Compellent Storage Center

DB2 UDB with Consistency Group

Compellent Corporate Office
Compellent Technologies
7625 Smetana Lane
Eden Prairie, Minnesota 55344

www.compellent.com
Contents

Preface ................................................................................................................................ 3
Scope .................................................................................................................................. 3
Audience ............................................................................................................................ 3
Customer Support ............................................................................................................ 3
Disclaimers ......................................................................................................................... 3
General Syntax .................................................................................................................. 4
Document Revision ........................................................................................................... 4
DB2 UDB Overview ............................................................................................................. 5
DB2 Instances .................................................................................................................... 5
Databases ............................................................................................................................ 6
Table Spaces ....................................................................................................................... 6
Containers ............................................................................................................................ 8
Compellent Overview ......................................................................................................... 9
Compellent Consistency Group ......................................................................................... 9
Configurations .................................................................................................................... 10
Creating Volumes .............................................................................................................. 10
Creating Consistency Group Replay Profile ................................................................. 11
Usage ................................................................................................................................ 15
Creating Replays ............................................................................................................... 15
Creating Replays via the GUI .......................................................................................... 15
Creating Replays via the CompCU .................................................................................... 16
Conclusion .......................................................................................................................... 18
Scope

This document describes the best practices on how to use Compellent Storage Center Consistency Group feature with DB2 UDB databases. It shows the step by step on how to create the profile and how to use it with the GUI and CompCU.

Audience

The primary target audience for this best practice is database administrators, system administrators, storage administrators, and architects who analyze, design, and maintain a robust database and storage system. Readers should be familiar with DB2 UDB table space containers and how they can affect the performance of a DB2 UDB database.

Customer Support

Compellent provides live support 1-866-EZSTORE (866.397.8673), 24 hours a day, 7 days a week, 365 days a year. For additional support, email Compellent at support@compellent.com. Compellent responds to emails during normal business hours.

Disclaimers

Information in this document is subject to change without notice.

© 2009 Compellent Technologies. All rights reserved. Reproduction in any manner without the express written permission of Compellent Technologies is strictly prohibited.

Trademarks used in this text are property of Compellent Technologies, or their respective owners.
## General Syntax

<table>
<thead>
<tr>
<th>Item</th>
<th>Convention</th>
</tr>
</thead>
<tbody>
<tr>
<td>Menu items, dialog box titles, field names, keys</td>
<td><strong>Bold</strong></td>
</tr>
<tr>
<td>Mouse click required</td>
<td>Click:</td>
</tr>
<tr>
<td>User Input</td>
<td>Monospaced Font</td>
</tr>
<tr>
<td>User typing required</td>
<td>Type:</td>
</tr>
<tr>
<td>Website addresses</td>
<td><a href="http://www.compellent.com">http://www.compellent.com</a></td>
</tr>
<tr>
<td>Email addresses</td>
<td><a href="mailto:info@compellent.com">info@compellent.com</a></td>
</tr>
</tbody>
</table>

## Document Revision

<table>
<thead>
<tr>
<th>Date</th>
<th>Revision</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>4/23/2010</td>
<td>A</td>
<td>Preliminary</td>
</tr>
</tbody>
</table>
**DB2 Instances**

An instance in DB2 for LUW is like a copy of RDBMS including all the processes that run DB2 and memory associated with that instance and some configuration parameters to control that instance. Anytime a new instance is created, that instance references the DB2 Database Manager program files that were stored on that server during the installation process; thus, each instance behaves like a separate installation of DB2 UDB.

Every instance controls access to one or more databases. Every database within an instance is assigned a unique name, has its own set of system catalog tables, and has its own configuration file. Below is an example of an instance named db2inst1.
Databases

A DB2 UDB database is a set of related objects. When you create a DB2 UDB database, you are establishing an administrative relational database entity that provides an underlying structure for an eventual collection of related objects such as tables, views, indexes, table spaces, etc.

Below is an example of a database named TESTDB.

Table Spaces

All data for a database is stored in a number of table spaces. You can think of a table space as being a child and a database as its parent, where the table space (child) cannot have more than one database (parent). Because there are different uses for table spaces, they are classified according to their usage and how they will be managed. There are five different table spaces by usage:

**Catalog table space**

There is only one catalog table space per database, and it is created when the CREATE DATABASE command is issued. Named SYSCATSPACE by DB2, the
catalog table space holds the system catalog tables. This table space is always created when the database is created.

**Regular table spaces**

Regular table spaces hold table data and indexes. It can also hold long data such as Large Objects (LOBs) unless they are explicitly stored in long table spaces. A table and its indexes can be segregated into separate regular table spaces, if the table spaces are database managed space (DMS). We will define the differences between DMS and system managed space (SMS) later in this white paper. At least one regular table space must exist for each database. The default is named USERSPACE1 when the database is created.

**Long table spaces**

Long table spaces are used to store long or LOB table columns and must reside in DMS table spaces. They can also store structured type columns or index data. If no long table space is defined, then LOBs will be stored in regular table spaces. Long table spaces are optional and none will be created by default.

**System temporary table spaces**

System temporary table spaces are used to store internal temporary data required during SQL operations such as sorting, reorganizing tables, creating indexes, and joining tables. At least one must exist per database. The default created with the database is named TEMPSPACE1.

**User temporary table spaces**

User temporary table spaces store declared global temporary tables. No user temporary table spaces exist when a database is created. At least one user temporary table space should be created to allow definition of declared temporary tables. User temporary table spaces are optional and none will be created by default.

**Table Space Management**

**System Managed Space (SMS)**

SMS table spaces are managed by the operating system. Containers are defined as regular operating system files and they are accessed via operating system calls. This means that all the regular operating system functions will handle the following: I/O will be buffered by the operating system, space will be allocated according to the operating system conventions, and the table space is automatically extended when it is necessary. However, containers cannot be dropped from SMS table spaces, and adding new ones is restricted to partitioned databases. The three default table spaces explained in the previous section are SMS.

**Database Managed Space (DMS)**

DMS table spaces are managed by DB2. Containers can be defined either as files (which will be fully allocated with the size given when the table space is created) or devices. DB2 will manage as much of the I/O as the allocation method and the operating system will
allow. Extending the containers is possible by using the \texttt{ALTER TABLESPACE} command. Unused portions of DMS containers can be also released (starting with version 8).

Below is an example of the table spaces.

Containers

Every table space has one or more containers. Again, you might think of a container as being a child and a table space as its parent. Each container can only belong to a single table space but a table space can have many containers. Containers can be added to or dropped from a DMS table space, and their sizes can be modified. Containers can only be added to SMS table spaces on partitioned databases in a partition, which does not yet have a container allocated for the table space. When new containers are added, an automatic rebalancing will start to distribute the data across all containers. Rebalancing will not prevent concurrent access to the database.

Below is an example of Containers.
Compellent Consistency Group

Compellent Consistency Group feature allows storage administrators to take a snapshot of a DB2 UDB database atomically. When creating a snapshot of a running DB2 database using storage functionality, you must ensure that all storage volumes (LUNs) that make up your database be atomically snapped because of multiple containers and transaction log files. Remember that DB2 writes to multiple container files, so without consistency group you cannot create a usable snapshot of a running database that spans multiple containers.

Without consistency group, in order to create a usable snapshot of an online database, the database table space must be configured with all container files in one volume or with just one container. Consistency Group feature gives you the ability to create a usable snapshot of an online database with multiple container files and transaction log files spread across volumes for performance.

Also, you can create a re-startable copy of a DB2 database with Consistency Group without having to suspend IO to database. This scenario is similar to having a power outage on the database server. At restart, DB2 performs crash recovery, rolling forward any changes that did not make it to the data files and rolling back changes that had not committed. However, roll-forward recovery using archive logs to a point-in-time after the re-startable copy is created is NOT supported.
Creating Volumes

Creating volumes for your DB2 database is simple. Once the volumes are created, you map the volumes to your server and configure the volumes at the operating system level.

Example: I have created four volumes for my database.
Creating Consistency Group Replay Profile

After creating the volumes for your database, you can create a Consistency Group replay profile and apply the profile to the volumes. The below steps show how to create this profile and apply it to the volumes:

1. Right click on Replay Profiles and select Create Consistent Replay Profile. Click “Continue” button.

2. Click the “Advanced” button.
3. Select both check boxes and enter 30 seconds and click “Continue” button.

4. Enter a name for this replay profile. The name of this replay profile will be used when creating replays of the database volumes. Click “Create Now” button.
5. DB2 replay profile has been created as shown. Right click on the DB2 replay profile and select “Apply to Volume(s)”. 

6. Expand the volume folder and select all volumes that make up the database and click “Continue”.
7. A confirmation window appears to allow you to verify the information. Click on "Apply Now" button to finish the process.
Creating Replays

You can create replays of your DB2 UDB database online via the GUI or CompCU (Compellent Command Line Utility). Before taking a replay of a DB2 database with either method, you need to suspend IO to the database first then take the replay and then resume IO to the database. Please refer to the sections below for both methods. If you want to create a re-startable copy of a database, you don’t need to suspend IO to the database before taking replays.

Creating Replays via the GUI

If you want to take a replay of your DB2 database online via the GUI, you need to perform the following tasks:

1. Log into your database as the instance user and issue the following commands:

   ```bash
   [db2inst1@lynx ~]$ db2 attach to db2inst1 user db2inst1 using db2inst1
   Instance Attachment Information
   Instance server = DB2/LINUXX8664 9.5.0
   Authorization ID = DB2INST1
   Local instance alias = DB2INST1
   
   [db2inst1@lynx ~]$ db2 connect to TESTDB
   Database Connection Information
   Database server = DB2/LINUXX8664 9.5.0
   SQL authorization ID = DB2INST1
   Local database alias = TESTDB
   
   [db2inst1@lynx ~]$ db2 set write suspend for database
   DB20000I  The SET WRITE command completed successfully.
   
   2. In the GUI, right click on the DB2 replay profile and select “Create Replay for Volumes”.
   ```
3. When the replay has been taken, issue the following commands to resume IO to the database.

```
[db2inst1@lynx ~]$ db2 set write resume for database
DB20000I The SET WRITE command completed successfully.
```

Creating Replays via the CompCU

If you want to take a replay of your DB2 database online via the CompCU you need to perform the following tasks:

**Prerequisite:** You need to install JRE 1.6 or higher on the database server and download CompCU.jar to the database server.

1. Log into your database server as the instance user and issue the following commands:

```
[db2inst1@lynx ~]$ db2 attach to db2inst1 user db2inst1 using db2inst1
Instance Attachment Information
    Instance server         = DB2/LINUXX8664 9.5.0
    Authorization ID        = DB2INST1
    Local instance alias    = DB2INST1

[db2inst1@lynx ~]$ db2 connect to TESTDB
Database Connection Information
    Database server         = DB2/LINUXX8664 9.5.0
    SQL authorization ID    = DB2INST1
    Local database alias    = TESTDB

[db2inst1@lynx ~]$ db2 set write suspend for database
DB20000I The SET WRITE command completed successfully.
```

2. Open another window to your database server and at the command prompt issue the CompCU command to take a replay. Please refer to the CompCU manual for the parameters.

```
[root@falcon]# java –jar CompCU.jar –host x.x.x.x –user Admin –password password –c “replayprofile createreplay –name DB2 –replayname testreplay –expire 60”
```
3. When the replay has been taken, issue the following commands to resume IO to the database.

```
[db2inst1@lynx ~]$ db2 set write resume for database
DB20000I The SET WRITE command completed successfully.
```

You can automate this process by using a shell script on UNIX or batch script on Windows.
With Compellent Consistency Group feature, database administrators can rest assure that they can take a snapshot of their DB2 databases at any time during working or non-working hours. This flexibility not only allows convenience to the DBAs but also save time for database refreshes.