7. Install and run Hive then use Hive to create, alter, and drop databases, tables,views, functions, and indexes.

THEORY: Hive defines a simple SQL-like query language to querying and managing large datasets called Hive-QL (HQL). It's easy to use if you're familiar with SQL Language. Hive allows programmers who are familiar with the language to write the custom MapReduce framework to perform more sophisticated analysis.

Uses of Hive:

- The Apache Hive distributed storage.
- Hive provides tools to enable easy data extract/transform/load (ETL)
- It provides the structure on a variety of data formats.
- By using Hive, we can access files stored in Hadoop Distributed File System (HDFS is used to querying and managing large datasets residing in) or in other data storage systems such as Apache HBase.

Limitations of Hive:

• Hive is not designed for Online transaction processing (OLTP), it is only used for the Online Analytical Processing.

• Hive supports overwriting or apprehending data, but not updates and deletes.

• In Hive, sub queries are not supported.

Why Hive is used inspite of Pig?

The following are the reasons why Hive is used in spite of Pig's availability:

- Hive-QL is a declarative language line SQL, PigLatin is a data flow language.
- Pig: a data-flow language and environment for exploring very large datasets.
- Hive: a distributed data warehouse.

Components of Hive:

Metastore : Hive stores the schema of the Hive tables in a Hive Metastore. Metastore is used to hold all the information about the tables and partitions that are in the warehouse. By default, the metastore is run in the same process as the Hive service and the default Metastore is DerBy Database.

SerDe : Serializer, Deserializer gives instructions to hive on how to process a record.

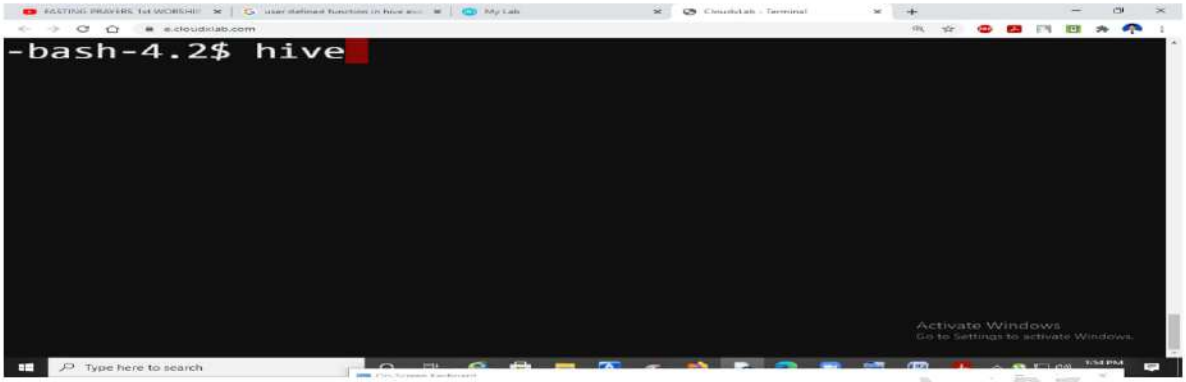
Hive Commands :

Data Definition Language (DDL)

DDL statements are used to build and modify the tables and other objects in the database.

Example : CREATE, DROP, TRUNCATE, ALTER, SHOW, DESCRIBE Statements.

Go to Hive shell by giving the hive and enter the command 'create database<data base name>' to create the new database in the Hive.



Create Database-Below is the syntax for creation of Databases in Hive.

```
CREATE (DATABASE|SCHEMA) [IF NOT EXISTS] database_name
  [COMMENT database_comment]
  [LOCATION hdfs_path]
  [WITH DBPROPERTIES (property_name=property_value, ...)];
```

Description of Arguments

IF NOT EXISTS – This optional but recommended to use, so that, if a database with same name already exists, then it will not try to create it again and will not show any error message.

COMMENT – It is also optional. It can be used for providing short description or any comment to database

LOCATION – It is also optional. By default all the hive databases will be created under default warehouse directory(set by the property hive.metastore.warehouse.dir) as user/hive/warehouse/database_name.db . But if we want to specify our own location then this option can be specified.

DBPROPERTIES – Optional but used to specify any properties of database in the form of (key, value) separated pairs.

Example

Below is the example database creation HiveQL, which covers all the above options.

```
CREATE DATABASE IF NOT EXISTS flightinfo
        COMMENT "Test Database created for flights"
        WITH DBPROPERTIES(
                'Date' = '2021-2-05',
                'Creator' = 'bhaska K',
                'Email' = 'bhskarjnvkp@gmail.com'
                );
```

```
hive> CREATE DATABASE IF NOT EXISTS flightinfo
    > COMMENT "Test Database created for flights"
    > WITH DBPROPERTIES('Date' = '2021-2-05','Creator' = 'bhaska K','Email' = 'bhskarjnvkp@gmail.com');
OK
Time taken: 0.23 seconds
```

Show database will show all database

```
hive> show databases;
OK
```

Hadoop Hive Alter Database Command-Hadoop Hive alter database is a statement used to change the properties of a databases in Hive. You can add or remove the database comments, properties etc by using alter database statements.

Describing databases: By default, the DESCRIBE output for a database includes the location and the comment, which can be set by the LOCATION and COMMENT clauses on the CREATE DATABASE statement. The additional information displayed by the FORMATTED or EXTENDED keyword includes the HDFS user ID that is considered the owner of the database, and any optional database properties. The properties could be specified by the WITH DBPROPERTIES clause if the database is created using a Hive CREATE DATABASE statement

```
ive> describe database flightinfo;
K
lightinfo               hdfs://cxln1.c.thelab-240901.internal:8020/apps/hive/warehouse/flightinfo.db    bhaskarjnvkp8113    USER
ime taken: 0.122 seconds, Fetched: 1 row(s)
ive>
```


ALTER DATABASE- Hadoop Hive alter database is a statement used to change the properties of a databases in Hive. You can add or remove the database comments, properties etc by using alter database statements.

The syntax for Hive alter database statement is as follows:

ALTER (DATABASE|SCHEMA) database_name SET DBPROPERTIES (property_name=property_value, ...);
ALTER (DATABASE|SCHEMA) database_name SET OWNER [USER|ROLE] user_or_role;
ALTER (DATABASE|SCHEMA) database_name SET LOCATION hdfs_path;

Below is the example of alter database in Hive:

```
hive> ALTER DATABASE flightinfo SET DBPROPERTIES ('Date' = '2021-02-05');
OK
Time taken: 0.232 seconds
hive>
```

Hive Drop Database -Hadoop Hive drop database is a statement used to drop the databases in Hive.

The syntax for Hive drop database statement is as follows:

DROP (DATABASE|SCHEMA) [IF EXISTS] database_name [RESTRICT|CASCADE];

The default behaviour is RESTRICT, where DROP DATABASE will fail if the database is not empty. To drop the tables in the database as well, use DROP DATABASE ... with CASCADE option.

Below is the example of drop database in Hive:

```
hive> drop database flightinfo;
OK
Time taken: 0.242 seconds
hive>
```

TYPES OF TABLES -Hive knows two different types of tables: Internal table and the External table. The Internal table is also known as the managed table. We can identify the internal or External tables using the DESCRIBE FORMATTED table_name statement in the Hive, which will display either MANAGED_TABLE or EXTERNAL_TABLE depending on the table type.

I. Hive Internal Table-Hive owns the data for the internal tables.It is the default table in Hive. When the user creates a table in Hive without specifying it as external, then by default, an internal table gets created in a specific location in HDFS.By default, an internal table will be created in a folder path similar to /user/hive/warehouse directory of HDFS. We can override the default location by the location property during table creation. If we drop the managed table or partition, the table data and the metadata associated with that table will be deleted from the HDFS.

II. Hive External Table-Hive does not manage the data of the External table.We create an external table for external use as when we want to use the data outside the Hive.External tables are stored outside the warehouse directory. They can access data stored in sources such as remote HDFS locations or Azure Storage Volumes.Whenever we drop the external table, then only the metadata associated with the table will get deleted, the table data remains untouched by Hive. We can create the external table by specifying the EXTERNAL keyword in the Hive create table statement.

CREATE TABLE Statement-Creates a new table and specifies its characteristics. While creating a table, you optionally specify aspects such as:

- Whether the table is internal or external.
- The columns and associated data types.
- The columns used for physically partitioning the data.
- The file format for data files.
- The HDFS directory where the data files are located.

Syntax:The general syntax for creating a table and specifying its columns is as follows:

```
CREATE [EXTERNAL] TABLE [IF NOT EXISTS] [db_name.]table_name
(col_name data_type [COMMENT 'col_comment'], ...)
[PARTITIONED BY (col_name data_type [COMMENT 'col_comment'], ...)]
[COMMENT 'table_comment']
[WITH SERDEPROPERTIES ('key1'='value1', 'key2'='value2', ...)]
[
```

[ROW FORMAT row_format] [STORED AS file_format]
]
[LOCATION 'hdfs_path']
[TBLPROPERTIES ('key1'='value1', 'key2'='value2', ...)]
[CACHED IN 'pool_name' [WITH REPLICATION = integer] | UNCACHED]

CREATE TABLE IF NOT EXISTS FlightInfo2006 (Month TINYINT, DayofMonth TINYINT,DayOfWeek TINYINT, CRSDepTime SMALLINT, CRSArrTime SMALLINT, FlightNum STRING, TailNum STRING,ActualElapsedTime SMALLINT, ArrDelay SMALLINT, DepDelay SMALLINT,Origin STRING, Dest STRING, Cancelled SMALLINT,CarrierDelay SMALLINT, WeatherDelay SMALLINT,NASDelay SMALLINT, SecurityDelay SMALLINT,LateAircraftDelay SMALLINT)

COMMENT 'Flight InfoTable'
PARTITIONED BY(Year SMALLINT)
ROW FORMAT DELIMITED
FIELDS TERMINATED BY ','
LINES TERMINATED BY '\n'
STORED AS TEXTFILE
TBLPROPERTIES ('creator'='Bhaskar', 'created_at'='Fri Feb 16:58:00 EDT 2021');

```
ive> CREATE TABLE IF NOT EXISTS FlightInfo2006 ( Month TINYINT, DayofMonth TINYINT,DayOfWeek TINYINT, CRSDepTime SMALLINT, CRSArrTime SMAL
INT, FlightNum STRING, TailNum STRING,ActualElapsedTime SMALLINT, ArrDelay SMALLINT, DepDelay SMALLINT,Origin STRING, Dest STRING, Cancell
d SMALLINT,CarrierDelay SMALLINT, WeatherDelay SMALLINT,NASDelay SMALLINT, SecurityDelay SMALLINT,LateAircraftDelay SMALLINT)
  > COMMENT 'Flight InfoTable'
  > PARTITIONED BY(Year SMALLINT )
  > ROW FORMAT DELIMITED
  > FIELDS TERMINATED BY ','
  > LINES TERMINATED BY '\n'
  > STORED AS TEXTFILE
  > TBLPROPERTIES ('creator'='Bhaskar', 'created_at'='Fri Feb 16:58:00 EDT 2021');
K
ime taken: 0.188 seconds
ive>
```


CREATE TABLE IF NOT EXISTS FlightInfo2008 (Year SMALLINT, Month TINYINT, DayofMonth TINYINT,DayOfWeek TINYINT, CRSDepTime SMALLINT, CRSArrTime SMALLINT, FlightNum STRING, TailNum STRING,ActualElapsedTime SMALLINT, ArrDelay SMALLINT, DepDelay SMALLINT,Origin STRING, Dest STRING, Cancelled SMALLINT,CarrierDelay SMALLINT, WeatherDelay SMALLINT,NASDelay SMALLINT, SecurityDelay SMALLINT,LateAircraftDelay SMALLINT)

COMMENT 'Flight InfoTable'
ROW FORMAT DELIMITED
FIELDS TERMINATED BY ','
LINES TERMINATED BY '\n'
STORED AS TEXTFILE

TBLPROPERTIES ('creator'='Bhaskar', 'created_at'='Fri Feb 16:58:00 EDT 2021');

```
hive> CREATE TABLE IF NOT EXISTS FlightInfo2008 (Year SMALLINT, Month TINYINT, DayofMonth TINYINT,DayOfWeek TINYINT, CRSDepTime SMALLINT, C
RSArrTime SMALLINT, FlightNum STRING, TailNum STRING,ActualElapsedTime SMALLINT, ArrDelay SMALLINT, DepDelay SMALLINT,Origin STRING, Dest S
TRING, Cancelled SMALLINT,CarrierDelay SMALLINT, WeatherDelay SMALLINT,NASDelay SMALLINT, SecurityDelay SMALLINT,LateAircraftDelay SMALLINT
)COMMENT 'Flight InfoTable'
    > ROW FORMAT DELIMITED
    > FIELDS TERMINATED BY ','
    > LINES TERMINATED BY '\n'
    > STORED AS TEXTFILE
    > TBLPROPERTIES ('creator'='Bhaskar', 'created_at'='Fri Feb 16:58:00 EDT 2021');
OK
Time taken: 0.216 seconds
hive> show tables;
OK
flightinfo2007
flightinfo2008
Time taken: 0.224 seconds, Fetched: 2 row(s)
hive>
```


'CREATE EXTERNAL' TABLE : The create external keyword is used to create a table and provides a location where the table will create, so that Hive does not use a default location for this table. An EXTERNAL table points to any HDFS location for its storage, rather than default storage.

CREATE EXTERNAL TABLE IF NOT EXISTS FlightInfo2009 (Year SMALLINT, Month TINYINT, DayofMonth TINYINT,DayOfWeek TINYINT, CRSDepTime SMALLINT, CRSArrTime SMALLINT, FlightNum STRING, TailNum STRING,ActualElapsedTime SMALLINT, ArrDelay SMALLINT, DepDelay SMALLINT,Origin STRING, Dest STRING, Cancelled SMALLINT,CarrierDelay SMALLINT, WeatherDelay SMALLINT,NASDelay SMALLINT, SecurityDelay SMALLINT,LateAircraftDelay SMALLINT)

COMMENT 'Flight InfoTable'
ROW FORMAT DELIMITED
FIELDS TERMINATED BY ','
LINES TERMINATED BY '\n'
STORED AS TEXTFILE
TBLPROPERTIES ('creator'='Bhaskar', 'created_at'='Fri Feb 16:58:00 EDT 2021');

```
hive> CREATE EXTERNAL TABLE IF NOT EXISTS FlightInfo2007 (Year SMALLINT, Month TINYINT, DayofMonth TINYINT,DayOfWeek TINYINT, CRSDepTime SM
ALLINT, CRSArrTime SMALLINT, FlightNum STRING, TailNum STRING,ActualElapsedTime SMALLINT, ArrDelay SMALLINT, DepDelay SMALLINT,Origin STRIN
G, Dest STRING, Cancelled SMALLINT,CarrierDelay SMALLINT, WeatherDelay SMALLINT,NASDelay SMALLINT, SecurityDelay SMALLINT,LateAircraftDelay
 SMALLINT)COMMENT 'Flight InfoTable'
    > ROW FORMAT DELIMITED
    > FIELDS TERMINATED BY ','
    > LINES TERMINATED BY '\n'
    > STORED AS TEXTFILE
    > TBLPROPERTIES ('creator'='Bhaskar', 'created_at'='Fri Feb 16:58:00 EDT 2021');
OK
Time taken: 0.103 seconds
hive> show tables;
OK
flightinfo2007
```

The Load operation is used to move the data into corresponding Hive table. If the keyword local is specified, then in the load command will give the local file system path. If the keyword local is not specified we have to use the HDFS path of the file.

LOAD data <LOCAL> inpath <file path> into table [tablename]

LOAD DATA LOCAL INPATH ' /home/bhaskarjnvkp8113/FLIGHT2008 / flightdata2008-1.csv' INTO TABLE FlightInfo2007;

```
hive> LOAD DATA LOCAL  INPATH '/home/bhaskarjnvkp8113/FLIGHT2008/flightdata2008-1.csv' INTO TABLE FlightInfo2007;
Loading data to table flightinfo.flightinfo2007
Table flightinfo.flightinfo2007 stats: [numFiles=1, numRows=0, totalSize=68600682, rawDataSize=0]
OK
Time taken: 1.531 seconds
```

LOAD DATA With PARTITION

LOAD DATA [LOCAL] INPATH 'path to file' [OVERWRITE] INTO TABLE 'table name' [PARTITION partition column1 = value1, partition column2 = value2,...]

LOAD DATA LOCAL INPATH ' /home/bhaskarjnvkp8113/FLIGHT2008 / flightdata2008-1.csv' INTO TABLE FlightInfo2005 PARTITION (month = 2);

```
hive> LOAD DATA LOCAL  INPATH '/home/bhaskarjnvkp8113/FLIGHT2008/flightdata2008-1.csv' INTO TABLE FlightInfo2005 PARTITION (month = 2);
Loading data to table flightinfo.flightinfo2005 partition (month=2)
Partition flightinfo.flightinfo2005{month=2} stats: [numFiles=1, numRows=0, totalSize=68600682, rawDataSize=0]
OK
Time taken: 1.867 seconds
hive> show partitions flightinfo2005;
OK
month=1
month=2
Time taken: 0.485 seconds, Fetched: 2 row(s)
```


LOAD DATA INPATH ' /user/bhaskarjnvkp8113/ flight' INTO TABLE FlightInfo2008;

```
hive> LOAD DATA  INPATH '/user/bhaskarjnvkp8113/flight' INTO TABLE FlightInfo2008;
Loading data to table flightinfo.flightinfo2008
Table flightinfo.flightinfo2008 stats: [numFiles=1, numRows=0, totalSize=68600682, rawDataSize=0]
OK
Time taken: 0.729 seconds
```

ALTER TABLE -Apache Hive ALTER TABLE command to change the structure of an existing table. You can add, modify existing columns in Hive tables.Below are the most common uses of the ALTER TABLE command: You can rename table and column of existing Hive tables.

- You can add new column to the table.
- Rename Hive table column.
- Add or drop table partition.
- Add Hadoop archive option to Hive table.

Below is syntax of commonly used ALTER TABLE commands:

```
ALTER TABLE [old_db_name.]old_table_name RENAME TO [new_db_name.]new_table_name;
ALTER TABLE name ADD COLUMNS (col_spec[, col_spec ...]);
ALTER TABLE name CHANGE column_name new_name new_type;
ALTER TABLE name REPLACE COLUMNS (col_spec[, col_spec ...]);
```

ALTER TABLE FlightInfo2005 ADD PARTITION (month = 4);

```
ive> ALTER TABLE  FlightInfo2005  ADD PARTITION (month = 4);
K
ime taken: 0.389 seconds
ive> show partitions flightinfo2005;
K
onth=1
onth=2
onth=4
ime taken: 0.479 seconds, Fetched: 3 row(s)
```

CREATING A VIEW-Views are generated based on user requirements. You can save any result set data as a view. The usage of view in Hive is same as that of the view in SQL. It is a standard RDBMS concept. We can execute all DML operations on a view.You can create a view at the time of executing a SELECT statement. The syntax is as follows:

CREATE VIEW [IF NOT EXISTS] view_name [(column_name [COMMENT column_comment], ...)]
[COMMENT table_comment]
AS SELECT ...

```
hive> CREATE VIEW avgdepdelay AS  SELECT DayOfWeek, AVG(DepDelay) FROM FlightInfo2008 GROUP BY DayOfWeek;
OK
Time taken: 0.503 seconds
hive>
```

You can see the contents of a view by
HIVE>SELECT * FROM avgdepdelay;

```
2021-02-05 15:00:54,747 Stage-1 map = 0%,  reduce = 0%
2021-02-05 15:01:08,449 Stage-1 map = 100%,  reduce = 0%, Cumulative CPU 6.82 sec
2021-02-05 15:01:16,815 Stage-1 map = 100%,  reduce = 50%, Cumulative CPU 10.11 sec
2021-02-05 15:01:22,009 Stage-1 map = 100%,  reduce = 100%, Cumulative CPU 11.83 sec
MapReduce Total cumulative CPU time: 11 seconds 830 msec
Ended Job = job_1607429201608_6509
MapReduce Jobs Launched:
Stage-Stage-1: Map: 1  Reduce: 2   Cumulative CPU: 11.83 sec   HDFS Read: 68619383 HDFS Write: 147 SUCCESS
Total MapReduce CPU Time Spent: 11 seconds 830 msec
OK
NULL    NULL
2       47.10423963007552
4       43.64284763552699
6       47.43961490461758
1       47.987120569227756
3       45.35003814773829
5       48.85494215602759
7       50.94043391465335
Time taken: 39.927 seconds, Fetched: 8 row(s)
hive> DROP VIEW avgdepdelay;
OK
Time taken: 0.208 seconds
```

Dropping a View Use the following syntax to drop a view:
DROP VIEW avgdepdelay;

```
hive> DROP VIEW avgdepdelay;
OK
Time taken: 0.208 seconds
hive> SELECT * FROM avgdepdelay;
FAILED: SemanticException [Error 10001]: Line 1:14 Table not found 'avgdepdelay'
```