or **Apply**; in this way you could save the SQL to file for record or future review, but execute the definition step right here in this window.I

Attention: If you select **Save to file** and click **OK,** the SQL or command will be saved in a file, but the *Run Now or Save SQL* window will close and you will have to find another way, outside Replication Center, to run the SQL or command.

If you would like to save a Replication Center-generated SQL or command to a file *and* run it now:

1. Select **Save to file.**
2. Fill out the *Save specifications* section of the *Run Now or Save SQL* window.
3. Click **Apply** here on the *Run Now or Save SQL* window.

   This will save the SQL or command to a file and leave the *Run Now or Save SQL* window open.
4. Select **Run now.**
5. Fill out the *Run specifications* section of the *Run Now or Save SQL* window.
6. Click **OK** here on the *Run Now or Save SQL* window.

   This will run the SQL or command, and close the *Run Now or Save SQL* window.

- If when you try to save the SQL to file you get a message box indicating that there was in IO error, there could be different causes for this:

- a file with the same name already exists. Replication Center can not over-write or replace existing files.
- Replication Center did not have authority to write in the directory that was chosen.
- The requested directory couldn't be found.

We recommend using a file manager (Windows Explorer) or command prompt (DOS Prompt) to go to the directory where you wanted to save the file and determine the possible cause of the IO Error message by examining the file already in the directory and/or by trying to create a small file in the directory such as with Notepad.

- Please note the 'Save multiple scripts in one file' check box immediately under **Save to file**. This is relevant when creating Subscription Sets with source-to-target members included , or when adding members to an existing Subscription Set. A member definition involves SQL steps at two or three servers:
  - At Apply Control Server, records are inserted into the control tables that that Apply reads to find out about the source-to-target member definition
  - At Source Server, records are inserted into the IBMSNAP_PRUNCNTL table that tells Capture about targets defined from a registered source table
  - At the Target Server, if different from the Apply or Capture Control Server, the target table is created if it doesn't already exist.

  If you leave 'Save multiple scripts in one file' unchecked, Replication Center will generate a different SQL file for each of the different severs involved. Replication Center will take the output file name you specify, and add '01', '02' etc. to the file name for each of the files created. Each of these files will contain 'CONNECT TO' statements for the indicated server.
- If you check **Save multiple scripts in one file**, Replication Center will create only one file containing all the SQL. Within the file, there will be 'CONNECT TO' statements before each of the groups of statements to be executed at a particular server.
- The *Run specifications* fields in the middle-left side of the window indicates the server the Replication Center will CONNECT TO to execute the SQL and the userid Replication Center will use on the Connection. Replication Center obtained this information from the *Manage Passwords for Replication Center* window*'s Database* tab.
- The *Save specifications* box in the middle-right side of the window indicates what system Replication Center will create the file that will contain the saved SQL and the userid and password that Replication Center will use. The userid/password information comes from the *Manage Passwords for*

Replication Center window*'s System* tab. Typically we would expect you to save the file on your own workstation (or to a network drive accessed by your workstation), but there is an option to specify another system and DB2 Administration Server (DASe) at that system could attempt to save the file there.

- To pick the name of the file and the directory where the SQL is to be saved, click the box with the three dots ('...') next to the *File Name* field. This is the last field under *Save Specifications.* You will presented with the *File Browser* window as in Figure 2-19.

Figure 2-19 File Browser for specifying file name and path

In the Replication *File Browser* window:

- select the appropriate disk drive in the lower right corner.
  - the current directory will be displayed in the upper right corner. Sub-directories will displayed in the Directories window below.
- If you wish to save into a sub-directory of the *Current directory*, double-click the sub-directory in the *Directories* field.
- If you wish to save into a sub-directory that is not under the *Current directory*, click the double dots ('..') at the top of the '*Directories*' list and the '*Current directory*' value will change to a higher level directory and the *Directories* field will present a list of sub-directories under the new Current Directory.
- When the *Current directory* field displays the path to the directory where you want to create the file:

- if there are already any files in that directory, their names will appear in the ‘Files’ field on the left.

  Note: Replication Center can not over-write or replace existing files. If you want to replace a file, go to Windows Explorer or a command prompt and erase or rename the existing file.

– type the name of the file you want to create into the ‘Path’ field.

– When saving SQL to a file, you may want to have a naming convention for the file names, for instance something to suggest

- the Server where the SQL would be executed,
- the type of definition activity,
- the name of the source table, subscription set, or target table.

For an example of how we filled in the Replication Center *File Browser* window to name the file *SAMP_CapCtl_MIXCAP.ddl* and save it into directory *d:\DB2Repl\ReplCtr,* see Figure 2-20.

Figure 2-20 Specifying file path and filename

2.13.1 Running Saved SQL Files Later

The SQL (or Commands) in Replication Center’s Saved- SQL files can be executed later from the Replication Center workstation or can be transferred to another system to be executed. We won’t cover that latter option here, but if you plan to do this, please read what follows regarding the contents of the Saved-SQL files from Replication Center.

Unfortunately, once the *Run now or SQL* window for the SQL has been closed, Replication Center itself doesn't have any facilities for opening the file later and running the SQL statements.

Some options for running the SQL in the file from your workstation would include:

- DB2 Command Window
  - a DOS-like Prompt, but operations within it have access to DB2 executable libraries on your workstation
- DB2 Command Center
  - a GUI tool with a facility for entering new queries or other SQL statement, or for importing scripts

SQL Statements in Replication Center Saved SQL Files

Before covering the use of DB2 Command Center and DB2 Command Window, let's discuss the contents of the SQL files created by Replication Center.

If you look at the contents of the SQL files generated by Replication Center you will notice that each separate SQL statement ends with a semi-colon (;), which is the standard statement delimiter for SQL. If you add any new statements to these files, they can use multiple lines in the file and each should be ended with a semi-colon (;).

Lines in the file that begin in the left margin with two dashes (--) are lines that won't be executed by DB2. These could be 'comment' lines, or they could be statements or commands that aren't to be executed. You can add comment lines to the file, if you start each comment line in the left margin with '--'. Also, you can make it so that existing statements in the file won't be executed by preceding the contents of the line with '--' in the left margin. It is also possible to make part of a statement not execute by putting that part of the statement on a line by itself that begins with '--'; if you do this, and it is the last line of the statement, make sure that you put a semi-colon, without '--', on the following line to indicate to DB2 where the statement ends.

In Example 2-1 are the first few lines of SQL generated by Replication Center's Create Capture Control table dialogue window:

Example 2-1 First few lines of generated SQL for Capture Control tables

```
CONNECT TO SAMPLE USER XXXXX USING XXXXX ;

CREATE TABLE ASN.IBMSNAP_REGISTER(
SOURCE_OWNER                    VARCHAR(30) NOT NULL,
SOURCE_TABLE                    VARCHAR(128) NOT NULL,
SOURCE_VIEW_QUAL                SMALLINT NOT NULL,
...
STOP_ON_ERROR                   CHAR(  1) WITH DEFAULT 'Y',
```

```
STATE                          CHAR(  1) WITH DEFAULT 'I',
STATE_INFO                     CHAR(  8))
IN USERSPACE1;
CREATE UNIQUE INDEX ASN.IBMSNAP_REGISTERX
ON ASN.IBMSNAP_REGISTER(
SOURCE_OWNER                   ASC,
....
```

The ‘CONNECT TO’ statement indicates that the statement(s) that follow are to be executed at DB2 server ‘SAMPLE’. The DB2 CONNECT statement is described in the *DB2 UDB Version 8 SQL Reference SC09-4845. T*he USER value indicates the userid to be sent with CONNECT statement when it is executed and the USING value password that is to be used; Replication Center has substituted ‘XXX’ for both the USER and USING values in this statement so it can not be used ‘as is.’ Your options with this CONNECT statement would be

- add two dashes (--) before ‘CONNECT’, so that the line becomes a comment and is not execute, and you should connect to SAMPLE by some other means before executing the contents of the file.

or

- put a valid userid and password in place of the two XXXs so that the CONNECT statement will execute successfully when the contents of the file are executed. The userid you use here should be one that has authority to execute the statements at the server indicated in the CONNECT statement.

The CREATE TABLE statement is over many lines; we have not shown them all.

You’ll notice the semi-colon after ‘IN USERSPACE1.’ This semi-colon marks the end of this CREATE TABLE statement. ‘CREATE UNIQUE INDEX’ is the beginning of the next SQL statement in the file; we have not shown all of that statement.

A SQL file created by Replication Center will likely have multiple CREATE TABLE, INSERT, UPDATE and/or DELETE statements in it. Following all the statements to be execute at the same DB2 Server will be a simple COMMIT statement. For SQL files that create control tables or that register tables, all the statements will likely be executed all at one server and there will be one COMMIT near the end of the file.

A file created by the Replication Center’s *Subscription Set* dialog window, if it adds members to an Subscription Set, will also include connections to multiple different DB2 servers if you checked on the Replication Centers ‘Run now or Save SQL’ window the option ‘Save multiple scripts in one file.’ This is because

creating a new member in a Subscription Set involves inserting records into Apply Control Tables, inserting records into Capture Control Tables, and, if the target table doesn't exist yet, creating the target table at the target server. The SQL file will have a COMMIT at each DB2 server before the CONNECT to the next DB2 server.

When creating control tables at Informix servers, registering tables in Informix, and creating new target tables in Informix, Replication Center will use DB2's Federated Server 'Set Passthru' facility to create tables, triggers, procedures at the Informix server. In that case, the SQL generated by Replication Center will contain

- a CONNECT TO statement to the DB2 Linux, UNIX, or Windows database that has the 'Server' definition for the Informix server,
- SET PASSTHRU for the federated database Server name for the Informix server
- SQL statements to be performed at the Informix; for instance CREATE TABLE, CREATE UNIQUE INDEX, CREATE TRIGGER, CREATE PROCEDURE statements. At the end of the statements to be execute at the Informix server, there should be a COMMIT,
- SET PASSTHRU RESET.
- If any tables were created at the Informix server in SET PASSTHRU mode, then after SET PASSTHRU will probably be CREATE NICNAME statements to put nicknames in place for the tables that were created.
- Following the CREATE NICKNAME statement may be one or more ALTER NICKNAME statements to change the local data type of some of the nicknames columns; this is done if the DB2 federated server's default type mappings caused the local type of a nickname column to be inappropriate for the data values that will be replicated through that column of the nickname.
- COMMIT at the federated server database that contains the new nicknames

DB2 Command Windows and saved SQL files

SQL in files created by Replication Center can be run in DB2 Command Window sessions. To open a DB2 Command Window, either click **Start-->Programs-->DB2-->Command Line Tools--> Command Window,** or in a DOS Prompt, enter'**db2cmd** and hit enter.

For example, for the SQL file saved in example above, open a DB2 Command Window, **cd** to the directory (*d:\DB2Repl\ReplCtr*) containing the file, and enter:

```
db2 -vtf SAMP_CapCtl_MIXCAP.ddl
```

If the CONNECT TO statements in the file were not preceded by dashes (--) then they will be executed and will establish the connection to SAMPLE where the rest of statements in the file will be executed; in this case, the CONNECT TO statements would need valid userid's and passwords after the USER and USING keywords respectively. For SQL generated by the Replication Center 'Create Subscription with Members' or 'Add Members' dialogue, if you specified that multiple scripts be saved in one file, find all the CONNECT statements in the file to add userid's/password's that are valid for each of the DB2 servers referenced, and execute all the statements in the file, at all the DB2 servers, in one use of **db2 -vtf filename**.

If the CONNECT TO statements in the file were preceded by dashes, we would first need to connect to SAMPLE, and then execute the file with **db2 -vtf filename**.

For instance, **cd** to the directory (*d:\DB2Repl\ReplCtr*) containing the file and enter:

```
db2 connect to sample user db2drs4 using db2drpwd
```

The server will responds with connection information seen in Example 2-2.

Example 2-2 Connection Information

```
   Database Connection Information

    Database server        = DB2/NT 8.1.0
    SQL authorization ID   = DB2DRS4
    Local database alias   = SAMPLE
```

Enter the following command:

```
db2 -vtf SAMP_CapCtl_MIXCAP.ddl
```

and the un-commented statements in the file will be executed at SAMPLE.

DB2 Command Window can be used to connect to and execute SQL statements at any DB2:

- DB2 Universal Database on Linux, UNIX, or Windows
- DB2 Universal Database for z/OS and OS/390
- DB2 Universal Database for iSeries

Using DB2 Command Center with saved SQL files

The DB2 Command Center has several 'tabs.' The *Script* tab can be used to import the contents of files, but it is the 'Database Connection' field on the *Interactive* tab that is used establish the connection to the DB2 server where the

statements in the imported script will be executed. You can import the script into the *Script* field of the *Script* tab either before or after you establish the connection to the appropriate server via the Database Connection field of the *Interactive* tab; but certainly you must establish the database connection before you execute the script. Here is an example of how to use the DB2 Command Center with a saved SQL file from Replication Center.

After opening the DB2 Command Center, select the 'Interactive' tab.

Select the '...' button at the right of the Database Connection field and you will be presented with a window where you can expand a tree of systems, instances, and databases until you see the icon of the 'database' where you want the script to execute. Select this icon and you will be presented with a dialogue window for entering the userid and password to be used on the connection.

Once the connection is establish, select Command Center's *Script* tab.

From the menu bar at the top of the window, select **Script-->Import**.

You will be presented with the *Import* file window.

In the *System Name* field near the top-left, select the system (probably your own workstation) where the file is. Select the appropriate disk drive in the *Drives* field at the lower right. Select the appropriate directory from the *Directories* window on the right. You can navigate to higher directories by selecting '..' from the directory list. Files in a selected directory will appear in the Files field on the left. Select the file you want to import so that its name appears in the Path field on the left and press 'OK'. You will be returned to the Command Center's *Script* tab with the contents of the file in the *Scripts* field. If you want, you can edit the script before running it. For instance, find CONNECT statements and precede each of them with two dashes (--) so that they won't execute.

You can then execute the script by either selecting the 'gears' icon in the upper left corner of the window, or **Script-->Execute** from the top menu bar, or by pressing your **Control** and **Enter** keys together at the same time.

DB2 Command Center can be used to execute Replication Center Save SQL files meant for execution with DB2 UDB on Linux, UNIX, and Windows and with DB2 UDB for iSeries. The Command Center should also work with DB2 for z/OS or OS/390.

2.14 Creating Control Tables - Quick or Custom

When Creating Capture, or Apply Control Tables from Replication Center, you will be presented the option of using the *Quick* or *Custom* dialogue windows. When creating the Monitor Control tales, the custom dialog is the only option. When Creating Capture or Apply Control Tables from Replication Center Launchpad, you will be presented with the *Quick* dialogue for creating control tables.

The *Quick* Capture and Apply Control tables dialogues ignore what may be in your Control Table Profile, make some assumptions for the tablespaces for the control tables, and offers fewer options for specifying tablespaces for control tables. The *Custom* dialogue window for creating Capture and Apply Control tables, when opened, will be filled in with information from your Control Table Profile and will give many more options for specifying the tablespaces to contain the control tables.

After you fill in the Create Capture or Apply Control Server dialogues, click ‘OK’ and then, on the *Run Now or Save SQL* window, select **Run Now** (upper left corner) and click either **Apply** or **OK** (lower right corner), the Capture or Apply Control Tables will be created and an icon with the server name will then appear under the Capture Control Server or Apply Control Server icons in the tree on the left side of the Replication Center main window.

2.15 Adding Capture and Apply Control Servers

If you **Save the SQL** created by the *Create Capture/Apply Control Table* dialogues and **Cancel** out of or exit from the *Run Now or Save SQL* without executing the SQL from Replication Center and then run that SQL outside of Replication Center to create the control tables, then the respective server name will not automatically appear under Capture or Apply Control servers in theReplication Center main window.

To add a Capture or Apply Control Server, under Replication Definitions, highlight either Capture or Apply Control Servers icon, whichever is appropriate, and among the available options, select **Add**. See Figure 2-21.

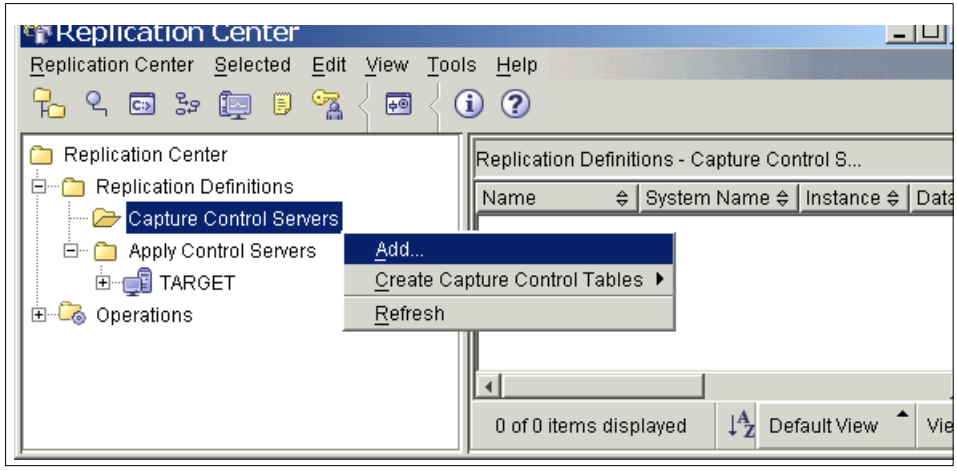


Figure 2-21 Adding a Capture Control Server

The *Add Capture or Apply Control Server* window should open. See Figure 2-22.

Figure 2-22 Add Capture Control Servers window

In our example we want to add SAMPLE as a Capture Control Server since we know that it has the Capture Control Tables. We would 'check' the box for SAMPLE under the Capture Control Server column. If our Replication Center Passwords profile has an entry for this server, then the Userid from that entry should be filled in when we click the Capture Control Server check box.

Not shown in the figure are the **OK**, **Cancel** and **Help** buttons in the lower right corner of this window. When the required information has been filled in, the **OK** button should become available and should be selected to close the window and add the new server name into the main Replication Center window.

As with the other Repication Center dialogs, the **OK** button will not be available to select until required fields - outlined in red - are filled in.

2.16 Replication Center objects for non-DB2 servers

When an Informix Server is made a Capture Control Server, that is, Capture Control tables have been created at an Informix server and nicknames for these Capture Control tables have been created in a DB2 ESE or DB2 Connect database, the Replication Center's windows will contain indication of the federated Server definition name for the Informix server. The Capture Control Server icon for the DB2 database that contains the nicknames for the control tables in Informix will indicate not only the DB2 database that contains the nicknames, but also the Server definition name for the Informix server. See Figure 2-23.

Figure 2-23 Capture Control Server icon for Informix

In the example, the text for the specific Capture Control Server's icon includes:

- the Server name for the Informix server in the DB2 ESE/Connect database
  - In this example, it is IDS_A23BK51:
- the name of the DB2 ESE or DB2 Connect database that contains the server definition to the Informix server.
  - In this example, it is FED_DB

We will see similar indicators elsewhere in the Replication Center if a Target server object includes federated access to an Informix server for target tables at an Informix server.

2.17 Creating registrations and subscriptions

Once a Capture or Apply Control Server have been added to the Replication Center main window, the dialogs for registering tables at Capture Control Servers and for creating Subscription Sets can be opened.

If the new server is a Capture Control Server, then it can be double-clicked to show the **Registered Tables** and **Registered Views** icons beneath it. One of these can be highlighted and the options for it shown so that **Register Tables** or

Register Views an be pickecd to open the *Register Table* or *Register a View* dialog windows.

If the Capture Control Server is for an Informix server, then the only icon under the server will‘**Registered Nicknames**. One of its options will be to **Register Nicknames** which will open the *Register Nicknames* dialog window.

After a table or nickname has been registered, its object in the Replication Center right window can be highlighted and a right-mouse click will show that one of the options is to create a Subscription Set.

If the new server is an Apply Control Server, then it can be ‘double-clicked’ to show the **Subscription Sets** and the ‘**Dependent Targets** icon beneath it. The **Subscription Sets** icon can be highlighted and its options displayed. One of these options is **Create** which can be selected to open the dialog window for creating a new subscription set and adding source-to-target members to the subscription set.

Figure 2-24 shows the Replication Center main window with Capture Control Server SAMPLE and Apply Control Server TARGET.

Figure 2-24 Opening Register Tables dialog window

Replication Center filters

The list of registered tables displayed in the right window in Figure 2-24 above might have been restricted to a small number by using a filter. The filter could have been created (or altered) by highlighting **Registered Tables** in the left

window and using right-mouse to show a list of options from which **Filter-->Create** was selected.

The filter could have been updated as indicated in Figure 2-25 so that only registered tables those schemas started with ‘DB2DR‘ and whose names started with ‘DEP’ would be displayed.

Figure 2-25 Filter example

2.18 Replication Center Launchpad

If you have specified that the *Replication Center Launchpad* will not be displayed every time you open Replication Center, you can open the Launchpad anytime by going to the Replication Center's menu bar and selecting **Replication Center--> Launchpad.**

The *Launchpad* provides a list of the major DB2 Replication set up and operations activities with an explaining graphic and summary description for each activity.

Placing your cursor over any of the activities on the left side of the Launchpad will highlight the appropriate part of the graph in the lower right, and display in the text window on the upper right a summary explanation of the activity.

Clicking on any of the activity icons on the left side of the *Launchpad* will open the Replication Center's dialogue window for executing the activity.

For **Creating Capture Control Tables** or **Creating Apply Control Tables**, the dialogue that will be opened will be the *Quick* dialog for creating these control tables, which has much fewer options and makes more assumptions about tablespace names than the *Custom* dialogue for creating control tables. If you want to use the *Custom* dialogue for creating control tables instead, close the Launchpad so you can access the Replication Center itself, under Replication Definitions, highlight the Capture Control Servers or Apply Control Servers icon, right mouse click to see options, and select **Create Capture Control Tables --> Custom** or **Create Apply Control Tables --> Custom**.

All the functions available through the Replication Center Launchpad, and more, are available through the main Replication Center window.

2.19 Trying replication with the DB2 SAMPLE database

If you have the opportunity to install DB2 on your workstation, we recommend that you do this so that you will have a DB2 where you can experiment with DB2 Replication. The architecture of DB2 Replication components is the same on all platforms; this common architecture is what permits Apply to replicate between different platforms. So DB2 on Windows or DB2 on Linux are valid environments for becoming familiar with Replication Center and the fundamentals of operating Capture and Apply for any platform.

You could do the following:

- create two databases in DB2 on your workstation; for instance SRC_DB and TGT_DB,
- create one or more tables to be your replication source tables in one those database
- use Replication Center to do the following:
  - Update your Replication Center Password Profile to add entries for these databases and for your system
  - Update Replication Center's Control Table profile for DB2 UDB UNIX an Windows Servers
  - create Capture Control tables in the database with the source tables
  - create Apply Control tables in the other database which is to be the Apply Control Server and Target Server.
  - Update Replication Center's Profiles for Sources and for Targets
  - Register your source tables
  - Create a Subscription Set for replicating the tables into the target server database. We would suggest a replication interval of 1 minute so that you will be able to check Apply problems quickly.
  - Start Capture and start Apply
  - check target table to make sure Apply has done the full refresh
  - update the source table and check the CD table to make sure Capture has inserted a change record
  - wait a minute and then check the target table to verify that Apply has replicated the update

You could also look at Capture and Apply's various operational reporting facilities through the Replication Center. You can also Create the Replication Monitor Control tables and start Replication Monitor to see what information if provides.

The DB2 UDB Sample database could be used as your source database. The DB2 UDB Sample database can be created by opening DB2 UDB's 'First Steps' window with **Start-->Programs-->DB2-->Setup Tools-->First Steps.** When you create the Sample database, several tables are created in SAMPLE, and several records are loaded into each of these tables. Tables that are useful for replication familiarization are DEPARTMENT, EMPLOYEE, and PROJECT tables.

Keep in mind that the tables create when the Sample database is created do not have any primary keys or unique indexes. DB2 Replication requires that target tables have either primary keys or unique indexes; you will see in the SQL that Replication Center's Create Subscription dialogue windows that target tables are created with unique indexes.

Apply uses the unique index/primary key values in updates and deletes to determine which record to update or delete in the target table. If you don't alter the tables of the Sample database to add primary keys, then when you set up a Subscription from one of these Sample tables, you will need to explicitly declare which column should be the primary key in the target table. This is not a hard decision to make since the appropriate column names are obvious: DEPTNO, EMPNO, PROJNO. Or if you want, you can add Primary Keys to the source tables before you define replication; these is easy to do in the DB2 Control Center. Highlight the table and right mouse click to see available options; select 'Alter Table', then the 'Keys' tab, and then the 'Add Primary Key' button to be presented with a list of columns from which one or more can be made part of a primary key. If you do this, when you create a subscription for this table, you will see that the primary key for the target is already designated in the Create Subscription dialogue.

2.20 More Replication Center tips

We have a few more tips for using the Replication Center

Adding databases while Replication Center is open

Replication Center only reads the local DB2 Database Directory when Replication Center is initialized. Replication Center will not have any awareness of any local databases created while it is open, or of any other additions to the local database directory made by using Catalog Database commands or by using Configuration Assistant.

You will need to close and re-open Replication Center to make it aware of any new DB2 servers that were configured while Replication Center was open.

Querying Capture and Apply control tables

Capture and Apply lock the control tables they update while they are updating them. If Capture is committing updates to its control tables frequently and Apply has a short replication interval, then these tables could be in a lock status when you query them for information causing your query either to take a long time or time-out. We suggest you end your SELECT statement with the words ‘with ur’ so that your query can complete quickly by allowing it to read the control tables even though Capture or Apply may have locked them for updates.

Operating system type of servers

Replication Center can use the Operating System Type to determine whether a specific DB2 server is DB2 for z/OS or OS/390, iSeries, or DB2 for Linux, UNIX, Windows. We recommend you specify the operating system type when you configure connectivity to a DB2 server. If the connection is configured using Configuration Assistant, in the *Add Database Wizard,* Operating System is specified on the *Node Options* tab. If the connection is configured using DB2 CLP commands, OSTYPE is specified in the **`catalog tcpip node`** command.

For details on configuring connections for Replication Center, see Appendix B, “Configuring Connections for Replication Center” on page dvii.

Userid and password prompts

When Replication Center is about to access any server, you will be prompted for a userid and password, unless there is a record for this server in the Replication Center Profile. Password records are added to the Replication Center Profile via the *Manage Passwords for Replication Center* dialog window which can be opened from the Replication Center main window by highlighting the **Replication Center** icon, right-mouse click, and select **Manage Passwords for Replication Center** from the available options. On the *Database* tab, you can add a record

for Replication Center to use when it connects to DB2 at the server. On the *System* tab, you can add a record for Replication Center to use when it uses DB2 Administration Server (DAS) to submit commands such as to start or stop Capture or Apply or check on their status.

Replication Center tracing

Replication Center's tracing is part of the tracing for all the DB2 Tools that come with DB2 Administrative Client. Replication Center tracing can be turned on by starting Replication Center from a command prompt (i.e. DOS-prompt) using the **db2rc** command and specifying **-tf *filename***.

Also, some Replication Center activity can be captured in the DB2 trace (**db2trc**) on the Replication Center workstation.

To trace Replication Center's activities using db2trc

- add DB2 Tools tracing to DB2 trace. To do this, on the Replication Center main window (or the main window of Control Center, Command Center, etc.), from the menu bar at the top, select **Tools > Tools Settings**. In the dialogue notebook that opens, on the **General** tab, check **Add Tools Tracing to DB2 Trace.**
- open Replication Center and do your activities just up to the point before the error. We will only want in the trace records for just the specific activity that had the error.
- open a DB2 Command Window and 'cd' to a directory where you want the trace output files.
- If you are not yet familiar with db2trc, look at its online help.

  db2trc -h

  shows the general format of db2trc command, and input parameters. It lists the major commands of db2trc: on, off, clear, dump, format, etc.

  db2trc *command* -u

  shows more details about usage of a specific db2trc command

- turn on db2trc.
  - for example, to turn on db2trc with a memory buffer of 8 megabytes

    db2trc on -l 8388608

    -l tells db2trc, if it runs out of memory in the trace buffer, over-write the first records captured. -i instead, would tell db2trc if it runs out of memory in the trace buffer to stop capturing records when the buffer is full and keep the first records captured.
  - in Replication Center, perform the activity you want to trace

- dump the contents of the db2trc buffer to a file. For example:

```
db2trc dump > RC_trc.dmp
```

- format the db2trc dump file just created

```
db2trc format RC_trc.dmp > RC_trc.fmt
```

We'll point here that one single activity in Replication Center could generate over a hundred thousand db2trc records with a trace dump file size of over 8,000,000 bytes. If when you format the trace, you get the indicator

```
Trace wrapped: YES
```

this means that db2trc, while tracing your activity, ran out of trace-buffer memory and wrote over the first trace records it captured. This could occur if db2trc was started with parameter `-l`.

```
Trace truncated: YES
```

would indicate that db2trc ran out of memory in the trace buffer and stopped capturing trace records. This could occur if db2trc was started with parameter `-i`.

If you want to look at the trace yourself to see if you can find the cause of a problem, you will probably find that the trace has many records. In the trace, you could try to find some word that is associated with the error to try to find records with useful information about the cause of a problem.

3

Replication control tables

This chapter discusses the following:

- Introduction to replication control tables
- Setting up capture control tables
- Setting up apply control tables
- Advanced considerations

3.1 Introduction to replication control tables

The replication control tables are the innards of replication. The components of replication communicate by accessing, updating, and creating the control tables' data. The control tables are viewable and manually able to be updated. In most circumstances manually updating is not recommended. This design makes DB2 Replication flexible.

The design divides the control tables into three sets based on the functionality they are related to: capture, apply, and monitoring. Each set of control tables is independent. The sets can be stored in separate databases, in separate instances, on separate servers. Unless the simplest of implementations, you will likely have varying numbers of each set.

Each of capture, apply, and monitoring control tables require different considerations about where to put them, how much space to allocate for them, and other configuration factors. Alert monitoring is an optional, but highly recommended activity. Often only by first monitoring in your test environment will you be able to address the considerations and tune your production environment well. We will describe creating the monitoring control tables in "Troubleshooting and monitoring" on page cccxxxvii. That chapter presents all the aspects of replication monitoring.

Here we will describe creating the capture and apply control tables. Included will be examples and advanced scenarios. The new in version 8, Replication Center (RC) will be used extensively. "Advanced considerations" on page clix describes ways to accomplish these tasks without using RC.

Tip: Replication Center includes excellent help. Clicking the **Help** button on most window will display detailed information about the window. The help often lists additional references.

Assumptions made in the following sections:

- Database Administrator (DBA) knowledge of table space creation and management. Basic configurations will be provided.
- All databases involved have been cataloged. See Section 2.6, "Configuring DB2 Connectivity for Replication Center" on page c for additional assistance.
- You have appropriate authority to connect to the computers, access the databases, and create table spaces and tables.

3.2 Setting up capture control tables

In this section we will be accomplishing *Replication Center Launchpad's* **1. Create the Capture Control Tables**, see Figure 3-1. The Launchpad provides *a* good description of each step. We will not be using the Launchpad, instead we will navigate the Replication Center (RC). This more involved method will assist you in developing the skills to customize replication to meet any of your needs.

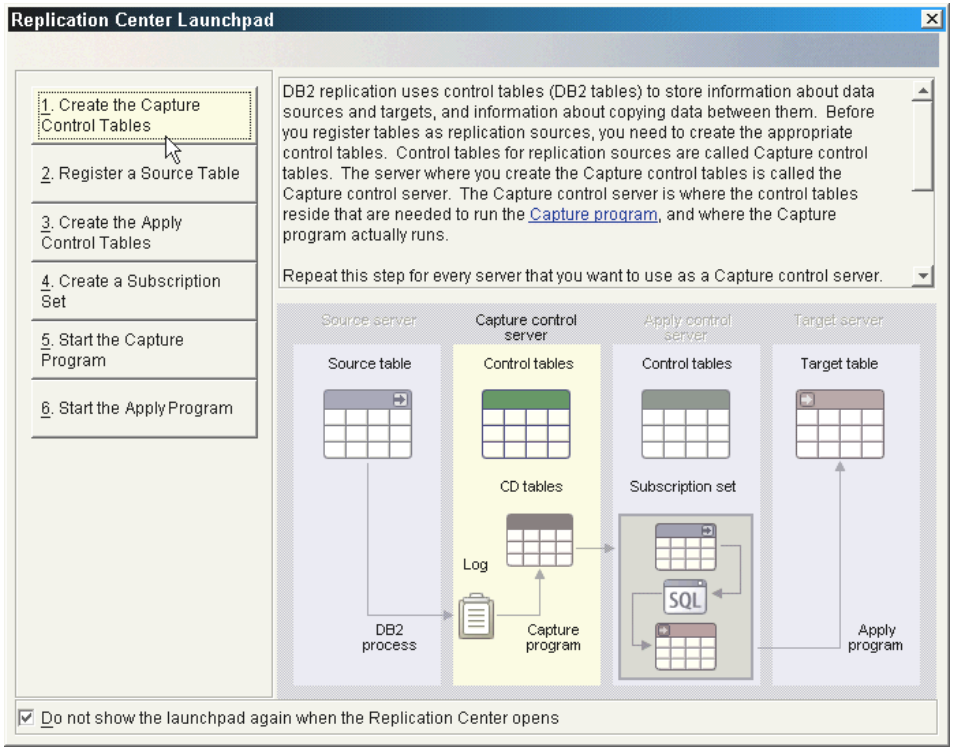


Figure 3-1 Launch Pad: Create capture control tables

Capture control tables must be created in the source database. Version 8 adds the ability to have multiple captures running verses a database, see 3.4.2, "Capture control tables, advanced considerations" on page clix

As documented in detail at Section 2.2, "Technical requirements for DB2 Replication Center" on page lxxxvi, the computer where RC is run can otherwise be uninvolved in replication.

Attention: Control tables setup on iSeries cannot be done from the RC, please see Section 3.2.2, "Platform specific issues, capture control tables" on page cl.

Within RC to open the *Create Capture Control Tables* window:

1. Double click on **Replication Definitions** so that the folders for **Capture Control Servers** and **Apply Control Servers** are shown.
2. Select **Capture Control Servers**.
3. From the menu bar at the top of the window choose **Selected —> Create Capture Control Tables —> Custom...**.

Tip: There are numerous ways to make the selects in RC described in this chapter. Often when an object is highlighted, clicking the right mouse will list the option you desire.

4. Select the database where you want to capture data, and then select **OK**.

Note: If you have not correctly set up RC to manage you password, you may be prompted for an id and password. See Section 2.11, “Managing your DB2 Replication Center Profile” on page cxiii for details on RC managing passwords.

3.2.1 Create capture control tables

Capture control table profiles were described in Section 2.11, “Managing your DB2 Replication Center Profile” on page cxiii. They populate the majority of the fields in the *Create Capture Control Tables* window. Setting up profiles is not required, but it will likely be desirable if you will be setting up a number of capture and apply servers from this computer. Here we use the default capture control table profiles.

The *Create Capture Control Tables* window allows you to choose which table space each capture control table is created in. By default, all the tables except IBMSNAP_UOW are put in one table space. The first control table in the list, IBMSNAP_REGISTER, is used to create the table space for the group. By selecting other control tables, you will see that they are set to *Use a table space already defined in this session*. Updating IBMSNAP_REGISTER’s table space name, will update all the tables that are set *Use a table space already defined in this session*. Selecting and updating any other table will affect only that table’s table space.

The capture control tables are briefly described in Table 3-1. The underlined tables are new in v8, and tables with stars * have changed since v7.

Table 3-1 List of the Capture Control Tables

Table name	Description
IBMSNAP_CAPENQ	Ensures only one Capture is running for a particular Capture Schema.
IBMSNAP_CAPMON	The data collected when monitoring Capture.
IBMSNAP_CAPPARMS	Capture Parameters, formerly known as IBMSNAP_CCPPARMS.
IBMSNAP_CAPSCHEMAS	Capture schemas.
IBMSNAP_CAPTRACE	Capture Trace, formerly known as IBMSNAP_TRACE. Contains the data collected when capture trace is run.
IBMSNAP_PRUNCNTL	Prune control information.
IBMSNAP_PRUNE_LOCK	Prune lock information.
IBMSNAP_PRUNE_SET	Prune set information.
IBMSNAP_REGISTER *	Registration information.
IBMSNAP_RESTART	Capture warm start configuration and data. Replaces v7’s IBMSNAP_WARMSTART.

Table name	Description
IBMSNAP_SIGNAL	Signals used to control the Capture program.
IBMSNAP_UOW *	Unit of work details.

These tables are described in more detail in Section 3.4.5, “Control tables described” on page clxv.

Note: A changed data (CD) table is created when a source is registered. A CD table is created for each registered table, see Chapter 4, “Replication Sources” on page clxix.

If you want to create the apply control tables in the same database, the option *Use this server as both a Capture and Apply Control Server* near the top of the window should be selected. When selected, the apply control tables will also be listed in the *Control Tables* section of the window. The first apply table, IBMSNAP_SUBS_MEMBR, is used to create the table space for the capture control tables group. The creation of the capture control tables is described in Section 3.3, “Setting up apply control tables” on page cliii. We discuss in Section 10.1, “End-to-end system design for replication” on page cdxlii why you may want to create the apply tables on the same server. At this point, we will only create the capture tables.

For this example we leave the parameters to their defaults. DB2 Multi-platform version 8 has complete functionality to modify table spaces. In previous versions, depending on the needs for simple management, we may have recommended putting the more dynamic tables in system managed spaces (SMS). For additional information on maintenance of DB2 Linux, UNIX, and Windows’ table spaces use DB2 Control Center, select a table space then from the menu bar at the top of the window choose **Selected —> Manage Storage...**.

Note: For detailed information on DB2 on Linux, UNIX, and Windows table spaces refer to the topic table space design in *DB2 UDB Administration Guide: Planning,* SC09-4822-00; the LIST TABLESPACES command in the *DB2 UDB Command Reference,* SC09-4828-00; and ALTER TABLESPACE command in the *DB2 UDB SQL Reference Volume 2,* SC09-4845-00.

The most active control table, other than the CD tables which are described in detail in Chapter 4, “Replication Sources” on page clxix, is the unit of work table (IBMSNAP_UOW). It grows and shrinks with the number of committed transactions on replicated tables and pruned. A row is inserted in the UOW table for each insert, delete, or update operation of a committed transaction against a registered replication source table.

You should initially over-estimate the space required by the table and monitor the space actually used to determine if any space can be recovered. The size of a row in the IBMSNAP_UOW is about 0.11 KB. It needs to be approximated as there are some variable length fields. We use the upper bounds as high estimates are greater than low estimates. After creating the capture control tables, you could follow the steps in Example 3-1 to verify this length.

Example 3-1 Size of row in UOW

```
CONNECT TO databaseName
DESCRIBE TABLE ASN.IBMSNAP_UOW
   This will identify each column and its size.
size = (10+10+1+18) character + 10 timestamp + (30+30) varchar
   Each of these types is 1 byte, therefore
size >= 51 bytes
size <= 109 bytes
```

You should delay consideration of sizing for the UOW table until after you have set up your registrations, see Chapter 4, “Replication Sources” on page clxix.

IBMSNAP_CAPMON and IBMSNAP_CAPTRACE can also fluctuate in size, but there growth can be limited by how the monitoring and tracing is used. You may want to re-examine the locations of these tables after reading Chapter 7, “Troubleshooting and monitoring” on page cccxxxvii. The UOW table, these two tables. and IBMSNAP_SIGNAL are pruned, see “Capture prunes applied changes” on page lxiv.

The RC is very good at assisting in tablespace creation, still you may want to refer to *DB2 UDB SQL Reference Volume 2* for the complete CREATE TABLESPACE syntax. The RC only can create database managed spaces (DMS). If for easy of maintenance, you decide to put some or all of the capture control tables in SMS, you will have to do it outside of RC. SMS provides poorer performance. If you do want to use SMS you would do something similar to Example 3-2 using the DB2 Command Center.

Example 3-2 Create SMS

```
CONNECT TO databaseName
CREATE TABLESPACE TSASNUOW MANAGED BY SYSTEM USING ('TSASNUOW')
```

If you do create table spaces outside of RC from the DB2 Control Center or using another interface, you would select *Use an existing table space* for each of the tables that you have prepared such table spaces for, see Figure 3-2.

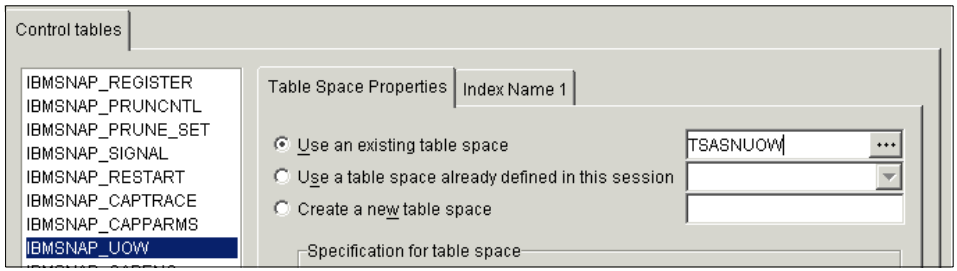


Figure 3-2 Use an existing table space

5. If you are using the defaults, or have completed your customizations, select **OK**.

Run now or save SQL

This window is described well in section 2.13, “Run Now or Save SQL” on page cxix and pictured in Figure 2-18 on page cxix.

1. A message box will appear identifying if the replication scripts were created successfully.

> **Note:** If the table space, buffer pools, or other related database objects do not exist, then the replication scripts will not be created successfully.The message will identify what is in error, and closing the message will result in the **Create Capture Control Table** window on top.

2. Read the information in the message box, and then **Close** the message box.
3. On top is now the *Run now or save SQL* window, see Figure 2-18 on page cxix.
4. We prefer to save a copy of the scripts before running. Saving the scripts is useful if you have a problem with RC. The scripts are SQL, and are a great reference. They can also be modified, and used to setup capture on another system of the same platform family. Select *Save to file* in the *Processing option* section.
5. In the *Save specifications* sections you can choose any of the systems cataloged. We save it to the machine that we are setting up capture on.
6. The *User ID* and *Password* fields are self explanatory.
7. Give a complete, existing path ending with a new filename to be created, example: `/home/prodDB2/repDDL/capCtrlCrt`.

Note: The CONNECT TO *database* statement in the script does not include your user identification and password.

8. Be careful to choose **Apply**, and not **OK**. If you choose **OK**, the script will be saved, but you will be back at the main RC window. If you choose **OK**, hen you could use the Command Center to import and after including your connectivity and authentication information, run the script.
9. select *Run now* in the *Processing option* section.
10. Confirm that the *Run Specifications* is filled in.
11. Select **OK**. A message box will inform you if successful. If successful, clicking **Close** in the message box will result in the RC window being on top.

Removing capture control tables

The Replication Center (RC) makes it easy to remove the control tables already created.

Within RC to remove previously created capture control tables:

1. Double click on **Replication Definitions** so that the folders for **Capture Control Servers** and **Apply Control Servers** are shown.
2. Select **Capture Control Servers**.
3. Select the database where you want to remove the control tables.
4. From the menu bar at the top of the window choose **Selected —> Drop Capture Control Tables**.
5. Confirm that the correct capture schema is shown in the drop down *Capture schema* field.
6. If you want RC to try and drop the table spaces as well, select *Drop table spaces used only by these control tables*.
7. Selecting **OK** will generate the SQL scripts.
8. A message box will inform you if successful. If successful, clicking **Close** in the message box will result in the **Run Now or Save SQL** window.

You can also use this method to drop them at a later time. You will be warned if the tables contain data from replication transactions. Be sure to stop all replication operations that involve these control tables before taking this action. Also, remember that you cannot easily recover a dropped tablespace.

3.2.2 Platform specific issues, capture control tables

DB2 for z/OS and OS/390

The capture control server on DB2 for z/OS and OS/390 is subsystem. When creating the capture control tables, the database for the tables should also be specified. Replication Center does not create the database. It should be created beforehand. Although an existing database can also be used, it is recommended to create a separate database for Capture.

The default and recommended table spaces created are different then on Linux, UNIX, and Windows.

Table 3-2 z/OS default table spaces

Table space name	Description
TSASNUOW	Stores IBMSNAP_UOW table.
TSASNCR	Stores tables with ROW lock size.
TSASNCP	Stores tables with PAGE lock size.

The default tablespace type used for all tablespaces is segmented.

The following can be specified for the tablespace:

- Number of pages per segment: Segment size can be optimized based on the sizes of tables in the tablespace. Refer to Section , “Sizing for capture control tables” on page clxii to estimate the table size.
- Locksize: It is recommended to accept the default locksize displayed for that tablespace.
- Encoding schema: If encoding schema is not provided, it defaults to the encoding schema of the database. If you are using V7 unicode, refer to See Appendix B, “UNICODE and ASCII encoding schemas on z/OS”, in *IBM DB2 UDB Replication Guide and Reference,* SC27-1121-00*,* for the usage and the restrictions.
- Buffer pool: You may prefer to specify a different buffer pool for performance reasons. The performance implications of specifying buffer pools is discussed on Chapter 10, “Performance” on page cdxli.
- Storage group: This parameter determines the physical volumes of the tablespace. If it is not specified, defaults to the storage group of the database.
- Minimum primary and secondary space allocation: You can alter the defaults based on your estimations. Refer to Section , “Sizing for capture control tables” on page clxii to estimate the tablespace sizes.

DB2 for iSeries and OS/400

The tables are automatically created when Data Propagator is installed on this platform family. Although the RC can be used for other tasks, it does not support creating control tables. See “Platform specific issues, apply control tables” on page clvii. Also , “Creating multiple sets of capture control tables” on page clx for how to create additional sets.

Also created, and changed from previous versions, are IBMSNAP_AUTHTKN, IBMSNAP_REG_EXT.

Non-DB2 relational sources, Informix

You must have the source, such as Informix, properly configured as a federated Server in a DB2 UDB ESE or DB2 Connect EE Version 8 database. See Appendix C, “Configuring federated access to Informix” on page dxxxvii for requirements and instructions for configuring federated access to Informix. The DB2 UDB ESE Version 8 server with the database containing the federated Server definition could be on the Informix server, on the target server, or anywhere between. See Chapter 10, “Performance” on page cdxli section on ‘End-to-end system design for replication.’ Particularly, see the sub-topic ‘Replicating from non-DB2.’

You must use the RC to create the control tables. If you are accessing more than one Informix source database from a single DB2 V8 federated database, then the nicknames for the Capture Control Tables in each Informix source database must have its own unique capture schema, see “Creating multiple sets of capture control tables” on page clx for details.

Select the *Use this DB2 federated server to capture changes on a non-DB2 server*. In the non-DB2 server field, select the name of the federated Server definition for the Informix server. This will create control tables on the source and nicknames are created on the DB2 federated database. See Table 3-3 for the control tables that are created.

Table 3-3 Listing of capture control tables for non-DB2 relational sources

Table name	Description
IBMSNAP_PRUNCNTL	Contains prune control information.
IBMSNAP_PRUNE_SET	Contains prune set information.
IBMSNAP_REG_SYNCH	Contains registration synchronization information.
IBMSNAP_REGISTER	Contains registration information.
IBMSNAP_SEQTABLE	Sequencing table, Informix only.

Table name	Description
IBMSNAP_SIGNAL	Contains signals used to control the Capture program.

These tables are described in more detail in 3.4.5, “Control tables described” on page clxv.

It is recommended that for an Informix server, that the remote schema for the Capture Control tables be in lower case. In the Manage Control Tables Profile, if you update it before opening the Create Control Tables window, select Platform Informix, then enter the remote schema in lower case and surrounded by double-quotes. For example:

```
Capture Schema = VIPER_IDS
Remote schema name = “db2repl”
```

Replication Center will use DB2 Federated Server’s *Set Passthru* capability to create the control tables in Informix and the procedures and triggers that also need to be created there. Triggers are created automatically on the signal table (IBMSNAP_SIGNAL) and the register synchronization table (IBMSNAP_REG_SYNCH). The IBMSNAP_SEQTABLE is used in Informix for the process of generating sequence numbers for the change records that will be inserted into staging tables. The IBMSNAP_REG_SYNCH table is used in the process of updating the signals in the IBMSNAP_REGISTER table in Informix before Apply reads this table to see if there are new changes available in the staging tables. All the control tables created in Informix, except the IBMSNAP_SEQTABLE, have an associated nickname that is created in the DB2 database that has the Server definition to Informix.

Important: Do not remove or change the triggers on IBMSNAP_SIGNAL or IBMSNAP_REG_SYNCH.

After you create the Capture Control Tables in Informix, and nicknames for them in a DB2 ESE or DB2 Connect EE database, you will notice in Replication Center’s Replication Definitions for Capture Control Servers, a separate object for the Informix server. The object will indicate the federated Server name for the Informix server and the DB2 database that contains the Server definition.

3.3 Setting up apply control tables

In this section we will be accomplishing **Replication Center Launchpad**'s *3. Create the Apply Control Tables*, see Figure 3-3. Note we have skipped *2. Register a Source Table*. Registering source tables and creating apply control tables are independent of each other. They can be done in either order. Refer to Chapter 4, "Replication Sources" on page clxix for details on registering source tables. The Launchpad provides *a* good description of creating apply control tables.

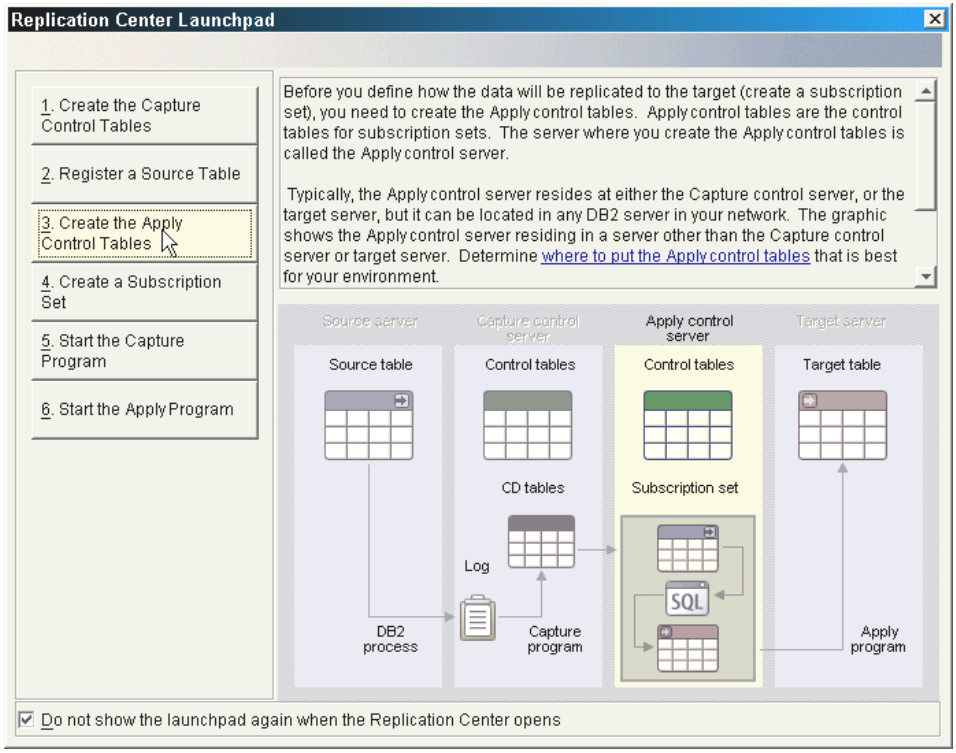


Figure 3-3 Launchpad: Create apply control tables

Creating apply control tables and their considerations involves many of the same concepts as creating capture control tables. As in 3.2, "Setting up capture control tables" on page cxliii, we will not be using the Launchpad. Instead, we will navigate the Replication Center (RC). Again, this more involved method will assist you in developing the skills to customize replication to meet any of your needs.

Attention: Control tables setup on iSeries cannot be done from the RC, please see 3.3.2, “Platform specific issues, apply control tables” on page clvii.

Unlike the capture control tables, the apply control tables can be on any database server. For now we will create the apply control tables on the target server. We will visit the issue of creating apply control tables on different database servers in 3.4.3, “Apply control tables, advanced considerations” on page clxi.

Within RC to open the **Create Apply Control Tables** window:

1. Double click on the **Replication Definitions** so that the folders for **Capture Control Servers** and **Apply Control Servers** are shown.
2. Select **Apply Control Servers**.
3. From the menu bar choose **Selected —> Create Apply Control Tables —> Custom...**.
4. Select the database where you want to replicate to, and then **OK**.

Note: If you have not correctly set up RC to manage you password, you may be prompted for an id and password. See 2.11, “Managing your DB2 Replication Center Profile” on page cxiii.

3.3.1 Creating apply control tables

The **Create Apply Control Tables** window allows you to choose which table space each apply control table is created in. This is the same as the **Create Capture Control Tables** window. Please review 3.2.1, “Create capture control tables” on page cxlv if you are not familiar with create control tables windows.

For Apply you cannot change the schema. Apply control tables always have a schema of ASN. By default all the tables are put in one table space.

The first control table in the list, IBMSNAP_SUBS_SET, is used to create the table space for the group. By selecting other control tables, you will see that they are set to *Use a table space already defined in this session*. Updating BMSNAP_SUBS_SET’s table space name, will update the group’s table space.

Selecting and updating any other table will affect only that table’s table space. See section 3.2.1, “Create capture control tables” on page cxlv for additional information on table spaces. The apply control tables (underlined tables are new in v8, and tables with stars * have changed since v7) are shown in Table 3-4.

Table 3-4 List of apply control tables

Table name	Description
IBMSNAP_APPENQ	Ensures only one Apply is running for an apply qualifier.
IBMSNAP_APPLYTRACE	Contains the data collected when monitoring Apply.
IBMSNAP_APPLYTRAIL *	Contains audit information about all subscription sets cycles performed by Apply.
IBMSNAP_APPPARMS	Contains apply parameters.
IBMSNAP_COMPENSATE	A temporary table for compensation processing for subscription-sets with more than 150 members.
IBMSNAP_SUBS_COLS *	Contains subscription columns.
IBMSNAP_SUBS_EVENT *	Contains subscription events for all subscription sets.
IBMSNAP_SUBS_MEMBR *	Contains subscription members for all subscription sets.
IBMSNAP_SUBS_SET *	Contains the subscription sets’ identifiers.

Table name	Description
IBMSNAP_SUBS_STMTS	Contains before or after SQL statements or stored procedures calls that you define for a subscription set.

Attention: The IBMSNAP_APPPARMS is not currently listed in the **Create Apply Control Tables** window. It is created. You can verify this or modify the properties of it's table space in the **Run now or save SQL window**. Unlike the IBMSNAP_CAPPARMS table which RC has the *Manage Values in CAPPARMS...* this table is not yet modifiable using the RC.

These tables are described in more detail in 3.4.5, "Control tables described" on page clxv.

Only IBMSNAP_APPLYTRACE and IBMSNAP_APPLYTRAIL possibly could require much space. You may want to manually prune these tables from time to time. These use of these tables will be discussed in detail in Chapter 7, "Troubleshooting and monitoring" on page cccxxxvii.

For our purposes we leave the parameters to their defaults.

Run now or save SQL

Select **OK** in the **Create Apply Control Tables** window, and follow the same steps you followed in "Run now or save SQL" on page cxlviii.

Removing apply control tables

Within RC to remove previously created apply control tables:

1. Double click on **Replication Definitions** so that the folders for **Capture Control Servers** and **Apply Control Servers** are shown.
2. Select **Apply Control Servers**.
3. Select the database where you want to remove the control tables.
4. From the menu bar at the top of the window choose **Selected —> Drop Apply Control Tables**.
5. If you want RC to try and drop the table spaces as well, select *Drop table spaces used only by these control tables*.
6. Selecting **OK** will generate the SQL scripts.
7. A message box will inform you if successful. If successful, clicking **Close** in the message box will result in the **Run Now or Save SQL** window.

You can also use this method to drop them at a later time. You will be warned if the tables contain data from replication transactions. Be sure to stop all replication operations that involve these control tables before taking this action. Also, remember that you cannot easily recover a dropped tablespace.

3.3.2 Platform specific issues, apply control tables

DB2 for z/OS and OS/390

Like the capture control server, the apply control server is a subsystem on DB2 for z/OS and OS/390. Therefore the database where the tables will be defined should also be specified. Replication Center does not automatically create the database. It can be a separate database created only for apply control server or an existing database.

Tables are created on two tablespaces. The tables are distributed to tablespaces based on the locksize.

Tablespace specifications are the same as capture control server. Refer to Section 3.2.2, “Platform specific issues, capture control tables” on page cl.

DB2 for iSeries and OS/400

The ASN schema and apply controls tables are automatically created when Data Propagator is installed on the iSeries. However, after the initial implementation if the apply controls tables are lost or corrupted in any way, they can be recreated by the using CRTDPRTBL command.

Enter the CRTDPRTBL on the command line then press enter. It will use ASN as the default schema name in the CAPCTLLIB parameter, which is the only parameter for this command. The following steps are performed when ASN is the specified for this parameter:

- Create the ASN schema, apply and capture control tables, if it doesn’t exist on the system.
- Create missing apply and capture control tables if ASN schema exist.
- Create missing indexes or triggers for apply and capture control tables if ASN schema exist.

Note: After you create the controls tables you need to grant authority to specific user profiles to access them. See chapter 2, in the *DB2 Universal Database Replication Guide and Reference,* SC27-1121-00 for details on using the GRTDPRAUT command to authorize a user to access these tables.

Apply Control Tables and non-DB2 target servers

When replicating to Informix, none of the Apply Control tables need to be created in the Informix target server. However, before a Subscription Set that will have target tables in Informix can be created, federated access to the Informix target server must first be defined in a DB2 UDB ESE Version 8 or DB2 Connect EE Version 8 database. See Appendix C, “Configuring federated access to Informix” on page dxxxvii for requirements and instructions for federated access to Informix.

It is not a requirement that the federated Server definition for the Informix target server be in the same DB2 database that contains the Apply Control Tables, but we expect that this will be the usual configuration. See Chapter 10, “Performance” on page cdxli section on ‘End-to-end system design for replication.’ Particularly, see the sub-topic ‘Replicating to non-DB2.’

3.4 Advanced considerations

At this point you have created the control tables necessary to perform replication. If you are new to replication you to DB2 and replication you may want to skip to Chapter 4, “Replication Sources” on page clxix. There will look at some additional considerations. Some of these will be factors of scenarios explored in Chapter 9, “Advanced Replication Topics” on page cdiii and Chapter 10, “Performance” on page cdxli.

3.4.1 Creating control tables at a command prompt

Some enterprises have numerous systems involved in replication. It would be tedious to create a large number of control table sets using RC. Thankfully, there are two alternates that allow running from a command prompt or scripting: modifying RC generated SQL, or using the new in version 8, ASNCLP.

For both the capture and apply control tables create examples we recommended saving the SQL to file before running them. The control tables are constructed using standard SQL. There is no special logic, and the SQL generated is readily understandable. It can be modified by hand or scripted to generate additional control table sets. Running the SQL directly can be useful when troubleshooting the creation of the control tables.

ASNCLP is a new 4th generational scripting language that can be used for replication definition operations. It is more abstract than SQL, so you can accomplish quite complex definitions in only a few lines, magnitudes less than the equivalent SQL. This addition to the DB2 Replication product is very new. We have been informed that is likely to be made available within the first year of version 8’s release. It will likely be first available on the DB2 Replication website

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

The version that we were supplied was excellent, but also restrictive in the ordering of the syntax.

3.4.2 Capture control tables, advanced considerations

Configuration

The space required for the IBMSNAP_UOW table grows and shrinks based on the number of rows inserted by the Capture program during a particular commit interval and on the number of rows that are pruned. 1.3.2, “Capture” on page xxxvii describes the unit of work role in replication. The dynamic nature of this table and importance in affect on replication latency suggests that it should

be included in your greatest performance configuration of drives. Also, putting it in it's own bufferpool can be effective in assisting obtaining good performance.

Creating multiple sets of capture control tables

New in version 8 is the ability to create multiple capture control tables verses the same database. Each Capture runs independently of other Captures. The Capture programs can concurrently read from the same DB2 log or journal. This redundancy may have a negative impact on performance. For z/OS data-sharing groups, multiple Capture programs can read from the logs of a data-sharing group.

If desired, you can register the same table to more than one Capture program. Each Capture must have its own set of capture control tables. Each set must have its own schema. The schema of a set must be unique amongst the capture control sets of that database.

Reasons why you might want to run multiple captures

You may want to separate the regularity of capturing changes for different tables. Here you have them run at different run-time priority and different capture characteristics such as pruning interval. This can improve throughput.

Or, different share holders can maintain their own replicas using the same source data, but different Capture programs.

On z/OS platforms, you can use multiple Capture programs to support a mixture of ASCII, EBCDIC, and UNICODE source tables within a single DB2 subsystem.

Notes for creating additional sets

To create additional sets, follow the same steps as 3.2.1, "Create capture control tables" on page cxlv. Things to take note:

- As before select the **Capture Control Servers** folder. We were tempted to select the database, but then neither right mouse click nor the menu bar choice **Selected** presented an action to create capture control tables.
- The default capture control table profiles will put these control tables in their own, new tablespaces. If you have modified the profile you may want to review the values shown in the **Create Apply Control Tables** window and the script generated, and shown in the **Run Now or Save SQL** window.
- Enter a unique schema, see Figure 3-4. RC will not give you an error when you select **OK** if you do not.

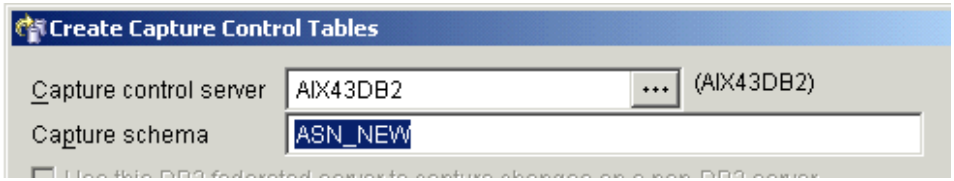


Figure 3-4 Enter a new, unique capture schema

DB2 for iSeries and OS/400

Use the CRTDPRTBL command to create additional capture control tables sets. Enter : 'CRTDPRTBL CAPCTLLIB(*NEWSCHEM*)' on the command line. The capture control tables are created in the schema specified in the CAPCTLLIB parameter. A maximum of 25 schemas can be created.

If the capture schema you specified already exist, then it will check and recreate any capture controls tables, indexes and triggers that are missing.

Note: After you create additional capture schemas, the capture controls tables needs authority to specific user profiles to access them. See chapter 2, in the *DB2 Universal Database Replication Guide and Reference,* SC27-1121-00 for details on using the GRTDPRAUT command to authorize a user to access these tables.

non-DB2 source server

The capability for multiple Capture Schemas permits a single DB2 UDB ESE or DB2 Connect EE Version 8 database to be used to replicate from multiple non-DB2 source servers, for instance, Informix. The nicknames in the DB2 database for the control tables at each Informix must have a different Capture Schema. For instance, before opening the Create Capture Control Tables for a particular Informix server, first open the Manage Control Tables Profiles, select platform Informix, and enter a different Capture Schema. Click Apply at the bottom of the Control Tables Profile window, and then Close, and then go and Create the Capture Control tables.

3.4.3 Apply control tables, advanced considerations

Where to create apply control tables

You can have any number of subscriptions in the same set of apply control tables. The apply control tables can be created on any database server that can be connected to where you run apply. As previously described, it is generally preferred to have the apply tables in the target database on each target server. This server is also where we recommend running capture. This keeps things simple, and generally gives the best performance as it minimizes network

communication. There are scenarios where you may want to trade the simplicity to achieve better performance, security or manageability. See 10.4, “Apply Performance” on page cdlxviii

Creating multiple sets of apply control table

Multiple sets are most desirable when you have multiple subscription servers, or different availability requirements. As described in the previous section, you will most often obtain the best performance by have apply control tables on each target server.

3.4.4 Sizing tablespaces for control tables

The size of the capture and apply control tables depend on the number of replication objects you define. For the most part, these tables are fairly small. The exception is the *capschema*.IBMSNAP_UOW table. The tables in this section will help you determine the space needed for the control tables.

Sizing for capture control tables

The capture control tables should be placed in at least two different tablespaces, one tablespace for *capschema*.IBMSNAP_UOW and one tablespace for all the other capture control tables.

There are no tablespaces on the iSeries platform. Informix control tables are created in the Informix source. Table 3-5 is sizing information for the capture control tables except for IBMSNAP_UOW. Control tables that are created only for DB2 source servers are marked as DB2. Control tables that are created for both DB2 and Informix source servers are marked as DB2 and Informix.

Table 3-5 Tablespace sizing for capture control tables

Capture control table	Row length	Number of rows
IBMSNAP_CAPSCHEMAS DB2	30	Number of capture schemas
IBMSNAP_CAPENQ DB2	9	none
IBMSNAP_CAPMON DB2	94	1 row inserted every Capture MONITOR_INTERVAL. Rows are eligible for pruning after MONITOR_LIMIT is reached
IBMSNAP_CAPPARMS DB2	1105	1

Capture control table	Row length	Number of rows
IBMSNAP_CAPTRACE DB2	1042	1 row for each Capture message. Rows are eligible for pruning after TRACE_LIMIT is reached.
IBMSNAP_PRUNCNTL DB2 and Informix	588	1 row for each subscription member (target table) that copies from a table registered in this capture schema
IBMSNAP_PRUNE_LOCK DB2	1	none
IBMSNAP_PRUNE_SET DB2 and Informix	74	1 row for each subscription set that is listed in IBMSNAP_PRUNCNTL
IBMSNAP_REGISTER DB2 and Informix	973	1 row for each registered table in this capture schema
IBMSNAP_RESTART DB2	50	1
IBMSNAP_SIGNAL DB2 and Informix	581	variable, depending on Apply and user activities. Rows with SIGNAL_STATE = 'C' (completed) are eligible for pruning.

The unit of work (UOW) table is volatile. Capture inserts a row into this table whenever a commit is issued for a transaction that involves replication sources. After the captured changes for the transaction have been applied to all replication targets, Capture deletes the associated rows from the UOW table.

The UOW table exists only on DB2 source servers. Table 3-6 is a worksheet to help you estimate the space needed for the UOW table.

Table 3-6 Sizing worksheet for IBMSNAP_UOW

You gather this information	Calculations
UOW row life = max(max(Apply interval),prune_interval) Apply interval is the frequency you run Apply to process changes. Prune interval is the frequency you prune changes	———— minutes

You gather this information	Calculations
UOW row rate = number of transactions (UOWs) involving replication sources that occur during the UOW row life	———— number of transactions
UOW minimum size	UOW row length (109) * UOW row rate
UOW exception factor = multiplier to account for delays or problems (like a network outage) that might prevent changes from being applied. You should always start with at least an exception factor of 2.	———— (should be 2 or more)
UOW adjusted size	UOW minimum size * UOW exception factor

Sizing for apply control tables

Table 3-7 is sizing information for the apply control tables:

Table 3-7 Tablespace sizing for apply control tables

Apply control table	Row length	Number of rows
IBMSNAP_APPENQ	18	1 for each Apply started on this server
IBMSNAP_APPLYTRACE	1060	1 row for each Apply message. Rows must be pruned manually.
IBMSNAP_APPLYTRAIL	1403	1 row each time Apply processes a subscription set. Rows must be pruned manually.
IBMSNAP_APPPARMS	1087	1 row for each apply qualifier
IBMSNAP_COMPENSATE	31	variable, used for conflict compensation in update anywhere replication
IBMSNAP_SUBS_COLS	483	1 row for each column in each target table
IBMSNAP_SUBS_EVENT	48	1 row for each posted event. Rows must be pruned manually.
IBMSNAP_SUBS_MEMBR	2570	1 row for each target table
IBMSNAP_SUBS_SET	207	1 row for each subscription set

Apply control table	Row length	Number of rows
IBMSNAP_SUBS_STMTS	1115	1 row for each SQL statement or stored procedure call defined for a subscription set

3.4.5 Control tables described

The *DB2 UDB Replication Guide and Reference* has a chapter titled “Table structures” which discusses each of the DB2 Replication tables. Currently, two tables are missing from that document, IBMSNAP_APPPARAMS and IBMSNAP_COMPENSATE.

IBMSNAP_APPPARMS

The Apply Parameter table overwrites the environment defaults for the Apply program. It is created on the apply control server. The table is read on start of Apply. Example 3-3 shows the apply parameter table’s data definition language (DDL). You can only have one row in this table for each apply qualifier. The table columns have the same properties as the Apply parameters of the same names. Table 3-8 describes the table columns. Any value in these columns will be overroad by values that you supply to Apply on start. The apply parameters are demonstrated in Chapter 6, “Operating Capture and Apply” on page cclxv. Detailed information for the parameters is available in the *DB2 UDB Replication Guide and Reference* in the chapter titled “Operating the Apply program.” Many of these parameters do not apply to, nor are created on iSeries.

Currently you have to manually run SQL to change the values of this table. Example 3-4 shows how to generate a row for an apply qualifier and updating one of the columns. This affects the behavior of Apply on that apply qualifier on subsequent starts. Until the functionality is added to RC, these parameters are not shown when you start Apply.

Example 3-3 IBMSNAP_APPPARMS DDL

```
CREATE TABLE ASN.IBMSNAP_APPPARMS(
APPLY_QUAL                      CHAR(18) NOT NULL,
APPLY_PATH                      VARCHAR(1040),
COPYONCE                        CHAR( 1) WITH DEFAULT 'N',
DELAY                           INT WITH DEFAULT 6,
ERRWAIT                         INT WITH DEFAULT 300,
INAMSG                          CHAR(  1) WITH DEFAULT 'Y',
LOADXIT                         CHAR(  1) WITH DEFAULT 'N',
LOGREUSE                        CHAR(  1) WITH DEFAULT 'N',
LOGSTDOUT                       CHAR(  1) WITH DEFAULT 'N',
NOTIFY                          CHAR(  1) WITH DEFAULT 'N',
OPT4ONE                         CHAR(  1) WITH DEFAULT 'N',
```

```
SLEEP                          CHAR(  1) WITH DEFAULT 'Y',
SQLERRCONTINUE                 CHAR(  1) WITH DEFAULT 'N',
SPILLFILE                      VARCHAR( 10) WITH DEFAULT 'DISK',
TERM                           CHAR(  1) WITH DEFAULT 'Y',
TRLREUSE                       CHAR(  1) WITH DEFAULT 'N')
IN TSASNAA;

CREATE UNIQUE INDEX ASN.IBMSNAP_APPPARMSX
ON ASN.IBMSNAP_APPPARMS(APPLY_QUAL ASC);
```

Table 3-8 IBMSNAP_APPPARMS Columns Described

Column name	Description
APPLY_QUAL	The Apply qualifier that identifies which subscription sets for Apply to run.
APPLY_PATH	Is used for all the file I/O of Apply. The value, if set, must be a valid path on the apply control server. Apply generates a operations log file in this directory. This directory will also be used for spill files, unless the SPILLFILES column is set, or overrode at Apply start.
COPYONCE	Update to ‘Y’ if you want Apply to stop after running each subscription set once.
DELAY	This value represents the amount of time in seconds to wait between apply runs. The default is 6. 10000 is the maximum valid value, and is equivalent to about 2.5 hours. The delay between runs is also influenced by the value of sleep of your subscription sets.
ERRWAIT	If error, how much time in seconds to wait before Apply retries. The default is 300 seconds (5 minutes). It can be set to zero seconds. We do not have information on a maximum value.
INAMSG	Update to ‘N’ if you want Apply not to issue a message when it becomes inactive.
LOADXIT	Update to ‘Y’ if you want Apply to invoke the ASNLOAD exit routine to refresh the target tables.

Column name	Description
LOGREUSE	Update to 'Y' if you want Apply to overwrite its operations log file when Apply is started.
LOGSTDOUT	Update to 'Y' if you want Apply to log it's operations to standard out as well as the operations log file.
NOTIFY	Update to 'Y' if you want Apply after processing a subscription to inform (notify) the ASNDONE exit routine.
OPT4ONE	Update to 'Y' if you want the Apply program to cache member and column data. Member or column changes require stop/restart of Apply. This is not feasible if there are multiple subscription sets.
SLEEP	Update to 'N' if you want Apply to terminate, instead of sleeping, when it finishes processing subscription sets.
SPILLFILE	The only valid value on Linux, UNIX, or WIndows is 'DISK'. On z/OS the default is VIO. You can specify to store it on disk, then the Apply program uses the specifications on the ASNAPLDD card to allocate spill files.
SQLERRCONTINUE	Update to 'Y' if you want Apply to continue when it encounters an SQL error message. This is not recommended for production environments.
TERM	Update to 'N' if you do not want Apply to terminate when DB2 terminates on the Apply Control Server.
TRLREUSE	Update to 'N' if you want to overwrite the IBMSNAP_APPLYTRAIL on Apply start. See Chapter 7, "Troubleshooting and monitoring" on page cccxxxvii for additional information about the apply trail.

Example 3-4 Initialize and modify the parameters for Apply

```
CONNECT TO databaseName
INSERT INTO ASN.IBMSNAP_APPPARMS (APPLY_QUAL) VALUES ('applyQualifer')
UPDATE ASN.IBMSNAP_APPPARMS SET applyParameter = newValue
```

```
   WHERE APPLY_QUAL = 'applyQualifer'
```

IBMSNAP_COMPENSATE

The compensate table was introduced in a version 7 fixpak. It is created on the apply control server. The compensation table is used for update-anywhere replication with conflict detection. It is automatically used when there are more than 150 subscription set members in a subscription set.

Example 3-5 IBMSNAP_COMPENSATE DDL

```
CREATE TABLE ASN.IBMSNAP_COMPENSATE(
APPLY_QUAL                      CHAR( 18) NOT NULL,
MEMBER                          SMALLINT NOT NULL,
INTENTSEQ                       CHAR( 10) FOR BIT DATA NOT NULL,
OPERATION                       CHAR(  1) NOT NULL)
IN TSASNAA;

CREATE UNIQUE INDEX ASN.IBMSNAP_COMPENSATX
ON ASN.IBMSNAP_COMPENSATE(
APPLY_QUAL                      ASC,
MEMBER                          ASC);
```

4

Replication Sources

This chapter discusses the following topics:

- What is a replication source?
- Define a replication source
- Views as replication sources

4.1 What is a replication source?

A replication source is the data you want to replicate and it can be any of the following:

- A DB2 table.
- A view defined on one DB2 table.
- A view defined as the inner join of DB2 tables.
- A non-DB2 relational source.
- A subset of DB2 table or non-DB2 relational source.

For replication sources on the iSeries, see 4.2.4, “iSeries replication sources” on page clxxxvi

Please refer to Chapter 3, “Registering tables and views as replication sources”, in *IBM DB2 Universal Database Replication Guide and Reference,* SC27-1121-00 for the list of supported non-DB2 relational sources.

4.2 Define a replication source from Replication Center

In order to replicate data, the replication source must be registered. The replication source is registered to a capture control server. Replication sources are selected from the objects of the capture control server. This implies your capture control server must be the one where your source data for replication resides. The capture control tables should already be created for that server andcapture control server should already be added to the Replication Center. See Chapter 3, “Replication control tables” on page cxli.

It is possible to use more than one capture schema within one capture control server. See Chapter 3.4.2, “Capture control tables, advanced considerations” on page clix.

4.2.1 Registering the replication sources

Registration is done from the Capture Control Servers folder which is under the Replication Definitions folder. Follow the steps below to access the screen to register a replication source:

1. Expand **Replication Definitions**.
2. Expand **Capture Control Servers**.
3. Find and expand the Capture server where your replication source resides.
   – If the data source is non-DB2, select the Capture Control Server icon that indicates the federated Server definition name for the non-DB2 source,

and the database name of the DB2 ESE database that contains this Server definition. For example: if the Capture Control Server name is '*IDS93_ITSO / FED_DB*',

- '*IDS93_ITSO*' is the Server definition name in a DB2 ESE database
- '*FED_DB*' is the name of the DB2 ESE database containing the Server definition.

4. If you have created more than one set of capture control tables for this capture control server, from the list of capture schemas expand the one you want to register.
5. From the option list, either select **Register Tables** or **Register Views**.
   – Or if the source server is non-DB2, select **Register Nicknames**

   In order to find out the differences related to registration of views, refer to Section 4.3, "Views as replication sources" on page cxcviii.

An alternative path to register sources from Replication Center, is the Launchpad. If you choose **Register a Source Table** from the Replication Center Launchpad, after specifying the replication source, *Register Tables* will display.

4.2.2 Selecting the replication sources

If you have chosen Register Tables, *Add Registerable Tables* automatically launches. See Figure 4-1.

If the source server is non-DB2 and you chose Register Nicknames, *Add Registerable Nicknames* window opens. If the source server is non-DB2, there needs to be a nickname for the source table in the DB2 ESE database that contains the federated Server definition for the non-DB2 source server. For stuations when a nickname does not yet exist, the Register Nicknames window has a button at the bottom left for creating nicknames. This button will open the Control Center's Create Nickname dialog. For an example of how to use Control Center's Create Nickname dialog, see Appendix C, "Configuring federated access to Informix" on page dxxxvii; specifically, look in the topic 'Creating federated objects using the Control Center.' If you call the Create Nickname dialog from Replication Center's Register Nicknames window, the new nickname will not automatically appear in the Selected nicknames field. Click the **Add** button to open the Add Registerable Nicknames window and use the filter their to retrieve the nickname just created.

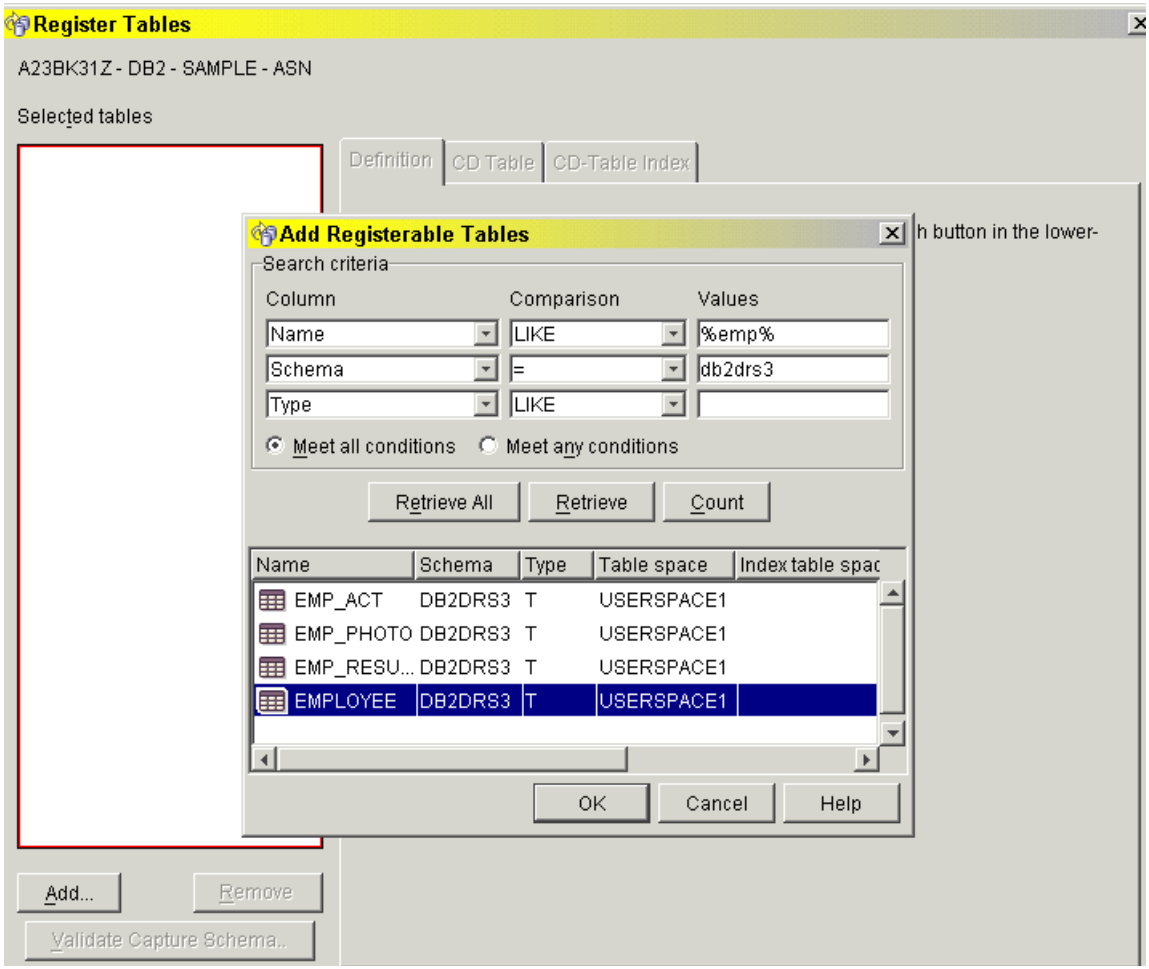


Figure 4-1 Add registerable tables

You select the tables you want to register as replication sources from the *Add Registerable Tables*.

If you use the **Retrieve All** button, your list will include all the registerable tables of the capture control server.

You can provide search criteria and use the **Retrieve** button on this window for filtering. The search criteria columns are based on the columns that exist in the DB2 catalog and specific to the platform of the capture control server. The most commonly used ones for DB2 UDB for UNIX and Windows are the name, schema. On Series the schema equates to libraries.

If you are unsure how many table descriptions may be selected, **Count** button gives the number of rows satisfying the selection. This is much more efficient than returning all the list.

You may find filtering even more useful, if the replication source server is DB2 UDB for z/OS, since the list will include all eligible tables from the subsystem catalog. Selection can be based on many different criteria among which database or creator are most commonly used.

It is not possible to register the Change Data (CD) tables.

The DB2 UDB for z/OS tables with EDITPROC or VALIDPROC defined on them can not be the replication sources thus does not appear on the registrable tables list.

Attention: This list does not include the sources which are already registered for this capture schema of the capture control server.

Important: If you want to capture a source multiple times, you need to define multiple capture schemas.

4.2.3 Defining the registration options

You can define options for more than one replication source on *Register Tables.* You see the screen where replication source options are specified on Figure 4-2. Besides the possibility of selecting more than one table with CTRL and Shift keys on *Add Registrable Tables*, you can also add tables using the **ADD** button below on the *Selected tables*

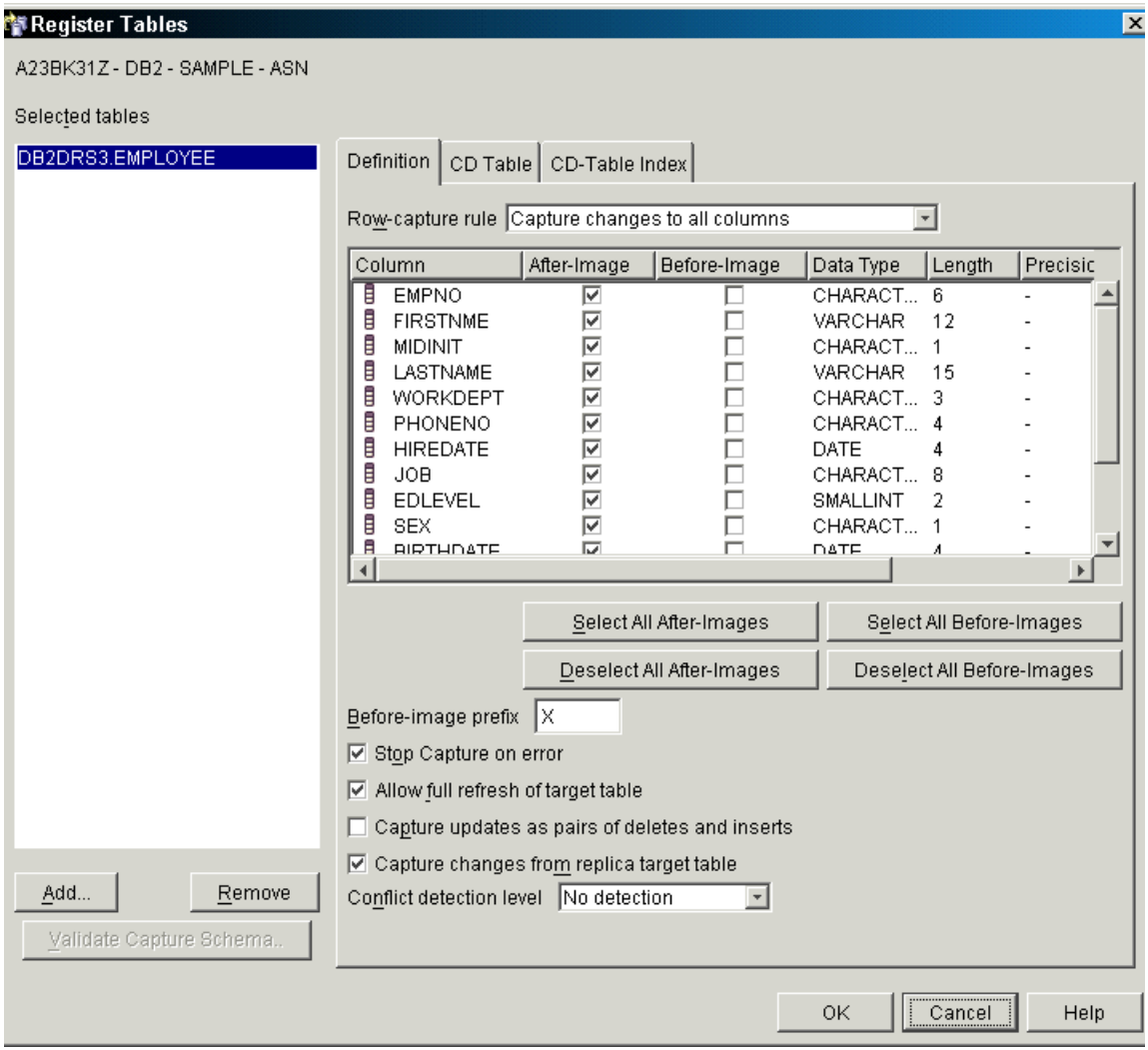


Figure 4-2 Register tables

There are a number of options you can specify when registering a source.

Note: If you are reading this book just to do hands-on example with Replication center, you can accept the defaults on this screen and continue with Section 4.3, “Views as replication sources” on page cxcviii.

Continue to read this section, if you want to read on registration options or CD table properties.

Registering the columns (vertical subsetting)

You specify the columns of the replication source which will be replicated to the target by checking the *after-image*s next to them on *Register Tables*. The default is to replicate all columns of the replication source. You can change the default by de-selecting the columns you do not want at the target. On this window you can also check the *before-image* boxes if you also need the unchanged data to be captured. Next to *column names* and *after-image*, *before-image* check boxes, there are informational columns like *Data Type* and *Length* which can not be changed.

If the capture control server is DB2 UDB for z/OS, columns with field procedures defined on them, should not be registered. Columns with structure data types (ADT) can not be registered, if the replication source is on DB2 UDB for UNIX and Windows.

One change-data (CD) table is created on the control server as a result of each successfully registered table. CD table is populated by the Capture program during the capture cycle with the changes to the replication source.

Selection of after-image value or after-image and before-image values of the columns affect how CD table is created. Every CD table contains columns which are used for synchronization of Capture and Apply programs. CD table also contains columns to hold changed data. It is also possible to store in the CD table, the unchanged form of the data. There will be one column for the changed data in the CD table, if you only marked the *after-image* for a column. The CD table will contain two columns to store changed data and the data before it is changed if you marked both the *after-image* and *before-image*. The columns created for storing before-image values in the CD table are prefixed by *Before-image prefix* you provide as another option at this window. The default for *Before-image prefix* is X. You can alter the default from the *Manage Source Object Profiles*. Selection of before-images as well as after-images has no effect on the columns defined for the target table.

In most cases, capturing only the changed data is enough for your requirements and you only mark for the after-images of the columns you need at the target. Capturing the before-image is not supported for most non-DB2 relational data sources. See *IBM DB2 Universal Database Replication Guide and Reference,* SC27-1121-00 for the list of supported non-DB2 relational data sources.

Capturing before-image together with after image

There are some cases where before-image of a column is necessary when applying the changes.

Target key columns are being updated:
A key for the target table is defined during subscription. Apply program uses this key to search the target table when applying the updates. If target keys are being updated, Apply program needs to know the before-image values of the target keys to correctly replicate the changes.

Figure 4-3 shows how Apply replicates the updates. There are differences on how it executes depending on whether you update the target key and whether or not you capture the before-image values.

In each of the examples, MGRNO of the department ‘A01’ is updated from ‘000010’ to ‘000020’. The target key in the first example is DEPTNO and is not updated. The Apply program searches the target based on DEPTNO and replicates the update correctly without any need of before-image values.

In the second example, MGRNO is specified as the target key which may be preferred by the DBA to take advantage of the index created for ad-hoc queries against the target table. No before-images values are captured for this example. The manager number (‘000010’) of the department with the department code ‘A01’ is updated to 000020 on the source table. Apply program uses the changed value (MGRNO=’000020’) to find the qualifying rows which does not exist on the target table. Apply then inserts a new row to the target table with (MGRNO=’000020’). Since Apply does not have the before-image value, row with (MGRNO=‘000020’) is replicated to the target table but old row (MGRNO=’000010’) is not deleted.

The target key is updated in the third example. The Apply program has the before-image values to search the target table. The update on the source table is correctly replicated to the target because Apply finds the qualifying rows on the target based on the before-image value of the target key.

Figure 4-3 How Apply replicates the updates to the target key

Important: During subscription, *Let the Apply program use before-image values to update target key columns* option must be checked and the before-image values of the target keys must be captured in order to have correct replication if target keys are being updated.

Replication source is part of Update-Anywhere model:
If in your configuration, there is a possibility that the target tables are also updated by applications other than Apply, then you must also run the Capture on the targets.

Restriction: Update Anywhere is not supported with non-DB2 source tables or target tables. DB2 Replication's Update Anywhere implementation requires that Capture run at both the source table and the target, and this is not possible when either the source or the target is in a non-DB2 database.

Since replication is an asynchronous operation, there is a probability that the same row is changed on both sides before being replicated to the other side. This is called an update-conflict. If one master and many replicas are defined (update-anywhere) , this conflict can be detected and resolved by accepting the change of master and undoing the update of the replica. Apply needs before-image values of the replica to undo the changes at the replica in case of conflicts.

If the model is peer-to-peer, since there is no master in this model, conflict detection is not possible and capture of before-image values of replica is not required. See Chapter 9.6, “DB2 Peer to peer replication” on page cdxxix for peer to peer setup.

Consider the replication environment in Figure 4-4. There is a HCUSTOMER table at the headquarters. It is replicated to branch offices BRANCH_1 and BRANCH_2 regularly. The HCUSTOMER table is updated at the headquarters. There are also applications updating the replica tables (BCUSTOMER) at branch offices independently. Both headquarters and branch offices are capture control servers and the CD tables created to be used for captured changes are called CD_HCUST and CD_BCUST respectively. Assume the following sequence of events that describes an update conflict between master (headquarters) and replica (one of the branches):

1. At time T1, customer record (say C1) is changed and committed at HCUSTOMER and this change is stored to CD_HCUST.
2. At time T2, C1 is changed and committed at BCUSTOMER and captured at CD_BCUST.
3. At time T3, during Apply cycle, C1 is read from CD_HCUST for replication to one of the branches.
4. Apply searches for C1 on CD_BCUST. A match indicates that an update conflict has occurred.
5. Change of BRANCH_1 is undone by using the before-image values on the CD_BCUST.

In update-anywhere configurations, Apply program before applying the change, searches the CD table on the target with matching keys values. If a match is found, the change of the replica is undone by using the before-image value and the master stays.

Figure 4-4 Update-anywhere replication

Important: Select the before-image values for the replicas for update anywhere model. Before-image values are not used for peer-to-peer model.

If before-image values are stored in the CD table as well as after-image values, increase on the CD table size should be considered,

Capture changes to all

Whenever any registered column is changed via insert, update or delete, this change is reflected to the CD table. It is also possible to store (in the CD table) any changes to the replication source even if none of the registered columns are changed. This implies that changes are captured without regard to columns registered.

Capturing every change is preferred from the performance point of view because Capture captures the changes without additional testing. Especially, if you have registered all the columns already, do not mark *Capture changes to registered columns only*. But, if the difference between the row size and the total registered

column size is big, then changes to all will have negative impact on the size of the CD table. You may also expect the log records produced due to inserts to CD table to be higher.

If you want Capture to capture whenever a registered column is changed only, select the *Capture changes to registered columns only* on the pull-down menu for *Row-capture rule.* Capturing changes to all columns is the default.

This option used to be set as a Capture start-up parameter (CHGONLY) which applied to all replication sources that Capture program is capturing data for. Since version 8, row-capture rule can be set while registering sources and thus the selection method is valid only for the sources registered on that window. This option is not available as a startup parameter anymore.

Full refresh only

If you specify *Don't capture changes* **(***Full refresh only***)** on the *Row-capture rule*, then the replication source will be replicated to the target as a whole on each replication cycle. If you choose full refresh only, no other option can be specified and the source table is directly copied to the target by the Apply program.

Figure 4-5 Full refresh only

On a full refresh only replication (see Figure 4-5), Apply reads the registration information from the capture control tables, accesses the replication source and replicates to the target.

Since the data on the replication source is not captured and Apply reads the replication source directly, full refresh only replication is possible without starting the Capture program.

If the replication source is a small table you may prefer full refresh only to differential (change capture) replication. Full refresh only replication does not have some of the overhead of change capture replication like updating the CD tables and generating log records for CD table updates. Not running a Capture program can also be regarded as a performance benefit.

This is the only way for replicating DB2 UDB for UNIX and Windows catalog tables. It is not possible to replicate DB2 UDB for z/OS catalog tables.

Stop on error

This option determines whether the Capture program terminates on every error or terminates when obliged to. The default for this option is yes and the Capture program terminates on every error occurred. But, there are errors that does not necessitate the Capture to terminate. If you set *Stop on error* to no, then Capture will not terminate on every error but will stop processing by either deactivating the registration or not activating the registration at all if the error occurred on first capture cycle of this registration.

Do not stop on error avoids Capture to terminate on following situations (*IBM DB2 Universal Database Replication Guide and Reference*, SC27-1121-00):

- The registration is not defined correctly.
- The Capture program did not find the CD table when it tried to insert rows of changed data.
- The Capture program started or reinitialized when the DATA CAPTURE CHANGES option on the (non OS/400) source table was set to OFF.

If the errors listed above occurred and do not stop on error is your option, STATE column of IBMSNAP_REGISTER table is set to ‘S’ (=stopped) and error message number associated with the failure is stored in STATE_INFO column for this registration.This allows you to take corrective action and then you must set the STATE column back to ‘I’ (=inactive), see Chapter 8.1.2, “Deactivating and activating registration table” on page ccclxxiv.

If the replication source is on DB2 UDB for z/OS and the tablespace is compressed, loss of compression directory causes Capture to set the STATE to ‘I’ for this replication source. Apply can immediately capstart this registration. In order to avoid Apply from capstart is to use USER or CMD/STOP signals to coordinate compression dictionary changes. If you are using DB2 compression utilities, you may prefer to coordinate the these utilities with Capture program. You can find how to coordinate these utilities with Capture program on Chapter

13, “Maintaining your replication environment”, in *IBM DB2 Universal Database Replication Guide and Reference, SC27-1121-00*.

Some errors like worker thread having a media-failure condition can not avoid Capture from terminating even if you specified do not stop on error.

Allow full refresh of target table

Full refresh is performed by the Apply program automatically for change capture replications if the Capture program is started with cold start mode or switched to cold mode. Disallowing full refresh of target table by this option prevents the Apply program doing full refresh automatically but does not remove the full refresh requirement and you should do the full refresh manually.

The default and common usage is allowing automatic full refresh of target table by the Apply program. You can change this default for any capture control server from *Manage source object profiles.* Some of the reasons for disallowing full refresh of target table are listed below.

- It is one of the steps of setting peer-to-peer replication. See 9.6, “DB2 Peer to peer replication” on page cdxxix.
- If the table requiring full refresh is a big table, replicating it completely with full refresh during heavy hours of operation may affect overall performance.

Capture updates as pairs of deletes and inserts

Normally every SQL statement (insert, update or delete) is captured and stored as it is in the CD table together with the statement type. There is an option for update statements: an update statement can be stored as a pair of delete and insert statements. Once, *Capture updates as pairs of deletes and inserts* is selected for a replication source, two rows, one delete and one insert, will be stored for all the updates to that source. This results in increase in the number of rows in the CD table and increase in the number of log records generated for the CD table but is necessary if you partition your data to the targets via row filtering and the columns you used for partitioning are updated. Row filtering is defined by a predicate during subscription. Apply checks this predicate against the captured changes and only the rows satisfying this criteria are applied to the target.If you update the columns specified in the predicate and if this update results in relocating the row to a different partition but do not capture the update as a delete/insert pair then this row will not be deleted from previous location.

Figure 4-6 Capture updates as pairs of inserts and deletes

Consider the example on Figure 4-6, the CUSTOMER table is replicated to two targets. First target is for premium customers whose account balances are more than 50000. Customers whose account balances less or equal to 50000 are regular customers and they are replicated to a different target. In order to replicate CUSTOMER to different targets two subscription sets (say S1 and S2) are defined. The premium customers predicate is defined as row filter for S1 and the predicate for regular customers is defined to S2. Apply before applying a change for either S1 or S2 first checks the *acct_balance* and skips it if it not qualifying. An update changes a customer's balance from 14000 to 64000 causing his category to be changed from regular to premium. If we assume that update is stored as an update in the CD table, during subscription cycle of S1, Apply will check the acct_balance which is 64000 and will find out that this row qualifies for S1 and will insert it into premium customers after receiving a notfound condition. Apply just skips this row during subscription cycle of S2 because it does not qualify for S2 though it has to be deleted from regular customers. If for this example, update is stored as one delete and one insert in the CD table, delete (with acct_balance=14000) will qualify during S2 and this operation will be applied to the regular customers and insert (with acct_balance=64000) will qualify for S1 and this operation will be applied to the premium customers.

If you use this option, two rows instead of one row is stored in your CD table. You should consider this when estimating the size of the CD table especially if updates are high in your environment.

Capture changes from replica target table

This option is related to update-anywhere model where data is changed at both capture and target sites and such that you choose one of the site as the master and others as replicas. You must consider whether to re-capture or not when registering for both the master and the replica target tables.

On peer-to-peer replications, where there are no master sites and all sites are replicas, capturing changes from the replica tables should be avoided. See Chapter 9.6, “DB2 Peer to peer replication” on page cdxxix.

The purpose of re-capturing is to propagate the change received from one site to others. If you have is one master and one replica as in the first example of Figure 4-7, where neither the master nor the replica has others to replicate the change received, you simply uncheck this option for both the master and the replica.

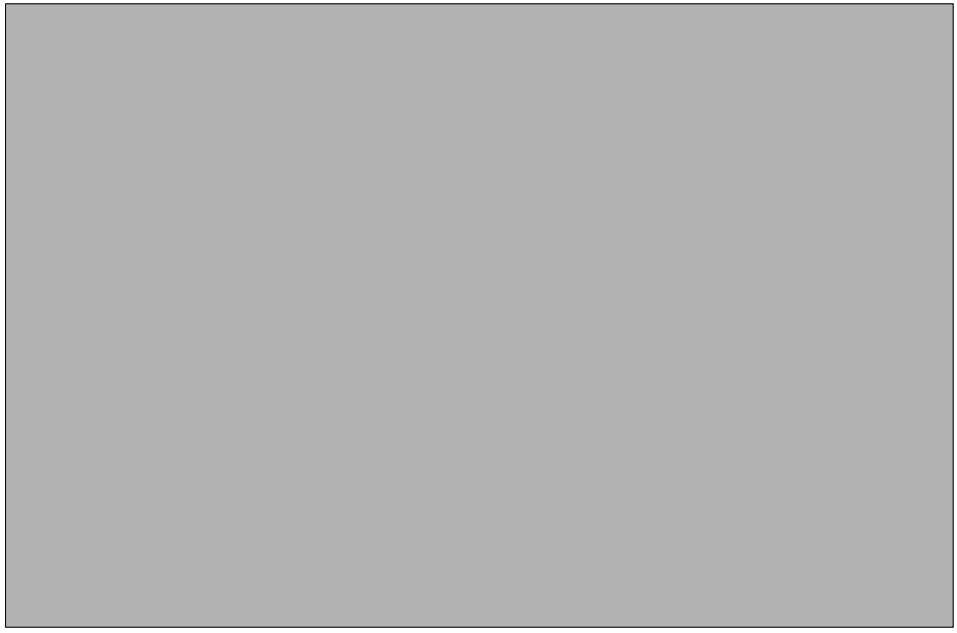


Figure 4-7 Re-capture changes

If you replicate to more than one replicas from the master as in the second example, the changes originating from replicas R1 and R2 has to be propagated to R2 and R1 respectively by the master. So, you need to re-capture changes at

the master. Re-capturing the changes originating from the master at the replicas is not necessary since they are not propagating data to any other site.

If you have distributed your data to the replicas by partitioning based on a key as in example three, then master need not re-capture the changes from the replicas as a row only belongs to one replica. A change originating from R1 will never be replicated to R2 because it is not in the range of rows replicated to R2.

If you are replicating from replicas to other replicas, you need to re-capture at the replica site which is replicating to others. On the forth example, R1 must re-capture to replicate changes originating both from M to R2 and from R2 to M.

If you are re-capturing a change, APPLY_QUAL column in IBMSNAP_UOW table which identifies the Apply program that applied the change, prevents this change to be propagated to the originating site again.

Attention: If on your registration, capture changes from replica tables is checked even if you do not need re-capturing, this may cause unnecessary updates to CD and capture control tables. Capture changes from replica tables is checked by default. Uncheck it, if re-capturing is unnecessary

Conflict detection level

Conflict detection level can be either no detection, standard detection or enhanced detection. Conflicts may occur at update anywhere and peer-to-peer models but it is not possible to detect the conflicts on the peer-to-peer model. The default for this option is no detection and setting anything else than the default is only meaningful for update anywhere replications.

If you choose standard conflict detection, Apply first searches the CD table for a matching key before applying the change at the target in order to detect the conflict but this checking can not be regarded as complete as there may be changes in the log which are not captured yet. Example 4-1 shows the occurrence of a conflict between a master and a replica which standard conflict detection method can not detect. T1,T2,T3,T4,T5 are being used to sequence the occurrence of events. V1 and V2 are the values stored to the row with the same key by master and replica.

Example 4-1 Standard conflict detection

```
T1 Replica stores V1 and commits. The change stays in the log but is not
inserted to the CD table.
T2 Master stores V2 and commits. The change is inserted to the CD table.
T3 Apply starts replication from master to replica. No conflicts are detected
since the change of replica is not in the CD table. Apply stores V2 to replica.
Both sites have V2.
T4 Capture at replica captures V1.
```

```
T5 Apply starts replication from replica to master. No conflict is detected. V2
is applied to master. Replica has V1 master has V2.
```

If you need to have a better checking, you may prefer enhanced conflict detection but the requirements of enhanced conflict detection may be hard to satisfy on every environment. It is aimed to make a complete check of changes made at the target on enhanced conflict detection. For this reason, Apply waits for the Capture to capture all the log records. This can be a never ending process. So, Apply locks the target in order to stop new changes. After all log records are read and captured to the CD table, conflict checking is performed and this checking covers all the changes made at the target.

Although, enhanced conflict detection can detect the conflicts that can not be detected by standard detection, it is almost impossible to implement this model in production environments where replication is continuous and tables are accessed by the applications almost all the time. This conflict detection method may be suitable for mobile users who occasionally connect to the server for replication and do not run applications until the replication ends.

4.2.4 iSeries replication sources

When your replication source and target server is a iSeries, there are additional registration functions we will discuss in this section. Such as remote journaling and capturing the relative record number (RRN). But first we need to consider the journal requirements.

Creating journals to register your source tables

If your source tables you are planning to replicate are already journaled, make sure the IMAGE parameter in the STRJRNPF command is *BOTH, then skip to remote journal heading.

Local Journal

Before registering your source tables to capture changed data, you must start journaling to those tables. If you specify the option for full refresh only, then you don't need to journal those source tables. The following will highlight the steps to create journals, journal receivers and starting your journals. Please, reference chapter 2 in *Replication Guide and Reference,* SC27-1121-00 for details on creating journals and managing journals

- First you need to create a journal receiver using the CRTJRNRCV command:

Example 4-2 Create Journal receiver

```
CRTJRNRCV JRNRCV(Journalreceiverlibrary/Journalreceivername)THRESHOLD(100000)
```

```
TEXT('DataPropagator Journal Receiver ')
```

- After you create a journal receiver, you will reference it in when you create the journal using the CRTJRN command:

Example 4-3 Create journal

```
CRTJRN JRN(Journallibrary/Jounralname)
JRNRCV(Journalreceiverlibrary/Journalreceivername) MNGRCV(*SYSTEM) DLTRCV(*YES)
TEXT('DataPropagator Journal')
```

- To start the journal for a source table use the STRJRNPF CL command. You can enter multiple source tables at one time.

Example 4-4 Start Journal

```
STRJRNPF FILE(Sourcetablelibrary/Sourcetable) JRN(Journallibrary/Journalname)
OMTJRNE(*OPNCLO)IMAGES(*BOTH)
```

Note: The Capture program require *BOTH for the IMAGES parameter. When ever you end journal for a source table, using the ENDJRNPF CL command. A full refresh is triggered to the target table.

Remote Journal

A remote journal is a copy of the source journal that resides on a target iSeries server. The remote journal provide the option to efficiently replicate journal entries to the remote journal that resides on one or more systems. The remote journal system management uses the following communications protocols for replicating the journal entries to the remote target severs:

- OptiConnect for OS/400.
- Systems Network Architecture (SNA).
- Transmission Control Protocol/Internet Protocol(TCP/IP).

The remote journal function, replicates journal entries to the remote system at the Licensed Internal Code layer. Moving the replication to this lower layer provides the following benefits:

- The remote system handles more of the replication overhead
- Overall system performance and journal entry replication performance is improved
- Replication to the remote journal can (optionally) occur synchronously
- Journal receiver save operations can be moved to the remote system.

Remote Journal Setup

To set up remote journal, we will highlight the following commands. For more information about the remote journal function and parameters, see *Backup and Recovery*, SC41-5304, and *OS/400 Remote Journal Function for High Availability and Data Replication*, SG24-5189.

To create a remote journal at target server, enter the ADDRMTJRN command at the source server were the local journal is located

Example 4-5 Create the remote journal

```
ADDRMTJRN RDB(RemoteDBname) SRCJRN(Journallibrary/Journalname) TEXT('Remote
journal from source system')
```

When you initially create a remote journal, the delivery state is inactive. Therefore, to start remote journal activity it needs to be activated, by performing the following steps at the source server:

- Enter the WRKJRNA JRN(*Journallibrary/Journalname*) CL Command to display the **Work with Journal Attributes** screen.
- Press the F16 to display the **Work with Remote Journal Information**. Type 13 on the option field, then F4 to display the CHGRMTJRN(**Change Remote Journal**) prompt screen.
- Enter on the delivery parameter: *ASYNC for asynchronous , *SYNC for synchronous as the journal entry delivery option.

Planning iSeries replication using remote journals

Conventional DB2 Data Propagator implementation, requires the replication source definitions, control tables and the Capture program to reside on the source server, which is the default setup.

Remote journal makes it possible to move the replication source definitions, the Capture program and its control tables away from the server on which the source tables reside, leaving more resources available on that server. Therefore, with the remote journaling, processor usage can be reduced, DASD can be saved, and performance can be improved significantly at the source server, which is normally the primary OLTP server. See, Figure 1-7 on page xxxix

With a remote journal implementation your replication source definitions and the Capture program could reside on the target server, along with the apply control tables, target tables and the Apply program. Therefore, the Capture and Apply program running concurrently on the same server will prevent the Apply program to create and use a spill file, because the Apply program will read directly from the CD table. This will also reduce the system resource used by the apply program.

If you plan to use remote journals:

- Create remote journal as described in the preceding heading: Remote Journal Setup
- Install DB2 Data Propagator for iSeries on the target server only.
- Run the CRTSQLPKG were the Capture program is running, pointing to the iSeries server were the source table resides.
- Proceed to chapter 2 in the *Replication Guide and Reference,* SC27-1121-00 for the rest of the procedure to setup replication on the iSeries.

Relative Record Number

Usually, the target table for a source uses the same key columns as the primary key columns in the source. The Apply program uses this key value to apply the data it has replicated from the source's CD table to the target. If you are registering an OS/400 table that does not have a primary key, a unique index, or a combination of columns that can be used as a unique index, you must register the table using the relative record numbers (RRN). When you choose to replicate using the RRN, both the CD table and the target table have an extra column, IBMQSQ_RRN of type INTEGER, which contains a unique value for each row. This column contains the RRN that corresponds to each source table row.

The RRN is used as a primary key for the source table row as long as the source table is not reorganized. When the source table is reorganized, the RRN of each source table row changes; therefore, the RRN in the CD and targett table rows no longer has the correct value that reflects the row's new position in the source table. Any time you reorganize a source table (to compress deleted rows, for example), a full refresh is triggered to all the target tables in the set of that source table. For this reason, place target tables that use RRN as primary keys in subscription sets with other targets that use RRNs, and not in sets with tables that use some other uniqueness factor.

Define replication source on the iSeries

You can define replication sources on the iSeries either from the Replication Center or using the ADDDPRREG CL command.

Using the Replication Center

As highlighted in Section 4.2.1, "Registering the replication sources" on page clxx and Section 4.2.2, "Selecting the replication sources" on page clxxi the following screen is displayed, when you register your source tables on an iSeries server, see Figure 4-8.

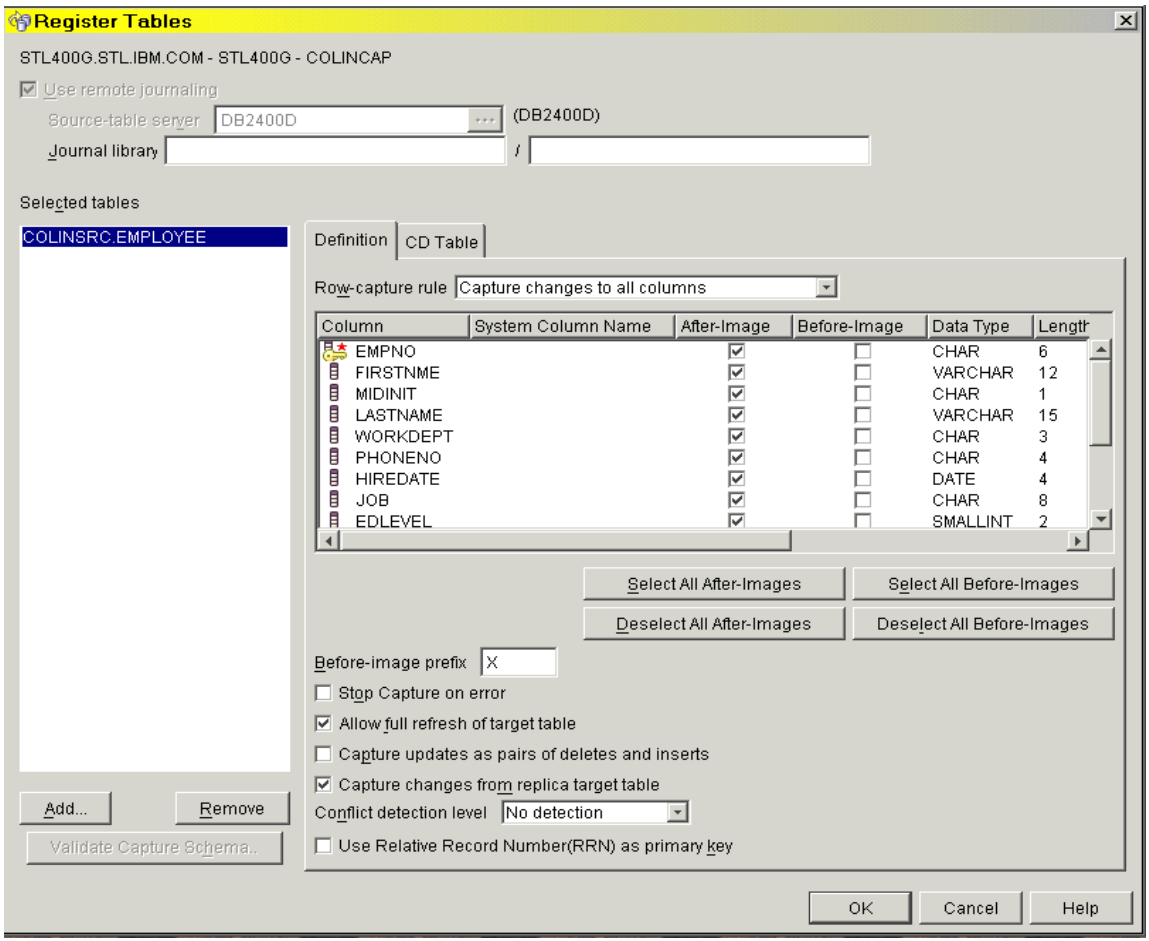


Figure 4-8 iSeries registered sources

We will describe the remote journaling and RRN option, that is indigenous on the iSeries. All the other replication sources functions are described in the sections mention in the preceding paragraph.

If you are not using remote journaling

- Retrieve your tables you want to register from the **Add Registerable Tables** window as described in Section 4.2.2, “Selecting the replication sources” on page clxxi
- Proceed to Section 4.2.3, “Defining the registration options” on page clxxiii for detail description to define replication sources

- If you want to use RRN as described in the proceeding heading, “Relative Record Number” on page clxxxix. Just click the check box **Use Relative Record Number (RRN) as primary key**

If you are using remote journaling

Note: Make sure you are defining your replication sources from the correct capture control server. Usually on the same server were the Apply program is running.

- Cancel the **Add Registerable Tables** window that is currently displayed
- Click the check box **Use remote journaling**
- Click the **...** by the **Source-table server** to display a window to select the source table server.
- Enter the remote journal library and name, which usually reside on the same server were the Capture and Apply program is running.
- Push the **Add** button to retrieve your tables you want to register from the **Add Registerable Tables** window as described in Section 4.2.2, “Selecting the replication sources” on page clxxi.
- Proceed to Section 4.2.3, “Defining the registration options” on page clxxiii for detail description to define replication sources
- If you want to use RRN as described in the proceeding heading, “Relative Record Number” on page clxxxix. Just click the check box **Use Relative Record Number (RRN) as primary key**

Using the ADDDPRREG CL command

This command will register a source table from an iSeries server. The screens on Figure 4-9 will give you an idea of the parameters that are used within the ADDDPRREG CL command. When it is entered on the command line, press the F4 key to display the prompt screen, then the F11 key to display the actual parameter names:

```
 Add DPR Registration (ADDDPRREG)

 Type choices, press Enter.

 Source table . . . . . .


   Library  . . . . . .                Name
 Capture control library  . . . .  ASN          Name, ASN
 Library for CD table . . . . . .  *SRCTBL      Name, *SRCTBL
 Name of CD table . . . . . .      *DEFAULT


 Source table type  . . . . . .    *USERTABLE, *POINTINTIME...
*POINTINTIME...
 Allow full refresh . . . . . .    *YES          *YES, *NO
 Text 'description' . . . . . .    *NONE

 Capture columns  . . . . . .      *ALL
               + for more values
 Capture relative record number    *NO           *YES, *NO

More...
 F3=Exit   F4=Prompt   F5=Refresh   F10=Additional parameters   F12=Cancel
 F13=How to use this display        F24=More keys
```

Figure 4-9 ADDDPRREG CL command screen prompt - 1st Screen

After you enter a required value for the **Source table** and **Library** press the F10 key for additional parameters, then the page down key, to display the screen on Figure 4-10.

```
 Add DPR Registration (ADDDPRREG)

Type choices, press Enter.

Record images  . . . . . .   *AFTER       *AFTER, *BOTH
```

Figure 4-10 ADDDPRREG CL command screen prompt - Last Screen

For detail information on the parameter values, you can use the field level help, by moving the cursor to the parameter and press the F1 key. Or refer to Chapter 18 in the *Replication Guide and Reference,* SC27-1121-00

4.2.5 CD Table

As a result of registration, a CD table is created on the capture control server where replication source resides.

Column name	Data type
EMPNO	CHARACTER
FIRSTNME	VARCHAR
MIDINIT	CHARACTER
LASTNAME	VARCHAR
WORKDEPT	CHARACTER
PHONENO	CHARACTER
HIREDATE	DATE
JOB	CHARACTER
EDLEVEL	SMALLINT
SEX	CHARACTER
BIRTHDATE	DATE
SALARY	DECIMAL
BONUS	DECIMAL
COMM	DECIMAL

Column name	Data type
IBMSNAP_COMMITSEQ	CHARACTER
IBMSNAP_INTENTSEQ	CHARACTER
IBMSNAP_OPERATION	CHARACTER
EMPNO	CHARACTER
XEMPNO	CHARACTER
FIRSTNME	VARCHAR
MIDINIT	CHARACTER
LASTNAME	VARCHAR
WORKDEPT	CHARACTER
XWORKDEPT	CHARACTER
PHONENO	CHARACTER
HIREDATE	DATE
JOB	CHARACTER
XJOB	CHARACTER
EDLEVEL	SMALLINT
SEX	CHARACTER
BIRTHDATE	DATE
SALARY	DECIMAL
XSALARY	DECIMAL
BONUS	DECIMAL
XBONUS	DECIMAL
COMM	DECIMAL
XCOMM	DECIMAL

Figure 4-11 Replication source and CD table example

On Figure 4-11, you see the replication source EMPLOYEE and the CD table created for EMPLOYEEE. During registration, after-images of all the columns and before-images for EMPNO, WORKDEPT, JOB, SALARY, BONUS and COMM are selected. The before-image prefix is X which is the default. This prefix is concatenated to the column names of the columns which will be used to store the before-images. All the columns except the first three are for storing changed or unchanged form of the data. The first two columns IBMSNAP_COMMITSEQ and IBMSNAP_INTENTSEQ are log sequence numbers of the commit log record and the change log record respectively. IBMSNAP_OPERATION is the operation type and is either ‘I’, ‘U’ or ‘D’.

Important: Capture inserts only the changes of the committed transactions to the CD table.

The data in the CD table may be pruned periodically by the Capture program depending on the value of the auto-prune parameter when Capture is starting.

It the CD table size is a concern, it is possible to decrease the data stored in it. *Capture changes to registered columns only* option available during registration allows rows to be inserted to the CD table only if one of the registered columns is changed. Another method to suppress capturing unnecessary rows is to define triggers on CD table. See Chapter 9.1.2, “Replicating row subsets” on page cdvi.

CD tablespace

From the **CD table** tab, you can change the attributes of the tablespaces of the CD table and the CD name and schema. The fields on the *CD Table* and *CD-Table Index* screens are filled by your source object profile which is customized by you from *Manage Source Object Profiles.*

At the *Manage Source Object Profiles,* you have the options to use the source name or the timestamp, concatenated with a prefix and/or suffix, as the naming convention for the CD table. The convention you can choose for the CD table schema are either the source’s schema or a specific one. This convention is used for all the CD tables definitions of a capture control server. If you are using source name but not the timestamp as the CD name convention and register a replication source more than once with different capture schemas, same name for the CD table will be produced on the second registration. You must either change the CD name for this replication source or use timestamp as CD naming convention.

DB2 UDB for UNIX and Windows

If you are using an existing tablespace, the CD table tablespace can be either SMS or DMS on DB2 UDB for UNIX and Windows. The tablespace created for CD from the Replication Center, is a DMS tablespace. You can alter the container definitions or add new containers. In order to make containers unique, you should consider including the server name and the capture schema in the path.The list of attributes that can be altered are as follows:

- Page size
- Extent size
- Prefetch size
- Buffer pool

DB2 UDB for z/OS

On DB2 UDB for z/OS if you prefer to use an existing tablespace for your CD table, you can choose any tablespace type you like. Replication Center creates CD table in either simple or segmented tablespace. If the source table is in a partitioned tablespace, CD tablespace can also be created as

partitioned. The partition key ranges of the source are used for the CD tablespace.

You must enter a database name for the tablespace. This database must be created since the script generated for registration only creates the tablespace. You can change the Lock size and Buffer pool specific to this tablespaces. The defaults in this screen are from your source object profile.

Storage group, Minimum primary space allocation and minimum secondary space allocation attributes are used for allocation of the dataset.

CD tables are volatile in size. Capture inserts rows into this table whenever a COMMIT is issued for a transaction that involves the associated replication source. After the captured changes for the transaction have been applied to all replication targets, the applied CD rows are eligible for pruning.

The CD table exists only on DB2 source servers.

Table 4-1 is a worksheet to help you estimate the space needed for the UOW table:

Table 4-1 Sizing worksheet for CD tables

You gather this information	Calculations
CD row life = max(max(Apply interval),prune_interval) Apply interval is the frequency you run Apply to process changes. Prune interval is the frequency you prune changes	———— minutes
CD row rate = number of successful SQL inserts, updates, deletes issued for the replication source table during the CD row life	———— number of changes
CD row length	21 bytes + length of registered columns
CD minimum size	CD row length * CD row rate
CD exception factor = multiplier to account for delays or problems (like a network outage) that might prevent changes from being applied. You should always start with at least an exception factor of 2.	———— (should be 2 or more)
CD adjusted size	CD minimum size * CD exception factor

CD-Table index

An index is automatically defined on the two log sequence columns of the CD table. Index schema, name, PCTFREE AND MINPCTFREE can be customized from the CD-Table index screen.

4.2.6 non-DB2 sources - CCD tables

You will notice on the Register Nicknames window that the tab for the staging table definition is titled ‘CCD table.’ Consistent Change Data (CCD) tables have a number of uses in DB2 Replication, both as target tables and as source tables. One of the uses of CCD’s is to fulfill the requirements for staging tables for a non-DB2 data source that doesn’t have Capture.

Registering a Nickname for a table in a non-DB2 source will create a staging table meeting the requirements for a CCD in the non-DB2 data source. The CCD tab of the Register Nicknames window will typically offer fewer options for a CCD table than the Register Table window will offer for CD tables. Among the options available are:

- schema of the CCD table to be created in the non-DB2 server
  - for Informix, it is recommended that the schema of the CCD table be in lower case. It should be enclosed in double-quotes here on the Register Nickname CCD tab, or, before that, in the Source Objects Profile for Platform Informix; for example: “db2repl”
- name of the CCD table in the non-DB2 server
- schema of the nickname to be created for the CCD table in the DB2 database containing the Server definition for the non-DB2 source server
- name of the nickname for the CCD table
- dbspace where the CCD table is to be created. This is an option when the non-DB2 server is Informix.

Replication Center will create the CCD table by using DB2 federated server’s Set Passthru capability. You will see in the SQL generated by Register Nicknames window a Set Passthru statement for the federated Server name of the non-DB2 source server. The Set Passthru statement will be followed by the Create Table statement for the CCD table. After creating the CCD table, the generated SQL will issue a commit at the non-DB2 server, then Set Passthru Reset to return to the DB2 database with Server definition, and then will create a nickname for the CCD.

4.2.7 non-DB2 sources - Capture Triggers and Procedures

When you register a nickname for a non-DB2 tables as a data source, triggers need to be added to the source table itself to insert records into the CCD table

whenever there is an insert, update, or delete to the source table. These triggers simulate Capture reading the log and inserting records in a staging table for updates to the source table. This is why we refer to them as ‘Capture Triggers.’ But in the case of the Capture *Triggers*, the insert into the CCD table is synchronous with and becomes part of the application transaction that updated the source table. The Capture Triggers are ‘After’ triggers; the insert into the CCD table is after the the insert to the source table, though it is still part of the application transaction. If the application transaction does not issue a commit, the new records in the staging table are not committed either. Also, the CCD table needs to have space for the records inserted by the triggers, or the application transaction can’t complete.

Three Capture Triggers get created on each source table. One for inserts on the source table to insert a record into the CCD table, another for updates, and another for deletes. If the source table already has ‘after’ triggers for inserts, udpates, and/or deletes, the logic of the Capture triggers and the existing triggers needed to be added together. Replication Center can not do this addition of the existing application trigger logic and the Capture Trigger logic. But Replication Center should sill generate the logic for the Capture Triggers so they can be copied from the generated SQL and added to any pre-existing triggers.

The Capture Triggers have a mechanism for creating a separate sequence number for each record inserted into each CCD table. This is to simulate unique, advancing sequence numbers of log records. For Informix, this mechanisms involves the IBMSNAP_SEQTABLE that was created with the Capture Control Tables.

In the case of Informix as a replication source, there are also ‘Capture Procedures’ created. This is because Informix allows only a limited number of characters in the logic of a trigger. To get around this restriction, Replication Center first creates a procedure containing the logic to insert a record in the CCD, and then creates a trigger on the source table that calls the procedure. You will see in the SQL generated that the three Capture Triggers (or in the case of Informix, three procedures each followed by a trigger) are created in the same Set Passthru session that creates the CCD table. Following the Create Procedure and Create Trigger statements, you will see a ‘commit’ that commits the creation of the CCD table and the procedures and triggers; this commit will be followed by Set Passthru Reset and then the Create Nickname statement for the CCD table.

The Register Nicknames generated SQL will also create or drop/re-create another trigger in the non-DB2 source server. In the case of Informix, a procedure and a trigger get created. This is the pruning trigger that deletes replicated records from the CCD’s when Apply udpates the IBMSNAP_PRUNCNTL records after it successfully replicates changes to target

tables. For the first table registered at a non-DB2 server, Replication Center creates the trigger (or procedure and trigger) to check for and delete replicated changes from the CCD table for the only registered table. When a second, third, etc. table is registered, the trigger/procedure is dropped and recreated, adding the additional CCD tables to the list that are checked for and pruned.

4.3 Views as replication sources

A view is a registrable source if it is a view over one table or a view over inner join of two or more tables. There are a number of requirements to successfully register a view but the following ones are the rules you must obey on the first place.

- At least one of the base tables of the view should be registered under the same capture schema.
- Columns of the registered tables referenced in the view should be registered.
- Correlation ids for base tables should be given (if the view contains more than one table or a function column).
- Columns should be referred with correlation ids (if the view contains more than one table).
- All base tables need to be qualified.

In order to register a view, you follow the steps explained at Registering the replication sources. When you select the **Register Views,** the *Add Registerable Views* automatically launches and its usage is very similar to *Add Registrable Tables.* At the bottom of *Register Views*, there is a **Create view button**. You can create your views using this button before registering them. There are no options for the views.

Restriction: Views in non-DB2 servers (e.g. Informix) are not supported as replication sources.

When you register a table, a CD table is created. When you register a view, a CD view is created. The underlying tables of the CD view differs depending on whether the underlying tables of the registered view are defined as full refresh or differential replication or whether registered or not.

The fields like the V*iew Schema*, V*iew Name*, etc. on the **CD views** can be altered.

4.3.1 Views over one table

Registered view inherits the differential refresh or full refresh characteristic of the base table. If the table is registered as full refresh only, then the view is also full refresh only. In this case, the CD view is created over the registered table since there is no CD table created for the base table replicated with full refresh only. The view over a table which is registered as a change capture replication source, is defined as a change capture replication source automatically. In this case, the CD view is defined over the CD table.

In order to create a view, follow the following steps:

1. Register the EMPLOYEE table as a replication source accepting all the defaults. (At least, the columns that appear on the view need to be selected).
2. Create the view on Example 4-6. You can create the view either using the **Create view button** or from other tools like Command Center, Command Line Processor. (Qualify the table as in the example).
3. Register the view VEMPLOYEE.

Example 4-6 A view on one table

```
CREATE VIEW VEMPLOYEE (NAME, PHONE)
AS (SELECT LASTNAME || MIDINIT || FIRSTNME, PHONENO
FROM DB2ADMIN.EMPLOYEE E)
```

As you will notice from the DDL created for this registration, a view over the CD table of EMPLOYEE is created. No additional data is captured as a result of view registration. Apply will access the CD table of EMPLOYEE and populate the target based on the definition of the view VEMPLOYEE.

4.3.2 Views over multiple tables

One of the most common reasons for registering an inner join view is to transform data while replicating. There are cases where there is a need to replicate data which resides on two or more tables at the capture site to only one table at the target. Assume the CUSTOMER and ACCOUNTS tables which are used to store customer information and account balances of the customers respectively. The finance department requires the customer names from CUSTOMER and account balances from ACCOUNTS. If views are being exploited, there is no need to replicate each table separately. Defining a view on two tables on the capture site enables only the required data to be transferred to the finance department as a single table.

If views over multiple tables are used as replication sources, refer to Table 4-2 in order to determine whether the replication is a full refresh or a differential refresh and to find out which tables are joined during replication. On the table, T1 and T2 are used to represent base tables and the abbreviations, CC, FR and NR are used for change capture (differential), full refresh and not registered. If both of the base tables are registered for change capture replications, two views referred as V1 and V2 in the table, are created.

Table 4-2 Change capture or full refresh replication of a view

T1	T2	The tables joined by the views	Change capture/full refresh
CC	CC	(V1:CD_T1,T2) (V2:T1,CD_T2)	CC
CC	FR/NR	(CD_T1,T2)	CC
FR	FR/NR	(T1,T2)	FR

In our example, the project number, name, department the project is assigned to, department name and manager of the projects whose duration more than a year, will be replicated to the target. The requirement is to replicate the columns which are spread to two tables to only one table at the target. The VPROJ_DEPT view on Example 4-7 is created for this purpose. The DEPARTMENT table is a stationary table which is not updated at the source on the contrary there are changes to the PROJECT table at any time at the source. PROJECT table is registered for differential refresh to propagate the changes. DEPARTMENT table is not a registered table.

Example 4-7 View on two tables, one differential refresh other not registered

```
CREATE VIEW VPROJ_DEPT AS
(SELECT P.PROJNO,P.PROJNAME,P.DEPTNO,D.DEPTNAME,D.MGRNO
FROM DB2ADMIN.PROJECT P, DB2ADMIN.DEPARTMENT D
WHERE P.DEPTNO = D.DEPTNO
AND DAYS(PRENDATE) - DAYS(PRSTDATE) < 365)
```

In order to try this example, follow the following steps:

- Register the PROJECT table accepting all the defaults on the *Register Tables.* (It must be registered for change capture and the columns that appear on the view should be registered).
- Create the view VPROJ_DEPT as in Example 4-7.
- Register the view accepting all the defaults on the *Register Views*.

After a full-refresh, this registration will enable any changes to the PROJECT table to be propagated to the target joined with the values in the DEPARTMENT table. This is a change capture replication. The CD view is defined over DEPARTMENT and CD table of PROJECT.

On the following example, the details of the employee activity on the projects will be replicated to the target together with the project name and the id of the employee who is responsible from the project. The view VEMP_ACT1 on Example 4-8 is defined to be used for this replication. Both of the tables, EMP_ACT and PROJECT are updated at the source. Since the updates on both tables need to be propagated, they are both registered for differential refresh.

Example 4-8 View on two tables, both of them differential refresh

```
CREATE VIEW VEMP_ACT1 AS
(SELECT A.EMPNO,A.PROJNO,A.EMSTDATE,A.EMENDATE,P.PROJNAME,P.RESPEMP
FROM DB2ADMIN.PROJECT P, DB2ADMIN.EMP_ACT A
WHERE P.PROJNO = A.PROJNO)
```

You can implement this example by following the steps below:

- Register the PROJECT table accepting all the defaults on the *Register Tables.* (It must be registered for change capture and the columns that appear on the view need to be registered).
- Register the EMP_ACT table accepting all the defaults on the *Register Tables.* (It must be registered for change capture and the columns that appear on the view need to be registered).
- Create the view VEMP_ACT1 as in Example 4-8.
- Register the view accepting all the defaults on the *Register Views*.

This registration will result in creation of two views. One joining PROJECT table with CD table of EMP_ACT and the other one joining EMP_ACT table with the CD table of PROJECT. Any change occurring on any of the base tables will be propagated to the target with a join on the other base table.

4.3.3 Restrictions on views

There are some restrictions for the registerable views. If you fail to obey these restrictions, you receive the error message; ASN1704E. This error message has a number of reason codes each for a certain error that can be made during registering a view. You can query the replication error message from Command Line Processor (CLP) like other DB2 error messages. We also provided the error message below for your reference.

ASN1704E The view “<viewowner.viewname>” cannot be registered. Reason code “<reason_code>”.

Explanation: The view cannot be supported by the Replication Capture mechanism, as defined. No script is generated. The following values are valid for the reason code:

- 0 None of the dependent tables for the view are registered.

- 1 The registered source columns on which the view is dependent are not registered.
- 2 The view is on an internal ccd.
- 3 The view is already registered.
- 4 The view has an 'OUTER JOIN' syntax.
- 5 The view includes more than one table or view column with a function, and no correlation is provided in the view definition for each table.
- 6 The view contains a reference to an aggregate function.
- 7 The view contains a subselect/subquery.
- 8 The view contains a reference to another view.
- 9 The view has an UNION.
- 10 No correlation is provided for the column.
- 11 The base table does not have the schema name.
- 12 The base table does not exist.
- 13 The view contains Table Expression as Table.
- 14 The dependent table does not exist.
- 15 A view on view cannot be registered.
- 16 The given source object is not a view.
- 17 This source view is a duplicate for this session.
- 18 The view definition cannot be supported.
- 19 The view has an asterisk (*) instead of a specific column name in the view definition.
- 20 The view contains the join of a CCD and a non-CCD table.

5

Subscription Set

In this chapter we will describe:

- A subscription set and subscription set members.
- Planning on the grouping of subscription sets and members
- Creating a subscription set and members using the Replication Center as the administration task.
- The subscription set and members attributes
- iSeries subscription set and members commands

5.1 Subscription Set and Subscription Set Members

After registering your source tables as described in the previous chapter, you establish the relationship between the target table, by creating subscription sets and subscription set members for specific target type tables or views, which contains the changed data from the registered source tables that is scheduled for replication by the subscription set. Before creating a subscription set you must create the Apply control tables. See,Chapter 3.1, “Introduction to replication control tables” on page cxlii .

5.1.1 Subscription attributes

The Subscription information that you define are stored in the following apply control tables. Details on all of these tables and columns is found in, chapter 23, “Table Structures”, *IBM DB2 Universal Database Replication Guide and Reference,* SC27-1121-00.

Subscription Set

When you create a Subscription Set, the following are some of the attributes defined in the IBMSNAP_SUBS_SET table:

- A name for the subscription set.
- The source and target server name.
- The Apply qualifier.
- When to start replication, how often to replicate, and whether to use interval timing, event timing, or both.
- Data blocking, if you expect large volumes of changes

If the Subscription Set is for replication from or to a non-DB2 server, such as Informix, the source or target server name will be the name of the DB2 ESE or DB2 Connect EE database containing the Server definition for the non-DB2 source/target server. The IBMSNAP_SUBS_SET table has two additional attributes, which will be filled in as appropriate for a Suscription Set replicating from or to a non-DB2 server:

- federated Server name of a non-DB2 source server
- federated Server name of a non-DB2 target server

Subscription Member

A subscription set usually have one or more subscription member for each target table or view, that is associated with a source table. When you create a Subscription Member the following are some of the attributes are defined in the IBMSNAP_SUBS_MEMBR:

- A name for the subscription set.

- The Apply qualifier.
- The source table or view and a target table or view
- Source and target schema
- Target table type
- The structure of the target table or view.
- The rows that you want replicated (SQL predicates)

When replicating from a non-DB2 server, the source table name and schema will be for the nickname for the source table.

When replicating to a non-DB2 server, the target table name and schema will be for the nickname for the target table.

Subscription Columns

A subscription columns table contains the target table or view columns definitions. When a Subscription column is created the following are some of the attributes defined in the IBMSNAP_SUBS_COLS:

- A name for the subscription set.
- The Apply qualifier.
- The target column name
- The target column type
- SQL column expression for data transformation.

If replication is to a non-DB2 target server, the target column name will be for the column of the nickname for the target table.

If replication is from a non-DB2 source server, the SQL column expression will reference the columns of the nickname for the source table.

Subscription Statement

A subscription statement table contains an SQL statement or procedure that is executed before or after the Apply programs runs. This is an optional table for your subscriptions. When a Subscription statement is created the following are some of the attributes defined in the IBMSANP_SUBS_STMTS:

- A name for the subscription set.
- The Apply qualifier.
- Execute before or after indicator
- Execute SQL statements or procedures before or after processing a subscription set

Subscription Event

A subscription statement table is an optional subscription table for defining a event name and time to start replicating. When a Subscription event is created the following are some of the attributes defined in the IBMSANP_SUBS_EVENT

- Event name
- Event starting timestamp

5.2 Subscription set and member planning

The following are rules and constraints to be aware of when planning to create subscription sets:

- The Apply program processes all members in a subscription set as a single group. Because of this, if any member of the subscription set requires a full refresh copy, all members within the entire set are full refreshed.
- All source tables in the members of a set must have the same Capture schema.
- If your are creating a subscription set on a iSeries systems, all source tables defined in the members of a subscription set must be journaled to the same journal.

Subscription Sets from non-DB2 servers

When replicating from non-DB2 servers, all source tables in a set must be from the same non-DB2 server.

Subscription Sets to non-DB2 target servers

Before creating a subscription set for replicating to a non-DB2 server (e.g. Informix), first a DB2 UDB ESE or DB2 Connect Version 8 server must first be configured for federated access to the non-DB2 server. Within a database in DB2 ESE/Connect, there needs to be a Server definition to the non-DB2 server. See Appendix C, “Configuring federated access to Informix” on page dxxxviifor requirements and instructions for configuring federated access to Informix. The database containing the Server definition to Informix could also be the Apply Control Server.

Grouping members to subscription sets

When you add members to a subscription set, you must determine how do you want to group them:

- Group all of your source target table into one subscription set
- Create separate subscription sets for each source and target table
- Create a small number of subscription sets, each with a number of source target tables.

The Apply program replicates the members within a subscription set as one (logical) transaction. Therefore, in your decision to group multiple members into one subscription sets, you could consider some of the following situations:

- The source tables are logically related to one another.
- The target tables have referential integrity constraints.
- The target tables could have a larger number of transactions process verses a targets table with the occasional transaction process
- Multiple members grouped within one subscription set, will make sure that replication for all members will begin at the same time.
- If the subscription set uses before or after SQL statements or stored procedures, those SQL statements or procedures will process all of the members within the subscription set.
- If there are no logical or referential integrity relationship between the source and target tables in a subscription set, you can group them into one subscription set or into several subscription sets.
- Limiting the number of subscription sets could make administration of the replication environment simpler.
- Increasing the number of subscription sets, could minimize the affect of replication failures.
- To easily resolve any errors that cause the Apply program to fail, add only a small number of members to a subscription set. You would likely locate the source of the problem more quickly if the set contains a small number of members.
- When one member of a subscription set fails, all of the data that has been applied to other members of the set is rolled back; so that no member can complete the cycle successfully unless all members do. The Apply program rolls back a failed subscription set to the last successful commit point, which could be within the current Apply cycle if you specified the commit_count keyword when you started the Apply program, see Figure 5-2 on page ccxiii.

5.2.1 Member definitions to non-DB2 targets servers

Replication Center can either create target tables in non-DB2 target servers or work with target tables that already exist. If the target table already exists, a nickname must also exist for it in the DB2 ESE/Connect database that contains the Server definition to the non-DB2 target server before the Create Subscription Set or Add Member definion is done in Replication Center.

If Replication Center creates the target table, it will use the DB2 federated server Set Passthru capability to create the target table in the non-DB2 server and then will create the nickname for the target table in the DB2 database that has the Server definition.

An advantage to letting Replication Center create the non-DB2 target table is that Replication Center is sure to make the data types, lengths, nullability, and primary key/unique index attributes of the target columns compatible with the source table columns. This compatibility includes altering the columns of the target nickname if needed to override the default type mappings in the federated server wrapper for the non-DB2 server. You can see the Create Table, Create Nickname, Alter Nickname DDL recommended by Replication Center without actually creating a table or nickname. When Creating the Subscription Set, or Adding the Member, specify a target nickname and remote target table name that don't exist; let Replication Center generate the SQL and save the SQL to file for comparison with the DDL that created the target table and created the nickname for it. You could also use 'Describe Table' at the replication source and on the target nickname to compare the column characteristics of an existing source table and an existing target nickname.

5.2.2 Subscription set and apply qualifiers planning

When you create a subscription set, you associate an Apply qualifier for that subscription set. The Apply qualifier initiate an instance of the Apply program with one or more subscription sets. Each subscription set is processed by only one Apply program, but each Apply program can process one or more subscription sets during each Apply cycle.

Subscription set and apply qualifier grouping

You can run as many instances of the Apply program (each with its own Apply qualifier) as required, and each Apply program can process as many subscription sets as you need. There are two basic options:

- Associate each Apply qualifier with one subscription set (each Apply program processes exactly one subscription set)
  - If speed is important, you can spread your sets among several Apply qualifiers, which allows you to run several instances of the Apply program at the same time. If you decide to have an Apply program process one subscription set, you can use the Apply program's OPT4ONE startup parameter, which loads the control-table information for the subscription set into memory. Using this option, the Apply program does not read the control tables for the subscription-set information for every Apply cycle.
  - Therefore, the Apply program performs better. However, the more Apply programs that you run, the more system resources they will use, and the slower their overall performance might be.
- Associate each Apply qualifier with multiple subscription sets (each Apply program processes many subscription sets)
  - One Apply program processing all of your subscription sets, could reduce the number of database connections needed to process the subscription

sets, also reduce the administration overhead for maintaining your replication environment.

- By using more than one Apply qualifier, you can run more than one instance of the Apply program from a single user ID.
- The Apply program tries to keep all sets for an Apply qualifier as current as possible. When an Apply cycle starts, the Apply program figure out which of the subscription sets contains the least current data and starts processing that set first. Therefore, if performance is not a concern, you might want to replicate a large number of subscription sets with one Apply qualifier.

  This could be the best option if you wait until after business hours before replicating.

- The Apply program processes the subscription sets sequentially. Therefore, your overall replication latency could increase.

If you have specific requirements for certain subscription sets, you can combine these two options. For example, you could have one Apply program process most of your subscription sets and thus take advantage of using one Apply program to process related subscription sets together, and you can have another Apply program process a single subscription set and thus ensure minimum replication latency for that subscription set. And by using two instances of the Apply program, you increase the overall processing performance for your subscription sets.

5.3 Define Subscriptions using the Replication Center

You can create, change and remove subscription sets and subscription members using the Replication Center as the administrator task, which give you the option to create the subscription set and add the subscription member later, or add the members while creating the subscription set. The following section will describe how to use the Replication Center to accomplish this task.

Before following these steps to use the replication center, you need to determine if you are going to create a empty subscription set first, then add the member later or add the member while creating the subscription or adding a member to an existing subscription set. Picking either of these options will determine at which entry point in Replication Center to create your subscriptions.

5.3.1 Create Subscription Sets with members

To create a subscription set while adding members, expand the **Replication Definitions** folder**----> Capture Control Servers** folder **--->**The database server that contains the registered tables or views. Click on the **Registered Tables** or

Registered Views folder. Right click on a registered source table or View on the right side of the display. Click on **Create Subscription Set** from the pop-up menu to display the Create Subscription notebook, see Figure 5-2 on page ccxiii.

5.3.2 Create Subscription Set without members

To create a subscription set without adding members, expand the **Replication Definitions** folder**----> Apply Control Servers** folder **--->**The Apply control server. Right click on the **Subscription Sets** folder. Click on **Create...** from the pop-up menu to display the Create Subscription notebook, see Figure 5-2 on page ccxiii.

5.3.3 Subscription Sets from non-DB2 servers

When creating a new subscription set from a non-DB2 server, in Replication Center, select the data base server object that includes the federated Server definition name for the non-DB2 serve *and* the name of the DB2 ESE/Connect database that contains this server definition. For example, if the Capture Control Server name in Replication Center is 'IDS_VIPER / FED_DB' , the federated Server name is 'IDS_VIPER' and the DB2 ESE/Connect database that contains this Server definition is 'FED_DB.'

5.3.4 Subscription Sets to non-DB2 servers

In Replication Center's Create Subscription Set window, when selecting the Target Server, you will click the '...' in the Target Server Alias field and be presented with the Select a Target Server window. This window will be filled in with the names of the DB2 servers known to Replication Center. Highlight the Database Alias for the server that you know to have the federated Server definition for the non-DB2 target, place your mouse-pointer over the 'Non-DB2 server' field, right-mouse click, and select 'Retrieve non-DB2 servers.' Additional lines will be added to the Selected a Target Servers list, one for each of the non-DB2 Servers defined in the Database Alias you just selected. Highlight the line that contains the federated Server definition for your intended non-DB2 target server, and click OK at the bottom of the window. In the Create Subscription Set window, the Target Server Alias name will show the federated Server definition name followed by the DB2 database name that contains this Server definition.

5.3.5 Adding subscription members to existing subscription sets

To add members to existing subscription sets, expand the **Replication Definitions** folder**--->**expand **Apply Control Servers--->**expand the Apply

control server**--->**click **Subscription Sets** folder (You can right click to create a filter and subset your list of subscription sets).

Next, do either of the following:

- Double click the subscription set you want work with, will display the Subscription Set Properties notebook similar to Figure 5-2 on page ccxiii. Then click the **Source to Target Mapping** page**--->Add** push button to display the Add registered source search criteria window**--->Retrieve all** or **Retrieve** using your search criteria to list the registered source tables**--->**Select one or more registered sources**---> OK** to display source to target mapping note book page similar to,Figure 5-3 on page ccxviiand follow the same steps to add subscription members.
- Select one or more subscription sets**--->**right click will display a pop up menu **--->Add Member** to display the **Add Member to Subscription Sets** note book page, see Figure 5-1 on page ccxii **---->**check box to select one or more subscription sets**--->Member Information** page ---> **Add** button to add subscription member**--->Retrieve all** or **Retrieve** using your search criteria to list the registered source tables**--->**Select registered sources**--->OK--->**select target type**--->Details** button to display **Member Properties** note book page see Figure 5-5 on page ccxxi and follow steps to define subscription member properties. You need to click **Details** button for each Subscription Set. Click .the **OK** button if active will generate the SQL script to update the apply control tables at the apply control server specified in Figure 5-2 on page ccxiii. See, Section 2.13.1, “Running Saved SQL Files Later” on page cxxiii. Also see Section 5.7, “SQL script description” on page cclv

5.3.6 Subscription Sets and Member Notebook

The following note book page is displayed when you select the option to add member to an existing subscription set.

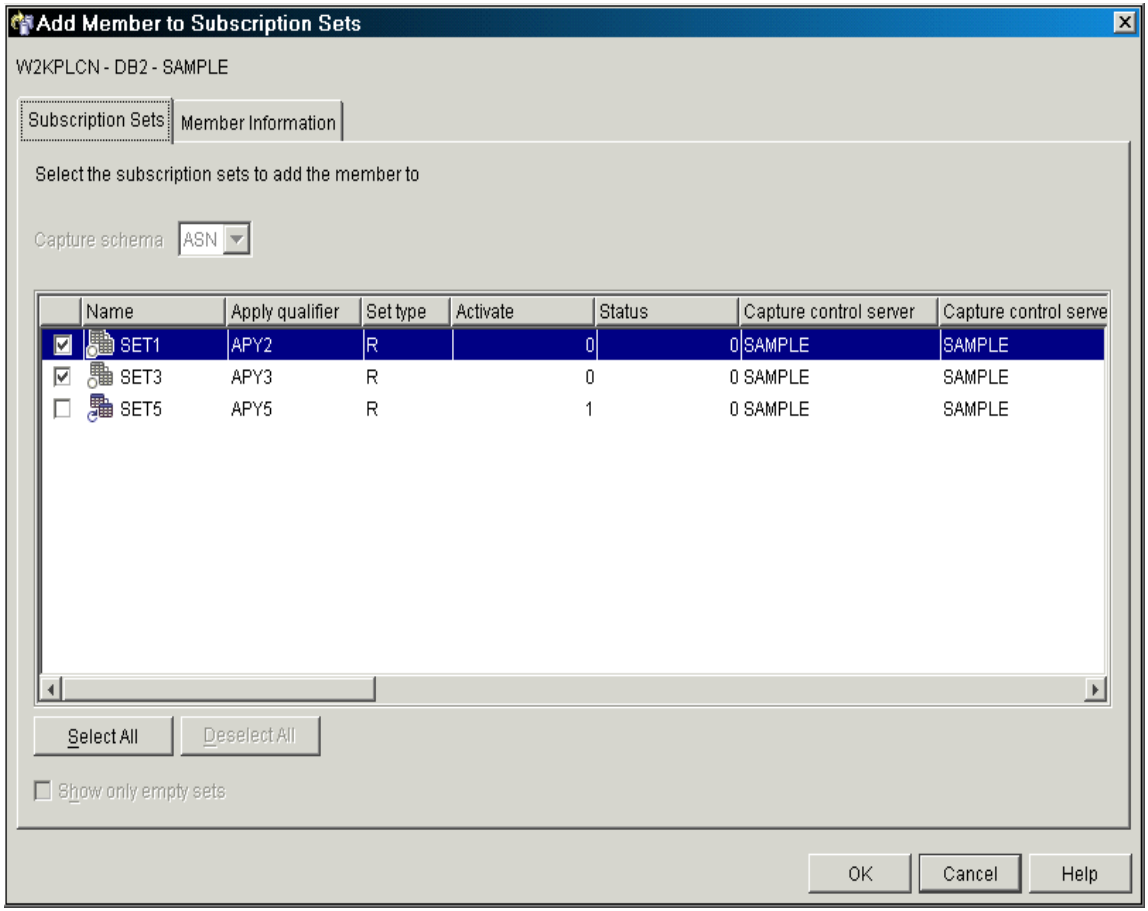


Figure 5-1 Add members to subscription set

See section Section 5.3.5, “Adding subscription members to existing subscription sets” on page ccx to use this notebook when adding subscription members.

The following is a description of the 4 pages in the Create Subscription Set notebook.

Note: You can also open this notebook from the launchpad clicking on option **4. Create a Subscription Set**, see chapter? for details.

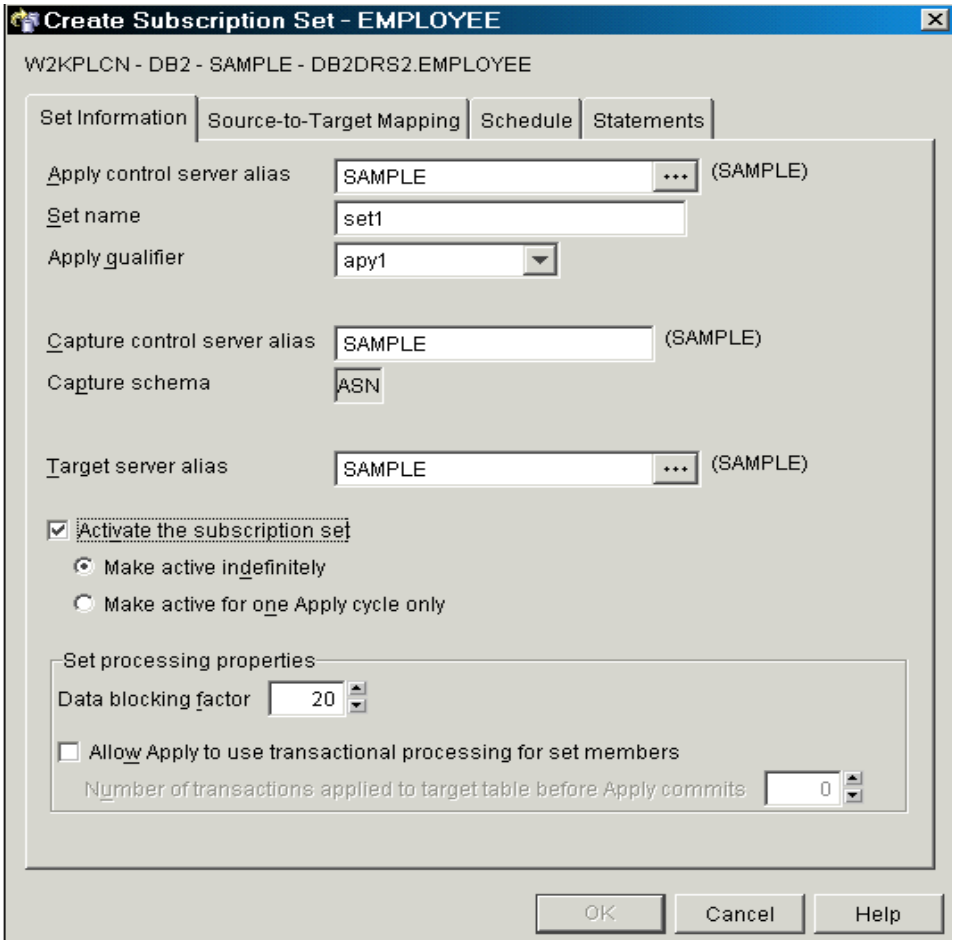


Figure 5-2 Create Subscription Set notebook

Set Information

- **Apply control server alias**
  - If you opened the Create Subscription notebook using the selection to add subscriptions member as indicated in section 5.3.1, “Create Subscription Sets with members” on page ccix you must select the Apply control server by clicking the**...** button. This will open the Select Server window. Select the Apply control server containing the Apply control tables for the subscription set your creating
  - If you opened the Create Subscription notebook choosing the select to add subscription members later as indicated in section 5.3.2, “Create Subscription Set without members” on page ccx, or from the launch pad.

Verify the Apply control server is the one you want. You can not modify this field. To change the server name, close the notebook and reopen it again from the correct server

- **Set Name**

  Type in the name for the subscription set, the name can be up to 18 characters long. This name uniquely identifies a group of source and targets tables within an apply qualifier, that is process by a separate apply program. You can define more than one subscription set within an apply qualifier.

- **Apply qualifier**
  - Type in a name for a new apply qualifier or click the down arrow to select from a list of existing apply qualifiers. The apply qualifier is case sensitive. Therefore, if you want lower or mixed case, you must delimit it with quotation marks. For example, “Apyqual1”. Lower or mixed case characters that are not delimited are changed to uppercase.
  - By using more than one Apply qualifier, you can run more than one instance of the Apply program from a single user ID. The Apply qualifier is used to identify records at the control server that define the work load of an instance of the Apply program; whereas the user ID is for authorization purposes only.
  - For example, assume that you want to replicate data from two source databases to the target tables on your computer. The data in source table A is replicated using full-refresh copying to target table A, and the data in source table B is replicated using differential-refresh copying to target table B. You define two subscription sets (one for table A and one for table B), and you use separate Apply qualifiers to allow two instances of the Apply program to copy the data at different times. You can also define both subscription sets using one Apply qualifier.

- **Capture control server alias**
  - This field contain the name of the control server were the registered tables and CD tables are located for this subscription set. If the registered tables are in a non-DB2 server, then the field name here will contain the federated Server definition name followed by the name of the DB2 database that contains this Server definition.
  - If you opened the Create Subscription notebook from the **Apply control server** folder as indicated in section 5.3.2, “Create Subscription Set without members” on page ccx you must select the Capture control server by clicking**...** button. This will open the Select Server window for you to select the Capture control server. When creating a Subscription Set to replicate from a non-DB2 server, select the select the Database Alias that has the ‘non-DB2 Server’ name for the non-DB2 source server.
  - If you opened the Create Subscription notebook from the Launchpad or the selection of registered tables as indicated in section 5.3.1, “Create

Subscription Sets with members" on page ccix. Verify the Capture control server is the one you want. You can not modify this field. To change the server name, close the notebook and open it again from the correct server

- **Capture schema**
  - This field contains the schema of the capture control tables containing the registered tables and CD tables for this subscription set. For non-DB2 sources, this will be the schema of the nicknames for the Capture Control Tables.
  - If you opened the Create Subscription notebook from the **Apply control server** folder as indicated in section 5.3.2, "Create Subscription Set without members" on page ccx, you must select the Capture schema from the list box when you click the arrow button.
  - If you opened the Create Subscription notebook from the Lauchpad or the selection of registered tables as indicated in section 5.3.1, "Create Subscription Sets with members" on page ccix. Verify the Capture schema is the one you want. You can not modify this field. To change the schema name, close the notebook and reopen it again from the correct schema.
- **Target server alias**

This field contain the name of the target server were the target table resides. Click the **...** button to open the Select Server window for you to select the target server. For non-DB2 targets, see the discussion on the previous page in 'Subscription Sets to non-DB2 Servers.'

- **Activate the subscription set**

  Click in this box to activate subscription, then select either of the following radio button:

  - **Make active indefinitely**
  - **Make active for one Apply cycle only**

  If you decide to deactivate the subscription set, you can activate it later, by right click the subscription set and select **activate** from the pop up menu, See fig.....

xref fig. That show list of sub to right click. will show after creation of sub set

- **Data blocking factor**

  Click the up or down arrow to change the number of minutes worth of captured data for apply to process during a single cycle. See Section 5.5, "Data Blocking" on page ccli

  Note: The default is **20**, changing it to **0** will disable the blocking factor function

- **Allow Apply to use transactional processing for set members**

  - Click on the box to change the way apply program replicate changes from the spill file.
  - Table mode is the default processing mode when this box is not checked, which means all changes from the spill files are applied to the corresponding target tables one at a time in the set. Then it issues a DB2 commit to commit all the changes to each of the target tables within the subscription set.
  - If the boxed is checked, apply processing will change to transactional mode, which means all changes from the spill file are open and process at the same time. The apply order to the target tables is the same as the source transaction order. Apply will issue DB2 commits at intervals that is specified in the: **Number of transactions applied to target table before Apply commits number** field. Click the up and down arrow to change the numeric value.

- When you complete the Set Information in Figure 5-2 on page ccxiii. Click the **Source-to Target** page to display the source to target mapping notebook, see Figure 5-3 on page ccxvii. Follow the steps as instructed to define the subscription member and columns.
- You can optionally click the **Schedule** page to display the notebook to schedule your replication, see Figure 5-16 on page ccxxxvi
- You can optionally click the **Statements** page to display the notebook to define the SQL statements or call procedure, which is executed before or after the apply program runs, see Figure 5-17 on page ccxxxviii
- The **OK** button if active will generate the SQL script to update the apply control tables at the apply control server specified in Figure 5-2 on page ccxiii. See Section 2.13.1, “Running Saved SQL Files Later” on page cxxiii. Also see Section 5.7, “SQL script description” on page cclv

Note: If you select the option to add subscription member later as shown in 5.3.2, “Create Subscription Set without members” on page ccx. The **OK** button is active, which indicates you can create an empty subscription.

Source-to Target Mapping

The following describes the source to target information that you create or add to the subscription set member table from the Source to Mapping notebook view, see Figure 5-3.

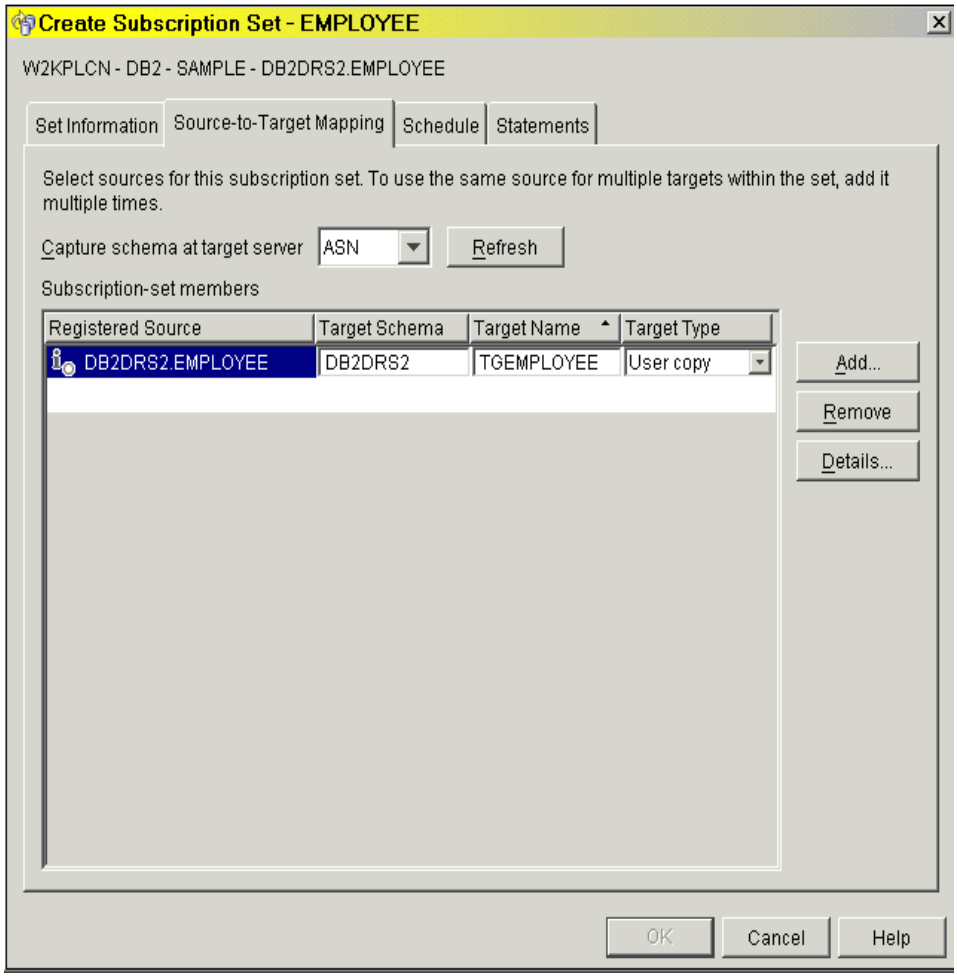


Figure 5-3 Source to Target mapping- Source and target tables

- **Registered Source**

These are the registered source tables described in Chapter 4, “Replication Sources” on page clxix. Click this field to highlight the Add..., Remove and Details... buttons

For non-DB2 sources, their will be two fields:

- **Registered nickname**

  Schema and nickname of the registered nickname

- **Remote source**

  Remote schema and remote table name at the non-DB2 server

- **Target Schema**

  The Target schema for the target table is defined here. The default name is the source table schema, you can type over this name if you wish to change it

- **Target Name**

  The name of the Target table is defined here. The target table could either be a new or existing table. The default target name is defined in the target object profile were you can establish a naming convention for target tables. You can change the name by typing over it.

- **non-DB2 targets tables**

  For non-DB2 target tables, the above two fields are replaced by **Target Nickname Schema** and **Target Nickname**, and there are two additional fields which can be viewed by scrolling to the right:

  - **Remote Target Schema**

    Name of the schema for the target table in the non-DB2 target server. The default values come from the Target Table Profile for the target server plaftorm if that was filled in before this window was opened. If the target server is Informix, it is recommended that the remote target schema be in lower case. If so, the value should be enclosed in double-quotes in the Remote Target Schema field.

  - **Remote Target Name**

    Name of the target table in the non-DB2 target server. The default values come from the Target Table Profile for the target server plaftorm if that was filled in before this window was opened.

- **Target Type**

  Click the arrow to select a target type from the drop down menu. See Chapter 5.4, “Target Types Descriptions” on page ccxli for more details on target types.

- The Member Properties column selection notebook page is displayed when you Click the **Details...** button to continue defining the subscription members, see Figure 5-5 on page ccxxi.

- The **OK** button if active will generate the SQL script to update the apply control tables at the apply control server specified in Figure 5-2 on page ccxiii. See Section 2.13.1, “Running Saved SQL Files Later” on page cxxiii. Also see Section 5.7, “SQL script description” on page cclv.

Note: If the target type is a **Consistent Change Data**, the CCD properties notebook page is displayed when you click the **Details...** button, see Figure 5-4 on page ccxx:

Note: If you select the Replica target type, there are additional pages shown for replica definition, see Figure 5-15 on page ccxxxv. Read the following pages from Figure 5-5 on page ccxxi for a description of the other notebook pages.

This note book page is displayed if your target type selection from Figure 5-3 is CCD.

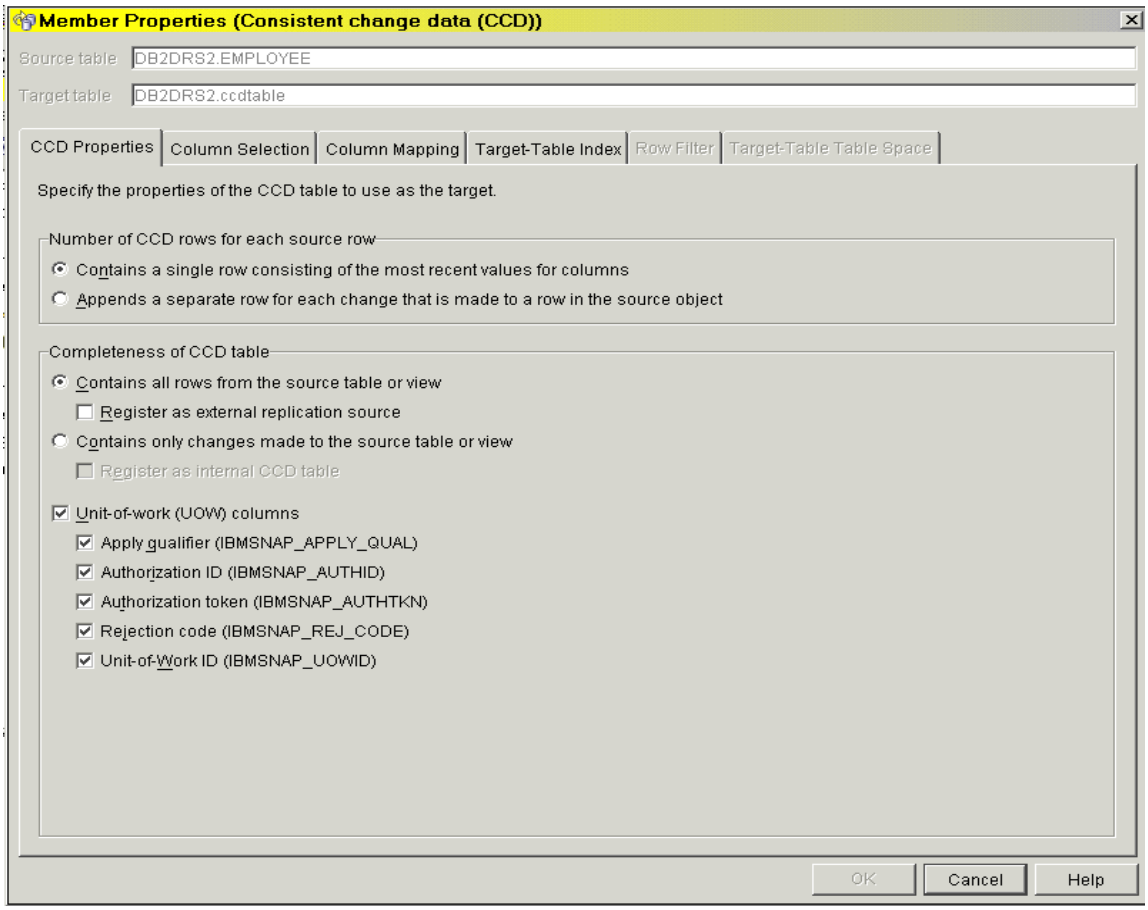


Figure 5-4 Member Properties -- CCD Properties

Click the appropriate radio button or box to select which type of CCD to be created. See Chapter 5.4.4, “CCD (consistent change data)” on page ccxlii for more details. Or click the **Help** button and then the “**define the properties of the CCD tables** link”, for a description of the radio buttons and box selection associations.

The column selections from the registered source tables or view to the target tables is performed in following notebook page show in Figure 5-5.

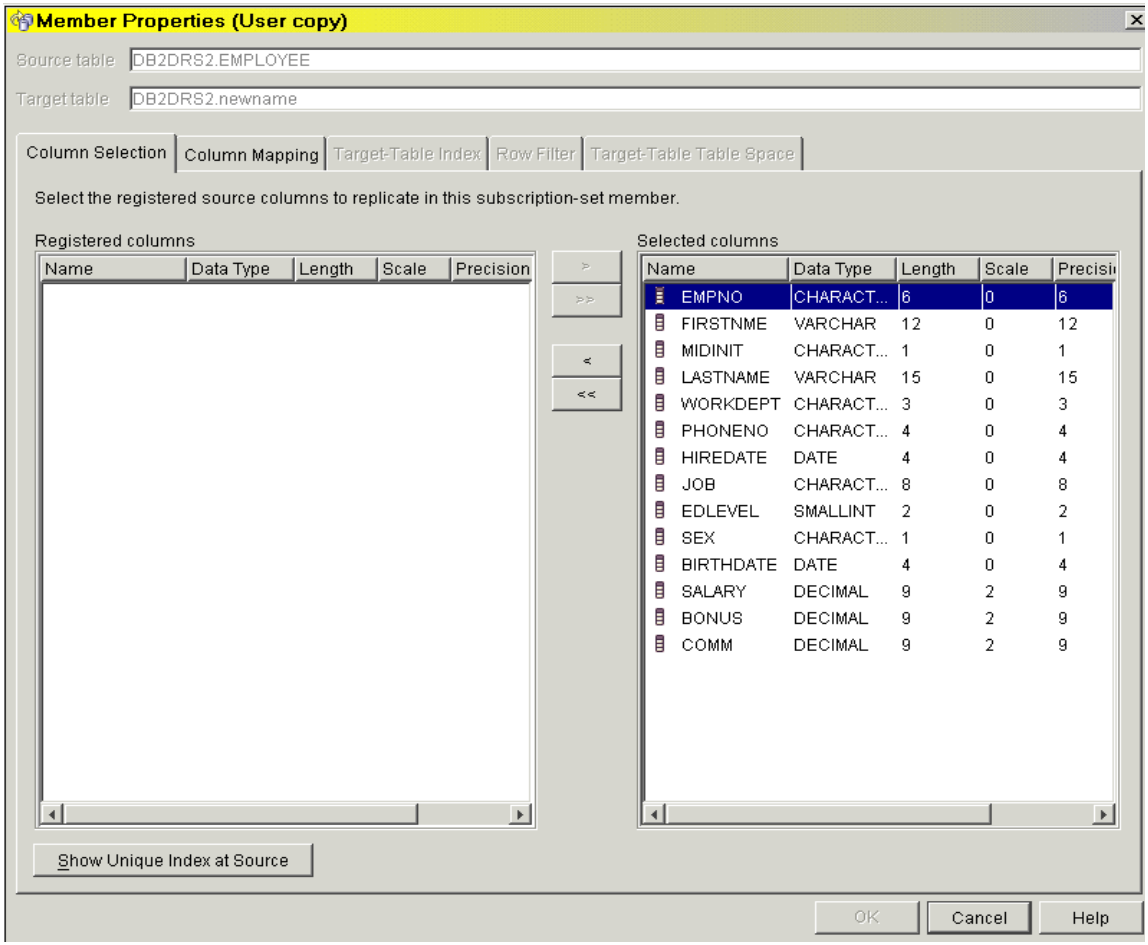


Figure 5-5 Member properties - Column Selection

All the columns are selected to the target table, which is shown on the right side of the notebook page, this is the default. If you don't want to replicate a particular column, click the column to highlight it, then click the < bottom to move the column to the registered column side. The << will move all the columns.

When the **column mapping** notebook page is selected the following notebook is displayed, see Figure 5-6

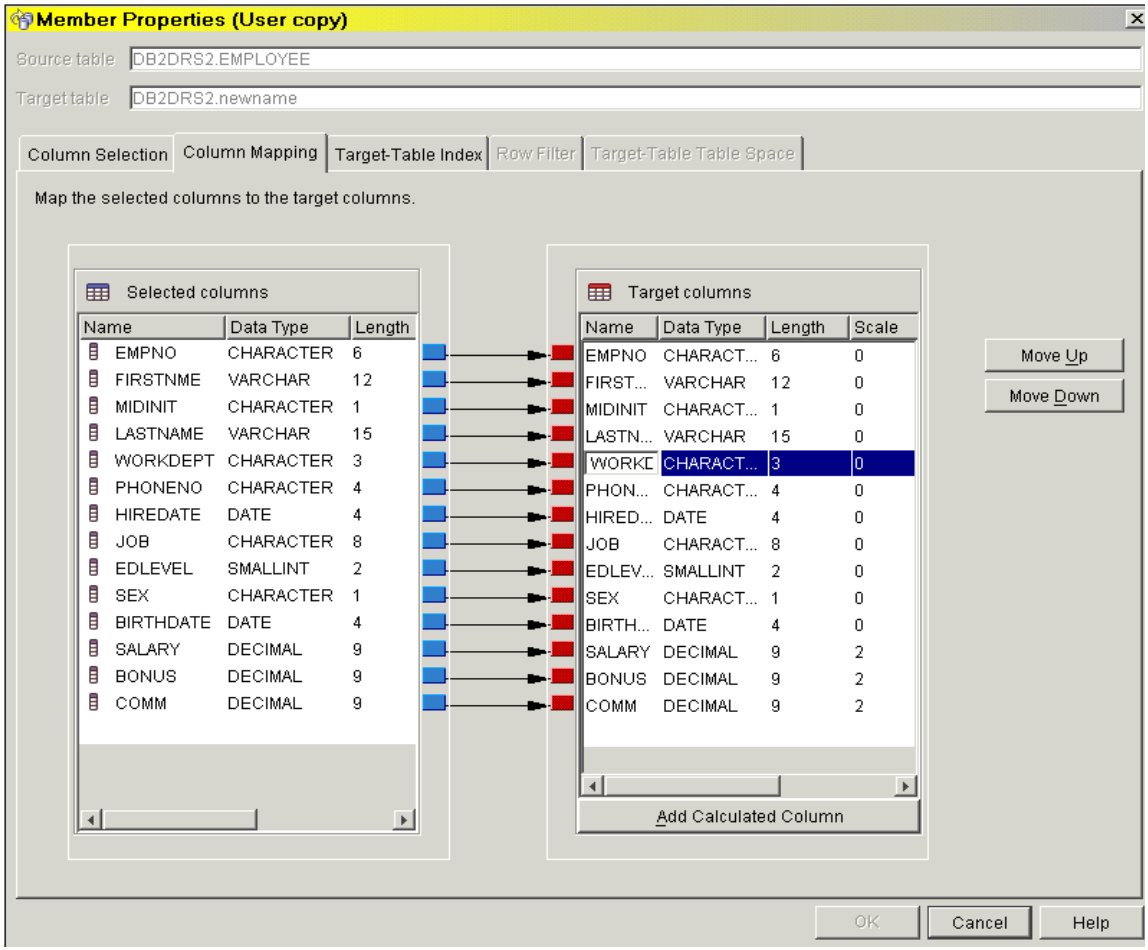


Figure 5-6 Member Properties - Column mapping

Column mapping from the selected targets column is performed on the notebook page shown in Figure 5-6. The mapping functions that is available depends on the existence of the target table. For a detail description on these mapping functions click the **Help** button**.**

- If the target table doesn't exist the following functions are available:
  - **`Move Up`** and **Move Down** button. Click on the target column to activate these buttons, which will enable you to change the position of the columns in the target table
  - Change target table column properties. Click the field to enter the appropriate and compatible value
  - **Add Calculated Column**. Click this push button to display the SQL Expression Builder window, see Figure 5-7 on page ccxxiv

- If the target table does exist the following functions are available:
  - The source and target column mapping arrows are not displayed, which means you have to manually do the mapping for each column. Clicking the arrow in the blue box then dragging the mouse to the circle in the red box will create the arrow, which indicates the source column is mapped to a corresponding target column.
  - For non-DB2 target tables, if a nickname already exists, we found the Column Mapping window will show, with arrows, Replication Center's 'guess' of the desired column mapping based on the names and attributes of the source table columns and the attributes of the nickname columns. If one or more of the source columns do not have arrows to any target nickname columns, this suggests strongly that the attributes of the target columns, or of the data types of the nickname columns, are not compatible with the source columns. Even if all the source columns are shown as mapped to nickname columns, you may still get warning messages that the nickname target columns do not have all the attributes of the source columns when the Create Subscription/Add Member window generates the SQL A typical warning message looks like:

    ```
    ASN1827W  The column "DEPTNO" of the target member
    "IFMX_TGT.TGDEPARTMENT" does not preserve a DB2 column attribute of
    the corresponding column "DEPTNO" of the source member
    "DB2DRS4.DEPARTMENT". Reason Code "4"
    ```

  - **Add Calculated Column**. Click this push button to display the SQL Expression Builder window, see Figure 5-7 on page ccxxiv

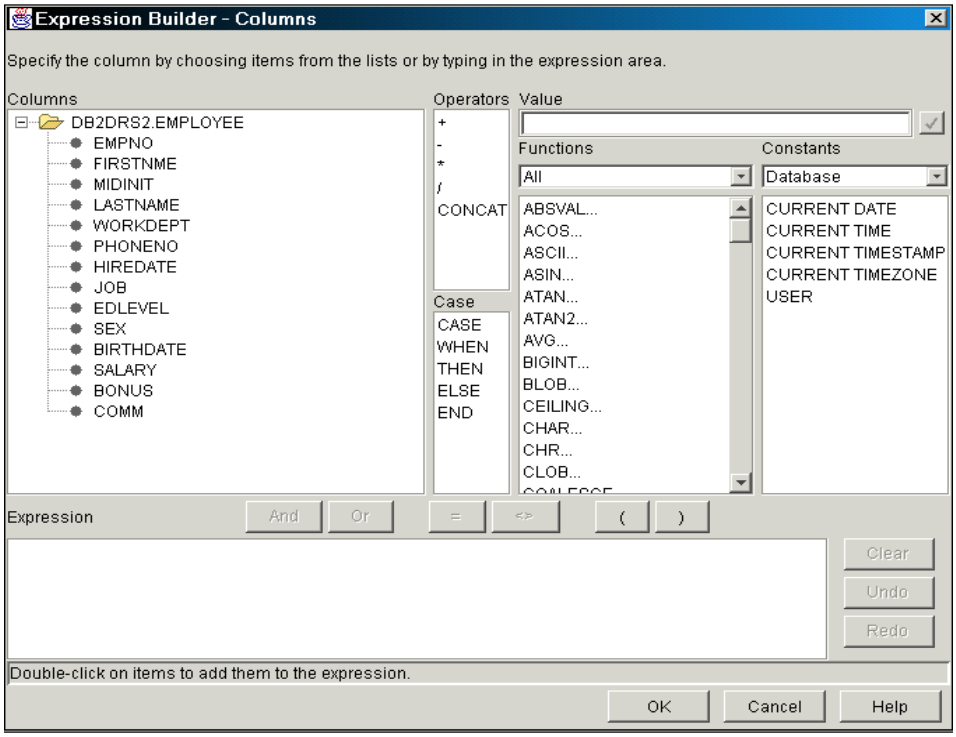


Figure 5-7 SQL Expression Builder Window

This window will assisted you to build an SQL expression to calculate values in a column in the target table based on columns that you have chosen to replicate from the replication source. Click on the **Help** button for detail instructions on how to use this window.

If the target type is either **Base or Change Aggregate** the **GROUP BY clause for aggregation** field is displayed on the bottom of the **Column Mapping** notebook page, see Figure 5-8

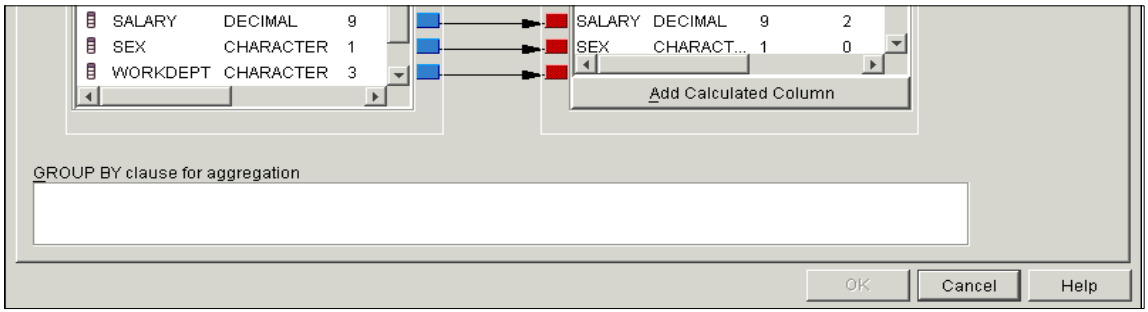


Figure 5-8 Member Properties - Group by in Column Mapping

Enter the column name to do a GROUP BY clause when aggregating data in Base or Changed Aggregate target types, see Chapter 5.4.3, “Aggregate tables” on page ccxlii or Chapter , “Base aggregate” on page ccxlii

When the **Target-Table Index** notebook page is selected the following notebook is displayed, see Figure 5-9

Note: The **Target-Table Index** notebook page is not shown if the target type is **Base or Change Aggregate**

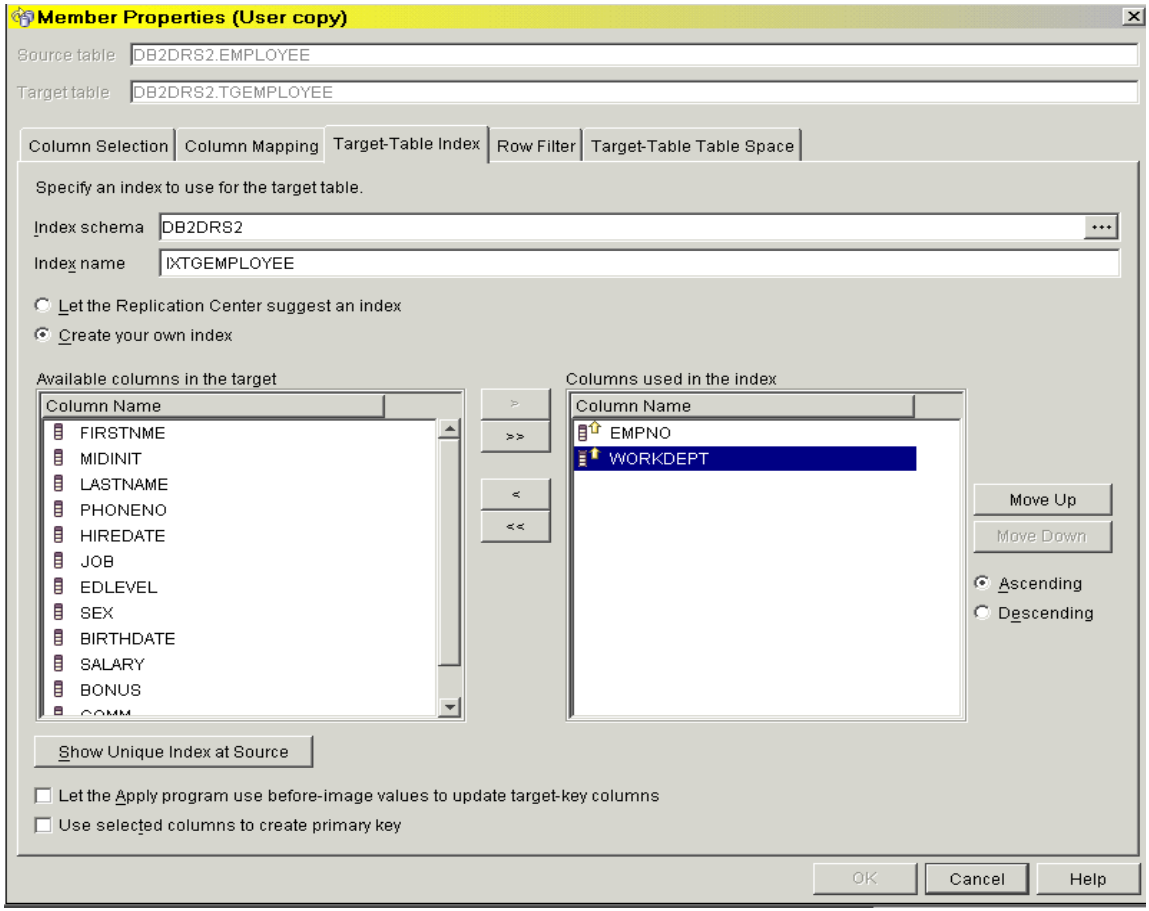


Figure 5-9 Member Properties - Target Table Index

The apply program requires a unique index or primary key to be defined on a condense target table, for change capture replication. The following are the target types that require a unique target index:

- User copy
- Point-in-time
- Replica

- Condense CCD

Creating or specifying an existing target table index is performed on the notebook page shown in Figure 5-9. The default **index schema** and **index name** comes from the target object profile. See Section 2.11, “Managing your DB2 Replication Center Profile” on page cxiii. These field will be disabled if the target table exist. If the target table doesn’t exist, these defaults could be used to create the target index. The buttons that are active depends on the existence of source table indexes and or primary keys.

Note: If the target index or primary key does exist see Figure 5-10 on page ccxxix

Xref Chapter on Registration before image

- **Let the Replication Center suggest an index** radio button. This button will use the index or primary key defined on the registered source table. It is only active if the source table index or primary key exist.
- **Create your own index** radio button. This button will allow you to select the columns from the: **Available columns in the target name** window, to define the target index. This button is still active for existing indexes.
- **Move up** and **Move Down.** These push button will change the position the columns in the index, selected in the: **Columns used in the index** window. These push buttons is active when you select the: **Create your own index** radio button and have more then one column selected to be used as the index.
- **Ascending** and **Descending**. These radio buttons changes the up or down direction of the yellow arrow, next to the column selected to be used as the index. These arrows indicate the sort order of the index.
- **Show Unique index at Source push button**. This button will show a list of columns that are defined as unique source columns the index. It will only show a list for existing unique source table indexes.
- **Let the Apply program use before-image values to update target-key columns** check box. Click this box if the columns defined in the target index is different from the columns defined in the source table. The before image is used from the CD table to update the column used in the target index, see Registration chapter. The before image is used by the apply program to delete the old key row and insert a new row with the new key value.
- **Use select columns to create primary key** check box. Click this box to create a unique primary key instead of a unique index. Will be inactive if a primary key exist on a existing target table.
- If you want to select specific rows from the registered source table, click the **Row Filter** page to add this member property, see Figure 5-13 on page ccxxxii.
- Click the **Target-Table Table Space** note book page to create a table space. SeeFigure 5-14 on page ccxxxiv

Note: The **Target-Table Table Space** page option is not displayed if the target table already exist, or the target table resides on a iSeries server.

- Clicking the **OK** push button when it becomes active, will take you back to the **Source-to-Target Mapping** page of the **Create Subscription Set** note book, see Figure 5-3 on page ccxvii

For a additional information on the target index functions, click the **Help** button --> **define a primary key or unique index on the target table** link on the browser.

The following note book page is displayed if the target index or primary key already exist.

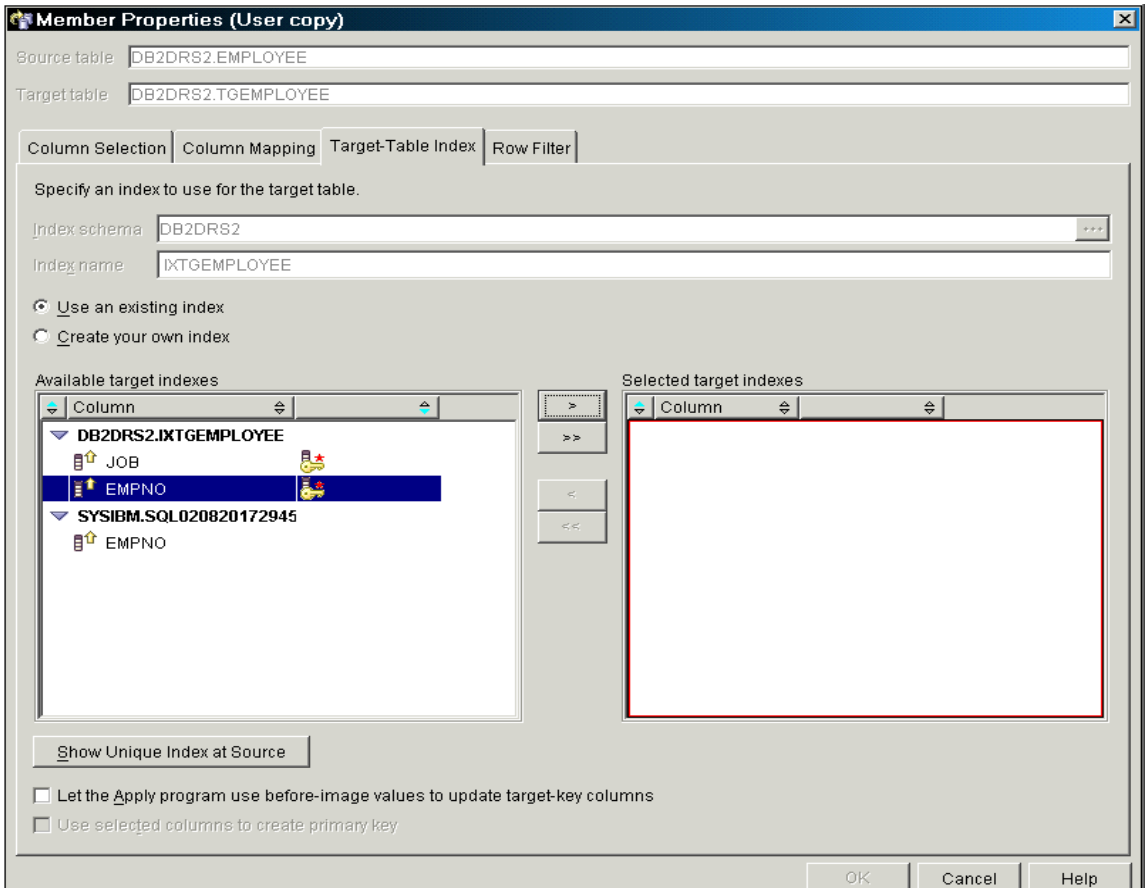


Figure 5-10 Member property - Use existing target index option

- **Use an existing index** radio button. Click this button and select which existing index you want to use, if there is more than one. Click the **>** button to move it the **Selected target indexes** window.
- **Create your own index** radio button. Click this button to create another unique target index. See Figure 5-11 on page ccxxx
- The other functions on this note book page are describe in Figure 5-9 on page ccxxvi

Option to create another target index, for existing target indexes.

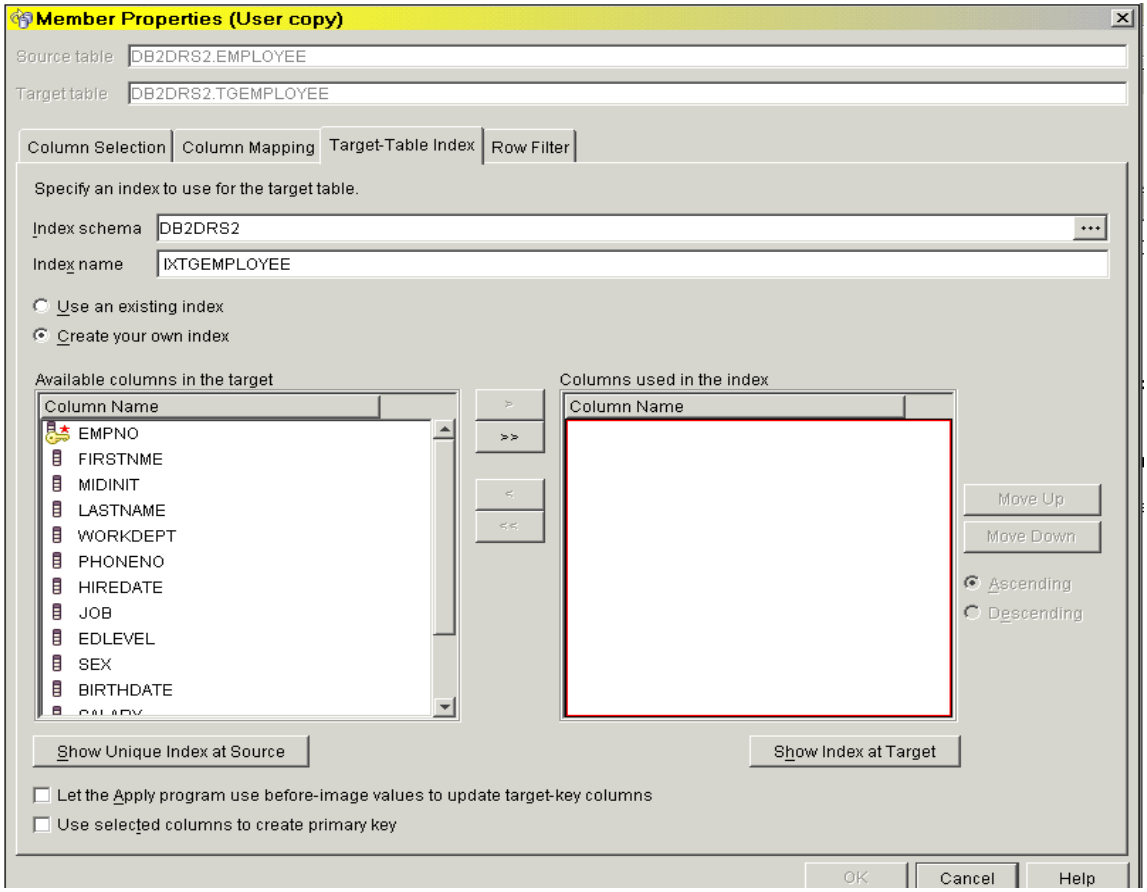


Figure 5-11 Member properties - Create another index

- Select available columns to create another target index, see Figure 5-9 on page ccxxvi
- **Show index at Target** push button. Click to display the a window showing the existing target indexes. See Figure 5-12

  For non-DB2 target tables, index information for the nickname will be shown. When the nickname for the existing target table was created, federated server retrieved information about the primary key or indexes on the target table and placed a record in the federated server's catalog table for each primary key and index. There is not a real index for the nickname in the DB2 database, only a record in its catalog about the primary key or index at the non-DB2 target server.

- The **OK** button will be active after you complete the target index options, indicating the member properties definition is complete and you can either go back to the Subscription Set note book page, see Figure 5-2 on page ccxiii. Or click the **Row filter** page, see Figure 5-13 on page ccxxxii

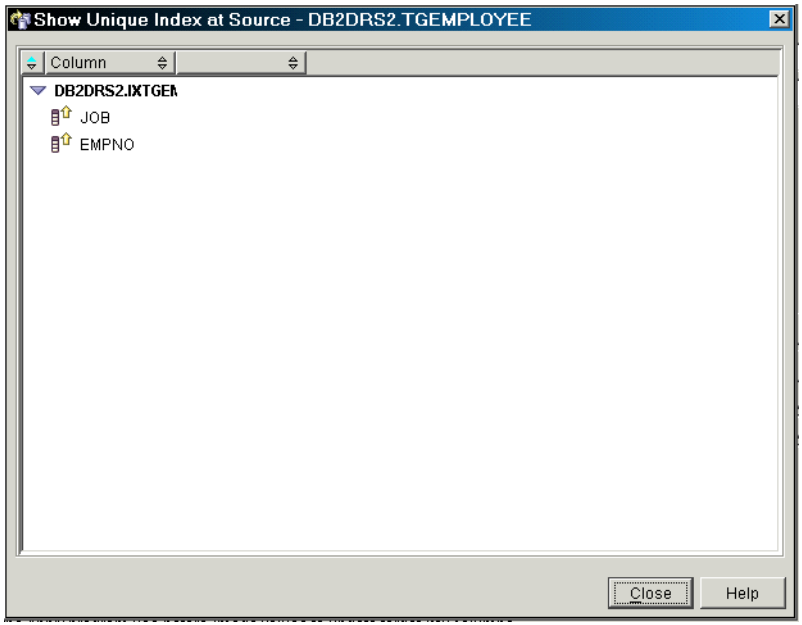


Figure 5-12 Show existing target indexes

Close button will take you back to Figure 5-14 on page ccxxxiv.

This note book page is displayed when you select the Row Filter page.

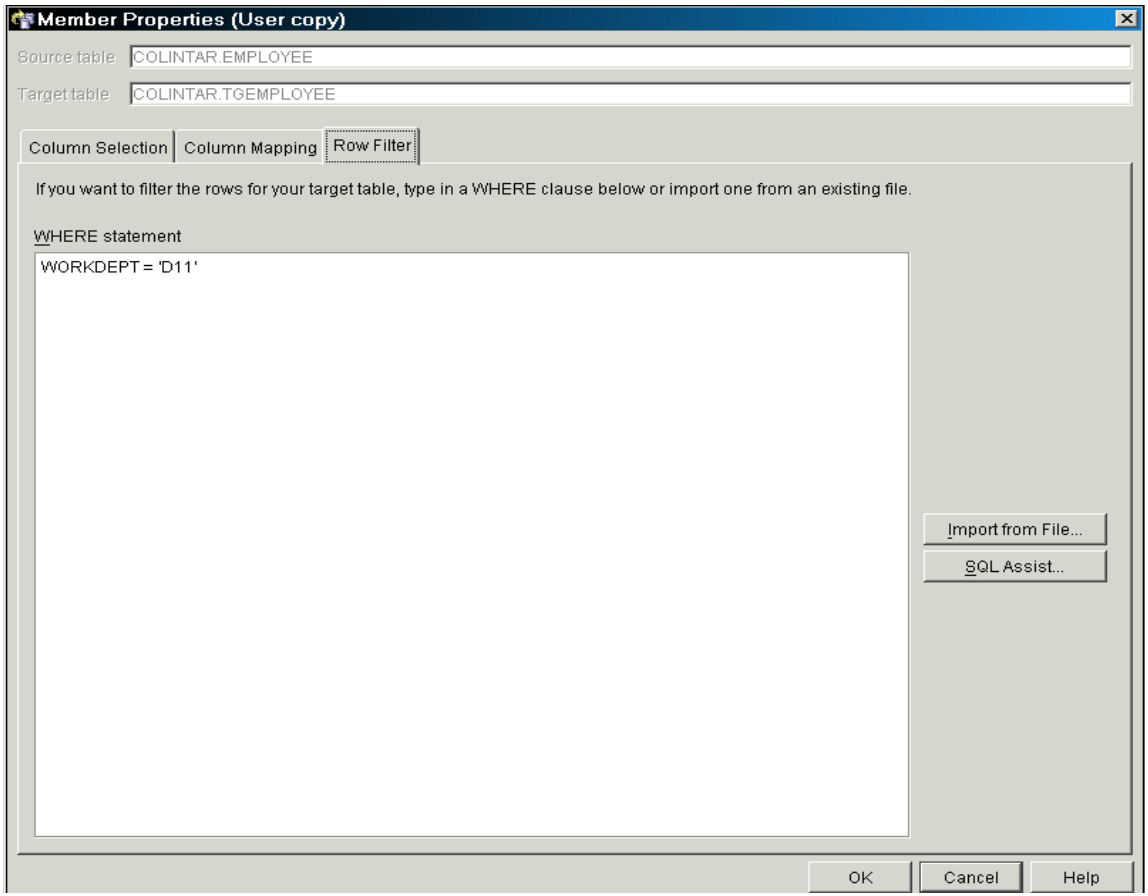


Figure 5-13 Member properties - Row selection

The Apply program, by default replicates all the rows from the source tables. However, if you only want to replicate specific rows to your target table, then this note book page provides that function. Enter a predicate in this window, which updates the PREDICATE column in the subscription member apply control table (IBMSNAP_SUBS_MEMBER). The apply program will use this predicate to select only the rows from the source table during full refresh or change updates.

For example: WORKDEPT=’D11’ predicate will select only the rows from the source table that contains ‘D11’ in column WORKDEPT. Notice that it is a WHERE clause SQL statement, but don’t enter the WHERE statement.

- You can type in a predicate, click the **Import from File** push button to bring in an existing predicate SQL script from a directory on your workstation or click the **SQL Assist** push button to help you build a predicate, see Figure 5-7 on page ccxxiv

- Clicking the **OK** push button when it becomes active, will take you back to the **Source-to-Target Mapping** page of the **Create Subscription Set** note book, see Figure 5-3 on page ccxvii

Note: Selecting records against the CD tables is not supported by the Replication Center, because the predicates entered on this page only applies to the source tables or views.

For example: To prevent deletes at the target table, by entering the following predicate: IBMSNAP_OPERATION <> 'D'. You need to manually update a column in the subscription member control table (IBMSNAP_SUBS_MEMBER) called UOW_CD_PREDICATES. The Apply program will use the predicate defined in this column against the CD or UOW table. The following SQL statement is a example to update this field.

UPDATE ASN.IBMSNAP_SUBS_MEMBR SET UOW_CD_PREDICATES ='IBMSNAP_OPERATION <>"D"'WHERE APPLY_QUAL ='apply_qual 'AND SET_NAME ='set_name 'ANDSOURCE_OWNER ='ALL 'AND SOURCE_TABLE ='source table'

|

This note book page is displayed when you select the Create target-table table space page. As you may have noticed in the previous screen captures, in some this table does not exist. Those are screen captures where the targer server is iSeries.

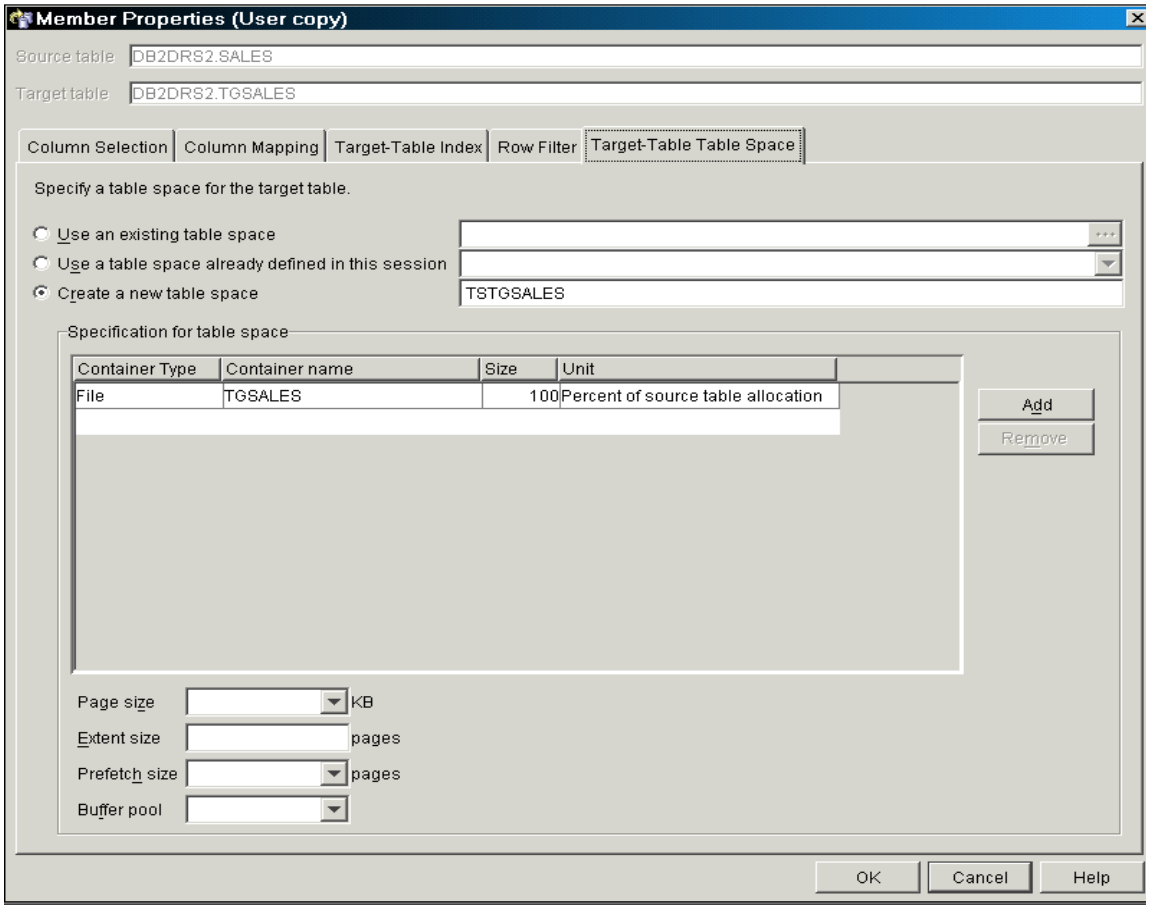


Figure 5-14 Member Properties - Create target-table table space

For non-DB2 target tables, the options for specifying the location of a target table to be created by Replication Center depend on the target server platform. For Informix target servers, the only option on this panel is to specify the *dbspace* for the target tables. After Replication Center's *Create Subscription* or *Add Member* window generates SQL, this SQL can be edited to add more details for the storage location of the target table before the SQL is run.

Attention: if your source table is in SMS you may want to pay particular attention to sizing an appropriately large DMS tablespace, or create an SMS tablespace outside of RC, and *Use an existing tablespace*.

Clicking **OK** when it becomes active, will take you back to the **Source-to-Target Mapping** page of the **Create Subscription Set** note book, see Figure 5-3 on page ccxvii

If your target table type is Replica, you can click the **Replica** page to display the Replica definitions notebook page, see Figure 5-15 on page ccxxxv to continue creating the subscription member replica target types.

This notebook page is displayed if the **replica** target type is selected from Figure 5-3 on page ccxvii.

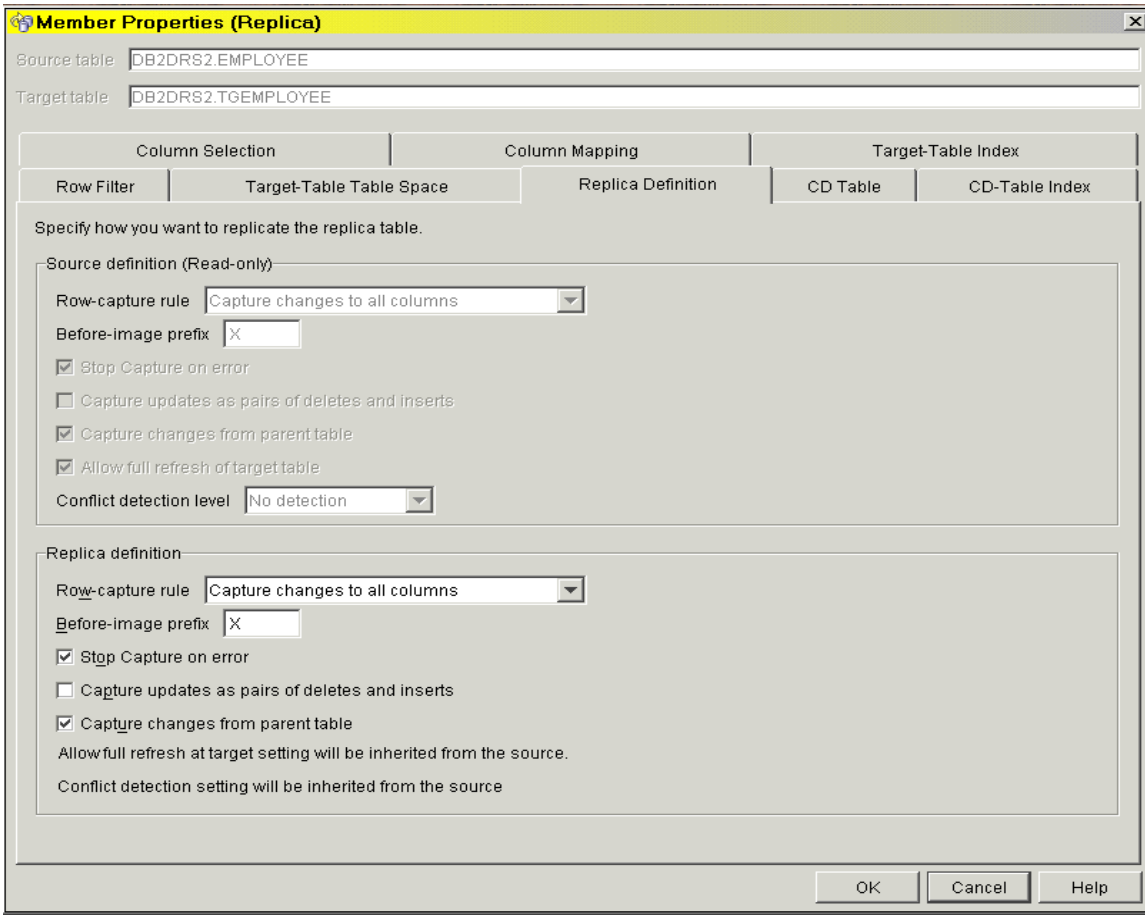


Figure 5-15 Member properties - Replica definitions

After defining the replica target table, you are prompt to definite the registration definition for the replica target table, which will create the CD table and update the capture control tables, that is used by the capture program running at the

target server. The check boxes in the **Replica definition** section of this page, also the **CD Table** and **CD table index** pages, is described in detail. See, Section 4.2.5, “CD Table” on page cxciii

- The **OK** button at this point will take you back to the Subscription Set properties note page, see Figure 5-2 on page ccxiii

Schedule

After creating or adding the member subscriptions from the **Source-to-Target Mapping**, clicking the **Schedule** page from the subscription set note book, see Figure 5-2 on page ccxiii, the following page is displayed, Figure 5-16.

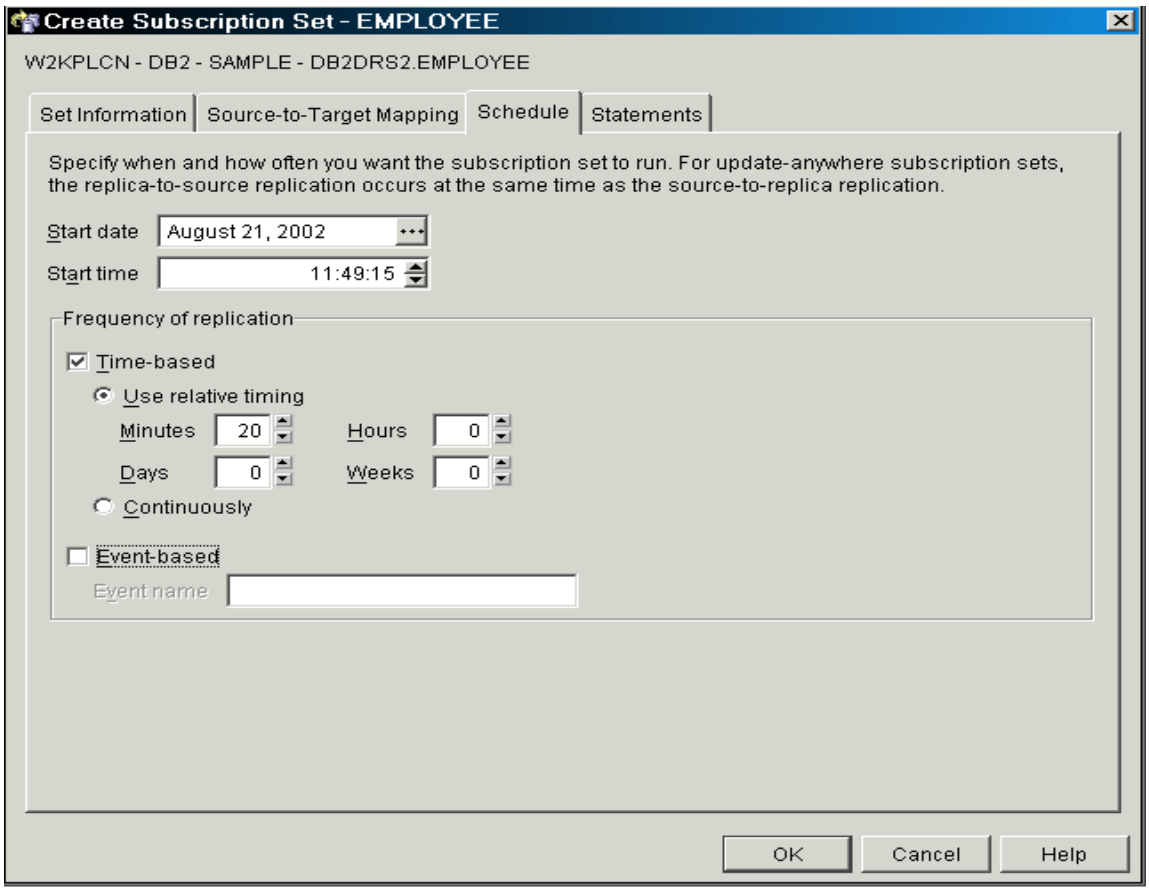


Figure 5-16 Subscription Set - Schedule replication

Scheduling your replication is specified on this note book page. There are two methods to schedule when to replicate, time based or event base, see Section 5.6, “Scheduling Replication” on page ccliii

- Time based method has two options:
  - Relative timing will schedule the apply program start and stop on a interval timing basis from 1 minute to 52 weeks, which will start from the **Start date** and **Start time** specified on this screen. Enter a numeric value by the **Minute**, **Hours**, **Days** or **Weeks** for specific interval timing options. For example: the page in Figure 5-16 indicates that replication will start on 8/21/02 at 11:49:15 and replicate every 20 minutes there after.
  - **Continuously**, clicking this radio button will schedule the apply program to run continuously until you stop it manually. For performance tip when you use this option see chapter?????

Xref chapter/section screen. also performance chap for Ccontine

- **Event-base**, click this box to schedule the apply program to start and stop, base on the start and stop times defined in the apply control table called: *IBMSNAP_SUBS EVENT.* This table is updated manually or automatically from a application program. The **event name** specified on this note book page correspond to the event name with a start and stop time in the events control table, see Section 5.6, “Scheduling Replication” on page ccliii.
- The **OK** button if active will generate the SQL script to update the apply control tables at the apply control server specified in Figure 5-2 on page ccxiii. See, Section 2.13.1, “Running Saved SQL Files Later” on page cxxiii. Also see Section 5.7, “SQL script description” on page cclv

Statements

After creating or adding the member subscriptions from the **Source-to-Target Mapping**, clicking the **Statements** page from the subscription set note book, see Figure 5-2 on page ccxiii, the following page is displayed,Figure 5-17 on page ccxxxviii

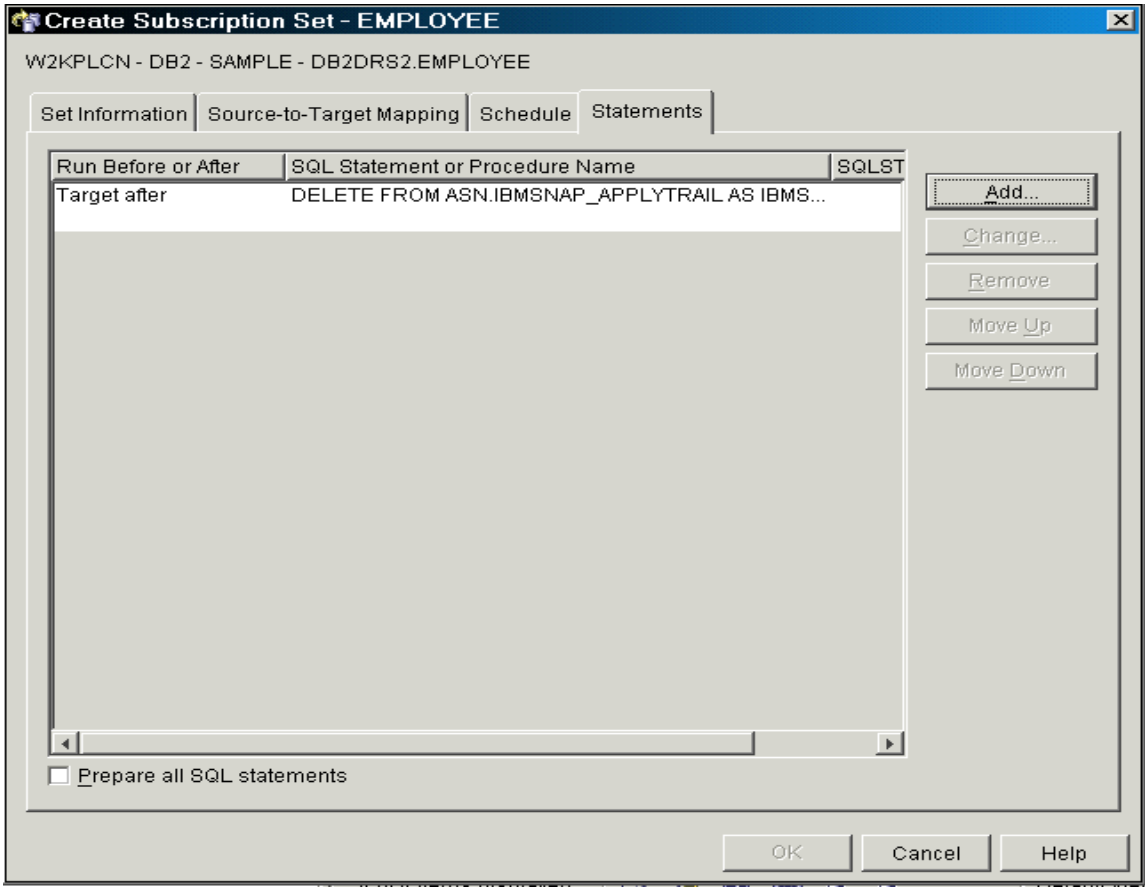


Figure 5-17 Subscription set - SQL before and after statement

You can create SQL statements or call procedures that are process each time the apply program process a subscription set. The SQL statements or procedures are updated in the *IBMSNAP_SUBS_STMTS* apply control table, that is defined from this note book page. For example, you could create a SQL statement to automate the process in maintaining the apply control tables.

- Clicking on the **Add** push button will display the following window to defined your SQL statement or procedure call, see Figure 5-17

- The **OK** button if active will generate the SQL script to update the apply control tables at the apply control server specified in Figure 5-2 on page ccxiii. See, Section 2.13.1, “Running Saved SQL Files Later” on page cxxiii. Also see Section 5.7, “SQL script description” on page cclv

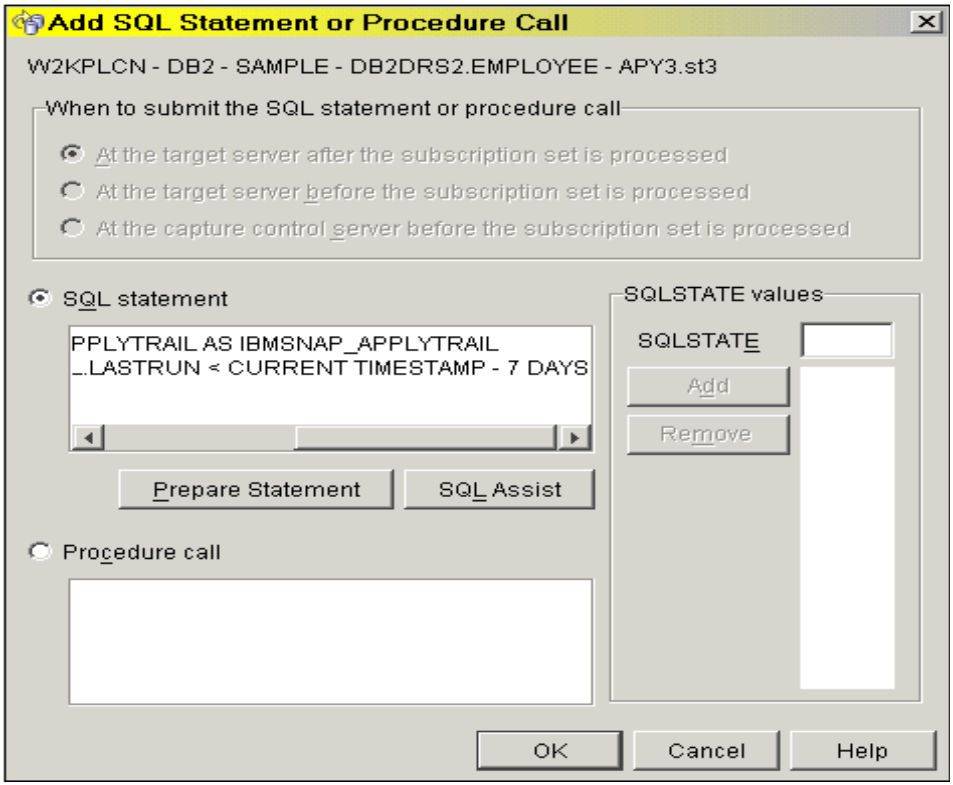


Figure 5-18 Add SQL Statement or Procedure

This is the window were you define the actual SQL statement or procedure calls:

- First click the radio button to specify on which server to run the SQL or procedure and whether you want to run it before or after the apply program run.
- If you select **SQL statement** radio button, you can either enter the SQL statement or press the **SQL Assist** button to assist you in creating the SQL statement. See Figure 5-19 on page ccxl.
- If you have typed an SQL statement, click **Prepare Statement**. The Replication Center will check the syntax of the SQL statements and checked that the objects to which they refer do in fact exist. If such an object does not exist, the Replication Center returns an error. If you know that an object referred to by the statement does not exist at the time you create the

subscription set, but will exist when the statement is run, you can ignore the error.

- If you select the **Procedure call** radio button you must enter a CALL before specifying the stored procedure name.
- **OK** button will either display SQL run screen or close this window and take you back to Figure 5-17 on page ccxxxviii

This window is display when the SQL Assist button is push from the Add SQL statement and Procedure call display Figure 5-17 on page ccxxxviii

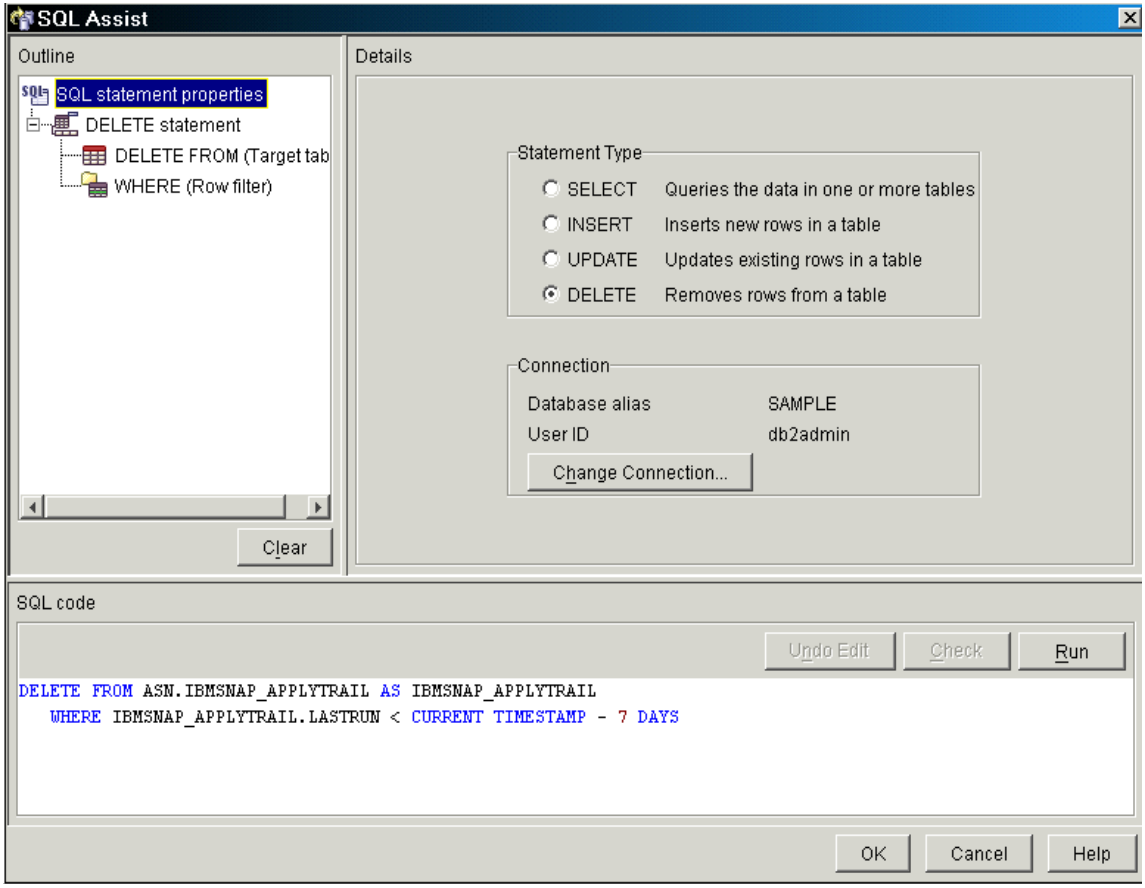


Figure 5-19 SQL Assistance

There are many functions of his screen to assist you in creating an SQL statement. Press the **HELP** key for detail instructions.

- The **OK** button will take you back to the Add SQL statement and procedure call, see Figure 5-17 on page ccxxxviii.

5.4 Target Types Descriptions

The type of target table you want depend on the how you want your data defined at the target and how your replication environment is defined. You could use an existing target table or create a new table.

Note the following restrictions:

- On non-DB2 relational target tables and indexes must follow the DB2 naming conventions.
- You can only select the following target type, if you source tables on non-DB2 relational databases:
- User copy
- Point-in-time
- External CCD
- You can only select the following target types, if your source tables on a OS/400 system is using the RRN column as a key column.
- Point-in-time
- External CCD
- For tables in a z/OS subsystem, the encoding scheme for the CD and UOW table must be the same if the Apply program will join these tables to satisfy a subscription-set WHERE clause for a user copy table.

5.4.1 User Copy

This table matches the data from the source file at the time of copy, that include only the columns defined in the subscription member. The structure can be same as the source table, or only a subset of the source columns. Before or after images and calculated column can also be defined. This target type also requires a unique index or primary key to apply the changes. Usually this target type is the most common for basic simple data replication. There are no additional replication control column.

5.4.2 Point-in-time

Similar to the features of a user copy with a timestamp column added, to indicate when the Apply program committed the row at the target. You can select this target type if you want to keep track of the time when the changes were applied to the target table.

5.4.3 Aggregate tables

This is a read only target table that summarizes the entire contents of a source table or changed data. The target columns are defined from the SQL column functions such as SUM, COUNT, MIN, MAX and AVG. These column contains the computed value of the SQL function not the actual source data. The data is appended to the target table at a specified timing interval. A timestamp is included to indicate when the Apply program performed the aggregation. There are two types of aggregation target tables:

Show actual setup e.g or show graphic e.g

Base aggregate

This target table type will summarize the entire contents of the source table during a replication cycle. For example, you could use this target type if you want to keep track of year to date sum or average sales by salesman or region.

Change aggregate

This target table will summarize the changes between replication cycles by reading the contents in the CD or CCD table, not the source table. For example, you could use this target type to track monthly sales totals by salesman, region or customer.

5.4.4 CCD (consistent change data)

This target type contains committed changes that occurred at the source, with replication control fields to determine the type operation from the source table (insert, update and delete). There are different types of CCD tables, depending on the your replication requirement. Therefore, you must decide where you want it located and what type of change data it must contain.

CCD tables attributes

The following table describes the columns definitions in a CCD table:

Table 5-1 CCD table column description

Column name	Description
User computed columns	User-defined columns that are derived from SQL expressions. The source data type can be converted to different target data types, by using computed columns with SQL scalar functions.
IBMSNAP_COMMITSEQ	The log or journal record sequence number of the captured commit statement. This value groups inserts, updates, and deletes by the original transactions for the source table.
IBMSNAP_INTENTSEQ	The log or journal record sequence number that uniquely identifies a change

Column name	Description
IBMSNAP_OPERATION	A flag to indicate the type of operation: I - insert, U - update and D - delete
IBMSNAP_LOGMARKER	The commit time at the Capture control server.
APPLY_QUAL	Optional Uniquely identifies which Apply program will process this CCD table.
IBMSNAP_REJ_CODE	Optional This value is set only during update-anywhere replication, if conflict detection is specified as standard or advanced when you define your replication source. It is not valid for non-DB2 relational targets because they cannot participate in update-anywhere configurations. The values are: 0 - A transaction with no known conflict. 1 - A transaction that contains a conflict where the same row in the source and replica tables have a change that was not replicated. When a conflict occurs, the transaction will be reversed at the replica table. 2 - A cascade-rejection of a transaction dependent on a prior transaction having at least one same-row conflict. When a conflict occurs, the transaction will be reversed at the replica table. 3 - A transaction that contains at least one referential-integrity constraint violation. Because this transaction violates the referential constraints defined on the source table, the Apply program will mark this subscription set as failed. Updates cannot be copied until the referential integrity definitions are corrected. 4 - A cascade-rejection of a transaction dependent on a prior transaction having at least one constraint conflict.
IBMSNAP_AUTHID	Optional The authorization ID associated with the transaction. It is useful for database auditing. AUTHID length is 18 characters. If you supply a longer value, it is truncated. For DB2 Universal Database for z/OS, this column is the primary authorization ID. For DB2 Universal Database for iSeries, this column has the name of the user profile ID under which the application that caused the transaction ran. This column holds a 10-character ID padded with blanks. This column is not automatically copied to other tables; you must select it and copy it as a user data column. This column can be selected as a user data column for a non complete CCD target table.
IBMSNAP_AUTHTKN	Optional The authorization token associated with the transaction. This ID is useful for database auditing. For DB2 Universal Database for z/OS, this column is the correlation ID. For DB2 Universal Database for iSeries, this column is the job name of the job that caused a transaction. This column is not automatically copied to other tables; you must select it and copy it as a user data column. This column can be selected as a user data column for a non complete CCD target table.

Column name	Description
IBMSNAP_UOWID	Optional The unit-of-work identifier from the log or journal record header for this unit of work.

Local or Remote CCD

- A *local* CCD tables resides in the source database server
- A *remote* CCD tables is located remotely from the source database. If there are many remote targets servers, a remote CCD table can be use as a source table to reduce network traffic from the source sever.

Complete or non complete

- A *complete* CCD table contains all of the data from the source table after the initial refresh, then afterward the change data is replicated from the source table.
- A *non complete* CCD table contains only the changes replicated from source table. No initial refresh is performed, thus leaving the CCD empty until the Apply program replicate the changed from the source table, but the original data.

Condense or non condense

- A *condense* CCD table contains the latest value for a row. Therefore, only one row exist with the same key value. If a row changed multiple times, the condense CCD table would contain just one row showing the latest result of the change.
- A non condense table contains all the changes that was made to source table, it could have multiple rows with the same key. Therefore, if a change was made 6 times to the same row on the source table, 6 rows will be inserted in the non condense CCD table showing the history of changes.

Define unique Indexes

If the CCD tables is defined as condense, a unique index or primary key must be created. A non condense CCD table contains multiple rows for the same key value, so an index or primary key is not required. See Figure 5-9 on page ccxxvi about creating target table indexes.

Types of CCD Tables

This table shows the different combination of attributes for a CCD table

Table 5-2 Type of CCD tables

Location	Complete	Condense	Description
Local	Yes	Yes	CCD table located in the source database, containing the same data as the replication source table
Local	Yes	No	CCD table located in the source database, containing the original data from the replication source table, and a history of subsequent changes
Local	No	Yes	CCD table located in the source database, containing only the latest change data
Local	No	No	CCD table located in the source database, containing all the change data
Remote	Yes	No	CCD table resides in a database access by the Apply program, containing the same data as the source table
Remote	Yes	No	CCD table resides in a database access by the Apply program, containing the original data from the replication source data, and a history of subsequent changes
Remote	No	Yes	CCD table resides in a database access by the Apply program, containing only the latest changes
Remote	No	No	CCD table resides in a database access by the Apply program, containing all the change data

Internal and External CCD tables

Depending on your replication environment, you can choose to register your complete CCD table as a replication source (an external CCD table), or you can set up your CCD table so that it is used implicitly as a source for replication (an internal CCD table).

External CCD tables

If you perform a full refresh on an external CCD table, the Apply program performs a full refresh on all target tables that use this external CCD table as a replication source. This process is referred to as a cascade full refresh. You can define more than one external CCD table for a replication source. An external CCD table can have any attributes you like (local or remote, complete or noncomplete, condensed or non condensed); however, if you use it to stage data, you must use a complete CCD table because the Apply program will use it to perform both full refresh and change replication.

Internal CCD tables

- Internal CCD tables are useful for staging changes. The Apply program uses the original source table for full refreshes, and it uses the internal CCD for change replication (instead of joining the CD and UOW table each time changes are replicated).
- Use an internal CCD table as a local cache for committed changes to a source table. The Apply program replicates changes from an internal CCD table, if one exists, rather than from CD tables.
- You can use an internal CCD table as an implicit source for replication without explicitly defining it as a replication source. When you add a subscription-set member, you can specify that you want an internal CCD table if the table has the following attributes:
  - It is a local CCD table. That is, the source server and the target server are the same database.
  - The CCD table is noncomplete.
  - No other internal CCD table exists for this replication source.

CCD Table as replication source for Multi Tier Staging

The basic replication environment is defined as a 2 tier configuration. Tier 1 is the replication source database and tier 2 is the Target database. If you want to replicate the target database from tier 2 to another two servers, an additional 3rd and 4th tier is added to the replication configuration using a CCD target table from the 2nd tier, see Figure 5-20 on page ccxlvii. The following will discuss the CCD target type in a multi tier environment.

Figure 5-20 CCD table multi tier replication

Complete, condense CCD tables

If you created this type of CCD table as a replication you can use it in the following ways:

- The staging table in a three-tier or four-tier distribution configuration. In this case, targets are maintained entirely from rows in the complete CCD table, with no additional overhead on the original source database.
- A CCD table that is maintained by an external program (such as IMS Data Propagator) and serves as a replication source for a CCD table in DB2 replication. In this case, the change data comes from an IMS database.

Typically you would create this table nearer the targets rather than local to the source to save network traffic during both change replication and full refreshes. Also, multiple updates to a single source table row will result in replication of a single row to all target tables.

Complete, non condense CCD table

- Do not define this type of CCD table as a replication source nor as an internal CCD table.

- Do not use this type of CCD table as a replication staging table because of the space consumed by saving each and every change.
- This table contains a complete set of rows initially, and it is appended with each and every row change. No information is overwritten, and no information is lost. Use this type of CCD table for auditing applications that require a complete set of rows.

Non complete, condense CCD table

You can define this type of CCD table as an internal CCD table for the replication source. By defining it as an internal CCD table, you access the original source during full refreshes only, and use the CCD table for updates. It is efficient because the join of the CD and UOW tables happens only once, to populate the CCD table, and then changes are replicated from the CCD table to all targets without the join overhead. Because the CCD table is noncomplete, full refreshes must come from the original source table. This type of table is also useful for condensing change data from a CD table. Condensing reduces the number of rows replicated to remote sites, and the number of SQL operations against the target tables, by eliminating all but the last change for each row.

Non complete, Non condense CCD table

In most cases, do not define this type of CCD table as a replication source. Use this type of table for auditing applications that do not require a complete set of rows, only the recently changed rows. You might define the CCD table as an internal CCD for a replication source when the remote targets themselves are non condensed. In this case, if there are many remote targets, you can benefit by avoiding repeated joins of the CD tables with the UOW table, as long as this benefit outweighs the cost of storing and maintaining the CCD table.

Internal CCD table with multiple targets

If you define an internal CCD table as a target, all other target tables associated with the source table will have changes replicated from that internal CCD table and not from the original source table. For this reason it is important to plan all the potential target tables to make sure that you define the internal CCD table correctly. If you do not include all the columns of the source table in the internal CCD table, but a target table includes all of those columns, then replication will fail. Similarly if the CD table used to maintain the internal CCD table does not include all the columns of the source table, but a target table includes all the columns, then replication will fail also.

Internal CCD tables do not support additional UOW table columns. If you define target CCD tables (with UOW columns) as a replication source, you cannot then define an internal CCD table. Do not use an internal CCD table if you already defined a target CCD table that includes UOW columns.

If you want to use column subsetting in an internal CCD table, review all previously-defined target tables to make sure that the internal CCD table definition includes all appropriate columns from the source tables. If you define the subscription set for the internal CCD table before you define any of the other subscription sets from this source, the other subscription sets are restricted to the columns that are in the internal CCD table.

5.4.5 Replica

Update anywhere replication, allows you replicate changes from the replica read/write target types table, then subsequently the source tables changes are are replicated to target server to update the replica target table. Therefore, at the source server we have the master table and the replica at the target.

An update anywhere replication is configured, with capture running at the source and target sever. The apply is configure to run only at the target server. The capture on the target server will apply any changes to the replica in the CD table at the target server, then the apply program will push those changes to the source system and apply the changes to the source tables. Any changes at the server is captured in the CD table, then the apply program will pull the changes as normal and update these changes to the replica target tables.

Restriction: Replica target tables can not be defined at non-DB2 target servers. A replica requires Capture at the target server, which is not possible with non-DB2 servers.

An update anywhere replication, could also be configured to have multiple replicas target table, against the same source table. A change at any of the replica target tables would replicate back to the source, then that change will eventually replicate to the other replicas target tables within this configuration. This is achieved because capture is running concurrently on all servers. See Figure 5-21

Figure 5-21 Update anywhere configuration with multi replicas

One of the issues to consider in a update anywhere replication is a conflict detection. If 2 users were updating the same row on both system at the same time. This would create a conflict detection situation when the same row is replicated to both servers at the same. The best way to prevent this problem is to design the application to avoid this if possible. However, during source tables registration, to help resolve this problem you need to know exactly what to do when a conflict detection occurs. There's a choice of three levels of confliction detections: None, Standard and Enhanced. See Section , "Capture changes from replica target table" on page clxxxiv for additional information

5.5 Data Blocking

Because of operations error, such as communication problems between your source and target sever, your Apply program not processing any changes for a period of time. Meanwhile, capture is still processing, which would generate a backlog of changes to replicate, when the Apply program resume processing, the backlog of changes could cause the following situations to occur.

- Network overload when transmitting the large backlog of changes from the server
- Spill file could overflow from memory, which causes additional overhead to the apply
- At the target server updating the target tables requires locking may rows, which could cause contention
- The logging resource to support the batch updates could exceed the target allocated log space.

Data blocking allows you to control the change data backlog situation by specify how many minutes worth of change data the Apply program can replicate during a subscription cycle. The number of minutes that you specify determines the size of the data block.This number is updated in the *MAX_SYNCH_MINUTES* column of the Subscription set table. To enter this number, see data blocking factor in Figure 5-2 on page ccxiii. When the backlog of change data is greater than the size of the data block, the Apply program changes a single subscription cycle into many mini apply cycles, which will reduce the backlog to manageable pieces. It also retries any unsuccessful mini-cycles and will reduce the size of the data block to match available system resources. See, Figure 5-22 on page cclii.

Figure 5-22 Data blocking

If replication fails during a mini-cycle, the Apply program will rerun the subscription set from the last successful mini-cycle, which is considered as another benefit in using data blocking. Instead of having to rerun the entire backlog of changes.

By default, the Apply program uses no data blocking, that is, it copies all available committed data that has been captured. If you enter a data-blocking value, the number of minutes that you set should be small enough so that all transactions for the subscription set that occur during the interval can be copied without causing the spill files or log to overflow.

The restrictions are:

- You cannot split a unit of work
- A previous mini cycle cannot roll back
- Only supported in change replication, cannot use data blocking to full refresh

5.6 Scheduling Replication

When you are designing your replication environment, you need to plan when to schedule replicating to your target tables. There are many factors that contributes on when you need to replicate your data.

The more frequent the Apply program replicates your changed data the less number of transactions it needs to apply. Therefore, less CPU cycles and lower system overhead.

If the target table application is not concerned on getting the changed data on a timely basis then you can schedule to replicate on a off peek time. Therefore, the changes are accumulated thru out the day, and applied at night as a large batch job.

If you need the changes replicated as soon as possible. For example: A sales order was receive by the customer from the web application, and it need to be process as soon possible on a target server. Therefore we need to schedule the apply program at target server to check on a frequent basis to see if there are any changed records to process.

As described in Section Figure 5-16, “Subscription Set - Schedule replication” on page ccxxxvi there are two methods to schedule replication: relative and event timing.

- Relative also known as interval timing is the simplest method of scheduling your subscription. The apply program will replicate the data at specific intervals. The timing interval could be continuos or specify the interval from one minute to one year. The interval is usually approximate, which depends on the workload or system resource available at the time. The interval pertains to all the subscription members within the subscription set. Therefore, all the tables will replicate within the subscription set at the same time. The timing interval can be changed from the replication center see Figure 5-16 on page ccxxxvi. You can make this change at any time.
  - For example, when you are creating your subscription, you can set the interval to 5 minute during the testing. When your subscriptions is ready for production you can change the interval time required for the target table application.
- Event timing is basically giving control to an application or a user to determine when to start replicating. This is accomplished by updating an event subscription control table (IBMSNAP_EVENT) with an event name that is associated with a subscription and a start and stop time, to trigger the apply program to start and stop replication. The following table shows the column specification of the IBMSNAP_EVENT table

Table 5-3 IBMSNAP_EVENT Column description

Column Name	Description
EVENT_NAME	The unique identifier of an event. This identifier is used to trigger replication for a subscription set
EVENT_TIME	An Apply control server timestamp of a current or future posting time. User applications that signal replication events provide the values in this column.
END_SYNCHPOINT	Optional. END_SYNCHPOINT A log sequence number that tells the Apply program to apply only data that has been captured up to this point. You can find the exact END_SYNCHPOINT that you want to use by referring to the signal table and finding the precise log sequence number associated with a timestamp. Any transactions that are committed beyond this point in the log are not replicated until a later event is posted. If you supply values for END_SYNCHPOINT and END_OF_PERIOD, the Apply program uses the END_SYNCHPOINT value because it then does not need to perform any calculations from the control tables to find the maximum log sequence number to replicate.
END_OF_PERIOD	Optional. A timestamp used by the Apply program, which applies only data that has been logged up to this point. Any transactions that are committed beyond this point in the log are not replicated until a later event is posted

Note, that this table is updated by a user or user application to post the events, not the replication center.

For example your batch application has to process a vital batch job step, before replicating the data. This batch job step run at different time each night, depending on the workload, so when this batch job finishes the next step is to execute SQL script to insert a row into the into the IBMSNAP_EVENT table, as follows:

CONECT TO Database USER username USING password;

INSERT INTO ASN.IBMSNAP_SUBS_EVENT (EVENT_NAME,EVENT_TIME)
VALUES ('NIGHTRUN',CURRENT TIMESTAMP +5 MINUTES)

Use the CONTECT statement if the apply control table exist on another server.

This SQL will post an event called NIGHTRUN, which will start the apply program to replicate the source data at the current system time plus 5 minutes after this SQL ends. The plus 5 minutes will ensure that a future event is posted in the events table, because when the apply program monitors this table, it is always looking for a future event to start replication, it will ignore past time events.

5.7 SQL script description

After you define your subscription using the replication center, the SQL script that is generated to create or change a subscription set and associated members, well be reviewed in this section. We'll breakdown the complete SQL script showing which control tables are updated at the capture and apply server.

The following SQL script is the first portion of creating a subscription set, which is inserting rows in the pruning control tables at the source server, to control the pruning process for the CD and UOW tables, and also initiating the handshaking process between the capture and apply servers.

```
CONNECT TO capture server USER XXX USING XXX;
INSERT INTO ASN.IBMSNAP_PRUNE_SET (
 APPLY_QUAL, SET_NAME, TARGET_SERVER, SYNCHTIME, SYNCHPOINT)
 VALUES ( 'APY1','SET1', 'SAMPLE', null, X'00000000000000000000' );

INSERT INTO ASN.IBMSNAP_PRUNCNTL (
 APPLY_QUAL, SET_NAME, CNTL_SERVER, CNTL_ALIAS,
 SOURCE_OWNER, SOURCE_TABLE, SOURCE_VIEW_QUAL,
 TARGET_OWNER, TARGET_TABLE, TARGET_SERVER,  TARGET_STRUCTURE, MAP_ID,
 PHYS_CHANGE_OWNER, PHYS_CHANGE_TABLE )
SELECT 'APY1', 'SET1', 'SAMPLE', 'SAMPLE','DB2DRS2','EMPLOYEE',0,'DB2DRS2','
TGEMPLOYEE', 'SAMPLE', 8, coalesce (char(max(INT(MAP_ID)+1)),'0'),'DB2DRS2',
'CDEMPLOYEE' FROM ASN.IBMSNAP_PRUNCNTL;
COMMIT ;
```

Figure 5-23 Subscription SQL Script - Update Pruning controls tables

In this part of the SQL script, the subscription set definition is updated at the target server.

```
CONNECT TO apply server USER XXX USING XXX;
INSERT INTO ASN.IBMSNAP_SUBS_SET (APPLY_QUAL,
SET_NAME,WHOS_ON_FIRST,SET_TYPE,  ACTIVATE, SOURCE_SERVER, SOURCE_ALIAS,
TARGET_SERVER, TARGET_ALIAS, STATUS, REFRESH_TYPE, SLEEP_MINUTES,EVENT_NAME,
MAX_SYNCH_MINUTES, AUX_STMTS, ARCH_LEVEL, LASTRUN, LASTSUCCESS,
CAPTURE_SCHEMA, TGT_CAPTURE_SCHEMA, OPTION_FLAGS, FEDERATED_SRC_SRVR,
FEDERATED_TGT_SRVR, COMMIT_COUNT, JRN_LIB, JRN_NAME) VALUES ('APY1','SET1',
'S','R',1, 'SAMPLE', 'SAMPLE','SAMPLE', 'SAMPLE',
 0,'R', 5, null, 0, 0, '0801','2002-08-27-10.43.31.0', null, 'ASN', null,
'NNNN',null,
 null,null, null, null );
```

Figure 5-24 Subscription SQL Script - Subscription Set control table

In this part of the SQL script the subscription member definition is updated at the target server.

```
INSERT INTO ASN.IBMSNAP_SUBS_MEMBR (APPLY_QUAL, SET_NAME, WHOS_ON_FIRST,
SOURCE_OWNER,SOURCE_TABLE, SOURCE_VIEW_QUAL,  TARGET_OWNER, TARGET_TABLE,
TARGET_STRUCTURE,TARGET_CONDENSED, TARGET_COMPLETE, PREDICATES,
UOW_CD_PREDICATES, JOIN_UOW_CD,  MEMBER_STATE,TARGET_KEY_CHG ) VALUES (
'APY1', 'SET1','S', 'DB2DRS2','EMPLOYEE', 0,'DB2DRS2','TGEMPLOYEE',
 8,'Y','Y','WORKDEPT = ''D11''', null, null, 'N','N' );
```

Figure 5-25 Subscription SQL Script - Subscription member control table

In this part of the SQL script the subscription column definition is created at the target server.

```
INSERT INTO ASN.IBMSNAP_SUBS_COLS (APPLY_QUAL, SET_NAME, WHOS_ON_FIRST,
 TARGET_OWNER, TARGET_TABLE, TARGET_NAME, COL_TYPE, IS_KEY,COLNO,EXPRESSION)
 VALUES ('APY1','SET1', 'S','DB2DRS2' ,'TGEMPLOYEE','EMPNO', 'A','Y',1,
'EMPNO');

INSERT INTO ASN.IBMSNAP_SUBS_COLS (APPLY_QUAL, SET_NAME, WHOS_ON_FIRST,
 TARGET_OWNER, TARGET_TABLE, TARGET_NAME, COL_TYPE, IS_KEY,COLNO,EXPRESSION)
 VALUES ( 'APY1', 'SET1', 'S','DB2DRS2', 'TGEMPLOYEE','FIRSTNME',
'A','N',2,'FIRSTNME');

INSERT INTO ASN.IBMSNAP_SUBS_COLS (APPLY_QUAL, SET_NAME, WHOS_ON_FIRST,
TARGET_OWNER, TARGET_TABLE, TARGET_NAME,  COL_TYPE, IS_KEY,COLNO,EXPRESSION)
VALUES ( 'APY1','SET1', 'S', 'DB2DRS2','TGEMPLOYEE','MIDINIT',
'A','N',3,'MIDINIT');
There is an INSERT statement for each column within the target table
```

Figure 5-26 Subscription SQL Script - Subscription column control table

In this part of the SQL script the subscription statement definition is updated at the target server.

.

```
INSERT INTO ASN.IBMSNAP_SUBS_STMTS( APPLY_QUAL, SET_NAME, WHOS_ON_FIRST,
BEFORE_OR_AFTER, STMT_NUMBER, EI_OR_CALL, SQL_STMT, ACCEPT_SQLSTATES) VALUES
('APY1','SET1','S','A',5,'E',
'DELETE FROM ASN.IBMSNAP_APPLYTRAIL IBMSNAP_APPLYTRAIL
 WHERE IBMSNAP_APPLYTRAIL.LASTRUN < CURRENT TIMESTAMP - 7 DAYS',NULL);

UPDATE ASN.IBMSNAP_SUBS_SET SET AUX_STMTS = AUX_STMTS + 1 WHERE APPLY_QUAL =
'APY1' AND SET_NAME = 'SET1' AND WHOS_ON_FIRST = 'S';COMMIT ;
```

Figure 5-27 Subscription SQL Script - Subscription statement control table

In this part of the SQL script the target table and index is created at the target server.

```
CONNECT TO apply server USER XXX USING XXX;
CREATE  TABLESPACE TSTGEMPLOYEE MANAGED BY DATABASE USING (FILE 'TGEMPLOYEE'
2048K);
CREATE TABLE DB2DRS2.TGEMPLOYEE( EMPNO CHARACTER(6) NOT NULL,FIRSTNME
VARCHAR(12) NOT NULL ,MIDINIT CHARACTER(1) NOT NULL ,LASTNAME VARCHAR(15)
NOT NULL , WORKDEPT CHARACTER(3),PHONENO CHARACTER(4),HIREDATE DATE ,JOB
CHARACTER(8), EDLEVEL SMALLINT  NOT NULL,SEX CHARACTER(1), BIRTHDATE DATE
,SALARY DECIMAL(9,2) ,BONUS DECIMAL(9,2), COMM DECIMAL(9,2)) IN
TSTGEMPLOYEE;
CREATE UNIQUE INDEX DB2DRS2.IXTGEMPLOYEE ON DB2DRS2.TGEMPLOYEE (EMPNO ASC);
COMMIT ;
```

Figure 5-28 Subscription SQL Script - Create target table and index

For target tables in non-DB2 servers, the SQL generated by Replication Center will include:

- CONNECT to the DB2 ESE/Connect database with the Server definition for the non-DB2 target server
- SET PASSTHRU to the non-DB2 server
- CREATE TABLE for the target table
- CREATE UNIQUE INDEX for the target table
- COMMIT for the create table and create unique index at the non-DB2 server
- SET PASSTHRU RESET

- CREATE NICKNAME for the target table
- ALTER NICKNAME statements if needed for any nickname columns to over-ride the default type mappings in the federated server wrapper and make the attibutes of these nickname columns compatible with the attributes of the source table columns.
- COMMIT for the create nickname and alter nickname

5.8 Create subscriptions using iSeries CL commands

The iSeries provides the option to create a subscriptions set and subscription member using native OS/400 CL Commands. The subscriptions created from these commands are only for replication on the iSeries. You can enter these commands, on the command line, or create a CL program and add these subscription commands in the program, then use the CALL commands to execute the CL program.

5.8.1 Add Subscription set - ADDDPRSUB

This command can either create an empty subscription set or subscription set with one member. The following screens will give you an idea of the parameters that are used within the ADDDPRSUB CL command. When it is entered on the command line, press the F4 key to display prompt screen, then the F11 key to display the actual parameter names:

```
Add DPR Subscription (ADDDPRSUB)

Type choices, press Enter.

Apply qualifier  . . . . . . . . . APYQUAL
Set name . . . . . . . . . . . . . SETNAME
Target table . . . . . . . . . . . TGTTBL        *NONE


  Library  . . . . . . . . . . .
Source table . . . . . . . . . . . SRCTBL        *NONE


  Library  . . . . . . . . . . .
Control server . . . . . . . . . . CTLSVR        *LOCAL
Source server  . . . . . . . . . . SRCSVR        *LOCAL
Capture control library  . . . . . CAPCTLLIB     ASN
Target capture control library   TGTCCLIB      *CAPCTLLIB
Target type  . . . . . . . . . . . TGTTYPE       *USERCOPY
Refresh timing . . . . . . . . . . TIMING        *INTERVAL

More...
```

Figure 5-29 ADDDPRSUB CL command screen prompt - 1st screen

After you enter a required value for the **Apply qualifier** and **Set name** press the F10 key for additional parameters, then the page down key, to display the following screens of parameters

```
 Add DPR Subscription (ADDDPRSUB)

 Type choices, press Enter.

 Interval between iterations:     INTERVAL
   Number . . . . . .                 1
   Interval . . . . . .               *DAY
                          + for more values
 Key columns  . . . . . . . . . KEYCOL         *SRCTBL

                          + for more values


                           Additional Parameters

 Activate subscription  . . . . . ACTIVATE       *YES
 Create target table  . . . . . . CRTTGTTBL      *YES
 Check target table format  . . . CHKFMT         *YES
More..
```

Figure 5-30 ADDDPRSUB CL command screen prompt - 2nd screen

Press the page down to display the 3rd screen.

```
 Add DPR Subscription (ADDDPRSUB)

 Type choices, press Enter.

 Source columns . . . . . . . . . COLUMN         *ALL

                          + for more values

 Unique key . . . . . . . . . . . UNIQUE         *YES
 Target columns:                  TGTCOL
   Column . . . . . . . . . . .                  *COLUMN

   New column . . . . . . . . .

                          + for more values
More...
```

Figure 5-31 ADDDPRSUB CL command screen prompt - 3rd screen

Press the roll down key to display the 4th screen.

```
 Add DPR Subscription (ADDDPRSUB)

 Type choices, press Enter.

 Calculated columns:             CALCCOL
   Column . . . . . .                *NONE

   Expression . . . . . .


                          + for more values
 Row selection expression . . . . ROWSLT          *ALL
More...
```

Figure 5-32 ADDDPRSUB CL command screen prompt - 4th screen

Press the page down to display the 5th screen

```
 Add DPR Subscription (ADDDPRSUB)

 Type choices, press Enter.

 SQL to run before:              SQLBEFORE
   SQL statement  . . . . . .                *NONE



   Server to run on . . . . . .
   Allowed SQL states . . . . . .
                          + for more values
                          + for more values
More...
 F3=Exit    F4=Prompt    F5=Refresh    F12=Cancel    F13=How to use this display
```

Figure 5-33 ADDDPRSUB CL command screen prompt - 5th screen

Press the page down to display the 6th screen

```
 Add DPR Subscription (ADDDPRSUB)

 Type choices, press Enter.

 SQL to run after:               SQLAFTER
   SQL statement  . . . . . .                *NONE



   Server to run on . . . . . .
   Allowed SQL states . . . . . .
                          + for more values
                          + for more values
 Maximum synchronization time:    MAXSYNCH
   Number . . . . . .                       0
   Interval . . . . . .                     *MIN
                          + for more values
 Commit count . . . . . . . . . . CMTCNT        *DEFAULT
 Target key change  . . . . . . . TGTKEYCHG     *NO
 Add DPR registration . . . . . . ADDREG        *NO

More...
```

Figure 5-34 ADDDPRSUB CL command screen prompt - 6th screen

Press the page down to display the last screen for this command.

```
 Add DPR Subscription (ADDDPRSUB)

Type choices, press Enter.

Federated server . . . . . . . . FEDSVR         *NONE

Bottom
```

Figure 5-35 ADDDPRSUB CL command screen prompt - Last screen

For detail information on the parameter values, you can use the field level help, by moving the cursor to the parameter and press the F1 key. Or refer to Chapter 18 in the *Replication Guide and Reference,* SC27-1121-00

5.8.2 Add subscription members - ADDDPRSUBM

This command will add a subscription set member to existing subscription set. The following screens will give you an idea of the parameters that are used within the ADDDPRSUBM CL command. When it is entered on the command line, press the F4 key to display prompt screen, then the F11 key to display the actual parameter names:

```
 Add DPR Subscription Member (ADDDPRSUBM)

 Type choices, press Enter.

 Apply qualifier  . . . . . .
 Set name . . . . . .
 Target table . . . . . .


   Library  . . . . . .                Name
 Source table . . . . . .


   Library  . . . . . .                Name
 Control server . . . . . .   *LOCAL
 Source server  . . . . . .   *LOCAL
 Target type  . . . . . .   *USERCOPY     *USERCOPY, *REPLICA...
 Key columns  . . . . . .   *SRCTBL
               + for more values
More...
```

Figure 5-36 ADDDPRSUBM CL command screen prompt - 1st screen

After you enter a required value for the **Apply qualifier**, **Set name, Target table** and **Source table** press the F10 key for additional parameters, then the page down key, to display the followings screens of parameters:

```
 Add DPR Subscription Member (ADDDPRSUBM)

 Type choices, press Enter.


                            Additional Parameters

 Create target table  . . . . . .   *YES          *YES, *NO
 Check target table format  . . .   *YES          *YES, *NO
 Source columns . . . . . . . . .   *ALL
               + for more values
 Unique key . . . . . . . . . . .   *YES          *YES, *NO
 Target columns:
   Column . . . . . . . . . . . .   *COLUMN
   New column . . . . . . . . . .
               + for more values
More...
```

Figure 5-37 ADDDPRSUBM CL command screen prompt - 2nd screen

Press the page down to display the last screen

```
 Add DPR Subscription Member (ADDDPRSUBM)

 Type choices, press Enter.

 Calculated columns:
   Column . . . . . . . . . . . .   *NONE
   Expression . . . . . . . . . .


               + for more values
 Row selection expression . . . .   *ALL


Target key change  . . . . . .   *NO          Character value, *NO, *YES
 Add DPR registration . . . . . .  *NO          *YES, *NO
Bottom
```

Figure 5-38 ADDDPRSUBM CL command screen prompt - Last screen

For detail information on the parameter values, you can use the field level help, by moving the cursor to the parameter and press the F1 key. Or refer to Chapter 18 in the *Replication Guide and Reference,* SC27-1121-00

6

Chapter 6.

Operating Capture and Apply

In this chapter, the following topics are discussed:

- Basic operations from Replication Center
- Basic operations from the command prompt
- Platform specific considerations for DB2 UDB for UNIX and Windows, DB2 UDB for z/OS and DB2 UDB for iSeries
- Troubleshooting the operations
- Capture and Apply parameters
- Other operations
- Using ASNLOAD for the initial load

6.1 Basic operations on Capture and Apply

The programs which enable the replication to take place according to the registrations and subscriptions, are Capture and Apply. There are cases where it is not necessary to run Capture but the Apply program must run in order to replicate anything from source to target. The basic operations on Capture and Apply are starting, stopping and querying the Capture and Apply programs.

There are several ways to start the Capture and Apply programs:

- Using the Replication Center interface
- Calling the command from the operating system prompt or shell prompt
- Starting as a service from the Services on Control Panel on Windows operating systems
- Using JCL to run as batch or started task on z/OS

Both Capture and Apply programs accept parameters. Some of these parameters are optional where a few of them are mandatory. These parameter values come from more than one resource. Capture and Apply uses the order below to assign values to parameters.

1. Start-up parameter
2. Parameter tables (IBMSNAP_CAPPARMS and IBMSNAP_APPPARMS)
3. Shipped defaults

6.1.1 Basic operations from the Replication Center

After you register the replication sources and create subscription sets from the Replication Center, you can also use Replication Center to start and/or stop Capture and/or Apply. You should have already added the Capture control server and/or apply control server to your Replication Center before defining the registrations and the subscriptions. You can follow the steps below to operate either Capture or Apply from Replication Center:

1. Expand **Operations**.
2. If you want to operate Capture, select **Capture Control Servers**.
   a. On the content pane, select and right click the server you want to operate.
   b. Select either **Start Capture** or **Stop Capture** depending on your choice.
3. If you want to operate Apply, expand **Apply Control Servers.**
   a. Expand the server you want to operate.
   b. Select the **Apply Qualifiers**.
   c. On the content pane, select and right-click the apply qualifier you want.
   d. Select either **Start Apply** or **Stop Apply** depending on your choice.

Alternatively you can start Capture and Apply from the *Launchpad* at Replication Center.

In this section, we describe how to start and stop Capture and Apply with the most common parameters. You can find other parameters on 6.2, “Capture and Apply Parameters” on page cccxxv.

The Capture and Apply programs can be operated from Replication Center or command prompt for all platforms. You can find the platform specific requirements, restrictions and alternative methods for DB2 UDB for UNIX and Windows on 6.1.3, “Considerations for DB2 UDB for UNIX and Windows” on page ccci, for DB2 UDB for z/OS on 6.1.4, “Considerations for DB2 UDB for z/OS” on page cccvi and for DB2 UDB for iSeries on 6.1.5, “Considerations for DB2 UDB for iSeries” on page cccxvi.

Attention: If you want to operate Capture and Apply for remote servers from the Replication Center, DB2 Administration Server (DAS) on the remote server should be started.

Attention: There are alternative ways to start Capture and Apply for DB2 UDB for z/OS. If you prefer to use Replication Center to operate Capture and Apply for z/OS, the DB2 Administration Server (DAS) should be installed on DB2 UDB for z/OS. See Section 6.1.4, “Considerations for DB2 UDB for z/OS” on page cccvi.

If you are operating Capture and Apply for iSeries go to Section 6.1.5, “Considerations for DB2 UDB for iSeries” on page cccxvi.

Starting Capture for DB2 for UNIX and Windows and z/OS

Replication Center accepts start-up parameters for Capture from *Start Capture* window and prepares and runs the command to start the Capture. You can see the *Start Capture* window displayed for *SAMPLE* database on Figure 6-1.

Attention: If the Capture control server is DB2 UDB for UNIX and Windows, the database must be enabled for replication. If it is not enabled, you will receive a message indicating that it is not configured for *archival logging*. You can enable it from Replication Center. In order to check if your database is enabled for replication and to enable it, refer to Section 6.1.3, “Considerations for DB2 UDB for UNIX and Windows” on page ccci.

The *Capture server* is the database you selected at the Replication Center to start capture for. Capture must read control information in order to start. The control information are on the DB2 tables whose schema are *capture schema*. Therefore *capture schema* is also essential for Capture to start. Default capture schema is ASN. If you are using multiple capture schemas for this capture control server or you used a capture schema different than ASN, you must specify it by highlighting it from the **Capture schema** list at the top of the screen.

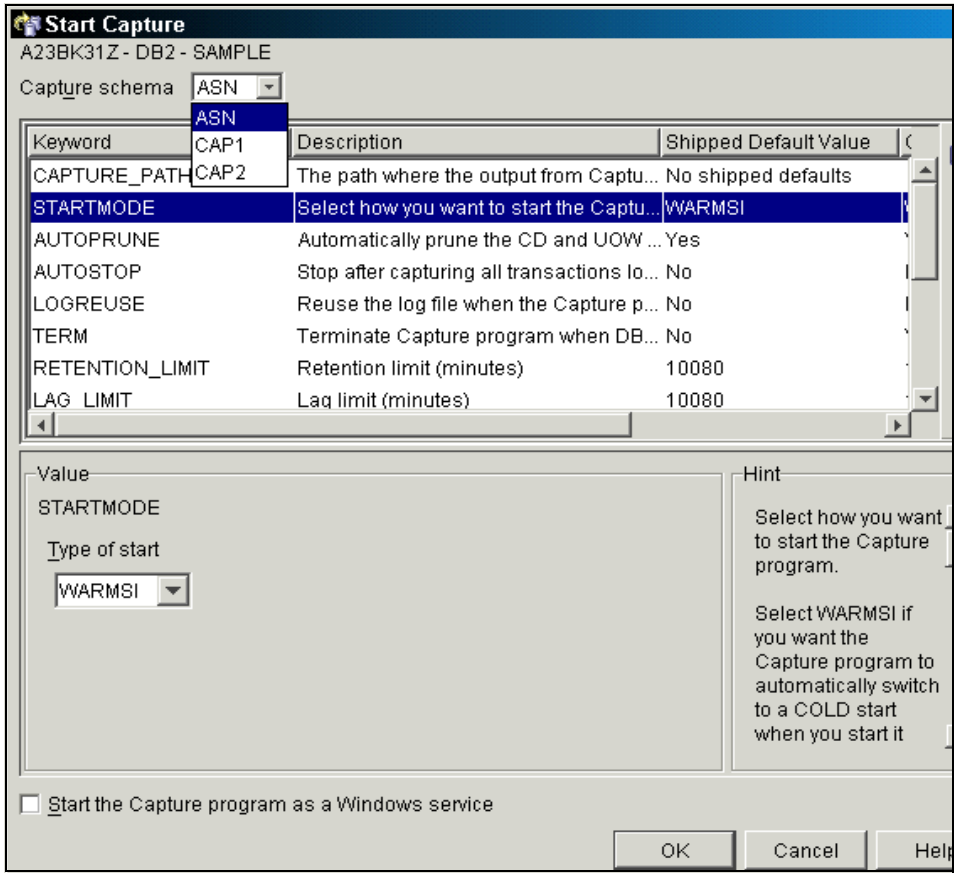


Figure 6-1 Start Capture parameter specification from Replication Center

Each parameter is specified with a keyword. Parameter values may originate from three different sources except for *capture_server*, *capture_schema* and *capture_path*:

- Each parameter has a shipped default.
- Each parameter has a default for that capture schema in the IBMSNAP_CAPPARMS. The defaults are stored as one row in this table and

this row is inserted when the table is created by the Replication Center. The values in this control table can be altered or set to null.

- A parameter may have an overridden value for this instance of Capture.

If a parameter is overridden at start-up, Capture assigns the start-up value for that parameter. If parameter is not overridden, reads the value for this parameter from IBMSNAP_CAPPARMS. In case the value in the IBMSNAP_CAPPARMS for this parameter is null or the row in IBMSNAP_CAPPARMS is deleted, assigns the shipped default.

It is not possible to change the shipped default anyway. The defaults created in the IBMSNAP_CAPPARMS table can be altered but not when starting the Capture from the Replication Center. It is only possible to give overridden values when starting Capture.

Capture directs all its file I/O to *capture_path*. This includes the log file, spill files and others. There is no shipped default for the capture_path. There is a column in IBMSNAP_CAPPARMS for capture_path but it is initially null. If you specify a path on the *Start Capture* window, then capture starts with this start-up value for capture_path. If you do not specify a path, then *capture_path* is set to the working directory you should give on *Run Specifications* at *Run Now or Save Command* window.

Capture creates a log file on the capture path.You can find the more about the log file on Section 6.1.2, “Basic operations from the command prompt” on page cclxxxviii.

The *startmode* can be either of the following:

WARMSI: Warm start, switch initially to cold start.
WARMNS: Warm start, never switch to cold start.
WARMSA: Warm start, always switch to cold as necessary.
COLD: Cold start.

If this is your first run of Capture for this schema, replication sources should be replicated to the targets with full refresh. A cold start initiates a full-refresh. All the start modes except WARMNS provide switch to cold start and can be used as the startmode on the first run. Generally, the safest way is to use WARMNS start mode that do not switch to cold start but this mode is not applicable for the first run. WARMSI also provides this safety and you do not have to change the startmode after the first run.

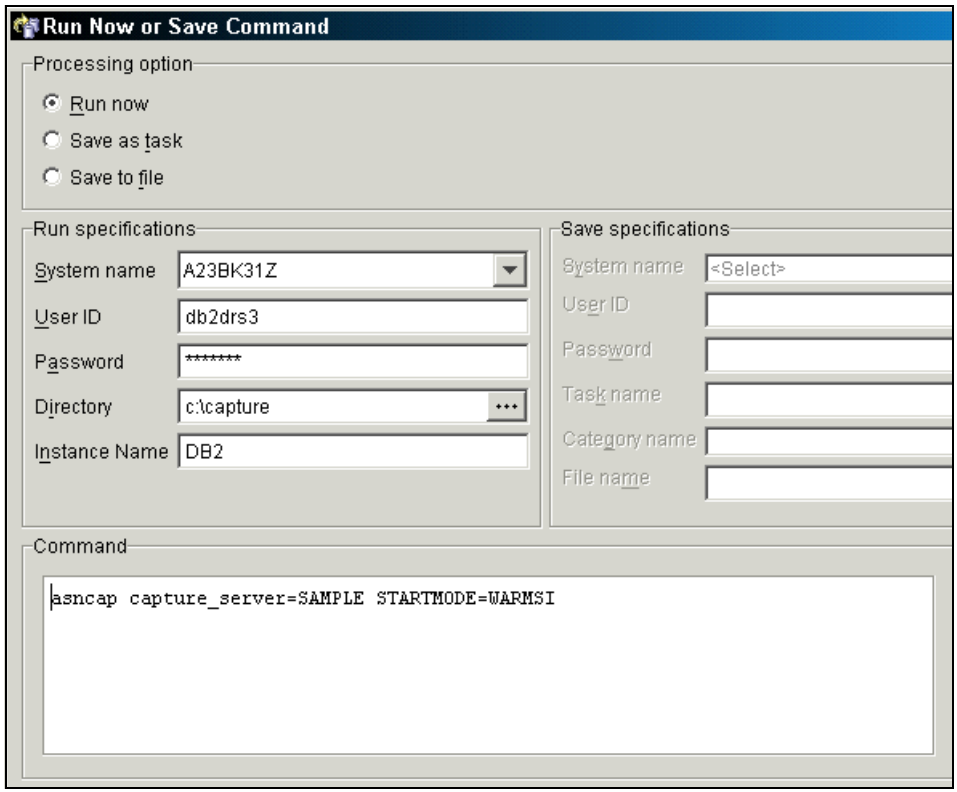


Figure 6-2 Start Capture

In our example on Figure 6-1, we have only overridden the startmode. Replication Center generates the command seen on Figure 6-2. *Capture_path* is not specified as a parameter. All the files will be created by Capture on the working directory specified at *Run Specifications*.

Note that *capture_server* is a mandatory parameter. ASN is the shipped default for *capture_schema*. These two parameters are essential for Capture to find the control tables and start running.

Capture can be started as a Windows service on Windows 2000 and Windows NT operating systems. If you mark the check-box on Figure 6-1 on page cclxviii, the command to create and start the service is prepared by Replication Center. You need to do some modifications before running the commands. Refer to Section , “Starting Capture and Apply as a Windows service” on page ccciv for necessary modifications. How to create a replication service (**asnscrt**) and drop a replication service (**asnsdrop**) are also explained on that section.

Starting Capture for iSeries

After following the steps specified earlier in this section, when expanding the operational folder from the Replication Center, then the capture control servers. The following window is displayed showing the default parameters for you to change when starting the Capture program.

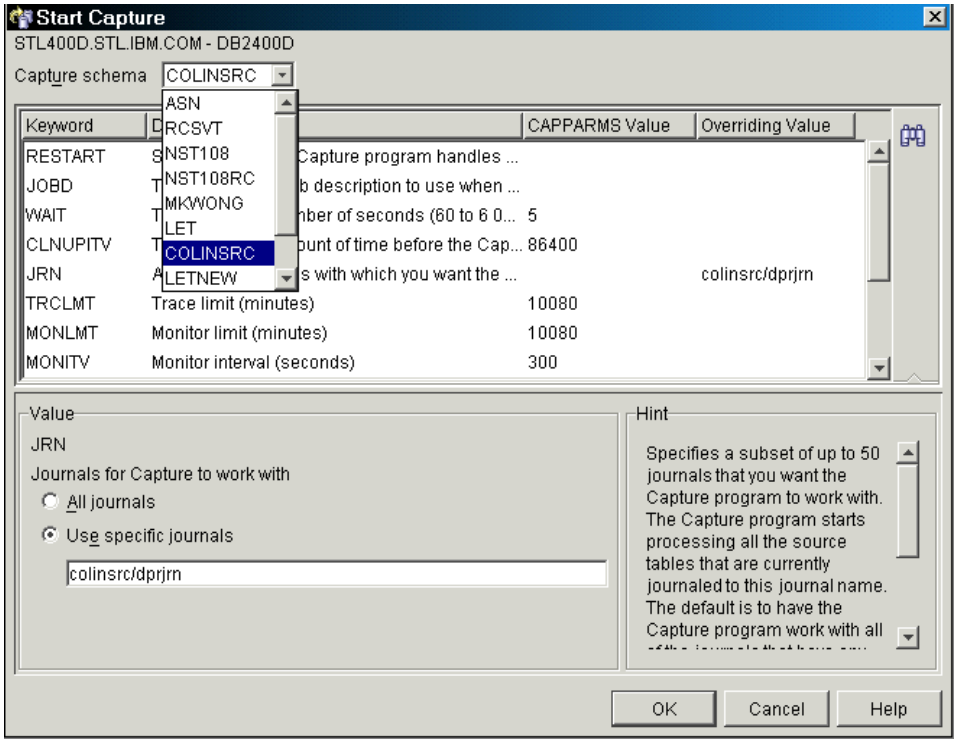


Figure 6-3 Start iSeries Capture program parameters

- The **Capture schema** is selected from this list box. In this example the capture control tables resides in a schema called *COLINSRC*. The default capture schema *ASN*.
- The **CAPPARMS Value** column contains defaults values for the cooresponding **Keyword** parameter. For example, the *CLNUPITV* keyword has a CAPPARMS default value of *86400*. The defaults values that are specified in that column comes from a table called IBMSNAP_CAPPARMS. When you change a value in that table, it becomes the new default value when ever you select this screen to start Capture, See 6.2.1, “Change Capture parameters” on page cccxxvi. However, you can override these values for only this instance of the Capture program.

- The other defaults values not defined in the CAPPARMS tables are changed in the **Value** section of the screen. The changes are only are this instance of the Capture program. For example:
  - The *JRN* overriding parameter value shown in Figure 6-3 is example on overriding the Capture program to use a specific journal to capture changes from the source tables that are registered only to a journal called *DPRJRN*. Instead of capturing changes from all the journal that contain registered source tables.

The Capture keyword parameter values are described in detail in Chapter 18 of the *DB2 Universal Database Replication Guide and Reference,* SC27-1121-00.

Click **OK** to generate the command to start the Capture program with the override values. See, Figure 6-4.

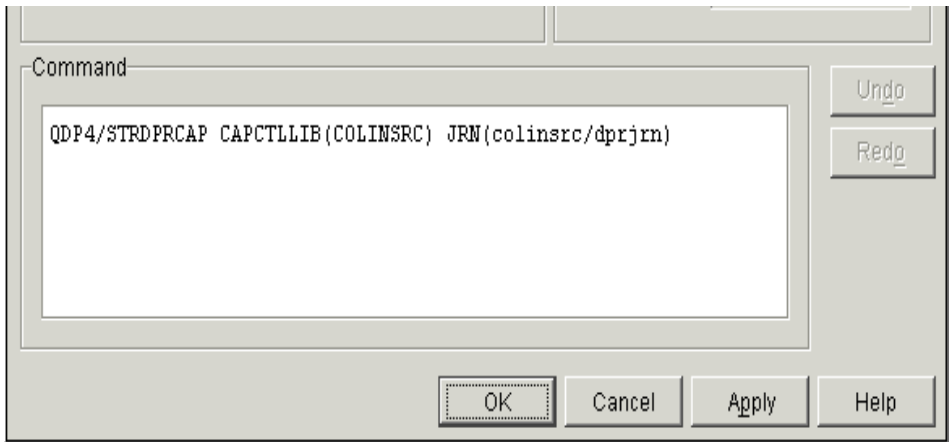


Figure 6-4 Start Capture command on the iSeries

As you can see the command that is generated is the STRDPRCAP command with the JRN parameter overridden with a specific journal. This command is sent to the iSeries to run the Capture program on the source server. See, Section 6.1.5, “Considerations for DB2 UDB for iSeries” on page cccxvi for details on this command.

Starting Apply for DB2 for UNIX and Windows and z/OS

Capture must run on the source server. Apply can run on any system but should be able to connect to source, target and and control server. If you start Apply on any other place than target, Apply runs in push mode. If you start Apply on the target server, Apply runs in pull mode. Running Apply in push or pull mode has a performance implication which is discussed at Chapter 10, “Performance” on page cdxli.

The server where the Apply control tables for the apply qualifier is its control server. The Apply control server must be configured as an application requestor (AR) for source and target servers. Source and target servers must be configured as application servers (AS).

If Apply is running on DB2 UDB for z/OS, refer to Section , “Configuring z/OS apply control server as AR” on page cccviii for connectivity of Apply.

If Apply is running on DB2 UDB for UNIX or Windows and:

- if the source server is a remote database, source server must be cataloged at the server where Apply is running.
- if the target server is a remote database, target server must be cataloged at the server where Apply is running.

In our sample configuration on Figure 6-5, capture control server (SAMPLE) is on Windows. The target (*AIXSAMP*) which is the apply control server is on AIX. Apply is running at the target in pull mode. Source is cataloged to the apply control server (*STHELENS*) with the following commands, since it is a remote server:

```
db2 catalog tcpip node nodewin1 remote 9.1.39.85 server 50000
db2 catalog database sample at node nodewin1
```

Figure 6-5 Sample replication configuration on Windows and AIX

When you start Apply from the Replication Center, you specify the system where Apply will run. The system name is given from the *System* pull-down menu on

top of *Start Apply* window. Apply is destined for STHELENS in our example on Figure 6-5.

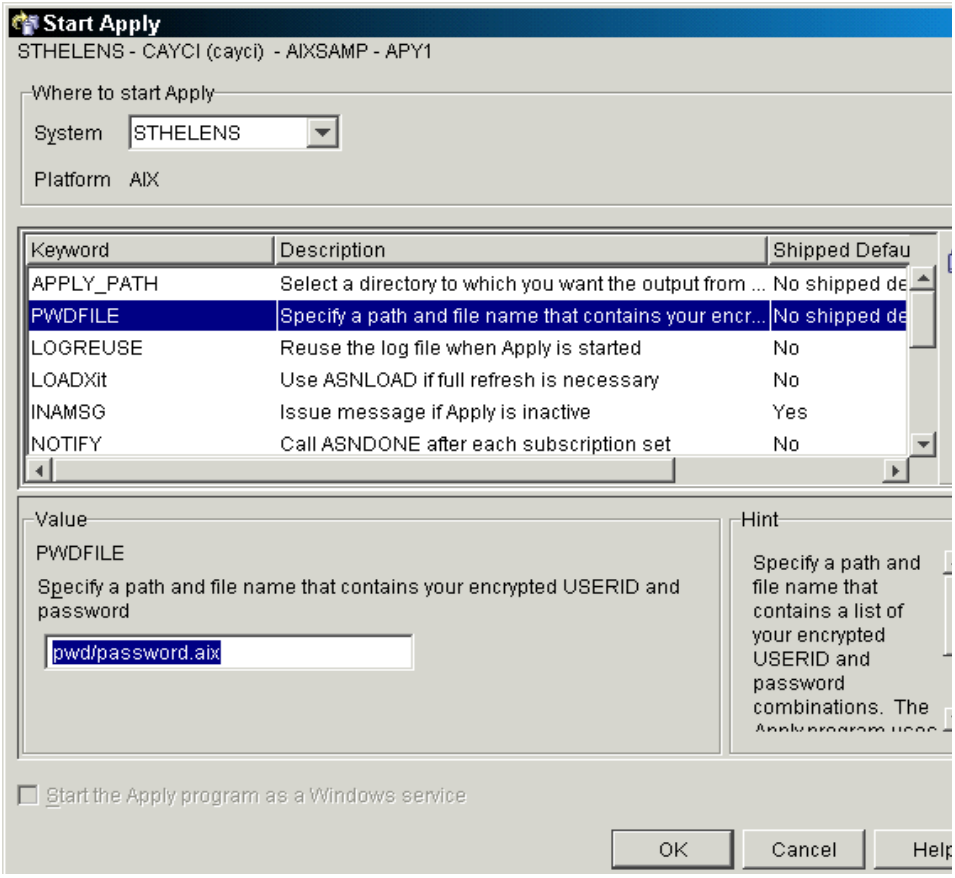


Figure 6-6 Start Apply parameter specification from Replication Center

The *control_server* and *apply_qual* are two mandatory parameters for Apply and they have already been specified before reaching this screen. You see control server (AIXSAMP) and apply qualifier (APY1) on the top left corner of the *Start Apply* window.The other parameters on this screen are all optional parameters. You can specify values for the parameters override the defaults from *Start Apply* window.

Parameter assignment method of Apply is very much like the Capture. Every parameter has a keyword. There are three possible values for each parameter, the shipped default, the defaults in the IBMSNAP_APPPARMS and the start-up value. Apply uses the same precedence as Capture such that shipped defaults

are overridden by the values in IBMSNAP_APPPARMS parameter table and start-up values override the values in the parameter table.

There is one set of apply control tables on an apply control server. There is always one row in IBMSNAP_CAPPARMS and this row is inserted by the Replication Center when capture control tables are created. The apply parameter table can have zero or more rows and it is not populated by the Replication Center. Each row defines the defaults for one apply qualifier. If there is a row for an apply qualifier that Apply is processing, Apply uses the defaults from this row to override the shipped defaults.

On this version of the product, the parameters values from the IBMSNAP_APPPARMS do not appear on the *Start Apply* window but will be available on a future fixpack. Although the values do not display on the Replication Center, Apply still uses the parameters from IBMSNAP_APPPARMS as described above. If you want to change shipped defaults for an apply qualifier, you can insert a row into the table with an SQL statement. The parameters that exist in IBMSNAP_APPPARMS are described in Chapter 3.4.5, “Control tables described” on page clxv.

The *apply_path* usage is also similar to Capture. This path is used for all the file I/O of Apply. If specified it is carried to the directory field on *Run Specifications*, if not specified, the directory must be filled on *Run Specifications.* Apply generates a log file like Capture does, in the directory of *apply_path*.

The password file is used by Apply to find the userid and password pairs to access the source and target servers. If the source or target servers are remote to apply control server and if configuration parameter of the source or target server indicates that authentication is required at the SERVER, a valid userid and password for that server must be present in the password file.

The password file is named *asnpwd.aut* by default. This password file is created and populated with **asnpwd** command. This command is explained on Section , “Maintaining the password file” on page cccxix. This command should be issued at the apply control server. Apply searches this file on the directory given in the *apply_path.* If you used a different name than the default, you should indicate the name with *pwdfile* parameter. You can also place this file to a sub-directory of *apply_path*. If you placed your password file under a sub-directory *apply_path*, you must indicate your sub-directory as well as the password file with *pwdfile* parameter.

Important: Password file which holds the passwords of userid’s from various platforms is *encrypted*.

In our example, we created and inserted the userid and password for *SAMPLE* database using the following commands at *STHELENS* before starting Apply. Our password file is created in the *pwd* directory under *apply_path*.

```
cd /home/cayci/pwd/
asnpwd init using password.aix
asnpwd add alias sample id db2drs3 password db2drs3 using password.aix
```

Note that init is issued only once for creating the password file but add should be issued for every database Apply connects and which requires authentication. In our case only the source database is remote.

Replication Center generates the following command and submits to *STHELENS* as the consequence of our selections at the *Start Apply* window.

```
asnapply CONTROL_SERVER=AIXSAMP APPLY_QUAL=APY1 PWDFILE=pwd/password.aix
```

One way to start Apply is to define it as a Windows service. This method is possible only when Apply is on the Window 2000 and Windows NT operating systems. If you mark the check-box at the bottom of the window on Figure 6-6 on page cclxxiv, Replication Center prepares the command to create and start the service based on the parameters provided on the *Start Apply* window. The command prepared by the Replication Center needs to be modified. Necessary modifications are explained at Section , “Starting Capture and Apply as a Windows service” on page ccciv. The command for creating the service(**asnscrt**) and the command for dropping the service(**asnsdrop**) are also explained on that section.

Stopping Capture and Apply

Stopping Capture or Apply from the Replication Center is straight forward. Refer to the beginning of this section to find how to stop Capture and Apply from Replication Center.

Attention: If you receive `ASN0506E, “The program could not attach to the replication communications message queue”` error message at the stop command, the most common reason is that the program (Capture or Apply) is already stopped or never started.

Starting Apply on the iSeries

The Capture program runs were the source table resides and the Apply programs could run on any target servers. For example, if the Apply and Capture was running on the same server and the target tables was on another server, then this configuration is in a push mode. When the Apply is running on a target server that is connecting to the source sever, it is a pull mode configuration. The

performance is more efficient in a pull mode configuration. Refer to Chapter 10, “Performance” on page cdxli on performance.

Xref chapter 4 and 10

Remote journal configuration is another option to run the Capture and Apply at the target server. See Chapter 4 replication sources@@@@@. and Chapter 1.

In this section we will consider pull mode configuration and make references to remote journal configurations.

Before starting the Apply program, the following procedures need to be accomplished with a valid user profile on the source and target server:

Connectivity configuration

Before the Apply program can connect to the source server. The DRDA communication connection must be configured:

Example 6-1 Configure connection

```
At the source server enter command: WRKRDBDIRE and note the relational database
(RDB) name for the *local remote location
The follow display is the WRKRDBDIRE screen, indicating DB2400D is the RDB the
target server will connect to:

Work with Relational Database Directory Entries
Position to  . . . . . .
Type options, press Enter.
   1=Add   2=Change   4=Remove   5=Display details   6=Print details

         Relational          Remote
 Option  Database            Location                  Text
         NST105              9.9.74.999
         MYNST103            9.30.74.43
         NST103              9.112.26.3
         DB2400D             *LOCAL

At the target server enter the ADDRDBDIRE command, F4 key to display the prompt
screen. Enter the following values to create the DRDA connection to the source
server:

Add RDB Directory Entry (ADDRDBDIRE)
Type choices, press Enter.
Relational database  . . . . . RDB          > DB2400D
Remote location:                RMTLOCNAME
  Name or address  . . . . . .              > 999.99.999.999
      Need to get IP address or Host name of the source sever
Type . . . . . .              > *IP
Text . . . . . . . . . . . . . TEXT           *BLANK

Port number or service program   PORT          *DRDA
```

```
Remote authentication method:    RMTAUTMTH
  Preferred method . . . . . .                *ENCRYPTED
  Allow lower authentication . .               *ALWLOWER
```

- If you want to connect to a source sever on Windows platform change the port parameter to 50000.
- Run SQL connect statement to test the connection to the source server database:
- CONNECT TO DB2400D USER *UserID* USING *Password*
- If you receive an authorization error when connecting to the source server and the userID and password is correct. At the target server enter CHGDDMTCPA. If password required parameter is: *YES. Then run the following command:
- ADDSVRAUTE USRPRF(*TargetUserprofile*) SERVER(*SourceRDB*) USRID(*SourceUserID*) PASSWORD*(SourcePassword*)

Create SQL packages

- You must now create SQL packages so that the Apply program can communicate with the source server. The packages are created in the ASN library. On other platforms they are created in the ASN schema. Enter the following command at the target server:

Example 6-2 Create Apply SQL package

```
CRTSQLPKG PGM(QDP4/QZSNAPV2) RDB(Source_server)
```

For remote journal configuration run the following command at the target server were the Capture and Apply program is running. The source server is were the actual source table reside:

```
CRTSQLKG PGM(QDP4/QZSNSQLF)RDB(Source table sever)
OBJTYPE(*SRVPGM)
```

- After creating the SQL packages you must grant *EXCECUTE privileges to these objects. Enter the following command at the source server:

Example 6-3 Grant authority to SQL packages

```
GRTOBJAUT OBJ(ASN/package_name )OBJTYPE(*SQLPKGUSER(subscriber
UserID )AUT(*OBJOPR *EXECUTE
```

If Data Propagator product exist at the Source server use the follow command:

GRTDPRAUT for details see chapter 18 in *DB2 Universe Database Replication Guide and Reference,* SC27-1121-00

Starting Apply

From the Replication Center when you select an apply qualifier from the Apply control server, the window on Figure 6-7 is displayed to start the Apply program at the target server:

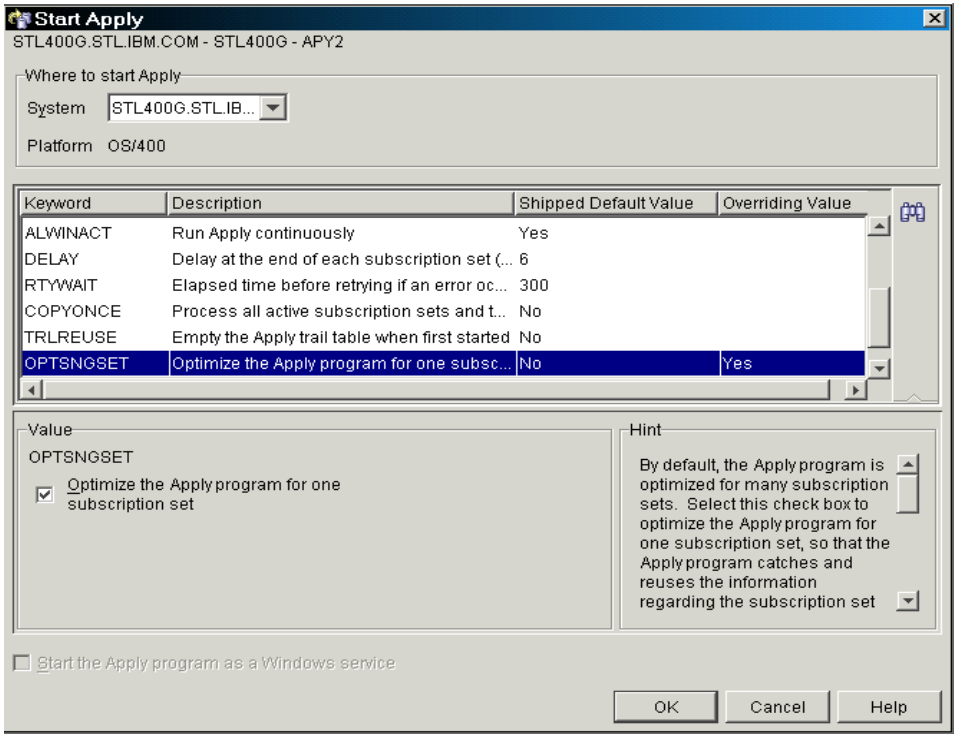


Figure 6-7 Start iSeries Apply.

Select the target sever from the **System** pull down list. The apply qualifier *APY1* for this Apply example was selected from the previous screen as described in the beginning of this section.

Select the Keyword and enter the value, which is displayed in the Overriding Value part of the display. For example, the OPTSNGSET parameter was overridden to YES. For details about these parameters refer to chapter 18 under STRDPRAPY command, in the *DB2 Universe Database Replication Guide and Reference,* SC27-1121-00.

The following STRDPRAPY command is generated showing the override of the OPTSNGSET parameter, when you click **OK** button, executed at the target server:

Example 6-4 STRDPRAPY command generated from the Replication Center

```
strdprapy CTLSVR(STL400G) APYQUAL(APY2) OPTSNGSET(Y)
```

Stopping and Capture and Apply Program

- To stop Capture Select **Stop Capture** from the menu when right click at the Capture control server display. The following screen is displayed:

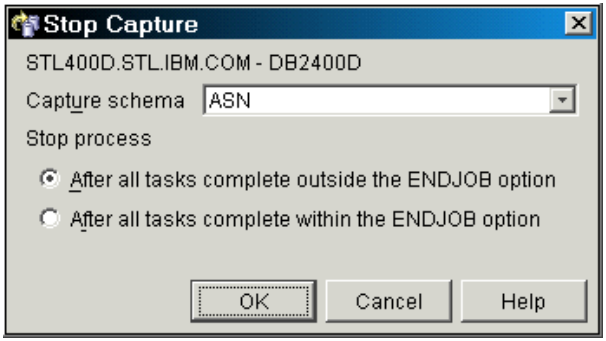


Figure 6-8 Stop Capture

  - Required to select the capture schema, and the option to end controlled by selecting **After all task complete outside the ENDJOB option**. Or selecting the other option to end Capture immediately
- Select the **Stop Apply** from the menu when you right click on the apply qualifier list from the apply control servers. Will display a window to select Apply control server with the option to end controlled or immediately.

Querying the Status of Capture and Apply

The status Capture and Apply can be queried from the Replication Center. In order to query the status of Capture, select **Operations** then **Capture Control Servers**. On the content pane, select and right-click the database you want to query and from the option list select **Query Status**. The Query Status screen seen on Figure 6-9 displays. Capture schema is chosen from the pull-down menu. In order to obtain the status, click the arrow. The status information refreshes at the time intervals you have chosen.

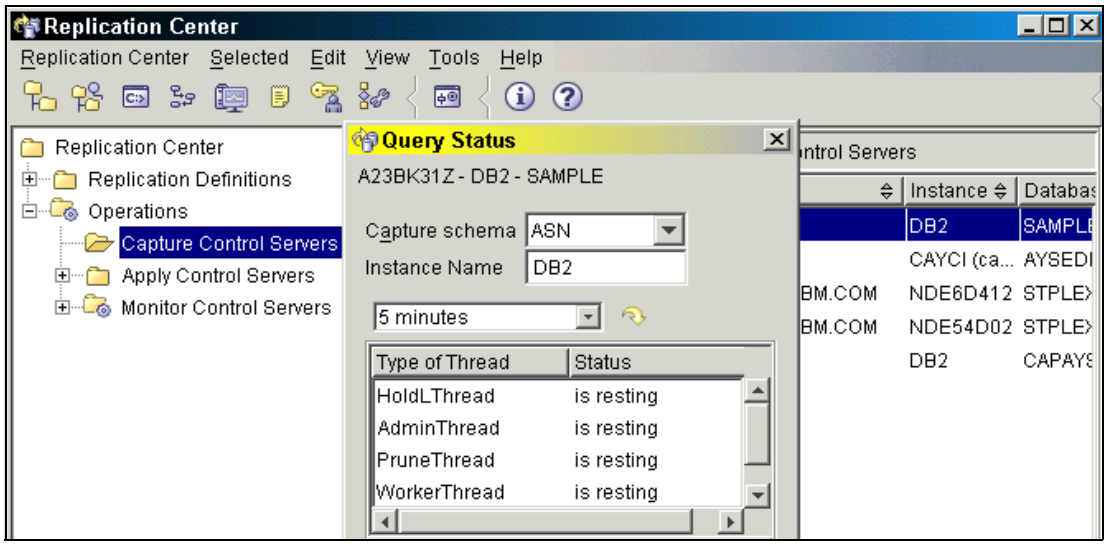


Figure 6-9 Query status from Replication Center

If Replication Center can attach to the communications message queue of the Capture, the threads and the status of the Capture threads are listed. Capture is a multi threaded application. Type of threads and their functions are as follows:

1. HOLDL thread: In order to ensure that only one Capture is capturing for this schema, locks the IBMSNAP_CAPENQ table of that schema. This thread is also called serialization thread.
2. ADMIN thread: This thread externalizes the statistics to IBMSNAP_CAPMON and writes messages to the Capture log and IBMSNAP_CAPTRACE.
3. PRUNE thread: This thread prunes the Change Data(CD), IBMSNAP_UOW, IBMSNAP_SIGNAL, IBMSNAP_CAPTRACE and IBMSNAP_CAPMON tables. If *auto_prune* is not set, then this thread stays in the resting state until prune command is issued. If Capture is run with auto-pruning, then prune thread rests until *prune_interval*.
4. WORKER thread: This thread reads the DB2 logs, captures changes into memory. Worker thread inserts changes to CD table and updates the commits to the IBMSNAP_UOW when the application commits.

Query status displays the four of the Capture threads. One more thread is used by Capture. This thread establishes the environment and starts the other threads.

The **is resting** and **working** states are indication of normal operation of Capture.

The status of Apply can also be queried in a similar way. Expand **Apply Control Servers** under **Operations**. Expand your apply control server and select **Apply Qualifiers**. On the content pane, select and right-click the qualifier you want to query and from the option list select **Query Status**. A *Query Status* screen similar to the *Query Status* screen of Apply displays. Two threads and states of Apply threads are being displayed.

1. HOLDL thread: This thread is for serialization. It stays in resting state all the time and prevents multiple applies to be started for the same apply qualifier for an apply control server.
2. WORKER thread: As evident from its name, this thread does the apply. Worker thread reads the CD table and UOW from the capture control server, updates the control tables of both capture control server and apply control server and updates the targets at the target server.

There is also an administrative thread which is not displayed on this screen but can be detected by issuing the command from the command line.

It is also possible to display the threads with DB2 or operating system commands. See Section , “Manage processes and threads” on page cccxx for the commands available on various platforms.

Query status may return an error on attaching the communications message queue for both Capture and Apply. One reason for not able to attach the communication message queue is that the program (Capture or Apply) is not started or stopped due to an error.

In a data sharing environment (z/OS) and if you start Capture with the DB2 data sharing group name, you must issue the status or trace command using the same group name. If you use the subsystem name instead of the group name you will receive the same error.

Querying Capture and Apply status on the iSeries

After starting the capture program from the Replication Center the following step will describe checking the status.

Checking Capture program status

The Capture program runs in batch within QZSNDPR sub system. On the command line enter: WRKSBSJOB QZSNDPR at the source server, to display all the Capture jobs running in that subsystem. If your running Apply on the same server you will also see the Apply programs, see Figure 6-13 on page cclxxxv.

```
 Work with Subsystem Jobs                    STL400D
                                                          09/11/02
15:37:54
 Subsystem  . . . . . . . . . . :   QZSNDPR

 Type options, press Enter.
   2=Change   3=Hold   4=End   5=Work with   6=Release   7=Display message
   8=Work with spooled files   13=Disconnect

 Opt  Job         User        Type     -----Status-----  Function
      COLINSRC    HCOLIN      BATCH    MSGW              PGM-QZSNCV81
      DPRJRN      HCOLIN      BATCH    ACTIVE            PGM-QZSNCV82
      LETAQ       RCTEST02    BATCH    ACTIVE            PGM-QZSNAPV2
      NST108RC    REPLSVT1    BATCH    MSGW              PGM-QZSNCV81
      PROMOTED    RCTEST01    BATCH    MSGW              PGM-QZSNCV81

 Parameters or command
 ===>
 F3=Exit      F4=Prompt    F5=Refresh    F9=Retrieve    F11=Display schedule
data
 F12=Cancel   F17=Top      F18=Bottom
```

Figure 6-10 WRKSBSJOB - Work with subsystem display for Capture

- The Capture program we started in Figure 6-3 on page cclxxi is shown on this display as COLINSRC job. This is the Capture controlling job that controls the journal job and pruning process. It will be the only Capture program displayed for each Capture Schema, if it's the first time starting capture for a group of registered source tables.
- When the full refresh process has successfully finish from the apply program, another Capture program starts up. This program is the Capture journal program, there's one program for each journal, for each capture schema. See, Figure 6-10, which shows *DPRJRN* job as the Capture journal job. The job name is represented by the actual journals name. The MSGW and ACTIVE status on this display indicate these jobs are working okay.
- The job log for both capture jobs will show details information if it's successful or message for any errors that occurred. From Figure 6-10. Enter option 5 by controlling job, to display the *Work with Job screen*. Select option 10 to *Display job log*, then F10 to display detail job log. The following example shows the job log when the Capture program starts successfully:

```
 System:   STL400D
Job . . :   COLINSRC      User . . :   HCOLIN       Number . . . :   028097

 >> CALL PGM(QDP4/QZSNCV81) PARM('COLINSRC  ' 'Y *LIBL/QZSNDPR 120 1
COLINSRC/
    DPRJRN')
    File IBMSN00021 in library COLINSRC changed.
    Capture process is starting with RESTART(*YES).
    Capture process has started.
    Capture has started clean up.
    Capture has completed clean up.
```

Figure 6-11 Successful Capture program job log for controlling job

- Enter option 5 on the WRKSMBJOB screen, see Figure 6-10 by the Capture journal job, select 10 to *Display job log,* then F10 to display detail job log. The following example shows the job log when the Capture journal job starts successfully:

```
 Display All Messages
                                                                System:
STL400D
 Job . . :   DPRJRN        User . . :   HCOLIN        Number . . . :
028098

  >> CALL PGM(QZSNCV82) PARM('COLINSRC   DPRJRN     0 1 -1
2002-09-11-15.34.23.
     205000 15 0 COLINSRC   ')
     Block mode started by the RCVJRNE exit program.
```

Figure 6-12 Successful Capture job log for the journal job

- If you can't find either the Capture or the journal program in the WRKSBMJOB display, that means an error occurred in the Capture programs and it has cancelled. Therefore, you need check the job log associated to these programs in QEZJOBLOG OUTQ to determine the problem. Use either WRKOUTQ OUTQ(QEZJOBLOG) or *WRKSPLF SELECT(YourUserID)*

Checking iSeries Apply program status

The Appy program runs also in batch within QZSNDPR sub system. There are a few method to check the status of your Apply program, we will cover 2 basic methods. Checking the job log on the iSeries and using the Replication Center Apply reports.

iSeries Apply program job logs

To check job logs, enter on the iSeries command line: WRKSBSJOB QZSNDPR at the target server, to display all the Apply jobs running in that subsystem.

```
 Work with Subsystem Jobs                    STL400G
                                                              09/11/02
18:58:44
 Subsystem  . . . . . . . . . . :   QZSNDPR

 Type options, press Enter.
   2=Change   3=Hold   4=End   5=Work with   6=Release   7=Display message
   8=Work with spooled files   13=Disconnect


 Opt  Job         User       Type     -----Status-----  Function
      APY2        HCOLIN     BATCH    ACTIVE            PGM-QZSNAPV2
      CMDTST      TKTONG     BATCH    ACTIVE            PGM-QZSNAPV2

Bottom
 Parameters or command
 ===>
 F3=Exit      F4=Prompt    F5=Refresh    F9=Retrieve    F11=Display schedule
data
 F12=Cancel   F17=Top      F18=Bottom
```

Figure 6-13 WRKSBSJOB - Work with sub system for Apply

- The Apply program we started in Figure 6-7 on page cclxxix is shown on this display as APY2 job, which is the name of the Apply qualifier. The status is active, but you need to view the job log, to check if any error occurred during the apply cycle. The procedure to view the job log is described in the preceding heading, see “Checking Capture program status” on page cclxxxii.
- The following is a sample of the Apply program job log to check for any message:

```
 Display All Messages
                                                                System:
STL400G
 Job . . :   APY2          User . . :   HCOLIN        Number . . . :
235188

  >> CALL PGM(QDP4/QZSNAPV2) PARM('APY2              ' '*LOCAL            '
'N'
      '*NONE               ' '*NONE               ' 'Y' 'Y' '    '
'00000300' '
     N' 'N' 'N')
     Database connection started over TCP/IP or a local socket.
     Printer device PRT01 not found. Output queue changed to QPRINT in
library
       QGPL.
     Printer device PRT01 not found. Output queue changed to QPRINT in
library
       QGPL.
     File ASNAS000 created in library QTEMP.
     Member ASNAS000 added to file ASNAS000 in QTEMP.
     Member TGPROJECT file TGPROJECT in COLINTAR cleared.
     Apply will be inactive for 1 minutes and 53 seconds.
     Apply will be inactive for 1 minutes and 57 seconds.
     Apply will be inactive for 1 minutes and 57 seconds.
Press Enter to continue.
F3=Exit   F5=Refresh   F12=Cancel   F17=Top   F18=Bottom
```

Figure 6-14 Successfull Apply program joblog

- Figure 6-14 shows a typical job log when the Apply program has started successfully. The numbered arrows indicates the following:
  1 - A successfull connection to the remote source server.
  2 - Create spill file in QTEMP.
  3 - Apply process 3 cycles and went to sleep approximately for 2 minutes. As specified in the subscription set.

If the Apply encountered a problem, the following job log is generated:

Need to add unsuccessfull log>>>>>>>>>>>>>>>>>>>>>>>>>>>>

complete this by colin

Figure 6-15 Unsuccessfull Apply program joblog

Apply reports from the Replication Center

On the Apply qualifier display to start and stop the Apply program, that same menu list contains another option to **Show Apply Report**. When you select that option window seen on Figure 6-16 is displayed:

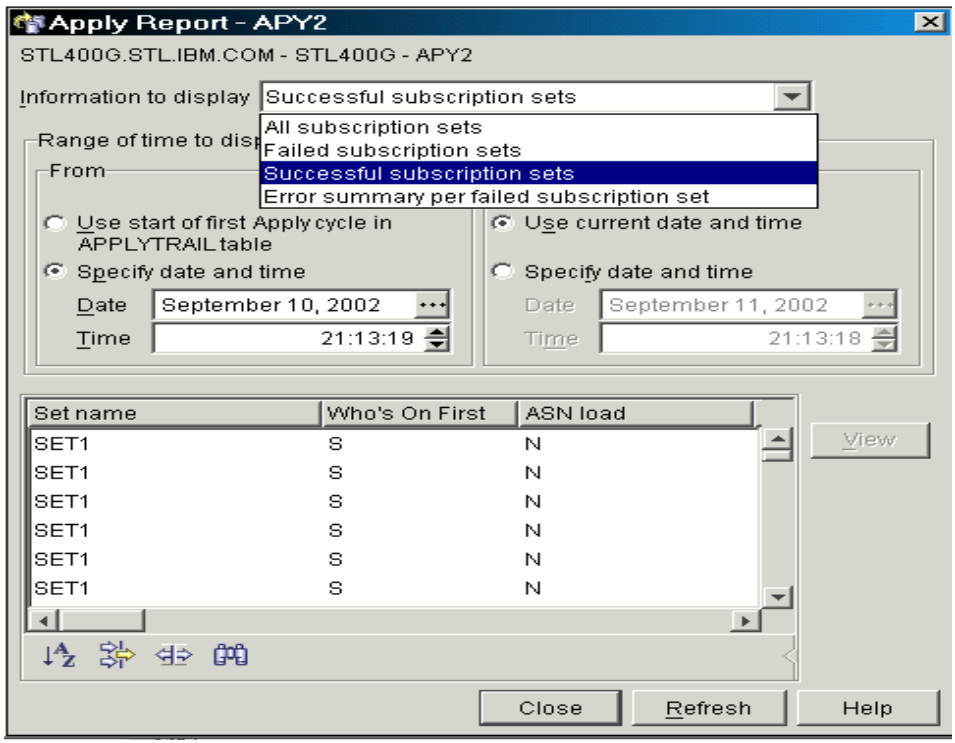


Figure 6-16 Apply report

This screen can help you to determine the status of your Apply program by selecting from the **Information to display** list and reviewing the results of your selection, after you click the **Refresh** button. This information is retrieved from the IBMSNAP_APPPLYTRAIL table.

6.1.2 Basic operations from the command prompt

This section does not cover basic operations for DB2 UDB for iSeries. Refer to Section 6.1.5, “Considerations for DB2 UDB for iSeries” on page cccxvi for DB2 UDB for iSeries operations.

It is also possible to start, stop or query the Capture and Apply from operating system command prompt or shell prompt. On this section the following commands with the functionality stated next to it, are explained:

asncap: start Capture
asnccmd: query and stop Capture
asnapply: start Apply
asnacmd: query and stop Apply

On Windows operating systems commands are run from **Command Window** which is under **IBM DB2 -> Command Line Tools** or you can issue the commands from any MS-DOS prompt.

On UNIX operating systems, commands are run from the shell prompt.

On z/OS, IBM DB2 DataPropagator V8.1 provides the data replication feature. For requirements, environment and general information on IBM DB2 DataPropagator V8.1 refer to Section , “General information on DB2 DataPropagator” on page cccvi. Capture and Apply run as UNIX System Services (USS) programs on z/OS. You can run the commands on z/OS either of the ways:

- After logging on to TSO, call **omvs** command to start USS shell command line.
- Use remote login program (**rlogin)** to access USS shell on z/OS from an UNIX or Windows environment.

At the USS profile, Data Propagator load library, DB2 load library and C run time libraries must be stated on the STEPLIB and data replication bin directory must be listed on the PATH. If you are using SDSNEXIT library for authorization exits, then it must also be listed at STEPLIB. LANG is optional. It is set if different code page is required. A sample profile is shown on Example 6-5.

Example 6-5 Sample USS profile Data Propagator

```
DPRLOAD=DPROPR.V810.LOAD
DB2LOAD=DSN.SDSNLOAD
CRUNLOAD=CEE.SCEERUN:CBC.SCLBLDLL
export STEPLIB=$(DPRLOAD):$(CRUNLOAD):$(DB2LOAD)
export PATH=/usr/lpp/db2repl_08_01/bin:$(PATH)
export LANG=en_US
```

You can also run the commands by a JCL as a batch job or started task on z/OS. Running commands using JCL is described on Section , “Using JCL to operate Capture and Apply” on page cccx.

On the examples in this section, commands are issued on one of the possible platforms chosen randomly but they can be run from every platform listed above.

From the command line, commands are called with the start-up parameters. Start-up parameters are not positional. They are given with a keyword. Commands have mandatory parameters, and optional parameters. Optional parameters are specified only to override the defaults for this run of Capture of Apply. In order to examine the complete syntax of the commands, refer to Chapter 17, “System Commands for replication (UNIX, Windows, z/OS)”, in *IBM DB2 Universal Database Replication Guide and Reference,* SC27-1121-00

asncap and asnccmd

The sample command on Figure 6-17 starts capture for *sample* capture control server.If the capture control server is DB2 UDB for Windows and UNIX, the *capture_server* is the database alias. You can find out the database alias of a database with **`list database directory`** command. Capture uses this information to connect to the capture control server. If the capture control server is DB2 UDB for z/OS, *capture_server* is the subsystem name.

If DB2DBDFT variable is not set, *Capture_server* should be specified on DB2 UDB for UNIX and Windows. If you have set the capture control server in DB2DBDFT then you can call **`asncap`** without parameters. The default for DB2 UDB for z/OS is DSN. You must specify it, if you have a different subsystem name.

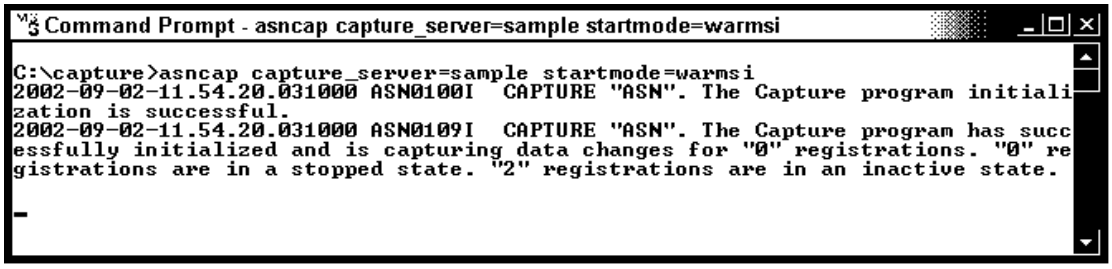


Figure 6-17 Start Capture from the command prompt

Capture_schema is also necessary for Capture. The default for *capture_schema* is ASN. You may be using different schemas due to exploitation of multiple schemas for your replication environment.

Startmode is an optional parameter. It is set on the example to change it to WARMSI from the default which is WARMNS. Capture warm start WARMNS and switch to cold start only if this is the first run of Capture for this capture control server with this schema.

The informational message issued by Capture on the example indicates the completion of re-initialization during warm start. Stopped registrations are the registered sources with state 'S' in the IBMSNAP_REGISTER table. They may be stopped by the Capture due to registration errors or set manually by he administrator during maintenance. Capture will not capture changes for a stopped registration until its state is set to 'I' manually. Capture will capture changes for inactive registrations (STATE='I').

If the *capture_path* is not given as in the example, it is defaulted to the directory where the **asncap** is issued. The naming convention for the log is as follows:

<instance name>.<capture_server>.<capture_schema>.CAP.log

The capture log generated by this invocation of Capture is seen on Example 6-6. All parameters values assigned for this instance of Capture and the method they are acquired appears in the log. *Capture_server* and *startmode* are obtained from the command line as start-up parameters, *capture_schema* and *capture_path* are assigned by system defaults and all other values are from the IBMSNAP_CAPPARMS table.

During start-up, the keys of the IPC queues created are listed on the log with message number ASN8008D.

Example 6-6 Capture log

```
2002-09-02-11.54.17.327000 <setEnvDprRIB> ASN8003D  "Capture" : "ASN" : Program "capture 8.1.0"
is starting.
2002-09-02-11.54.19.820000 <asnParmClass::printParms> ASN0529I  "Capture" : "ASN" : The value
of "CAPTURE_SERVER" was set to "SAMPLE" at startup by the following method: "COMMANDLINE".
2002-09-02-11.54.19.830000 <asnParmClass::printParms> ASN0529I  "Capture" : "ASN" : The value
of "CAPTURE_SCHEMA" was set to "ASN" at startup by the following method: "DEFAULT".
2002-09-02-11.54.19.830000 <asnParmClass::printParms> ASN0529I  "Capture" : "ASN" : The value
of "LOGREUSE" was set to "N" at startup by the following method: "IBMSNAP_CAPPARMS".
2002-09-02-11.54.19.830000 <asnParmClass::printParms> ASN0529I  "Capture" : "ASN" : The value
of "LOGSTDOUT" was set to "N" at startup by the following method: "IBMSNAP_CAPPARMS".
2002-09-02-11.54.19.830000 <asnParmClass::printParms> ASN0529I  "Capture" : "ASN" : The value
of "TERM" was set to "Y" at startup by the following method: "IBMSNAP_CAPPARMS".
2002-09-02-11.54.19.830000 <asnParmClass::printParms> ASN0529I  "Capture" : "ASN" : The value
of "CAPTURE_PATH" was set to "C:\capture\" at startup by the following method: "DEFAULT".
2002-09-02-11.54.19.830000 <asnParmClass::printParms> ASN0529I  "Capture" : "ASN" : The value
of "AUTOSTOP" was set to "N" at startup by the following method: "IBMSNAP_CAPPARMS".
2002-09-02-11.54.19.830000 <asnParmClass::printParms> ASN0529I  "Capture" : "ASN" : The value
of "STARTMODE" was set to "WARMSI" at startup by the following method: "COMMANDLINE".
```

```
2002-09-02-11.54.19.830000 <asnParmClass::printParms> ASN0529I  "Capture" : "ASN" : The value
of "RETENTION_LIMIT" was set to "10080" at startup by the following method: "IBMSNAP_CAPPARMS".
2002-09-02-11.54.19.830000 <asnParmClass::printParms> ASN0529I  "Capture" : "ASN" : The value
of "LAG_LIMIT" was set to "10080" at startup by the following method: "IBMSNAP_CAPPARMS".
2002-09-02-11.54.19.830000 <asnParmClass::printParms> ASN0529I  "Capture" : "ASN" : The value
of "COMMIT_INTERVAL" was set to "30" at startup by the following method: "IBMSNAP_CAPPARMS".
2002-09-02-11.54.19.830000 <asnParmClass::printParms> ASN0529I  "Capture" : "ASN" : The value
of "PRUNE_INTERVAL" was set to "300" at startup by the following method: "IBMSNAP_CAPPARMS".
2002-09-02-11.54.19.830000 <asnParmClass::printParms> ASN0529I  "Capture" : "ASN" : The value
of "SLEEP_INTERVAL" was set to "5" at startup by the following method: "IBMSNAP_CAPPARMS".
2002-09-02-11.54.19.830000 <asnParmClass::printParms> ASN0529I  "Capture" : "ASN" : The value
of "AUTOPRUNE" was set to "Y" at startup by the following method: "IBMSNAP_CAPPARMS".
2002-09-02-11.54.19.830000 <asnParmClass::printParms> ASN0529I  "Capture" : "ASN" : The value
of "TRACE_LIMIT" was set to "10080" at startup by the following method: "IBMSNAP_CAPPARMS".
2002-09-02-11.54.19.830000 <asnParmClass::printParms> ASN0529I  "Capture" : "ASN" : The value
of "MONITOR_LIMIT" was set to "10080" at startup by the following method: "IBMSNAP_CAPPARMS".
2002-09-02-11.54.19.830000 <asnParmClass::printParms> ASN0529I  "Capture" : "ASN" : The value
of "MONITOR_INTERVAL" was set to "300" at startup by the following method: "IBMSNAP_CAPPARMS".
2002-09-02-11.54.19.830000 <asnParmClass::printParms> ASN0529I  "Capture" : "ASN" : The value
of "MEMORY_LIMIT" was set to "32" at startup by the following method: "IBMSNAP_CAPPARMS".
2002-09-02-11.54.19.830000 <Asnenv:setEnvIpcQRcvHdl> ASN8008D  "Capture" : "ASN" : "Created"
IPC queue with key(s) "(OSSEIPC0tempDB2.SAMPLE.ASN.CAP.IPC, OSSEIPC1tempDB2.SAMPLE.ASN.CAP.IPC,
OSSEIPC2tempDB2.SAMPLE.ASN.CAP.IPC)".
2002-09-02-11.54.20.031000 <CWorkerMain> ASN0100I  CAPTURE "ASN". The Capture program
initialization is successful.
2002-09-02-11.54.20.031000 <CWorkerMain> ASN0109I  CAPTURE "ASN". The Capture program has
successfully initialized and is capturing data changes for "0" registrations. "0" registrations
are in a stopped state. "2" registrations are in an inactive state.
2002-09-02-11.59.20.332000 <PruneMain> ASN0111I  CAPTURE "ASN". The pruning cycle started at
"Mon Sep 02 11:59:20 2002".
2002-09-02-11.59.20.362000 <PruneMain> ASN0112I  CAPTURE "ASN". The pruning cycle ended at "Mon
Sep 02 11:59:20 2002".
```

The figure on Figure 6-18 shows the **asncap** and **asnccmd** commands issued from uss on z/OS. Note that DB2 subsystem name is specified for *capture_server*. Capture runs as a process under uss on z/OS and a process id (pid=200 for this example) is assigned. A capture log is created on the directory where command is issued.

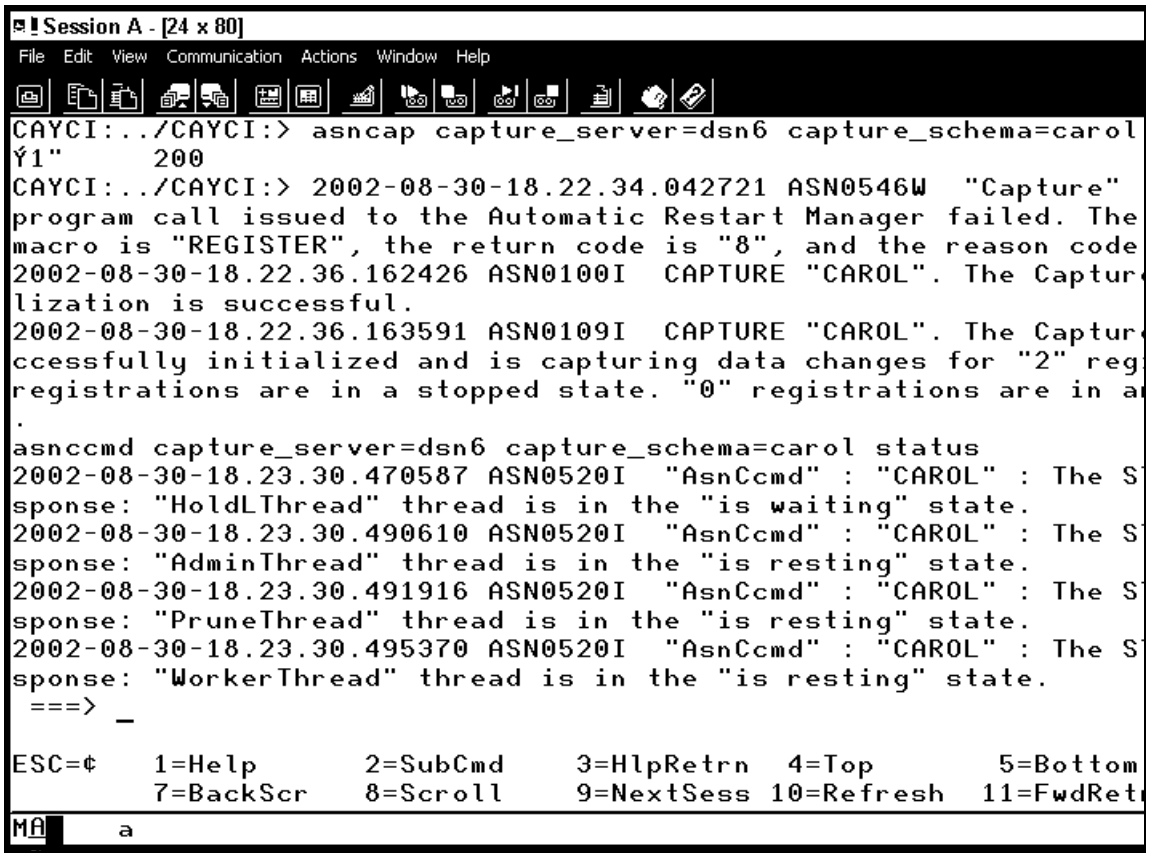


Figure 6-18 Start Capture and query status from USS shell

The status of Capture is queried using **asnccmd** command with status option. If Capture is running, the status of each thread is displayed as in Figure 6-18. Refer to Section 6.1.1, “Basic operations from the Replication Center” on page cclxvi for the functions of the threads. For this example, the *capture_server* on the command, is the subsystem name because the capture control server is DB2 UDB for z/OS. If the capture control server were DB2 UDB for UNIX or Windows, it must be the one of the database alias listed in the database directory.

If capture is not running, you will receive `ASN0506E, “The program could not attach to the replication communications message queue”` error message.

Three possible ways of stopping Capture are:

- You can stop it with **asnccmd** command as in the example below.

```
asnccmd capture_server=sample stop
```

- You can cancel it with CTRL-C.
- You can stop it manually by inserting a row into the IBMSNAP_SIGNAL table using the SQL below:

```
INSERT INTO schema.IBMSNAP_SIGNAL (SIGNAL_TYPE, SIGNAL_SUBTYPE,
SIGNAL_STATE) VALUES('CMD ','STOP ','P')
```

asnapply and asnacmd

When starting Apply from the command line, the apply *control_server* and the *apply_quali*fier are mandatory. If Apply is on DB2 UDB for UNIX and Windows and either the source or the target server is a remote instance which requires authentication or a DB2 subsystems on z/OS, userid password combination for that server is also required to successfully connect to this system.

The following example shows starting and querying Apply for the replication scenario on Figure 6-5 on page cclxxiii. On the shell prompt of AIX seen on Figure 6-19, following are performed:

- A password file is created by **asnpwd**.
- Userid, password pair is added to the password file for SAMPLE and AIXSAMP databases.
- A row is inserted to the IBMSNAP_APPPARMS to override the shipped value for the *delay* parameter for APY1 apply qualifier for all Apply cycles.
- Apply is run as a background process with **asnapply**.
- Status of Apply is checked with **asnacmd**.

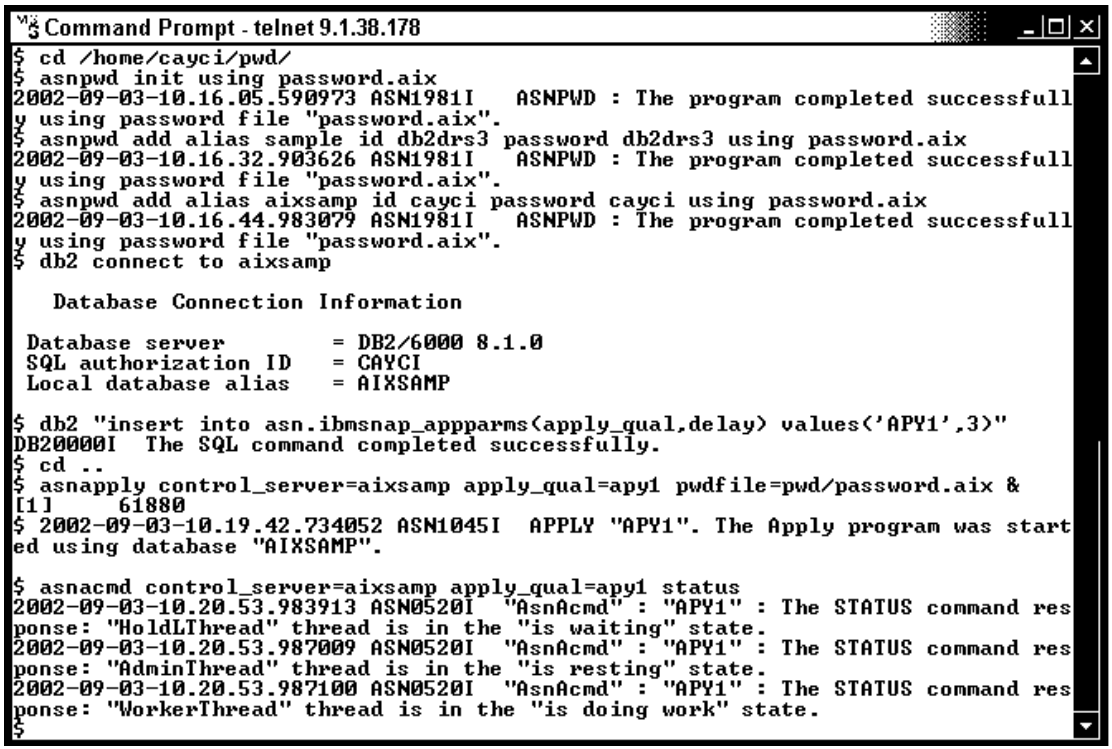


Figure 6-19 Start Apply and query status from the command prompt

The *apply_path* is not given on the **asnapply** command. Therefore it is defaulted to the directory where the command is issued. The naming convention for the log is as follows:

<instance name>.<apply_server>.<apply_qual>.APP.log

The log file is seen on Example 6-7. The *control_server*, *apply_qual* and *pwdfile* are start-up parameters. The password file is located under the *pwd* sub-directory of *apply_path* and it is named *password.aix*. All other values are received from IBMSNAP_APPPARMS. Although we have only inserted the *delay* parameter, values for other columns are filled by the defaults (which are the shipped defaults) defined on the IBMSNAP_APPPARMS. For this reason, all the parameters other than overridden during start up are marked as they are obtained from the IBMSNAP_APPPARMS.

Apply log is also used by Apply for recording errors encountered during Apply cycle.

Example 6-7 Apply log

```
2002-09-03-10.19.40.468340 <setEnvDprRIB> ASN8003D  "Apply" : "" : Program "apply 8.1.0" is
starting.
2002-09-03-10.19.42.584883 <asnParmClass::printParms> ASN0529I  "Apply" : "APY1" : The value of
"CONTROL_SERVER" was set to "AIXSAMP" at startup by the following method: "COMMANDLINE".
2002-09-03-10.19.42.585683 <asnParmClass::printParms> ASN0529I  "Apply" : "APY1" : The value of
"APPLY_QUAL" was set to "APY1" at startup by the following method: "COMMANDLINE".
2002-09-03-10.19.42.585755 <asnParmClass::printParms> ASN0529I  "Apply" : "APY1" : The value of
"LOGREUSE" was set to "N" at startup by the following method: "IBMSNAP_APPPARMS".
2002-09-03-10.19.42.585825 <asnParmClass::printParms> ASN0529I  "Apply" : "APY1" : The value of
"LOGSTDOUT" was set to "N" at startup by the following method: "IBMSNAP_APPPARMS".
2002-09-03-10.19.42.585894 <asnParmClass::printParms> ASN0529I  "Apply" : "APY1" : The value of
"TERM" was set to "Y" at startup by the following method: "IBMSNAP_APPPARMS".
2002-09-03-10.19.42.585964 <asnParmClass::printParms> ASN0529I  "Apply" : "APY1" : The value of
"PWDFILE" was set to "pwd/password.aix" at startup by the following method: "COMMANDLINE".
2002-09-03-10.19.42.586038 <asnParmClass::printParms> ASN0529I  "Apply" : "APY1" : The value of
"APPLY_PATH" was set to "/home/cayci/" at startup by the following method: "DEFAULT".
2002-09-03-10.19.42.586110 <asnParmClass::printParms> ASN0529I  "Apply" : "APY1" : The value of
"NOTIFY" was set to "N" at startup by the following method: "IBMSNAP_APPPARMS".
2002-09-03-10.19.42.586181 <asnParmClass::printParms> ASN0529I  "Apply" : "APY1" : The value of
"SLEEP" was set to "Y" at startup by the following method: "IBMSNAP_APPPARMS".
2002-09-03-10.19.42.586272 <asnParmClass::printParms> ASN0529I  "Apply" : "APY1" : The value of
"ERRWAIT" was set to "300" at startup by the following method: "IBMSNAP_APPPARMS".
2002-09-03-10.19.42.586344 <asnParmClass::printParms> ASN0529I  "Apply" : "APY1" : The value of
"COPYONCE" was set to "N" at startup by the following method: "IBMSNAP_APPPARMS".
2002-09-03-10.19.42.586414 <asnParmClass::printParms> ASN0529I  "Apply" : "APY1" : The value of
"TRLREUSE" was set to "N" at startup by the following method: "IBMSNAP_APPPARMS".
2002-09-03-10.19.42.586489 <asnParmClass::printParms> ASN0529I  "Apply" : "APY1" : The value of
"SPILLFILE" was set to "DISK" at startup by the following method: "IBMSNAP_APPPARMS".
2002-09-03-10.19.42.586617 <asnParmClass::printParms> ASN0529I  "Apply" : "APY1" : The value of
"LOADXIT" was set to "N" at startup by the following method: "IBMSNAP_APPPARMS".
2002-09-03-10.19.42.586689 <asnParmClass::printParms> ASN0529I  "Apply" : "APY1" : The value of
"SQLERRCONTINUE" was set to "N" at startup by the following method: "IBMSNAP_APPPARMS".
2002-09-03-10.19.42.586762 <asnParmClass::printParms> ASN0529I  "Apply" : "APY1" : The value of
"OPT4ONE" was set to "N" at startup by the following method: "IBMSNAP_APPPARMS".
2002-09-03-10.19.42.586829 <asnParmClass::printParms> ASN0529I  "Apply" : "APY1" : The value of
"DELAY" was set to "3" at startup by the following method: "IBMSNAP_APPPARMS".
2002-09-03-10.19.42.586897 <asnParmClass::printParms> ASN0529I  "Apply" : "APY1" : The value of
"INAMSG" was set to "Y" at startup by the following method: "IBMSNAP_APPPARMS".
2002-09-03-10.19.42.726915 <Asnenv:setEnvIpcQRcvHdl> ASN8008D  "Apply" : "APY1": "Created" IPC
queue with key(s) "(0x30000073)".
2002-09-03-10.19.42.734052 <CPCIMPC(08/07)> ASN1045I  APPLY "APY1". The Apply program was
started using database "AIXSAMP".
2002-09-03-10.19.43.095065 <CPREST(01/00)> ASN1044I  APPLY "APY1". The Apply program will
become inactive for "5" minutes and "0" seconds.
2002-09-03-10.20.52.786209 <cmdsEngine> ASN8008D  "Apply" : "APY1" : "Attached to" IPC queue
with key(s) "(0x30000075)".
2002-09-03-10.20.52.786325 <cmdsEngine> ASN8008D  "Apply" : "APY1" : "Detached from" IPC queue
with key(s) "(0x30000075)".
```

The command used for querying the status of Apply is **asnacmd** with status option. If Apply is running, the state of each thread is displayed as in Figure 6-19. Refer to Section 6.1.1, “Basic operations from the Replication Center” on page cclxvi for the functions of these threads.

If the apply control server is DB2 UDB for z/OS then the parameter specification changes slightly:

- If the apply control server is DB2 UDB for UNIX and Windows, the database alias is specified as the *control_server* in the command. If the apply control server is DB2 UDB for z/OS, the *control_server* must be the location name of the subsystem.
- If apply control server is DB2 UDB for z/OS, the subsystem id is provided to Apply with *db2_subsystem* parameter.

Apply can run anywhere provided that connectivity to source, target and apply control servers are configured. Apply program needs the LOCATION and subsystem id name of the apply control server to connect and access the apply control tables. There is one set of apply control tables per DB2 subsystem. All information for each Apply qualifier are in the control tables. *Apply_qual* qualifies the necessary rows in the control tables for the subscription set processed by Apply.

The *pwdfile* parameter of Apply is not valid for DB2 UDB for z/OS.

Figure 6-20 Sample replication configuration for z/OS to z/OS

The subsystem D7DP is defined as both the Capture and Apply control servers and subsystem DSN7 is the target server on the sample replication configuration of Figure 6-20.

The following command can be used for starting Apply for the replication configuration on Figure 6-20.

```
asnapply control_server=d7dp db2_subsystem=d7dp apply_qual=apy1
```

control_server=d7dp, *d7dp* is the location name and **db2_subsystem=d7dp**, *d7dp* is the subsystem name where apply will run.As a coincidence location name and subsystem name are same for our sample. The location name must be assigned to *control_server* and subsystem name must be assigned to *db2_subsystem* in the commands.

Two possible ways of stopping Apply are:

- You can stop it with asnacmd command as in the example below.

  ```
  asnacmd control_server=aixsamp apply_qual=apy1 stop
  asnacmd control_server=d7dp db2_subsystem=d7dp apply_qual=apy1 stop
  ```

- You can cancel it with CTRL-C.

If you tried the stop or query status of Apply and if Apply is not running, you will receive `ASN0506E, "The program could not attach to the replication communications message queue"` error message.

Using commands to operate Capture and Apply on the iSeries

The following command are used to start and stop the Capture and Apply programs.

- STRDPRCAP - Start Capture program
- ENDDPRCAP- End Capture program
- STRDPRAPY- Start Apply program
- ENDDPRAPY - End Apply program

STRDPRCAP and ENDDPRCAP

Enter STRDPRCAP, then F4 and F11 key to display the prompt screens on Figure 6-21 to start the Capture program.

```
 Start DPR Capture (STRDPRCAP)

 Type choices, press Enter.

 Restart after end  . . . . . . . RESTART        *YES
 Job description  . . . . . . . . JOBD           QZSNDPR
   Library  . . . . . . . . . . .                  *LIBL
 Wait . . . . . . . . . . . . . . WAIT           120
 Clean up interval:               CLNUPITV
   Wait time  . . . . . . . . . .                *DFT
   Start clean up . . . . . . . .                *IMMED
 Capture control library  . . . . CAPCTLLIB    > COLINSRC
 Journal  . . . . . . . . . . . . JRN            *ALL
   Library  . . . . . . . . . . .
                          + for more values

 Trace limit  . . . . . . . . . . TRCLMT         *DFT
 Monitor limit  . . . . . . . . . MONLMT         *DFT
 Monitor interval . . . . . . . . MONITV         *DFT
 Memory limit . . . . . . . . . . MEMLMT         *DFT

More...
 F3=Exit   F4=Prompt   F5=Refresh   F12=Cancel   F13=How to use this display
```

Figure 6-21 STRDPRCAP - Start capture program. First screen

Press the page down key to display the last screen of prompts (Figure 6-22.).

```
 Start DPR Capture (STRDPRCAP)
Type choices, press Enter.

Retention period . . . . . . . . RETAIN         *DFT
Lag limit  . . . . . . . . . . . LAG            *DFT
Force frequency  . . . . . . . . FRCFRQ         *DFT
```

Figure 6-22 STRDPRCAP - Start Capture program. Last screen

The default values from the IBMSNAP_CAPPARMS are used as described in Figure 6-3 on page cclxxi, they overridden when you enter another value on this prompt screen. See, Section 6.2.1, “Change Capture parameters” on page cccxxvi

The command prompt screen on Figure 6-23 is the ENDDPRCAP

```
End DPR Capture (ENDDPRCAP)

Type choices, press Enter.

How to end . . . . . . . . . . . . OPTION         *CNTRLD
Capture control library  . . . . CAPCTLLIB       ASN
Reorganize control tables  . . . RGZCTLTBL       *NO
```

Figure 6-23 ENDDPRCAP - End Capture program.

The CAPCTLIB is new to version 8. It will end Capture for the Capture schema specified, which contains the Capture controls tables

The RGZCTLTBL is a new parameter in version 8, that performs a Reorganize Physical File Memebr (RGZPFM) over the CD and UOW tables opened during that instance of the Capture program. Therefore, if you started Capture for a specific journal, only those CD tables that are associated with the registered source table for that jounal are reorganized. If you start Capture using the default value to use all journals then all the CD tables are reorganized for all the registered source table within the Capture schema. Beware that when you select to reorganized your CD tables, it could delay the end Capture process. However, it could improve the performance of running Capture.

STRDPRAPY and ENDDPRAPY

The STRDPRAPY command when prompt will display the screen on Figure 6-24 to start Apply programs for a specific Apply qualifier.

```
 Start DPR Apply (STRDPRAPY)

 Type choices, press Enter.

 User . . . . . . . . . . . . . . . USER          *CURRENT
 Job description  . . . . . . . . . JOBD          QZSNDPR
   Library  . . . . . . . . . . . .                 *LIBL
 Apply qualifier  . . . . . . . . . APYQUAL      > APY2
 Control server . . . . . . . . . . CTLSVR        *LOCAL
 Trace  . . . . . . . . . . . . . . TRACE         *NONE
 Full refresh program . . . . . . . FULLREFPGM     *NONE
   Library  . . . . . . . . . . . .
 Subscription notify program  . . SUBNFYPGM      *NONE
   Library  . . . . . . . . . . . .
 Inactive message . . . . . . . . . INACTMSG       *YES
 Allow inactive state . . . . . . . ALWINACT       *YES
 Delay  . . . . . . . . . . . . . . DELAY          6
 Retry wait time  . . . . . . . . . RTYWAIT        300
 Copy Once  . . . . . . . . . . . . COPYONCE       *NO
 Trail Reuse  . . . . . . . . . . . TRLREUSE       *NO

More...
 F3=Exit   F4=Prompt   F5=Refresh   F12=Cancel   F13=How to use this display
 F24=More keys
```

Figure 6-24 STRDPRAPY - Start Apply program first screen

Press page down key display the next screen (Figure 6-25).

```
 Start DPR Apply (STRDPRAPY)

 Type choices, press Enter.

 Optimize single set  . . . . . . OPTSNGSET       *NO

Bottom
```

Figure 6-25 STRDPRAPY - Start Apply program last screen

ENDDPRAPY when prompt will display the screen on Figure 6-26 to end Apply for a specific Apply qualifier.

```
End DPR Apply (ENDDPRAPY)

Type choices, press Enter.

User . . . . . . . . . . . . . . . USER          *CURRENT
How to end . . . . . . . . . . . . OPTION        *CNTRLD
Apply qualifier  . . . . . . . . . APYQUAL       *USER
Control server . . . . . . . . . . CTLSVR        *LOCAL
```

Figure 6-26 ENDDPRAPY - End Apply program.

For details about all these parameters refer to chapter 18 under in the *DB2 Universe Database Replication Guide and Reference,* SC27-1121-00.

6.1.3 Considerations for DB2 UDB for UNIX and Windows

Before starting the Capture

Capture reads the changed data from DB2 logs. Since data replication is an asynchronous process, this data must stay in the log for sufficient amount of time until Capture reads it. If the capture control server is DB2 UDB for UNIX and Windows, logging type must be suitable for replication.

Circular and archival logging are two possible logging methods on DB2 for UNIX and Windows. At circular logging, log data is overridden whenever DB2 does not need that data for crash recovery. This method is not supported for replication because data in the log must be kept until Capture reads it. Circular logging is the default for databases created at DB2 UDB for UNIX and Windows. You must enable archival logging for databases used as Capture control servers on DB2 for UNIX and Window before starting the Capture. You can find how to detect the current logging type of the database and how to enable a database for replication on DB2 UDB for UNIX and Windows below.

How to detect logging type?

1. From the Replication Center: You can detect if the database is using archival logging thus enabled for replication by examining the legend displayed next to the capture control server *Name* at the *Operations*.

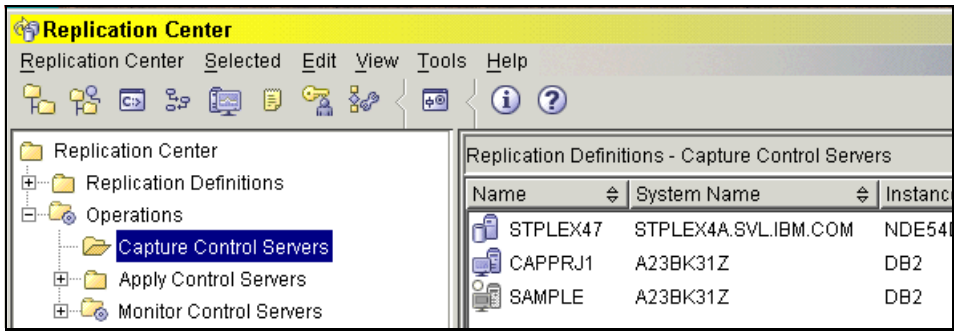


Figure 6-27 Enablement for replication

SAMPLE on Figure 6-27 is using circular logging, the legend next to it indicates that it is not enabled for replication. The other servers on the figure are enabled for replication.

2. From Command Window or Command Window: You can detect the logging of the database by issuing the following DB2 command.

   ```
   db2 get db cfg for sample
   ```

Circular logging is in effect for this database if both LOGRETAIN and USEREXIT parameters are OFF.

3. From Control Center: You can also use the Control Center to control your database logging type. After launching the Control Center, there are two places you can query this configuration parameter: follow the path **Systems -> Instances -> Databases,** right-click on your database and either select **Configure Database Logging** or **Configure Parameters**.

How to enable for replication?

This is accomplished in two steps: update the database configuration and backup the database. We describe below how to set the LOGRETAIN parameter. A database is also enabled for replication if USEREXIT parameter is ON. Refer to DB2 Information Center to learn how to implement user exit for database recovery.

1. From the Replication Center: You can change the logging type to archival and backup the database without leaving the Replication Center. Right-click the database to be enabled after selecting **Replication Definitions -> Capture Control servers** and from the pull-down menu select **Enable Database for Replication**. This action initiates a call to the DB2 command to update the LOGRETAIN value of the database to RECOVERY and launches *Backup Database*. A database automatically becomes backup pending after changing its log type to archival. An offline backup should be taken to switch it to normal

state. This is the only allowed option on Backup Database wizard which you launched by selecting **Enable Database for Replication**.

2. From Command Window or Command Center: The following are the commands you can issue for changing the logging type and backing up the database from CLP or Command center. The database name is *SAMPLE*. The backup file will be generated to a directory named *backup*. All the connected applications should disconnect before taking an offline backup.

```
cd \
mkdir backup
db2 terminate
db2 update database configuration for sample using logretain yes
db2 backup database sample to 'c:\backup'
```

   The file and directory structure used in the example is for Windows platforms, you have to alter them appropriately for UNIX platforms.

3. From Control Center: You can change the logging parameter from two different wizards at the Control Center like you did when you query this parameter. Follow **Systems -> Instances -> Databases**, right-click on your database and select **Configure Database Logging** or **Configure Parameters**. If you change the logging type to archival from **Configure Database Logging**, a backup is also created whereas if the LOG_RETAIN is changed from the **Configure Parameters**, you should backup the database afterwards.

Configuring UNIX/Windows apply control server as AR

The server that the Apply program runs must be configured as Application Requestor for remote capture control and target servers.

The DB2 for UNIX and Windows has inbuilt AR functionality if the remote server(s) are also DB2 UDB for UNIX and Windows. Configuration can be done from Client Configuration Assistant (CCA) or from the command line using the commands:

```
CATALOG TCPIP NODE node-name REMOTE host-name SERVER service-name
```

where *node-name* is any name which will be used to catalog databases under the node, *host-name* is either the IP adress or the hostname and *service-name* is the port number remote DB2 uses.

```
CATALOG DATABASE database-name AS alias AT NODE node-name
```

where *node-name* is the name used on catalog tcpip node command, *database-name* is the name of the remote database alias you want to catalog, *alias* is the name you want use in the connect statements.

You can list other options available for these commands from the command line by using **db2 ? catalog**.

If the remote server(s) are DB2 UDB for z/OS, the communication protocol used is Distributed Relational Database Architecture (DRDA). DB2 Connect or DB2 UDB ESE V8 for UNIX and Windows possesses the DRDA AR functionality.

Besides cataloging the remote server to the database directory and node directory, DB2 UDB for z/OS subsystem must also be cataloged to the DCS directory. You can use the following command to catalog the remote z/OS subsystem as a DCS database:

```
CATALOG DCS DATABASE database-name AS location-name
```

where *database-name* is any name you choose *location-name* must be the location name of the remote DB2 subsystem.

CCA can also be used to configure client connectivity.

Consider the DB2 subsystem with the following location name, port number and hostname:

Location name: STPLEX4A_DSN7.
Port number: 8020
Hostname: splex4a.stl.ibm.com

This remote DB2 subsystem can be cataloged to the DB2 UDB for UNIX and Windows DRDA AR with the following commands:

```
db2 catalog tcpip node stplex2 remote stplex4a.stl.ibm.com server 8020
db2 catalog database stpdsn7 at node stplex2 authentication dcs
db2 catalog dcs database stpdsn7 as stplex4a_dsn7
```

You can check your definitions with the following commands:

```
db2 list database directory
db2 list node directory
db2 list dcs directory
```

Starting Capture and Apply as a Windows service

Replication Center prepares the commands to create and start a Capture and Apply service based on the parameters specified on *Start Capture* and *Start Apply* windows. There are also commands to create and drop Windows services for starting Capture and Apply.

Capture and Apply can run as a service only on Windows 2000 and Windows NT operating systems.

The following commands define Capture and Apply as a windows services respectively:

```
asnscrt -C <db2-instance> account password <asncap-command>
asnscrt -A <db2-instance> account password <asnapply-command>
```

where *db2-instance* is DB2 instance service name, *account* and *password* are the account and password you logged on to Windows.

You can obtain DB2 instance service name from the **Start Menu** of Windows by following the path **Settings -> Control Panel -> Administrative Tools -> Services**. On the *Services* window, select and right-click the DB2 instance and choose **Properties**. Use the Service Name on the *General Panel* as the DB2 instance name on **asnscrt** command.

Account must always start with a period and a backslash (.\).

The following examples creates Capture and Apply services:

```
asnscrt -c DB2-0 .\db2drs3 db2drs3 asncap capture_server=sample
capture_path=c:\capture
asnscrt -a DB2-0 .\db2drs3 db2drs3 asnapply control_server=applydb
apply_qual=xyz1 apply_path=c:\apply
```

If *capture_path* or *apply_path* are not specified, it is defaulted to the path in DB2PATH. If DB2PATH is not set, then the path is defaulted to DB2 installation directory.

The following naming convention is used for the Capture service:

DB2.<instance>.<alias>.CAP.<schema>

where *instance* is the DB2 instance service name, *alias* is the database alias for capture control server, *schema* is the capture schema.

The service name created for the above **asnscrt -c** command is: *DB2-DB2-0.SAMPLE.CAP.ASN*.

The following naming convention is used for the Apply service:

DB2.<instance>.<alias>.APP.<qualifier>

where instance is the is the DB2 instance service name, alias is the database alias for apply control server, qualifier is the apply qualifier.

The service name created for the above **asnscrt -a** command is: *DB2-DB2-0.APPLYDB.APP.ASN*.

The replication service created can be dropped with the following command:

```
asnsdrop <service-name>
```

where *service-name* is the service name of Capture or Apply.

An example command created by the Replication Center is as follows:

```
ASNSCRT -A <db2_instance> .\<user_ID> <password> -START asnapply
CONTROL_SERVER=APPLYDB APPLY_QUAL=XYZ1 APPLY_PATH=c:\apply PWDFILE=psw/psw.psw
```

db2_instance, *user_ID* and *password* should be provided as explained above.

6.1.4 Considerations for DB2 UDB for z/OS

General information on DB2 DataPropagator

Unlike UNIX and Windows platforms, data replication is performed by a separate product on z/OS. The latest data replication product is IBM DB2 DataPropagator V8.1 for z/OS.

You can find some information about requirements, environment and installation of DB2 DataPropagator V8.1 below.

- The operating system must be OS/390 Version 2 Release 10 (5647-A01) or higher, with the Version 2 Release 10 Language Environment or higher.
- This version of DB2 DataPropagator V8.1 requires DB2 UDB for z/OS V6 (5645-DB2) with PTF UQ56678 or DB2 UDB for OS/390 and z/OS V7 (5675-DB2) with PTF UQ62179.
- SMP installation of the product is required.
- Capture, Apply, Monitor and Trace programs of DB2 DataPropagator V8.1 are z/OS Unix System Services (uss) applications.
- HFS installation is required for NLS message services and for running the programs listed above from the uss.
- The profile of uss users must include the required load libraries in STEPLIB as in Example 6-5 on page cclxxxviii.
- Uss users must include */usr/lpp/db2repl_08_01* on the PATH in their profile. The */bin* directory under this directory contains the executables as empty files with sticky bits.
- ASNAPLX and ASNCAP modules which read DB2 log IFI must be placed on an APF-authorized library.
- DB2 DataPropagator V8.1 must be installed on all systems in the data-sharing group.

- The Capture, Apply, Monitor and Trace programs can be run from JCL, BPX Batch as well as from uss. See , “Using JCL to operate Capture and Apply” on page cccx.
- If Replication Center will be used for operating Capture, Apply or Monitor programs in this subsystem, DAS (DB2 Administration Server) should be installed and configured.
- Capture, Apply and Monitor are DB2 application programs. They must be bound manually. Necessary DBRM’s to bind these packages come in the installation libraries. There are totally three plans and twenty-five packages bound under four collections. List of plans and packages are seen on Table 6-1. On the third column of the table, the list of packages that should be bound are listed. The packages must be grouped into collections as seen on the second column of the table. The collection name is selected arbitrarily. You may prefer a different collection name according to your site’s standards. Each collection must be included in the PKLIST of the bind command for the plan given on the first column. The plans for DB2 DatatPropagator V8.1 are ASNTC810 for Capture, ASNTA810 for Apply and ASNTM810 for Monitor.
- Binding ASNLOAD package under Apply plan (ASNTA810) is only required if you plan to refresh your target tables with the ASNLOAD exit routine by using *loadxit* parm in Apply. If you are running Apply on z/OS, the default sample program will call DB2 for z/OS Crossloader utility on ASNLOAD. The crossloader package DSNUGSQL and DSNUTILS store procdure package should also be bound into Apply plan (ASNTA810).
- Apply packages must be bound to source, apply control and target servers. The Apply plan on the server where you run Apply must include all the apply packages from the source, control and target servers which replicates data to and from. It is recommended to use generic location name on the PKLIST of Apply plan. This is recommended in order not to re-bind the Apply plan after every new source or target server definition to the subscription sets Apply processes.
- If the subsystem which Capture, Apply or Monitor are running is DB2 UDB for OS/390 and z/OS V7 and unicode encoding scheme is selected as default for your subsystem then ENCODING(EBCDIC) option in your bind commands is required.
- The following are the recommended isolation levels for the packages in the following plans:

  All packages in ASNTC810: ISOLATION(UR), except ASNCCPWK and ASNREG which are ISOLATION(CS).

  All packages in ASNTA810: ISOLATION(UR), except ASNLOAD and ASNAFET which is ISOLATION(CS).

All packages in ASNTM810: ISOLATION(UR) except ASNMDATA which is ISOLATION(CS).

The recommended isolation level for all the plans is UR.

- KEEPDYNAMIC(YES) bind option is recommended for better performance.

Table 6-1 DB2 DataPropagator V8.1 plans and packages

PLAN	COLLECTION	PACKAGES
ASNTC810 ASNTA810 ASNAM810	ASNCOMMON	ASNDBCON ASNMSGT ASNSQLCF ASNSQLCZ
ASNTC810	ASNCAPTURE	ASNADMIN ASNCCPWK ASNCDINS ASNCTSQL ASNCMON ASNPRUNE ASNREG ASNTXS ASNUOW
ASNTA810	ASNAPPLY	ASNAAPP ASNAWPN ASNACMP ASNAFET ASNAISO ASNAMAN ASNAPPWK ASNAPRS ASNLOAD
ASNAM810	ASNMONITOR	ASNMDATA ASMONIT ASMNUPDT

Configuring z/OS apply control server as AR

The connectivity of Apply program to capture control and target servers is essential. Apply accesses the capture control tables and CD tables at the capture control server and accesses the target server to propagate the updated data. The server which Apply is running must be configured as Application Requestor (AR) to capture control and target servers. Capture control and target servers are Application Servers (AS).

The protocol used for communication of DB2 servers on various platforms is called Distributed Relational Database Architecture (DRDA). In this section, we

describe how to configure DB2 UDB for z/OS subsystem as a DRDA AR using TCP/IP protocol.

DRDA AR functionality is provided by Distributed Data Facility (DDF) for DB2 UDB for z/OS. DDF requires connectivity configuration information of the servers for outbound connections to be present in the Communications Database (CDB).

The tables of CDB listed below are used for definitions of outbound requests of a DB2 subsystem:

- SYSIBM.LOCATIONS: If the AS is a DB2 UDB for z/OS, this table holds the LOCATION name and port number of the AS. If the AS is DB2 UDB for UNIX and Windows, this table holds the database alias name in the database directory and port number of the AS.
- SYSIBM.IPNAMES: This table holds the IP addresses (or host names) of the AS defined in SYSIBM.LOCATIONS.
- SYSIBM.USERNAMES: This table holds the authentication information of the AS's that require translation for authentication.

Consider the configuration on Figure 6-20 on page ccxcvi. Subsystem D7DP is the apply control server. Target server is subsystem DSN7. D7DP must be configured as AR to DSN7. The location name of AS is required.

Location name: STPLEX4A_DSN7.

The following insert statement to the SYSIBM.LOCATIONS table of D7DP is sufficient to configure the communication in this case because DSN7 and D7DP subsystems are on the same z/OS system:

```
INSERT INTO SYSIBM.LOCATIONS(LOCATION,LINKNAME)
VALUES('STPLEX4A_DSN7','LNKDSN7')
```

Assume D7DP is also propagating to the target server A23BK31Z seen on Figure 6-5 on page cclxxiii. Configuration information of A23BK31Z is as follows:

Database alias: SAMPLE
IP adress: 9.1.39.85
Port number: 50000
Userid:db2drs3
Password:db2drs3

The AS communication configuration is inserted to the CDB of D7DP:

```
INSERT INTO SYSIBM.LOCATIONS(LOCATION,LINKNAME,PORT)
VALUES('SAMPLE','LNKBK31',50000)
INSERT INTO SYSIBM.IPNAMES(LINKNAME,SECURITY_OUT,USERNAMES,IPADDR)
VALUES('LNKBK31','P','O','9.1.39.85')
```

```
INSERT INTO SYSIBM.USERNAMES (TYPE,LINKNAME,AUTHID,NEWAUTHID,PASSWORD)
VALUES('O','LNKBK31','CAYCI','db2drs3','db2drs3')
```

where CAYCI is a RACF userid on z/OS where D7DP runs.

The value assigned to *LOCATION* column in the SYSIBM.LOCATIONS is the location name of the AS (if the AS is DB2 UDB for z/OS) or database alias (if the AS is DB2 UDB for UNIX and Windows). *LINKNAME* given is any name chosen to associate the row in SYSIBM.LOCATIONS table to the row in SYSIBM.IPNAMES.

The *LINKNAME* in SYSIBM.IPNAMES is the linkname from SYSIBM.LOCATIONS. The *IPADDR* column is either a real IP adress or host name.

The *AUTHID* in the SYSIBM.USERNAMES is an userid from the AR system that needs to be translated. *NEWAUTHID* and *PASSWORD* columns define the authentication information for AS. You insert this translation information if the AS is DB2 UDB for UNIX and Windows and AUTHENTICATION(SERVER) is specified on the database manager configuration. Translation is also required for DB2 UDB for z/OS AS's where TCP/IP *ALREADY VERIFIED* is set to NO.

Using JCL to operate Capture and Apply

Besides Replication Center and uss shell, Capture and Apply programs can be run by the JCL as batch jobs or started tasks.

Whether run as a batch job or started task, commands can be either called using BPXBATCH or with a native program call in the JCL.

In the JCL, parameters are used with the associated keywords as in other platforms. The parameters are passed to the commands via PARM in the EXEC card. The operating system limit on the number of characters that can be specified on PARM is 100.Therefore, it may not be possible to specify all required parameters within this limit. There are two recommendations to overcome this problem:

- Update the parameter tables IBMSNAP_CAPPARMS and IBMSNAP_APPPARMS with your defaults. It is possible to specify all the parameters except *capture_server* and *capture_schema* in the IBMSNAP_CAPPARMS tables. You can specify all the parameters in the IBMSNAP_APPPARMS table except *apply_path*, *apply_qual* and *control_server.* There is one IBMSNAP_CAPPARMS table per capture schema with one row in it whereas there is one IBMSNAP_APPPARMS per subsystem which may have zero or more rows. Each row in IBMSNAP_APPPARMS contains the defaults for one apply qualifier.

- Use the minimum number of characters for the keywords that will differentiate them from other keywords. For example, *capture_sc* can be used instead of *capture_schema.*

Important: It is recommended to exploit the parameter tables IBMSNAP_CAPPARMS and IBMSNAP_APPPARMS to specify the frequently used parameters.

Although, DataPropagator programs can be run from the JCL, they are still the uss programs and they use HFS as default file system. If either *capture_path* or *apply_path* do not exist in the parameter table or not specified during start-up, the home directory of the user who submitted the JCL or RACF userid of the started task, is assigned as the path for file I/O. You can also give a directory which exists in HFS as absolute *capture_path* or *apply_path.*

If you are using JCL you may not prefer to use uss shell to browse the log created by Capture or Apply. It is possible to forward the file I/O to z/OS datasets. If the *capture_path* or *apply_path* starts with //, this enables the Capture and Apply programs to direct their file I/O to *sequential datasets on z/OS*. The sequential datasets are created by system defaults. The // in the path indicates that HLQ of the dataset will be the userid who submitted the start Capture or start Apply job.

Important: It is possible to create the log files of Capture and Apply as sequential datasets of z/OS. You append “//’ in front of the *capture_path* or *apply_path*, where “//’ stands for the userid.

If you do not want userid as the HLQ of the dataset, you must append a quotation mark infront of “//”.

Assume that *capture_path* is set to “//SYSADM”, dataset will be created with the following naming convention:
<userid>.SYSADM.<capture_server>.<capture_schema>.CAP.LOG.

If *capture_path* is set to “‘//SYSADM”, dataset will be created with the following naming convention: *SYSADM.<capture_server>.<capture_schema>.CAP.LOG.*

The sample JCL’s used in this section are for starting Capture and Apply for the replication configuration seen on Figure 6-20 on page ccxcvi.

Example 6-8 is a sample JCL for starting Capture as a batch job on z/OS. The Capture control server is D7DP, which is a DB2 subsystem and the capture schema is CAP1. Some of the defaults are altered with following SQL statements in the IBMSNAP_CAPPARMS table before submitting this JCL.

```
UPDATE CAP1.IBMSNAP_CAPPARMS SET CAPTURE_PATH='//SY4A'
UPDATE CAP1.IBMSNAP_CAPPARMS SET STARTMODE='WARMSI'
```

The log file for Capture of Example 6-8, is created as a sequential dataset named: *CAYCI.SY4A.D7DP.CAP1.CAP.LOG,* You can browse this sequential dataset from ISPF.

Example 6-8 Sample JCL for starting Capture

```
//CAPCAP1 JOB USER=CAYCI,NOTIFY=CAYCI,
//         MSGCLASS=H,MSGLEVEL=(1,1),
//         REGION=0M,TIME=50
/*JOBPARM SYSAFF=SY4A
//ASNCAP   EXEC PGM=ASNCAP,
//  PARM='ENVAR("LANG=en_US")/CAPTURE_SERVER=D7DP CAPTURE_SC=CAP1'
//STEPLIB  DD  DISP=SHR,DSN=DPROPR.V810.BASE.TESTLIB,
//             UNIT=SYSDA,VOL=SER=RMS002
//         DD  DISP=SHR,DSN=SYS1.SCEERUN
//         DD  DISP=SHR,DSN=DSN.D7DP.SDSNLOAD
//CAPSPILL DD DSN=&&CAPSPL,DISP=(NEW,DELETE,DELETE),
//            UNIT=SYSDA,SPACE=(CYL,(50,100)),
//            DCB=(RECFM=VB,BLKSIZE=6404)
//CEEDUMP  DD   DUMMY
//SYSTERM  DD   SYSOUT=*
//SYSUDUMP DD   DUMMY
//SYSPRINT DD   SYSOUT=*
```

Capture keeps the uncommitted changes in memory until the *memory_limit* (Capture parameter) is reached. When this limit is reached, Capture spills to VIO using a default value. You can override the default VIO allocation of Capture by providing a CAPSPILL DD card in your capture job.

A sample JCL is provided on Example 6-9 to start Apply for the replication configuration seen Figure 6-20 on page ccxcvi. Before starting Apply, the following insert statement is done to direct file I/O of Apply to sequential dataset:

```
INSERT INTO ASN.IBMSNAP_APPPARMS(APPLY_QUAL,APPLY_PATH) VALUES('APY1','//SY4A')
```

Example 6-9 Sample JCL for starting Apply

```
//APPAPY1  JOB USER=CAYCI,NOTIFY=CAYCI,
//         MSGCLASS=H,MSGLEVEL=(1,1),
//         REGION=0M,TIME=NOLIMIT
/*JOBPARM SYSAFF=SY4A
//APPLYX   EXEC PGM=ASNAPPLY,
//  PARM='ENVAR("LANG=en_US")/control_se=D7DP
//             db2_subsystem=D7DP apply_qual=APY1'
//STEPLIB  DD  DISP=SHR,DSN=DPROPR.V810.BASE.TESTLIB,
//             UNIT=SYSDA,VOL=SER=RMS002
```

```
//         DD  DISP=SHR,DSN=SYS1.SCEERUN
//         DD  DISP=SHR,DSN=DSN.DSN7.SDSNLOAD
//CEEDUMP  DD   DUMMY
//SYSTERM  DD   SYSOUT=*
//SYSUDUMP DD   DUMMY
//SYSPRINT DD   SYSOUT=*
//ASNASPL  DD  DSN=&&ASNASPL,DISP=(NEW,DELETE,DELETE),
//          UNIT=VIO,SPACE=(CYL,(11,7)),
//          DCB=(RECFM=VB,BLKSIZE=6404)
```

On the sample JCL, the *control_server* is the location name and *db2_subsystem* is the subsystem id. *Apply_qual* defines the subscription set which will be processed by this Apply program.

The log file allocation is similar to Capture. If an *apply_path* that implies to open the log as a sequential dataset is not specified, the log file goes to HFS under the home directory of the userid who submitted the job or RACF userid of the started task. The log file for Apply of Example 6-9, is created as a sequential dataset named: *CAYCI.SY4A.D7DP.APY1.APP.LOG.*

The spill file specifications are provided with ASNASPL DD card. This DD card on the Apply JCL is optional. If it does not exist in the JCL, the default unit for spill file is VIO for DB2 UDB for z/OS. If you want to direct it to disk or manage the allocation parameters, you must provide DD statement as in the Example 6-9.

Several new parameters are added to Capture in this version. All the new parameters are used by Capture running on DB2 UDB for z/OS. Besides, some of the old parameters which were not supported on DB2 UDB z/OS are supported in this platform too. There are no positional parameters anymore. The parameters seen as bold on the first column are old positional parameters. Old parameters and corresponding new parameters and defaults for new parameters for Capture are shown on Table 6-2.

Table 6-2 Old and new Capture parameters

OLD PARAMETER	NEW PARAMETER	DEFAULT FOR NEW
DB2_subsystem_name=	capture_server	capture_server=DSN
TERM (default) NOTERM	term=y term=n	term=y
WARM (default) WARMNS COLD	startmode=warmns startmode=warmsa startmode=warmsi startmode=cold	startmode=warmns

OLD PARAMETER	NEW PARAMETER	DEFAULT FOR NEW
PRUNE (default) NOPRUNE	autoprune=y autoprune=n	autoprune=y
NOTRACE(default) TRACE	N/A	N/A
SLEEP (n)(default=0)	sleep_interval	sleep_interval=5
ALLCHG CHGONLY	N/A as Capture parameter. Specified when registrating source.	N/A
N/A	capture_path	home dir. of user in HFS
N/A	capture_schema	capture_schema=ASN
N/A	autostop=n autostop=y	autostop=n
N/A	commit_interval	commit_interval=30
N/A	lag_limit	lag_limit=10080
N/A	logreuse=n logreuse=y	logreuse=n
N/A	logstdout=n logstdout=y	logstdout=n
N/A	memory_limit	memory_limit=32
N/A	monitor_limit	monitor_limit=10080
N/A	monitor_interval	monitor_interval=300
N/A	prune_interval	prune_interval=300
N/A	retention_limit	retention_interval=10080
N/A	trace_limit	trace_limit=10800

The old parameter, corresponding new parameter and defaults for the new parameter are seen on Table 6-3. There are no positional new parameters, the old parameters which were positional are shown in bold.

Table 6-3 Old and new Apply parameters

OLD PARAMETER	NEW PARAMETER	DEFAULT FOR NEW
Apply_qual	apply_qual	N/A

OLD PARAMETER	NEW PARAMETER	DEFAULT FOR NEW
DB2_subsystem_name	db2_subsystem	DSN
Control_server_name	control_server	N/A
LOADX NOLOADX (default)	loadxit=n loadxit=y	loadxit=n
MEM (default) DISK	spillfile=mem spillfile=disk	spillfile=mem
INAMSG (default) NOINAMSG	inamsg=y inamsg=n	inamsg=y
NOTIFY NONOTIFY (default)	notify=n notify=y	notify=n
SLEEP NOSLEEP (default)	sleep=y sleep=n	sleep=y
DELAY(n) (default=6)	delay	delay=6
ERRWAIT(n) (default=300)	errwait	errwait=300
NOTRC (default) TRCERR TRCFLOW	N/A	N/A
N/A	apply_path	home dir. of user in HFS
N/A	copyonce=n copyonce=y	copyonce=n
N/A	logreuse=n logreuse=y	logreuse=n
N/A	logstdout=n logstdout=y	logstdout=n
N/A	opt4one=n opt4one=y	opt4one=n
N/A	trlreuse=n trlreuse=y	trlreuse=n
N/A	term=y term=n	term=y
N/A	sqlerrorcontinue=n sqlerrorcontinue=y	sqlerrorcontinue=n

The Capture or Apply jobs started with JCL can be stopped in an orderly way with the modify command as follows: **`/f capcap1,stop`** and **`/f asnapy1,stop`** where *asnapy1* and *capcap1* are jobnames in the JOB card.

6.1.5 Considerations for DB2 UDB for iSeries

IBM DB2 Data Propagator for iSeries can only run on V5 R2 OS/400 V5. The lastest PTF must be applied after installation.

Remote journal does not support LOB and view replication

Setup Apply to remote source server as specified in , “Starting Apply on the iSeries” on page cclxxvi

Quering the IBMSNAP_RESTART Capture control table is another option to check if the Capture program is running. The STATUS column should contain a Y. Indicating the Capture program is currently running for each journal processed within the Capture schema. There should be 1 row for each journal.

When replicating to an existing target table, Apply will fail when processing changes from the source, to a target table that is not journaled. This is usually a problem to existing User-Copy target type tables.

6.1.6 Troubleshooting the operations

In this section, we try to address the problems that may occur when operating (starting or stopping) Capture or Apply only. It is not aimed to give troubleshooting information for replication problems.

The following are some guidelines if you have problems operating Capture or Apply:

- If Capture or Apply is not starting you may need to check if prerequisites for either Capture or Apply are satisfied for your environment. The prerequisites for Capture and Apply are discussed on , “Prerequisites for Capture” on page cccxviii and , “Prerequisites for Apply” on page cccxviii.
- Both Capture and Apply are run as processes. If Capture or Apply does not start, you may consider checking if a process is started. There are several operating system commands to display Capture and Apply processes. See , “Manage processes and threads” on page cccxx for the alternative commands that display information of processes and threads.

- Although Capture and Apply are successfully started, data propagation is not performed according to the replication definitions, problem must be investigated based on the error messages in the log and the contents of the capture and apply control tables. The command to analyze the control information is explained on Section , “Analyzing the control tables” on page cccxxiv.
- If there are performance problems, you can collect and analyze the trace data with *asntrc* command. Refer to Chapter 10.4.13, “Time spent on each of Apply’s sub-operations” on page cdlxxxiii on how to benefit the *asntrc* data for performance problems.

iSeries troubleshooting operations

- On the iSeries, Capture and Apply runs as seperate batch jobs in QZSNDPR sub system. Therefore, you need to check if those jobs are still running. See, “Querying Capture and Apply status on the iSeries” on page cclxxxii to work with QZSNDPR sub system.
- When starting Capture on the iSeries, make sure you are specifying the correct capture schema name, if you using more than one in your replication environment. Also check if you are specifying the correct journal names, if you are not using the default value to use all journals.See, Figure 6-3 on page cclxxi and Figure 6-21 on page ccxcviii.
- Make sure you are up todate on applying the latest PTF for licence program 5722-DP4 DB2 DataPropagator.
- After applying new PTF, you have to recreate the SQL packages. See heading, “Create SQL packages” on page cclxxviii
- When Apply is connecting to a remote source server, check connectivity configuration. See heading, “Connectivity configuration” on page cclxxvii
- There must be at least one active subscription set for the Apply qualifier, and the subscription set must contain at least one of the following:
  - Subscription-set member
  - SQL statement or procedure
- Produce apply traces:
  - Turn Apply trace on when starting Apply program from the STRDPRAPR command and specify value in the TRACE prarmeter. See, Figure 6-21 on page ccxcviii
  - Use WRKDPRTRC command. See chapter 18, *DB2 Universal Database Replication Guide and Reference,* SC27-1121-00
- Run ANZDPR to analyze your replication configuration to help you determine the program with your Capture or Apply program. See, “Analyzing the control tables” on page cccxxiv.

Prerequisites for Capture

- If the capture control server is DB2 UDB for UNIX and Windows, database should be enabled for replication. See Section 6.1.3, “Considerations for DB2 UDB for UNIX and Windows” on page ccci.
- If the capture control server is DB2 UDB for z/OS, requirements listed on Section , “General information on DB2 DataPropagator” on page cccvi. They are also listed shortly below:
  - ASNCAP must on an APF-authorized library.
  - DB2 DataPropagator V8.1 is only compatible with V6 and V7. There are PTF requirements for these versions.
  - Requirement for Unix System Services and LE.
  - PATH and STEPLIB should be set properly on uss profile.
  - DB2 DataPropagator packages and plan should be bound.
- User running the Capture must have certain authorizations. The authorization requirements are specific to platform. See Chapter 2, “Setting up for replication”, in *IBM DB2 Universal Database Replication Guide and Reference,* SC27-1121-00 for authorization requirements of Capture.

Prerequisites for Apply

- If the apply control server is DB2 UDB for z/OS, all the requirements listed above for Capture are valid except that there is no requirement for Apply program to be on an APF-authorized library.
- If Apply is on DB2 UDB for z/OS, Apply packages should also be bound to each server which Apply connects, manually,
- Connectivity is very important for Apply. Apply must successfully connect to Capture control server in pull mode and must also successfully connect to target in push mode. For this reason, Apply should be configuration as DRDA Application Requestor for Capture control server in pull mode and DRDA AR for target server in push mode. Refer to Section , “Configuring z/OS apply control server as AR” on page cccviii and Section , “Configuring UNIX/Windows apply control server as AR” on page ccciii for configuring DB2 UDB for z/OS or UNIX/Windows apply control servers as AR.
- If the apply control server is DB2 UDB for UNIX and Windows, a password file must exist with userid, password pairs of the remote servers for authenticating remote servers. See , “Maintaining the password file” on page cccxix.
- There must be at least one active subscription set for the Apply qualifier.
- The subscription set must contain at least one of the following:
  - Subscription-set member
  - SQL statement

– Procedure

- Refer to Chapter 2, “Setting up for replication”, in *IBM DB2 Universal Database Replication Guide and Reference, SC27-1121-00* for authorizations required for operating Apply.

Maintaining the password file

The password file might be required if apply control server is on DB2 UDB for UNIX and Windows. The userid, password combinations are kept encrypted in this file. Default name of this file is *asnpwd.aut*. Apply searches this file under *apply_path* by default. If a different name or different path is used this should be specified with *pwdfile* parameter.

If the either capture control server or target server is a remote server, Apply should pass userid, password information when connecting to:

- DB2 UDB for z/OS or DB2 UDB for OS/400 servers.
- DB2 UDB for UNIX and Windows server and AUTHENTICATION=SERVER is specified on the database manager configuration file of the remote server.

Password file is maintained by **asnpwd** command. Following functions are available with **asnpwd.**

1. Initialize the password file: This should be done once for the apply control server. The password file is created either by the default name or by the name specified. The password file is created on the directory command is run or it can be directed to another directory with *using* keyword. The following are two examples of usage:

```
asnpwd init
asnpwd init using os390pwd.aut
```

2. Add userid, password combinations for the servers: This operation is repeated for every database alias or DB2 subsystem on z/OS Apply connects during apply cycle and require authentication.

```
asnpwd add alias sample id db2drs3 password db2drs3
asnpwd add alias stpdsn7 id cayci password cayci using os390pwd.aut
```

3. Modify either userid or password:

```
asnpwd modify alias sample id db2drs3 password db2drs9
asnpwd modify alias stpdsn7 id cayci password db299db2 using os390pwd.aut
```

4. Delete an entry from the password file:

```
asnpwd delete alias sample
asnpwd delete alias stpdsn7 using os390pwd.aut
```

The command options can be queried from the command line by entering

```
asnpwd ?
```

You can find the complete syntax of the command on Chapter 17,“System commands for replication (UNIX, Windows, z/OS)”, in *IBM DB2 Universal Database Replication Guide and Reference,* SC27-1121-00

Manage processes and threads

Capture and Apply are both DB2 applications. It is possible to display them with DB2 commands like any other DB2 application.

Display commands on UNIX and Windows

The commands run on this section display the Capture and Apply started for the replication configuration seen on Figure 6-5 on page cclxxiii.

The first **list applications** command on Example 6-10, displays the threads of the Capture application. Capture uses five threads. This command is issued from the server where *SAMPLE* database is cataloged locally. SAMPLE is the capture control server running on Windows.

The second **list applications** command on Example 6-10, displays the threads of the Apply application. This command is issued from the server where Apply runs which is an AIX system. Apply has four threads on the Apply control server.

The third **list applications** command on Example 6-10, displays the Apply thread on capture control server. This command is issued from the capture control server.

Example 6-10 List applications

```
C:\>db2 list applications
Auth Id  Application    Appl.      Application Id                 DB       # of
         Name           Handle                                    Name    Agent
-------- -------------- ---------- ------------------------------ -------- ----
DB2DRS3  asncap.exe     11         *LOCAL.DB2.00CC07002037        SAMPLE   1
DB2DRS3  asncap.exe     10         *LOCAL.DB2.00CBC7002036        SAMPLE   1
DB2DRS3  asncap.exe     9          *LOCAL.DB2.00CB87002035        SAMPLE   1
DB2DRS3  asncap.exe     8          *LOCAL.DB2.00CB07002034        SAMPLE   1
DB2DRS3  asncap.exe     6          *LOCAL.DB2.00C9C7002032        SAMPLE   1
$ db2 list applications
Auth Id  Application    Appl.      Application Id                 DB       # of
         Name           Handle                                    Name    Agent
-------- -------------- ---------- ------------------------------ -------- ----
CAYCI    asnapply       14         *LOCAL.cayci.083847002624      AIXSAMP  1
CAYCI    asnapply       13         *LOCAL.cayci.082837002623      AIXSAMP  1
CAYCI    asnapply       12         *LOCAL.cayci.081827002622      AIXSAMP  1
CAYCI    asnapply       10         *LOCAL.cayci.080817002620      AIXSAMP  1
C:\>db2 list applications
Auth Id  Application    Appl.      Application Id                 DB       # of
```

```
         Name           Handle                                  Name    Agent
-------- -------------- ---------- ------------------------------ -------- ----
DB2DRS3  asnapply       16         G90126B2.001B.083847002625     SAMPLE   1
$ ps -ef | grep -i 32896
   cayci 32896 27922   0 17:26:19  pts/0  0:00 asnapply control_server=aixsamp
apply_qual=APY1
```

The *ps* command displays the process of Apply. This is a UNIX command and can not be issued from Windows.

Display commands on z/OS

The commands used in this section display the processes, threads, IPC queues of the sample jobs to start Capture on Example 6-8 on page cccxii and the sample job to start apply on Example 6-9 on page cccxii.

It is possible to display DB2 applications with *display thread* command as in Example 6-11. On the example, this command is used to display the DB2 threads of Capture application on DB2 UDB for z/OS. Capture has five threads. These threads are HOLDL, ADMIN, PRUNE, WORKER and the one that starts the others. They are explained on Section , “Querying the Status of Capture and Apply” on page cclxxx. You can find out the jobname of start capture job (*CAPCAP1*), authorization id (*CAYCI*), planname (*ASNT810*) and ASID of the job (*00AF*) from the output.

The status of the process (that these threads belong) can be displayed from USS. On Example 6-11, the process is displayed by *d omvs* command. On the example, USS process is displayed for a particular userid. It is also possible to display the same information by giving the address space id. See ”*z/OS V1R4.0 MVS System Commands”,* SA22-7627-04 for the complete documentation of the command.

The last command issued from the console on Example 6-11, displays the TCB’s of this process. There are five TCBS’s corresponding to five threads.

You can display the process information with UNIX command from the uss shell. A sample *ps* command is seen as the last command on Example 6-11.

Example 6-11 Display Capture threads on z/OS

```
#D7DP- DISPLAY THREAD(*)
DSNV401I  #D7DP- DISPLAY THREAD REPORT FOLLOWS -
DSNV402I  #D7DP- ACTIVE THREADS -
NAME     ST A   REQ ID           AUTHID   PLAN     ASID TOKEN
D7DP     RA *     0 028.DBAA  02 SYSOPR            040A    39
 V445-G91E4A2B.G4A3.B82EA9FFC629=39 ACCESSING DATA FOR 9.30.74.43
SERVER   RA *     5 db2bp.exe    CAYCI    DISTSERV 040A   626
 V437-WORKSTATION=A23BK31Z, USERID=cayci,
```

```
      APPLICATION NAME=db2bp.exe
 V445-G9012755.H004.005A06182424=626 ACCESSING DATA FOR 9.1.39.85
DB2CALL  T       24 CAPCAP1      CAYCI    ASNTC810 00AF   627
DB2CALL  T        5 CAPCAP1      CAYCI    ASNTC810 00AF   628
DB2CALL  T     3679 CAPCAP1      CAYCI    ASNTC810 00AF   629
DB2CALL  T     2556 CAPCAP1      CAYCI    ASNTC810 00AF   630
DB2CALL  T       28 CAPCAP1      CAYCI    ASNTC810 00AF   631
-D OMVS,U=CAYCI
 BPXO040I 11.38.56 DISPLAY OMVS 395
 OMVS     000D ACTIVE          OMVS=(00,4A)
 USER     JOBNAME  ASID        PID       PPID STATE   START     CT_SECS
 CAYCI    CAPCAP1  00AF   16777523          1 HRI--- 11.11.54    131.39
   LATCHWAITPID=         0 CMD=ASNCAP
COMMAND INPUT ===>                                          SCROLL ===> CS
-D OMVS,PID=16777523
 BPXO040I 11.49.23 DISPLAY OMVS 552
 OMVS     000D ACTIVE          OMVS=(00,4A)
 USER     JOBNAME  ASID        PID       PPID STATE   START     CT_SECS
 CAYCI    CAPCAP1  00AF   16777523          1 HRI--- 11.11.54    131.39
   LATCHWAITPID=         0 CMD=ASNCAP
  THREAD_ID        TCB@     PRI_JOB  USERNAME   ACC_TIME SC  STATE
  1C55A8D000000000 007AAE88                        .158 STA  YU
  1C5677D000000001 007AA660                        .008 SWT JY
  1C4C4F6000000002 007AD058                     122.812 SPM JY
  1C4CA9F000000003 007B5088                        .191 STA JY
  1C4D1E6000000004 007AD2E8                       7.020 STA JY
CAYCI:../CAYCI:> ps -ef
     UID        PID       PPID  C    STIME TTY       TIME CMD
   CAYCI        122          1  - 12:15:23 ?         0:02 OMVS
   CAYCI        140        184  - 12:16:15 ttyp0030  0:00 ps -ef
   CAYCI        184        122  - 12:15:23 ttyp0030  0:02 -sh
   CAYCI   16777523          1  - 11:11:55 ?         2:11 ASNCAP
```

Same commands are used on Figure 6-12 to display Apply. Apply has four threads running on the Apply server. Three of them are HOLDL, WORKER and ADMIN threads. They are explained on Section , “Querying the Status of Capture and Apply” on page cclxxx.

Example 6-12 Display Apply threads on z/OS

```
#D7DP- DISPLAY THREAD(*)
 DSNV401I  #D7DP- DISPLAY THREAD REPORT FOLLOWS -
 DSNV402I  #D7DP- ACTIVE THREADS -
NAME     ST A   REQ ID           AUTHID   PLAN     ASID TOKEN
DB2CALL  T      100 APPAPY1      CAYCI    ASNTA810 00BD   632
DB2CALL  T        3 APPAPY1      CAYCI    ASNTA810 00BD   633
DB2CALL  T       13 APPAPY1      CAYCI    ASNTA810 00BD   634
DB2CALL  T      135 APPAPY1      CAYCI    ASNTA810 00BD   635
```

```
-D OMVS,ASID=00BD
 BPXO040I 13.57.40 DISPLAY OMVS 801
 OMVS     000D ACTIVE          OMVS=(00,4A)
 USER     JOBNAME  ASID        PID       PPID STATE   START     CT_SECS
 CAYCI    APPAPY1  00BD   33554615          1 HRI--- 13.49.32       .27
   LATCHWAITPID=         0 CMD=ASNAPPLY
-D OMVS,PID=33554615
 BPXO040I 13.58.44 DISPLAY OMVS 813
 OMVS     000D ACTIVE          OMVS=(00,4A)
 USER     JOBNAME  ASID        PID       PPID STATE   START     CT_SECS
 CAYCI    APPAPY1  00BD   33554615          1 HRI--- 13.49.32       .28
   LATCHWAITPID=         0 CMD=ASNAPPLY
  THREAD_ID        TCB@     PRI_JOB  USERNAME   ACC_TIME SC  STATE
  1C4D6C0000000000 007CA718                        .171 STA  YU
  1D835C3000000001 007ABE88 OMVS                   .003 SWT JK
  1D8E945000000002 007AB798                        .015 CLO JY
  1E25671000000003 007AAE88 OMVS                   .061 SLP JS
CAYCI:../CAYCI:> ps -ef
     UID        PID       PPID  C    STIME TTY       TIME CMD
   CAYCI   33554615          1  - 13:49:32 ?         0:00 ASNAPPLY
```

Display IPC queue on UNIX and z/OS

When Capture or Apply is starting, an IPC queue is created. The message **ASN8008D** seen on Figure 6-13 is from the capture log. The same message also appears on the Apply log. The IPC queue keys used for Capture and Apply is listed on this message. The *ipcs* command can be issued from uss on z/OS or UNIX shell. This command displays the IPC queues. The IPC queue is always generated under */tmp* directory.

Example 6-13 IPC queue of Capture on z/OS

```
ASN8008D  "Capture" : "CAP1" : "Attached to" IPC queue with key(s)
"(0xf04c5c06, 0xf14c5c06, 0xf24c5c06)".
CAYCI:../tmp:> ipcs | grep CAYCI
m     204806 0xf04c5c06 --rw-rw----    CAYCI    #QXXD
s     397320 0xf14c5c06 --ra-ra----    CAYCI    #QXXD
s     397321 0xf24c5c06 --ra-ra----    CAYCI    #QXXD
```

It is also possible to locate the associated IPC queue on the Apply log. The *ipcs* command on Example 6-14 which is issued on AIX, displays the IPC queue of Apply.

Example 6-14 IPC queue of Apply on AIX

```
2002-09-06-17.26.22.460339 <Asnenv:setEnvIpcQRcvHdl> ASN8008D  "Apply" : "APY1"
: "Created" IPC queue with key(s) "(0x30000084)".
$ ipcs | grep 0x30000084
```

```
q    917553 0x30000084 --rw-rw----    cayci    staff
```

Analyzing the control tables

If data is not propagating as defined in the registrations and subscriptions, the information in the Capture and Apply control tables is very critical in determining the problem. It is also recommended to query the control tables periodically in order to detect the problems which are not evident.

The command that analyzes the control tables is *asnanalyze.* This command is run from the command line and produces an htm output which can be displayed from a browser. If the command is run without any parameter, information on how to use this command, is displayed. The complete syntax of this command is on Chapter 17,“System commands for replication (UNIX, Windows, z/OS)”, in *IBM DB2 Universal Database Replication Guide and Reference, SC27-1121-00.*

The following is an example of *asnanalyze* commands:

```
asnanalyze -db sample aixsamp -la detailed -fn a020906.htm -pw pass1.pwd
```

Detailed analysis of databases *sample* and *aixsamp* is requested on the example. First database is capture control server and the second one is apply control server. The output (*a020906.htm*) produced for the above command contains the following:

- Contents of capture control tables for all capture schemas from *SAMPLE*.
- Contents of apply control tables from *AIXSAMP*.
- CD table column analysis.
- List of Capture and Apply packages and some of the bind options (like isolation level) from DB2 catalog.
- Database alias connection summary.
- Results of queries run against the control tables to diagnose the cause of existing Capture and Apply problems.
- Statistics on the number of rows that are eligible for pruning in the CD and UOW tables.
- Subset of table and tablespace statistics of control tables from DB2 catalog.
- List of (registered) tables with insufficient indexes
- Results of queries run against the control tables to detect subscription definition errors.
- Inconsistencies which are detected on the control tables.

There are several options exist for this command which limits or increases the amount of information gathered. You should consider using this command with the appropriate options based on your requirements.

Note: iSeries users can run this command from your workstation to analyze the Capture and Apply control tables on the iSeries. As regards the preceeding asnanalyze command example the -db parameter could be the iSeries RDB name. Then make sure you have the password table created on the workstation with the user ID and password to access the iSeries. See chapter 17, in *DB2 Universal Database Replication Guide and Reference,* SC27-1121-00 on the asnpwd command to create the password table

The ANZDPR command on the iSeries could also analyze your iSeries Apply and Control tables and output htm file, similar to the asnanayze described in this section. You can find details of this command in Chapter 18, from the *DB2 Universal Database Replication Guide and Reference,* SC27-1121-00.

6.2 Capture and Apply Parameters

Capture and Apply programs assign values to parameters from different resources:

- Parameters have shipped values.
- IBMSNAP_CAPPARMS which is the parameter table of Capture, is populated with shipped defaults when capture control tables are created. It is possible to change the defaults from Replication Center or by SQL. If a row exists in the table, the not null values override the shipped defaults of Capture parameters for a particular capture schema.
- IBMSNAP_APPPARMS table of Apply is created by Replication Center. It is possible to insert parameter values for each apply qualifier. The not null values on each row override the shipped defaults of Apply parameters for a particular apply qualifier.
- When starting Capture and Apply, parameter values can be overridden.

The shipped default of *capture_server* of Capture and *control_server* of Apply are not hard-coded on DB2 UDB for UNIX and Windows but defaults to DB2DBDFT registry variable.

The *capture_path* and *apply_path* also do not have shipped defaults but if they are not set at start-up or not specified at the parameter table, are assigned a value depending on how they are run. If the programs run from the Replication Center, the *working directory* at *Run now or Save command* window is assigned. If the programs run from the command line, the current directory is the path. If

run as a Windows service, the path in DB2PATH (which defaults to DB2 installation directory) is assigned.

It is possible to update Capture parameter values in IBMSNAP_CAPPARMS from the Replication Center. The assigned values of the Capture parameters can also be changed dynamically from the Replication Center.

6.2.1 Change Capture parameters

All parameters of Capture except *capture_server* and *capture_schema* exist in IBMSNAP_CAPPARMS. All the Capture parameters that exist in IBMSNAP_CAPPARMS except *logstdout* can be updated from Replication Center.

You can update the values in IBMSNAP_CAPPARMS by following the path **Operations -> Capture Control Servers**. On the content pane, select and right-click the capture control server. From the option list select **Manage Values in CAPPARMS**.

Attention: The updates made to the IBMSNAP_CAPPARMS will not be effective until Capture is recycled as Capture reads this table only during start-up.

If you want your change to be effective without stopping Capture, from the Replication Center, follow the path **Operations -> Capture Control Servers**. On the content pane, select and right-click the capture control server. From the option list select **Change Operational Parameters**.

Most of the Capture parameters except *capture_schema*, *capture_server, capture_path*, *startmode* and *logstdout* can be changed dynamically. The changes made will be effective immediately but will be lost after Capture stops.

Attention: Dynamic changes made to the parameters are not updated to IBMSNAP_CAPPARMS and will be lost when Capture is stopped.

6.2.2 Capture parameters

The Capture parameters can be grouped as follows based on their usage:

- A number of parameters are essential. The *capture_server*, *capture_schema*, *capture_path* and *startmode*. Although it is possible to set defaults to all of them, these parameters are the ones that should be determined before starting Capture. We already described these parameters throughout the chapter as they are essential for the start of Capture.

- Performance parameters: The parameters below are performance related. They may be changed after tuning the replication environment. These parameters and their impact on performance are described in detail on Chapter 10.2, “Capture Performance” on page cdxlix.
  - *autoprune*
  - *commit_interval*
  - *prune_interval*
  - *lag_limit*
  - *memory_limit*
  - *retention_limit*
  - *sleep_interval*
- Some of the parameters define the working environment of Capture. The parameters *logreuse*, *logstdout, trace_limit, term, monitor_limit, monitor_interval* can be included into this group. They can be set based on your requirements. These parameters are described shortly below:
  - *logreuse*: Default is n. If no l*ogreuse* is requested, Capture appends to the log file even after restart. If *logreuse* is requested previous information in the log is deleted during start-up.
  - *logstdout:* Default is n. The messages in the log file are also directed to the standard output if *logstdout* is set to *y*.
  - *trace_limit:* Default is 10080 (min). This is the prune threshold for rows in IBMSNAP_CAPTRACE.
  - *monitor_interval*: Default is 300 (sec). This is the interval Capture writes to IBMSNAP_CAPMON table.
  - *monitor_limit*: Default is 10080 (min). This is the interval Capture writes to IBMSNAP_CAPMON table.
- Specific use parameters: The parameter *autostop* can be included in this group. Default for *autostop* is *n*. If *autostop* is requested, Capture stops when reaches the end of log. The mobile users may benefit from *autostop*, otherwise it must be set to *y*.

6.2.3 Apply Parameters

The Apply parameters can also be categorized based on their usage:

- The following parameters can be regarded as essential.
  - *apply_qual*
  - *control_server*
  - *db2_subsystem (valid only for z/OS)*
  - *apply_path*
  - *pwdfile (valid only for UNIX and Windows)*

These parameters are explained in detailed under the topics how to Start Apply.

- The following are performance related parameters which are described on Chapter 3.4.5, “Control tables described” on page clxv and explained in detail on the related to performance topics on Chapter 10.4.10, “Apply operations and Subscription set parameters” on page cdlxxvi.
  - *delay*
  - *opt4one*
  - *spillfile*
- There are also parameters of Apply which are related to the working environment of the program. The parameters *logreuse, logstdout, inamsg*, *term, trlreuse, errorwait, sqlerrorcontinue* can be combined in that group. These parameters can be set based on your choice according to your sites resources (like disk) and your requirements. These parameters are described on Chapter 3.4.5, “Control tables described” on page clxv.
- The following can be grouped under special use parameters:
  - *copyonce:* Default is n. If it is set to y, Apply processes the subscription sets of the apply qualifier once and stops.
  - *notify:* Default is n. If it is set to y, Apply calls the ASNDONE exit routine after every successful subscription set processing.
  - *sleep:* Default is y. The default behavior of Apply is to process the subscription sets until no more data to replicate is left. It then sleeps for a certain period of time until it wakes up again and starts processing. When *sleep* is set to n, apply terminates when no data is left for replication. This parameter is suitable for mobile users who periodically connect to the Capture server and process the subscription sets altogether.
  - *loadxit:* This parameter is explained on Section 6.4, “Using ASNLOAD for the initial load” on page cccxxxi.

6.3 Other operations

We have covered how to start, query and stop Capture and Apply until this section. In this section, we cover other possible operations of Capture. These are are prune control tables, reinitialize capture, suspend and resume capture.

6.3.1 Pruning control tables

Data in some of the control tables accumulate during replication. It is essential to prune the information in the control tables for two reasons:

- to improve the performance of Apply.
- to release the storage used by expired control data.

Pruning can either be done automatically or by command. Whether it is done automatically or manually, data in the following tables are pruned:

- CD and UOW tables: Rows are pruned based on the LSN value updated to SYNCPOINT column on IBMSNAP_PRUNE_SET by Apply or *retention_limit*.
- IBMSNAP_SIGNAL: Rows are pruned based on SIGNAL_STATUS or *retention_limit*.
- IBMSNAP_CAPTRACE: Rows are pruned based on *trace_limit*.
- IBMSNAP_CAPMON: Rows are pruned based on *monitor_limit*.

The *retention_limit*, *trace_limit* and *monitor_limit* are Capture parameters.

Pruning is done by the PRUNE thread. This thread can work concurrently with the WORKER thread. Replication Center assigns locksize row for all DB2 UDB for UNIX and Windows capture tables and CD table. Default locksize used is either page or row for DB2 UDB for z/OS capture tablespaces and CD tablespace. Pruning is not serialized with capture and does not affect capture latency with default locksizes. If you change the locksize to table or you have control tables or CD tables already defined with locksize table, PRUNE thread may conflict with WORKER thread. You should consider to alter locksize of capture control tables and CD table to default which Replication Center assigns.

Important: It recommended to have minimum number of rows in the control tables for the efficiency of Apply program. Capture with *autoprune* set to yes, prunes the control tables at *prune_intervals* and guarantees that control tables only keep necessary data for replication.

Automatic pruning is highly recommended but if you have reasons to keep data in the control tables longer than data replication requires, there is a command to prune data from the control tables on demand:

```
asnccmd capture_server=<server> capture_schema=<schema> prune
```

where *server* is the database alias if capture control server is DB2 UDB for UNIX and Windows database but subsystem name if capture control server is on DB2 UDB for z/OS.

This command prunes data from the capture control tables of a particular schema.

This operation can also be done from the Replication Center. Follow the path **Operations -> Capture Control Servers**. On the content pane, select and

right-click the capture control server. From the option list select **Prune Capture Control Tables**. You will be prompted for the capture schema. The command prepared by the Replication Center is displayed on the *Run now or Save Command* window.

If you are manually pruning control tables, the information about the number of eligible rows in CD and UOW tables on **asnanalyze** report can help you to determine if you need pruning.

6.3.2 Reinitializing Capture

Capture reads the active registrations from IBMSNAP_REGISTER at start-up. If a registration option is altered, there is no automatic mechanism to warn Capture of that change. This should be done manually. The following reinitialization command enables the Capture to re-read IBMSNAP_REGISTER.

```
asnccmd capture_server=<server> capture_schema=<schema> reinit
```

where *server* is the database alias if capture control server is DB2 UDB for UNIX and Windows database but subsystem name if capture control server is on DB2 UDB for z/OS.

This operation can also be done from the Replication Center. Follow the path **Operations -> Capture Control Servers**. On the content pane, select and right-click the capture control server. From the option list select **Reinitialize Capture**. You will be prompted for the capture schema. The command prepared by the Replication Center is displayed on the *Run now or Save Command* window.

The following registration options can be altered:

- Row-capture rule
- Before-image prefix
- Stop Capture on error
- Allow full refresh of target table
- Capture updates as pairs of deletes and inserts
- Capture changes from replica table
- Conflict detection level

The message ASN0023I is recorded to the Capture log indicating that the reinitialization command is issued.

If a new registration is made, Capture becomes aware of this new source when Apply signals during subscription cycle and reads the registration from the IBMSNAP_REGISTER. Reinitialization is not necessary for this case.

Reinitialization with reinit command is not meaningful when a registration is deleted or the CD table is altered. Refer to Chapter 8.1.3, “Removing registrations.” on page ccclxxviii and Chapter 8.2.7, “Adding a new column to a source and target table” on page cccxc how these operations accomplished.

6.3.3 Suspend and resume Capture

Suspend command switches the Capture threads to **is resting** state. By this way, the CPU usage and memory usage of Capture is minimized without stopping it. The operation of Capture can then be resumed again. The primary reason of suspending Capture is to save CPU and memory during heavy workload.

The following commands are used to suspend and resume the operations of Capture:

```
asnccmd capture_server=<server> capture_schema=<schema> suspend
asnccmd capture_server=<server> capture_schema=<schema> resume
```

where *server* is the database alias of capture control server if it is DB2 UDB for UNIX and Windows database but subsystem name if capture control server is on DB2 UDB for z/OS.

These operations can also be done from the Replication Center. Follow the path **Operations -> Capture Control Servers**. On the content pane, select and right-click the capture control server. From the option list either select **Suspend Capture** or **Resume Capture**. You will be prompted for the capture schema. The command prepared by the Replication Center is displayed on the *Run now or Save Command* window.

These commands result on ASN0028I (for suspend) and ASN0029I (for resume) messages to be written to Capture log. You can detect when Capture is suspended and resumed by locating the messages in the log.

6.4 Using ASNLOAD for the initial load

Apply initially does a full refresh for a replication source. During full refresh, Apply accesses not the CD table but directly to the source and inserts the rows to the target table. Inserts to the target table can be the performance bottleneck if the table being replicated is a large one.

It is possible to optimize the full refresh process by deploying DB2 utilities. DB2 utilities are called by the ASNLOAD exit routine. There are different load alternatives offered by ASNLOAD.

The source and compiled version of this routine is shipped by DB2 so that it is possible to alter the routine based on your site requirements.

6.4.1 Using ASNLOAD on DB2 UDB for UNIX and Windows

Using ASNLOAD as is

If your definitions satisfy the following prerequisites of ASNLOAD, you can use it by setting *loadxit* parameter to *y* when starting Apply.

- The source and target columns must have matching data types.
- Target table only contains the columns from the replication definition.

ASNLOAD offers alternative load methods:

- Crossloader: This function is available on DB2 UDB for UNIX and Windows V8 and DB2 UDB for z/OS V7. It is basically a cursor based select from source and load into target from that cursor. This function must exist at the target. Apply must run on target and target must be a DB2 table. Crossloader uses two part name (owner.table) for the select statement. For this reason source table can be one of the following:
  - Local DB2 table
  - Federated source
  - Remote DB2 table with user supplied nickname (nickname specified on IBMSNAP_SUBS_MEMBR, see , “Specifying the preferred method” on page cccxxxii).
- Export/Load: Export from the source, load into target.
- Export/Import: Export from the source, import into target.

You can indicate which method to use but if you leave it to default, the method is determined depending on the source and target types.

Specifying the preferred method

The preferred method for a subscription member is updated to LOADX_TYPE column of IBMSNAP_SUBS_MEMBR. You must update this column with one of the following values:

- 1 : stands for *do not call ASNLOAD for this member*. If *loadxit* is set for Apply, initial load of all the members of all subscription sets Apply processes are done by calling ASNLOAD. By setting the LOADX_TYPE column to 1 for a member you can exclude this member from initially loaded by ASNLOAD.
- 2 : is for *user defined* load. Set this value if you have altered ASNLOAD based on your site requirements.
- 3 : *Crossloader*

- 4 : *export/load*
- 5 : *export/import*

If LOADX_TYPE is null, the method is selected by ASNLOAD.

If crossloader is the preferred method and source is a remote DB2 database to target, then you must create a nickname for the source on the target server and update the schema and nickname to LOADX_SRC_N_OWNER and LOADX_SRC_N_TABLE columns of IBMSNAP_SUBS_MEMBR. This is not required if source is not a DB2 source.

Important: You can determine the LOAD type of ASNLOAD by updating LOADX_TYPE column on IBMSNAP_SUBS_MEMBR.

Configuration File

It is possible to pass some parameters to ASNLOAD through configuration file (*asnload.ini*). This file is on *samples\repl* under the installation directory.

If you want to pass parameters to ASNLOAD, you must copy the configuration file to *apply_path* and update. The parameters are related to utility statements prepared by ASNLOAD. *UID* and *PWD* if specified used on connect statement. The possible keywords are seen on Example 6-15. A comma at the beginning of a line makes that line comment. Parameters grouped under *COMMON* are valid for all databases. The parameter values for a certain database are grouped under the database name alias in square brackets. Database assignments override the assignments made under *COMMON*. If a value is not assigned to a parameter neither under database nor under COMMON, defaults in ASNLOAD program are used.

Example 6-15 Sample configuration file for ASNLOAD

```
[COMMON]

[SAMPLE]
COPY=Y
COPYTO=c:\apply\bkp

;--- POSSIBLE KEYWORDS:---
;COPY  {Y/N}
;COPYTO {Path of the Copyimage}
;LOBFILE {List of Lobfilesbasenames}
;LOBPATH {List of Lobpaths}
;MAXLOBS {max number of lobfiles that can be created with a given list of
Lobfilenames}
;UID     {only if compiled sample is used}
;PWD     {only if compiled sample is used}
```

```
;DATA_BUFFER_SIZE
;DISK_PARALLELISM
;CPU_PARALLELISM
;
```

The database alias may be the source, target or the apply control server. The following can be specified:

- An userid and password for connect. If not specified for a database alias connect will be tried without userid and password.
- A backup copy of the target database can be generated while loading if the target database is forward recovery enabled (LOGRETAIN or USERCOPY set to ON on database configuration).
- If replication source has LOB columns, LOBPATH, LOBFILE and the limit on the number of LOB files can be specified.
- DATA_BUFFER_SIZE, DISK_PARALLELISM, CPU_PARALLELISM can be coded for improving the performance of the utility.

Important: Apply searches for the configuration file under the *apply_path*. If you want to pass parameters, copy *asnload.ini* to *apply_path*.

Message files

Message files are created by ASNLOAD under *apply_path*. Assume *apply_qual* used is, APY1. Following message files will be created:

- *APY1.trc*
- *asnloadAPY1.msg*
- *asnaEXPT.msg* if EXPORT is used as the method
- *asnaIMPT.msg* if IMPORT is used as the method
- *asnaLOAD.msg* if LOAD is used as the method

Customizing ASNLOAD

It is possible to customize the ASNLOAD since the source is also shipped with the product. The source (*asnload.smp*) is on directory *samples\repl* under the installation directory. The source is a C program with SQL calls. The following steps can be used for customization:

- There are instructions on how to modify the source on the sample. Rename (with sqc extension) and modify the source according to your site needs.
- Precompile, compile and linkedit. The program must be linked to a directory in the PATH.
- The LOADX_TYPE must be set to 2 at IBMSNAP_SUBS_MEMBR and start Apply with *loadxit* parameter.

6.4.2 Using ASNLOAD on DB2 UDB for z/OS

ASNLOAD on DB2 UDB for z/OS calls Crossloader for initial load of targets. There are no other shipped methods available.

- Apply must be started with *loadxit* set to *y*.
- Crossloader runs on DB2 UDB for V7.
- ASNLOAD calls DSNUGSQL. DSNUGSQL is the stored procedure which provides Crossloader functionality.
- Apply must run in the target.
- Three part names (server.owner.table) are used on select statement of Crossloader. There is no restriction for the source server for this reason.
- At the shipped source code, ASNLOAD loads data to the tablespace with **`LOG(NO)`** option. The *backup pending state* of the tablespace is reset by **`REPAIR NOCOPYPEND`** command on the shipped code. If you prefer to backup your tablespace after LOAD or modify the LOAD options you should modify ASNLOAD.
- The source of ASNLOAD is also shipped on DB2 UDB for z/OS. It is a C program issuing DB2 calls. ASNLOAD can be customized and prepared with your program preparation procedures on your site.

Using ASNLOAD on

7

Troubleshooting and monitoring

This chapter discusses the following:

- Troubleshooting and monitoring introduced
- Basic replication troubleshooting
- Replication alert monitoring
- Other monitoring
- Advanced troubleshooting

7.1 Troubleshooting and monitoring introduced

In the previous chapters, you were introduced to DB2 Replication and walked through setting up your replication environment. Here we look at troubleshooting problems that you may encounter in your replication environment. This is primarily achieved through monitoring. Monitor can be used to ensure that replication is working as desired. Also, monitoring allows early detection of possible problems. In the later chapters, you will be shown how to configure, tune, and maintain your environment for advanced scenarios, and performance.

Troubleshooting and monitoring combines both general DB2 tools, and replication specific tools. After this introduction we will first present some basic troubleshooting. Then we identify the tools available to monitor your replication environment. Lastly, we go into troubleshooting in more depth. For the scenarios in this chapter, the following environments were primarily used:

- Windows NT / DB2 ESE v8 Open Beta 2
- AIX 4.3.3 / DB2 ESE v8 Open Beta 2
- Redhat Linux 7.3 / DB2 ESE v8 Open Beta 2

There are many components involved in DB2 Replication: Replication Center, Capture, Apply, DB2 data definition (DDL), authorization, DB2 connectivity, and so on. If the source of a DB2 Replication problem is not obvious, it is necessary to take steps to narrow the scope of your search for the cause. Throughout the chapters we have suggested troubleshooting specific to the topic discussed. Section 1.5, “DB2 Replication V8 close up” on page liii and the detailed discussion of Capture and Apply’s sub-operations throughout Chapter 10, “Performance” on page cdxli are useful in providing ideas of what to look for as you investigate the problem.

This chapter assumes that you have basic DB2 database administration and replication knowledge.

7.1.1 DB2 Administration Client

The DB2 Administration Client is greatly enhanced from previous versions. The additional monitoring functionality and utilities simplify administration work and allow easier scheduling and monitoring. DB2 Replication takes advantage of these, and also complements them with some tools specific to replication including the DB2 Replication Alert Monitor. The majority of the capabilities can be accessed through a variety of interfaces. Using the DB2 Administration Client is advantageous as it presents complex data in meaningful ways.

7.2 Basic replication troubleshooting

Many of the common user errors from previous versions cannot occur in this version, or are more detectable. An example is both Capture and Apply with now function well if **asncap** or **asnapply** are passed the database server names in lowercase. Still, there will undoubtable be situations that you have to spend some effort troubleshooting.

DB2 Replication, and DB2 itself on each platforms, offer facilities for narrowing problem cause. First, you will likely look at online help, use RC to check the status of Capture and Apply, or look in the Capture or Apply program logs. Those topics are described in this section. If this does not allow you to resolve the problem, you will likely use SQL selects from the various control tables, CD, source, and target tables. You may also want to monitor for conditions to try and catch the problems occurrence. If you still do not have resolution, you may use tools such as **asnanalyze** (on iSeries use ANZDPR) or **asntrc** to obtain more detailed information. Those topics will be described in the later sections.

7.2.1 Getting help

Information center

DB2 Information Center is a web page (HTML) based help system. One of it's interfaces is Java based and integrated into the DB2 Administration Client. From the Replication Center (RC) you can access the information center from the menu bar, **Help —> Information Center**. From this Java interface, selecting any topic will load a web page on that topic into your web browser. You need to have a recent version of your web browser with Java and Javascript enabled. The web browser based information is now consistent across Linux, Windows, and UNIX, including the documentation search facility.

Tip: When not using the help, we prefer viewing and searching through the whole documents. For this reason, we tend to download the PDF versions of the documents from the DB2 support sites.

The documentation for DB2 on Linux, UNIX, and Windows no longer includes a troubleshooting guide. The information that was contained in it is now in the documents on the topics. The documents are now more task oriented.

Note: Information center searches are exact phrase. Boolean, wild-card, and partial word searches do not work.

Administration client

As well as the information center, and the other selections under the menu bar's **Help** tab, help is available in two ways. If you stop your mouse for a few seconds over in the Administration client for many elements a brief description will be shown in a box on top of the window. These descriptions are termed *infopops*. Most windows in the client have **Help** buttons. Clicking on them will load your web browser with the page that describes the topic. The web page also provides links to related information.

If information does not appear correct in the RC or other graphical tools, try selecting from the menu bar, **View —> Refresh**. This will ensure that what you see in the window is up to date.

Restriction: When cataloging new DB2 systems and databases, it is currently necessary to shutdown RC and all the administration client's graphical tools to have the catalog entries to be shown in RC. We have been informed that this issue will likely be resolved in the version 8 fixpak 2 time frame.

DB2 ?

DB2 operations return information, warning or error messages. Many commands return a combination of these. These messages identify the success or failure of an operation, and if a failure, often how to correct the problem. Figure 7-1 shows the messages returned one time when I selected **OK** from the *Register Tables* window.

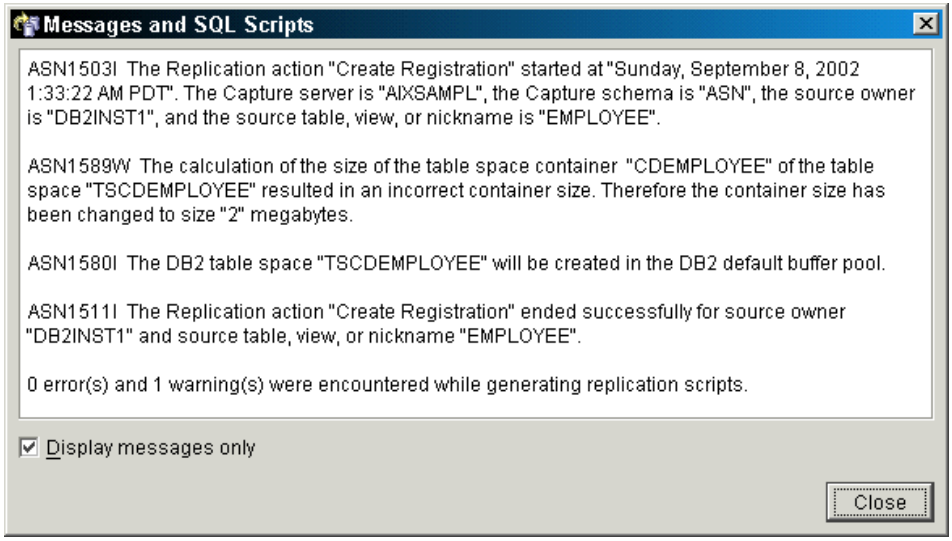


Figure 7-1 DB2 messages

There are three informational messages, one warning message, and zero error messages. Each message begins with an identifier. They are ASN1503I, ASN1589W, ASN15801I, and ASN1511I. Detailed information about a particular message can be looked up using the Command Center or other DB2 Command Line Processor (CLP) interfaces, see Example 7-1.

Example 7-1 Detailed output of a DB2 Message

```
db2 => ? asn1589

ASN1589W The calculation of the size of the table space
          container  "<container>" of the table space
          "<tspace>" resulted in an incorrect container size.
          Therefore   the container size has been changed to size
          "<size>" megabytes.

Explanation:

The calculation of the table space container size has resulted in
a value that is too low to be used in a valid table space
container definition. To ensure that the definition will be
accepted by DB2, a replication specific minimum container size
has been provided for the table space container definition.

User Response:

For the calculation based on a percentage of the current source
table size, check whether the source table contains data and if
the statistics of the source table are up to date (using the
RUNSTATS utility). For the calculation based on a number of rows,
check whether the number of rows is realistic.
```

All DB2 messages begin with an identifier of the following format: a prefix of three characters for the component, a four or five digit number, and a single character identifying the severity. Table 7-1 shows prefixes that are commonly encountered doing DB2 Replication.

Table 7-1 Common Message Identify Prefixes

Prefixes	Description
ASN	DB2 Replication.
DBA	Database Administration tools.
DB2	CLP.
SQL	database manager (the instance) when a warning or error condition has been detected.

The four or five digit message number unique identifies that message from other messages with the same prefix. As shown in Example 7-1, the ending is option when you enter the command to see the detailed information about a message. The ending provides important information about the nature of a message. Table 7-2 shows the endings that are most commonly encountered doing DB2 Replication.

Table 7-2 Message Identify Endings

Ending	Description
I	Information.
W	Warning.
N	Error, used by messages with a prefix of SQL.
E	Error.
C	Critical error, usually a crash.

There are often quoted strings in the message, example “<captureSchema>”. These have values particular to the result the message is reporting. Sometimes you will see the quotes together, ““. This means there is no value for this variable.

The detailed response gained by ? *messageID* contains three sections: message, explanation, and user response. If there are multiply explanations, they will be numbered, and the user response will contain the same numbering. Often this information will allow you to resolve the problem.

When seeking assistance from DB2 Customer Support, or others, the exact message identify and the quoted springs are important. If we were seeking asstance for an ASN0068E problem, we would say, “We are encountering ASN0068E, insert statement too long for CD table with strings “ASN”, and “ASN.manyLargeColumnsTable””. We would also provide the exact output of **db2level**, if on Linux, UNIX, or Windows. Simply, looking up the meaning of the DB2 message would likely have allowed us to solve the problem ourselves.

Full information about the DB2 message format, and a listing of all messages are in the *DB2 UDB Message Reference Volume 1 & 2*.

Files generated

There are a large number of files that contain troubleshooting and debugging information.

instance.database.captureSchema.CAP.log

The Capture program log. This file is generated in CAPTURE_PATH and contains a trail of the Capture program’s programmatic status information. This

file is plain text. It can usually be readily interpreted. Still, it is mainly for problem investigation by DataPropagator support and development. An example is `db2inst1.SAMPLE.ASN.CAP.log`.

instance.database.applyQualifier.APP.log

The Apply program log. This file is generated in APPLY_PATH and contains a trail of the Apply program's programmatic status. This file is plain text. It can usually be readily interpreted. Still, it is mainly for problem investigation by DataPropagator support and development. An example is `DB2.REPSAMPL.WINSAM.APP.log`.

captureSchema.TRC and applyQualifier.TRC

A trace is common computing terminology to describe recording steps in a program in a verbose manner. The DB2 family of products can generates **`db2trc`** files, CLI traces, JDBC traces, a few specialized for replication, and a number of others. Capture and Apply traces are automatically generated in CAPTURE_PATH and APPLY_PATH respectively, generally only for abends and abnormal errors. DB2 replication traces can also be manually created.

db2diag.log

On Linux, UNIX, and Windows, DB2 logs fairly low level programatic status. This is generated at the location identified by the database manager configuration parameter DIAGPATH. DIAGPATH is where all automatically generated DB2 diagnostics files are on these platforms. There are four diagnostic levels set by the database manager configuration parameter DIAGLEVEL. Unless advised by a DB2 Customer Support Analyst leave do not change DIAGLEVEL from three. To view these parameters, as the instance owner:

```
db2 get dbm cfg
```

Replication Analyzer (asnanalyze and ANZDPR)

DB2 Replication Analyzer reads the various replication control tables and provides a snapshot of the replication environment. The analyzer also reads DB2 catalog tables for information relevant to replication. The output from the analyzer is in the form of an HTML file that can be viewed with a web browser. The output is useful for comparing the settings in the various replication control tables and the configuration of the staging tables to determine possible problem causes. The output of analyzer is frequently requested by IBM Customer Support while assisting in solving problems with replication.

`asnanalyze` runs on Linux, UNIX, and Windows and can connect to and read the contents of the replication control tables and catalog tables on any platform including z/OS and iSeries. **`asnanalyze`** is describe in the *DB2 UDB Version 8 Replication Guide and Reference* SC27-1121-00 chapter "System commands for replication (UNIX, Windows, z/OS)". **`asnanalyze`**'s online help can also be viewed

by entering **asnanalyze** at a command prompt with no input parameters. *Section 7.5.1, "asnanalyze and ANZDPR" on page ccclxv* contains an example of running **asnanalyze**

ANZDPR runs on iSeries and can collect information from replication control tables and the system libraries. ANZDPR is described in *DB2 UDB Version 8 Replication Guide and Reference,* SC27-1121-00, chapter "System commands for replication (OS/400)".

Replication Trace (asntrc and WRKDPRTRC)

With DB2 Replication Version 8, one of the new changes is in the tracing facilities. There is now one tracing facility, **asntrc**, that can trace both Capture and Apply. When **asntrc** process is started, input parameters indicate the capture schema or apply qualifier of the Capture or Apply that is to be traced. Multiple traces can be started, formatted, and stopped without stopping Capture or Apply.

The output of **asntrc** is useful for understanding where in Capture or Apply's various sub-operations a problem is occurring and for showing more detail about a problem. Like with Replication Analyzer, the output of asntrc is frequently requested by IBM Customer Support while assisting in solving problems with replication.

asntrc runs on Linux, UNIX, Windows, and on UNIX System Services (USS) on z/OS. **asntrc** is described in *DB2 UDB Version 8 Replication Guide and Reference* SC27-1121-00 chapter "System commands for replication (UNIX, Windows, z/OS)". **asntrc** also has online help which can viewed by entering **asntrc -help** at a command prompt. Section 7.5.2, "DB2 Replication trace" on page ccclxix descripts running **asntrc**, and the sections which follow that section show examples for both Capture and Apply. There is another example specific to viewing Apply's Performance Trace Records in Section 10.4.13, "Time spent on each of Apply's sub-operations" on page cdlxxxiii.

WRKDPRTRC runs on iSeries and traces an Apply job. The apply qualifier specified when WRKDPRTRC is started to indicate the Apply job that is to be traced. WRKDPRTRC is described in *DB2 UDB Version 8 Replication Guide and Reference* SC27-1121-00 chapter "System commands for replication (OS/400)". WRKDPRTRC also has online help that can be viewed by moving your cursor to the command in a display list and pressing F1 key. Within the WRKDPRTRC prompt screens, help for specific parameters can be seen by moving your cursor to the parameter and pressing F1.

DB2 Trace

The **db2trc** command is part of the DB2 for Linux, UNIX, and Windows product. It will capture each operation of the DB2 processes and threads. It is a very low

level diagnostic tool. The most common manner that you would be requested by DB2 Customer Support to run it is shown in Example 7-2.

Example 7-2 Taking a db2trc

```
Move sqllib/db2dump/db2diag.log to db2diag.logOLD ** or a name of your choice
Now collect the trace:
   db2trc on -l 8000000 -e -1 -t
   **command to trace **
   db2trc dump <filename.trc>  ** filename as PMR # is usually preferred **
   db2trc off   ** do not forget  to do this step!!
Now format the trace file (located in 'pwd'):
   db2trc fmt <filename.trc> <filename.fmt>   ** note if it wrapped
Also create a flow of the trace dump:
   db2trc flw <filename.trc> <filename.flw>
```

db2support

Introduced in a recent version 7 fixpak of DB2 for Linux, UNIX, and Windows is the **db2support** tool. It is shipped with version 8 on those platforms. This collects a fairly complete picture of the DB2 environment. Entering the command with no options will give complete syntax. Most often the basic invocation is all that is desired:

```
db2support . -d databaseAlias -c
```

Attention: Do not run **db2support** while experiencing a severe performance degradation -- contact DB2 Customer Support.

The output will be packaged into a file in the zip compress format in the directory, '.' that you invoked it. Unfortunately, the command does not know about the DB2 Replication environment, so you will have to manually collect those files.

How to get assistance

There is excellent DB2 online help at:

http://www-3.ibm.com/software/data/support/

From there you can navigate to DB2 product family support websites such as DB2 DataPropagator Support:

http://www-3.ibm.com/software/data/dpropr/support.html,

Or to platform specific resources such as the DB2 for Linux, UNIX, and Windows Support:

http://www.ibm.com/software/data/db2/udb/winos2unix/support.

iSeries does not have a support web site dedicated to DB2 and replication. There main support site is very good:

http://www-912.ibm.com/

There is also the USENET newsgroup comp.databases.ibm-db2 which has a very active user community. The DB2 Magazine http://www.db2mag.com is worth subscribing to.

IBM DB2 Customer Support can be reached in the United States of America at 1-800-237-5511. The world wide number is 1-404-238-1234 (charges may apply). Both phone numbers have live operators at all times of every day.

7.2.2 Status of Capture and Apply

From any system where you have the DB2 Administration Client installed, you can check the status of capture and apply. The capture and apply control servers must be cataloged, and be added to the RC. When you start Capture or Apply from the Replication Center, the message displayed recommands using the *Query Status* window for more information on the status of the program. We recommand looking at the *Capture Messages* and the *Apply Report* window first.

Attention: On iSeries and OS/400 to check the status use the WRKSBSJOB *subsystem* command, where *subsystem* is the name of the subsystem. In most cases the subsystem is QZSNDPR, unless you created your own subsystem description.

In the list of running jobs, look for the jobs you're interested in. The journal job is named after the journal to which it was assigned. If a job is not there, use the Work with Submitted Jobs (WRKSBMJOB) system command or the Work with Job (WRKJOB) system command to locate the job. Find the job's log (joblog) to verify that it completed successfully or to deterine the cause of failure.

Capture Messages

The *Capture Messages* window displays the Capture program's messages. These are rows in the capture control table IBMSNAP_CAPTRACE. The amount of data contained in this table is limited by the Capture configuration element TRACE_LIMIT, seven days by default.

To view the *Capture Messages* window:

1. Within RC double click on **Operations** so that the folders for **Capture Control Servers**, **Apply Control Servers**, and **Monitor Control Servers** are showing.

2. Select **Capture Control Servers**.
3. In the right pane, select the control server you are interested in seeing the status of Capture for.
4. From the menu bar at the top of the window choose **Selected —> Show Capture Messages**. Figure 7-2 shows the *Capture Messages* window after selecting **Retrieve**.

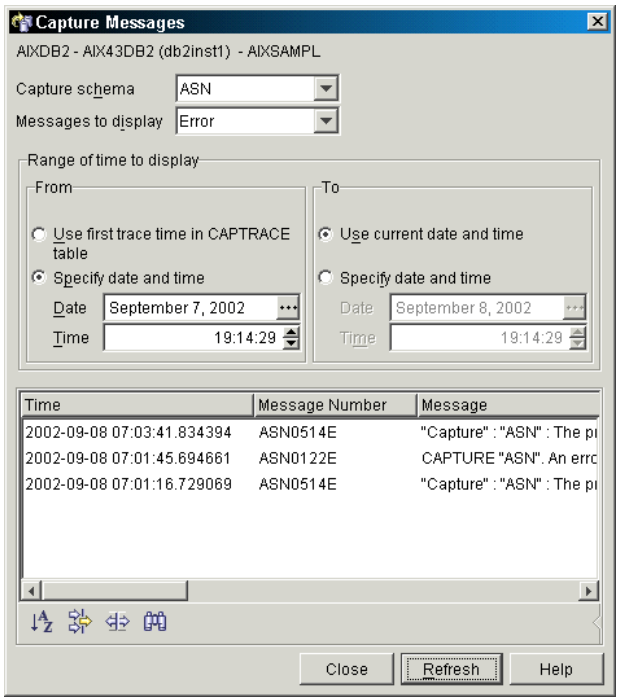


Figure 7-2 Capture Messages

You will likely have to scroll right to see the whole message. Clicking on any of the messages will highlite it, making reading the message as you scroll easier. This is also a good opportunity to use the knowledge gained on “DB2 ?” on page cccxl.

None of the messages shown are current. By default the *From* is *Specify date and time*, selecting **Retrieve** will show the previous 24 hours.

If we had another capture schema on this capture server we could retrieve it’s messages using the drop down menu *Capture Schema*. It is often useful to select **All** for *Messages to display* to view all of the most recent messages. They will likely be informational messages.

Apply Report

The *Apply Report* window displays the Apply program's messages. These are rows in the apply control table IBMSNAP_APPTRACE. The amount of data contained in this table is not limited. You will want to manually delete rows on a regular basis. To view the Apply program's messages:

1. Within RC double click on **Operations** so that the folders for **Capture Control Servers**, **Apply Control Servers**, and **Monitor Control Servers** are showing.
2. Double click **Apply Control Servers**.
3. Select the control server you are interested in seeing the status of Apply for.
4. Select **Apply Qualifies**.
5. In the right pane select the apply qualifier you are interested in.
6. From the menu bar at the top of the window choose **Selected —> Show Apply Report**. Figure 7-3 shows the *Apply Report* window.

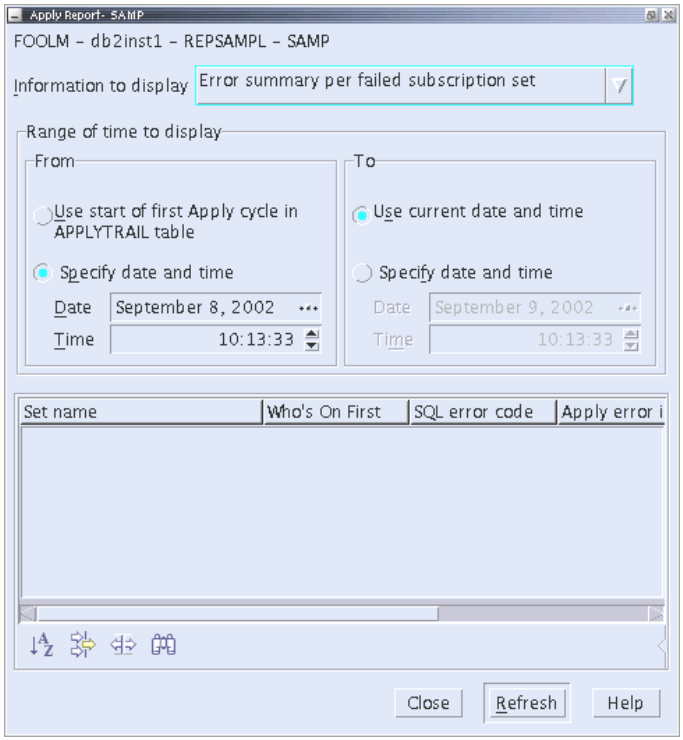


Figure 7-3 Apply Report

Unlike the *Capture Messages* window, here there is no indication of the time of the event. The usefulness of this window is in the selection of *Information to display*. The information is divided by subscription sets, and you can select **All subscription sets**, **Failed subscription sets**, **Successful subscription sets**, or the default of **Error summary per fairled subscription set**.

Clicking **Refresh** will display the set information for that selection. This may result in a message box with ASN1560E with quoted string “SQL0100W No row was found for FETCH, UPDATE or DELETE; or the result of a query is an empty table.”

When you try to query status, stop, or otherwise try to collect information from Capture or Apply, you may get SQL2220 “12” and ASN0506E “*theCommand*” “*captureSchema*” or “*applyQualifier*”. The ASN0506E message describes not being able to attach to the replication communications message queue. The cause is very likely that the program is not running.

Query Status

This status tool was explained in detail in “Querying the Status of Capture and Apply” on page cclxxx. Included there is using the **`status`** option for **`asnccmd`** and **`asnacmd`**. Near that section the process threads are also described. We find it generally more useful to look at the Capture and Apply logs.

Show Capture Throughput Analysis

This tool covered in detail in Chapter 10, “Performance” on page cdxli, specifically see 10.2.7, “Capture’s throughput” on page cdlxiii.

Show Apply Throughput Analysis

This tool is covered in detail in Chapter 10, “Performance” on page cdxli, speicifically see 10.4.12, “Apply Throughput” on page cdlxxxii.

Show End-to-End Latency

This tool is covered in detail in Chapter 10, “Performance” on page cdxli, specifically see 10.4.11, “End-to-end latency of replication” on page cdlxxx.

Log files

The log files *instance.database.captureSchema*.CAP.log and *instance.database.applyQualifier.*APP.log are often your best tool for start and stop failures. View them in your preferred text editor. Start at the end and work backwards. Example 7-3 shows a portion of a capture log.

Example 7-3 CAP.log

```
2002-09-09-08.48.37.271000 <CWorkerMain> ASN0100I  CAPTURE "ASN". The Capture
program initialization is successful.
2002-09-09-08.48.37.271000 <CWorkerMain> ASN0109I  CAPTURE "ASN". The Capture
program has successfully initialized and is capturing data changes for "1"
registrations. "0" registrations are in a stopped state. "0" registrations are
in an inactive state.
2002-09-09-08.53.37.603000 <PruneMain> ASN0111I  CAPTURE "ASN". The pruning
cycle started at "Mon Sep 09 08:53:37 2002".
2002-09-09-08.53.37.603000 <PruneMain> ASN0112I  CAPTURE "ASN". The pruning
cycle ended at "Mon Sep 09 08:53:37 2002".
2002-09-09-08.54.09.138000 <handleCAPSTART> ASN0104I  CAPTURE "ASN". In
response to a CAPSTART signal with MAP_ID "2
```

Further troubleshooting of the operations has been covered in Section 6.1.6, “Troubleshooting the operations” on page cccxvi.

7.2.3 Platform specific troubleshooting

As replication involves many software, hardware and network components, it is useful to use the tools that are specific to your environment in determining a problems cause. DB2 Replication is good at making the DB2 Replication tools available and consistent across the platforms supported.

z/OS

Check System Services Address Space (*ssid*MSTR where *ssid* is your DB2 subsystem), and the job message log (JESMSGLG) output listing at the System Display and Search Fascility (SDSF) to detect some of the errors of Capture and Apply. This tooling will keep track of errors like timeout, deadlock, resource unavailable, extend failure, or lock escalation experienced by Capture and Apply.

iSeries

Care has been taken to include iSeries troubleshooting content throughout the topics covered in this book, particularly in Chapter 6, “Operating Capture and Apply” on page cclxv. If the topics presented here do not offer assistance, please refer to the topic of interest in the other chapters.

7.3 Replication alert monitoring

As you have seen the DB2 Replication products, like the DB2 UDB engines, have built-in monitoring capabilities. The fascilities allow basic monitoring, as well as combinational and sophisticated monitoring. It can be done both interactively and by schedule. New in version 8 is alert monitoring.

Tip: If you are not interested in otherwise using Linux in your replication solution, it can make an ideal monitor server. All linux distributions include SMTP services either preconfigured or otherwise easily configured. The previous DB2 versions difficulty getting Java to work well on UNIX and Linux seems resolved. The installation and administration client's graphical user interfaces are near identical in appearance on Windows. The performance of the interface is comparable, if it does not favor Linux.

7.3.1 Creating monitoring control tables

Creation of the monitoring control tables follows similar steps as the capture and apply control tables, Chapter 3, "Replication control tables" on page cxli. The intention is to have one or more monitor control servers for multiple replication servers. A monitor server can be on DB2 for Linux, UNIX, Windows, and z/OS.

The control tables cannot be on iSeries. It can be monitored from one of the other platforms.

Within RC, to open the *Create Monitor Control Tables* window:

1. Double click on **Operations** so that the folders for **Capture Control Servers**, **Apply Control Servers** and **Monitor** are shown.
2. Select **Monitor Control Servers**.
3. From the menu bar at the top of the window choose **Selected —> Create Monitor Control Tables...**.
4. Select the database where you want to create the monitor control tables, and then select **OK**.

The monitor control tables are divided into three tablespaces by default: one for IBMSNAP_ALERTS, one for IBMSNAP_MONTRACE and IBMSNAP_MONTRAIL, and one for the rest of the control tables. The purpose of each table is briefly described in Table 7-3. This is followed by Table 7-4 presenting calculations to assist you with sizing the monitoring table's table spaces.

Table 7-3 Monitor control tables

Table name	Description
IBMSNAP_ALERTS	All alerts issued by the Replication Alert Monitor are recorded.
IBMSNAP_CONDITIONS	Contains conditions to contact on occurance.
IBMSNAP_CONTACTGRP	Contains individuals that make up contact groups.

Table name	Description
IBMSNAP_GROUPS	Contains the contact groups and their description.
IBMSNAP_MONENQ	Future use.
IBMSNAP_MONPARMS	Monitor parameters.
IBMSNAP_MONSERVERS	Information on which control servers where monitored and when.
IBMSNAP_MONTRACE	Every action of the monitor is recorded here.
IBMSNAP_MONTRAIL	Contains information about each monitoring cycle.

Table 7-4 Tablespace sizing for monitor control tables

Monitor control table	Row length	Number of rows
IBMSNAP_ALERTS	1172	1 row for each alert detected. Rows are eligible for puning based on ALERT_PRUNE_LIMIT.
IBMSNAP_CONDITIONS	372	1 row for each condition defined
IBMSNAP_CONTACTGRP	254	1 row for each contact group
IBMSNAP_CONTACTS	1415	1 row for each individual contact
IBMSNAP_GROUPS	1151	1 row for each contact group
IBMSNAP_MONENQ	18	none
IBMSNAP_MONPARMS	48	1
IBMSNAP_MONSERVERS	2570	1 row for each monitored server
IBMSNAP_MONTRACE	207	1 row for each Alert Monitor message. Rows must be pruned manually.
IBMSNAP_MONTRAIL	1115	1 row for each monitor cycle (MONITOR_INTERVAL). Rows must be pruned manually.

7.3.2 Create contacts

Unfortunately, the Replication Center does not use the contact information that you may have added during installation or later to the health center. To add contacts and groups for replication alerts:

1. Within RC double click on **Operations** so that the folders for **Capture Control Servers**, **Apply Control Servers**, and **Monitor Control Servers** are showing.
2. Double click **Monitor Control Servers**.
3. Double click the monitor control server are interested in creating contacts on.
4. Select **Contacts**.
5. From the menu bar at the top of the window choose **Selected —> Create Contact —> Person...**.
6. In the *Create Contact* enter a name, e-mail address, and, if you desire, a description. There is also a check box for *Address is a pager* which will result in pager friendly messages being generated.
7. Selecting **OK** generates the SQL.
8. In the *Run Now or Save SQL* window select **OK**. See Section 2.13, “Run Now or Save SQL” on page cxix if you desire assistance with this window.
9. The contact will now be listed in the right pane of the main *Replication Center* window.

If you want different people to receive notification for specific events, then repeat the steps just described. If you want multiply people to receive notification for a particular event after adding these persons, select **Group...** instead of **Person...**. Create a group and add the persons that you have previously created.

7.3.3 Alert Conditions

Now that we have a person to notify, we can consider the conditions that warrant notification. The alert conditions belong to three categories:

- Status of the Capture and Apply programs
- Thresholds being exceeded
- Miscellaneous events

Attention: You must have a contact created to setup alert conditions.

To select alert conditions:

1. Within RC double click on **Operations** so that the folders for **Capture Control Servers**, **Apply Control Servers**, and **Monitor Control Servers** are showing.
2. Double click **Monitor Control Servers**.
3. Double click the monitor control server are interested in creawting contacts on.

4. Select **Monitor Qualifiers**.
5. From the menu bar at the top of the window choose **Selected —> Select Alert Conditions for Capture Schemas** or **Select Alert Conditions for Apply Qualifiers or Subscription Sets.**
6. Whichever you select, the windows are almost identical. The *Select Alert Conditions for Capture Schemas* window is shown in Figure 7-4.

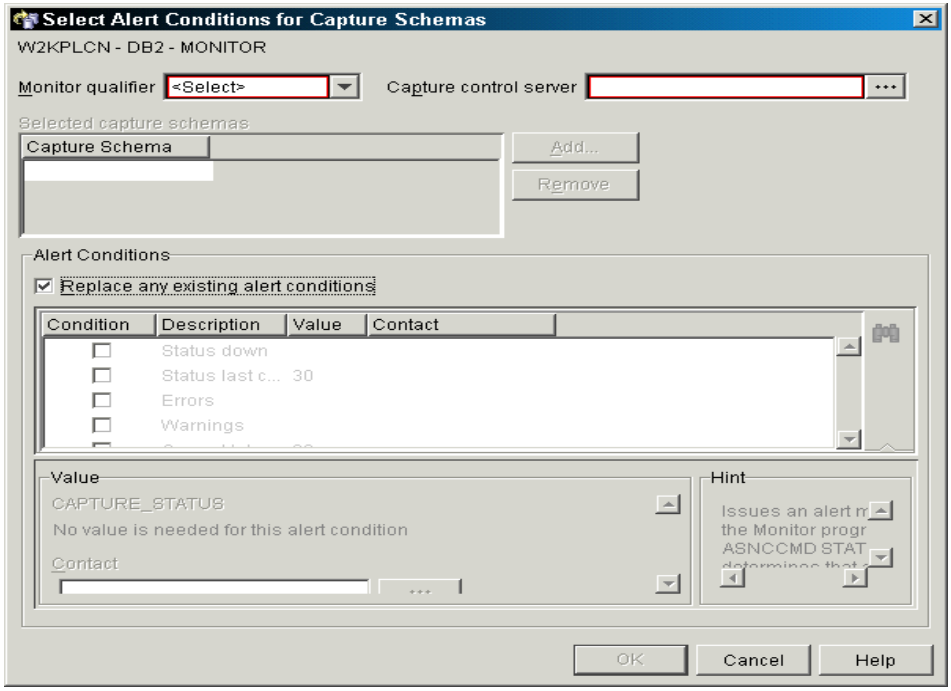


Figure 7-4 Select alert condition for capture

7. You have to select an existing qualifier or enter a new qualifier.
8. Select a capture control server by clicking **...**.
9. **Add...** capture schemas.
10. By default it will replace any existing alert conditions. If this is not desired uncheck that box.
11. Select alert conditions, identify a value as needed, and decide who is contacted. The *Hint* provides sufficient information about the majority of the alert conditions and their possible values. It is worth noting while considering the conditions that the monitor runs on an interval specified by `monitor_interval` at monitor start. Some of the conditions will be better grasped after reading the later chapters.

Attention: The Replication Alert Monitor does not monitor triggers associated with non-DB2 relational databases used as sources in a federated database system.

12. Select contacts. This is contained in the *Values* section. It may be hidden from view, scroll that portion if this is the case.
13. Select **OK** to generate an SQL script.
14. In the *Run Now or Save SQL* window select **OK**. See Section 2.13, “Run Now or Save SQL” on page cxix if you desire assistance with this window.

7.3.4 The replication monitor program

Now in the main *Replication Center* window the monitor qualifier will be listed. Selecting it will show all of the capture schemas and apply qualifiers or subscriptions sets that have alert conditions. The ability to modify an alert is very limited, and if you do create many alerts to workaround this, it become unwieldly.

With the monitor qualifier selected, choosing **Selected** with show you the new options of **Show Alerts**, **Start Monitor**, **Stop Monitor**, and **Reinitialize Monitor**. These options behave the same as the corresponding options for Capture and Apply. You can start the monitor from the command line using **asnmcmd**. **Reinitialize Monitor** will load the changed values for that monitor qualify. This is unlike Capture and Apply, Monitor does not dynamically collect new and changed parameters and alert conditions.

7.3.5 Receiving an alert

When an alert condition is encountered, the Monitor will attempt to notify you using the SMTP `email_server` parameter. It will attempt to reach you each time a paticular alert condition is encountered up to `max_notifications_per_alert` within `max_notifications_minutes`. These values defaults are three times within one hour. Example 7-4 shows a received email. If you set up the contact as a pager then the message will be even more susinct.

Example 7-4 Email notification

```
To:      ldbudd@ca.ibm.com
From:    dpmon@server.com
Subject: Monitor: MONQUAL: Alerts issued
Monitor: MONQUAL: This is a report from the Replication Alert Monitor
on server MONDB.
2002-01-01-03.00.00 ASN5152W Capture latency exceeded threshold.
Server is "CORP". Capture Schema is "ASN". Capture latency "720"
seconds, threshold "600" seconds.
```

```
2002-01-01-03.10.20 ASN5056W Memory used by Capture exceeded threshold.
Server is "CORP".
Capture schema is "ASN". Current Memory is "26", Threshold is "24".
```

7.3.6 Replication monitoring example

We will set up and run a simple scenario to show the configuration and operation of the DB2 Replication version 8 Alert Monitor.

In Replication Center we created monitor control tables co-located with an apply control server TGT_NT, and then used Replication Center to create contacts and then alert conditions for both Capture and Apply. The entries for these two groups of alerts are shown in in Figure 7-5.

Figure 7-5 Capture and Apply alert entries in Replication Center

We can highlight either the C (Capture component alert grouping) or the A (Apply component alert grouping), right-mouse click, and select **Properties** to see the alert conditions in place. Figure 7-6 is an example of the apply alert conditions properties, showing the apply alert conditions that we set using the *Select Alert Conditions for Apply Qualifiers or Subscriptions* dialog.

Figure 7-6 Alert conditions for Apply

We have set a low latency threshold on an alert condition for when full-refreshes occur to force the generation of an alert. We can see our alert conditions by querying the ASN.IBMSNAP_CONDITIONS table at the monitor control server. Example 7-5 shows our query. The columns of the conditions table are described in detail in the *DB2 UDB Veresion 8 Replication Guide and Reference,* SC27-1121-00, chapter “Table structures”.

Example 7-5 Query of monitor conditions table

```
SELECT MONITOR_QUAL, SERVER_ALIAS,
COMPONENT, SCHEMA_OR_QUAL, SET_NAME,
CONDITION_NAME, PARM_INT, PARM_CHAR, CONTACT
FROM ASN.IBMSNAP_CONDITIONS
WHERE MONITOR_QUAL='MIXMON'
```

Figure 7-7 shows the result of this query displayed in DB2 Command Center.

Figure 7-7 Monitor conditions displayed in Command Center

We can then start the alert monitor to check for alert conditions at our capture and apply control servers. We force Apply to do a full-refresh without stopping Capture or Apply. This can be done in Replication Center. In the left pane we double click on **Replication Definitions —> Apply Control Servers —> *ourControlServer* —> Subscription Set**. We highlite the subscription set MIXSET in the right pane. Then choose from the menu bar **Selected —> Full Refresh —> Automatic**. The SQL that is generated, when run, connects to the apply control server and sets the last success, synch time, and synch point of a set each to NULL, causing a full-refresh.

Before starting Replication Monitor, we first designate a working directory and copy or create a asnpwd type password file containing the userids and passwords the Replication Monitor needs to connect to the capture and apply control servers.

On Windows, for a Monitor working directory, we create a directory:

```
mkdir d:\DB2Repl\MIXMON.
```

At the command prompt we **cd** to that directory and use **asnpwd** to create a password file MIXMON.aut containing userid's and passwords to connect to SRC_NT and TGT_NT, our capture and apply control servers respectively. **asnpwd** is described in Chapter 6, "Operating Capture and Apply" on page cclxv. **asnpwd** has online help which can be viewed from a command prompt by entering **asnpwd -help**. There is additional information in the *DB2 UDB Version 8 Replication Guide and Reference,* SC27-1121-00, chapter "System commands for replication (UNIX, Windows, z/OS)".

Alert Monitor can be started from the Replication Center. To do this, we click on the icon for a monitor qualifier, right mouse click, and select **Start Monitor** from among the options. Figure 7-8 shows the *Start Monitor* window.

Figure 7-8 Start Monitor window example in Replication Center

We start alert monitor with `RunOnce=Y` instead of specifying a `Monitor_Interval` since we just want to collect any alerts that might have been generated by a short run of Apply.

Besides using Replication Center, DB2 Replication Alert Monitor can be started at a command prompt. Below is the command entered it in this example:

```
d:\DB2Repl\MIXMON\asnmon monitor_server=TGT_NT monitor_qual=MIXMON
runonce=y monitor_path=d:\DB2Repl\MIXMON pwdfile=MIXMON.aut
```

In this example:

- `monitor_server=TGT_NT`

  TGT_NT is the DB2 server containing the monitor control tables.

- `monitor_qual=MIXMON`

  MIXMON is the monitor qualifier.

- `runonce=y`

  Instead of setting a regular `Monitor_Interval`, we tell monitor to just run once, then we check for the occurrence of alert conditions.

- `monitor_path=d:\DB2Repl\MIXMON`

  The working directory for this monitor.

- `pwdfile=MIXMON.aut`

  The asnpwd-type password file we have created in the monitor working directory for monitor to use to get the userid's and passwords to connect to the capture and apply control servers.

In this example, monitor runs for about one minute and then stops. In that time frame we also received email notification.

If we look in Replication Center at our monitor qualifier MIXMON, we notice in the right pane that the icon for apply qualifier MIXQUAL has a red light next to it indicating an alert, se Figure 7-9.

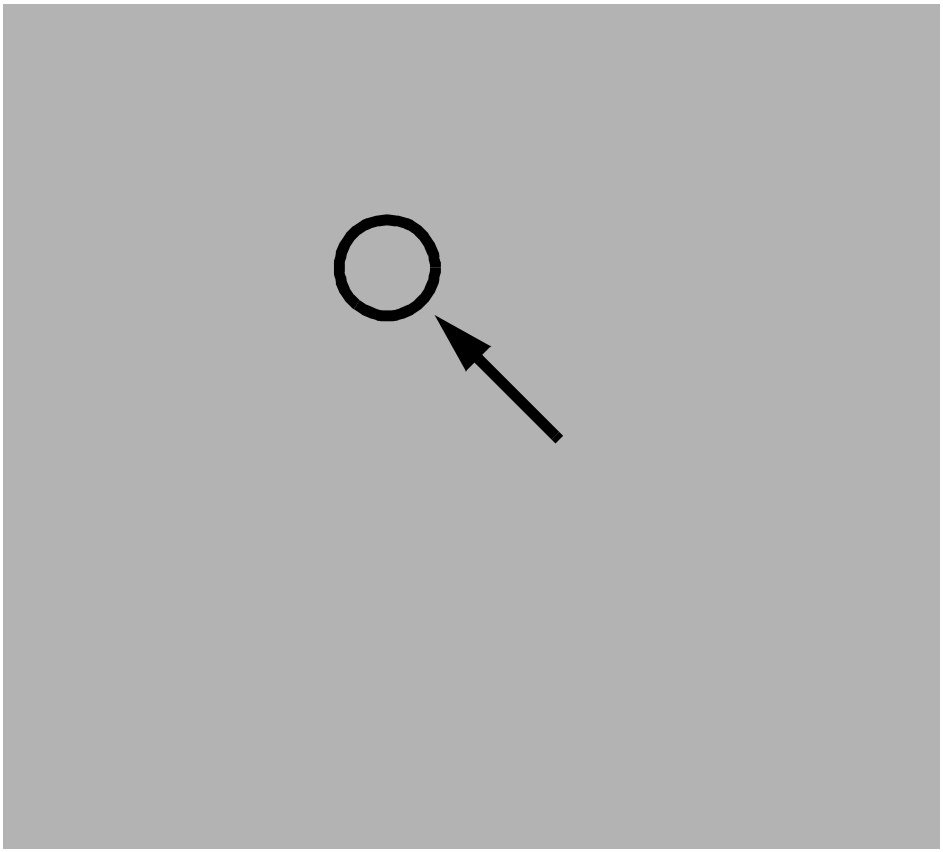


Figure 7-9 Monitor Alert identifier

After the run, we use query to look in the ASN.IBMSNAP_ALERTS table at the monitor control server (TGT_NT) to see if there are any records. We expect at least one record since we cold start Capture, which forces Apply to do a full-refreshes for all members.

We can display the alerts themselves in the Replication Center. We highlight the icon for MIXQUAL, right-mouse, and select **Show alerts.** In the *Show Alerts* window, we can specify a range of times for which we want to see alerts and then click **Retrieve**. Since we know we do not have many alerts, we accept the defaults. When we click **Retrieve**, the alerts are displayed, see Figure 7-10.

Figure 7-10 Alerts detail display in Replication Center

We could also query the ASN.IBMSNAP_ALERTS table at the monitor server to see the records for the alerts. Example 7-6 shows the query. Here we will use DB2 Command Center to do our query. Figure 7-11 shows the result of the query

Example 7-6 Query of Alerts

```
SELECT MONITOR_QUAL, COMPONENT, SERVER_ALIAS, SCHEMA_OR_QUAL,
SET_NAME, CONDITION_NAME, OCCURRED_TIME
FROM ASN.IBMSNAP_ALERTS WHERE MONITOR_QUAL='MIXMON'
```

Figure 7-11 Replication alert records in the ASN.IBMSNAP_ALERTS table

7.3.7 Using JCL to start monitoring on z/OS

If the monitor control server is DB2 UDB for z/OS, you have the alternative to start monitoring with JCL, see Example 7-7.

Example 7-7 Sample JCL to start monitor on z/OS

```
//ASNMON   EXEC PGM=ASNMON,
//  PARM='/monitor_server=V71A monitor_qual=MIXMON
//             monitor_path=//JAYAV8'
//STEPLIB  DD  DISP=SHR,DSN=DPROPR.V810.BASE.TESTLIB,
//             UNIT=SYSDA,VOL=SER=RMS002
//         DD  DISP=SHR,DSN=SYS1.SCEERUN
//         DD  DISP=SHR,DSN=DB2A.SDSNLOAD
//CEEDUMP  DD   DUMMY
//SYSTERM  DD   SYSOUT=*
//SYSUDUMP DD   DUMMY
//SYSPRINT DD   SYSOUT=*
```

ASNMON is a DB2 application program. This program must be bound to a DB2 subsytem which is the monitor server and to all monitored servers. See Chapter 6.1.4, “Considerations for DB2 UDB for z/OS” on page cccvi for the recommended bind parameters.

Monitor_server is the db2 subsystem id name of the monitor control server.

7.4 Other monitoring

DB2 includes a variety of other types of monitoring including snapshot and monitoring not dependent on replication monitoring control tables.

7.4.1 Examining historic data

This is also aspect of performance tuning and examined in Chapter 7, “Troubleshooting and monitoring” on page cccxxxvii. Historical data is derived from Apply trail (IBMSNAP_APPLYTRAIL), Apply trace (IBMSNAP_APPLYTRACE), Capture monitor (IBMSNAP_CAPMON), and Capture trace (IBMSNAP_CAPTRACE).

7.4.2 Health center

The health center is the Replication Alert Monitor for the non-replication elements of DB2 on Linux, UNIX, and Windows. Whether it uses tables or not is not transparent to us. The command syntax to configure it is UPDATE ALERT CFG FOR DATABASE ... . To access the *Health Center* window from RC or other main administration client windows, select **Tools —> Health Center**. The health center is shown in Figure 7-12. It seems to default to the three lit shapes. You will likely find your instances and databases clicking the four lit shapes button. Clicking on the instance and databases you can configure and start the health monitoring

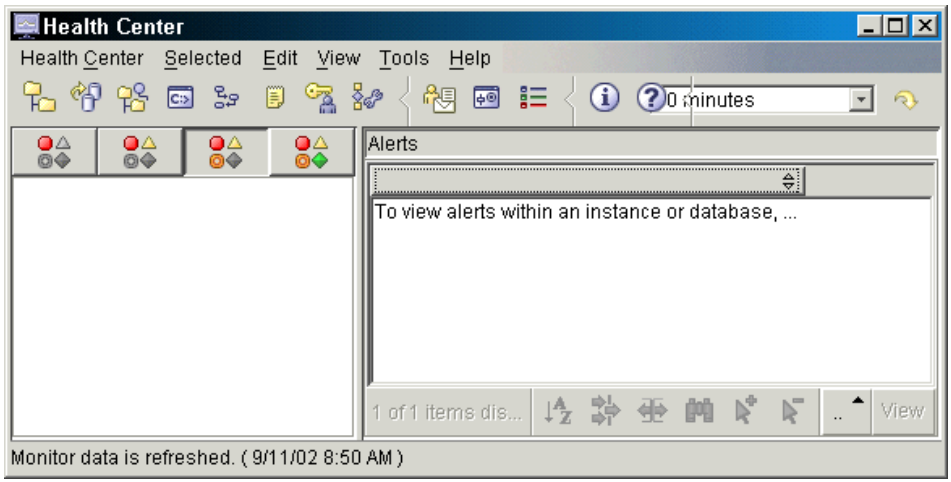


Figure 7-12 Health Center

7.4.3 System Monitoring

DB2 on Linux, UNIX, and Windows includes snapshots and monitoring documented in the *DB2 UDB System Monitor Guide and Reference Version 8*, SC09-4847-00.

7.5 Advanced troubleshooting

We have been informed that there is plans to include a troubleshooting section on the DataPropagator website:

http://www-3.ibm.com/software/data/dpropr/support.html

This is expected to be present within the first year of version 8 availability.

7.5.1 asnanalyze and ANZDPR

asnanalyze is described in the *DB2 UDB Version 8 Replication Guide and Reference,* SC27-1121-00, chapter “System commands for replication (UINX, Windows, z/OS)”. ANZDPR, an iSeries CL command, is described in the same document, in chapter “System commands for replication (OS/400)”.

asnanalyze, though it only runs on UNIX and Windows, can also obtain information from Replication Control tables and the DB2 system catalog on z/OS,

and OS/390. **asnanalyze** uses DB2 connectivity from the system it is run on to the DB2 servers from which it collects information.

The output of **asnanalyze** is an HTML file that can be read using a web browser.

Besides the information on asnanalyze in the *Replication Guide and Reference,* SC27-1121-00, **asnanalyze** also has online help. To view this help, at a command prompt, enter

```
asnanalyze
```

with no keywords or parameters specified , and the online help will be still be displayed.

asnanalyze example:

This example was run on Windows. It could also be run on Linux or UNIX.

DB2 connectivity has been defined and tested to both the capture control server (SRC_NT) and the apply control server (TGT_NT).

- Go to a directory where you want the HTML output

  ```
  cd d:\DB2Repl\analyze\
  ```

- create a password file using **asnpwd** to provide the userid's and password **asnanalyze** will need to connect to the capture control server (SRC_NT) and the apply control server (TGT_NT):
  - First, create the password file analyze.aut:

    ```
    asnpwd init using analyze.aut
    ```

  - Next, add an entry for the capture control server:

    ```
    asnpwd add alias src_nt id db2drs4 password db2drs4 using analyze.aut
    ```

  - Then, add an entry for the apply control server:

    ```
    asnpwd add alias tgt_nt id db2drs4 password db2drs4 using analyze.aut
    ```

- Run analyze for the capture schema MIXCAP and apply qualifier MIXQUAL, Example 7-8 shows the single command listed over multiple lines for clarity.

Example 7-8 asnanalyze command

```
asnanalyze -db SRC_NT TGT_NT
           -la standard
           -ct 2
           -cm 2
           -sg 2
           -tl 2
           -at 2
           -aq MIXQUAL
```

```
-cs MIXCAP
-od d:\DB2Repl\analyze
-fn MIXanlyz.htm
-pw analyze.aut
```

Explanation of input parameter values:

- -db SRC_NT - our Capture Control Server
- TGT_NT - our Apply Control Server
- -la standard - provide standard level of detail in the output
- -ct 2 - get CAPTRACE records for passed 10 days; default is 3 days
- -cm 2 - get CAPMON records for passed 10 days; default is 3 days
- -sg 2 - get SIGNAL records for passed 10 days; default is 3 days
- -tl 2 - get APPLYTRAIL records for passed 10 days; default is 3 days
- -at 2 - get APPLYTRACE records for passed 10 days; default is 3 days
- -aq MIXQUAL - Apply Qualifier to retrieve recrods for
- -cs MIXCAP - Capture Schema to retrieve records for
- -od d:\DB2Repl\analyze - output directory for analyze output file; default is current directory
- -fn MIXanlyz.htm - output file from analyzer
- -pw analyze.aut - the DProp password file containing userids/passwords for Analyze to use on connections to SRC_NT and TGT_NT; default is asnpwd.aut

Output messages to the terminal look like Example 7-9.

Example 7-9 Analyzer output to terminal

```
Analyzer Started....
Fetch completed for SRC_NT - Capture Control tables
No ASN.IBMSNAP_SUBS_SET table.
No ASN.IBMSNAP_SUBS_SET table.
No ASN.IBMSNAP_SUBS_MEMBR table.
No ASN.IBMSNAP_SUBS_COLS table
No ASN.IBMSNAP_SUBS_STMTS table.
No ASN.IBMSNAP_SUBS_EVENT table.
No ASN.IBMSNAP_APPLYTRAIL table.
No ASN.IBMSNAP_APPLYTRACE table.
Fetch completed for SRC_NT - Apply Control tables
No ASN.IBMSNAP_CAPSCHEMAS table.
Probably the capture control tables are of Pre V8 archietecture please
try with
```

```
 Pre V8 ANALYZER -> ANALYZE
Fetch completed for TGT_NT - Capture Control tables
Fetch completed for TGT_NT - Apply Control tables
Broad analysis started....
Compare CD vs Source Column completed
Compare CD vs Source Column completed
Check Index completed
Check Subscription completed
Analyzer run completed. Total time taken is 0 minutes - 2 seconds.
File d:\DB2Repl\analyze/MIXanlyze.htm contains the output of the
Analyzer.
```

We can open the output file (MIXanlyze.htm) with a web browser. At top the them HTML document is a table of contents, see Example 7-10. Each entry in the HTML document is a link to the respective section of the output.

Example 7-10 Analyser output table of contents

```
Table Of Contents
SRC_NT Control table detail
SRC_NT Packages and plans (link not available if OS/400)
SRC_NT Change data table (CD) column analysis
SRC_NT Internal consistent change data table (CCD) column analysis
SRC_NT Subscription target key synopsis
SRC_NT Federated DB nickname details from SYSIBM.SYSCOLUMNS (link not available
if OS/400)
TGT_NT Control table detail
TGT_NT Packages and plans (link not available if OS/400)
TGT_NT Change data table (CD) column analysis
TGT_NT Internal consistent change data table (CCD) column analysis
TGT_NT Subscription target key synopsis
TGT_NT Federated DB nickname details from SYSIBM.SYSCOLUMNS (link not available
if OS/400)
Server connection summary
How many rows in each CD table are eligible for pruning
How many rows in each unit of work table are eligible for pruning
Selected DB2 for z/OS table,tablespace information
Selected DB2 Common Server, Febderated, workstation UDB tablespace information
Missing table or view references
Inconsistency - Orphan Details
Incorrect or inefficient indexes
Incorrect or inefficient tablespace LOCKSIZE
Subscription errors, omissions or anomalies
Apply process summary
When your CAPTURE program is not capturing
CAPTURE tuning troubles
Additional Findings
```

7.5.2 DB2 Replication trace

asntrc can be run on Linux UNIX, Windows, and z/OS. , is described in the *DB2 UDB Version 8 Replication Guide and Reference,* SC27-1121-00, chapter "System commands for replication (UNIX, Windows, z/OS)". **asntrc -help** provides online help. WRKDPRTRC is described in the same document in the chapter "System commands for replication (OS/400)".

Below we show one an of running **asntrc**. There is another example of using **asntrc** in Section 10.4.13, "Time spent on each of Apply's sub-operations" on page cdlxxxiii. That example is specific to finding Apply TRACE PERFORMANCE RECORDS.

asntrc tips

There could be several Capture, Apply, and Monitor processes running simultaneously on a system, every time we enter an **asntrc** command, we need to specify:

- -db parameter with a value indicating the control server database for that Capture, Apply, or Monitor process being traced.
- -cap, -app, or -mon to indicate whether the command is for a Capture, Apply, or Monitor process,
- -schema or -qualifier with a value indicating the the Capture Schema, Apply Qualifier, or Monitor Qualifier of the process being traced

The above parameters need to be specified when we do any of the following **asntrc** commands:

- on - turning on asntrc for a Capture, Apply, or Monitor process
- off - turning off anstrc
- clr - clear the memory buffers of asntrc
- kill - kill an asntrc that can't be stopped with 'off'
- dmp - dump the current asntrc buffers to a file
- fmt - format the current asntrc buffers to a file
- v7fmt - format the current asntrc buffers to a file, in the format of a Capture/Apply V5/6/7 trace
- flw - to format current asntrc buffers in abbreviated format

For dump, specify the output filename after the keyword **dmp**.

For **fmt**, **v7fmt**, and **flw**, at the end of the command, also specify **>** and an output file name.

Attention: If you are tracing a Capture, Apply, or Monitor process that you suspect will stop abnormally, when you start **asntrc** (**asntc on**), you will want to specify the **-fn** parameter followed by an output filename. This will force asntrc to continuously dump its buffer. This can have a negative impact on performance, but otherwise the trace may not be dumped succesfully.

The above is also true when tracing an Apply that is started with `CopyOnce='Y'`, since it will also release its memory when it stops, including any **asntrc** buffers.

The contents of the file will be in **asntrc**'s dump. **asntrc fmt**, **v7fmt**, or **flw** can then be used to create a file containing formatted **asntrc** output.

When formatting the contents of an existing **asntrc** dump output file, use **-fn** *filename* to indicate the **dmp** file being formatted, and **>***outfilename* to indicate the name of the formatted output file to create.

7.5.3 asntrc example for Apply

In this example we will show using **asntrc** to obtain a trace for a specific Apply.

Capture is already running.

Apply was started with the following command:

```
D:\DB2Repl\MIXQUAL\asnapply control_server=TGT_NT apply_qual=MIXQUAL
apply_path=d:\DB2Repl\MIXQUAL pwdfile=MIXQUAL.aut
```

This apply qualifier has only one subscription set. We start **asntrc** with the following command:

```
asntrc on -db TGT_NT -app -qualifier MIXQUAL -b 5M
```

The parameters we supplied are:

- -db TGT_NT: the Apply Control Server
- -app: trace Apply
- -qualifier MIXQUAL: the apply qualifier of the Apply process we're tracing
- -b 5M: the asntrc buffer is 5 MegaBytes

Note: If we were tracing an Apply that we suspect may terminate abnormally, we would start **asntrc** before starting Apply. Also, we would have specified a filename for trace records to ensure the buffers will be written to disk. **asntrc** will be started with:

```
asntrc on -fn MIXQUAL.dmp -db TGT_NT -app -qualifier MIXQUAL
```

We watch the ASN.IBMSNAP_APPLYTRAIL table looking for a new record with the Apply Qualifier of the Apply process we are tracing. Then the asntrc buffers should contain the information from one complete cycle. We check the APPLYTRAIL table with the following query:

```
SELECT COUNT(*) FROM ASN.IBMSNAP_APPLYTRAIL WHERE
APPLY_QUAL='MIXQUAL'
```

The result of this query indicates that there is a new apply trail record. We dump the contents of the asntrc buffers to a file.

```
asntrc dmp MIXQUAL.dmp -db TGT_NT -app -qualifier MIXQUAL
```

In this example:

- dmp MIXQUAL.dmp: the output file

Note: If we had wanted to bypass the dump file and just format the asntrc buffers, we might have used:

```
asntrc v7fmt -db TGT_NT -app -qualifier MIXQUAL > MIXQUAL.v7fmt
```

We can turn off asntrc now since, one way or another, we've written the trace records out to a file. We stop **asntrc** with the following command:

```
asntrc off -db TGT_NT -app -qualifier MIXQUAL
```

If we just dumped the asntrc records to a file, and didn't format them yet, we can create a file with the formatted trace output with the following command:

```
asntrc v7fmt -fn MIXQUAL.dmp > MIXQUAL.v7fmt
```

In this example:

- -fn MIXQUAL.dmp: the input file to asntrc v7fmt
- > MIXQUAL.v7fmt: the output file in the format of a Apply V5/6/7 trace

7.5.4 asntrc example for Capture

In this example we will show the comands to trace Capture.

Capture was started with the following command:

```
asncap capture_server=SRC_NT capture_schema=MIXCAP
capture_path=D:\DB2Repl\MIXCAP startmode=warmns
```

We turn **asntrc** on with the following command:

```
asntrc on -db SRC_NT -cap -schema MIXCAP -b 5M
```

In this example:

- -db SRC_NT: the Capture Server
- -cap: indicates we are tracing a Capture process
- -schema MIXCAP: the Capture schema of the Capture process being traced
- -b 5M: the asntrc buffer is 5 MegaBytes

Note: As with the asntrc for Apply, if we suspect Capture might terminate abnormally, we turn **asntrc** on before starting Capture. When we start asntrc we include **-fn *filename*** so that asntrc would write all trace records to a file:

```
asntrc on -fn MIXCAP.dmp -db SRC_NT -cap -schema MIXCAP
```

We let Capture run through whatever event we want to trace. We update one of the registered tables and issued a commit so that Capture has a transaction to track in memory. Then Caputre inserts records into the CD table and the unit of work table. We could query the CD table to see when Capture has inserted new records.

We then dump the asntrc records to a file, with the following command:

```
asntrc dmp MIXCAP.dmp -db SRC_NT -cap -schema MIXCAP
```

We can turn the trace off with the following command:

```
asntrc off -db SRC_NT -cap -schema MIXCAP
```

And then we created a formatted trace file from the contents of the dump file, with the following command:

```
asntrc v7fmt -fn MIXCAP.dmp > MIXCAP.v7fmt
```

8

Maintaining Your Replication Environment

In this chapter we will discuss how to maintain your replication environment. After implementing your replication configuration, there will, no doubt, continue to be structural changes to your source or target tables specifications. Therefore, we will show you how to reflect these changes to your replication environment, with minimal impact to your day to day replication. We will also discuss maintaining capture and apply control tables to maintain optimum replication performance on the specific platform that is processing either the capture or apply. Also briefly look into system maintenance that could impact your replication environment.

The following topics will be discuss:

- Maintaining registrations
- Maintaining subscriptions
- Promoting replication definitions to another system
- Maintaining replication control tables
- Full refreshing target tables

8.1 Maintaining Registrations

The following section will discuss maintaining registrations sources after implementing your initial replication configuration. As your application progresses with continuous use, more request will come from the users to make changes to the current database design. These changes at the source server will effect the replication registration, as follows:

8.1.1 Adding new registrations

Adding new registered tables to a current replication configuration can occur at any time with out impacting your current capture operations. You not need to stop and start Capture or re-initialize Capture. A new registration is inactive until it is accessed by a subscription. The Apply program issues a CAPSTART signal to Capture the first time a source table is used for a subscription set. This signal tells Capture to activate the registration and begin capturing changes. , “Capture starts capturing changes” on page lxi describes the signal process. The steps to add a new registered table are described in Chapter 4., “Replication Sources” on page clxix.

8.1.2 Deactivating and activating registration table

The state of a registration is kept in the STATE column of *capschema*.IBMSNAP_REGISTER. The state can have one of these three values:

- I — inactive, no CAPSTART signal yet
- A — active, capturing changes
- S — stopped, deactivated

Deactivating registration tables

Before deleting or changing a registered table attribute, it is a good practice to deactivate the registered table. This will ensure that the Capture program has completed processing all of its captured entries. Another reason to deactivate a register table is to temporary stop capture for a particular registered table, and continue capturing the changes for the other registered tables.

When you are deactivating a registration table, the Capture program will only stop capturing changes. The CD tables, subscriptions sets, registration attributes are still defined within your replication environment.

All subscription sets associated with the deactivated registered table needs to be deactivated also, just in case the apply program reactivate the registered table while you are in the process or deleting or making changes or before you are

ready to reactive it. These subscription sets will be effected when it encounters a deactivated registered table and Capture is no longer processing it changes. Therefore, removing the subscription member that is associated with the deactivated register table will allow you to continue processing those subscription set.

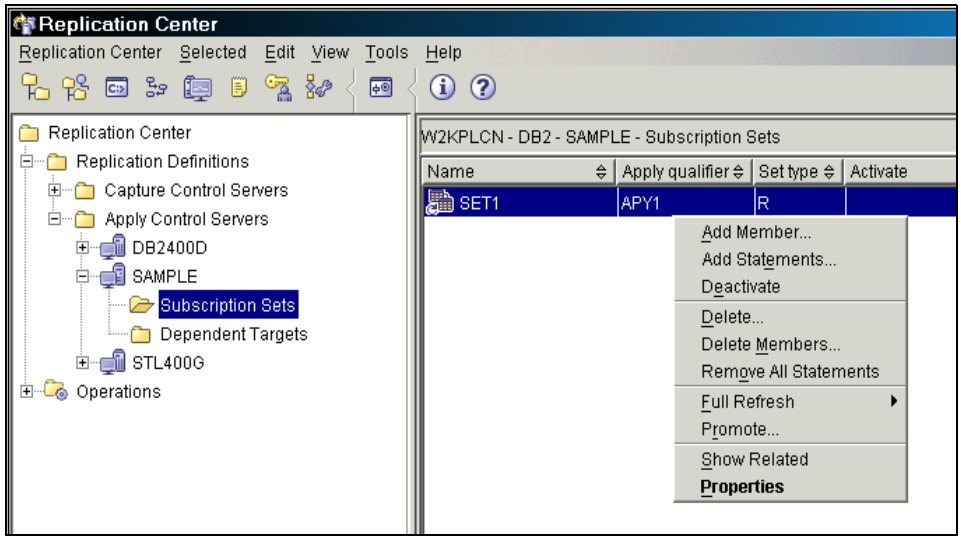


Figure 8-1 Deactivate Subscription Set

There are two methods to deactivate a registration table

- Using the replication center:

  First you need to deactivate all the associated subscription sets. Figure 8-1 shows how to deactivate a single subscription set:

Expand the **replication Definition**—>**Apply Control Servers** folder—>The Apply control server**—>Subscription Sets** folder (Right click to filter).Right click the actual subscription set—>**Deactivate**

The next step is to deactivate the registration table, see Figure 8-2:

Expand the **replication Definition**—>**Capture Control Servers** folder—>The Capture control server—> Registration folder (Right click to filter).Right click the actual subscription—>**Stop Capturing Changes**

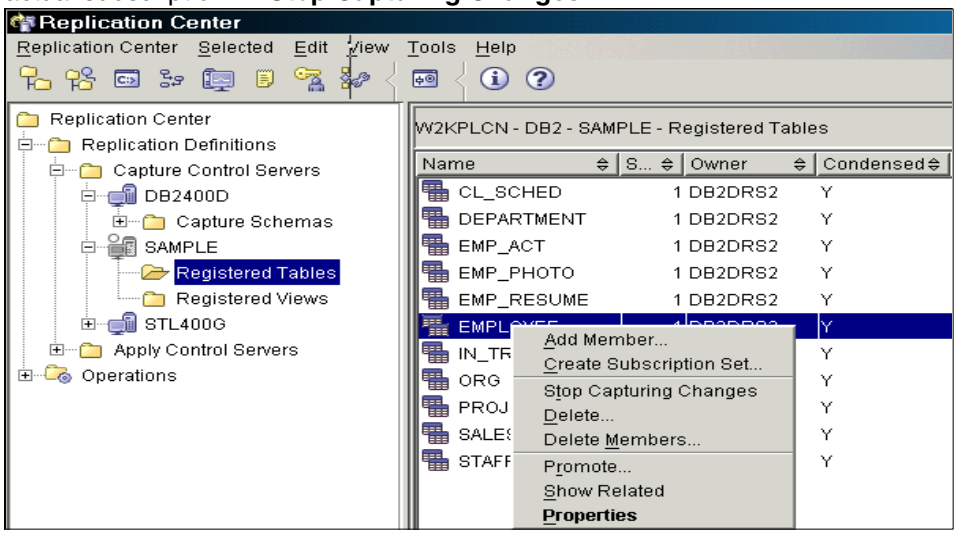


Figure 8-2 Deactivating registration tables

When you click the option to stop capturing changes, the IBMSNAP_SIGNAL table is updated with a CAPSTOP in the SIGNAL_SUBTYPE column and a 'P' value for the SIGNAL_STATE, indicating this signal is pending and during the next cycle when the capture program is running, this registration will become inactive. The IBMSNAP_REGISTER table will have a 'I' value in the STATE column.

- Manually inserting the CAPSTOP signal in the IBMSNAP_SIGNAL table as shown in

Example 8-1 Manually deactivating a registration

```
CONNECT TO capture control server
INSERT INTO capschema.IBMSNAP_SIGNAL
   (SIGNAL_TIME,SIGNAL_TYPE,SIGNAL_SUBTYPE,SIGNAL_INPUT_IN,
    SIGNAL_STATE,SIGNAL_LSN)
VALUES
   (CURRENT TIMESTAMP,'CMD','CAPSTOP','sourceschema.sourcetable',
    'P',NULL)
```

Activating registrations tables

When your registration is temporarily inactive and you are ready to reactivate your registration to resume capturing, all you need to is activate the associated subscription sets using the replication center, see Section 8.2.2, "Deactivating

and activating subscriptions" on page ccclxxxi. Then after you start the Apply program, it will update the IBMSNAP _SIGNAL table with a CAPSTART subtype signal, which will tell the Capture program to reactive the registration.

The Capture program could have deactivated the registration, because of an unexpected error during the capture process. When this occurs the following conditions happens:

- The STATE column value is set to an 'S' in the IBMSNAP_REGISTER table if the STOP_ON_ERROR column value is set to an 'N'. See , "Stop on error" on page clxxxi for details about this setting.
- When STATE column is set to 'S', Capture will no longer capture changes for this registration until the problem is resolved.

To activate the registration, which was deactivated because of the unexpected error during the Capture program to need to do the following:

- Resolve the problem that causing there errors in Capture program, so that registration is eligible to be activated.
- At the Capture control server run the SQL statement in Figure 8-3 to reset the STATE column value in the IBMSNAP_REGISTER table:

```
UPDATE Schema .IBMSNAP_REGISTER SET STATE ='I '

WHERE

SOURCE_OWNER = SrcSchema AND SOURCE_TABLE ='SrcTbl 'AND
SOURCE_VIEW_QUAL =SrcVwQual AND STATE ='S ';
```

Figure 8-3 SQL to activate registration after unexpected capture errors

Schema is the name of the Capture schema, *SrcSchema* is the registered source table schema, *SrcTbl* is the name of the registered source table, and *SrcVwQual* is the source-view qualifier for this source table. After the STATE column is set to I (Inactive), the Capture program will start capturing data when the CAPSTART signal is received, from the Apply program.

Attention: Since capturing was stopped for this registration, there may have been changed in the log that were missed. Capture does not go back in the log or journal to look for these changes. Setting the STATE to 'I' forces a full refresh of all subscriptions sets that copy from this source table.

8.1.3 Removing registrations.

When you remove a registration the following items are effected:

- The registration entry is removed from the IBMSNAP_REGISTER table.
- For DB2 sources, the corresponding CD table is dropped.
- For Informix sources, the CCD table, CCD nickname, Capture triggers and Capture stored procedures are dropped.

The actual source table or view are not effected in any way. They still remain on the source server.

The remove a registration, make sure you deactivate it first, see Section 8.1.2, “Deactivating and activating registration table” on page ccclxxiv. Then from the replication center, see Figure 8-2 on page ccclxxvi to display the menu, select **Delete** option to display the registered tables you want to remove, see Figure 8-4:

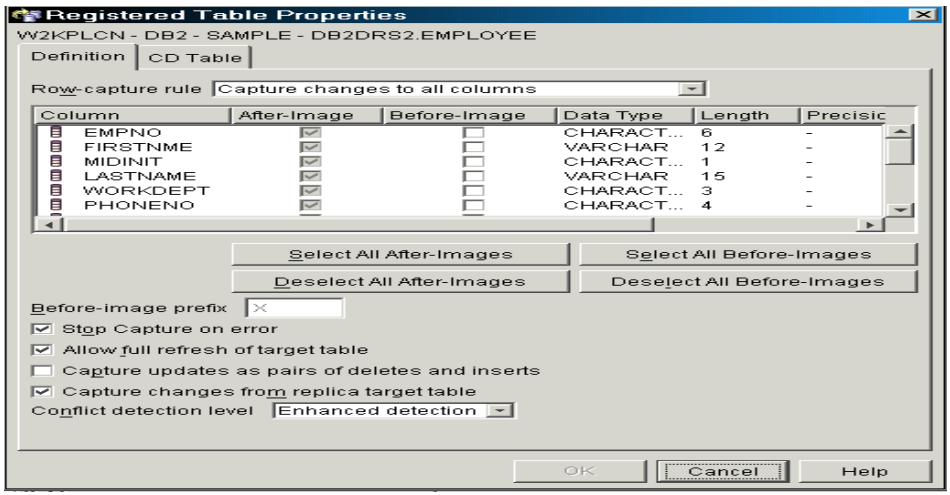


Figure 8-4 Removing registrations

Click **OK** to display the Run Now or Save SQL screen.

If your registered tables reside on iSeries server, then you have an option to remove your registrations using the RMVDPRREG CL command. To use this command enter it on the iSeries command line and press the F4 key, then press the F11 key to display the actual parameter names, see Figure 8-5:

```
Remove DPR Registration (RMVDPRREG)

Type choices, press Enter.

Capture control library  . . . . CAPCTLLIB      ASN
Source table . . . . . . . . . . SRCTBL


  Library  . . . . . . . . . . .
```

Figure 8-5 iSeries CL command to remove registrations

For field level help move the curser to the parameter and press the F1 key. Or refer to Chapter 18 in the *Replication Guide and Reference,* SC27-1121-00

8.1.4 Changing capture schemas

You may find that, for performance reasons, you wish to create a new capture schema and move some of your current registrations and subscriptions to the new schema. Refer to Chapter 12, “Making changes to your replication environment” in *IBM DB2 Universal Database Replication Guide and Reference, SC27-1121-00,* for the steps needed to change capture schemas.

8.1.5 Changing registration attributes for registered tables

There are limited changes that can be made to an existing registered table, which can be performed at any time. These changes can be made from the replication center. See, Figure 8-2 on page ccclxxvi and click **Properties** from the menu to display the window in Figure 8-6:

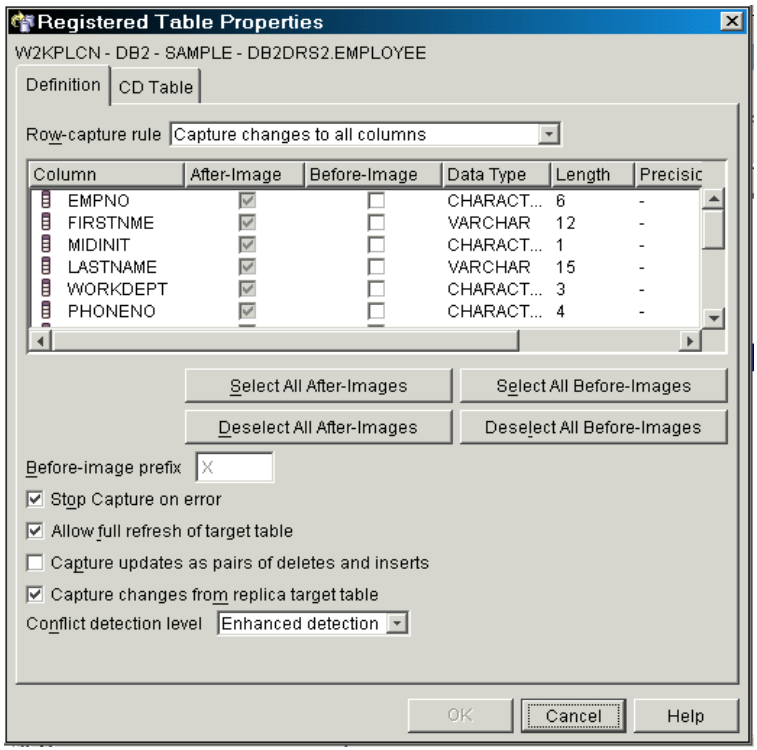


Figure 8-6 Registered table properties

The changes that are made to an existing registered tables, as indicated from this window are updated in the IBMSNAP_REGISTER table. The following are the columns that are changed:

- **Row capture rule** — CHGONLY
- **Before image prefix** — BEFORE_IMG_PREFIX

Restriction: You cannot change an existing BEFORE_IMG_PREFIX. You can use this screen to add a before image prefix if you entered blank for this field when the table was originally registered.

- **Stop Capture on error** — STOP_ON _ERROR
- **Allow full refresh of target tables** — DISABLE_REFRESH
- **Capture updates as pairs of deletes and inserts** — CHG_UPD_TO_DEL_INS
- **Capture changes from replica targets tables** — RECAPTURE
- **Conflict detection level** — CONFLICT_LEVEL

Detail information on these columns is found in Chapter 4., “Replication Sources” on page clxix.

8.2 Maintaining Subscriptions

A replication environment is never a static solution, development is usually on-going, especially in a Business Intelligence application. The target tables definitions are always changing, as per the users demands.

This section will describe the on-going maintenance to the subscriptions, within your replication environment, after the initial implementation. Making sure these changes have minimum impact on the day to day replication operations. Enabling you to, cater to your end user replication request and demands.

Any maintenance that are made to the registered source table, has to be reflected on the associated subscription set.

8.2.1 Adding new Subscriptions Sets

Adding new subscription can be accomplished at any time without impacting your current replication environment. Need to make sure that source table are registered at the source server, see Chapter 5, “Subscription Set” on page cciii.

8.2.2 Deactivating and activating subscriptions

Deactivating a subscription set will only suspend the Apply program from processing any changes after it complete its current processing cycle. All the entries in the subscription set and members tables are still there. Deactivating a subscription set is accomplished from the replication center. See Figure 8-1 on page ccclxxv. From the same menu to deactivate a subscription, you can activate it by clicking the **Activate** option, then select **One Time Only** or **Indefinitely**

There are 2 periods of deactivation to observe that require additional processing.

Depending on how long your subscription is deactivated, will determine if you need to run these additional steps to prevent system performance problems in the future.

Short period of Deactivation

- A short period is determine if you want to suspend apply, so that you made changes to the subscription set or member. Then after the change you made you will immediately activate the subscription set and then resume processing.

- Or may be an error occurred during a particular apply cycle and while you are resolving the unexpected error, you can deactivate the subscription to prevent other changes to be applied, while you are resolving the problem. When you have fixed the problem then you can active the subscription and resume processing.

Long period of Deactivation

If for some reason you need to deactivate the subscription for a long period of time and you have no idea when you are going to resume processing, then you must consider the following situation as a caution for system and Apply program performance problems.

The Capture program pruning process will retrieve information from active Apply programs. Therefore, when the subscription sets is deactivated for a long period of time, the pruning information at that point in time becomes old and the UOW and CD tables are not prune as quickly and efficiently for active registrations that are associated with an inactive subscription. This old pruning information could degrade the performance of remaining active subscriptions and degrade system performance, by using CPU cycles to perform unnecessary pruning. Keep in mind, depending on the amount data being capture, the retention limit (normally set to the default value of 7 days) could initiate the pruning process to the UOW and CD tables, which could also impact system and Apply processing performance.

To prevent these unnecessary pruning process, you can run the following SQL to reset the pruning information for subscription sets that is inactive for a long period of time:

```
UPDATE CaptureSchema.IBMSNAP_PRUNE_SET
SET SYNCHPOINT = x '00000000000000000000'AND
SYNCHTIME = NULL
WHERE
APPLY_QUAL ='ApplyQual'AND
SET_NAME ='SetName'AND
TARGET_SERVER ='TargetServer';
UPDATE CaptureSchema.IBMSNAP_PRUNCNTL
SET SYNCHPOINT =NULL AND
SYNCHTIME =NULL
WHERE
APPLY_QUAL ='ApplyQual'AND
SET_NAME ='SetName'AND
TARGET_SERVER ='TargetServer';
```

Figure 8-7 SQL to reset pruning for deactivated subscription sets

Run this SQL at the capture server to reset the pruning information in the IBMSNAP_PRUNE_SET and IBMSNAP_PRUNCNTL control tables for deactivated subscriptions sets.

When a subscription set is deactivated, you should deactivate all associated registered tables, to prevent capturing data you don't need.

8.2.3 Changing Subscription sets

Subscription set name

The following steps will assist you to change the name of a subscription set, without having to remove and then recreate it with a different subscription set name, along with all associated members.

1. First, make sure the subscription set is deactivated, see Section 8.2.2, "Deactivating and activating subscriptions" on page ccclxxxi
2. From the Apply control server run the SQL shown in Figure 8-8 to change the subscription set name in the IBMSNAP_SUBS_SET, IBMSNAP_SUBS_MEMBER and IBMSNAP_SUBS_COLS:

```
UPDATE ASN.IBMSNAP_SUBS_SET SET SET_NAME ='NewSetName'
WHERE
APPLY_QUAL ='ApplyQual'AND SET_NAME ='ExistSetName';
UPDATE ASN.IBMSNAP_SUBS_MEMBR SET SET_NAME ='NewSetName'
WHERE
APPLY_QUAL ='ApplyQual'AND SET_NAME ='ExistSetName';
UPDATE ASN.IBMSNAP_SUBS_COLS SET SET_NAME ='NewSetName'
WHERE
APPLY_QUAL ='ApplyQual'AND SET_NAME ='ExistSetName';
```

Figure 8-8 SQL to change Apply subscription sets name

3. If your subscription set has any SQL before and after statements or procedures, then run the SQL shown in Figure 8-9 also at the Apply control server to update the IBMSNAP_SUBS_STMTS table:

```
UPDATE ASN.IBMSNAP_SUBS_STMTS SET SET_NAME ='NewSetName'
WHERE
APPLY_QUAL ='ApplyQual'AND SET_NAME ='ExistSetName';
```

Figure 8-9 Change subs set name in subs set statement table

4. From the Capture control server run the SQL shown in Figure 8-10 to change the subscription set name in the IBMSNAP_PRUNE_SET and IBMSNAP_PRUNCNTL tables:

```
UPDATE CaptureSchema.IBMSNAP_PRUNE_SET SET SET_NAME ='NewSetName '
WHERE
APPLY_QUAL ='ApplyQual'AND SET_NAME ='ExistSetName'AND TARGET_SERVER
='Target_Server ';
UPDATE CaptureSchema.IBMSNAP_PRUNCNTL SET SET_NAME ='NewSetName'
WHERE
APPLY_QUAL ='ApplyQual'AND SET_NAME ='ExistSetName'AND TARGET_SERVER
='Target_Server';
```

Figure 8-10 Change subscription set name in the pruning control tables

5. If you started the Apply program using the OPT4ONE parameter for UNIX, Windows and z/OS or OPTSNGSET parameter on the iSeries, then you have to stop and start the Apply program for new subscription set name to take effect, see 6.2.3, “Apply Parameters” on page cccxxvii for details on this parameter.
6. Reactivate the subscription to resume Apply processing, see Section 8.2.2, “Deactivating and activating subscriptions” on page ccclxxxi

Subscription apply qualifiers

There might be an occasion to change the apply quailifier, to break up the number subscription sets using the same apply qualifier. Changing the apply qualifier could help to balance out the workload.

The following steps will assist you to change the name of apply qualifier, without having to remove and then recreate the subscription set with a new apply qualifier, along with all associated members.

1. First, make sure the subscription set is deactivated, see Section 8.2.2, “Deactivating and activating subscriptions” on page ccclxxxi
2. From the Apply control server run the SQL shown in Figure 8-11 to change the subscription set name in the IBMSNAP_SUBS_SET, IBMSNAP_SUBS_MEMBER and IBMSNAP_SUBS_COLS:

```
UPDATE ASN.IBMSNAP_SUBS_SET SET APPLY_QUAL ='NewApplyQual'
WHERE
APPLY_QUAL ='ExistApplyQual 'AND SET_NAME ='SetName';
UPDATE ASN.IBMSNAP_SUBS_MEMBR SET APPLY_QUAL ='NewApplyQual'
WHERE
APPLY_QUAL ='ExistApplyQual'AND SET_NAME ='SetName';
UPDATE ASN.IBMSNAP_SUBS_COLS SET APPLY_QUAL ='NewApplyQual'
WHERE
APPLY_QUAL ='ExistApplyQual'AND SET_NAME ='SetName';
```

Figure 8-11 SQL to change apply qualifier at the apply control server

3. If your subscription set has any SQL before and after statements or procedures, then run the SQL shown in Figure 8-12 at the Apply control server to update the IBMSNAP_SUBS_STMTS table:

```
UPDATE ASN.IBMSNAP_SUBS_STMTS SET APPLY_QUAL ='NewApplyQual'
WHERE
APPLY_QUAL ='ExistApplyQual'AND SET_NAME ='SetName';
```

Figure 8-12 Change subs set name in subs set statement table

4. From the Capture control server run the SQL shown in Figure 8-13 to change the apply qualifier in the IBMSNAP_PRUNE_SET and IBMSNAP_PRUNCNTL tables:

```
UPDATE CaptureSchema.IBMSNAP_PRUNE_SET SET APPLY_QUAL ='NewApplyQual'
WHERE
APPLY_QUAL ='ExistApplyQual'AND SET_NAME ='SetName'AND TARGET_SERVER
='Target_Server ';
UPDATE CaptureSchema.IBMSNAP_PRUNCNTL SET APPLY_QUAL ='NewApplyQual'
WHERE
APPLY_QUAL ='ExistApplyQual'AND SET_NAME ='SetName'AND TARGET_SERVER
='Target_Server'
```

Figure 8-13 SQL to change the apply qualifier in the pruning control tables

5. Repeat steps 2 to 4 to change the apply qualifier for additional subscription sets.
6. If you started the Apply program using the OPT4ONE parameter for UNIX, Windows and z/OS or OPTSNGSET parameter on the iSeries, then you have to stop and start the Apply program for new subscription set name to take effect, see 6.2.3, “Apply Parameters” on page cccxxvii for details on this parameter.
7. Reactivate the subscription to resume Apply processing, see Section 8.2.2, “Deactivating and activating subscriptions” on page ccclxxxi.

Note: If you setup monitoring definitions then you need to change them, by removing and recreate new monitor definitions with the new subscriptions name or apply qualifiers as described previously in this section. You can use the replication center to make these changes. See Chapter 7., “Troubleshooting and monitoring” on page cccxxxvii on replication monitoring details

8.2.4 Removing Subscription sets

If a subscription set is not required to replicate data, you have the option to remove it from your replication environment.

When you remove a subscription set you have the option to drop the target table and index, and the apply and control server tables are updated

To delete the subscription set from the Replication Center, deactivate it first to make sure the Apply program has finish processing it, see Section 8.2.2, “Deactivating and activating subscriptions” on page ccclxxxi. Then display the following window by selecting the **Delete** option from the menu in Figure 8-1 on page ccclxxv. Figure 8-14 shows the Delete Subscription Set window:

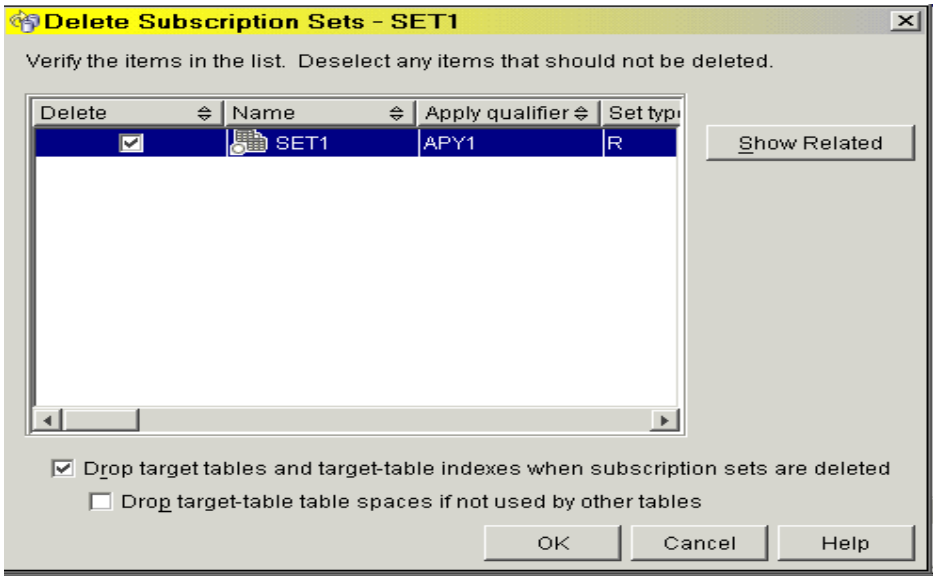


Figure 8-14 Remove subcription sets

Click **OK** to display the Run Now or Save screen. Review SQL script before selecting the option to run it.

Note, the click box to drop the target table and indexes.

When you remove a subscription set, you could either deactivate or remove all associated registered tables, to prevent capturing data you don’t need.

Remove subscription set on the iSeries

If your subscription sets reside on the iSeries server, then you have the option to remove your subscriptions sets using the RMVDPRSUB CL command. To use

this command enter it on the iSeries command line and press the F4 key, then press the F11 key to display the actual parameter names, see Figure 8-15:

```
Remove DPR Subscription (RMVDPRSUB)

Type choices, press Enter.

Apply qualifier  . . . . . . . . . APYQUAL
Set name . . . . . . . . . . . . . SETNAME
Control server . . . . . . . . . . CTLSVR          *LOCAL
Remove members . . . . . . . . . . RMVMBRS         *NO
Remove DPR registration  . . . . . RMVREG          *NO
Delete target table  . . . . . . . DLTTGTTBL       *NO
```

Figure 8-15 iSeries CL command to remove subscription sets

For field level help move the curser to the parameter and press the F1 key. Or refer to Chapter 18 in the *Replication Guide and Reference,* SC27-1121-00

8.2.5 Adding members to existing subscription sets

When you add a member to existing subscription set, a full refresh will occur for the entire subscription set. If you want to full refresh then see 5.3.5, “Adding subscription members to existing subscription sets” on page ccx. If you don’t want to force a full refresh, the following procedures describes how to prevent it for the entire subscription set.

1. First, make sure the subscription set is deactivated, see Section 8.2.2, “Deactivating and activating subscriptions” on page ccclxxxi
2. From the Capture control server run the SQL shown in Figure 8-16 to change the apply qualifier in the IBMSNAP_PRUNE_SET and IBMSNAP_PRUNCNTL tables:

```
UPDATE CaptureSchema.IBMSNAP_PRUNCNTL SET SYNCHPOINT =
(SELECT SYNCHPOINT FROM CaptureSchema.IBMSNAP_PRUNE_SETWHERE APPLY_QUAL
='ApplyQual'AND SET_NAME ='SetName'AND TARGET_SERVER ='Target_Server'),
SYNCHTIME = (SELECT SYNCHTIME FROM CaptureSchema.IBMSNAP_PRUNE_SET WHERE
APPLY_QUAL ='ApplyQual 'AND SET_NAME ='SetName 'AND TARGET_SERVER
='Target_Server')
WHERE
APPLY_QUAL ='ApplyQual 'AND SET_NAME ='SetName'AND
TARGET_SERVER='Target_Server';
```

Figure 8-16 SQL to update pruning control tables to resume apply processing

This SQL will ensure the Apply program will resume processing from the correct starting point in the CD for each member in the subscription set.

3. Add members to the subscription set. See, Section 5.3.5, “Adding subscription members to existing subscription sets” on page ccx.
4. Insert the CAPSTART signal in the IBMSNAP_SIGNAL table at the capture server to indicate a new subscription member was added.
   – First we need to locate the map ID for the new subscription member to insert the CAPSTART signal as shown in Figure 8-17:

```
SELECT MAP_ID FROM CaptureSchema.IBMSNAP_PRUNCNTL
WHERE
SOURCE_OWNER ='SrctableSchema'AND SOURCE_TABLE ='SrcTbSetl'AND
SOURCE_VIEW_QUAL =SrcVwQual AND APPLY_QUAL ='ApplyQual 'AND SET_NAME
='SetName'AND TARGET_SERVER ='Target_Server 'AND TARGET_OWNER ='TgtSchema
'AND TARGET_TABLE ='TgtTbl'WITH UR;
```

Figure 8-17 Locate subscription member map ID

   – With the MAP ID the SQL shown in Figure 8-18 will insert the CAPSTART signal

```
INSERT INTO CaptureSchema.IBMSNAP_SIGNAL (SIGNAL_TIME,SIGNAL_TYPE,
SIGNAL_SUBTYPE,SIGNAL_INPUT_IN,
SIGNAL_STATE,SIGNAL_LSN) VALUES (CURRENT TIMESTAMP,'CMD ',
'CAPSTART ','Mapid ','P ',NULL);
```

Figure 8-18 Insert the CAPSTART signal

5. Ensure the CAPSTART signal is picked up by the capture program, by running the SQL in Figure 8-19 to check the IBMSNAP_PRUNCNTL table.

```
SELECT SYNCHPOINT FROM captureSchema.IBMSNAP_PRUNCNTL
WHERE
MAP_ID ='Mapid 'WITH UR;
```

Figure 8-19 SQL to check CAPSTART signal was started

Continue to run this SQL and check the SYNCHPOINT value is not null.

6. Using your method of choice, load the source table data into the subscription member target table.
7. If you started the Apply program using the OPT4ONE parameter for UNIX, Windows and z/OS or OPTSNGSET parameter on the iSeries, then you have to stop and start the Apply program, see 6.2.3, “Apply Parameters” on page cccxxvii for details on this parameter

8. At the apply control server run the SQL in Figure 8-20 to update IBMSNAP_SUBS_SET, which will activate the subscription set, making the Apply program to immediately process the subscription set and prevent a full refresh.

```
UPDATE ASN.IBMSNAP_SUBS_SET SET SYNCHPOINT=NULL,
SYNCHTIME =CURRENT TIMESTAMP,LASTSUCCESS =CURRENT TIMESTAMP,
LASTRUN = CURRENT TIMESTAMP -X MINUTES,STATUS =0,ACTIVATE =1WHERE
APPLY_QUAL ='ApplyQual 'AND SET_NAME ='SetName';
```

Figure 8-20 SQL for Apply to start subscription set and prevent full refresh.

9. At this point the subscription set should resume processing with the addition of a new member.

8.2.6 Changing attributes of subscription sets

If you need to make a change to existing subscription set, you can use the replication center, which only allows you to change the following attributes:

- Schedules for applying updates (time-based replication or event-based replication)
- Subscription statements
- WHERE clause predicates of subscription-set members
- Commit count
- Data blocking value (MAX_SYNCH_MINUTES)

The following describes how to make these changes using the replication center:

- First deactivate the subscription set to prevent the Apply program from processing the subscription set while you are making the changes. The Apply program could still continue to process the other subscription sets defined within this Apply instance. See Section 8.2.2, “Deactivating and activating subscriptions” on page ccclxxxi.
- See Figure 8-1 on page ccclxxv and select **Properties** from the menu to display the window in Figure 5-2 on page ccxiii. See Section 5.3.1, “Create Subscription Sets with members” on page ccix for details on changing the subscription set attributes.
- To activate a subscription set, see Section 8.2.2, “Deactivating and activating subscriptions” on page ccclxxxi. Which will resume processing this subscription set in the Apply program.

8.2.7 Adding a new column to a source and target table

If a new column was added to a source table, and you need to replicate it. The following procedures will reproduce the new column, so that the Capture and Apply program could include the new column, without disrupting your current replication environment:

Note: If your registered a source table on the iSeries with RRN as the primary key, then you can't use the following procedures. The RRN has to be the last column in the CD table. Therefore, you have to remove the registration, add the new column to the source table, then reregister specifying RRN to be captured. See Section 8.1.3, "Removing registrations." on page ccclxxviii and Section 8.1.1, "Adding new registrations" on page ccclxxiv specifically on the iSeries.

Non DB2 restrictions: You cannot use these steps to add columns to registered sources on non-DB2 relational databases. A registration for a non-DB2 relational source includes a set of triggers used for capturing changes. You cannot alter these triggers. Therefore, if you need to add new column to this source table and need to replicate the data in these columns, you must drop and re-create the existing registered source.

1. Quiesce all activity against the source table were you want to add a new column
2. Depending on your platform server were you are running Capture, use one of the following methods to stop the Capture program:
   - Use the Stop Capture window from the Replication Center
   - asnccmd system command (Windows, UNIX, z/OS)
   - ENDDPRCAP iSeries CL command

   See , "Stopping Capture and Apply" on page cclxxvi for details on stopping the Capture program

   - If you need to keep the Capture program active during this procedure, you can insert a USER signal in the signal (IBMSNAP_SIGNAL) table after stopping activity against then source table. Wait for the Capture program to process the USER signal. After the Capture program processes the USER signal, the Capture program has no more activity to process against the associated CD table and no longer requires access to this CD table.
3. Deactivate the subscription set, see Section 8.2.2, "Deactivating and activating subscriptions" on page ccclxxxi.
4. Add the new column to the source table using the SQL ALTER ADD statement. This is done through the DB2 Command Center or any other tool available for your platform. The ALTER for the source table is not done through the Replication Center

5. Adding the new column to the CD table is done using the Replication Center. See Figure 8-6 on page ccclxxx, to display the *Registered Table Properties* window. Scroll down the list of columns, looking for the new column that was added in step 4. When you find it, select either **After-Image**, **Before-image** or both, then click **OK** button to generate and run the SQL script, running the SQL alter to the CD table.
6. Start the Capture program, if it was stopped. The Capture program automatically initialize the registration and captures the changes to these new columns when the Capture program first reads log or journal entries with the new columns.
7. Quiesce all activity against the target table.
8. Add the new column to the target table using the SQL ALTER statement.
9. Adding the new column to a subscription is done using the Replication center. See Figure 5-2 on page ccxiii to display the *Create Subcription Set* note page **--->Source to Target Mapping** page, see Figure 5-3 on page ccxvii. Select the member **--->Details...**, see Figure 5-5 on page ccxxi. Move the new column from the *Registered columns* to the *Selected columns* **---> Column Mapping** page, see Figure 5-6 on page ccxxii. Map the new column in the *Selected columns* to the *Target columns*. Clicking the arrow in the blue box, then dragging the mouse to the circle in the red box will create the arrow, indicating the source and target column are mapped.Click **---->OK --->OK** from the *Create Subcription Set* note page to generate and run the SQL script.
10. Reactivate the subscription set, see Section 8.2.2, “Deactivating and activating subscriptions” on page ccclxxxi to resume processing this subscription set in the Apply program.

8.3 Promoting a replication configuration to another system

Once you have developed and tested your replication configuration, the next step is to move that configuration to your production system. You can use the Replication Center promote function to copy your definitions so that you do not have to recreate them through the Replication Center. You can also use the PROMOTE function to duplicate your configuration on other systems.

Important: The promote function generates scripts to duplicate an existing replication configuration. It does not connect to the new server to validate the scripts.

You can promote:

- Registered tables
- Registered views
- Subscription sets

Restriction: You cannot use the promote function to copy replication definitions to or from non-DB2 (Informix) databases. You cannot use the promote function to copy replication definitions that include iSeries remote journals. You only promote replication definitions to like systems. For example, if your existing definitions are for DB2 z/OS, they cannot be promoted to DB2 for Windows and Unix. They can only be promoted to another DB2 z/OS system.

8.3.1 Promoting registered tables

The steps to promote registered tables are:

- Expand the **Replication Definitions—>Capture Control Servers** folder—>The Capture control server—> **Registered Tables** folder.
- Left-click on one or more registered tables to select them for promotion. Hold down the shift key while clicking to select multiple tables.
- Right-click on one of the selected tables and left-click on **Promote** in the list of operations.
- On the **Promote Registered Tables** window, enter information about the new registration. The **Capture control server alias** is required. All other fields are optional. If you leave them blank then the new registration will have the same schemas as the old registration and SQL to create the source table will not be generated. The input fields are:

- **Capture control server alias** — database alias of the server which will receive the promoted registrations. This field is required.
- **Capture schema** — capture schema for the promoted registrations.
- **Change CD table schema** — new schema for CD tables in the promoted registrations (optional).

Important: The PROMOTE SUBSCRIPTION SET dialog, described in the next section, does not have an input field for this new CD table schema. If you change it here, then you will need to manually update the promote subscription set scripts. This will be corrected in a future fixpack.

- **Create source tables** — check this box to create source tables at the new server and to optionally change the source table schema at that server.
- Click **OK** to generate the promotion scripts.

- Review the messages in the Messages and SQL Scripts window for any errors and then click on **CLOSE** to close the Messages window.
- Refer to 2.13, “Run Now or Save SQL” on page cxix for the procedure to **Run Now or Save SQL.** The only option available for promote scripts is **Save SQL.**
-

8.3.2 Promoting registered views

Important: You must promote the registered tables referenced in the source view before promoting the source view.

The steps to promote registered views are:

- Expand the **Replication Definitions—>Capture Control Servers** folder**—>**The Capture control server**—> Registered Views** folder.
- Left-click on one or more registered views to select them for promotion. Hold down the shift key while clicking to select multiple views.
- Right-click on one of the selected views and left-click on **Promote** in the list of operations.
- On the **Promote Registered Views** window, enter information about the new registration. The **Capture control server alias** is required. All other fields are optional. If you leave them blank then the new registration will have the same schemas as the old registration and SQL to create the source view will not be generated. The input fields are:

- **Capture control server alias** — database alias of the server which will receive the promoted registrations.
- **Capture schema** — capture schema for the promoted registrations.
- **Change CD view schema** — new schema for CD views in the promoted registrations (optional).
- **Change CD table schema** — new schema for CD tables in the promoted registrations (optional).

Important: The PROMOTE SUBSCRIPTION SET dialog, described in the next section, does not have an input field for the new CD view and CD table schema. If you change it here, then you will need to manually update the promote subscription set scripts. This will be corrected in a future fixpack

- **Create source views** — check this box to create the source views. You can also optionally:
  - create any unregistered base tables.
  - specify a new schema for the promoted views.
- Click **OK** to generate the promotion scripts.

► Review the messages in the Messages and SQL Scripts window for any errors and then click on **CLOSE** to close the Messages window.

► Refer to 2.13, “Run Now or Save SQL” on page cxix for the procedure to **Run Now or Save SQL.** The only option available for promote scripts is **Save SQL.**

►

8.3.3 Promoting subscription sets

When you promote a subscription set, all the subscription members are promotes as well.

The steps to promote registered tables are:

► Expand the **Replication Definitions—>Apply Control Servers** folder—>The Apply control server—> **Subscription Sets** folder.

► Left-click on one or more subscription sets to select them for promotion. Hold down the shift key while clicking to select multiple sets.

► Right-click on one of the selected sets and left-click on **Promote** in the list of operations.

► On the **Promote Subscription Set** window, enter information about the new subscription set. All of the fields are optional. If you leave them blank, the

promoted subscription set will have the same values as the old set. Here are the input fields:

- **Apply control server alias** — database alias of the server which will receive the promoted subscription.
- **Capture control server alias** — database alias of the capture control server for this promoted subscription.
- **Target server alias** — database alias of the target server for this promoted subscription
- **Apply qualifier** — apply qualifier for this promoted subscription.
- **Name of subscription set** — set name for the promoted subscription.
- **Capture schema** — capture schema used at the source server for this subscription.
- **Schema of source tables or views** — source schema used by the promoted subscription
- **Schema of target tables or views** —target schema used by the promoted subscription
- If the subscription set includes replica target tables, there are two additional input fields:
  - **Capture schema** — capture schema used at the target server for this subscription
  - **CD table schema** — schema of the CD table used at the target server
- Click **OK** to generate the promotion scripts.

- Review the messages in the Messages and SQL Scripts window for any errors and then click on **CLOSE** to close the Messages window.
- Refer to 2.13, “Run Now or Save SQL” on page cxix for the procedure to **Run Now or Save SQL.** The only option available for promote scripts is **Save SQL.**

8.4 Maintaining capture and apply control servers

Maintenance of a replication control server involves the following tasks:

- Pruning replication control tables that are not pruned automatically.
- Gathering statistics on control tables to optimize performance.
- Reorganizing dynamic control tables to reclaim space.
- Recovering replication tables
- Rebinding replication packages and plans
- Managing DB2 logs or journals

8.4.1 Manually pruning replication control tables

Section 6.3.1, “Pruning control tables” on page cccxxviii discusses the automatic pruning of capture control tables, either automatically by Apply, through the Replication Center, or by command.

There are several replication tables that are not pruned by Capture or Apply and must be pruned manually. These tables are:

- ASN.IBMSNAP_APPLYTRAIL at the apply control server
- ASN.IBMSNAP_APPLYTRACE at the apply control server
- ASN.IBMSNAP_MONTRAIL at the monitor control server
- CCD tables at the target server populated by an Apply program.

You can use the Apply program to automatically prune these tables based on an SQL DELETE statement that you supply.

Using Apply to prune apply control tables

You define an empty subscription set with the SQL statements needed to prune ASN.IBMSNAP_APPLYTRACE and ASN.IBMSNAP_APPLYTRAIL. Here are the steps to set up this pruning:

1. Expand the **Replication Definitions—>Capture Control Servers** folder**—>**The Capture control server**—> Registered Tables** folder, left-click on any registered table and choose **Create Subscription Set** from the list.

   This new set will have no members, so the registered table and capture control information will never be used.

2. Supply the following information on the **Create Subscription Set** notebook:
   - **Apply control server alias** — location of tables to be pruned.
   - **Set name** — any name you choose.
   - **Apply qualifier** — any existing or new apply qualifier.
   - **Target server alias** — must be the same as apply control server alias.
   - Check **Activate the subscription set**.
3. Click on the **Statements** tab of the **Create Subscription Set** notebook and click the **Add** button.
4. Do the following tasks on the **Add SQL Statement or Procedure Call** window**:**
   a. Choose **At the target server after the subscription set is processed**.
   b. Choose **SQL Statement** and enter the SQL statement shown in Example 8-2:

Example 8-2 SQL statement to prune APPLYTRAIL table

```
DELETE FROM ASN.IBMSNAP_APPLYTRAIL
```

```
WHERE STATUS = 0 and (CURRENT DATE - 7 DAYS) > DATE(LASTRUN)
This will keep all rows reporting an error and all rows for Apply processing in
the last week.
```

c. Click **OK** to return to the **Create Subscription Set** notebook.

5. Repeat steps 3 and 4, but use the SQL statement in Example 8-3 to prune from the APPLYTRACE table:

Example 8-3 SQL statement to prune ASN.IBMSNAP_APPLYTRACE

```
DELETE FROM ASN.IBMSNAP_APPLYTRACE
WHERE (CURRENT DATE - 7 DAYS) > DATE(TRACETIME)
This will keep all Apply messages for the last week.
```

6. Click on the **Schedule** tab of the **Create Subscription Set** notebook and set the relative timing for the subscription set
7. Click **OK** to generate the subscription scripts.
8. Review the messages in the Messages and SQL Scripts window for any errors and then click on **CLOSE** to close the Messages window.
9. Refer to 2.13, “Run Now or Save SQL” on page cxix for the procedure to **Run Now or Save SQL.**
10. If this is a new apply qualifier, start Apply. If this is an existing Apply qualifier, there is nothing further to be done.

Using Apply to prune ALERTS and CCD tables

You use the procedure described in “Using Apply to prune apply control tables” on page cccxcvi to prune the ASN.IBMSNAP_ALERTS table and optionally, any CCD tables. The key points to remember are:

- The **Target server alias** of the set must point to the server where the tables you want to prune are located. The Apply program that processes this set must be able to connect to that server.
- The SQL statement must contain logic to delete only the rows you no longer need. You determine what you need to keep. The timestamp columns in the table can be used as a basis for pruning:
  - ASN.IBMSNAP_MONTRAIL — LASTRUN
  - CCD tables — IBMSNAP_LOGMARKER

8.4.2 RUNSTATS for replication tables

Refer to , “Section 6.3.1, “Pruning control tables” on page cccxxviii discusses the automatic pruning of capture control tables, either automatically by Apply,

through the Replication Center, or by command.” on page cccxcvi for information on gathering statistics for your replication control tables.

Important: You should use the RUNSTATS utility to update statistics in the DB2 catalog for the CD and UOW tables when those tables are populated with change information that reflects the maximum activity on your registered tables. Apply performance will be degraded if the statistics are updated with low values that do not represent the normal state of these tables.

8.4.3 REORG for replication tables

The CD and UOW tables are *volatile.* Change information is inserted in these tables and then deleted when no longer needed. You can reorganize these DB2 tables to eliminate fragmentation and reclaim space using:

- DB2 for Windows and Unix **`REORG`** command
- DB2 for z/OS REORG utility with the PREFORMAT option
- DB2 for iSeries the ENDDPRCAP has a reorganize parameter call RGZCTLTBL. See, Figure 6-23 on page ccxcix

We recommend that you reorganize the following replication tables weekly:

- Capture control servers
  - all CD tables
  - IBMSNAP_CAPMON
  - IBMSNAP_CAPTRACE
  - IBMSNAP_UOW
- Apply control servers
  - IBMSNAP_APPLYTRACE
  - IBMSNAP_APPLYTRAIL
- Monitor control servers
  - IBMSNAP_MONTRAIL
  - IBMSNAP_MONTRACE
  - IBMSNAP_ALERTS

Important: If these tables will not be available during the reorganization, SUSPEND Capture, stop Apply, or stop the Alert Monitor while the reorganization takes place.

8.4.4 Rebinding replication packages and plans

The Capture and Apply programs must always be bound with ISOLATION UR. If you have procedures that automatically rebind packages and plans on a regular basis, be sure that all the replication packages are bound with ISOLATION UR. If you need to rebind after apply maintenance, be sure to use ISOLATION UR.

8.4.5 Recovering source tables, replication tables, or target tables

If a device failure or some other disaster occurs, you may need to restore tables from a backup image. The best possible case is that you can use the DB2 backup and logs to recover the tables to the current point in time (the end of the DB2 logs). If that is possible, then no special action is needed.

In some cases, you may not be able to recover to the end of the DB2 logs, possibly because the backup or logs is not available, or because you wish to set the registered tables back to a previous point in time. The impact on replication depends on which tables are involved.

If you recover a registered table to a previous point in time, target tables that copy from this registered table may no longer be valid. Capture may have captured changes that occurred after your recovery point and Apply may have processed applied those changes to the target.

Recovering a registered table to a previous point in time

If are user copies or replicas copy from this registered table, you must do a full refresh of the target tables to synchronize them with the source tables. The procedure is:

1. Stop Capture for this registration. Refer to , “Deactivating registration tables” on page ccclxxiv for the procedure to deactivate (stop) the registration.
2. Recover the registered table to the desired point in time
3. Restart the stopped registration from a new starting point with the SQL shown in Figure 8-21:

```
UPDATE Schema .IBMSNAP_REGISTER SET STATE ='I '

WHERE

SOURCE_OWNER = SrcSchema AND SOURCE_TABLE ='SrcTbl 'AND
SOURCE_VIEW_QUAL =SrcVwQual AND STATE ='S ';
```

Figure 8-21 SQL to activate registration after unexpected capture errors

4. Apply will full refresh all the members in the subscription set that contains the target table. You can choose to only refresh the member that copied from the recovered register table. Refer to 8.2.5, “Adding members to existing subscription sets” on page ccclxxxvii for details.

If consistent change data (CCD) tables copy from this registered table, then the procedure depends on the type of (CCD):

- Non-condensed with one row for every change to the registered table.

DELETE all rows from the CCD where IBMSNAP_LOGMARKER is greater than the recovery point in time. This is valid whether the CCD is complete or non-complete.

- Condensed and complete with one row for every row in the registered table, and one rows for every row that has been deleted from the registered table

  Copy all rows from the CCD where IBMSNAP_LOGMARKER is less than the recovery point in time and IBMSNAP_OPERATION = 'D' to preserve the record of deletions. Unload these rows to a file or store them in a different table.

  Force a full refresh of the CCD table using the technique above for user copies and replicas. All these rows will have an IBMSNAP_OPERATION = 'I' and IBMSNAP_LOGMARKER = current timestamp.

  Copy the saved deletion rows back to the CCD table.

- Condensed and not complete with one row for every changed row in the registered table.

  There is no way to recover this data. The previous values for the changed rows are not available.

Recovering control tables to a previous point in time

You cannot recover replication control tables to a previous point in time without corrupting your replication environment. Many control tables include SYNCHPOINT values which have relationships to other control tables. If you must recover your control tables to a previous point in time, then you should start Capture with the COLD parameter and force a reset of all control tables and a full refresh of all target tables.

Recovering target tables to a previous point in time

You can recover target tables to a previous point in time, but you then assume the responsibility for the accuracy of the data. You may need to do this if someone inadvertently deletes rows from a target table. A better solution is to full refresh the table from the source rather than recover.

8.4.6 Managing DB2 logs and journals used by Capture

Chapter 13, "Making changes to your replication environment" in the *IBM DB2 Universal Database Replication Guide and Reference, SC27-1121-00,* includes detailed instructions for managing logs and journals.

The DB2 log interfaces used by Capture can read log records from archived DB2 logs, as long as the logs are still available.

Important: You must ensure that no log or journal is deleted or moved before Capture has processed all the changes in the log or journal. If the Capture program requests a log record and the log file containing that record is not available, Capture will stop. This is not a recoverable error. The resolution is to start Capture with the COLD parameter and full refresh all your target tables.

You can use the *capshema*.IBMSNAP_RESTART table and the DB2 command **db2flsn** on DB2 for Windows and Unix capture control servers to determine which logs have been completely processed by Capture and can be removed from the system.

The *capschema*.IBMSNAP_RESTART table and the DSNJU004 (Print Log Map Utility) can be used on DB2 for OS/390 and z/OS capture control servers to determine which logs are no longer needed by Capture.

DB2 DataPropagator for iSeries includes a journal receiver exit program (**DLTJRNRCV**) which is registered automatically when you install DB2 DataPropagator. You specify DLTRCV(*YES) and MNGRCV(*SYSTEM) on the **CHGJRN** or **CRTJRN** command to use this exit program to prevent the deletion of journals until Capture is finished processing them.

8.5 Full refresh procedures

You may need to force a full refresh if a target table becomes corrupted or is lost or is no longer synchronized with the source table. A full refresh would also be needed if DB2 logs or journals are not available for Capture. The Replication Center can be used to update control tables so that Apply will automatically do the full refresh for you or so that you can do the full refresh yourself. You can also update the control tables directly, but we recommend that you use the Replication Center for this task, or at least use the Replication Center to generate the SQL statements needed for a full refresh.

8.5.1 Automatic full refresh

Restriction: Apply cannot do the full refresh if the registered table was defined with the option **Allow full refresh of target table** unchecked. If this option was not checked when the source table was registered, then you must use the manual refresh technique described in **8.5.2, “Manual full refresh” on page cdii**.

Use this procedure to tell Apply to do a full refresh for a subscription set:

- Expand the **Replication Definitions—>Apply Control Servers** folder—>The Apply control server—> **Subscription Sets** folder.
- Left-click on one subscription set to select it for refresh. key while clicking to select multiple sets.
- Right-click on the selected set, left-click on **Full Refresh** in the list of operations and then choose **Automatic** from the next list.
- Refer to 2.13, “Run Now or Save SQL” on page cxix for the procedure to **Run Now or Save SQL**.

Important: If you started the Apply program using the OPT4ONE parameter for UNIX, Windows and z/OS or OPTSNGSET parameter on the iSeries, then you have to stop and start the Apply program after requesting the refresh. See 6.2.3, “Apply Parameters” on page cccxxvii for details on this parameter.

8.5.2 Manual full refresh

Use this procedure when you wish to do the full refresh yourself:

- Expand the **Replication Definitions—>Apply Control Servers** folder—>The Apply control server—> **Subscription Sets** folder.
- Left-click on one subscription set to select it for refresh.
- Right-click on the selected set, left-click on **Full Refresh** in the list of operations and then choose **Manual** from the next list.
- Follow the prompts to update the capture control tables and deactivate the subscription set, unload the source data, load the target table, and update the apply control tables and activate the subscription set.

  You can use any utility or command you wish to unload the source and load the target.

8.5.3 Bypassing the full refresh

You can bypass the full refresh using the **Full Refresh — Manual** technique described in **8.5.2, “Manual full refresh” on page cdii**. The capture and apply control tables must still be updated. That part of the process cannot be skipped. You can skip the unload and load steps if you are confident that your source and target table are exactly synchronized. When you do this, you assume responsibility for the integrity of the data in the target table.

9

Advanced Replication Topics

This chapter covers these advanced topics:

- Filtering during replication
- Transformations during replication
- Replication of large objects
- Replication of DB2 Spatial Extender data
- Update anywhere replication
- Peer to peer replication

9.1 Replication filtering

There are many different ways to filter data during replication and it is not always easy to determine which is the best method to use for a particular requirement. This topic is a summary of several different methods you can use. Some of the methods are defined with the Replication Center. Others require that you code simple triggers. We refer to two different kinds of filtering:

- Column (or vertical) filtering which is a subset of columns for replication
- Row (or horizontal) subsetting which is a subset of rows for replication

9.1.1 Replicating column subsets

Column filtering can be done during the Capture process and/or during the Apply process. There are advantages and disadvantages for each method.

Important: Do not forget that each user copy, replica, and condensed CCD target table must include one or more columns that uniquely identify each target table row. Those target table types must have a primary key or unique index defined. Be sure to include the columns needed for uniqueness in your column list, regardless of the method you use to filter columns.

Capture column filters

Capture supports column filtering, specified when you register a source table. It is always wise to include as many of the source table columns as possible when registering a source table, so that you will be prepared for any future requests. You may wish to exclude a column because:

- the size of a column would impact CD table storage and Apply performance.

  A large column used for scratch pad comments in a table may not be needed on the receiving system. If the column is large, but contains important data, then it cannot be excluded.

- the column is not needed at the receiving system.

  An encrypted password column is probably not needed at a data warehouse target, so it could be excluded.

- the column contains confidential information that should not be copied.

  A customer's credit history or income information would not be copied to a salesperson's computer, but other customer information might be needed. The sensitive columns should be excluded.

When you register a source table using the Replication Center, you choose the columns you want to include in the Change Data table. Only those columns are

available for replication. This has the obvious advantage of reducing the size of the CD table. If a large number of changes occur to source table columns that are not in the CD table, even greater space savings and performance improvements can be gained if you choose the option “Capture changes to registered columns only”.

The disadvantage is that the excluded columns are not available if needed in the future.

As an alternative, you can filter columns using a source table view. Example 9-1 illustrates of a source table view that filters out unwanted columns:

Example 9-1 Subsetting columns using a source table view

```
Source table is created as:
  CREATE TABLE SAMP.EMPLOYEE (EMPNO CHAR(6) NOT NULL, FIRSTNME CHAR(10),
                         LASTNAME CHAR(10), HIREDATE DATE, SALARY DECIMAL(6,2),
                         PRIMARY KEY(EMPNO))
You do not want to replicate the SALARY column, so you create a view:
  CREATE VIEW SAMP.EMPVIEW (EMPNO,FIRSTNME,LASTNAME,HIREDATE)
    AS SELECT E.EMPNO,E.FIRSTNME,E.LASTNAME,E.HIREDATE
         FROM EMPLOYEE E
```

You must register both the source table and the source table view for replication.

The advantage of this technique is that the SALARY column is kept confidential. If you chose all the EMPLOYEE columns when registering the source table, then the SALARY column is available for other replication scenarios that may need it. If you are already using a view for other reasons, such as joining data, then it makes sense to include the filtering logic in this view as well.

The disadvantage of this technique is that you must create the view and the view registration, in addition to the source table registration.

Apply column filters

Apply supports column filtering, specified when you define the source-to-target table column mapping. The columns displayed in the column mapping list are the columns of the CD table, so the data may already have been filtered when the source table was defined as a replication source or a source table view.

There is no performance penalty in Apply regardless of the option chosen. Apply builds a column list for a select against the source table or view (full refresh) or against the CD table or CD table view (change processing) based on the entries in ASN.IBMSNAP_SUBS_COLS at the apply control server.

An advantage of filtering columns at the subscription member level is that this only restricts the columns for a particular subscription member, so the only limit for other replication is the columns chosen for the CD table.

You can use any combination of these three options:

- Exclude columns during source table registration
- Exclude columns with a source table view
- Exclude columns when defining a subscription member

9.1.2 Replicating row subsets

Row filtering involves an SQL predicate. Capture reads DB2 log records through a DB2 interface, not SQL. There is no predicate available to restrict the rows that Capture inserts into the CD table. You can use a trigger on the CD tables to skip the capture of changes.

Apply selects from the source table or view (full refresh) or the CD table or CD table view (change processing), so it is possible to specify predicates to limit the rows that are replicated.

Important: Excluding rows during replication can have a negative impact on the data integrity and consistency of your target tables. Certainly, excluding rows means that your source and target tables will not match.

DB2 Capture row filters

You can exclude rows from the CD table during the Capture process with a before insert trigger on the CD table. Capture recognizes the special SQLSTATE 99999 and will skip over the current change if the trigger returns 99999. You can use this trigger to:

- exclude all source table deletes.

  If you never, ever want to replicate deletes, you can use a CD table trigger to prevent them from being captured. Example 9-2 shows a trigger to skip deletes:

Example 9-2 CD table trigger to exclude captured deletes

```
CREATE TRIGGER SAMP.DEPT_SKIPDELETES
NO CASCADE
BEFORE INSERT ON SAMP.DEPARTMENT
REFERENCING NEW AS CD
FOR EACH ROW MODE DB2SQL
WHEN (CD.IBMSNAP_OPERATION = 'D')
SIGNAL SQLSTATE '99999' ('CD DELETE FILTER')
```

Skipping deletes may cause unusual behavior in your target table if key values are re-used. If a row is deleted from the source and then a new row is inserted into the source with the old key values, the existing row in the target table will be updated with the new values. If you need to preserve the old values, you may need to specify additional key columns for the target or change the target type to a non-condensed CCD.

- skip all changes based on column values in the change.

  Example 9-3 is a trigger to exclude all changes where the LOCATION column is equal to TEXAS:

Example 9-3 CD table trigger to exclude changes based on a column value

```
CREATE TRIGGER SAMP.DEPT_SKIPTEXAS
NO CASCADE
BEFORE INSERT ON SAMP.DEPARTMENT
REFERENCING NEW AS CD
FOR EACH ROW MODE DB2SQL
WHEN (CD.LOCATION = 'TEXAS')
SIGNAL SQLSTATE '99999' ('CD TEXAS FILTER')
```

During a full refresh, all the rows with LOCATION = 'TEXAS' will be copied to the target table unless you specify a subscription member row filter.

The Capture Throughput Analysis Report shows the number of rows skipped because of a CD table trigger or because you chose to capture only changes to your selected columns.

The disadvantage of using triggers to skip changes is that those changes are not available for replication to *any* location. If you have some subscriptions which need those changes and some that do not, a better choice is to use the UOW_CD_PREDICATES column for the members that do *not* need the changes.

Informix Capture row filters

Changes to Informix source tables are captured by triggers on the source table. You can add logic to the generated triggers to filter out rows during the Capture process. You should never return a non-zero code in these triggers, since that would cause the original transaction to fail.

Example 9-4 is the modification to the generated Informix insert trigger which skips all rows where the column CITY is set to ‘Denver’:

Example 9-4 Informix row filter

```
The generated trigger is:
CREATE TRIGGER “sampifx”.itccdcustomer
insert on “sampifx”.customer
referencing new as new for each row (execute procedure.....

The modified trigger is:
CREATE TRIGGER “sampifx”.itccdcustomer
insert on “sampifx”.customer
referencing new as new
for each row when (new.city <> ‘Denver’) (execute procedure....
```

You can skip all deletes to an Informix source table by dropping the delete trigger generated from the Replication Center.

Apply row filters

The Row filter for a subscription member is the primary method for filtering rows during replication. The filter is a predicate clause that you specify when defining a subscription member. This clause is added to the select which Apply issues for the source table (full refresh) or the CD table (change processing), so the clause can refer to any columns in the source table that have been registered for Capture. The row filter is stored in the PREDICATES column of the ASN.IBMSNAP_SUBS_MEMBR table at the apply control server.

Example 9-5 is a filter for a subscription member that will receive only rows where the location is TEXAS:

Example 9-5 Row filter to include only rows with a certain value

```
ASN.IBMSNAP_SUBS_MEMBR PREDICATES COLUMN is
        LOCATION = ‘TEXAS’
```

In this case, LOCATION is called the *partitioning key*, since it is used to partition the source table during replication. If this was a data distribution scenario, you might have a second subscription set for a different target server, the same target table, and a row filter of LOCATION = ‘CALIFORNIA’. If the column used for the *partitioning key* may be updated at the source server, then you should also choose the “Capture updates as pairs of deletes and inserts” option when registering the source table. This ensures that, if the LOCATION value is changed from ‘TEXAS’ to ‘CALIFORNIA’, then the corresponding rows will be deleted from the target server receiving only ‘TEXAS’ data and inserted on the target server receiving only ‘CALIFORNIA’ data.

In some cases, you may need to apply a filter only to the Apply select from the CD table. For this, you use the UOW_CD_PREDICATES. The Replication Center does not currently have an input method for this column in the ASN.IBMSNAP_SUBS_MEMBR at the apply control server, so you must update it manually. If you need to refer to UOW table columns in your predicate and your target table is a user copy, then you must also manually set the JOIN_UOW_CD column in ASN.IBMSNAP_SUBS_MEMBR to 'Y' at the apply control server. The CD filter and the Join specification will be added to the Replication Center in a future fixpack.

Suppose that you did not want to replicate deletes to a target table. Other target tables need the deletes, so you cannot use a Capture trigger to skip captured deletes. You would insert IBMSNAP_OPERATION <> 'D' in the UOW_CD_PREDICATES for the subscription member for that target table.

Another example is the case where you want to skip all changes made by a certain authorization userid (perhaps a batch job). The capschema.IBMSNAP_UOW table includes the authorization id for each unit of work. If your target table type is user copy, you must set JOIN_UOW_CD to 'Y' in the subscription set member entry to make the authorization id available to Apply. You do not need to update JOIN_UOW_CD for other target table types. Insert IBMSNAP_AUTHID <> 'skipid' in the UOW_CD_PREDICATES in ASN.IBMSNAP_SUBS_MEMBR.

Source table views can also be used to subset source table and CD table data before it is applied. Views may be required if your filtering predicates are very long. The ASN.IBMSNAP_SUBS_MEMBR predicates columns are limited in length:

- 1024 bytes for row filter column PREDICATES
- 1024 bytes for CD filter column UOW_CD_PREDICATES

9.2 Replication transformations

You can transform data during replication using these techniques

- Change Data table triggers
- Source table views
- Subscription columns based on SQL expressions
- SQL statements or stored procedures issued after Apply processes changes

9.2.1 Capture transformations

You can use a CD table trigger to do some transformation of the change data during the Capture process. Remember that this trigger will be executed every time a change is inserted to the CD table, so the trigger should be as simple and efficient as possible.

Important: The CD table trigger cannot change the data type or length of a captured column. Those attributes are fixed when the CD table is created and must match the source table column attributes so that DB2 log records can be decoded. If the column attributes of your target are different from the column attributes of your target, this transformation should be done either by a source table view or by using SQL expressions in the source to target table column mapping.

Example 9-6 is a trigger that sets default values for a source table column with a null value:

Example 9-6 CD table trigger to set default values for a null column

```
CREATE TRIGGER SAMP.DEPT_FIXLOC
NO CASCADE
BEFORE INSERT ON SAMP.CDDEPARTMENT
REFERENCING NEW AS NEW
FOR EACH ROW MODE DB2SQL
WHEN (NEW.LOCATION IS NULL)
BEGIN ATOMIC
SET NEW.LOCATION = 'Unknown';
END
```

Example 9-7 shows a trigger that sets default values for a column based on the value given for some other column in the source table.

Example 9-7 set values in one column based on another column

```
CREATE TRIGGER SAMP.DEPT_FIXLOC
NO CASCADE
BEFORE INSERT ON SAMP.CDDEPARTMENT
REFERENCING NEW AS NEW
FOR EACH ROW MODE DB2SQL
WHEN (NEW.LOCATION IS NULL)
BEGIN ATOMIC
SET NEW.LOCATION =
 CASE
   WHEN NEW.ADMRDEPT = 'A00' THEN 'Palo Alto'
   WHEN NEW.ADMRDEPT = 'D01' THEN 'Dallas'
   WHEN NEW.ADMRDEPT = 'E01' THEN 'Istanbul'
   WHEN NEW.ADMRDEPT = 'F01' THEN 'Toronto'
   WHEN NEW.ADMRDEPT = 'G00' THEN 'Pembrooke Pines'
   ELSE 'Unknown'
 END;
END
```

Example 9-8 is a trigger where the reference information is stored in a separate table at the source server:

Example 9-8 set values in one column based on another table

```
CREATE TRIGGER SAMP.DEPT_FIXLOC
NO CASCADE
BEFORE INSERT ON SAMP.CDDEPARTMENT
REFERENCING NEW AS NEW
FOR EACH ROW MODE DB2SQL
WHEN (NEW.LOCATION IS NULL)
BEGIN ATOMIC
SET NEW.LOCATION =
(SELECT L.LOCNAME FROM SAMP.LOCATIONS WHERE NEW.ADMRDEPT = L.DEPT);
END
```

All of the triggers in these examples were tested using DB2 Universal Database for Windows V8 and the DB2 SAMPLE database.

9.2.2 Source table views

Source table views are a powerful tool for transforming data during replication. You can define a view over a single table to transform the data to meet target

table requirements or define a view with an inner join to bring data from multiple source tables together for replication.

You can define a simple view over a single table that uses SQL expressions to change the source data. Example 9-9 is such a view:

Example 9-9 Source table view over one table

```
The source table is defined as:
CREATE TABLE EMPLOYEE
  (EMPNO CHAR(6) NOT NULL,
   FIRSTNME VARCHAR(12), MIDINIT CHAR(1), LASTNAME VARCHAR(15),
   PRIMARY KEY(EMPNO))

The target table is defined as:
CREATE TABLE EMPLOYEE
  (EMPLOYEE_NUMBER INTEGER NOT NULL,
   EMPLOYEE_NAME VARCHAR(40),
   PRIMARY KEY(EMPLOYEE_NUMBER)

The source table view to map this transformation is:
CREATE VIEW REPEMPLOYEE
  (EMPLOYEE_NUMBER, EMPLOYEE_NAME)
  AS SELECT INTEGER(E.EMPNO),
            VARCHAR(E.LASTNAME CONCAT ',' CONCAT E.FIRSTNME CONCAT ' '
                    CONCAT E.MIDINIT,40)
   FROM EMPLOYEE E
```

You register the source table EMPLOYEE and then the source table view REPEMPLOYEE. When you subscribe to the source view, the names, data types and lengths of the columns already meet the target table definitions.

A join view is more complex in terms of processing, since it involves multiple source tables and multiple CD tables. When you register a join view, the Replication Center creates a join view for each CD table.

Example 9-10 shows a join view with two source tables:

Example 9-10 Source table view over two tables

```
CREATE VIEW REPEMPLOYEE
  (EMPLOYEE_NUMBER, EMPLOYEE_NAME, DEPARTMENT, LOCATION, MANAGER_NUMBER)
  AS SELECT INTEGER(E.EMPNO),
            VARCHAR(E.LASTNAME CONCAT ',' CONCAT E.FIRSTNME CONCAT ' '
                    CONCAT E.MIDINIT,40),
            E.WORKDEPT, D.DEPTNAME, D.MGRNO
     FROM EMPLOYEE E, DEPARTMENT D
     WHERE E.WORKDEPT = D.DEPTNO
```

Example 9-11 shows the CD table views created when you register this source table view:

Example 9-11 CD table views

```
CDEMPLOYEE view
CREATE VIEW REPCDEMPLOYEE
  (<IBMSNAP columns>,
  EMPLOYEE_NUMBER, EMPLOYEE_NAME, DEPARTMENT, LOCATION, MANAGER_NUMBER)
  AS SELECT INTEGER(E.EMPNO),
            VARCHAR(E.LASTNAME CONCAT ',' CONCAT E.FIRSTNME CONCAT ' '
                    CONCAT E.MIDINIT,40),
            E.WORKDEPT, D.DEPTNAME, D.MGRNO
     FROM CDEMPLOYEE E, DEPARTMENT D
     WHERE E.WORKDEPT = D.DEPTNO

CDDEPARTMENT view
CREATE VIEW REPCDDEPARTMENT
  (<IBMSNAP columns>,
  EMPLOYEE_NUMBER, EMPLOYEE_NAME, DEPARTMENT, LOCATION, MANAGER_NUMBER)
  AS SELECT INTEGER(E.EMPNO),
            VARCHAR(E.LASTNAME CONCAT ',' CONCAT E.FIRSTNME CONCAT ' '
                    CONCAT E.MIDINIT,40),
            E.WORKDEPT, D.DEPTNAME, D.MGRNO
     FROM EMPLOYEE E, CDDEPARTMENT D
     WHERE E.WORKDEPT = D.DEPTNO
```

A problem can occur with this technique if the following happens:

1. There is an EMPLOYEE row with WORKDEPT = 'AAA' and a DEPARTMENT row with DEPTNO = 'AAA', so the target table has a row with EMPLOYEE_NUMBER = 'AAA'
2. The 'AAA' rows in EMPLOYEE and DEPARTMENT are deleted
3. This delete is never replicated because the CD table view shows a change with 'AAA', but the source table in the join no longer has this value.

You can work around the *double delete* problem by modifying the CD table views. Example 9-12 shows the modification:

Example 9-12 CD table view modification for double delete problem

```
Join condition for CD table views
     WHERE E.WORKDEPT = D.DEPTNO

Modified predicate to include deletes regardless of the join predicate
     WHERE E.WORKDEPT = D.DEPTNO OR IBMSNAP_OPERATION = 'D'
```

If your view has a predicate, you may need a different modification. The goal of the modification is to always include any deletes in the CD table, regardless of whether or not the deletes meet the join condition or any other predicates that you have specified. Apply will ignore any deletes for rows that do not exist in the target table, so there is no penalty for replicating deletes that do not match the join condition or any predicates you provide.

9.2.3 Apply Transformations

You can transform existing data and add new data during the Apply process.

Calculated columns

The Replication Center Member Properties window has a Column Mapping tab. You use the Column Mapping page to map source table columns to target table columns. You use the Change Calculated Column button to provide SQL expressions which define columns on the target table.

In Example 9-9 on page cdxii, we showed a source table view that transformed EMPLOYEE table. That same transformation could be accomplished by using the SQL expressions to create calculated columns.

CASE expressions can be used to provide conditional logic for a calculated column. You can map source table columns to different column names, different data types and include DB2 special register values like timestamp or user.

You can also map a calculated column to a literal value. In a data consolidation scenario, you might want to have a column in your target table that identifies the source server of the row. The expression for this calculated column would be a literal like ‘SAMPLE’ representing the source server for that member.

Target table triggers

You can use target table triggers to transform data during replication. If your target table is a replica, remember that it is also updated by user applications and your trigger will execute for those changes as well as changes processed by Apply. If Apply encounters an unexpected SQL code from a change, it will not be able to continue processing, so be careful when coding triggers that return non-zero SQL states.

9.2.4 Before and after SQL statements

When creating a subscription set, you can define SQL statements or stored procedure calls that Apply will execute before or after it processes a set of changes. These statements are *not* executed for each change that Apply processes. The changes being processed are not available to your SQL

statements or stored procedure except as they exist in the CD table (BEFORE statements) or the target table (AFTER statements).

Suppose that you have a target table column called FIXED that is defined as a calculated column with a value of NULL. You then create a stored procedure which selects all the rows from the target table where FIXED is null, uses business logic to cleanse and validate those rows, and then updates the target table with the new values, including FIXED = ‘YES’. This stored procedure is defined to run at the target server after the subscription set is processed, so that Apply will automatically call it every time changes are processed.

9.3 Replication of large objects

Large objects (*LOB*s) are DB2 columns defined as CLOB (character large object) or BLOB (binary large object) and Informix columns defined as clob/text (character large object) or blob/byte (binary large object).

Large objects are limited in size to 2GB on DB2 servers. Informix large object columns can be greater than 2GB, but the DB2 federated functions places a 2GB limit on LOBs replicated from Informix. Currently DB2 V8 federated function does not support insert, update, or delete for Informix nicknames with LOB columns. This means that replication of LOBs *to* Informix targets is not supported.

LOB columns cannot be used as primary keys or unique indexes, so any source or target table containing a LOB must also have at least one other column.

9.3.1 DB2 LOB replication

The primary difference between replication of LOB columns and that of other columns is that the LOB data is not stored in the CD table. Example 9-13 shows a source table and a CD table with a LOB column:

Example 9-13 CD table with a LOB column

```
Source table is:
CREATE TABLE EMP_PHOTO
  (EMPNO CHAR(6) NOT NULL, PHOTO_FORMAT VARCHAR(10) NOT NULL,
   PICTURE BLOB(100K),
   PRIMARY KEY(EMPNO))

CD table is:
CREATE TABLE CDEMP_PHOTO
  (<IBMSNAP columns>,
   EMPNO CHAR(6) NOT NULL, PHOTO_FORMAT VARCHAR(10) NOT NULL,
   PICTURE CHAR(1))
```

Note that, in the CD table, the PICTURE column is defined as one character. Capture puts a ‘U’ in this column if the BLOB column PICTURE is updated in the source table.

When a subscription member is defined that copies from EMP_PHOTO and the PICTURE column is selected, the COL_TYPE value in ASN.IBMSNAP_SUBS_COLS at the apply control server is set to ‘L’ for this target table column.

When Apply processes this subscription member, the 'L' in ASN.IBMSNAP_COLS indicates that this is a LOB column and requires special handling:

- If this is a full refresh, then the LOB column and the other columns are selected from the source table and inserted into the target table.
- If this is change processing and:
  - the change is an *insert,* then the PICTURE value from the source table is joined with the change data table row and the result is inserted into the target table.
  - the change is a delete, then the PICTURE column is ignored. It is not needed to find the target table row that should be deleted.
  - the change is an update:
    - if the CD table PICTURE column is 'U', then the PICTURE value from the source table is joined with the change data table row and the result is used to update the target table.
    - If the CD table PICTURE column is not 'U', then the PICTURE column is ignored and only the non-LOB columns are used to update the target table.

This method conserves space in the CD table and minimizes the amount of data which is transferred to the target server.

9.3.2 Informix LOB replication

Replication to an Informix target with LOBs is not supported. The DB2 V8 wrapper for Informix does not allow insert/update/delete of LOB data.

Replication from an Informix source with LOBs can be accomplished with the following technique

- Register the Informix source table, but do *not* select the LOB columns. Save the generated SQL to a file for further modification. Do NOT execute the SQL.
- Modify the saved SQL:
  - Add a placeholder column for the LOB column to the CCD table. This column should have the same name as the source LOB column and be defined as CHAR(1).
  - Change the update stored procedure (up<ccdtablename>). The last statement in the stored procedure is an insert statement. Add the placeholder column name to the end of the insert column list and 'U' to the end of the values list.

- Run the modified SQL using the DB2 Command Center. Note that the termination character is the # sign and that you may have to edit the CONNECT statement to add your userid and password. If you run the script from the DB2 command window, the command is **db2 -vtd# -f** <filename>.
- Create a subscription set and subscription member in the usual way.

9.4 Replication of DB2 Spatial Extender data

Attention: Time limits prevented the Redbook team from testing this solution. It has been tested by IBM development and several customers, so we have included it in this book.

Spatial data is a representation of geographic features that exist at specific locations. Examples of spatial data are coordinates that represent a specific address, lines that represent a river or a road, and polygons that represent a city boundary or a lake. Refer to `http://www.software.ibm.com/data/spatial` for information about the DB2 Spatial Extender.

The DB2 Spatial Extender is a set of stored procedures and functions that allow you to store and manipulate spatial data in DB2 tables. The spatial data is stored in columns with structured data types that cannot be processed by Capture.

Example 9-14 is a DB2 table with a spatial data column.

Example 9-14 DB2 table with a spatial data column

```
CREATE TABLE SAMP.CUSTOMER
(ID INTEGER NOT NULL,NAME VARCHAR(30),
 ADDRESS CHAR(30),CITY CHAR(28),STATE CHAR(2),ZIP CHAR(5),
 INCOME DOUBLE,PREMIUM DOUBLE,CATEGORY SMALLINT,
 LOCATION ST_POINT,
 PRIMARY KEY (ID))

The LOCATION is registered as a spatial column using the DB2 spatial extender
stored procedure ST_register_spatial_column and populated with encoded values.
```

The DB2 Spatial Extender includes functions which can be used to convert (*cast)* spatial columns to text. Views and triggers using the spatial functions can be defined to replicate spatial data:

1. Create a view of the source table which casts the spatial column to datatype of varchar as shown in Example 9-15:

Example 9-15 Source table view for spatial replication

```
CREATE VIEW SAMP.CUSTVIEW
 (ID,NAME,ADDRESS,CITY,STATE,ZIP,INCOME,PREMIUM,CATEGORY,LOCATION_TEXT)
AS
 SELECT C.ID,C.NAME,C.ADDRESS,C.CITY,C.STATE,C.ZIP,C.INCOME,C.PREMIUM,
        C.CATEGORY,
        CAST(db2gse.ST_Astext(C.LOCATION) AS VARCHAR(50))
```

```
You must determine the correct function to convert the spatial data type to
text and also determine the appropriate length.
```

2. Create the target table, including a column for the text representation of the spatial data as shown in Example 9-16:

Example 9-16 Target table for spatial replication

```
CREATE TABLE TARGET.CUSTOMER
(ID INTEGER NOT NULL,NAME VARCHAR(30),
 ADDRESS CHAR(30),CITY CHAR(28),STATE CHAR(2),ZIP CHAR(5),
 INCOME DOUBLE,PREMIUM DOUBLE,CATEGORY SMALLINT,
 LOCATION_TEXT VARCHAR(40),
 LOCATION ST_POINT,
 PRIMARY KEY (ID))

The LOCATION is registered as a spatial column using the DB2 spatial extender
stored procedure ST_register_spatial_column.
```

3. Register the source table (SAMP.CUSTOMER) using the Replication Center. Do *not* register the LOCATION column.
4. Register the source view (SAMP.CUSTVIEW) using the Replication Center. Include the LOCATION_TEXT column in the registration.
5. Create a subscription set with a new member (or add a member to an existing subscription set). The source is SAMP.CUSTVIEW. Map all the columns from the view to the target table. Note that there is no mapping for the spatial column. This column will be populated by a trigger on the target table.
6. Create a trigger on the target table that converts the LOCATION_TEXT column to spatial data in the LOCATION column. Example 9-17 is the trigger for the TARGET.CUSTOMER TABLE:

Example 9-17 Target table trigger to convert text to spatial

```
CREATE TRIGGER TARGET.CUST_SPATIAL_TRIG
NO CASCADE
BEFORE INSERT ON TARGET.CUSTOMER
REFERENCING NEW AS NEW
FOR EACH ROW MODE DB2SQL
SET NEW.LOCATION =
     db2gse.ST_PointFromText(NEW.LOCATION_TEXT,db2gse.coordref()..srid(1))

In this example, LOCATION is defined as a point, so the ST_PointFromText
function is called for the conversion. Refer to the DB2 Spatial Extender
documentation for information on the function parameters.
```

7. Start Capture and Apply. Apply will select the spatial data as text (LOCATION_TEXT) from the source table view (full refresh) or the CD table

view (change processing) and the target table trigger will convert the text to the spatial representation (LOCATION).

9.5 Update anywhere replication

Update anywhere replication is the bidirectional exchange of data between two or more servers. One server is the designated master server; all other servers are called replica servers. Changes made at each replica server are copied to the master server. Changes made at the master server are copied to the replica server, including changes that came to the master server from other replicas.

In this section, we examine in detail the setup and operations for update anywhere replication. The master server is called MASTER and the replica server is called REPLICA.

Update anywhere is best used for applications where the number of changes at the replica sites is not high. There is one Apply at each replica processing changes in both directions. The changes from the MASTER to the REPLICA are processed with a pull (block fetch from the MASTER CD table and local inserts, updates, deletes to the REPLICA target). The changes from the REPLICA to the MASTER are processed with a push (local fetch from the REPLICA CD and remote inserts, updates, deletes to the MASTER target). A pull configuration is less efficient, so changes move from the REPLICA to the MASTER at a slower rate. There is also a performance cost is you choose to detect and compensate for conflicts during replication.

Update anywhere replication is not an appropriate choice for high availability requirements, unless the REPLICA will be maintained as a standby system with no application activity during normal operations. Refer to the next section for a discussion of replication and high availability.

9.5.1 Administration — defining update anywhere replication

For detailed information on these tasks, refer to Chapters 2, 3, 4, and 5 of this book.

Defining database servers as replication sources and targets

You must define the capture control tables at each server and the apply control tables at each replica for the update anywhere replication. Table 9-1 shows what would be created:

Table 9-1 Control tables for update anywhere replication

Server	Capture control tables	Apply control tables
MASTER	YES	NO
REPLICA	YES	YES

As you will see later on, you will run Capture on both servers, but Apply only on the REPLICA server.

Defining replication source tables

You register the source tables on the MASTER server using the Replication Center. There are several options that are important for update anywhere:

- Before-image columns are *not* needed.
- Capture changes from replica target table is not needed for this example. If you have multiple replicas, then you would check this so that changes made at one replica will flow to the other replicas through the master.
- Conflict detection level can be No detection, Standard detection, or Enhanced Detection. Refer to , “Conflict detection level” on page clxxxv for a discussion of the three levels.

The source table is registered in the normal way:

- If the source table is not already defined with DATA CAPTURE CHANGES, the source table is altered to have that attribute.
- A change data table, called a *CD table*, is created to hold the captured changes.
- A row is inserted into the capture control table *capschema*.IBMSNAP_REGISTER with information about the registration, including the name of the CD table.

Defining update anywhere subscriptions

You use the Replication Center to define a subscription set and members for the update anywhere scenario. Refer to Chapter 5., “Subscription Set” on page cciii for a detailed description of this process.

This is what takes place when you define a subscription set with a member that has target table type of replica:

- A row is inserted into the apply control table ASN.IBMSNAP_SUBS_SET for replication from the MASTER to the REPLICA. The WHOS_ON_FIRST column in ASN.IBMSNAP_SUBS_SET for this set is ‘S’.
- A row is inserted into the apply control table ASN.IBMSNAP_SUBS_SET for replication from the REPLICA to the MASTER. The WHOS_ON_FIRST column in ASN.IBMSNAP_SUBS_SET for this set is ‘F’.
- Two rows are inserted into the apply control table ASN.IBMSNAP_SUBS_MEMBR for each target table. One row has the WHOS_ON_FIRST column set to ‘S’ and the other has the WHOS_ON_FIRST column set to ‘F’.
- Two rows are inserted into the apply control table ASN.IBMSNAP_SUBS_COLS for each column in the target table. One row

has the WHOS_ON_FIRST column set to ‘S’ and the other has the WHOS_ON_FIRST column set to ‘F’.

- If the replica target table does not already exist, it is created with DATA CAPTURE CHANGES. If the target table does exist, but does not have the DATA CAPTURE CHANGES attribute, then the target table is altered to include DATA CAPTURE CHANGES.
- The replica target table is registered at the REPLICA server and a CD table is created to hold changes made at the replica. This CD table includes the before-image values for each change, since they be needed to reverse conflicting transactions.
- Rows are inserted into capschema.IBMSNAP_PRUNE_SET and capschema.IBMSNAP_PRUNCNTL at both the MASTER and the REPLICA.

It may be easier to think of this as two different replication scenarios:

- WHOS_ON_FIRST = ‘S’ copies from a table registered on the MASTER to a target table on the REPLICA.
- WHOS_ON_FIRST = ‘F’ copies from a table registered on the REPLICA to a target table on the MASTER.

9.5.2 Operations — Capture and Apply

Chapter 6 provides detailed information by platform about operating Capture and Apply. In this section, we will discuss what Capture and Apply do when processing changes for an update anywhere scenario.

Capture initializes global information

You start Capture at the MASTER server and the REPLICA server with a capture schema and capture control server name. The first time Capture runs for a capture schema/capture control server combination, it inserts a row into *capschema*.IBMSNAP_REGISTER with the GLOBAL_RECORD column set to Y and the SYNCHPOINT/SYNCHTIME columns set to the current point in the DB2 log. Capture reads the *capschema*.IBMSNAP_REGISTER table to find registrations. A registration is not active until Apply has signalled that a full refresh has been done.

Apply full refreshes the target table

Apply is started with an apply qualifier and the name of the apply control server. Apply connects to the apply control server and selects rows from ASN.IBMSNAP_SUBS_SET which have a matching apply qualifier. Apply ignores any sets which are not active (ACTIVATE column is 0). A new set has SYNCHPOINT and SYNCHTIME and LASTSUCCESS values of NULL in ASN.IBMSNAP_SUBS_SET. This tells Apply that a full refresh is needed.

In this case, there are two subscription sets with the same apply qualifier and set name that are processed together by Apply. One set has WHOS_ON_FIRST = ‘F’ (processed First) and one has WHOS_ON_FIRST = ‘S’ (processed second) in ASN.IBMSNAP_SUBS_SET.

Replica to master full refresh — WHOS_ON_FIRST = ‘F’
Apply selects the member information from ASN.IBMSNAP_SUBS_MEMBR at the apply control server and updates the *capschema*.IBMSNAP_PRUNCNTL at the REPLICA capture control server with the SYNCHPOINT column = x’00000000000000000000’ and the SYNCHTIME column = CURRENT TIMESTAMP for each member.

Then, Apply inserts one row into *capschema*.IBMSNAP_SIGNAL at the REPLICA capture control server for each member in the set. The value for SIGNAL_TYPE is CMD, for SIGNAL_SUBTYPE is CAPSTART and for SIGNAL_INPUT_IN is the MAP_ID of the subscription member from the *capschema*.IBMSNAP_PRUNCNTL table at the REPLICA capture control server.

A full refresh is *not* done for this set. The assumption is that the MASTER server is already populated.

Apply then updates the LASTSUCCESS and SYNCHTIME columns for this set in ASN.IBMSNAP_SUBS_SET and changes the MEMBER_STATE for each member in ASN.IBMSNAP_SUBS_MEMBR to L to indicated that the target tables have been loaded. This is done at the apply control server.

Master to replica full refresh — WHOS_ON_FIRST = ‘S
Apply selects the member information from ASN.IBMSNAP_SUBS_MEMBR at the apply control server and updates the *capschema*.IBMSNAP_PRUNCNTL at the MASTER capture control server with the SYNCHPOINT column = x’00000000000000000000’ and the SYNCHTIME column = CURRENT TIMESTAMP for each member.

Then, Apply inserts one row into *capschema*.IBMSNAP_SIGNAL at the MASTER capture control server for each member in the set. The value for SIGNAL_TYPE is CMD, for SIGNAL_SUBTYPE is CAPSTART and for SIGNAL_INPUT_IN is the MAP_ID of the subscription member from the *capschema*.IBMSNAP_PRUNCNTL table at the MASTER capture control server.

The REPLICA target tables (replicas) are loaded from the MASTER source tables using the same process described in 1.5, “DB2 Replication V8 close up” on page liii for read only target tables. If your REPLICA target tables have referential constraints, then you must do a manual refresh yourself to avoid violating the constraints or use a modified ASNLOAD exit which loads the data in

the correct order. Another technique is to remove the constraints, do the full refresh, and then put the constraints back.

Apply then updates the LASTSUCCESS and SYNCHTIME columns for this set in ASN.IBMSNAP_SUBS_SET and changes the MEMBER_STATE for each member in ASN.IBMSNAP_SUBS_MEMBR to L to indicated that the target tables have been loaded.

Capture starts capturing changes

Apply inserted a row into the *capschema*.IBMSNAP_SIGNAL table at both the MASTER and the REPLICA servers, so Capture will start capturing changes on both servers. Capture requests log records from DB2. When Capture detects the insert that Apply executed for the *capschema*.IBMSNAP_SIGNAL table, it begins to capture changes. This process is exactly the same as the process described in 1.5, “DB2 Replication V8 close up” on page liiifor DB2 Capture with read only tables.

Apply processes changes

Apply checks for active sets (ACTIVATE =1) in ASN.IBMSNAP_SUBS_SET at the apply control server. Apply then reviews the active sets to see if any are eligible for processing. A set is eligible if the sleep_minutes plus the timestamp value of LASTRUN is greater than the current time or if the EVENT_NAME has been posted in the ASN.IBMSNAP_SUBS_EVENT table.

In this case, there are two subscription sets with the same apply qualifier and set name that are processed together by Apply. One set has WHOS_ON_FIRST = ‘F’ (processed First) and one has WHOS_ON_FIRST = ‘S’ (processed second) in ASN.IBMSNAP_SUBS_SET. Conflict detection is done when copying from the REPLICA server to the MASTER server. The MASTER always wins any conflict in an update anywhere scenario. Apply always uses transactional processing for replica set members.

Replica changes to master — WHOS_ON_FIRST = ‘F’

If a set is eligible, then Apply does the following:

- Reads the ASN.IBMSNAP_SUBS_STMTS table and issues any SQL statements that were defined for execution BEFORE Apply runs.
- Reads the member and column mapping information from the ASN.IBMSNAP_SUBS_MEMBR and ASN.IBMSNAP_SUBS_COLS tables for the WHOS_ON_FIRST = ‘F’ set (REPLICA to MASTER).
- Sets a *lower bound* for changes equal to the SYNCHPOINT in the ASN.IBMSNAP_SUBS_SET table for each subscription set.
- Connects to the REPLICA capture control server and checks the global SYNCHPOINT in the *capschema*.IBMSNAP_REGISTER. This LSN is the

upper bound for changes. If the lower and upper bounds are equal, then there are no changes to process.

- Selects changes for each member from the REPLICA CD tables between the lower and upper bounds. The select includes the SQL transformations and row filters defined for the member and is a join of the CD and UOW tables.
- Stores the result set in a spill file on the server where Apply was started. There is one spill file for each subscription member. At this point, all the unprocessed changes from the REPLICA are in the spill files.
- If conflict detection is either standard or enhanced, then:
  - Connects to the MASTER capture control server and selects the key columns of all unprocessed changes.
  - Compares the spill file key columns from REPLICA changes with the key column values from MASTER changes. If any match, then this is a conflic.t
  - Uses the before-image values in the REPLICA CD table to reverse the conflicting change and all other changes in that same unit of work. If there are later units of work which include changes to rows that have been reversed (compensated), then those units of work are also reversed.
  - Updates the IBMSNAP_REJ_CODE in the *capschema*.IBMSNAP_UOW table at the REPLICA capture control server for each unit of work that was reversed.
  - Inserts an audit row into ASN.IBMSNAP_APPLYTRAIL at the apply control server with STATUS = -1 indicating that conflicts have been detected and compensated.
  - Selects changes for each member from the REPLICA CD tables between the lower and upper bounds, excluding any compensated units of work, and replaces the spill files.
- Reads the spill files and issues inserts, updates, deletes against the MASTER target tables. Changes may need to be reworked when issuing inserts, updates, and deletes.
- Executes any SQL statements from the ASN.IBMSNAP_SUBS_STMTS which are marked to be run AFTER Apply processing.
- Updates the ASN.IBMSNAP_SUBS_SET SYNCHPOINT and SYNCHTIME columns for this set (WHOS_ON_FIRST = ‘F) at the apply control server with the LSN of the upper bound and the timestamp of the upper bound. The MEMBER_STATEs in ASN.IBMSNAP_MEMBR for all members of the set are set to S.
- Updates the *capschema*.IBMSNAP_PRUNE_SET SYNCHPOINT column for this set at the MASTER capture control server with the upper bound LSN.

- Inserts an audit row into ASN.IBMSNAP_APPLYTRAIL at the apply control server.

Master changes to replica — WHOS_ON_FIRST = 'S'

Changes copied from MASTER to REPLICA are processed with the same steps documented in 1.5, "DB2 Replication V8 close up" on page liii for read only target tables.

Capture prunes applied changes

Pruning is described in Chapter 1. The only difference for update anywhere is that pruning does not delete CD table or UOW table rows for units of work that were compensated due to conflicts. These rows are kept until the Capture retention_limit is reached. You can use this information to analyze and correct conflicts.

9.6 DB2 Peer to peer replication

Peer to peer replication is bidirectional. It differs from update anywhere replication:

- There is no MASTER server.
- Capture and Apply run on all peer servers.
- Administration cannot be done totally through the Replication Center.

The primary use of peer to peer replication is for high availability and disaster recovery. The advantages of peer to peer over a hardware solution (disk mirroring) or log shipping are:

- Disk mirroring and/or log shipping may not be available on the desired platform.
- Replication may be relatively inexpensive compared to the other solutions, both in the initial cost and ongoing administration.
- Peer to peer can operate across distances that are not easily supported, if supported at all, by disk mirroring.
- Applications can run on any peer system, so query workload can be balanced across the peer configuration. Note that this is *not* workload balancing for change processing. Each peer server must be able to handle local changes as well as the replicated changes from all other peers.

The disadvantages of peer to peer are:

- Replication is *ASYNCHRONOUS*. There is always some delay in copying changes from one system to another. At any given point in time, two peer systems may not exactly match. If a failure occurs on one system and application activity is transferred to the second system, there may be changes that have not yet been replicated to the second system. The replication delay is called *latency.* Refer to *“Configuring for low-latency replication” on page cdxci* for tips on reducing this delay.
- Apply cannot do conflict detection and compensation. “Conflict detection using triggers” on page cdxxxvi describes a method to resolve this problem.
- Most of the peer to peer setup is done through the Administration Center, but there are some updates to control tables that must be done manually. “Administration and operations for peer to peer replication” on page cdxxix includes all the manual updates needed.

9.6.1 Administration and operations for peer to peer replication

Peer to peer replication is really two separate definitions, one for each direction in the bidirectional exchange of data.

We assume that the initial peer system (PEER1) already exists with application tables defined and populated. Please refer to Chapters 2, 3, 4, and 5 for detailed descriptions of the Replication Center tasks.

There are two configurations for peer to peer, both designed to prevent the capture of changes made by Apply:

- Assign a special userid to Apply with a UOW_CD_PREDICATE that screens the changes. This method has the advantage that you can use the Replication Center to administer and monitor the peer to peer environment. The disadvantage is that changes are captured twice, once when they are made to the source table and once when Apply processes them. Changes are only applied once.
- Modify the capture and control tables to resemble half of an update anywhere configuration so that Capture will not recapture changes made by Apply. The advantage here is that you do not have to use a special userid for Apply and changes are only captured once. The disadvantage is that the Replication Center and Alert Monitor cannot be used for this configuration after the manual modifications are done.

In the steps that follow, we will refer to these methods as the *APPLYID* method or the *UAHALF* method to distinguish tasks that are unique to a particular method.

Adding the second peer (PEER2)

First, you must create the capture and apply control tables at each peer server, register the source tables, and create the subscriptions using the Replication Center:

1. Create capture and apply control tables in PEER1 and PEER2.
2. Register the source tables at PEER1. Before-image columns are not needed. Be sure to choose the following options:
   - No Allow full refresh of target table.
   - No Capture changes from replica target table.
   - Conflict detection: No detection.

Important: You must set all of the above correctly or peer to peer replication will not work properly.

3. Define a subscription set with members from PEER1 to PEER2. The definitions are for standard user copies except where noted in this list:
   - Apply control server: PEER2.
   - Set name: anything you wish.
   - Apply qualifier: anything you wish.

Important: If you are using the UAHALF method, then the same Apply qualifier must be used for both directions of replication.

- Capture control server: PEER1.
- Capture schema: capschema for PEER1 capture control tables.
- Target server: PEER2.
- Check Activate the subscription set.
- Check Allow Apply to use transactional processing for set members.
- Specify a number for the Number of transactions.
- Schedule: Relative timing: 0 minutes.

4. Start Capture on PEER1.
5. Use the Replication Center to do a Manual Full Refresh from PEER1 to PEER2. Load the tables on PEER2 using your favorite utility.
6. Register the source tables at PEER2. Before-image columns are not needed. Be sure to choose the following options:
   - No Allow full refresh of target table.
   - No Capture changes from replica target table.
   - Conflict detection: No detection.

Important: You must set all of the above correctly or peer to peer replication will not work properly.

7. Define a subscription set with members from PEER2 to PEER1. The definitions are for standard user copies except where noted in this list:
   - Apply control server: PEER2.
   - Set name: anything you wish.
   - Apply qualifier: for UAHALF method, must match the step3 qualifier.

Important: If you are using the UAHALF method, then the same Apply qualifier must be used for both directions of replication.

   - Capture control server: PEER2.
   - Capture schema: capschema for PEER2 capture control tables.
   - Target server: PEER1.
   - Check Activate the subscription set.
   - Check Allow Apply to use transactional processing for set members.
   - Specify a number for the Number of transactions.
   - Schedule: Relative timing: 0 minutes.

8. Start Capture on PEER2.

9. Use the Replication Center to do a Manual Full Refresh from PEER1 to PEER2..

Important: This step updates the capture and apply control tables. DO NOT LOAD THE PEER1 TABLES. These tables are already populated with data.

The setup for the two peer to peer methods is different from this point on.

APPLYID peer to peer method

These are the next steps for the APPLYID method:

1. Identify a userid that will dedicated for Apply. This is the userid that will be used to start Apply. It should be used only for that purpose. You should use the same userid for both PEER1 and PEER2 to make this solution less complex. The userid, called APPLYID in this example, must be granted the necessary privileges for Apply.
2. If the peer servers are DB2 for Windows and Unix, then use the **asnpwd** command to create a password file on PEER1 and PEER2 that Apply will use to connect to both the source and target servers. On z/OS, modify the started task or JCL for Apply to ensure that this userid is used.
3. Update JOIN_UOW_CD and UOW_CD_PREDICATE for all the members in the subscription sets at PEER1 and PEER2.

Important: This SQL should be issued at PEER1 and PEER2

UPDATE ASN.IBMSNAP_SUBS_MEMBR SET JOIN_UOW_CD = 'Y', UOW_CD_PREDICATE='IBMSNAP_AUTHID<>''APPLYID''' WHERE TARGET_TABLE IN ('target-table1','target-table2','target-tablen')

The APPLYID literal is enclosed in two single quotes (not double quotes) and should be the userid identified for Apply in step 10.

4. Start Apply at both PEER1 and PEER2.

UAHALF peer to peer method

These are the next steps for the UAHALF method:

1. Change the source table type to REPLICA.

Important: This SQL must be run at both PEER1 and PEER2

```
UPDATE capschema.IBMSNAP_REGISTER SET
SOURCE_STRUCTURE=7 WHERE SOURCE_OWNER = 'xxxxxx' AND
SOURCE_TABLE in ('source-table1','source-table2','source-tablen')
```

2. Change the target table structure to REPLICA.

Important: This SQL must be run at both PEER1 and PEER2

```
UPDATE capschema.IBMSNAP_PRUNCNTL SET
TARGET_STRUCTURE=7 WHERE APPLY_QUAL = 'apply qualifier'

UPDATE ASN.IBMSNAP_SUBS_SET SET WHOS_ON_FIRST='F',
SET_TYPE='U' WHERE APPLY_QUAL = 'apply qualifier'

UDPATE ASN.IBMSNAP_SUBS_MEMBR SET WHOS_ON_FIRST='F',
TARGET_STRUCTURE=7 WHERE APPLY_QUAL = 'apply qualifier'

UDPATE ASN.IBMSNAP_SUBS_COLS SET WHOS_ON_FIRST='F',
WHERE APPLY_QUAL = 'apply qualifier'

UDPATE ASN.IBMSNAP_SUBS_STMTS SET WHOS_ON_FIRST='F',
WHERE APPLY_QUAL = 'apply qualifier'
```

3. Start Apply at PEER1 and PEER2.

9.6.2 Adding another peer

Adding new peer servers after the second one requires more planning. When adding PEER2, we knew that PEER1 data could be used to full refresh the PEER2 tables. When adding PEER3, you cannot arbitrarily choose a server for the full refresh, since that server might not have all the changes from the other peer servers. Here are the steps to add PEERn to a peer to peer configuration with x existing peers.

Define replication from existing peers to PEERn

1. Define x subscription sets from PEER1 through PEERx.The definitions are for standard user copies except where noted in this list:
   - Apply control server: PEERn.
   - Set name: anything you wish.
   - Apply qualifier: for UAHALF method, must match the PEER apply qualifier.

Important: If you are using the UAHALF method, then the same Apply qualifier must be used for all peer subscriptions.

   - Capture control server: PEER1 through PEERx.
   - Capture schema: capschema for PEER capture control tables.
   - Target server: PEERn.
   - Check Activate the subscription set.
   - Check Allow Apply to use transactional processing for set members.
   - Specify a number for the Number of transactions.
   - Schedule: Relative timing: 0 minutes.
   - Member definitions are the same as usual.
2. Use the Replication Center to do a Manual Full Refresh for each of the x subscription sets you just created.

Important: Do not load the target table at PEERN.

3. Identify a peer server that you will use as a source for the full refresh of PEER3. This can be any peer server. For our example, we will use PEER1. The SYNCHPOINTs in the capture and apply control tables are used to ensure that PEER1 can be used for a full refresh without loss of data:
   a. When a manual refresh is done, a signal is sent to Capture and Capture updates the *capschema*.IBMSNAP.PRUNE_SET table with the starting point for that subscription set. The starting point for the new peer will be different on each existing peer. If there are 3 existing peers, then the values in PRUNE_SET might be as shown in Table 9-2:

Table 9-2 Starting points for PEERn subscription sets

Capture Control Server	PRUNE_SET starting LSN
PEER1	x'BB7D0000000000000000'
PEER2	x'AC340000000000000000'
PEER3	x'42778200000000000000'

b. The ASN.IBMSNAP_SUB_SET at our chosen full refresh server PEER1 has a SYNCHPOINT column for each subscription set. We use this column to make sure that any changes on the other peer servers that happened BEFORE the PEERn starting point have been copied to PEER1. Suppose Table 9-3 that lists the values at PEER1:

Table 9-3 Changes processed at PEER1

Capture control server	PEER1 subscription set	PEER1 SYNCHPOINT
PEER2	PEER2-TO-PEER1	x'BB7D0000000000000000'
PEER3	PEER3-TO-PEER1	x'ABFF0000000000000000'

c. You must wait until the ASN.IBMSNAP_SUBS_SET SYNCHPOINT values at PEER1 are greater than or equal to capschema.IBMSNAP_PRUNE_SET values at all the other peers before loading the new PEERn with the PEER1 data. In our example, PEER3 changes that happened before PEERn's starting point are not yet copied to PEER1. You must wait for these changes to be copied to PEER1 before you do the unload the data from PEER1 and load the data on PEERn. Do not continue with the next steps until you have loaded data to PEERn.

4. Register the source tables at PEERn. Before-image columns are not needed. Be sure to choose the following options:
   - No Allow full refresh of target table.
   - No Capture changes from replica target table.
   - Conflict detection: No detection.

Important: You must set all of the above correctly or peer to peer replication will not work properly.

5. Define x subscription set with members from PEERn to PEER1 through PEERx. The definitions are for standard user copies except where noted in this list:
   - Apply control server: PEER1 through PEERx.
   - Set name: anything you wish.
   - Apply qualifier: for UAHALF method, must match the PEERx qualifier.

Important: If you are using the UAHALF method, then the same Apply qualifier must be used for all subscriptions.

   - Capture control server: PEERn.
   - Capture schema: capschema for PEERn capture control tables.
   - Target server: PEER1 through PEERx.
   - Check Activate the subscription set.

- Check Allow Apply to use transactional processing for set members.
- Specify a number for the Number of transactions.
- Schedule: Relative timing: 0 minutes.
- Members are defined as usual.

6. Start Capture on PEERn
7. Use the Replication Center to do a Manual Full Refresh for each of the x subscription sets you just created.

Important: Do not load the target tables at PEER1 through PEERx.

8. Complete the setup by following the instructions for the method you have chosen:
   - “APPLYID peer to peer method” on page cdxxxii.
   - “UAHALF peer to peer method” on page cdxxxiii.
9. Start Apply at PEERn.

9.6.3 Conflict detection using triggers

In a peer to peer configuration, Apply does not have enough information to identify conflicts based on the key values, since only the changes from the source system are available.

One way that you can identify conflicts is by maintaining a timestamp in your application tables. A conflict occurs if the timestamp of a change processed by Apply is earlier than the timestamp of the row being changed by Apply. You may also wish to include server information in the application table to act as a tie-breaker if the timestamps are equal. When using this method, you must:

- Always start Apply with a specific userid so that the conflict triggers can identify changes that are coming from Apply rather than from your applications.
- Add a timestamp column and server column to your application tables.
- Create a stored procedure or function that converts timestamps to a universal time so that timestamp comparisons are valid across time zones.
- Prevent deletes to your application tables so that conflicts can be detected. If deletes are needed your applications must do logical deletes which update some column in the row to indicate it is no longer in use.
- Always start Apply with the SQLERRORCONTINUE parameter so that Apply will skip changes rejected by the conflict triggers.

This is an example of the conflict triggers for a peer to peer configuration with two peer servers, PEER1 and PEER2. The peer to peer method used is APPLYID, described in “APPLYID peer to peer method” on page cdxxxii. The utime function converts all timestamps to a universal time. There is one table in the configuration, SAMP.ORDERS.

Application table changes

Example 9-18 shows the definition of SAMP.ORDERS and the two new columns added for conflict detection:

Example 9-18 Adding conflict columns to application tables

```
CREATE TABLE SAMP.ORDERS
    (ORDERNUM SMALLINT NOT NULL, ORDER_QUANTITY INTEGER,
     PRIMARY KEY(ORDERNUM))

ALTER TABLE SAMP.ORDERS ADD COLUMN DELETED SMALLINT NOT NULL DEFAULT 0

ALTER TABLE SAMP.ORDERS ADD COLUMN LAST_UPDATED TIMESTAMP NOT NULL
    DEFAULT CURRENT TIMESTAMP

ALTER TABLE SAMP.ORDERS ADD COLUMN LAST_UPDATED_SITE CHAR(18) NOT NULL
    DEFAULT 'PEER1'
NOTE: The default for LAST_UPDATED_SITE for PEER2 application tables is set to
'PEER2' instead of 'PEER1'.
```

Triggers to populate conflict columns

These triggers populate the added columns, LAST_UPDATED and LAST_UPDATED_SITE, whenever a change is made to the application table. Example 9-19 shows the triggers before modifications to prevent conflicts:

Example 9-19 Triggers to populate conflict columns

```
CREATE  TRIGGER CC_ORDERS_INSERT
   NO CASCADE BEFORE INSERT
   on SAMP.ORDERS
REFERENCING NEW AS NEW
FOR EACH ROW MODE DB2SQL
BEGIN ATOMIC
  SET NEW.LAST_UPDATED = CURRENT TIMESTAMP;
  SET NEW.LAST_UPDATED_SITE = CURRENT SERVER;
END

CREATE  TRIGGER CC_ORDERS_UPDATE
   NO CASCADE BEFORE UPDATE
   ON SAMP.ORDERS
REFERENCING NEW AS NEW OLD AS OLD
```

```
FOR EACH ROW MODE DB2SQL
BEGIN ATOMIC
  SET NEW.LAST_UPDATED = CURRENT TIMESTAMP;
  SET NEW.LAST_UPDATED_SITE = CURRENT SERVER;
END
```

Triggers to detect conflicts

These triggers detect conflicts and report the error to Apply. If an insert is processed by Apply and the row already exists, then the insert is changed to an update, so there is no need for conflict detection in the insert trigger. There are no deletes on the target table, so there is no need for a delete trigger.

Example 9-20 shows the update trigger after modifications to prevent conflicts:

Example 9-20 Trigger with modifications to prevent conflicts

```
CREATE  TRIGGER CC_ORDERS_UPDATE
   NO CASCADE BEFORE UPDATE
   ON SAMP.ORDERS
REFERENCING NEW AS NEW OLD AS OLD
FOR EACH ROW MODE DB2SQL
BEGIN ATOMIC
  DECLARE CHANGETS TIMESTAMP;
  DECLARE LASTTS TIMESTAMP;
  SET CHANGETS = UTIME(NEW.LAST_UPDATED);
  SET LASTTS = UTIME(OLD.LAST_UPDATED);
  IF USER = 'APPLYID' AND CHANGETS < LASTTS
     THEN SIGNAL SQLSTATE '99997' ('CHANGE TIMESTAMP < CURRENT TIMESTAMP');
  END IF;
  IF USER = 'APPLYID' AND CHANGETS = LASTTS AND NEW.LAST_UPDATED_SITE <>‘PEER1’
     THEN SIGNAL SQLSTATE ‘99998’ (‘CHANGE TIMESTAMP = CURRENT TIMESTAMP’);
  END IF;
  IF USER <> ‘APPLYID’ THEN
     SET NEW.LAST_UPDATED = CURRENT TIMESTAMP;
     SET NEW.LAST_UPDATED_SITE = CURRENT SERVER;
  END IF;
END
```

If the USER (CURRENT USER on z/OS) is Apply, then conflict checking is done:

- If the new LAST_UPDATED is earlier than the current LAST_UPDATED, then reject the change.
- If the new LAST_UPDATED is the same as the current LAST_UPDATED and the change was not made on the PEER1 server, then reject the change.
- If there is no conflict, process the change, using LAST_UPDATED and LAST_UPDATED_SITE from the source server.

If the USER is not Apply, then set the current values for LAST_UPDATED and LAST_UPDATED_SITE.

Apply parameter to skip rejected changes

The update conflict trigger returns non-zero SQLSTATEs if a conflict is detected. You can define these states to be any 5 character value that if not already used by DB2 for an SQLSTATE. In this example, we use SQLSTATEs 99997 and 99998 to indicate that a conflict has been detected.

You use the Apply SQL error processing option to skip rejected conflicts and report the rejected changes. First you must create a file listing the SQLSTATEs you want Apply to ignore. The filename is *applyqualifier*.sqs. It must be located in the apply_path directory that you specify when starting Apply. For our example, the apply qualifier is PEERQUAL, so the file would be named PEERQUAL.sqs. Example 9-21 lists the contents of PEERQUAL.sqs:

Example 9-21 File with conflict SQLSTATEs used by Apply

```
99997
99998
```

You start Apply with the parameter SQLERRORCONTINUE set to 'Y'. When this parameter is used, Apply will compare any non-zero SQLSTATEs to the list in the sqs file. If it finds a match, it will write the change information to a file named *applyqualifier*.err (PEERQUAL.err for our example), skips the failing change and continues on with the processing for all other changes. Apply reports this in the apply control table ASN.IBMSNAP_APPLYTRAIL with a STATUS code of 16.

Important: Only the one change that was rejected is skipped. Any other changes in the unit of work containing the rejected change will be processed by Apply.

10

Performance

In this chapter we will discuss:

- end-to-end system designs for replication
- Capture performance
- Apply performance
- configuring replication for low latency
- development benchmark results

DB2 Replication Version 8, like previous versions, is still SQL based, meaning that the most operations depend on DB2 to do the bulk of work in response to requests, in the form of SQL, from Capture and Apply.

We will discuss Capture and Apply separately, focusing on the key sub-operations they each perform. The principal new aspect with Version 8 is Capture's accumulating infomation about whole transactions in memory before inserting records to CD tables.

Lastly, DB2 Replication Development performed a few benchmarks on pSeries (AIX) and we can provide some of the resutls.

But first it would be good to discuss overall system design for replication.

10.1 End-to-end system design for replication

The major sub-operations involved in DB2 Replication are listed below:

1. Capture at the source server reads the DB2 Log or iSeries Journal and accumulates records into memory for each transaction.
2. When Capture reads the commit for the transaction from the log/journal, Capture inserts records for a transaction into replication staging tables at the source server
3. Apply fetches records in multi-record blocks from the staging table at the source server and writes these records into Apply spill files on the system where Apply is running.
4. Apply inserts, updates, and deletes records in the target tables at the target server from the records in the Apply spill files.

We purposely did not specify the location of Apply above because Apply uses DB2 connection capabilities to perform both the fetch from the source server and the insert/update/delete to the target tables. Apply can be on any system that has DB2 Connectivity to the Apply Control Server, the Source/Capture Control Server, and the Target Server.

10.1.1 *Pull* replication system design

Since the fetch can obtain multiple records at once and the insert/update/delete is always one record at a time, the best performing design, if source tables and target tables are on different systems, is usually the ‘pull’ design, which uses Apply on the same system as the target tables. This is depicted in Figure 10-1.

Figure 10-1 Replication - 'pull' system design

10.1.2 *Push* replication system design

Figure 10-2 shows the 'push' design, which uses Apply on a different system than the system containing the target system. Expectations should be that for the same volume of changes, the pull design should be faster than the push design. How much faster of course depends on a number of factors. If there is only one row changed at the source per replication interval, than they should be equivalent. But as the number of rows changed per interval increases, the performance advantage of doing local updates to the target table will increase. The other main factor affecting the difference between pull and push is the network between the Apply system and the target server. Lower data rate, higher utilization, and more router hops will all make the individual insert, update, delete operations slower.

For those wondering why Apply won't block the updates to the target table, the answer is that data integrity is a higher priority for DB2 Replication than performance. Apply needs to get the result of inserting, updating, deleting each record in the spill file to the target table so that it can detemine if an insert or update needs to be re-worked in order to satisfy the data-integrity objective of making the target table row look like the corresponding source table row.

Figure 10-2 Replication - 'push' system design

10.1.3 iSeries-to-iSeries replication with remote journalling

When replicating between iSeries systems, there is an alternative design involving remote journalling that can deliver better performance than the normal 'pull' design. If the journals for the source tables have remote-journals on the target system, then Capture and Apply can both run on the target system. With iSeries, if the CD table and the target table are on the same system, then Apply

will perform the insert/update/deletes to the target table from a select from the CD table and eliminate the intermediate step of writing all the change records first to a spill file and then doing the insert/update/deletes from the spill file.

Also, iSeries is very efficient in the way it sends records to a remote journal. DRDA is not used for this function, where as DRDA is used if the CD tables are local to the source table and Apply at the target system fetches the changes from the CD table using the usual ‘pull’ design.

Figure 10-3 depicts iSeries-iSeries replication using remote journalling.

Figure 10-3 Replication iSeries-to-iSeries using remote journalling

In the figure, we’ve labeled the insert into the target table as steps ‘3/4’ to be consistent with our original description of the operations involved replication. Step 3 - the fetch from the CD table - is in effect imbedded as a ‘select’ into the insert operations to the target table.

There are some restrictions for replication between iSeries systems using remote journalling:

- view registrations are not supported if any of the the underlying registered tables are using remote journals for capturing changes
- replication of LOBs and DataLinks is not supported via remote journals

10.1.4 Replicating to non-DB2 servers

The implications of pull versus push should also be consider in designing replication involving non-DB2 sources or targets. Figure 10-4 shows the components when replicating to Informix.

Figure 10-4 Replication to Informix

As far as Apply is concerned, Figure 10-4 depicts a ‘pull’ scenario since the nicknames for the target tables are in a local DB2 database on the same system with Apply. But if the Informix server is on a different system from the DB2 that

has the target nicknames, then we have the performance considerations of a push design. If the DB2 federated server must be on a different system than the Informix target tables, then the federated server should ideally be in the same building and on the same highspeed network segment - with low utiliziation - as the Informix server.

10.1.5 Replicating from non-DB2

When replicating from a non-DB2 data source, the blocking that Apply does to fetches changes from the federated server that has the source and CD table nicknames will be translated into a blocked fetch from the non-DB2 data source, so a pull design, using Apply close to the target tables should perform the best. If the target server is DB2 ESE Version 8, then the same database that contains the target tables could also contain the nicknames for the source/CD tables and for the Capture Control tables in the non-DB2 data source. If the target server is DB2 for z/OS or OS/390, or iSeries, then the DB2 ESE or DB2 Connect with source and Capture Control nicknames could be either alongside the mainframe or iSeries system, or near or on the non-DB2 source system, or anywhere in between. Figure 10-5 depicts a pull design for replicating from Informix.

Figure 10-5 Replicating from Informix - pull system design

In the figure, trying to be consistent with the beginning discussion on major operations in replication, we've labeled the Capture triggers as performing operations '1/2' since they simulate the function of both the 1) log-read and the 2)insert into the CD tables that is performed by Capture in a DB2 system.

10.2 Capture Performance

The Capture operations we'd like to discuss are:

1. reading the DB2 log or iSeries journal
2. holding transactions until commit for them is read from the log
3. inserting into the CD tables
4. pruning CD and IBMSNAP_UOW tables, and other control tables.

1 - 3 are involved in Capture making changes available for Replication by Apply. Of the three, inserting into the CD tables is usually the slowest and usually the focus of attention when trying to improve Capture performance.

We'll discuss the characteristics of the CD tables and the IBMSNAP_UOW table in a separate topic since their characteristics also affect Apply's performance.

10.2.1 Reading the DB2 Log

Capture uses DB2's interface for providing log records to applications. On z/OS, it is the DB2's Instrumentation Facility Interface (IFI) and on Linux, UNIX, and Windows it is the DB2 Asynchronous Read Log API. Which is to say Capture itself is not reading the log file; it is DB2 that is reading the log file and providing the log records to Capture via the interface.

DB2 will not provide Capture a log record until the record has been written to the log file on disk. But if a log record is still in memory in a log buffer, DB2 will provide it to Capture from the memory buffer rather than perform a disk I/O to get the record for Capture. Thus, if Capture is run while there is update activity in DB2, it will help Capture's performance if DB2 has more memory allocated for log buffers as this will increase the probability that DB2 can provide the log records to Capture without the time delay of disk reads.

On z/OS, the parameters that increase DB2's memory allocations for log buffers are specified on DB2 installation panel for *Active Log Data Sets* (DSNTIPL)

- OUTBUFF - DB2 for z/OS and OS/390 Versions 6 and 7
- WRTTHRSH - DB2 for z/OS and OS/390 Version 6 only

On Linux, UNIX, and Windows, it is

- LOGBUFSZ in the Database Configuration.

  The default value for LOGBUFSZ is 8 pages. We recommend specifying a larger value.

If Capture 'lags behind' so that it can not process the log records as fast as DB2 is creating them and so DB2 has to provide log records from disk from the active log file, then attention to the placement of the log file on the disk system will affect how quickly DB2 can provide the records to Capture, which will affect Capture performance. But the the active log file may be optimally placed already as it's location on the disk system also affects DB2's performance of updates for applications.

If Capture is not running when there is update activity in DB2 and so Capture has to 'catch up' for hours, days, weeks, and so on, and you are not going to reload the target tables and COLD start Capture, then availability and placement of the DB2 archive log files will affect DB2's performance in providing the log records to Capture. If the point in the log that Capture has to start reading from is in a DB2 archive log file that is on tape, then when Capture starts DB2 will request that the tape be loaded so DB2 can start providing the log records requested by Capture. For better performane, if Capture has not been running for some time, we recommend that the archive log files be recalled from tape to disk before starting Capture.

A possible consideration with Capture Version 8 is when Capture is stopped while there is a long-running transaction in progress involving registered tables. When Capture is restarted, even if it is right after it stopped, it will ask DB2 for log records from a point in the log where the long transaction started.

In capshema.IBMSNAP_REGISTER record where GLOBAL_RECORD='Y', the SYNCHTIME value gives an indication of the timestamps in the log records that Capture is currently reading. This value can be compared with current system time to find out how far Capture is behind. We will discuss this more under Capture latency.

SLEEP_INTERVAL Capture parameter

If Capture receives from DB2 the most current log record and, because update activity in DB2 is low, there is not another log record immediately available, Capture goes to sleep (i.e. becomes inactive) for an interval as specified by the SLEEP_INTERVAL Capture parameter. In a DB2 for z/OS data sharing environment, Capture goes to sleep if the log record provided by DB2 does not have an efficient amount of data for Capture to process; usually, this means that the log buffer from DB2 is less than half full.

Without having a SLEEP_INTERVAL, the altrnative would be for Capture to flood DB2 with requests for the next log record until there is enough activity in DB2 for DB2 to have another log record to give to Capture.

The default value for SLEEP_INTERVAL is 5 seconds. It can be set to a different value through the CAPPARMS table, by starting Capture with a different value, or

by changing the current value using '**asnccmd...chgparms sleep_interval *n***' where '*n*' is the number of seconds you want Capture to be inactive if it gets to the end of the DB2 log.

DB2 for z/OS data sharing log-merge and Capture

If there is low data-update activity in a DB2 for z/OS data sharing group so that Capture becomes current and goes to sleep, then when Capture becomes active again and asks DB2 for the next log record, DB2 for z/OS will have to merge the logs of the members of the sharing group to give Capture the next log record to see if it has enough data in it to be worth processing. The DB2 for z/OS data sharing log merge process can take a lot of resources, particularly CPU.

If there are periods of low update-activity across a DB2 for z/OS sharing group you may want to consider doing one of the following:

- specifying the value for the sleep_interval parameter when Capture is started, or specifying a value for Sleep_Inteval in the capschema.IBMSNAP_CAPPARMS table.
- changing the Sleep_Interval of a running Capture by using the command '**asnccmd...chgparms sleep_interval *n*'** where '*n*' is the number of seconds you want Capture to sleep when it becomes current in reading the DB2 log.

10.2.2 Reading iSeries Journals

On iSeries, Capture uses Journal Receivers to read the records in the journals for the source tables. Capture can read from journals that are journalling changes for other tables that are not replicated.

If the target system for replication is another iSeries system, consider using remote journalling to the target system. See the discussion earlier in 10.1.3, "iSeries-to-iSeries replication with remote journalling" on page cdxliv.

10.2.3 Collecting transaction information in memory

If Capture can hold all the insert/update/delete log records for all inflight transactions for registered tables in memory, then performance should be optimal. If it cannot, than it will have to 'spill' some of the current and all new incoming transaction records to the I/O system. Capture will do this until some of the transactions currently tracked in memory complete and their memory can be re-used. On Linux, UNIX, and Windows, Capture spilling to the I/O system guarantees disk I/O activity. On z/OS, where Capture spills to Virtual I/O (VIO), Capture spilling will not cause actual disk I/O activity unless VIO does not have enough memory to contain the Capture transaction files.

To reduce the number of concurrent transactions that Capture has in memory and the amount of memory needed, and therefore the likelihood of Capture spilling transactions to file, consider the following:

- Have the applications that update the replicaiton source tables commit more frequently. We know this may not be an option in many cases, but where it is, it will help. Reducing the commit scope of the source applications also has implications for Apply since Apply must fetch all the changes made by a single transaction at once, no matter how small the blocking factor (MAX_SYNCH_MINUTES) is. Also, if Apply is using transational replication for a Subscription Set (i.e. COMMIT_COUNT for the set has a value), Apply has to apply all the changes from the same source transaction to a target table in the same commit, even if COMMIT_COUNT=1.
- Running multiple Captures so that the total number of transactions simultaneously in flght are handled by more than one Capture each with its own memory_limit. Before configuring and starting a second Capture consider the implications of two Captures operating simultaneously. A single Apply Subscription Set can replicate from only one Capture Schema and CD and UOW tables that the two Captures are inserting into should be configured to avoid contention between the two Captures. There may be other considerations as well.

Capture's memory usage can be monitored, for instance to detect if it is close to having to spill transactions to I/O or has been spilling to I/O. This is indicated in the records Capture writes to the capschema.IBMSNAP_CAPMON table in the columns:

- MONITOR_TIME - time the record was written to the CAPMON table
- CURRENT_MEMORY - Capture's current memory allocation in KiloBytes
  - Note: this value is in KiloBytes, but the Capture parameter Memory_Limits is in MegaBytes.
- TRANS_PROCESSED - number of transactions processed since the last CAPMON entry.
- TRANS_SPILLED - number of transactions spilled to I/O since the last CAPMON entry
- MAX_TRANS_SIZE - largest transaction since the last CAPMON entry.

This information from the CAPMON table can be displayed in the Replication Center. In the left window, expand **Replication Center --> Operations --> Capture Control Servers**, highlight the Capture control server server in the right window, right-mouse click and select **Show Capture Throughput Analysis**. The *Capture Throughput Analysis* window will open. In the upper part of the window, select the *Capture Schema* for the Capture you want throughput numbers for and **Memory Usage** from the *Information to be displayed* field. You can select *From* and *To* time periods in the window fields below, or accept the defaults and select time intervals if you don't want the display to have one record for each record in

the CAPMON table. Then click **Retrieve** from the buttons at the bottom of the window and the memory usage information will be displayed in the result-display area at the bottom of the window.

Note, in the Memory Usage display. the memory used by Capture will be displayed in KiloBytes though Capture's memory_limit parameter is specified in MegaBytes.

To see the Transactions Processed and Spilled information in the Replication Center, at the top of the Show Capture Throughput Analysis window, select **Number of Transactions Commited** in the *Information to be displayed* field and then **Retrieve** at the bottom of the window.

We found when occasionally when trying to switch from **Transactions Committed** to **Memory Usage** in the *Show Capture Throughput Analysis* window that we had to close the window and re-open from the options available for a source server in the rightside of the main Replication Center window.

The frequency with which Capture writes records to the CAPMON table is determined by Capture parameter MONITOR_INTERVAL; the value is specified in seconds and the default shipped value is 300 (5 minutes). In times when there is a possibility of Capture running out of memory, you might lower the Monitor_Interval so that Capture rights records to the CAPMON table more frequently and set a Monitor Condition for CAPTURE_MEMORY so that an alert will be sent when a current_memory threshold is reached.

MEMORY_LIMIT Capture parameter

The maximum memory that Capture can use to hold insert/update/delete log records is determined by Capture's Memory_Limit parameter. Default shipped value is 32 Megabytes. This can be over-ridden by a value in the capschema.IBMSNAP_CAPPARMS table and that in turn can be over-ridden by specifying a different value for memory limit when Capture is started. The Capture Memory_Limit can also be changed while Capture is running by using the **asnccmd** command with **chgparms** and specifying a new Memory_Limit.

Note: On z/OS, it is recommended that a memory limit not be specified on Capture's job card and that Capture's memory_limit parameter be used to indicate how much memory Capture is allowed to use. That is, on Capture's job card, specify REGION=0M

Capture's current setting for Memory_Limit can be determined using the asnacmd command, specifying the Capture_Server, Capture_Schema, and **qryparms** . This command can be issued by the Replication Center if the Capture Server supports DB2 Administration Server (DASe). In the Replication

Center, highlight a particular Capture Server, right-mouse click and select **Change Operational Parameters**. In the *Change Operational Parameters* window, select the *Capture Schema* in the upper left corner, and the shipped defaults, CAPPARMS values, and current values currently in effect should be displayed for that Capture. It may take a few minutes for Replication Center to retrieve and fill-in the current settings.

Capture's memory limit can be changed several ways:

- **asnacmd**, specifying Capture_Server, Capture_Schema, 'chgparms Memory_Limit=*new value'*

  For example:

  ```
  asnacmd capture_server=AIX43DB2 capture_schema=MIXCAP chgparms
  Memory_Limit=64
  ```

  will change Capture's Memory_Limit to 64 MegaByes.

- in Replication Center using the *Change Operational Parameters* window described above. Change the current value for Memory Limit and press OK or Apply in the lower right corner. Replication Center will first generate the **asnccmd...chgparms** command in a *Run Now or Save SQL* window and then you can execute it. Note: If you don't have DB2 Administrative Server (DASe) at the Capture Server, then you can't do this with the Replication Center.

If Capture spills transactions to I/O

If Capture runs out of memory for holding inflight transactions, it will create files to hold the overflow from the current transactions that are causing the out-of-memory condition, and will write each new transactions out to file until enough of the current transactions in memory have completed to free up memory for new transactions. Raising Capture's Memory_Limit via 'asnccmd ...chgparms' will not cause Capture to retrieve the currently spilled transactions from disk and put them all in memory, but it will give Capture memory so that new transactions will not be spilled to I/O.

When Capture spills transactions to I/O, it creates a file for each separate transaction that it is collecting log records for. Capture removes each file when Capture reads the commit for the transaction from the DB2 log and has inserted records for the transaction into the CD tables and into the IBMSNAP_UOW table.

On z/OS, these files are created in VIO. When VIO is out of memory, VIO will write files to disk. Mapping VIO to disks with fast read/write access and no contention will help Capture performance.

An alternative on z/OS, if you don't want Capture to spill to VIO, is to have Capture spill transactions to a directory file specified on a CAPSPILL DD card in

the Capture start JCL. Either UNIT=VIO or UNIT=SYSDA can be specified on this CAPSPILL DD card. Specifying UNIT=VIO on such a card would be equivalent to taking the default. UNIT=SYSDA would permit specifying a particular directory where you want Capture to create transaction spill files. In any case, the transaction files created by Capture will be temporary, and each file will be deleted by Capture after it has inserted the data from the file into the CD tables.

Here is an example of what a CAPSPILLS DD card might look like in the Capture start JCL:

```
//CAPSPILL DD DSN=&&CAPSPL,DISP=(NEW,NEW,DELETE),
// UNIT=SYSDA.SPACE=(CYL(50,100)),
// DCB=(RECFM=VB,BLKSIZE=6404)
```

On Linux, UNIX, and Windows, Capture will create the spilled transaction files in the file system directory specified by the Capture_Path start parameter. Their file names will be their transaction ID and extension '.000.' Capture will remove each file when it receives a commit from the DB2 log for the transaction and has inserted records for the transaction into the CD tables and the IBMSNAP_UOW table. Captures performance will be better if CAPTURE_PATH specifies a file system directory where there will be fast read/write access and no contention when Capture has to spill transactions.

10.2.4 Capture Insert into CD and UOW tables

Inserting records into CD tables and into the capshema.IBMSNAP_UOW table should be the gating factor for Capture's throughput, assuming that the DB2 log-read and management of inflight transactions discussed above can be done through memory. And the main factors affecting Capture's performance inserting into the CD tables and UOW table will probably be the characteristics of these tables themselves. Their characterisitcs also affect Apply's performance in fetching changes to be applied to replication targets and Capture's performance when pruning replicated changes from these tables. Therefore, the characteristics of the CD tables and UOW table will be disucssed in a separate topic.

With DB2 Replication Version 8, it can be expected that Capture will be inserting records into the CD tables in groups as Capture detects from the log-read the commit associated with a transaction involving registered tables. For short transactions that udpate one or a few tables and commit frequently, there should be just a few inserts to a few CD tables. For long transactions that update one, few, or many tables and commit infrequently, there should be a large number of inserts, possibly to many CD tables, when Capture detects the commit for a transaction involving registered tables.

With Replication Version 8, there are several occurrences where Capture, when if detects the commit for a transaction involving registered tables, will not insert records into CD table for every insert/update/delete record for a registered table:

- CHGONLY='Y' in the registration for a source table, and none of the columns updated match any of registered columns, which are the 'data' columns of the respective CD table. If not all the columns of source table are being made available for replication, then CHGONLY on the registration for this table could save INSERT activity on the CD table and, in turn, Apply update activity on the target.

  However, the reduction in CD table size and insert activity to the CD table does have a cost, which is the CPU that Capture will use to compare the before and after image of the column values in the log record to determine whether any of the columns with different before and after images in log or journal match the registered columns. If all,or even most, of the columns of a source table are registered, then CHGONLY on the registration will increase Capture's overhead unnecessarily; Capture is going to insert a record into the CD table anyway for almost every update to the source table.

  To find out if a registration has CHGONLY='Y':

  - In the Replication Center left window, select **Replication Center-->Replication Definitions-->Capture Control Server-->*capture control server name*-->Registered Tables.** Highlight the registered table in the right window, right-mouse click, and select **Properties.** In the *Registered Table Properties* window, at the top of the Definition tab, check the *'Row-capture rule* field's value. 'Capture changes to all columns' (default) means CHGONLY='N'. 'Capture changes to registered columns' means CHGONLY='Y'.
  - or query the capschema.IBMSNAP_REGISTER table at the Capture Control Server for the record for the source table and check the value of the CHGONLY column.

- RECAPTURE='N' in capschema.IBMSNAP_REGISTER record for a source table, and Apply replicates into the source table. RECAPTURE='N' tells Capture that if it detects that the application that insert/update/deleted the source table is Apply, don't insert any records into the CD tables. The application updating a replication source table could be Apply when:
  - using a three-tier replication environment where one Apply replicates to an intermediate User Copy table that is registered, and then another Apply replicates to down-stream user copies. If this is the case, then RECAPTURE='Y' should be set so that the first Apply's changes will be captured for replication by the second Apply.

    Actually, a more efficient design might be to have the intermediate table be a CCD, which would have all the control information needed by the down-stream Apply and avoid the overhead of an intermediate Capture.

- the replication source table is a Master or a Replica in a Update Anywhere replication configuration. RECAPTURE needs to be =Y' only if the changes made the source table by Apply need to be replicated to other table. This is the case for the Master if changes made at one Replica need to be replicated to other Replicas via the central Master. RECAPTURE can be 'N' at Replicas if they will not be used for replicating to other down-stream copies of the table, and RECAPTURE can also be 'N' for Replicas in a Peer-to-Peer update-anywhere configuration.

To find out if changes made by Apply are being recaptured:

- In the Replication Center, open the *Registered Tables Properites* window for a registered table as described above. At the bottom of the Defintion tab, see if **'Capture changes from replica target tables** is checked.
- Or, query the capschema.IBMSNAP_REGISTER table at the Capture Control Server for the record for the source table and check the value of the RECAPTURE column.

- 'before' triggers on CD tables to tell Capture not to insert. It is possible to create a 'before' trigger on a CD table that evaluates the values of an insert and returns control to Capture without doing an insert if certain conditions are met. For instance, if you don't want to replicate deletes, the before trigger on a CD table could detect the IBMSNAP_OPERATION of the insert into a CD table and return control to Capture if IBMSNAP_OPERATION='D'.

There are also conditions where Capture will insert more than one record in the CD tables for each update of the source tables.

- CHG_UPD_TO_DEL_INS='Y' in a registration
  - don't use this feature unless you really need for Apply to delete a record at one target table or in one partition of the target table and insert a record in another target table or in another partition of the target table as a result of the update. This feature will cause Capture to insert two records into the CD table for every update of the source table and for Apply to perform both a delete and an insert at the target table for every update of the source table.
    - If the source application updated a primary key/unique index column of the source table, and you want to effect the same update at the target table, register the 'before image' of the source table's primary key/unique index columns. Then on Subsription Members from this source table specify TARGET_KEY_CHG. With this combination, Capture will do only one insert into the CD table for each source table update, and Apply will only do one update at the target table for each update that was performed on the source table.

To find out if source table updates are captures as an 'delete' record and an 'insert' record in the CD table

- In the Replication Center, open the *Registered Tables Properites* window for a registered table as described above. At the bottom of the Defintion tab, see if **Capture updates as pairs of deletes and inserts** is checked. The default is for this not to be checked.
- Or, query the capschema.IBMSNAP_REGISTER table at the Capture Control Server for the record for the source table and check the value of the CHG_UPD_TO_DEL_INS column.

Decreasing row-length of inserts into CD table

There is an 'advanced technique' for improving Capture's throughput, and saving space of CD tables, by only registering the primary key/unique index columns of a source table and having Apply replicate from a view that joins the CD table with source table itself to get the values it needs for all the columns of a target table. With this technique, the CD table takes less space, Capture's insert won't take as long since it won't have as many values, and Capture's pruning deletes from CD table won't take as long since the number of values removed won't be as great. We won't go into all the details of setting up this technique but will point out some issues associated with it:

- This technique bypasses one of DB2 Replication's normal data integrity rule for making the data in the target table consistent with the source table data *as of specific points in time*. With this technique, the primary key/unique index values replicated will be at best seconds behind the data in the source table. But the non-key/non-unique index values are the current values from the source table itself. In many cases this won't matter; the users wanted the most current values from the records of the source table anyway. But it is worth bringing to attention if you are considering using this technique.
- This technique also adds a join to Apply's fetch of new changes at the source server. The join between the CD table and the source table will have to be on the primary key/unique index columns of the source table. The corresponding columns of the CD table should be included in the unique index of the CD table.
- definitely do NOT specify CHGONLY in the registration of the source table. The technique assumes that Capture will insert a record into the CD table for every update of the source table, but the CD table only has the primary key/unique columns which will probably not be changed by the update of the source table.

COMMIT_INTERVAL Capture parameter

At the Commit_Interval, Capture stops inserting changes into the CD tables and IBMSNAP_UOW table and does the two following steps:

1. updates the restart information in the IBMSNAP_RESTART table regarding Capture's current point in the DB2 log, and issues a commit.

- This commit also commit the records inserted into the CD tables and UOW table since the last ‘commit_interval’ processing.

2. updates the new-change signals in the IBMSNAP_REGISTER table and issues a commit.
   - The new-change signals in the IBMSNAP_REGISTER table are the first information read by Apply when it connects to the Capture Control server and tell Apply whether it should prepare and execute selects from the CD tables.
   - We’ll provide more detail about the ‘new-change’ signals in the section on achieving Low-Latency replication.

Then Capture returns to reading the DB2 Log and inserting records into the CD tables and UOW table.

Apply will not fetch new changes from the source server any more frequently than Capture updates the new-change signals in the IBMSNAP_REGISTER table. But Capture’s Commit_Interval processing does ‘interfere’ with Capture’s throughput in inserting into CD tables and Capture’s ability to keep up with the applications that are updating the source tables. So a balance may need to be struck when trying to achieve low end-to-end latency replication. If Capture is setting the new-change signals too frequently, its overall insert throughput to the CD tables may not be able to keep up with source applications updates of the soure tables, which will be counter-productive to achieving low end-to-end latency.

iSeries: COMMIT_INTERVAL and FRCFRQ

For Capture on iSeries, Capture commits changes from the journals to the CD tables, making the new changes available for Apply, at a frequency determined by the COMMIT_INTERVAL value in capschema.IBMSNAP_CAPPARMS. The COMMIT_INTERVAL specified in CAPPARMS can be over-ridden when Capture is started (STRDPRCAP) by specifying parameter FRCFRQ with a value when Capture is started. Both COMMIT_INTERVAL and FRCFRQ are specified in seconds. The range for FRCFRQ is 30-600 seconds.

10.2.5 Capture Pruning

With DB2 Replication Version 8, Capture’s pruning of CD tables still is asking DB2 at the source server to perform DELETEs on records in the CD tables and UOW table. But wtih Version 8, there are several features that make this process more efficient and less disruptive to Capture’s overall throughput:

- separating Capture into separate threads for inserting into CD/UOW tables and for pruning means that Capture does not have to stopped the flow of changes going into the CD tables in order to prune. Therefore, Capture can

prune the CD and UOW tables during periods of activity on source tables without affecting Capture's latency.

Note: Locksize on CD tables and IBMSNAP_UOW table should not be at table-level. If the locksize of the CD and UOW tables is at table-level, then Capture's PRUNING thread, which is deleting records form the CD and UOW tables, could slow down Capture's WORKER thread that is inserting changes into these tables.

On z/OS and OS/390, CD tables and IBMSNAP_UOW locksize should be set to PAGE or ANY. It should not be at row-level.

On Linux, UNIX, and Windows, CD tables and IBMSNAP_UOW locksize can be at row-level, which is the default LOCKSIZE when a table is created.

If you wish, you can still defer the pruning workload by starting Capture with NO_PRUNE so that it does not automatically prune. Then **`asnccmd...prune`** can be used to make Capture prune at a specific time.

- since the CD table now contains the IBMSNAP_COMMITSEQ values for each record, Capture can prune CD tables without joining them with the IBMSNAP_UOW table.
- Capture performs intermediate commits when deleting from CD tables, and from the UOW table. Fewer rows will be locked in the CD tables at once and shorter strings of deletes will be logged per transaction when Capture is pruning.

The time when Capture will prune is controlled by two of the Capture operations parameters, which can be entered when Capture is started, can be changed with **asnccmd ...chgparms** or can be specified in the capschema.IBMSNAP_CAPPARMS table.:

- AUTOPRUNE
  - if 'N', Capture only prunes when it receives the prune command via 'asncmmd...prune'
  - if 'Y', Capture prunes at the Prune_Interval
- PRUNE_INTERVAL
  - interval, in seconds, between times when Capture is to prune, if AUTOPRUNE='Y'. Default setting is 300 seconds (5 minutes).

Capture's performance when pruning will largely be governed by the characteristics of the CD tables and the IBMSNAP_UOW table. Since their characteristics also affect Capture's insert throughput and Apply's performance when fetching changes to be replicated, the characteristics of the CD tables and the UOW table are discussed in a separate topic.

Capture pruning activity is recorded in records Capture writes to the capschema.IBMSNAP_CAPTRACE table. In the CAPTRACE records for pruning, the first 8 characters of the DESCRIPTION value will be ‘ASN0105I’. In the characters farther on in the DESCRIPTION value will be the number of rows pruned.

The number of rows pruned each time Capture has pruned can also be seen via the Replication Center. In the left window, expand **Replication Center --> Operations --> Capture Control Servers**, highlight the Capture control server server in the right window, right-mouse click and select Show Capture Throughput Analysis. The Capture Throughput Analysis window will open. In the upper part of the window, select the Capture Schema for the Capture you want throughput numbers for and ‘Number of Rows Pruned from CD tables’ from the ‘Information to be displayed’ field. Fill in the From- and To- time in the middle of the window or take the default and then press Retrieve from the buttons at the bottom of the display. In the display window, one record will be display for each time that Capture pruned, giving the date and time that Capture pruned and the number of rows pruned from all CD tables.

If when you press Retrieve you get an error message SQL0100W ‘No Records found...’ it is possibly because there are no records in the capschema.IBMSNAP_CAPTRACE starting with ‘ASN0105I’ which suggests that Capture has not pruned recently.

DB2 resources for number of locks

Because the Capture pruning thread could be deleting records from the CD and IBMSNAP_UOW table at the same time that the Capture worker thread is inserting into these tables, we recommend that the locking level for CD and UOW tables not be at table-level or higher. On z/OS or OS/390, the locking level on CD and UOW tables should be PAGE or ANY; on Linux, UNIX, and Windows, it can be ROW. Therefore, Capture could be making a lot of individual lock requests. We recommend that more recources be allocated in the DB2 system for locks in order to avoid having Capture’s row locks be escalated to table locks which will drastically slow down either pruning or inserting into the CD tables.

In DB2 on Linux, UNIX, and Windows, specify a large LOCKLIST value in the Database Configuration.

10.2.6 Capture’s latency

Capture’s latency can be determined from the entries in the capschema.IBMSNAP_CAPMON table. In each record, the MONITOR_TIME value is the current timestamp at the time Capture inserted the record into this table. The SYNCHTIME is the current SYNCHTIME value from the capschema.IBMSNAP_REGISTER GLOBAL_RECORD=’Y’; the source of this

SYNCHTIME value would be the timestamp in a DB2 log record; this would be Log record last read when Capture last did its Commit_Interval processing.

```
SELECT MONITOR_TIME, MONITOR_TIME - SYNCHTIME AS CAPTURE_LATENCY FROM
capschema.IBMSNAP_CAPMON
```

will give you a picture of Capture's latency over time.

Or Replication Center can provide the same information. In the left-window tree, select **Replication Center --> Operations --> Capture Control Server**. In the right window, highlight the capture control server you are interested in, right-mouse click, and select **Show Capture Latency**. The *Capture Latency* window will open.

Pick a *Capture Schema* at the top of the window, a *from*- and *to*-time in the middle of the window and/or an interval from the *Time Intervals* pulldown, and press **Retrieve.** Figure 10-6 is an example of the Replication Center's *Capture Latency* window for a Capture whose Commit_Interval is 30 seconds (the default), showing the latency statistics by hour.

Figure 10-6 Capture Latency window

As you can see, the Average Latency for this capture is less than the Commit_Interval, so this Capture is keeping up with the changes to source applications.

10.2.7 Capture's throughput

The number of records that Capture is inserting into CD tables is recorded by Capture in the records it inserts into the capschema.IBMSNAP_CAPMON table which we also mentioned above regarding indicators of Capture's memory usage and spilling transactions to disk. For monitoring Capture's throughput the relevant columns of the capschema.IBMSNAP_CAPMON table are:

- MONITOR_TIME - the date and time when Capture inserted into CAPMON
- CD_ROWS_INSERTED - rows inserted into all CD tables during the interval

- CHG_ROWS_SKIPPED - update records evaluated but not inserted into CD tables because the registration had CHGONLY='Y' and the updated column was not among the columns of the CD table.
- RECAP_ROWS_SKIPPED - insert/update/delte records evaluated but not inserted into CD tables because the registration for the source table has RECAPTURE='N' and the application that update the source table was Apply
- TRIGR_ROWS_SKIPPED - insert/update/delete records evaluated but not inserted into CD tables because a before trigger on the CD table returned control to Capture without performing the insert to the CD table.

The above numbers in the CAPMON table can be displayed in the Replication Center. In the left window, expand **Replication Center --> Operations --> Capture Control Servers**, highlight the Capture control server server in the right window, right-mouse click and select **Show Capture Throughput Analysis**. The *Capture Throughput Analysis* window will open. In the upper part of the window, select the *Capture Schema* for the Capture you want throughput numbers for and **Number of Rows inserted from log or skipped** from the '*Information to be displayed*' field. You can select *From* and *To* time periods in the window fields below, or accept the defaults and select time intervals if you don't want a record in the display result for each CAPMON record. Then click **Retrieve** from the buttons at the bottom of the window and the throughput information will be displayed in the result-display area at the bottom of the window.

10.3 CD tables and the IBMSNAP_UOW table

The three major operations on the CD tables and UOW table are:

1. Capture inserting records for source table transactions
2. Apply fetching changes to be applied at target tables
3. Capture Pruning

The characteristics of the UOW table will not be important for Apply's replication of changes if

- all your target tables are User Copies or Replicas and
  - you don't source any table columns from IBMSNAP_UOW table columns
  - you don't use any IBMSNAP_UOW columns in member predicates (WHERE clauses)

If this is the case, then Apply will not touch the UOW table at all. However, if the any of your target tables are CCD's or Point-in-Time tables, then Apply will need to involve the UOW table since the values for some of the replication-information columns in CCD and Point-in-Time tables come only from the UOW table.

The CD tables and the IBMSNAP_UOW table should have

- only a single index each so that Capture's insert and pruning (deleting) performance will only cause DB2 to have to write /remove one record in the CD table's tablespaces and one record to the CD's tables index's tablespaces. On z/OS, the indexes should be Type 2. Replication Center will create the IBMSNAP_UOW table and the CD tables with only one index. If for some reason you want more columns to be indexed (such as for Subsription Member predicates used by Apply), add them to the index that was defined by the Replication Center; don't create a second index.
  - the UOW table is created when the Capture Control Tables are created. A CD table is created when a source table is Registered. The Create Table DDL that Replication Center generates to create the UOW table and CD tables will include the Create Unique Index statement for the UOW/CD table indexes; the Create Unique Index statement will specify the columns that are used by Capture to find the records to prune and by Apply to find and order the records to fetch.
- place CD table's index in a separate tablespace from the CD table so that Capture's insert and pruning activity won't have to go back and forth between two places in the same tablespace file as it, say, inserts a record in the table and inserts a related record in the index, then insert next record in the table and the its related reoord in the index.
- set locking for the CD and UOW table at row-level so that Capture's worker thread can insert records into the CD and UOW table at the same time that

Capture's pruning thread is deleting records. With DB2 Replication Version 7 and before, the recommendation was for table-level locking since Capture did not insert and prune at the same time.

- if Apply will be replicating at short intervals from the CD table, create a bufferpool for all the CD tables and for the IBMSNAP_UOW table so that when Apply asks the source DB2 for records from the CD (and at times that it is joined with the UOW table), the source DB2 can fulfill the requests from records still in bufferpool pages from when Capture just inserted the records into the CD tables and UOW table.
- run RUNSTATS only once for each CD table and for the IBMSNAP_UOW table when they are full of records. Both Apply's fetch, and Capture's pruning deletes have WHERE clauses for the columns that are indexed in the CD and UOW table, and are dynamically prepared. If the statistics indicate that the CD tables and IBMSNAP_UOW tables are large, DB2 will use the indexes on the CD tables and UOW table to perform the fetch for Apply and the pruning deletes for Capture. If the statistics indicate these tables are small, such as when RUNSTATS is done right after Capture pruning, then DB2 will use tablescans to process Apply's fetch and Capture's pruning.

Attention: If RUNSTATS is performed regularly for all tables in a DB2 on z/OS, Linux, UNIX, or Windows, it is recommended the the CD and IBMSNAP_UOW table be excluded from the tables upon which this is done. It is important for replication performance that the source server have relatively accurate statistics for the CD and UOW tables, but if the CD and UOW tables are included in a list of tables upon which regular RUNSTATS is performed, there is the distinct possiblity that these tables will have few records when RUNSTATS is performed; but later in the day, after much activity in the DB2 system, these tables could be large. Since the statistics will indicate they are small, DB2 will use table scans, instead of index access, when Apply replicates from these tables. Even if there are only a few records for Apply to fetch at each iteration, DB2 will have to scan the whole CD (and UOW) tablespaces each time. There are many reports of DPROP having high CPU and many page fetches; most frequently these occurrences can be attributed to the statistics for the CD and UOW tables indicating these tables are small when in fact they are large.

- on z/OS and OS/390, if there is many updates being replicated, the REORG utility can be run for the CD and IBMSNAP_UOW tables weekly. the PREFORMAT option should be specified.
- On iSeries, CD tables and the IBMSNAP_UOW table should be re-oganized periodically to reclaim unused space, such as space no longer needed for records that have been pruned by Capture. The CL command to do this is RGZPFM.

RGZPFM can be performed by Capture when it stops if RGZCTLBL *YES is specified with the End-Capture command (ENDDPRCAP).

If RGZPFM is done manually (i.e. not by Capture), then you will need to find out the library and short file name for the CD table to specify with RGZPFM command. This information can be obtained from System Catalog View QSYS2.SYSTABLES using the CD_OWNER and CD_TABLE values for the CD table from the IBMSNAP_REGISTER table. The query can be entered using STRSQL. The query would look like this:

```
SELECT SYSTEM_TABLE_SCHEMA,SYSTEM_TABLE_NAME
FROM QSYS2.SYSTABLES
WHERE TABLE_OWNER='cd_owner' AND TABLE_NAME='cd_table'
```

The SYSTEM_TABLE_SCHEMA and SYSTEM_TABLE_NAME values in QSYS2.SYSTABLES are the library and short file name for the CD table.

10.4 Apply Performance

The Apply operations we'd like to discuss are:

1. fetching changes from the source server to the spill file
2. inserting, updating, deleting from the spill file to the target server

We expect that inserting, updating, deleting to the target tables should be the gating factor in Apply performance since Apply must do these operations one spill-file record at a time. The fetch from the source server should, in comparison, be much faster since Apply is able to fetch records from the source server in multi-record blocks.

There are things that could make the insert/update/delete to the target tables even slower, such as when Apply is on a different system than the target table and has to 'push' the changes to the target server over a network.

When replicating to a non-DB2 server, such as to Informix, though Apply may be on the same system as the nicknames for the target tables, if the Informix server that contains the target tables themselves is on a different system, then, though for Apply it is a 'pull' scenario, federated server will have to 'push' each change across a network to Informix.

10.4.1 Fetching changes from the source server

To fetch changes from the source server, Apply connects to the source server using an SQL Connect statement, determines from the capschema.IBMSNAP_REGISTER table if there are any new changes for replication, and then executes SELECT statements against the CD tables to fetch the changes. If any target tables are CCD's or Point-in-Time type targets, or if the target tables have columns sourced from the IBMSNAP_UOW table, or member predicates on columns of the UOW table, then the SELECT to obtain new changes will involve a join between each CD table and the UOW table.

In any case, DB2 at the source server will do most of the work, executing the SELECT statement's sent by Apply.

The SELECT statements from the CD tables will be dynamically prepared by the DB2 source server each time that Apply replicates. Statistics about the CD tables in the DB2 source servers catalog will help the source server's optimizer determine whether to use the CD tables index, or to do a table scan, when executing the SELECT statement for Apply. The statistics do not have to be 100% accurate; they just need to provide an estimate of the characteristics of the CD tables, just enough, really, to help the optimizer make the best decision on whether to use the index or not. The characteristics of the CD tables, and the

statistics about them in the DB2 system catalog at the source server, will dramatically affect how long it takes the source server to execute Apply's select statement, and how many resources, particularly CPU and page-fetches, are required.

The SELECT statement from Apply will have a WHERE clause which will include the IBMSNAP_COMMITSEQ column of the CD table. The SELECT statement will also include an ORDER BY clause including both the IBMSNAP_COMMITSEQ and IBMSNAP_INTENTSEQ columns of the CD table. The ORDER BY clause will put the records from the CD table into the Apply spill file in the order of the SQL operations on the source table. This way updates will be applied to the target by Apply in the same order they were performed on the source table; thus if the same row of the source table was updated during a replication interval, the last update to the source table will be the last update in the spill file, and the last update Apply will perform on the target. The unique index that was created by the Replication Center when the source table was Registered would have included both of the COMMITSEQ and INTENTSEQ columns.

If the target table is a CCD or a Point-in-Time type, or if a user copy with columns sourced from the IBMSNAP_UOW table, or if there are member predicates referencing UOW columns, then the SELECT statement sent the source server by Apply will involve a join between the CD and the UOW table on the IBMSNAP_COMMITSEQ column in both these tables. When the Replication Center created the Capture Control tables, the generated SQL included a Unique Index for the UOW table that included the IBMSNAP_COMMITSEQ column. Statistics that are a good estimate for the capschema.IBMSNAP_UOW table in the DB2 source server catalog will help the DB2 optimizer make a good decision on whether to use the index on the UOW table or not.

If a particular Subscription Member includes WHERE clauses for columns of the source table or columns of the UOW table, then adding these columns the index of the CD table and UOW table respectively could also affect performance of Apply's SELECT statement at the source server.

Apply's SELECT from the CD tables and from the UOW table should be executed with isolation Uncommited Read (UR, 'Dirty Read') so it will not be affected by any locks on the CD table and UOW table by Capture, which is inserting records into these tables. Apply's using UR on this SELECT won't compromise the integrity of the replicated data since Capture won't begin to insert any records into the CD and UOW table related to a transaction involving the respective Registered source tables until Capture has seen a commit for that transaction in the source server's DB2 log. The use of UR on the SELECT from the CD and UOW table was determined when Apply's packages were bound at the source server.

Apply's SELECT statement from the CD tables and UOW should also be executed with blocking so that multiple records can be transported from source server to Apply's spill files in a single block and in a single network packet.

Apply on Linux, UNIX, and Windows autobinds at the Capture Control Server if it can't find its packages the first time it connects to the source server. Apply's autobind for these packages specifies isolation and blocking as appropriate. Most of Apply's packages are bound with ISOLATION UR; one is bound with ISOALTION CS. All are bound with BLOCKING ALL.

Apply on z/OS and iSeries does not autobind its packages.

For zOS, sample JCL for binding Apply packages and plans are shipped with IBM DB2 DataPropagator for z/OS Version 8. The sample JCL for binding each package specifies correctly whether the package should be bound with ISOLATION(UR) or (CS). The packages used for fetching data from the CD and UOW tables specify ISOLATION UR; the default blocking behavior for fetching data when ISOLATION UR is to fetch multiple records in a block.

When replicating from a non-DB2 server, such as from Informix, Apply will be sending the federated server SELECT statements referencing the nicknames for the CCD tables. Federated server will in turn send Informix a select statement for the ccd tables. Federated server will request that the results of this select be blocked. Since the results are blocked across the network to from Informix to the federated server and from the federated server to Apply, there will in most cases be little advantage to the federated server being close to the Informix server or close to the Apply system. If the target server is a DB2 ESE Version 8 on Linux, UNIX, or Windows, there will be slight advantage to letting the target system also be the 'Capture Control Server' containing nicknames for the source tables in Informix; with this configuration, the results from Informix are only blocked once since they go straight from Informix to the system with Apply. If the federated server and the target server are different system - which is unfortunately required when the target server is DB2 for z/OS and iSeries - then the same result will be blocked twice: first from Informix the federated server (DB2 ESE or DB2 Connect on Linux, UNIX, or Windows) and from the federated server to the target server.

10.4.2 Caching Apply's dynamic SQL at source servers

Apply's fetch from a CD table will be dynamically prepared each time Apply connects to the source server, but will always have the same form every time.

The same columns will be referenced in the SELECT and the same columns will always be referenced in the WHERE clause of this select.

If the package the source server prepared the last time Apply fetched from a CD table is still in memory, the source server will re-use that package rather than execute Apply's prepare request.

Note: Also, for DB2 for z/OS and OS/390 to keep Apply's dynamically prepared SQL statements in memory for use by Apply the next time Apply connects to a DB2 for z/OS or OS/390 subsystem or sharing group, Apply's packages need to be bound with BIND PACKAGE option KEEPDYNAMIC(YES) specified. The sample BIND PACAKGE JCL provided with Apply on z/OS should include this option.See *DB2 for z/OS and OS/390 Command Reference* for a description of bind options.

10.4.3 Specifying more memory for caching dynamic SQL packages

DB2 for z/OS and OS/390 and DB2 for Linux, UNIX, and Windows both have facilities for keeping the packages of previously prepared dynamic SQL statements in memory.

In DB2 for z/OS and OS/390, on the Application Programming Defaults panel (DSNTIP4), indicate CACHE DYNAMIC SQL - YES. Howeer, if Dynamic SQL caching is enabled, more memory should be allocated for the EDM Pool which is used for other purposes besides caching packages for dynamic SQL; for instane, the EDM Pool is used for caching the packages for Static SQL. EDM Pool size is set on the CLIST CALCULATIONS panel (DSNTIPC). See *DB2 for z/OS Installation Guide* and *DB2 for z/OS Adminstration Guide* for more information.

In DB2 for Linux, UNIX, and Windows, a package for previously prepared dynamic SQL statement is automatically stored in memory and is kept there until the memory for this purpose needs to be used for the package of a more recently used SQL statement. The memory allocated for holding packages in memory, both for static SQL packages and for packages for dynamically prepared SQL statements, is indicated by the Database Configuration parameter PCKCACHESZ. The default value is -1, meaning that the memory alloation wil be a multiple of the MAXAPPLS parameter. You can specify with the PCKCACHESZ the exact amount of memory you want allocated in the range from 32 - 524,288 pages. Allocating more memory for PCKCACHESZ will increase the probability that when Apply connects again to source and/or target server, that the previously prepared package for Apply's dynamic SELECT statements will still be in memory and the delay associated with preparing these statement again will be avoided. See the *DB2 UDB Version 8 Administration Guide -Performance SC09-4821* chapter on Configuring DB2 for more information about PCKCACHESZ.

10.4.4 Transporting changes across the network

Hopefully transporting the changes from the source server to the spill file on the Apply system takes the least amount of time of any of the sub-operations performed by Apply.

The ‘packetizing’ of the records sent from the source server to Apply is done at more than one level. At the distributed database connectivity level there is the blocking of the results. At the network level, each distributed database block may have to be broken up into multiple network packets if the network packet size is less than the distributed database blocksize. For replication performance, we would recommend maximizing both.

For DB2 on Linux, UNIX, and Windows the distributed database blocksize is determined by the Database Manager Configuration parameter RQRIOBLK. The default value is large - 32,767 bytes- but can be even larger - 65,535 bytes. This block size is in effect when replicating between DB2 on Linux, UNIX, and Windows, and also when replicating from DB2 for z/OS or iSeries to DB2 on Linux, UNIX, and Windows.

When replicating from DB2 for z/OS to DB2 for z/OS, block size is determined by the Extra Blocks Req and Extra Blocks Srv on the DB2 for z/OS and OS/390 Distributed Data Facility installation panel (DSNTIP5). The default setting is the maximum allowed. See the *DB2 for z/OS and OS/390 Installation Guide.*

We won’t cover here how to set the network packet size. That should be investigated in TCP/IP documentation and discussed with your network administrators.

10.4.5 Apply spill files

The records that Apply fetches from the CD tables at the source server will be put in spill files on the system where Apply is running. There will be one spill file for each target tables. When replicating LOBs or datalinks there will also be a separate spill file for each LOB value or data link value fetched. The spill files are not persistent. Apply does not create a spill file until it has a result set from the source server to put in a spill file. At the end of each replication cycle, Apply erases the spill files.

Before discussing the spill files location in the Apply system’s file system, we’ll mention that the spill files do not have to be created on disk in two circumstances:

- on z/OS, the default location for the spill files is in memory. This is controlled by the Apply start parameter ‘spill_file’ which is also a column in the capschema.IBMSNAP_APPPARMS table, and by what is specified on a

ASNAPL DD card in the JCL that is used to start Apply. If the start parameter 'spill_file' specifies disk, then location of the spill files will be control by the ASNAPL DD card. If that card is not specified, then the default location for the spill files is VIO, then the spill files could still be in memory but that will be determined by resources available to VIO. If the ASNAPL DD card is included in the Apply start JCL, then the directory location specified on that card will govern where the spill files are created.

- when replicating from iSeries to iSeries, if remote journalling is used from the source server to the target server so that the CD tables can be on same system as the target tables, Apply can replicated directly from a CD table to the target table and avoid altogether the steps of open and writing to the spill file with the results from the source server and then opening and reading from the spill file to apply the changes to the target tables.

Apply will create its spill files in the directory indicated by the Apply_Path start variable. There should be adequate space in that directory for the spill files. If Apply is replicating at short intervals so that the volume of changes fetched is not great than not as much disk space may be needed for spill files. But if Apply is replicating at long intervals during which there could be lots of changes to the source tables, then the number of records fetched by Apply could be large and more disk space is needed for spill files. Row length of a CD table at the source server times the number of changes replicated each interval provides an estimate of the size of the spill file. If you are doing this calculation, don't forget the IBMSNAP_COMMITSEQ, IBMSNAP_INTENTSEQ, and IBMSNAP_OPERATION columns; there combined length is 21 bytes.

The Apply_Path should be selected so that there is minimal contention between Apply's I/O with the spill files and other activity on the system with Apply. For instance, if Apply is on the same system as the target tables - which is the optimal 'pull' replication system design for performance - then the Apply_Path should not be to the same disk as the DB2 log, the storage groups / containers of the tablespaces for the target tables or their indexes.

10.4.6 Apply updating the target tables

To apply the changes currently in the spill files to the target tables, Apply executes a Connect to the target server, insert/update/deletes to all the target tables in the set from all the records in the spill files, and then issues a commit. Thus, as with the fetch from the source server, the DB2 at the target server is really doing most of the work. The exception to the 'one-commit-at-end' rule will be discuss later with COMMIT_COUNT. In any case, Apply takes each record from each spill file and applies it to the appropriate target table with a single insert, update, or delete statement as appropriate, waiting for the result from the target server before processing the next spill file record. Apply has to wait for the result of each insert, update, or delete so it can determine whether an insert

needs to be reworked into an update or an update needs to be reworked into an insert before moving onto the next spill file record. And it needs to process the spill file records in order so that it guarantees that updates are applied in the order in which they were applied at the source table. That way, if the the same record in a source table was updated multiple times, the values of the last update to the sourde server will also be the values of the last update to that Apply does to the target table.

We'll also point out here that when Apply does an update at a target table, it sets values for all the columns not in the primary key or unique index. The CD table records at the source server for updates do not indicate which column was updated. To make sure the column with the change is also changed at the target, Apply just sets all the non-key/non-unique index columns of the target with the after-image values from the update record from the CD table.

Apply's insert/update/deletes to the target tables typically take up the largest slice of time of a replication cycle. And the amount of time taken to insert/update/delete the target tables can be expected to be even longer if Apply is not on the same system as the target tables and so each insert/update/delete, and its result back to Apply, has to traverse a network.

When replicating to a non-DB2 server, such as Informix, remember to think about whether Apply and the nicknames for the target tables are on the same system as the Informix server that contains the target tables themselves. If not, though Apply is configured for the faster 'pull' system design, each insert/update/delete that Apply does still has to cross the network from the federatd server where Apply is running to the Informix server.

10.4.7 Target table and DB2 log characteristics at the target server

The target server's performance of Apply's insert/update/deletes to the target tables will be affected by the usual factors for insert/update/delete performance. We'll list some of these factors:

- fewer indexes on the target tables will make for faster inserts and deletes since fewer underlying files will need to be updated. If the users' queries of the target tables have WHERE clauses referencing columns not in the primary key or unique index, consider adding these columns to the unique index over creating a second index on the target table.
- the tablespaces for the target tables and their indexes, and the characteristics of those tablespaces should be specified to optimize both Apply's insert, update, delete operations and the users' queries. The target table and its index should not be in the same tablespace so that when Apply inserts or deletes records in the target table, the target server won't have to perfom I/O two places on the same disk path to execute the insert or delete.

- RUNSTATS should be run for the target tables so that the optimizer at the target server will pick the fastest access plan for executing Apply's insert, update, and deletes, as well as the users' queries.
- the DB2 log files, the tablespace containers for the table, the tablespace containers for the index, and the Apply_Path diretory that contains the spill files should ideally be on separate disk drivers to minimize the I/O contention as DB2 reads from the spill file and writes to table tablespace, index tablespace, and log file.

10.4.8 Caching Apply's dynamic SQL at the target server

Apply's insert, update, and delete statements for each of target tables will be dynamically prepared each time Apply connects to the target table, but will always have the same form every time that Apply replicates into a particular target table

- the same columns will be in the insert statements
- the same columns in the WHERE clauses and SET clause of the update statements
- the same columns in the WHERE clause of the delete statements.

If Apply will be replicating a Subscription Set at short intervals or continuously, performance will be improved if the package created for the previous iteration is still available in memory. DB2 at the target server will use the package in memory and won't actually execute the prepare request from Apply.

Please see the discussion in "Specifying more memory for caching dynamic SQL packages" on page cdlxxi regarding how to allocate more resources in DB2 for caching dynamic SQL statement packages in memory. Also, see the note there about binding Apply's packages at DB2 for z/OS and OS/390 with bind option KEEPDYNAMIC(YES).

10.4.9 Querying target tables while Apply is replicating

If Apply will be replicating with low intervals, or continuously, to target tables while the users are at work and querying the target tables then there is the distinct possibility that the users could experience slow response time to their queries if some steps are not taken since Apply will be acquiring locks on the target table as it updates it. Some things to consider so that Apply and the users can coexist with good performance:

- if the user's analysis will not be compromised by using Uncommitted Read (UR, 'Dirty Read') consider specifying that for the users so they can read through Apply's locks

- specify a low level of l locksize on the target table so that there is less possibility that the users result will need a record that Apply currently has locked. For instance, on z/OS specify a locksize of PAGE or ANY, and on Linux, UNIX, and Windows use the default locksize which is row-level. Allocate more lock resources in the DB2 database. If there are not enough lock resources, then the row locks could be escalated to tables locks and cause the users to experience poor query performance. In DB2 for Linux, UNIX, and Windows, lock resources are specified at the database level wth the LOCKLIST parameter. Default and maximum values for LOCKLIST are: :
  - DB2 on UNIX: default = 100 4K pages; range = 4 - 60,000 pages
  - DB2 on Windows: default = 25 4k pages; range = 4 - 60,000 pages
- specify a lower, non-zero/non-null value for COMMIT_COUNT for the Apply Susbscription Set. Specifying a non-zero COMMIT_COUNT causes Apply to issue intermediate commits which will unlock all the target tables rows that were locked. Apply will starting locking another set of target table rows as it processes the next set of spill file records but release these locks as it performs the next intermediate commit. The intermediate commits caused by specifying a low/non-zero COMMIT_COUNT could slow Apply's over-all throughput and this should be balanced against the benefit of having Apply lock fewer records in the target tables.

  We will have more information on COMMIT_COUNT in the next section.

10.4.10 Apply operations and Subscription set parameters

There are a number of Apply operations parameters and parameters associated with individual Subscription Sets that we should discuss relative to their affect on Apply's performance.

Subscription Set parameters

The Subscription Set parameters are parameters that are in place for a Subscriptoin Set in the ASN.IBMSNAP_SUBS_SET table at the Apply Control Server. You can query this table, looking for the record for the Apply_Qual and Set_Name for a particular set. Or you can view them in the Replication Center. In the Replication Center left window, select **Replication Center-->Replication Definitions-->Apply Control Servers-->*apply control server name*-->Subscription Sets.** Highlight a subscription set name in the right window, right-mouse click, and select **Properties.**

Sleep_Minutes

Sleep_Minutes is the interval, in minutes, between replication iterations. If Sleep_Minutes=0, then Apply's frequency for replicating the set is determined by the value of Apply's operations parameter Delay which will be discussed below. SLEEP_MINUTES is a column of the IBMSNAP_SUBS_SET table. In the Replication Center's *Subscription Set Properties* window, the Sleep_Minutes is

indicated on the *Schedule* tab, in the section on *Frequency of replication* but it is indicated in Minutes + Hours + Days + Weeks rather than just in minutes. If 'Continuously' is checked, then Sleep_Minutes = 0.

Max_Synch_Minutes

- Max_Synch_Minutes, or the blocking factor, tells Apply how to determine which changes to fetch from the source server if there are lots of changes available from a long period of time. If Max_Synch_Minutes is null (no value), than Apply fetches all available changes that have not been previously replicated. If Max_Synch_Minutes has a value, it is in minues, and it tells Apply to check the timestamps associated with the transactions of the changes in the CD table and to only fetch changes within a certain range. If that is not all the changes eligible to be replicated, then Apply, after if applies a set of changes to the target table, will return to the source server and fetch another block of changes. With prior versions of DB2 Replication, the blocking factor could be used for two purposes:
  - to reduce the amount of space required for Apply spill files.
  - to get Apply in effect to perform intermediate connects at the target server.

  With DB2 Replication Version 8, the second objective can be achieved with the Commit_Count parameter on a subscription set.

  Overall, replicating the changes using the blocking factor can decrease Apply's overall throughput in two ways:
  - Apply will have to do more connections to source and target servers to replicate an equivalent number of changes.
  - Apply, while connected to the source server, has an additional calculation to make to determine the range of changes to be fetched.

We recommended that Max_Synch_Minutes (blocking factor) for a Subscription Set be 0 or null (no value) unless the blocking is needed. The blocking factor for a Subscription set can be checked two ways:

- In the Replication Center left window, select **Replication Center-->Replication Definitions-->Apply Control Servers-->*apply control server name*-->Subsriptions.** In the right window, highlight the set name, right-mouse click, and select **Properties.** On the *Set Information* tab, under Set processing properties, check the *Data blocking factor*.
- Query the ASN.IBMSNAP_SUBS_SET table at the Apply Control Server wiht the Apply_Qual and Set_Name values for the set and check MAX_SYNCH_MINUTES.

Commit_Count

If the target tables of a Subscription Set have Referential Integrity between them, or you would just like Apply to perform intermediate commits at the target server

so that fewer rows will be locked by Apply at any point in time, you can specify that Apply open all the spill files together and apply the changes to all the targets in the same order in which the changes were made to all the sources tables of the set. Apply does this if the Commit_Count value for a Subscription Set has a value. If the value is zero (0), Apply applies all the changes from all the spill files in the order they were done on all the soure tables and does one Commit at the end. If Commit_Count > 0, then Apply will perform intermediate commits; if Commit_Count is 2, then Apply will issue a commit after it has applied the changes of every 2 transactions as indicated by the different IBMSNAP_COMMITSEQ values of the records in the spill files. If Commit_Count has no value (i.e. is null) then Apply opens each spill file in succession, applying all the changes from one spill file to its target before opening the next spill and applying its changes.

In any case, if Commit_Count has any value, even if that value is 0, Apply has to do some additional processing with the spill files to determine the order to apply the changes from all the spill files. Also, if Commit_Count>0, then Apply is doing intermediate commits at the target server, which also impedes Apply's overall throughput in applying changes to the target tagbles.

The additional processing by Apply to do the Commit_Count evaluations is probably not significant. Howerver intermediate commits could have a significant effect on Apply's throughput. We would recommend if you are trying to squeeze out every unnecessary bit of processing by Apply in order to decrease the latency of the data in the target tables that you not have any RI between the target tables and that you leave 'Number of transactions applied at the target table before Apply commits' (i.e. Commit_Count) with no value.

You can check the Commit_Count value for a set either in the Replication Center or by querying the Apply Control Tables.

- In the Replication Center, open the *Subscription Set Properties* window for a set as described above for checking Max_Synch_Minutes. On the Set Information tab, under Set processing properties, check *Number of transactions applied to target table before Apply commits*.
- Query the ASN.IBMSNAP_SUBS_SET table at the Apply Control Server with the Apply_Qual and Set_Name values for the set and check COMMIT_COUNT

Apply operations parameters

Here we will discuss some of Apply's operational parameters and their implications for performance.

DELAY

The Delay parameter has no bearing for Subscription Sets which are interval based and have Sleep_Minutes (replication interval) of 1-minute or greater, and for Subscription Sets that are ‘event’ based.

The Delay parameter is relevant for Subscription Sets that are ‘Continuous’; that is, their Sleep_Minutes=0 in the record for the set in in the ASN.IBMSNAP_SUBS_SET table at the Apply Control Server.

Delay determines the number of seconds that an Apply is to use as the interval for the Set that it is replicating. The default value is 6 seconds. The range is 0 seconds to 6 seconds. the value can be set when Apply is started, or it can be set by inserting/updating a record into ASN.IBMSNAP_APPPARMS table at the Apply Control Server for the Apply_Qual of the set.

Delay=0 will make Apply, any time it isn’t actually replicating any changes, connect to the source server to check the new-change signals in the IBMSNAP_REGISTER table. This is probably OK if the source server has lots of activity so there are almost always new changes for Apply to replicate. . But if there are not any changes, Apply could flood the source server with connection requests.

Therefore, we recommand Delay=1 when trying to achieve low-latency replication.

OPT4ONE

Apply normally does checks of its Control Tables after it finishes a replication and also, if it goes inactive even for a sub-second, when it returns from inactivity. Apply is checking to see if there are changes to its control information and whether any Subscription Sets have become eligible, based on time interval event, for immediate processing. This select from Apply Control Tables does not take very long, but if you’re interested in squeezing all possible unnecessary steps out Apply’s cycle, you can have an Apply process just read the Control Tables for the information about the only set it processes when Apply starts, and not take time to re-read this information again. To do this, when Apply starts, include the parameter OPT4ONE=YES, or you can specify OPT4ONE=’Y’ in the record in ASN.IBMSNAP_APPARMS for this Apply qualifier. For instance:

```
UPDATE ASN.IBMSNAP_APPPARMS SET OPT4ONE='YES' WHERE APPLY_QUAL='MIXQUAL'
```

Or, if there is not yet a record in APPARMS for this Apply’s Qualifier, you can insert a record with:

```
INSERT INTO ASN.IBMSNAP_APPPARMS (APPLY_QUAL,OPT4ONE) VALUES ('MIXQUAL','Y')
```

Seeing Apply's operations parameters

Unlike with Capture, there is not a way to query the parameters of a running Apply. However, if you set the parameters for an Apply Qualifier via the ASN.IBMSNAP_APPPARMS table at the Apply Control Server, and don't over-ride these settings when you start Apply, then you can see the operational parameters for an Apply process by querying the APPPARMS table.

We showed above how to insert and update a record in the APPPARMS table. You could also query this table with the APPLY_QUAL value for an Apply process to see the operations parameters for that Apply process.

10.4.11 End-to-end latency of replication

The latency of the data for the target tables of a set can be determined from the ASN.IBMSNAP_SUBS_SET SYNCHTIME for the set. The SYNCHTIME value originated as the timestamp in a DB2 Log record at the source server. This value was first placed in the capschema.IBMSNAP_REGISTER GLOBAL_RECORD where it reflected the latency of Capture at the time that Apply last successfully replicated from the source server. But before substracting SYNCHTIME for a set from a current timestamp at the Apply Control Server, you should compare current timestamps from Capture and Apply Control Servers to determine the time difference between the two system and add or subtract this amount as appropriate in your latency calculation.

A history of the latency of the data at the targets can be determined at the Apply Control Server with:

```
SELECT ENDTIME, (ENDTIME - LASTRUN) + SOURCE_CONN - SYNCHTIME) AS APPLY_LATENCY
FROM ASN.IBMSNAP_APPLYTRAIL WHERE SET_NAME = 'set_name' AND
APPLY_QUAL='apply_qualifier'
```

- ENDTIME is the Apply Control Server timestamp when it finished a subscription cycle.
- LASTRUN is the Apply Control Server timestamp when it started a subscription cycle
  - ENDTIME - LASTRUN is the time to execute a subscription cycle
- SOURCE_CONN is a Capture Control Server timestamp when Apply connects to the Capture Control Server at the start of a Subscription cycle.
- SYNCHTIME is the SYNCHTIME from capschema.IBMSNAP_REGISTER GLOBAL_RECORD=Y'
  - SOURCE_CONN - SYNCHTIME indicates Capture's latency

So the whole calculation can be summarized as

- (time it took Apply to complete the replication) + (Capture's latency at the time the replicaiton cycle was executed)

The same information is available through the Replication Center. In the left-window tree, select **Replication Center --> Operations --> Apply Control Servers-->*Apply_Control_Server_Name.*** In the right window, highlight the Apply Qualifier for which you want the latency, right-mouse click, and select **End-to-End-Latency** from the options. The *End-to-End Latency* window opens. Select the *From*-Time and *To*-Time and/or select a *Time Interval* and press **Refresh**. A report of Average, Minimum, and Maximum Latency will appear for the Subscriptions Set included with that Apply Qualifier. See Figure 10-7 for an example of the *End-to-End Latency* window.

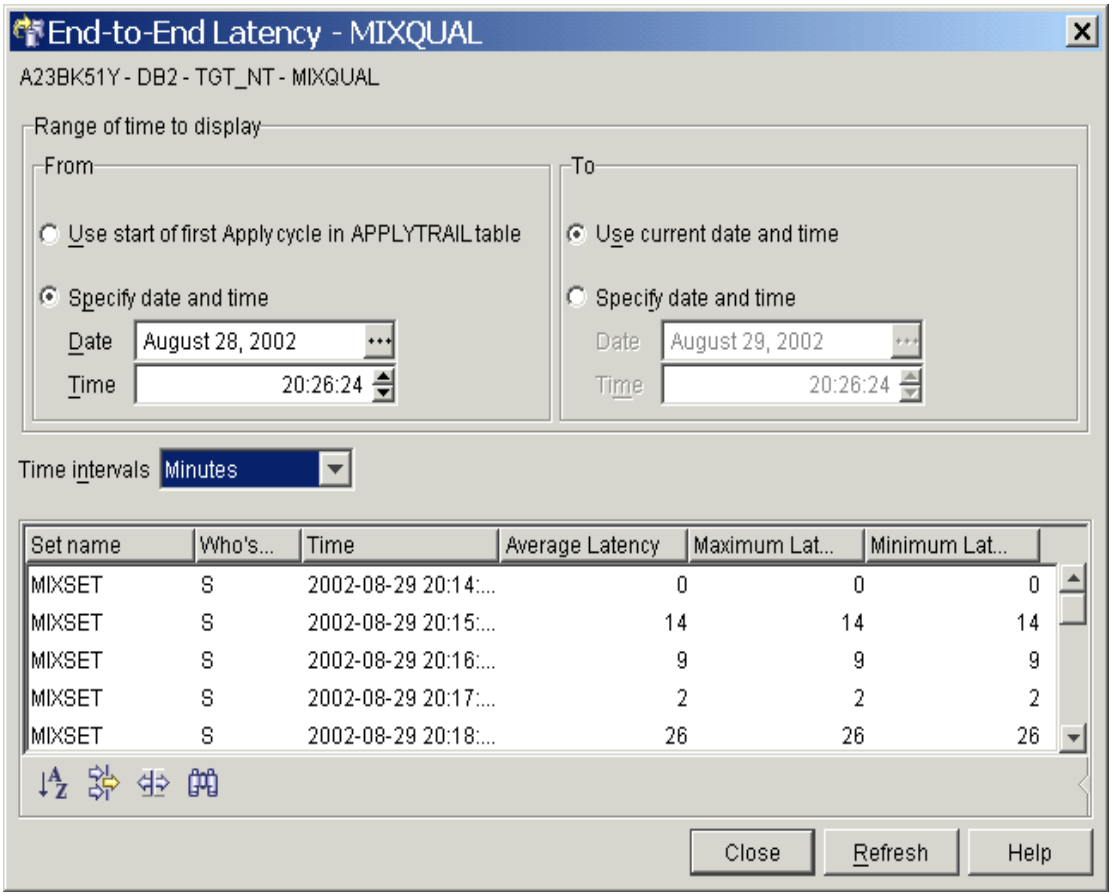


Figure 10-7 Replication Center - Apply End-to-End Latency

10.4.12 Apply Throughput

The number of records that Apply insert/update/deletes to the target tables of a Subscription set is recorded in the records Apply inserts into the ASN.IBMSNAP_APPLYTRAIL at the end of each replication cycle.

The relevant columns in the IBMSNAP_APPLYTRAIL table are:

- APPLY_QUAL - the Apply Qualifier of the Apply process that created this record
- SET_NAME - the Subscription set name associated with this cycle
- WHOS_ON_FIRST - indicates if target was an Update-Anywhere Master; in which case, the value is 'F'. In all other cases, the value is 'S'. In short, if the target is a User Copy, CCD, Point-in-Time, or Replica, the value will be 'S'
- LASTRUN - timestamp at start of a replication cycle
- END_TIME - timestamp at the end of the replication cycle.
  - END_TIME - LASTSUCCESS gives the elapsed time of the replication cycle represented by this record in APPLY_TRAIL.
- ASNLOAD - was ASNLOAD used.
- FULL_REFRESH - was full refresh done.
- EFFECTIVE_MEMBERS - number of members in the set for which there wre changes.
- SET_INSERTED - number of rows inserted into all target tables of the set.
- SET_DELETED - number of rows deleted from all target tables in the set
- SET_UPDATED - number of rows updated in all target tables in the set.
- SET_REWORKED - number of inserts and updates that were reworked for all target tables of the set.

The same information can be obtained through the Replication Center. In the left window, expand **Replication Center-->Operations-->Apply Control Servers-->*control server name*-->Apply Qualifiers** and highlight the Apply Qualifier name in the right window. Rightmouse click and select **Show Apply Throughput Analysis** from the options. The *Apply Throughput Analysis* window will open.

In the *Information to display* field, select either **Rows fetched** or **Elapsed Time**.

If you select **Rows Fetched** you will be able to see, for each set procesed by this Apply Qualifier, the *Rows Inserted*, *Rows_Updated*', *Rows_Deleted*, and *Rows Reworked.*

If in the *Time Intervals* field you select **No Time Interval**, the display will be one record for each replication cycle that was executed, essentially showing the values from each record in the APPLYTRAIL table.

If in the *Rate* field you select **No Time Interval**, the data displayed will be of the total records inserted, updated, deleted, reworked per interval. If in the *Rate* field

you select **Rows/Second**, the display will be the average rows/second inserted, updated, deleted, reworked.

If in the *Time Intervals* field you select **Seconds, Minutes, Hours, Days**, etc, then the display will be aggregates of the APPLYTRAIL statistics with one record for each interval selected (second, minute, etc).

10.4.13 Time spent on each of Apply's sub-operations

It is possible, using the asntrc facility to calculate the amount of time that Apply spends on each of its sub-operations. We will be looking at the TRACE PERFORMANCE RECORD's in the output of Apply's formatted trace. This technique could be useful when trying to find what to address when there is a performance issue. It will tell us such things as whether Apply is spending more time fetching changes from the source server than applying the changes to the target tables. If there are multiple members in a set, this technique can determine if more time is spenton one of the members than the others considering the number of records fetched and applied.

We will be using asntrc. asntrc is described in the *DB2 Unversal Database Version 8 Replication Guide and Reference SC27-1121* in the chapter on 'System Commands for Replication.' asntrc also has online help. To display asntrc's help, in a command prompt, enter:

```
asntrc -help
```

We will show an example of using asntrc to get the TRACE PERFORMANCE RECORDs, describe the fields in the Trace Performance Records, and then show how we calculate the time spent by Apply fetching changes for each member of a set, and the time spent by Apply to apply the records from each spill file to the corresponding target table.

In our example, Capture is running and we are making updates to two source tables: MICKS.SRCTAB and MICKS.SRCTAB2. Apply could be running when we turn asntrc on, but in our case we begin with Apply stopped. We will turn asntrc on, and then start Apply with COPYONCE=Y so that Apply will do exactly one replication cycle and stop. The trace won't be too large, and we should find one set of PERFORMANCE TRACE RECORDS to analyze when we format the trace.

In our example, we will also be starting asntrc specifying that it write all trace records to file; this is not the default behavior and may actually be slowing Apply down a little. We could also not specify an output file when we start asntrc so trace records would only be held in memory. Then we would need explicitly to enter an asntrc command while the asntrc is still on to write the asntrc memory buffers to file bfore turning asntrc off.

Let's proceed.

1. We start asntrc with this command:

   ```
   asntrc on -fn MIXQUAL.dmp -db TGT_NT -app -qualifier MIXQUAL
   ```

   In this example:

   on: asntrc is turned on with this command.

   -fn MIXQUAL.dmp: asntrc will write all records to the MIXQUAL.dmp file as it is tracing.

   -db TGT_NT: we specify the Apply Control Server database

   -app: we tell asntrc to trace Apply (not Capture)

   -qualifier MIXQUAL: we tell asntrc the Apply Qualifier of the Apply process to be traced

2. We start Apply, specifiying that it perform just one cycle:

   ```
   asnapply control_server=TGT_NT apply_qual=MIXQUAL
   apply_path=d:\DB2Repl\MIXQUAL pwdfile=MIXQUAL.aut copyonce=Y
   ```

   In this example:

   control_server=TGT_NT: our Apply Control Server database

   apply_qual=MIXQUAL: our Apply Qualifier

   apply_path=d:\DB2Repl\MIXQUAL: the directory containing the password file and also where we want Apply to create the spill files.

   pwdfile=MIXQUAL.aut: the file containing userid/passwords that Apply will use to connect to Apply Control Server, Capture Control (source) Server, and Target Server. The file was created and records added to it using the **asnpwd** command.

   copyonce=Y: we want Apply to replicate one subscription set cycle and stop

3. We wait while Apply starts, goes through one cycle and stops. First we see the message that Apply is started. A little while later, we see a message that Apply is stopped.

4. We stop the asntrc.

   ```
   asntrc off -db TGT_NT -app
   ```

   In this example:

   off: asntrc is turned off

   -db TGT_NT: the database specified for the trace instance we're stopping

   -app: we're stopping a trace that was for Apply

5. We format the asntrc 'dmp' output so we can look for the Peformance Trace Records:

```
asntrc v7fmt -fn MXQUAL.dmp > MIXQUAL.v7fmt
```

In this example:

v7fmt: provide a formatted trace similar to the Apply V7 trace. The alternative would have been ‘fmt’ to provide a trace in the newer DB2 Replication trace format that looks somewhat like a DB2 for Linux, UNIX, Windows formatted trace.

-fn MIXQUAL.dmp: the input file to be formatted

> MIXQUAL.v7fmt: the output file from the trace-format operation

6. We use an editor to look at the formatted trace to find the Performance Trace Records. In this example, we are on Windows and use Notepad. The formatted traces are large, even for just one Apply replication cycle. The editor’s (i.e. Notepad) ‘find’ capability can take us to the Performance Trace Records. The Performance Trace Records look like in Example 10-1

Example 10-1 Performance Trace Records in formatted asntrc for Apply

```
===== PERFORMANCE TRACE RECORD =====
S, MIXSET, 1, S, 2002/09/04 15:02:05, 2002/09/04 15:02:05, 2002/09/04 15:02:05,
2002/09/04 15:02:05, 2002/09/04 15:02:11, 2002/09/04 15:02:11, , , 2002/09/04
15:02:36, 2002/09/04 15:02:36

M, MIXSET, 1, 0, 2002/09/04 15:02:05, 2002/09/04 15:02:05, 2002/09/04 15:02:05,
2002/09/04 15:02:08, 2002/09/04 15:02:11, 2002/09/04 15:02:15, 200, 5, 2, 0

M, MIXSET, 1, 1, 2002/09/04 15:02:08, 2002/09/04 15:02:08, 2002/09/04 15:02:08,
2002/09/04 15:02:11, 2002/09/04 15:02:15, 2002/09/04 15:02:20, 50, 150, 2, 0
```

We see there are two types of Performance Trace Records:

- S - performance trace record for the entire Subscription Set
- M - performance trace record for a specific Member in a Subscription Set.

The fields at the beginning of the S and M records indicate several things.

1st field=S or M

2nd field=MIXSET: the Subscription Setname

3rd field=1: the numerical number of this Performance Trace Record. This number becomes relevant if there is more than one group of Performance Trace Records in the trace output. In this case, we are looking at either the first or the only set of Performance Trace Records in the trace output.

4th field of S record=S: WHOS_ON_FIRST value. ‘S’ means the target is a User Copy, CCD, Point-in-Time, or a Replica. If ‘M’, the target is the Master in a Update-Anywhere replication.

4th field of M record=0 or 1. Indicates the Member Number within the set. To reconcile the Member Number to the Target Table name, go back to the top of formatted trace and find ‘mem_info’ entries. See Example 10-2:

Example 10-2 Member info in formatted anstrc output for Apply

```
------------------------
mem_info i = 0
------------------------
SOURCE_OWNER    = MICKS
SOURCE_TABLE    = SRCTAB
SOURCE_VIEW_QUAL = 0
TARGET_OWNER    = MICKS
TARGET_TABLE    = TGSRCTAB
....
------------------------
mem_info i = 1
------------------------
SOURCE_OWNER    = MICKS
SOURCE_TABLE    = SRCTAB2
SOURCE_VIEW_QUAL = 0
TARGET_OWNER    = MICKS
TARGET_TABLE    = TGSRCTAB2
```

So we see:

Member 0 = source table MICKS.SRCTAB - target table MICKS.TGSRCTAB

Member 1 =source table MICKS.SRCTAB2 - target tale MICKS.TGSRCTAB2

The meanings of the timestamp fields in the S records are given in Table 10-1

Table 10-1 Description of Apply ‘S’ Performance Trace Records:

Timestamp Field Number	Meaning
1st timestamp	Before Connect to Apply Control Server to read set information from Apply Control Tables
2nd timestamp	After Connect to Control Server
3rd timestamp	Before Connect to Capture Control (Souce) Server Between the 2nd and 3rd timestamp, Apply reads the information about the Subscription Set from the Apply Control Tables

Timestamp Field Number	Meaning
4th timestamp	After Connect to Capture Control (Soure) Server
5th timestamp	Before Connect to Target Server Between the 4th and 5th timestamps, Apply: - reads the capschema.IBMSNAP_REGISTER table to learn if there are new changes and to find out names of CD tables - fetches changes from CD tables for all members in the set for which there are new changes. See the M records' 1st, 2nd, 3rd, and 4th timestamps to find out how long the source server took to prepare and execute the SELECT FROM the CD tables. Note: this timestamp should follow all the 'M' record's 4th timestamps, which marked end of fetches from the CD tables at the source server.
6th timestamp	After Connect to Target Server
7th timestamp	Before Opening Spill Files Value provided only if the Set's Commit_Count is not null. That is, this is the timestamp when Apply opened all the spill files. If Commit_Count=null, spill files were opened in succession. Look at M-records' 5th timestamp
8th timestamp	After Closing Spill Files Value provided only if Set's Commit_Count is not null Between the 7th and 8th timestamp, Apply applies the changes from all the spill files to all the target tables
9th timestamp	Before Connect to Source Server to update Prune_Set
10th timestamp	After Connect to Source Server to update Prune_Set Between the 9th and 10th timestamps, Apply updates the synchpoint and synchtime for the Subscription Set in the capschema.IBMSNAP_PRUNE_SET table.

The meaning of the timestamp and number fields in the 'M' records are shown in Table 10-2

Table 10-2 Description of Apply Trace 'M' Performance Trace Records

Timestamp/Numeric Field Number	Meaning
1st timestamp	At Source Server, before preparing Select from CD table Note: this timestamp should follow 'S' record's 4th timestamp
2nd timestamp	At Source Server, after preparing Select from CD table Between the 1st and 2nd timestamps, the source server optimizes and compiles Apply's dynamic SQL SELECT statement for the CD table
3rd timestamp	At Source Server, open cursor before fetch from CD table
4th timestamp	At Source Server: after last fetch from CD table Between the 3rd and 4th timestamp, the source server provides the results of the SELECT statement for the CD table.
5th timestamp	At Apply server: open spill file Value provided only if Set's Commit_Count is null. If Set's Commit_Count is not null, all spillfiles are opened together. See S record 7th timestamp.
6th timestamp	At Apply server: close spill file Between the 5th and 6th timestamp, Apply updates the target table with the changes from the spill file. Value provided only if Set's Commit_Count is null. If Set's Commit_Count is not null, all spillfiles are closed together. See S record 7th timestamp.
1st numeric field	Number of rows inserted to target table
2nd numeric field	Number of rows updated in target table
3rd numeric field	Number of rows deleted from target table
4th numeric field	Number of rows reworked to insert or update of target table

Before proceeding, we'll point out that in our example that our Subscription Set has Commit_Count = null. Apply is opening each spill file in succession, not together. Apply opened the spill file for Member 0, applied all the changes to TGSRCTAB, and closed that spill file. Then Apply opened the spill file for Member 1, applied all the changes to TGSRCTAB2, and then closed that spill file.

So the S record in our example has no values for the 7th and 8th timestamps and the M records have values for their 5th and 6th timestamps, indicating when Apply opened and closed each spill file. If the Set's Commit_Count had a value, even if it was zero, the S record would have 7th and 8th timestamps indicating

when the two spill files were opened and closed togehter, and the M records would not have 5th and 6th timestamps.

Here we'll repeat in Example 10-3 the Performance Trace Records we're using in the calculations below of Apply's time to fetch the changes at the source server and the time to apply the changes to the target tables.

Example 10-3 Peformance Trace Records used in our calculations

```
===== PERFORMANCE TRACE RECORD =====
S, MIXSET, 1, S, 2002/09/04 15:02:05, 2002/09/04 15:02:05, 2002/09/04 15:02:05,
2002/09/04 15:02:05, 2002/09/04 15:02:11, 2002/09/04 15:02:11, , , 2002/09/04
15:02:36, 2002/09/04 15:02:36

M, MIXSET, 1, 0, 2002/09/04 15:02:05, 2002/09/04 15:02:05, 2002/09/04 15:02:05,
2002/09/04 15:02:08, 2002/09/04 15:02:11, 2002/09/04 15:02:15, 200, 5, 2, 0

M, MIXSET, 1, 1, 2002/09/04 15:02:08, 2002/09/04 15:02:08, 2002/09/04 15:02:08,
2002/09/04 15:02:11, 2002/09/04 15:02:15, 2002/09/04 15:02:20, 50, 150, 2, 0
```

Now here are our calculations:

1. Total Elapsed Time for the Subscription Cycle:

   ```
   S 10th timstamp - S 1st timestamp = 15:02:36 - 15:02:05 = 31 seconds
   ```

2. Time for Apply to fetch changes for Member 0 (CDSRCTAB):

   ```
   M0 4th timestamp - M0 3rd timestamp = 15:02:08 - 15:02:05 = 3 seconds
   ```

3. Time for Apply to fetch changes for Member 1 (CDSRCTAB2):

   ```
   M1 4th timestamp - M1 3rd timestamp = 15:02:11 - 15:02:08 = 3 seconds
   ```

4. Time for Apply to insert/update/delete Member 0 (TGSRCTAB)

   ```
   M0 6th timestamp - M0 5th timestamp = 15:02:22 - 15:02:11 = 11 seconds
   ```

5. Time for Apply to insert/update/delete Member 1 (TGSRCTAB2):

   ```
   M1 6th timestamp - M1 5th timestamp = 15:02:35 - 15:02:22 = 13 seconds
   ```

Note: We learned the name of the CD tables by querying the IBMSNAP_REGISTER table at the Capture Control Server. Or in the Replication Center, under **Replication Center-->Replication Definitions-->Capture Control Servers-->*capture control server name*-->Registered Tables,** we higlighted the source table, right mouse click, and select **Properties.** In the *Registered Tables Properties* window, select the *CD Table* tab and look at the *CD Table schema* and *CD Table name* fields.

In our example, the number of records insert/update/deleted/re-worked to each of the target tables is in the numeric fields at the end of the 'M' records in Example 10-1:

- TGSRCTAB had
  - 200 records inserted
  - 5 records updated
  - 2 records deleted
  - no inserts or updates reworked
- TGSRCTAB2 had:
  - 50 records inserted
  - 150 records updated
  - 2 records deleted
  - 0 records reworked.

We'll note in this case that the fetches from the source server took less time (3 seconds and 3 seconds respectively for the two members) than the insert/update/deletes to the targets (11 seconds and 13 seconds respectively). This is as expected. If the fetches from the CD tables had taken longer than the updates of the targets, we would suspect that there might be something done that could done at the source server to improve the total throughput on this subscription set.

10.5 Configuring for low-latency replication

Here we will discuss the settings used to lower the latency - that is, the time differential - of the data in target tables when using DB2 Replication. A big assumption in the following discussion is that both Capture and Apply can keep up with the source application. That is, that Capture can insert records in the CD tables at the same rate as the source application, and so can Apply at the target tables. If Capture can't keep up with source application, then of course then the changes available in the CD tables for Apply to fetch will have a greater and greater time differential from the data in the source tables. And if Apply can't keep up, then changes will be sitting in the staging tables for greater lengths of time before Apply fetches them.

To make Apply check at the source server for new changes more frequently:

- when creating the Subscription Set, on the *Create Subscription Schedule* tab, under *Frequency of Replication*, specify **Time-based - Continuous**.
  - The setting in ASN.IBMSNAP_SUBS_SET will be REFRESH_TYPE='R' and 'SLEEP_MINUTES='0'.
- when starting Apply, or in the record for this Apply Qualifier in the ASN.IBMSNAP_APPPARMS table, give attention to the DELAY parameter, which specifies the frequency in seconds that Apply will connect to the source server to check for new changes

Note: Delay=0 is not recommended as it could cause Apply to flood the Source Server with connection requests.

We recommend Delay=1.

To make Capture set signals more frequently that new changes are available:

- when starting Capture, or in the record in capschema.IBMSNAP_CAPPARMS at the Capture Control Server, give attention to the COMMIT_INTERVAL value.
  - At the Commit_Interval, Capture stops inserting changes into the CD tables and updates the new-change signals in the capschema.IBMSNAP_REGISTER table. Specifically:
    - In the IBMSNAP_REGISTER record where GLOBAL_RECORD='Y', Capture sets SYNCHPOINT to the Log Sequence Number (LSN) that it has last read.

      This value is the first thing that Apply checks when it connects to the source server. If different from the SYNCHPOINT in the ASN.IBMSNAP_SUBS_SET record for this set at the Apply Control Server, it tells Apply that there has been some kind of update activity at the source server since DB2 is adding records in the Log.

- In the IBMSNAP_REGISTER record for each source table for which Capture has inserted records into CD tables since the last Commit_Interval processing, Capture sets CD_NEW_SYNCHPOINT to the highest IBMSNAP_COMMITSEQ value of records recently inserted into the CD table.

  If Apply detects that IBMSNAP_REGISTER GLOBAL_RECORD='Y' SYNCHPOINT has 'advanced', Apply then checks the CD_NEW_SYNCHPOINTs for all the tables in the set. If any of them are greater than the SYNCHPOINT for the set back at the Apply Control Server, then this indicates to Apply that there are new changes for that source table. Apply then includes that CD table on the list to fetch from.

– The Commit_Interval default setting is 30 seconds.
  - A different value can be set through the Replication Center: **Replication Center-->Operations-->Capture Control Servers**. Highlight the Capture Control Server by name in the right window, right-mouse click and select '**Manage Values in CAPPARMS**.'

The above probably seems confusing when first read. It might be good to give an example of what happens with Apply Delay=1 and Capture Commit_Interval=15:

1. every 1 second, Apply connects to the source server and reads the SYNCHPOINT in the IBMSNAP_REGISTER record where GLOBAL_RECORD='Y'. Apply does this 14 times (i.e. once a second for 14 seconds) without finding a new value and so Apply goes back to sleep for part of a second and then wakes up and connects again to the source server to checkthe SYNCHPOINT in IBMSNAP_REGISTER's GLOBAL_RECORD
2. the 15th time, Apply checks, it finds a new value for SYNCHPOINT in IBMSNAP_REGISTER GLOBAL_RECORD because 15 seconds is the interval at which Capture updates this value if there's been any update activity at all in DB2.
3. Apply then checks CD_NEW_SYNCHPOINT in each of the IBMSNAP_REGISTER records for source tables in the set. For those with new values, Apply puts the CD table on the list to fetch from.
4. Apply selects from the CD tables with new changes, puts the changes into spill files, connects to the target server and insert/update/deletes the target tables with the new changes.
5. Apply goes back to 1) above

The temptation, to decrease latency, is to set Capture's Commit_Interval=1 second, but this can be counter-productive since Capture has to stop inserting into the CD tables in order to do the Commit_Interval processing. If Capture overall rate of inserting into the CD tables is less than the rate that the source

applications are updating the source tables, the data available in the CD tables will be farther and farther behind and so will the data in the target tables.

We have recommended trying Capture Commit_Interval of 10, or 15 seconds and seeing if Capture is keeping up. The lowest end-to-end latencies we have seen have been 15-20 seconds. What is achievable in your environment will depend on whether Capture can keep up with the source applications while setting the new-change signals for Apply at short commit_intervals.

Low latency when replicating from non-DB2 servers

When replicating from non-DB2 data sources, such as Informix, lower latencies may be achievable. This is because it is Apply itself that causes the new-change signal to be set. When Apply connects to the federated databse with the nicknames for the source tables, before it reads the IBMSNAP_REGISTER nicknem for the new-change signal, executes an SQL statement that updates the IBMSNAP_REG_SYNCH nickname. Federated server will take the update to this nickname and send an update for the ibmsnap_reg_synch table to Informix. The ibmsnap_reg_synch table in Informix has a trigger, created by the Replication Center, that calls a procedure that, among other things, checks the CCD tables and updates the synchpoint values in the ibmsnap_register table in Informix. Then Apply reads the IBMSNAP_REGISTER nickname for the ibmsnap_register table in Informix, checking for the new-change signal. If Apply is replicating with Delay=1, then the new-change signal is being set every second, unless the number of changes fetched takes over a second to fetch and apply at the target tables.

10.6 Develoment Benchmarks

DB2 Replication development performed some benchmarks with Version 8 while the product was still in beta status. We will provide some of the results here, with caveat that what is achievable on your systems, with your data and application workloads, could be different. Also, the technicians who performned the benchmarks provided some tuning recommendations; they have been covered throughout this chapter.

10.6.1 Benchmark system configuration

The systems involved in the benchmarks were all IBM pSeries RISC System/6000's. Operating system level was AIX 4.3.3. DB2 Unversal Database for AIX was used for all the replication servers in these benchmarks. Source/Capture Control Server and Target/Apply Control Server were on different machines. The Pull design was used; that is Apply on the target server was used to replicate the Subscription Sets. Largest system used was a 12-way S80 with 16GB memory and 128 SSA disk drives.

The application workload updating the source tables was the OLTP-type IRWW workload. This workload was run simulating 200 to 600 users which generates 174 to 466 transactions/second respectively. The tables and their row-lengths are summarized in Table 10-3

Table 10-3 IRWW Tables' Row Lengths

Table Name	Row Length
CUSTOMER	600 bytes
DISTRICT	100 bytes
HISTORY	60 bytes
ITEM	90 bytes
NEWORDERS	14 bytes
ORDERLINE	66 bytes
ORDERS	33 bytes
WHAREHOUSE	95 bytes
STOCK	310 bytes

provides some characteristics of the IRWW workload itself, with Capture running on the same system:

Table 10-4 IRWW workload characteristics

Number of users	Transaction/Second	Rows changed/second
200	174	1700
300	276	2500
400	374	3200
500	424	3500
600	466	4000

Regarding the configuration of the source database that contain the IRWW tables and the CD tables for them, one tablespace was create to contain all the CD tables plus the IBMSNAP_UOW table and another tablespace to contain all the indexes for the CD tables and the UOW table. Both were specified with following:

- pagesize 4096
- managed by database
- using (raw) device:
  - the CD_UOW table tablespace had 16 disk paths specified
  - the CD_UOW index tablespace had 4 disk paths specified
- extentsize 16
- prefetchsize 64
- bufferpool specified
- overhead 24.10000
- transferrate 0.900000

Also, RUNSTATS was run for the CD tables and the statistics were reasonably accurate throughout the benchmark.

The Database Configuration parameters updated at the source/Capture Control Server were as indicated in Table 10-5.

Table 10-5 Benchmark soure server Database Configuration parameters

Database Configuration parameter	Setting during benchmark
LOCKLIST	2048 pages
DFT_DEGREE	1
MAXLOCKS	80
AVG_APPLS	1
STMTHEAP	4096 pages

Database Configuration parameter	Setting during benchmark
DFT_QUERYOPT	5
NUM_IOCLEANERS	4
NUM_IOSERVERS	16

The benchmark team has since the benchmark recommended the DB2 Replicaiton users in a high volume enviornment consider Database Configuratoin parameters as follows:

- BUFFPAGE = 7500 pages
- DBHEAP = 5000 pages
- LOGBUFSZ = 256 or 512 pages
- PCKCACHESZ = 5000 pages
- LOCKLIST = 10000 pages

They also recommend setting the following DB2 Profile Registry (db2set) variables:

- DB2_RR_TO_RS = YES
- DB2_MMAP_READ = OFF (for AIX)
- DB2_MMAP_WRITE = OFF (for AIX)

10.6.2 Benchmark results

When the IRWW workload was running with 500 users doing 424 transactions/second which changed 3500 rows per second, with Capture running at the same time, Capture could keep up. That is, Capture was able to insert rows into the CD tables at the same rate - 3500 rows/second - that IRWW was inserting, updating, and deleting rows of the IRWW source tables. When the IRWW workload was stepped up to 600 users doing 466 transactions/second changing 4000 rows/second, Capture lagged behind, only able to insert 3000 rows/second into the CD tables.

The IRWW workload was then run without Capture running, and then stopped, so that there was then absolutely no activity on the source server. A single Capture was then started to read through the log records just created by IRWW. The highest throughput rate seen for Capture, running by itself with no other activity on the system, was 5730 rows/second. With multiple Captures, higher aggregate Capture throughput was observed, but the increase was not linear with the increase in number of Captures running. Two Captures' highest throughput was 8425 rows/second, and three Captures' highest throughput was 9183 rows/second.

For a single Apply, running on the system with the target tables, the maximum Apply throughput seen was 2703 rows/second. For two Apply's the maximum throughput was 3804 rows/second.

Runs were done with Capture pruning regularly while Capturing changes and while Apply was fetching and applying changes to targets. The elapsed time for Capture to capture all the changes and for Apply to get them into the targets was slower if Capture was allowed to prune, but only by 1 percent or less total elapsed time.

End-to-end latency observed was excellent with up to 2396 rows changed per second. Table 10-6 provides a summary of the latencies seen for different change rates.

Table 10-6 Benchmark target table latency

Throughput (rows/second)	Min (seconds)	Max (seconds)	Avg (seconds)
76	1	6	2
223	1	6	2
398	1	7	2
1323	1	8	3
2396	1	17	4
2932	1	44	12
3448	1	71	25

Some of the replication settings specified during these latency tests were:

- Capture Commit_Interval = 1 second
  - The benchmark team reports that Capture Version 8's throughput does not seem to be as affected by frequent stops to update the new-change signals in the IBMSNAP_REGISTER table and commit new records in the CD and UOW table. But we are still cautious in recommending that Commit_Interval be set this low.
- Apply Sleep_Minutes = 0
  - This is the setting for Continuous Replication. How frequently Apply connects to the source server looking for new changes will be determined by the Apply start parameter Delay.
- Delay=0
  - This setting will cause Apply to never be inactive. If it is not replicating changes it is connecting to the source server to check for new changes. In this high volume workload there were almost certainly always be new

changes to fetch. In a lower volume environment, Apply oculd flood the source server with connection requests. We would still recommend Delay=1 as the minimum for this parameter.

Part 1

Appendixes

A

DB2 Admin Client and DB2 Connect PE install

Following is an example of the steps to install DB2 Connect Personal Edition on a Windows 2000 PC.

We put the product CD into the workstation's CD-ROM drive, and select

Start > Run

In the Run dialogue window, we selected **Browse.**

In the *Browse* dialogue window that appeared, we selected the icon for the CD-Drive, and 'setup' appeared among the available files/folders. We selected **setup** and **Open** to returned to the *Run* dialogued window. In the *Open* field, we now had E:\setup.exe. We selected **OK** to start the install of DB2.

If we had downloaded the DB2 Administration Client from IBM and placed it in a directory and unzipped it, we would find the file *setup.exe* among the files/directories in which you unzipped the download file. You would select this *setup* file in the **Start > Run** dialogue.

Once we selected **OK** in the Run dialogue window, we got a message box: *Loading the DB2 Setup Launchpad*.

Then we see the *DB2 Setup Launchpad*, as in Figure A-1

Figure A-1 DB2 Connect PE - Setup Launchpad

We selected **Install Products** from the *Setup Launchpad* window.

We accepted the **License Agreement** in the next window and hit **Next.**

The next dialog window that appears is *Select the Installation Type*. It asks us if we want our installation to be

- Typical
  - with or without Additional Functions - Data Warehousing
- Compact, or
- Custom

We selected **Custom** so that we could see the individual components available for installation, even though **Typical without Data Warehousing** would be fine.

Next was a dialogue window with heading *Select the Installation Action***.** We selected **Install on this computer** and **Next.**

This brought us to the window with title *Select the features you want to install*.

This window includes a selection list as in Figure A-2

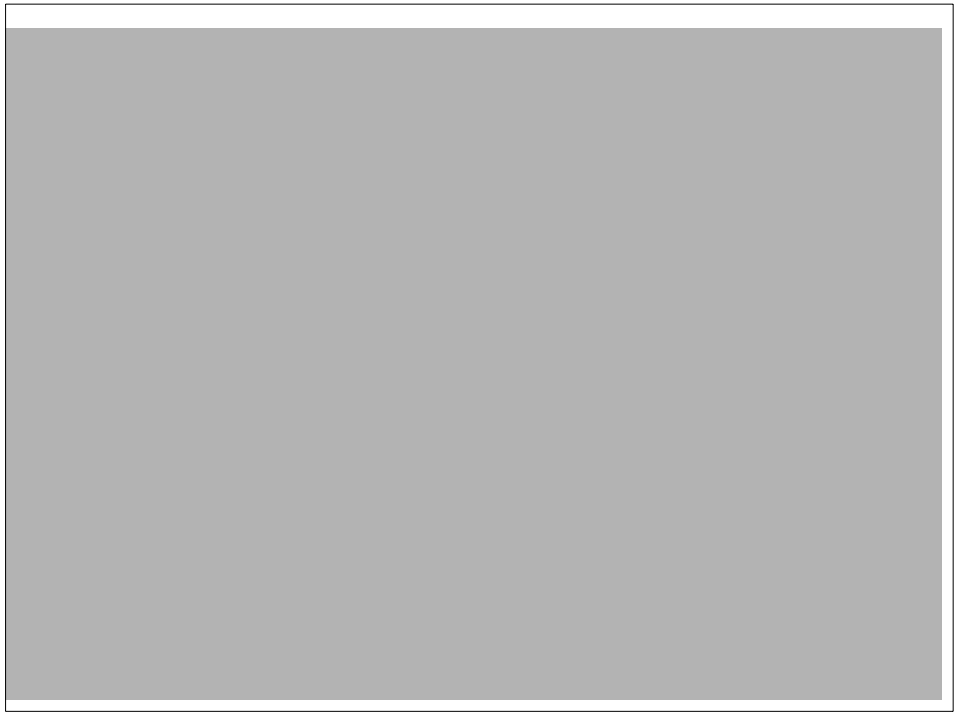


Figure A-2 DB2 Connect PE - features selection window

All the features in the list are in the ‘selected’ status. Clicking the ‘+’ by each icon, we can see the sub-features that are available.

We can ‘de-select’ a feature from installation by clicking the icon next to it and then selecting the line with red-X from among the options presented.

The sub-features under Client Support are:

- Interfaces
- Base Client Support
- System Bind Files
- Java Runtime Environment
- LDAP Exploitation
- XML Extender
- Communications Protocol

We definitely want Interfaces, Base Client Support, System Bind Files, Java Runtime Environment, and, under Communications Protocols, TCP/IP.

Under **Base Support --> Interfaces**, we find:

- JDBC Support
- MDAC 2.7
- ODBC Support
- OLE DB Support
- SQLJ Support

JDBC is needed by DB2 Replication Center, Control Center, Command Center, and Configuration Assistant. The other interfaces may be needed with other software we have on our workstation.

We should probably accept the Application Development Tools though some, if not all, the features in the list will certainly not be needed to administer DB2 Replication. Application Development Tools may not be included if we were installing Administration Client instead of DB2 Connect Personal Edition.

Under Administrative Tools we find:

- Control Center
- Client Tools
- Command Center
- Configuration Assistant
- Event Analyzer

`Replication Center is included in` **`Control Center.`**

All the Administrative Tools may be useful and we should accept them.

Server Support probably would not be included if we were installing DB2 Adminsitration Client instead of DB2 Connect Personal Edition. But in a DB2 Connect Personal Edition installation we need this feature. This feature provides the connectivity to DB2 for z/OS and OS/390 and to iSeries.

We do not need the Business Intelligence features to administer DB2 Replication.

Before we click **Next** on the *Select Features to Install* window, we note that the default directory and path for the install are:

- Drive: C:\
- Directory: *C:\Program Files\IBM\SQLLIB*

The default drive and directory are fine with us.

We go ahead and press **Next.**

We next get a window with the title *Select the Languages to be installed*. **English** has been pre-selected for us. This is fine with us, so we click **Next** yet again.

Finally, we get the window is titled *Start Copying Files***,** and it contains a full list of the DB2 features to be installed. We review the list and click **Install.**

Next is the window titled *Installing DB2***.** This takes several minutes.

Then we get the window *Setup is Complete*. We read the informational message about DB2 documentation and click **Finish.**

We are returned to what looks like the *DB2 Installation Launchpad*, but it is now titled *DB2 First Steps*. At this point, you can explore some of the options offered here. We select the last option on the list: **Exit**

To verify installation of Replication Center we go to:

Start > Programs > DB2 > Administrative Tools > Replication Center

B

Appendix B.

Configuring Connections for Replication Center

This appendix discusses the information needed and the steps required to configure connections from a Replication Center workstation to source, Capture Control, Apply Control, target, and DB2 federated servers. By DB2 federated servers we mean DB2 ESE Version 8 databases (or database in other IBM product with exactly the same federated server capability as DB2 ESE Version 8).

In this appendix we will discuss establishing connectivity to:

- DB2 Universal Database for z/OS and OS/390
  - direct connection
  - connection via a DB2 Connect server
- DB2 for iSeries (AS/400)
  - direct connection
  - connection via a DB2 Connect server
- DB2 Universal Database for Linux and for UNIX
- DB2 Unversal Database for Windows

In each section we will first discuss the information that needs to be obtained from the data source and then how to configure the connection on the Replication Center workstation.

We will give examples using DB2 UDB Version 8 Configuration Assistant and using DB2 UDB Linux, UNIX, Windows commands. The DB2 Commands are described in the *DB2 UDB Version 8 Command Reference*. If you do not have it in hard copy or online, it can be found at:

http://www.ibm.com/software/data/db2/library

Select 'DB2 Universal Database'

B.1 Connecting Directly to DB2 for z/OS and OS/390

Information Required from DB2 for z/OS or OS/390 system

We'll need:

- Hostname or IP Address
- DB2 for z/OS Sub-system or Sharing Groups' Location Name
- DB2 for z/OS Sub-system/Sharing Group's TCP/IP listener port.

After you obtain the hostname or IP address of the zOS or OS/390 server, you should confirm TCP/IP connectivity, for instance using **`ping`** `from a DOS or UNIX command prompt.`

A DB2 for z/OS sub-system/sharing group's Location Name and TCP/IP listener port number is registered in its Bootstrap Dataset. (BSDS). If you can't look at the contents of the BSDS, two other ways to get this information are:

- Use command *prefix*DIS DDF

  where *prefix*=DB2 sub-system prefix that was defined in DB2 for z/OS Installation panel DSNTIPM

  The command can be issued from z/OS Console or SDSF System Log.

- Search in the MVS Console Log for the start message for the DB2 for z/OS sub-system's Distributed Data Facility (DDF).

  We are searching for DSNL004I.

- Run DB2 for z/OS Utility DSNJU004. The information we are looking for is at the very bottom of the output.

Whether you use *prefix*DIS DDF, the DSNL004I message in the System Log, or the utility DSNJU004, in the output look for the LOCATION and TCPPORT values for DDF.

In our example, the values are:

- System name: stplex4a.stl.ibm.com
- LOCATION: STPLEX4A_DSN6
- TCPPORT: 8008

Configuring Connection to DB2 for z/OS or OS/390

First, verify TCP/IP connetivity to the z/OS or OS/390 mainframe system. From a DOS-prompt, you could ping the mainframe system. For example:

```
C:\>ping stplex4a.stl.ibm.com
Pinging stplex4a.stl.ibm.com [9.30.4.77] with 32 bytes of data:
Reply from 9.30.4.77: bytes=32 time<10ms TTL=58
```

Then we can configure DB2 connectivity to DB2 for z/OS or OS/390 on that mainframe.

First we'll use the Configuration Assistant.

We open it with

Start>Programs>DB2>Setup Tools>Configuration Assistant.

Once *Configuration Assistant* opens, we want to add a new database. To do this click **Selected** from the menu bar and **Add database using Wizard** as in Figure B-1.

Figure B-1 Configuration Assistant - Add new database

The *Add a Database wizard* starts with the *Source* panel. The word 'Source' is in the upper lefthand corner of the notebook. On the Source panel we select **Manually Configure a Connection to a Database** and click **Next.**

The Protocol tab is next**.** we select **TCP/IP,** and the following tabs now appear on the left margin of the Add Database wizard notebook.

- Source - tab is already filled in
- Protocol -current tab
- TCP/IP - next tab to fill in
- Database (grayed)
- Data source (grayed)
- Node Options (grayed)
- Systems Options (grated)
- Security Options (grayed)

The Add Database wizard will walk us through each of these tabs. We can go forward or backward among the tabs to check or change information.

Here is how we fill in the information in each of the remaining tabs:

- Protocol
  - TCP/IP - already checked above
  - The database resides on a host or AS/400 - **be sure to check this**.
  - Connect directly to the server - check this also
- TCP/IP tab:
  - Hostname: stplex4a.stl.ibm.com
  - Service name: [blank]
  - Port: 8008
- Database

  Note that the description says, 'For z/OS and OS/390 databases, specify the Location name.'

  - Database name: STPLEX4A_DSN6
  - Alias: will automatically be first 8 characters of 'Database name' Can over-type.
- Data Source

  Note the description says this is for registering the data source as an ODBC System DSN. We want this.

  - Register this data base for ODBC - leave checked
  - As System data source - leave checked
  - Data source name: - we accept the default 'STPLEX4A'
  - Optimize for application - we accept 'None'
- Node Options
  - Operating System - OS/390 or z/OS
  - Remote instance name - [blank]
- System Options
  - System name - stplex4a.stl.ibm.com - we accept the default
  - Hostname - stplex4a.stl.ibm.com -we accept the default
  - Operating System - OS/390 or z/OS - we accept the default
- Security Options
  - Use authentication value in server's DBM Configuration - checked
- DCS Options
  - We make no entries.

We click **Finish** and are returned to the Configuration Assistant main window with an entry added for our DB2 for z/OS sub-system STPLEX4A as in Figure B-2.

Figure B-2 Configuration Assistant - DB2 for z/OS entry

The same connection could be configured in DB2 Command Line Processor using the following commands in Example B-1.

Example: B-1 DB2 CLP commands for connection direct to DB2 for z/OS

```
db2 catalog tcpip node stplex4a remote stplex4a.stl.ibm.com server 8008 system
stplex4a ostype os390

db2 catalog dcs db stplex4a as stplex4a_dsn6
db2 catalog db stplex4a at node stplex4a
db2 catalog system odbc data source stplex4a
```

Attention: If the DB2 CLP commands are used to configure the connection, be sure in the **`catalog tcpip node`** command to include OSTYPE. Replication Center uses this value to recognize that the server is a DB2 for z/OS or OS/390.

We'll cover testing connections and binding packages in "Testing Connections and Binding Packages" on page dxxxii since the techniques will be the same for DB2 for z/OS and OS/390, iSeries, and DB2 on Linux, UNIX, and Windows.

B.2 Connecting to DB2 for z/OS via DB2 Connect

Information needed from DB2 Connect server

If you'll actually be connecting to DB2 for z/OS via an intermediate DB2 Connect or DB2 ESE Server, you will need information from that intermediate system as if you were connecting to a DB2 UDB database on that system. We'll point out that if the DB2 Connect or DB2 ESE Server has DB2 Administration Service configured and running (which is the default during installation) then you may be able to configure the connection from your workstation to the DB2 for z/OS by asking Configuration Assistant the information to get from the DB2 Connect/ESE server are:

- hostname or IP address

  In our case,

  hostname = sthelens.almaden.ibm.com

  IP address = 9.1.38.178

- TCP/IP port number of the DB2 Server or DB2 Connect instance. If you will be depending on Configuration Assistant 'Search' capability, you'll only need the DB2 instance name.

  On the DB2 UDB/Connect system, logged in as the DB2 instance owner:

  ```
  db2 get dbm cfg | grep SVCENAME
  ```

  In our case, the result is:

  ```
  TCP/IP Service name                          (SVCENAME) = DB2_aix43db2
  ```

  If the DB2 Connect server is on Windows, where **grep** isn't available, you can either scan the whole output of the **db2 get dbm cfg** command looking for SVCENAME, or use the **DB2 Administration Tool --> Control Center**, highlighting the instance name for the DB2 Connect Server, right-mouse click, and select **Configure Parameters** from the available options. In the DBM Configuration window, SVCENAME is under *Communications*.

  If the result of this command is a name (as it is in our case) rather than a port number, then we need to check the /etc/services file to find out the associated port number:

  ```
  $ cat /etc/services | grep DB2_aix43db2
  ```

  In our case, the result is:

  ```
  DB2_aix43db2    60012/tcp
  ```

  On windows the TCP/IP Services file is in the folder:

  - c:\Winnt\System32\Drivers\etc

- DB Alias name in the DB2 Database Directory entry for the DB2 for z/OS sub-system. shows the command to obtain the DB2 Alias name.

Example: B-2 DB2 z/OS Alias name at DB2 Connect server

```
db2 list db directory | more

   Database 1 entry:
   Database alias                       = STPLEX4A
    Database name                        = STPLEX4A_DSN6
    Node name                            = STPLEX4A
    Database release level               = a.00
    Comment                              =
    Directory entry type                 = Remote
    Catalog database partition number    = -1
```

In summary, the information we needed is:

- system name = sthelens.almaden.ibm.com
- port number = 60012
- Database Alias = STPLEX4A

Configuring DB2 z/OS connection via DB2 Connect server

The difference from doing the connection directly to the DB2 for z/OS sub-system, is that we have to configure the connection as if the DB2 for z/OS sub-system were a database at the DB2 Connect Server.

First, verify TCP/IP connectivity to the DB2 Connect server. For instane, at a DOS-prompt using ping:

```
C:\>ping sthelens.almaden.ibm.com
Pinging sthelens.almaden.ibm.com [9.1.38.178] with 32 bytes of data:
Reply from 9.1.38.178: bytes=32 time=10ms TTL=255
```

Next we configure DB2 connectivity.

Using *Configuration Assistant*:

- Source
  - Manually configuration a connection to the data source.

  Here is where we could try 'Search the network' and see if DB2 Administration Server at the DB2 Connect server can provide all the information needed to configuration by 'point-and-click.'
- Protocol
  - TCP/IP
  - Database physically resides on host or OS/400 system - check this
  - Connect to server via gateway - check this also
- TCP/IP

- Hostname: sthelens.almaden.ibm.com - our AIX system with DB2 ESE.
- Service - [blank]
- Port - 60012 - TCP/IP listener port of DB2 ESE on sthelens

- Database
  - Database name: STPLEX4A - the Database Alias name in the Database directory on sthelens
- Data Source
  - Register as ODBC System DSN - STPLEX4
- Node Options
  - Operating system - AIX
  - Instance name - aix43db2
- System Options
  - no changes (accepted defaults)
- Security Options
  - no changes (accepted default values)
- Finish

The new entry in Configuration Assistant should appear as in Figure B-3.

Figure B-3 Configuration Assistant - DB2 for z/OS via DB2 Connect Gateway

The DB2 CLP Commands to do the same configuration are in Example B-3:

Example: B-3 DB2 CLP commands for B2 for z/OS via DB2 Connect server

```
db2 catalog tcpip node aix43db2 remote sthelens.almaden.ibm.com server 60012
system sthelens ostype aix

db2 catalog db stplex4a at node aix43db2
db2 catalog system odbc data source stplex4a
```

Please note in this example:

- we picked a node name that is the same as the DB2 instance name on the AIX server.
- sthelens.almaden.ibm.com is the hostname of our DB2 Server on AIX.
- 60012 is the TCP/IP listener port of the DB2 Server (or DB2 Connect) on the AIX server.

Note: when configuring the connection through a DB2 Connect or DB2 Server on Linux, UNIX, or Windows, we don't execute the 'db2 catalog dcs db...' command.

As before when we configured the connection directly to DB2 for z/OS, we can test the connection and bind using Configuration Assistant.

B.3 Connecting directly to iSeries (AS/400)

Information needed from iSeries

We'll need:

- hostname or IP address
- Relational Database name of the iSeries system

Note: We'll also need the iSeries DRDA-TCP/IP listener port number, but we expect that to be the standard, which is 446.

When you have the hostname or IP address of the iSeries system, we recommend that you verify TCP/IP connectivity, such as by opening a DOS or UNIX command prompt and using **`ping`**.

To get the Relational Database name of the iSeries system, we need to log onto the iSeries system and look at the output of command WRKRDBDIRE. We are looking for the entry where Remote Location = *LOCAL.

On our iSeries system STL400G.STL.IBM.COM, we find in the second page of output for WRKRDBDIRE as in Figure B-4.

Figure B-4 iSeries Relational Database name in WRKRDBDIRE

So in our case,

- hostname = stl400g.stl.ibm.com
- Relational Database name = STL400G

We'll point out some other things that need to be in place an iSeries system for it to be accessed from DB2 Connect on Linux, UNIX, or Windows:

- Library/Collection NULLID must exist. It will be used for the DB2 Connect Utility, CLP, and ODBC packages. It can be created with the CL command:
  `CRTLIB LIB(NULLID)`
- The DDM TCP/IP job needs to have certain attributes. They can be checked/changed with command
  `CHGDDMTCPA`

  On the resulting entry screen, we need to see

  ```
  Autostart server *YES
  Password required *YES
  ```

  If you get security errors (SQL30082) when trying to access from DB2 Connect, try changing the Password setting in CHGDDMTCPA.
- The DDM TCP/IP Server job needs to be running. It can be started with:
  `STRTCPSVR SERVER (*DDM)`

  To see if DDM is running:

  ```
  WRKACTJOB SBS(QSYSWRK)
  ```

  and check for QRWTLSTN in the QSYSWRK subsystem.
- If the CCSID of the iSeries system is 66535 (which is the default CCSID for iSeries systems), then the CCSID of the iSeries Userid specified on the connection from DB2 Connect needs to be changed. The CCSID of AS/400 user HCOLIN can be changed to the standard English CCSID '37' with the command:
  `CHGUSRPRF USRPRF(HCOLIN)) CCSID(37)`

In summary, the information we needed from the iSeries system is:

- system name = stl400g.stl.ibm.com
- port number = 446 (DRDA listener port on iSeries)
- Relational Database name = STL400G

Configuring connection direct to iSeries

Back on Windows, we can open the Configuration Assistant with

Start>Programs-->DB2-->Setup Tools-->Configuration Assistant

Once there, we can configure a new connection to an iSeries system by choosing **Selected --> Add Database using wizard.**

Here is how we complete the configuration for our example using Configuration Assistant:

- Source
  - Manually configure a connection to a database
- Protocol
  - TCP/IP
  - Database resides on a host or AS/400 - check this
  - Connect directly to the server - check this also
- TCP/IP
  - Hostname: stl400g.stl.ibm.com
  - Service name: [blank]
  - Port: 446 - this is the DRDA listener port on iSeries
- Database
  - Database name: STL400G (the RDB name from WRKRDBDIRE)
- Data Source
  - Register this data source for ODBC
  - As System DSN
  - Data Source name: STL400G
  - Optimize for application: None
- Node Options
  - Operating system: OS/400
  - Instance: [blank]
- System options
  - System name: stl400g.stl.ibm.com
  - Hostname: stl400g.stl.ibm.com
  - Operating system: OS/400
- Security Options
  - User authentication value in server's DBM configuration
- DCS Options
  - No specifications
- **Finish**

Our new entry for the iSeries in the Configuration Assistant looks as in Figure B-5.

Figure B-5 Configuration Assistant - iSeries connection entry

As indicated before, we can use the Configuration Assistant to test the connection to the iSeries and to bind DB2 Connect Utility, CLP, and ODBC packages to the iSeries. From the menu bar, select ‘Selected’ and then ‘Bind’ or ‘Test Connection’.

The connection to the same iSeries system could have also been configured with the following DB2 CLP commands in Example B-4:

Example: B-4 DB2 CLP commands for connection direct to iSeries

```
db2 catalog tcpip node stl400g remote stl400g.stl.ibm.com server 446 system
stl400g ostype os400

db2 catalog dcs db stl400g as stl400g
db2 catalog db stl400g at node stl400g
db2 catalog system odbc data source stl400g
```

Attention: If the DB2 CLP commands are used to configure the connection, be sure in the **`catalog tcpip node`** command to include OSTYPE. Replication Center uses this value to recognize that the server is an iSeries.

We’ll cover testing connections and binding packages in “Testing Connections and Binding Packages” on page dxxxii since the techniques will be the same for DB2 for z/OS and OS/390, iSeries, and DB2 on Linux, UNIX, and Windows.

B.4 Connecting to iSeries via DB2 Connect server

The information needed and the process followed to obtain it are similar to that in “Connecting to DB2 for z/OS via DB2 Connect” on page dxiii. The intermediate DB2 Connect or DB2 ESE Server will need to be already configured to connect to the iSeries system and we will need information about the DB2 Connect / DB2 ESE Server and the entry in its Database Directory for the iSeries system. The collection of this information is discussed in Connecting to DB2 for z/OS via DB2 Connect.

In our example here, the needed infomation is:

- TCP/IP hostname = sthelens.almaden.ibm.com
- TCP/IP port number = 60012
- Database Alias = STL400G

Configuring connection to iSeries via DB2 Connect server

Here is how we configured the connection via our DB2 ESE Server on AIX:

- Source
  - Manually configure

  Here is where we could try ‘Search the network’ and see if DB2 Administration Server at the DB2 Connect server can provide all the information needed to configuration by ‘point-and-click.’
- Protocol
  - TCP/IP
  - Database physically resides on host or AS/400 - check this
  - Connect to the server via the gateway - check this
- TCP/IP
  - Hostname: stlhelens.almaden.ibm.com
  - Service name: [blank]
  - Port: 60012
- Database
  - Database name: STL400G
  - Alias: STL400G
- Data Source
  - Register this database for ODBC
  - As System Data Source
  - Data source name: STL400G
- Node options
  - Operating system: AIX
  - Instance name: aix43db2
- System options
  - System name: stlhelens.almaden.ibm.com
  - Hostname: sthelens.almaden.ibm.com

- Operating system: AIX

► Security Options
  - Use authentication value in Server's DBM Configuration

Finish

To do the same configuration using DB2 Command Line on Windows, we used the following commands in Example B-5:

Example: B-5 DB2 CLP commands for connection to iSeries via DB2 Connect server

```
db2 catalog tcpip node aix43db2 remote sthelens.almaden.ibm.com server 60012
system sthelens ostype aix

db2 catalog db stl400g at node sthelens

db2 catalog system odbc data source stl400g
```

B.5 Connecting to DB2 UNIX or Linux server

Before covering the information to be gathered, we'd like to point out that if the DB2 on UNIX or Linux server has its DB2 Administration Server running you may be able to configure the connection from you workstation to the databases in DB2 on UNIX or Linux very easily in DB2 Configuration Assistant with minimum of information. For DB2 for Linux, UNIX, and Windows server products, the default installation behavior is to configure and Autostart the Administration Server. The DB2 Configuration Assistant on your workstation can use the DB2 Administration Client to connect to the server and send it commands to find information about DB2 instances and databases on the server and bring this information back for display in Configuration Assistant. That being the case, you would just need to know:

- hostname or IP address
- DB2 instance name
- DB2 database name

for the database you want to configure connection to.

Here we'll assume you may need to configure the connection from your workstation to the DB2 for Linux or UNIX 'manually,' keying the information required.

The information you'll need about the DB2 for UNIX or Linux server are:

- hostname or IP address
- TCP/IP Listener Port number of the DB2 instance
- the Database Alias name of the database.

After obtaining the hostname or IP address, we recommend verifying TCP/IP connectivity, such as by going to a DOS or UNIX command prompt on the Replication Center workstation and using **ping.**

To get the TCP/IP port number and database name, we need to log into the DB2 server system as the DB2 instance owner, for instance using telnet.

To get the TCP/IP port number we enter:

```
db2 get dbm cfg | grep SVCENAME
```

Result is:

```
TCP/IP Service name (SVCENAME) = DB2_aix43db2
```

Since the SVCENAME value is a Service *name* (DB2_aix43db2), and not a port *number* we need to look in */etc/services* to get the TCP/IP listener port number that is associated with the DB2 instance's TCP/IP Service name. The command to do this is, in our example, is:

```
cat /etc/services | grep DB2_aix43db2
```

The result is:

```
DB2_aix43db2    60012/tcp
DB2_aix43db2_1  60013/tcp
DB2_aix43db2_2  60014/tcp
DB2_aix43db2_END        60015/tcp
```

The number we want is the first one: 60012.

To get the Database Alias name, we need to look at the DB2 instance's Database Directory. Example B-6 shows the command to do this.

Example: B-6 Obtaining DB2 UNIX Alias from DB directory

```
db2 list db directory | more

 Database alias                       = AIX43DB2
  Database name                        = AIX43DB2
  Local database directory             = /db2
  Database release level               = a.00
  Comment                              =
  Directory entry type                 = Indirect
  Catalog database partition number    = 0
```

So, in summary, the information we needed was:

- TCP/IP hostname = sthelens.almaden.ibm.com
- TCP/IP listener port = 60012Database Alias = AIX43DB2'

Configuring Connection to DB2 server on UNIX and Linux

Firstl we should verify TCP/IP connectivity to the DB2 server.

We'll use ping at a DOS-prompt:

```
c:\>ping sthelens.almaden.ibm.com
Pinging sthelens.almaden.ibm.com [9.1.38.178] with 32 bytes of data:
Reply from 9.1.38.178: bytes=32 time=10ms TTL=255
```

Next we'll configure DB2 connectivity using both the DB2 Configuration Assistant and the DB2 CLP commands.

We open DB2 Configuration Assistant here on our workstation by doing:

Start>Programs>DB2>Setup Tools>Configuration Assistant.

Once there, we go to the Tool Bar at the top and pick **Selected --> Add database using wizard.**

We fill out the Add database wizard's notebook as follows:

- Source
  - Manually configure a connection to the database

  Note: here is where you could select 'Search the network' and see if Configuration Assistant can 'find' the DB2 server system and get the information from it to let you do the connection configuration all by 'point-and-click.'
- Protocol
  - TCP/IP

  We leave unchecked 'Database physically resides on host or AS/400'
- TCP/IP
  - Hostname: sthelens.almaden.ibm.com
  - Service name: [blank]
  - Port: 60012
- Database
  - Database name: AIX43DB2
  - Database alias: AIX43DB2
- Data source
  - Register this database for ODBC
  - As system data source
  - Data source name: AIX43DB2
- Node Options
  - Operating system: AIX
  - Instance name: aix43db2
- System Options
  - System name: sthelens.almaden.ibm.com
  - Hostname: sthelens.almaden.ibm.com
  - Operating System: AIX
- Security Options
  - Use authentication value in Server's DBM Configuration

Finish

The DB2 CLP commands to accomplish the same thing are in Example B-7:

Example: B-7 DB2 CLP commands for connection to DB2 on UNIX

```
db2 catalog tcpip node aix43db2 remote sthelens.almaden.ibm.com server 60012
system sthelens ostype aix

db2 catalog db aix43db2 at node aix43db2
db2 catalog system odbc data source aix43db2
```

Our new entry for the DB2 for Linux/UNIX database in DB2 Configuration Assistant appears as in Figure B-6.

Figure B-6 Configuration Assistant - DB2 on Linux or UNIX connection entry

We'll cover testing connections and binding packages in "Testing Connections and Binding Packages" on page dxxxii since the techniques will be the same for DB2 for z/OS and OS/390, iSeries, and DB2 on Linux, UNIX, and Windows.

B.6 Connecting to DB2 on Windows

We recommend that you use TCP/IP, rather than Named Pipes or NetBIOS, as network communications protocol for your connection from DB2 Administrative Client on your workstation to a DB2 Windows server. For one thing, the DB2 Administration client-to-server communications is only support over TCP/IP and DB2 Administration Server (DAS) will enable you to start and stop the DB2 Replication Capture, Apply, and Monitor processes from DB2 Replication Center on your workstation.

As with DB2 on Linux and UNIX, the DB2 Administration Server may be configured and running on the DB2 Windows server and make configuring the connection from your workstation to the DB2 database on the Windows server much easier using Client Configuration Assistant. On the Configuration Assistant's 'Add Database using wizard' 'Source' panel, you will select 'Search the Network' and Configuration Assistant will prompt you for information. You will need to know the System name, Instance Name (probably 'DB2') and database name.

We'll do the configuration 'manually' here.

Information from DB2 Windows server.

The information we need from the DB2 Windows server is the same as from a DB2 Linux or UNIX server:

- TCP/IP Hostname (or IP address)
- DB2's instance's TCP/IP Listener Port
- DB2 Database Alias name

In our example, the system name is 'micks'

To obtain the TCP/IP Listener Port and Database Alias name we need to go to the DB2 Windows server.

To get the TCP/IP Listener Port, we need to see the DB2 instance's Database Manager Configuration. We can go to a DB2 Command Window prompt and enter:

```
db2 get dbm cfg | more
```

and look for 'TCP/IP Service Name SVCENAME' in the result. There are many parameters in the output. SVCENAME is near the bottom.

An alternative is open the DB2 Control Center on the DB2 Windows server, select the instance (DB2), right mouse click, and select 'Configure Parameters.' Look under 'Communications' for SVCENAME. We find

```
SVCENAME micksdb2
```

Since the SVCENAME is a Service name and not a TCP/IP Port number, we need to look in the Windows TCP/IP services file on the DB2 Windows server to find out the port number. The services file is in folder:

c:\WINNT\system32\drivers\etc

We can open the 'services' file with Notepad and use Edit>Find to look for 'micksdb2'. We find the entry:

```
micksdb2 3846/tcp
```

We'll point out here that these are a customized Service Name and TCP/IP Port number for DB2 on Windows. If the DB2 Installation process on the DB2 Windows server had configured the TCP/IP communications, the Service Name would be more like db2c_DB2 and TCP/IP port number would probably be 50000.

To obtain the database alias name, we can either use another command in DB2 Command Window or we can use the Control Center. Example B-8 shows the command to do this.

Example: B-8 Obtaining the DB2 Windows Alias name

```
db2 list db directory

 Database alias                       = SAMPLE
  Database name                        = SAMPLE
  Local database directory             = c:\DB2
  Database release level               = a.00
  Comment                              =
  Directory entry type                 = Indirect
  Catalog database partition number    = 0
```

In the Control Center on the DB2 Windows Server, if we look under the instance 'DB2' at the Databases, we find the database 'SAMPLE.' If the database we are looking for had an Alias that was different from the database name, what we would see is something like 'MICKSDB(SAMPLE)' where SAMPLE is the real Database name and MICKSDB is the Database Alias. It is the Database Alias that we need. In our case here, the Database Name and Database Alias are the same: SAMPLE.

So, in summary, the information we needed from the DB2 Windows server was:

- System Name: MICKS
- DB2 TCP/IP Listener Port: 3846

– DB2 Database Alias: SAMPLE

Configuring Connection to DB2 Server on Windows

We'll do this in our example using both the DB2 Configuration Assistant and the DB2 CLP commands.

When we have the hostname or IP address of the DB2 Windows server, we should verify TCP/IP connectivity, such as by opening a DOS prompt on the Replication Center workstation and using **ping.**

Then we open DB2 Configuration Assistant here on our workstation by doing:

Start>Programs>DB2>Setup Tools>Configuration Assistant.

Once there, we go to the Tool Bar at the top and pick **Selected > Add database using wizard.**

We fill out the Add database wizard's notebook as follows:

- Source
  - Manually configure a connection to the database

  Note: here is where you could select **Search the network** and see if Configuration Assistant can 'find' the DB2 server system and get the information from it to let you do the connection configuration all by 'point-and-click.'
- Protocol
  - TCP/IP

  We leave unchecked 'Database physically resides on host or AS/400'
- TCP/IP
  - Hostname: micks.stl.ibm.com
  - Service name: [blank]
  - Port: 3846
- Database
  - Database name: SAMPLE
  - Database alias: SAMPLE.

  Note: If we had DB2 UDB on our workstation containing a database SAMPLE, we could use a Database alias for the SAMPLE database at the DB2 Windows server to avoid conflicting entries in the DB2 Database Directory here on our workstation. Or, prior to configuring the connection to the remote DB2 Windows server, we could either drop our local SAMPLE database, or uncatalog it and catalog it again with an Database Alias name.
- Data source

- Register this database for ODBC
- As system data source
- Data source name: SAMPLE

- Node Options
  - Operating system: Windows
  - Instance name: DB2
- System Options
  - System name: micks.stl.ibm.com
  - Hostname: micks.stl.ibm.ocm
  - Operating System: Windows
- Security Options
  - Use authentication value in Server's DBM Configuration

Finish

The DB2 CLP commands to accomplish the same thing are shown in Example B-9

Example: B-9 DB2 CLP Commands for connection to DB2 on Windows

```
db2 catalog tcpip node micks remote micks.stl.ibm.com server 3846 system
micks ostype win

db2 catalog db sample at node micks

db2 catalog systm odbc data source sample
```

The resulting Configuration Assistant entry is as in Figure B-7.

Figure B-7 Configuration Assistant - DB2 on Windows connection entry

We can test the connection to the DB2 Windows server, and, if needed, bind packages also using Configuration Section. See the next section, Testing Connections and Binding Packages.

B.7 Testing Connections and Binding Packages

There are many options for testing DB2 connectivity from your workstation to your replication source, target, and to other DB2 servers, including to ones providing federated access to non-DB2 data sources. We'll cover two here: DB2 Configuration Assistant and DB2 Command Line Processor. Also, if you need to bind packages at your DB2 servers, you can also do that either via the Configuration Assistant or DB2 CLP.

We're finding that the DB2 Administrative Client appears to be automatically binding the packages it needs at DB2 data sources and so no explicit steps may be required to bind the packages needed by Control Center, Command Center, Replication Center, Command Line Processor, and ODBC/CLI applications. But if you see 'SQL0805' or '-805,' or 'package NULLID.____' not found' messages when using Replication Center, Control Center, Command Center or other software on your workstation to access the DB2 for z/OS data source, it will be because some packages need to be bound from the workstation to the data source. We'll cover how to do that here.

To test the connection to a DB2 server using the Configuration Assistant, open the *Configuration Assistant*, highlight a particular database, and in the Tools bar at the top, select **Selected-->Test Connection**. See Figure B-8.

Figure B-8 Configuration Assistant - Selected - Test connection

The Test Connection window opens as in in Figure B-9.

Figure B-9 Configuration Assistant - Test Connection window

We select the following options:

- Standard - tests DB2 Command Line Processor and embedded SQL interface
- CLI and ODBC - may be used for query tools and other ODBC software
- JDBC - will definitely be used by the DB2 Administration tools that are written in Java

We include a User ID (with password) that we got from the data source (mainframe host, iSeries, Linux, UNIX, or Windows), and press **Test Connection.** Next we might see a black 'Dos Prompt' type window while the connection request is being process, then we should see a result window like in Figure B-10.

Figure B-10 Configuration Assistant - Connection Test successful

Some errors we could see, and their causes are:

- SQL1336: We entered the wrong system name in Configuration Assistant.
- SQL30081- TCP/IP -‘10061’ or ‘10060’: At the data source, or the intermediate DB2 Connect Server, the DB2 TCP/IP Listener isn’t running, the DB2 Service isn’t started, or on our workstation we entered the DB2 TCP/IP Listener port information incorrectly in Configuration Assistant.
- SQL30082: The userid we entered is not correct, doesn’t exist at the data source, or the password we entered is not correct.

We can also make the connection test in a DB2 Command Line Processor window to DB2 for z/OS or OS/390, iSeries, or DB2 for Linux, UNIX, Windows server. Example B-10shows how to do this. In the example, we enter the **connect** command without specifying a password, so we are prompted for the password to use on the connection.

Example: B-10 Testing connectivity using DB2 CLP

```
db2 connect to sample user micks

Enter current password for micks:
```

```
Database Connection Information
Database server        = DB2/NT 8.1.0
SQL authorization ID   = MICKS
 Local database alias   = SAMPLE
```

If we need to bind any packages from our workstation to a DB2 for z/OS or OS/390, iSeries, or DB2 Linux, UNIX, Windows server, we can do that either using Client Configuration Assistant or DB2 Command Line Processor.

Using Configuration Assistant, highlight the particular database, and on the tool bar, select '**Selected-->Bind**'. The *Bind* dialogue box opens, like in Figure B-11.

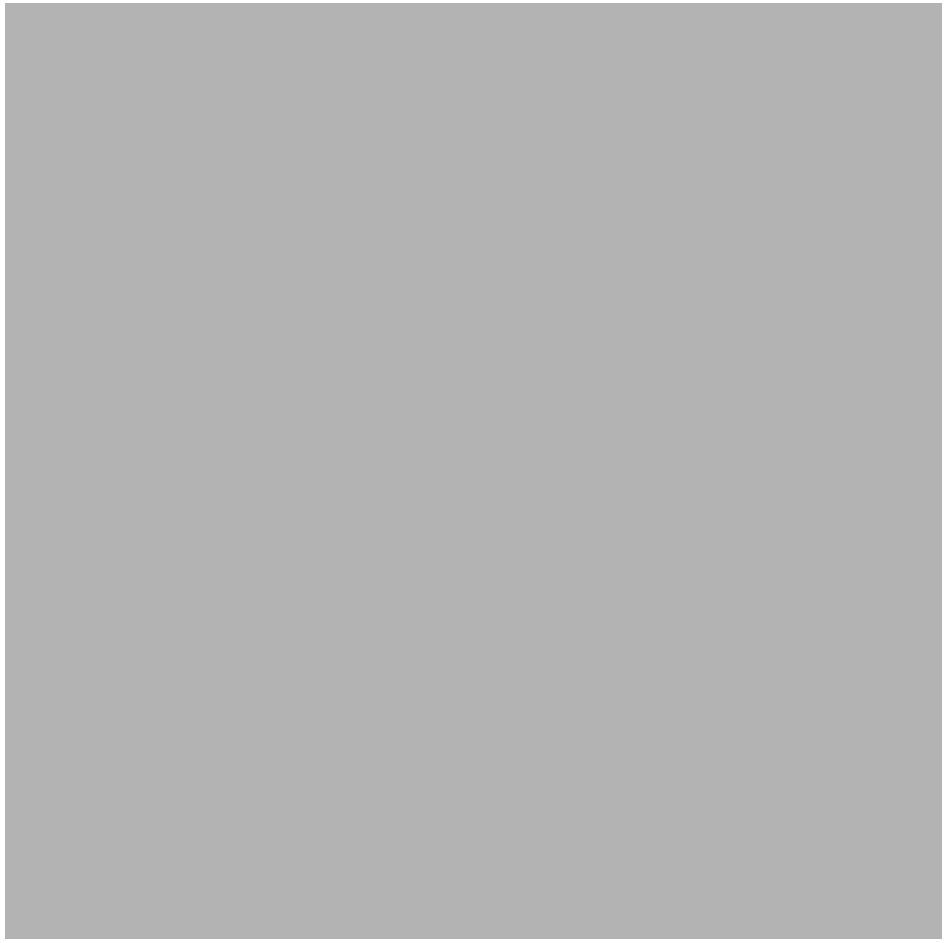


Figure B-11 Configuration Assistant - Bind dialog window

In this window, we can select one of the groups of DB2 Client packages (in the example, we've selected the 'CLI/ODBC Support' packages, or we can indicate another BND file on our workstation.

If we want to override any of the default values for the options associated with the BND files, we can focus on Bind Options in the middle of the window and press Add and we will be presented with a list of Options from which we can select and specify a value.

Under connection information, we fill in a userid we obtain from the data source system that can connect to the data source and can bind packages.

We press the **Bind** button in the lower right corner.

The Results tab should move to the foreground and show us if our packages are successfully bound or if we get have any errors.

We could also bind packages using a DB2 Command Window. The BND files for the DB2 Client facilities, such as for Command Line Processor, ODBC/CLI, REXX, and DB2 Utilities, are in the folder

c:\Program Files\IBM\SQLLIB\bnd

They can be bound in groups, rather than individually, using the *.LST files.

To bind packages using DB2 CLP, you must first connect to the data source where you wish to bind the packages. See the DB2 CLP Connect example above. You should 'cd' to the directory containing the BND files. On Windows, that would be an example of the command to bind the appropriate BND files to DB2 for z/OS is:

```
db2 bind @ddcsmvs.lst blocking all sqlerror continue
```

The Bind command is described in the *DB2 UDB Version 8 Command Reference* SC09-4828.

Again, it appears that DB2 Administrative Client is automatically binding all the packages it needs at each DB2 data source and so it should be unnecessary for you to have to explicitly bind any packages to use Replication Center. Binding is described here just in case it is needed for some reason.

C

Configuring federated access to Informix

In this appendix we will cover the details of configuring federated access from DB2 ESE Version 8 to an Informix IDS server. Federated access is required for replication from Informix and for replication to Informix. We will cover the configuration steps in this order:

- technical requirements for federated access to Informix
- obtaining information from the Informix server
- software required on the DB2 ESE system
- installation steps for DB2 ESE to enable federated access
- updating a DB2 ESE instance for federated access
- oonfiguring federated objects in a DB2 ESE database

C.1 Technical requirements for federated access

There are requirements both at the Informix server and at the DB2 ESE Version 8 server.

At the Informix IDS server

The Informix server can be either IDS or XPS version 7.2 or later. -GLS and -SE Informix servers are not supported for federated access from DB2 ESE Version 8.

If DB2 ESE Version 8 will be on a different server than the Informix, the Informix server's TCP/IP protocol listener, which is required for connections from Informix Client SDK, must be configured and running. We will cover how to check for this and to get the information about the TCP/IP protocol listener.

At the DB2 ESE Version 8 server

Federated access to Informix is supported by both DB2 ESE Version 8 or DB2 Connect Enterprise Edition Version 8.

Informix Client SDK 2.4 or later must be installed on the DB2 ESE or DB2 Connect Version 8 server. Informix I-Connect is not enough. I-Connect does not include the 'archive' libraries that contain the API's called by the DB2 ESE Version 8 Informix wrapper. Those API's are only in the that 'archive' libraries that come with Informix Client SDK.

C.2 Information from the Informix server

From the Informix server, you will need:

- a userid that will have the authorities and privileges required to set up replication and to do the operations required by Apply
  - if replicating from Informix
    - read the Informix system catalog
    - create tables, procedures, and triggers
  - if replicating to Informix
    - if target tables already exist, need to be able to insert, update, delete records into the target tables
    - if target table don't yet exist, create table
- information about the Infomrix TCP/IP protocol listener. See the discussion below on how to obtain this information
- the name of the database within the Informix server that has the target tables or the source tables for replication. Informix database names are case

sensitive. This can be found using Informix's dbaccess. In our example, it will be 'stores_demo'

Information about the Informix TCP/IP protocol listener

What we need to do is verify that Informix server has an operational TCP/IP protocol listener and to obtain the information about the Informix TCP/IP listener to configure a sqlhosts entry at the DB2 ESE Version 8 server.

The operational listeners for an Informix server are determined from the specifications in a number of different variables and files:

ONCONFIG environment variable setting indicates the name of the active onconfig file for an Informix server instance.

Within the active onconfig file the DBSERVERNAME and DBSERVERALIASES values indicate the relevant entries in the local sqlhosts file.

Within the local sqlhosts file, the entries with dbservername values matching the onconfig DBSERVERNAME and DBSERVERALIASES values tell the server:

- the system name
- the listeners to activate
- the TCP/IP service name or port number for the TCP/IP listeners.

If the TCP/IP listener(s) are to be active and the sqlhosts file indicates a service name (rather than port number) than there needs to be an entry in the /etc/services file to indicate the associated port number for that service name.

We will cover an example of gathering this information for an Informix IDS Version 9.3 server running on AIX:

The system name is viper.svl.ibm.com

Our userid on that system is gjuner2

We telnet to tat system.

If we can, we **su** to the userid for the Informix instance, probably 'informix.'

Use **echo** or **env** to find out the value of ONCONFIG. In our case:

```
ONCONFIG=onconfig.inf93
```

We also need to know the Informix main directory. This is indicated by environment variable INFORMIXDIR. On this system:

```
INFORMIXDIR=/informix/93server
```

We should aslo see the setttings for INFORMIXSERVER and INFORMIXSQLHOSTS. The latter indicates if the active sqlhosts file for this instance is in another directory besides the default location, which is

/INFORMIXDIR/etc. In our case, we find that INFORMIXSQLHOSTS is not set, which means that the active sqlhosts file is in /INFORMIXDIR/etc.

We will need to see the active onconfig file and the sqlhosts file. The will be in INFORMIXDIR/etc; in otherwords, in

/informix/93server/etc

We **cd** to that directory. We use **ls** to see the names of the files in that directory to verify that the onconfig file we are looking for (*onconfig.inf93*) and the *sqlhosts* file are there.

We need to see the DBSERVERNAME and DBSERVERALIASES values in the active onconfig file. We could use **vi** or we could use **cat** with **grep**:

```
cat onconfig.inf93 | grep DBSERVERNAME
```

The result is

```
DBSERVERNAME    inf93 #Name of default database server
```

Then we check DBSERVERALIASES

```
cat onconfig.inf93 | grep DBSERVERALIASES
```

The result is:

```
DBSERVERALIASES #Alternate DB servernames
```

(Since there is no value between the variable name and the '#', the variable is not set with a value).

Next we need to look in the *sqlhosts* file. It is also in INFORMIXDIR/etc. The file is probably not large so we could use **vi**. In sqlhosts we find:

```
inf724          onsoctcp        anaconda        ifmx724
inf92           onsoctcp        python          ifmx92
inf93           onsoctcp        viper           ifmx93
inf731          onsoctcp        boa             ifmx731
```

For those who are not familiar with format of *sqlhosts* entries:

- first value is the dbservername
- second value is the Informix protocol
- third value is the system name
- fourth value is the TCP/IP service name or port number.

The record we are looking for is the third one (inf93). It indicates:

- dbservername = inf93
- Informix protocol = onsoctcp
  - This is for one of the Informix TCP/IP protocols. This what we need for Informix Client SDK to connect to this Informix server from the DB2 ESE Version 8 system using federated access.

– TCP/IP Service name = ifmx93

Since the fourth field is a service name, not a port number, we need to look in the systems TCP/IP services file (etc/services) to make sure there is an entry there and to find out the port number. We can use cat with grep:

cat /etc/services | grep ifmx93

The result is:

```
ifmx93          1652/tcp # Informix online v9.3 viper
```

Let's summarize the infomration we have obtained from the Informix server:

– userid: gjuner2
– Informix DB server name: inf93
– Informix TCP/IP protocol listener: onsoctcp
– Informix server system name: viper.svl.ibm.com
– Informix server's TCP/IP service name: infx93
– Informix server's TCP/IP port: 1652
– Informix database name: stores_demo

For an Informix server on Windows, the DBSERVERNAME of the Informix/Windows server can be found in the Windows Register. Open the Windows Registry (at a DOS command prompt, enter **regedit),** then select **HKEY_LOCAL_MACHINE-->SOFTWARE-->Informix-->Online-->*dbservername***

In the *Environment* folder under the our *dbservername,* we can find more information, including the name of the *onconfig* file. The *onconfig* file referenced can be found in INFORMIXDIR/etc. Typically that would be in

```
c:\Program Files\informix\etc
```

The *sqlhosts* entries on the Infomix/Windows server machine can also be found in the Windows Registry. Select **HKEY_LOCAL_MACHINE-->SOFTWARE-->Informix-->SQLHOSTS-->*server name***

If the protocol is 'olsoctcp' for the SQLHOSTS entry referenced by DBSERVERNAME in the onconfig information, this is ok.

The TCP/IP services file on Windows can be found in the folder *c:\Winnt\System32\Drivers\etc*.

C.3 Software installation on the DB2 system

If DB2 ESE Version 8 is on a different system then the Informix server, than Informix Client SDK must be installed on the DB2 system. And DB2 ESE Version 8 or DB2 Connect Enterprise Edition Version 8 must be installed.

Informix Client SDK

We won't cover the installation details here. Please consult Informix documentation. Installation of Client SDK would be covered in the *Informix Client Products Installation Guide for UNIX, Linux, and Windows.* Configuration of sqlhosts entries is covered in the *Informix Dynamic Server Administration Guide* in the chapter on Client/Server Communications. The Informix product documentation is online at:

```
http://www-3.ibm.com/software/data/informix/pubs/library/
```

When the Informix Client SDK installation is complete, you should have the information for the INFORMIXDIR environment variable. In our case it is

```
INFORMIXDIR=/home/informix
```

If the Informix client software was previously installed on the system, you can verify that you have the Client SDK by going to the INFORMIXDIR/lib directory and using **ls *.a** (on AIX) to verify that archive libraries of the Client SDK are there. For instance on our AIX system, in */home/informix/lib* we find *netstub.a*.

If we cd to INFORMIXDIR/lib/esql, we find more of the archive libraries of the Informix Client SDK that DB2 federated access will need. The Informix Client SDK archive libraries on AIX end with the extension '.a'.

DB2 ESE or DB2 Connect EE Version 8

DB2 ESE or DB2 Connect Version 8 will need to be installed by root.

root should run **./db2setup** from the installation CD or downloaded installation files. Among the installation choices, be sure the installed filesets include:

```
db2_08_01.dj              8.1.0.0    C    DB2 Data Source Support
db2_08_01.djinx           8.1.0.0    C    Informix Data Source Support
db2_08_01.djx             8.1.0.0    C    Relational Connect Common
```

If there are fixpacks available for DB2 Version 8. the latest should be downloaded at this time and applied before the next step (djxlink).

djxlink

On AIX, Solaris, HP-UX, and Linux, the wrapper library that DB2 uses to interface with the Informix Client SDK has to be built by a link between the

wrapper input library and Informix Client SDK libraries. The wrapper input library for creating the Informix wrapper on AIX is

/usr/opt/db2_08_01/lib/libdb2STinformixF.a

On Solaris and Linux, this would be */opt/IBM/V8.1/libdb2STinformixF.so*

On HP-UX, it would be */opt/IBM/V8.1/libdb2STinformix.sl*

It is important that djxlink be run *after* each fixpack is applied so that that wrapper library that is in use is at the same fixpack level as the DB2 engine.

djxlink needs to be run by root.

Just before running djxlink, root needs to export the INFORMIXDIR variable to point to the location of Informix Client SDK. In our example:

```
export INFORMIXDIR=/home/informix
```

For creating the Informix wrapper library, use (on AIX):

/usr/opt/db2_08_01/bin/djxlinkInformix

djxlinkInformix will write detailed warning/error messages to

/usr/opt/db2_08_01/lib/djxlinkInformix.out

If successful, djxlinkInformix will create the Informix wrapper library:

/usr/opt/db2_08_01/lib/libdb2informix.a

Note: on Windows, djxlink is not run. DB2's Informix wrapper library (db2informix.dll) is a dynamic link library (dll) as are the Informix Client SDK libraries.

C.4 Configuring Informix sqlhosts

The Informix *sqlhosts* file on the DB2 ESE or DB2 Connect EE system needs to have an entry for the Informix server.

Informix sqlhosts is normally in the directory INFORMIXDIR/etc. In our example, that is */home/informix/etc*

The format of the entries on the client is the same as on the Informix server. This is described in the *Informix Dynamic Server Administration Guide* in the chapter on Client/Server Connectivity. In our example the record in sqlhosts here on the DB2 ESE system looks like this:

```
inf93 onsoctcp viper.svl.ibm.com 1652
```

The meaning of the values in the four fields is:

- 1st field - dbservername: inf93.
  - This will be setting for the Node option in our federated Server definition for the Informix server.
- 2nd field - Informix protocol: onsoctcp
- 3rd field - hostname of the Informix server: viper.svl.ibm.com
- 4th field - TCP/IP port number or service name: 1652
  - If the value were a name instead of a number, there would need to be an entry in /etc/services here on the DB2 system to resolve this service name to the port number that the Informix server is listening on.

On Windows, Informix sqlhosts entries are recorded in the Windows Registry (**regedit**). Select
HKEY_LOCAL_MACHINE-->SOFTWARE-->Informix-->SQLHOSTS.

If there is not an entry yet for the Informix server, it can be added using Informix-Connect's setnet32.

If the service name and port for the Informix server needs to be added to the TCP/IP services file (such as because the SQLHOSTS entry has a service name instead of a port number), then Notepad or some other editor can be used to add the entry. On Windows, the TCP/IP services file is in the directory c:\Winnt\System32\Drivers\etc.

C.5 Preparing the DB2 instance for federated

Before adding federated objects for an Informix server to a DB2 ESE Version 8 database, we must check for and potentially do some configuration of the DB2 instance. We'll note which steps are required on UNIX and Windows respectively. More steps are required on UNIX. The steps are:

- check/set Informix variables in db2dj.ini file (UNX)
- check/set DB2 Profile (db2set) variable DB2_DJ_INI (UNIX)
- check/set System Enviornment Variables (UNIX and Windows)
- check/set DB2 Database Manager Configuration parameter FEDERATED (UNIX and Windows)

These steps are performed by the DB2 instance owner. In our example, that is 'aix43db2'.

As the instance owner, you might verify that the Informix wrapper library now appears in the instance's /sqllib/lib sub-directory since this library was created by

root using djxlinkInformix. **cd** to */home/aix43db2/sqllib/lib/* and use **ls** to find *libdb2informix.a*

db2dj.ini

db2dj.ini file contains the environment variables that non-DB2 data source client software requires. In otherwords, we need to specify in *db2dj.ini* the environment variables required by Informix Client SDK.

The *db2dj.ini* file is normally in the instance owner's /sqllib/cfg sub-directory.

For instance, in our example, */home/aix43db2/sqllib/cfg/db2dj.ini*.

If the file db2dj.ini is not in that directory, it can be created with an editor (vi).

The db2dj.ini variables required for use with Informix Client SDK are INFORMIXDIR and INFORMIXSERVER. Another variable - INFORMIXSQLHOSTS - is required if the Informix sqlhosts file is not in the directory INFORMIXDIR/etc.

- INFORMIXDIR - the main directory of Informix Client SDK here on the DB2 system.
- INFORMIXSERVER - the dbservername from one of the sqlhosts entries on the DB2 system
- INFORMIXSQLHOSTS - the path to the sqlhosts file if it is not in INFORMIXDIR/etc

In our example, db2dj.ini did not exist yet. We used vi to create it and made the following entries:

```
INFORMIXDIR=/home/informix
INFOMRIXSERVER=inf93
INFORMIXSQLHOSTS=/home/informix/etc
```

Note: we really didn't need to specify INFORMIXSQLHOSTS, since the *sqlhosts* file is in the usual directory.

Attention: do ***not*** use variable names in the values specified in db2dj.ini. This will cause unpredictable DB2 errors, including crashes. For instance:

```
INFORMIXSQLHOSTS=INFORMIXDIR/etc
```

could cause DB2 to *crash*.

Note: *db2dj.ini* is not used on Windows. On Windows, federated server uses Windows System Environment Variables.

DB2 Profile Registry (db2set) variables

On UNIX, the DB2_DJ_INI variable in the DB2 Profile Registry (db2set) must be set to indicate to DB2 the full path to the db2dj.ini file.

For instance:

```
db2set DB2_DJ_INI=/home/aix43db2/sqllib/cfg/db2dj.ini
```

Warning: it is important that the setting include the full path to the *db2dj.ini* file.

Note: DB2_DJ_INI is not used on Windows, since DB2 federated server on Windows does not use a *db2dj.ini* file.

There is an optional DB2 Profile Registry variable - DB2_DJ_COMM - that will tell DB2 to load wrapper libraries whenever DB2 is started. This way, the first time the wrapper is used, there won't be a delay while it is loaded into memory. Once a wrapper is loaded into memory, it remains there until DB2 is stopped.Here is an example to tell DB2 to load the Informix wrapper library into memory on AIX.

```
db2set DB2_DJ_COMM=libdb2informix.a
```

Note: db2set DB2_DJ_COMM can be used on Windows to load db2informix.dll whenever the DB2 service is started.

System Environment Variables

For some of the non-DB2 server source client software used with DB2 federated access, there is also the requirement that the client software's /bin directory be added to the PATH of the DB2 instance owner. But this is not true for Informix Client SDK on UNIX.

On Windows, the variables that were set in db2dj.ini on UNIX need to be set as System Environment variables. These are:

- INFORMIXDIR
- INFORMIXSERVER

On windows, the probable value for INFORMIXDIR will be *c:\Program Files\informix*

On windows, the value for INFORMIXSERVER should be found from the SQLHOSTS entries either in the Windows Registry (regedit) under

```
\HKEY_LOCAL_MACHINE\SOFTWARE\Informix\SQLHOSTS
```

...or in Informix Connect configuration tool SetNet32

Database Manager Configuration parameter FEDERATED

To configure and use federated database objects in a DB2 database, the Database Manager Configuration for the DB2 instance must have Federated =Yes. You can check the current setting either in the DB2 Control Center or using the **db2 get dbm cfg** command.

In Control Center, in the left window, go to the icon for the DB2 instance that will contain the database where the nicknames will be created for the Informix source or target tables. Highlight the icon, right-mouse click to see your options, and select 'Configure Parameters.' Look under 'Environment' parameters for FEDERATED. If the value is 'NO' it can be set to YES in the Control Center. DB2 will have to be stopped and started in the instance for the new setting to be in effect.

At command prompt, on UNIX you can use:

```
db2 get dbm cfg | grep FEDERATED
```

On Windows, in a DB2 Command Window, you can use

```
db2 get dbm cfg | more
```

FEDERATED is among the first 10 parameters in the output.

In our case, the result is:

```
Federated Database System Support (FEDERATED) = YES
```

If the value is 'NO' it can be set to YES with:

```
db2 update dbm cfg using federated yes
```

C.6 Configuring federated objects in a DB2 database

The federated database objects for access to Informix can be configured either in the DB2 Control Center or using SQL statements entered via DB2 CLP or in a DB2 script file. We will show both.

The federated database objects to be created in the DB2 database are:

- Wrapper - registers the Informix wrapper in the DB2 database so that it can be referenced in Server definitions for Informix servers.
- Server - registers a Server definition for each Informix server that is to be accessed. The Server definition declares
  - a Server name by which the data source will be referenced in the DB2 database.
  - the type and version of the federated Server,

  - the Wrapper to be used with this federated Server
  - options that specify connectivity information,
  - other options that may be required or could improve performance
    - for replicatoin to/from informix, the Server Option IUD_APP_SVPT_ENFORCE must be specified and set to 'N'

  The Server definition depends on a Wrapper already being defined for use with the specified Server Type

- User Mapping - registers a mapping of a userid that accesses the DB2 database to a userid at a federated Server. The User Mapping provides the Federated Server function with the userid to specify in the under-the-covers connections that federated server will make to an Informix server on behalf of a specific DB2 database user.
- Nicknames - registers a remote table or view in the DB2 database. A nickname is a two-part name adhering to the same naming rules as for tables and views in the DB2 database. The first part of the name is the schema. If when the nickname is created only *one* part is specified, that one part will be the second part of the two-part name and schema (the first part) for the nickname will be the default schema. The default schema is the userid itself that is creating the nickname. The nickname specification must include the remote schema and the remote name of the data source object for which it is created. Once created, the nickname can be referenced in Select, Insert, Update, and Delete statements made to the DB2 database, and Federated Server will make an under-the-covers connection to the data source to execute the appropriate action. Mutliple nicknames can be referenced in an SQL statement. SQL statements can also include local tables/views as well as nicknames. Federated Server can do joins between multiple data sources and between local DB2 data and remote data source data.

C.6.1 Creating federated objects using SQL statements

You must be connected to a DB2 database in an instance that has been configured for federated access.The statements that follow are described in the *DB2 UDB SQL Reference*.

Create Wrapper

Before creating the wrapper, you might check for the appropriate wrapper library. On AIX, this would be in the instance owner's /sqllib/lib directory and the library name is libdb2informix.a

On Windows, this would be in c:\Program Files\IBM\SQLLIB\bin and the library name is db2informix.dll

At a DB2 CLP prompt (db2=>) or in a DB2 script file:

```
CREATE WRAPPER INFORMIX
```

At a UNIX command prompt or in a Windows DB2 Command Window:

```
db2 create wrapper informix
```

Note: Wrappers, once they have been defined, can be found in the SYSCAT.WRAPPERS catalog view.

Create Server

Before defining a Server for an Informix data source we need:

- the INFORMIX wrapper to have already been defined.
- the dbservername for the Informix data source from the sqlhosts file here on the DB2 server. In our case, the value is 'inf93'
- the database name at the Informix server that contains the tables that will be replication sources or into which we will create replication target tables. In our case, that is 'stores_demo'

At DB2 CLP prompt (db2=>) or in DB2 script file:

```
CREATE SERVER IDS_VIPER
TYPE INFOMRIX VERSION 9.3 WRAPPER INFORMIX
OPTIONS
(NODE 'inf93',
DBNAME 'stores_demo',
IUD_APP_SVPT_ENFORCE 'N',
FOLD_ID 'N',
FOLD_PW 'N' )
```

At UNIX command prompt:

```
db2 "create server ids_viper type informix version 9.3 wrapper informix options
(node 'inf93',dbname 'stores_demo',iud_app_svpt_enforce 'N',fold_id 'N',fold_pw
'N' ) "
```

In Windows DB2 Comand Window:

```
db2 create server ids_viper type informix version 9.3 wrapper informix options
(node 'inf93',dbname 'stores_demo',iud_app_svpt_enforce 'N',fold_id 'N',fold_pw
'N' )
```

In this example:

- SERVER IDS_VIPER: the Server name we are making up to reference this Informix server
- TYPE INFOMRIX: the appropriate type for Informix
- VERSION 9.3 : the version of the Informix server
- WRAPPER INFORMIX: the wrapper we use with Informix
- NODE 'inf93': dbservername in sqlhosts for the Informix server. This value is case sensitive

- DBNAME ‘stores_demo’: database at the Informix server. This value is case sensitive.
- IUD_APP_SVPT_ENFORE ‘N’ : option we need to specify to enable insert/update/delete (i.e. replication) with Informix
- FOLD_ID/FOLD_PW ‘N’: these are optional. They tell federated server when it connects to this Informix server, attempt the connection only once and use the REMOTE_AUTHID/REMOTE_PASSWORD values of the User Mapping exactly as they are without folding them to either upper or lower case.

Note: Servers, once they have been defined, can be found in the SYSCAT.SERVERS catalog view. The options are in SYSCAT.SERVEROPTIONS.

Create User Mapping

User Mappings are always recommended even if the Informix server and DB2 are on the same system and the same userid can access both.

At a DB2 CLP prompt (db2=>) or in DB2 script file:

```
CREATE USER MAPPING FOR AIX43DB2 SERVER IDS_VIPER
OPTIONS (REMOTE_AUTHID ‘gjuner2, REMOTE_PASSWORD ‘gjunerpw’)
```

At UNIX prompt:

```
db2 “create user mapping for aix43db2 server ids_viper options (remote_authid
‘gjuner2’,remote_password ‘gjunerpw’ ) “
```

At Windows DB2 Command Windows prompt:

```
db2 create user mapping for aix43db2 server ids_viper options (remote_authid
‘gjuner2’,remote_password ‘gjunerpw’ )
```

In this example:

- aix43db2: the userid that connects to the DB2 database
- SERVER IDS_VIPER: the Server name for the specific Informix server
- REMOTE_AUTHID ‘gjuner2’: the userid at the Informix server
- REMOTE_PASSWORD ‘gjunerpw’: the password at the Informix server

Set Passthru

Before trying to create any nicknames, it would be a good idea to test the Server and User Mapping definitions. We can also verify the schema and name of an Informix table for which we will create a nickname.

At DB2 CLP prompt (db2=>)

```
set passthru ids_viper
```

  - Note: this statement does not exercise the Server defintion, it just tells DB2 to send the next statement to the specified server.

```
select count (*) from systables
```

– This statement causes DB2 to use the Server and User Mapping information to attempt a connection to the Informix server through the Infomrix Client SDK. If the connection is successful, DB2 sends the SQL statement. This statement queries a catalog table.

```
select count(*) from informix.customer
```

– This statement queries a table for which we wish to create a nickname. We specify both the schema and the table name in order to verify both parts of the name.

```
set passthru reset
```

– This statement returns us to the world of the DB2 database where Informix tables can only be accessed via nicknames.

Create Nickname

Create Nickname will cause DB2 to

- connect to the Informix server using the Server and User Mapping information
- query the Informix catalog for information about the Informix table or view for which are creating the nickname.
- insert records into the DB2 catalog for the nickname.

To create a nickname, we need both the schema and the table/view name of the objects at the Informix server. Also, if we will be doing Selects in the DB2 database referencing this nickname, performance will be better if DB2 has accurate statistics for the nickname. These are gathered from Informix when the nickname is created, if there are statistics. The Informix command to update the statistics in the Informix catalog for a table is ‘update statistics’. For instance:

```
update statistics for table informix.customer
```

The Informix update statistics command can be run in a DB2 Set Passthru session to an Informix server.

At a DB2 CLP prompt (db2=>) or in a DB2 script file:

```
CREATE NICKNAME IDS_VIPER.CUSTOMER FOR IDS_VIPER.”informix”.”customer”
```

At an AIX or Windows DB2 Command Windows prompt:

```
db2 create nickname ids_viper.customer for ids_viper.\”informix\”.\”customer\”
```

Note, in the example immediately above: the backslash (\) is used before the double-quotes (“) that surround the remote schema name and the remote table name to tell AIX and Windows not to do their normal operation that they do when they find a double-quote within a command. The backslash is not needed when the statement is entered in DB2 CLP session (db2=>) or in a DB2 script file

In this example

- IDS_VIPER (1st) will be the schema of the nickname
- CUSTOMER will be the nickname part of the nickname
- IDS_VIPER (2nd) is the Server name for the Informix server. The Node option of the Server definition points to an Informix server instance, and the DBNAME option points to a specific database within the Informix server instance.
- informix is the schema of the table for which we are creating the nickname.
  - the value is enclosed in double-quotes so that DB2 will not fold it to upper case before querying Informix about the remote table
- customer is the name of the table for which we are creating the nickname
  - the value is enclosed in double-quotes so that DB2 will not fold it to upper case before querying Informix about the remote table

Once a nickname has been created, there are records for the nickname in the DB2 Catalog. Look in the following Catalog Views:

- SYSCAT.TABLES - a record for the nickname. Type='N'
- SYSCAT.TABOPTIONS - information about the remote table. There are multiple records for each nickname.
- SYSCAT.COLUMNS - one record for each column of the nickname. Shows the DB2 data types of the columns.
- SYSCAT.TABOPTIONS - information about the columns of the remote table, including there data types. There are multiple records for each column.
- SYSCAT.INDEXES - information about indexes on the remote table, such as which columns are indexed. The DB2 optimizer uses this information to help choose the best plan for queries involving the nickname. There is not a real index in DB2 for the remote table.

C.6.2 Creating federated objects using the Control Center

We will now demonstrate the above steps, but this time using the Control Center.

Our workstation that is running the Control Center has already been configured to connect to the FED_DB database on the AIX server sthelens.almaden.ibm.com.

We open the *Control Center* and expand the tree in the left window to get the FED_DB database and choose the **Federated Database objects**. See Figure C-1.

Figure C-1 Control Center - accessing federated database objects

Create rapper

With the '**Federated Database Objects** icon highlighted, right-mouse click and select **Create Wrapper**.

The *Create Wrapper* dialog window opens.

Select **INFORMIX** in the pulldown window. See Figure C-2

Figure C-2 Control Center - Create Wrapper window

When you select INFORMIX as the wrapper name, the Library name should automatically be filled in.

Click **OK** to create the wrapper.

In the *Control Center*'s right window, the INFORMIX wrapper should appear ,and also it should be an object under Federated Database Objects in the tree in the left window.

Create Server

In the *Control Center*'s tree in the left window, highlight the *INFORMIX* wrapper icon and, if necessary, click the '+' to expand the tree below.

Highlight the ***Server*** icon and right mouse click and then select **Create** from among the options to open the *Create Server* dialog window.

Key in the *Server Name.*

Select the *Server Type* from the pull down list

Key in the *Version* of the Informix server

For the Node, key in the *dbservername* value from the appropirate *sqlhosts* entry.

For the *Database*, key in the name of the database at the Informix server.

When filled in the the *Create Server* window should appear as in Figure C-3.

Figure C-3 Control Center - Create Server window

Don’t click OK yet.

First, click ***Options,*** to open the Server *Options* window.

Check and highlight each of the options one at a time:

- **IUD_APP_SVPT_ENFORCE** - change from Y to **N**
- **FOLD_ID** - change from U to **N**
- **FOLD_PW** - change from U to **N**

See Figure C-4.

Figure C-4 Control Center - Server Options window

Now click **OK** on the *Options* window.

Then **OK** on the *Create Server* window.

An icon for the new server definion should now appear under the Server icon under the INFORMIX icon in the tree in the Control Center's left window.

Create User Mapping

Click the '+' sign next to the icon for the server just created in the Control Center's left window. Icons for User Mappings, Nicknames, and Remote Tables should appear. Highlight the **User Mappings** icon, right-mouse click, and select **Create** from the options. The *Create User Mappings* window should open.

In the left window, select your own *userid* that you are using on your current connection to the DB2 database and then the '**>**' button in the middle of the window to put this userid in the right window.

In the *Remote Userid* fields below, put the Userid and Password that can access the Informix server. See Figure C-5

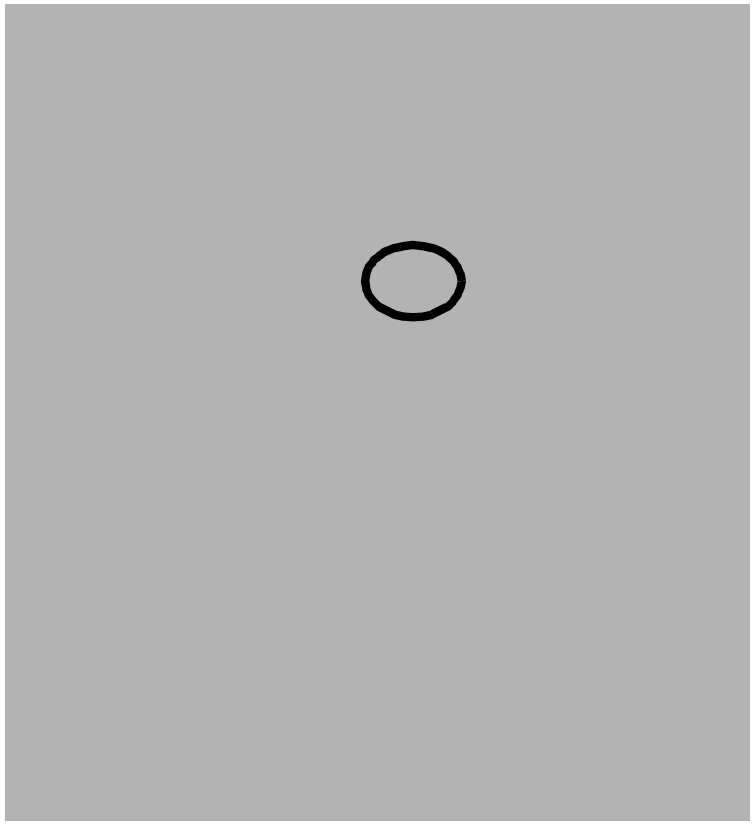


Figure C-5 Control Center - Create User Mapping window

Now click **OK** to create the User Mapping.

The new User Mapping should appear as an object in the Control Center's right window.

If your password changes at the Informix server, you can find this User Mapping object in the Control Center, highlight it, right mouse click, and select **Alter** to open the *Alter User Mapping* window where you can change the *Remote_Password*.

Testing the Server/User Mapping

The above steps have only made entries into the DB2 catalog. None of the defintions have been tested. To test these definitions, you could go to a DB2 Command Line Processor Window (db2=>), connect to the DB2 database

(FED_DB) and use the commands in the ‘SQL Statements’ section above for Set Passthru to test the definitions.

You can also do this test here in the *Control Center* using the *Create Nickname* dialog.

Under the icon for the Server, highlight the **Nicknames** icon, right mouse , and select **Create**. The *Filter Table for Nicknames* window should open.

Check the box for **Remote Table Name**, be sure the operator is ‘**=**’, and in the values field put the name of a known table. For Informix, we’ll use ‘systables’ which is one of the Informix catalog tables. Then click the **Count** button. Under the covers, federated server will use the information in our Server and User Mapping definitions to connect to the Informix server and query Informix for the number of tables whose name is ‘systables.’ We should get a response in the window, ‘Number of objects meeting this filter criteria: 1.’ See Figure C-6.

Figure C-6 Control Center - server connection test with nickname filter

Create Nickname

With the Control Center we can create one or many nicknames at once.

As you will see, the *Create Nickname* dialog can build a list of remote tables pre-checked to have nicknames created for them. We can if we want customize the nickname schema for one or all of the nicknames that will be created at once.

To get to that point, we need to go through the *Filter Tables for Nicknames* dialog that we demonstrate above in the connection test to the Informix server.

In the *Filter Table for Nicknames*, we can use the different operators in the middle of the window with different values in the fields on the right side of the window to get various lists of remote tables from the Informix server. If you use the **LIKE** operator, put the percent sign (%) before or after the value specified to get all remote tables that meet the criteria.

In our example here we just want to create a nickname for some of the tables in the ‘stores_demo’ database. In the filter, we fill in the schema of the tables of the stores_demo database. In this case it is ‘informix’ and we use the ‘=’ operator and click OK.

The *Create Nicknames* window opens with the list of remote tables whose schema is ‘informix.’ In this case, that includes both the stores_demo tables and the catalog tables.

You may need to re-size the window to see all the fields. They are:

- Create check box, with check mark as the default setting.
- Nickname - two part name for the nickname that will be created
- Local Schema - the nickname’s schema
- Remote Schema - the schema of the table in the Informix database
- Remote Name - the name of the table in the Informix database.

In our case, before proceeding, we want a remote table list that excludes the Informix catalog tables. We go back to the filter. For the *Remote Table Name*, we select the NOT LIKE operator and put ‘sys%’ in the Values field. Then we click ‘**OK**’. The Create Nickname window now has only ‘user data’ tables and not the Informix catalog tables. The *Create Nickname* window now appears as in Figure C-7.

Figure C-7 Control Center - Create Nicknames - defaults

We want to:

- create nicknames only for the ‘customer’ and ‘orders’ tables
- we want the nickname schema to be the same as our Server name
- the nickname ‘name’ can be the same as the remote table name, but it should be upper case.

To accomplish this, we uncheck the check box under *Create* for all but the ‘customer’ and ‘orders’ table. Then we click the **Change All** button. This opens the *Change All Nicknames* window.

In the *Schema* field, we select **Custom** from the pulldown. In the field that now becomse available we key the Server name: ‘IDS_VIPER’.

We note that in the *Nickname* field the value is **Remote table name**

We then go to the *Fold* field in the lower left, and select **Upper Case** from the pulldown.

We click **OK** on the *Change All Nicknames* window. Now our *Create Nicknames* window looks like in Figure C-8.

Figure C-8 Control Center - Create Nicknames with custom settings

We see that only the nicknames we want are checked, and they have the schema and upper-case folding we wanted. We can click **Show SQL** to see the SQL that would be used to create the nickname. We click **OK** to create the two nicknames.

The nicknames now appear in the Conrol Center's right window. See Figure C-9.

Figure C-9 Control Center with nicknames

If we higlight one of the nicknames and right mouse click, we can see our options include **Sample Contents.** We can, if we want, see a few records from the table in Informix.

D

Additional material

This redbook refers to additional material that can be downloaded from the Internet as described below.

Locating the Web material

The Web material associated with this redbook is available in softcopy on the Internet from the IBM Redbooks Web server. Point your Web browser to:

ftp://www.redbooks.ibm.com/redbooks/SG24????

Alternatively, you can go to the IBM Redbooks Web site at:

ibm.com/redbooks

Select the **Additional materials** and open the directory that corresponds with the redbook form number, SG24????.

Using the Web material

The additional Web material that accompanies this redbook includes the following files:

File name	*Description*
????????.zip	????Zipped Code Samples????

????????.zip	????Zipped HTML Documents????
????????.zip	????Zipped Presentations????

System requirements for downloading the Web material

The following system configuration is recommended:

Hard disk space:	????MB minimum????
Operating System:	????Windows/UNIX????
Processor:	???? or higher????
Memory:	????MB????

How to use the Web material

Create a subdirectory (folder) on your workstation, and unzip the contents of the Web material zip file into this folder.

Glossary

Your glossary term, acronym or abbreviation. Term definition. Sort terms: **highlight rows>Table>Sort>Column1>Sort**

Term1. Term1 definition.

Term2. Term2 definition.

Term3. Term3 definition.

Related publications

The publications listed in this section are considered particularly suitable for a more detailed discussion of the topics covered in this redbook.

IBM Redbooks

For information on ordering these publications, see “How to get IBM Redbooks” on page dlxviii.

- *????full title???????,* xxxx-xxxx
- *????full title???????,* xxxx-xxxx
- *????full title???????,* xxxx-xxxx

Other resources

These publications are also relevant as further information sources:

- *Administration Made Easier: New and Improved Tools in DB2 Universal Database*, by Jason Gartner found at http://www7b.software.ibm.com/dmdd/library/techarticle/0207gartner/0207gartner.html
- IBM DB2 Universal Database Administration Guide: Planning, Version 8, SC09-4822-00
- IBM DB2 Universal Database Command Reference, Version 8, SC09-4828-00
- IBM DB2 Universal Database Message Reference Volume 1, Version 8, GC09-4840-00
- IBM DB2 Universal Database Replication Guide And Reference, Version 8, SC27-1121-00
- IBM DB2 Universal Database SQL Reference Volume 2, Version 8, SC09-4845-00
- IBM DB2 Universal Database System Monitor Guide and Reference, Version 8, SC09-4847-00.
- z/OS V1R4.0 MVS System Commands, SA22-7627-04
- *????full title???????,* xxxx-xxxx

Referenced Web sites

These Web sites are also relevant as further information sources:

- DB2 Replication home page
- http://www.software.ibm.com/data/dpropr
- DB2 Replication Support home page

  http://www.software.ibm.com/data/dpropr/support.html
- DB2 home page

  http://www.software.ibm.com/data/db2
- DB2 Support home page

  http://www.software.ibm.com/data/support
- DB2 for Linux, UNIX, and Windows Support home page

  http://www.software.ibm.com/data/db2/udb/winos2unix/support
- Informix home page

  http://www.software.ibm.com/data/informix
- DB2 Spatial Extender home page

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

How to get IBM Redbooks

You can order hardcopy Redbooks, as well as view, download, or search for Redbooks at the following Web site:

ibm.com/redbooks

You can also download additional materials (code samples or diskette/CD-ROM images) from that site.

IBM Redbooks collections

Redbooks are also available on CD-ROMs. Click the CD-ROMs button on the Redbooks Web site for information about all the CD-ROMs offered, as well as updates and formats.

Index

Symbols

.IBMSNAP_REGISTER cdlvi
.profile cv

Numerics

3270 terminal emulator ciii
5250 terminal emulator ciii

A

Activate ccxv
Add ccxxiii, cclix, cclxiii
add a Capture or Apply Control Server cxxx
Add a Database wizard dx
Adding database cxxxviii
Adding new Subscriptions Sets ccclxxxi
Administration Client lxxxvi, xciii, xcvii–xcviii
Administration Client, lxxvii
Administration Client. xciv
Administration Tools xcviii
Advanced Replication Topics cdiii
AIX lxxxvii
alert conditions lxvii, cccliii
Alert Monitor xxxvi
Allow ccxv
ANZDPR ccclxv
APP.log cccxliii
Application Programming Defaults cdlxxi
Application Requestor cccviii
Apply xxxvi, ccxiii–ccxiv, dxxxviii
apply control server liv
Apply Parameters cccxxvii
Apply parameters cccxiv
Apply Performance cdlxviii
apply qualifier lv
Apply Report cccxlvi, cccxlviii
Apply Throughput cdlxxxii
Apply throughput cdxcvii
Apply Transformations cdxiv
APPLY_PATH cccxliii
Apply_Path cdlxxiii
apply_path cclxxv
APPLY_QUAL cdlxxxii
apply_qual cclxxiv
Apply's sub-operations cdlxxxiii
AS/400 ciii, dvii
ASN.IBMSNAP_APPARMS cdlxxix
ASN.IBMSNAP_APPLYTRAIL cdlxxxii
ASN.IBMSNAP_APPPARMS cdxci
ASN.IBMSNAP_SUBS_SET cdlxxvi, cdlxxx, cdxci
ASN1560E cccxlix
asnacmd ccxciii, cccxlix
asnanalyze cccxxiv, cccxxxix, cccxliii–cccxliv, ccclxv
ASNAPLDD cdlxxiii
asnapply ccxciii, cccxxxix
asncap cclxxxix, cccxxxix
asncap and asnccmd cclxxxix
asnccmd cclxxxix, cccxlix
ASNCLP lxxix
ASNLOAD lx, cccxxxi, cdlxxxii
asnmcmd ccclv
ASNMON ccclxiv
asnpwd cclxxv, ccclix, ccclxvi
asntrc cccxxxix, cccxliv, ccclxx, cdlxxxiii
asntrc example ccclxx
Asynchronous Read Log API cdxlix
Automatic full refresh cdi
Automatic Restart Manager (ARM) lxxi
AUTOPRUNE cdlx

B

Base ccxxiv
Base aggregate lvii
Bidirectional exchange of data xxxiii
BIND PACKAGE cdlxxi
Binding Package dxxxii
BLOCKING cdlxx
blocksize cdlxxii
Boolean cccxxxix
BPXBATCH cccx
bufferpool cdxcv
BUFFPAGE cdxcvi
Bypassing the full refresh cdii

C

CACHE DYNAMIC SQL cdlxxi
caching dynamic SQL cdlxxi, cdlxxv
CAPPARMS cdl
CAPSPILL cdliv
Capture xxxvi, ccxiv–ccxv
capture control server liii
Capture Latency cdlxii
Capture Messages cccxlvi
Capture parameters cccxiii, cccxxvi
capture schema xxxix, cclxviii
Capture spill cdli, cdliv
Capture Throughput cdlxiv
Capture throughput cdxcvi
Capture transformations cdx
Capture triggers cdxlviii
Capture updates as pairs of deletes and inserts cdlviii
CAPTURE_MEMORY cdliii
CAPTURE_PATH cccxliii
capture_path cclxix
capture_server cclxx
catalog db dxii
catalog dcs db dxii
catalog system odbc data source dxii
catalog tcpip node dxii
CCD xli, ccxlii, ccxlvi
CCSID dxviii
CD table xxxviii
CD tables cdxlix, cdlv, cdlix, cdlxv, cdlxviii, cdxcii, cdxcv
CD_NEW_SYNCHPOINT cdxcii
CD_ROWS_INSERTED cdlxiii
Change ccxlii
Change aggregate lvii
Change Capture parameters cccxxvi
Change Data xxxviii
changes from replica target tables cdlvii
CHG_ROWS_SKIPPED cdlxiv
CHG_UPD_TO_DEL_INS cdlvii
CHGDDMTCPA dxviii
CHGONLY cdlvi
CHGUSRPRF dxviii
CL commands ciii
CLI dxxxiii
Client Access xciii, ciii
CLIST CALCULATIONS cdlxxi
Collection NULLID dxviii
Command Center ciii, cvi, cxi
Command Line Processor c, ciii
COMMIT_COUNT cdlii, cdlxxiii, cdlxxvi
Commit_Count cdlxxvii
COMMIT_INTERVAL cdlviii
Commit_Interval cdxcii, cdxcvii
comp.databases.ibm-db2 cccxlvi
Complete ccxliv–ccxlv
Condense ccxliv–ccxlv
Configuration Assistant c, dx
connectivity of Apply cccviii
consistent change data xli
Consolidation of data from remote systems xxxii
contact or contact group lxviii
Continuously ccxxxvii
Control Center lxxvi, ciii, cvi, cxi, dxlvii, dlii
Control Table Profile cxxix
Control Tables cxvii, cxxix
control tables xxxvi
Control Tables Profile cxv
control_server cclxxiv
CPU lxxxvi, cdlxvi
Create cclix
Create Monitor Control cccli
Create Nickname dli, dlviii
Create Server dxlix, dliv
Create User Mapping dl, dlvi
Create Wrapper dxlviii, dliii
CRTLIB dxviii
CURRENT_MEMORY cdlii

D

DAS xlvii
DASe lxxx, cdliii
Data ccxv
Data blocking factor cdlxxvii
data sharing cdl–cdli
Data Source dxi
Database dxi
Database Administration Server xlvii
Database Manager Configuration dxlvii
DataJoiner xc
DataJoiner Replication Administration lxxvi
DataJoiner Version 2 lxxxiii
DB2 Administration Client cccxxxix
DB2 Administration Server lxxxix, cdliii
DB2 Administration Tools ciii
DB2 Administrative Server lxxx
DB2 Connect xc, dvii, dxiii, dxxxviii

DB2 Connect EE lxxxix
DB2 Connect Enterprise Edition xciii–xciv
DB2 Connect Personal Edition lxxx, xciii–xcv, xcviii
DB2 Customer Support cccxlv–cccxlvi
DB2 Customer Support Analyst cccxliii
DB2 Database Directory dxiii
DB2 ESE lxxxix–xc, xciii, xcviii, dvii, dxiii, dxxx-vii–dxxxviii
DB2 for OS/390 xciii
DB2 for z/OS xciii
DB2 for z/OS or OS/390 ciii
db2 get dbm cfg dxiii
DB2 instance owner cv
DB2 LOB replication cdxvi
DB2 log cdxlix, cdlxxiv
DB2 Peer to peer replication cdxxix
DB2 Personal Edition xcviii
DB2 Profile Registry lxxxix, cdxcvi, dxlvi
DB2 Replication V8 close up liii
DB2 Runtime Client lxxx
DB2 Runtime client xciii
DB2 trace cxxxix
DB2 UDB Developers Edition xciv
DB2 UDB Enterprise Server Edition xciv
DB2 UDB Personal Edition xciv
DB2 Universal Database for Linux, UNIX, Windows xciii
DB2 V8 replication from 30,000 feet xxxvi
DB2/400 xciii
DB2_DJ_COMM dxlvi
DB2_DJ_INI dxlvi
DB2COMM lxxxix
db2diag.log cccxliii
db2dj.ini dxlv
DB2INSTANCE cv
db2mag cccxlvi
db2profile cv
db2rc cv
db2repl.prf cxiii
db2set lxxxix, cdxcvi, dxlvi
db2setup xcviii
db2support cccxlv
db2trc cxxxix, cccxliii–cccxliv
DBHEAP cdxcvi
DBNAME dl
DBSERVERALIASES dxxxix
DBSERVERNAME dxxxix
dbservername dxl, dxlix
DCS Options dxi
DDF lxxxix
DDM lxxxix
Deactivating and activating subscriptions ccclxxxi
Define ccix, ccxliv
DELAY cdlxxix, cdxci
Delay cdxcvii
Dependent Targets cxxxiii
DIAGLEVEL cccxliii
DIAGPATH cccxliii
Dirty Read cdlxix, cdlxxv
DIS DDF dix
disk lxxxvi
Distributed Data Facility lxxxix, cdlxxii, dix
Distribution of data to other locations xxxi
DJRA lxxvi
djxlink xcii, dxlii
djxlinkInformix dxliii
Downloads xciv
DRDA-TCP/IP listener port dxvii
DSNJU004 dix
DSNL004I dix
DSNTIP4 cdlxxi
DSNTIP5 cdlxxii
DSNTIPC cdlxxi
DSNTIPM dix
dynamic SQL cdlxx, cdlxxv

E

EDM Pool cdlxxi
EFFECTIVE_MEMBERS cdlxxxii
email ccclv
enable archival logging ccci
END_TIME cdlxxxii
End-to-End Latency cdlxxxi
End-to-end latency cdlxxx, cdxcvii
etc/services dxli
Event ccxxxvii
extentsiz cdxcv
External ccxlv
Extra Blocks Req cdlxxii

F

FEDERATED dxlvii
federated liii
federated database objects dxlvii
Federated server cdxciii
federated server lxxvii, lxxxiii, xc, cdxlvii, cdlxviii, cdlxx, dvii

fetch cdxlii, cdlxviii
filters cxxxiii
fixpacks xciv
FOLD_ID dl, dlv
FOLD_PW dl, dlv
full refresh xlii
Full refresh procedures cdi
FULL_REFRESH cdlxxxii
full-refresh ccclviii

G

GLOBAL_RECORD cdl, cdlxxx, cdxci
GROUP ccxxiv
Grouping ccvi

H

Hardware requirements lxxxvi
HFS cccvi
hostname dix
HP 9000 lxxxvi
HP-UX lxxxvii, xcvii

I

IBM Customer Support cccxliii
IBM Personal Communications ciii
IBM z/Series lxxxvi
IBMSNAP_ALERTS cccli, ccclxi
IBMSNAP_APPPARMS cccxxv, cdlxxii
IBMSNAP_APPTRACE cccxlviii
IBMSNAP_CAPMON cdlii, cdlxi
IBMSNAP_CAPPARMS cccxxv, cdli, cdlx
IBMSNAP_CAPTRACE cccxlvi, cdlxi
IBMSNAP_COMMITSEQ cdlx, cdlxix, cdlxxiii, cdlxxviii, cdxcii
IBMSNAP_CONDITIONS cccli
IBMSNAP_CONTACTGRP cccli
IBMSNAP_GROUPS ccclii
IBMSNAP_INTENTSEQ cdlxix, cdlxxiii
IBMSNAP_MONENQ ccclii
IBMSNAP_MONPARMS ccclii
IBMSNAP_MONSERVERS ccclii
IBMSNAP_MONTRACE cccli–ccclii
IBMSNAP_MONTRAIL cccli–ccclii
IBMSNAP_OPERATION cdlxxiii
IBMSNAP_REG_SYNCH cdxciii
IBMSNAP_REGISTER cdl, cdlxviii, cdlxxx, cdxci
IBMSNAP_UOW cdlv, cdlxv, cdxcv
I-Connect dxxxviii
IFI cdxlix
infopops cccxl
Information Center cxi
Information center cccxxxix
Informix xxx, lxxv, lxxvii, lxxxiii–lxxxiv, lxxxix, cxxxi, cdxlvi, cdlxviii, cdlxx, cdlxxiv, cdxciii, dxxxvii
Informix Client SDK xc–xci, dxxxviii, dxlii
Informix LOB replication cdxvii
INFORMIXDIR dxxxix, dxlii, dxlv
INFORMIXSERVER dxxxix, dxlv
INFORMIXSQLHOSTS dxxxix, dxlv
Internal ccxlv–ccxlvi, ccxlviii
Introduction to DB2 Replication V8 xxix
IP address dix
IRWW cdxciv
iSerie ciii
iSeries lxxxii, lxxxix, xciii, cccl, cdlxxiii, dvii
isolation levels cccvii
IUD_APP_SVPT_ENFORCE xcii, dxlviii, dlv
IUD_APP_SVPT_ENFORE dl

J

Java cccxxxix
Java Runtime Environment lxxxvii
Javascript cccxxxix
JavaToolbox/400 lxxxii
JCL to operate Capture and Apply cccx
JDBC lxxix, dxxxiii
joblog cccxlvi
JOIN_UOW_CD lxiii
Journal cdxlii
journal cdxlix
Journal Receivers cdli
Journals cdli
JRE lxxxvii

K

KEEPDYNAMIC(YES) cdlxxi, cdlxxv

L

LASTRUN cdlxxxii
latency cdxxix, cdlxi
Launchpad cvii, cxxxv
LDAP lxxxvii
Linux lxxxvi–lxxxvii, lxxxix, xcvii, ciii, cv, dvii
Linux for z/Series lxxxvii

Local ccxliv
LOCATION dix
Location ccxlv
Locking level cdlx
LOCKLIST cdlxi, cdlxxvi, cdxcv–cdxcvi
Locksize cdlx
locksize cdlxxvi
Log cdxlii
log record sequence number lix
Log Sequence Number cdxci
LOGBUFSZ cdxlix, cdxcvi
log-merge cdli
low-latency cdxci
LSN lix, cdxci

M

Maintaining capture and apply control servers Maintenance of a replication control server involves the followi cccxcv
Maintaining Registrations ccclxxiv
Maintaining Subscriptions ccclxxxi
Maintaining Your Replication Environment ccclxxiii
Managing DB2 logs and journals used by Capture cd
Manual full refresh cdii
Manually pruning replication control tables cccxcvi
MAX_SYNCH_MINUTES cdlii
Max_Synch_Minutes cdlxxvii
MAX_TRANS_SIZE cdlii
Member ccvii
memory lxxxvi
MEMORY_LIMIT cdliii
Monitor Condition cdliii
monitor control tables lxvii
monitor qualifier lxvii
Monitor Server xlvi
MONITOR_INTERVAL cdliii
Monitor_Interval ccclix
MONITOR_TIME cdlii

N

Named Pipes lxxxvii
NetBIOS lxxxvii
network cdxliii, cdlxxii
network adapter lxxxvi
network packet size cdlxxii
newsgroup cccxlvi
Nickname xcii
Nicknames dxlviii
nicknames lxxxiv, cdlxx
NODE dxlix
Node Options dxi
non ccxviii
non-DB2 cdxlvii
non-DB2 server lxxxiii, xc, cxxxi, cdxlvi, cdlxviii, cdlxx, cdlxxiv, cdxciii, dxlvi
NULLID dxxxii
Number of transactions applied at the target cdlxxviii

O

ODBC dxxxiii
olsoctcp dxli
ONCONFIG dxxxix
onconfig xc
onsoctcp dxl
Operating cxxxviii
Operating System Type cxxxviii
Operational Navigator ciii
OPT4ONE cdlxxix
Oracle xc
ORDER BY cdlxix
OS/390 lxxxix, dvii
OSTYPE cxxxviii
OUTBUFF cdxlix
Overview of the IBM replication solution xxx

P

package NULLID dxxxii
packages cccviii
page fetches cdlxvi
pagesize cdxcv
Passwords cxiii
PATH cv, dxlvi
PCKCACHESZ cdlxxi, cdxcvi
PDF cccxxxix
Peer-to-peer xl
plans cccviii
Point ccxli
Point-in-time lvi
prefetchsiz cdxcv
prefixDIS DDF dix
procedures lxxxiv, xc, dxxxviii
process cccxx
profile cxiii
Promoting a replication configuration to another sys-

tem cccxcii
Protocol dxi
prune cccxxix
PRUNE_INTERVAL cdlx
Pruning cdlix
pruning lv, cdxlix
Pull cdxlii
pull cdlxviii
Push cdxliii
push cdlxviii
Putting the pieces together xlviii

Q

Query Status cclxxx, cclxxxii, cccxlvi
Querying target tables cdlxxv
QZSNDPR cccxlvi

R

Rebinding replication packages and plans cccxcviii
RECAP_ROWS_SKIPPED cdlxiv
RECAPTURE cdlvi
Recovering source tables, replication tables, or target tables cccxcix
Red Hat Linux lxxxvii
Redbook environment lxxiii
Redbooks Web site dlxviii
Contact us xxviii
Referential Integrity cdlxxvii
REFRESH_TYPE cdxci
regedit dxli
REGION cdliii
Registered ccxvii–ccxviii
Registered Nicknames cxxxiii
Registered Tables cxxxii
Registered Views cxxxii
registering a source table liv
reinit cccxxx
Reinitialize Monitor ccclv
Relational Connect xc
Relational Database dxvii
Relative ccxxxvii
Remote ccxviii
remote journaling xxxix
remote journalling cdxliv, cdlxxiii
Remote Location dxvii
REMOTE_AUTHID dl
REMOTE_PASSWORD dl
REORG for replication tables cccxcviii
Replica lvii, ccxlix
Replicating column subsets cdiv
Replicating row subsets cdvi
Replication Center xxxvi, lxxv, dvii
Replication Center Profile cxiii, cxxxviii
Replication Center profile lxxxii, cxiii
Replication Center Profile. cxiii
Replication Center tracing cxxxviii
Replication filtering cdiv
Replication of DB2 Spatial Extender data cdxix
Replication of large objects cdxvi
replication source clxx
Replication transformations cdx
resume cccxxxi
Rework lxiii
rework cdlxxiv
RISC System/6000 lxxxvi
root xcviii, dxliii
row filters lvi
Row-capture rule cdlvi
Run now or Save SQL cii
RunOnce=Y ccclix
RUNSTATS cdlxvi, cdlxxv, cdxcv
RUNSTATS for replication tables cccxcvii
Runtime Client xciv

S

SAMPLE cxxxvi
Sample Contents dlxii
Schedule ccxxxvi
SDSF cccl
SecureWay lxxxvii
Security Options dxi
Server xcii, dxlvii
Server Option xcii, dxlviii
Server Options dlv
Service name dxiii
Set ccxiii–ccxiv
Set Passthru lxxxiv, xcii, dl
SET_DELETED cdlxxxii
SET_INSERTED cdlxxxii
SET_NAME cdlxxxii
SET_REWORKED cdlxxxii
SET_UPDATED cdlxxxii
setnet32 dxliv
Show ccxxx
Show Alerts ccclv, ccclxi
Show alerts ccclxi

SIGNAL lxi
SLEEP_INTERVAL cdl
SLEEP_MINUTES cdxci
Sleep_Minutes cdlxxvi, cdxcvii
Software Requirements lxxxvi
Solaris lxxxvii, xcvii
Source ccxvii
Source Object Profiles cxv
spill file lxiii, cdxlii, cdlxviii, cdlxxii
SPUFI ciii
SQL file lxxxii
SQL Server xc
SQL0100W cccxlix
sqlhosts xc–xci, dxxxix, dxliii, dxlix
ssid cccl
ssidMSTR cccl
staging tables cdxlii
start Apply cclxxii
Start Capture cclxvii
Start Monitor ccclv
startmode cclxix
Statements ccxxxviii
states of Apply threads cclxxxii
status of the Capture threads cclxxxi
status of the threads cclxxxi
Stop Monitor ccclv
STRTCPSVR dxviii
Subscription cciv–ccvi, ccviii, ccx, ccxii
subscription members xlii
subscription set xlii
Subscription Set parameters cdlxxvi
Subscription Sets cxxxiii
subsystem cccxlvi
Sun Solaris SPARC lxxxvi
Support cccxlv
SuSE Linux lxxxvii
suspend cccxxxi
suspend and resume the operations of Capture cccxxxi
SVCENAME dxiii
Sybase xc
SYNCHPOINT lix, cdxci
SYNCHTIME cdl, cdlxxx
SYSCAT.COLUMNS dlii
SYSCAT.INDEXES dlii
SYSCAT.SERVEROPTIONS dl
SYSCAT.SERVERS dl
SYSCAT.TABLES dlii
SYSCAT.TABOPTIONS dlii
SYSCAT.WRAPPERS dxlix
system design cdxlii
System Display and Search Fascility cccl
System Environment Variables dxlvi
System kernel parameters lxxxviii, xcvii
System Options dxi
System Services Address Space cccl

T

tablespace cdxcv
Target ccxv, ccxviii
Target Object Profiles cxvi
target server liv
Target table cdlxxiv
target table cdxlii, cdlxviii
TCP/IP lxxxvii, dix
TCP/IP Service Name lxxxix
TCP/IP services dxxviii
TCPPORT dix
Teradata xc
Testing Connections dxxxii
Testing the Server/User Mapping dlvii
thread cccxx
Time ccxxxvii
Tools Settings cxi
TRACE PERFORMANCE RECORD cdlxxxiii
TRACE_LIMIT cccxlvi
tracing cxxxix
TRANS_PROCESSED cdlii
TRANS_SPILLED cdlii
transational replication cdlii
trigger cdxciii
triggers lxxxiv, xc, dxxxviii
TRIGR_ROWS_SKIPPED cdlxiv
Types ccxlv

U

Uncommited Read cdlxix
Uncommitted Read cdlxxv
UNI cv
Unit of Work xxxviii
UNIX lxxxix, xcviii, ciii, dvii
Unix System Services cccvi
UOW_CD_PREDICATES lviii
Update Anywhere xl
Update anywhere replication cdxxii
update statistics dli
UR cdlxix, cdlxxv

USENET cccxlvi
User ccxli
User copy lvi
User Mapping xcii, dxlviii
Userid and password prompts cxxxviii

V

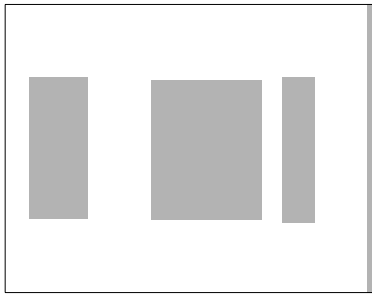
VIO cdli, cdliv, cdlxxiii
Virtual I/O cdli

W

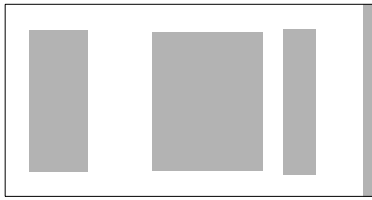
web browser cccxxxix, cccxliii
What's new in DB2 Replication V8 lxix
WHOS_ON_FIRST cdlxxxii
Why use replication? xxxi
Windows lxxxvi, lxxxix, xcix, ciii, cv, dvii
Windows 2000 lxxxvii
Windows 98 lxxxvii
Windows Explorer civ
Windows ME lxxxvii
Windows Registry xcix, dxli
Windows service ccciv
Windows XP lxxxvii
Windows.NET lxxxvii
Windows/NT lxxxvii
Wrapper xcii, dxlvii
WRKACTJOB dxviii
WRKDPRTRC cccxliv
WRKJOB cccxlvi
WRKRDBDIRE dxvii
WRKSBMJOB cccxlvi
WRKSBSJOB cccxlvi
WRTTHRSH cdxlix

Z

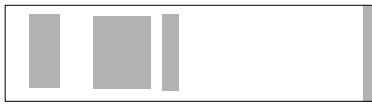
z/OS lxxxix, cccl, dvii



(1.5" spine)
1.5"<-> 1.998"
789 <->1051 pages



(1.0" spine)
0.875"<->1.498"
460 <-> 788 pages



(0.5" spine)
0.475"<->0.875"
250 <-> 459 pages

(0.2"spine)
0.17"<->0.473"
90<->249 pages

(0.1"spine)
0.1"<->0.169"
53<->89 pages

(2.5” spine)
2.5”<->nnn.n”
1315<-> nnnn pages

(2.0” spine)
2.0” <-> 2.498”
1052 <-> 1314 pages

Draft Document for Review September 16, 2002 4:39 pm

IBM Data Replication V8

What is new for version 8

Develop on How to use new Replication Center capabilities in V8 and to integrate relational connect

Covers Windows, UNIX, zSeries, iSeries

All you need to know IBM Data Replication V8

INTERNATIONAL TECHNICAL SUPPORT ORGANIZATION

BUILDING TECHNICAL INFORMATION BASED ON PRACTICAL EXPERIENCE

IBM Redbooks are developed by the IBM International Technical Support Organization. Experts from IBM, Customers and Partners from around the world create timely technical information based on realistic scenarios. Specific recommendations are provided to help you implement IT solutions more effectively in your environment.

For more information:
ibm.com/redbooks

SG24-6828-00 ISBN