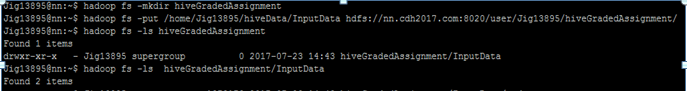
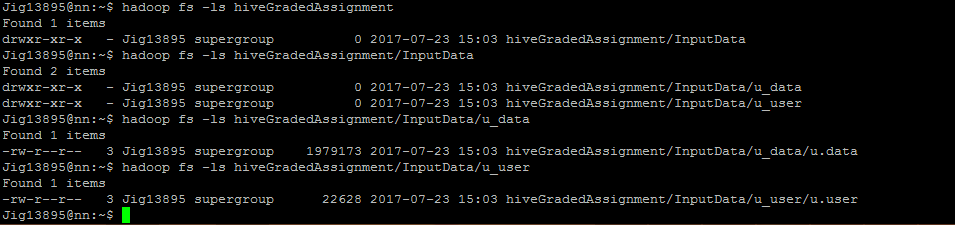
**Moving Input Data to HDFS**

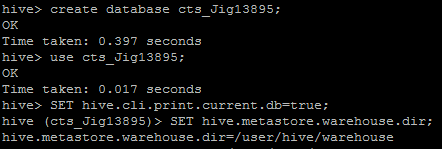
Creating InputData folder in HDFS



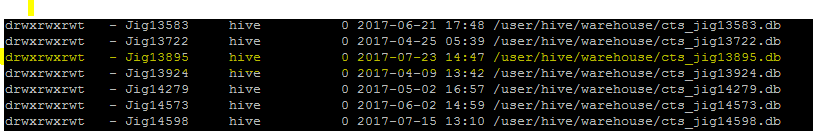
Moved the input files into corresponding sub directories in the InputData folder



New database is created with the name of cts\_Jig13895



Thus we can see the directory for the above database in warehouse



Initially there are no files under the created directory by hive.



**1. Create an external table u\_data for u.data in HDFS.**

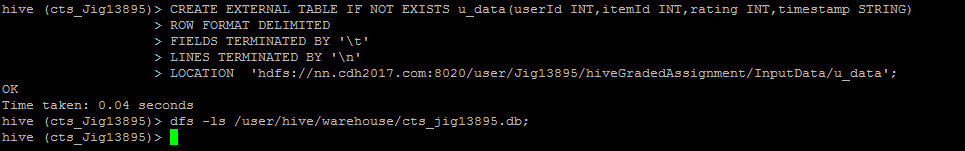
CREATE EXTERNAL TABLE IF NOT EXISTS u\_data(userId INT,itemId INT,rating INT,timestamp STRING)

ROW FORMAT DELIMITED

FIELDS TERMINATED BY '\t'

LINES TERMINATED BY '\n'

LOCATION 'hdfs://nn.cdh2017.com:8020/user/Jig13895/hiveGradedAssignment/InputData/u\_data';



Since we created external tables, the files in HDFS are not moved under the created database. HDFS has full control of the data.

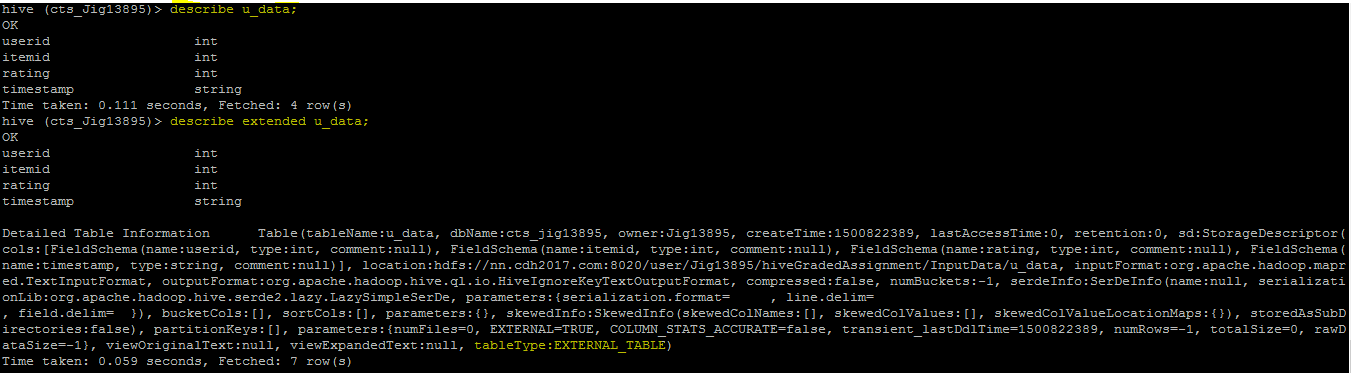


**2. See the field descriptions of u\_data table**

describe u\_data;

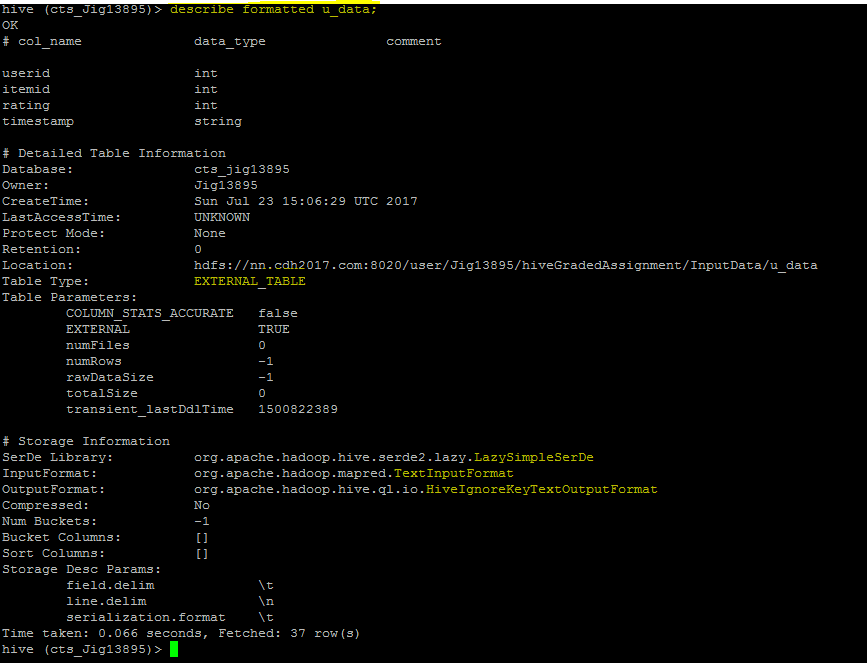
describe extended u\_data;

The description shows it is an external table.



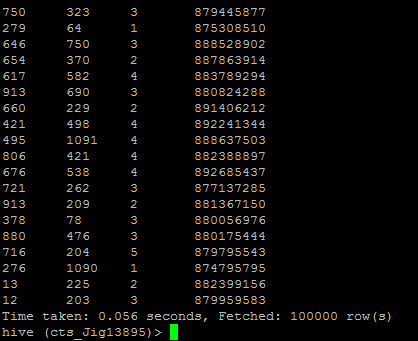
describe formatted u\_data;

The description shows it is an external table as well as it lists the input and output format.

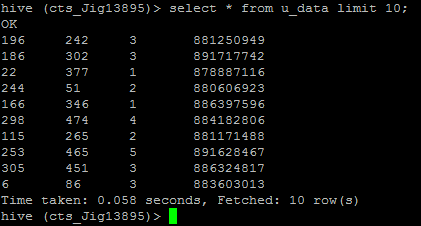


**3. Show all the data in the newly created u\_data table**

select \* from u\_data;

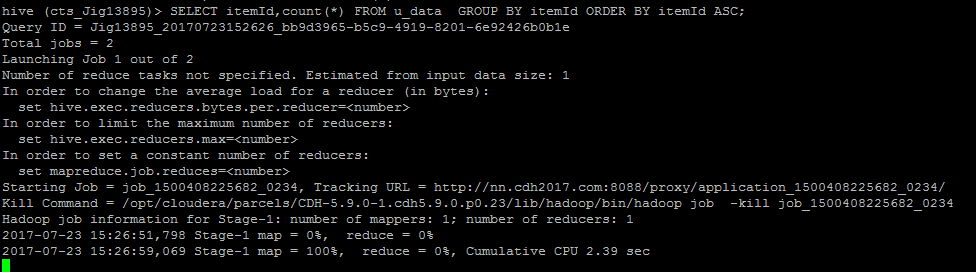


select \* from u\_data limit 10;



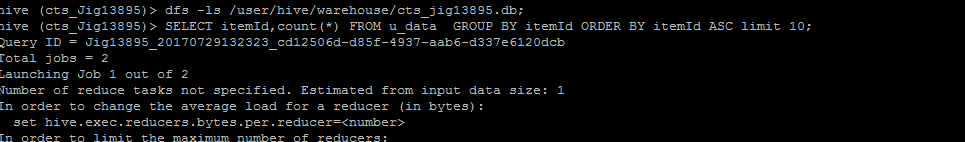
**4. Show the numbers of item reviewed by each user in the newly created u\_data table**

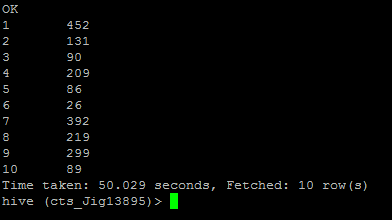
SELECT itemId,count(\*) FROM u\_data GROUP BY itemId ORDER BY itemId ASC;



Limit to 10 items

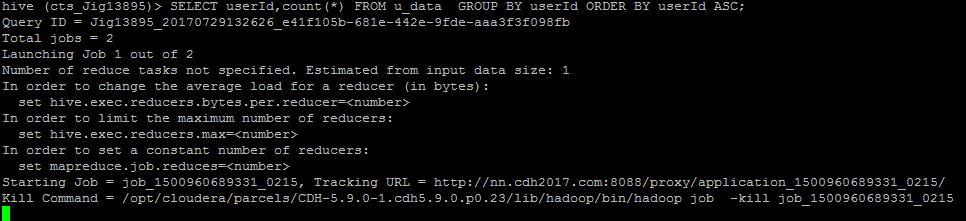
SELECT itemId,count(\*) FROM u\_data GROUP BY itemId ORDER BY itemId ASC limit 10;



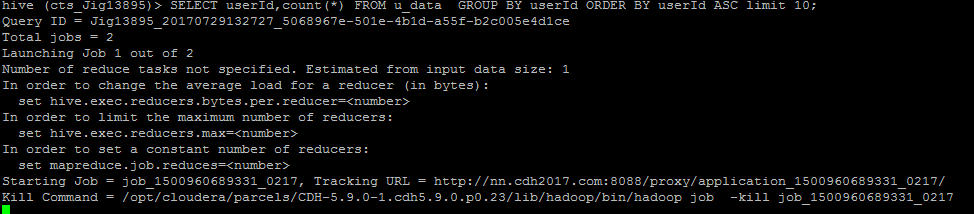


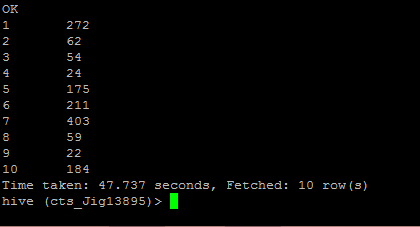
**5. Show the numbers of users reviewed each item in the newly created u\_data table**

SELECT userId,count(\*) FROM u\_data GROUP BY userId ORDER BY userId ASC;



Limit to 10 items





**6. Create an external table u\_user for u.user in HDFS .**

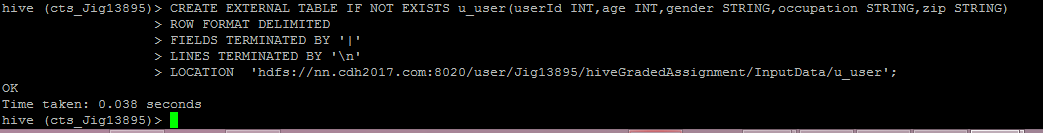
CREATE EXTERNAL TABLE IF NOT EXISTS u\_user(userId INT,age INT,gender STRING,occupation STRING,zip STRING)

ROW FORMAT DELIMITED

FIELDS TERMINATED BY '|'

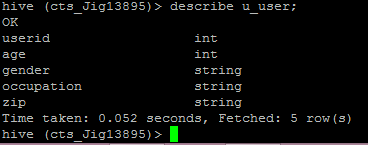
LINES TERMINATED BY '\n'

LOCATION 'hdfs://nn.cdh2017.com:8020/user/Jig13895/hiveGradedAssignment/InputData/u\_user';

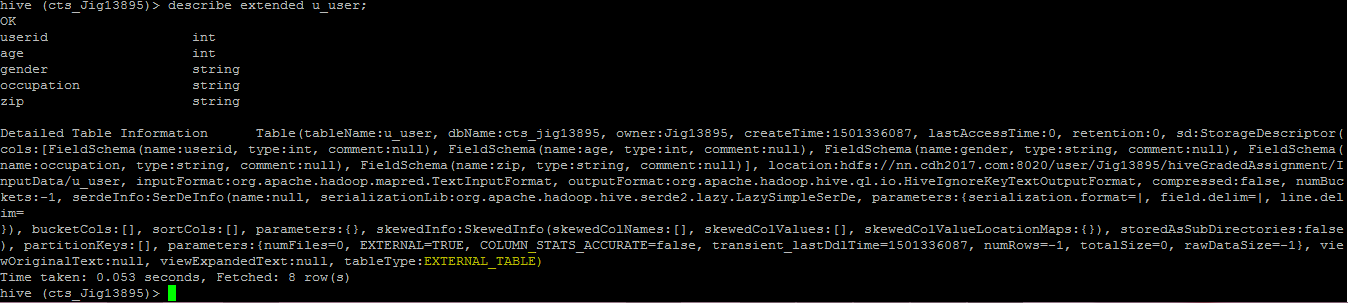


**7. See the field descriptions of u\_user table**

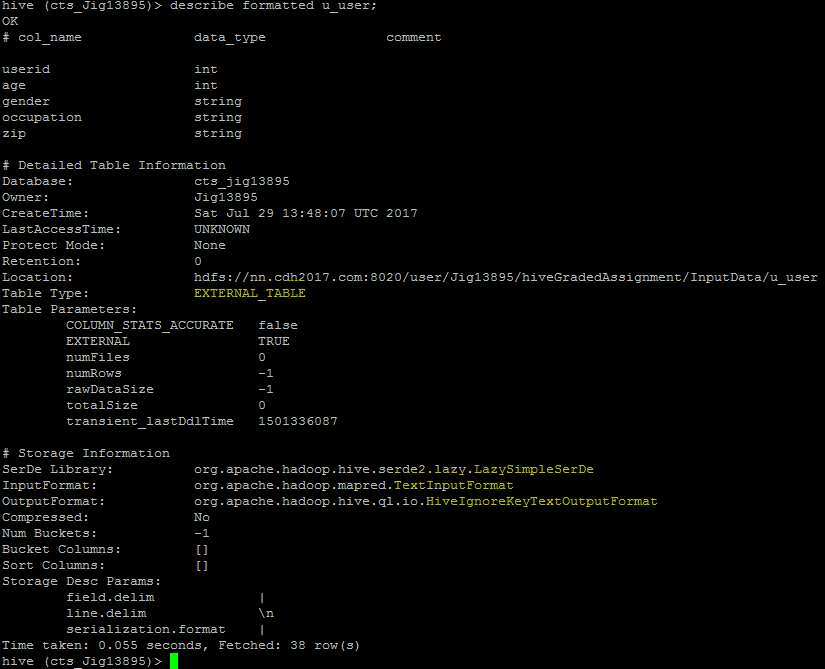
describe u\_user;



describe extended u\_user;



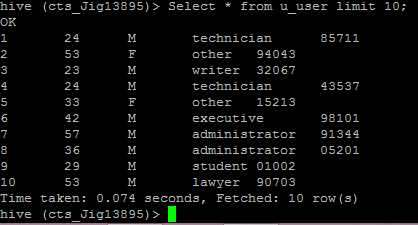
describe formatted u\_user;



**8. Show all the data in the newly created user table**

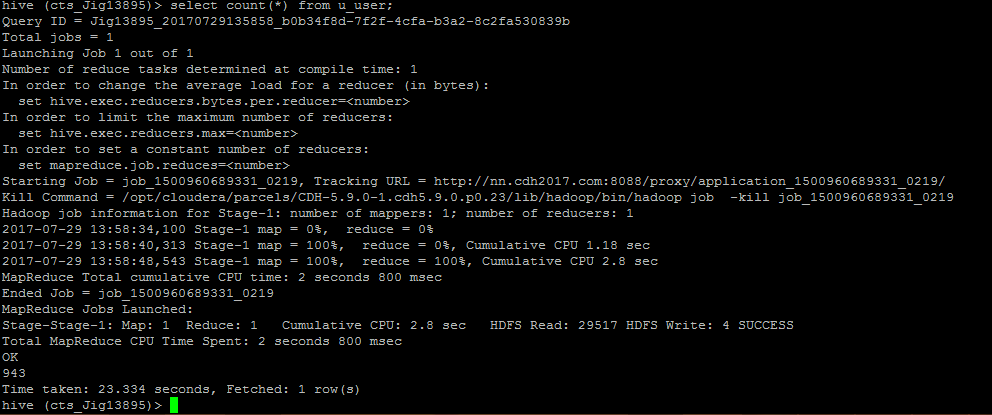
Select \* from u\_user;

Select \* from u\_user limit 10;



**9. Count the number of data in the u\_user table**

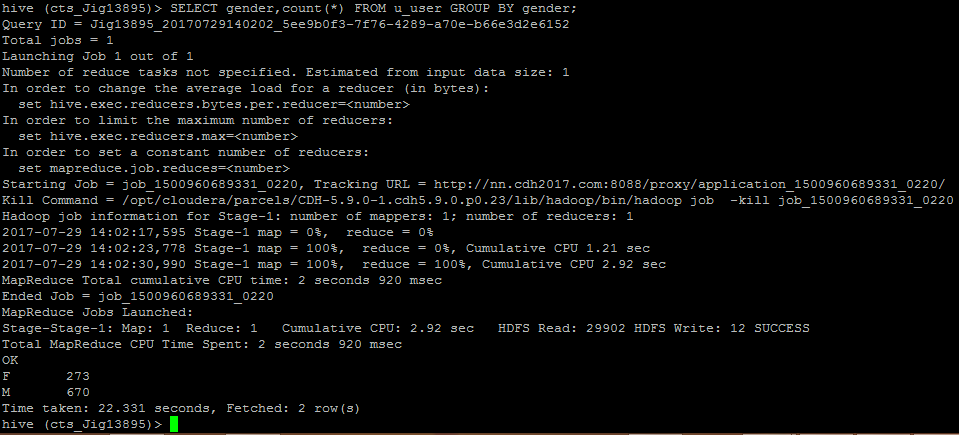
select count(\*) from u\_user;



Output is **943**

**10. Count the number of user in the u\_user table genderwise**

SELECT gender,count(\*) FROM u\_user GROUP BY gender;

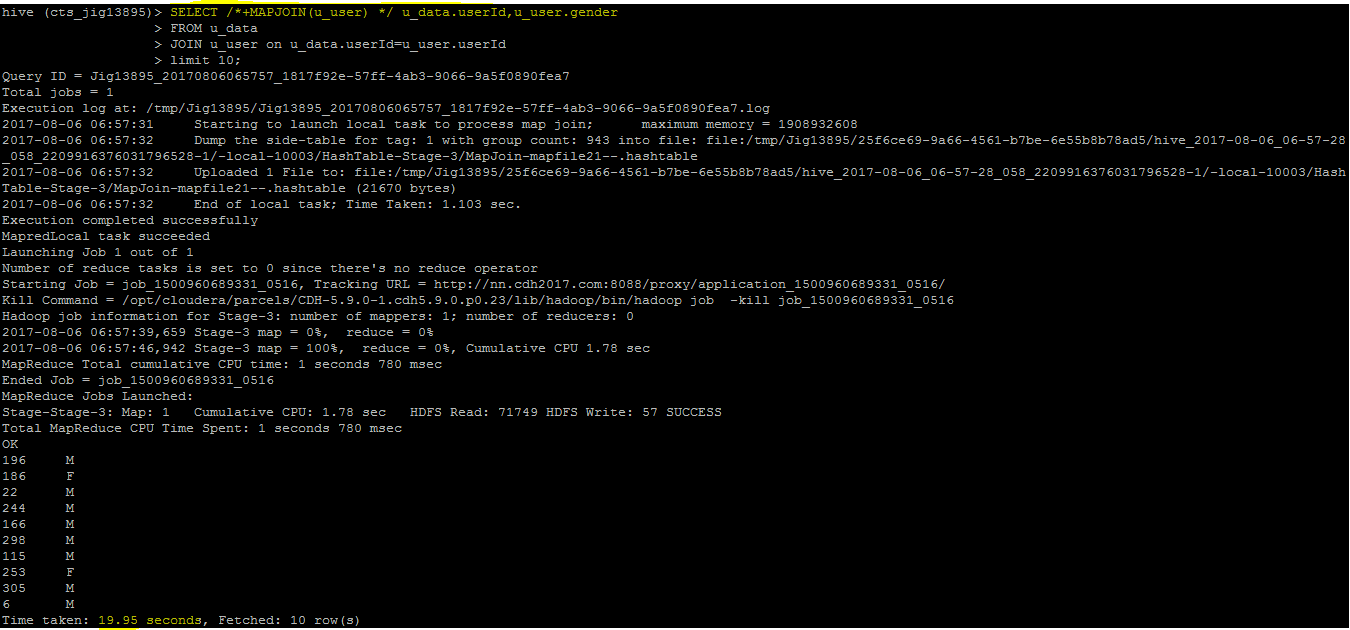
Output: **male -273; Female-670 ;**

**11. Join u\_data table and u\_user tables based on userid - Perform a reduce side join and map side join for the same and compare the time taken in both cases.**

**Map Side Join:**

SELECT /\*+MAPJOIN(u\_user) \*/ u\_data.userId,u\_user.gender

FROM u\_data

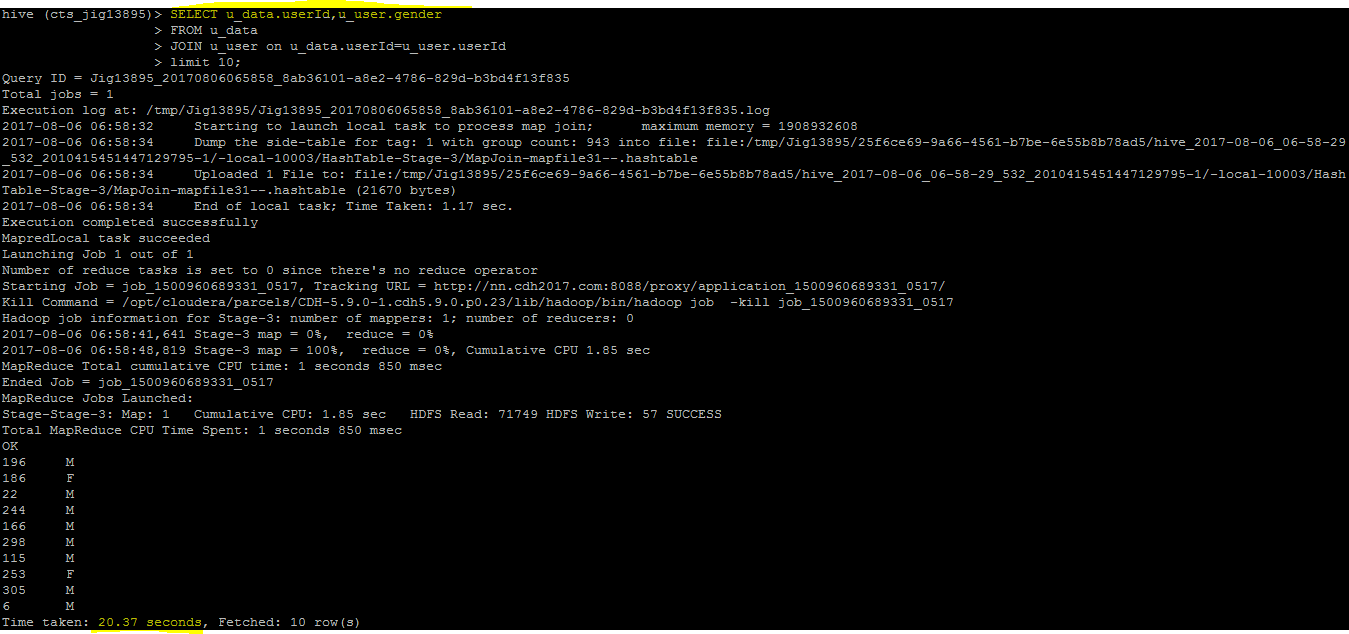
JOIN u\_user on u\_data.userId=u\_user.userId limit 10;

**Total seconds:19.95 seconds**

**Reduce Side Join:**

SELECT u\_data.userId,u\_user.gender

FROM u\_data

JOIN u\_user on u\_data.userId=u\_user.userId limit 10;

**Total time:20.37 seconds**

**Comparision of map side and reduce side join from the results**

🡪The map side join take less time to complete than the reduce side join

🡪So it can reduce the cost incurred at the shuffle stage and the reduce stage

**12. Create a partitioned table u\_user\_partitioned, partitioned by occupation column**

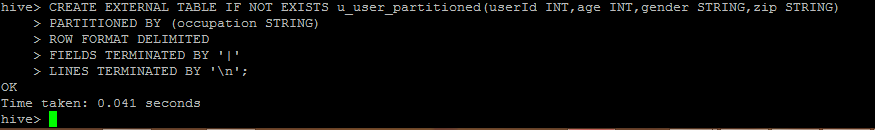
CREATE EXTERNAL TABLE IF NOT EXISTS u\_user\_partitioned(userId INT,age INT,gender STRING,zip STRING)

PARTITIONED BY (occupation STRING)

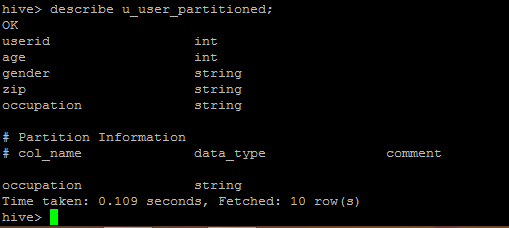
ROW FORMAT DELIMITED

FIELDS TERMINATED BY '|'

LINES TERMINATED BY '\n';



The table u\_user\_partitioned by the occupation column



To enable dynamic partition ,the dynamic partition mode parameter should be set to non strict



Insert the data into partition table from u\_user table

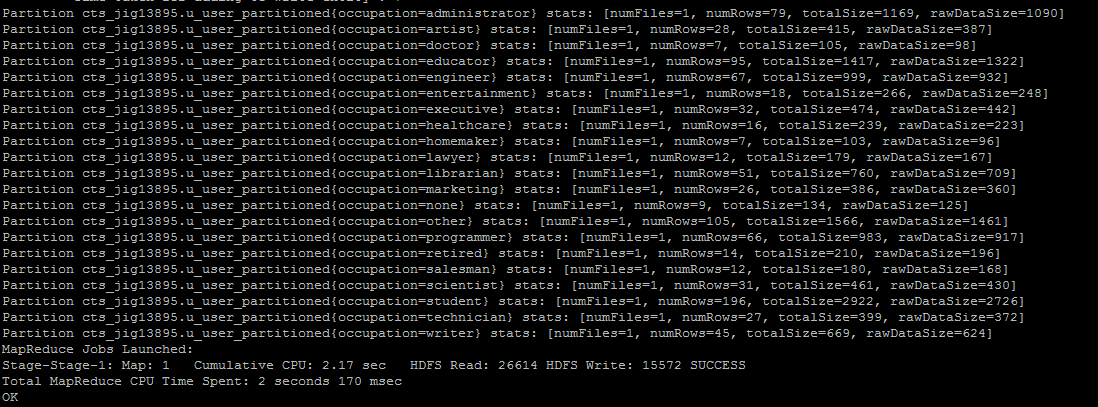
INSERT OVERWRITE TABLE u\_user\_partitioned

PARTITION(occupation)

SELECT userId,age,gender,zip,occupation

FROM u\_user;

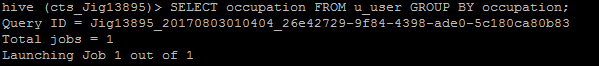




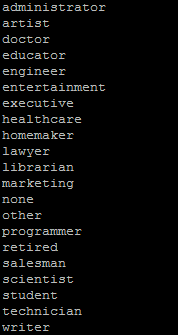
**13. Find out the total number of male and total number of female only for the most common occupation – you can hard code the occupation/ use subqueries. - Perform the query on both un-partitioned table and partitioned table. - Compare and report the performance differences.**

Select the distinct occupations

SELECT occupation FROM u\_user GROUP BY occupation;

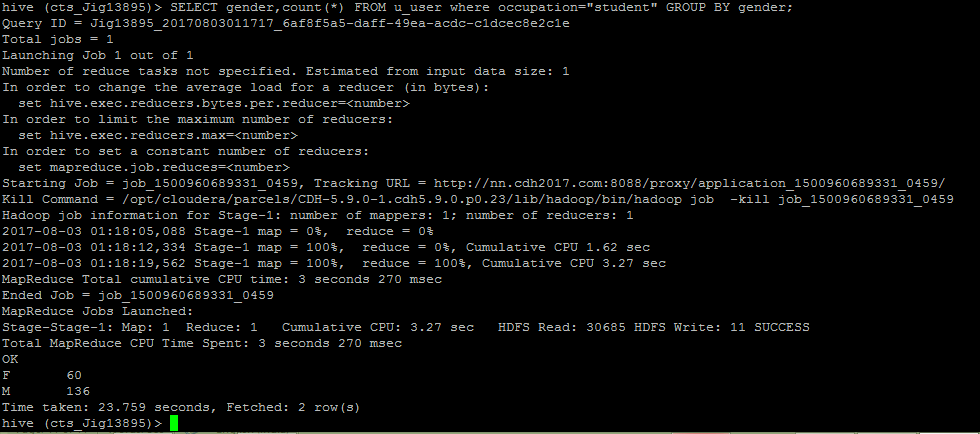


The distinct occupations are



The total number of male and total number of female for the occupation **student** from **u\_user** table

SELECT gender,count(\*) FROM u\_user where occupation="student" GROUP BY gender;



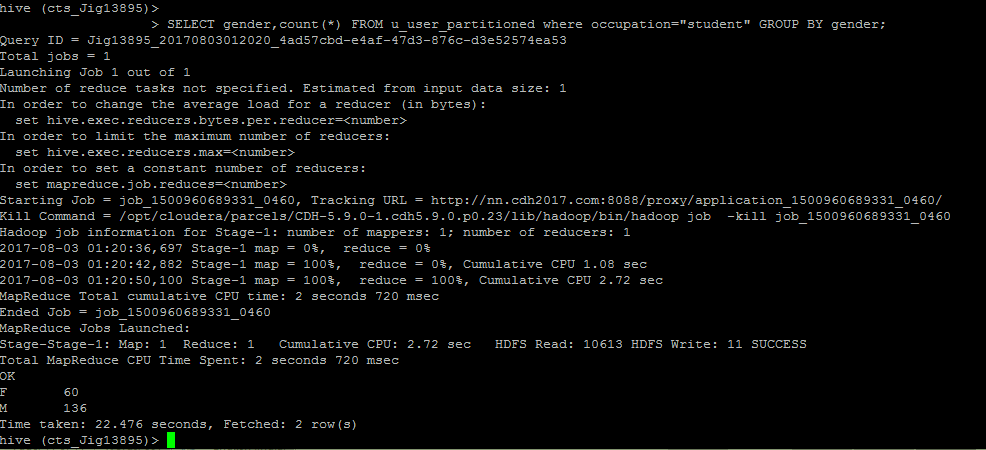
Total male=**136** ; Total female=**60**

Total:**196**

Time taken:**23.7 seconds**

The total number of male and total number of female for the occupation **student** from **u\_user\_partitioned** table

SELECT gender,count(\*) FROM u\_user\_partitioned where occupation="student" GROUP BY gender;



Total male=**136** ; Total female=**60**

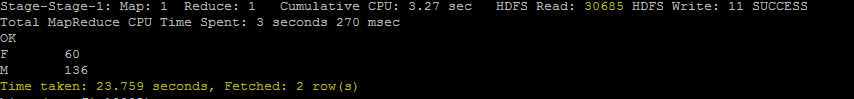
Total:**196**

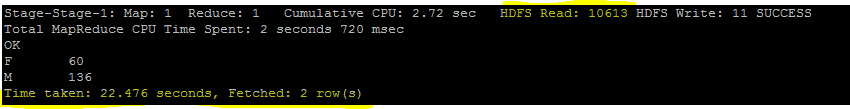
Time taken:**22.47 seconds**

Comparison of u\_user and u\_user\_partitioned tables from the results

🡪The time taken for the user\_partitioned table is comparatively less than the u\_user table

🡪The number of reads from HDFS is reduced in case of a select from u\_user\_partitioned table





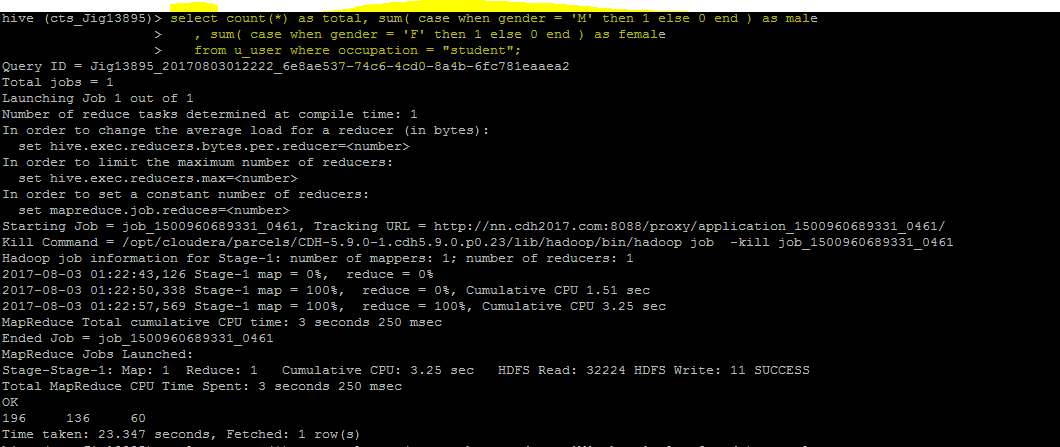
🡪In case of partitioned tables the reduced stage takes less time

**Using case statement:**

select count(\*) as total, sum( case when gender = 'M' then 1 else 0 end ) as male

, sum( case when gender = 'F' then 1 else 0 end ) as female

from u\_user where occupation = "student";

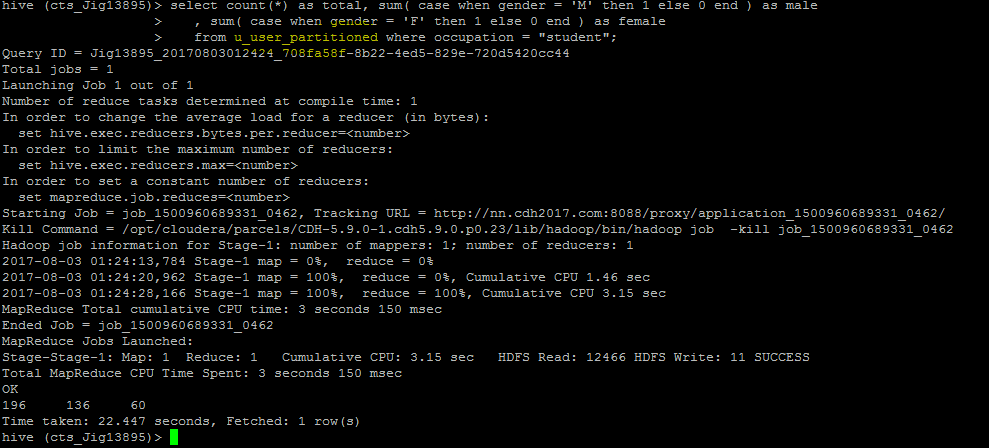


Time taken:23.3 seconds

select count(\*) as total, sum( case when gender = 'M' then 1 else 0 end ) as male

, sum( case when gender = 'F' then 1 else 0 end ) as female

from u\_user\_partitioned where occupation = "student";s



Time taken:22.44 seconds