

# CUDB Node Commands and Parameters

## User Guide

## **Copyright**

© Ericsson AB 2016–2018. All rights reserved. No part of this document may be reproduced in any form without the written permission of the copyright owner.

## **Disclaimer**

The contents of this document are subject to revision without notice due to continued progress in methodology, design and manufacturing. Ericsson shall have no liability for any error or damage of any kind resulting from the use of this document.

## **Trademark List**

All trademarks mentioned herein are the property of their respective owners. These are shown in the document *Trademark Information*.



# Contents

<b>1</b>	<b>Introduction</b>	<b>1</b>
1.1	Purpose and Scope	1
1.2	Revision Information	1
1.3	Target Groups	8
1.4	Typographic Conventions	8
<b>2</b>	<b>CUDB Commands</b>	<b>9</b>
2.1	CUDB Command	10
2.2	Management Commands	10
2.2.1	cudbAnalyser	10
2.2.2	cudbApplicationCounters	12
2.2.3	cudbApplyConfig	14
2.2.4	cudbCheckConsistency	14
2.2.5	cudbCheckReplication	17
2.2.6	cudbCollectInfo	19
2.2.7	cudbConsistencyMgr	22
2.2.8	cudbDataBackup	27
2.2.9	cudbDbDiskManage	30
2.2.10	cudbDsgMastershipChange	31
2.2.11	cudbDsgProvisioningManage	35
2.2.12	cudbGetLogs	37
2.2.13	cudbHaState	38
2.2.14	cudbLdapFeRestart	39
2.2.15	cudbManageBCServer	42
2.2.16	cudbManageLibDataDist	43
2.2.17	cudbManageMsgSrvServer	44
2.2.18	cudbManageStore	46
2.2.19	cudbPmJobReload	52
2.2.20	cudbPrepareStore	53
2.2.21	cudbReallocate	57
2.2.22	cudbReconciliationMgr	64
2.2.23	cudbRemoteTrust	65
2.2.24	cudbResumeProvisioningNotification	67
2.2.25	cudbServiceContinuity	68
2.2.26	cudbSwBackup	69
2.2.27	cudbSwVersionCheck	73
2.2.28	cudbSystemDataBackupAndRestore	77
2.2.29	cudbSystemStatus	82
2.2.30	cudbTakeAllMasters	88
2.2.31	cudbUnitDataBackupAndRestore	89



2.2.32	cudbUpdateUserInfo	92
2.3	Internal Commands	92
2.3.1	cudbAppCheckerManager	92
2.3.2	cudbBCServersRestart	92
2.3.3	cudbCopyAclsConfFile	92
2.3.4	cudbEvipConfigExtension	92
2.3.5	cudbEvipEncapsulator	93
2.3.6	cudbExecuteAllBlades	93
2.3.7	cudbFollowLdapfeLogs	93
2.3.8	cudbManageDsGroup	93
2.3.9	cudbManageNode	93
2.3.10	cudbManageSite	93
2.3.11	cudbMpstat	93
2.3.12	cudbOomConfigurator	93
2.3.13	cudbParallelCommandRun	93
2.3.14	cudbRemoveNode	93
2.3.15	cudbSdpInfo	94
2.3.16	cudbSetDsgMaster	94
2.3.17	cudbSetPartitionStatus	94
2.3.18	cudbTpsStat	94
<b>Glossary</b>		<b>95</b>
<b>Reference List</b>		<b>96</b>



# 1 Introduction

This document describes the Ericsson Centralized User Database (CUDB) commands and command options, along with a complete reference on their options.

## 1.1 Purpose and Scope

This document describes the CUDB commands and command options with a complete reference on their options, syntax, output and some examples. The following topics are out of the scope of this document:

- Linux Operating System (OS) commands. Refer to [LDE Management Guide](#) for information on the Linux Distribution Extension (LDE) commands.
- The process of updating the CUDB Configuration Model. Refer to [CUDB Node Configuration Data Model Description](#) for the steps of updating the configuration model.

Most of the commands provide the output by means of logging information. Refer to [CUDB Node Logging Events](#) for more details about logging events in CUDB.

## 1.2 Revision Information

### Rev. A

This document is based on 1/1553-CSH 109 067/9, and contains the following changes:

- Structural and content rearrangement throughout the document.
- Virtualization terminology updates throughout the document.
- Removed obsolete information.
- [CUDB Commands](#) on page 9: Updated the general description of commands and the available user groups.
- [cudbAnalyser](#) on page 10: Updated syntax and examples of use of the `cudbAnalyser` command.
- [cudbApplicationCounters](#) on page 12: Updated syntax information of `cudbApplicationCounters`.



- [cudbApplyConfig](#) on page 14 and [cudbApplyConfig](#) on page 14: Updated sections with information on command deprecation. Updated output in examples of use.
- [cudbCheckConsistency](#) on page 14: Updated syntax, list of commands option, and examples of use.
- [cudbCheckReplication Configuration File](#) on page 18: Updated syntax of the `--behind-limit` option of the `cudbCheckReplication` command.
- [cudbConsistencyMgr](#) on page 22: Updated syntax and command option descriptions of the `cudbConsistencyMgr` command.
- [cudbDataBackup](#) on page 27: Updated syntax and command option information of `cudbDataBackup`.
- [cudbDbDiskManage](#) on page 30: Updated syntax and command option descriptions of the `cudbDbDiskManage` command.
- [cudbDsgMastershipChange](#) on page 31: Updated the command description, syntax, `--force` option description, and output examples of the `cudbDsgMastershipChange` command.
- [cudbDsgProvisioningManage Syntax](#) on page 35: Updated syntax of `cudbDsgProvisioningManage` command.
- [cudbLdapFeRestart](#) on page 39: Updated the command syntax, and updated the output examples of the `cudbLdapFeRestart` command.
- [cudbManageBCServer](#) on page 42: Updated syntax of the `cudbManageBCServer` command. Removed `-help` command option.
- [cudbManageStore Command Options](#) on page 46: Updated kickstart action of `cudbManageStore` command. Action is no longer restricted. Updated value of `-d | --ds <dsId>` command option.
- [cudbPrepareStore Command Options](#) on page 54: Updated value of `-d | --ds <dsId>` command option.
- [cudbReallocate Command Options](#) on page 58: Updated information regarding the `--nthreads` command.
- [cudbReallocate Output](#) on page 60, [cudbSwBackup Command Options](#) on page 70, [cudbSwBackup Examples of Use](#) on page 71, and [cudbEvipConfigExtension](#) on page 92 : Added information related to removed PLDB nodes.
- [cudbRemoteTrust](#) on page 65
- [cudbRemoteTrust Syntax](#) on page 65: Updated command.
- [cudbRemoteTrust Command Options](#) on page 66: Updated the note of a command and added a new command with a note.: Added a new paragraph.



- [cudbSwBackup](#) on page 69: Updated command syntax, and updated the examples of use of the cudbSwBackup command.
- [cudbSystemDataBackupAndRestore](#) on page 77 and [cudbSystemDataBackupAndRestore Output](#) on page 79: Updated information on command execution. Updated example information.
- [cudbSystemStatus Examples of Use](#) on page 84: Added 0AM1 to LDAP counter in the example.
- [cudbUnitDataBackupAndRestore](#) on page 89: Updated command syntax, and updated the output example of the cudbUnitDataBackupAndRestore command.
- [cudbUpdateUserInfo](#) on page 92: Updated with information on deprecation.

## Rev. B

Other than editorial changes, this document has been revised as follows:

- [cudbDsgMastershipChange Syntax](#) on page 32: Updated Format 1 and 2 with [ --no-prompt ].
- [cudbDsgMastershipChange Command Options](#) on page 32: Updated --force command.
- [cudbLdapFeRestart Syntax](#) on page 40: Updated command.
- [cudbRemoteTrust Command Options](#) on page 66: Updated network and VIP names.
- [cudbSystemDataBackupAndRestore Syntax](#) on page 77: Updated Format 3 and 4 with [ --no-prompt ].
- [cudbSystemDataBackupAndRestore Command Options](#) on page 78: Updated --restore <backup\_path> command.
- [cudbSystemDataBackupAndRestore Output](#) on page 79: Added warning to example.
- [cudbMpstat](#) on page 93: Moved and updated section, cudbMpstat command is available for Ericsson personnel only.
- Added information relevant to the Key Performance Indicators (KPIs).

## Rev. C

Other than editorial changes, this document has been revised as follows:

- [cudbSystemDataBackupAndRestore Syntax](#) on page 77: Removed -R This command option has been removed.



- [cudbSystemDataBackupAndRestore Command Options](#) on page 78: Removed -R This command option has been removed.
- [cudbSystemStatus Syntax](#) on page 82: Added `cudbSystemStatus -B` command.
- [cudbSystemStatus Command Options](#) on page 83: Added `-B | --new-bc-status` command and updated `-b | --bc-status` command description. Updated `cudbSystemStatus -B` command in note.
- [cudbSystemStatus Output](#) on page 84: Updated output description.
- [cudbSystemStatus Examples of Use](#) on page 84: Updated output example for Checking BC clusters.

#### Rev. D

Other than editorial changes, this document has been revised as follows:

- Updated Ericsson personnel information.

#### Rev. E

Other than editorial changes, this document has been revised as follows:

- [cudbTakeAllMasters Requisites](#) on page 88: Updated requisites.

#### Rev. F

Other than editorial changes, this document has been revised as follows:

- [cudbManageBCServer Syntax](#) on page 42: Added `-restart -no_check`.
- [cudbManageBCServer Command Options](#) on page 43 and [cudbSystemStatus Output](#) on page 84: Updated the terminology of BC cluster.
- [cudbManageBCServer Command Options](#) on page 43: Added a new command option, `-restart -no_check`, to the list.
- [cudbSystemStatus Command Options](#) on page 83: Updated `-r` option, which is now deprecated.
- [cudbSystemStatus Examples of Use](#) on page 84: Added new list item with examples in case there are disabled node(s) and/or disabled DS(s) in the system.
- [cudbManageNode](#) on page 93: New section describing the `cudbManageNode` command.





## Rev. G

Other than editorial changes, this document has been revised as follows:

- [cudbCollectInfo Requisites](#) on page 20: Added restrictions for the `cudbCollectInfo` command.
- [cudbCollectInfo Command Options](#) on page 20: Added caution regarding `-a`, `-d`, and `-n` options.
- [cudbCollectInfo Output](#) on page 21: Added example output messages and a note.
- [cudbCollectInfo Examples of Use](#) on page 22: Added example for performing all log collection actions on all nodes.
- [cudbHaState Output](#) on page 38: Updated the output of the `cudbHaState` command.
- [cudbLdapFeRestart](#) on page 39: Updated the description, syntax and command options, and added a warning.
- [cudbSystemStatus Examples of Use](#) on page 84: Updated the output of the `cudbSystemStatus` command.
- [cudbManageMsgSrvServer](#) on page 44: Added new section.

## Rev. H

Other than editorial changes, this document has been revised as follows:

- [cudbCollectInfo Output](#) on page 21: Updated output.
- [cudbPmJobReload](#) on page 52: Updated `cudbPmJobReload` command, which is now deprecated.
- [cudbReallocate Output](#) on page 60: Updated note with command used for a particular DSG.
- [cudbSwBackup Syntax](#) on page 70: Updated syntax with `cudbSwBackup -U | --Unschedule '<cron expression>'`.
- [cudbSwBackup Command Options](#) on page 70: Added `-U | --Unschedule <cron expression>` command option and a note.

## Rev. J

Other than editorial changes, this document has been revised as follows:

- [cudbDsgProvisioningManage](#) on page 35: Updated the list of what to consider when using `cudbDsgProvisioningManage` command.



- [cudbReallocate](#) on page 57: Updated cudbReallocate command description.
- [cudbReallocate Syntax](#) on page 58: Updated syntax with [-f | --force].
- [cudbReallocate Command Options](#) on page 58: Updated command description, -n | --nthreads is now restricted for Ericsson personnel. Updated the list of default values for -n | --nthreads configuration, and note. Added -f | --force command option.
- [cudbReallocate Output](#) on page 60: Updated list of example output messages.
- [cudbReallocate Examples of Use](#) on page 63: Added subsections [How to Check the Occupancy Percentage and Memory Level of All DSGs](#) on page 63 and [How to Force Reallocation if One DSG is Blocked for Provisioning of DEs](#) on page 63.
- [cudbSystemStatus Command Options](#) on page 83: Updated note.

#### Rev. K

Other than editorial changes, this document has been revised as follows:

- Updated Ericsson personnel information.

#### Rev. L

Other than editorial changes, this document has been revised as follows:

- [cudbDsgMastershipChange](#) on page 31: Updated description.
- [cudbDsgMastershipChange Output](#) on page 32: Updated with Automatic Mastership Change (AMC) behavior.
- **Section 2.3.10 cudbLicensingTool**: Removed section.

#### Rev. M

Other than editorial changes, this document has been revised as follows:

- [cudbSwBackup Output](#) on page 71: Updated output information.
- [cudbSwBackup Examples of Use](#) on page 71: Updated output information.
- [cudbManageMsgSrvServer](#) on page 44: Updates section title and cudbManageMsgSrvServer command.
- [cudbManageMsgSrvServer Syntax](#) on page 45: Removed **cudbManageMsgSrvServer h** from the list of formats.



- [cudbManageMsgSrvServer Command Options](#) on page 45: Removed h from the list of command options.
- [cudbManageMsgSrvServer Examples of Use](#) on page 45: Updated **cudbManageMsgSrvServer status** command.
- [cudbAppCheckerManager](#) on page 92: Added subsection.

## Rev. N

Other than editorial changes, this document has been revised as follows:

- [cudbCheckConsistency Command Options](#) on page 15: Updated the **object-tables** command in the list of commands.

## Rev. S

Other than editorial changes, this document has been revised as follows:

- [cudbApplyConfig](#) on page 14: Removed `init` command option.
- [cudbManageDsGroup](#) on page 93: Added section.
- [cudbRemoveNode](#) on page 93: Added section.

## Rev. T

Other than editorial changes, this document has been revised as follows:

- Structural and content rearrangement throughout the document: Requisites, Syntax, Command Options and Output are now directly under relevant command chapter. Command name concatenated to subsection titles.
- [cudbServiceContinuity](#) on page 68: Updated command description. Updated list of requisites. Updated command output.
- [cudbSwBackup Examples of Use](#) on page 71: Removed obsolete information.
- [cudbSystemDataBackupAndRestore Syntax](#) on page 77: Updated Format (1) with `[-o | --copy-over-oam-vip]`.
- [cudbSystemStatus Command Options](#) on page 83: Added `-o | --copy-over-oam-vip` command option.
- [cudbUnitDataBackupAndRestore Syntax](#) on page 90: Updated Format (1) and Format (2) with `-o | --copy-over-oam-vip`.
- [cudbTakeAllMasters](#) on page 88: Updated command description. Updated list of requisites. Updated command output.



- [cudbUnitDataBackupAndRestore Command Options](#) on page 90: Added -o | --copy-over-oam-vip command option.
- [cudbSetPartitionStatus](#) on page 94: option -ps is available for all system administrators and operators.

### Rev. U

Other than editorial changes, this document has been revised as follows:

- [cudbApplyConfig](#) on page 14: Updated Ericsson personnel information.
- [cudbUpdateUserInfo](#) on page 92: Updated Ericsson personnel information.

## 1.3 Target Groups

This document is intended for system administrators and operators who are familiar with Linux systems. Users of this document must possess moderate knowledge of the CUDB system and its Operation and Maintenance (OAM) procedures.

## 1.4 Typographic Conventions

Typographic Conventions can be found in the following document:

- [Typographic Conventions](#)



## 2 CUDB Commands

The following sections list and describe all CUDB commands that can be used within a Secure Shell (SSH) session in the CUDB nodes belonging to a CUDB system. The commands are grouped into the following categories:

- **Management commands:** These commands can be executed both by operators and Ericsson personnel. See [Management Commands](#) on page 10 for the alphabetical list and description of these commands.

**Note:** Some of the options of the management commands can be executed by Ericsson personnel only. These options are indicated with the "Restricted for Ericsson personnel" phrase.

- **Internal commands:** These commands can be executed by Ericsson personnel only. See [Internal Commands](#) on page 92 for the alphabetical list of these commands.

The commands in this document can be executed from any CUDB node, unless specific conditions or restrictions are specified in the *Requisites* section of the related command.

For each command, the following information is included:

- Description and execution requisites
- Use and Options:
  - Options and their default values
  - Syntax of use
  - Output
- Examples of use

The commands described in this document can be used by connecting to any CUDB node through an SSH interface, and getting authenticated by the system. The commands can be executed with the `cudb0operator` or `cudbadmin` users through the `sudo` Linux command. The available rights of these users are as follows:

- The `cudb0operator` user can execute commands that provide information about the CUDB status without affecting the node or the system. The *Requisites* subsections of the commands always state if `cudb0operator` users can run the specific command. For more information about these users and the associated security groups, refer to [CUDB Security and Privacy Management](#) and to [CUDB Users and Passwords](#).



- The `cudbadmin` user can execute all the commands that the `cudbOperator` user can, and can also run commands that execute actions which may affect system status, behavior, or performance. Use of this user therefore requires OAM skills.

By default, the location of the commands described in the document is set on the `PATH` system variable during installation time.

Several commands offer a debug option. This option activates the `DEBUG` log level, whose `DEBUG` log messages are stored in the same location as the log messages belonging to other levels. The location of the log messages for each component are listed in [CUDB Node Logging Events](#).

**Note:** The debugging option of the commands is used only when required by Ericsson personnel for troubleshooting purposes. This is because the amount of logging information can be very high during debugging, which can alter system working conditions.

All commands include a help option as well, used to provide information on the command and its available options.

For further information on CUDB system administration procedures, refer to [CUDB System Administrator Guide](#).

## 2.1 CUDB Command

## 2.2 Management Commands

Commands in this section can be executed both by operators and Ericsson personnel.

### 2.2.1 `cudbAnalyser`

The `cudbAnalyser` command performs a log analysis on the logs collected by `cudbGetLogs`. For more information about this command, refer to [CUDB Logchecker](#).

#### `cudbAnalyser` Requisites

`cudbOperator` users have no access to run this command.

#### `cudbAnalyser` Syntax

- Format (1): `cudbAnalyser -l | --logfile <logfile> -p | --previous-logfile <previous-logfile> [-s | --save-counter] [-q | --quiet] [-d | --debug] [-S | --send-alarm] [-w | --weight-threshold <threshold>]`



- Format (2): `cudbAnalyser -a | --auto-check [-s | --save-counter] [-q | --quiet] [-d | --debug] [-S | --send-alarm] [-w | --weight-threshold <threshold>]`
- Format (3): `cudbAnalyser -h | --help`

### cudbAnalyser Command Options

The following command options can be used:

- `-h | --help`: Displays command help message and exits.
  - `-l <logfile> | --logfile <logfile>` : Points to the newer log file.
  - `-p | --previous-logfile <previous-logfile>` : Points to an older logfile.
- Note:** The timestamp of `<logfile>` must be newer than `<previous-logfile>` to receive a correct differential analysis.
- `-s | --save-counter`: Saves a counter under `/home/cudb/monitoring/preventiveMaintenance/cudb_analyser_error_counter.log` which shows the number of failing checks. This is needed for ESA integration.
  - `-a | --auto-check`: The script checks the last two log files automatically. The use of the `--logfile` and `--previous-logfile` switches is not necessary in this case.
  - `-q | --quiet`: Makes the error printouts less verbose.
  - `-d | --debug`: Restricted for Ericsson personnel.
  - `-S | --send-alarm`: Sends an alarm to ESA, based on the results.
  - `-w | --weight-threshold <threshold>` : Prints errors if WEIGHT is greater or equal to the weight threshold (default value is 2). When performing a detailed system audit, set this value to 0 to print more errors.

### cudbAnalyser Output

#### Example 1

```
Started on ACTIVE SC...
[INFO] Checking files: //home/cudb/monitoring/preventiveMaintenance/ \
/CUDb_50_201307231233.log //home/cudb/monitoring/preventiveMaintenance/ \
/CUDb_50_201307231231.log logfile versions:0.0.60/0.0.60
[ERROR] OS: network stat shows errors (Severity: Minor)
> CUDb50 PL0 eth1/statistics/rx_dropped 27 K packets
> CUDb50 PL1 eth1/statistics/rx_dropped 27 K packets
> CUDb50 DS1_0 eth1/statistics/rx_dropped 27 K packets
> CUDb50 DS1_1 eth1/statistics/rx_dropped 27 K packets
```

#### 2.2.1.1

### cudbAnalyser Examples of Use

- `<CUDb_node_prompt> cudbAnalyser --help`



- <CUDB\_node\_prompt> **cudbAnalyser --auto-check --save-counter**
- <CUDB\_node\_prompt> **cudbAnalyser -l /home/cudb/monitoring/preventiveMaintenance/CUDB\_28\_201108190300.log -p /home/cudb/monitoring/preventiveMaintenance/CUDB\_28\_201108160300.log --send-alarm**
- <CUDB\_node\_prompt> **cudbAnalyser -l /home/cudb/monitoring/preventiveMaintenance/CUDB\_28\_201108190300.log -p /home/cudb/monitoring/preventiveMaintenance/CUDB\_28\_201108160300.log --save-counter**
- <CUDB\_node\_prompt> **cudbAnalyser --logfile /home/cudb/monitoring/preventiveMaintenance/CUDB\_28\_201108190300.log --previous-logfile /home/cudb/monitoring/preventiveMaintenance/CUDB\_28\_201108190300.log**

## 2.2.2

### cudbApplicationCounters

The `cudbApplicationCounters` command executes application counters defined by the counter configuration file. Refer to [CUDB Application Counters](#) for more information.

#### cudbApplicationCounters Requisites

`cudbOperator` users have no access to run this command.

#### cudbApplicationCounters Syntax

```
cudbApplicationCounters -C | --Counter-file  
<counter_configuration_file> -U | --Unique <uniqueNumber> [-f | --  
file-config <configuration_file>] [-u | --user-facility  
<facility_level>] [-l | --licensing] [-D | --Debug] [-h | --help]
```

#### cudbApplicationCounters Command Options

The following command options can be used:

- **-C | --Counter-file <counter\_configuration\_file>** : Selects the file which contains the information required for generating counters and storing them in Processing Layer Database (PLDB).
- **-f | --file-config**: Sets the XML file with the data of the entire CUDB system.
- **-U | --Unique**: Unique number greater than 0 between process instances.
- **-u | --user-facility**: Restricted for Ericsson personnel.
- **-l | --licensing**: Restricted for Ericsson personnel.





- -D | --Debug: Restricted for Ericsson personnel.
- -h | --help: Displays command help message and exits.

### **cudbApplicationCounters Output**

The command output shows the command version and OAM library version when executed successfully:

```
cudbApplicationCounters Ver.(1.3.1) OAM Lib.Version (1.1.0)
```

#### **2.2.2.1**

### **cudbApplicationCounters Examples of Use**

#### **Steps**

1. Connect to the CUDB node with the following command:  

```
<CUDB_node_prompt> ssh <admin_user>@<CUDB_Node_IP_Address>
```
2. Connect to the System Controller (SC) of the node where the PLDB master is located:  

```
<CUDB_node_prompt> ssh <admin_user>@<CUDB_Node_OAM_IP_Address>
```
3. Execute the cudbApplicationCounters command as follows:  

```
<CUDB_node_prompt> cudbApplicationCounters -C /cluster/home/
cudb/oam/performanceMgmt/appCounters/config/<app_counter>.conf
-U 1 -u LOCAL1
```

The expected output must look similar to the below example:

```
cudbApplicationCounters Ver.(1.3.1) OAM Lib.Version (1.1.0)
```



## 2.2.3 cudbApplyConfig

---

### Attention!

The use of this command is restricted to installation only. To make configuration model changes persistent in regular operation and maintenance procedures, use the administrative operation `applyConfig`. Refer to [CUDB Node Configuration Data Model Description](#) for more information on `applyConfig`.

---

## 2.2.4 cudbCheckConsistency

The `cudbCheckConsistency` command checks if the slave Data Store (DS) Units hold approximately as many rows in their tables as the corresponding master DS Unit in the same DS Unit Group (DSG). The lightweight consistency check relates to PLDB units as well. The check is considered to be failed if the difference of the row counts of any Structured Query Language (SQL) table exceeds a certain value, given in percentage. Tables with only a few rows can be skipped.

`cudbCheckConsistency` can be executed manually, but it also runs regularly as a `cron` job. When executed manually, the default configuration values can be overridden with command line options. When it finds a failure while running as a `cron` job, it raises an alarm on the node which hosts the failed slave DS Unit. For further information about alarms related to this command, refer to [Storage Engine, Potential Data Inconsistency between Replicas Found in DS and Storage Engine, Potential Data Inconsistency between Replicas Found in PLDB](#).

**Note:** If any changes are made in the configuration file on a CUDB node, then the exact same configuration changes must be performed on all the other CUDB nodes as well. This is needed because `cudbCheckConsistency` operates at the CUDB system level instead of the CUDB node level. After the configuration is updated in the whole system, the following command must be issued on every SC of every CUDB node:

```
/etc/init.d/cudbCheckConsistencySrv restart
```

### `cudbCheckConsistency` Requisites

`cudbOperator` users have no access to run this command.

### `cudbCheckConsistency` Syntax

- Format (1): `cudbCheckConsistency [-l | --locked] [-a | --alarms] [-m | --maxdiff-limit <LIMIT>] [-c | --check-limit <LIMIT>] [-v | --verbose <LEVEL>]`
- Format (2): `cudbCheckConsistency -h | --help`



- Format (3): `cudbCheckConsistency -o | --object-tables`

### cudbCheckConsistency Command Options

The following command options can be used:

- `-h | --help`: Displays command help message and exits.
- `-l | --locked`: Refuses to run if started anywhere else than on the PLDB master CUDB node, on the HA active SC. This option is useful and mandatory for running the program as a `cron` job.
- `-a | --alarms`: Allows to raise alarms if the test fails. This option is useful and mandatory when the program runs as a `cron` job. When started interactively, there is no need to raise alarms.
- `-v | --verbose <LEVEL>`: Sets the verbosity level to the defined `<LEVEL>`. Accepted values: '1' or '2'. A value of '1' means that detailed information about the progress of the program is printed to the console, but these extra details do not go into the system log. Level '2' means that extra details are written both to the console and to the system log.
- `-m | --maxdiff-limit <LIMIT>`: Uses the defined value of `<LIMIT>` as the maximal tolerable difference in percentage (`MAXDIFF_LIMIT` configuration value). `<LIMIT>` must be a non-negative floating point value rounded to two decimal digits in the `[ 0.00 ; 100.00 ]` closed interval. For example, "0.58" means 0.58% difference is allowed. The test fails if the difference is greater than the value defined for `<LIMIT>`.
- `-c | --check-limit <LIMIT>`: Does not check tables that have fewer rows than the value defined for `<LIMIT>` in the master DS Unit (`CHECK_LIMIT` configuration value). `<LIMIT>` must be a non-negative integer.
- `-o | --object-tables`: Checks for any possible differences in the `OBJECT_CLASSES` and the `MULTI_VALUE_OBJECTS` tables. Tables located on slave PLDB units will be checked against the tables located on master PLDB replica. Tables located on slave DS units will be checked both against the tables located on the corresponding master DS replica and the tables located on master PLDB replica.

**Note:** This option must not be run as a `Cron` job, but only from the command line.

### cudbCheckConsistency Output

The following example output is displayed:

```
[info] cudbCheckConsistency is running with options: '', MAXDIFF_LIMIT: 1.00%, CHECK_LIMIT: 100
[info] Acquiring mastership information.
[info] Checking consistency in slave DS units.
[info]   Node42-DSG0 consistency: OK - row count difference 0.00%
[info]   Node42-DSG1 consistency: OK - row count difference 0.00%
[info]   Node42-DSG2 consistency: OK - row count difference 0.00%
[info] Summary: slaves checked: 3 -> PASSED: 3, FAILED: 0, UNKNOWN: 0.
```



In this output, Node42-DSG0 refers to the PLDB.

### **cudbCheckConsistency Configuration File**

The configuration file of the command is located in the following directory:

```
/home/cudb/monitoring/replicaConsistency/config/
cudbCheckConsistency.conf
```

This file is directly included by a Bash shell script with the following prerequisites:

- Comment lines must start with a # character.
- Settings must follow the VARIABLE=VALUE pattern without white spaces around the = operator.

#### ***MAXDIFF\_LIMIT=1.00***

Maximum tolerable row count difference in PERCENTS between tables in the master DS Unit and the slave DS Unit. The difference is counted table-by-table, and if the difference is greater than the configured tolerance limit in any of the tables, the verdict is FAILED.

The value must be a floating point number with exactly two decimal digits, in the [ 0.00 ; 100.00 ] closed interval.

Default value: 1.00

Command line option: -m | --maxdiff-limit <LIMIT>

#### ***CHECK\_LIMIT=100***

If a table in the master DS Unit has fewer rows than this check limit, then the table is skipped.

The value must be an integer number in the [ 0 ; INT\_MAX ) left-closed right-open interval, where INT\_MAX is the constant defined in *POSIX.1-2008 Base Specifications, Issue 7*.

Default value: 100

Command line option: -c | --check-limit <LIMIT>

#### ***CRON\_TIMESPEC='37 0 \* \* \*'***

Cron job time specification. If this value is changed, then the value must also be changed on every other CUDB node of the CUDB system, and the new value must be installed to cron on both SCs of every CUDB node. On a specific CUDB node, after saving the new value, the new settings can be installed to cron by issuing the following command on an SC:

```
ssh OAM1 '/etc/init.d/cudbCheckConsistencySrv restart'
```



```
ssh OAM2 '/etc/init.d/cudbCheckConsistencySrv restart'
```

The value must be either "disabled" or a valid cron job time specification.

Default value: '37 0 \* \* \*'

Command line option: -

### 2.2.4.1

#### cudbCheckConsistency Examples of Use

Check the following examples of executing the command:

— <CUDB\_node\_prompt> **cudbCheckConsistency**

When executed like above, the command checks the consistency manually with the default values given in the configuration file. No alarms are raised.

— <CUDB\_node\_prompt> **cudbCheckConsistency -m 0.00 -c 0**

When executed like above, the command checks the consistency strictly so that no difference is accepted, and all the tables are checked. No alarms are raised.

### 2.2.5

#### cudbCheckReplication

The `cudbCheckReplication` command checks if the active database cluster replication channels are functional in the CUDB system. `cudbCheckReplication` can be executed manually, but it also runs regularly as a cron job. When executed manually, the default configuration values can be overridden with command line options. When it finds a failure while running as a cron job, it raises an alarm on the node which hosts the failed slave DS Unit. For further information about alarms related to this command, refer to *Storage Engine, Replication Stopped Working in DS, and Storage Engine, Replication Stopped Working in PLDB*.

**Note:** If any changes are made in the configuration file on a CUDB node, then the exact same configuration changes must be performed on every other CUDB node as well. This is required because `cudbCheckReplication` operates on the CUDB system level instead of the CUDB node level. After the configuration is updated in the entire system, execute the below command on every SC of every CUDB node:

```
/etc/init.d/cudbCheckReplicationSrv restart
```

#### cudbCheckReplication Requisites

`cudbOperator` users have no access to run this command.



### **cudbCheckReplication Syntax**

- Format (1): `cudbCheckReplication [-l | --locked] [-a | --alarms] [-b | --behind-limit <LIMIT>] [-v | --verbose <LEVEL>]`
- Format (2): `cudbCheckReplication -h | --help`

### **cudbCheckReplication Command Options**

The following command options can be used:

- `-h | --help`: Displays command help message and exits.
- `-l | --locked`: Refuses to run if started anywhere else than on the PLDB master CUDB node, on the HA active SC. This option is useful and mandatory for running the program as a cron job.
- `-a | --alarms`: Allows to raise alarms if an open replication channel does not work. This option is useful and mandatory for running the program as a cron job. When started interactively, there is no need to raise alarms.
- `-b | --behind-limit <LIMIT>` : After injecting test data into the master DS Unit of a DSG, the command waits for the specified amount of seconds (defined with <LIMIT> ) before starting to check if the data showed up in the slave DS Units. The value of <LIMIT> must be a positive integer, and its configuration variable is BEHIND\_LIMIT.
- `-v | --verbose <LEVEL>` : Sets the verbosity level to the defined <LEVEL> . Accepted values are '1' or '2'. A value of '1' means that detailed information about the execution progress is printed to the console, but these extra details are not included in the system log. Level '2' means that the extra details are written both to the console and to the system log.

### **cudbCheckReplication Output**

The command can display the following example output:

```
[info] cudbCheckReplication is running with options: '', BEHIND_LIMIT: 10
[info] Acquiring mastership information.
[info] Acquiring DSG status information.
[info] Injecting verification data to master DS units.
[info] Sleeping 10 seconds to wait for replication.
[info] Checking replication in slave DS units.
[info]   Node42-DSG0 replication: OK
[info]   Node42-DSG1 replication: OK
[info]   Node42-DSG2 replication: OK
[info] Summary: channels checked: 3 -> PASSED: 3, FAILED: 0, UNKNOWN: 0.
```

### **cudbCheckReplication Configuration File**

The configuration file of the command is located in the following directory:



/home/cudb/monitoring/replicaConsistency/config/  
cudbCheckReplication.conf

This file is directly included by a Bash shell script with the following prerequisites:

- Comment lines must start with a # character.
- Settings must follow the VARIABLE=VALUE pattern without white spaces around the = operator.

#### ***BEHIND\_LIMIT=10***

This setting assumes that any data injected to a master DS Unit are replicated within the defined amount of seconds to the slave DS Units. If the verification data does not arrive after the amount of seconds defined with **BEHIND\_LIMIT**, the test verdict is FAILED. The value must be an integer number in the [ 1 ; INT\_MAX ) left-closed right-open interval, where INT\_MAX is the constant defined in *POSIX.1-2008 Base Specifications, Issue 7*.

Default value: 10

Command line option: -b | --behind-limit <LIMIT>

#### ***CRON\_TIMESPEC='7 0 \* \* \*'***

This setting is the `cron` job time specification. If this value is changed, then the value must also be changed in every other CUDB node of the CUDB system, and the new value must be installed to `cron` on both SCs of every CUDB node. On a specific CUDB node, after saving the new value, you can install the new settings to `cron` by issuing the following command on the SCs:

```
ssh OAM1 '/etc/init.d/cudbCheckReplicationSrv restart'
```

```
ssh OAM2 '/etc/init.d/cudbCheckReplicationSrv restart'
```

The value must be either "disabled" or a valid `cron` job time specification.

Default value: '7 0 \* \* \*'

Command line option : -

### **2.2.5.1 cudbCheckReplication Examples of Use**

Use the command as shown below to check the replication channels with a wait limit of 5 seconds, and ensure that the command prints only basic progress information, and does not raise any alarm if a check fails:

```
<CUDB_node_prompt> cudbCheckReplication -b 5
```

### **2.2.6 cudbCollectInfo**

The `cudbCollectInfo` command is used to create a tarball of CUDB logs.



## cudbCollectInfo Requisites

cudbOperator users have no access to run this command.

Restrictions of cudbCollectInfo command:

- Only one instance of cudbCollectInfo command can be executed in one CUDB node
- Only one instance of cudbCollectInfo command at a time can collect logs from a target CUDB node
- Only one system-wide cudbCollectInfo command at a time can be executed in the CUDB system
- A system-wide cudbCollectInfo command precludes any other cudbCollectInfo command from running anywhere in the system

## cudbCollectInfo Syntax

```
cudbCollectInfo [-n | --node <node id>] [-d | --ds <dsg id>] [-a | --action <action name>] [-c | --no-compress] [-e | --no-encrypt] [-h | --help] [-x | --exit-on-error]
```

## cudbCollectInfo Command Options

The following command options can be used:

- `-h | --help`: Displays command help message and exits.
- `-n | --node <node id>` : Defines the node where logs must be collected. Use the ID of a CUDB node to define a specific node, or "all" to collect logs for all nodes. If the `-n` option is not used, the logs will be automatically collected from all the nodes in the system.
- `-d | --ds <dsg id>` : Defines the DSGs where the logs must be collected. Use the ID of the DSG to define a specific DSG, or "all" to collect logs for all DSGs. This option only applies to `ndb_error_reporter` logs, which are retrieved from local DS Units belonging to the specified DSGs. If the `-d` option is not used, the logs will be automatically collected from all the DSGs.
- `-c | --no-compress`: Disables the compression of the collected logs to an archive.
- `-e | --no-encrypt`: Disables the GPG encryption of the `tgz` archive.
- `-x | --exit-on-error`: Exits the command when an error is found during collection. Default behavior is not to exit.
- `-a | --action <action name>` : `<action name>` is the name of the log collecting action to execute. If the `-a` option is not used, all collecting actions will be performed as if the `-n all` option is used.





The available actions are as follows:

- `ndb_error_reporter`
- `collect_info`
- `sm`
- `esa`
- `cluster_config`
- `measurement`
- `ldapfe`
- `system_status`
- `bc_server`
- `evip`
- `ddci`
- `bc_client`
- `upgrade`
- `all`: This action is equivalent to specifying all other actions together.

### **cudbCollectInfo Output**

The following list contains some example output messages:

#### **— `cudbCollectInfo -n 1`**

```
Creating dir for node 1 (/local/tmp/cudb_collect_info_20120412-104337/1) ... OK
```

```
Waiting 1 to finish ... OK
```

```
CREATING ARCHIVE ... OK
Encrypting archive ... OK
Removing unencrypted archive ... OK
```

```
REMOVING RAW DATA ... OK
```

```
Fetch the file: /local/tmp/cudb_collect_info_20120412-104337.c
```

In case `cudbCollectInfo` is already running on a local node or on a remote node (from another terminal or from another node in the system), the following output messages are printed to the console:

#### **— `cudbCollectInfo -n <local node-id>`**

```
cudbCollectInfo is already running on this node.
```



— **cudbCollectInfo -n <remote node-id>**

cudbCollectInfo is already running on node <Node ID where the command is executing>.

— **cudbCollectInfo -n <remote node-id>**

cudbCollectInfo is already running for node <target node ID> on node <Node ID where the command is executing>.

**Note:** These last three outputs are displayed if cudbCollectInfo is already running somewhere in the CUDB system. If no other cudbCollectInfo process is running, this output is displayed due to a previous cudbCollectInfo process finished with an unexpected error and lock file was not removed from the CUDB node where it was executing. To unlock cudbCollectInfo, execute the following command in the CUDB nodes where the lock file is: `rm -rf /cluster/tmp/cudbcollectinfo*`

Do not delete this file if the command is up and running, to prevent unexpected behaviors.

### 2.2.6.1 cudbCollectInfo Examples of Use

Use the command as shown below to perform all log collection actions on Node 39:

```
<CUDB_node_prompt> cudbCollectInfo -n 39 -a all
```

Use the command as shown below to perform all log collection actions on all nodes of the system, disabling the GPG encryption of the tgz archive:

```
<CUDB_node_prompt> cudbCollectInfo -e
```

### 2.2.7 cudbConsistencyMgr

The cudbConsistencyMgr command manages consistency check tasks in the CUDB system. Refer to [CUDB Consistency Check](#) for more information on how to use the command and perform consistency check tasks.

#### cudbConsistencyMgr Requisites

The user permissions of executing cudbConsistencyMgr are as follows:

- Root permission is required to run the command.
- cudbadmin can run this command through sudo without password.
- cudbOperator users have no access to run this command without password.



## cudbConsistencyMgr Syntax

- Format (1): `cudbConsistencyMgr -h | --help`
- Format (2): `cudbConsistencyMgr -v | --version`
- Format (3): `cudbConsistencyMgr -l | --list`
- Format (4): `cudbConsistencyMgr -t | --taskhistory`
- Format (5): `cudbConsistencyMgr -r | --remove <taskId> [-f | --force]`
- Format (6): `cudbConsistencyMgr -o | --order ms -n | --node <nodeid> -p | --pl [--max-replica-lag <milliseconds>] [--alarm-severity-limit <threshold>] [-D | --debug] [--debugStderr] [--task-debug] [--task-verbose]`
- Format (7): `cudbConsistencyMgr -o | --order ms -n | --node <nodeid> -d | --dsg <dsgid> [--max-replica-lag <milliseconds>] [--alarm-severity-limit <threshold>] [-D | --debug] [--debugStderr] [--task-debug] [--task-verbose]`

## cudbConsistencyMgr Command Options

The following command options can be used:

- `-h | --help`: Displays command help message and exits.
- `-v | --version`: Prints version information and exits.
- `-l | --list`: Lists elements of the Pending Task List (PTL) and the Running Task List (RTL), then exits.
- `-t | --taskhistory`: Prints the contents of the local task history log on the screen, then exits.
- `-r | --remove <taskId>` : Removes the task specified by `<taskId>` from the PTL.
- `-f | --force`: Forces the removal of the task by trying to follow the task even in the RTL.
- `-o | --order ms`: Orders master-slave consistency check between the master and slave replicas selected by the `-d | --dsg`, `-p | --pl` and `-n | --node` command options.
- `-n | --node <nodeid>` : Defines the node ID of the slave replica which is subject to the consistency check.
- `-p | --pl`: Selects the PLDB for consistency check.



- `-d | --dsg<dsgid>` : Selects the DSG with the specified `<dsgid>` for consistency check.

**Note:** Either the `-p | --pl`, or `-d | --dsg` command option must be specified when executing the command.

- `[--max-replica-lag <milliseconds>]` (default value: 10000): This parameter sets the maximum tolerable replication distance between the master and the slave. The value is provided in milliseconds, and it defines the execution conditions for the check task. The value of 10000 means that the check task is not started, and respectively a running task is stopped, if the replication lag reaches or exceeds 10000 ms.
- `[--alarm-severity-limit <threshold>]` (default: 500): This parameter is used to control the severity of the alarm that is raised when inconsistency is found. If the number of inconsistent rows is higher than the `<threshold>` value, then a major severity alarm is raised instead of the default severity.
- `[--task-verbose]`: Restricted for Ericsson personnel.
- `[--task-debug]`: Restricted for Ericsson personnel.
- `-D | --debug`: Restricted for Ericsson personnel.
- `--debugStderr`: Restricted for Ericsson personnel.

### cudbConsistencyMgr Output

The output of the `cudbConsistencyMgr` command depends on the specified command options. Some examples are provided below.

#### Scheduling-Related Output

If the scheduling is successful, then `cudbConsistencyMgr` prints the task ID as shown below, which can be used to track the task:

Task UTC\_2014-09-23-17-05-42\_N242\_U0000001955 is put into the pending task list and will be executed on node 242.

#### Task Listing-Related Output

The output of the command uses two lists: the PTL lists the scheduled tasks, while the RTL lists the currently running check tasks. Examples for both lists are shown below:

```
[Site 1]
      RTL:
            UTC_2014-09-09-11-00-10_N121_U0000000849
            checkType=ms,source=S1-N121-D1,check=S2-N122-D1,maxReplica→
Lag=2000,alarmSeverityLimit=500,\
            verboseMode=off,debugMode=off
      PTL:
```

[Site 2]



```

RTL:
    UTC_2014-09-09-11-00-16_N122_U0000000793
    checkType=ms,source=S2-N122-D2,check=S1-N121-D2,maxReplica →
Lag=2000,alarmSeverityLimit=500,\
    verboseMode=off,debugMode=off
PTL:
    UTC_2014-09-09-11-00-17_N122_U0000000794
    checkType=ms,source=S2-N122-D2,check=S1-N121-D2,maxReplica →
Lag=2000,alarmSeverityLimit=500,\
    verboseMode=off,debugMode=off
    UTC_2014-09-09-11-00-21_N122_U0000000795
    checkType=ms,source=S2-N122-D3,check=S1-N121-D3,maxReplica →
Lag=2000,alarmSeverityLimit=500,\
    verboseMode=off,debugMode=off
    UTC_2014-09-09-11-00-22_N122_U0000000796
    checkType=ms,source=S2-N122-D3,check=S1-N121-D3,maxReplica →
Lag=2000,alarmSeverityLimit=500,\
    verboseMode=off,debugMode=off
    UTC_2014-09-09-11-00-23_N122_U0000000797
    checkType=ms,source=S2-N122-D3,check=S1-N121-D3,maxReplica →
Lag=2000,alarmSeverityLimit=500,\
    verboseMode=off,debugMode=off
    UTC_2014-09-09-11-00-23_N122_U0000000798
    checkType=ms,source=S2-N122-D3,check=S1-N121-D3,maxReplica →
Lag=2000,alarmSeverityLimit=500,\
    verboseMode=off,debugMode=off
    UTC_2014-09-09-11-00-23_N122_U0000000799
    checkType=ms,source=S2-N122-D3,check=S1-N121-D3,maxReplica →
Lag=2000,alarmSeverityLimit=500,\
    verboseMode=off,debugMode=off

```

## Task History-Related Output

The output in this case consists of log lines describing the following information:

- The time when each task was added to the PTL.
- The time when each task was started and finished.
- The result of the task executions.

The results are ordered by time, and are listed from the oldest to the newest, as shown below:

```

2014-09-23 15:53:49+02:00 SC_2_1 UTC_2014-09-23-13-53-48_N243_U0000000586: task fo →
und in PTL
checkType=ms,source=S2-N243-D1,check=S1-N242-D1,maxReplicaLag=2000,alarmSeverityLi →
mit=500,verboseMode=off,debugMode=off
2014-09-23 15:53:49+02:00 SC_2_1 UTC_2014-09-23-13-53-48_N243_U0000000586: task ex →
ecution started
2014-09-23 15:54:00+02:00 SC_2_1 UTC_2014-09-23-13-53-48_N243_U0000000586: task ex →
ecution finished with result:
Successful completion, no difference found. Exit code: 0.

```

### 2.2.7.1

## cudbConsistencyMgr Examples of Use

Check the following examples of executing the command:

- Execute the command as shown below to order a slave consistency check of DSG 33 on node 100:

```

<CUDB_node_prompt> cudbConsistencyMgr --order ms --node 100 --
dsg 33

```



Alternatively, use the shortened syntax of the command options as follows:

```
<CUDB_node_prompt> cudbConsistencyMgr -o ms -n 100 -d 33
```

- Execute the command as shown below to order a slave consistency check of the PLDB on node 200:

```
<CUDB_node_prompt> cudbConsistencyMgr --order ms --node 200 --pl
```

Alternatively, use the shortened syntax of the command options as follows:

```
<CUDB_node_prompt> cudbConsistencyMgr -o ms -n 200 -p
```

- Execute the command as shown below to list the running and pending tasks:

```
<CUDB_node_prompt> cudbConsistencyMgr --list
```

- Execute the command as shown below to remove a known task from the PTL:

```
<CUDB_node_prompt> cudbConsistencyMgr --remove  
UTC_2014-09-02-11-32-46_N47_U0000000007
```

Alternatively, use the shortened syntax of the command option as follows:

```
<CUDB_node_prompt> cudbConsistencyMgr -r  
UTC_2014-09-02-11-32-46_N47_U0000000007
```

- Execute the command with the `--force` option as shown below to remove a known task from the RTL, or even abort its execution:

```
<CUDB_node_prompt> cudbConsistencyMgr --remove  
UTC_2014-09-02-11-32-46_N47_U0000000008 --force
```

Alternatively, use the shortened syntax of the command option as follows:

```
<CUDB_node_prompt> cudbConsistencyMgr -r  
UTC_2014-09-02-11-32-46_N47_U0000000008 -f
```

- Execute the command as shown below to check the history of the scheduled tasks:

```
<CUDB_node_prompt> cudbConsistencyMgr --taskhistory
```

Alternatively, use the shortened syntax of the command option as follows:

```
<CUDB_node_prompt> cudbConsistencyMgr -t
```



## 2.2.8

### cudbDataBackup

The `cudbDataBackup` command executes a data backup in the complete CUDB system. Refer to [CUDB Backup and Restore Procedures](#) for further details about when to use it.

#### cudbDataBackup Requisites

`cudbOperator` users have no access to run this command.

#### cudbDataBackup Syntax

- Format (1): `cudbDataBackup [-T | --Timeout <seconds>] [-R | --Retries-number <number_of_retries>] [-t | --time <seconds>] [-S | --Slack-backup <minutes>] [-F | --Force] [-q | --DisablePG] [-L | --Parallel-mode] [-u | --user-facility <facility>] [-p | --print] [-D | --Debug]`
- Format (2): `cudbDataBackup -s | --schedule <cron_expression> [-T | --Timeout <seconds>] [-R | --Retries-number <number_of_retries>] [-t | --time <seconds>] [-S | --Slack-backup <minutes>] [-F | --Force] [-q | --DisablePG] [-L | --Parallel-mode] [-u | --user-facility <facility>] [-p | --print] [-D | --Debug]`
- Format (3): `cudbDataBackup -U | --Unschedule <cron_expression> [-T | --Timeout <seconds>] [-R | --Retries-number <number_of_retries>] [-t | --time <seconds>] [-S | --Slack-backup <minutes>] [-F | --Force] [-q | --DisablePG] [-L | --Parallel-mode] [-u | --user-facility <facility>] [-p | --print] [-D | --Debug]`
- Format (4): `cudbDataBackup -h | --help`

#### cudbDataBackup Command Options

The following command options can be used:

- `-T | --Timeout <seconds>` : Restricted for Ericsson personnel.
- `-R | --Retries-number <number_of_retries>` : Defines the number of retry attempts to establish the connection to the CUDB node where a specific piece of the data backup program is taken, if it fails the first time. The default value is 2.
- `-t | --time <seconds>` : Defines the time (in seconds) between retries. The default value is 5 seconds.
- `-S | --Slack-backup <minutes>` : Defines the time (in minutes) available to finish the provisioning operations before starting the backup. The default value is 2 minutes.



- `-F | --Force`: Forces the backup, even if the notification to the Provisioning Gateway (PG) fails.
- `-q | --DisablePG`: Disables PG notification.
- `-L | --Parallel-mode`: Orders a backup for each CUDB node element (that is for the PLDB and all DS Units like DS1, DS2, and so on) at the same time instead of one after another (that is, not when the previous one is finished).
- `-u | --user-facility <facility>` : Restricted for Ericsson personnel.
- `-s | --schedule`: Creates a periodic system data backup according to the defined `<cron_expression>` . The cron expression follows the below UNIX format:

`<minute> <hour> <day_of_month> <month_of_year> <day_of_week>`

The accepted values and value ranges are as follows:

- `<minute>`: `<0 - 59>`
- `<hour>`: `<0 - 23>`
- `<day_of_month>`: `<1 - 31>`
- `<month_of_year>`: `<1 - 12>`
- `<day_of_week>`: `<0 - 7>`

**Note:** Values 0 and 7 for `<day_of_week>` both mean the same day, Sunday.

Every value can be replaced with an asterisk ("`*`"), meaning all possible values for the field. See the below execution for an example:

```
cudbDataBackup -s "5 0 * * *"
```

**Note:** In case the `-s | --schedule` option is used, then any additional `cudbDataBackup` options listed in this section must be stated after this option.

- `-U | --Unschedule`: Removes a periodic system data backup according to the defined `<cron_expression>` . The cron expression follows the same format as described for the `-s | --schedule` command option. This command option must be executed with the same optional arguments that were used to set the schedule.
- `-p | --print`: Prints the information of the scheduled periodic system data backup.
- `-D | --Debug`: Restricted for Ericsson personnel.





— -h | --help: Displays command help message and exits.

### **cudbDataBackup Output**

The command output is a log of the performed changes. The command result is provided at the end of the log.

#### **2.2.8.1**

### **cudbDataBackup Examples of Use**

The following example procedure shows how to execute cudbDataBackup with the -S | --Slack-backup option:

#### **Steps**

1. Connect to the CUDB node with the following command:

```
<CUDB_node_prompt> ssh <admin_user>@<CUDB_Node_IP_Address>
```

2. Connect to the SC with the following command:

```
<CUDB_node_prompt> ssh <admin_user>@<CUDB_Node_OAM_IP_Address>
```

3. Execute the cudbDataBackup command as follows:

```
<CUDB_node_prompt> cudbDataBackup -S 5
```

The expected output must be similar to the below example:

```
/home/cudb/systemDataBackup
Listening for current PLDB and DSGs status reports (may take up to 2 →
minutes)
Starting backup...
Before calling pg_notification
cudb_backup      ver(1.3.4)
BEGIN MANAGEMENT FOR LAST BACKUP Backup
2013-10-31_15-09 finished successfully in :
  PLDB in CUDB node 27
    NDB node PL0
    NDB node PL1
  DS#2 in CUDB node 27
    NDB node DS2_0
    NDB node DS2_1
  DS#1 in CUDB node 29
    NDB node DS1_0
    NDB node DS1_1
Attempting to de-block Provisioning Gateway. This may take up to a co →
uple of minutes.
```



## 2.2.9 cudbDbDiskManage

The `cudbDbDiskManage` command manages the space of the Binary Large Object (BLOB) attributes stored on the disk storage system for certain CUDB object classes.

### cudbDbDiskManage Requisites

`cudbOperator` users have no access to run this command.

### cudbDbDiskManage Syntax

- Format (1): `cudbDbDiskManage -a | --allocate <objectClass> <size>`
- Format (2): `cudbDbDiskManage -l | --list <objectClass>`
- Format (3): `cudbDbDiskManage -h | --help`

### cudbDbDiskManage Command Options

The following command options can be used:

- `-a | --allocate <objectClass> <size>` : Allocates extra space for the BLOB attributes stored on the disk storage system belonging to the object class defined with `<objectClass>` . The `<size>` is in megabytes and is rounded up to the following multiple of 128 as long as the provided value is greater than 128; otherwise, the exact size is added.
- `-l | --list <objectClass>` : Lists the occupied and free space for storing attribute values on the disk storage system for the object class defined with `<objectClass>` . Refer to [CUDB Application Schema Update](#) for more information on the available object classes.
- `-h | --help`: Displays command help message and exits.

### cudbDbDiskManage Output

The command reports `OK` in case of success, and `error` in case of failure.

### 2.2.9.1 cudbDbDiskManage Examples of Use

The following example procedure shows how to allocate a set amount of space on the disk storage system for a BLOB object with `cudbDbDiskManage`:

#### Steps

1. Connect to the CUDB node with the following command:

```
<CUDB_node_prompt> ssh <admin_user>@<CUDB_Node_IP_Address>
```



2. Connect to the SC with the following command:

```
<CUDB_node_prompt> ssh <admin_user>@<CUDB_Node_OAM_IP_Address>
```

3. <CUDB\_node\_prompt> **cudbDbDiskManage -a**  
 <BLOB\_object\_name\_stored\_on\_disk> <Allocated\_disk\_space\_MB>

The expected output of the above command must look similar to the below example:

```
Obtaining SQL Servers Information ...
NODE <DS_ID
or PL_ID> : <type>
Connection strings : <mysql ip:port><mysql ip:port>
Disk space assigned to the object class <BLOB_object_name_stored_on_disk>→
  before increase : <x> Megabytes
Disk space assigned to the object class <BLOB_object_name_stored_on_disk>→
  after increase : <x> Megabytes
```

**Note:** The value of <x> in the above output must be greater than, or equal to the value defined with <Allocated\_disk\_space\_MB>

## 2.2.10

### cudbDsgMastershipChange

The **cudbDsgMastershipChange** command is used to move the master of a DSG to the selected node. It must be launched on the CUDB node that will hold the new master replica.

When executed, the command first checks the PLDB replication status on the node where the command is invoked. If the PLDB is unable to synchronize or replication status info is not available, the mastership change will be rejected. After this, the command checks if the new master replica has a longer delay than the time specified by the **-t time** parameter: if it does, it rejects the mastership change. Then, for a period of four seconds, all write operations toward the current master are rejected to allow the new master replica to catch up with the current master. After this four-second period passes, the command checks if the new master is completely synchronized with the current master to avoid any potential data loss. If the check succeeds, the master of the specified DSG or PLDB is changed to the node.

In case Automatic Mastership Change (AMC) is active, the system periodically tries to move the master of each DSG to a DSG replica with higher priority, unless the master was intentionally moved to a degraded replica.

To avoid undesired master movements, disable the AMC feature. Refer to **CUDB High Availability** for more information.

### cudbDsgMastershipChange Requisites

**cudbOperator** users have no access to run this command.



### **cudbDsgMastershipChange Syntax**

- Format (1): `cudbDsgMastershipChange -d | --dsg <dsgId> [-t | --time <seconds>] [-D | --Debug] [--force [--no-prompt]]`
- Format (2): `cudbDsgMastershipChange -p | --pl [-t | --time <seconds>] [-D | --Debug] [--force [--no-prompt]]`
- Format (3): `cudbDsgMastershipChange -h | --help`

### **cudbDsgMastershipChange Command Options**

The following command options can be used:

- `-d | --dsg <dsgId>` : Identifies the DSG where the master change is executed. The value of `<dsgId>` must be between 1 and 255.
- `-p | --pl`: Refers to the local PLDB cluster.
- `-t | --time <seconds>` : Defines the time (in seconds) of the allowed synchronization delay between the current master replica and the future master replica to perform the mastership change. In case the current master replica is not close enough, the command exits and does not perform the mastership change. The default value is 3 seconds.
- `-D | --Debug`: Restricted for Ericsson personnel.
- `--force`: Forces to complete the mastership change, even in the following cases:
  - The PLDB is unable to synchronize on the node where the command is invoked.
  - The PLDB replication status information is unavailable on the node where the command is invoked.
  - Replication channels are unavailable.
  - The replication in the new target master does not align with the current master.

This option is not exclusive with `--time`. To force a mastership change with great distance between the new master and the current master, use a great value for the `--time` parameter.

- `-h | --help`: Displays command help message and exits.
- `--no-prompt`: Restricted for Ericsson personnel only.

### **cudbDsgMastershipChange Output**

The possible outputs for the command are as follows:



— In case of successful mastership change:

- Processing DSG mastership change...  
cudbDsgMastershipChange: Success.  
DSG 2 master replica moved from node N2 to N3 successfully.
- Processing PL mastership change...  
cudbDsgMastershipChange: Success.  
PL master replica moved from node N2 to N3 successfully.
- Processing DSG mastership change...  
cudbDsgMastershipChange: Success.  
The node already has the master replica of the specified DSG.

— In case of failed mastership change:

- Processing DSG mastership change...  
cudbDsgMastershipChange: Failed.  
DSG 2 master replica can not be moved to node N3 because replicat →  
ion with node N2 is not aligned.
- Processing DSG mastership change...  
cudbDsgMastershipChange: Failed.  
Replication channel is not up.

**Note:** In the two cases above, an error message is also provided describing the situation. In these cases, the `--force` option can be used to proceed with the mastership change. However, consider that in this case, the mastership change most probably results in data loss because the candidate master cannot be synchronized with the current master. In case of data loss, replication fails, and the `Storage Engine, Unable to Synchronize Cluster in DS, Major` or `Storage Engine, Unable to Synchronize Cluster in PLDB, Major` alarms are raised. Refer to the appropriate alarm OPI for more information on how to clear the alarm.

- Processing DSG mastership change...  
cudbDsgMastershipChange: Failed.  
Replication is delayed more than 3 secs.

**Note:** In the latter example, the `--time` option can be used to increase the allowed synchronization time delay between replicas. If the mastership still cannot be changed because the candidate master cannot align with the current master, the `--force` option can be used in addition to the `--time` option to force the change.

- Processing DSG mastership change...  
cudbDsgMastershipChange: Failed.  
DSG 2 master replica can not be moved. Check site SM leader logs.



- Processing DSG mastership change...  
cudbDsgMastershipChange: Failed.  
The DSG specified is masterless.
  - Processing DSG mastership change...  
cudbDsgMastershipChange: Failed.  
The DS hosting the specified DSG in the node where the command is →  
invoked is disabled.
  - Processing DSG mastership change...  
cudbDsgMastershipChange: Failed.  
The node hosting the specified DSG has PL down or no SM available →  
.
  - Processing DSG mastership change...  
cudbDsgMastershipChange: Failed.  
The specified DSG is not alive.
  - Processing DSG mastership change...  
cudbDsgMastershipChange: Failed.  
There is no replica distance info in BC cluster.
  - Processing DSG mastership change...  
cudbDsgMastershipChange: Failed.  
The PLDB is unable to synchronize on the node where the command i →  
s invoked.
  - Processing DSG mastership change...  
cudbDsgMastershipChange: Failed.  
Replication status info for PLDB on the node where the command is →  
invoked is not available.
- When attempting to move the DSG master to a degraded replica, the following warning message is shown:

```
Processing DSG mastership change...
WARNING: The master will be moved to a degraded replica.
cudbDsgMastershipChange: Success.
DSG 1 master replica moved from node N1 to N2 successfully.
```

### 2.2.10.1 cudbDsgMastershipChange Examples of Use

The following example procedure shows how to change the master of DSG 2 with cudbDsgMastershipChange:

#### Steps

1. Connect to the CUDB node with the following command:

```
<CUDB_node_prompt> ssh <admin_user>@<CUDB_Node_IP_Address>
```



2. Connect to the SC with the following command:

```
<CUDB_node_prompt> ssh <admin_user>@<CUDB_Node_OAM_IP_Address>
```

3. Execute the command as follows:

```
<CUDB_node_prompt> cudbDsgMastershipChange -d 2
```

The expected output must be similar to the below example:

```
Processing DSG mastership change...
cudbDsgMastershipChange: Success.
DSG 2 master replica moved from node 80 to 81 successfully.
```

## 2.2.11 cudbDsgProvisioningManage

The `cudbDsgProvisioningManage` command is used to disable or enable a specific DSG during the provisioning of Distribution Entries (DEs). When using the command, consider the following:

- When the provisioning lock is set on a DSG, no more DEs are assigned automatically to that DSG. Provisioning and reallocation operations cannot be executed on that specific DSG. However, this provisioning lock can be overridden (and therefore new DEs can be created on a provisioning-locked DSG) by specifying the affected DSG in the provisioning operation as the place where the new DEs must be stored or by specifying force option in reallocation operation.
- The provisioning lock does not affect every provisioning-related operation, but only the creation of new DEs. Existing DEs can still be deleted on a provisioning-locked DSG, and Lightweight Directory Access Protocol (LDAP) operations (Add, Add/Modify, Delete, and so on) can also be performed on data stored under the existing DEs.

### cudbDsgProvisioningManage Requisites

`cudbOperator` users have no access to run this command.

### cudbDsgProvisioningManage Syntax

- Format (1): `cudbDsgProvisioningManage -e | --enable <DSGId> [-D | --Debug]`
- Format (2): `cudbDsgProvisioningManage -d | --disable <DSGId> [-D | --Debug]`
- Format (3): `cudbDsgProvisioningManage -s | --status <DSGId> [-D | --Debug]`
- Format (4): `cudbDsgProvisioningManage -h | --help`



### **cudbDsgProvisioningManage Command Options**

The following command options can be used:

- `-e | --enable <DSGId>` : Enables provisioning in the specified DSG. `<DSGId>` must be a valid DSG identifier present in the system.
- `-d | --disable <DSGId>` : Disables provisioning in the specified DSG. `<DSGId>` must be a valid DSG identifier present in the system.
- `-s | --status <DSGId>` : Shows the provisioning status of the specified DSG. `<DSGId>` must be a valid DSG identifier present in the system. The possible return values are enable or disable for provisioning.
- `-D | --Debug`: Restricted for Ericsson personnel.
- `-h | --help`: Displays command help message and exits.

### **cudbDsgProvisioningManage Output**

An example output of the command is shown below:

```
cudbDsgProvisioningManage: Failed.  
Invalid syntax
```

```
cudbDsgProvisioningManage: Failed.  
Unrecognized command line argument or arguments
```

```
cudbDsgProvisioningManage: Failed.  
DSG %d is not valid
```

```
cudbDsgProvisioningManage: Failed.  
Unexpected error.
```

```
cudbDsgProvisioningManage: Success.  
DSG %d is already enabled.
```

```
cudbDsgProvisioningManage: Success.  
DSG %d is already disabled.
```

```
cudbDsgProvisioningManage: Success.  
DSG %d is enabled for provisioning.
```

```
cudbDsgProvisioningManage: Success.  
DSG %d is disabled for provisioning
```

#### **2.2.11.1 cudbDsgProvisioningManage Examples of Use**

The following example procedure shows how to enable provisioning on DSG 1 with `cudbDsgProvisioningManage`:





## Steps

1. Connect to the CUDB node with the following command:

```
<CUDB_node_prompt> ssh <admin_user>@<CUDB_Node_IP_Address>
```

2. Connect to the SC with the following command:

```
<CUDB_node_prompt> ssh <admin_user>@<CUDB_Node_OAM_IP_Address>
```

3. Execute the command as follows:

```
<CUDB_node_prompt> cudbDsgProvisioningManage --enable 1
```

The expected output must be similar to the below example:

```
cudbDsgProvisioningManage: Success
```

## 2.2.12

### cudbGetLogs

The `cudbGetLogs` command collects preventive maintenance logs for log analysis. For more information about this command, refer to [CUDB Logchecker](#).

#### cudbGetLogs Requisites

`cudbOperator` users have no access to run this command.

#### cudbGetLogs Syntax

```
cudbGetLogs [-d | --debug] [-b | --bash-debug] [-c | --consistency-threshold <threshold>] [-f | --force-c-check] [-h | --help]
```

#### cudbGetLogs Command Options

The following command options can be used:

- -d | --debug: Restricted for Ericsson personnel.
- -b | --bash-debug: Restricted for Ericsson personnel.
- -c | --consistency-threshold <threshold> : Sets the lightweight consistency check error threshold (the default value is 500).
- -f | --force-c-check: Forces the lightweight consistency check to run.
- -h | --help: Displays command help message and exits.



## cudbGetLogs Output

An example output for the command is provided below:

```
Starting /opt/ericsson/cudb/OAM/bin/cudbGetLogs ...
Grep logs and creating /home/cudb/monitoring/preventiveMaintenance/CUDb_107_201201241136.log ...
The log file is saved as : /home/cudb/monitoring/preventiveMaintenance/CUDb_107_201201241136.log
```

### 2.2.12.1 cudbGetLogs Examples of Use

- Execute the command as shown below to run it in normal mode:

```
<CUDb_node_prompt> cudbGetLogs
```

- Execute the command as shown below to run it in debug mode:

```
<CUDb_node_prompt> cudbGetLogs --debug
```

- Execute the command as shown below to run it in debug mode and also force a lightweight consistency check to run with a threshold of 1:

```
<CUDb_node_prompt> cudbGetLogs --debug --force-c-check --
consistency-threshold 1
```

### 2.2.13 cudbHaState

The cudbHaState commands automatically prints the cluster status.

#### cudbHaState Requisites

cudbOperator users have no access to run this command.

#### cudbHaState Syntax

**cudbHaState**

#### cudbHaState Command Options

Not applicable.

#### cudbHaState Output

The output shows the LDE and AMF cluster state, the CoreMW, COM state and the SI and SU states. The output must be similar to the below example:

```
LOTC cluster uptime:
-----
Sat Apr 20 18:33:30 2013
LOTC cluster state:
-----
Node safNode=SC_2_1 joined cluster | Sat Apr 20 18:33:40 2013
Node safNode=SC_2_2 joined cluster | Sat Apr 20 18:33:30 2013
Node safNode=PL_2_3 joined cluster | Sat Apr 20 18:35:39 2013
```



```
Node safNode=PL_2_4 joined cluster | Sat Apr 20 18:35:43 2013
Node safNode=PL_2_5 joined cluster | Sat Apr 20 18:35:43 2013
Node safNode=PL_2_6 joined cluster | Sat Apr 20 18:35:43 2013
Node safNode=PL_2_7 joined cluster | Sat Apr 20 18:35:48 2013
Node safNode=PL_2_8 joined cluster | Sat Apr 20 18:35:48 2013
Node safNode=PL_2_9 joined cluster | Sat Apr 20 18:35:53 2013
Node safNode=PL_2_10 joined cluster | Sat Apr 20 18:35:48 2013
AMF cluster state:
-----
saAmfNodeAdminState."safAmfNode=SC-1,safAmfCluster=myAmfCluster": Unlocked
saAmfNodeOperState."safAmfNode=SC-1,safAmfCluster=myAmfCluster": Enabled
saAmfNodeAdminState."safAmfNode=SC-2,safAmfCluster=myAmfCluster": Unlocked
saAmfNodeOperState."safAmfNode=SC-2,safAmfCluster=myAmfCluster": Enabled
saAmfNodeAdminState."safAmfNode=PL-3,safAmfCluster=myAmfCluster": Unlocked
saAmfNodeOperState."safAmfNode=PL-3,safAmfCluster=myAmfCluster": Enabled
saAmfNodeAdminState."safAmfNode=PL-4,safAmfCluster=myAmfCluster": Unlocked
saAmfNodeOperState."safAmfNode=PL-4,safAmfCluster=myAmfCluster": Enabled
saAmfNodeAdminState."safAmfNode=PL-5,safAmfCluster=myAmfCluster": Unlocked
saAmfNodeOperState."safAmfNode=PL-5,safAmfCluster=myAmfCluster": Enabled
saAmfNodeAdminState."safAmfNode=PL-6,safAmfCluster=myAmfCluster": Unlocked
saAmfNodeOperState."safAmfNode=PL-6,safAmfCluster=myAmfCluster": Enabled
saAmfNodeAdminState."safAmfNode=PL-7,safAmfCluster=myAmfCluster": Unlocked
saAmfNodeOperState."safAmfNode=PL-7,safAmfCluster=myAmfCluster": Enabled
saAmfNodeAdminState."safAmfNode=PL-8,safAmfCluster=myAmfCluster": Unlocked
saAmfNodeOperState."safAmfNode=PL-8,safAmfCluster=myAmfCluster": Enabled
saAmfNodeAdminState."safAmfNode=PL-9,safAmfCluster=myAmfCluster": Unlocked
saAmfNodeOperState."safAmfNode=PL-9,safAmfCluster=myAmfCluster": Enabled
saAmfNodeAdminState."safAmfNode=PL-10,safAmfCluster=myAmfCluster": Unlocked
saAmfNodeOperState."safAmfNode=PL-10,safAmfCluster=myAmfCluster": Enabled
CoreMW HA state:
-----
CoreMW is assigned as ACTIVE in controller SC-2
CoreMW is assigned as STANDBY in controller SC-1
COM state:
-----
COM is assigned as ACTIVE in controller SC-2
COM is assigned as STANDBY in controller SC-1
SI HA state:
-----
saAmfSISUHAState."safSu=SC-2,safSg=2N,safApp=ERIC-CUDB_CUDBOI"."safSi=2N-1": active(1)
saAmfSISUHAState."safSu=SC-1,safSg=2N,safApp=ERIC-CUDB_KPICENTRAL"."safSi=2N-1": active(1)
saAmfSISUHAState."safSu=SC-2,safSg=DS1_2N,safApp=ERIC-CUDB_CS"."safSi=DS1_2N-1": active(1)
saAmfSISUHAState."safSu=SC-2,safSg=DS2_2N,safApp=ERIC-CUDB_CS"."safSi=DS2_2N-1": active(1)
saAmfSISUHAState."safSu=SC-2,safSg=PLDB_2N,safApp=ERIC-CUDB_CS"."safSi=PLDB_2N-1": active(1)
saAmfSISUHAState."safSu=SC-2,safSg=2N,safApp=ERIC-CUDB_LDAPFE_MONITOR"."safSi=2N-1": active(1)
saAmfSISUHAState."safSu=SC-1,safSg=2N,safApp=ERIC-CUDB_CUDBOI"."safSi=2N-1": standby(2)
saAmfSISUHAState."safSu=SC-1,safSg=2N,safApp=ERIC-CUDB_LDAPFE_MONITOR"."safSi=2N-1": standby(2)
saAmfSISUHAState."safSu=SC-1,safSg=2N,safApp=ERIC-CUDB_MSGSRV_MONITOR"."safSi=2N-1": active(1)
saAmfSISUHAState."safSu=SC-1,safSg=DS2_2N,safApp=ERIC-CUDB_CS"."safSi=DS2_2N-1": standby(2)
saAmfSISUHAState."safSu=SC-1,safSg=DS1_2N,safApp=ERIC-CUDB_CS"."safSi=DS1_2N-1": standby(2)
saAmfSISUHAState."safSu=SC-1,safSg=PLDB_2N,safApp=ERIC-CUDB_CS"."safSi=PLDB_2N-1": standby(2)
saAmfSISUHAState."safSu=PL-3,safSg=2N,safApp=ERIC-CUDB_SOAP_NOTIFIER"."safSi=2N-1": active(1)
saAmfSISUHAState."safSu=PL-4,safSg=2N,safApp=ERIC-CUDB_SOAP_NOTIFIER"."safSi=2N-1": standby(2)
SU States:
-----
Status OK
```

## 2.2.13.1 cudbHaState Examples of Use

<CUDB\_node\_prompt> **cudbHaState**

## 2.2.14 cudbLdapFeRestart

The **cudbLdapFeRestart** command is used to perform a rolling restart on the LDAP Front Ends (FEs).

By default, the LDAP FE processes in the node are restarted one by one. An LDAP FE process is not restarted until the previous one is not fully operative (waiting up to a maximum of 120 seconds). In case the restarted LDAP FE process is not fully



operative in period of 120 seconds, restart of the LDAP FEs is aborted. This ensures that the LDAP connections are equally balanced between all LDAP FEs after executing the command.

**Note:** The current connections of the LDAP FEs are dropped when the processes are restarted.

### **cudbLdapFeRestart Requisites**

`cudbOperator` users have no access to run this command.

### **cudbLdapFeRestart Syntax**

```
cudbLdapFeRestart [-f | --force] [-p | --parallel [--no-prompt]]  
[-h | --help]
```

### **cudbLdapFeRestart Command Options**

The following command options can be used:

- `-p | --parallel`: Restricted for Ericsson personnel.

---

---

#### **Caution!**

Never use the command option above in a live system. Use it only during node installation, otherwise traffic loss and degradation can occur.

---

---

- `--no-prompt`: Restricted for Ericsson personnel.
- `-h | --help`: Displays command help message and exits.
- `-f | --force`: Continue with the restart, even if any `slapd` process fail to start.

---

---

#### **Caution!**

After executing a forced restart, in case there is an error in the LDAP-FE configuration, all LDAP-FEs will be down.

---

---

### **cudbLdapFeRestart Output**

The command output is a log of the command progress. The result of the command is provided at the end of the log.



### 2.2.14.1 cudbLdapFeRestart Examples of Use

The following example procedure shows how to execute cudbLdapFeRestart:

#### Steps

1. Connect to the CUDB node with the following command:

```
<CUDB_node_prompt> ssh <admin_user>@<CUDB_Node_IP_Address>
```

2. Connect to the SC with the following command:

```
<CUDB_node_prompt> ssh <admin_user>@<CUDB_Node_OAM_IP_Address>
```

3. Execute the command as follows:

```
<CUDB_node_prompt> cudbLdapFeRestart
```

The expected output must be similar to the below example:

```
Reading necessary data from /home/cudb/common/config/cudbSystem.xml
Restarting slapd process in host 10.22.89.3
Stopping slapd...OK
Making sure cudbLdapFeMonitor is running (to start slapd)...OK
Waiting for slapd to be ready...OK
slapd ready after 24 seconds
-----
Restarting slapd process in host 10.22.89.4
Stopping slapd...OK
Making sure cudbLdapFeMonitor is running (to start slapd)...OK
Waiting for slapd to be ready...OK
slapd ready after 56 seconds
-----
Restarting slapd process in host 10.22.89.5
Stopping slapd...OK
Making sure cudbLdapFeMonitor is running (to start slapd)...OK
Waiting for slapd to be ready...OK
slapd ready after 10 seconds
-----
Restarting slapd process in host 10.22.89.6
Stopping slapd...OK
Making sure cudbLdapFeMonitor is running (to start slapd)...OK
Waiting for slapd to be ready...OK
slapd ready after 12 seconds
-----
Restarting slapd process in host 10.22.89.7
Stopping slapd...OK
Making sure cudbLdapFeMonitor is running (to start slapd)...OK
Waiting for slapd to be ready...OK
slapd ready after 12 seconds
-----
Restarting slapd process in host 10.22.89.8
```



```
Stopping slapd...OK
Making sure cudbLdapFeMonitor is running (to start slapd)...OK
Waiting for slapd to be ready...OK
ready after 11 seconds
-----
Restarting slapd process in host 10.22.89.9
Stopping slapd...OK
Making sure cudbLdapFeMonitor is running (to start slapd)...OK
Waiting for slapd to be ready...OK
slapd ready after 7 seconds
-----
Restarting slapd process in host 10.22.89.10
Stopping slapd...OK
Making sure cudbLdapFeMonitor is running (to start slapd)...OK
Waiting for slapd to be ready...OK
slapd ready after 15 seconds
-----
Restarting of slapd processes is finished
```

## 2.2.15 cudbManageBCServer

The `cudbManageBCServer` command manages the Blackboard Coordination (BC) servers in the CUDB node. Refer to [CUDB High Availability](#) for further information about BC servers and the BC cluster.

### `cudbManageBCServer` Requisites

This command acts on the BC server running on the blade or Virtual Machine (VM) where it is executed.

### `cudbManageBCServer` Syntax

- Format (1): `cudbManageBCServer -check`
- Format (2): `cudbManageBCServer -check4`
- Format (3): `cudbManageBCServer -checkMode`
- Format (4): `cudbManageBCServer -start`
- Format (5): `cudbManageBCServer -stop`
- Format (6): `cudbManageBCServer -restart`
- Format (7): `cudbManageBCServer -restart -no_check`
- Format (8): `cudbManageBCServer -h | --help`



### **cudbManageBCServer Command Options**

The following command options can be used:

- **-check**: Checks if a BC server is running on the blade or VM.
- **-check4**: By using a four-letter command, this option checks if the process is running and if it is connected to a BC cluster with the majority of BC servers.
- **-checkMode**: Checks if the specific BC server is the leader of the cluster, or just a follower.
- **-start**: Starts a BC Server in the blade or VM.
- **-stop**: Stops a BC Server in the blade or VM.
- **-restart**: Restarts a BC Server in the blade or VM.
- **-restart -no\_check**: Restricted for Ericsson personnel.
- **-h | --help**: Displays command help message and exits.

### **cudbManageBCServer Output**

When executed with the **-checkMode** option, the command returns an output similar to the below example:

```
cudbManageBCServer -checkMode
Actual Host=SC_2_1
Mode: follower
```

#### **2.2.15.1 cudbManageBCServer Examples of Use**

```
<CUDB_node_prompt> cudbManageBCServer -checkMode
```

#### **2.2.16 cudbManageLibDataDist**

The **cudbManageLibDataDist** command checks if a specific CUDB node is using the default distribution policy or a custom one. Refer to *CUDB System Administrator Guide* for further details about using the command.

### **cudbManageLibDataDist Requisites**

**cudbOperator** users have no access to run this command.

### **cudbManageLibDataDist Syntax**

```
cudbManageLibDataDist [-l | --load] [-0 | --unload] [-s | --status] [-h | --help]
```



### **cudbManageLibDataDist Command Options**

The following command options can be used:

- -1 | --load: Loads a new library stored in the following temporary folder: /home/cudb/oam/configMgmt/commands/tmp.
- -0 | --unload: Unloads the library stored in /usr/lib/.
- -s | --status: Checks the status of the distribution policy.
- -h | --help: Displays command help message and exits.

### **cudbManageLibDataDist Output**

The command must return an output similar to the below example:

```
Customer distribution policy active
```

#### **2.2.16.1**

### **cudbManageLibDataDist Examples of Use**

The following example procedure shows how to check the status of the distribution policy with `cudbManageLibDataDist -s`:

#### **Steps**

1. Connect to the CUDB node with the following command:  

```
<CUDB_node_prompt> ssh <admin_user>@<CUDB_Node_IP_Address>
```
2. Connect to the SC with the following command:  

```
<CUDB_node_prompt> ssh <admin_user>@<CUDB_Node_OAM_IP_Address>
```
3. Execute the command as follows:  

```
<CUDB_node_prompt> cudbManageLibDataDist -s
```

The expected output must be similar to the below example:

```
CUDB default distribution policy active
```

#### **2.2.17**

### **cudbManageMsgSrvServer**

The `cudbManageMsgSrvServer` command manages the Messaging Service servers (MsgSrv) in the CUDB node. Refer to [CUDB High Availability](#) for further information about BC servers and the BC cluster.





### **cudbManageMsgSrvServer Requisites**

This command acts on the MsgSrv server running on the blade or VM where it is executed.

### **cudbManageMsgSrvServer Syntax**

- Format (1): **cudbManageMsgSrvServer status**
- Format (2): **cudbManageMsgSrvServer start**
- Format (3): **cudbManageMsgSrvServer stop**
- Format (4): **cudbManageMsgSrvServer restart**

### **cudbManageMsgSrvServer Command Options**

The following command options can be used:

- **status**: Checks if a MsgSrv server is running on the blade or VM.
- **start**: Starts a MsgSrv server in the blade or VM.
- **stop**: Stops a MsgSrv server in the blade or VM.
- **restart**: Restarts a MsgSrv server in the blade or VM.

### **cudbManageMsgSrvServer Output**

The result the command depends on the option used. For example:

- **status**:  
 Status of Messaging Service server: <UP|DOWN>, pid:<PID>
- **start**:  
 Starting Messaging Service server OK or Messaging Service server  
 already running, pid: <PID>
- **stop**:  
 Stopping Messaging Service server OK

## **2.2.17.1**

### **cudbManageMsgSrvServer Examples of Use**

```
<CUDb_node_prompt> cudbManageMsgSrvServer status
```



## 2.2.18 cudbManageStore

The `cudbManageStore` command executes administrative orders over clusters installed in the CUDB node. Refer to *CUDB System Administrator Guide* for more information about using the command.

### cudbManageStore Requisites

The requisites of executing this command are as follows:

- `cudbOperator` users have no access to run this command.
- The command requires execution confirmation for some options.

### cudbManageStore Syntax

- Format (1A): `cudbManageStore -p | --pl -o | --order <orderName> [--no-prompt] [-l | --location <backupPath>] [--Script-path | -S <sqlPath>] [--no-start-replication | -n] [ --no-ldapfe-restart | -x ] [--restore-stored-procedures | -s] [--fresh-backup | -f]`
- Format (1B): `cudbManageStore -p | --pl -o | --order <initndb> -i | --ndbid <ndbid>`
- Format (2A): `cudbManageStore -d | --ds {<dsId>,<dsId>,...} -o | --order} <orderName> [-l | --location <backupPath>] [--Script-path | -S <sqlPath>] [--no-start-replication | -n] [ --no-ldapfe-restart | -x ] [--restore-stored-procedures | -s] [ --fresh-backup | -f ]`
- Format (2B): `cudbManageStore -d | --ds <dsId> -o | --order <initndb> -i | --ndbid <ndbid>`
- Format (3): `cudbManageStore -a | --all -o | --order <orderName> [--no-prompt] [-l | --location <backupPath>] [--Script-path | -S <sqlPath>] [--no-start-replication | -n] [--restore-stored-procedures | -s] [ --fresh-backup | -f ]`
- Format (4): `cudbManageStore -h | --help`

### cudbManageStore Command Options

The following command options can be used:

- `-p | --pl`: Specifies the local PLDB cluster.
- `-d | --ds <dsId>` : Specifies the local DS clusters. `<dsId>` can take the following values:



- 1 to 17: The dsId corresponding to each DSG cluster configured in the CUDB node. Applies to the specified DSG storage clusters.
  - all: Applies to all DSG clusters configured in the CUDB node.
- -a | --all: Refers to all the local clusters (both PLDB and DSG).
- -o | --order <ordername> : Executes the specified administrative order. The allowed orders in a CUDB storage engine are as follows:
- **backup**: Performs a cluster data backup.
  - **initialize**: Starts the cluster from scratch and rules out all its previous data.
  - **maintenance**: Takes the cluster out of service. Not involved in master movement decisions.
  - **initndb**: Restricted for Ericsson personnel.
  - **kickstart**: This order starts replication in the cluster. The order can be used only when a CUDB node is initially installed, when a system data restore or a group data restore is executed, or when a new DSG or PLDB replica is added into the CUDB system.
  - **mysqldslvrestart**: Restricted for Ericsson personnel.
  - **mysqldch1mstrestart**: Restricted for Ericsson personnel.
  - **mysqldch2mstrestart**: Restricted for Ericsson personnel.
  - **mysqldacsrestart**: Restricted for Ericsson personnel.
  - **ndbrestart**: Performs a rolling restart on the database nodes in the cluster.
  - **masterreplicationrestart**: Restarts master processes for a specific storage or DSG
  - **ready**: Joins back the cluster to the CUDB system from maintenance mode.
  - **refresh**: Forces the cluster to read configuration.
  - **restart**: Restarts the cluster taking into account its previous data.
  - **restore**: Loads a previous data backup in the cluster.
  - **start**: Starts the cluster, taking into account its previous data when it was stopped.
  - **status**: Shows the cluster status. The possible return values are as follows:



- is being initialized
- is being restored
- is being performing a backup
- is being stopped
- is Stopped
- is Maintenance Mode
- is being (re)Started
- is being refreshed
- executing kickstart order
- is in ready mode
- **stop**: Stops all processes belonging to the cluster.
- **toggleddebug**: Toggles debug log for the Cluster Supervisor (CS) processes.
- **mgmnoderestart**: Forces to restart the NDB mgm in the node.
- **restorestoredprocedures**: Restores stored procedures for application counters.
- **--no-prompt**: Restricted for Ericsson personnel.
- **-l | --location <backupPath>** : Specifies the backup path from which backup files can be obtained from each NDB node belonging to the involved database cluster. If this path is not provided when the **restore** order is requested, then the default path of the cluster (specified with the BackupDataDir entry of the **config.ini** file in the [ndbd default] section) is applied.
- **-S | --Script-path**: Specifies a SQL script to be run after the **restore** process is successfully finished. If <sqlPath> is not provided, the default script applies. This option is used only to restore the process over a data partition belonging to the PLDB.
- **-i | --ndbid**: Restricted for Ericsson personnel.
- **-n | --no-start-replication**: Used to disable automatic replication start after a successful restore operation. Use this option only after a Group Data Restore. Refer to [CUDB Backup and Restore Procedures](#) for further information.
- **-x | --no-ldapfe-restart**: Used to disable reestablishing connections to the LDAP FE.



- `-f` | `--fresh-backup`: Restricted for Ericsson personnel.
- `-h` | `--help`: Displays command help message and exits.
- `-s` | `--restore-stored-procedures`: Stored procedures of application counters are also restored after a successful **restore** order.

**Note:** The `-l` | `--location`, `-S` | `--Script-path`, `-n` | `--no-start-replication` and `-s` | `--restore-stored-procedures` options are valid only for the **restore** order.

---

## Attention!

Some DSG cluster manipulation orders, such as starting a stopped cluster, or a cluster with in maintenance status, may raise or clear several SNMP alarms for a short period of time. However, once the order is executed, the situation regarding alarms reaches a stable state, coherent with the system state after executing the orders.

---

### cudbManageStore Output

An example of command output is shown below:

```
cudbManageStore stores to process: pl.
Store pl is in ready mode.
cudbManageStore command successful.
```

## 2.2.18.1

### cudbManageStore Examples of Use

The following examples show how to execute `cudbManageStore`. Before all command executions, the following procedure must be performed:

#### Steps

1. Connect to the CUDB node with the following command:

```
<CUDB_node_prompt> ssh <admin_user>@<CUDB_Node_IP_Address>
```

2. Connect to the SC with the following command:

```
<CUDB_node_prompt> ssh <admin_user>@<CUDB_Node_OAM_IP_Address>
```

**Example 2** Some examples of using `cudbManageStore` are provided below:

- Execute the `cudbManageStore` command as follows to stop the PLDB cluster in the current CUDB node:

```
<CUDB_node_prompt> cudbManageStore --pl -o stop
```

The expected output must be similar to the below example:



```
WARNING: Stopping a PLDB replica will take PLDB and all DSG replicas →
on this node out of service. Are you sure you want to continue (y/n)?
```

```
y
```

```
Action will be executed
```

```
cudbManageStore stores to process: pl.
```

```
Launching order stop to pl in dsgroup 0.
```

```
Waiting for stop order(s) to be completed in CUDB Node 44 for stores →
: pl.
```

```
stop order finished succesfully in CUDB Node 44 for store pl.
```

```
stop order(s) completed in CUDB Node 44 for stores : pl.
```

```
Stores where order stop was succesfully completed: pl.
```

```
cudbManageStore command successful.
```

- Execute the `cudbManageStore` command as follows to restart the PLDB and all DSG clusters in the current CUDB node:

```
<CUDB_node_prompt> cudbManageStore -a --order restart
```

The expected output must be similar to the below example:

```
WARNING: Restarting the PLDB replica will take PLDB and all DSG repli →
cas on this node out of service for a while. Are you sure you want to →
continue (y/n)?
```

```
y
```

```
Action will be executed
```

```
cudbManageStore stores to process: pl ds1 (in dsgroup2) ds2 (in ds →
group3).
```

```
Launching order restart to pl in dsgroup 0.
```

```
Launching order restart to ds1 in dsgroup 2.
```

```
Launching order restart to ds2 in dsgroup 3.
```

```
Waiting for restart order(s) to be completed in CUDB Node 44 for sto →
res : pl ds1 ds2.
```

```
restart order finished succesfully in CUDB Node 44 for store ds1.
```

```
restart order finished succesfully in CUDB Node 44 for store ds2.
```

```
restart order finished succesfully in CUDB Node 44 for store pl.
```

```
restart order(s) completed in CUDB Node 44 for stores : pl ds1 ds2.
```

```
Stores where order restart was succesfully completed: ds1 ds2 pl.
```

```
cudbManageStore command successful.
```

- Execute the `cudbManageStore` command as follows to create a backup for the local DS cluster 1:

```
<CUDB_node_prompt> cudbManageStore --ds 1 -o backup
```

The expected output must be similar to the below example:

```
cudbManageStore stores to process: ds1 (in dsgroup1).
```

```
Starting Backup ...
```



```

Launching order Backup for ds1 in dsgroup 1.
Waiting for backup order(s) to be completed in CUDB Node 121 for stor
es : ds1.
backup order finished successfully in CUDB Node 121 for store ds1.
BACKUP-999 renamed in PL_2_7 to /local/cudb/mysql/ndbd/backup/BACKUP/ →
BACKUP-2016-05-12_12-13
BACKUP-999 renamed in PL_2_8 to /local/cudb/mysql/ndbd/backup/BACKUP/ →
BACKUP-2016-05-12_12-13
backup order(s) completed in CUDB Node 121 for stores : ds1.
Stores where order backup was succesfully completed: ds1.
cudbManageStore command successful.

```

- Execute the `cudbManageStore` command as follows to move all local clusters in this CUDB node to maintenance mode:

```
<CUDB_node_prompt> cudbManageStore -d all -o maintenance
```

The expected output must be similar to the below example:

```

WARNING: Getting a DS cluster into maintenance mode will take it out →
of service. Are you sure you want to continue (y/n)?
y
Action will be executed

```

```

cudbManageStore stores to process: ds1 (in dsgroup1) ds2 (in dsgroup →
2).
Launching order maintenance to ds1 in dsgroup 1.
Launching order maintenance to ds2 in dsgroup 2.
Waiting for maintenance order(s) to be completed in CUDB Node 29 for →
stores : ds1 ds2.
maintenance order finished succesfully in CUDB Node 29 for store ds1.
maintenance order finished succesfully in CUDB Node 29 for store ds2.
maintenance order(s) completed in CUDB Node 29 for stores : ds1 ds2.
Stores where order maintenance was succesfully completed: ds1 ds2.
cudbManageStore command successful.

```

- Execute the `cudbManageStore` command as follows to restore backup and application counter procedures on local DS cluster 1:

```
<CUDB_node_prompt> cudbManageStore --ds 1 -o restore --location
<backupPath> -s
```

The expected output must be similar to the below example:

```

WARNING: Restoring a DS cluster will replace previous contents. Are y →
ou sure you want to continue (y/n)?
y
Action will be executed

Starting restore in CUDB Node 42 for store ds1,
backup path /local/cudb/mysql/ndbd/backup/BACKUP/BACKUP-2016-04-11_15 →
-46,

```



```
sql scripts path /home/cudb/storageEngine/config/schema/ds/internal/r →
estoreTempSql.
Waiting for restore order(s) to be completed in CUDB Node 42 for stor →
es : ds1.
restore order finished successfully in CUDB Node 42 for store ds1.
restore order(s) completed in CUDB Node 42 for stores : ds1.
Stores where order restore was successfully completed: ds1.
Closing connections for all blades of DSUnitGroup 1.
Successfully restored application counter procedures for ds1.
cudbManageStore command successful.
```

## 2.2.19

### cudbPmJobReload

---

#### Attention!

This command is deprecated. To restart the ESA Performance Management (PM) agent, execute `/sbin/service esapma restart` in both SCs instead.

---

The `cudbPmJobReload` command restarts the PM agent to allow it to get new configuration. Refer to [CUDB System Administrator Guide](#) for more information on when to use it.

#### cudbPmJobReload Requisites

The requisites of using the command are as follows:

- This command is executed in all CUDB nodes to ensure that the same PM configuration is used in the whole CUDB system.
- `cudbOperator` users have no access to run this command.

#### cudbPmJobReload Syntax

```
cudbPmJobReload [-t | --time <timeout>] [-D | --Debug] [-u | --
user-facility <facility_level>] [-h | --help]
```

#### cudbPmJobReload Command Options

The following command options can be used:

- `-t | --time-out <timeout>` : Sets the timeout, in seconds, to try the operations. The default value is 10 seconds.
- `-D | --Debug`: Restricted for Ericsson personnel.
- `-u | --user-facility <facility-level>` : Restricted for Ericsson personnel.





— -h | --help: Displays command help message and exits.

### **cudbPmJobReload Output**

When executed, the command must return an output similar to the below example:

```
Stopping PmAgent in node 10.22.27.10 ... OK
Waiting for ESA PmAgent to go off line.
ESA PmAgent has been successfully stopped.
Starting PmAgent in node 10.22.27.10 ... OK
```

## **2.2.19.1**

### **cudbPmJobReload Examples of Use**

The following example shows how to execute `cudbPmJobReload`:

#### **Steps**

1. Connect to the CUDB node with the following command:

```
<CUDB_node_prompt> ssh <admin_user>@<CUDB_Node_IP_Address>
```

2. Connect to the SC with the following command:

```
<CUDB_node_prompt> ssh <admin_user>@<CUDB_Node_OAM_IP_Address>
```

3. Execute the command as follows:

```
<CUDB_node_prompt> cudbPmJobReload
```

The expected output must be similar to the below example:

```
Stopping PmAgent in node 10.22.28.1 ... OK
Waiting for ESA PmAgent to go off line.
ESA PmAgent has been successfully stopped.
Starting PmAgent in node 10.22.28.1 ... OK
```

## **2.2.20**

### **cudbPrepareStore**

The `cudbPrepareStore` command creates databases and tables for the PLDB or the specified DS Unit. Refer to *CUDB System Administrator Guide* for further details on when to use it.

#### **cudbPrepareStore Requisites**

`cudbOperator` users have no access to run this command.



### cudbPrepareStore Syntax

- Format 1: `cudbPrepareStore -p | --pl [ -s | --scope <scopeValue>]`
- Format 2: `cudbPrepareStore -d | --ds <dsId> [ -s | --scope <scopeValue>]`
- Format 3: `cudbPrepareStore -a | --all [ -s | --scope <scopeValue>]`

### cudbPrepareStore Command Options

The following command options can be used:

- `-p | --pl`: Runs the command for the local PLDB cluster.
- `-d | --ds <dsId>` : Runs the command for the local DS cluster defined with `<dsId>` . The value of `<dsId>` is the ID of the specific DS Unit, indicated either with an integer between 1-17 (for example, 1 stands for DS1, while 17 means DS17), or with "all" if the command must be executed for all local DS Units.
- `-a | --all`: Runs the command for all local clusters (both PLDB and DSG) configured in the specific CUDB node.
- `-s | --scope <scopeValue>` : Optional command option defining the scope of the command. The value of `<scopeValue>` can be either `appsrv` (referring to application scope) or `all` (referring to all scope).

**Note:** If `--scope <scopeValue>` is not used, the command is executed with the `--scope all` setting by default.

### cudbPrepareStore Output

An example output of the command is shown below:

```
Starting
Obtaining SQL Servers Information ...
Done
Skipping Step 1 ...
Executing Step 2 ...
..ok
Applying /home/cudb/storageEngine/config/schema/ds/appSrv/cudb-ds.sql on →
10.22.23.5:15010
..ok
Applying /home/cudb/storageEngine/config/schema/ds/appSrv/identities-ds. →
sql on 10.22.23.5:15010
..ok
Applying /home/cudb/storageEngine/config/schema/ds/appSrv/msc-ds.sql on →
10.22.23.5:15010
..ok
```



```

Applying /home/cudb/storageEngine/config/schema/ds/appSrv/csps-ds.sql on →
10.22.23.5:15010
..ok
Applying /home/cudb/storageEngine/config/schema/ds/appSrv/nph-ds.sql on →
10.22.23.5:15010
..ok
Applying /home/cudb/storageEngine/config/schema/ds/appSrv/auth-ds.sql on →
10.22.23.5:15010
..ok
Applying /home/cudb/storageEngine/config/schema/ds/appSrv/ftest-ds.sql o →
n 10.22.23.5:15010
..ok
Applying /home/cudb/storageEngine/config/schema/ds/appSrv/avg-ds.sql on →
10.22.23.5:15010
..ok
Applying /home/cudb/storageEngine/config/schema/ds/appSrv/eps-ds.sql on →
10.22.23.5:15010
..ok
Applying /home/cudb/storageEngine/config/schema/ds/appSrv/ims-ds.sql on →
10.22.23.5:15010
..ok
Applying /home/cudb/storageEngine/config/schema/ds/appSrv/sm-ds.sql on 1 →
0.22.23.5:15010
..ok
Applying /home/cudb/storageEngine/config/schema/ds/appSrv/ldap-indexes-d →
s.sql on 10.22.23.5:15010
..ok
..ok
Applying /home/cudb/storageEngine/config/schema/ds/appSrv/cudb-ds.sql on →
10.22.23.6:15010
..ok
Applying /home/cudb/storageEngine/config/schema/ds/appSrv/identities-ds. →
sql on 10.22.23.6:15010
..ok
Applying /home/cudb/storageEngine/config/schema/ds/appSrv/msc-ds.sql on →
10.22.23.6:15010
..ok
Applying /home/cudb/storageEngine/config/schema/ds/appSrv/csps-ds.sql on →
10.22.23.6:15010
..ok
Applying /home/cudb/storageEngine/config/schema/ds/appSrv/nph-ds.sql on →
10.22.23.6:15010
..ok
Applying /home/cudb/storageEngine/config/schema/ds/appSrv/auth-ds.sql on →
10.22.23.6:15010
..ok
Applying /home/cudb/storageEngine/config/schema/ds/appSrv/ftest-ds.sql o →
n 10.22.23.6:15010
..ok
Applying /home/cudb/storageEngine/config/schema/ds/appSrv/avg-ds.sql on →
10.22.23.6:15010
..ok

```



```

Applying /home/cudb/storageEngine/config/schema/ds/appSrv/eps-ds.sql on →
10.22.23.6:15010
..ok
Applying /home/cudb/storageEngine/config/schema/ds/appSrv/ims-ds.sql on →
10.22.23.6:15010
..ok
Applying /home/cudb/storageEngine/config/schema/ds/appSrv/sm-ds.sql on 1→
0.22.23.6:15010
..ok
Applying /home/cudb/storageEngine/config/schema/ds/appSrv/ldap-indexes-d→
s.sql on 10.22.23.6:15010
..ok
Done
Finished

```

### 2.2.20.1 cudbPrepareStore Examples of Use

The following example shows how to prepare the databases and tables on DS1 with cudbPrepareStore in application scope:

#### Steps

1. Connect to the CUDB node with the following command:

```
<CUDB_node_prompt> ssh <admin_user>@<CUDB_Node_IP_Address>
```

2. Connect to the SC with the following command:

```
<CUDB_node_prompt> ssh <admin_user>@<CUDB_Node_OAM_IP_Address>
```

3. Execute the command as follows:

```
<CUDB_node_prompt> cudbPrepareStore --ds 1 -s appsrv
```

#### Example

The expected output must be similar to the below example:

```

Starting
Obtaining SQL Servers Information ...
Done
Skipping Step 1 ...
Executing Step 2 ...
..ok
Applying /home/cudb/storageEngine/config/schema/ds/appSrv/cudb-ds.sql on 10.22.23.5:1501→
0
..ok
Applying /home/cudb/storageEngine/config/schema/ds/appSrv/identities-ds.sql on 10.22.23.→
5:15010
..ok
Applying /home/cudb/storageEngine/config/schema/ds/appSrv/msc-ds.sql on 10.22.23.5:15010
..ok
Applying /home/cudb/storageEngine/config/schema/ds/appSrv/csps-ds.sql on 10.22.23.5:1501→
0
..ok
Applying /home/cudb/storageEngine/config/schema/ds/appSrv/nph-ds.sql on 10.22.23.5:15010
..ok
Applying /home/cudb/storageEngine/config/schema/ds/appSrv/auth-ds.sql on 10.22.23.5:1501→
0

```



```

..ok
Applying /home/cudb/storageEngine/config/schema/ds/appSrv/ftest-ds.sql on 10.22.23.5:15010
..ok
Applying /home/cudb/storageEngine/config/schema/ds/appSrv/avg-ds.sql on 10.22.23.5:15010
..ok
Applying /home/cudb/storageEngine/config/schema/ds/appSrv/eps-ds.sql on 10.22.23.5:15010
..ok
Applying /home/cudb/storageEngine/config/schema/ds/appSrv/ims-ds.sql on 10.22.23.5:15010
..ok
Applying /home/cudb/storageEngine/config/schema/ds/appSrv/sm-ds.sql on 10.22.23.5:15010
..ok
Applying /home/cudb/storageEngine/config/schema/ds/appSrv/ldap-indexes-ds.sql on 10.22.23.5:15010
..ok
..ok
Applying /home/cudb/storageEngine/config/schema/ds/appSrv/cudb-ds.sql on 10.22.23.6:15010
..ok
Applying /home/cudb/storageEngine/config/schema/ds/appSrv/identities-ds.sql on 10.22.23.6:15010
..ok
Applying /home/cudb/storageEngine/config/schema/ds/appSrv/msc-ds.sql on 10.22.23.6:15010
..ok
Applying /home/cudb/storageEngine/config/schema/ds/appSrv/csps-ds.sql on 10.22.23.6:15010
..ok
Applying /home/cudb/storageEngine/config/schema/ds/appSrv/nph-ds.sql on 10.22.23.6:15010
..ok
Applying /home/cudb/storageEngine/config/schema/ds/appSrv/auth-ds.sql on 10.22.23.6:15010
..ok
Applying /home/cudb/storageEngine/config/schema/ds/appSrv/ftest-ds.sql on 10.22.23.6:15010
..ok
Applying /home/cudb/storageEngine/config/schema/ds/appSrv/avg-ds.sql on 10.22.23.6:15010
..ok
Applying /home/cudb/storageEngine/config/schema/ds/appSrv/eps-ds.sql on 10.22.23.6:15010
..ok
Applying /home/cudb/storageEngine/config/schema/ds/appSrv/ims-ds.sql on 10.22.23.6:15010
..ok
Applying /home/cudb/storageEngine/config/schema/ds/appSrv/sm-ds.sql on 10.22.23.6:15010
..ok
Applying /home/cudb/storageEngine/config/schema/ds/appSrv/ldap-indexes-ds.sql on 10.22.23.6:15010
..ok
Done
Finished

```

## 2.2.21 cudbReallocate

The `cudbReallocate` command performs the Subscription Reallocation (also known as "reallocation") of the Distribution Entries (DEs) between DSGs. When executing the command, the complete DE subtree is moved. If one or more DSGs is disabled for provisioning of DEs, reallocation will not be performed to such DSGs. In case when DSG disabled for provisioning of DEs is specified as reallocation destination, reallocation will be stopped. When the reallocation destination is not specified, the command will select one or more suitable DSGs which are not blocked for provisioning as reallocation targets. In both cases, reallocation can be forced if optional parameter is used.

Refer to [CUDB Subscription Reallocation](#) for more information on when to use the command.

Use the reallocation command during the low-traffic hours and use it repeatedly. Either way, use it with small chunks of subscribers (`--list`) or small percentages



(`--entriespercentage`) at a time, to make sure that reallocating those small chunks of data fits into the low-traffic time window.

### **cuDbReallocate Requisites**

The requisites of using the command are as follows:

- Execute the command in the node where the PLDB master replica is located. For more information, refer to [CUDB System Administrator Guide](#).
- `cuDbOperator` users have no access to run this command.

### **cuDbReallocate Syntax**

- Format (1): Reallocate by moving DS entries from the source DSG:  
**cuDbReallocate -s | --source <dsgId> [-d | --destination <dsgId>] -p | --entriespercentage <number 1-100> [-o | --output <filepath>] [-D | --Debug] [-f | --force] [-n | --nthreads <number>]**
- Format (2): Reallocate a list of elements to a specified destination:  
**cuDbReallocate -d | --destination <dsgId> -l | --list <filepath> [-o | --output <filepath>] [-D | --Debug] [-f | --force] [-n | --nthreads <number>]**
- Format (3): Request the status of DSGs relevant to the reallocate command:  
**cuDbReallocate -u | --status [-D | --Debug]**
- Format (4): Show command help message: **cuDbReallocate [-h | --help]**

### **cuDbReallocate Command Options**

The following command options can be used:

- `-u | --status`: Prints the occupancy percentage and the memory level of all DSGs in the system. The memory occupation of the DSG is determined by the replica with the highest occupation. The possible status of the output are the following:
  - Below the eligible level.
  - Below the stop level.
  - Below the warning level.
  - Above the warning level.
  - Above the full level.



- `-s | --source <dsgId>` : Defines the source DSG with `<dsgId>` . The value of `<dsgId>` must be an integer between 1-255. Refer to [CUDB Node Configuration Data Model Description](#) for more information.
- `-d | --destination <dsgId>` : Defines the destination DSG on which entries are reallocated. The value of `<dsgId>` must be an integer between 1-255. If this command option is not used, then the command selects one or more suitable DSGs as reallocation targets automatically.
- `-p | --entriespercentage <number 1-100>` : Defines the percentage of the DS entries in the specific source DSG that is required to be moved away. The value must be an integer between 1-100.
- `-l | --list`: Defines the file containing the list of primary network identities which identify the DEs to reallocate. The file contains lines of strings with the following format: `netIDType:= netIDValue`. The file is case sensitive.

Typical values of `netIDType` for common applications are as follows:

- IMPI
- IMPU
- IMSI
- IPADDRESS
- MSISDN

An example for these values is provided below:

`IMSI:=99999`

`MSISDN:=100003`

- `-o | --output` : Defines the output file where information on DE reallocation failures is stored. The default name of the output file is `cudbReallocationResult <YYYY-MM-DD-HH24-MI>.txt`, and is stored in the current directory. The file is deleted if the `cudbReallocate` command finishes without errors. The contents of the output file depend on whether the `--list` option was used when executing the command:
  - If the `--list` option was used when executing the command, the output file contains one line per failed network identity, with the following format:
 

```
netIDType:= netIDValue; result <result>
```

In the above example, the value of `<result>` can be as follows:

    - `already moved`, if the entry was already in the destination DSG.



- `invalid` entry, if the specified network identity cannot be translated into a valid DE.
- `not moved`, in case of any other faulty situation.
- If the `--list` option was not used when executing the command, then each line in the output file states the DN of one of the DEs which could not be moved. If the DN was moved, but there was an error deleting it from the source DSG, an extra "moved"-suffix is added to the DN.
- `-n` | `--nthreads`: Restricted for Ericsson personnel.
- `-D` | `--Debug`: Restricted for Ericsson personnel.
- `-f` | `--force`: Forces reallocation if destination DSG is disabled for provisioning.
  - If destination DSG is specified and that DSG is disabled for provisioning, `--force` option allows reallocation over that DSG.
  - If destination DSG is not specified and one or more DSGs are disabled for provisioning, `--force` option allows reallocation over such DSGs.
- `-h` | `--help`: Displays command help message and exits.

### **cudbReallocate Output**

The following list contains some example output messages:

- Reallocating from DSG <In case the reallocation command is triggered<id>> t→  
o DSG <id>.  
Reallocation is ongoing.  
Reallocation is ongoing, approximately 10% entries reallocated so far →  
.  
...  
Reallocation is finished. All entries reallocated.  
cudbReallocate: Success.
- Interrupt signal caught.  
Aborting reallocation process. Please wait...  
Reallocation terminated by user.
- Reallocating from DSG <id> to DSG <id>.  
cudbReallocate: Partial Success.  
Reallocate DSG <n> partially succeeded - unable to move some entries. →  
  
Consult output file for entries that could not be moved.
- Reallocating from DSG <id> to DSG <id>.  
cudbReallocate: Partial Success.  
Reallocate DSG <n> partially succeeded - All entries were not moved to destination DSG <y>→





- Refer to output file for details of failure entries where moved.  
 An entry of "already moved" indicates the entry is already in the destination DSG.  
 An entry of "invalid entry" indicates that the entry was not found or is not a primary network identity.  
 An entry of "not moved" indicates that could not be moved due to a temporary failure.
- Reallocating from DSG <id> to DSG <id>.  
 cudbReallocate: Partial Success.  
 Reallocate DSG <n> partially succeeded - All entries were not moved to destination DSG <y>→  
  
 No entries were moved to destination DSG <y>.  
 An entry of "already moved" indicates the entry is already in the destination DSG.  
 An entry of "invalid entry" indicates that the entry was not found or is not a primary network identity.  
 An entry of "not moved" indicates that could not be moved due to a temporary failure.
  - cudbReallocate: Failed.  
 Unexpected error.
  - cudbReallocate: Failed.  
 Invalid format: unrecognised option '<option>'  
 ...
  - cudbReallocate: Failed.  
 Reconciliation is ongoing.
  - cudbReallocate: Failed.  
 Source DSG <n> is not valid.
  - cudbReallocate: Failed.  
 Invalid format: Invalid DSG value, should be in range 1 - 255
  - cudbReallocate: Failed.  
 The list <inputFile> could not be found or could not be opened.
  - cudbReallocate: Failed.  
 The output <outputFile> already exists.
  - cudbReallocate: Failed.  
 <outputFile>  
 Reallocate DSG <n> failed - DS master change detected in source DSG <y>.
  - cudbReallocate: Failed.  
 <outputFile>  
 Reallocate DSG <x> failed - DS master change detected in destination DSG <y>.
  - cudbReallocate: Failed.  
 <outputFile>  
 Reallocate DSG <n> failed - PLDB master change detected in destination.
  - cudbReallocate: Failed.  
 <outputFile>  
 Reallocate DSG <n> failed - PLDB master change detected in destination →  
 n.
  - cudbReallocate: Failed.  
 Invalid format: Parameter [list] requires parameter [destination]  
 ...



- cudbReallocate: Failed.  
Invalid format: Parameter [source] and parameter [list] are mutually exclusive  
...
- cudbReallocate: Failed.  
Invalid format: Source and destination DSG IDs should not be identical  
1  
...
- cudbReallocate: Failed.  
PLDB master is not in the local node.
- cudbReallocate: Failed.  
PLDB is not present in the local node.
- cudbReallocate: Failed.  
CudbReallocate is already running.
- cudbReallocate: Failed.  
The output <outputFile> file opening error.
- cudbReallocate: Failed.  
Destination DSG <n> is not valid.
- cudbReallocate: Failed.  
CudbReallocate is already running.

**Note:** This output is displayed after an unexpected error in the command execution. This output is also displayed if reallocation is in process. If no other reallocation process is running, this output is displayed due to a previous reallocation process finished with an unexpected error. To unlock cudbReallocate for a particular DSG, use the following command:

```
rm -rf /cluster/home/  
cudb/.lock.SC_2_?.cudbReallocate.DSG<x>
```

x - the ID of the particular DSG

Do not delete this file if the command is up and running, to prevent unexpected behaviors.

If destination DSG is blocked for provisioning, the example of output message is:

- Destination DSG 255 is blocked for reallocation.



## 2.2.21.1 cudbReallocate Examples of Use

### 2.2.21.1.1 How to Check the Occupancy Percentage and Memory Level of All DSGs

The following example shows how to check the occupancy percentage and memory level of all DSGs in the system with the cudbReallocate command:

#### Steps

1. Connect to the CUDB node with the following command:

```
<CUDB_node_prompt> ssh <admin_user>@<CUDB_Node_IP_Address>
```

2. Connect to the SC with the following command:

```
<CUDB_node_prompt> ssh <admin_user>@<CUDB_Node_OAM_IP_Address>
```

3. Execute the command as follows:

```
<CUDB_node_prompt> cudbReallocate --status
```

The expected output must be similar to the below example:

```
cudbReallocate: Success
DSG1 occupancy: 40%, Above the warning level
DSG2 occupancy: 40%, Below the stop level
```

**Note:** In case of error, the output file is generated with no moved entries.

### 2.2.21.1.2 How to Force Reallocation if One DSG is Blocked for Provisioning of DEs

The following example shows how to force reallocation if one DSG is blocked for provisioning of DEs and it is specified as reallocation destination:

#### Steps

1. Connect to the CUDB node with the following command:

```
<CUDB_node_prompt> ssh <admin_user>@<CUDB_Node_IP_Address>
```

2. Connect to the SC with the following command:

```
<CUDB_node_prompt> ssh <admin_user>@<CUDB_Node_OAM_IP_Address>
```

3. Execute the command as follows:

```
<CUDB_node_prompt> cudbReallocate -p 20 -s 1 -d 255 -f
```

The expected output must be similar to the below example:



```
cudbReallocate: Failed.  
Destination DSG 255 is blocked for reallocation.
```

## 2.2.22 cudbReconciliationMgr

The `cudbReconciliationMgr` command allows to check, print, and modify the pending reconciliation task list. Refer to [CUDB Data Storage Handling](#) for further details about when to use it.

### cudbReconciliationMgr Requisites

`cudbOperator` users have no access to run this command.

### cudbReconciliationMgr Syntax

- Format (1): `cudbReconciliationMgr -c | --check <dsId>`
- Format (2): `cudbReconciliationMgr -a | --add <dsId>`
- Format (3): `cudbReconciliationMgr -l | --list`
- Format (4): `cudbReconciliationMgr [-h | --help]`

### cudbReconciliationMgr Command Options

The following command options can be used:

- `-c | --check <dsId>` : Checks if the DSG identified with `<dsId>` is in the Pending Task List.
- `-a | --add`: Adds a new pending reconciliation task for the DSG if the DS master is on the current CUDB node.
- `-l | --list`: Lists all pending tasks, that is, the contents of the Pending Task List.
- `-h | --help`: Displays command help message and exits. If the command is executed without options, the command use and help information is shown.

### cudbReconciliationMgr Output

The command output is a log of the execution progress. The command result is provided at the end of the log.

### 2.2.22.1 cudbReconciliationMgr Examples of Use

The following examples show how to execute `cudbReconciliationMgr`. Before all command executions, the following procedure must be performed:



## Steps

1. Connect to the CUDB node with the following command:

```
<CUDB_node_prompt> ssh <admin_user>@<CUDB_Node_IP_Address>
```

2. Connect to the SC with the following command:

```
<CUDB_node_prompt> ssh <admin_user>@<CUDB_Node_OAM_IP_Address>
```

## Example 3

Some examples of using `cudbReconciliationMgr` are provided below:

- Execute `cudbReconciliationMgr` as shown below to check if DS1 is in the Pending Task List:

```
<CUDB_node_prompt> cudbReconciliationMgr -c 1
```

If DS1 is on the Pending Task List, the command has no output. Otherwise, the output provides the error(s) fetched from the database.

- Execute `cudbReconciliationMgr` as shown below to add a new pending reconciliation task for DS2:

```
<CUDB_node_prompt> cudbReconciliationMgr -a 2
```

If the task for DS2 is added, the command has no output. Otherwise, the output provides the error(s) fetched from the database.

## 2.2.23

### `cudbRemoteTrust`

The `cudbRemoteTrust` command is used in scalability procedures during installation to establish new SSH trust relationships between the installed CUDB node and the rest of the CUDB nodes.

It is also used to disable legal warning banner to be displayed for internal CUDB logins. Refer to Ericsson personnel for more information.

### `cudbRemoteTrust` Requisites

The requisites of using the command are as follows:

- This command must be executed in all CUDB nodes.
- `cudbOperator` users have no access to run this command.

### `cudbRemoteTrust` Syntax

```
cudbRemoteTrust [-l | --local] [-a | --all] [-p | --persistent]
[-b | --banner] [-h | --help]
```



### cudbRemoteTrust Command Options

The following command options can be used:

- **-l | --local**: Establishes the trust relationships in the current blade or VM of the CUDB node where the command is executed.
- **-a | --all**: Establishes the trust relationships in every blade or VM of the CUDB node where the command is executed.
- **-p | --persistent**: Stores the trust relationship settings in the following location:

`/home/cudb/common/config/sshTrust`

**Note:** This command option can be executed only if the **--all** option is also executed.

This command option implies the execution of **--banner** option, too.

- **-b | --banner**: Disables legal warning banner for logins, targeting the local CUDB node over SSH, in case the source address of the SSH connection is a `SITE_VIP` from the `CUDB_SITE` network. For example, when an internal CUDB login is initiated from a remote node towards the local node.

**Note:** This command option will also be executed when the **--all --persistent** combined options are executed.

- **-h | --help**: Displays command help message and exits.

### cudbRemoteTrust Output

Successful execution provides no output.

#### 2.2.23.1

### cudbRemoteTrust Examples of Use

The following example shows how to execute `cudbRemoteTrust` on every blade or VM of the current CUDB node:

#### Steps

1. Connect to the CUDB node with the following command:

```
<CUDB_node_prompt> ssh <admin_user>@<CUDB_Node_IP_Address>
```

2. Connect to the SC with the following command:

```
<CUDB_node_prompt> ssh <admin_user>@<CUDB_Node_OAM_IP_Address>
```

3. Execute the command as follows:

```
<CUDB_node_prompt> cudbRemoteTrust -a
```



## 2.2.24 cudbResumeProvisioningNotification

The `cudbResumeProvisioningNotification` command is used to send a notification to the Provisioning Gateway (PG) to resume provisioning. Refer to *Storage Engine, Backup Notification Failure To Provisioning Gateway* for further details about when to use it.

### cudbResumeProvisioningNotification Requisites

`cudbOperator` users have no access to run this command.

### cudbResumeProvisioningNotification Syntax

```
cudbResumeProvisioningNotification [-u | --user-facility  
<facility>] [-D | --debug] [-h | --help]
```

### cudbResumeProvisioningNotification Command Options

The following command options can be used:

- `-u | --user-facility <facility>` : Restricted for Ericsson personnel.
- `-D | --debug`: Restricted for Ericsson personnel.
- `-h | --help`: Displays command help message and exits.

### cudbResumeProvisioningNotification Output

The command output is a log of the changes performed. The command result is provided at the end of the log.

## 2.2.24.1 cudbResumeProvisioningNotification Examples of Use

The following example shows how to execute `cudbResumeProvisioningNotification`:

### Steps

1. Connect to the CUDB node with the following command:

```
<CUDB_node_prompt> ssh <admin_user>@<CUDB_Node_IP_Address>
```

2. Connect to the SC with the following command:

```
<CUDB_node_prompt> ssh <admin_user>@<CUDB_Node_OAM_IP_Address>
```

3. Execute the command as follows:

```
<CUDB_node_prompt> cudbResumeProvisioningNotification
```



## 2.2.25 cudbServiceContinuity

The `cudbServiceContinuity` command is used to change DSG mastership manually to working that have been left in minority after a CUDB system split situation. The `cudbServiceContinuity` command maximizes service, allowing traffic operation with the resources available in the minority partition. New masters will be elected for all DSGs configured in that minority partition.

**Note:** PLDB mastership is not affected by the command.

Refer to [CUDB High Availability](#) for further details about using the command.

### cudbServiceContinuity Requisites

The requisites of using the command are as follows:

- The partition must be in a system split situation of minority to accept this command.
- The command requires execution confirmation.

### cudbServiceContinuity Syntax

`cudbServiceContinuity [-h | --help]`

### cudbServiceContinuity Command Options

The following command option can be used:

- `-h | --help`: Displays command help message and exits.

### cudbServiceContinuity Output

The command can return the following output messages:

- In case of a minority system split situation, the command returns the following success message:

```
cudbServiceContinuity
WARNING: This command will take all masters excluding PLDB in this partition! →
Are you sure you want to continue (y/n)? y
The node /cudb/actions already existed
Set in path: /cudb/actions DATA: serviceContinuity
Service Continuity successfully executed
```

- In case the execution of the command is not confirmed (n is selected instead of y), the command returns the following message:

```
cudbServiceContinuity
WARNING: This command will take all masters excluding PLDB in this partition! →
Are you sure you want to continue (y/n)? n
No action executed
```





- When executed with the `-h` or `--help` option, the command returns following output:

```
cudbServiceContinuity -h
=====
                        USE of cudbserviceContinuity
=====
Format : cudbserviceContinuity [-h | --help]
Where:
        -h | --help           Shows command usage and help
```

- In case of a majority system split situation, the command returns the following failure message:

```
cudbServiceContinuity
WARNING: This command will take all masters excluding PLDB in this partition! →
Are you sure you want to continue (y/n)? y
SC can not be set in a majority (majority[1, 2, 3] AR[] NS[])
Command failed
```

- In case of a symmetrical split situation, the command returns the following failure message:

```
cudbServiceContinuity
WARNING: This command will take all masters excluding PLDB in this partition! →
Are you sure you want to continue (y/n)? y
SC can not be set in a evenSplit (evenSplit[2] AR[] NS[])
Command failed
```

**Note:** If the command fails for other reasons than the ones above, contact the next level of maintenance support.

### 2.2.25.1 cudbServiceContinuity Examples of Use

```
<CUDB_node_prompt> cudbServiceContinuity
```

### 2.2.26 cudbSwBackup

The `cudbSwBackup` command performs a software and configuration backup in the CUDB node. The command rotates software and configuration backup files automatically, and asks for confirmation when the oldest backup is about to be deleted. Refer to *CUDB System Administrator Guide* for more information on using the command.

#### cudbSwBackup Requisites

The requisites of using the command are as follows:

- `cudbOperator` users have no access to run this command.
- The command requires execution confirmation for some options.



### cudbSwBackup Syntax

- Format (1): `cudbSwBackup [-c | --create <backup name>] [[-f | --force] -r | --restore <backup name>] [-d | --delete <backup name>] [-i | --info] [-l | --list] [-p | --print] [-h | --help]`
- Format (2): `cudbSwBackup -o | --delete-oldest -c | --create <backup name>`
- Format (3): `cudbSwBackup [--no-prompt] --restore <backup name>`
- Format (4): `cudbSwBackup -s | --schedule '<cron expression>'`
- Format (5): `cudbSwBackup -U | --Unschedule '<cron expression>'`

### cudbSwBackup Command Options

The following command options can be used:

- `-c | --create <backup name>` : Creates a new CUDB software and configuration backup.
  - `-r | --restore <backup name>` : Restores an already created CUDB software and configuration backup.
  - `--no-prompt`: Restricted for Ericsson personnel.
  - `-d | --delete <backup name>` : Deletes an already created CUDB software and configuration backup.
  - `-i | --info`: Prints information about the CUDB and LDE backup location and file format.
  - `-l | --list`: Lists the created CUDB software and configuration backups.
  - `-s | --schedule <cron expression>` : Creates a periodic software and configuration backup according to the defined `<cron expression>` .
  - `-U | --Unschedule <cron expression>` : Remove a periodic software and configuration backup according to the defined `<cron expression>` .
- Note:** The format of the `<cron expression>` follows the standard UNIX cron expression format.
- `-p | --print`: Prints the information on the scheduled periodic software and configuration backup.
  - `-o | --delete-oldest`: Removes the oldest backup without confirmation message, if more than 5 backups exist.

**Note:** This option must be followed by the `-c | --create` option.



- `-f` | `--force` : Allows SW restore when the mysqls of the PLDB are down (sql backup is not restored) or the CUDB node does not have one defined.

**Note:** If the PLDB cluster is defined, is up, and replication is working after the LDE cluster reboot, restore the system monitor tables by executing the `cudbApplyConfig -s init` command. Otherwise, refer to the procedure in [Storage Engine, Unable to Synchronize Cluster in PLDB, Major](#) to get replication up and running and to restore the sql backup.

- `-h` | `--help`: Displays command help message and exits.

### cudbSwBackup Output

The output of the command is a log of the performed actions. An example output is provided below:

```
Pack and compress all directories and files under /home/cudb/* except swbackup, systemDat →
aBackup and automatedBackupStorage directories.
```

```
IMM data persisted
Obtaining SQL Servers Information ...
Creating tables backup into file /cluster/home/cudb/swbackup/newbackup-cudbSmpConfig.sql
Connecting to host 10.22.49.5:15000 , database: cudb_system_monitor
Let's tar the directory /cluster/home/cudb excluding backup directory itself /cluster/hom →
e/cudb/swbackup/
```

...

```
Let's execute the lde-brf command: cluster brf create -l newbackup -t system -m newbackup
The resulting file is saved in /cluster/storage/no-backup/newbackup.tar
Snapshot cleanup completed
newbackup/config.md5sum
newbackup/config.metadata
newbackup/config.tar.gz
newbackup/software.md5sum
newbackup/software.metadata newbackup/software.tar.gz
Backup successfully created.
The backup files: newbackup, are located in directories:
/cluster/home/cudb/swbackup and
/cluster/storage/no-backup
```

#### 2.2.26.1 cudbSwBackup Examples of Use

The following example shows how to create a new software and configuration backup (named newbackup) with `cudbSwBackup`:

##### Steps

1. Connect to the CUDB node with the following command:



```
<CUDB_node_prompt> ssh <admin_user>@<CUDB_Node_IP_Address>
```

2. Connect to the SC with the following command:

```
<CUDB_node_prompt> ssh <admin_user>@<CUDB_Node_OAM_IP_Address>
```

3. Execute the command as follows:

```
<CUDB_node_prompt> cudbSwBackup -c newbackup
```

The expected output must be similar to the below example:

```
Let's execute the lde-brf command: cluster brf create -l newbackup -t →
system -m newbackup
```

```
The resulting files will be saved in /cluster/storage/no-backup/newba →
ckup.tar
```

```
Snapshot cleanup completed
```

```
newbackup/config.md5sum
```

```
newbackup/config.metadata
```

```
newbackup/config.tar.gz
```

```
newbackup/software.md5sum
```

```
newbackup/software.metadata
```

```
newbackup/software.tar.gz
```

```
Backup successfully created.
```

```
The backup files: newbackup, are located in directories:
```

```
    /cluster/home/cudb/swbackup and
```

```
    /cluster/storage/no-backup
```

```
<CUDB_node_prompt> cudbSwBackup -i
```

```
CUDB SW Backups are stored in two directories:
```

```
    /cluster/home/cudb/swbackup and
```

```
    /cluster/storage/no-backup
```

```
The backup files are:
```

```
    - One tar backup file
```

```
    - One gz backup file
```

The following example shows how to restore a previously created software and configuration backup (named newbackup) with cudbSwBackup:

4. Connect to the CUDB node with the following command:

```
<CUDB_node_prompt> ssh <admin_user>@<CUDB_Node_IP_Address>
```

5. Connect to the SC with the following command:

```
<CUDB_node_prompt> ssh <admin_user>@<CUDB_Node_OAM_IP_Address>
```

6. Execute the command as follows:

```
<CUDB_node_prompt> cudbSwBackup --restore newbackup
```

The expected output must be similar to the below example:



```

WARNING: Restoring the CUDB software will replace previous software a→
nd configuration for the CUDB node. Are you sure you want to continue→
(y/n)?y
Check for ongoing or pending applyConfig...
Waiting for OIWorker to write to file and release lock.
Obtaining SQL Servers Information ...
Checking SQL connections...

```

## 2.2.27

### cudbSwVersionCheck

The `cudbSwVersionCheck` command gathers information on the installed software packages from both the object model and LDE, and then prints the current installation state for each node in the LDE cluster. It also compares the current state to the official package list of the installed CUDB release, and shows the differences between the reference file and the currently installed packages.

The package information listing and the comparison of the installed packages can be requested separately with the use of the `--packages` and `--compare` options, respectively. If none of these options are used when executing the command, both actions are performed.

#### cudbSwVersionCheck Requisites

`cudbOperator` users have no access to run this command.

#### cudbSwVersionCheck Syntax

```

cudbSwVersionCheck [-p | --packages] [-c | --compare] [(-f | --
reffile <REFFILE>) | (-d | --refdir <REFDIR>)] [-h | --help]

```

#### cudbSwVersionCheck Command Options

The following command options can be used:

- `-h | --help`: Displays command help message and exits.
- `-p | --packages`: Prints package installation status.
- `-c | --compare`: Compares the package installation state to the `cudbReference` file and prints the differences, if there is any.

**Note:** Omitting the `-c` and `-p` options has the same effect as using both of them.

- `-f | --reffile <REFFILE>` : Restricted for Ericsson personnel.
- `-d | --refdir <REFDIR>` : Restricted for Ericsson personnel.

**cudbSwVersionCheck Output**

An example output is provided below:

**Example 4**

CUDB\_67 SC\_2\_1# cudbSwVersionCheck

Checking SW on blades..... →  
 .....

SW num	2_1	2_2	2_3	2_4	2_5	2_6	2_7	2_8	2_9	2_10
1	X0	X0								
2			X0	X0	X0	X0	X0	X0	X0	X0
3	X0	X0								
4	X0	X0	X0	X0	X0	X0	X0	X0	X0	X0
5	X0	X0								
6	X0	X0								
7	X0	X0	X0	X0	X0	X0	X0	X0	X0	X0
8	X0	X0	X0	X0	X0	X0	X0	X0	X0	X0
9	X0	X0								
10	X0	X0								
11	X0	X0	X0	X0	X0	X0	X0	X0	X0	X0
12	X0	X0								
13	X0	X0								
14	X0	X0								
15	X0	X0								
16	X0	X0								
17	X0	X0								
18	X0	X0	X0	X0	X0	X0	X0	X0	X0	X0
19			X0	X0	X0	X0	X0	X0	X0	X0
20	X0	X0	X0	X0	X0	X0	X0	X0	X0	X0
21	X0	X0								
22			X0	X0	X0	X0	X0	X0	X0	X0
23	X0	X0								
24	X0	X0								
25	X0	X0								
26	X0	X0	X0	X0	X0	X0	X0	X0	X0	X0
27	X0	X0	X0	X0	X0	X0	X0	X0	X0	X0
28	X0	X0	X0	X0	X0	X0	X0	X0	X0	X0
29	X0	X0	X0	X0	X0	X0	X0	X0	X0	X0
30	X0	X0	X0	X0	X0	X0	X0	X0	X0	X0
31	X0	X0	X0	X0	X0	X0	X0	X0	X0	X0
32	X0	X0	X0	X0	X0	X0	X0	X0	X0	X0
33	X0	X0	X0	X0	X0	X0	X0	X0	X0	X0
34			X0	X0						
35	X0	X0								
36	X0	X0								
37			X0	X0	X0	X0	X0	X0	X0	X0
38	X0	X0	X0	X0	X0	X0	X0	X0	X0	X0
39	X0	X0	X0	X0	X0	X0	X0	X0	X0	X0
40	X0	X0	X0	X0	X0	X0	X0	X0	X0	X0



41	X0	X0		X0	X0	X0	X0	X0	X0	X0	X0	
42	X0	X0										
43				X0	X0	X0	X0	X0	X0	X0	X0	
44	X0	X0										
45	X0	X0		X0	X0	X0	X0	X0	X0	X0	X0	
46	X0	X0		X0	X0	X0	X0	X0	X0	X0	X0	
47	X0	X0										
48	X0	X0										
49				X0	X0	X0	X0	X0	X0	X0	X0	
50	X0	X0										
51		X0										
52	X0	X0		X0	X0	X0	X0	X0	X0	X0	X0	
53	X0	X0										
54	X0	X0										
55	X0	X0										
56	X0	X0										
57	X0	X0										
58	X0	X0										

## LEGEND:

X: queried by the "cmw-repository-list --node <blade> <sw\_string>" comma →  
nd

O: queried by the "immlist <sw\_string>" command

SDPs imported to the LOTC cluster:

-----

SW num	SW string	Used
1	ERIC-COREMW_SC-CXP9017565_1-R5K01	X0
2	ERIC-LINUX_PAYLOAD-CXP9013152_3-R4F04	X0
3	ERIC-CUIDB_SMBC_PROCESS-CXP9030366_6-R1A02	X0
4	ERIC-CUIDB_BC_SERVER-CXP9030364_6-R1A01	X0
5	ERIC-CUIDB_BC_CLIENT-CXP9030367_6-R1A02	X0
6	ERIC-CUIDB_LDAPFE_MONITOR-CXP9015809_6-R1A10	X0
7	ERIC-CUIDB_CBATMPATCHES-CXP9030357_6-R1A10	X0
8	ERIC-COREMW_OPENSFA-CXP9017656_1-R5K01	X0
9	ERIC-CUIDB_APPCOUNTERS-CXP9015318_6-R1A10	X0
10	ERIC-CUIDB_BC_SERVER_MONITOR-CXP9030379_6-R1A01	X0
11	ERIC-COREMW_COMMON-CXP9017566_1-R5K01	X0
12	ERIC-COM-CXP9017585_2-R6K01	X0
13	ERIC-ComSa-CXP9017697_3-R2K01	X0
14	ERIC-CUIDB_ESA-CXP9015816_6-R1A09	X0
15	ERIC-CUIDB_CURATOR-CXP9030365_6-R1A02	X0
16	ERIC-CUIDB_BCCLUSTER_CMD-CXP9030380_6-R1A01	X0
17	ERIC-CUIDB_BACKUP_RESTORE-CXP9015818_6-R1A10	X0
18	ERIC-CUIDB_BACKUP_CMD-CXP9016194_6-R1A05	X0
19	ERIC-CUIDB_LDAPFE-CXP9015319_6-R1A10	X0
20	ERIC-CUIDB_KEEPLIVE-CXP9015322_6-R1A10	X0
21	ERIC-CUIDB_CS_SC_CMD-CXP9030358_6-R1A10	X0
22	ERIC-CUIDB_CS_PL_CMD-CXP9019599_6-R1A10	X0
23	ERIC-CUIDB_CS-CXP9015812_6-R1A10	X0



24		ERIC-CUDB_COUNTCOLLECTOR-CXP9015814_6-R1A07		X0	
25		ERIC-CUDB_LIB_REALLOC-CXP9030372_6-R1A10		X0	
26		ERIC-CUDB_LIB_BCCLI-CXP9030363_6-R1A01		X0	
27		ERIC-CUDB_JAVA-CXP9015823_6-R1A04		X0	
28		ERIC-CUDB_MANAGE_BCSERVER_CMD-CXP9030381_6-R1A01		X0	
29		ERIC-CUDB_LIB_BOOST-CXP9030227_6-R1A04		X0	
30		ERIC-CUDB_LIB_ALARMS-CXP9030217_6-R1A10		X0	
31		ERIC-CUDB_LIB_SMMCP-CXP9015821_6-R1A10		X0	
32		ERIC-CUDB_LIB_LDAP-CXP9030233_6-R1A10		X0	
33		ERIC-CUDB_LIB_AMF-CXP9017480_6-R1A10		X0	
34		ERIC-CUDB_LDAP_SOAP_NOTIF-CXP9016181_6-R1A10		X0	
35		ERIC-CUDB_LDAP_CMD-CXP9030234_6-R1A03		X0	
36		ERIC-CUDB_REALLOC_CMD-CXP9030373_6-R1A10		X0	
37		ERIC-CUDB_MYSQL_NDBD-CXP9030225_6-R1A02		X0	
38		ERIC-CUDB_MYSQL_COMMON-CXP9030223_6-R1A10		X0	
39		ERIC-CUDB_LIB_LOGGER-CXP9030219_6-R1A10		X0	
40		ERIC-CUDB_LIB_CDQX-CXP9030216_6-R1A10		X0	
41		ERIC-CUDB_LIB_CONF_MGMT-CXP9030218_6-R1A10		X0	
42		ERIC-CUDB_MYSQL_MGMD-CXP9030224_6-R1A02		X0	
43		ERIC-CUDB_PL_PLATFORM-CXP9015820_6-R1A10		X0	
44		ERIC-CUDB_LOADMONITOR-CXP9017650_6-R1A10		X0	
45		ERIC-CUDB_LIB_XERCES-CXP9030220_6-R1A01		X0	
46		ERIC-CUDB_LIB_ZKCLI-CXP9030362_6-R1A01		X0	
47		ERIC-CUDB_PDSH_CMD-CXP9030391_6-R1A01		X0	
48		ERIC-CUDB_OI-CXP9015813_6-R1A10		X0	
49		ERIC-CUDB_MYSQL_SERV-CXP9030226_6-R1A02		X0	
50		ERIC-CUDB_RECONCILIATION-CXP9015808_6-R1A10		X0	
51		ERIC-CUDB_NODE_CONFIG-CXP9015320_6-R1A10		X0	
52		ERIC-CUDB_SECURITY-CXP9016002_6-R1A10		X0	
53		ERIC-CUDB_SYSHEALTH-CXP9030087_6-R1A10		X0	
54		ERIC-CUDB_SC_PLATFORM-CXP9015817_6-R1A10		X0	
55		ERIC-CUDB_SLES_SCREEN_CMD-CXP9030371_6-R1A01		X0	
56		ERIC-LINUX_CONTROL-CXP9013151_3-R4F04		X0	
57		ERIC-CUDB_SW_MGMT-CXP9015815_6-R1A10		X0	
58		ERIC-CUDB_SM_CLIENT-CXP9015810_6-R1A10		X0	

SW Version of the CUDB Node:

-----

Reference: /home/coremw\_appdata/incoming/cudb-install-temp/cudbReference

Version: CUDB13B CXP9020214/6 R1A10

Differences:

reference <|> current sw vers →

ion

----- →

---

ERIC-CUDB\_SM\_PROCESS-CXP9015811\_6-R1A10

<





### 2.2.27.1 cudbSwVersionCheck Examples of Use

Execute the command as follows to print the package installation status and compare the package installation state to the contents of the cudbReference file:

```
<CUDB_node_prompt> cudbSwVersionCheck --packages --compare
```

### 2.2.28 cudbSystemDataBackupAndRestore

The `cudbSystemDataBackupAndRestore` command automates the system data backup and restore processes. In case of `kickstart` command failure, the execution is not stopped. Furthermore, a retry mechanism exists, which executes the `kickstart` command a maximum of 2 more times with a 5-second delay in-between them. The encountered errors are stored and displayed in a summary at the script exit. In case of error, the exit code of `cudbSystemDataBackupAndRestore` is the number of errors encountered during execution. The `cudbSystemDataBackupAndRestore` script exits when running `kickstart_replication` and leaves most of the slave DS replica without the replication started. Refer to [CUDB Backup and Restore Procedures](#) for further details on when to use the command.

**Note:** Do not run the command `cudbSystemDataBackupAndRestore` while configuration changes are being applied.

**Note:** To avoid collision with the Self-Ordered Backup and Restore function, do not perform system data backup or restore if a Self-Ordered Backup and Restore process is running on any node of the system.

See [cudbSystemStatus](#) on page 82 for further information about the status of the Self-Ordered Backup and Restore process.

#### cudbSystemDataBackupAndRestore Requisites

`cudbOperator` users have no access to run this command.

#### cudbSystemDataBackupAndRestore Syntax

- Format (1): `cudbSystemDataBackupAndRestore -c | --create [-F | --fast-copy] [-f | --forced] [-o | --copy-over-oam-vip] [-b | --cudbdatabackup-option <option_string>]`
- Format (2): `cudbSystemDataBackupAndRestore -c | --create -k | --keep-local [ -f | --forced ] [ -b | --cudbdatabackup-option <option_string>]`
- Format (3): `cudbSystemDataBackupAndRestore --restore <backup_path> [-F | --fast-copy] [-s | --cudbdatarestore-option <option_string>] [--no-prompt]`



- Format (4): `cudbSystemDataBackupAndRestore --restore-date <backup_date> [ -F | --fast-copy ] [ -f | --forced ] [ -s | --cudbdatarestore-option <option_string> ] [--no-prompt]`
- Format (5): `cudbSystemDataBackupAndRestore -h | --help`

### cudbSystemDataBackupAndRestore Command Options

The following command options can be used:

- `-h | --help`: Displays command help message and exits.
- `-c | --create`: Creates a system data backup.
- `--restore <backup_path>` : Restores a system data backup from the specified directory path.
- `--restore-date <backup_date>` : Restores a system data backup created on the defined date. The format of `<backup_date>` is YYYY-MM-DD\_HH24-MI.

**Note:** The `--create`, `--restore` and `--restore-date` options are mutually exclusive.

- `--no-prompt`: This command can be executed by Ericsson personnel only.
- `-k | --keep-local`: Forces to keep the backup files on the CUDB nodes where they are generated. If this option is used, the created backups will not be collected to the CUDB node where the command was ordered.

**Note:** If `-k | --keep-local` is omitted, the generated backup files are collected to the CUDB node where the command was ordered, provided that there is enough storage space. The files are copied to the cluster area in the following directory:

`/home/cudb/automatedBackupStorage`

- `-F | --fast-copy`: Disables the default SCP speed limit when copying backup files from one node to another.
- Note:** Using this option yields a much higher network use.
- `-f | --forced`: Disables the checking of `/home/cudb/systemDataBackup` in case of backup creation or backup restore with the `-R` option. Furthermore, in case of backup creation, the checking of `/home/cudb/automatedBackupStorage` is disabled.
- `-o | --copy-over-oam-vip`: Transfer backup files over OAM\_VIP address.
- `-b | --cudbdatabackup-option <option_string>` : Option arguments to pass to the `cudbDataBackup` command. The default value is `-L`.



- `-s | --cudbdatarestore-option <option_string>` : By default, `-L` is included, but it can be discarded. Additional options to use when doing a system data restore (`cudbDataRestore` parameters):
  - `-T | --Timeout <seconds>` : Restricted for Ericsson personnel.
  - `-R | --Retries-number <number of retries>` : Defines the number of retry attempts to establish the connection to the CUDB nodes where the specific DS Units or the PLDB must be restored. The default value is 2.
  - `-S | --Script-path <script>` : Defines the path where the post-restore commands are located. Do not use this option in a production environment.
  - `-u | --user-facility <facility>` : Enables Syslog facility for logging. Do not use this option in a production environment.
  - `-L | --parallel-mode`: Orders a restore for each CUDB node element (that is for the PLDB and all DS Units like DS1, DS2, and so on) at the same time instead of one after another (that is, not when the previous one is finished). The option does not guarantee that the restore takes place on the PLDB first, then on the DS Units in parallel.
  - `-D | --Debug`: Restricted for Ericsson personnel.

## cudbSystemDataBackupAndRestore Output

The following output examples are provided below:

- The below example output indicates normal operation:

```
CUDB_44 SC_2_2# cudbSystemDataBackupAndRestore -c
Checking /home/cudb/automatedBackupStorage on local node
-----

Local node: ... EMPTY

Checking /home/cudb/systemDataBackup on nodes
-----

44: ... EMPTY
43: ... EMPTY

CREATE PART
-----

/home/cudb/systemDataBackup
Listening for current PLDB and DSGs status reports (may take upto 2 minutes)
Starting backup...
Before calling pg_notification
BackupStart [INFO] - PS Notification trying stop notification was successfully→
sent.
PS Notification successfully sent to : 172.31.233.139
PS Notification failed to : NONE
PS not notified : NONE

cudb_backup      ver(1.3.4)

BEGIN MANAGEMENT FOR LAST BACKUP
```



```
Backup 2014-05-14_20-11 finished successfully in :
PLDB in CUDB node 44
  NDB node PL0
  NDB node PL1
  NDB node PL2
  NDB node PL3
DSG#1 in CUDB node 43
  NDB node DS1_0
  NDB node DS1_1
DSG#255 in CUDB node 43
  NDB node DS2_0
  NDB node DS2_1
Attempting to de-block Provisioning Gateway. This may take up to a couple of minutes.
```

#### COLLECT PART

```
-----

Copying backup piece from node 44 for dsg 0 for ndb node id 3 ... OK
Copying backup piece from node 44 for dsg 0 for ndb node id 4 ... OK
Copying backup piece from node 44 for dsg 0 for ndb node id 5 ... OK
Copying backup piece from node 44 for dsg 0 for ndb node id 6 ... OK
Copying backup piece from node 43 for dsg 1 for ndb node id 3 ... OK
Copying backup piece from node 43 for dsg 1 for ndb node id 4 ... OK
Copying backup piece from node 43 for dsg 255 for ndb node id 3 ... OK
Copying backup piece from node 43 for dsg 255 for ndb node id 4 ... OK
```

Please copy /home/cudb/automatedBackupStorage to a safe location to be able to restore the backup pieces when needed!

#### ERROR SUMMARY

None.

- The below example output shows an error caused by executing `cudbSystemDataBackupAndRestore` with the `--restore` option, while configuration changes are being applied:

```
CUDB_210 SC_2_2# cudbSystemDataBackupAndRestore --restore /home/cudb/automatedBackupStorage
```

#### VALIDATE PART

```
-----

NODE: 210
  DSG: 0
    NDB: 3
    NDB: 4
    NDB: 5
    NDB: 6
  DSG: 1
    NDB: 3
    NDB: 4
  DSG: 255
    NDB: 3
    NDB: 4
```

#### DISTRIBUTE PART

Creating /home/cudb/systemDataBackup on node 210

```
Node 210 has DSG 0      -> copying ... id3-OK id4-OK id5-OK id6-OK
Node 210 has DSG 1      -> copying ... id3-OK id4-OK
Node 210 has DSG 255    -> copying ... id3-OK id4-OK
Creating /home/cudb/systemDataBackup on node 244
```

```
Node 244 has DSG 0      -> copying ... id3-OK id4-OK id5-OK id6-OK
```



```

Node 244 has DSG 1      -> copying ... id3-OK id4-OK
Node 244 has DSG 255   -> copying ... id3-OK id4-OK
EXTRACT PART
-----

Node 210 has DSG 0      -> extracting ... id3-OK id4-OK id5-OK id6-OK
Node 210 has DSG 1      -> extracting ... id3-OK id4-OK
Node 210 has DSG 255    -> extracting ... id3-OK id4-OK
Node 244 has DSG 0      -> extracting ... id3-OK id4-OK id5-OK id6-OK
Node 244 has DSG 1      -> extracting ... id3-OK id4-OK
Node 244 has DSG 255    -> extracting ... id3-OK id4-OK

RESTORE PART
-----

Starting System Data Backup with command cudbDataRestore -B 2015-07-27_12-40 ->
L 2>&1 ...
cudbDataRestore      ver(1.3.6)

Trying for restore the backup from [2015-07-27_12-40] data files.

Error executing the restore command in PL/DS node in CUDBNode#244
Error executing the restore command in PL/DS node in CUDBNode210. Failed due p>
arallel execution of cudbApplyConfig command.
Check logs and cudbApplyConfig.lock file in /home/cudb/oam/configMgmt/commands>
/config directory!
Restores process finished with error.

ERROR: unable to restore backup

ERROR SUMMARY
-----

ERROR: unable to restore backup
WARNING: System data restoration will execute for the complete CUDB system wit>
h the data from the system backup.
Are you sure you want to continue (y/n)? y

```

- The below example output shows another error in another erroneous scenario:

```

ERROR SUMMARY
-----

Unable to kickstart pl on node 24
Unable to kickstart dsg 1 on node 24
Unable to kickstart dsg 2 on node 24

```

### 2.2.28.1 cudbSystemDataBackupAndRestore Examples of Use

- Execute the command as follows to create a system data backup with the SCP limit disabled:

```
<CUDB_node_prompt> cudbSystemDataBackupAndRestore --create --fast-copy
```

- Execute the command as follows to create a system data backup with parallel mode enabled and PG notifications disabled:

```
<CUDB_node_prompt> cudbSystemDataBackupAndRestore -c -b "-q -L"
```



- Execute the command as follows to restore a system data backup from a specified location:

```
<CUDB_node_prompt> cudbSystemDataBackupAndRestore --restore /  
home/cudb/cudbSystemDataBackup
```

- Execute the command as follows to restore a system data backup created on the defined date:

```
<CUDB_node_prompt> cudbSystemDataBackupAndRestore --restore-date  
2014-01-01_08-30
```

## 2.2.29

### cudbSystemStatus

The `cudbSystemStatus` command automatically prints the general system status and the local node information. The output contains information on the following components:

- CUDB software version
- Blackboard Coordination (BC) server process status and BC cluster information.
- System Monitor (SM) status
- Master and Slave cluster status
- Network DataBase (NDB) status
- Replication status and active channel
- Alarms
- Database cluster server connection
- CUDB processes

**Note:** If the `cudbSystemStatus` command is executed on a system with different hardware types, the command output will also contain information on the hardware types used in the system. This hardware information is listed below the CUDB software version output.

### cudbSystemStatus Requisites

This command can be accessed by `cudbOperator` user using `sudo`.

### cudbSystemStatus Syntax

```
cudbSystemStatus [-a | --alarms] [-s | --sm-status] [-c | --  
cluster-status] [-C | --new-cluster-status] [-m | --check-mysql]
```



```
[ -p | --check-cudbprocess ] [ -b | --bc-status ] [ -B | --new-bc-status ] [ -r | --replication-status ] [ -R | --new-replication-status ] [ -t | --hardware-type ] [ -v | --version ] [ -h | --help ]
```

### cudbSystemStatus Command Options

The following command options can be used:

- `-a` | `--alarms`: Prints alarms.
- `-b` | `--bc-status`: Restricted for Ericsson personnel.
- `-B` | `--new-bc-status`: Prints the status of the BC process.
- `-s` | `--sm-status`: Prints the SM status.
- `-c` | `--cluster-status`: Restricted for Ericsson personnel.
- `-C` | `--new-cluster-status`: Prints cluster status.
- `-m` | `--check-mysql`: Checks database cluster server connections.
- `-p` | `--check-cudbprocess`: Checks CUDB processes relevant to the correct system behavior.
- `-r` | `--replication-status`: Restricted for Ericsson personnel.

---

### Attention!

This command option is deprecated. Use the `-R` option instead.

---

- `-R` | `--new-replication-status`: Prints replication status. The possible output of the option for the DSGs are as follows:
  - `M`: Master
  - `Sn`: Slave - replication channel `<#n>` is active.
  - `S?`: Slave - Replication status is unknown.
  - `[S]`: Slave - replication channels are down.
  - `Xm`: Wrong state - masterless.
  - `Xu`: Wrong state - unreachable.
- `-t` | `--hardware-type`: Prints the value of the `hwType` attribute of each CUDB node and whether each DSG in the CUDB system is running on hybrid hardware (that is, whether the DS Units of a certain DSG are hosted on nodes



with different hardware types). Refer to CUDB Node Configuration Data Model Description for more information about the hwType attribute.

- -v | --version: Prints the CUDB version.
- -h | --help: Displays command help message and exits.

**Note:** If no options are used when executing the command, cudbSystemStatus runs as if the -B, -v, -s, -C, -R, -a, -m, and -p options have been used. If at least one command option is executed, the command collects information related only to that command option.

Also, if no options are used when executing the command on a system with different hardware types, cudbSystemStatus runs the -t | --hardware-type option in addition to the options listed above.

**Note:** In hybrid systems, where some DSGs consist of different hardware platforms, memory usage values are recalculated for DSGs located on nodes with higher configured system memory for database cluster. Memory levels shown by the cudbSystemStatus command for those DSGs are different from those reported by database cluster client.

### cudbSystemStatus Output

The command shows information on the relevant system status. BC cluster information is showing status of the BC clusters in all the sites (unless the node is disabled, then the output is Disabled instead of showing the status of BC servers on that node), while SM information shows the status of the SM leader elected in every site.

If, for any reason, the BC cluster does not have the majority of BC servers, enough to work as a cluster, it is indicated in the output, and no SM leader appears on the affected site.

#### 2.2.29.1

### cudbSystemStatus Examples of Use

Perform the following steps to execute the command:

#### Steps

1. Connect to the CUDB node with the following command:

```
<CUDB_node_prompt> ssh <admin_user>@<CUDB_Node_IP_Address>
```

2. Connect to the SC with the following command:

```
<CUDB_node_prompt> ssh <admin_user>@<CUDB_Node_OAM_IP_Address>
```

3. Execute the command as follows:

```
<CUDB_node_prompt> cudbSystemStatus
```





## Example 5 cudbSystemStatus

- The expected output must be similar to the below example:

```
<CUDB_node_prompt> cudbSystemStatus

Execution date: Sat Aug 10 11:41:23 CEST 2013

CUDB Software Version:
!- CUDB DESIGN DISTRIBUTION: CUDB13B CXP9020214/6 R1B549

Checking BC clusters:

[Site 1]

    SM leader: Node 10 OAM1

    Node 10
        BC server in OAM1 ..... running
        BC server in OAM2 ..... running (Leader)
        BC server in PL2 ..... running

[Site 2]

    SM leader: Node 11 OAM1

    Node 11
        BC server in OAM1 ..... running
        BC server in OAM2 ..... running (Leader)
        BC server in PL2 ..... running

[Site 3]

    SM leader: Node 9 OAM2

    Node 9
        BC server in OAM1 ..... running
        BC server in OAM2 ..... running (Leader)
        BC server in PL2 ..... running

Checking System Monitor BC status in local node:

    SM-BC in OAM1 ..... running
    SM-BC in OAM2 ..... running

Checking Clusters status:
Node 9:
    PL Cluster (29%) .....OK
    DSG1 Cluster (23%) .....OK
    DSG255 Cluster (23%) .....OK
Node 10:
    PL Cluster (29%) .....OK
    DSG1 Cluster (23%) .....OK
    DSG255 Cluster (23%) .....OK
Node 11:
    PL Cluster (29%) .....OK
    DSG1 Cluster (22%) .....OK
    DSG255 Cluster (22%) .....OK

Checking NDB status:
    PL NDB's (2/2) .....OK
    DS1 NDB's (2/2) .....OK
    DS2 NDB's (2/2) .....OK

Checking Replication Channels in the System:
Node   | 9 | 10 | 11 |
=====|==|===|====|
PLDB   |___|___|___|___|
DSG 1  |___|___|___|___|
DSG 255|___|___|___|___|

Printing Detailed Replication Status for the Slave Replicas:
Node 9:
    Replication in DSG0(Chan=1) .... Up -- Delay = 0.0 seconds, no. of pe→
nding changes = 0
    Replication in DSG1(Chan=1) .... Up -- Delay = 0.0 seconds, no. of pe→
nding changes = 0
```



```

Replication in DSG255(Chan=1) .... Up -- Delay = 0.0 seconds, no. of pe→
nding changes = 0
Node 10:
Replication in DSG0(Chan=1) .... Up -- Delay = 0.0 seconds, no. of pe→
nding changes = 0
Replication in DSG1(Chan=1) .... Up -- Delay = 0.0 seconds, no. of pe→
nding changes = 0
Replication in DSG255(Chan=1) .... Up -- Delay = 0.0 seconds, no. of pe→
nding changes = 0
Node 11:
There are no Slave clusters

Printing Alarms...
[Aug 09 16:12:17]( Root Login Failed @172.80.122.10 )
[Aug 09 16:16:00]( Root Login Failed @10.22.10.199 )

Checking MySQL server connection:
MySQL Master Servers connection .....OK
MySQL Slave Servers connection .....OK
MySQL Access Servers connection .....OK

Checking Process:
OAMs.....
Cluster Supervisor.....Running
System Monitor BC.....Running
Reconciliation process.....Running in: OAM1
Management Server Process (ndb_mgmd).....Running
KeepAlive process.....Running
Alarm Correlator.....Running
ESA.....Running
LDAP counter.....Running in: OAM1
Log Handler process.....Running
KpiCentral process.....Running in: OAM1
Messaging Service servers.....Running
PLs.....
Storage Engine process (ndbd).....Running
LDAP FE.....Running
KeepAlive process.....Running
MySQL server process (Master).....Running
MySQL server process (Slave).....Running
MySQL server process (Access).....Running
CudbNotifications process.....Running
LDAP FE Monitor process.....Running
DSs.....
Storage Engine process (ndbd).....Running
LDAP FE.....Running
KeepAlive process.....Running
MySQL server process (Master).....Running
MySQL server process (Slave).....Running
MySQL server process (Access).....Running
LDAP FE Monitor process.....Running

```

- In case of systems with different hardware types, the beginning of the expected output must be similar to the below example:

```

<CUDB_node_prompt> cudbSystemStatus
Execution date: Sat Mar 12 14:27:48 CET 2016

CUDB Software Version:
!- CUDB DESIGN DISTRIBUTION: CUDB16A CXP9020214/9 R2A99

Checking Hardware Type:
This system is working on following hardware types: EBS_GEP3, EBS_GEP5.
...

```

- In case the `-t | --hardware-type` option is used on hybrid systems, the expected output must be similar to the below example:

```

<CUDB_node_prompt> cudbSystemStatus -t
Execution date: Sat Mar 12 14:43:28 CET 2016

NodeId hwType
49 EBS_GEP3
82 EBS_GEP3
154 EBS_GEP5
157 EBS_GEP5

```



```
DsgId isHybrid
0 true
11 false
12 false
13 false
14 false
15 false
16 false
21 false
22 false
23 false
24 false
25 false
26 false
27 false
28 false
```

- In case replication is stopped in one or more DSGs and Self-Ordered Backup and Restore is enabled, the expected output is similar to the below example:

Checking Replication Channels in the System:

```
Node      | 9 | 10 | 11
=====
PLDB      |   |   |   |
DSG 1     | S1 | S1 | M
DSG 255   | Xu | S1 | M
```

Printing Detailed Replication Status for the Slave Replicas:

```
Node 9:
  Replication in DSG0(Chan=1) .... Up -- Delay = 0.0 seconds, no. of pe→
ding changes = 0
  Replication in DSG1(Chan=1) .... Up -- Delay = 0.0 seconds, no. of pe→
ding changes = 0
  Replication in DSG255(Chan=1) .... Dsg is Unreachable
Self-Ordered Backup and Restore is run→
ning: restoring backup
Node 10:
  Replication in DSG0(Chan=1) .... Up -- Delay = 0.0 seconds, no. of pe→
ding changes = 0
  Replication in DSG1(Chan=1) .... Stopped - From Thu Apr 14 08:54:02 CES→
T 2016
Self-Ordered Backup and Restore is run→
ning: transferring backup
  Replication in DSG255(Chan=1) .... Up -- Delay = 0.0 seconds, no. of pe→
ding changes = 0
Node 11:
  There are no Slave clusters
```

- In case there are disabled node(s) and/or disabled DS(s) in the system, the expected output is similar to the below examples:

#### A) One Unit Disabled (DSG3 on Node 41):

Checking Replication Channels in the System:

```
Node      | 40 | 41
=====
PLDB      |   |   |
DSG 1     | S1 | S1 | M
DSG 3     | M  | Xu |
```

Printing Detailed Replication Status for the Slave Replicas:

```
Node 40:
  Replication in DSG0(Chan=1) .... Up -- Delay = 0.0 seconds, no. of pen→
ding changes = 0
  Replication in DSG1(Chan=1) .... Up -- Delay = 0.0 seconds, no. of pen→
ding changes = 0
Node 41:
  Replication in DSG3 .... Stopped - Data Store is disabled
```

#### B) One Node Disabled (Node 40):

Checking Replication Channels in the System:

```
Node      | 40 | 41 | 130 | 131
```



```
=====
PLDB  |  | Xu | S1 | M | S1 |
DSG 1 |  | Xu |  | M | S1 |
DSG 3 |  | Xu | M |  |  |
DSG 255 |  | S1 | M | S1 |
=====
```

Printing Detailed Replication Status for the Slave Replicas:

```
Node 40: Disabled
  Replication in DSG0      .... Stopped - Data Store is disabled
  Replication in DSG1      .... Stopped - Data Store is disabled
  Replication in DSG3      .... Stopped - Data Store is disabled
Node 41:
  Replication in DSG0(Chan=1) .... Up -- Delay = 0.0 seconds, no. of pending →
  changes = 0
  Replication in DSG255(Chan=1) .... Up -- Delay = 0.0 seconds, no. of pending →
  changes = 0
Node 130:
  There are no Slave clusters
Node 131:
  Replication in DSG0(Chan=1) .... Up -- Delay = 0.0 seconds, no. of pen →
  ding changes = 0
  Replication in DSG1(Chan=1) .... Up -- Delay = 0.0 seconds, no. of pen →
  ding changes = 0
  Replication in DSG255(Chan=1) .... Up -- Delay = 0.0 seconds, no. of pen →
  ding changes = 0
```

## 2.2.30

### cudbTakeAllMasters

The `cudbTakeAllMasters` command is used to change mastership manually to a working partition in case of emergency (such as a CUDB system split situation), when a surviving partition is available in the CUDB system with reduced service. `cudbTakeAllMasters` maximizes the service in the surviving partition, allowing normal operation with the resources available in this partition.

Refer to [CUDB High Availability](#) for further details about the use of this command.

---

### Attention!

Execute this command only as a manual intervention in the special case that some sites are failing, and the system is in a split situation with reduced service.

---

### cudbTakeAllMasters Requisites

The requisites of using this command are as follows:

- The partition must be in a system split situation of minority or symmetrical split to accept this command.
- The command requires execution confirmation.

### cudbTakeAllMasters Syntax

**cudbTakeAllMasters** [-h|--help]



### cudbTakeAllMasters Command Options

The following command option can be used:

- `-h` | `--help`: Displays command help message and exits.

### cudbTakeAllMasters Output

The command can return the following output messages:

- When executed with the `-h` or `--help` option, the command returns the following output:

```
cudbTakeAllMasters -h
=====
      USE of cudbTakeAllmasters
=====
Format : cudbTakeAllmasters [-h | --help]
```

Where:

`-h` | `--help`                      Displays command help message and exits

- In case of a majority system split situation, the command may return the following failure message:

```
cudbTakeAllMasters
WARNING: This command will take all masters in this partition! →
\
      Are you sure you want to continue (y/n)? y
TAM can not be set in a majority (majority[1, 2] AR[] NS[])
Command failed
```

**Note:** If the command fails for other reasons than the ones above, contact the next level of maintenance support.

#### 2.2.30.1 cudbTakeAllMasters Examples of Use

```
<CUDB_node_prompt> cudbTakeAllMasters
```

#### 2.2.31 cudbUnitDataBackupAndRestore

The `cudbUnitDataBackupAndRestore` command resolves data inconsistencies between replicas by creating a data backup on the master replica and restoring this data backup on the inconsistent slave replica. Refer to [CUDB Backup and Restore Procedures](#) for further details about when to use it

#### cudbUnitDataBackupAndRestore Requisites

The requisites of using the command are as follows:



- The command location must be included in the PATH system environment variable.
- `cudbOperator` users have no access to run this command.
- The command requires execution confirmation for some options.

### **cudbUnitDataBackupAndRestore Syntax**

- Format (1): `cudbUnitDataBackupAndRestore (-p | --pl) (-n | --node) <nodeId> [(-f | --fast-copy)] [-q | --queued-copy] [-o | --copy-over-oam-vip] [--no-prompt]`
- Format (2): `cudbUnitDataBackupAndRestore (-d | --dsg) <dsgId> (-n | --node) <nodeId> [(-f | --fast-copy)] [-q | --queued-copy] [-o | --copy-over-oam-vip] [--no-prompt]`
- Format (3): `cudbUnitDataBackupAndRestore -h | --help`

### **cudbUnitDataBackupAndRestore Command Options**

The following command options can be used:

- `-p | --pl`: Sets the PLDB cluster as the cluster to operate on. The command does not work, just exits with an error message, if any of the DS clusters in the node is the master replica of its data partition.
- `-d | --dsg <dsgId>` : Sets the DSG cluster defined with `<dsgId>` as the DSG cluster to operate on.
- `-n | --node <nodeId>` : Sets the slave node defined with the `<nodeId>` as the slave node on which the backup will be restored,
- `-h | --help`: Displays command help message and exits.
- `-f | --fast-copy`: Disables the default SCP speed limit when copying backup files from one node to another. Consider that this yields a much higher network use.
- `-q | --queued-copy`: Restricted for Ericsson personnel.
- `-o | --copy-over-oam-vip`: Transfer backup files over OAM\_VIP address.
- `--no-prompt`: Restricted for Ericsson personnel.

### **cudbUnitDataBackupAndRestore Output**

An example output of the command is provided below:



## Example 6

```
CUDB_41 SC_2_1# cudbUnitDataBackupAndRestore -d 1 -n 42
WARNING: Restoring the DS cluster will replace previous contents. Are you sure you want to continue →
(y/n)? y
Action will be executed

CREATE PART
-----

creating backup on node 41

cudbManageStore stores to process:  ds1 (in dsgroup1).

Starting Backup ...
Launching order Backup for ds1 in dsgroup 1.
Obtaining Mgm Information.
Trying backup on mgmt access 1, wait a moment ...
ndb_mgm 10.22.41.1 2372 -e "START BACKUP 999 WAIT COMPLETED"
..ok
BACKUP-999 renamed in PL_2_7 to /local/cudb/mysql/ndbd/backup/BACKUP/BACKUP-2016-04-11_15-46
BACKUP-999 renamed in PL_2_8 to /local/cudb/mysql/ndbd/backup/BACKUP/BACKUP-2016-04-11_15-46

Backup finished successfully for store ds1.
Stores where order backup was successfully completed:  ds1.
cudbManageStore command successful.

DIR CREATE PART
-----

creating backup directory /local/cudb/mysql/ndbd/backup/BACKUP/BACKUP-2016-04-11_15-46 on node 42 on →
blade 10.22.42.7
creating backup directory /local/cudb/mysql/ndbd/backup/BACKUP/BACKUP-2016-04-11_15-46 on node 42 on →
blade 10.22.42.8

COPY PART
-----

copying backup files from node 41 blade 10.22.41.7:/local/cudb/mysql/ndbd/backup/BACKUP/BACKUP-2016- →
04-11_15-46 ⇒
to node 42 blade 10.22.42.7:/local/cudb/mysql/ndbd/backup/BACKUP/BACKUP-2016-04-11_15-46 ⇒
copying backup files from node 41 blade 10.22.41.8:/local/cudb/mysql/ndbd/backup/BACKUP/BACKUP-2016- →
04-11_15-46 ⇒
to node 42 blade 10.22.42.8:/local/cudb/mysql/ndbd/backup/BACKUP/BACKUP-2016-04-11_15-46

RESTORE PART
-----

Restoring backup on node 42

cudbManageStore stores to process:  ds1 (in dsgroup1).

Launching restore order in CUDB Node 42 to store ds1 in dsgroup 1.
Starting restore in CUDB Node 42 for store ds1, ⇒
backup path /local/cudb/mysql/ndbd/backup/BACKUP/BACKUP-2016-04-11_15-46,⇒
sql scripts path /home/cudb/storageEngine/config/schema/ds/internal/restoreTempSql.
Waiting for restore order(s) to be completed in CUDB Node 42 for stores :  ds1.
restore order finished successfully in CUDB Node 42 for store ds1.
restore order(s) completed in CUDB Node 42 for stores :  ds1.
Stores where order restore was successfully completed:  ds1.
Closing connections for all blades of DSUnitGroup 1.
No stored procedures were found to restore.
cudbManageStore command successful.

Performing cleanup
delete backup directory /local/cudb/mysql/ndbd/backup/BACKUP/BACKUP-2016-04-11_15-46 on node 42 on b →
lade 10.22.42.7
delete backup directory /local/cudb/mysql/ndbd/backup/BACKUP/BACKUP-2016-04-11_15-46 on node 42 on b →
lade 10.22.42.8
cudbUnitDataBackupAndRestore successfully ended
```



### 2.2.31.1 cudbUnitDataBackupAndRestore Examples of Use

- Execute the command as follows to create a backup of DSG2 on the master node, and restore it on node 39:

```
<CUDB_node_prompt> cudbUnitDataBackupAndRestore -dsg 2 --node 39
```

- Execute the command as follows to create a backup of the PLDB on the master node, and restore it on node 40:

```
<CUDB_node_prompt> cudbUnitDataBackupAndRestore -p -n 40
```

### 2.2.32 cudbUpdateUserInfo

---

---

#### Attention!

This command is deprecated. To update the local node configuration with the latest changes of LDAP users, use the administrative operation `updateUserInfo`. Refer to [CUDB Node Configuration Data Model Description](#) for more information on `updateUserInfo`.

---

---

## 2.3 Internal Commands

This section lists the internal commands of the CUDB system, available for Ericsson personnel only.

### 2.3.1 cudbAppCheckerManager

This command can be executed by Ericsson personnel only.

### 2.3.2 cudbBCServersRestart

This command can be executed by Ericsson personnel only.

### 2.3.3 cudbCopyAclsConfFile

This command can be executed by Ericsson personnel only.

### 2.3.4 cudbEvipConfigExtension

This command can be executed by Ericsson personnel only.





### 2.3.5 `cudbEvipEncapsulator`

This command can be executed by Ericsson personnel only.

### 2.3.6 `cudbExecuteAllBlades`

This command can be executed by Ericsson personnel only.

### 2.3.7 `cudbFollowLdapfeLogs`

This command can be executed by Ericsson personnel only.

### 2.3.8 `cudbManageDsGroup`

This command can be executed by Ericsson personnel only.

### 2.3.9 `cudbManageNode`

This command can be executed by Ericsson personnel only.

### 2.3.10 `cudbManageSite`

This command can be executed by Ericsson personnel only.

### 2.3.11 `cudbMpstat`

This command can be executed by Ericsson personnel only.

### 2.3.12 `cudbOomConfigurator`

The `cudbOomConfigurator` command is used only by a restricted process.

### 2.3.13 `cudbParallelCommandRun`

This command can be executed by Ericsson personnel only.

### 2.3.14 `cudbRemoveNode`

This command can be executed by Ericsson personnel only.



### 2.3.15 **cudbSdpInfo**

This command can be executed by Ericsson personnel only.

### 2.3.16 **cudbSetDsgMaster**

This command can be executed by Ericsson personnel only.

### 2.3.17 **cudbSetPartitionStatus**

This command can be executed by Ericsson personnel only, although option `-ps` is available for all system administrators and operators.

#### **cudbSetPartitionStatus Command Options**

The following command options can be used:

— `-ps` | `--printPartitionStatus`: Prints the current partition status.

### 2.3.18 **cudbTpsStat**

This command can be executed by Ericsson personnel only.



## Glossary

For the terms, definitions, acronyms and abbreviations used in this document, refer to CUDB Glossary of Terms and Acronyms



# Reference List

## CUDB Documents

1. CUDB Node Configuration Data Model Description
2. CUDB Node Logging Events
3. CUDB Security and Privacy Management
4. CUDB Users and Passwords 3/00651-HDA 104 03/10
5. CUDB System Administrator Guide
6. CUDB Logchecker
7. CUDB Application Counters
8. Storage Engine, Potential Data Inconsistency between Replicas Found in DS
9. Storage Engine, Potential Data Inconsistency between Replicas Found in PLDB
10. Storage Engine, Replication Stopped Working in DS
11. Storage Engine, Replication Stopped Working in PLDB
12. CUDB Consistency Check
13. CUDB Backup and Restore Procedures
14. CUDB Application Schema Update
15. CUDB High Availability
16. Storage Engine, Unable to Synchronize Cluster in DS, Major
17. Storage Engine, Unable to Synchronize Cluster in PLDB, Major
18. CUDB Subscription Reallocation
19. CUDB Data Storage Handling
20. Storage Engine, Backup Notification Failure To Provisioning Gateway
21. CUDB Glossary of Terms And Acronyms



## Other Ericsson Documents

1. LDE Management Guide