## horizontal line



Assignment 02

**Implementation of a system for sales data analytics using Hadoop Eco-System**

**─** 

Group 141 Members

Kumar Anurag - 2021FC04016

Das Shubhankar Jagannath Arpita - 2021FC04154

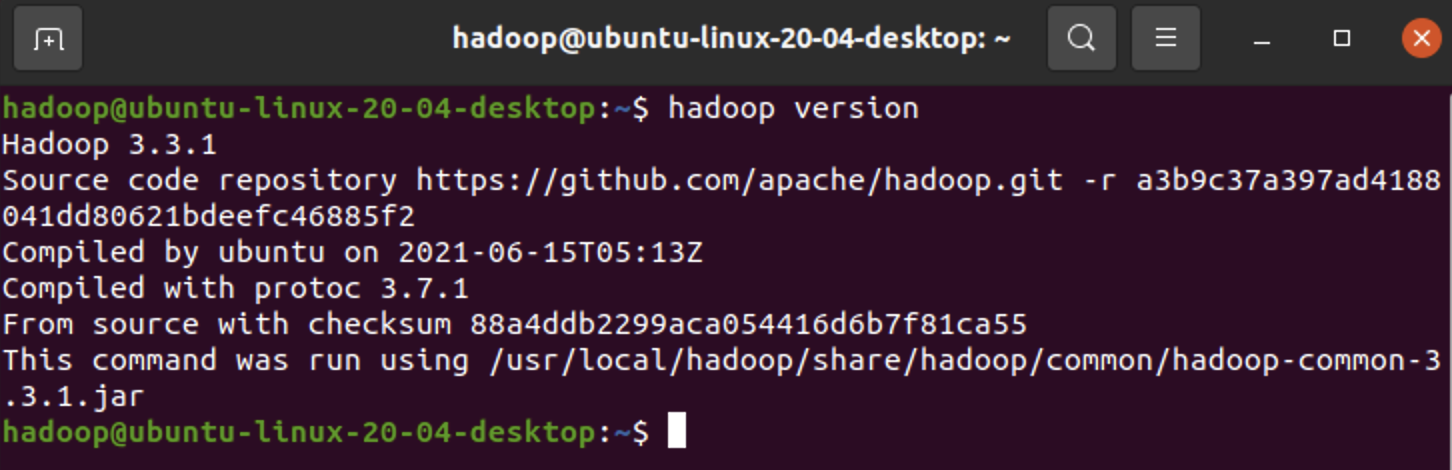
Amit Kumar - 2021FC04433

Siddhant Shivam - 2021FC04090

# SETTING UP THE HADOOP & HDFS

## 1.1 Hadoop Version

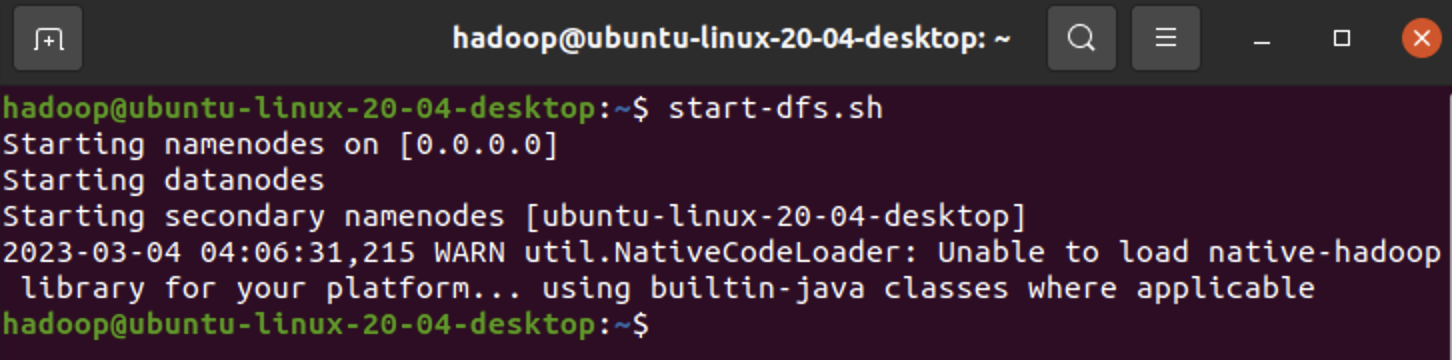
Printing the current Hadoop version.



## 

## 1.2 Starting the DFS services

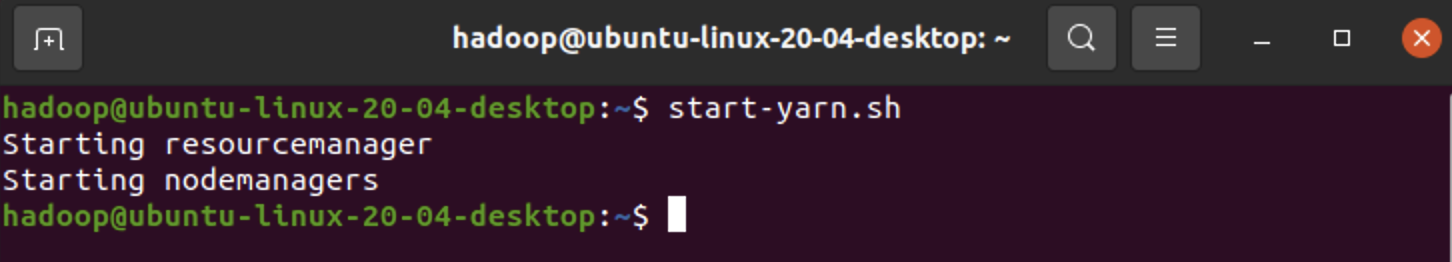
We are starting the DFS services using the **start-dfs.sh** command.



## 

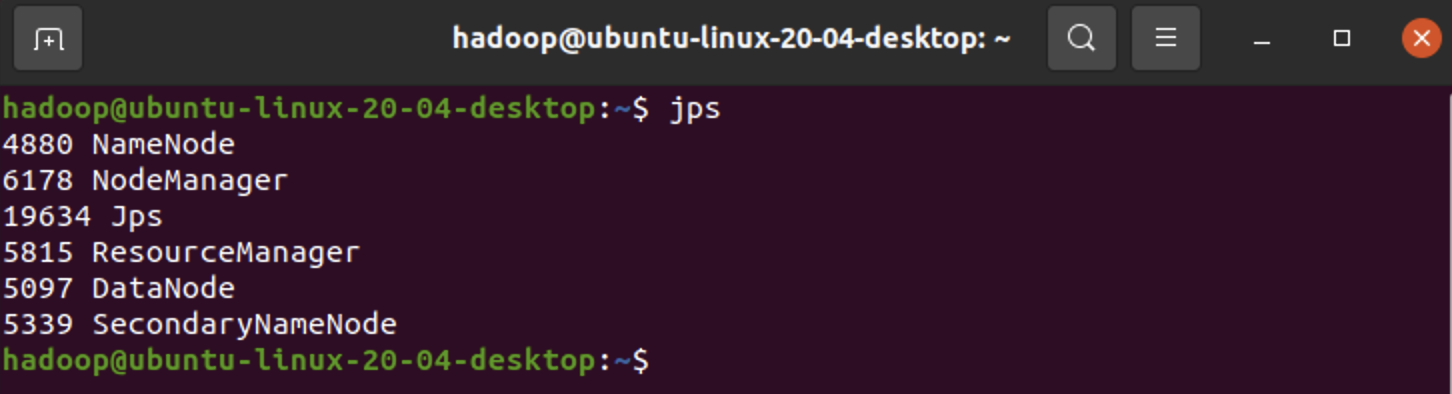
## 

## 1.3 Starting the Yarn services

We are starting the yarn services using the **start-yarn.sh** command.

## 1.4 Verifying the running components

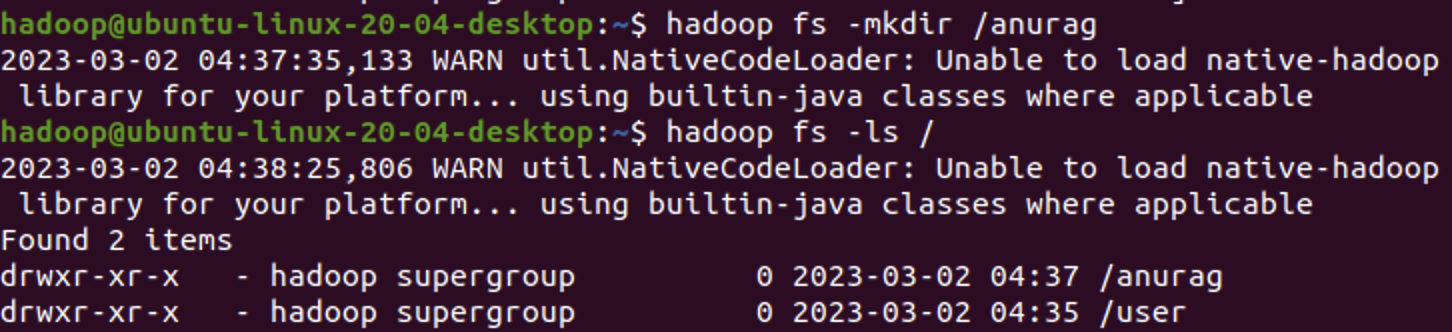
We are verifying the running components using the **jps** command.



## 

## 1.5 Creating a directory in HDFS for storing the dataset

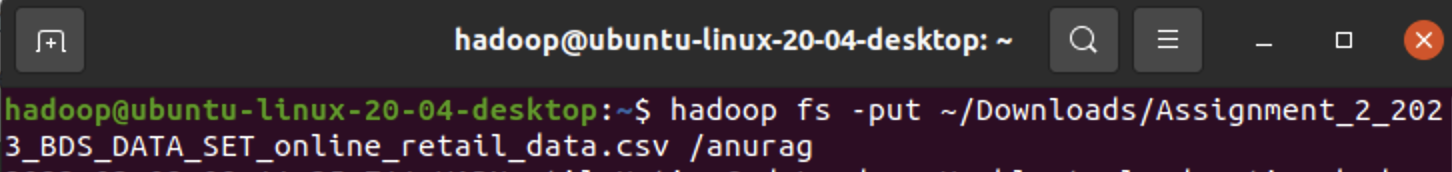
We used the **-mkdir** command to create a new directory named **anurag** and **-ls** to show the created folder.

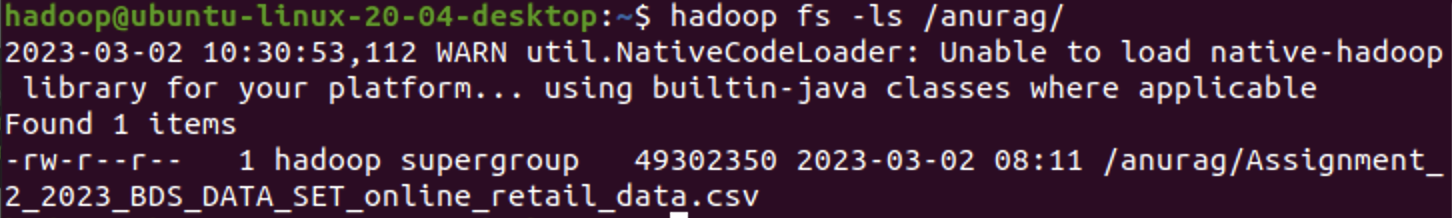


**Note**: We will always get a warning of **NativeCodeLoader**, the reason is that the native Hadoop Library “**$HADOOP\_HOME/lib/native/libhadoop.so.1.0.0**” was actually complied to 32 bit and the OS we are using is Ubuntu 64 bit so that’s why it will show this warning at step. However, it won’t impact any Hadoop functionality, so kindly ignore it.

## 1.6 Uploading dataset into the Hadoop Distributed File System

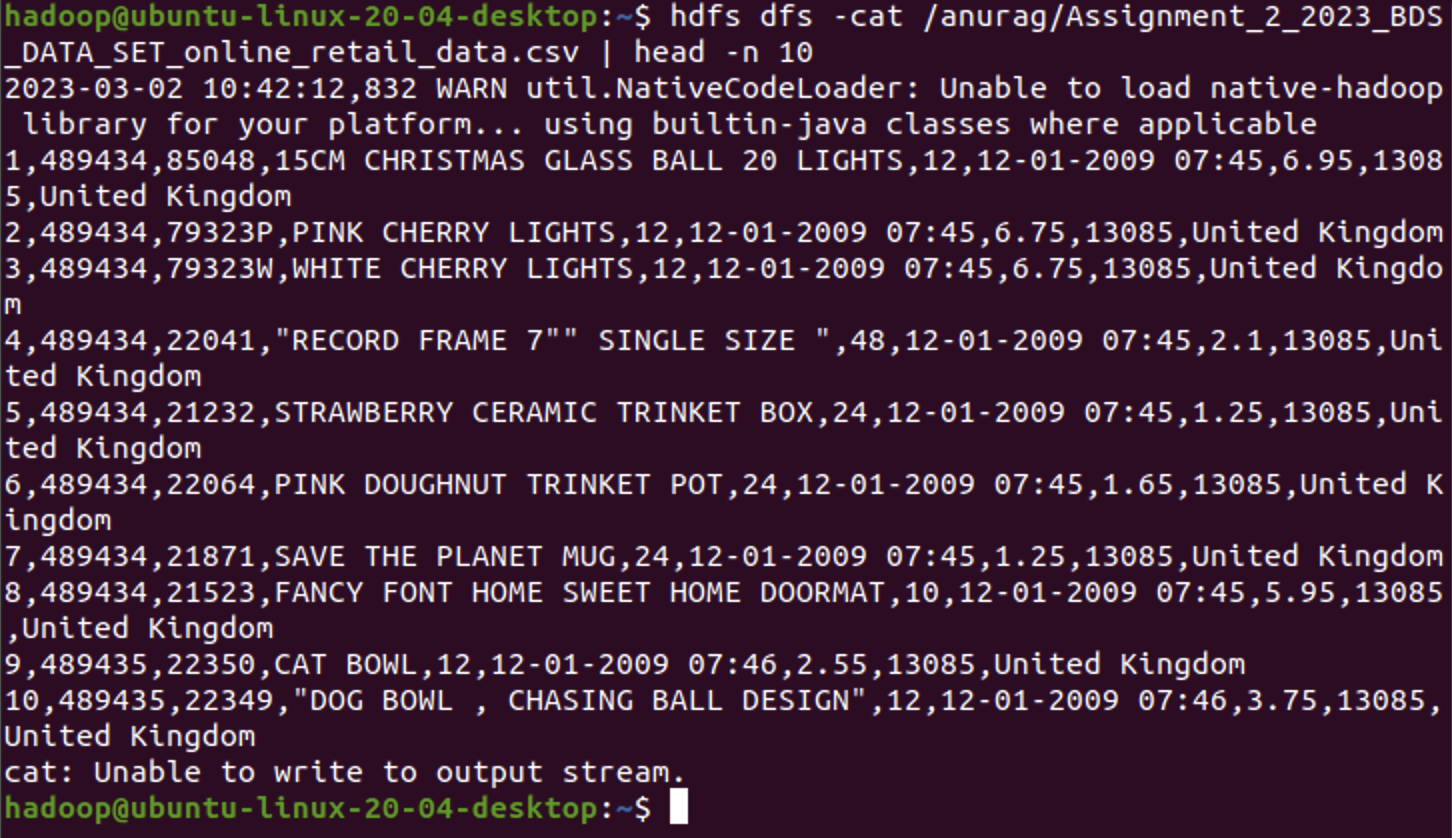
Using **-put** command, we have uploaded the given dataset into the created hadoop directory.





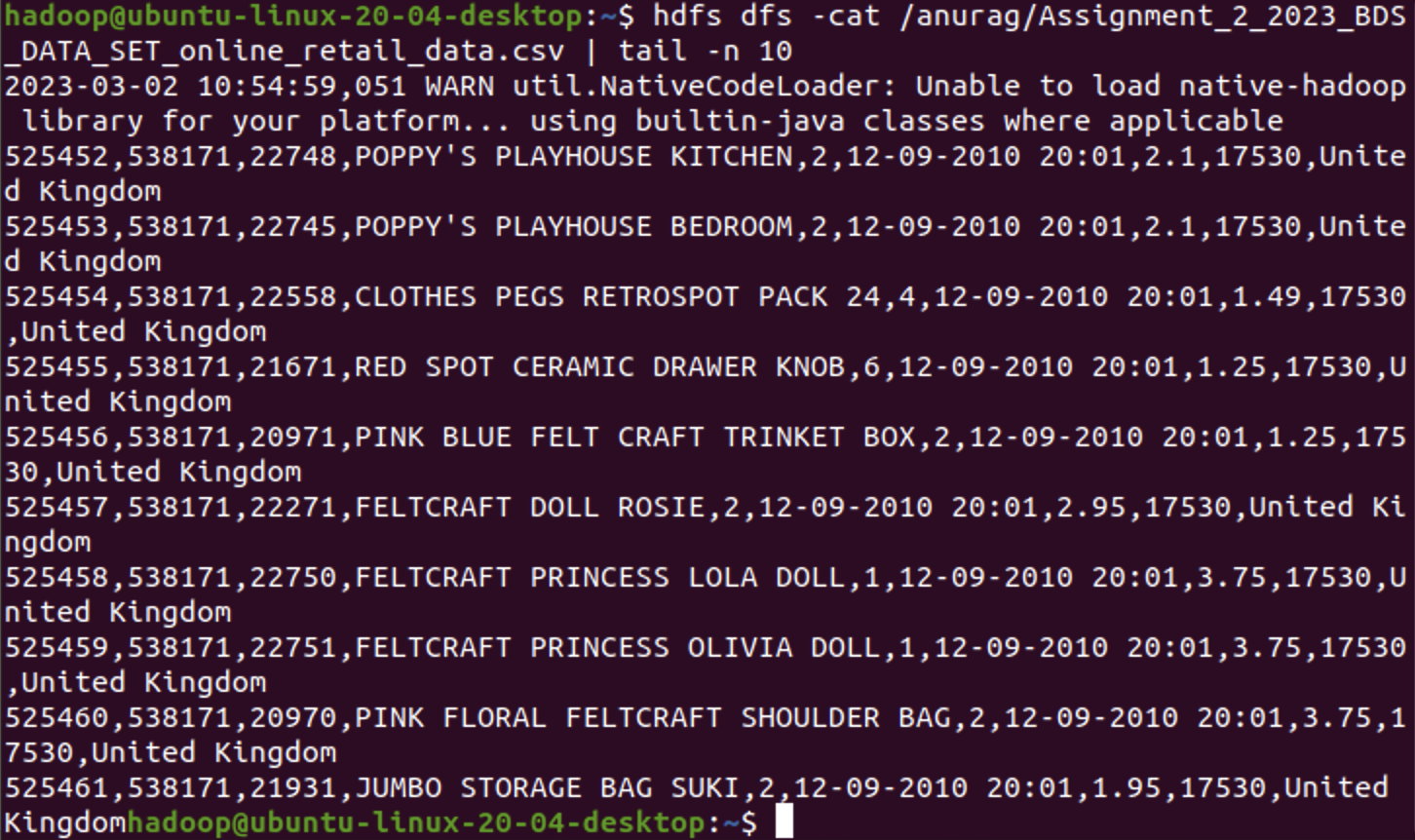
## 1.7 Printing the first 10 rows of the given dataset

Using the **-cat** and **head** commands, we have printed the first 10 rows of the given dataset.



## 1.8 Printing the last 10 rows of the given dataset

Using the **-cat** and **tail** commands, we have printed the last 10 rows of the dataset.



# 

# 

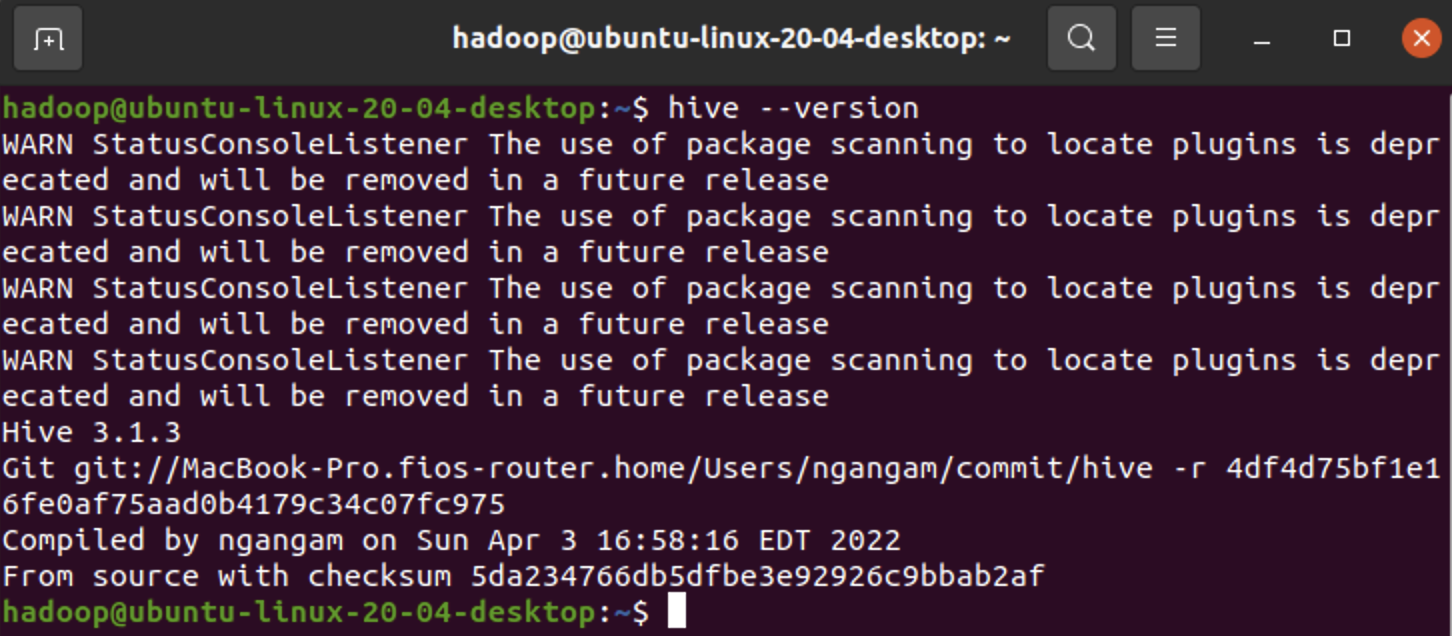
# 

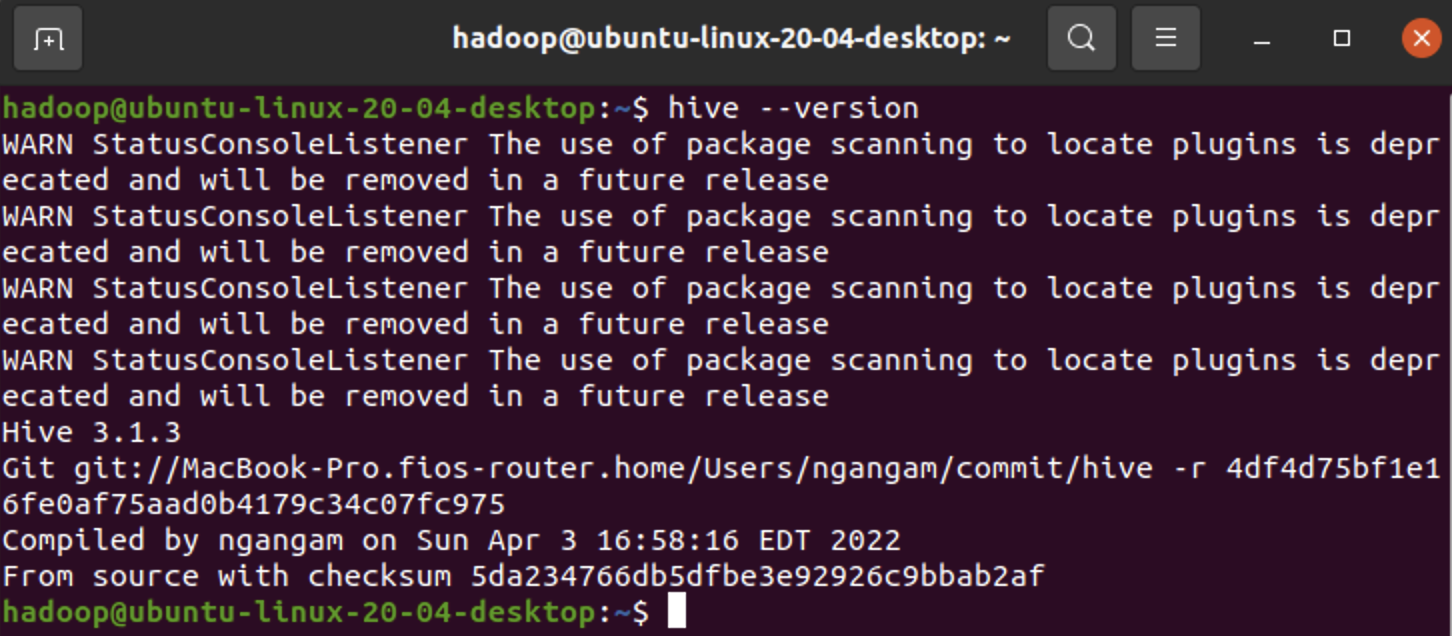
# 

# SETTING UP THE HIVE

## 2.1 Hive Version

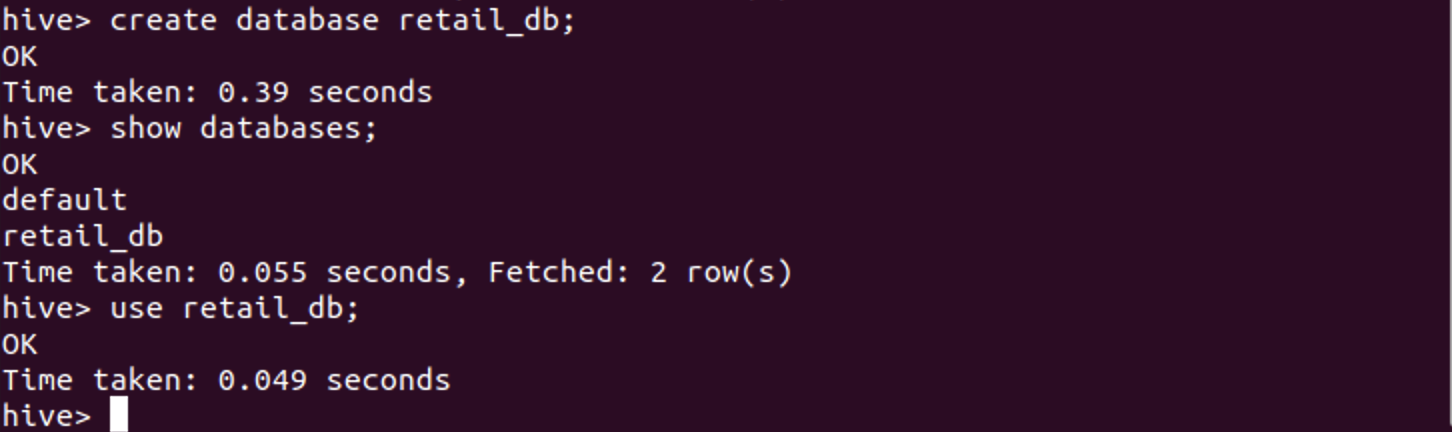
Printing the hive version.





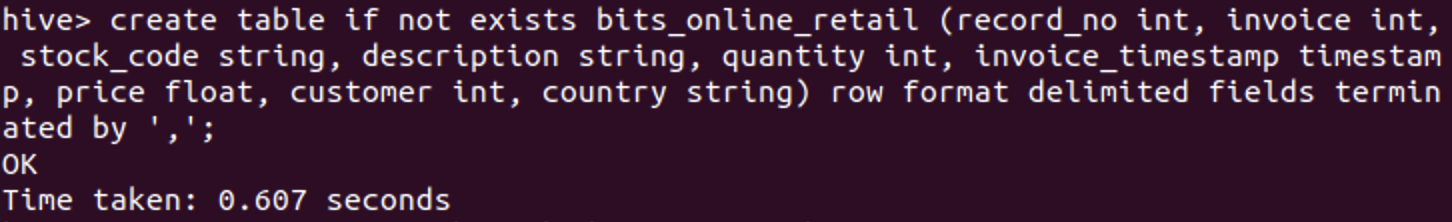
## 2.2 Creating database in Hive

Using the create database query, we have created a new database named **retail\_db**.



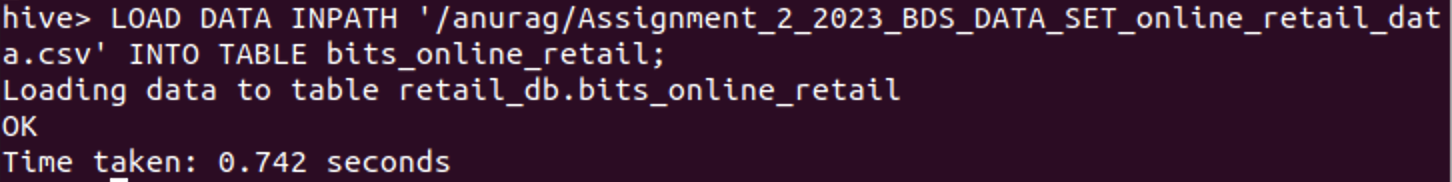
## 2.3 Creating a table in Hive

Using create table query, we have created a new table named **bits\_online\_retail**.



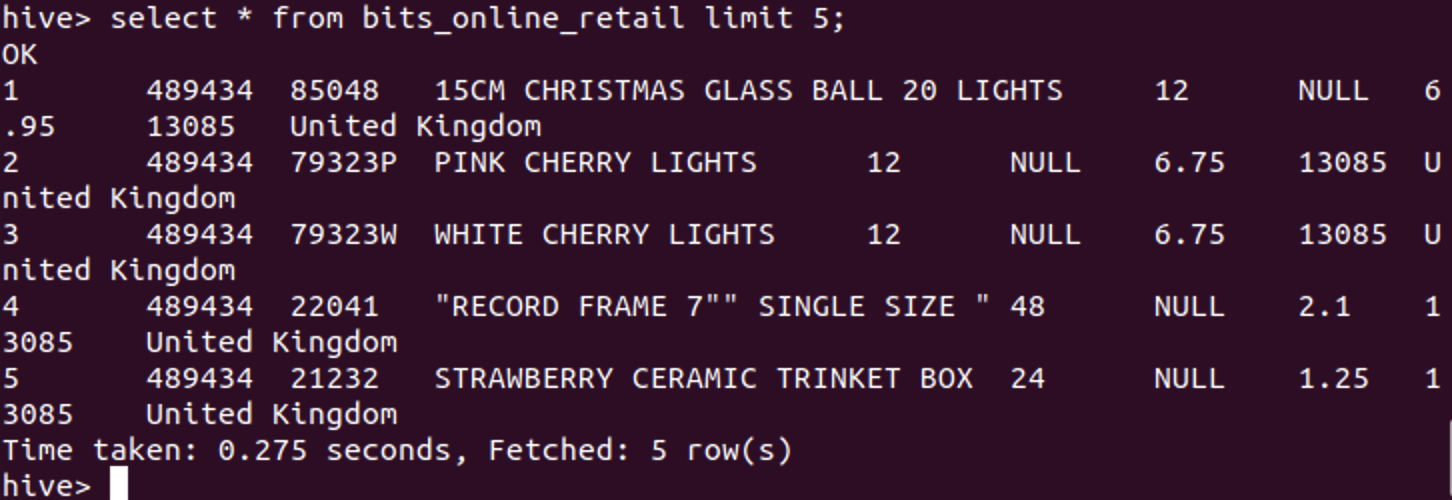
## 2.4 Importing the values in the Hive table from Hadoop stored database

Using the **LOAD DATA** query, we have imported the csv file values from the Hadoop database that we created earlier.



## 2.5 Printing first 5 rows of Hive table

Using **select** and **limit** query, we have printed the first 5 rows of the table **bits\_online\_retail**.

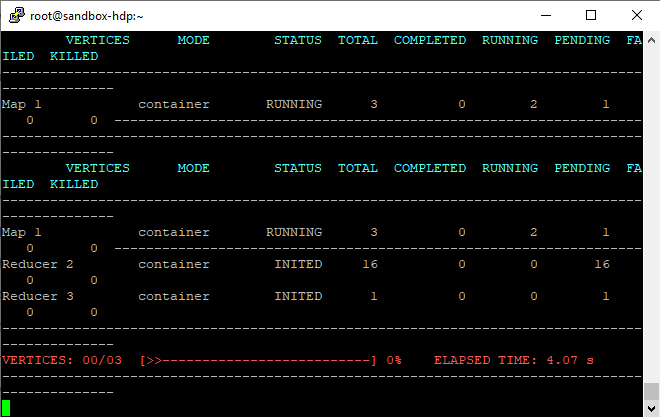


# ANALYTICS

As we know Hive queries make use of MapReduce function so in order to make sure, the system doesn’t go in hung state, so we have executed the queries inside the Sandbox.

## MapReduce:

For answering the below questions, we have made use of MapReduce() function of Hive and Hadoop. Below is the snippet for reference.



## 

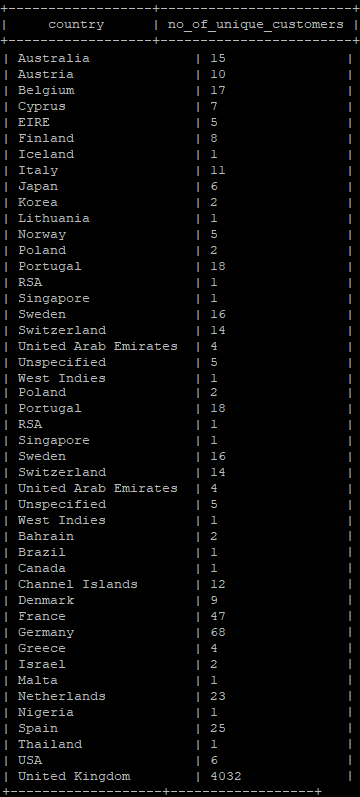
## 

## 

## 3.1 Total number of unique customers in the "given country".

For counting the unique customers, we are making use of the **DISTINCT** keyword and also we are making sure that we don’t count any **NULL** values so we are putting an extra where clause.

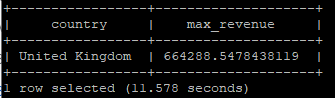




## 3.2 Country from which the maximum revenue was collected from sales in the month of March 2010.

For getting this output, first we have used **GROUP BY** country, so that we can get country-wise data, then we have used **SUM** function to calculate to total revenue, after that we set the condition of March 2010 with the help of **WHERE** clause, and finally we have sorted the results in **DESC** order using **ORDER BY** clause.

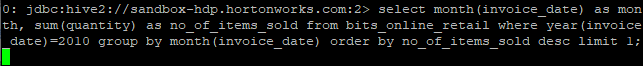


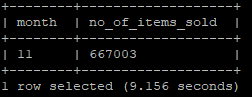


## 

## 3.3 Month of 2010 in which maximum number of items were sold.

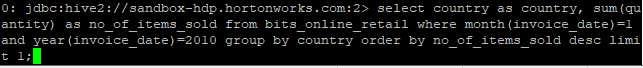
Using the **month()** function, we have extracted the month from invoice column, then by applying the **sum()** function on quantity column, we have calculated the total no of items sold, and finally we have arranged the data in **DESC** order using the **ORDER BY** clause.

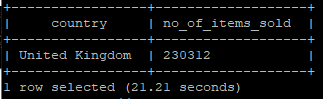




## 3.4 In the month of January 2010, find the country in which the maximum number of items were sold.

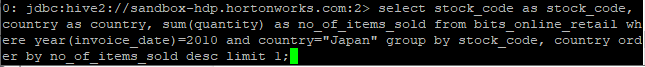
Using the **sum()** function, we have calculated the total number of items sold and with the help of **month()** function we have set the **WHERE** clause to January month and with **year()** function, we set the year to 2010. Finally, we arranged the data in **DESC** using **ORDER BY** clause and since we need only the country with maximum no of items sold so we have used **LIMIT** clause to 1 so that it can only print the highest top record.

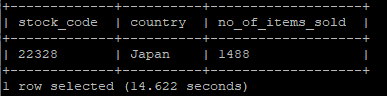




## 3.5 The StockCode of the item with the highest number of sales in the "given country" in the year 2010.

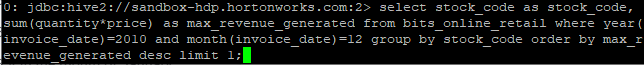
As we have a column named **stock\_name** in our table **bits\_online\_retail**, so we have selected the same using **SELECT**, then have used **GROUP BY** clause on **stock\_code** and **country**. Finally, we arranged the series in **DESC** order using the **ORDER BY** clause and printed the required top row using the **LIMIT** keyword.

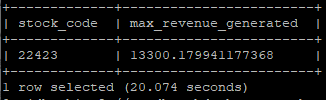




## 3.6 StockCode of the item for which the maximum revenue was received by sales in the month of December 2010.

With the help of the **sum()** function, we have calculated the total revenue generated and using **WHERE** clause we have set the month to 12 i.e. December and the year to 2010. We have set the **GROUP BY** clause to **stock\_code** so that the data is grouped as per the stock codes. Finally, we have arranged the rows in **DESC** order using the **ORDER BY** clause and we have printed the required top most row using the **LIMIT** keyword.

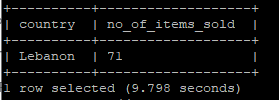




## 3.7 The country in which the minimum number of sales happened in 2010.

Using the **sum()** function, we have calculated the total no of sales for each country. Also, with the help of the **WHERE** clause, we have set the year condition to 2010. Finally, we arranged the rows in **ASC** order of their no of sales by making use of **ORDER BY** clause and with **LIMIT** keyword, we have just printed the desired top most row.





## 

# EVALUATOR’S SETUP

## Instructions:

1. Prior Hadoop and Hive setup is required for executing the queries.
2. Load the CSV dataset in the Hadoop distributed file system from the local system.
3. Import the CSV from HDFS to Hive Metastore.
4. In the Hive Shell, now we can execute the above mentioned queries one by one.

## 

## 

## 

## 

## 

## 

## 

## 

## 

## 

## 

## **-x-x- Thank You -x-x-**