**RUNNING HiveQL :**

**(In Terminal) –**

**(Method 1)**

1]student@Ubantu:~$ **start-dfs.sh**

2]student@Ubantu:~$ **start-yarn.sh**

3]student@Ubantu:~$ **jps**

4]student@Ubantu:~$ **cd /usr/local/hive/bin**

OR

student@Ubantu:~$ **cd ~/Downloads/hive/bin**

5] student@Ubantu:~/Downloads/hive/bin$ **schematool -initSchema -dbType derby**

student@Ubantu:~/Downloads/hive/bin$ **cd ..**

student@Ubantu:~/Downloads/hive$ **rm -rf metastore\_db**

student@Ubantu:~/Downloads/hive$ **rm -rf derby.log**

student@Ubantu:~/Downloads/hive$ **cd ~/Downloads/hive/bin**

student@Ubantu:~/Downloads/hive/bin$ **./schematool -initSchema -dbType derby**

\*\*\* schemaTool failed \*\*\*

student@Ubantu:~/Downloads/hive/bin$ **rm -rf metastore\_db**

student@Ubantu:~/Downloads/hive/bin$ **rm -rf derby.log**

student@Ubantu:~/Downloads/hive/bin$ **./schematool -initSchema -dbType derby**

Initialization script completed

6]**hive**

Hive>

**Method 2 :**

1] student@Ubantu:~$ **start-all.sh**

2]student@Ubantu:~$ **cd /usr/local/hive/bin**

OR

student@Ubantu:~$ **cd ~/Downloads/hive/bin**

3]**hive**

**OR**

**./hive**

**QUERIES :**

Write HiveQL for Flight Reservation

a) Creating, Dropping, and altering tables.

b) Insert values in tables/ Load the data from flight dataset

c) Display data from table

d) Create index.

e) Join

**a) Creating, Dropping, and Altering Tables**

**Creating a table for flights:**

CREATE TABLE flight\_reservation (

flight\_id STRING,

flight\_name STRING,

source STRING,

destination STRING,

departure\_time STRING,

arrival\_time STRING,

seats\_available INT

)

ROW FORMAT DELIMITED

FIELDS TERMINATED BY ','

STORED AS TEXTFILE;

**Dropping the table:**

DROP TABLE flight\_reservation;

*(You can recreate it later after drop for practice.)*

**Altering the table:**

Example: Add a new column like price

ALTER TABLE flight\_reservation ADD COLUMNS (price DOUBLE);

Example: Change column name (say, from "flight\_name" to "airline\_name")

ALTER TABLE flight\_reservation CHANGE flight\_name airline\_name STRING;

**b) Insert values or Load Data**

**Method 1: Manually Insert values**

INSERT INTO TABLE flight\_reservation VALUES

('F001', 'Indigo', 'Pune', 'Delhi', '06:00', '08:00', 120, 4500.00),

('F002', 'SpiceJet', 'Mumbai', 'Bangalore', '09:00', '11:00', 100, 4000.00);

**Method 2: Load Data from a Flight Dataset**

Suppose you have a CSV file like flights.csv:

Sample flights.csv content:

F001,Indigo,Pune,Delhi,06:00,08:00,120,4500

F002,SpiceJet,Mumbai,Bangalore,09:00,11:00,100,4000

First, put your CSV file into HDFS:

hdfs dfs -mkdir /user/hive/warehouse/flights

hdfs dfs -put /home/your\_username/flights.csv /user/hive/warehouse/flights/

Then load it into the Hive table:

LOAD DATA INPATH '/user/hive/warehouse/flights/flights.csv' INTO TABLE flight\_reservation;

**c) Display data from table**

Simple select:

SELECT \* FROM flight\_reservation;

Specific columns:

SELECT flight\_id, source, destination FROM flight\_reservation;

**d) Create Index**

Hive does **not** support traditional RDBMS indexes directly, but you can create **indexes** using HiveQL like this:

CREATE INDEX idx\_source\_destination

ON TABLE flight\_reservation (source, destination)

AS 'COMPACT'

WITH DEFERRED REBUILD;

**To rebuild the index:**

ALTER INDEX idx\_source\_destination ON flight\_reservation REBUILD;

**To show the index:**

SHOW INDEX ON flight\_reservation;

**e) Join**

Suppose you have another table airlines:

CREATE TABLE airlines (

airline\_name STRING,

rating INT

)

ROW FORMAT DELIMITED

FIELDS TERMINATED BY ','

STORED AS TEXTFILE;

Insert some sample data:

INSERT INTO TABLE airlines VALUES

('Indigo', 4),

('SpiceJet', 3);

**Now JOIN them:**

SELECT fr.flight\_id, fr.source, fr.destination, al.rating

FROM flight\_reservation fr

JOIN airlines al

ON (fr.airline\_name = al.airline\_name);