**ISIT312 Big Data Management**

## Session 4, 2021

## Exercise 3

## Introduction to Hive

**In this exercise, you will get familiar with how to start Hive Server 2, how to use command line and graphical user interfaces to Hive, how to create internal and external tables in Hive, and how the relational view of data provided by Hive is implemented in HDFS.**

***Be careful when copying the Linux commands in this document to your working Terminal, because it is error-prone. Maybe you should type those commands by yourself***

**Prologue**

Login to your system and start VirtualBox.

When ready start a virtual machine ISIT312-BigDataVM-07-SEP-2020.

**(1) Where is Hive Server 2 ?**

Open a new Terminal window and process the following commands in Terminal window.

echo $HIVE\_HOME

echo $HIVE\_CONF\_DIR

The messages tell you where Hive is installed and where is Hive's configuration folder. Configuration folder contains a file hive-site.xml that includes the values of Hive configuration parameters.

Process a statement that lists the contents of hive-site.xml.

cat $HIVE\_CONF\_DIR/hive-site.xml

At the beginning of quite long list of messages you will get the following fragment of XML document.

<property>

<name>javax.jdo.option.ConnectionURL</name>

<value>jdbc:derby:;databaseName=/usr/share/hive/metastore\_db;

create=true</value>

<description>

JDBC connect string for a JDBC metastore.

To use SSL to encrypt/authenticate the connection, provide database-specific SSL flag in the connection URL.

For example, jdbc:postgresql://myhost/db?ssl=true for

postgres database.

</description>

</property>

A value of a property javax.jdo.option.ConnectionURL is jdbc:derby:;databaseName=/usr/share/hive/metastore\_db;create=true and it tells us what relational DBMS is used to implement Metastore (Derby) and where Metastore is located (/usr/share/hive/metastore\_db). Metastore (data dictionary or data repository in traditional DBMSs) contains all information about the mappings of Hive relational tables into the files in HDFS. **Deletion or re-initialization of Metastore means that all such mappings are lost !** Data located in HDFS is not changed. If you would like to reinitialize metastore (you probably do not need to do it now) then you have to first remove the present metastore\_db folder from $HIVE\_HOME

rm –rf $HIVE\_HOME/metastore\_db

and the process schematool program in the following way.

$HIVE\_HOME/bin/schematool -initSchema -dbType derby

Process a statement:

ls $HIVE\_HOME/bin

hiveserver2 is Hive2 Thrift server that will be used to access HDFS visible as a collection of relational tables. beeline is a command line interface to Hive2 server, but we use Zeppelin in this lab. schematool is a program for initialization of Hive Metastore.

**(2) How to start Metastore service and Hive Server 2 ?**

To start Hive’s metastore service, open ***a Terminal window***, type:

$HIVE\_HOME/bin/hive --service metastore

A message shows that metastore is up and running:

SLF4J: Actual binding is of type [org.apache.logging.slf4j.Log4jLoggerFactory]

To start hiveserver2, open ***another*** ***Terminal window*** and type:

$HIVE\_HOME/bin/hiveserver2

The same message shows that hiveserver2 is up and running.

**[IMPORTANT] Don’t use Zeppelin to run the above two commands, because the %sh interpreter has a timeout threshold.**

You can use Hive’s own interface to interact with Hive. Open another new Terminal window and process the following command.

$HIVE\_HOME/bin/beeline

Process the following cmmand in front of beeline> prompt.

!connect jdbc:hive2://localhost:10000

The statement connects you through JDBC interface to Hive 2 server running on a localhost and listening to a port 10000.

Press Enter when prompted about user name. Press Enter when prompted about password. The system should reply with a prompt

0: jdbc:hive2://localhost:10000>

To find what databases are available process the following command.

show databases;

At the moment only default database is available. We shall create new databases in the future. To find what tables have been created so far process a statement

show tables;

**(3) How to create an internal relational table ?**

To work with Hive in Zeppelin, you need to use the hive interpreter in the Zeppelin paragraphs.

Open a new Zeppelin paragraph. Type the following in the beginning of the paragraph:

%hive

which indicates that the Hive interpreter is used.

In the following contents, we use Beeline to interact with Hive.

To create a single column relational table process the following statement in "beeline window".

create table hello(message varchar(50));

Try again

show tables;

and

describe hello;

in "beeline window".

Hive created an internal relational table hello in HDFS. Is it possible to find a location of the table in HDFS ?

**(4) How to find a location of internal relational table in HDFS ?**

If you use Zeppelin then open a new paragraph and use %sh as the interpreter.

If you use command line interface then move to "Hadoop window"

Next, process the following command.

$HADOOP\_HOME/bin/hadoop fs -ls /user

Note a new folder hive created in HDFS folder. Use ↑(up) arrow key to redisplay a command processed just now and at the end of the command add /hive, i.e. process the following command.

$HADOOP\_HOME/bin/hadoop fs -ls /user/hive

The results show that a folder hive is not empty and it contains a folder warehouse. Use the following command to investigate what are the contents of a folder warehouse.

$HADOOP\_HOME/bin/hadoop fs -ls /user/hive/warehouse

And here we find our relational table hello implemented as a folder in HDFS. One more time, try to find what is in hello folder.

$HADOOP\_HOME/bin/hadoop fs -ls /user/hive/warehouse/hello

There is nothing there because the table is empty.

**(5) How to insert a row into an internal relational table ?**

If you use Zeppelin then return to %hive paragraph,

If you use command line interface then return to "beeline window".

To insert a row into a relational table hello process the following statement.

insert into hello values ('Hello world !');

Note, that insertion of a row takes some time. In the future we shall not use this way to populate Hive tables.

What about HDFS ? What has changed in HDFS after insertion of a row ? Return to "Hadoop window" and process the most recently processed command again.

$HADOOP\_HOME/bin/hadoop fs -ls /user/hive/warehouse/hello

A new file has been added to /user/hive/warehouse/hello HDFS folder.

Process the following command to list the contents of a new file.

$HADOOP\_HOME/bin/hadoop fs -cat /user/hive/warehouse/hello/000000\_0

And here we have a row recently inserted into a table hello.

Return to "beeline window" and insert few more rows. Then return to "Hadoop window" and list the contents of hello folder in the following way.

$HADOOP\_HOME/bin/hadoop fs -ls /user/hive/warehouse/hello

Try the following command to list the contents of all rows.

$HADOOP\_HOME/bin/hadoop fs -cat /user/hive/warehouse/hello/000000\*

**(6) How to create and how to process SQL script ?**

Right click Terminal icon at the bottom of task bar and open a new Terminal window.

Use a command

gedit hello.hql

to open a text editor with an empty file hello.hql.

Insert into the file the following lines.

insert into hello values('Hello HQL !');

select \* from hello;

Save a file and quit gedit editor. When you open a new Terminal window and you start gedit editor your current folder is your home folder and because of that the edited file is saved in your home folder. Note, that beeline has been started from you home folder as well. This is why you can process a script file hello.hql through beeline without providing a path to the script file. Now return to "beeline window" and process HQL script just created with the following command.

!run hello.hql

If you would like to save a report from processing of HQL script then you should first process a command

!record hello.rpt

then process HQL script with

!run hello.hql

and finally stop recording with

!record (no file name !)

Your report from processing of HQL script is stored in a file hello.rpt in the current folder of beeline which in this case is your home folder. Return to Terminal window used to create HQL script and process the following command to list the contents of a report.

cat hello.rpt

Now, you can close a Terminal window used to create HQL script file.

**(7) How to use SQL Developer ?**

To start SQL Developer leftclick at the second icon from top of task bar (an icon with a pretty large and green triangle) and wait for a while.

When ready look at Connections subwindow in the left upper corner of SQL Developer window and find hive connection there. It is a connection to Hive Server 2 we have already created for you. Rightclick at a connection icon (hive) and pick Connect option.

SQL Developer asks about Connection Information. Type any name you like and any password you like, and click at OK button.

SQL Developer adds default database to hive connection in Connections subwindow. Leftclick at a small + in front of default name. SQL Developer add Tables item. Leftclick at a small + in front of Table item, and so on ... Finally, left click at hello. SQL Developer lists the contents of hello table in its main window.

Now, retun to hive panel in SQL Developer main window. To process HQL statement type

select \* from hello where message like '%world !';

into worksheet window and leftclick at a large green triangle in the left upper corner of hive panel.

To access HQL script use option File->Open from the main SQL Developer menu and pick a file hello.hql in bigdata home folder.

To process the script click at an icon with a small green triangle next to an icon with a large green triangle. Confirm that you would like to use hive connection. A report from processing appears at the bottom of hello.hql panel. You can use a "floppy disk" icon to save it.

To disconnect SQL Developer from Hive Server 2 rightclick at hive connection icon and pick Disconnect option.

Use File->Exit option to quit SQL Developer. You can save HQL from worksheets into the files if you like.

**(8) How to load data into an internal relational table ?**

In this step you can use either Zeppelin interface or beeline command interface or SQL Developer to Hive Server 2. In Zeppelin interface use %hive paragraph.

A specification below is consistent with Zeppelin or beeline command interfaces.

As we found earlier loading data to an internal relational table with insert statement takes considerable amount of time. Additionally, Hive is not a typical database system that should be used to process online transactions where a small amount of data is collected from a user and inserted into a relational table. Hive supposed to operate on the large amounts of data that should be inserted into the relational table for more convenient processing with HQL. In this step, we practice a way how a large and well formatted data set can be loaded into an internal relational table.

Right click at Terminal icon at the bottom of task bar and open a new window. Start gedit editor and create a new file called names.tbl. Insert into the file the following 3 lines, save the file, and quit gedit.

James,Bond,35

Harry,Potter,16

Robin,Hood,120

Start gedit again and create HQL script names.hql with the following create table statement.

create table names(

first\_name VARCHAR(30),

last\_name VARCHAR(30),

age DECIMAL(3) )

row format delimited fields terminated by ',' stored as textfile;

Save the file, quit gedit, and close Terminal window.

Now, move to "beeline window" and process HQL script names.hql script with!run command.

!run names.hql

To load into a relational table names data from a file names.tbl process the following statement in "beeline window".

load data local inpath 'names.tbl' into table names;

Verify the results with

select \* from names;

**(9) How to load data into an external relational table ?**

When loading data into an internal relational table data located in a local file system get replicated in HDFS. It is much better to move data into HDFS and "overlap" ("cover") it with a definition of an external relational table.

To do so copy a file names.tbl to HDFS. Move to "Hadoop window" and process the following commands.

$HADOOP\_HOME/bin/hadoop fs -put names.tbl /user/bigdata

$HADOOP\_HOME/bin/hadoop fs -ls /user/bigdata

Right click at Terminal icon at the bottom of task bar and open a new window. Start gedit editor and create HQL script enames.hql with the following create table statement.

create external table enames(

first\_name VARCHAR(30),

last\_name VARCHAR(30),

age DECIMAL(3) )

row format delimited fields terminated by ','

stored as textfile location '/user/bigdata';

Save the file, quit gedit, and close Terminal window.

Move to "beeline window" and process a script file enames.hql in the following way.

!run enames.hql

Finally, list the contents of an external table enames.

select \* from enames;

If you move to "Hadoop window" and list the names of relational tables located in HDFS /user/hive/warehouse with

$HADOOP\_HOME/bin/hadoop fs -ls /user/hive/warehouse/

then only the relational tables hello and names are listed. This is because an external relational table enames is located in /user/bigdata/names.tbl. An external relational table is equivalent to a definition of the tables stored in Metastore and mapped on a file in HDFS /user/bigdata/names.tbl.

Move to "beeline window" and drop an external relational table enames.

drop table enames;

Then, move to "Hadoop window" and check if a file names.tbl still exists in /user/bigdata and it is not empty.

$HADOOP\_HOME/bin/hadoop fs -ls /user/bigdata

$HADOOP\_HOME/bin/hadoop fs -cat /user/bigdata/names.tbl

Deletion of an external relational table deletes its definition from Metastore only. Now, move to "beeline window" and drop an internal table names.

drop table names;

Finally, move to "Hadoop window" and check if a file names.tbl still exists in /user/hive/warehouse.

$HADOOP\_HOME/bin/hadoop fs -ls /user/hive/warehouse/

Deletion of an internal relational table deletes its definition in Metastore and a file in HDFS that implements the table.

**(10) How to create a database ?**

In this step you can use either Zeppelin interface or beeline command interface or SQL Developer to Hive Server 2. In Zeppelin interface use %hive paragraph .

Move to "beeline window" if you use command line interface. To create a new database tpchr process the following statement.

create database tpchr;

A database is created as a new folder tpchr.db in /user/hive/warehouse folder in HDFS. To verify location the database move to "Hadoop window" and process the following command.

$HADOOP\_HOME/bin/hadoop fs -ls /user/hive/warehouse

A database can be created in a location different from /user/hive/warehouse. For example, process the following statement in "beeline window".

create database other location '/user/hive';

A database other is created in /user/hive folder however no new folder is created. Process the following command in "Hadoop window".

$HADOOP\_HOME/bin/hadoop fs -ls /user/hive

To get more information about default, tpchr, and other databases process the following commands in "beeline window".

show databases;

describe database default;

describe database tpchr;

describe database other;

Move to SQL Developer and refresh with blue arrows Connections subwindow. To restore information about the databases leftclick at small plus in front of hive connection.

When connected beeline assumes that initially you are connected to default database.

The current database is a default database. It is possible to change a current database with use command. Process the following statement in "beeline window".

use other;

create table hello(message varchar(50));

Refresh Connections subwindow in SQL Developer and leftclick at small + in front of other database icon and later on at Tables item to show the tables created in other database.

Move to "Hadoop window" and process a command:

$HADOOP\_HOME/bin/hadoop fs -ls /user/hive

to find location of hello table in HDFS. Note, that it is possible to have many tables with the same names in different databases (like in MySQL).

**(11) How to access a database ?**

To access a database as your current database use a command use. For example, to make other database your current database process a statement use in "beeline window".

use other;

insert into hello values( 'Hello James !');

To access a relational table located in a database that is not your currrent database you have to prefix a table name with a database name and with . (dot). For example, change a current database in "beeline window" to default.

use deafult;

To access a relational table hello located in other database process the following statement.

select \* from other.hello;

**(12) How to drop a database ?**

To drop other database process a statement:

drop database other;

The system cannot drop a nonempty database. Drop a table hello first and then drop a database other.

drop table hello;

drop database other;

**(13) How to create and how to access a relational table with a column of type array ?**

It is possible in Hive to create relational tables with columns of type array, map, struct, named struct, and union.

From the relational database theory point of view such tables are not in 1NF and they are also called as nested tables, unnormalized tables or 0NF tables.

Make default database your current database. First, we create a table friend that contains information about friends.

create table friend(

name varchar(30),

friends array<string> );

Use describe command to verify the structures of the table. Next, we shall insert few rows into the table.

insert into friend select 'James',array('Kate','John');

insert into friend select 'John', array(NULL);

insert into friend select 'Kate', array();

insert into friend select 'Harry',array('James');

Use select statement to verify the contents of a relational tables friend. Note a difference between NULL and array[]). It is possible to bind different interpretations to NULL and empty array (array([])).

select \* from friend;

To select a particular element from an array we provide a number of an element in an array. For example, to list the first 3 friends of each person process the following select statement.

select name, friends[0], friends[1], friends[2]

from friend;

**(14) How to create and how to access a relational table with a column of type map ?**

Create a relational table workshop to keep information about the workshops, types of tools available at each workshop and total number of tools for each type.

create table workshop(

name varchar(50),

tools map<string,int> );

Use describe command to verify the structures of the table. Next, we shall insert few rows into the table.

describe workshop;

insert into workshop

select 'XYZ Ltd.',map('screwdriver',30,'hammer',1);

insert into workshop select 'Mitra10', map('hammer',1);

select \* from workshop;

We use a key value to select a particular element from a map. For example, to select the total number of hammers in each workshop process the following select statement.

select name, tools['hammer'] hammers

from workshop;

To select workshops that have at least one hammer process the following select statement.

select name, tools['hammer'] hammers

from workshop

where tools['hammer'] > 0;

**(15) How to create and how to access a relational table with a column of type struct ?**

Create a relational table employee to keep information about employees.

create table employee(

enumber decimal(7),

address struct<city:string,street:string,house:int,flat:int> );

Use describe command to verify the structures of the table. Next, we shall insert few rows into the table.

describe employee;

insert into employee

select 007, named\_struct('city','London',

'street','Victoria St.','house',7,'flat',77);

insert into employee

select 123, named\_struct('city','Dapto',

'street','Station St.','house',1,'flat',0);

select \* from employee;

To select a particular element from a structure we provide a field name. For example, to select the names of cities the employees live in we process the following statement.

select enumber, address.city city

from employee;

To find all employees living London we process the following statement.

select \*

from employee

where address.city = 'London';

**(16) How to create a new connection in SQL Developer ?**

Assume that we would like to create a new connection in SQL Developer to a new database. To create a new connection leftclick at Connections icon in Connection subwindow of SQL Developer window.

Next leftclick at large green + (plus) at the top of Connections subwindow. SQL Developer opens New/Select Database Conmnection window.

Type a name of a new connection into Connection Name field. A connection name is up to you.

A user name and password are also up to you.

Next, left click at Hive tab to make it active (initially Oracle tab is active).

Type localhost into Host name field.

Type 10000 into Port field.

Type a name of a database created earlier, e.g. tpchr into Database field.

Do not change the contents of Driver field.

When ready leftclick at Test button. When "Success" word is displayed in a left lower corner of New/Select Database Connection window leftclick at Connect button. SQLDeveloper creates a new connection in Connections subwindow and opens a new tab in the main window.

By choosing tabs in the main Window of SQL Developer you can use different connections to process HQL statements.

*End of Exercise 3*