HP NonStop SQL DDL Replicator User's Guide

Abstract

HP NonStop™ SQL DDL Replicator Software replicates NonStop SQL DDL operations to one or more backup systems.

Product Version

NonStop SQL DDL Replicator AAC

Supported Release Version Updates (RVUs)

N/A

Part Number	Published
545799-003	June 2010

Document History

Part Number	Product Version	Published
545799-001	NonStop SQL DDL Replicator 1.0	October 2008
545799-002	NonStop SQL DDL Replicator AAB	January 2010
545799-003	NonStop SQL DDL Replicator AAC	June 2010

Legal Notices

Printed in the US

What's New in This Manual

Manual Information

Abstract

HP NonStop™ SQL DDL Replicator Software replicates NonStop SQL DDL operations to one or more backup systems.

Product Version

NonStop SQL DDL Replicator AAC

Supported Release Version Updates (RVUs)

N/A

Part Number	Published
545799-003	June 2010

Document History

Part Number	Product Version	Published
545799-001	NonStop SQL DDL Replicator 1.0	October 2008
545799-002	NonStop SQL DDL Replicator AAB	January 2010
545799-003	NonStop SQL DDL Replicator AAC	June 2010

New and Changed Information

Note that SPR AAC, while classified as a Time Critical Fix, is a major upgrade to AAB and contains new configuration options that obsolete prior configuration options.

- A new appendix has been added how to describe SDR testing configurations and options. See Appendix C, Testing SDR for details.
- The global parameters EXTRACT and REPLICATION have been replaced by the new parameter DDLCAPTURE. See DDLCAPTURE { ENABLE[D] | DISABLE[D] | REQUIRED | TEST[ING} on page 6-6 for details.
- The RDFCONFIG global parameter has been changed. Existing RDFCONFIG global settings will not be processed. See RDFCONFIG { RDF-control-subvol | DEFAULT } { AUTO[MATIC] | MANUAL | NOREPL[ICATE] } on page 6-7 for details on the new options.
- A new STATUS * command has been added. See STATUS * on page 6-17 or details.

- Paragraph SDR Fundamental Concepts on page 4-1 describing SDR basic rules has been added to Section 4, SDR Operations.
- Clarifications and corrections have been made throughout the document.

About This Manual

This manual describes the uses, installation, and all other operational aspects of the HP NonStop SDR software.

This software product replicates SQL/MP DDL operations to one or more backup systems.

Notation Conventions

Hypertext Links

Blue underline is used to indicate a hypertext link within text. By clicking a passage of text with a blue underline, you are taken to the location described. For example:

To start the SDR monitor, see START MONITOR on page 6-15.

General Syntax Notation

The following list summarizes the notation conventions for syntax presentation in this manual.

UPPERCASE LETTERS. Uppercase letters indicate keywords and reserved words; enter these items exactly as shown. Items not enclosed in brackets are required. For example:

```
MAXATTACH
```

lowercase italic letters. Lowercase italic letters indicate variable items that you supply. Items not enclosed in brackets are required. For example:

```
file-name
```

[ ] Brackets. Brackets enclose optional syntax items. For example:

```
TERM [\system-name.]$terminal-name
```

```
INT[ERRUPTS]
```

A group of items enclosed in brackets is a list from which you can choose one item or none. The items in the list may be arranged either vertically, with aligned brackets on each side of the list, or horizontally, enclosed in a pair of brackets and separated by vertical lines. For example:

```
FC [ num ]
   [ -num]
   [ text]
```

```
K [ X | D ] address-1
```

{ } Braces. A group of items enclosed in braces is a list from which you are required to choose one item. The items in the list may be arranged either vertically, with aligned braces on each side of the list, or horizontally, enclosed in a pair of braces and separated by vertical lines. For example:

```
LISTOPENS PROCESS { $appl-mgr-name }
                  { $process-name  }

ALLOWSU { ON | OFF }
```

| Vertical Line. A vertical line separates alternatives in a horizontal list that is enclosed in brackets or braces. For example:

```
INSPECT { OFF | ON | SAVEABEND }
```

… Ellipsis. An ellipsis immediately following a pair of brackets or braces indicates that you can repeat the enclosed sequence of syntax items any number of times. For example:

```
M address-1 [ , new-value ]...

[ - ] {0|1|2|3|4|5|6|7|8|9}...
```

An ellipsis immediately following a single syntax item indicates that you can repeat that syntax item any number of times. For example:

```
"s-char..."
```

Punctuation. Parentheses, commas, semicolons, and other symbols not previously described must be entered as shown. For example:

```
error := NEXTFILENAME ( file-name ) ;

LISTOPENS SU $process-name.#su-name
```

Quotation marks around a symbol such as a bracket or brace indicate the symbol is a required character that you must enter as shown. For example:

```
"[" repetition-constant-list "]"
```

Item Spacing. Spaces shown between items are required unless one of the items is a punctuation symbol such as a parenthesis or a comma. For example:

```
CALL STEPMOM ( process-id ) ;
```

If there is no space between two items, spaces are not permitted. In the following example, there are no spaces permitted between the period and any other items:

```
$process-name.#su-name
```

Line Spacing. If the syntax of a command is too long to fit on a single line, each continuation line is indented three spaces and is separated from the preceding line by

a blank line. This spacing distinguishes items in a continuation line from items in a vertical list of selections. For example:

```
ALTER [ / OUT file-spec / ] LINE

   [ , attribute-spec ]...
```

Change Bar Notation

Change bars are used to indicate substantive differences between this edition of the manual and the preceding edition. Change bars are vertical rules placed in the right margin of changed portions of text, figures, tables, examples, and so on. Change bars highlight new or revised information. For example:

> The message types specified in the REPORT clause are different in the COBOL85 environment and the Common Run-Time Environment (CRE).
>
> The CRE has many new message types and some new message type codes for old message types. In the CRE, the message type SYSTEM includes all messages except LOGICAL-CLOSE and LOGICAL-OPEN.

HP NonStop SQL DDL Replicator User's Guide

1 Introducing SDR

This manual describes HP NonStop SQL DDL Replicator (SDR). This section is an introduction to SDR. The information is organized as follows:

Introduction

HP NonStop SQL DDL Replicator (SDR) is a companion to the NonStop Remote Database Facility (RDF) subsystem. It provides the ability to automatically replicate SQL DDL operations to a backup system in the same way that RDF replicates database updates to a backup database.

The initial release of SDR supports NonStop SQL/MP. All further references to SQL implies SQL/MP only. Also, all further references to DDL refer to NonStop SQL/MP DDL operations.

Note. Since SDR is an extension of RDF, this document assumes the reader has a good working knowledge of RDF and frequently makes reference to the *HP NonStop RDF System Management Manual for H-Series RVUs (RDF 1.8)* in the following form: (RDF 5. – Stopping RDF), which is the paragraph heading “Stopping RDF” in chapter 5 of the manual.

RDF uses TMF audit trail information to capture changes to a primary database and to replicate those changes to one or more backup copies of the database, usually on one or more remote systems. In the case of a planned or unplanned outage of a primary database system, the backup copies may be used to continue operations after a takeover operation.

Since the necessary information is not present in the TMF audit trail, RDF does not replicate DDL operations such as CREATE, DROP, and ALTER. In order to safely perform these operations in a standard RDF environment, HP recommends stopping all primary system applications, and stopping TMF and RDF (See RDF 5. – RDF and NonStop SQL/MP DDL Operations).

SDR performs a series of operations to both capture DDL operations and to replicate them on the backup database. It does this quickly, automatically, without user intervention and without affecting applications or stopping TMF or RDF. SDR replicates all DDL operations that are executed by SQLCI, user applications, or access utilities like ODBC. No changes to user applications are required.

Product Overview

SDR installs quickly. Once the product files are in place, you use the SDR command interpreter, SDRCOM, to issue a few commands that set up a small configuration database and to install SDR on your system.

SDR is privileged, so the installation of the product files must be performed by SUPER.SUPER. It does not require a cold load, application outage, or stopping RDF or TMF. All other operations can be performed by a member of the SUPER group.

SDR must be installed on both RDF primary and backup systems.

SDR requires licensing on each system where it is installed. Before attempting to install SDR, you must obtain a product licenses from HP, as instructed on the product CD.

During the installation process, components of SQL are updated and become dependent on the presence of SDR. Be sure to read the installation section carefully before attempting to remove or alter the SDR software.

Once installed and licensed, SDR is ready to capture all DDL operations performed on the system. This happens automatically and is invisible to the SQL users. DDL can be executed in both SQLCI and in applications; no application or operational changes are necessary.

When a DDL operation has been captured, SDR uses the RDF configuration information to translate the DDL to the backup database. SDR coordinates with RDF to execute the DDL at precisely the right time.

In general, you do not need to start and stop SDR, even if you are starting, stopping or reinitializing RDF. The SDR monitor is started when SDR is installed and should also be started whenever the system is cold-loaded. It automatically monitors the RDF state and activates SDR replication when RDF is started. SDR does not perform any DDL replication if RDF is stopped.

SDR Components

SDR consists of 4 program files and 6 data files. Each of the program files contains privileged code and must be licensed.

Program Files

SDR Command Interpreter

The command interpreter, SDRCOM, is the primary user interface for installing, configuring, and controlling SDR.

SDR Monitor ($ZSDR)

The SDR monitor (SDRMON) is a NonStop process pair with the reserved name $ZSDR. It should be started at cold load time and left running at all times, regardless of the state of RDF or TMF. If the SDR monitor is executing and properly licensed, then SDR is active.

SDR Runtime

The SDR runtime (SDRRUNTM) captures DDL operations and stores them in appropriate locations for replication at a later time. It does not require RDF to be executing to gather and store the necessary information for later use.

SDR Updater

The SDR updater (SDRUPDT) is a program that performs the actual replication of DDL at the appropriate time. It will only be executing if RDF is started.

Data Files

The following SDR components are files containing information required by SDR.

SDRERROR

A specially formatted file containing all information, warning, and error messages issued by the SDRCOM and, sometimes, by the monitor and updater processes.

SDRHELP

A specially formatted file containing HELP information issued by the SDRCOM.

ZSDRTMPL

An EMS template file containing templates for the SDR subsystem. These templates should be installed in your system as described in the HP DSM Services Manual.

SDRFLTR

An EMS filter for viewing SDR events.

ZCOMTMPL and COMFLTR

An EMS template file and filter for viewing events for the both RDF and SDR. This is an unsupported component of the SDR product, as it may not be consistent with the installed release of RDF on your system. But, it can be useful to monitor SDR as it operates in conjunction with RDF.

NonStop Kernel and RDF Compatibility

SDR is a release-independent product. There is only one version of SDR and it runs on all supported NonStop servers supported by HP: NonStop S-series, NonStop H-series and NonStop J-series.

SDR is also compatible with all versions of RDF (IMP, IMPX and ZLT) starting with SPR T0346AAJ (released summer 2000), with a few exceptions. SDR will not work with T0346A07 SPRs ABJ or ABO, or with base release T0346H08. SDR checks usage of RDF and issues a warning message if you are using an incompatible release of RDF.

2 Installing SDR

This section describes the installation of SDR. It covers the following topics:

Prerequisites

SDR works closely with RDF and must be installed on the RDF primary and backup systems. Versions of SDR software must be the same on all systems.

Read the Software Release Document before you install SDR.

SDR works with RDF and RDF/IMP as described in NonStop Kernel and RDF Compatibility on page 1-4.

Like RDF, you must be a member of the SUPER group to perform SDR operations. To install the SDR product files, you must be SUPER.SUPER, as described below in Install Product Files.

In anticipation of the SDR installation, the operators should be made aware of the RDF events that will no longer require their intervention. See paragraph Changes to RDF Operations below for a description of these changes.

Initial Installation

To insure that the backup system is ready for replication, first install SDR on the backup system.

For each system where SDR is to be installed, have your SDR licensing instructions available. Proceed as follows:

1. Move Files from the Product Media to the Installation Subvolume.
2. Install Product Files.
3. Start an EMS distributor.
4. Create the SDR System Database
5. Install the SDR License

6. Enable SQL for DDL Replication
7. Install EMS templates
8. Update System Coldload Procedures

Repeat the installation on the all the systems in the RDF network.

Move Files from the Product Media to the Installation Subvolume

The SDR product media contains the files listed in Table 2-1. Instructions for placing the SDR software in the ISV subvolume ZISDR are delivered on the distribution media. After the initial installation, these files can also be copied from another system on your network, or copied using FTP over the Internet from an HP site.

The subvolume ZISDR contains the following files:

Table 2-1. ZISDR Subvolume

FIle Name	Contents
SDRPAK	A PAK file containing the SDR product files listed in Table 2-2
INSTALL	TACL macro for installing SDR
T2828*nnn*	Software release document where *nnn* designates the PVU level
ZSDRTMPL	EMS templates for SDR events

The actual operational product files for SDR are listed in Table 2-2.

Table 2-2. SDR Product Files in SDRPAK

File Name	Contents
SDRERROR	Error and warning messages
COMFLTR	EMS filter for RDF and SDR events
SDRFLTR	EMS filter for SDR events
SDRHELP	Help text for command interpreter SDRCOM
SDRCOM	SDR command interpreter
SDRMON	Monitor program
SDRRUNTM	SDR library
ZCOMTMPL	EMS template file for SDR and RDF events
ZSDRTMPL	EMS template file for SDR events

Install Product Files

The INSTALL macro is used for the initial SDR installation and for subsequent SDR product updates.

When installing SDR for the first time, INSTALL performs the following steps:

1. Restores the product files from ZISDR to the operational subvolume
2. Licenses the SDR object files.

You must be logged on as SUPER.SUPER to run the INSTALL macro. Because local SUPER.SUPER access is required to license object files, INSTALL cannot be run from a different system.

The default operational subvolume is $SYSTEM.SDR but you can install the files in other locations. Do not install the files in $SYSTEM.SYSTEM or $SYSTEM.SYS*nn*.

To install SDR in $SYSTEM.SDR:

1. Log on as SUPER.SUPER
2. Type the following TACL commands:

```
VOLUME $vol.ZISDR
RUN INSTALL
```

Running INSTALL:

```
> INSTALL
HP Nonstop(tm) SQL DDL Replicator(tm) Software File Installer

*** SDR software will be installed in subvolume $system.sdr

UNPAK - File decompression program - T1255G06 - (2008-06-03)

Archive version: 1
File Mode RESTORE Program - T9074G08 (21JUL2008) (AFO)
(C)2000 Compaq (C)2006 Hewlett Packard Development Company, L.P.
Drives: (\SIERRA.$X0HN)
System: \ATOM  Operating System: H06  Tape Version: 3
Backup options:  NO AUDITED, BLOCKSIZE 8, NO IGNORE, NO OPEN, PARTONLY OFF,
                 INDEXES IMPLICIT
*WARNING-7147*  Files created and stored via OSS and SQL/MX objects are not
                 supported.
Restore time: 18Mar2010 11:00  Backup time:  8Mar2010 14:57           Page: 1

Tape: 1       Code             EOF      Last modif   Owner RWEP   Type Rec Bl

$SYSTEM.SDR
 COMFLTR       101              582  8May2008 15:59    -1   NUNU
 SDRCOM        100          3907336  8Mar2010 14:51    -1   NUNU
 SDRERROR                    270336  1Mar2010 10:35    -1   NUNU   K    128  4
 SDRFLTR       101              304  8May2008 15:59    -1   NUNU
 SDRHELP                     114688  3Mar2010 16:10    -1   NUNU   K   1928  4
 SDRMON        100          5677056  8Mar2010 14:54    -1   NUNU
 SDRRUNTM      100          2748416  8Mar2010 14:55    -1   NUNU
 SDRUPDT       100          3696640  8Mar2010 14:57    -1   NUNU
 ZCOMTMPL      839           233472  4Mar2010 10:25    -1   NUNU   K    510  4
 ZSDRTMPL      839            73728  4Mar2010 10:25    -1   NUNU   K    510  4

Summary Information

Files restored = 10  Files not restored = 0
********************************************************************************
*** SDR software file installation is complete. Proceed to installation   ***
*** instructions described in the user manual.                             ***
********************************************************************************
```

If for any reason you want to install SDR in a location other than $SYSTEM, specify the target subvolume when you run the INSTALL macro:

```
RUN INSTALL $vol.SDR
```

Start an EMS distributor

Before you install the SDR EMS templates in the system templates, you can view SDR messages through an EMS distributor, using the SDR templates and filter. Enter the following TACL commands in a separate terminal window:

```
ADD DEFINE =_EMS_TEMPLATES, CLASS MAP, FILE ZSDRTMPL
EMSDIST TYPE P, COLLECTOR $0, TEXTOUT $HOME, FILTER SDRFLTR
```

The EMS distributor will display SDR events.

Create the SDR System Database

The SDR System Database (SysDB) is created using the SDR command interpreter SDRCOM. When starting SDRCOM during the initial installation, a warning message appears directing you to create a SysDB and license the SDR product.

You must be a member of the SUPER group in order to create the SysDB. If you are not logged on as a SUPER user, SDRCOM displays an error message and does not create the SysDB.

The SUPER user that creates the SYSDB becomes the SDR *owner*. The SDR owner is similar to the RDF owner, except that any SUPER user that has update access to the SysDB REGISTRY table is considered to be an SDR owner.

The SDR owner can start and stop the monitor, issue the INSTALL SDR command, and perform the SDR control functions described in Section 5, SDR Monitoring and Control under Controlling Replication.

To create the SDR SysDB:

1. Log on as a SUPER user
2. Invoke SDRCOM from the TACL prompt:

```
7> sdrcom
SHP Nonstop(tm) SDR(tm) Command Interpreter(T2828V01) - System \SIERRA
(C)2010 Hewlett Packard Development Company, L.P.
(C)2010 Carr Scott Software Incorporated

*Warning* The SysDB cannot be found. Issue the CREATE SYSDB command to
* 1601  * create it. Enter HELP CREATE-SYSDB; for more information.

*Warning* Use of SDR requires licensing. To obtain the
* 1801  * licensing key, send an e-mail to LICENSE.MANAGER@HP.COM.
*       * Please provide your system serial number and order number.
*       * This system has serial number 35507.
```

3. Create the SDR SysDB tables by issuing the CREATE SYSDB command.:

```
SDRCOM 1? create sysdb;
--- SysDB table $SYSTEM.SDRSYSDB.REGISTRY created.
--- Starting Monitor process $ZSDR

SDR Monitor 1.9.1 - 29MAR2010 -- $ZSDR (0,221) - System \SIERRA
Started at Mar 18 2010 11:09:20, elapsed time = 0:00:00
Backup process cpu 1, no takeovers
Monitor hometerm: $0
EMS event logging level is NORMAL.

*Warning* Use of SDR requires licensing. To obtain the
* 1801  * licensing key, send an e-mail to LICENSE.MANAGER@HP.COM.
*       * Please provide your system serial number and order number.
*       * This system has serial number 35507.
SDR monitor $ZSDR (0,221) is monitoring 0 RDF environment(s).
Current SDRUPDT processes:    0
Total SDRUPDT processes:      0
```

CREATE SYSDB performs the following operations:

1. Creates the REGISTRY table in subvolume $SYSTEM.SDRSYSDB. If creating tables on $SYSTEM is difficult due to system security or if $SYSTEM is not audited, specify an alternative volume using the ON *volume* option. SysDB may be created on any audited volume.

   The REGISTRY table is owned by the SUPER user that issues the CREATE SYSDB command and has a default security of “NOOO”. The REGISTRY table owner is the SDR owner who is authorized to manage SDR.

   Use the SECURE option to secure REGISTRY differently. Setting the write security to “G” will allow all SUPER users to manage SDR.

2. Creates an SQL catalog on the volume specified in the CREATE SYSDB command, in a subvolume called SDRCATLG. Use the CATALOG option to specify a different catalog.

3. Starts start the SDR monitor $ZSDR. This is the default name for the SDR monitor.

See CREATE SYSDB on page 6-9 for details on the command options.

Install the SDR License

Follow the licensing procedure described in the documentation that accompanies the SDR software. The license is installed by entering the SDRCOM license command as it is supplied by HP. If you move the SDR to a different system, you must request a license for that system.

After you have successfully installed your SDR license, SDRCOM displays the state of your license:

To display the status of your SDR software license subsequently, use the Monitor STATUS LICENSE command.

Note. If you are upgrading your SDR license from a limited license to a permanent license, simply install the new license received from the License Manager. Installing a valid license is not disruptive and does not require any interruption of SQL operations.

Enable SQL for DDL Replication

If you are installing SDR on a system that hosts the production application or database and intend to test SDR on the system, please consult Appendix C, Testing SDR before you enable SQL for DDL replication.

After you have successfully installed your SDR license, SDRCOM notifies you that SQL is not yet configured to perform DDL capture and replication:

```
*Warning* $SYSTEM.SYSTEM.SQLCAT is not properly configured for use with
SDR.
* 1669  * This file must be the NonStop SQL/MP product program and it
*       * must be activated for use with SDR with the following command:
*       *
*       *            INSTALL SDR
*       *
```

To enable DDL replication, enter the INSTALL SDR command. Because the SQL utilities might be in use at the time the INSTALL SDR command is executed, SDRCOM performs the following operations to insure that SQL is correctly enabled for SDR, without impacting the current users (here we assume SDR is installed in $SYSTEM.SDR):

- Notifies you that the SQL utilities will be unavailable for a brief moment, and prompts for permission to proceed. If you are ready, enter Y.
- Duplicates the utilities to $SYSTEM.ZASDR.SQL*util*Z.
- Prepares the duplicated copy SQL*util*Z.
- Renames $SYSTEM.SYSTEM.SQL*util* to $SYSTEM.ZASDR.SQL*util*,where it continues to execute, if it is in use.

  Note. . The next time INSTALL SDR is executed, SDRCOM will check that the renamed files in ZASDR are now closed. If they are still running, INSTALL SDR will terminate with an error message instructing you to stop them.

- Renames the prepared $SYSTEM.ZASDR.SQL*util*Z to $SYSTEM.SYSTEM.SQL*util*, enabled for SDR when the next user executes the SQL utilities.

You are now able to replicate DDL operations for your RDF protected audited SQL tables.

Install EMS templates

The file ZSDRTMPL contains EMS templates for all events generated by SDR. The file ZCOMTMPL contains templates for SDR and RDF events. Please refer to the *DSM Template Services Manual* for instructions about installing system templates.

You can use the EMS filter file SDRFLTR with an EMS distributor to display SDR events, as described above in Start an EMS distributor on page 2-4.

Update System Coldload Procedures

To insure that the replication of DDL operations is active, the monitor process must be running. You should include the start of the monitor process with the other operational steps that normally follow a system cold load.

Start the monitor process after TMF has been started and before any DDL operation is performed on your SQL database. To start the SDR monitor process, use the following TACL command:

```
RUN $SYSTEM.SDR.SDRCOM START MONITOR
```

Updating SDR Software

The version of SDR must be the same on the primary and backup systems. If you update SDR software on the any of the participating systems, you must update the software on all systems. For a safe and minimal outage, the SDR installation steps should be executed in the following sequence:

1. On both the primary and backup systems, restore the product media as instructed in Move Files from the Product Media to the Installation Subvolume
2. On the primary system:
    1. Make sure that there are no DDL operations in progress on the primary system. Use STATUS UPDATE to verify that no operations are in progress.
    2. Disallow SQL DDL operations during the period that SDR is unable to capture them for later replication:
        - In SDRCOM: ALTER GLOBAL DDLCAPTURE REQUIRED
    3. In SDRCOM: STOP UPDATE
3. On the backup system (or systems):
    1. Replace the SDR software components in the product subvolume using the INSTALL macro as described below in Install Product Files for SDR Update. The INSTALL macro restarts the SDR monitor.
    2. Install the EMS templates. See Install EMS templates on page 2-7.
4. On the primary system:
    1. Replace the SDR software components in the product subvolume using the INSTALL macro as described below in Install Product Files for SDR Update. The INSTALL macro restarts the monitor with UPDATE.
    2. Install the EMS templates. See Install EMS templates on page 2-7.
    3. RESET the DDLCAPTURE global parameter if it was set only for the duration of the installation: RESET GLOBAL DDLCAPTURE.

You do not need to re-issue the INSTALL SDR command.

Install Product Files for SDR Update

When replacing SDR product files, the INSTALL does additional processing to account for the possibility that the SDR files are in use. INSTALL proceeds as follows:

- Warns you that while the software is being replaced, DDL replications could be lost, and tells you what to do to prevent it.
- Prompts you for permission to proceed with the installation. If you are ready, enter Y.
- If the updater object file SDRUPDT is opened, instructs you to STOP UPDATE on the primary SDR system and to restart INSTALL.
- If other SDR product files are opened, instructs you to close them.
- Stops the SDR monitor if it is executing.
- If the SDRRUNTM is opened because it is in use by a prepared SQL utility, restores the new SDRRUNTM as NEWRUNTM in the SDR subvolume.
- Executes the SDRCOM command INSTALL RUNTIME to replace the existing SDR runtime with the new version. See description of the command below on page 6-11,

  Note also that INSTALL RUNTIME aborts with no change if any of the SQL*util* objects duplicated to the ZASDR.SQL*util* for previous installation is still in use and cannot be purged.
- After the product files have been replaced, restarts the monitor. On the primary system, INSTALL restarts the SDR monitor specifying the UPDATE option.

Enabling DDL Capture when Updating NonStop SQL Software

If you install a new version of the operating system or a new version of SQL, you need to enable the new SQL components for DDL capture and replication, as described above in Enable SQL for DDL Replication on page 2-6.

Disabling DDL Capture

To disable SDR DDL capture, set the DDLCAPTURE global parameter to DISABLED using SDRCOM command ALTER:

```
SDRCOM 1? alter global ddlcapture disabled;
--- SDR Global SDRDDLCAPTURE updated.
--- Monitor $ZSDR globals refreshed.
```

SDR displays a message in the EMS log and STATUS SDR shows the following:

```
SDRCOM 2? status sdr

* Warning 1810 * SQL DDL Capture is disabled.
SDR monitor $ZSDR (0,817) is monitoring 1 RDF environment(s).
Current SDRUPDT processes:    1
Total SDRUPDT processes:      6
-----------
RDF ATOMJ (\ATOM -> \SIERRA) at  4/28/08 13:52:42
Current State: Normal

SDR Updater \SIERRA.$Y27G, holding
No replications performed by this SDR updater.
```

Removing SDR Software

If you have enabled SQL for SDR, you must remove SDR from the SQL utilities before you remove the SDR software from your system. If SDR is not “uninstalled” and the SDR software is removed from your system, DDL operations on the primary system will fail.

To remove SDR from your system:

1. In SDRCOM, issue the UNINSTALL SDR command. Like INSTALL SDR, UNINSTALL SDR performs the necessary duplications and rename operations to account for the possibility that the SQL utilities are in use at the time the command is executed. UNINSTALL SDR also stops the SDR monitor.
2. Use SQLCI to purge the REGISTRY table in the SDRSYSDB subvolume.
3. Purge the SDR software in SDR subvolume $SYSTEM.SDR or the subvolume where it was installed.
4. Update your system startup and shutdown procedures to remove the commands that start and stop the SDR monitor process.

Changes to RDF Operations

The function of SDR is to automate the replication of SQL DDL statement to backup tables, thereby performing operations that were previously done manually by operators. Moreover, to perform these functions, SDR coordinates its activities with RDF by monitoring RDF events. The installation of SDR on your system therefore calls for some changes in the way operators respond to a few specific RDF events that are reported on the backup system, mostly by ignoring them.

ZRDF-EVT-Msg-736

```
Waiting to obtain FILEINFO on file $vol.ZASDRnnn.SDRDEPOT,
error 11
```

This message from the RDF Updater will appear after the initial installation of SDR the the first time a SQL DDL operation is performed on a table residing on $*vol*. It will appear for each volume protected by RDF at the first DDL operation captured and replicated and will not appear again. The *$vol*.ZASDR*nnn*.SDRDEPOT file is automatically created by SDR and the operator can safely ignore the message.

ZRDF-EVT-Msg-908

```
A file is prepared for SQL DDL operation
```

Prior to the installation of SDR, this message from the RDF Updater was used to notify the operator that an SQL DDL operation had been performed on the primary system, that all RDF updaters had processed the required audit and that it was now safe to execute the same DDL operation on the backup database. The operator was also advised to obtain the name of the source file involved in the operation on the primary system by looking for the last RDF 733 event.

The replication of the DDL operation is now automatically executed on the backup table by SDR and the RDF 733 event is no longer reported. The operator should ignore this message. The message will be followed by the RDF 807 event that signals that the SDR Updater has restarted RDF updates.

ZRDF-EVT-Msg-700, ZRDF-EVT-Msg-705

(*with parallel volume-sharing RDF configurations*)

```
File system error 10 on $vol.ZASDRnnn.SDRDEPOT,SNO n,RBA rba
and
File system error 71 n $vol.ZASDRnnn.SDRDEPOT,SNO n,RBA rba
```

```
File open error 12 on $vol.ZASDRnnn.SDRDEPOT
```

These messages from the RDF Updater appear if SDR is installed on systems where RDF is deployed in parallel configurations. Parallel configurations refer to sites where two or more RDF configurations are defined between same primary and backup systems and replicate to the same backup disk volume.

An example: Between \PRIM and \BACK, PRIMA replicates $P.SUBA to \BACK.$P.SUBA and PRIMB replicates $P.SUBB to \BACK.$P.SUBB.

In such RDF configurations, the operator can safely ignore the file system errors 10 and 71 and the open error 12 reported on the $*vol*.ZASDR*nnn*.SDRDEPOT file by the RDF Updaters.

3 Configuring SDR

This section discusses the SDR configuration options and, because many functions of SDR depend on proper configuration of RDF, a few pertinent RDF considerations. The chapter is organized as follows:

RDF Considerations

RDF Topologies

SDR supports all RDF topologies (as described in the RDF - 1 – Features).

RDF Configuration

For the most part, you configure RDF in the normal way, without regard for SDR. However, there are a few RDF configuration options you need to consider.

REPLICATEPURGE

The REPLICATEPURGE option instructs RDF to automatically replicate Enscribe file purge operations (RDF 3 – Setting Global Parameters). Some installations disable this option in order to retain the backup database in case a user accidentally purges a critical primary file. Others perform regular file purges and would be inconvenienced if RDF stopped updating every time this happened.

When RDF REPLICATEPURGE is ON, SDR adopts the RDF setting and replicates DROP operations. If RDF automatically replicates purges, SDR automatically replicates DROP operations.

When RDF REPLICATEPURGE is OFF, there is a significant difference between RDF and SDR. While RDF simply ignores the purge (this usually leads to an updater error 10 at some future time), SDR issues EMS messages and holds off replication until the user decides to either execute or cancel the DROP statement.

NETWORK and NETWORKMASTER

The proper definition of an RDF network is of great importance to SDR if you perform DDL on distributed tables with secondary partitions or indexes remote from the primary

table partition. See the discussion below in Configuring the SDR Network for further details.

Replicating Physical VOLUMEs in an SMF Environment

The RDF VOLUME configuration determines how SDR translates the DDL for execution on the backup systems. When configuring RDF to replicate databases on SMF virtual volumes, it is usually not necessary to specify the physical volumes. If, however, you specify PHYSVOL in DDL statements, you must specify the physical volume mappings in RDF.

INCLUDE and EXCLUDE

SDR stores the DDL information it needs for replication in files called depot files, that reside in subvolumes on the replicated volumes named ZASDR*nnn,* where *nnn* is the primary node number. These files are described in detail in paragraph SDR Depot Files on page 4-3.

In order to function correctly, the depot files in ZASDR*nnn* subvolumes must be replicated by RDF.

- More recent version of RDF, starting with SPRs H08^ABR and A07^ABS, automatically replicate ZASDR*nnn* subvolumes. In fact, ZASDR*nnn* subvolumes cannot be excluded from the RDF configuration.
- When prior versions of RDF are in use, SDR checks the RDF configuration to insure that the ZASDR* subvolumes are replicated by RDF. If ZASDR* subvolumes have been excluded, or have not been included, SDR issues a warning to notify you that the RDF EXCLUDE or INCLUDE specification will cause SDR to malfunction.

RDF subvolume name mapping using a mapfile (RDF 12) does not permit mapping subvolumes beginning with X, Y, or Z, so it cannot interfere with SDR replication.

RDF EMS Events and the LOGFILE

SDR monitors the operation of RDF in many ways, but one primary input is RDF events in the EMS event log. RDF writes events to both $0 and any EMS collector configured RDF as the RDF LOGFILE. SDR always monitors the LOGFILE collector if it is configured. Otherwise, it monitors the $0 collector.

You must not configure “pre-log filtration” to remove the RDF events from the event log that SDR is monitoring.

Configuring SDR

The following paragraphs describe global settings that control the capture and replication of DDL operations.

The global settings are changed using the ALTER command and reset to their default with the RESET command.

```
ALTER [GLOBAL] parameter-and-value
```

To issue the ALTER and RESET commands you must be a SUPER user and have update access to the REGISTRY table in the SDRSYSDB subvolume.

Configuring SDR Capture

Configuring SDR Replication

Configuring SDR Capture

The following paragraphs describe global settings that control DDL capture:

Enabling and Disabling SDR

Requiring DDL Capture

Retaining Captured DDL

Handling DDL Operations performed under a User Transactions

Enabling and Disabling SDR

It is highly recommend to leave the SDR monitor running at all times. If it becomes necessary to disable DDL capture for any reason, it can be done by setting the DDLCAPTURE global parameter to DISABLED.

Requiring DDL Capture

When a DDL operation is performed on a SQL table, operational problems could prevent SDR from capturing and preserving the DDL statement for replication. For example, if the SDR monitor is stopped, DDL capture is impossible, or if a security error prevents SDR from accessing its files, then the statement cannot be stored. The capture problem on the primary table might cause the backup table to become inconsistent or unusable.

By default, SDR allows the DDL operation to be performed on the primary table, even if it cannot be captured for replication. But you can choose to make the DDL replication mandatory by not allowing the DDL operation to succeed on the primary table if it cannot be captured by SDR. This is done by setting the global parameter DDLCAPTURE to REQUIRED (instead of the default ENABLED).

Obviously, mandatory DDL replication could have an adverse affect on applications that perform DDL. But, if all DDL is performed using ad hoc SQLCI commands, it is a useful safeguard against accidental corruption of the backup database.

Retaining Captured DDL

At the time you perform either primary database updates or DDL operations, you might not have configured RDF yet, or even have a backup database. You can decide, weeks after an application has started updating a database on the primary system, to initialize RDF and replicate all the updates that were performed in past weeks to a backup database. All you need to do is save the TMF audit trails.

Similarly, SDR captures and preserves the DDL being executed on a primary table, with no knowledge of RDF. Indeed, DDL capture is completely independent of DDL replication. This is very similar to the use of TMF that is independent of RDF.

The de-coupling of DDL capture from replication is why the DDL retention period is important. It can be set to any value between 1 day and 60 days. The default is one week. Set retention higher if you plan to initialize RDF to a date that goes back in time more than one week. To change the default retention period, set the global parameter RETENTION to the desired value.

Note. SDR allows one extra day past the configured retention period when deleting old records from SDRDEPOT files on the backup system. In the normal case, old records are deleted on the primary system and those deletes are quickly replicated by RDF to the backup system. The delay prevents the backup system from deleting old records before the primary system has a chance to do so.If the backup SDR deletes records before the primary SDR, RDF issues EMS message 700, error 11 on the SDRDEPOT file. This has no ill effect but is an additional distraction for the operator.

Handling DDL Operations performed under a User Transactions

Certain types of DDL operations can be done in a user-initiated transaction. For example, you can DROP and CREATE a table as a single transaction; if the transaction fails, the original table is still available. User transactions containing DDL operations are rare, but they are allowed.

User transactions complicate the DDL replication. At the time SDR replicates the DDL, the outcome of the user transaction is not known.

You can configure how you want SDR to handle DDL operations that are executed in a user transaction, by setting the global parameter USERTRANSACTION to one of the following options:

- ABORT: disallow DDL inside a user transaction. If a user attempts such an operation, the transaction is aborted, the DDL operation fails, and the database is rolled back to the state it was when the transaction began. SDR writes a message to the home terminal and to EMS whenever it aborts a transaction.
- ASSUMECOMMIT: assume that user transactions containing DDL always commits. SDR ignores the fact that the DDL is part of a user transaction and always replicates it.

- HOLD: allow the operator to manually review each DDL operation executed under a user transaction, and decide if it should be executed or canceled. SDR and RDF updating is suspended until the operation is either executed or canceled.

  HOLD is the default.

Configuring SDR Replication

The following paragraphs describe global settings that control the replication of DDL operations:

Setting CREATE Ownership and Security

Setting the User ID for Replicated DDL Operations

Automatic Catalog Creation

Replicating Unaudited Tables

Configuring the SDR Network

Be sure to set the global parameters on the primary and backup systems. Unlike RDF, SDR does not copy its configuration files to the backup system, because a single instance of SDR can service many RDF configurations, both primary and backup. Use INFO GLOBALS with the OBEYFORM option to capture and reenter the SDR global settings from one system to another.

If SDR global settings are not the same on both the primary and backup systems, SDR may not operate properly.

Setting CREATE Ownership and Security

You can configure the user ID that will own the tables, indexes and views resulting from the replication of CREATE operations, by setting the global parameter CREATEID to the desired value. By default, the owner of the new SQL objects is the owner of the primary SQL object.

You can also specify a security string to add to CREATE TABLE, VIEW, and CATALOG operations if none is explicitly specified, by setting the global parameter CREATESECURITY.

Setting the User ID for Replicated DDL Operations

By default, all replicated DDL operations other than CREATE (see discussion above) are performed under the super id.

You can choose instead to have the operations performed under the same user ID that performed the operation on the primary system, by setting the global parameter ACCESSID to USERID.

Automatic Catalog Creation

If a DDL operation references a catalog that does not exist on the backup, SDR can be configured to create the catalog automatically, by setting the global parameter AUTOCREATECATALOG to ON.

SDR creates a catalog automatically when audit records indicate that a catalog table is being updated, but only if the primary catalog resides on a subvolume that is replicated by RDF (included or, not excluded, in the RDF configuration).

When a catalog is updated on the primary system, the updated rows are placed in the TMF audit trail. If RDF is configured to replicate updates in the catalog subvolume, RDF requires that the catalog tables exist on the backup system. If they do not, then the RDF updater stalls, issuing event 736 messages.

If AUTOCREATECATALOG is set, then the SDR updater detects that RDF is looking for the catalog table and creates the catalog. SDR creates the catalog even if DDLCAPTURE is DISABLED and the CREATE CATALOG was not captured.

Replicating Unaudited Tables

SDR is designed to operate in conjunction with RDF, to perform DDL replication automatically and at exactly the right time, relative to RDF audited data replication. Thus, it is designed primarily for audited tables.

SDR can be configured to replicate DDL for unaudited SQL operations, by setting the global parameter UNAUDITEDDDL. It cannot, however, coordinate DDL replication with any data operations, so this feature might have limited usefulness in your environment.

Configuring the SDR Network

To create and perform DDL operations on distributed tables, with partitions and indexes on multiple systems, SDR must have an accurate picture of all the RDF configurations, including the primary and backup systems for each one.

If you have installed RDF IMPX, it is preferable that you use the RDF NETWORK configuration to do this. RDF has excellent verification capabilities to ensure that the configuration is complete and consistent.

If you do not have RDF IMPX (or do not want to configure RDF for network transactions) you can configure SDR to specify the primary and backup systems for all of the affected RDF configurations, using the NETWORK global parameter.

Note that, within SDR, an SDR network configuration overrides any RDF network configuration.

4 SDR Operations

This section describes how SDR captures, replicates and executes DDL operations and various special processing cases. The topics covered are the following:

SDR Monitor

The SDR monitor process has a reserved process name, $ZSDR. It is a NonStop process pair to ensure that a CPU or network failure does not interrupt SDR processing.

The SDR monitor has three basic functions:

1. Maintain configuration options for the SDR software that captures the DDL operations.
2. Monitor the state of RDF and start an SDR updater process on the backup system for each active RDF configuration.
3. Implement an efficient resource locking protocol to coordinate the capture of DDL operations that are performed on distributed tables.

Once SDR is installed, the SDR monitor process should be executing at all times. It is required for the capture of DDL operations.

The SDR monitor must also be executing on the RDF backup system for SDR replication to occur.

The system cold load procedures should be updated to start the monitor right after TMF has been started. Unlike RDF, the SDR monitor does not need to be stopped in order to perform routine maintenance operations. Even stopping TMF does not require SDR to be stopped.

SDR Fundamental Concepts

There are two fundamental rules or concepts about SDR that users must keep in mind when trying to manage SDR. Users that have managed RDF systems can make incorrect assumptions about how SDR works. Understanding the following “rules of the road” will help eliminate some confusion about what SDR is doing.

Rule 1 - DDL Capture

The “capture” or extraction of SQL DDL on any system is, and must be, completely independent of any RDF configuration. RDF can be configured to replicate database audit (and now SQL DDL) that was generated in the past, so SDR does not know whether or not you will configure and start RDF on a system at some future time. The SQL DDL must be already captured, along with the other TMF audit, to replicate everything.

DDL capture performed by SDR is analogous to TMF capture of database updates in the audit trail. TMF does not make any assumptions about whether or not you intend to replicate database updates using RDF. It simply captures the updates. The same is true of SDR capturing DDL statements; SDR has no knowledge of the RDF configuration, and it captures all SQL DDL operations.

Rule 2 - DDL Replication

The “replication” of SQL DDL is driven by RDF. There is no independent replication of SQL DDL by SDR unless RDF is actively replicating database updates.

You may notice SDR capturing SQL DDL on the backup system and wonder if SDR will replicate these operations to the primary system. Replication will NOT occur, unless you have configured and started RDF to replicate database audit from the backup to the primary system.

Understanding and remembering these basic rules is essential to dispel misconceptions and confusion about how SDR actually works.

DDL Capture

SDR installation modifies SQL software to enable DDL capture, but in a way that is invisible to your applications and database maintenance activities. Installing SDR does not change the way you use SQL. You should see no significant operational or performance change when performing any SQL operations on the primary system.

The following paragraphs discuss topics related to the capture and storage of DDL statements on the primary system:

DDL Capture Artifacts

DDL Capture Operation

Special cases

DDL Capture Artifacts

There are three artifacts used by SDR to perform DDL capture:

- SDR Depot Files
- Stop RDF Update Audit Records

SDR Depot Files

SDR depot files are automatically created by SDR as needed. A depot file is an audited Enscribe key-sequenced file. A depot file is created on each primary database volume that has a SQL object (table, index, so on) that is the target of a captured DDL statement.

The name of the subvolume where a depot file is stored is a combination of the reserved name ZASDR and the Expand node number. The depot filename is always SDRDEPOT.

For example, the depot file $DATA.ZASDR004.SDRDEPOT is created for SQL objects residing on $DATA, on the primary RDF system with an Expand node number of 4.

The replication of the SDRDEPOT files to the backup system is described below in Creating the SDR Depot Files on the Backup on page 4-8.

Stop RDF Update Audit Records

Stop RDF Update (SRU) audit records are described in (RDF 5 – Performing Shared Access DDL Operations). When SDR is not installed, SRU audit records are generated by SQL to signal the completion of a WITH SHARED ACCESS DDL operation. When RDF updaters encounter an SRU record, they stop at an appropriate location to allow you to manually perform the DDL operation on the backup system and then manually restart RDF updating.

The SRU record contains the name of the SQL table or index that was the target of the WITH SHARED ACCESS operation. The file name is used by RDF for informational purposes only; it is written to the EMS log to alert the operator to the need to manually replicate the DDL and restart RDF updating.

When SDR is installed, NonStop SQL no longer generate SRU audit records for WITH SHARED ACCESS operations, because SDR replicates the DDL operations automatically.

You will, however, see SRU messages produced by RDF in the EMS log, because SDR generates SRU messages to coordinate its operation with RDF. For operational purposes, these EMS messages should be ignored.

Marker Files

Marker files are used to satisfy an artificial requirement of RDF that each file specified in a Stop RDF Update audit record must exist. If the file does not exist, the RDF Extractor aborts and cannot be restarted without reinitializing RDF.

Marker files are created in the same subvolume as depot files. They are empty, unaudited, and unstructured Enscribe files, with names like

$DATA.ZASDR004.AA000123. Marker files are managed automatically by SDR. They occupy no disk space. SDR purges marker files when they are no longer needed.

▲ **WARNING.** Purging Marker files manually can cause RDF to abort with no possibility of restart. You would need to reinitialize RDF to an audit location after the SRU record that references the marker file, and likely lose database audit that keeps the backup database synchronized.

DDL Capture Operation

You can perform DDL operations in an application program, in SQLCI, or in other subsystems such as ODBC or DBA/m. In any of these cases, SDR is informed at various points of the DDL processing and performs some additional operations as described in this chapter.

DDL capture is completely independent of the RDF configuration. In the general case, RDF need not be configured when the primary DDL operation occurs.

When a DDL operation occurs, SQL performs numerous catalog updates and file label operations. It usually performs these as a single TMF transaction. During the DDL processing, SDR captures the operation, including environmental information such as DEFINEs and default subvolumes.

SQL eventually commits or aborts the transaction used to perform the DDL change. If the transaction commits, SDR writes a description of the DDL to a depot file, which is replicated by RDF to the RDF backup system(s). SDR also creates the marker file to prevent the RDF Extractor from aborting.

Later, when RDF is replicating the catalog audit for the DDL, it will replicate the SDR depot records and then encounter the SRU audit record and shut down the RDF updaters. SDR processing from this point is described below.

Special cases

Create Catalog

User Transactions

Distributed Tables and Indexes

Unaudited Tables and Indexes

SQLCI DUP Utility

Create Catalog

In the DDL operation to create a catalog, SQL first creates all the audited catalog tables, opens them, and inserts the initial catalog rows into the tables. This is all done under a single transaction.

In the typical case, this causes RDF to stall if the catalog is not yet created on the backup system. When the RDF updater processes the audited catalog inserts, it does

not know that they apply to a catalog and cannot discard them. More specifically, RDF events 736 appear in the EMS log, when the RDF updater detects the SQL updates to the catalog table VERSIONS:

Waiting to obtain FILEINFO on file *$vol.catsubvol*.VERSIONS, error 11

SDR detects this situation and updates the depot file with a CREATE CATALOG before the audited catalog inserts. The CREATE CATALOG will be replicated before the RDF updater would stall at the catalog inserts. The downside of this strategy is that there is a very slight chance that the primary CREATE CATALOG could be aborted and SDR could create a backup catalog where no primary catalog was created. But, this is not likely to have any adverse effect on maintaining a consistent duplicate database.

CREATE Collation

SDR replicates CREATE COLLATION on the backup system in two steps: first it translates the name of the collation and the input edit file used by the collation compiler, then it executes a SQL CREATE COLLATION statement.

SQL in turn executes the CREATE COLLATION in two steps: first it creates a special audited file to be used for output by the collation compiler, then it invokes the collation compiler that makes audited updates to its output file.

Because there are additional files involved, there are potential issues in replicating CREATE COLLATION relating to the use of these files.

The collation compiler input file

- Since edit files are not replicated by RDF, you must ensure that the same character processing rules are present on both systems before executing the CREATE COLLATION on the primary system.

  If the input edit file does not exist on the backup system, SDR will continue to retry the CREATE COLLATION until the input file is created or until you cancel the operation.

- Not only must the input edit file exist on the backup system, but it must also be in sync with the input file on the primary. Problems will occur if the primary input file is updated and an out-of-date file still exists on the backup.

Note. If you have automated the replication of non database files between your primary and backup systems, be sure to include the collation compiler input edit file into your automatically replicated file sets.

The audited collation output file

The audited updates that the collation compiler makes to its output file cause RDF to stall, because the corresponding audited compiler output file does not exist on the backup system.

Without SDR, the operator has to create a collation manually to allow RDF to continue processing.

With SDR, the operator no longer needs to intervene. SDR detects the absence of the audited file and inserts a CREATE COLLATION into the depot file, before the collation compiler has a chance to generate its audited updates. Thus, CREATE COLLATION is replicated before RDF encounters the compiler generated audited updates.

△ **Caution.** The downside of this strategy is that the CREATE COLLATION operation could fail on the primary system, for example due to syntax error in the compiler input, but succeed on the backup system. This can be avoided by making sure that both systems have the same input edit file.

User Transactions

Normally, SQL creates and commits a transaction for each DDL operation. No user transaction management is required. But, you can execute DDL in a transaction that you create and commit. This leaves SDR with a dilemma, as it does not know the final status of the user transaction when it is time to replicate the DDL.

See paragraph Handling DDL Operations performed under a User Transactions on page 3-4 for a discussion on the options available to deal with this situation.

Note. If you have a serious need to execute DDL in a user transaction, please contact HP product support.

Distributed Tables and Indexes

If a table or index is distributed, with partitions or indexes on multiple systems, RDF updaters on all systems must be stopped at the same point in processing their primary system audit. SDR writes SRU audit to the audit trail of every referenced system.

Since SDR might be processing another distributed DDL operation at the same time, this leads to the possibility of confusion and deadlock. Two sequences of SRU audit records sent to remote audit trails might be inserted in different orders. One RDF might be waiting for one distributed DDL replication while another RDF is waiting for a different DDL replication.

To avoid this problem, SDR obtains an SDR lock on each of the affected systems before writing any of the SRU audit. The SDR monitor on each system manages the lock. If SDR cannot immediately obtain all the locks it needs, it releases all the locks, waits for a short time, and tries again. A random backoff algorithm ensures resolution within a short time.

If the SDR monitor on any of the affected systems is not accessible, the DDL operation will be terminated with an error message.

Unaudited Tables and Indexes

SDR writes every DDL operation to a depot file, but it writes the SRU audit record only if the target SQL object is audited. Since the SRU audit initiates SDR replication, DDL for unaudited objects is not replicated. See Replicating Unaudited Tables on page 3-6 for options available to process DDL replication on unaudited tables.

SQLCI DUP Utility

The DUP command executes several internal DDL operations to create tables and indexes and to alter table attributes. DDL for DUP operations are never captured or replicated by SDR.

DDL Replication

The following paragraphs discuss topics relating to the replication of the captured DDL statements to the backup system(s):

SDR Updater Processes

Replication Operations

SDR Updater Processes

The SDR monitor is responsible for monitoring all RDF activity. It does so by monitoring RDF events sent to collector $0. The SDR monitor also makes a periodic check of RDF and eventually makes the proper adjustments, even if the RDF events are not available. When the SDR monitor detects an active RDF subsystem, it starts an SDR updater process on the RDF backup system.

The SDR updater monitors events sent to the EMS collector that is configured as the RDF LOGFILE. Although RDF always sends events to both $0 and any configured LOGFILE, some installations might have a filter to eliminate RDF events. The SDR updater depends on analyzing RDF events for proper operation.

Although the SDR updater is started and managed from the primary RDF system, the SDR monitor must also be executing on the backup system.

Each SDR updater manages a specific RDF configuration. By RDF configuration, we refer to the configuration identified by an RDF control subvolume of format *node*[*suffix-character*]. If there are multiple active RDF control subvolumes, the SDR monitor will start multiple SDR updater processes, one for each control subvolume. For example, if on system \PRIM there are RDF control subvolumes PRIMA and PRIMB, there will be two SDR updaters on the backup system, one for PRIMA and one for PRIMB.

Normal vs. Takeover Operations

When RDF is operating in normal mode, the SDR monitor on the primary RDF system manages the SDR updater on the backup system. When RDF is operating in takeover

mode, it is assumed that the primary system is unavailable; the SDR monitor on the backup system manages the SDR updater.

EMS Messages

In contrast to RDF, each SDR component, including the SDR updater, logs all EMS events to the system that it was started from. When RDF is in normal mode, all SDR EMS events are sent to the primary system EMS collector. When RDF is in takeover mode, all SDR events are sent to the backup system EMS collector. This maintains a better time line of all SDR events and avoids problems comparing primary and backup EMS logs.

Replication Operations

Replicating and executing DDL statements entails the following operations:

Creating the SDR Depot Files on the Backup

Processing Stop RDF Update Audit Records

Translating DDL

Executing DDL

Restarting RDF Update

Creating the SDR Depot Files on the Backup

The SDR depot files $*vol*.ZASDRnnn.SDRDEPOT contain the DDL statements captured on the primary system. The creation of the depot files is not replicated to the backup by RDF. When RDF first attempts to insert updates into the depot file before it exists on the backup, it issues event 736. The SDR updater detects the event signaling that RDF is looking for the file and creates the depot file.

Processing Stop RDF Update Audit Records

When the RDF Extractor processes an SRU audit record, it filters it based on the associated file name. If the volume or subvolume is not protected by the RDF configuration, the SRU record is discarded.

If the SRU record is accepted, it is sent to the RDF receiver who places it in every updater image trail. If the SRU record is processed, it causes all RDF updaters to shut down, with the backup database consistent up to the point of the associated DDL statement. When all updaters have shut down, RDF issues an EMS 908 event. When the SRU is associated with SDR, the filename will have the form $vol.ZASDR*nnn*.AA*mmmmmm* where *nnn* is the primary Expand node number, and *mmmmmm* is a unique sequence number assigned by SDR.

Note that the 908 shutdowns do not stop normal RDF extractor-to-receiver processing. All audit produced on the primary database is still being transmitted and stored on the

backup, so there is no increased exposure to lost transactions on the backup in case of a takeover.

In RDFCOM, a STATUS RDF command will show an “Update NSA Stopped” state. Prior to SDR, this is the point where system operators would manually execute the DDL operation and restart the RDF updaters.

If SDR is installed, it is critical to notify the operators that they should not be issuing an RDFCOM START UPDATE command when RDF is in this state.

SDR automatically restarts the RDF updaters after it has replicated the DDL. If this doesn’t happen promptly, you should investigate the problem with SDRCOM. If you need to bypass the DDL operation to restart the RDF updaters, use the SDRCOM CANCEL command.

You may also issue an RDFCOM START UPDATE command and SDR will simply not replicate the DDL. SDR has no memory of which DDL operations have not been replicated. It simply monitors RDF and performs the associated DDL operation when RDF enters the “Update NSA Stopped” state.

When RDF enters the Update NSA Stopped state, the SDR updater accesses the RDF backup copy of the depot file where the associated original DDL statement was stored. Note that the SRU file name will have a subvolume beginning with ZASDR.

If the SRU file subvolume does not begin with ZASDR, then the audit was not produced by SDR and SDR will ignore it. It may have been caused by a WITH SHARED ACCESS operation occurring when SDR DDL capture is disabled. You must deal with this situation using standard RDF (i.e., manual) methods.

Translating DDL

When the DDL has been fetched from the depot file, it is translated for execution on the backup system. SDR uses the RDF configuration to replace file names and catalog names with their RDF backup equivalents. SDR uses the RDF volume table, any includes and excludes, and the mapfile if configured.

During the translation process, SDR distinguishes between the target SQL object and any secondary file reference. The target object is the first file name in the DDL statement, specifying the table or index or view being created or updated. All other names, such as catalogs, partitions, physical volumes, a CREATE LIKE table, or a table being indexed are secondary references.

If the DDL target volume is not replicated, or the target file name or subvolume is excluded, SDR skips the operation and restarts the RDF updaters. If RDF replicates

the target name, but does not replicate some secondary reference, SDR signals an error and waits for operator intervention.

Note. Many RDF configurations have excluded SQL catalog subvolumes. If your DDL references a catalog on an excluded subvolume of a replicated volume, SDR will still attempt to replicate the operation. It will use the same subvolume on the RDF backup volume.

But, if the backup catalog does not exist, SDR (and RDF) will wait for operator intervention. You will have to either create the catalog manually or CANCEL the DDL replication.

This rule does not apply to CREATE CATALOG or DROP CATALOG on a subvolume that is excluded in RDF. Also, in this situation, AUTOCREATECATALOG will not create a catalog.

Appendix A, SQL DDL Statements provides information on how each DDL statement is translated.

Executing DDL

When the DDL has been successfully translated, SDR executes it on the backup database.

If the command drops a SQL object, then SDR refers to the REPLICATEPURGE setting of RDF. If the setting is ON, the DROP is executed; otherwise SDR issues an EMS message and awaits operator intervention.

When executing replicated DDL, SDR reports SQL errors in the EMS log and waits for the operator to take action. See Controlling Replication in SDR Monitoring and Control.

If the DDL statement alters or drops a table that does not exist on the backup, the operation is discarded and the RDF updaters are restarted.

When executing the DDL for a CREATE operation, SDR assumes the user identity that executed the primary system DDL. Thus, the same user will own both primary and backup SQL objects. You can configure the user ids that SDR should use for CREATE operations. See Setting CREATE Ownership and Security on page 3-5 for details.

SDR executes the DDL for other types of access as the SUPER user, just as RDF does for data replication. You can configure SDR to perform replicated DDL operations other that CREATE as the user who performed the operation on the primary. See Setting the User ID for Replicated DDL Operations on page 3-5 for details.

Restarting RDF Update

Once the backup DDL is successfully executed, or canceled by the operator, SDR sends RDF a message to restart its updaters. SDR then waits for the next RDF “Update NSA Stopped” state.

SDR and RDF Takeover

SDR operation in a planned or unplanned RDF takeover is completely automatic; no operator intervention is required.

An RDF takeover can occur only if the primary system has failed (or is unavailable on the Expand network) or if RDF has been stopped on the primary system. Either of these events will cause the SDR updaters started from the primary system to shut down.

When you issue the RDF TAKEOVER command, the SDR monitor on the backup system immediately starts an SDR updater for that RDF configuration. If there are any SDR-captured DDL operations in the remaining RDF image trail, RDF will shut down on each SRU audit record, SDR will execute the DDL operation and then restart the RDF takeover. Eventually, RDF will enter the “Takeover Complete” state and both RDF and SDR will shut down.

Note that, for complex RDF topologies, the SDR monitor can start SDR updaters for both primary and takeover RDF configurations on the same system.

Distributed Table Replication

If a table or index has partitions and indexes on remote systems, the DDL capture process has written a SRU audit record to the audit trail of each system that has a partition or an index of the table. On each one of those systems, RDF is expected to enter the “Update NSA Stopped” state for the same file name. (Actually, each SRU file name contains a primary volume on the remote system, but the ZASDR*nnn*.AA*mmmmmm* part of the file name will be the same.)

The SDR depot record contains a list of all systems that received an SRU audit record, including the volume name used in each remote SRU audit record. SDR must wait until all such systems have entered that state for the correct file name.

In order to translate distributed table DDL operations, SDR needs an accurate description of the RDF network, which is a list of RDF control subvolumes and their primary and backup systems. This must be specified in either the SDR configuration, in networks where RDF IMP is running, or in the RDF configuration, where RDF IMP/X is running.

SDR accesses the RDF configurations for each of the systems that are referenced in the DDL operation, and uses that information to translate the DDL to reference the appropriate backup systems and volumes.

Then, SDR is ready to execute the DDL operation and restart RDF updating on all of the listed systems.

5 SDR Monitoring and Control

This section describes the commands available to query the status of SDR and control its operations:

Checking Status

EMS Log

SDR sends numerous event messages to the EMS collector. The default collector is $0, but you can specify a different collector by setting the EMSCOLLECTOR global parameter to an alternate collector. Unlike RDF, SDR sends messages to a single collector. RDF always sends events to $0 as well as an alternate collector.

The SDR monitor and updaters log informational events, such as monitor and updater activity, action events that require operator intervention and critical events. All events are logged in the EMS collector on the primary system. Consequently, it is recommended you monitor the EMS log for SDR events, in the same manner you do for RDF events.

To facilitate the monitoring of the EMS log, the SDR product subvolume contains the EMS template and filter files ZSDRTMPL and SDRFLTR that restricts the event display to SDR events.

For an example of how you can start an EMS distributor to display only SDR events, see Start an EMS distributor on page 2-4.

For your convenience, the SDR product subvolume also contains EMS template and filter files ZCOMTMPL and COMFLTR that can be used to display RDF events in addition to SDR events.

The SDR Log

The SDR log (SDRLOG) is a key-sequenced Enscribe file where replicated DDL operations are recorded. SDRLOG resides on the backup system in the SDRSYSDB subvolume.

SDRLOG is created when the first DDL operation is replicated successfully. All replicated DDL operations are logged, even if they do not succeed.

The SDR updater detects if the log fills up and deletes the oldest records, making room for new records. The SDR update also detects retries resulting in the same error and does not write a new log entry that would be a copy of the previous one.

The format of the SDRLOG entries is proprietary. Use the READLOG command to examine the content of the log file. The display is similar in format to the output of the STATUS UPDATE command described below.

STATUS Commands

Used in conjunction with the EMS log, the STATUS commands provide information on the state of the participants in the DDL replication:

STATUS MONITOR

STATUS SDR

STATUS RDF

STATUS UPDATE

STATUS *

STATUS MONITOR

STATUS MONITOR or the equivalent, MONITOR STATUS, displays information about the monitor process and the objects it manages. In the example below, the monitor has just been started and no replication has taken place yet:

```
SDR Monitor 1.9.0 - 01JUN2008 -- $ZSDR (5,627) - System \ATOM
Started at May 1 2008  16:56:45, elapsed time = 17:02:04
Backup process $ZSDR (4,650), no takeovers
Monitor hometerm: $0
SDR License Information
License is                  Valid
Maximum logical processors   8

SDR monitor $ZSDR (5,627) is monitoring 1 RDF environment(s).
Current SDRUPDT processes:    1
Total SDRUPDT processes:      1
Monitor activity log is inactive

Monitor tracing is inactive.

Memory Pool Usage Statistics
Size = 2,097,066, current use = 976, maximum use = 5,088
```

STATUS SDR

STATUS SDR shows a combined summary of RDF and SDR information. It provides a simple way to look for abnormal conditions. In the example below, STATUS SDR shows that the SDR updater has been stopped:

```
SDRCOM 25? STATUS SDR
SDR monitor $ZSDR (5,627) is monitoring 1 RDF environment(s).
Current SDRUPDT processes:    0
Total SDRUPDT processes:      2
*** SDR update is stopped ***

-----------
RDF ATOMJ (\ATOM -> \SIERRA) at  5/ 2/08 10:17:22
Current State: Normal

SDR Update OFF
```

STATUS RDF

STATUS RDF shows the state of both primary and backup RDF configurations on the system. Each RDF configuration is identified by the RDF control subvolume with the name format *node*[*suffix-character*]. For example, on \PRIM, there can be primary RDF control subvolumes named PRIM, PRIMA, PRIMB, and so on. All RDF control subvolumes are stored on $SYSTEM.

The SDR command STATUS RDF shows RDF information that is pertinent to SDR. The DETAIL option adds volume mapping information to the display. In the example below, the RDF status for the ATOMJ configuration is displayed.

```
SDRCOM 31? STATUS RDF ATOMJ,DETAIL
RDF Configurations on System \ATOM --  5/ 2/08 10:19:35

RDF ATOMJ (\ATOM -> \SIERRA)
------------------------------------------------------------
Status                    Normal
Config Version            8
Initialized                5/ 1/08 16:46:57
Monitor Process           $JDFM (1,604)
Last Config Modification   5/ 1/08 16:53:50
Log File                  $0
Access Id                 255,255 (SUPER.SUPER)
Owner

Primary -> Backup      In/Excludes Mapfile
$QA     -> $ORANGE      2/0
$QA2    -> $BANANA      2/0
$QA3    -> $PEACH       2/0
$QA4    -> $GRAPE       2/0
$QA5    -> $LEMON       2/0
```

Note that the SDRCOM STATUS RDF is not a substitute for the RDFCOM equivalent command. SDR shows only the information that is of a particular concern to SDR processing.

STATUS UPDATE

The STATUS UPDATE command shows the state of SDR updaters that are controlled from a selected system. The DETAIL option adds the DEFINES and the text of the last replicated DDL statement to the display. In the example below, the last statement was a CREATE TABLE LIKE.

```
SDRCOM 35?   STATUS UPDATE ,DETAIL
SDR monitor $ZSDR (5,627) is monitoring 1 RDF environment(s).
Current SDRUPDT processes:    1
Total SDRUPDT processes:      3
RDF Control Subvol   ATOMJ
SDR Updater Process  \SIERRA.$X7DC (8,375)
RDF Primary Sys      \ATOM
RDF Backup Sys       \SIERRA
State                Updater executing

SDR Operation               Count  Failed
SDRDEPOT fetch                  1
DDL translation                 1
SQL process start               1
DDL execution                   1
RDF update restart              1

Total processing time         3.068 seconds
Average processing time       3.068 seconds

Last DDL replication     5/ 2/08 10:38:35
Stop RDF Update file    $QA3.ZASDR099.AA000008
Depot file              $PEACH.ZASDR099.SDRDEPOT
Depot record key:       $QA3    00000008
SQL statement:          CREATE TABLE \ATOM.$QA3.JCCSDSQL.JOJO

ADD DEFINE =DEPT, CLASS MAP, FILE $PEACH.JCCSDSQL.DEPT

CREATE TABLE \SIERRA.$PEACH.JCCSDSQL.JOJO LIKE =DEPT CATALOG
\SIERRA.$ORANGE.JCCSDCAT;
--- SQL operation complete.
```

STATUS *

The STATUS * command, with a DETAIL option, is a combined form of STATUS RDF, STATUS SDR and STATUS UPDATE.

When submitting a problem report, you should always submit the output of STATUS *,DETAIL with the problem description.

Controlling Replication

The SDR monitor process running on the primary system and the updaters running on the backup system are controlled from the primary system using two types of commands:

Monitor-Level Commands

Updater-level Command

Monitor-Level Commands

The SDR monitor starts and stops the SDR updaters on the backup system. The following commands are related to monitor functions:

START UPDATE

STOP UPDATE

RESTART UPDATE

START UPDATE

START UPDATE starts all SDR updaters or the updater for a specific RDF configuration.

```
SDRCOM 39? START UPDATE
SDR updater ATOMJ will be started.
```

Specifying HOLD starts the updaters in the hold state, even if you have released the updater prior to issuing the START.

Use the RELEASE command to remove the updater from the HOLD state.

STOP UPDATE

STOP UPDATE stops all SDR updaters or the updater for a specific RDF configuration. The STATUS UPDATE command shows the updater as “Not executing”:

```
SDRCOM 37? STOP UPDATE
SDR updater ATOMJ will be stopped.

SDRCOM 37? STATUS UPDATE
SDR monitor $ZSDR (5,627) is monitoring 1 RDF environment(s).
Current SDRUPDT processes:    0
Total SDRUPDT processes:      3
*** SDR update is stopped ***

RDF Control Subvol   ATOMJ
SDR Updater Process  Not executing
RDF Primary Sys      \ATOM
RDF Backup Sys       \SIERRA
State                Updater idle (Update OFF)
```

RESTART UPDATE

RESTART UPDATE stops and then starts each SDR updater. The updaters can be restarted in a HOLD state.

In the following example, STATUS UPDATE shows the SDR updater is in a HOLD state, RESTART UPDATE is used to stop it and restart it and the subsequent STATUS UPDATE shows the updater executing:

```
SDRCOM 60? STATUS UPDATE
SDR monitor $ZSDR (5,627) is monitoring 1 RDF environment(s).
Current SDRUPDT processes:    1
Total SDRUPDT processes:      8
RDF Control Subvol   ATOMJ
SDR Updater Process  \SIERRA.$X8LB (3,410)
RDF Primary Sys      \ATOM
RDF Backup Sys       \SIERRA
State                Updater executing (Update HOLD)

*** This updater is in HOLD state. ***

SDR Operation               Count  Failed
SDRDEPOT fetch                  0
DDL translation                 0
SQL process start               0
DDL execution                   0
RDF update restart              0

No replications performed by this SDR updater.

SDRCOM 61? RESTART UPDATE
Restarting SDR updater ATOMJ with HOLD OFF.

SDRCOM 62? STATUS UPDATE
SDR monitor $ZSDR (5,627) is monitoring 1 RDF environment(s).
Current SDRUPDT processes:    1
Total SDRUPDT processes:      9
RDF Control Subvol   ATOMJ
SDR Updater Process  \SIERRA.$X8MZ (4,379)
RDF Primary Sys      \ATOM
RDF Backup Sys       \SIERRA
State                Updater executing

SDR Operation               Count  Failed
SDRDEPOT fetch                  0
DDL translation                 0
SQL process start               0
DDL execution                   0
RDF update restart              0

No replications performed by this SDR updater.
```

Updater-level Command

As mentioned in paragraph SDR Updater Processes on page 4-7, an SDR updater manages one RDF configuration or RDF control subvolume. This is why SDR updater commands operate on a single SDR updater process, identified by the RDF control subvolume associated with this SDR updater.

HOLD RDF-control

RELEASE RDF-control

EXECUTE RDF-control

CANCEL RDF-control

RETRY RDF-control

You must be a SUPER user to execute SDR updater commands.

HOLD *RDF-control*

HOLD places the specified updater in the hold state. When in the hold state, an updater will fetch and translate a pending DDL operation, but will not execute it.

In the example below, the HOLD command places the updater on hold as shown by the subsequent STATUS UPDATE:

```
SDRCOM 63? HOLD ATOMJ

SDRCOM 64? STATUS UPDATE
SDR monitor $ZSDR (5,627) is monitoring 1 RDF environment(s).
Current SDRUPDT processes:    1
Total SDRUPDT processes:      9
RDF Control Subvol   ATOMJ
SDR Updater Process  \SIERRA.$X8MZ (4,379)
RDF Primary Sys      \ATOM
RDF Backup Sys       \SIERRA
State                Updater executing

*** This updater is in HOLD state. ***

SDR Operation               Count  Failed
SDRDEPOT fetch                  0
DDL translation                 0
SQL process start               0
DDL execution                   0
RDF update restart              0

No replications performed by this SDR updater.
```

RELEASE *RDF-control*

RELEASE removes an updater from a HOLD state. If there is a pending DDL operation, the updater executes it immediately.

In the following example, the SDR updater is on hold and there is a pending DDL statement as shown by STATUS SDR. The subsequent RELEASE causes the updater to execute the pending statement:

```
SDRCOM 65? STATUS SDR
SDR monitor $ZSDR (5,627) is monitoring 1 RDF environment(s).
Current SDRUPDT processes:    1
Total SDRUPDT processes:      9
-----------
RDF ATOMJ (\ATOM -> \SIERRA) at  5/ 2/08 12:09:35
Current State: Update NSA Stopped
All 5 updaters are awaiting a SQL DDL operation.
File name is $QA3.ZASDR099.AA000010.

SDR Updater \SIERRA.$X8MZ
Last DDL replication     5/ 2/08 12:09:31
Stop RDF Update file    $QA3.ZASDR099.AA000010
Depot file              $PEACH.ZASDR099.SDRDEPOT
Depot record key:       $QA3    00000010
SQL statement:          CREATE TABLE \ATOM.$QA3.JCCSDSQL.JOJO3.

SDRCOM 66? RELEASE ATOMJ
```

EXECUTE *RDF-control*

If an updater has a pending DDL operation, EXECUTE causes the updater to execute the pending DDL operation immediately. The operation may be pending because the updater is in the hold state or because the operation requires manual intervention. An example of the latter is a DROP operation when REPLICATEPURGE has not been enabled.

In the example below, the updater is in a hold state and holding a DROP TABLE statement as shown by STATUS UPDATE. The subsequent EXECUTE causes the updater to execute the DROP statement

```
SDRCOM 6? STATUS UPDATE
SDR monitor $ZSDR (5,627) is monitoring 1 RDF environment(s).
Current SDRUPDT processes:    1
Total SDRUPDT processes:      9
RDF Control Subvol   ATOMJ
SDR Updater Process  \SIERRA.$X8MZ (4,379)
RDF Primary Sys      \ATOM
RDF Backup Sys       \SIERRA
State                Updater executing

*** This updater is in HOLD state. ***

SDR Operation               Count  Failed
SDRDEPOT fetch                 20
DDL translation                20
SQL process start               2
DDL execution                   2
RDF update restart              2

Total processing time         5.432 seconds
Average processing time       2.716 seconds

Last DDL replication     5/ 2/08 12:49:56
Stop RDF Update file    $QA3.ZASDR099.AA000012
Depot file              $PEACH.ZASDR099.SDRDEPOT
Depot record key:       $QA3    00000012
SQL statement:          DROP TABLE \ATOM.$QA3.JCCSDSQL.JOJO4

SDRCOM 7? EXECUTE ATOMJ
SDRCOM 8?
```

CANCEL *RDF-control*

If an updater has a pending DDL operation, either because the updater is in hold or because the updater is unable to execute the operation, CANCEL discards the pending DDL and the RDF updaters are restarted.

```
SDRCOM 10? STATUS UPDATE
SDR monitor $ZSDR (5,627) is monitoring 1 RDF environment(s).
Current SDRUPDT processes:    1
Total SDRUPDT processes:      9
RDF Control Subvol   ATOMJ
SDR Updater Process  \SIERRA.$X8MZ (4,379)
RDF Primary Sys      \ATOM
RDF Backup Sys       \SIERRA
State                Updater executing

*** This updater is in HOLD state. ***

SDR Operation               Count  Failed
SDRDEPOT fetch                 30
DDL translation                30
SQL process start               3
DDL execution                   3
RDF update restart              3

Total processing time         8.029 seconds
Average processing time       2.676 seconds

Last DDL replication     5/ 2/08 13:09:59
Stop RDF Update file    $QA3.ZASDR099.AA000013
Depot file              $PEACH.ZASDR099.SDRDEPOT
Depot record key:       $QA3    00000013
SQL statement:          DROP TABLE \ATOM.$QA3.JCCSDSQL.JOJO3


SDRCOM 11? CANCEL ATOMJ
SDRCOM 8?
```

RETRY *RDF-control*

If an updater has a pending DDL operation that it is unable to execute, then RETRY causes the updater to make a new attempt to execute the DDL operation. Note that an updater will retry a failed DDL operation every minute. Issuing the RETRY explicitly eliminates the delay and causes the updater to retry immediately.

In the example below a table cannot be created on the backup because there is already a table of that name. In such cases, SDR displays errors in the EMS log to notify the operator that the CREATE TABLE operation could not be performed. The error is also reported by STATUS SDR or STATUS UPDATER, as shown below in first STATUS UPDATER.

The operator purges the table on the backup system and issues a RETRY. The second STATUS UPDATE shows that the table was created successfully:

```
SDRCOM 8?STATUS UPDATER
SDR monitor $ZSDR (3,978) is monitoring 1 RDF environment(s).
Current SDRUPDT processes:    1
Total SDRUPDT processes:      7
RDF Control Subvol   ATOMJ
SDR Updater Process  \SIERRA.$Z3JF (0,410)
RDF Primary Sys      \ATOM
RDF Backup Sys       \SIERRA
State                Updater executing

SDR Operation               Count  Failed
SDRDEPOT fetch                  1
DDL translation                 1
SQL process start               1
DDL execution                   1       1 ( 4 retries )
RDF update restart              0

DDL replication time     6/25/08 17:36:27
Stop RDF Update file    $QA5.JASDR099.AA000006
Depot file              $LEMON.JASDR099.SDRDEPOT
Depot record key:       $QA5    00000006
SQL statement:          CREATE TABLE \ATOM.$QA5.JCCSDSQL.MYTAB
*** SQL DDL execution error: -1221

---> At this point, the operator purges the table on \SIERRA

SDRCOM 17? RETRY ATOMJ

SDRCOM 3? status update
SDR monitor $ZSDR (3,978) is monitoring 1 RDF environment(s).
Current SDRUPDT processes:    1
Total SDRUPDT processes:      7
RDF Control Subvol   ATOMJ
SDR Updater Process  \SIERRA.$Z3JF (0,410)
RDF Primary Sys      \ATOM
RDF Backup Sys       \SIERRA
State                Updater executing

SDR Operation               Count  Failed
SDRDEPOT fetch                  1
DDL translation                 1
SQL process start               1
DDL execution                   1       1 ( 0 retries )
RDF update restart              1

DDL replication time     6/25/08 17:40:21
Stop RDF Update file    $QA5.JASDR099.AA000006
Depot file              $LEMON.JASDR099.SDRDEPOT
Depot record key:       $QA5    00000006
SQL statement:          CREATE TABLE \ATOM.$QA5.JCCSDSQL.MYTAB
*** SQL DDL execution error: 0
```

6 SDR Commands

The command interpreter for SDR, SDRCOM, is used to install, configure, control and monitor SDR. This section describes the command interpreter and is organized as follows:

Running the Command Interpreter

Even if you can run SDRCOM as any user, most configuration and management commands must be performed by a member of the SUPER group.

Use this TACL RUN command to start the SDRCOM command interpreter:

```
[RUN] $SYSTEM.SDR.SDRCOM[/run-options/] [monitor] [cmd]
```

run-options

process options for any TACL RUN command.

monitor

If an alternate monitor is configured, specifies the name of the alternate monitor process. The monitor process name must be a local name, starting with a dollar sign (“$”).

cmd

specifies a command.

SDRCOM sets the completion code when the it stops. The completion code values are:

0: Normal completion

1: Warnings messages were issued during the session

2: Error messages were issued during the session

During a SDRCOM session, you can display the completion code by using the ENV command. Use the RESET command to restore the completion code to 0. The RESET

command also restores all parameters displayed by the ENV command to their default values.

Command Syntax

SDRCOM is a conversational-mode program. Most commands are single-line commands and are terminated by the end of line; such commands can be continued by placing an ampersand (&) at the end of the line.

Alternatively, if you wish to enter several commands on a single line, or if you do not wish to use a continuation character to enter long commands, you can choose to use a semicolon (;) to terminate each command. To use (or reset) the (;) as a termination character, use the SEMICOLON command.

Commands are free-format: spaces are not significant except within character strings. The commands are not case-sensitive.

Guardian defines can be used to specify command parameters:

- CLASS MAP defines to specify parameters that designate files or tables
- CLASS CATALOG defines to specify command parameters that designate subvolumes

Command Description Overview

The SDR configuration and management commands are shown in Table 6-1.

In addition to the SDR specific commands, Table 6-2 shows a list of utility commands supplied for general use.

The monitor process also accepts commands from SDRCOM; the commands are shown in Table 6-3. To enter a monitor command, precede the command with the keyword MONITOR.

Common programs in the $SYSTEM.SYSTEM (or SYSnn) subvolume can be run directly by typing the name of the program; these programs are shown in Table 6-4; refer to the RUN[D] command for the complete syntax. TACL macros cannot be run from SDRCOM.

Table 6-1. SDR Management Commands

ALTER	Changes the value of a global parameter
CANCEL	Restarts an RDF updater without performing current DDL operation
CREATE SYSDB	Creates the SDR system database
EXECUTE	Thaws the SDR updater and allows it to replicate the current DDL statement
HOLD	Puts an SDR updater in hold state
INFO GLOBALS	Displays SDR global values

Table 6-1. SDR Management Commands

INFO RUNTIME	Lists SQL/MP components that are using the SDR runtime
INSTALL RUNTIME	Replaces current SDRRUNTM with new SDRRUNTM
INSTALL SDR	Enables SQL software for DDL replication
READLOG	Displays entries from the SDR log file
RELEASE	Releases an SDR updater from a HOLD state
RESET	Sets a global value to the original default value
RESTART UPDATE	Restarts an SDR updater
RETRY	Instructs an SDR updater to retry a DDL operation
START MONITOR	Starts monitor processes
START UPDATE	Starts SDR updater on backup node
STATUS RDF	Displays RDF status information pertinent to SDR
STATUS SDR	Displays the status of SDR operations
STATUS UPDATE	Displays the status of SDR updaters
STOP MONITOR	Stops the monitor process
STOP UPDATE	Stops SDR updaters
UNINSTALL SDR	Removes SDR software from SQL components

Table 6-2. SDR Utility Commands

Command Name	Description
ABEND MONITOR	Abends the monitor process.
ALTER FILE	Changes the file attributes of a file set
COMMENT	Indicates the line is a comment line
COPY	Copies data from any input file to any output file
DELAY	Suspends the execution of the command interpreter
ENV	Displays current volume and other environmental settings
EXIT	Terminates SDRCOM session
FC AND!	Re-runs a previous command
FILEINFO	Displays information for Enscribe files
FILES	Displays a simple list of all files
HISTORY	Displays a list of previous commands
LOG	Output a copy of command input and output to a file
MONITOR	Sends a command to a monitor process
OBEY	Runs a sequence of commands from a file
OUT	Directs all command output to a file
RESET	Resets all environmental settings to their starting values

Table 6-2. SDR Utility Commands (continued)

Command Name	Description
RUN[D]	Runs an external process during a SDRCOM session
SEMICOLON	Toggles the use of a semicolon (;) as a command delimiter
STATUS MONITOR	Displays information about the monitor process
TIME	Displays the current time and date
VOLUME	Sets the SDRCOM session volume and subvolume

Table 6-3. Monitor commands

Command Name	Description
LOG	Starts and stops monitor activity logging
STATUS	Displays information about the state of the monitor
BACKUPCPU	Alters the backup cpu of the monitor process
SWITCH	Causes primary and backup processes to exchange roles

Table 6-4. Executable External Programs

AXCEL	BIND	C	COBOL85
DCOM	DDL	DSAP	ECOBOL
EDIT	ELD	ENABLE	ENFORM
ENMC	ENOFT	EPTAL	ERROR
FTP	FUP	INSPECT	MEASCOM
NLD	NMC	NMCOBOL	NOFT
OCA	PATHCOM	PERUSE	PTAL
SAFECOM	SPOOLCOM	SQLCI	SQLCOMP
TAL	TEDIT	TMFCOM	TNSVU
VPROC			

SDR Configuration and Management Commands

ALTER

Sets or changes global configuration values for SDR. Global parameters are the default configuration settings for SDR.

```
ALTER [ GLOBAL ] parameter-and-value

parameter-and-value is

{ ACCESS[ID] { USER[ID] | SUPER[ID] }                                  }
{                                                                      }
{ AUTOCREATECATALOG { ON | OFF }                                       }
{                                                                      }
{ CREATEID { user-ID | user-name }                                     }
{                                                                      }
{ CREATESECURITY “rwep”                                                }
{                                                                      }
{ DDLCAPTURE
       { ENABLE[D] | DISABLE[D] | REQUIRE[D] | TEST[ING]}              }
{                                                                      }
{ EMSCOLLECTOR [ $collector ]                                          }
{                                                                      }
{ KEEPPHYSVOL { ON | OFF }                                             }
{                                                                      }
{ NETWORK RDF-control-subvol, primary-node, backup-node                }
{                                                                      }
{ RDFCONFIG { RDF-control-subvol | DEFAULT }
              { AUTO[MATIC] | MANUAL | NOREPL[ICATE] }                  }
{                                                                      }
{ RETENTION n { HOURS | DAYS | WEEKS }                                 }
{                                                                      }
{ UNAUDITEDDDL { OFF | ON }                                            }
{                                                                      }
{ USERTRAN[SACTION] { ABORT | HOLD | [ASSUME]COMMIT }                  }
```

parameter-and-value

ACCESSID { USER[ID] | (SU{PER[ID] }

specifies the user ID under which the DDL replication operations are performed, with the exception of CREATE TABLE or CREATE INDEX operations (see CREATEID { user-ID | user-name } below).

Setting the global ACCESSID to USER[ID] instructs the SDR updater to use the same user ID to perform the DDL operation on the backup as was used to perform the DDL operation on the primary system.

The default is SUPER[ID], which causes all DDL operations, except CREATE, to be performed under SUPER,SUPER.

AUTOCREATECATALOG { ON | OFF }

specifies that SDR should automatically create catalogs on the backup system, whenever RDF indicates that a catalog is required. For automatic catalog creation, RDF must be configured to replicate data for the files in the catalog subvolume.

Automatic catalog creation is independent of any other SDR facility. RDF may be referencing a catalog due to reasons unrelated to an SDR DDL operation.

The default is OFF.

CREATEID { user-ID | *user-name* }

sets the owning user ID of any SQL object being created.

The default owner is the user ID that created the primary object.

CREATESECURITY “rwep”

sets the SECURE option of any SQL object being created.

DDLCAPTURE { ENABLE[D] | DISABLE[D] | REQUIRED | TEST[ING}

controls the capture of SQL DDL operations on the primary:

ENABLE[D]

enables the capture of all SQL DDL operations, providing

a. the SDR monitor is running, and

b. SDR is licensed and,

c. SDR has been INSTALLed in the SQL utilities.

DISABLE[D]

disables the capture of SQL DDL operations.

REQUIRE[D]

the capture of SQL DDL operations is enabled and, if for any reason the DDL operations cannot be captured by SDR, the DDL operations are not executed on the primary system.

If SDR aborts a DDL operation on the primary system, a message is logged to EMS to notify the user that DDL capture is REQUIRED, but an environmental problem is preventing SDR from capturing the DDL operation.

Note. As a usage example, you are advised to set DDLCAPTURE to REQUIRED when you install an update to the SDR software, at least until the installation is completed on all the systems in the RDF network. See “Updating SDR Software” on page 2-8 for further details.

TEST[ING]

disables the capture of DDL operations unless one of the following enablers is present:

- the CLASS MAP DEFINE =_SDR_TEST DEFINE has been set or,
- an empty file named ENABLSDR is present in the default subvolume for the process creator user id.

EMSCOLLECTOR $*collector*

specifies a collector process name for all SDR EMS messages.

The default collector is $0.

KEEPPHYSVOL { ON | OFF }

specifies that PHYSVOL specifications in CREATE and ALTER statements should always be retained, even if they lead to errors in replication.

The DEFAULT is ON.

NETWORK *RDF-control-subvol*, *primary-node*, *backup-node*

used for the replication of DDL operations performed on distributed tables or indexes, to provide SDR with the description of the RDF network nodes.

RDF-control-subvol is an RDF configuration in the format *node*[*suffix-character*].

primary-node and *backup-node* are the names of the primary and backup systems respectively

Specifying NETWORK is not required and is not recommended for RDF/IMPX installations, where SDR extracts the necessary network information from the RDF NETWORK configuration.

There is no SDR “master” network node. A complete description of the RDF network must be entered on every node, both primary and backup.

There can be no more than 31 NETWORK globals.

RDFCONFIG { *RDF-control-subvol* | DEFAULT }
{ AUTO[MATIC] | MANUAL | NOREPL[ICATE] }

used to support parallel RDF configurations (most commonly production and test) that may share RDF protected volumes, to avoid disrupting the specified RDF configuration (most commonly production).

RDF-control-subvol is an RDF configuration in the format *node*[*suffix-character*].

DEFAULT applies to all RDF configurations that do not have an explicit RDFCONFIG setting.

AUTOMATIC

The SDR Updater automatically replicates SQL DDL for the specified RDF configuration, subject to any other control such as HOLD, RELEASE and so on.

AUTOMATIC is the default.

MANUAL

If a SQL DDL target object (table, index, so on) is protected by an RDF configuration, the SDR Updater enters a hold state to allow the user to manually examine and control replication. The user should issue a CANCEL or EXECUTE command to proceed and restart RDF updating.

NOREPL[ICATE]

The SDR Updater does not replicate any SQL DDL operation for the specified RDF configuration and restarts RDF updaters immediately when the RDF updaters stop for the NSA Stop Update generated by SDR.

Note that AUTOCREATECATALOG is not active when NOREPLICATE is configured.

RETENTION n { HOURS | DAYS | WEEKS }

sets the retention period for SDR Depot records that store captured DDL operations.

The minimum retention period is 24 hours. If the user sets it to a smaller value, SDR rounds it up to 1 day.

The maximum retention time is 60 days.

The default is 1 week.

UNAUDITEDDDL { ON | OFF }

specifies how SDR will process DDL statements for unaudited tables.

If set to ON, SDR will replicate and apply DDL statements for unaudited tables, even if DML statements are not replicated by RDF.

The default is OFF

USERTRAN[SACTION] { ABORT | HOLD | [ASSUME]COMMIT }

specifies how SDR will process DDL statements executed under a user transaction.

ABORT

causes SDR to disallow the execution of a DDL statement within a user transaction. NonStop SQL already disallows the execution of many DDL statement within a user transaction; this further extends the prohibition to include all DDL statements.

When a DDL statement is attempted under a user transaction, SDR writes a message to the home terminal of the application program and to EMS and aborts the transaction, which causes the SQL statement to terminate with error -1353.

HOLD

causes SDR to write a message in the EMS log and wait for the user to issue an SDRCOM command to either EXECUTE or CANCEL the statement.

HOLD is the default.

[ASSUME]COMMIT

causes SDR to take the optimistic view that the transaction will eventually be committed and to replicate the DDL statement to the backup system.

CANCEL

Instructs the SDR updater that manages the specified RDF configuration to restart the RDF updaters without executing a DDL operation that was previously attempted and did not complete.

Restarting the RDF updaters that are awaiting completion of the DDL statement replication means that RDF simply proceeds and consequently, the DDL statement is never replicated.

Certain SDR configuration or processing errors make it impossible to determine the information necessary to restart the RDF updaters. For example, an incomplete RDF or SDR network specification can make it impossible to restart updaters on all participating network systems. In this case, CANCEL has no effect and you must use RDFCOM to restart the updaters.

```
CANCEL RDF-control-subvol
```

RDF-control-subvol

the RDF configuration in the format *node*[*suffix-character*] that replicates the SQL object that is the target of the DDL operation.

CREATE SYSDB

Creates the SDR Configuration Database. SysDB consist of the SQL REGISTRY that resides in a subvolume named SDRSYSDB. REGISTRY is updated with registration information and global parameters.

SysDB must be created on an audited volume.

```
CREATE SYSDB [ ON volume ] [, option ]

option is

{ CATALOG SQL-catalog                                    }
{                                                        }
{ SECURE "rwep"                                          }
```

volume

volume where SysDB will be created. The default is $SYSTEM.

option

CATALOG *SQL-catalog*

is the SQL catalog subvolume where the SysDB table is registered.

You can also select the catalog by specifying a catalog attribute for the =_DEFAULTS define.

If the SQL-catalog parameter is omitted, SDRCOM creates a SQL catalog on the same volume as the SysDB in a subvolume called SDRCATLG.

Following the creation of the system database, you should comply with the registration instructions.

If you create the SysDB on a subvolume other than $SYSTEM.SDRSYSDB, then some precautions must be taken, mainly to avoid creating more than one SysDB, because SDR searches all disks for the SYSDB and issue an error if more than one is found.

SECURE *“reap”*

specifies the READ, WRITE, EXECUTE and PURGE security attributes of SysDB tables.

After you create SysDB, comply with the product licensing instructions.

EXECUTE

Directs the specified SDR updater to execute the current DDL statement. EXECUTE is typically used when the SDR updater is in a hold state or when the operation requires user intervention.t.

```
EXECUTE RDF-control-subvol
```

RDF-control-subvol

an RDF configuration in the format *node*[*suffix-character*] that replicates the SQL object that is the target of the DDL operation.

HOLD

Instructs the SDR updater that manages the specified RDF configuration to cease replication of DDL operations. The updater remains active. If a DDL operation is ready to be executed, RDF remains in the Update NSA Stopped state.

```
HOLD RDF-control-subvol
```

RDF-control-subvol

the RDF configuration in the format *node*[*suffix-character*] that.replicates the SQL objects that are the target of DDL operations managed by the SDR updater.

INFO GLOBALS

Displays the list of configured SDR global parameters.

```
INFO GLOBAL[S]
```

See command ALTER above for a list of global parameters and values.

INFO RUNTIME

Lists the SQL/MP components currently executing that are using the SDR runtime.

```
INFO RUNTIME
```

INSTALL RUNTIME

Performs the file duplications and renames required to install the new SDRRUNTM while SQL utilities are still executing with the old SDRRUNTM:

- Renames existing SDRRUNTM to OLDRUNTM
- Renames and NEWRUNTM to SDRRUNTM
- Duplicates and renames SQL utilities so that new instances of SQL processes will use the new SDRRUNTM

INSTALL RUNTIME is executed by the INSTALL macro when a new version of SDR is installed on the system.

```
INSTALL RUNTIME new-runtime
```

new-runtime

the new SDR runtime that replaces the current SDRRUNTM.

Note that

- Unlike INSTALL SDR, the INSTALL RUNTIME command does not need the SDR monitor to be executing nor does it require the product to be licensed. This allows the INSTALL RUNTIME to be executed by the INSTALL macro.
- INSTALL RUNTIME does not perform the functions of INSTALL SDR: It does not prepare the SQL utilities, nor does it check that they are prepared. It merely duplicates executing utilities and renames them so that newly launched SQL processes will use the new SDRRUNTM.

INSTALL SDR

Enables the replication of DDL statements by preparing a number of NonStop SQL components residing in $SYSTEM.SYSTEM.

You must be a SUPER user with update access to the REGISTRY table to perform this command.

INSTALL SDR must be repeated when a new version of SQL is installed on your system. You can safely re-execute INSTALL SDR.

Once SDR has been enabled, you must not remove SDR objects from your system without first reversing the installation by issuing the UNINSTALL SDR command.

```
INSTALL SDR
```

READLOG

Displays the entries in the SDR log file SDRLOG. The display is similar to the output of STATUS UPDATE.

The SDR log file SDRLOG is an Enscribe key-sequenced file, residing on the backup system in the SysDB subvolume (SDRSYSDB). Its record format is proprietary and the file can only be displayed with READLOG.

```
READLOG backup-system [, options ]

options:

{ COUNT n                                                    }
{ DETAIL                                                     }
{ LAST n                                                     }
{ PRIMARY primary system                                     }
{ RDF[CONFIG] RDF-control-subvol                             }
{ STOP stop-time                                             }
{ TIME start-time                                            }
```

backup-system

the name of the RDF backup system where the DDL operation was executed.

backup-system is a required parameter because the SDR log resides on the system where the DDL operations are executed and is examined most often from the primary system.

options:

COUNT *n*

the maximum number of entries displayed from the log.

If count is omitted, all entries within the specified interval are displayed.

DETAIL

displays the full list of the SQL statements and associated DEFINEs.

LAST *n*

displays the *n* most recent entries from SDRLOG, between the TIME and STOP times (if specified). Specifying LAST overrides COUNT.

If LAST is omitted, all entries within the specified interval are displayed.

PRIMARY *primary-system*

the name of the system from which the DDL operation was captured.

This option is used if multiple RDF primary systems use the same RDF backup system, to select the name of a specific primary system.

RDF[CONFIG] *RDF-control-subvol*

displays only entries generated by an SDR updater that manages the specified RDF configuration.

RDF-control-subvol is in the format *node*[*suffix-character*].

STOP *stop-time*

time of the latest entry to display from the SDR log file.

stop-time is [[*date*] *time*] or [[*time*] *date*]
where: *time* is *hh*:*mm*[:*ss*] and *date* is *dd mmm yyyy* or *mmm dd yyyy*
(example: 11:15:00 30 May 2008)

TIME *start-time*

time of the earliest entry to display from the SDR log file.

start-time is [[*date*] *time*] or [[*time*] *date*]
where: *time* is *hh*:*mm*[:*ss*] and *date* is *dd mmm yyyy* or *mmm dd yyyy*
(example: 10:30:00 Jun 15 2008)

RELEASE

Releases an SDR updater in HOLD state.

```
RELEASE RDF-control-subvol
```

RDF-control-subvol

an RDF configuration in the format *node*[*suffix-character*] that replicates the SQL objects that are the target of DDL operations managed by the SDR updater.

RESET

Resets the value of a global parameter to its original default.

```
RESET [ GLOBAL ] parameter
RESET GLOBAL *
```

parameter

the global parameter to reset. In this case the keyword GLOBAL is optional.

*

all global parameters. In this case, the keyword GLOBAL is required.

For a list of global parameters, see command description for ALTER on page 6-5.

RESTART UPDATE

Instructs the monitor to stop and then restart the SDR updater that manages the specified RDF configuration (or stop and restart all SDR updaters).

```
RESTART UPDATE [ RDF-control-subvol ][, HOLD ]
```

RDF-control-subvol

the RDF configuration in the format *node*[*suffix-character*] that replicates the SQL objects that are the target of the DDL operations managed by the SDR updater.

If *RDF-control-subvol* is omitted, all SDR updaters are stopped and restarted.

HOLD

instructs the monitor to restart the SDR updater(s) in the HOLD state.

RETRY

Instructs the SDR updater that manages the specified RDF configuration to retry executing a DDL operation that was previously attempted and did not succeed.

```
RETRY RDF-control-subvol
```

RDF-control-subvol

an RDF configuration in the format *node*[*suffix-character*] that replicates the SQL object that is the target of the DDL operation.

START MONITOR

Starts a SDR monitor process.

You must be a SUPER user with update access to the REGISTRY table to start the monitor. The monitor process executes as the user who issues the START MONITOR command.

```
START MONITOR [ process ] [, option...]

option is:

{ BACKUP cpu-number                                          }
{                                                            }
{ EMSLEVEL { CRITICAL | DETAIL | NORMAL | TRACE }            }
{                                                            }
{ HOMETERM file-name                                         }
{                                                            }
{ PRIMARY cpu-number                                         }
{                                                            }
{ PRIORITY process-priority                                  }
{                                                            }
{ UPDATE { HOLD | OFF }                                      }
{                                                            }
```

process

specifies the name of the SDR monitor process to be started.

option

BACKUP *cpu-number*

cpu (in the range of 0 to 15) of the backup monitor processes. Used when the monitor process is started.

EMSLEVEL { CRITICAL | DETAIL | NORMAL | TRACE }

sets the granularity of EMS event reporting:

CRITICAL displays only messages that are critical or that require operator intervention.

DETAIL displays all EMS messages generated by the SDR monitor and SDR updaters.

NORMAL displays essential informational messages, action messages and critical messages. NORMAL is the default.

TRACE displays low level activity for problem analysis and should be used only when advised by HP support.

HOMETERM *file-name*

specifies a home terminal for the monitor process. Used when the monitor process is started.

PRIMARY *cpu-number*

cpu (in the range of 0 to 15) of the primary monitor processes. Used when the monitor process is started.

PRIORITY *process-priority*

specifies the priority (in the range of 1 to 199) for the monitor process. Used when the monitor process is started.

UPDATE { HOLD | OFF }

instructs the monitor to start (or not start) the SDR updaters.

If UPDATE HOLD is specified, the updaters are started in the HOLD state.

If UPDATE is omitted or UPDATE OFF is specified, the monitor does not start the SDR updaters.

The default is OFF.

START UPDATE

Starts the SDR updater that manages the specified RDF configuration (or all SDR updaters).

SDR updaters that are already executing are not affected by this command.

```
START UPDATE [ RDF-control-subvol ] [, HOLD ]
```

RDF-control-subvol

an RDF configuration in the format *node*[*suffix-character*] that replicate the SQL objects that are the target of DDL operations managed by the SDR updater.

If *RDF-control-subvol* is omitted, all SDR updaters are started.

HOLD

instructs the monitor to start the updater(s) in the HOLD state.

STATUS *

Combined form for STATUS SDR, STATUS RDF, and STATUS UPDATER.

```
STATUS * [, DETAIL ]
```

DETAIL

adds RDF volume information, the list of defines and the text of the last DDL statement performed by the SDR updater to the display.

STATUS RDF

Display the status of RDF. The command is not a substitute for its RDFCOM namesake. It displays RDF information that is pertinent to SDR operations.

```
STATUS RDF [ RDF-control-subvol ][, DETAIL ]
```

RDF-control-subvol

an RDF configuration in the format *node*[*suffix-character*].

DETAIL

adds the RDF volume information to the display.

STATUS SDR

Displays the state of DDL replication.

```
STATUS SDR [ \node ]
```

node

node name of the system for which the SDR updater information is requested.

The default is the local node

STATUS UPDATE

Displays information about SDR updaters that manage the specified RDF configuration and for the specified system.

```
STATUS UPDATE [ \node ] [ RDF-control-subvol ] [, DETAIL ]
```

node

node name of the system for which the SDR updater information is requested.

The default is the local node.

RDF-control-subvol

an RDF configuration in the format *node*[*suffix-character*] that replicates the SQL objects that are the target of DDL operations managed by the SDR updater.

If *RDF-control-subvol* is omitted all RDF configurations are assumed.

DETAIL

Displays the defines and the text of the last replicated DDL statement.

STOP MONITOR

Performs an orderly shutdown of the monitor process..

```
STOP MONITOR
```

STOP UPDATE

Stops an SDR updater.

```
STOP UPDATE [ RDF-control-subvol ]
```

RDF-control-subvol

an RDF configuration in the format *node*[*suffix-character*] that replicates the SQL objects that are the target of DDL operations managed by the SDR updater.

If *RDF-control-subvol* is omitted, all SDR updaters are stopped.

UNINSTALL SDR

Reverses the installation of SDR by unpreparing the NonStop SQL components that had been enabled for DDL replication when issuing the INSTALL SDR command.

You must be logged on as a member of the SUPER group to perform this command.

You must UNSINSTALL SDR before removing the SDR product files from your system.

```
UNINSTALL SDR
```

Utility Commands

ABEND MONITOR

Causes the monitor to stop and produce a saveabend file. Use this command only when you are required to supply information about a monitor problem and are instructed to do so by product support.

```
ABEND MONITOR [ * | process-name ]
```

*

(asterisk) stops all configured monitor processes.

process-name

specifies the name of the SDR monitor process to be stopped. The default is the current monitor for the session.

ALTER FILE

Utility command similar to the FUP ALTER command. Changes attributes of Enscribe files. Unlike the FUP ALTER command, the SDRCOM ALTER FILE command can alter a collection of files specified as a file set.

```
ALTER FILE file-set {, specification }

specification is

{ AUDIT | NO AUDIT                                                    }
{                                                                     }
{ AUDITCOMPRESS | NO AUDITCOMPRESS                                    }
{                                                                     }
{ BUFFERED | NO BUFFERED                                              }
{                                                                     }
{ CLEARONPURGE | NO CLEARONPURGE                                      }
{                                                                     }
{ CODE file-code                                                      }
{                                                                     }
{ LOCKLENGTH key-length                                               }
{                                                                     }
{ MAXEXTENTS size                                                     }
{                                                                     }
{ NOPURGEUNTIL day month year }                                       }
{                                                                     }
{ OWNER group-num,user-num                                            }
{                                                                     }
{ SECURE "rwep"                                                       }
{                                                                     }
{ SERIALWRITES | NO SERIALWRITES                                      }
{                                                                     }
{ VERIFIEDWRITES | NO VERIFIEDWRITES                                  }
{                                                                     }
{ RESETBROKEN                                                         }
```

file-set

TACL-style file name pattern specifying a collection of files.

specification

AUDIT | NO AUDIT

specifies whether TMF auditing is on. If NO is specified, auditing is off.

The audit mode is propagated to alternate key files.

AUDITCOMPRESS | NO AUDITCOMPRESS

specifies whether or not compression of audit records is occurring for this file.

BUFFERED | NO BUFFERED

specifies whether buffered writes are performed. If NO BUFFERED is specified, writes are not buffered.

The default is buffered mode for audited files and not buffered for non audited files.

CLEARONPURGE | NO CLEARONPURGE

erases disk free space when files are purged.

CODE *file-code*

numeric file code of the file. *file-code* is an integer between 0 and 65535. Codes 100 to 999 are reserved for use by HP.

LOCKLENGTH *key-length*

the byte count of the record key for generic locks. *key-length* is between 0 and the key length of the file.

MAXEXTENTS *size*

the maximum disk allocation extents. *size* is an integer between 16 and 978, where the maximum value depends on the free space in the file label.

NOPURGEUNTIL day-month-year

date after which a PURGE of the file is allowed. For example: 28 Feb. 2001

OWNER *group-num*, *user-num*

the user ID of the owner of the files. For example: 100,004.

RESETBROKEN

resets BROKEN flag in the file label for non audited files.

SECURE “rwep”

Guardian security string.

SERIALWRITES | NO SERIALWRITES

specifies whether serial writes to the mirrored disk are performed. If NO SERIALWRITES is specified, parallel writes are performed. The default is NO SERIALWRITES.

VERIFIEDWRITES | NO VERIFIEDWRITES

sets the mode of file writes: verified or not verified. The default is NO VERIFIEDWRITES.

COPY

Similar to the FUP COPY command with enhancements. Copies records from an input file to an output file.

```
COPY in-file, out-file [, copy-options ]

copy-options is:
{ control-options                                                    }
{ in-options                                                         }
{ out-options                                                        }
{ display-options                                                    }

control-options is:
{ COUNT num-records                                                  }
{                                                                    }
{ FIRST { ordinal-record-num |                                       }
{ { KEY {record-spec|key-value [, key-value]}|                       }
{ { key-spec ALTKEY key-value [, key-value ] }                       }
{                                                                    }
{ FROMLAST                                                           }
{                                                                    }
{ UPSHIFT                                                            }

in-options is:
{ BLOCKIN n                                                          }
{                                                                    }
{ COMPACT | NO COMPACT                                               }
{                                                                    }
{ COMP[ARELEN] n                                                     }
{                                                                    }
{ EBCDICIN                                                           }
{                                                                    }
{ EXACT                                                              }
{                                                                    }
{ RECIN n                                                            }
{                                                                    }
{ REVERSE                                                            }
{                                                                    }
{ REWINDIN | NO REWINDIN                                             }
{                                                                    }
{ SKIPMATCH                                                          }
{                                                                    }
{ TRIM trim-character                                                }
{                                                                    }
{ UNLOADIN | NO UNLOADIN                                             }
{                                                                    }
{ UNSTRUCTURED                                                       }
{                                                                    }
{ VARIN
                                                          ...more
```

```
out-options is:

{ BLOCKOUT n                                                    }
{                                                               }
{ EBCDICOUT                                                     }
{                                                               }
{ FOLD                                                          }
{                                                               }
{ PAD pad-character                                             }
{                                                               }
{ RECOUT n                                                      }
{                                                               }
{ REWINDOUT | NO REWINDOUT                                      }
{                                                               }
{ UNLOADOUT | NO UNLOADOUT                                      }
{                                                               }
{ UNSTROUT                                                      }
{                                                               }
{ UPDATE                                                        }
{                                                               }
{ VAROUT                                                        }

display-options

{ [O]CTAL                                                       }
{                                                               }
{ [D]ECIMAL                                                     }
{                                                               }
{ [H]EX                                                         }
{                                                               }
{ [A]SCII                                                       }
{                                                               }
{ BYTE                                                          }
{                                                               }
{ NO HEAD                                                       }
```

in-file

file containing data to be copied.

in-file can be a process, tape, terminal, or disk file. Disk files include edit files, Enscribe structured and unstructured files. If the *in-file* is omitted, the SDRCOM IN file is used.

out-file

file to receive data to be copied.

out-file can be a process, tape, terminal, a printer or disk file. Disk files include edit files, Enscribe structured and unstructured files. If the out-file is omitted, the SDRCOM OUT file is used.

control-options

COUNT *num-records*

the number of records or rows to be copied. If omitted, all records are copied.

FIRST *ordinal-record-num*

the starting record of the input file to copy. If omitted, the copy starts at the first record or row in the input file.

ordinal-record-num

the number of records or rows from the beginning of the file that are to be skipped. The first record in a file is record zero.

KEY { *record-spec* | *key-value* }

the primary key value for the starting record or row of a structured disk file.

record-spec is an integer on the range of 0 to 4294967295.

- for unstructured files, *record-spec* is the starting relative byte address.
- for relative files, *record-spec* is the starting record number
- for entry-sequenced files, *record-spec* is the *ordinal-record-number*

key-value applies only to key-sequenced files and specifies the approximate position of the starting record; *key-value is* specified as a string or as integer byte values in the range of 0 to 255.

The key-value is entered as follows:

"[" { string } [, { string }] "]"

{ 0:255 } [{ 0,255 }]

where the integers represent the byte values. To specify a list of strings, enclose each string in quotation marks separated by a comma and enclose the list in square brackets.

key-specifier

the alternate key tag (a 2-byte string or a 16-bit integer) designating the alternate key to be used for positioning.

ALTKEY *key-value* [, *key-value*]

the alternate key of the starting record or row. The format of *key-value* is described above.

UPSHIFT

convert lowercase characters to uppercase.

in-options

BLOCKIN *n*

number of bytes between 1 and 32767 in an input block that is requested in a single physical read operation. When BLOCKIN is not specified, the RECIN value is used. The default is device dependent: 80 bytes for terminal, 132 bytes for process and unstructured files.

COMPACT | NO COMPACT

zero length records should (or should not) be skipped when copied to the output file. The default is COMPACT.

COMPACT applies only for copying relative files

COMP[ARELEN] *n*

use generic positioning for compare length *n* on the record key (primary or alternate) specified in the FIRST KEY option. The compare length is between 1 and 255 and must be less than or equal to the key specified.

EBCDICIN

translate input characters from EBCDIC to ASCII.

EXACT

exact positioning on the record key (primary or alternate) specified in the FIRST KEY option.

FROMLAST

position on the last record in the key range specified in the FIRST KEY option.

RECIN *n*

the maximum number of bytes in an input record. *n* is between 1 and 4096. When RECIN is not specified, the BLOCKIN value is used with a maximum of 4096.

REVERSE

reads the input file from the starting record in reverse order.

REWINDIN | NO REWINDIN (*magnetic tapes only*)

input tape is rewound (or not rewound) when the EOF is read from the tape. If NO REWINDIN is specified, the tape remains positioned without rewinding.The default is REWINDIN. This option also applies to labeled tapes.

SHARE

the file is to be opened in shared exclusion mode. The default is protected.

SKIPMATCH

position the input file to the record immediately following the one whose key matches the specified key. The entire key must be supplied. If the file is not key sequenced, an error 46 is returned.

This option applies to key sequenced files only.

TRIM *trim-character*

delete any trailing character matching the *trim-character*. The character is specified in ASCII using quotation marks or as an integer in the range 0 to 255.

UNLOADIN | NO UNLOADIN (*magnetic tapes only*)

input tape is unloaded (or not unloaded) after the tape has been rewound. The default is UNLOADIN. This also applies to labeled tapes.

UNSTRUCTURED

open and access the input file using the unstructured option. This option can be used for Enscribe unstructured or structured files where the file structure is ignored.

VARIN

read variable length blocked records. These records can be produced by using the VAROUT COPY command option described below. Each record is preceded by a one word indicator that contains the record length in bytes.

out-options

BLOCKOUT *n*

number of bytes from 1 to 32767 in an output record. When BLOCKOUT is not specified, the RECOUT value is used. The default is device dependent: 80 bytes for terminal, 132 bytes for a process and unstructured files. If BLOCKOUT is greater than RECOUT, the output block is filled with RECOUT-value length records until the block contains BLOCKOUT-value bytes or the last output record is encountered.

EBCDICOUT

translate output characters from ASCII to EBCDIC.

FOLD

output records longer than the output record length will be divided into as many output records as needed to copy the entire record.

PAD *pad-character*

output records shorter than the output record length will be padded with *pad-character*. Specify *pad-character* as an ASCII character in quotation marks or as an integer in the range of 0 to 255.

RECOUT *n*

maximum length between 1 and 4096 of an output record.

REWINDOUT | NO REWINDOUT (*magnetic tapes only*)

output tape is rewound (or not rewound) after the copy operation has completed. If NO REWINDOUT is specified, the tape remains positioned without rewinding. The default is REWINDOUT. This option also applies to labeled tapes.

UNLOADOUT | NO UNLOADOUT (*magnetic tapes only*)

output tape is unloaded (or not unloaded) after rewinding. The default is UNLOADOUT. This option also applies to labeled tapes.

UNSTROUT

opens and writes the output file using the unstructured access option. This option is used for Enscribe unstructured files or structured files where the file structure is ignored.

UPDATE

records are updated rather than inserted into the output file. An error is returned if the record to update is not present in the file.

This option is valid for key sequenced files only. If the output file is not key sequenced, the option is ignored.

VAROUT

write variable length blocked records. Each record is preceded by a one word indicator that contains the record length in bytes.

Variable length records are word aligned in the output block. The last record in each block is followed by a terminator (-1) if there is space in the block.

The FOLD and PAD options are not supported when varout is specified.

display-options

Display options differ slightly from FUP display options: output data transformations (RECOUT, BLOCKOUT, PAD and so on) are applied before formatting the data. Thus, formatted data is displayed as the data would be written to a disk file.

[O]CTAL

display the output in octal and ASCII format.

[D]ECIMAL

display the output in decimal and ASCII format

[H]EX

display the output in hexadecimal and ASCII format

[A]SCII

display the output in ASCII format. This option is ignored if combined with OCTAL, HEX, DECIMAL or BYTE display options.

NO HEAD

omit the heading preceding each record when one of the *display-options* is specified.

COMMENT

Causes the command interpreter to ignore the remainder of the current line. COMMENT is not a multiline command or terminated with a semicolon and may not appear within the lines of a multiline command.

```
COMMENT any-text
```

In addition to the explicit COMMENT command, a pair of dashes (– –) causes the interpreter to ignore all remaining text on the current line. A dash-dash comment may appear within a multiline command.

DELAY

Suspends the execution of SDRCOM for the specified interval.

```
DELAY { n-centisecs | n SEC[ONDS] | n MIN[UTES] }
```

ENV

Displays the current setting of all environmental variables.

```
ENV
```

EXIT

Terminates the SDRCOM session. A CTRL–Y has the same effect.

```
EXIT
```

The EXIT command does not require a semi-colon and cannot be followed by any text.

FC AND!

Runs previous commands found in the command history. FC permits a command to be edited before execution. “!” runs a command without editing. The commands are not multiline commands and are not terminated with a semicolon.

```
FC [ integer | -integer | text ]

! [ integer | -integer | text ]
```

The desired command can be specified in one of four ways:

- The default is the immediately previous command.
- *integer* (positive) specifies the ordinal number of a command in the history buffer (See HISTORY).
- *-integer* (negative) specifies a relative command in the history buffer, with the most recent command having the value –1.
- *text* selects the most recent command that starts with the specified text.

FILEINFO

Displays information about Enscribe files.

```
FI[LEINFO] file-set [, DETAIL ]
or
FID file-set
```

file-set

is a TACL-style file name pattern specifying a collection of files. If *file-set* is not specified, the current subvolume is assumed.

DETAIL

displays detailed information. If not specified, SDRCOM displays one line of information per file in the file-set.

FID *file-set*

abbreviates FILEINFO DETAIL.

FILES

Displays the 8-character filename of the files in the file-set.

```
FILES file-set
```

file-set

is a TACL-style file name pattern specifying a collection of files. If *file-set* is not specified, the current subvolume is assumed.

HELP

Lists the help options for SDR commands, MONITOR commands and the SDR defines.

```
HELP
{ ALL                                                                }
{                                                                    }
{ SDR-COMMAND                                                        }
{                                                                    }
{ command [ DETAIL | EXAMPLE ]                                       }
{                                                                    }
{ SDR-DEFINES                                                        }
{                                                                    }
{ define [ USAGE ]                                                   }
{                                                                    }
{ GLOBALS                                                            }
{                                                                    }
{ MONITOR [ monitor-command ]                                        }
```

ALL

displays the list of all commands.

SDR-COMMANDS

requests the list of all SDR commands available.

command [DETAIL | EXAMPLES]

a SDR command. Help displays the syntax and description of the command.

Multi-word commands are entered with hyphens.

For example, to obtain help on START UPDATE type:

HELP start-update

If DETAIL is specified and if detail help information exists, a description of the command parameters is displayed in addition to the syntax.

If EXAMPLES is specified, Help displays examples of the command.

If neither DETAIL nor EXAMPLES is specified, Help displays both as possible subtopics, when applicable.

MONITOR [*monitor-command*]

displays the syntax and description of commands that can be sent to the SDR monitor process via SDRCOM. If no monitor command is specified, HELP displays a list of the commands as subtopics.

monitor-commands is one of: BACKUPCPU, LOG, SECURITY, STATUS or SWITCH or TRACE.

SDR-DEFINES

requests the display of the list of DEFINEs for which HELP is available.

define-name [USAGE]

is the name of the DEFINE. HELP displays the syntax and a description of the DEFINEs.

If USAGE is specified, Help displays information about the usage of the DEFINE, otherwise Help displays USAGE as a possible subtopic.

GLOBALS

requests a display of the list of all the possible global parameters that can be set for SDR using the ALTER command.

HISTORY

Lists the saved commands in the history buffer. These commands can be run using the FC or bang (!) commands.

```
HISTORY count
```

count

the number of commands to display. The default is 10. If fewer commands are in displays the opens by file name. It is the default.

detected deadlock. The command displays a list of transactions and processes that are participants in the deadlock and prompts the user to select either a process to abend or a transaction to abort to resolve the deadlock.

RESOLVE must follow the DEADLocks keyword on the command line.

LOG

Collects a history of the SDRCOM session to a file. LOG TO starts the logging process, and LOG STOP terminates the logging.

```
LOG { TO filename [CLEAR] | STOP }
```

TO *filename*

starts logging to *filename*.

filename may be a disk file, a printer, or another terminal. If the log file does not exist, SDRCOM creates the log file as an edit file.

CLEAR

clears the log file of existing data.

STOP

closes the current log file and stops logging.

MONITOR

The MONITOR command sends a command to the monitor process. Refer to Monitor Commands on page 6-34 for a list of valid monitor commands

```
MONITOR command-text
```

command-text

is all text up to the end of the current command line. A MONITOR command cannot be continued on multiple lines.

OBEY

Reads and runs a sequence of commands from another device. The commands are run serially until end-of-file is detected. An OBEY file may not contain an OBEY command.

```
OBEY filename
```

filename

identifies the file containing a sequence of SDRCOM commands.

OUT

Directs the output of the SDRCOM session to another file. Interactive terminal prompts will continue to appear on the original input device.

```
OUT [ filename ]
```

filename

identifies an output file and directs the SDRCOM session output to that file. If filename is omitted, output will be directed to the original process OUT file.

RESET

Changes all of the environmental variables to their original settings.

```
RESET
```

RUN[D]

Runs a program during a SDRCOM session. When the program terminates, the session resumes.

```
RUN[D] filename[/run-options/] [command]
```

filename

specifies a program file to be run.

run-options

specifies standard TACL process options, including the following:

CPU	EXTSWAP	LIB	NOWAIT	PRI
DEBUG	IN	MEM	OUT	SWAP
DEFMODE	INSPECT	NAME	PFS	TERM

command

specifies any command line to be passed to the process in the startup message.

The RUN[D] command is not a multiline command and is not terminated with a semicolon. Any semicolon is passed to the process as part of the startup command.

Refer to Table 6-4 for a list of standard programs, such as EDIT and SQLCI, that can be run by simply typing the program name. These commands are equivalent to specifying “RUN $SYSTEM.SYSTEM.” before the name and otherwise follow the complete RUN command syntax.

SEMICOLON

Toggles the use of semicolon (“;”) as a command termination character in SDRCOM.

```
SEMICOLON { ON | OFF }
```

{ ON | OFF }

ON sets the command termination character to semicolon (“;”). Using the semicolon allows the user to enter multiple commands on a single line or multiline commands without requiring the continuation character ampersand (“&”).

OFF removes the use of semicolon (“;”) as a command termination character.

The default is OFF.

STATUS MONITOR

Determines the status of the monitor process by sending the monitor inquiries.

```
STATUS MONITOR
```

TIME

Displays the current local time of day as obtained from the local Guardian timekeeping function. You may also request the local time as seen by another system on your Expand network.

```
TIME [ systemname ]
```

systemname

identifies an Expand system for which the local system time is desired.

VOLUME

Changes the default volume and/or subvolume for filename expansion. VOLUME can be abbreviated to V.

```
V[OLUME] [ volume | subvolume | volume.subvolume ]
```

volume

specifies a new default volume; does not alter the default subvolume.

subvolume

specifies a new default subvolume; does not alter the default volume.

Monitor Commands

Each of the following commands are recognized by the monitor process. To send a command to the monitor, use the MONITOR command.

EMSLEVEL

Sets the level of detail displayed in the EMS log.

```
MONITOR EMSLEVEL { CRITICAL | DETAIL | NORMAL | TRACE }
```

CRITICAL

displays only messages that are critical or that require operator intervention.

DETAIL

displays all EMS messages generated by the SDR monitor and SDR updaters.

NORMAL

displays essential informational messages, action messages and critical messages.

NORMAL is the default.

TRACE

displays low level activity for problem analysis and should be used only when advised by HP support.

LOG

Initiates or terminates a monitor activity log.

```
MONITOR LOG { TO file [CLEAR] | STOP }
```

TO *file* [CLEAR]

specifies the name of a log file and starts logging. CLEAR empties the file before logging starts.

If *file* already exists, information is appended to the end of the file unless CLEAR is specified. If *file* ends with a number (for example LOG001), additional files will be allocated with incrementing numbers when the first file becomes full.

STOP

terminates logging.

Activities that are logged include:

- Starting and stopping the log.
- Monitor opens and closes by requesting processes.
- Fault tolerance process events, such as a backup process takeover.
- Local CPU failures and reloads.
- Remote network status changes

Note. Information in the activity log is for informational and diagnostic purpose. It is not intended to be an official or reliable record for continuing management or control purposes. The content and format of the log is subject to change without notification.

STATUS

Displays information about various objects.

```
MONITOR STATUS [ LICENSE ]
```

LICENSE

requests information about the SDR license for the current system.

BACKUPCPU

The BACKUPCPU command specifies the cpu for the monitor backup process.

```
MONITOR BACKUPCPU [ cpu ]
```

cpu

specifies the cpu number where the monitor backup process should be started. If omitted, the backup process is terminated.

SWITCH

The SWITCH command causes the monitor primary and backup processes to exchange roles.

```
MONITOR SWITCH
```

A SQL DDL Statements

This appendix describes the SQL DDL statements that are replicated and, in some cases, specific information about how SDR translates them. References to catalogs, physical volumes, and WITH SHARED ACCESS are also discussed.

DDL Statements

Table A-1 lists the DDL statements processed by SDR with notes on special processing, where applicable.

Table A-1.

DDL Operation	Considerations and Processing Notes
ALTER CATALOG	
ALTER COLLATION	
ALTER INDEX	
ALTER TABLE	ALTER TABLE NO AUDIT is replicated. ALTER TABLE AUDIT is not replicated unless unaudited replication is selected by setting the global parameter UNAUDITEDDDL to ON.
ALTER VIEW	
COMMENT	
CREATE CATALOG	See discussion in paragraph Create Catalog on page 4-4.
CREATE COLLATION	This statement references an Enscribe edit file. Like all other secondary references, RDF must be configured to replicate the backup edit file name. The backup edit file should be created before executing the CREATE COLLATION statement.
CREATE CONSTRAINT	
CREATE SYSTEM CATALOG	Is not replicated. It must be performed as a local operation when SQL is first installed.
CREATE INDEX	
CREATE TABLE	CREATE TABLE LIKE references an existing table for column definitions. This table must be present on the backup system, with the name that RDF would use if it were replicating that table. If the table is partitioned, all partitions must be replicated by RDF.

DDL Operation	Considerations and Processing Notes
CREATE VIEW	A shorthand view is a SQL object that is neither audited nor unaudited. When a shorthand view is created on the primary system, SDR assumes it is audited and captures it just as it would an audited table creation. On the backup system, the SDR updater examines the tables referenced in the CREATE VIEW statement. If any of those tables are audited, SDR replicates the CREATE VIEW; otherwise, it considers the view to be unaudited and does not replicate it unless the UNAUDITEDDDL global parameter is set to ON.
DROP CATALOG	DROP processing depends on how RDF REPLICATEPURGE is set. See REPLICATEPURGE discussion in Section 3, Configuring SDR.
DROP CONSTRAINT	DROP processing depends on how RDF REPLICATEPURGE is set. See REPLICATEPURGE discussion in Section 3, Configuring SDR.
DROP INDEX	DROP processing depends on how RDF REPLICATEPURGE is set. See REPLICATEPURGE discussion in Section 3, Configuring SDR.
DROP TABLE	DROP processing depends on how RDF REPLICATEPURGE is set. See REPLICATEPURGE discussion in Section 3, Configuring SDR.
DROP VIEW	DROP processing depends on how RDF REPLICATEPURGE is set. See REPLICATEPURGE discussion in Section 3, Configuring SDR.
DUP	DUP is an SQLCI utility, not a standard DDL operation. It is not replicated by SDR.
PURGE	PURGE is an SQLCI utility that is converted to a DROP statement for each SQL object that is referenced by the file set. The individual DROP statements are replicated separately by SDR. References to programs and Enscribe files are not processed by SDR
SECURE	SECURE is an SQLCI utility that is converted by SQLCI to an ALTER statement for each SQL object that is referenced by the file set. The individual ALTER statements are replicated separately by SDR. References to programs and Enscribe files are not processed by SDR.
UPDATE STATISTICS	

SQL object in Statement

The target SQL object is the first SQL object name in the statement. In general (except for a CREATE), if the target SQL object does not exist, the SQL error is ignored and SDR proceeds to restart the RDF updaters. CREATE, of course, will not get a SQL error if the target SQL object does not exist.

If there is a secondary reference (such as partition, catalog, LIKE table, etc.) to a SQL object that does not exist, the resulting SQL error causes SDR to issue an EMS message and wait for operator intervention.

Catalog References in Statement

Catalog names, including default catalog names, must be translated to the backup file names using the RDF configuration. If you have an RDF mapfile and wish to map catalog names, then the filename $*vol.subvol.*CATALOG must be appropriately mapped.

SDR will translate catalog references even if catalog subvolumes are excluded in the RDF configuration. They will be translated to the same subvolume name on the backup volume. Note that the volume must be protected by RDF.

This does not apply to CREATE CATALOG and DROP CATALOG. Also, AUTOCREATECATALOG will not create a catalog when the catalog subvolume is excluded in the RDF configuration.

PHYSVOL References in Statement

Some DDL statements support the assignment of virtual volumes to a specific physical volume, or PHYSVOL. When SDR processes PHYSVOL references, the RDF configuration must contain the physical volumes in its volume table. Also, the virtual and physical volume assignments on the backup system must mirror the primary system. Otherwise, the translated SQL statement will fail.

If you do not require the replication of physical volume assignments on the backup system, set the SDR global option KEEPPHYSVOL to OFF.

WITH SHARED ACCESS Clauses

Some DDL statements support, or even require, a WITH SHARED ACCESS clause. The clause provides the ability to control the time at which a DDL operation is committed, because the commit can disrupt application access to the database. Since applications are not making transactional access to the backup database, SDR always specifies COMMIT WHEN READY.

A SQL DDL statement containing a WITH SHARED ACCESS clause will normally cause RDF to stop the updaters when it is safe to replicate the DDL statement on the backup system. The RDF updater stop is identified by the name of the SQL object.

But, when SDR is actively capturing DDL statements, the normal RDF updater stop is cancelled. Instead, SDR causes the RDF updaters to stop for all SQL DDL statements, and specifies an SDR marker file name.

B SDR EMS Messages

In each message description, the following information appears:

- Message number
- Message text
- Cause—the condition or error that produced the message
- Effect—the effect of the condition or error on the system
- Recovery—the steps required to recover from a reported error

Informational Messages

100

```
Monitor initializing
```

Cause. The SDR monitor process is initializing itself.

Effect. The monitor process is not ready to service requests until initialization is completed.

Recovery. Informational message only; no corrective action is needed.

101

```
Monitor started
```

Cause. The SDR monitor process has completed initialization.

Effect. The monitor is ready to service requests.

Recovery. Informational message only; no corrective action is needed.

102

```
Monitor stopped
```

Cause. The SDR monitor process is stopping due to a user request. The EMS message identifies the user that made the request.

Effect. The monitor stops

Recovery. Restart the monitor.

103.

```
Monitor switched to backup
```

Cause. The SDR monitor primary process or its CPU has failed. The monitor backup process has taken over.

Effect. The monitor is ready to service requests.

Recovery. Informational message. No action is required.

104

```
Monitor backup created
```

Cause. The SDR monitor primary process has created a backup process.

Effect. The monitor process is now fault-tolerant.

Recovery. Informational message. No action is required.

105

```
Monitor backup failed
```

Cause. The SDR monitor backup process has failed.

Effect. The monitor primary process continues operation, but the monitor process is no longer fault-tolerant.

Recovery. Determine the cause of the failure. If the cause can be corrected, use the MONITOR BACKUPCPU command to start the backup process.

106

```
Monitor initialization failed
```

Cause. The SDR monitor process failed to initialize itself.

Effect. The monitor process abends. SQL DDL operations will not be replicated.

Recovery. Report the error information to product support.

107

```
Monitor status information
```

Cause. The monitor process has displayed status information.

Effect. Status information is placed in the EMS log.

Recovery. Informational message only; no corrective action is needed.

151

```
SDR updater process starting
```

Cause. The monitor process has started an SDRUPDT process to perform replication. The message contains the remote node and the process name.

Effect. SDR replication processing for the specified node is active.

Recovery. Informational message only; no corrective action is needed.

152

```
SDR updater process stopping
```

Cause. The SDR monitor is stopping an SDRUPDT process. This normally occurs when an RDF configuration is aborted.

Effect. SDR will not be replicating SQL DDL for the specified

Recovery. No action required unless the updater was stopped by error.

155

```
SDR monitoring an RDF configuration
```

Cause. The SDR monitor has found an RDF configuration that it should monitor.

Effect. The SDR monitor will monitor the RDF configuration.

Recovery. Informational message only; no corrective action is needed.

156

```
SDR updater process has stopped
```

Cause. An SDRUPDT process has stopped.

Effect. If an error has occurred, the SDR monitor will restart the SDRUPDT process.

Recovery. Correct the error described in the message. The process will be automatically restarted.

157

```
A utility process used by SDR has died
```

Cause. A utility process started by SDR has unexpectedly terminated.

Effect. Replication of SQL DDL is interrupted.

Recovery. SDR should restart the process. If the problem persists, contact product support.

158

```
SDR Updater draining
```

Cause. The SDR monitor is draining an SDR updater process. This normally occurs when an RDF configuration is shut down. The updater may be processing a final SQL DDL operation.

Effect. SDR will not be replicating SQL DDL for the specified RDF environment.

Recovery. Do not issue a START RDF until the SDR updater has terminated.

175

```
NSA STOP UPDATE message sent
```

Cause. SDR has inserted an NSA STOP UPDATE message in the TMF audit trail containing the specified file name.

Effect. When processed by RDF, the SRU message will cause the RDF updaters to stop and the SDR updater to replicate the associated SQL DDL.

Recovery. Informational message only; no corrective action is needed.

176

```
SDR updater executing
```

Cause. An SDR updater process has started and is monitoring an RDF configuration.

Effect. The SDR updater will replicate SQL DDL for the RDF configuration.

Recovery. Informational message only; no corrective action is needed.

177

```
SDR updater stopped
```

Cause. An SDR updater has stopped, usually due to a shut down of an RDF configuration.

Effect. The SDR updater will no longer replicate SQL DDL for the RDF configuration.

Recovery. Informational message only; no corrective action is needed.

178

```
SDR started a process
```

Cause. SDR has started a utility process.

Effect. The utility process will perform services required by SDR.

Recovery. Informational message only; no corrective action is needed.

179

```
SDR write to Depot file
```

Cause. SDR has written a SQL DDL statement to the specified Depot file.

Effect. SDR will replicate the SQL DDL statement at some future time.

Recovery. Informational message only; no corrective action is needed.

180

```
RDF updater stopped
```

Cause. SDR has detected that RDF updaters have stopped and that the SQL DDL operation to be replicated was initiated on a remote primary system.

Effect. The local SDR ignores the RDF updater shut down. SDR on the backup system must replicate the SQL DDL and then restart the RDF updaters on all affected systems.

Recovery. Informational message only; no corrective action is needed.

181

```
RDF updater restarted
```

Cause. After SDR has replicated SQL DDL, it tells the RDF monitor to restart the RDF updaters.

Effect. RDF updating is restarted.

Recovery. Informational message only; no corrective action is needed.

182

```
DDL statement executed
```

Cause. SDR executed a SQL DDL statement on the backup system.

Effect. SQL DDL on a primary system is replicated on a backup system.

Recovery. Informational message only; no corrective action is needed.

183

```
SQL error display
```

Cause. A SQL error has occurred for a replicated statement and SDR displays the SQL error message.

Effect. The SQL error may indicate the cause of a previously reported problem.

Recovery. Informational message; consult the other messages to determine whether a problem needs to be corrected.

184

```
SQL DDL statement cancelled
```

Cause. The user has requested that a pending SQL DDL replication be cancelled.

Effect. The RDF updaters are restarted without replicating the SQL DDL.

Recovery. Informational message only; no corrective action is needed.

185

```
SQL DDL statement on hold
```

Cause. The user has requested that a pending SQL DDL replication be held.

Effect. The SDR updater will await further instructions to cancel or execute the SQL DDL.

Recovery. Use the SDRCOM to issue either the CANCEL or EXECUTE command.

186

```
SDR Depot file created
```

Cause. An SDR depot file has been created.

Effect. SDR will use the depot file to capture SQL DDL statements.

Recovery. Informational message only; no corrective action is needed.

187

```
SQL catalog created
```

Cause. Replication of a SQL DDL statement required the creation of a SQL catalog.

Effect. The required SQL catalog was created.

Recovery. Informational message only; no corrective action is needed.

188

```
Unprotected target
```

Cause. The target of a SQL DDL operation is not protected by the indicated RDF configuration.

Effect. The DDL operation is ignored and SDR proceeds.

Recovery. No corrective action is needed.

191

```
SQL DDL statement retrieved from Depot file
```

Cause. A SQL DDL statement was fetched from the backup depot file.

Effect. The SDR updater obtained a SQL DDL statement for replication.

Recovery. Informational message only; no corrective action is needed.

Critical Messages

402

```
SDR monitor intentionally abended
```

Cause. The monitor process has been intentionally abended to produce a saveabend file for problem diagnosis.

Effect. The primary monitor process abends. The backup process takes over and continues processing.

Recovery. Send the saveabend file to product support.

403

```
Invalid SDR license
```

Cause. An operation was attempted, but no license for that service has been installed.

Effect. The operation is rejected.

Recovery. Obtain and install the necessary license.

407

```
Version mismatch between SDR components
```

Cause. The Monitor process and the Runtime have incompatible versions.

Effect. The program abends.

Recovery. Verify that the runtime library and monitor are the same version. Contact product support for further assistance.

410

```
Error accessing object file
```

Cause. The object file is secured to prevent reading by the user id executing the process.

Effect. The process abends.

Recovery. Resecure the object file to allow read access.

411

```
Internal error detected (assertion failure)
```

Cause. A logic error has been detected by the process.

Effect. The process abends.

Recovery. Report the error information to product support.

412

```
Segment resize error
```

Cause. An attempt to resize the extended segment failed.

Effect. The process abends.

Recovery. The error number is provided in the message. Determine the reason for the failure and correct it.

413

```
Memory pool allocation error
```

Cause. An attempt to allocate memory in the extended segment failed.

Effect. The process abends.

Recovery. Since the product usually resizes the segment to satisfy memory requests, this error should not occur. Report this error to product support.

414

```
Allocate segment error
```

Cause. Allocation of the extended segment failed.

Effect. The process abends.

Recovery. The error number is provided in the message. Determine the reason for the failure and correct it.

415

```
TMF operation failure
```

Cause. A TMF operation needed by SDR failed.

Effect. The process abends.

Recovery. Either correct the TMF environmental problem or report the error information to product support.

419

```
Utility program process trap
```

Cause. A utility program encountered a program trap.

Effect. The process abends.

Recovery. Report the error information to product support.

420

```
Native utility program process trap
```

Cause. A native mode utility program encountered a program trap.

Cause. The process abends.

Recovery. Report the error information to product support.

433

```
System contains more logical CPUS than license permits
```

Cause. The referenced product is licensed for a maximum number of logical NSK processors (MAXCPUS). The number of active processors exceeds the license maximum.

Effect. Your license is not valid and you cannot use the product.

Recovery. Contact the license manager requesting a valid license with sufficient number of logical processors. Please provide your system serial number and order number.

450

```
Version mismatch between the SDR updater and the SDR monitor
```

Cause. The SDRUPDT process detected that the SDR monitor that started it has a different version.

Effect. The SDRUPDT process abends.

Recovery. Install the same version of SDR on the primary and backup systems.

451

```
SDR monitor on the backup system is not running
```

Cause. An SDRUPDT process on a remote system was not able to communicate with the SDR monitor process on the remote system.

Effect. The SDRUPDT process stops. No replication is performed.

Recovery. Start the monitor on the remote system.

452

```
Unable to send NSA UPDATE message
```

Cause. SDR could not insert a NSA STOP UPDATE message in the TMF audit trail. The error code is displayed in the message.

Effect. The RDF updaters will not stop at the proper time to replicate the SQL DDL. The SQL DDL will not be replicated.

Recovery. Report this error to product support.

453

```
Unable to access a file
```

Cause. SDR was unable to access the specified file. The error code is displayed in the message.

Effect. The SDR operation that required file access is aborted. If the SDR updater could not access an RDF configuration then it will not be able to monitor that RDF configuration. If SDR is unable to access a depot file, SQL DDL replication will fail.

Recovery. Correct the file access problem. SDR should retry the operation after the error condition has been resolved.

454

```
Unable to create a file
```

Cause. SDR was unable to create a depot, marker, or log file. The error code is displayed in the message.

Effect. SDR will terminate the operation that required the file. That could cause a primary SQL DDL operation to fail.

Recovery. Correct the file creation problem.

455

```
Unable to access a file label
```

Cause. SDR was unable to obtain the label of a SQL table. The error code is displayed in the message.

Effect. SDR will not be able to replicate a SQL DDL statement. SDR may have caused a SQL DDL statement to fail.

Recovery. Correct the file label access problem.

456

```
SDR updater or SDR runtime unable to authenticate a user
```

Cause. The SDR runtime or updater was unable to authenticate a user ID. SDR must perform certain operations using a different user id and invokes USER_AUTHENTICATE_.

Effect. A required operation fails.

Recovery. Correct the authentication problem. SDR should retry the operation.

457

```
SQL DDL statement not recognized
```

Cause. SDR was unable to parse the SQL DDL statement.

Effect. SDR can not translate a SQL DDL statement for execution on the backup system.

Recovery. Report this error to product support. Use a manual method to replicate the SQL DDL on the backup system and then issue the SDRCOM CANCEL command to proceed with RDF updating.

458

```
SDR disabled by user
```

Cause. A SQL DDL statement was executed when SDR was installed but DDL capture was disabled by the user.

Effect. The SQL DDL statement is neither captured nor replicated.

Recovery. To replicate SQL DDL, alter the SDR DDLCAPTURE global value to Enabled.

459

```
IOEdit error
```

Cause. SDR encountered an error when using the IOEDIT routines to communicate with EMSDIST.

Effect. SDR cannot monitor system events and will fail to replicate SQL DDL.

Recovery. This is an internal error. Contact product support.

460

```
SDR monitor down
```

Cause. The SDR monitor is down.

Effect. SDR cannot perform critical functions when the monitor is down.

Recovery. Use SDRCOM to start the SDR monitor.

461

```
SDR monitor cannot communicate with SDR updater
```

Cause. SDR monitor was unable to contact the SDR updater.

Effect. Orderly SDR processing may be affected.

Recovery. Contact product support.

462

```
SDR updater CPU failure
```

Cause. A processor failure caused a SDRUPDT process to terminate.

Effect. SDR updating is temporarily interrupted.

Recovery. No corrective action is needed. SDR will restart SDRUPDT in another processor.

463

```
SDR updater node failure
```

Cause. A backup node or communications failure has caused a SDRUPDT process to terminate.

Effect. SDR updating is temporarily interrupted.

Recovery. No corrective action is needed. SDR will restart the updater when the backup node is available.

464

```
SDR monitor unable to authenticate user
```

Cause. The SDR monitor was unable to authenticate a user ID. SDR must perform certain operations using a different user id and invokes USER_AUTHENTICATE_.

Effect. A required operation fails.

Recovery. Correct the authentication problem. SDR should retry the operation.

465

```
Error communicating with an EMS distributor
```

Cause. SDR encountered an error communicating with EMSDIST.

Effect. SDR may be unable to determine the state of an RDF configuration.

Recovery. Investigate and correct the error; contact product support if the problem persists.

466

```
File names inconsistent in SRU
```

Cause. The RDF configuration indicates that RDF updaters have stopped for different NSA STOP UPDATE records.

Effect. RDF updaters are stopped and cannot be restarted.

Recovery. Internal error. Contact product support.

467

```
RDF waiting on NSA STOP UPDATE not generated by SDR
```

Cause. RDF updaters have processed an NSA STOP UPDATE record that was not generated by SDR. Either SDR is not installed properly or has been disabled.

Effect. SDR replication of SQL DDL will not occur.

Recovery. Use standard RDF management of an NSA stop event.

468

```
Invalid NSA STOP UPDATE record
```

Cause. The file name in a NSA STOP UPDATE record is invalid.

Effect. The SQL DDL will not be replicated.

Recovery. Internal error. Contact product support.

469

```
Unable to access SDR depot file
```

Cause. The SDR depot file could not be accessed for the indicated file system error.

Effect. The SQL DDL will not be replicated.

Recovery. If the indicated error cannot be corrected, contact product support.

470

```
Missing record in the SDR depot file
```

Cause. RDF updaters received a NSA STOP UPDATE record, but the associated SDR SQL DDL could not be found in the depot file.

Effect. The SQL DDL will not be replicated.

Recovery. Internal error. Contact product support.

471

```
Invalid record in SDR depot file
```

Cause. The SDR depot file has invalid records.

Effect. The SQL DDL will not be replicated.

Recovery. Internal error. Contact product support.

472

```
Process creation failure
```

Cause. The monitor process has attempted to start a process but the operation failed. The message contains PROCESS_CREATE_ error and error detail code.

Effect. SDR replication is not active.

Recovery. Correct the cause of the failure and the process will be restarted.

473

```
Error communicating with a utility program
```

Cause. SDR was unable to communicate with the indicated utility program.

Effect. The effect depends on the utility program and the operation being attempted.

Recovery. If the error cannot be corrected, contact product support.

474

```
Unable to stop SDR updater
```

Cause. The SDR monitor is unable to STOP an SDR updater process.

Effect. SDR may be unable to replicate SQL DDL for the specified RDF environment.

Recovery. Manually terminate the named SDR Updater process.

475

```
SDR updater terminating
```

Cause. The SDR monitor that started a SDRUPDT process has terminated.

Effect. The SDR updater shuts down. Replication of SQL DDL is interrupted.

Recovery. Restart the SDR monitor to restart the SDRUPDT process.

476

```
RDF updater stopped and waiting on SDR
```

Cause. Updaters for the RDF configuration are stopped by a NSA STOP UPDATE record generated by SDR.

Effect. RDF updating is suspended awaiting the replication of SQL DDL by SDR.

Recovery. Investigate why SDR is not replicating the SQL DDL.

477

```
User transaction aborted by SDR
```

Cause. SQL DDL was executed under a user transaction; the user transaction was aborted by SDR.

Effect. The SQL DDL statement fails.

Recovery. Either change the application or change the SDR configuration to deal with user transactions in another way.

478

```
SDR holding on user transaction
```

Cause. SQL DDL was executed under a user transaction; SDR is holding the replication of the SQL DDL.

Effect. SDR and RDF updating are suspended awaiting user input.

Recovery. Review the SQL DDL and either EXECUTE or CANCEL it.

479

```
SDR holding on DROP TABLE
```

Cause. A SQL DROP statement was executed, but can not be replicated on the backup without permission. (RDF Replicate Purge is OFF)

Effect. SDR and RDF updating are suspended awaiting user input.

Recovery. Review the SQL DDL and either EXECUTE or CANCEL it.

480

```
SDR cannot obtain network lock
```

Cause. A SQL DDL statement references object names on multiple systems. To coordinate replication on each system, the SDR monitor must be executing on each system. Access to a required SDR monitor failed.

Effect. The SQL DDL operation fails.

Recovery. Correct the cause of the error. Confirm that the SDR monitor is executing on the referenced system.

481

```
Contention for SDR network lock
```

Cause. There is contention for a distributed SQL DDL lock.

Effect. SDR retries the lock request indefinitely; the SQL DDL operation does not complete until it succeeds.

Recovery. If the operation is not successful after many retries, investigate the status of the indicated SDR monitor.

482

```
SDR waiting for RDF SRU
```

Cause. A distributed SQL DDL statement is being replicated and SDR is waiting for a remote RDF configuration to process the NSA STOP UPDATE record and stop the updaters.

Effect. SDR and RDF updating are suspended awaiting remote RDF configurations to stop updating.

Recovery. No corrective action is needed. SDR will complete the operation when all RDF configurations have stopped updating.

483

```
Error accessing RDF configuration
```

Cause. An error occurred when attempting to access an RDF configuration file.

Effect. The RDF configuration will not be monitored for SDR replication.

Recovery. Contact product support.

484

```
RDF network configuration is incomplete
```

Cause. The RDF NETWORK configuration does not include all nodes required to replicate a distributed SQL DDL statement.

Effect. SDR and RDF updating are suspended awaiting user input.

Recovery. Either correct the RDF NETWORK configuration or CANCEL the SQL DDL statement. Note that you may also configure a NETWORK inside SDR.

485

```
Error accessing RDF network configuration
```

Cause. The RDF NETWORK configuration could not be accessed due to a file system error.

Effect. SDR and RDF updating are suspended awaiting error correction.

Recovery. Either correct the RDF NETWORK configuration or CANCEL the SQL DDL statement. Note that you may also configure a NETWORK inside SDR.

486

```
Target of SQL DDL operation missing
```

Cause. A DROP operation referenced SQL object that did not exist.

Effect. The DROP operation is ignored and SDR proceeds.

Recovery. No corrective action is needed.

487

```
SDR updater abend
```

Cause. An SDRUPDT process has abended.

Effect. The SDR monitor will restart the process. If the process continues to abend, no DDL replication will occur.

Recovery. Contact product support.

488

```
SDR updater lost
```

Cause. An SDRUPDT process has stopped or abended, but no process-stop message was received. This may be caused by a CPU failure.

Effect. The SDRUPDT process is restarted automatically.

Recovery. Informational message only; no corrective action is needed.

489

```
SQL DDL operation aborted
```

Cause. An SDR runtime error was detected and a transaction for a SQL DDL statement was aborted.

Effect. The SQL DDL statement fails.

Recovery. Correct the cause of the runtime error or change the SDR configuration to make SDR replication optional.

490

```
SDR required but impaired by environmental condition
```

Cause. SDR DDL capture is configured as REQUIRED, but some error is preventing SDR replication.

Effect. The SQL DDL statement fails.

Recovery. Correct the cause of the error or change the SDR configuration to make DDL capture enabled but not required.

491

```
SDR updater in hold state
```

Cause. A SQL DDL statement is ready to be replicated, but the SDR updater is in the hold state.

Effect. SDR and RDF updating are suspended awaiting user input.

Recovery. Either EXECUTE or CANCEL the SQL DDL statement. You may also RELEASE the updater from the hold state.

492

```
File translation error
```

Cause. It was not possible to translate a primary SQL object name to a corresponding name on the backup. Usually due to missing volumes in the RDF configuration.

Effect. SDR and RDF updating are suspended awaiting user input.

Recovery. Determine the missing volume definition and correct the RDF configuration. Otherwise, CANCEL the SQL DDL replication.

C Testing SDR

Overview

Testing of SDR is dependent on the resources available. If you have a separate testing environment with RDF configured between two development and or testing nodes, then there is less concern that SDR testing might impact production activities. But, many installations have only two systems: the production database and backup database nodes, with the backup used for testing and development. For this reason, SDR supports a special testing configuration, so that SDR can be tested on the production node, without impacting the production work.

The SDR testing capability supports parallel (production and test) RDF configurations. This capability allows the user to enable SQL for SDR but keeps DDL capture disabled, except for the specific users and programs that perform test DDL operations.

This testing capability is used to avoid replicating operations stored in the TMF audit trail that affect the production RDF. Such operations include updates to the SDRDEPOT files and the insertion of STOP-RDF-UPDATE messages. The production RDF can be isolated from DDL operations that are replicated by the test RDF.

This appendix describes how RDF and SDR can be configured for testing.

Setting up the Test Environment

1. It is simpler if the test RDF is configured to replicate separate volumes from the production RDF. The testing facility does not isolate RDF configurations that share volumes.
   If, through the use of INCLUDE and EXCLUDE options, the test and production RDF configurations share a volume, you will see specific messages in the EMS log on the backup system that can be safely ignored. See the description of the messages in paragraph ZRDF-EVT-Msg-700, ZRDF-EVT-Msg-705 on page 2-11.
2. Install SDR on both primary and backup systems, as described in Section 2, Installing SDR, up to the step Enable SQL for DDL Replication.
3. In SDRCOM, set the global parameter DDLCAPTURE to TESTING, on the primary and backup systems:

   ```
   ALTER DDLCAPTURE TESTING
   ```

4. In SDRCOM, enable SQL for SDR:

   ```
   INSTALL SDR
   ```

5. To prevent accidental updates, turn off replication by setting the RDFCONFIG global to NOREPLICATE for all RDF configurations on the primary and backup systems. In SDRCOM

   ```
   ALTER RDFCONFIG DEFAULT NOREPLICATE
   ```

Testing

To execute testing scenarios:

1. DDLCAPTURE TESTING disables the capture SQL DDL operations, unless one of 2 specific enablers is present:
   - the CLASS MAP DEFINE =_SDR_TEST DEFINE is set prior to executing SQLCI or any program performing SQL DDL or,
   - an empty file named ENABLSDR is present in the default subvolume. In this case, the default subvolume refers to the default subvolume for the process creator user id, not the current subvolume set by the VOLUME command.

   To activate one of the enabling methods:
   - Add a DEFINE for =_SDR_TEST when executing DDL operations in SQLCI or running any program performing SQL DDL:

     ADD DEFINE =_SDR_TEST, CLASS MAP, file $DUMMY

   or
   - Create an empty file named ENABLSDR in your default subvolume.
2. Enable DDL replication for the test RDF configuration. In SDRCOM:

   ALTER RDFCONFIG *test-RDF-control-subvolume* AUTOMATIC

You are now ready to execute SQLCI or any application performing SQL DDL operations.

Moving to production

Once the testing of SDR has completed, the following configuration changes will activate DDL replication for all RDF configurations. In SDRCOM:

1. ALTER DDLCAPTURE ENABLED (or the higher level REQUIRED)
2. RESET GLOBAL RDFCONFIG

Index

Symbols

A

C

D

E

I

M

N

R

S

T

U

V

Z