Chapter 21.

September 2018

Welcome to the September 2018 edition of DataStax Developer's Notebook (DDN). This month we answer the following question(s);

My company is investigating using DataStax for our new Customer/360 system in our customer call center. I'm a developer, and do not know how to administer DataStax Enterprise, but, I need to know how to backup and restore tables for my programming and unit tests. Can you help ?

Excellent question ! DataStax Enterprise (DSE) cab be backed up and restored using DataStax Operations Center (DSE Ops Center), including activities to block stores like Amazon S3, other. You can also perform sstabledump(s), and table unloads and loads, including bulk unloads and loads.

But, as you seek to perform these activities as part of your unit tests, we are going to detail table backup and restore using snapshots; faster, less code, easily automated.

Software versions

The primary DataStax software component used in this edition of DDN is DataStax Enterprise (DSE), currently release 6.0.2. All of the steps outlined below can be run on one laptop with 16 GB of RAM, or if you prefer, run these steps on Amazon Web Services (AWS), Microsoft Azure, or similar, to allow yourself a bit more resource.

For isolation and (simplicity), we develop and test all systems inside virtual machines using a hypervisor (Oracle Virtual Box, VMWare Fusion version 8.5, or similar). The guest operating system we use is CentOS version 7.0, 64 bit.

21.1 Terms and core concepts

As stated above, ultimately the end goal is to backup and restore DataStax Enterprise (DSE) tables as part of (programming) unit tests and similar. For reasons of simplicity, speed, less programming, other, this document details using DSE snapshots to backup and restore.

Comments:

– DataStax Enterprise (DSE) Operations Center (DSE Ops Center) backup and restore is detailed here,

  `https://docs.datastax.com/en/opscenter/6.5/opsc/online_help/services/opscBackupService.html`

  `https://docs.datastax.com/en/dse/6.0/dse-admin/datastax_enterprise/operations/opsBackupRestoreTOC.html`

– You can also backup and restore tables using sstabledump, by use of the bulk loaders, and also table load and unload.

Review of the Runtime Environment

Example 21-1displays a specific entry from the cassandra.yaml files, and a specific directory listing. A code review follows.

Example 21-1 Finding (system) keyspaces on a new DSE cluster.

```
cassandra.yaml
   data_file_directories:
     - /opt/dse/node1/data/data


cd /opt/dse/node1/data/data
ls -l
   drwxr-xr-x  4 root root 4096 Jul  1 11:59 dse_leases
   drwxr-xr-x  3 root root 4096 Jul  1 11:59 dse_perf
   drwxr-xr-x  5 root root 4096 Jul  1 11:58 dse_security
   drwxr-xr-x  3 root root 4096 Jul  1 11:59 dse_system
   drwxr-xr-x  3 root root 4096 Jul  1 11:59
dse_system_local
   drwxr-xr-x  3 root root 4096 Jul  1 11:59 solr_admin
   drwxr-xr-x 18 root root 4096 Jul  1 11:58 system
   drwxr-xr-x  5 root root 4096 Jul  1 11:58 system_auth
   drwxr-xr-x  7 root root 4096 Jul  1 11:58
system_distributed
   drwxr-xr-x 12 root root 4096 Jul  1 11:58 system_schema
   drwxr-xr-x  4 root root 4096 Jul  1 11:58 system_traces
```

Relative to Example 21-1, the following is offered:

– The cassandra.yaml at 1000 lines offers 100 or so identifiers, capacities, tunables, and other settings related to DSE Core. (900 lines of comments, blank lines, other.)

– One such entry specifies the single or set of directories that store DSE keyspaces. DSE keyspaces are a logical entity that specify the replication factor, other, specific to the DSE tables contained within.

– The directory listing above details the keyspaces contained in a nearly new DSE cluster. Per the settings above, any newly created (normal, end user) keyspaces, and the tables it will contain, will be located in this parent directory.

Given the creation of a new keyspaces titled, ks_6254, and a table within same titled, cust-orders, the display from Example 21-1 changes. Example as displayed in Example 21-2. A code review follows.

Example 21-2   Adding (normal, end user) keyspaces to a DSE cluster; effect

```
...
drwxr-xr-x  3 root root 4096 Jul  1 11:59 dse_system_local
drwxr-xr-x  3 root root 4096 Jul  1 11:59 ks_6254
drwxr-xr-x  3 root root 4096 Jul  1 11:59 solr_admin
         ...

cd ks_6254
ls -l
   drwxr-xr-x 3 root root 4096 Jul  1 11:59
cust_orders-8eda10 ...

cd cust*
```

```
ls -l
   drwxr-xr-x 2 root root 4096 Jul  1 11:59 backups


cd back*
ls -l
   (All above just empty directories)
```

Relative to Example 21-2, the following is offered:

– With the creation of a new keyspace titled, ks_6254, a new directory entry is created in the target directory. (The directory specified in cassandra.yaml.)

– Inside said keyspace is a listed for any tables contained in same. In this example, we created a DSE table titled, cust_orders.

Note: DSE table names are suffixed with a UUID at the operating system level, to ensure uniqueness as the same named table may me dropped and recreated.

– As a newly created table, the contents of the directory specific to said table may be empty. (No data has been added or flushed yet.) There will be a backups directory, which may also be initially empty.

After adding data to the table above (to ensure we have something to restore; proof that this works), we execute the following:

```
nodetool cleanup ks_6254

nodetool snapshot -t ks_6254-snap1 ks_6254
```

The "cleanup" command is just a nervous tick; DSE can be configured to automatically snapshot a table upon receipt of a truncate command. The cleanup will remove (retire) any of these (other) snapshots.

The "snapshot" command calls for a backup of the named keyspace, and labels that snapshot, ks_6254-snap1 (a variable name of our choosing).

Example 21-3 details how the contents of the data directory will have changed. A code review follows.

Example 21-3 Contents of the data directory after the snapshot.

```
pwd
```

```
/opt/dse/node1/data/data/ks_6254/cust_orders-8eda10d17d5811
e8839b2baf872cd95b

ls -l
   -rw-r--r-- 2 root root   47 Jul  1 12:04
aa-1-bti-CompressionInfo.db
   -rw-r--r-- 2 root root  135 Jul  1 12:04 aa-1-bti-Data.db
   -rw-r--r-- 2 root root   10 Jul  1 12:04
aa-1-bti-Digest.crc32
   -rw-r--r-- 2 root root   16 Jul  1 12:04
aa-1-bti-Filter.db
   -rw-r--r-- 2 root root   73 Jul  1 12:04
aa-1-bti-Partitions.db
   -rw-r--r-- 2 root root    0 Jul  1 12:04 aa-1-bti-Rows.db
   -rw-r--r-- 2 root root 4782 Jul  1 12:04
aa-1-bti-Statistics.db
   -rw-r--r-- 2 root root   94 Jul  1 12:04 aa-1-bti-TOC.txt
   drwxr-xr-x 2 root root 4096 Jul  1 11:59 backups
   drwxr-xr-x 3 root root 4096 Jul  1 12:04 snapshots

cd snapshots
ls -l
      drwxr-xr-x 2 root root 4096 Jul  1 12:04 ks_6254-snap1
cd ks*
ls -l
   -rw-r--r-- 2 root root   47 Jul  1 12:04
aa-1-bti-CompressionInfo.db
   -rw-r--r-- 2 root root  135 Jul  1 12:04 aa-1-bti-Data.db
   -rw-r--r-- 2 root root   10 Jul  1 12:04
aa-1-bti-Digest.crc32
   -rw-r--r-- 2 root root   16 Jul  1 12:04
```

```
aa-1-bti-Filter.db
   -rw-r--r-- 2 root root   73 Jul  1 12:04
aa-1-bti-Partitions.db
   -rw-r--r-- 2 root root    0 Jul  1 12:04 aa-1-bti-Rows.db
   -rw-r--r-- 2 root root 4782 Jul  1 12:04
aa-1-bti-Statistics.db
   -rw-r--r-- 2 root root   94 Jul  1 12:04 aa-1-bti-TOC.txt
   -rw-r--r-- 1 root root   31 Jul  1 12:04 manifest.json
   -rw-r--r-- 1 root root  942 Jul  1 12:04 schema.cql
```

Relative to Example 21-3, the following is offered:

- In this directory listing, we see that any contents of the table that resided only in memory are flushed.
- And for the first time, the snapshots sub-directory has an entry for the named snapshot. That directory too has entries, largely similar to the table proper. (Plus a schema.cql and manifest.json file.)

Per this Example, Truncate the Table

Continuing with this example, we would truncate the table, and run a SELECT to confirm there are in fact, no rows in the table. Example as shown in Example 21-4. A code review follows.

Example 21-4 Truncating the example table.

```
cqlsh> truncate table ks_6254.cust_orders;
cqlsh> select * from ks_6254.cust_orders;
    (0 rows)


pwd
    /opt/dse/node1/data/data/ks_6254/cust_orders- \
                 8eda10d17d5811e8839b2baf872cd95b/snapshots


ls -l
   drwxr-xr-x 2 root root 4096 Jul  1 12:04 ks_6254-snap1
   drwxr-xr-x 2 root root 4096 Jul  1 12:11
truncated-1530468681275-cust_orders
```

Relative to Example 21-4, the following is offered:

- The truncate table and following SELECT confirm that the table is now empty.
- As stated above, the truncate command itself creates snapshot.

Restoring from snapshot

DSE snapshots can be restored to the source node, or another node or DSE cluster entirely, as detailed here.

```
https://docs.datastax.com/en/dse/6.0/dse-admin/datastax_enterprise/o
perations/opsSnapshotRestoreNewCluster.html
```

In our continuing example, we wish to restore from snapshot.

Note: *If you had copied any snapshots you wanted to preserve off to the side*, you could at this time run a,

```
nodetool clearsnapshot --all
```

"all" deletes all snapshots, and this command can accept specifically named snapshots.

With the table now empty, copy the snapshot back into the original table location via a,

```
cp \

/opt/dse/node1/data/data/ks_6254/cust_orders-8eda10d17d5811e8839b2ba
f872cd95b/snapshots/ks_6254-snap1/*  \

/opt/dse/node1/data/data/ks_6254/cust_orders-8eda10d17d5811e8839b2ba
f872cd95b


# In effect, a standard Linux copy command,

# cp   source   destination

# cp   snapshot_dir /*  data_dir
```

Last step

The work we performed was local to one node and at the operating system level. As such, we should run a "nodetool refresh". Example as shown,

```
nodetool refresh ks_6254 cust_orders
```

The nodetool command, as shown, will ensure that the DSE cluster knows about our changes, and correct for any data entropy with any other DSE nodes that may have been present.

21.2 Complete the following

At this point in this document we have completed a brief how to related to ones means of backup and recovery using DataStax Enterprise (DSE).

Make a keyspace, table, add data, create a snapshot, truncate the table, and restore. Confirm that your data is present.

21.3 In this document, we reviewed or created:

This month and in this document we detailed the following:

- An overview of several DataStax Enterprise means to backup and recover data.
- Detailed a specific and complete means to accomplish same using DSE snapshots.

Persons who help this month.

Kiyu Gabriel, Matt Atwater, and Jim Hatcher.

Additional resources:

Free DataStax Enterprise training courses,

```
https://academy.datastax.com/courses/
```

Take any class, any time, for free. If you complete every class on DataStax Academy, you will actually have achieved a pretty good mastery of DataStax

Enterprise, Apache Spark, Apache Solr, Apache TinkerPop, and even some programming.

This document is located here,

```
https://github.com/farrell0/DataStax-Developers-Notebook
https://tinyurl.com/ddn3000
```