

Experiment 4

Shashwat Shah

60004220126

TY BTECH COMPSB

Aim: Execute hive commands to load, insert, retrieve, update or delete data in tables.

Theory: Hive is a data warehouse infrastructure tool to process the structured data in Hadoop. It resides on top of Hadoop to summarise Big Data, and makes querying and analysing 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.

Commands:

- 1) Show databases
Show all the existing databases.
- 2) Create database student
Creates a database named student.
- 3) Use student
Use the database student for executing queries.
- 4) Create table student
(id int, name varchar(20));
Partitioned by (load-date date)
Clustered by (id) into 3 buckets
Stored as ORC FOR EDUCATIONAL USE TBLPROPERTIES ('transactional' = 'true');

- 5) Insert into Student position (load_date = '2023-03-09') values (101, 'xyz', 123456);
- 6) Select * from student
To display contents of table.
- 7) update student set name = 'xyz' where id = 101;
Update the values in the table.
- 8) delete from student where name = 'xyz';
Delete the entry from that table.
- 9) alter table student rename to stud;
- rename the table.
- 10) Relational operations
select * from stud where CGPA < 9.0;
- 11) Join in hive
select * from customer c join orders o on (c.id = o.customer_id)
- 12) Views in hive
create view if not exists customers_vw as
select * from customers where address = 'Mumbai'
- 13) Drop the view
drop view if exists customers_vw;

Conclusion: We have successfully executed HIVE queries using HQL in hadoop.



BDI

HIVE Query Language

<http://127.0.0.1:4200> sandbox login: root
root@sandbox.hortonworks.com's password:
Last login: Thu Mar 9 06:30:54 2024 from 172.17.0.2
[root@sandbox ~]# hive

Logging initialized using configuration in file:/etc/hive/2.5.0.0-1245/0/hive-log4j.properties

Show databases already existing in Hive

```
hive> show
databases; OK
bdiexample default
foodmart retail
xademo
Time taken: 0.025 seconds, Fetched: 5 row(s)
```

Creating a new database

```
hive> create database studentdb;
OK
Time taken: 0.072 seconds
```

Using the database for executing queries

```
hive> use studentdb;
OK
Time taken: 0.042 seconds
```

Creating a table in Hive

The most used optional clauses in creating a table are:

- IF NOT EXISTS – You can use IF NOT EXISTS to avoid the error in case the table is already present. Hive checks if the requesting table already presents,
- EXTERNAL – Used to create external table
- TEMPORARY – Used to create temporary table.
- ROW FORMAT – Specifies the format of the row.
- FIELDS TERMINATED BY – By default Hive use ^A field separator, To load a file that has a custom field separator like comma, pipe, tab use this option.
- PARTITION BY – Used to create partition data. Using this improves performance.
- CLUSTERED BY – Dividing the data into a specific number for buckets.
- LOCATION – You can specify the custom location where to store the data on HDFS.
- The *Optimized Row Columnar* (ORC) file format provides a highly efficient way to store Hive data. Using ORC files improves performance when Hive is reading, writing, and processing data. Other file formats supported are : JSON, Text, Sequence, etc.

```
hive> create table student(id int, name varchar(20), branch varchar(20), mobile int)
> PARTITIONED BY (load_date date)
> CLUSTERED BY(id) INTO 3 BUCKETS
> STORED AS ORC TBLPROPERTIES ('transactional'='true'); OK
Time taken: 0.657 seconds
```

#Inserting values in the table

```
hive> insert into student partition (load_date = '2024-03-09') values
(101,'Manav','Computer',123456);
```

Query ID = root_20240309064202_db1b9f50-234f-42ed-b028-22d9ba0ff933

Total jobs = 1

Launching Job 1 out of 1

Status: Running (Executing on YARN cluster with App id application_1678329811540_0003)

VERTICES	STATUS	TOTAL	COMPLETED	RUNNING	PENDING	FAILED	KILLED
Map 1	SUCCEEDED	1	1	0	0	0	0

VERTICES: 01/01 [=====>>] 100% ELAPSED TIME: 5.54 s

Loading data to table studentdb.student partition (load_date=2024-03-09)

Partition studentdb.student{load_date=2024-03-09} stats: [numFiles=1, numRows=0, totalSize=858, rawDataSize=0]

OK

Time taken: 10.075 seconds

```
hive> insert into student partition (load_date = '2024-03-09') values (102,'Devraj','Computer',341256);
```

Query ID = root_20240309064238_73e473b3-adc1-4fe9-818c-e077dbed113c

Total jobs = 1

Launching Job 1 out of 1

Status: Running (Executing on YARN cluster with App id application_1678329811540_0003)

VERTICES	STATUS	TOTAL	COMPLETED	RUNNING	PENDING	FAILED	KILLED
Map 1	SUCCEEDED	1	1	0	0	0	0

VERTICES: 01/01 [=====>>] 100% ELAPSED TIME: 4.71 s

Loading data to table studentdb.student partition (load_date=2024-03-09)
Partition studentdb.student{load_date=2024-03-09} stats: [numFiles=2, numRows=0, totalSize=1720, rawD ataSize=0]
OK
Time taken: 5.983 seconds

To display contents of a table

```
hive> select * from student;
OK
101      Manav Computer      123456 2024-03-09 102      Devraj Computer      341256 2024-
03-09
```

Time taken: 0.216 seconds, Fetched: 2 row(s)

Order By Clause (Ordering the result in descending order; by default ascending order)

```
hive> select id, name from student order by id desc;
```

Query ID = root_20240309064323_da0caba1-19a2-433e-b579-4505f17dcabc
Total jobs = 1
Launching Job 1 out of 1
Status: Running (Executing on YARN cluster with App id application_1678329811540_0003)

	VERTICES	STATUS	TOTAL	COMPLETED	RUNNING	PENDING	FAILED	KILLED
Map 1	SUCCEEDED	3	3	0	0	0	0	0
Reducer 2	SUCCEEDED	1	1	0	0	0	0	0

VERTICES: 02/02 [=====>>] 100% ELAPSED TIME: 16.65 s

OK

```
102      Devraj
101      Manav
```

Time taken: 17.594 seconds, Fetched: 2 row(s)

Inserting a new tuple in the table

```
hive> insert into student partition (load_date = '2024-03-09') values (103,'Sahil','Mechanical',98762
3);
```

Query ID = root_20240309064439_bfdf6f3d-ce54-4cf6-b31e-65bc23b689e1
Total jobs = 1
Launching Job 1 out of 1

Status: Running (Executing on YARN cluster with App id application_1678329811540_0003)

~~VERTICES~~ ~~STATUS~~ ~~TOTAL~~ ~~COMPLETED~~ ~~RUNNING~~ ~~PENDING~~ ~~FAILED~~ ~~KILLED~~

Map 1 SUCCEEDED 1 1 0 0 0 0

~~VERTICES: 01/01 [=====>>] 100%~~ ELAPSED TIME: 3.67 s

Loading data to table studentdb.student partition (load_date=2024-03-09)
Partition studentdb.student{load_date=2024-03-09} stats: [numFiles=3, numRows=0, totalSize=2579, rawD
ataSize=0]
OK
Time taken: 5.262 seconds

Display contents of the table after new entry

```
hive> select * from student;
OK
101    Manav Computer      123456 2024-03-09 102    Devraj Computer      341256 2024-
03-09 103    Sahil Mechanical      987623 2024-03-09
```

Time taken: 0.121 seconds, Fetched: 3 row(s)

Group By Clause (Grouping the result w.r.t branch here)

hive> SELECT branch,count(*) FROM student GROUP BY branch;

Query ID = root_20240309064938_1c36c2cc-682f-4956-bed3-4897f9bd3be4 Total jobs
= 1
Launching Job 1 out of 1

Status: Running (Executing on YARN cluster with App id application_1678329811540_0003)

~~VERTICES~~ ~~STATUS~~ ~~TOTAL~~ ~~COMPLETED~~ ~~RUNNING~~ ~~PENDING~~ ~~FAILED~~ ~~KILLED~~

Map 1 SUCCEEDED 3 3 0 0 0 0
Reducer 2SUCCEEDED 1 1 0 0 0 0

VERTICES: 02/02 [=====>>] 100% ELAPSED TIME: 20.15 s

OK

```
Computer 2
Mechanical 1
```

Time taken: 21.075 seconds, Fetched: 2 row(s)

Update the value in table

Note: update, delete statements will work only if during the creation of table transactional property is set to 'true'. `hive> update student set name='Sayli' where id = 103;`

Query ID = root_20240309070310_dee26faf-f65d-4c8a-b966-bc18cc118ccc

Total jobs = 1

Launching Job 1 out of 1

Tez session was closed. Reopening...

Session re-established.

Status: Running (Executing on YARN cluster with App id application_1678329811540_0004)

VERTICES		STATUS TOTAL COMPLETED RUNNING PENDING FAILED KILLED					
Map 1	SUCCEEDED	3	3	0	0	0	0
Reducer 2	SUCCEEDED	1	1	0	0	0	0
VERTICES: 02/02 [=====>>] 100% ELAPSED TIME: 26.39 s							

Loading data to table studentdb.student partition (load_date=null)

Time taken for load dynamic partitions : 464

Loading partition {load_date=2024-03-09}

Time taken for adding to write entity : 5

Partition studentdb.student{load_date=2024-03-09} stats: [numFiles=4, numRows=0, totalSize=3457, rawDataSize=0]

OK

Time taken: 36.217 seconds

```
hive> select * from student;
```

OK

```
101 Manav Computer      123456 2024-03-09 102 Devraj
Computer                341256 2024-03-09
103 Sayli Mechanical      987623 2024-03-09
```

Time taken: 0.262 seconds, Fetched: 3 row(s)

#Delete the entry from the table

`hive> delete from student where name = 'Sayli';`

Query ID = root_20240309070624_382e39b5-755b-41c2-8f04-362939800cbf

Total jobs = 1

Launching Job 1 out of 1

Status: Running (Executing on YARN cluster with App id application_1678329811540_0004)

VERTICES	STATUS	TOTAL	COMPLETED	RUNNING	PENDING	FAILED	KILLED
Map 1	SUCCEEDED	3	3	0	0	0	0
Reducer 2	SUCCEEDED	1	1	0	0	0	0

VERTICES: 02/02 [=====>>] 100% ELAPSED TIME: 19.68 s

Loading data to table studentdb.student partition (load_date=null)

Time taken for load dynamic partitions : 516

Loading partition {load_date=2024-03-09}

Time taken for adding to write entity : 0

Partition studentdb.student{load_date=2024-03-09} stats: [numFiles=5, numRows=0, totalSize=3990, rawDataSize=0]

OK

Time taken: 22.826 seconds

hive> select * from student;

OK

101 Manav Computer 123456 2024-03-09

102 Devraj Computer 341256 2024-03-09 Time taken: 0.319 seconds, Fetched: 2 row(s)

Rename a table using the ALTER statement

hive> alter table student rename to stud;

OK

Time taken: 0.539 seconds

Check whether table is renamed

hive> show tables; OK

stud

values_tmp ____

____table 1 ____

values_tmp____table 2

values_tmp_table_3

Time taken: 0.119 seconds, Fetched: 4 row(s)

Add new column in the table using ALTER statement

hive> alter table stud add columns(cgpa double);

OK

Time taken: 0.441 seconds

CGPA column is added. Initially its value in all rows will be NULL as we haven't entered CGPA value.

hive> select * from stud;

OK

101 Manav Computer 123456 NULL 2024-03-09

Changing Column Name with ALTER statement and also changing the datatype from varchar to String.

```
hive> ALTER TABLE stud CHANGE name sname String;
OK
Time taken: 0.478 seconds
```

Updating the CGPA values in table (i.e. changing the NULL Value)

```
hive> update stud set cgpa = 7.9 where id = 101;
Query ID = root_20240309071806_94217582-6ff2-408e-addf-3af8a3655e94 Total
jobs = 1
Launching Job 1 out of 1
Tez session was closed. Reopening... Session re-established.
Status: Running (Executing on YARN cluster with App id application_1678329811540_0005)
```

VERTICES								STATUS TOTAL COMPLETED RUNNING PENDING FAILED KILLED							
Map 1								SUCCEEDED	3	3	0	0	0	0	0
Reducer 2								SUCCEEDED	1	1	0	0	0	0	0

VERTICES: 02/02 [=====>>] 100% ELAPSED TIME: 33.39 s

```
Loading data to table studentdb.stud partition (load_date=null)
Time taken for load dynamic partitions : 417
Loading partition {load_date=2024-03-09}
Time taken for adding to write entity : 0
Partition studentdb.stud{load_date=2024-03-09} stats: [numFiles=6, numRows=0, totalSize=4956,
rawDataSize=0]
OK
Time taken: 45.136 seconds
```

```
hive> update stud set cgpa = 8.8 where id = 102;
```

```
Query ID = root_20240309071919_6f74a8dc-b5a9-406b-81af-fea4bc3a09e8 Total
jobs = 1
Launching Job 1 out of 1
Status: Running (Executing on YARN cluster with App id application_1678329811540_0005)
```

VERTICES								STATUS TOTAL COMPLETED RUNNING PENDING FAILED KILLED							
Map 1								SUCCEEDED	3	3	0	0	0	0	0
Reducer 2								SUCCEEDED	1	1	0	0	0	0	0

VERTICES: 02/02 [=====>>] 100% ELAPSED TIME: 12.69 s

Loading data to table studentdb.stud partition (load_date=null)

Time taken for load dynamic partitions : 183

Loading partition {load_date=2024-03-09}

Time taken for adding to write entity : 1

Partition studentdb.stud{load_date=2024-03-09} stats: [numFiles=7, numRows=0, totalSize=5926, rawDataSize=0]

OK

Time taken: 14.739 seconds

Check whether the CGPA values are updated.

```
hive> select * from stud;
```

OK

```
101  Manav Computer  123456 7.9      2024-03-09
102  Devraj Computer 341256 8.8      2024-03-09
```

Time taken: 0.196 seconds, Fetched: 2 row(s)

Performing Aggregate Functions (SUM, MAX, MIN, AVG, COUNT) on table

```
hive> select sum(cgpa) from stud;
```

Query ID = root_20240309074116_00e58970-8700-4fd9-897b-15e8ce2a3d5c

Total jobs = 1

Launching Job 1 out of 1

Tez session was closed. Reopening...

Session re-established.

Status: Running (Executing on YARN cluster with App id application_1678329811540_0006)

VERTICES	STATUS	TOTAL	COMPLETED	RUNNING	PENDING	FAILED	KILLED
----------	--------	-------	-----------	---------	---------	--------	--------

Map 1	SUCCEEDED	3	3	0	0	0	0
-------------	-----------	---	---	---	---	---	---

Reducer 2	SUCCEEDED	1	1	0	0	0	0
-----------------	-----------	---	---	---	---	---	---

VERTICES: 02/02 [=====>>] 100% ELAPSED TIME: 25.89 s

OK

```
16.700000000000003
```

Time taken: 39.321 seconds, Fetched: 1 row(s)

```
hive> select max(cgpa) from stud;
```

Query ID = root_20240309074240_a75dc01e-f8c6-4694-b6d1-45012b7652e6

Total jobs = 1

Status: Running (Executing on YARN cluster with App id application_1678329811540_0006)

OK

Time taken: 14.61 seconds, Fetched: 1 row(s)

Launching Job 1 out of 1

Status: Running (Executing on YARN cluster with App id application_1678329811540_0006)

OK

Time taken: 19.969 seconds, Fetched: 1 row(s)

Launching Job 1 out of 1

Status: Running (Executing on YARN cluster with App id application_1678329811540_0006)

VERTICES	STATUS	TOTAL	COMPLETED	RUNNING	PENDING	FAILED	KILLED
----------	--------	-------	-----------	---------	---------	--------	--------

```

Map 1 ..... SUCCEEDED    3    3    0    0    0    0
Reducer 2 .....SUCCEEDED    1    1    0    0    0    0

```

VERTICES: 02/02 [=====>>] 100% ELAPSED TIME: 16.41 s

OK

8.350000000000001

Time taken: 18.007 seconds, Fetched: 1 row(s)

hive> select count(cgpa) from stud;

Query ID = root_20240309074649_0342e8d0-ab88-40a0-9306-16aca6cdb51e

Total jobs = 1

Launching Job 1 out of 1

Status: Running (Executing on YARN cluster with App id application_1678329811540_0006)

VERTICES STATUS TOTAL COMPLETED RUNNING PENDING FAILED KILLED

```

Map 1 ..... SUCCEEDED    3    3    0    0    0    0
Reducer 2 .....SUCCEEDED    1    1    0    0    0    0

```

VERTICES: 02/02 [=====>>] 100% ELAPSED TIME: 8.17 s

OK

2

Time taken: 9.295 seconds, Fetched: 1 row(s)

Operators in HIVE (Arithmetic, Relational, Logical)

Relational Operators

These operators are used to compare two operands. The following table describes the relational operators available in Hive:

Operator	Operand	Description
A = B	all primitive types	TRUE if expression A is equivalent to expression B otherwise FALSE.
A != B	all primitive types	TRUE if expression A is not equivalent to expression B otherwise FALSE.
A < B	all primitive types	TRUE if expression A is less than expression B otherwise FALSE.
A <= B	all primitive types	TRUE if expression A is less than or equal to expression B otherwise FALSE.
A > B	all primitive types	TRUE if expression A is greater than expression B otherwise FALSE.

A >= B	all primitive types	TRUE if expression A is greater than or equal to expression B otherwise FALSE.
A IS NULL	all types	TRUE if expression A evaluates to NULL otherwise FALSE.
A IS NOT NULL	all types	FALSE if expression A evaluates to NULL otherwise TRUE.
A LIKE B	Strings	TRUE if string pattern A matches to B otherwise FALSE.
A RLIKE B	Strings	NULL if A or B is NULL, TRUE if any substring of A matches the Java regular expression B , otherwise FALSE.
A REGEXP B	Strings	Same as RLIKE.

Arithmetic Operators

These operators support various common arithmetic operations on the operands. All of them return number types. The following table describes the arithmetic operators available in Hive:

Operators	Operand	Description
A + B	all number types	Gives the result of adding A and B.
A - B	all number types	Gives the result of subtracting B from A.
A * B	all number types	Gives the result of multiplying A and B.
A / B	all number types	Gives the result of dividing B from A.
A % B	all number types	Gives the remainder resulting from dividing A by B.
A & B	all number types	Gives the result of bitwise AND of A and B.
A B	all number types	Gives the result of bitwise OR of A and B.
A ^ B	all number types	Gives the result of bitwise XOR of A and B.
~A	all number types	Gives the result of bitwise NOT of A.

Logical Operators

The operators are logical expressions. All of them return either TRUE or FALSE.

Operators	Operands	Description
A AND B	boolean	TRUE if both A and B are TRUE, otherwise FALSE.
A && B	boolean	Same as A AND B.
A OR B	boolean	TRUE if either A or B or both are TRUE, otherwise FALSE.
A B	boolean	Same as A OR B.
NOT A	boolean	TRUE if A is FALSE, otherwise FALSE.
!A	boolean	Same as NOT A.

```
hive> select * from stud where cgpa < 9.0;
OK
```

101	Manav Computer	123456 7.9	2024-03-09
102	Devraj Computer	341256 8.8	2024-03-09

Time taken: 0.473 seconds, Fetched: 2 row(s)

hive> select * from stud where cgpa >= 7.5;			
OK			
101	Manav Computer	123456 7.9	2024-03-09
102	Devraj Computer	341256 8.8	2024-03-09

Time taken: 0.219 seconds, Fetched: 2 row(s)

Joins in HIVE

Basically, for combining specific fields from two tables by using values common to each one we use Hive JOIN clause.

In other words, to combine records from two or more tables in the database we use JOIN clause. **Types of Joins in Hive**

1. Inner join in Hive
2. Left Outer Join in Hive
3. Right Outer Join in Hive
4. Full Outer Join in Hive

a. Inner Join

Basically, to combine and retrieve the records from multiple tables we use Hive Join clause. Moreover, by **using the primary keys and foreign keys** of the tables JOIN condition is to be raised.

b. Left Outer Join

On defining HiveQL Left Outer Join, even if there are no matches in the right table it returns all the rows from the left table.

To be more specific, even if the ON clause matches 0 (zero) records in the right table, then also this Hive JOIN still returns a row in the result. Although, it returns with NULL in each column from the right table.

In addition, it returns all the values from the left table. Also, the matched values from the right table, or NULL in case of no matching JOIN predicate.

c. Right Outer Join

Basically, even if there are no matches in the left table, HiveQL Right Outer Join returns all the rows from the right table.

To be more specific, even if the ON clause matches 0 (zero) records in the left table, then also this Hive JOIN still returns a row in the result. Although, it returns with NULL in each column from the left table.

In addition, it returns all the values from the right table. Also, the matched values from the left table or NULL in case of no matching join predicate.

d. Full Outer Join

The major purpose of this HiveQL Full outer Join is it combines the records of both the left and the right outer tables which fulfills the Hive JOIN condition. Moreover, this joined table contains either all the records from both the tables or fills in NULL values for missing matches on either side.

Creating and inserting values in 2 tables namely, customers and orders to execute JOINS in HIVE.


```
hive> create table customers(id int, name String, age int, address String, salary int)
> partitioned by (load_date date)
> clustered by(id) into 3 buckets
> stored as orc tblproperties ('transactional'='true'); OK
Time taken: 0.842 seconds
```

```
hive> insert into customers partition (load_date = '2024-03-09') values
(1,"Ross",25,"Mumbai",25000);
```

Query ID = root_20240309080253_358e5b90-f99b-4fc3-a2cd-41571ab6b21c
Total jobs = 1
Launching Job 1 out of 1
Tez session was closed. Reopening...
Session re-established.
Status: Running (Executing on YARN cluster with App id application_1678329811540_0007)

VERTICES	STATUS	TOTAL	COMPLETED	RUNNING	PENDING	FAILED	KILLED
Map 1	SUCCEEDED	1	1	0	0	0	0

VERTICES: 01/01 [=====>>] 100% ELAPSED TIME: 5.21 s

Loading data to table studentdb.customers partition (load_date=2024-03-09)
Partition studentdb.customers{load_date=2024-03-09} stats: [numFiles=1, numRows=0, totalSize=865, rawDataSize=0]
OK
Time taken: 14.342 seconds

```
hive> insert into customers partition (load_date = '2024-03-09') values
(2,"Mike",27,"Bhopal",35000);
```

Query ID = root_20240309080416_207775fb-ecc4-40c0-a216-281a829b9581
Total jobs = 1
Launching Job 1 out of 1
Status: Running (Executing on YARN cluster with App id application_1678329811540_0007)

VERTICES	STATUS	TOTAL	COMPLETED	RUNNING	PENDING	FAILED	KILLED
Map 1	SUCCEEDED	1	1	0	0	0	0

VERTICES: 01/01 [=====>>] 100% ELAPSED TIME: 7.01 s

Loading data to table studentdb.customers partition (load_date=2024-03-09)
Partition studentdb.customers{load_date=2024-03-09} stats: [numFiles=2, numRows=0, totalSize=1735, rawDataSize=0]
OK

Time taken: 8.722 seconds

```
hive> insert into customers partition (load_date = '2024-03-10')values(3,"Albin",24,"Pune",50000);
```

Query ID = root_20240309085540_1bc1239b-2b0d-444f-bea5-e1c69576df93

Total jobs = 1

Launching Job 1 out of 1

Status: Running (Executing on YARN cluster with App id application_1678329811540_0008)

~~-----VERTICES-----STATUS-----TOTAL COMPLETED RUNNING PENDING FAILED KILLED~~

Map 1 SUCCEEDED 1 1 0 0 0 0

~~VERTICES: 01/01 [----->>] 100% ELAPSED TIME: 3.34 s~~

Loading data to table studentdb.customers partition (load_date=2024-03-10)

Partition studentdb.customers{load_date=2024-03-10} stats: [numFiles=1,
numRows=0, totalSize=863, rawDataSize=0]

OK

Time taken: 5.857 seconds

```
hive> select * from customers;
```

OK

1 Ross 25 Mumbai 25000 2024-03-09

2 Mike 27 Bhopal 35000 2024-03-09

3 Albin 24 Pune 50000 2024-03-10

Time taken: 0.302 seconds, Fetched: 3 row(s)

```
hive> create table orders(oid int, customer_id int, amount int);
```

OK

Time taken: 0.606 seconds

```
hive> insert into orders values(101, 2, 20000);
```

```
hive> insert into orders values(102, 2, 25000);
```

```
hive> insert into orders values(104,4,10000); hive> select * from orders;
```

OK

101 2 20000

102 2 25000

103 1 30000

104 4 10000

Time taken: 0.174 seconds, Fetched: 4 row(s)

Executing Inner Join

hive> select * from customers c join orders o >
on (c.id = o.customer_id);
Query ID = root_20240309090029_4b55b07e-4508-478f-a05b-4eaca94e4190
Total jobs = 1
Launching Job 1 out of 1
Status: Running (Executing on YARN cluster with App id application_1678329811540_0008)

VERTICES	STATUS	TOTAL	COMPLETED	RUNNING	PENDING	FAILED	KILLED
Map 1	SUCCEEDED	6	6	0	0	0	0
Map 2	SUCCEEDED	1	1	0	0	0	0

VERTICES: 02/02 [=====>>] 100% ELAPSED TIME: 19.48 s

OK

1	Ross 25	Mumbai 25000	2024-03-09	103	1	30000
2	Mike 27	Bhopal 35000	2024-03-09	101	2	20000
2	Mike 27	Bhopal 35000	2024-03-09	102	2	25000

Time taken: 20.319 seconds, Fetched: 3 row(s)

Executing LEFT OUTER JOIN

hive> select * from customers c left outer join orders o
> on (c.id = o.customer_id);
Query ID = root_20230309090159_2d9c595a-7a7b-45d5-ad28-81e65ac22b64
Total jobs = 1
Launching Job 1 out of 1
Status: Running (Executing on YARN cluster with App id application_1678329811540_0008)

VERTICES	STATUS	TOTAL	COMPLETED	RUNNING	PENDING	FAILED	KILLED
Map 1	SUCCEEDED	6	6	0	0	0	0
Map 2	SUCCEEDED	1	1	0	0	0	0

VERTICES: 02/02 [=====>>] 100% ELAPSED TIME: 13.92 s

OK

1	Ross 25	Mumbai 25000	2023-03-09	103	1	30000
2	Mike 27	Bhopal 35000	2023-03-09	101	2	20000

2	Mike	27	Bhopal	35000	2024-03-09	102	2	25000
3	Albin	24	Pune	50000	2024-03-10	NULL	NULL	NULL

Time taken: 15.026 seconds,
 Fetched: 4 row(s)

select * from customers c right outer join orders o >
on (c.id = o.customer_id);
Query ID = root_20240309090237_47fe2670-0826-4677-845e-b2cd58660148
Total jobs = 1
Launching Job 1 out of 1
Status: Running (Executing on YARN cluster with App id application_1678329811540_0008)

VERTICES	STATUS	TOTAL	COMPLETED	RUNNING	PENDING	FAILED	KILLED

Map 1	SUCCEEDED	6	6	0	0	0	0
Map 2	SUCCEEDED	1	1	0	0	0	0

VERTICES: 02/02 [=====>>] 100% ELAPSED TIME: 10.82 s

OK

2	Mike	27	Bhopal	35000	2024-03-09	101	2	20000
2	Mike	27	Bhopal	35000	2024-03-09	102	2	25000
1	Ross	25	Mumbai	25000	2024-03-09	103	1	30000
NULL	NULL	NULL	NULL	NULL	NULL	104	4	10000

Time taken: 11.516 seconds, Fetched: 4 row(s)

Executing RIGHT OUTER JOIN

Executing FULL OUTER JOIN

```
hive> select * from customers c full outer join orders o
> on (c.id = o.customer_id);
Query ID = root_20230309090307_762521ad-0c69-4310-83f2-5d8061121d8a
Total jobs = 1
Launching Job 1 out of 1
Status: Running (Executing on YARN cluster with App id application_1678329811540_0008)
```

```
-----
VERTICES  STATUS TOTAL COMPLETED RUNNING PENDING FAILED KILLED
-----
Map 1.....SUCCEEDED   6     6     0     0     2     0
Map 3.....SUCCEEDED   1     1     0     0     0     0
Reducer 2 .....SUCCEEDED   1     1     0     0     0     0
-----
```

VERTICES: 03/03 [=====>>] 100% ELAPSED TIME: 18.78 s

```
-----
OK
1  Ross  25  Mumbai 25000 2023-03-09 103 1 30000
2  Mike  27  Bhopal 35000 2023-03-09 101 2 20000
2  Mike  27  Bhopal 35000 2023-03-09 102 2 25000
3  Albin 24  Pune   50000 2023-03-10 NULL NULL NULL
```

```
NULL NULL NULL NULL NULL NULL 104      4    10000
```

Time taken: 19.841 seconds, Fetched: 5 row(s)

Views in HIVE

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.

Creating a View on customers table for address Mumbai.

```
hive> create view if not exists customers_vw
> as select * from customers where address="Mumbai"; OK
Time taken: 0.248 seconds
```

Displaying the results of above view

```
hive> select * from customers_vw;
OK
1  Ross  25  Mumbai 25000 2024-03-09 Time taken: 0.146 seconds, Fetched: 1 row(s)
```

Creating a view on complete customers table instead of one single row (condition based)

```
hive> alter view customers_vw as select * from customers;
OK
Time taken: 0.212 seconds
```


Displaying the results of above view

```
hive> select * from customers_vw;  
OK  
1    Ross  25  Mumbai 25000 2024-03-09  
2    Mike  27  Bhopal 35000 2024-03-09  
Time taken: 0.11 seconds, Fetched: 2 row(s)
```

Dropping the view created

```
hive> drop view if exists customers_vw;  
OK  
Time taken: 0.462 seconds  
hive> select * from customers_vw;  
FAILED: SemanticException [Error 10001]: Line 1:14 Table not found 'customers_vw'
```