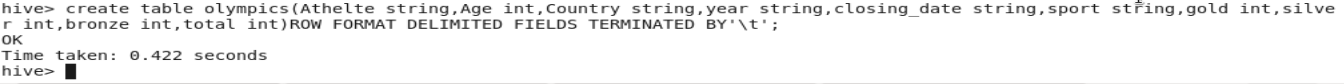
**Advanced Hive:**

**Task 1:**

**Create table Olympics with the columns as shown in the command below:-**

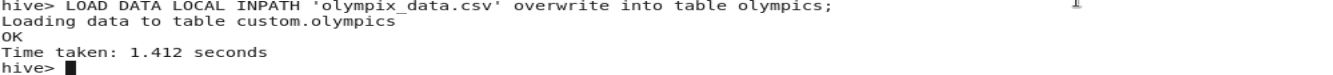
**create table olympics(Athelte string,Age int,Country string,year string,closing\_date string,sport string,gold int,silver int,bronze int,total int)**

**ROW FORMAT DELIMITED FIELDS TERMINATED BY'\t';**



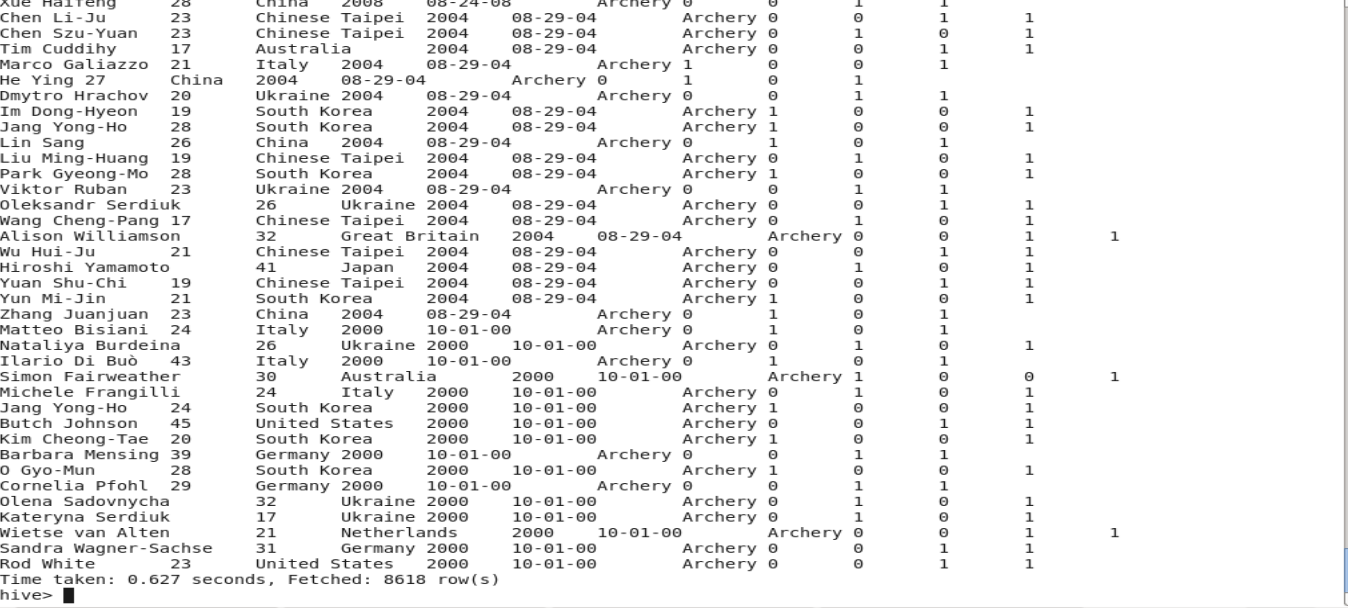
**Now load the table with the dataset ,this dataset should be present in /home/acadgild:-**

**LOAD DATA LOCAL INPATH 'olympics.csv' overwrite into table olympics;**



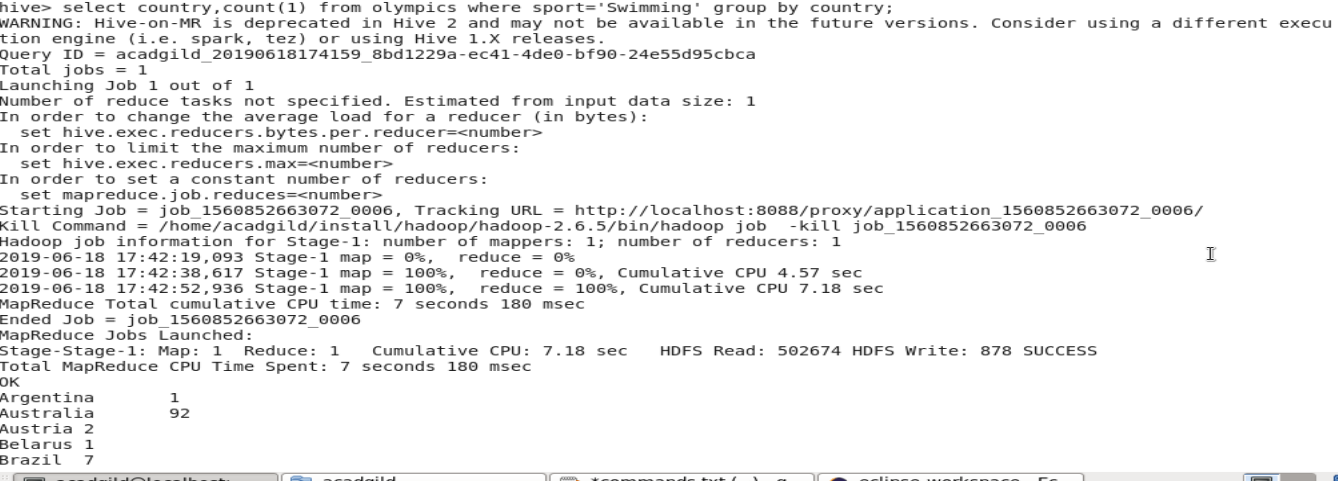
**Query the table to see loaded data:-**

**select \* from Olympics;**



**1. Write a Hive program to find the number of medals won by each country in swimming:-**

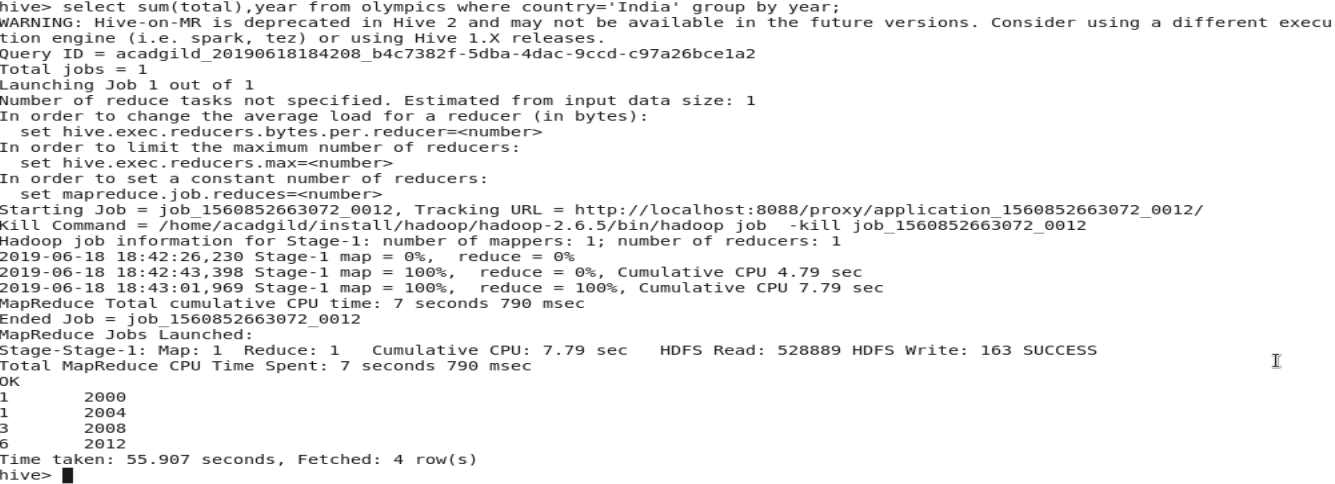
**select country,count(1) from olympics where sport='Swimming' group by country;**





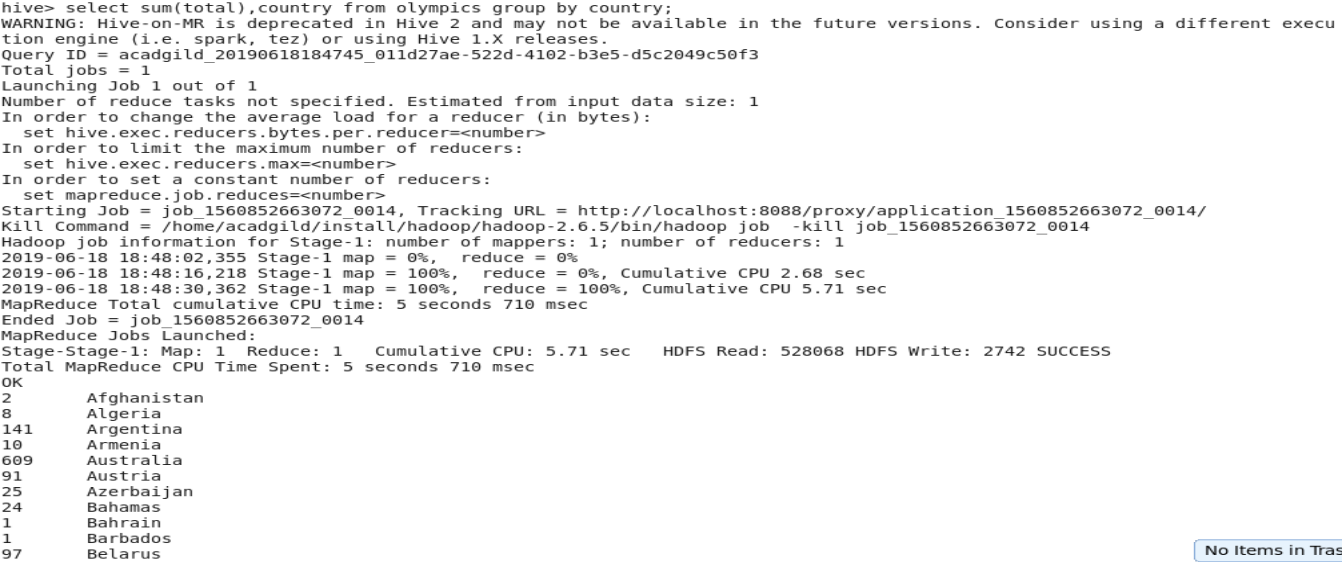
**2. Write a Hive program to find the number of medals that India won year wise.**

**select sum(total),year from olympics where country='India' group by year;**



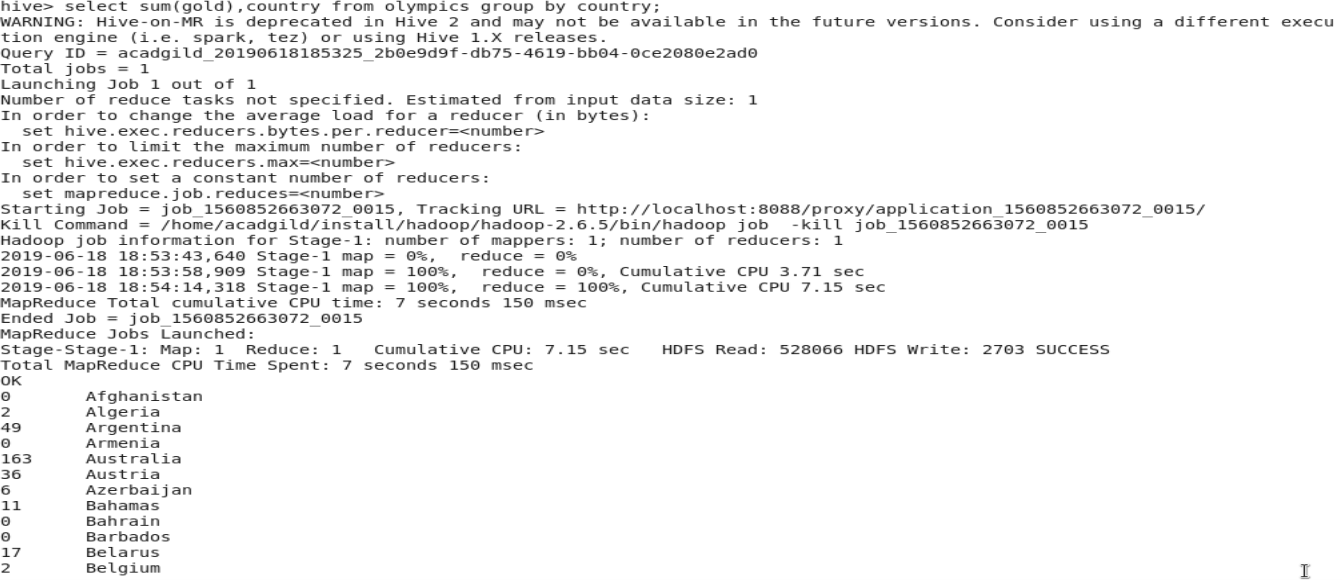
**3. Write a Hive Program to find the total number of medals each country won.**

**select sum(total),country from olympics group by country;**



**4. Write a Hive program to find the number of gold medals each country won.**

**select sum(gold),country from olympics group by country;**

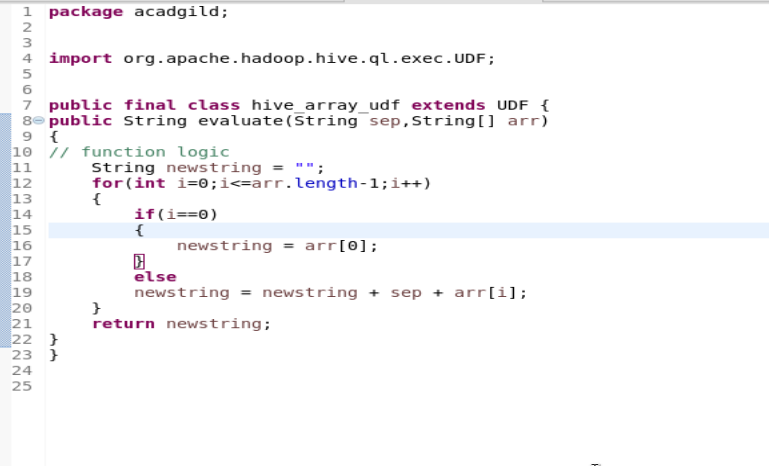


**Task 2**

**Create a java class with the logic of appending words given in the array with the separator,**

**Both separator and array are passed as arguments to the evaluate function.**

Below is the screenshot of the java class written in Eclipse .This is tested using a main class then only its JAR was created in the next step.



**Create the JAR file of the class at /home/acadgild**

**Now add the JAR file to HIVE:-**

**ADD JAR /home/acadgild/hive\_array\_udf.jar**



**Once added then list the jars to see the JAR and the path:-**

Command: **list jar;**



**Now create a temporary function**

**CREATE TEMPORARY FUNCTION hive\_array\_udf AS ‘acadgild.hive\_array\_udf’;**

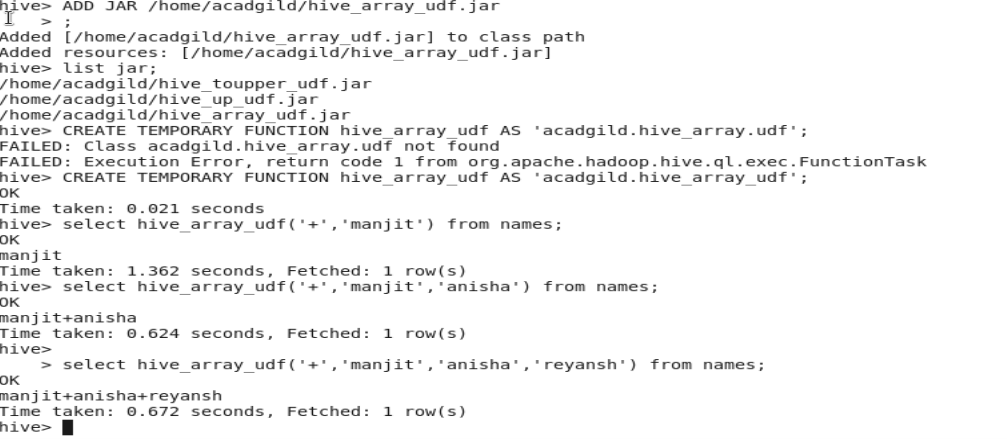


Where hive\_array\_udf is thefunction name which will refer to the function of the JAR file with the same name under the package acadgild.

Now call the function of the java using select statement. Here we are using name table the column of which we are not referring,we are just passing the separator and array values.

**Select hive\_array\_udf(‘+’,’manjit’,’anisha’);**

**Select hive\_array\_udf(‘+’,’manjit’,’anisha’,’reyansh’);**



**TASK 3:**

**Create the table student with the columns as shown in the command below:-**

**CREATE TABLE student(std\_id int,std\_name string,std\_loc string) clustered by (std\_id) into 5 buckets stored as orc TBLPROPERTIES('transactional'='true');**

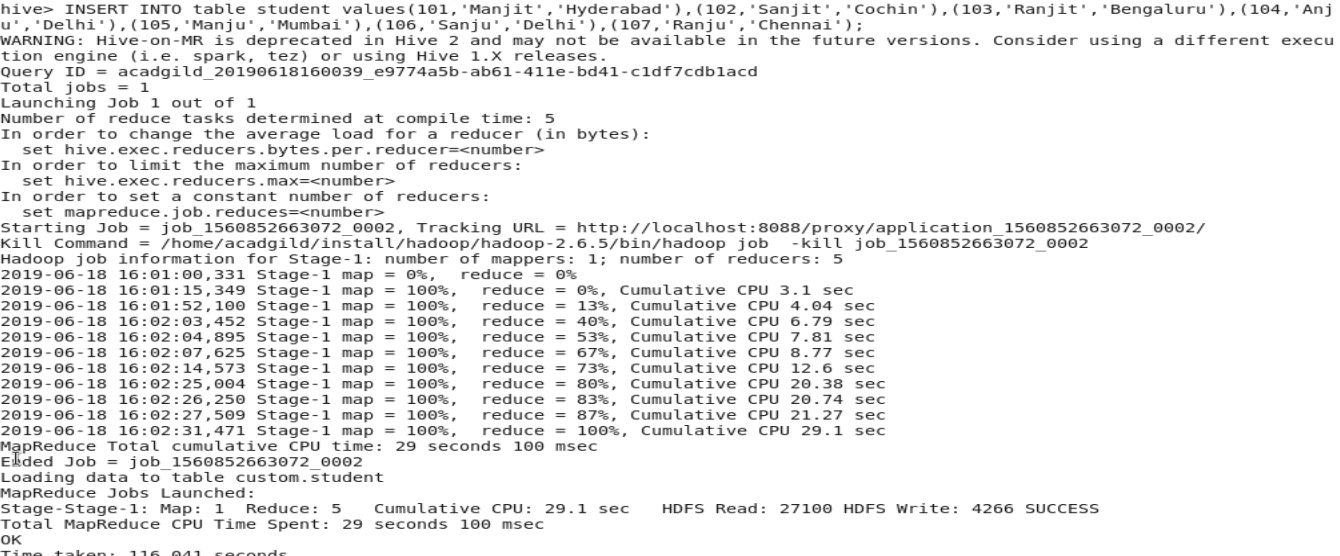
**Here we are clustering on the std\_id and using 5 buckets and file format as ORC.**



**Now insert records into the table student:-**

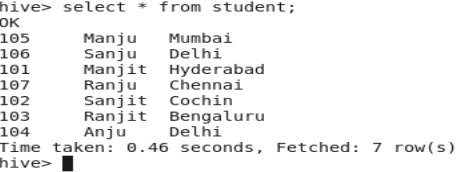
**INSERT INTO table student values(101,’Manjit’,’Hyderabad’),(102,’Sanjit’,’Cochin’),(103,’Ranjit’,’Bengaluru’),**

**(104,’Anju’,’Delhi’),(105,’Manju’,’Mumbai’),(106,’Sanju’,’Delhi’),(107,’Ranju’,’Chennai’);**



**Now check whether the records got inserted into the student table or not using the below command:-**

**Select \* from student;**



**Now update the bucketing column of the table:-**

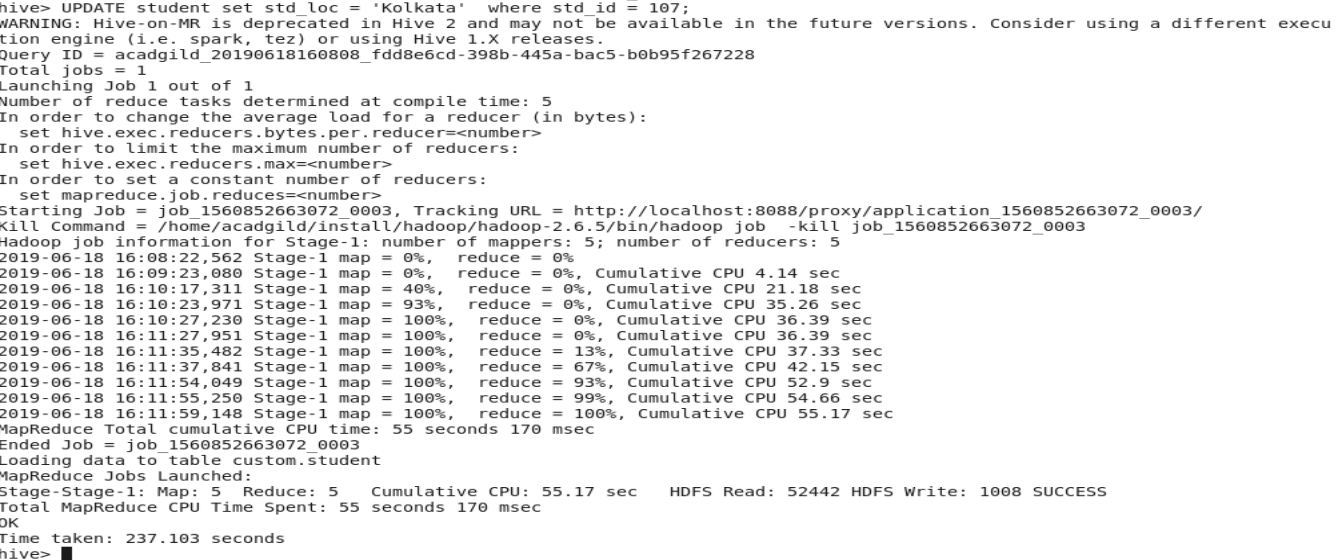
**UPDATE student set std\_id=108 where std\_id=107;**

**We get an error here saying that bucketing column can’t be updated.**



**Now update the std\_loc of a record:-**

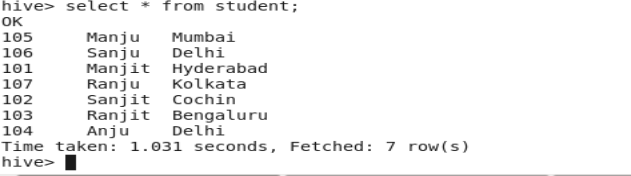
**Update student set std\_loc=’Kolkata’ where std\_id=107;**



**Check the updated record:-**

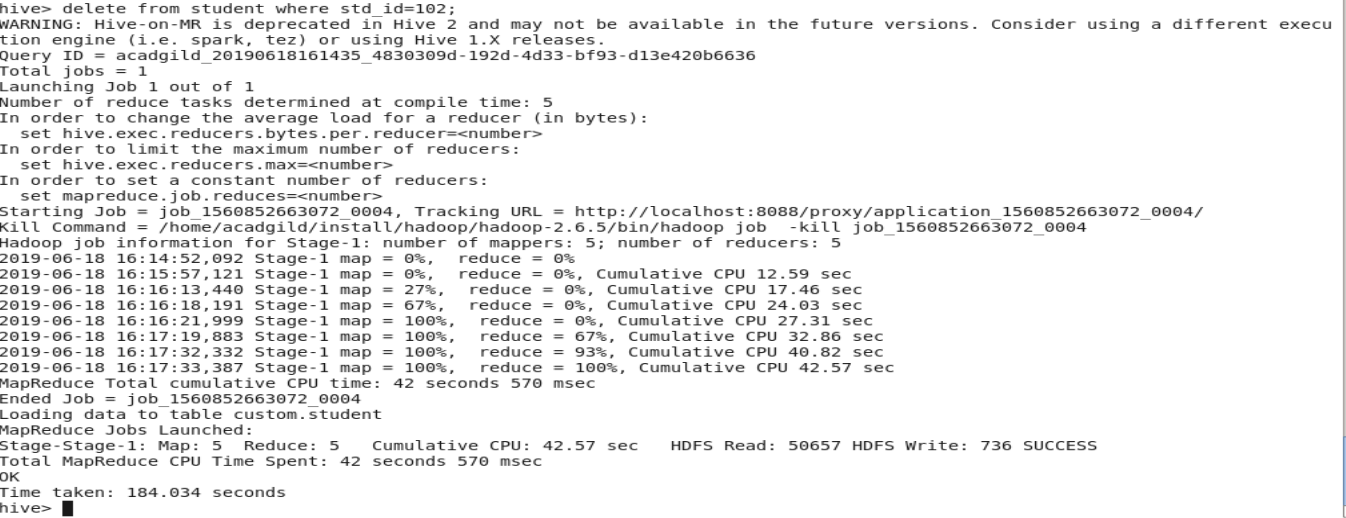
**Here the std\_loc of the student with std\_id 107 is changed to ‘Kolkata’.**

**Select \* from student:-**



**Now lets delete a records where the std\_id=102.**

**Delete from student where std\_id=102;**



**Now check if the record got deleted or not :-**

**Select \* from student;**

