CS300 Couchbase NoSQL Server Administration

Lab 7.2 Exercise Manual

Release: 6.5.1

Revised: June 22nd, 2020

Lab #7.2: Backup/Restore using cbbackupmgr tool

Objective: This lab will walk you through how to backup a bucket's files and restore it to a live cluster.

Overview: **The following high-level steps are involved in this lab:**

- Use cbbackupmgr to backup data in a bucket to some files
- Use cbbackupmgr to restore the files back into a live bucket

Enterprise Backup Tutorial

This tutorial gives examples of how to use all of the commands in the 'cbbackupmgr' tool effectively.

This tutorial shows how to take backups and restore data using cbbackupmgr. This tutorial uses a cluster that contains both the travel-sample and beer-sample buckets installed and requires you to modify some of the documents in the travel-sample bucket.

Using this cluster we will show how the incremental/merge approach taken by cbbackupmgr reduces time and overhead on your cluster.

Configuring a Backup

Before getting started with cbbackupmgr you must first decide the directory where to store all of your backups. This directory is referred to as the backup archive. The backup archive contains one or more backup repositories. These backup repositories are where your backups will be contained. The easiest way to think of a backup repository is that it corresponds directly to a single cluster that you want to back up. The backup repository also contains a configuration for how to back that cluster up. A backup repository is created by using the config sub-command. In this tutorial we will use a backup archive located at /data/backup. The backup archive is automatically created if the directory specified is empty. Below is an example of how to create a backup repository called "cluster" which backs up all data and index definitions from all buckets in the target cluster.

May need to become root to write in the /data/backup dir.

```
$ cbbackupmgr config --archive /data/backup --repo cluster

Backup repository `cluster` created successfully in archive
`/data/backup`
```

One of the most important aspects of backup repository creation is that you can configure that backup repository in many different ways to change the way backups in each backup repository are taken. Let's

say you want a separate backup of only the index definitions in the travel-sample bucket. To do this you can create a separate backup repository called "single" using the following command:

```
$ cbbackupmgr config --archive /data/backup --repo single \
--include-buckets travel-sample --disable-data

Backup repository `single` created successfully in archive
`/data/backup`
```

The config sub-command provides many options in order to customize how you backup your data. For more information about the available options and how they are used, see cbbackupmgr config.

Backing up a Cluster

Now that you have created some backup repositories let's take a look at the backup archive to see what it looks like. The easiest way to do this is to use the list sub-command. This sub-command is used to examine a backup archive and gives information about how much data is stored in it. To see the entire backup archive, run the following command:

```
$ cbbackupmgr list --archive /data/backup

Size      Items          Name
0B        -              /
0B        -              + cluster
0B        -              + single
```

The list sub-command returns a directory print out of all of the backup repositories and backups in your backup archive. Since there are no backups yet you can just see your archives list in the output of this command. There is also information about how much disk space each folder and file contains and, if applicable, how many items are backed up in those folders/files. For more information about the list sub-command, see cbbackupmgr list.

Now that you have your backup repositories configured it's time to start taking backups. Since the backup repository contains all of the configuration information for how the backup should be taken you just need to specify the backup repository name and the information for the target cluster you intend to back up.
Below is an example of how to take a backup on the "cluster" backup repository. Let's assume that your cluster is running on localhost.

```
$ cbbackupmgr backup --archive /data/backup --repo cluster \
--cluster couchbase://127.0.0.1 --username Administrator --password
couchbase
```

```
Backing up to 2016-03-22T10_26_08.933579821-07_00
Copied all data in 6s (Avg. 6.67MB/Sec)        38894 items / 40.02MB
travel-sample           [==================================] 100.00%
beer-sample             [==================================] 100.00%

Backup successfully completed
```

When the backup command is executed, by default it prints out a progress bar which is helpful to understand how long your backup will take to complete and the rate of data movement. While the backup is running, the progress bar gives an estimated time to completion, and when the backup completes, but this changes to the average backup rate. Information is also provided on the total data and items already backed up and the current rate of data movement. If the backup completes successfully, the tool prints the message "Backup completed successfully" as the last line.

Let's also run the backup on the "single" backup repository to see how the two backup runs differ.

```
$ cbbackupmgr backup --archive /data/backup --repo single \
--cluster couchbase://127.0.0.1 --username Administrator --password
couchbase

Backing up to 2016-03-22T10_33_20.812668465-07_00
Copied all data in 1s (Avg. 480B/Sec)                  0 items / 480B
travel-sample           [==================================] 100.00%
```

Since the "single" backup repository is only configured to back up index definitions for the travel-sample bucket you do not see a progress bar for the beer-sample bucket. You can also see that the backup executed quicker since there was considerably less data to actually back up.

Now that you have backups in your backup archive let's take a look at how the state of our backup archive has changed by using the list sub-command.

```
$ cbbackupmgr list --archive /data/backup

Size      Items          Name
154.25MB  -              /
154.21MB  -              + cluster
154.21MB  -                  + 2016-03-22T10_26_08.933579821-07_00
55.85MB   -                      + beer-sample
298B      0                          bucket-config.json
55.84MB   7303                       + data
55.84MB   7303                           shard_0.fdb
2B        0                          full-text.json
10.07KB   8                          gsi.json
784B      1                          views.json
98.36MB   -                      + travel-sample
300B      0                          bucket-config.json
98.35MB   31591                      + data
98.35MB   31591                          shard_0.fdb
2B        0                          full-text.json
```

```
10.07KB   8                          gsi.json
1.72KB    1                          views.json
40.08KB   -                  + single
40.08KB   -                      + 2016-03-22T10_33_20.812668465-07_00
40.08KB   -                          + travel-sample
300B      0                              bucket-config.json
28.00KB   0                              + data
28.00KB   0                                  shard_0.fdb
2B        0                              full-text.json
10.07KB   8                              gsi.json
1.72KB    1                              views.json
```

Now that you have some backups defined, the output of the list sub-command is much more useful. You can see that the "cluster" backup repository contains one backup with a name corresponding to the time the backup was taken. That backup also contains two buckets and you can see various files in each of those backups with their size and item counts. The "single" backup repository also contains one backup, but this backup only contains the travel-sample bucket and contains 0 data items.

One of the most important features of cbbackupmgr is that it is an incremental-only backup utility. This means that once you back up some data, you will never need to back it up again. In order to simulate some changes on the cluster you can run the 02-modify.sh script from the backup-tutorial GitHub repository mentioned at the beginning of the tutorial. If you do not have this script then you need to modify two documents and add two new documents to the travel-sample bucket. After you modify some data, run the backup sub-command on the "cluster" backup repository again.

```
$ cbbackupmgr backup --archive /data/backup --repo cluster \
--cluster couchbase://127.0.0.1 --username Administrator --password
couchbase

Backing up to 2016-03-22T14_00_38.668068342-07_00
Copied all data in 3s (Avg. 18.98KB/Sec)             4 items / 56.95KB
travel-sample            [====================================] 100.00%
beer-sample              [====================================] 100.00%

   Backup successfully completed
```

In this backup notice that since you updated 2 items and created two items, this is all that needs to be backed up during this run. Now list the backup archive using the list sub-command. You can see that the backup archive looks something like this:

```
$ cbbackupmgr list --archive /data/backup

Size      Items          Name
254.31MB  -              /
254.28MB  -              + cluster
```

```
154.19MB  -                 + 2016-03-22T10_26_08.933579821-07_00
55.84MB   -                     + beer-sample
298B      0                         bucket-config.json
55.83MB   7303                    + data
55.83MB   7303                        shard_0.fdb
2B        0                       full-text.json
9.99KB    8                       gsi.json
784B      1                       views.json
98.35MB   -                     + travel-sample
300B      0                         bucket-config.json
98.34MB   31591                   + data
98.34MB   31591                       shard_0.fdb
2B        0                       full-text.json
9.99KB    8                       gsi.json
1.72KB    1                       views.json
100.08MB  -                 + 2016-03-22T14_00_38.668068342-07_00
50.03MB   -                     + beer-sample
298B      0                         bucket-config.json
50.02MB   0                       + data
50.02MB   0                           shard_0.fdb
2B        0                       full-text.json
9.99KB    8                       gsi.json
784B      1                       views.json
50.05MB   -                     + travel-sample
300B      0                         bucket-config.json
50.04MB   4                       + data
50.04MB   4                           shard_0.fdb
2B        0                       full-text.json
9.99KB    8                       gsi.json
1.72KB    1                       views.json
40.08KB   -             + single
40.08KB   -                 + 2016-03-22T10_33_20.812668465-07_00
40.08KB   -                     + travel-sample
300B      0                         bucket-config.json
28.00KB   0                       + data
28.00KB   0                           shard_0.fdb
2B        0                       full-text.json
10.07KB   8                       gsi.json
1.72KB    1                       views.json
```

Restoring a Backup

Now that you have some backup data let's restore that data backup to the cluster. In order to restore data you just need to know the name of the backup that you want to restore. To find the name you can use the list sub-command in order to see what is in our backup archive. The backup name will always be a timestamp. For example, let's say you want to restore the 2016-03-22T10_26_08.933579821-07_00 from the "cluster" backup repository. In order to do this, run the following command:

```
$ cbbackupmgr restore --archive /data/backup --repo cluster \
--cluster http://127.0.0.1:8091 --username Administrator --password
couchbase \
--start 2016-03-22T14_00_16.892277632-07_00 \
--end 2016-03-22T14_00_16.892277632-07_00 --force-updates

(1/1) Restoring backup 2016-03-22T14_00_16.892277632-07_00
Copied all data in 2s (Avg. 19.96MB/Sec)       38894 items / 39.91MB
travel-sample            [====================================] 100.00%
beer-sample              [====================================] 100.00%

Restore completed successfully
```

In the command above, notice the use of the --start and --end flags to specify the range of backups you want to restore. Since you are only restoring one backup, specify the same value for both --start and --end. The --force-updates flags skip Couchbase conflict resolution. This tells cbbackupmgr to force overwrite key-value pairs being restored even if the key-value pair on the cluster is newer than the one being restored. If you look at the two values that were updated on the cluster, you will now see that they have been reverted to what they were at the time we took the initial backup. If you used the script in the backup-tutorial GitHub repository to update documents then you can use the 03-inspect.sh script to see the state of the updated documents after the restore.

You can also use the restore sub-command to exclude data that was backed up from the restore and provide various other options. For more information on restoring data, see cbbackupmgr restore.

Merging backups

Using an incremental backup solution means that each backup you take increases the disk space. Since disk space in not infinite you need to be able to reclaim this disk space. In order to do this, use the merge sub-command to merge two or more backups together. Since there are two backups in the "cluster" backup repository, you can merge these backups together using the following command:

```
$cbbackupmgr merge --archive /data/backup --repo cluster \
--start 2016-03-22T14_00_16.892277632-07_00 \
--end 2016-03-22T14_00_38.668068342-07_00

Merge completed successfully
```

After merging the backups together you can use the list sub-command to see the effect of the merge sub-command on the backup archive.

```
$ cbbackupmgr list --archive /data/backup
Size      Items          Name
154.41MB  -              /
154.37MB  -              + cluster
154.37MB  -                  + 2016-03-22T14_00_38.668068342-07_00
```

```
55.84MB    -                        + beer-sample
298B       0                            bucket-config.json
55.83MB    7303                         + data
55.83MB    7303                             shard_0.fdb
2B         0                            full-text.json
9.99KB     8                            gsi.json
784B       1                            views.json
98.53MB    -                        + travel-sample
300B       0                            bucket-config.json
98.52MB    31593                        + data
98.52MB    31593                            shard_0.fdb
2B         0                            full-text.json
9.99KB     8                            gsi.json
1.72KB     1                            views.json
40.08KB    -                + single
40.08KB    -                    + 2016-03-22T10_33_20.812668465-07_00
40.08KB    -                        + travel-sample
300B       0                            bucket-config.json
28.00KB    0                            + data
28.00KB    0                                shard_0.fdb
2B         0                            full-text.json
10.07KB    8                            gsi.json
1.72KB     1                            views.json
```

You can see from the list command that there is now a single backup in the "cluster" backup repository. This backup has a name that reflects the name of the most recent backup in the merge. It also has 31593 data items in the travel-sample bucket. This is two more items than the original backup you took because the second backup had two new items. The two items that were updated were de-duplicated during the merge so they do not add extra items to the count displayed by the list sub-command.

For more information on how the merge command works as well as information on other ways the merge command can be used, see cbbackupmgr merge.

Removing a Backup Repository

If you no longer need a backup repository, you can use the remove sub-command to remove the backup repository. Below is an example showing how to remove the "cluster" backup repository.

```
$ cbbackupmgr remove --archive /data/backup --repo cluster

Backup repository `cluster` deleted successfully from archive
`/data/backup`
```

If you now run the list sub-command you will see that the "cluster" backup repository no longer exists. For more information on the remove sub-command, see cbbackupmgr remove.

Couchbase

Lab-7.2: cbbckupmgr tool &cbexport/cbimport

Cbexport and cbimport

```
[root@Couchbase01 bin]# ./cbexport json -c couchbase://127.0.0.1 -u Administrator -p couchbase -b gamesim-sample -o /data/list.json -f list
Json exported to `/data/list.json` successfully

[root@Couchbase01 bin]# ls -alh /data/list.json
-rw-rw----. 1 root root 94K Jan 11 19:57 /data/list.json

[root@Couchbase01 bin]# more /data/list.json
[{
    "experience": 337,
    "hitpoints": 10,
    "jsonType": "player",
    "level": 2,
    "loggedIn": false,
    "name": "Jan0",
    "uuid": "e7b3823b-6d56-4e62-b7e4-e8acbbfba0c8"
}
,
{
    "experience": 14248,
    "hitpoints": 23832,
    "jsonType": "player",
    "level": 141,
    "loggedIn": true,
```

```
    "name": "Aaron1",
    "uuid": "78edf902-7dd2-49a4-99b4-1c94ee286a33"
}
,
{"jsonType":"item","name":"Axe_ce6667f2-5dea-4b27-bc98-
c744b309aacf","uuid":"ce6667f2-5dea-4b27-bc98-
c744b309aacf","ownerId":"Melinda0"},
{
    "experienceWhenKilled": 19,
    "hitpoints": 3583,
    "itemProbability": 0.46180518984163765,
    "jsonType": "monster",
    "name": "Wild-man6",
    "uuid": "cc87dcee-36fd-4f33-a64e-bf979099c462"
}
,
{"jsonType":"item","name":"Mace_b8c0bb46-90da-45d1-bbec-
e5144beed9f0","uuid":"b8c0bb46-90da-45d1-bbec-
e5144beed9f0","ownerId":"Trond2"},
{"jsonType":"item","name":"Goblinshredder_6d2f21f1-9ec2-4570-b8ae-
e84bb98178df","uuid":"6d2f21f1-9ec2-4570-b8ae-
e84bb98178df","ownerId":"Steve1"},
{"jsonType":"item","name":"Corrupted_Ash_bringer_12443571-b22a-4fe3-
850a-d167a0e4a321","uuid":"12443571-b22a-4fe3-850a-
d167a0e4a321","ownerId":"Keith1"},
{"jsonType":"item","name":"Axe_c8c4d9fd-22e8-4669-9745-
7dd653ab048a","uuid":"c8c4d9fd-22e8-4669-9745-
7dd653ab048a","ownerId":"Tony0"},
{"jsonType":"item","name":"Corrupted_Ash_bringer_c9e7ae7c-bb34-4696-
967c-4cee358bb686","uuid":"c9e7ae7c-bb34-4696-967c-
4cee358bb686","ownerId":"Volker2"},
{"jsonType":"item","name":"Axe_c2e955fa-0848-4bc7-8654-
02d5d42e4db8","uuid":"c2e955fa-0848-4bc7-8654-
02d5d42e4db8","ownerId":"Melinda0"},
```

```
{"jsonType":"item","name":"Mood_of_Shadowsong_7ee4e41c-195e-4177-98c8-
776f80b66ca4","uuid":"7ee4e41c-195e-4177-98c8-
776f80b66ca4","ownerId":"Dale1"},

{"jsonType":"item","name":"Meatwagon_f31e738e-d4a8-4e9d-afdb-
d7a1dd0e240d","uuid":"f31e738e-d4a8-4e9d-afdb-
d7a1dd0e240d","ownerId":"Jan1"},

{"jsonType":"item","name":"Katana_a5df8154-7ee5-471f-b23b-
0eddd4f45bfe","uuid":"a5df8154-7ee5-471f-b23b-
0eddd4f45bfe","ownerId":"Mike1"},

{"jsonType":"item","name":"Bristleblitzstriker_768034bd-1d7d-4137-
ae66-10a387f162da","uuid":"768034bd-1d7d-4137-ae66-
10a387f162da","ownerId":"Damien2"},

{"jsonType":"item","name":"Meatwagon_cf6773eb-344f-44a2-979b-
e68772f326b7","uuid":"cf6773eb-344f-44a2-979b-
e68772f326b7","ownerId":"Dale0"},

{"jsonType":"item","name":"Forsaken_Catapult_a65d7abe-dc15-4206-a9f4-
75557de9c47b","uuid":"a65d7abe-dc15-4206-a9f4-
75557de9c47b","ownerId":"Aliaksey1"},

{

    "experienceWhenKilled": 70,

    "hitpoints": 279,

    "itemProbability": 0.38271562657579794,

    "jsonType": "monster",

    "name": "Fachen0",

    "uuid": "f75abb6b-a753-4cbb-a244-167d278e42e1"

}

,

{"jsonType":"item","name":"Forsaken_Catapult_091bc471-18e6-43ca-b418-
557aaef31224","uuid":"091bc471-18e6-43ca-b418-
557aaef31224","ownerId":"Volker2"},

{

    "experience": 16870,

    "hitpoints": 21405,

    "jsonType": "player",
```

Couchbase


```
    "level": 168,
    "loggedIn": false,
    "name": "Claire2",
    "uuid": "141f8848-4a22-4f9c-bbd6-4305d8bd4b0b"
}
,
{
    "experienceWhenKilled": 58,
    "hitpoints": 4397,
    "itemProbability": 0.13102747831116571,
    "jsonType": "monster",
    "name": "Wild-man7",
    "uuid": "5bbb942d-f9cd-4b20-8cb5-31a6d159acd2"
}
```

Note: you may need to make a "default" bucket to import to.

```
[root@Couchbase01 bin]# ./cbimport json -c couchbase://127.0.0.1 -u
Administrator -p couchbase -b default  -d file:///data/list.json -g
key::%name% -f list
Json `file:///data/list.json` imported to `http://127.0.0.1:8091`
successfully
[root@Couchbase01 bin]#
```

This concludes Lab 7.2