DATA INGESGTION AND PIPELINE ANALYSIS WITHIN MySQL, HDFS AND HIVE.

COURSE: BDM 1024

Group Members

Mahima Akula

Modupeola Omodunni Oyatokun

Jumoke Yekeen

Chibuike Okoroama

Diksha

Harish Kundal

SUBMITTED ON 29-07-2023

TABLE OF CONTENTS:

1. INTRODUCTION:

1.1. PROJECT OVERVIEW:

For this project (BDM 1024 project), sales_data.csv file was cleaned, data type formats were set, additional unique ID column was added, and we made sure that the data is in .csv format before we ingested into cloudera. It was important that we do not tamper with the data so that meaningful insights are not lost as result of such action.

1.2. DATA INGESTION AND PREPARATION:

The challenge was to ingest a .csv file into cloudera local drive, analyze it in Hive and store the analyzed results in MySQL. It is expected that a minimum of five queries to analyze the data is expected. The data to be ingested into cloudera was in the local drive of computer, therefore we needed to transfer it into cloudera.

To transfer the data into the HDFS, we used FileZilla to connect to cloudera local drive and then transfer the file into cloudera local drive. Figure 1 below shows sales_data.csv located in the local cloudera drive. By simply using the Linux command *ls* we visualized different files including the file that will be analyzed.

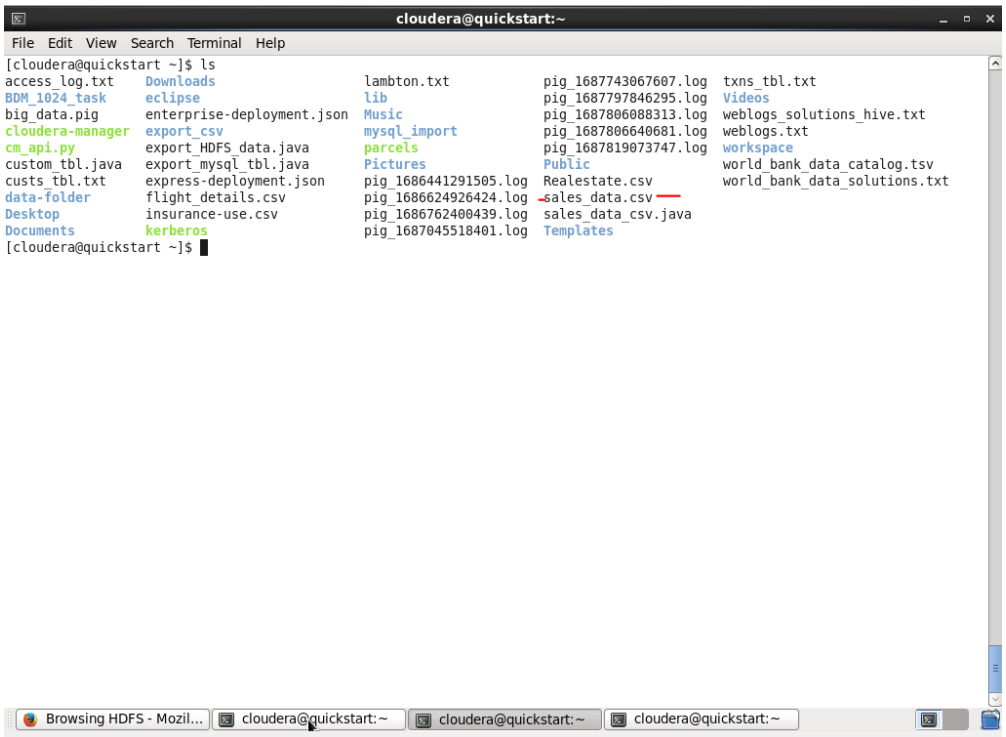


Figure 1. show sale_data.csv in the cloudera local drive

Now that we have our data, sales_data.csv in the local drive, next thing is to display the data in the terminal. In our project we used head -n 5 to display few results but we wanted to see the entire

data. Figure 2 below shows a display of sale_data.csv data content. This way we confirmed that the right data was transferred. However, we could not do any further analysis on the data than display it on the terminal. Therefore, the challenge of analyzing the data in Hive and storing it in MySQL persists.

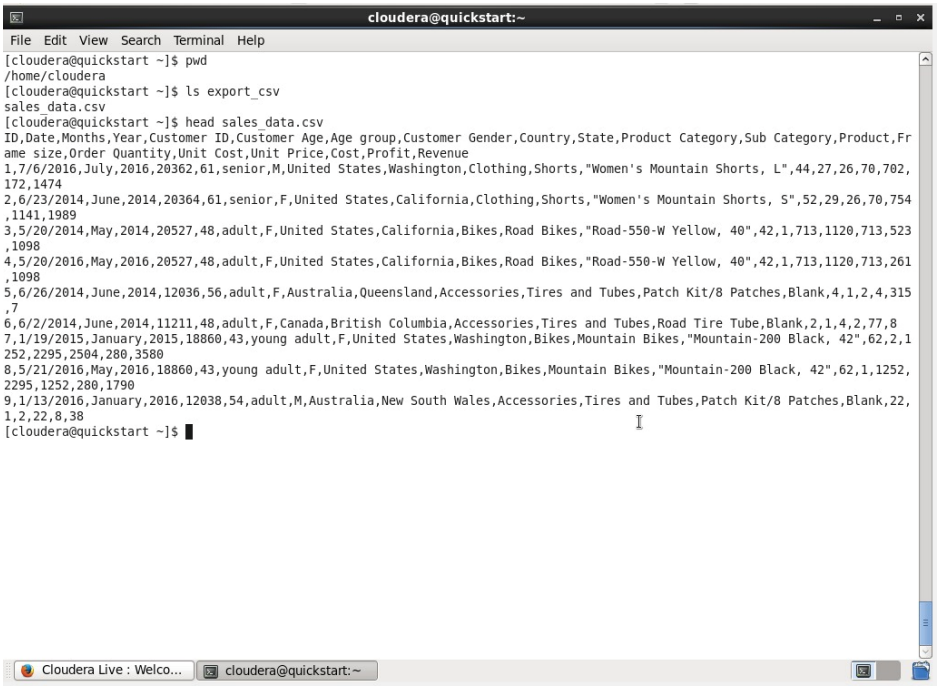


Figure 2 shows display of sale_data.csv on the terminal

2. DATA TRANSFER:

2.1. CREATING DATABASE IN MySQL:

According to the project task, it is mandatory that we ingest the sale_data.csv into a table in MySQL database in cloudera, therefore we had to confirm that the was no replica table in the MySQL database. Figure 3 shows that we log into MySQL with error in cloudera by typing MySQL -u root -p in the cloudera terminal. With MySQL syntax show databases, figure 4 shows the different databases present in my MySQL database. On this note we must create a new database with a new table that is with the same schema as the sale_data.csv file before we can ingest it into the MySQL database.

2.2. CREATING TABLE IN MySQL:

Figure 4 shows the creation of a database named project_1024 in MySQL. A table with similar .csv schema was created, then described to confirm that the schema matches properly. As mentioned in introduction, the ID column added to the .csv data was done to provide uniqueness to the data. So, the ID was made the primary key instead.

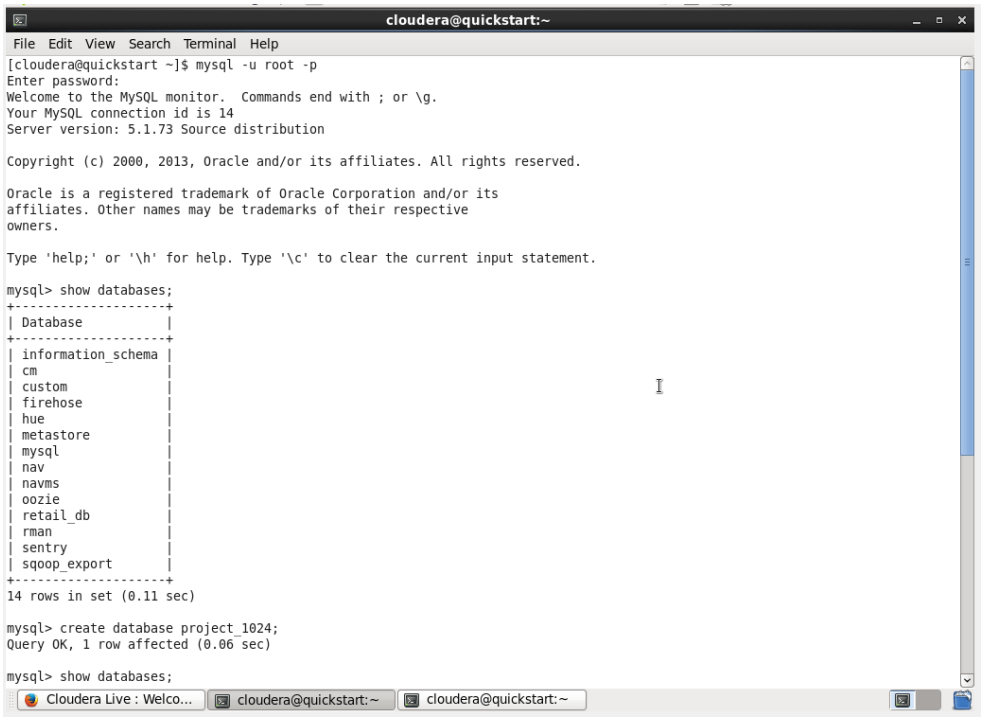


Figure 3.

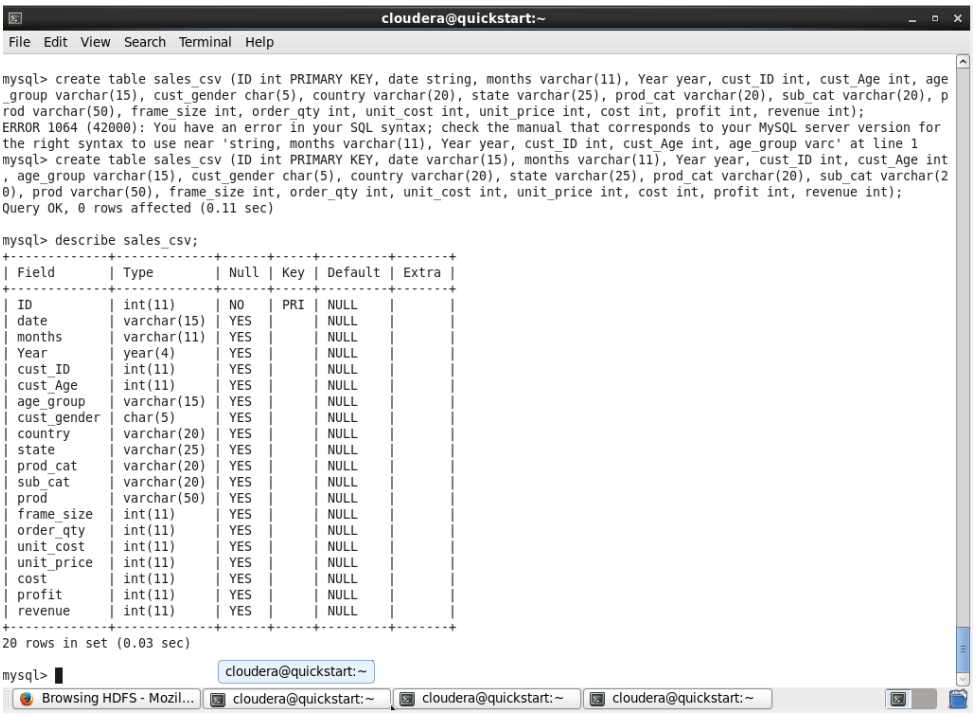


Figure 4 shows MySQL table create and schema description.

After creating the sale_csv table in MySQL, we ingested the sale_data.csv date into the table from the MySQL terminal simply by using the load data infile location of data syntax as shown in figure 5. A total of 113036 rows were imported which is the same as .csv.

2.3. RESULTS OF UPLOADING DATA INTO MYSQL:

Therefore, our data is completely ingested into MySQL sales_csv table in project_1024 database. With the select * from the sales_csv table we can see that our data has been successfully ingested.

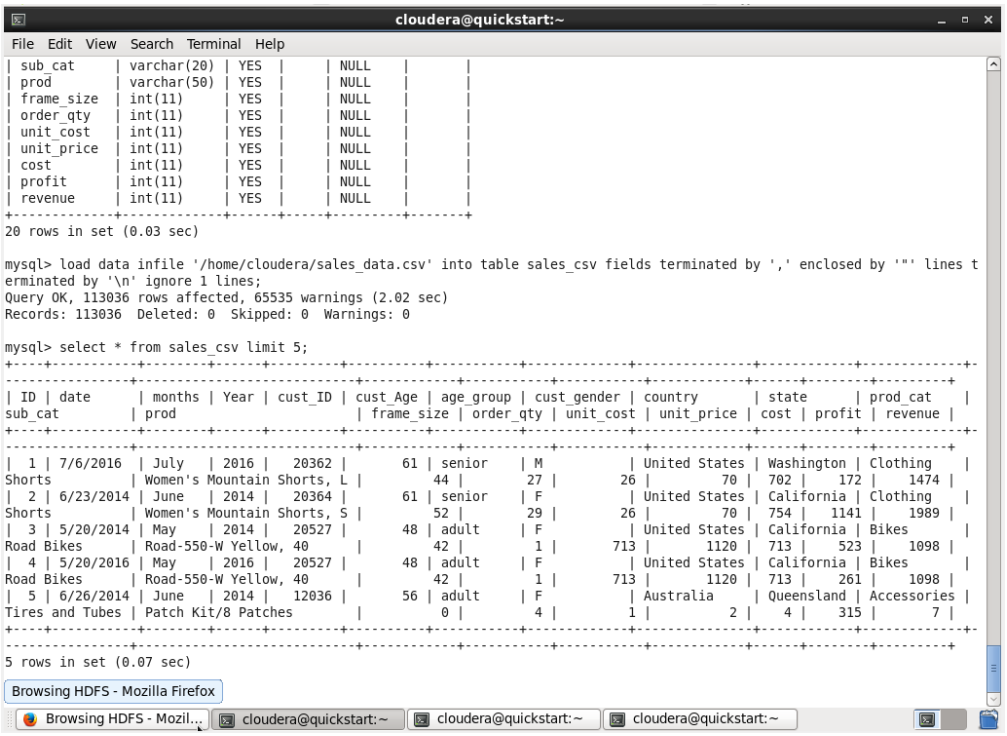


Figure 5 shows ingestion of csv file into MySQL and display of data.

3. EXPORTING DATA FROM MySQL TO HDFS USING SQOOP:

The uniqueness of this project is that we can move that between frameworks in Hadoop. However, the challenge of analyzing our data in Hive persists. To resolve this challenge, we must import our data into HDFS in cloudera then we can ingest the data into Hive. To achieve this, we must make sure make sure that we can transfer the entire MySQL table into HDFS, and we have created a database and exact same table schema in Hive for data ingestion.

3.1. TRANSFERRING DATA FROM MYSQL TO HDFS USING SQOOP:

Sqoop, which is a tool for importing and exporting bulk data between Hadoop and external data storages, we used it to transfer sales_csv into HDFS by using the Sqoop import –connect as shown in figure 6 below. Common mistakes made is to run this syntax on the same terminal as MySQL, however it is important we do it on a different cloudera HDFS terminal other than the MySQL terminal. The Sqoop import was split into two with the use ID in the data, which was made primary key, as highlighted in the syntax in figure 6, the data was successfully ingested into the HDFS. From HDFS, we can transfer the data into Hive for analysis. The ability to move data between framework HDFS and HDFS is why Sqoop is important for our structured data.

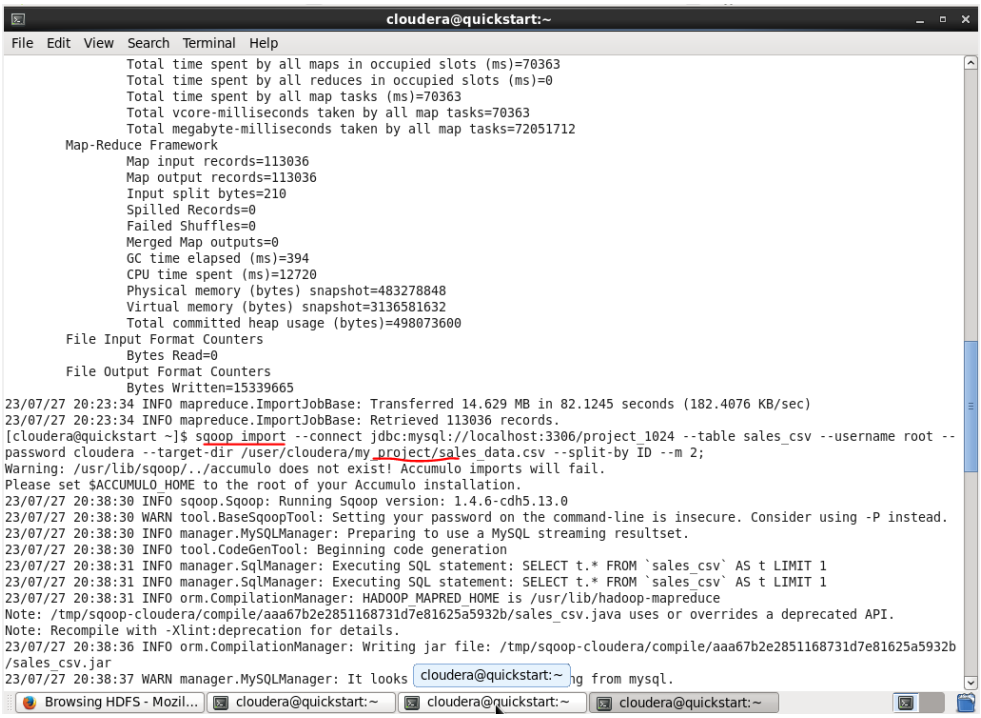


Figure 6 shows the use of Sqoop to import data into HDFS.

3.2. DATA INGESTION INTO HIVE:

From HDFS we ingested the data into Hive for further analysis. To achieve this, we used the load data inpath to ingest into the Hive. For sure, a table of similar schema was created before the ingestion of data from HDFS to Hive. Figure 7 shows the syntax used for ingestion.

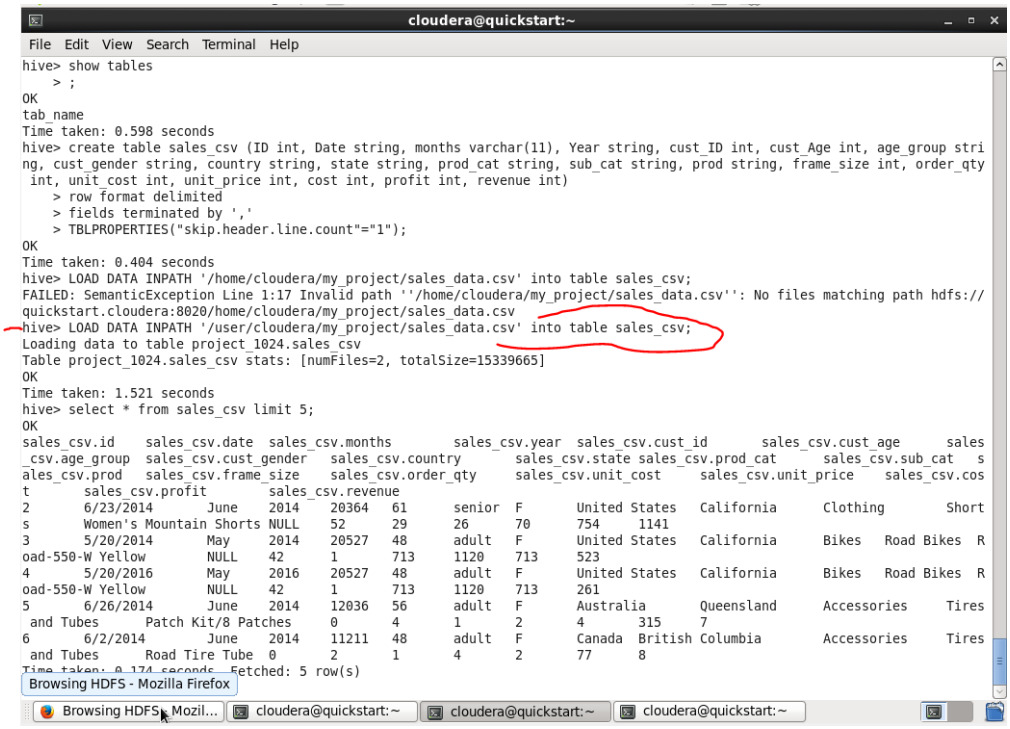


Figure 7 shows loading of data from HDFS into Hive

Currently we have successfully ingested our data into Hive for analysis. Here we will derive a few insights from our data. The purpose is to understand our data better. Furthermore, after generating results from Hive queries, we must store them and export the results into MySQL. The importance of this is to understand how to ingest data between HDFS frameworks.

So, to export the query results from hive to MySQL, then we must create individual result tables with the schema matching. Once the result is viewed, we create a table for it and store the result inside the table. The result table is different from the entire data table as shown in figure 9. To store the result differently in Hive, we used the syntax in figure 10.

4. DATA ANALYSIS USING HIVE:

After demonstrating how the data was transferred between Hive and MySQL, we will explain the insights derived from the queries.

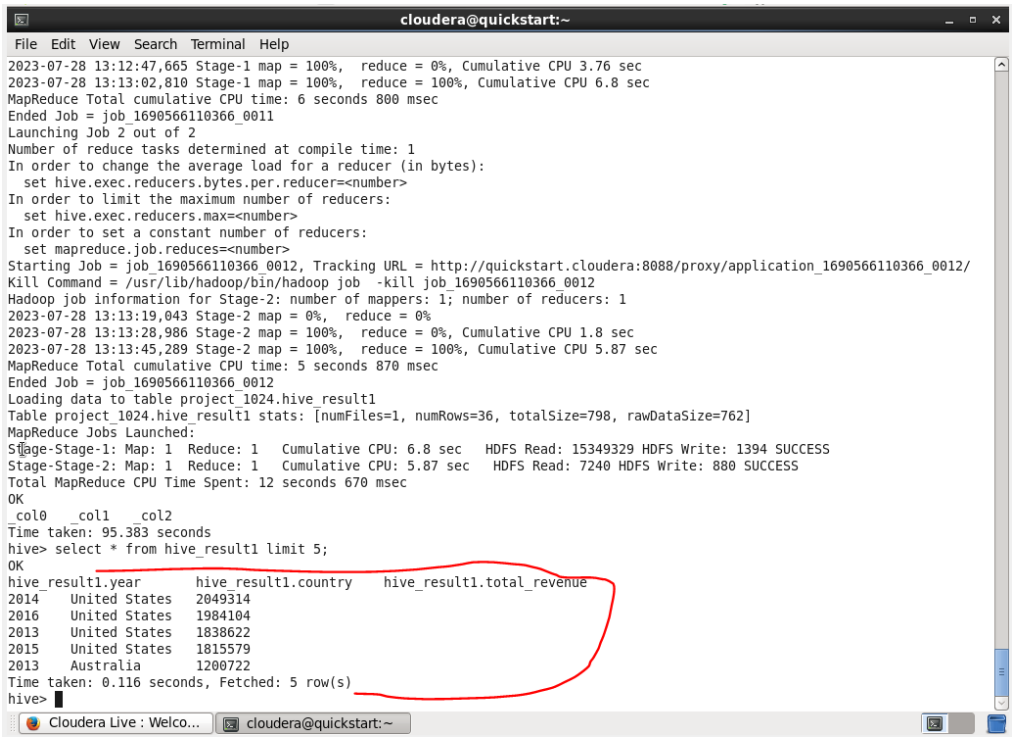


Figure 8 shows the query result.

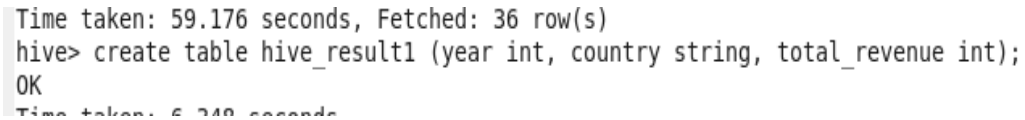


Figure 9 shows the creation of query result table for storing the result in Hive.

4.1. EXPORTING HIVE QUERY RESULTS TO HIVE RESULT TABLES:

We have created 5 different tables using hive query and inserting the data into those tables.

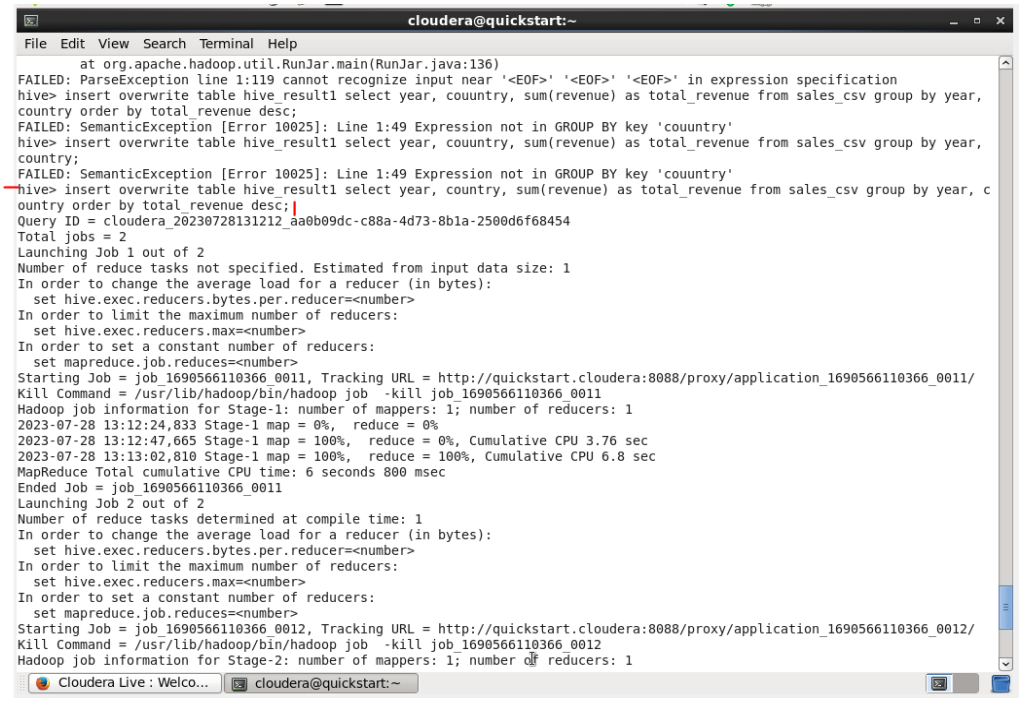


Figure 10 show syntax for inserting query result into a separate table

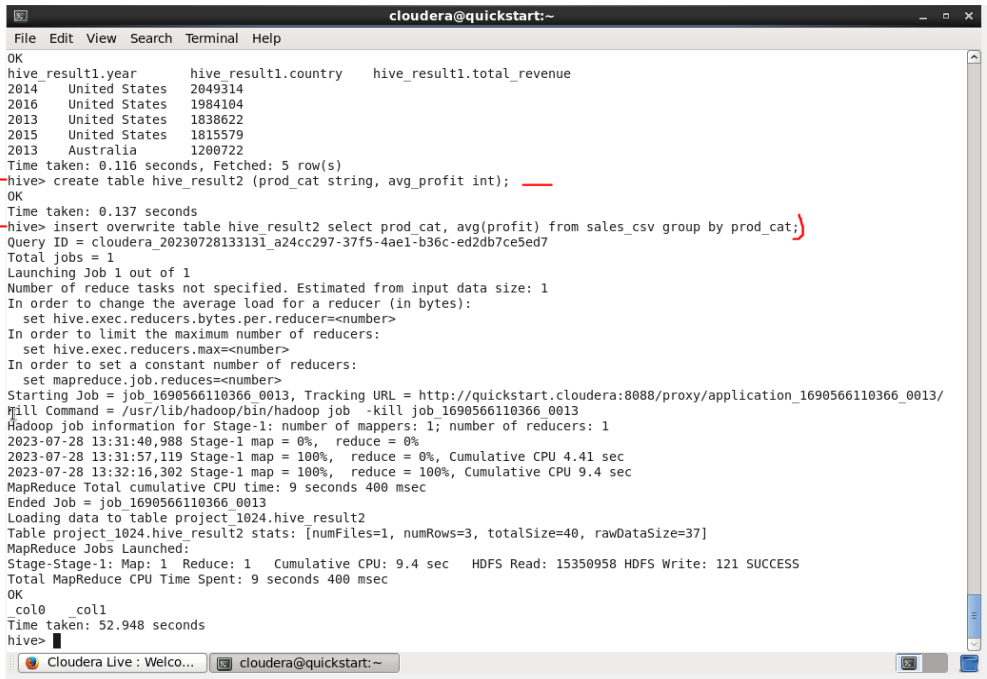


Figure 11 shows creation of hive_result2 and insert query result into it.

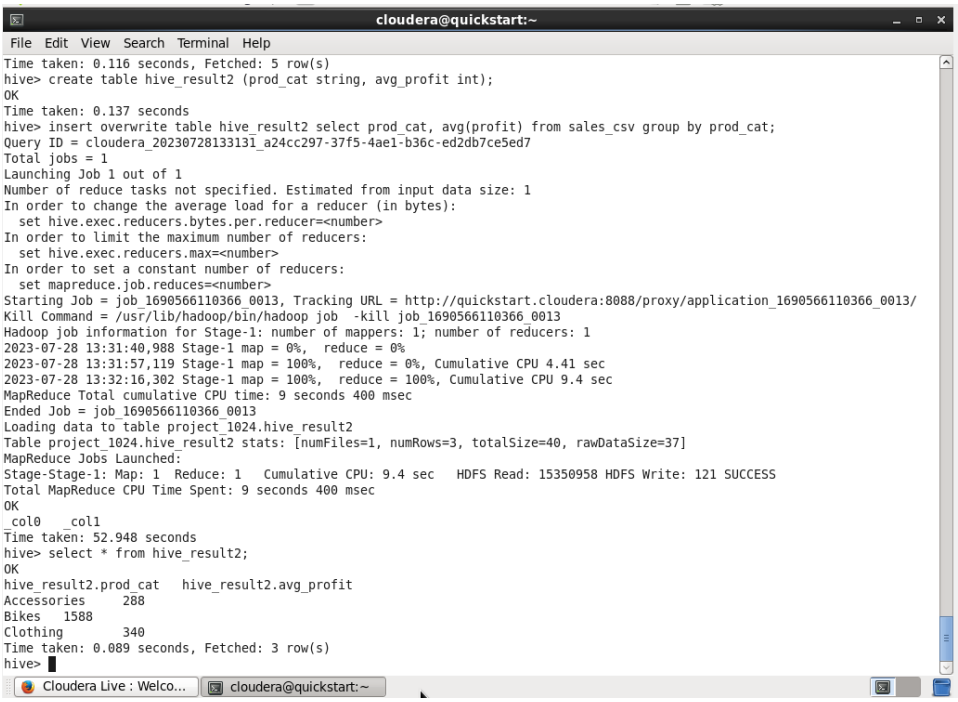


Figure 12 shows result insertion into hive_result2 and result is displayed.

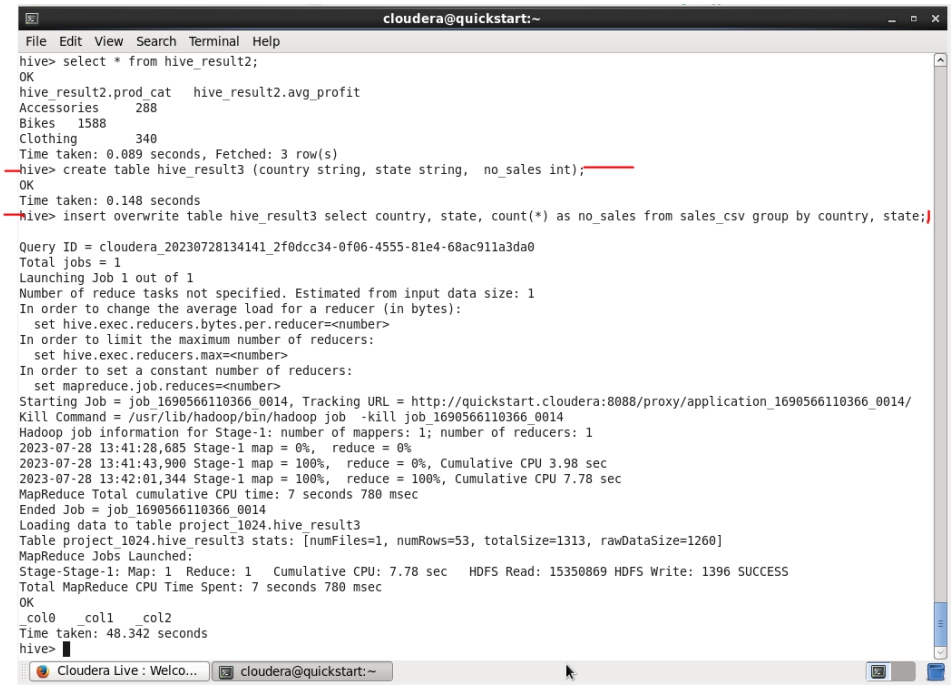


Figure 13 shows creation hive_result3 and insertion of result query syntax

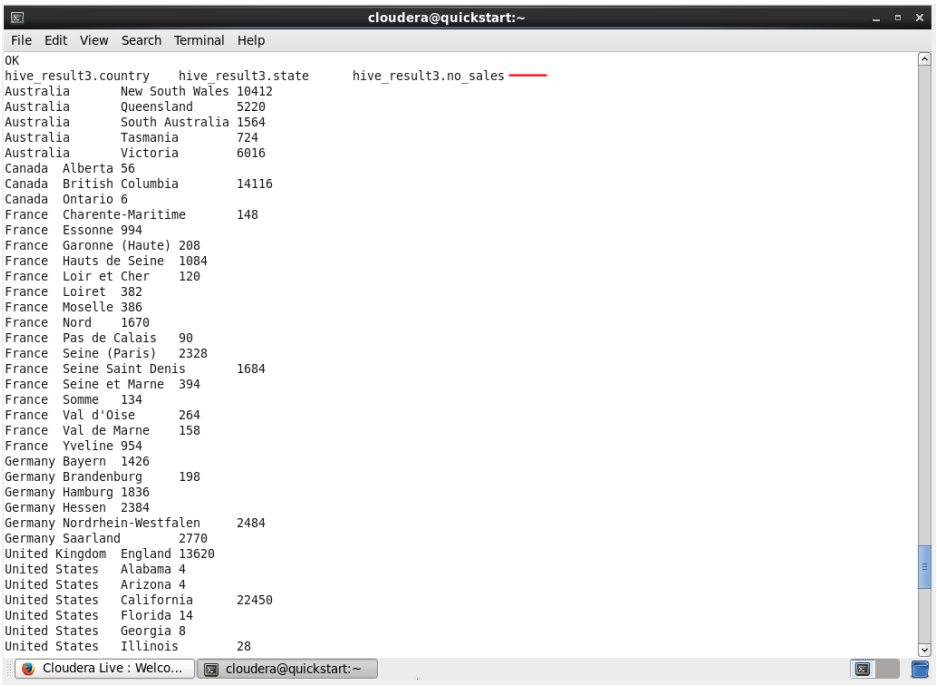


Figure 14 shows the display of hive_result3 table content.

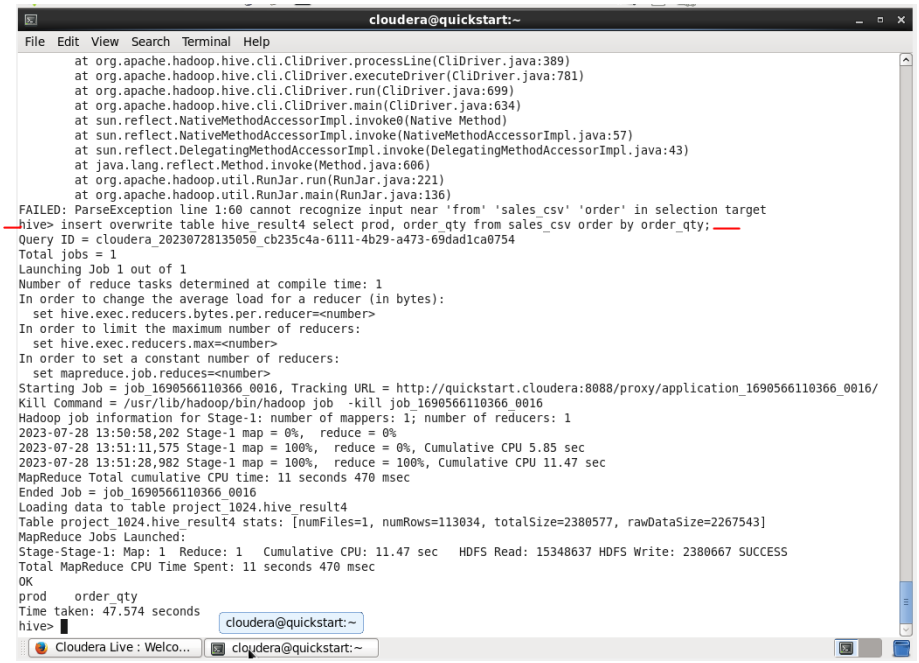


Figure 15 shows creation hive_result4 and insertion of result query syntax

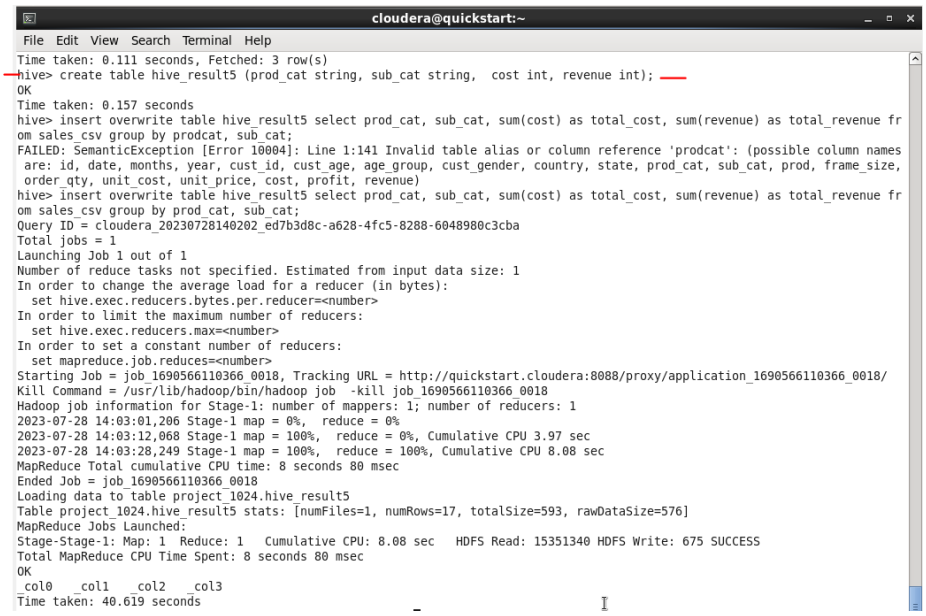


Figure 16 shows creation hive_result5 and insertion of result query syntax.

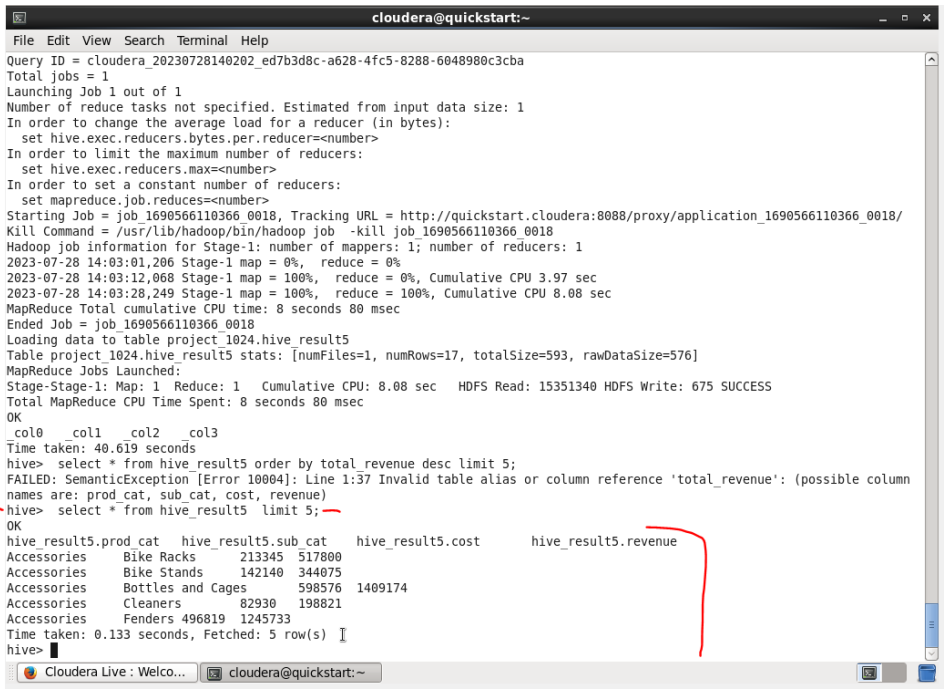


Figure 16 shows creation hive_result5 results.

Previously we have been saving our different Hive results by *inserting overwrite*-- syntax as shown in the above figures because we are preparing them for export to MySQL. Remember we are dealing with structured data, therefore the table schemas between origin and destination must be the same. Here the data is cleaned further by removing the pipe symbol before saving it into HDFS.

In our project we created hive_result_db database and created respective result tables with exact matching hive schemas in the above-mentioned database. Since we are moving data between HDFS frameworks, this means we must save (dump) individual results into HDFS first, then from there we export to MySQL.

Note, before exporting to MySQL, we had to save into HDFS using the *insert overwrite directory to save* in HDFS. Figure 17 shows how we exported our Hive data (hive_result1) to MySQL (result1). The consequent figures will show the syntax for saving the Hive result into HDFS.

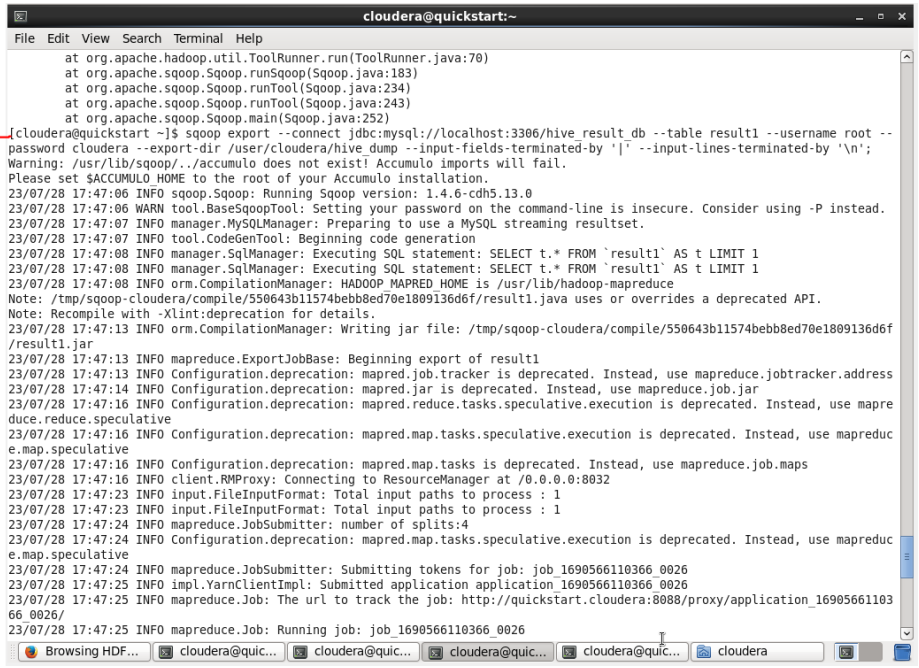


Figure 17 shows export syntax to MySQL specified DB and table.

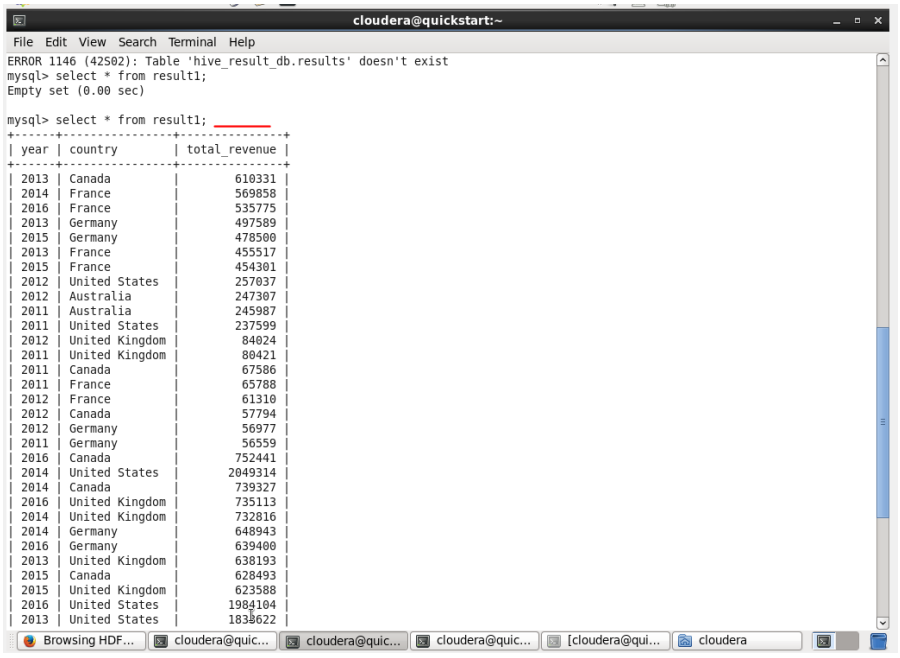


Figure 18 shows the result exported Into MySQL result1 table.

4.2. EXPORTING DATA FROM HIVE TO MySQL:

Like we mentioned before, the insert syntax for saving Hive results into HDFS is clearly shown in figure 19. While selecting all the results in Hive_result table, it also removes the Pipe '|' delimiter then save the data in HDFS. This makes it easier for each table to be exported into MySQL.

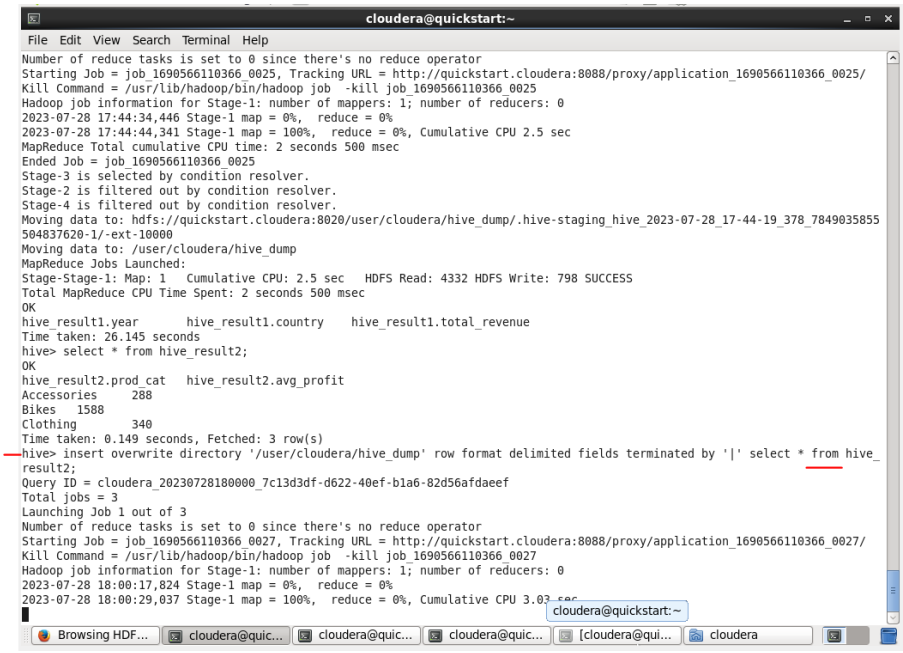


Figure 19 shows syntax for saving Hive_result2 data into HDFS.

For table result2 as an example, we could see that in figure 19, we saved the hive data in HDFS, in figure 20, we exported hive result into respective MySQL table and made sure that the appropriate delimiters are specified.

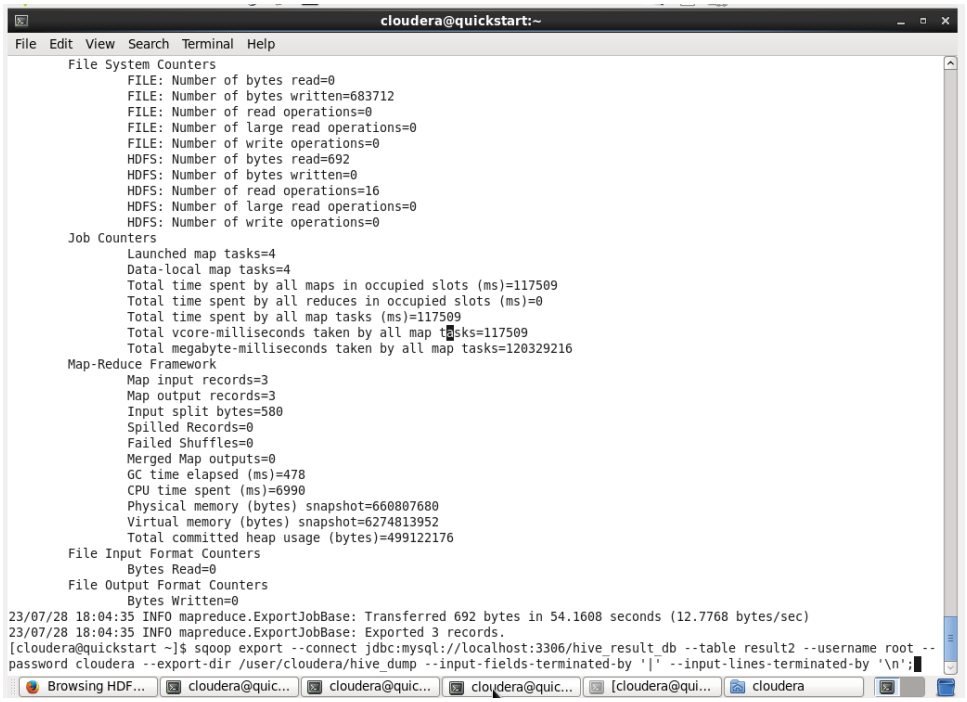


Figure 20 shows the syntax for exporting data into MySQL.

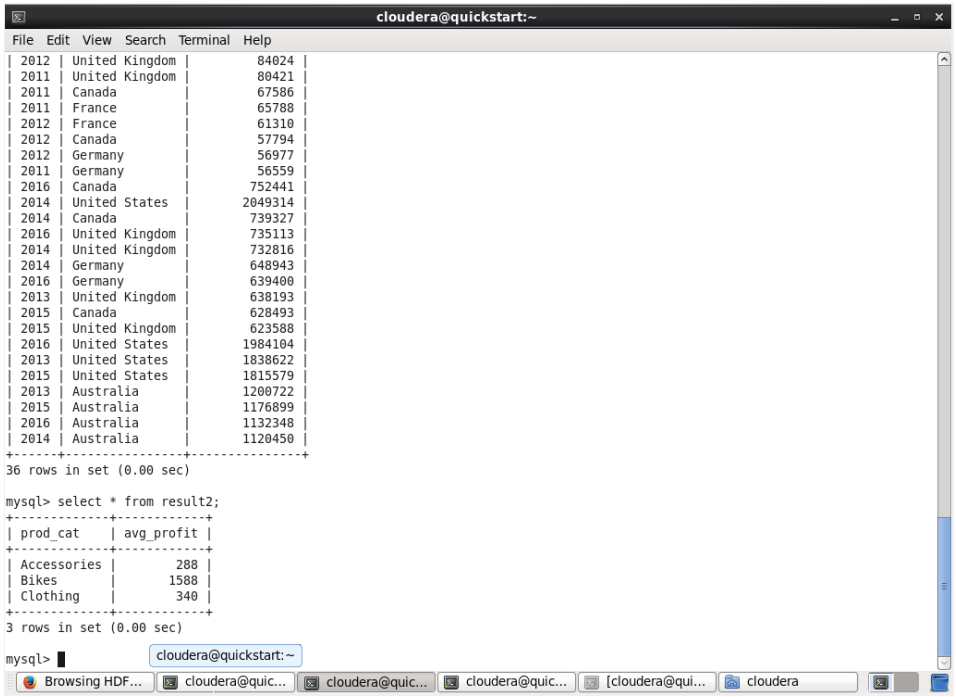


Figure 21 shows the result of MySQL data which matches Hive_result.

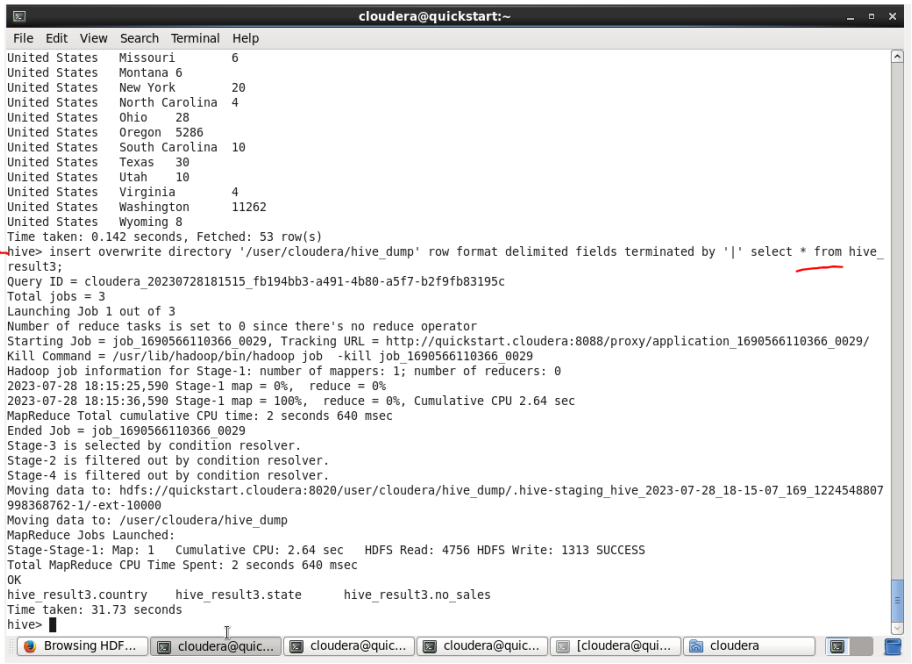


Figure 22 shows syntax for saving hive_result3 data into HDFS.

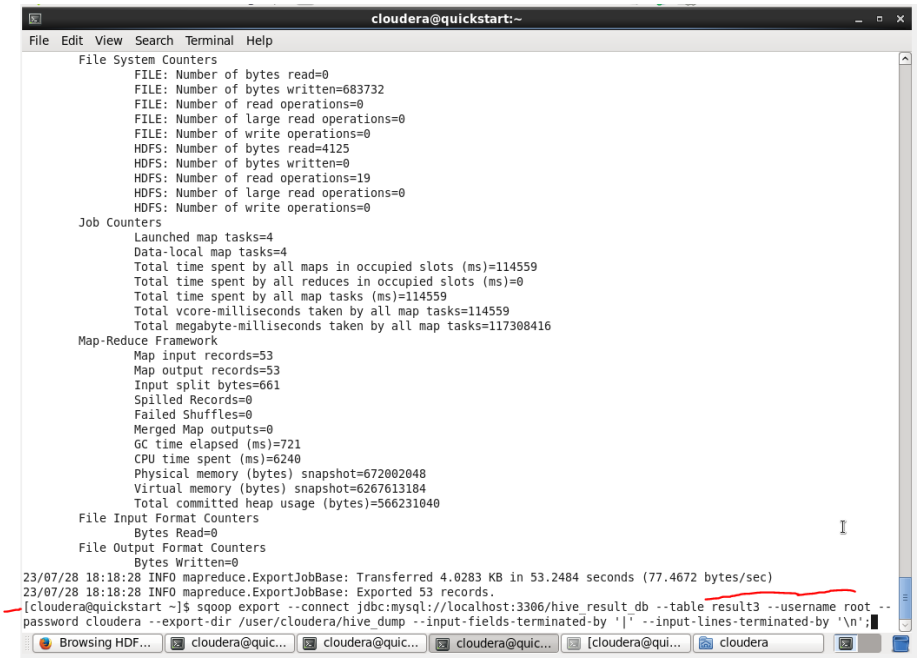


Figure 23 syntax shows sucessful export of data into MySQL result3 table.

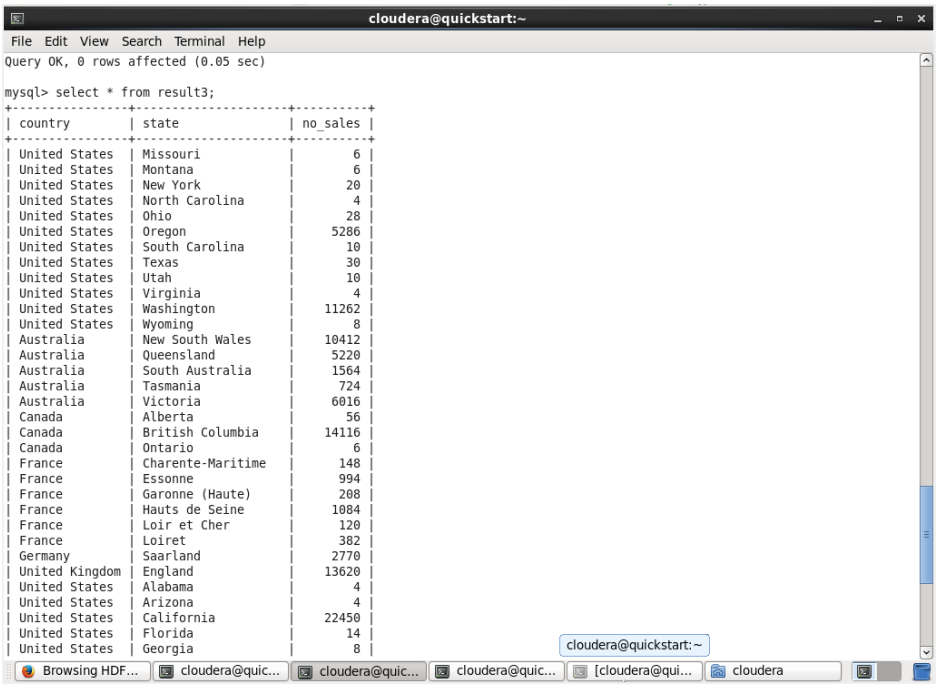


Figure 24 shows result of MySQL result3 table successful transferred data

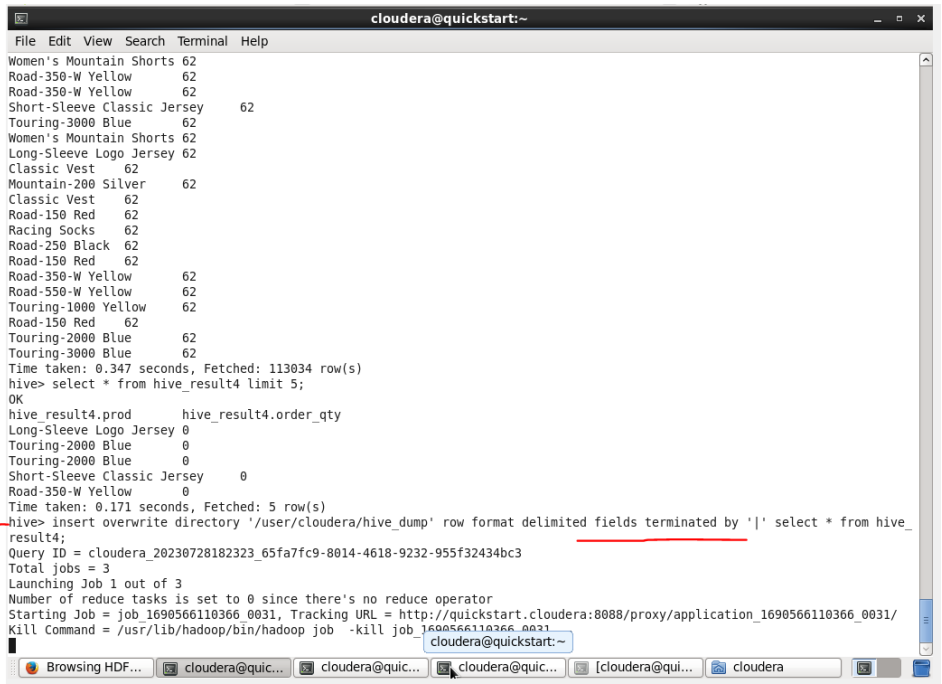


Figure 25 shows the syntax for saving Hive_result4 into HDFS.

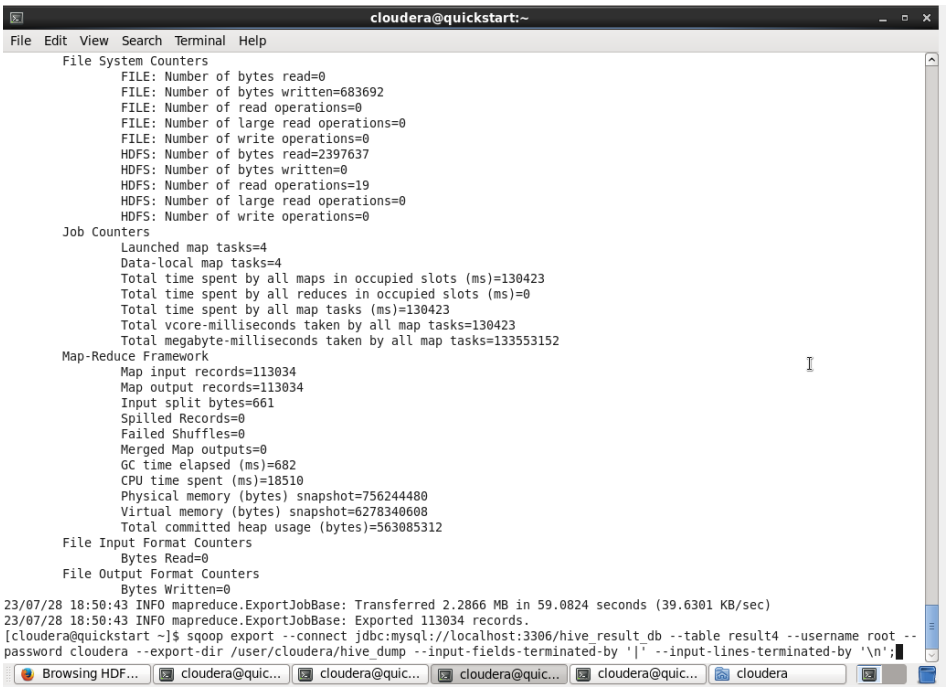


Figure 26 shows successful export of hive_result4 from HDFS to MySQL result4 table

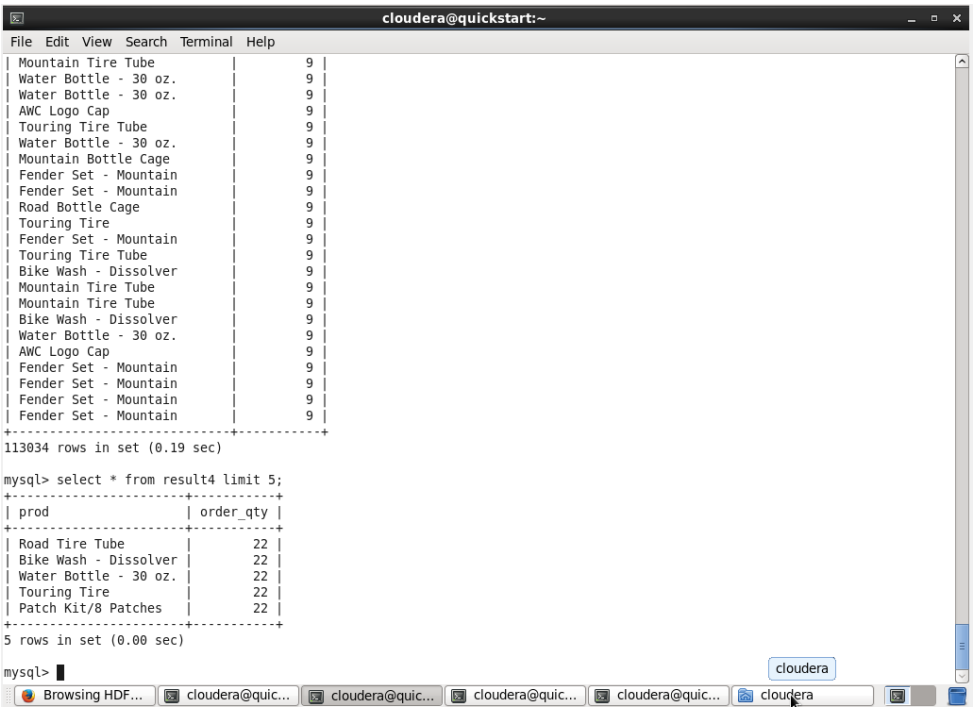


Figure 27 shows successful display of exported hive table into MySQL.

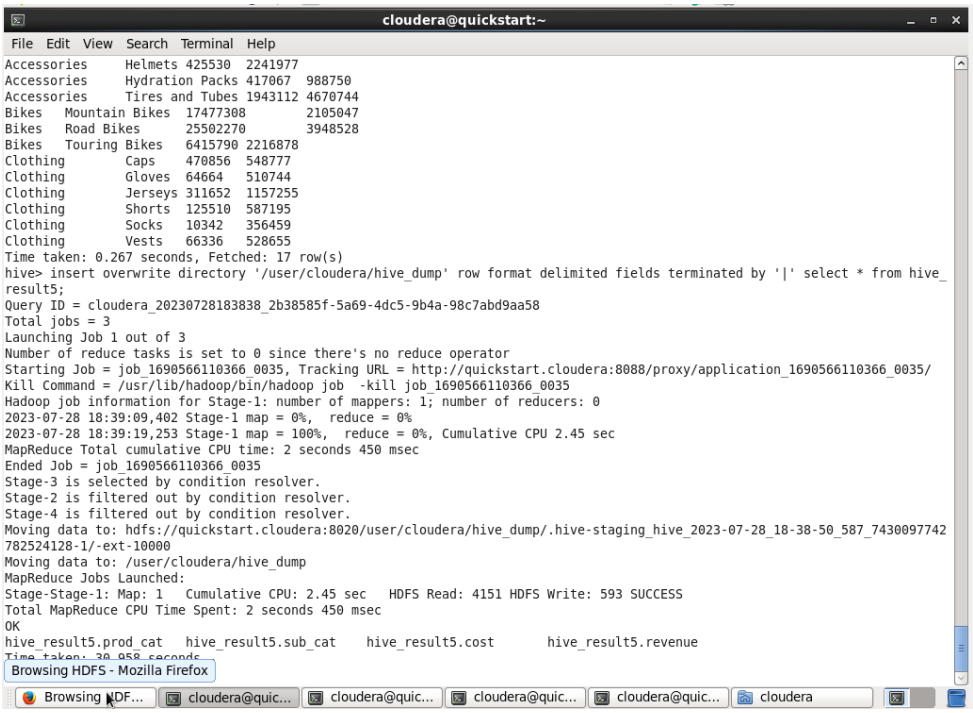


Figure 28 shows syntax for saving hive_result5 data into HDFS

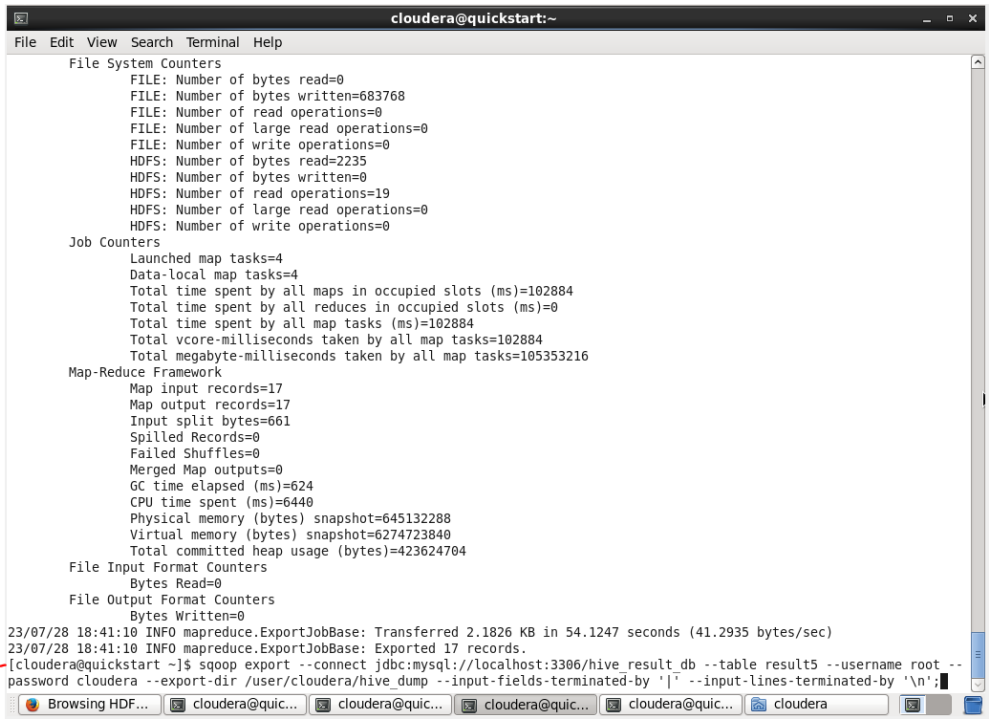


Figure 29 shows syntax for successful export of hive_result5 into MySQL result5 table.

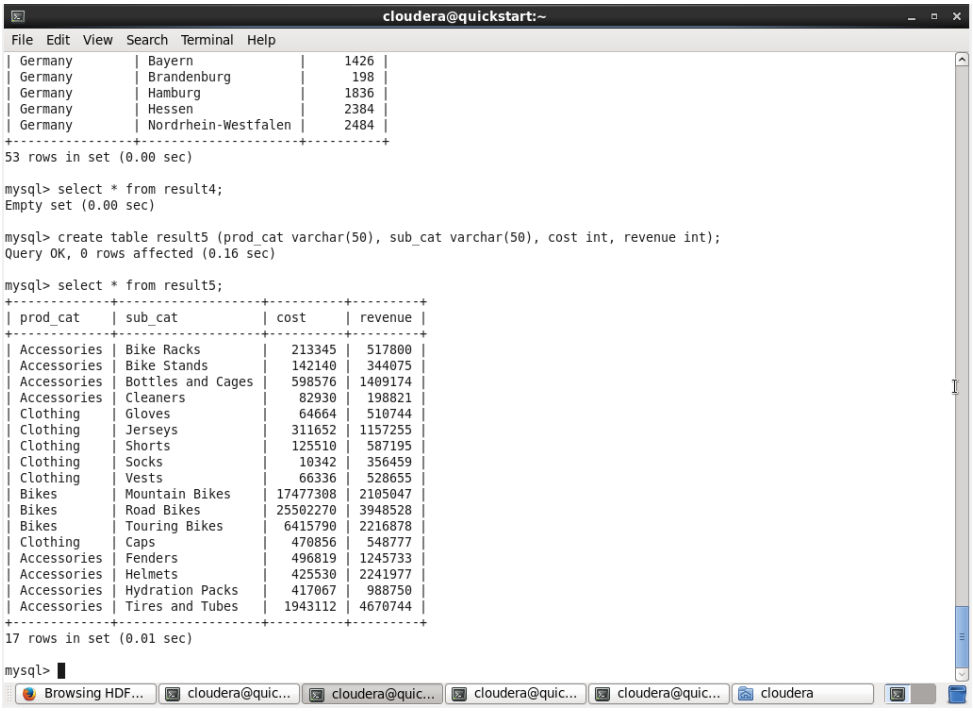


Figure 30 shows the result of MySQL result5 table.

5. DATA INSIGHTS:

The data is a historical sales data of a company that sells different outdoor products which includes bike and its accessories, clothes, vest for both male and female. For this project, we have limited analysis to 5 queries to draw insight from product sales.

Data indicates sales to USA, Europe, and Australia. USA, being the world biggest market by GDP, recorded the biggest ever sales of outdoor products. In 2014, 2016 and 2013 were the strongest sales in the USA. Australia being the Darwin loves outdoor activities with favorable weather showed second strong sales. As shown in Figure 31, we can see the revenues by year.

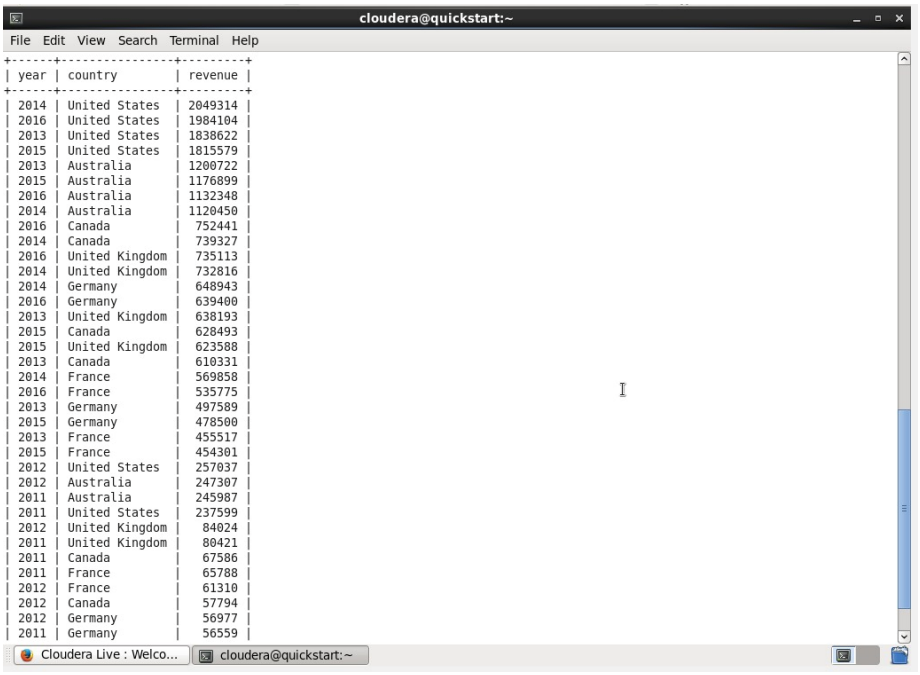


Figure 31 country with most revenue generated.

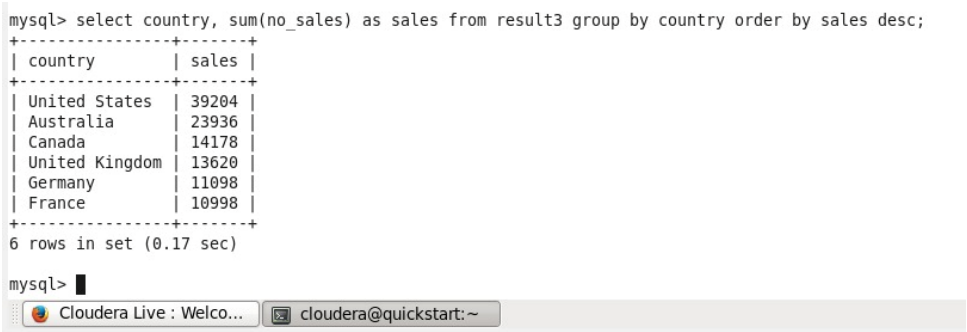
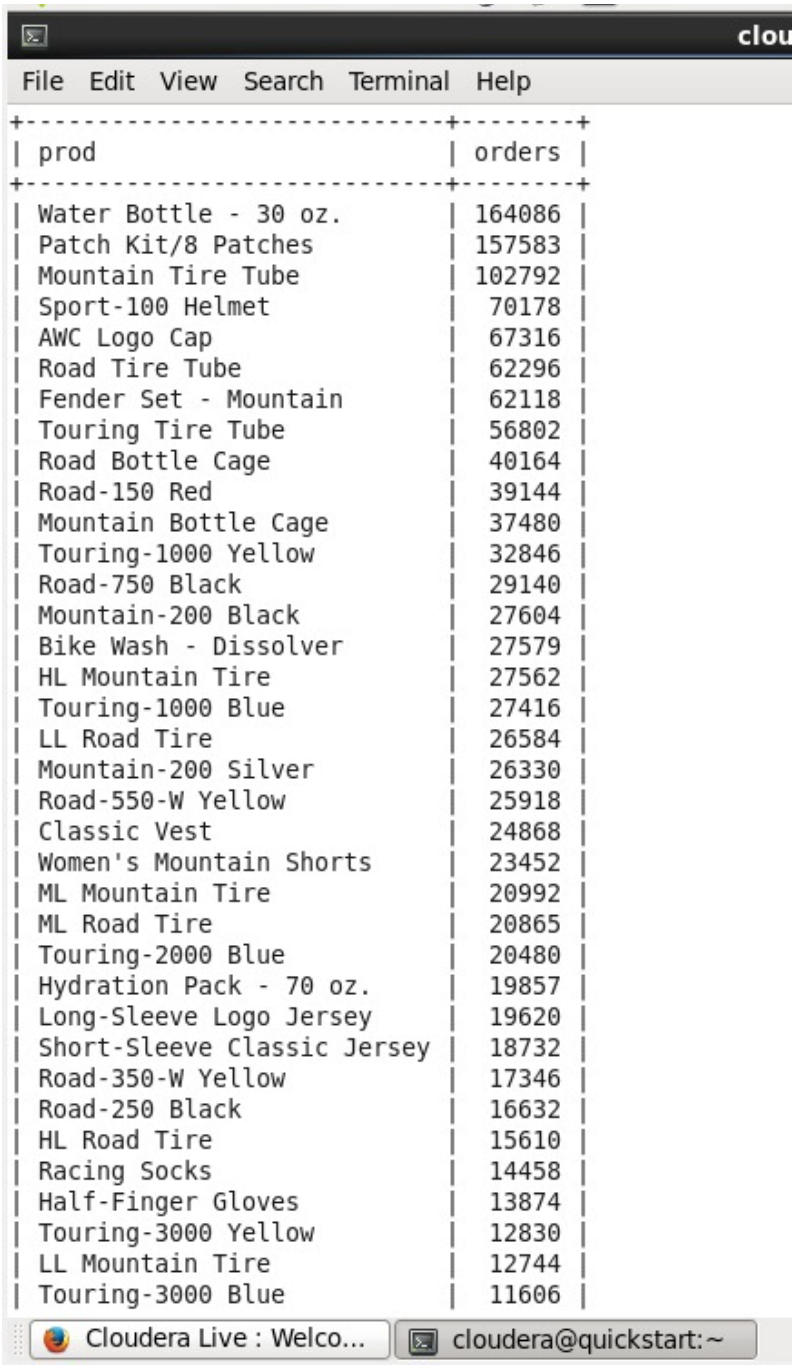


Figure 32 USA and Australia is the biggest market for outdoor.

However, most of the sold products ordered were water bottles, patch kit and mountain tire tube. Which show strong outdoor activities because of the need to stay hydrated and repairs of already bought bikes. It is important to monitor the inventory of accessories so that accessories are available for customers. Figure 33 shows the products bought by customers.



prod	orders
Water Bottle - 30 oz.	164086
Patch Kit/8 Patches	157583
Mountain Tire Tube	102792
Sport-100 Helmet	70178
AWC Logo Cap	67316
Road Tire Tube	62296
Fender Set - Mountain	62118
Touring Tire Tube	56802
Road Bottle Cage	40164
Road-150 Red	39144
Mountain Bottle Cage	37480
Touring-1000 Yellow	32846
Road-750 Black	29140
Mountain-200 Black	27604
Bike Wash - Dissolver	27579
HL Mountain Tire	27562
Touring-1000 Blue	27416
LL Road Tire	26584
Mountain-200 Silver	26330
Road-550-W Yellow	25918
Classic Vest	24868
Women's Mountain Shorts	23452
ML Mountain Tire	20992
ML Road Tire	20865
Touring-2000 Blue	20480
Hydration Pack - 70 oz.	19857
Long-Sleeve Logo Jersey	19620
Short-Sleeve Classic Jersey	18732
Road-350-W Yellow	17346
Road-250 Black	16632
HL Road Tire	15610
Racing Socks	14458
Half-Finger Gloves	13874
Touring-3000 Yellow	12830
LL Mountain Tire	12744
Touring-3000 Blue	11606

Figure 33 most orders products.

Without doubt we can confirm the countries with the most revenue and we can see in figure 33 that the top 10 products are predominantly accessories. So, with confirmation from figure 34, we confirm that accessories are the leading source of revenue for this company.

Therefore, we recommend that the company further analysis their inventory to maintain enough products for their customer especially towards inventory limits. Secondly, the company can increase their advertising campaign to drive sales of other products, ask for product feedback to drive seasonal growth further higher.

6. CONCLUSION:

From the analysis we can see that the company sales seasonal products which generates good sales. Their biggest market is USA and Australia, so the company must continue run targeted advert to improve sales, especially before the beginning of the season and expand their advert promoting healthy outdoor activities.

7. CHALLENGES:

While transferring our Hive result data from HDFS to MySQL, we noticed that our query would run successfully but the table in MySQL is empty. After further investigation, we noticed we did not include the right delimiter causing our not be clean enough for transfer. After further investigation, we found the right syntax that removes the pipe '|' thereby making our data to be successfully transferred from HDFS to MySQL. It is a lesson learned. Figure 35 is what our data looks like without removing the pipe delimiter.

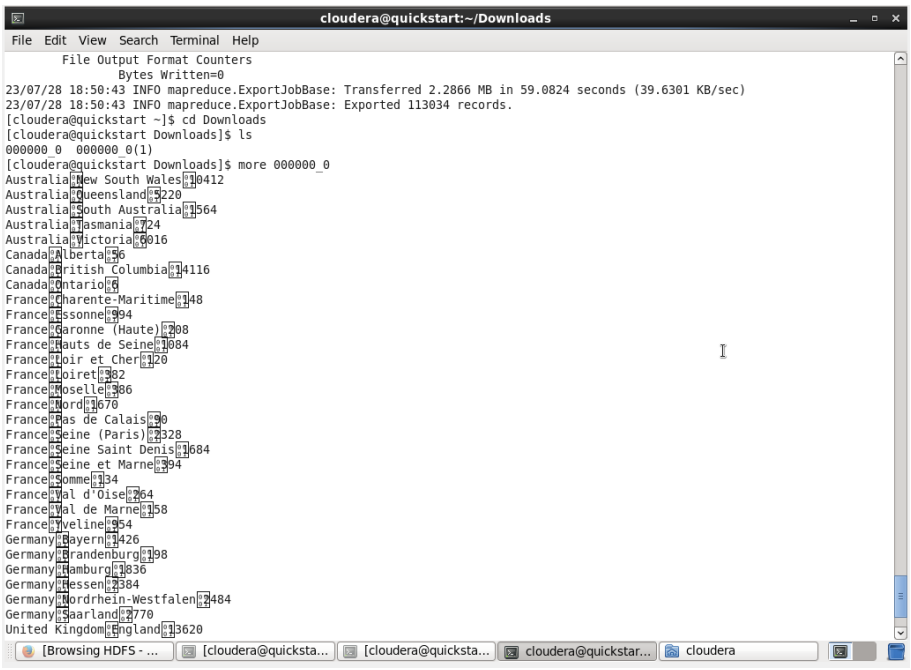


Figure 34 shows hive-result table result with unknown delimiter.

8. PROJECT PARTICIPANTS:

PROJECT PARTICIPANTS	**STUDENT ID**
Student Name: Mahima Akula	C0908140
Student Name: Modupeola Omodunni Oyatokun	C0895705
Student Name: Jumoke Yekeen	C0900481
Student Name: Chibuike Okoroama	C0892150
Student Name: Diksha	C0908141
Student Name: Harish Kundal	C0906990