

ASSIGNMENT NO: 3

Title: Write an application using HiveQL for flight information system which will include

- Creating, Dropping, and altering Database tables.
- Creating an external Hive table.
- Load table with data, insert new values and field in the table, Join tables with Hive
- Create index on Flight Information Table
- Find the average departure delay per day in 2008

Problem Statement: Study and write an application using HiveQL for flight information system.

Software and Hardware Requirement: PC with Ubuntu

Theory:

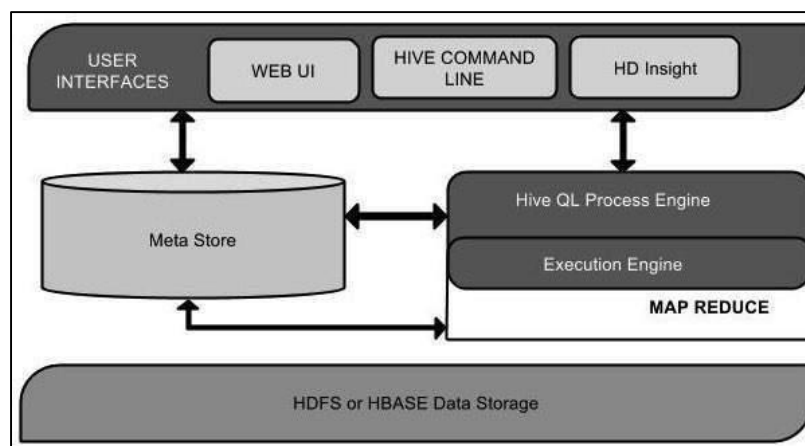
What is Hive?

Hive is a data warehouse infrastructure tool to process structured data in Hadoop. It resides on top of Hadoop to summarize Big Data, and makes querying and analyzing easy.

Initially Hive was developed by Facebook, later the Apache Software Foundation took it up and developed it further as an open source under the name Apache Hive.

Architecture of Hive:

The following component diagram depicts the architecture of Hive:



Hive chiefly consists of three core parts:

- **Hive Clients:** Hive offers a variety of drivers designed for communication with different applications. For example, Hive provides Thrift clients for Thrift-based applications. These clients and drivers then communicate with the Hive server, which falls under Hive services.

- **Hive Services:** Hive services perform client interactions with Hive. For example, if a client wants to perform a query, it must talk with Hive services.
- **Hive Storage and Computing:** Hive services such as file system, job client, and meta store then communicates with Hive storage and stores things like metadata table information and query results.

Hive's Features:

These are Hive's chief characteristics:

- Hive is designed for querying and managing only structured data stored in tables
- Hive is scalable, fast, and uses familiar concepts
- Schema gets stored in a database, while processed data goes into a Hadoop Distributed File System (HDFS)
- Tables and databases get created first; then data gets loaded into the proper tables
- Hive supports four file formats: ORC, SEQUENCEFILE, RCFILE (Record Columnar File), and TEXTFILE
- Hive uses an SQL-inspired language, sparing the user from dealing with the complexity of MapReduce programming. It makes learning more accessible by utilizing familiar concepts found in relational databases, such as columns, tables, rows, and schema, etc.

Limitations of Hive:

Of course, no resource is perfect, and Hive has some limitations. They are:

- Hive doesn't support OLTP. Hive supports Online Analytical Processing (OLAP), but not Online Transaction Processing (OLTP).
- It doesn't support subqueries.
- It has a high latency.
- Hive tables don't support delete or update operations.

- Hive provides a high-level SQL-like interface for users to interact with data stored in HDFS, enabling them to perform complex data analytics and processing tasks on large-scale datasets. It abstracts the underlying complexities of Hadoop and provides a familiar querying experience to users.

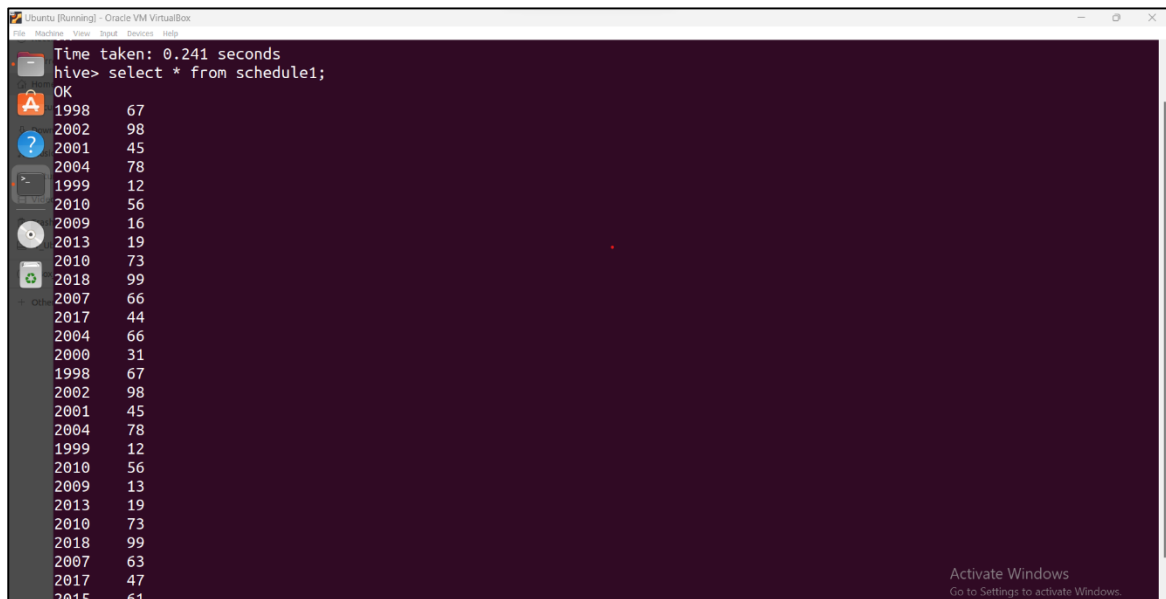
STEPS TO PERFORM:

HIVE COMMANDS

- 1) create database flightinfo;
- 2) show databases;
- 3) use flightinfo;
- 4) create table delay(year int, delay int)
 row format delimited
 fields terminated by ',';
- 5) desc delay;
- 6) load data local inpath '/home/user/airinfo' into table delay;
- 7) select * from delay;
- 8) exit from hive
- 9) hdfs dfs -mkdir /viman
- 10) hdfs dfs -put airinfo /viman
- 11) go to hive
- 12) show databases;
- 13) use flightinfo;
- 14) select * from delay;
- 15) load data local inpath
- 16) go to browser
- 17) load data inpath '/viman/airinfo' into table delay;
- 18) create table delay1(year int, delay int)
 row format delimited
 fields terminated by ',';
- 19) load data inpath '/viman/airinfo' into table delay;
- 20) select * from delay1;
- 21) go to browser
- 22) localhost:9870 (Just Type localhost in browser)
- 23) Then go to utilities and then go to log and check 'viman' file

OUTPUT:

1.Importing Table from local file to hive database:

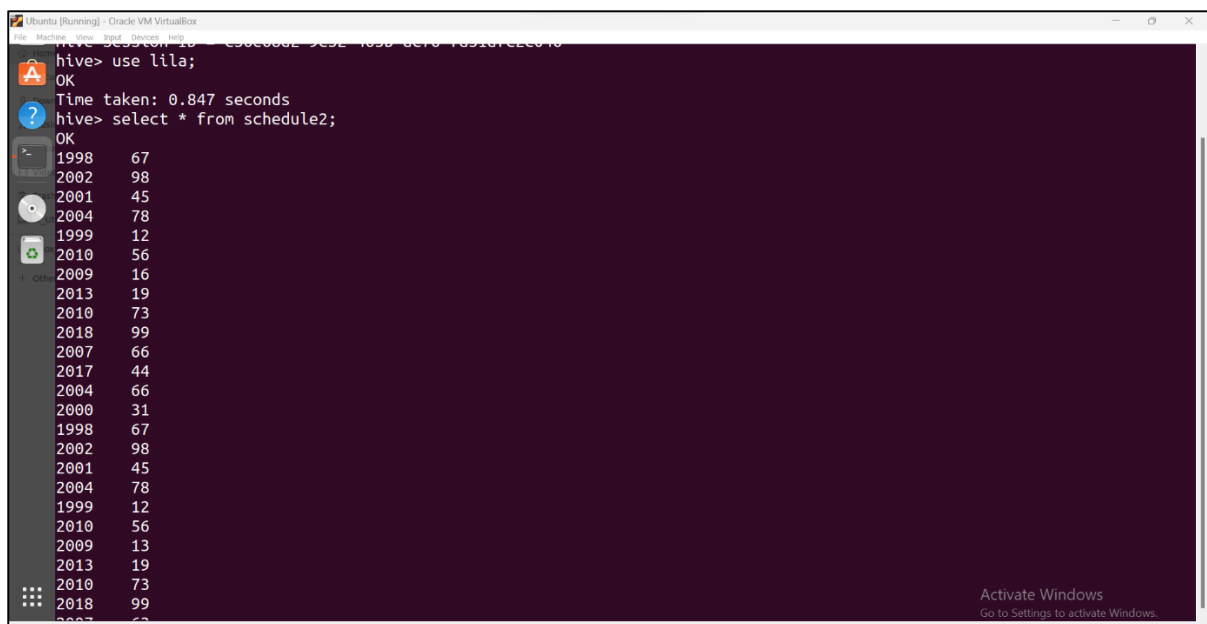


The screenshot shows a terminal window titled 'Ubuntu [Running] - Oracle VM VirtualBox'. The terminal displays the following output:

```
Time taken: 0.241 seconds
hive> select * from schedule1;
OK
1998      67
2002      98
2001      45
2004      78
1999      12
2010      56
2009      16
2013      19
2010      73
2018      99
2007      66
2017      44
2004      66
2000      31
1998      67
2002      98
2001      45
2004      78
1999      12
2010      56
2009      13
2013      19
2010      73
2018      99
2007      63
2017      47
2015      64
```

An 'Activate Windows' watermark is visible in the bottom right corner of the terminal window.

2.Importing data from local file to hdfs to Hive database:



The screenshot shows a terminal window titled 'Ubuntu [Running] - Oracle VM VirtualBox'. The terminal displays the following output:

```
hive> use lila;
OK
Time taken: 0.847 seconds
hive> select * from schedule2;
OK
1998      67
2002      98
2001      45
2004      78
1999      12
2010      56
2009      16
2013      19
2010      73
2018      99
2007      66
2017      44
2004      66
2000      31
1998      67
2002      98
2001      45
2004      78
1999      12
2010      56
2009      13
2013      19
2010      73
2018      99
2007      63
```

An 'Activate Windows' watermark is visible in the bottom right corner of the terminal window.

Average:

```
hive> select * from delay1 where year=2010
      > ;
OK
2010      13
2010      17
2010       5
Time taken: 0.435 seconds, Fetched: 3 row(s)
```

```
hive> select avg(delay) from delay1;
Query ID = ninad_20230518234851_722badbc-5ac8-40a5-b3df-609ff87a2a2c
Total jobs = 1
Launching Job 1 out of 1
Number of reduce tasks determined at compile time: 1
In order to change the average load for a reducer (in bytes):
  set hive.exec.reducers.bytes.per.reducer=<number>
In order to limit the maximum number of reducers:
  set hive.exec.reducers.max=<number>
In order to set a constant number of reducers:
  set mapreduce.job.reduces=<number>
Starting Job = job_1684432744922_0001, Tracking URL = http://ninad-VirtualBox:8088/proxy/application_1684432744922_0001/
Kill Command = /home/ninad/hadoop-3.3.5/bin/mapred job -kill job_1684432744922_0001
Hadoop job information for Stage-1: number of mappers: 1; number of reducers: 1
2023-05-18 23:49:11,317 Stage-1 map = 0%, reduce = 0%
2023-05-18 23:49:22,133 Stage-1 map = 100%, reduce = 0%, Cumulative CPU 7.35 sec
2023-05-18 23:49:28,371 Stage-1 map = 100%, reduce = 100%, Cumulative CPU 10.28 sec
MapReduce Total cumulative CPU time: 10 seconds 280 msec
Ended Job = job_1684432744922_0001
MapReduce Jobs Launched:
Stage-Stage-1: Map: 1 Reduce: 1 Cumulative CPU: 10.28 sec HDFS Read: 14334 HDFS Write: 106 SUCCESS
Total MapReduce CPU Time Spent: 10 seconds 280 msec
OK
12.125
Time taken: 39.257 seconds, Fetched: 1 row(s)
```

Conclusion:

Hence, Successfully completed application using HiveQL for flight information system.