Vertica Knowledge Base Article

Copying Data Between Similar Vertica Clusters

Document Release Date: 10/25/2018

Legal Notices

Warranty

The only warranties for Micro Focus International plc products and services are set forth in the express warranty statements accompanying such products and services. Nothing herein should be construed as constituting an additional warranty. Micro Focus shall not be liable for technical or editorial errors or omissions contained herein.

The information contained herein is subject to change without notice.

Restricted Rights Legend

Confidential computer software. Valid license from Micro Focus required for possession, use or copying. Consistent with FAR 12.211 and 12.212, Commercial Computer Software, Computer Software Documentation, and Technical Data for Commercial Items are licensed to the U.S. Government under vendor's standard commercial license.

Copyright Notice

Trademark Notices

Adobe™ is a trademark of Adobe Systems Incorporated.

Microsoft® and Windows® are U.S. registered trademarks of Microsoft Corporation.

UNIX® is a registered trademark of The Open Group.

This product includes an interface of the 'zlib' general purpose compression library, which is Copyright © 1995-2002 Jean-loup Gailly and Mark Adler.

Contents

Copying Data Between Similar Vertica Clusters

This is the second document in a three-part series on backup and restore, see parts 1 and 3 below:

Part 1: Copy and Restore Data from a Vertica Cluster to a Backup

Part 3: Copying Data Between Two Dissimilar Vertica Clusters

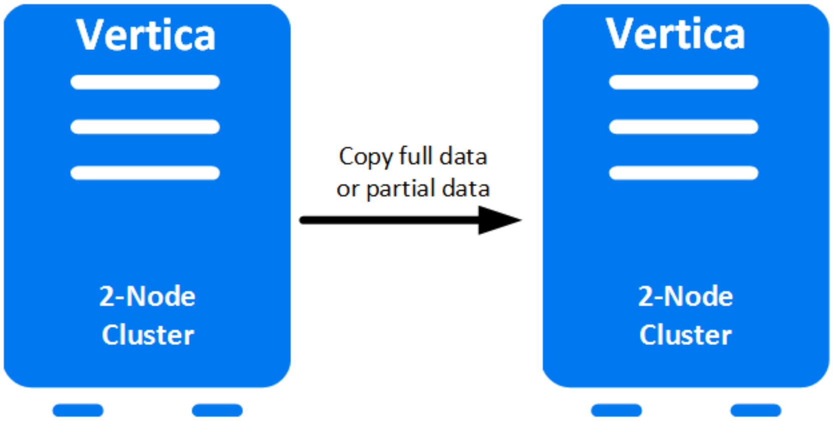
Copying Data Between Similar Vertica Clusters: PDF Format

Part 2: Copying Data Between Similar Vertica Clusters

You can replicate your Vertica database on another cluster, with these options:

- Replicate your entire Vertica database—Copy the catalog, schema, projections, and partitions.
- Replicate part of your Vertica database—Copy only specific tables.

For the purpose of demonstration and conciseness, the example in this document uses a 2-node cluster, but it's best to use a 3-node cluster with a K-safety level of 1.



When performing replication, you must make sure that your source and target clusters are similar. The two clusters must have the *same* number of nodes, node names, dbadmin user, and Vertica database version.

Other prerequisites vary, depending on whether you are performing a full or partial replication. See Copy a Full Cluster from Source to Target Cluster (Copycluster) or Copy Specific Tables Between Similar Vertica Clusters for information specific to the type of replication you want to perform.

Source and Target Clusters

For the purpose of this document:

- The primary cluster is referred to as the *source*.
- The secondary cluster is referred to as the *target*.
- Backup always occurs on the source, and restore always occurs on the target.
- The IP addresses for the source cluster nodes are 10.100.0.66 and 10.100.0.77.
- The IP addresses for the target cluster nodes are 10.100.0.88 and 10.100.0.99.

Copy a Full Cluster from a Source to a Target Cluster (Copycluster)

To copy the entire database from one Vertica cluster to another Vertica cluster, you can use the `vbr` option, `--copycluster`. Vertica backs up the data from the source cluster and restores the data to the target cluster in a single operation.

> Caution When copying data, the `vbr` script with the `--copycluster` option overwrites all existing data in the target cluster.

Prerequisites:

- The target cluster must have a different IP address than the source cluster.
- The source and the target cluster must have the same:
  - Number of nodes
  - Node names
- The databases on the source and target clusters must share the same:
  - dbadmin user
  - Database name
  - Vertica version
- You must set up passwordless SSH between the source and the target clusters.

How to Copy a Full Cluster Between Similar Vertica Clusters

To copy your entire database from one cluster to another similar cluster, follow these steps:

1. On the target cluster, install Vertica with the same number of nodes and same Vertica version as your source cluster.

```
$ sudo /opt/vertica/sbin/install_vertica --accept-eula --license
'/opt/vertica/config/licensing/vertica_community_edition.license.key' --
point-to-point --dba-user-password-disabled --ssh-identity
```

2. On the target cluster, create a database with the same database name as the source database, `mydatabase`.

```
$ /opt/vertica/bin/admintools -t create_db -s 10.100.0.88,10.100.0.99 -d
```

```
mydatabase
Info: no password specified, using none
Database with 1 or 2 nodes cannot be k-safe and it may lose data if it
crashes
Database mydatabase created successfully.
```

After you create the database, it is up and running. Before you can copy the database, you must first stop it.

3. On the target cluster, stop the `mydatabase` database.

```
$ /opt/vertica/bin/admintools -t stop_db -d mydatabase
Info: no password specified, using none
Connecting to database
Issuing shutdown command to database
Database mydatabase stopped successfully
```

4. On the source cluster, create a configuration file called `copyclusterconfig.ini`. Run the `vbr` script with the `--setupconfig` option and answer the prompts.

```
$ /opt/vertica/bin/vbr --setupconfig
Number of restore points (1): 3
Object restore mode (coexist, createOrReplace or create)
(createOrReplace): coexist
Vertica user name (dbadmin):
Node v_verticadb1_node0001
Backup host name (no default): 10.100.0.88
Backup directory (no default):
Node v_verticadb1_node0002
Backup host name (no default): 10.100.0.99
Backup directory (no default):
Change advanced settings? (n) [y/n]: n
Config file name (backup_snapshot.ini):copyclusterconfig.ini
Saved vbr config to copyclusterconfig.ini
```

5. On the source cluster, verify that your source database is up and running. Copy your entire database to the target cluster by running the `vbr` script with the `--copycluster` option.

> Caution When copying data, the `vbr` script with the `--copycluster` option overwrites all existing data in the target cluster.

```
$ /opt/vertica/bin/vbr --config-file copyclusterconfig.ini --task
copycluster
```

```
Starting copy of database mydatabase.
Participating nodes: v_mydatabase_node0001, v_mydatabase_node0002.
Enter vertica password:
Snapshotting database.
Snapshot complete.
Syncing data to destination cluster.
[==================================================] 100%
Reinitializing destination catalog.
Copycluster complete!
```

6. Start the target database, and verify that Vertica copied the data accurately.

```
$ admintools -t start_db -d mydatabase
Info: no password specified, using none
Starting nodes:
v_mydatabase_node0001 (10.100.0.88)
v_mydatabase_node0002 (10.100.0.99)
Starting Vertica on all nodes. Please wait, databases with large catalog
may take a while to initialize.
Node Status: v_mydatabase_node0001: (DOWN) v_mydatabase_node0002: (DOWN)
Node Status: v_mydatabase_node0001: (DOWN) v_mydatabase_node0002: (DOWN)
Node Status: v_mydatabase_node0001: (DOWN) v_mydatabase_node0002: (DOWN)
Node Status: v_mydatabase_node0001: (DOWN) v_mydatabase_node0002: (DOWN)
Node Status: v_mydatabase_node0001: (UP) v_mydatabase_node0002: (UP)
Database mydatabase started successfully

$ vsql

=> \dt store_sales*
List of tables
Schema | Name                 | Kind  | Owner
-------+----------------------+-------+---------
store  | store_sales          | table | dbadmin
store  | store_sales_fact     | table | dbadmin
store  | store_sales_original | table | dbadmin
(3 rows)
```

Advantages and Limitations of Full Database Replication

Advantages	Limitations
• You can create a full copy of the source cluster database.	• All nodes on the source cluster must be up.

• If you run copycluster multiple times, the copy is incremental.	• The target cluster must be stopped. • To restore data, the source and target clusters must have the same: • Number of nodes • Database name • dbadmin users • Vertica version

Copy Specific Tables Between Similar Vertica Clusters

Suppose, you have a Vertica database with 50 tables on the source cluster and you have backed up 50 tables on the target clusters. You add 5 new tables and want to copy the 5 new tables to the target cluster to replicate data between the 2 Vertica clusters.

In this situation, you can just copy specific tables from one cluster to another, as long as your clusters are identical.

Prerequisites:

- The target cluster database name can be same or different than the source cluster database name.
- The source and the target cluster must have the same:
  - Number of nodes
  - Vertica version
  - dbadmin username
- The source and the target cluster can have different:
  - Node names
  - Database name
- The target cluster must have a different IP address than the source cluster.

How to Copy Specific Tables Between Similar Vertica Clusters

To copy a table, follow these steps, which lead you through creating a table on your source cluster and copying it to the target.

1. On the source cluster, create a table named `cluster1_table1`.

```
$ vsql
=> CREATE TABLE cluster1_table1 (i INT NOT NULL) PARTITION BY i;
CREATE TABLE

=> COPY cluster1_table1 FROM stdin;
Enter data to be copied followed by a newline.
```

```
End with a backslash and a period on a line by itself.
>> 1
>> 2
>> 3
>> 4
>> 5
>> \.

=> SELECT * FROM cluster1_table1;
i
---
2
3
5
1
4

=> \dt
                       List of tables
Schema |           Name           | Kind  | Owner
-------+--------------------------+-------+---------
public | cluster1_table1          | table | dbadmin
public | table_that_wont_be_moved | table | dbadmin
```

2. On the source cluster, create a backup configuration file `object_specific_replication.ini`. Run the `vbr` script with the `--setupconfig` option and answer the prompts.

```
$ /opt/vertica/bin/vbr --setupconfig
Snapshot name (backup_snapshot):
Destination Vertica DB bin directory (only required for object
replication) (/opt/vertica/bin):
Number of restore points (1): 3
Specify objects (no default): public.cluster1_table1
Object restore mode (coexist, createOrReplace or create)
(createOrReplace): coexist
Vertica user name (dbadmin):
Save password to avoid runtime prompt? (n) [y/n]: n
Node v_mydatabase_node0001
Backup host name (no default): 10.100.0.88
Backup directory (no default): /vertica/backuplocation
Node v_mydatabase_node0002
```

```
Backup host name (no default): 10.100.0.99
Backup directory (no default): /vertica/backuplocation
Change advanced settings? (n) [y/n]: n
Config file name (test1.ini): object_specific_replication.ini
Saved vbr config to object_specific_replication.ini.
```

You can choose from the following options for the object restore mode.

- `coexist`: When you specify `coexist` for the object restore mode, Vertica creates all restored objects with the prefix `<backup>_<timestamp>_<object_name>`. This option allows existing and restored objects to exist simultaneously.
- `createOrReplace`: When you specify `createOrReplace` for the object restore mode, Vertica creates an object on the target cluster with the same name as the object on the source cluster. If an object with the same name exists on the target cluster, Vertica drops the existing object and replaces it with a new object with the same name as on the source cluster.
- `create`: When you specify `create` for the object restore mode, Vertica creates an object on the target cluster with the same name as the object on the source cluster. However, with the `create` option, Vertica cannot replace any existing object with the same name.

For this example the name of the backup configuration file is `object_specific_replication.ini`.

```
$ cat object_specific_replication.ini

[Misc]
snapshotName = backup_snapshot
dest_verticaBinDir = /opt/vertica/bin
restorePointLimit = 3
objects = public.cluster1_table1
objectRestoreMode = coexist

[Database]
dbName = mydatabase
dbUser = dbadmin
dbPromptForPassword = True

[Mapping]
v_mydatabase_node0001 = 10.100.0.88:/vertica/backuplocation
v_mydatabase_node0002 = 10.100.0.99:/vertica/backuplocation
```

3. On the source cluster, in the `object_specific_replication.ini` file, change the database name to the target database name and map the nodes appropriately. The

changed text is in **bold.**

```
$ cat object_specific_replication.ini
[Misc]
snapshotName = backup_snapshot
dest_verticaBinDir = /opt/vertica/bin
restorePointLimit = 3
objects = public.cluster1_table1
objectRestoreMode = coexist

[Database]
dbName = targetdb
dbUser = dbadmin

[Mapping]
v_mydatabase_node0001 = 10.100.0.88:vertica/backuplocation
v_mydatabase_node0002 = 10.100.0.99:vertica/backuplocation
```

4. On the source cluster, replicate the tables from the source cluster to the target cluster by running the `vbr` script with the `replicate` option.

```
$ vbr -t replicate -c object_specific_replication.ini
Configured backup directories in Mapping are ignored for object
replication.
Starting replication of objects ['public.cluster1_table1'] from
mydatabase.
Participating nodes: v_mydatabase_node0001, v_mydatabase_node0002.
Enter vertica password:
Snapshotting the source database.
Snapshot complete.
Copying catalog snapshot from source to destination.
Preparing destination database for replicating objects.
Prep complete, start syncing files.
Complete syncing files, removing snapshot from the source database.
Finalizing object replication.
Object replication complete!
```

5. This step is only applicable, if you specify `coexist` for the object restore mode. Move the data from `cluster1_table1` to another table in the target database. To move your data using `move_partitions_to_table`, the two tables must have identical schema, projections, and partitions. When you move your data, Vertica removes the data from its original location.

a. On the target cluster, to view both the schemas, use the `\dt` command.

```
$ vsql
=> \dt
                              List of tables
               Schema                 |     Name        | Kind  |
Owner
--------------------------------------+-----------------+-------+--
------
backup_snapshot_20160203221054_public | cluster1_table1 | table |
dbadmin
public                                | cluster2_table2 | table |
dbadmin
```

b. To view the existing data in the table `public.cluster2_table2`, use the following command:

```
=> SELECT * FROM public.cluster2_table2;
i
----
15
13
12
14
11
```

c. Move the data from `backup_snapshot_20160203221054_cluster1_table1` to `public.cluster2_table2` using the following command:

```
$ vsql
=> \dt
                              List of tables
                Schema                |      Name      | Kind |
Owner
--------------------------------------+----------------+------+---
-----
backup_snapshot_20160203221054_public |cluster1_table1 |table
|dbadmin
public                                |cluster2_table2 |table
|dbadmin

=> SELECT MOVE_PARTITIONS_TO_TABLE('backup_snapshot_20160203221054_
public.cluster1_table1', 1,5,'public.cluster2_table2');
MOVE_PARTITIONS_TO_TABLE
```

```
-----------------------------------------------
5 distinct partition values moved at epoch 17.

=> SELECT * from cluster2_table2;
i
-----
 5
 3
 2
11
15
15
13
12
 4
 1
```

Now, `public.cluster2_table2` has data from both tables, and `backup_snapshot_20160203221054_public.cluster1_table1` has no data.

Advantages and Limitations of Copying Specific Tables Between Similar Vertica Clusters

Advantages	Limitations
• The source and the target clusters can be up. • Perform multiple, incremental backups. • Sync tables bidirectionally to different backup locations. • Perform fast incremental backups at regular intervals (if no significant changes). • Choose individual tables to migrate. • Faster than full backup, so saves disk space at the backup location. • Can have different database name and dbadmin users between source and target clusters.	• The tables must have the same schema and projection definition. • You must manually move the user-defined libraries and functions in the source and the target clusters. • User-defined libraries and functions in the source and the target clusters should be the same.

For More Information

- Vertica Community Edition
- Vertica User Community
- Vertica Documentation