3. Write an application using HiveQL for flight information system which will include
      a. Creating, Dropping, and altering Database tables.
      b. Creating an external Hive table.
      c. Load table with data, insert new values and field in the table, Join tables with Hive
      d. Create index on Flight Information Table
      e. Find the average departure delay per day in 2008.

SOLUTION

cd /usr/local/hive/bin

./hive

```
(A) $ $HIVE_HOME/bin hive --service cli
(B) hive> set hive.cli.print.current.db=true;
(C) hive (default)> CREATE DATABASE ourfirstdatabase; OK
Time taken: 3.756 seconds
(D) hive (default)> USE ourfirstdatabase; OK
Time taken: 0.039 seconds
(E) hive (ourfirstdatabase)> CREATE TABLE our_first_table
(
> FirstName STRING,
> LastName STRING,
> EmployeeId INT);
OK
Time taken: 0.043 seconds
hive (ourfirstdatabase)> quit;
```

1) Creating, Dropping, and altering Database tables.

hduser@ubuntu:~$ start-all.sh

hduser@ubuntu:~$ hive

hive> set hive.cli.print.current.db=true;

hive (default)> create database ourfirstdatabase;

OK

Time taken: 1.955 seconds

hive (default)> use ourfirstdatabase;

OK

Time taken: 0.048 seconds

hive (ourfirstdatabase)> create table our_first_table(firstname string,lastname string,employeeid int);

OK

Time taken: 0.873 seconds

hive> show tables;

OK

our_first_table

Time taken: 0.114 seconds, Fetched: 1 row(s)

//set db properties use alter command

hive> ALTER DATABASE ourfirstdatabase SET DBPROPERTIES

   > ('creator'=Swapnali Ware',

   > 'created_for'='Learning Hive DDL');

OK

Time taken: 0.315 seconds

hive> DESCRIBE DATABASE EXTENDED ourfirstdatabase;

OK

ourfirstdatabase   hdfs://localhost:54310/user/hive/warehouse/ourfirstdatabase.db hduser USER {created_for=Learning Hive DDL, creator=Swapnali Ware}

Time taken: 0.091 seconds, Fetched: 1 row(s)

hive> DROP DATABASE ourfirstdatabase CASCADE;

OK

Time taken: 1.691 seconds

2) `Creating an external Hive table.`

In Hive terminology, external tables are tables not managed with Hive. Their purpose is to facilitate importing of data from an external file into the metastore.

The external table data is stored externally, while Hive metastore only contains the metadata schema. Consequently, dropping of an external table does not affect the data.

In this tutorial, you will learn how to create, query, and drop an external table in Hive.

Creating an External Table in Hive – Syntax Explained

When creating an external table in Hive, you need to provide the following information:

- **Name of the table** – The `create external table` command creates the table. If a table of the same name already exists in the system, this will cause an error. To avoid this, add `if not exists` to the statement. Table names are case insensitive.
- **Column names and types** – Just like table names, column names are case insensitive. Column types are values such as `int`, `char`, `string`, etc.
- **Row format** – Rows use native or custom SerDe (Serializer/Deserializer) formats. Native SerDe will be used if the row format is not defined, or if it is specified as delimited.
- **Field termination character** – This is a `char` type character which separates table values in a row.
- **Storage format** – You can specify storage formats such as textfile, sequencefile, jsonfile, etc.
- **Location** – This is the HDFS directory location of the file containing the table data.
- The correct syntax for providing this information to Hive is:

- create external table if not exists [external-table-name] (
- [column1-name] [column1-type], [column2-name] [column2-type], …)
- comment '[comment]'
- row format [format-type]
- fields terminated by '[termination-character]'
- stored as [storage-type]
- location '[location]';

3) Download data set of 2007 & 2008 from
http://stat-computing.org/dataexpo/2009/the-data.html

Variable descriptions

	Name	Description
1	Year	1987-2008
2	Month	1-12
3	DayofMonth	1-31
4	DayOfWeek	1 (Monday) - 7 (Sunday)
5	DepTime	actual departure time (local, hhmm)
6	CRSDepTime	scheduled departure time (local, hhmm)
7	ArrTime	actual arrival time (local, hhmm)
8	CRSArrTime	scheduled arrival time (local, hhmm)
9	UniqueCarrier	unique carrier code
10	FlightNum	flight number
11	TailNum	plane tail number
12	ActualElapsedTime	in minutes
13	CRSElapsedTime	in minutes
14	AirTime	in minutes
15	ArrDelay	arrival delay, in minutes
16	DepDelay	departure delay, in minutes
17	Origin	origin IATA airport code
18	Dest	destination IATA airport code
19	Distance	in miles

20	TaxiIn	taxi in time, in minutes
21	TaxiOut	taxi out time in minutes
22	Cancelled	was the flight cancelled?
23	CancellationCode	reason for cancellation (A = carrier, B = weather, C = NAS, D = security)
24	Diverted	1 = yes, 0 = no
25	CarrierDelay	in minutes
26	WeatherDelay	in minutes
27	NASDelay	in minutes
28	SecurityDelay	in minutes
29	LateAircraftDelay	in minutes

Integer type data can be specified using integral data types, INT. When the data range exceeds the range of INT, you need to use BIGINT and if the data range is smaller than the INT, you use SMALLINT. TINYINT is smaller than SMALLINT.

Int 2 bytes storage size -32,768 to 32,767 or 4 bytes storage size -2,147,483,648 to 2,147,483,647

Unsigned Int 0 to 65,535 / 0 to 4,294,967,295

Big int 8 bytes storage -2^63(-9,223,372,036,854,775,808) to 2^63-1(9,223,372,036,854,775,807)

smallint 2 bytes storage -2^15(-32,768) to 2^15-1 32,767)

tinyint 0 to 255

CREATE TABLE IF NOT EXISTS FlightInfo2007

  (

Year SMALLINT, Month TINYINT, DayofMonth TINYINT,

DayOfWeek TINYINT,

 DepTime SMALLINT, CRSDepTime SMALLINT, ArrTime SMALLINT,CRSArrTime SMALLINT,

   UniqueCarrier STRING, FlightNum STRING, TailNum STRING,

ActualElapsedTime SMALLINT, CRSElapsedTime SMALLINT,

AirTime SMALLINT, ArrDelay SMALLINT, DepDelay SMALLINT,

Origin STRING, Dest STRING,Distance INT,

TaxiIn SMALLINT, TaxiOut SMALLINT, Cancelled SMALLINT,

CancellationCode STRING, Diverted SMALLINT,

CarrierDelay SMALLINT, WeatherDelay SMALLINT,

NASDelay SMALLINT, SecurityDelay SMALLINT,

LateAircraftDelay

SMALLINT)

COMMENT 'Flight InfoTable'

ROW FORMAT DELIMITED

FIELDS TERMINATED BY ','

STORED AS TEXTFILE

TBLPROPERTIES ('creator'='PSB ', 'created_at'='Tues Dec 5 3:00:00 EDT 2017');

OK

> LINES TERMINATED BY '\n'

Time taken: 0.292 seconds

hive> load data local inpath '/home/student/Desktop/2007.csv' into table FlightInfo2007;

hive> CREATE TABLE IF NOT EXISTS FlightInfo2008 LIKE FlightInfo2007;

hive> load data local inpath '/home/hduser/Desktop/2008.csv' into table FlightInfo2008;

hive> CREATE TABLE IF NOT EXISTS myFlightInfo (

Year SMALLINT, DontQueryMonth TINYINT, DayofMonth

TINYINT, DayOfWeek TINYINT, DepTime SMALLINT, ArrTime SMALLINT,

UniqueCarrier STRING, FlightNum STRING,

AirTime SMALLINT, ArrDelay SMALLINT, DepDelay SMALLINT,

Origin STRING, Dest STRING, Cancelled SMALLINT,

CancellationCode STRING)

COMMENT 'Flight InfoTable'

PARTITIONED BY(Month TINYINT)

ROW FORMAT DELIMITED

FIELDS TERMINATED BY ','

LINES TERMINATED BY '\n'

STORED AS RCFILE TBLPROPERTIES ('creator'='swapnali ware',

'created_at'='Mon sep 2 14:24:19 EDT 2017');

OK

Time taken: 1.697 seconds

hive> INSERT OVERWRITE TABLE myflightinfo PARTITION (Month=1)

> SELECT Year, Month, DayofMonth, DayOfWeek, DepTime,

> ArrTime, UniqueCarrier,FlightNum, AirTime, ArrDelay, DepDelay, Origin,

> Dest, Cancelled,CancellationCode FROM FlightInfo2008 WHERE Month=1;

hive> FROM FlightInfo2008 INSERT INTO TABLE myflightinfo

> PARTITION (Month=2) SELECT Year, Month, DayofMonth, DayOfWeek, DepTime,

> ArrTime, UniqueCarrier, FlightNum,

> AirTime, ArrDelay, DepDelay, Origin, Dest, Cancelled,

> CancellationCode WHERE Month=2

> INSERT INTO TABLE myflightinfo

> PARTITION (Month=12)

> SELECT Year, Month, DayofMonth, DayOfWeek, DepTime,

> ArrTime, UniqueCarrier, FlightNum,

> AirTime, ArrDelay, DepDelay, Origin, Dest, Cancelled,

> CancellationCode WHERE Month=12;

hive> SHOW PARTITIONS myflightinfo;

OK

month=1

month=12

month=2

Time taken: 0.344 seconds, Fetched: 3 row(s)

Load table with data, insert new values and field in the table, Join tables with Hive

hive> CREATE TABLE myflightinfo2007 AS

  > SELECT Year, Month, DepTime, ArrTime, FlightNum,

  > Origin, Dest FROM FlightInfo2007

  > WHERE (Month = 7 AND DayofMonth = 3) AND

  > (Origin='JFK' AND Dest='ORD');

hive>SELECT * FROM myFlightInfo2007;

OK

Year	Month	DepTime	ArrTime	FlightNum	Origin	Dest
2007	7	700	834	5447	JFK	ORD
2007	7	1633	1812	5469	JFK	ORD
2007	7	1905	2100	5492	JFK	ORD
2007	7	1453	1624	4133	JFK	ORD
2007	7	1810	1956	4392	JFK	ORD
2007	7	643	759	903	JFK	ORD
2007	7	939	1108	907	JFK	ORD
2007	7	1313	1436	915	JFK	ORD

2007	7	1617	1755	917	JFK	ORD
2007	7	2002	2139	919	JFK	ORD

Time taken: 1.219 seconds, Fetched: 10 row(s)

hive> CREATE TABLE myFlightInfo2008 AS

   > SELECT Year, Month, DepTime, ArrTime, FlightNum,

   > Origin, Dest FROM FlightInfo2008

   > WHERE (Month = 7 AND DayofMonth = 3) AND

   > (Origin='JFK' AND Dest='ORD');

hive> SELECT * FROM myFlightInfo2008;

OK

2008	7	930	1103	5199	JFK	ORD
2008	7	705	849	5687	JFK	ORD
2008	7	1645	1914	5469	JFK	ORD
2008	7	1345	1514	4392	JFK	ORD
2008	7	1718	1907	1217	JFK	ORD
2008	7	757	929	1323	JFK	ORD
2008	7	928	1057	907	JFK	ORD
2008	7	1358	1532	915	JFK	ORD
2008	7	1646	1846	917	JFK	ORD
2008	7	2129	2341	919	JFK	ORD

Time taken: 0.424 seconds, Fetched: 10 row(s)

JOIN

Hive>SELECT m8.Year, m8.Month, m8.FlightNum, m8.Origin, m8.Dest, m7.Year, m7.Month, m7.FlightNum, m7.Origin, m7.Dest FROM myFlightinfo2008 m8 JOIN myFlightinfo2007 m7 ON m8.FlightNum=m7.FlightNum;

2008	7	5469	JFK	ORD	2007	7	5469	JFK	ORD
2008	7	4392	JFK	ORD	2007	7	4392	JFK	ORD

2008	7	907	JFK	ORD	2007	7	907	JFK	ORD
2008	7	915	JFK	ORD	2007	7	915	JFK	ORD
2008	7	917	JFK	ORD	2007	7	917	JFK	ORD
2008	7	919	JFK	ORD	2007	7	919	JFK	ORD

hive> SELECT m8.FlightNum,m8.Origin,m8.Dest,m7.FlightNum,m7.Origin,m7.Dest FROM myFlightinfo2008 m8 FULL OUTER JOIN myFlightinfo2007 m7 ON m8.FlightNum=m7.FlightNum;

1217	JFK	ORD	NULL	NULL	NULL
1323	JFK	ORD	NULL	NULL	NULL
NULL	NULL	NULL	4133	JFK	ORD
4392	JFK	ORD	4392	JFK	ORD
5199	JFK	ORD	NULL	NULL	NULL
NULL	NULL	NULL	5447	JFK	ORD
5469	JFK	ORD	5469	JFK	ORD
NULL	NULL	NULL	5492	JFK	ORD
5687	JFK	ORD	NULL	NULL	NULL
NULL	NULL	NULL	903	JFK	ORD
907	JFK	ORD	907	JFK	ORD
915	JFK	ORD	915	JFK	ORD
917	JFK	ORD	917	JFK	ORD
919	JFK	ORD	919	JFK	ORD

Time taken: 10.33 seconds, Fetched: 14 row(s)

hive>SELECT m8.Year,m8.Month,m8.FlightNum,m8.Origin,m8.Dest,m7.Year,m7.Month,m7.FlightNum,m7.Origin,m7.Dest FROM myFlightinfo2008 m8 LEFT OUTER JOIN myFlightinfo2007 m7 ON m8.FlightNum=m7.FlightNum;

2008	7	5199	JFK	ORD	NULL	NULL	NULL	NULL	NULL
2008	7	5687	JFK	ORD	NULL	NULL	NULL	NULL	NULL
2008	7	5469	JFK	ORD	2007	7	5469	JFK	ORD

2008	7	4392	JFK	ORD	2007	7	4392	JFK	ORD
2008	7	1217	JFK	ORD	NULL	NULL	NULL	NULL	NULL
2008	7	1323	JFK	ORD	NULL	NULL	NULL	NULL	NULL
2008	7	907	JFK	ORD	2007	7	907	JFK	ORD
2008	7	915	JFK	ORD	2007	7	915	JFK	ORD
2008	7	917	JFK	ORD	2007	7	917	JFK	ORD
2008	7	919	JFK	ORD	2007	7	919	JFK	ORD

4. Create index on Flight Information Table

hive> CREATE INDEX f08_index ON TABLE flightinfo2008 (Origin) AS

  > 'COMPACT' WITH DEFERRED REBUILD;

OK

Time taken: 1.124 seconds

hive> ALTER INDEX f08_index ON flightinfo2008 REBUILD;

hive>SHOW INDEXES ON FlightInfo2008;

OK

f08_index   flightinfo2008   origin   default__flightinfo2008_f08_index__ compact

Time taken: 2.549 seconds, Fetched: 1 row(s)

hive> SELECT Origin, COUNT(1) FROM

  > flightinfo2008 WHERE Origin = 'SYR' GROUP BY Origin;

hive> DESCRIBE default__flightinfo2008_f08_index__;

OK

origin   string

_bucketname   string

_offsets   array<bigint>

Time taken: 0.927 seconds, Fetched: 3 row(s)

hive> SELECT Origin, SIZE(`_offsets`)

   > FROM default__flightinfo2008_f08_index__ WHERE origin='SYR';

OK

SYR   12032

Time taken: 0.705 seconds, Fetched: 1 row(s)

hive> CREATE VIEW avgdepdelay AS

   > SELECT DayOfWeek, AVG(DepDelay) FROM

   > FlightInfo2008 GROUP BY DayOfWeek;

hive> SELECT * FROM avgdepdelay;

3   8.289761053658728

6   8.645680904903614

1   10.269990244459473

4   9.772897177836702

7   11.568973392595312

2   8.97689712068735

5   12.158036387869656

e) Find the average departure delay per day in 2008.
#calculate average delay
hive> select sum(delay) from hbase_flight_new;

🎬 Day 5 under the results in Step (B) — had the highest number of delays.

🎬 Step (A): We want to point out that Hive's Data Definition Language (DDL) also includes the CREATE VIEW statement, which can be quite useful. In Hive, views allow a query to be saved but data is not stored as with the Create Table as Select (CTAS) statement.

🎬 When a view is referenced in HiveQL, Hive executes the query and then uses the results which could be part of a larger query. This can be very useful to simplify complex queries and break them down into logical components.

🎬 Additionally, the GROUP BY clause, which gathers all the days per week and allows the AVG aggregate function to provide a consolidated answer per day.

🎬 After we answered our question above about average flight delays per day,