Hive Case Study

By -Arjun Bhatnagar, Syed Mohammad and Rehan Shaikh

Problem Statement

With online sales gaining popularity, tech companies are exploring ways to improve their sales by analysing customer behaviour and gaining insights about product trends. Furthermore, the websites make it easier for customers to find the products they require without much scavenging. Needless to say, the role of big data analysts is among the most sought-after job profiles of this decade. Therefore, as part of this assignment, we will be challenging you, as a big data analyst, to extract data and gather insights from a real-life data set of an e-commerce company.

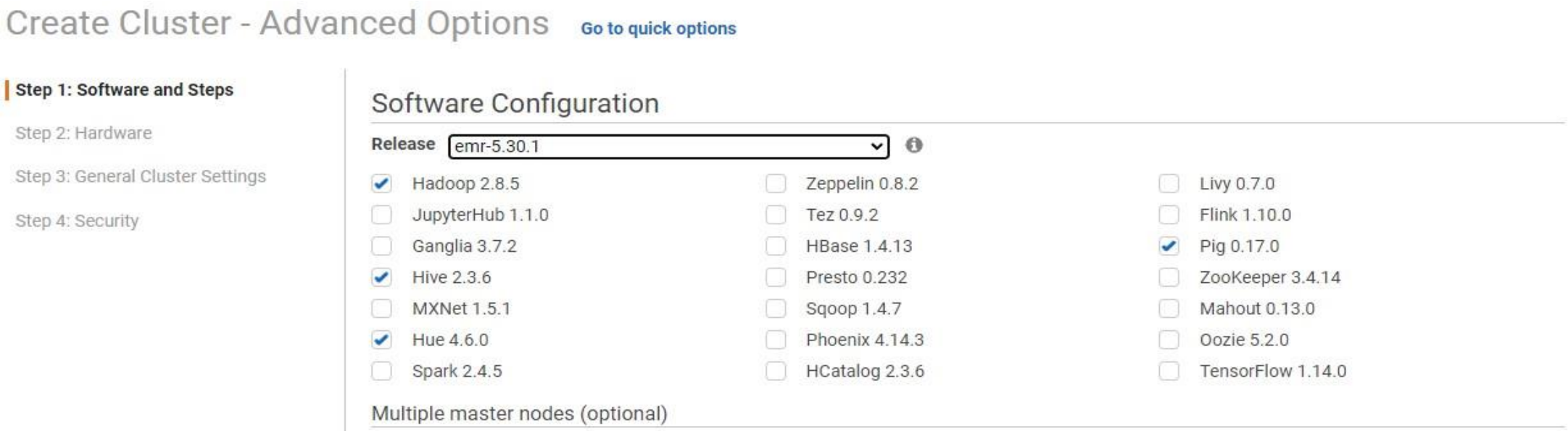
Data for this case study was available in:

https://e-commerce-events-ml.s3.amazonaws.com/2019-Oct.csv
https://e-commerce-events-ml.s3.amazonaws.com/2019-Nov.csv

In this case study we used a **2-node** EMR cluster with both the master and core nodes as **M4.large** and we've used a emr-5.29.0 release for this case study.

Creating EMR Cluster

We login in to the AWS , go to the Console and then to EMR Home Page → Click on CreateCluster → Advanced Options → Select release emr-5.30.1 and select required services for the case study.



In this case study as suggested we are using a **2-node** EMR cluster with both the master and core nodes as **M4.large**.

Cluster Nodes and Instances

Choose the instance type, number of instances, and a purchasing option. Learn more about instance purchasing options

Console options for automatic scaling have changed. Learn more

Node type	Instance type	Instance count	Purchasing option
Master Master - 1	**m4.large** 2 vCore, 8 GiB memory, EBS only storage EBS Storage: 32 GiB Add configuration settings	1 Instances	On-demand Spot Use on-demand as max price
Core Core - 2	**m4.large** 2 vCore, 8 GiB memory, EBS only storage EBS Storage: 64 GiB Add configuration settings	2 Instances	On-demand Spot Use on-demand as max price

+ Add task instance group

Named the cluster as Hive Assignment

Step 1: Software and Steps

Step 2: Hardware

Step 3: General Cluster Settings

Step 4: Security

General Options

Cluster name hive assignment

- [x] Logging
  - S3 folder s3://aws-logs-769438569402-us-east-1/elasticmap
- [ ] Log encryption
- [x] Debugging
- [x] Termination protection

Now go to Security > Choose “demokey9900” EC2 Key-Pair and then click on create cluster

Create Cluster - Advanced Options

Go to quick options

Step 1: Software and Steps

Step 2: Hardware

Step 3: General Cluster Settings

Step 4: Security

Security Options

EC2 key pair demokey9900

- [x] Cluster visible to all IAM users in account

Permissions

(●) Default ( ) Custom

Use default IAM roles. If roles are not present, they will be automatically created for you with managed policies for automatic policy updates.

EMR role EMR_DefaultRole [ ] Use EMR_DefaultRole_V2

Our cluster Hive Assignment is created and launched successfully and is now in “Ready” state.

Summary | Application user interfaces | Monitoring | Hardware | Configurations | Events | Steps | Bootstrap actions

Summary

ID: j-3TJV2HDU0Q2DO
Creation date: 2022-12-06 11:57 (UTC+5:30)
Elapsed time: 12 minutes
After last step completes: Cluster waits
Termination protection: On Change
Tags: – View All / Edit
Master public DNS: ec2-3-85-177-126.compute-1.amazonaws.com
Connect to the Master Node Using SSH

Configuration details

Release label: emr-5.30.1
Hadoop distribution: Amazon 2.8.5
Applications: Hive 2.3.6, Pig 0.17.0, Hue 4.6.0
Log URI: s3://aws-logs-769438569402-us-east-1/elasticmapreduce/
EMRFS consistent view: Disabled
Custom AMI ID: --

“demo9900” is the Key Pair we are using

Key pairs (1/2) Info

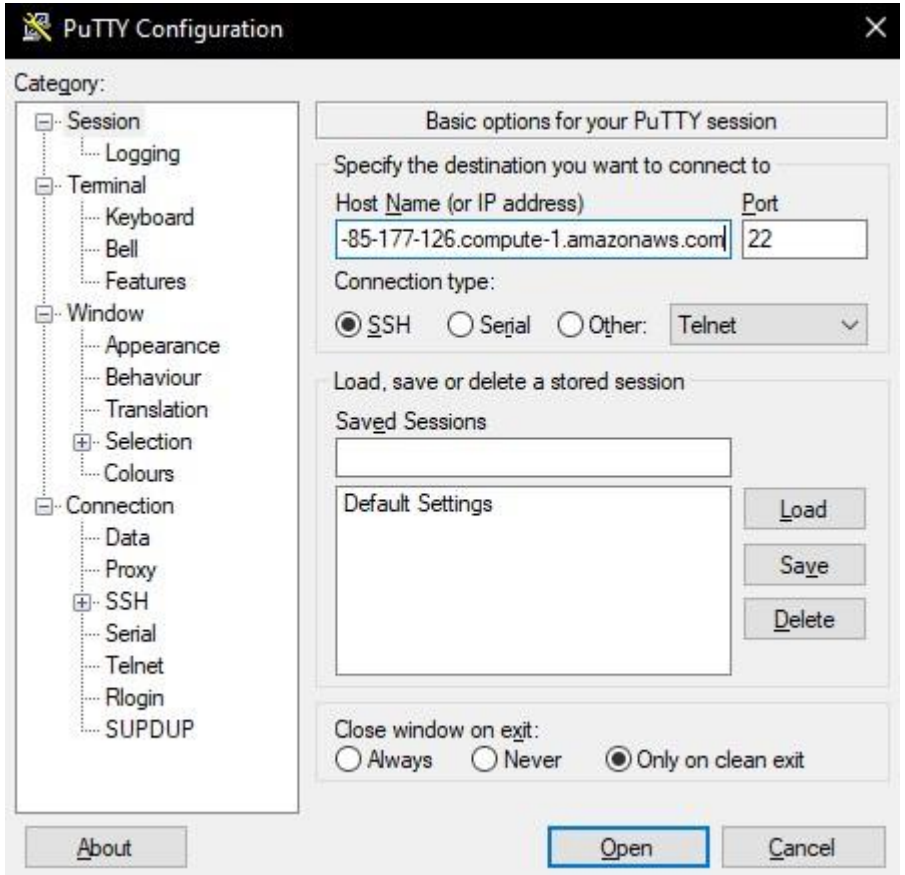
Search

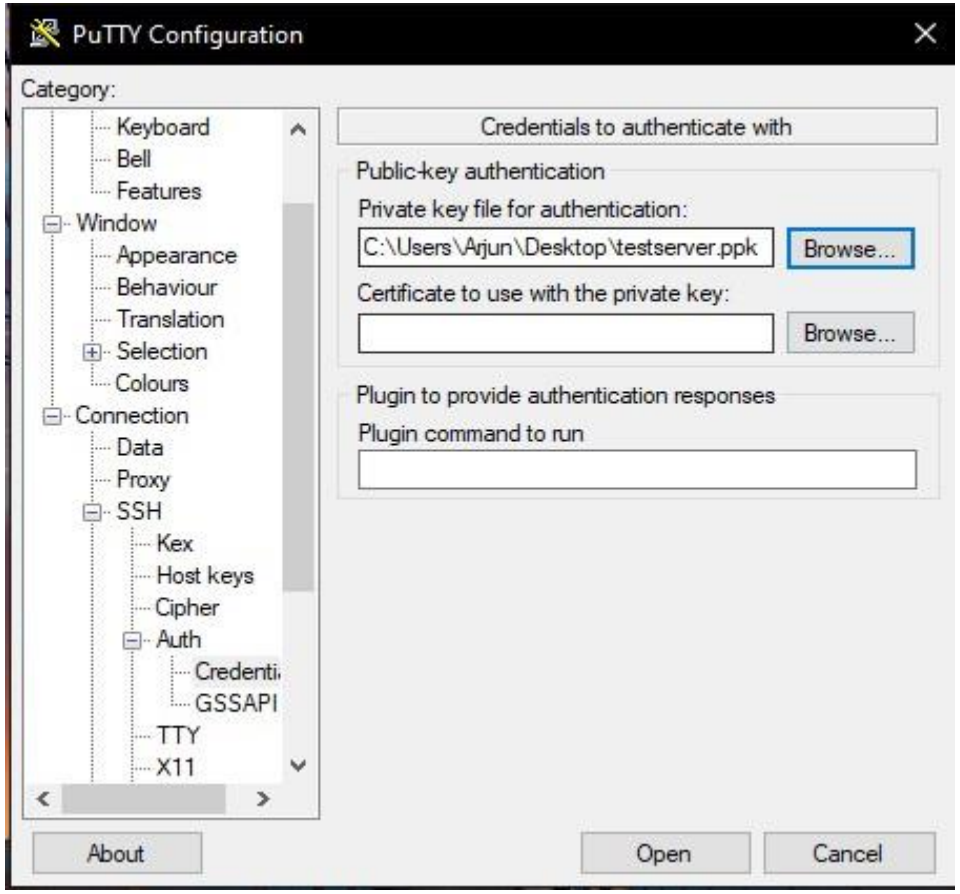
Name	Type	Created	Fingerprint	ID
demokey9900	rsa	2022/11/15 15:48 GMT+5:30	b2:05:df:19:56:45:26:df:9a:48:6c:d1:b5:...	key-0e9b35659974edf07
vockey	rsa	2022/11/15 15:16 GMT+5:30	b1:31:90:18:29:a0:63:ec:fb:3b:68:07:ff:...	key-093f02372f7a09b9e

Hadoop & Hive Querying

Open PuTTY and give the Host Name as “hadoop@” followed by the Master DNS address from the EMR cluster summary page.

Then click on SSH -> Auth and load the .ppk Key Pair File.





Launching EMR:

```
hadoop@ip-172-31-50-127:~
login as: hadoop
Authenticating with public key "imported-openssh-key"
Last login: Tue Dec  6 06:40:25 2022

       __|  __|_  )
       _|  (     /   Amazon Linux 2 AMI
      ___|\___|___|

https://aws.amazon.com/amazon-linux-2/

EEEEEEEEEEEEEEEEEEEE MMMMMMMM           MMMMMMMM RRRRRRRRRRRRRRR
E::::::::::::::::::E M:::::::M         M:::::::M R::::::::::::::R
EE::::::EEEEEEEEE:::E M::::::::M       M::::::::M R::::::RRRRRR::::R
  E::::E       EEEEE M:::::::::M     M:::::::::M RR::::R      R::::R
  E::::E             M::::::M:::M   M:::M::::::M   R:::R      R::::R
  E::::::EEEEEEEEEE  M::::M M:::M M:::M M::::M   R:::RRRRRR::::R
  E::::::::::::::E   M::::M  M:::M:::M  M::::M   R::::::::::RR
  E::::::EEEEEEEEEE  M::::M   M::::M   M::::M   R:::RRRRRR::::R
  E::::E             M::::M    M:::M    M::::M   R:::R      R::::R
  E::::E       EEEEE M::::M     MMM     M::::M   R:::R      R::::R
EE::::::EEEEEEEE:::E M::::M             M::::M   R:::R      R::::R
E::::::::::::::::::E M::::M             M::::M RR::::R      R::::R
EEEEEEEEEEEEEEEEEEEE MMMMMM             MMMMMM RRRRRRR      RRRRRR

[hadoop@ip-172-31-50-127 ~]$
```

Creating a directory – “hiveassignment” :

```
hadoop fs -mkdir /hiveassignment
```

```
[hadoop@ip-172-31-49-96 ~]$ hadoop fs -mkdir /hiveassignment
```

Checking the directory:

```
hadoop fs -ls /
```

We see that a directory, “hiveassignment” has been created

```
ommand [genericoptions] [commandoptions]

hadoop@ip-172-31-50-127 ~]$ hadoop fs -ls
hadoop@ip-172-31-50-127 ~]$ hadoop fs -ls /
ound 5 items
rwxr-xr-x   - hdfs   hadoop          0 2022-12-06 06:35 /apps
rwxr-xr-x   - hadoop hadoop          0 2022-12-06 06:45 /hiveassignment
rwxrwxrwt   - hdfs   hadoop          0 2022-12-06 06:37 /tmp
rwxr-xr-x   - hdfs   hadoop          0 2022-12-06 06:35 /user
rwxr-xr-x   - hdfs   hadoop          0 2022-12-06 06:35 /var
hadoop@ip-172-31-50-127 ~]$
```

Loading Data:

Since the size of the data is large we'll load the data into HDFS from S3 an into the local storage.

```
hadoop distcp 's3://upgrad-hiveassignment/hiveassignment/2019-Oct.csv' /hiveassignment/2019-Oct.csv
hadoop distcp 's3://upgrad-hiveassignment/hiveassignment/2019-Nov.csv' /hiveassignment/2019-Nov.csv
```

```
[hadoop@ip-172-31-50-127 ~]$ hadoop distcp 's3://hivebucketstudy/2019-Nov.csv' /hiveassignment/2019-Nov.csv
22/12/06 07:46:45 INFO tools.DistCp: Input Options: DistCpOptions{atomicCommit=false, syncFolder=false, deleteMissing=false, igno
reFailures=false, overwrite=false, skipCRC=false, blocking=true, numListstatusThreads=0, maxMaps=20, mapBandwidth=100, sslConfigu
rationFile='null', copyStrategy='uniformsize', preserveStatus=[], preserveRawXattrs=false, atomicWorkPath=null, logPath=null, sou
rceFileListing=null, sourcePaths=[s3://hivebucketstudy/2019-Nov.csv], targetPath=/hiveassignment/2019-Nov.csv, targetPathExists=f
alse, filtersFile='null'}
22/12/06 07:46:45 INFO client.RMProxy: Connecting to ResourceManager at ip-172-31-50-127.ec2.internal/172.31.50.127:8032
22/12/06 07:46:51 INFO tools.SimpleCopyListing: Paths (files+dirs) cnt = 1; dirCnt = 0
22/12/06 07:46:51 INFO tools.SimpleCopyListing: Build file listing completed.
22/12/06 07:46:51 INFO Configuration.deprecation: io.sort.mb is deprecated. Instead, use mapreduce.task.io.sort.mb
22/12/06 07:46:51 INFO Configuration.deprecation: io.sort.factor is deprecated. Instead, use mapreduce.task.io.sort.factor
22/12/06 07:46:51 INFO tools.DistCp: Number of paths in the copy list: 1
22/12/06 07:46:51 INFO tools.DistCp: Number of paths in the copy list: 1
22/12/06 07:46:51 INFO client.RMProxy: Connecting to ResourceManager at ip-172-31-50-127.ec2.internal/172.31.50.127:8032
22/12/06 07:46:52 INFO mapreduce.JobSubmitter: number of splits:1
22/12/06 07:46:52 INFO mapreduce.JobSubmitter: Submitting tokens for job: job_1670308675445_0001
22/12/06 07:46:53 INFO impl.YarnClientImpl: Submitted application application_1670308675445_0001
22/12/06 07:46:53 INFO mapreduce.Job: The url to track the job: http://ip-172-31-50-127.ec2.internal:20888/proxy/application_1670
308675445_0001/
22/12/06 07:46:53 INFO tools.DistCp: DistCp job-id: job_1670308675445_0001
22/12/06 07:46:53 INFO mapreduce.Job: Running job: job_1670308675445_0001
22/12/06 07:47:04 INFO mapreduce.Job: Job job_1670308675445_0001 running in uber mode : false
22/12/06 07:47:04 INFO mapreduce.Job:  map 0% reduce 0%
22/12/06 07:47:23 INFO mapreduce.Job:  map 100% reduce 0%
22/12/06 07:47:27 INFO mapreduce.Job: Job job_1670308675445_0001 completed successfully
22/12/06 07:47:28 INFO mapreduce.Job: Counters: 38
        File System Counters
                FILE: Number of bytes read=0
                FILE: Number of bytes written=172831
                FILE: Number of read operations=0
                FILE: Number of large read operations=0
                FILE: Number of write operations=0
                HDFS: Number of bytes read=353
                HDFS: Number of bytes written=545839412
                HDFS: Number of read operations=12
                HDFS: Number of large read operations=0
                HDFS: Number of write operations=4
                S3: Number of bytes read=545839412
                S3: Number of bytes written=0
                S3: Number of read operations=0
                S3: Number of large read operations=0
                S3: Number of write operations=0
        Job Counters
                Launched map tasks=1
                Other local map tasks=1
                Total time spent by all maps in occupied slots (ms)=643040
                Total time spent by all reduces in occupied slots (ms)=0
                Total time spent by all map tasks (ms)=20095
```

```
[hadoop@ip-172-31-50-127 ~]$ hadoop distcp 's3://hivebucketstudy/2019-Oct.csv' /hiveassignment/2019-Oct.csv
22/12/06 07:48:03 INFO tools.DistCp: Input Options: DistCpOptions{atomicCommit=false, syncFolder=false, deleteMissing=false, igno
reFailures=false, overwrite=false, skipCRC=false, blocking=true, numListstatusThreads=0, maxMaps=20, mapBandwidth=100, sslConfigu
rationFile='null', copyStrategy='uniformsize', preserveStatus=[], preserveRawXattrs=false, atomicWorkPath=null, logPath=null, sou
rceFileListing=null, sourcePaths=[s3://hivebucketstudy/2019-Oct.csv], targetPath=/hiveassignment/2019-Oct.csv, targetPathExists=f
alse, filtersFile='null'}
22/12/06 07:48:04 INFO client.RMProxy: Connecting to ResourceManager at ip-172-31-50-127.ec2.internal/172.31.50.127:8032
22/12/06 07:48:08 INFO tools.SimpleCopyListing: Paths (files+dirs) cnt = 1; dirCnt = 0
22/12/06 07:48:08 INFO tools.SimpleCopyListing: Build file listing completed.
22/12/06 07:48:08 INFO Configuration.deprecation: io.sort.mb is deprecated. Instead, use mapreduce.task.io.sort.mb
22/12/06 07:48:08 INFO Configuration.deprecation: io.sort.factor is deprecated. Instead, use mapreduce.task.io.sort.factor
22/12/06 07:48:08 INFO tools.DistCp: Number of paths in the copy list: 1
22/12/06 07:48:08 INFO tools.DistCp: Number of paths in the copy list: 1
22/12/06 07:48:08 INFO client.RMProxy: Connecting to ResourceManager at ip-172-31-50-127.ec2.internal/172.31.50.127:8032
22/12/06 07:48:10 INFO mapreduce.JobSubmitter: number of splits:1
22/12/06 07:48:10 INFO mapreduce.JobSubmitter: Submitting tokens for job: job_1670308675445_0002
22/12/06 07:48:10 INFO impl.YarnClientImpl: Submitted application application_1670308675445_0002
22/12/06 07:48:10 INFO mapreduce.Job: The url to track the job: http://ip-172-31-50-127.ec2.internal:20888/proxy/application_1670
308675445_0002/
22/12/06 07:48:10 INFO tools.DistCp: DistCp job-id: job_1670308675445_0002
22/12/06 07:48:10 INFO mapreduce.Job: Running job: job_1670308675445_0002
22/12/06 07:48:19 INFO mapreduce.Job: Job job_1670308675445_0002 running in uber mode : false
22/12/06 07:48:19 INFO mapreduce.Job:  map 0% reduce 0%
22/12/06 07:48:37 INFO mapreduce.Job:  map 100% reduce 0%
22/12/06 07:48:40 INFO mapreduce.Job: Job job_1670308675445_0002 completed successfully
22/12/06 07:48:40 INFO mapreduce.Job: Counters: 38
        File System Counters
                FILE: Number of bytes read=0
                FILE: Number of bytes written=172837
                FILE: Number of read operations=0
                FILE: Number of large read operations=0
                FILE: Number of write operations=0
                HDFS: Number of bytes read=355
                HDFS: Number of bytes written=482542278
                HDFS: Number of read operations=12
                HDFS: Number of large read operations=0
                HDFS: Number of write operations=4
                S3: Number of bytes read=482542278
                S3: Number of bytes written=0
                S3: Number of read operations=0
```

Checking the loaded files:

```
hadoop fs -ls /hiveassignment
```

```
hadoop@ip-172-31-50-127 ~]$ hadoop fs -ls /hiveassignment
ound 2 items
rw-r--r--   1 hadoop hadoop  545839412 2022-12-06 07:47 /hiveassignment/2019-Nov.csv
rw-r--r--   1 hadoop hadoop  482542278 2022-12-06 07:48 /hiveassignment/2019-Oct.csv
```

We can confirm that the datasets were loaded successfully.

Launching HIVE:

```
[hadoop@ip-172-31-50-127 ~]$ hive

Logging initialized using configuration in file:/etc/hive/conf.dist/hive-log4j2.properties Async: false
hive>
```

Creating Database "upgrad_assignment":

```
hive> Create database if not exists upgrad_assignment;
OK
Time taken: 1.179 seconds
```

```
Create database if not exists upgrad_assignment;
use upgrad_assignment;
```

```
hive> use upgrad_assignment;
OK
Time taken: 0.052 seconds
```

Creating an External Table, Sales:

```
create External table if not exists sales(event_time timestamp,event_type
string,product_id string,category_id string,category_code string,brand string,price float,
user_id bigint,user_session string) ROW FORMAT SERDE
'org.apache.hadoop.hive.serde2.OpenCSVSerde'
WITH SERDEPROPERTIES ("separatorChar"=",","quoteChar"="\"","escapeChar"="\\")
stored as textfile
```

```
hive> create external table if not exists ecom (event_time TIMESTAMP ,
    > event_type STRING , product_id STRING , category_id STRING ,
    > category_code STRING , brand STRING , price FLOAT , user_id BIGINT ,
    > user_session STRING ) COMMENT 'ecom table' ROW FORMAT DELIMITED
    > FIELDS TERMINATED BY ',' lines terminated by '\n' stored as textfile;
OK
Time taken: 1.204 seconds
hive>
```

```
Location
'/hiveassignment' TBLPROPERTIES("skip.header.line.count"="1");
desc sales;
```

```
hive> desc ecom;
OK
event_time              timestamp
event_type              string
product_id              string
category_id             string
category_code           string
brand                   string
price                   float
user_id                 bigint
user_session            string
Time taken: 0.19 seconds, Fetched: 9 row(s)
```

Loading the data into the table:

```
hive> load data inpath '/hiveassignment/2019-Oct.csv' into table sales;
hive> load data inpath '/hiveassignment/2019-Nov.csv' into table sales;
```

```
hive>  load data inpath '/hiveassignment/2019-Oct.csv' into table sales;
Loading data to table upgrad_assignment.sales
OK
Time taken: 3.264 seconds
```

```
hive> load data inpath '/hiveassignment/2019-Nov.csv' into table sales;
Loading data to table upgrad_assignment.sales
OK
Time taken: 1.149 seconds
```

Q1. Find the total revenue generated due to purchases made in October.

```
hive> set hive.cli.print.header=true;
hive> select sum(price) from sales where Month(event_time)=10 and
event_type='purchase';
```

```
hive> load data inpath '/hiveassignment/2019-Oct.csv' into table sales;
Loading data to table upgrad_assignment.sales
OK
Time taken: 1.264 seconds
hive> load data inpath '/hiveassignment/2019-Nov.csv' into table sales;
Loading data to table upgrad_assignment.sales
OK
Time taken: 0.479 seconds
hive> set hive.cli.print.header=true;
hive> select sum(price) from sales where Month(event_time)=10 and event_type='purchase';
Query ID = hadoop_20221206091542_e78b8a22-abb8-478a-bf05-480bdd8ec247
Total jobs = 1
Launching Job 1 out of 1
Status: Running (Executing on YARN cluster with App id application_1670308675445_0015)

--------------------------------------------------------------------------------
        VERTICES      MODE        STATUS  TOTAL  COMPLETED  RUNNING  PENDING  FAILED  KILLED
--------------------------------------------------------------------------------
Map 1 .......... container     SUCCEEDED     12         12        0        0       0       0
Reduce 2 ...... container     SUCCEEDED      1          1        0        0       0       0
--------------------------------------------------------------------------------
VERTICES: 02/02  [==========================>>] 100%  ELAPSED TIME: 56.89 s
--------------------------------------------------------------------------------
OK
_c0
1211538.4300000328
Time taken: 61.857 seconds, Fetched: 1 row(s)
hive>
```

Here the query takes 61.85 seconds which can be optimized by creating dynamic partition and then compare the execution time.

Dynamic Partitioning and Bucketing:

```
hive> set hive.exec.dynamic.partition=true;
hive> set hive.exec.dynamic.partition.mode=nonstrict;
```

```
hive> set hive.exec.dynamic.partition=true;
hive> set hive.exec.dynamic.partition.mode=nonstrict;
```

Creating a new table by name sales_dp to store the dataset which we partitioned by using 'event_type' and clustered by 'user_id'.

Desc sales_dp;

```
hive> create external table if not exists sales_dp (event_time TIMESTAMP ,event_type STRING , product_id STRING , category_id STRING ,category_
code STRING , brand STRING , price FLOAT , user_id BIGINT ,user_session STRING ) partitioned by (event_type string) clustered by(user_id)into 5
 buckets ROW FORMAT SERDE 'org.apache.hadoop.hive.serde2.OpenCSVSerde' stored as textfile;
OK
Time taken: 0.023 seconds
hive> desc sales_dp;
OK
col_name        data_type       comment
event_time              string                  from deserializer
product_id              string                  from deserializer
category_id             string                  from deserializer
category_code           string                  from deserializer
brand                   string                  from deserializer
price                   string                  from deserializer
user_id                 string                  from deserializer
user_session            string                  from deserializer
event_type              string

# Partition Information
# col_name              data_type               comment

event_type              string
Time taken: 0.104 seconds, Fetched: 14 row(s)
```

Loading the data into the new table, sales_dp from the old sales table:

```
insert into sales_dp partition(event_type) select event_time, product_id, category_id,
category_code, brand, price, user_id, user_session, event_type from sales
```

```
hive> create external table if not exists sales_dp (event_time TIMESTAMP ,event_type STRING , product_id STRING , category_id STRING ,category_
code STRING , brand STRING , price FLOAT , user_id BIGINT ,user_session STRING ) partitioned by (event_type string) clustered by(user_id)into 5
 buckets ROW FORMAT SERDE 'org.apache.hadoop.hive.serde2.OpenCSVSerde' stored as textfile;
OK
Time taken: 0.023 seconds
hive> desc sales_dp;
OK
col_name        data_type       comment
event_time              string                  from deserializer
product_id              string                  from deserializer
category_id             string                  from deserializer
category_code           string                  from deserializer
brand                   string                  from deserializer
price                   string                  from deserializer
user_id                 string                  from deserializer
user_session            string                  from deserializer
event_type              string

# Partition Information
# col_name              data_type               comment

event_type              string
Time taken: 0.104 seconds, Fetched: 14 row(s)
```

Executing the same query again for Q1.

```
select sum(price) from sales_dp where Month(event_time)=10 and
event_type='purchase';
```

```
hive> select sum(price) from sales_dp where Month(event_time)=10 and
    > event_type='purchase';
Query ID = hadoop_20221206092751_8373c2e8-ebff-4812-8be2-d0f3d68b21e8
Total jobs = 1
Launching Job 1 out of 1
Status: Running (Executing on YARN cluster with App id application_1670308675445_0016)

--------------------------------------------------------------------------------
        VERTICES      MODE        STATUS  TOTAL  COMPLETED  RUNNING  PENDING  FAILED  KILLED
--------------------------------------------------------------------------------
Map 1 .......... container     SUCCEEDED      4          4        0        0       0       0
Reducer 2 ...... container     SUCCEEDED      1          1        0        0       0       0
--------------------------------------------------------------------------------
VERTICES: 02/02  [==========================>>] 100%  ELAPSED TIME: 28.33 s
--------------------------------------------------------------------------------
OK
_c0
1211538.4299999247
Time taken: 29.449 seconds, Fetched: 1 row(s)
```

Notice how the time taken has reduced drastically due to partitioning and bucketing. Earlier it took almost 57 seconds for the query to run however now it took only 28.33 seconds.

Answer: The total sales in the month of October is 1211538.42

Q2. Write a query to yield the total sum of purchases per month in a single output.

```
select Month(event_time) as Month, sum(price) as sum, COUNT(event_type) as cnt
from sales where event_type='purchase' group by Month(event_time);
```

```
hive> select Month(event_time) as Month, sum(price) as sum, COUNT(event_type) as cnt from sales where event_type='purchase' group by Month(event_time);
Query ID = hadoop_20221206092920_de8f6b50-3fc8-4d44-aclf-132153e0668b
Total jobs = 1
Launching Job 1 out of 1
Status: Running (Executing on YARN cluster with App id application_1670308675445_0016)

--------------------------------------------------------------------------------
        VERTICES      MODE        STATUS  TOTAL  COMPLETED  RUNNING  PENDING  FAILED  KILLED
--------------------------------------------------------------------------------
Map 1 .......... container     SUCCEEDED     12         12        0        0       0       0
Reducer 2 ...... container     SUCCEEDED      6          6        0        0       0       0
--------------------------------------------------------------------------------
VERTICES: 02/02  [==========================>>] 100%  ELAPSED TIME: 53.27 s
--------------------------------------------------------------------------------
OK
month   sum     cnt
11      1531016.9000000155      322417
10      1211538.4300000328      245624
Time taken: 53.927 seconds, Fetched: 2 row(s)
hive>
```

Answer:
In the month of October the total purchases are 245624 and sales is 1211538.42
In the month of November the total purchases are 322417 and sales is 1531016.90

Q3. Write a query to find the change in revenue generated due to purchases from October to November.

```
WITH Monthly_Revenue AS ( SELECT SUM(CASE WHEN date_format(event_time, 'MM')=10 THEN
price ELSE 0 END) AS Oct_Revenue, SUM(CASE WHEN date_format(event_time, 'MM')=11 THEN
price ELSE 0 END) AS Nov_Revenue FROM sales WHERE event_type= 'purchase' AND
date_format(event_time, 'MM') in ('10', '11') ) SELECT Nov_Revenue, Oct_Revenue,
(Nov_Revenue - Oct_Revenue) AS Revenue_Difference FROM Monthly_Revenue;
```

```
hive> WITH Monthly_Revenue AS ( SELECT SUM(CASE WHEN date_format(event_time, 'MM')=10 THEN price ELSE 0 END) AS Oct_Revenue, SUM(CASE WHEN date_format(event_time, 'MM')=11
 THEN price ELSE 0 END) AS Nov_Revenue FROM sales WHERE event_type= 'purchase' AND date_format(event_time, 'MM') in ('10', '11') ) SELECT Nov_Revenue, Oct_Revenue, (Nov_Re
venue - Oct_Revenue) AS Revenue_Difference FROM Monthly_Revenue;
Query ID = hadoop_20221206093920_ffa53a28-0207-4610-86da-787c748cf4fa
Total jobs = 1
Launching Job 1 out of 1
Status: Running (Executing on YARN cluster with App id application_1670308675445_0016)

--------------------------------------------------------------------------------------------------
        VERTICES      MODE        STATUS  TOTAL  COMPLETED  RUNNING  PENDING  FAILED  KILLED
--------------------------------------------------------------------------------------------------
Map 1 ..........  container     SUCCEEDED     12         12        0        0       0       0
Reducer 2 ......  container     SUCCEEDED      1          1        0        0       0       0
--------------------------------------------------------------------------------------------------
VERTICES: 02/02  [==========================>>] 100%  ELAPSED TIME: 79.71 s
--------------------------------------------------------------------------------------------------
OK
nov_revenue     oct_revenue     revenue_difference
1531016.9000000157      1211538.4300000328      319478.469999983
Time taken: 80.258 seconds, Fetched: 1 row(s)
```

Answer: We can see that the difference in the revenue is 319478.47

Q4. Find distinct categories of products. Categories with null category code can be ignored.

```
select distinct split(category_code,'\\.')[0] as Categories from sales_dp where
split(category_code,'\\.')[0]<>'';
```

```
hive> select distinct split(category_code,'\\.')[0] as Categories from sales_dp where
    > split(category_code,'\\.')[0]<>'';
Query ID = hadoop_20220102071052_af73873d-0f35-414f-8bf1-a299f35b0985
Total jobs = 1
Launching Job 1 out of 1
Status: Running (Executing on YARN cluster with App id application_1641103761278_0005)

--------------------------------------------------------------------------------
        VERTICES      MODE        STATUS  TOTAL  COMPLETED  RUNNING  PENDING  FAILED  KILLED
--------------------------------------------------------------------------------
Map 1 .......... container     SUCCEEDED     14         14        0        0       0       0
Reduce 2 ...... container     SUCCEEDED      5          5        0        0       0       0
--------------------------------------------------------------------------------
VERTICES: 02/02  [==========================>>] 100%  ELAPSED TIME: 57.04 s
--------------------------------------------------------------------------------
OK
categories
furniture
appliances
accessories
apparel
sport
stationery
Time taken: 57.803 seconds, Fetched: 6 row(s)
```

Answer: We can see that the distinct categories are Furniture, Appliances, Accessories, Apparel, sport and stationery.

Q5. Find the total number of products available under each category.

```
select split(category_code,'\\.')[0] as category , count(product_id) as Prodcount from
sales group by split(category_code,'\\.')[0] order by Prodcount desc;
```

```
hive> select split(category_code,'\\.')[0] as category , count(product_id) as Prodcount from sales group by split(category_code,'\\.')[0] order by Prodcount desc;
Query ID = hadoop_20221206094955_30e51cff-c2df-4194-89f1-d529af4c5258
Total jobs = 1
Launching Job 1 out of 1
Status: Running (Executing on YARN cluster with App id application_1670308675445_0016)

--------------------------------------------------------------------------------------------
        VERTICES      MODE        STATUS  TOTAL  COMPLETED  RUNNING  PENDING  FAILED  KILLED
--------------------------------------------------------------------------------------------
Map 1 .......... container     SUCCEEDED     12         12        0        0       0       0
Reducer 2 ...... container     SUCCEEDED      1          1        0        0       0       0
Reducer 3 ...... container     SUCCEEDED      1          1        0        0       0       0
--------------------------------------------------------------------------------------------
VERTICES: 03/03  [==========================>>] 100%  ELAPSED TIME: 83.64 s
--------------------------------------------------------------------------------------------
OK
category        prodcount
        8594895
appliances      61736
stationery      26722
furniture       23604
apparel 18232
accessories     12929
category_code   2
sport   2
Time taken: 84.299 seconds, Fetched: 8 row(s)
```

Answer: Total number of products under each category is as follows: Appliances – 61736 ; Stationery – 26722; Furniture – 23604; Apparel – 18232; Accessories – 12929 and sport – 2.

Q6. Which brand had the maximum sales in October and November combined?

```
WITH Max_Sales_Brand AS ( SELECT brand, SUM(CASE WHEN date_format(event_time, 'MM')=10
THEN price ELSE 0 END) AS Oct_Sales, SUM(CASE WHEN date_format(event_time, 'MM')=11 THEN
price ELSE 0 END) AS Nov_Sales FROM sales WHERE ( event_type='purchase'AND
date_format(event_time, 'MM') in ('10','11') AND brand <> '') GROUP BY brand ) SELECT
brand, Nov_Sales + Oct_Sales AS Total_Sales FROM Max_Sales_Brand ORDER BY Total_Sales DESC
LIMIT 1;
```

```
hive> WITH Max_Sales_Brand AS ( SELECT brand, SUM(CASE WHEN date_format(event_time, 'MM')=10 THEN price ELSE 0 END) AS Oct_Sales, SUM(CASE WHEN date_format(event_time, 'MM
')=11 THEN price ELSE 0 END) AS Nov_Sales FROM sales WHERE ( event_type='purchase'AND date_format(event_time, 'MM') in ('10','11') AND brand <> '') GROUP BY brand ) SELECT
 brand, Nov_Sales + Oct_Sales AS Total_Sales FROM Max_Sales_Brand ORDER BY Total_Sales DESC LIMIT 1;
Query ID = hadoop_20221206095538_c38b5cd2-fb92-4135-8f1f-a75fb8c69f79
Total jobs = 1
Launching Job 1 out of 1
Status: Running (Executing on YARN cluster with App id application_1670308675445_0016)

--------------------------------------------------------------------------------------------------
        VERTICES      MODE        STATUS  TOTAL  COMPLETED  RUNNING  PENDING  FAILED  KILLED
--------------------------------------------------------------------------------------------------
Map 1 .......... container     SUCCEEDED     12         12        0        0       0       0
Reducer 2 ...... container     SUCCEEDED      4          4        0        0       0       0
Reducer 3 ...... container     SUCCEEDED      1          1        0        0       0       0
--------------------------------------------------------------------------------------------------
VERTICES: 03/03  [==========================>>] 100%  ELAPSED TIME: 90.42 s
--------------------------------------------------------------------------------------------------
OK
brand   total_sales
runail  148297.94000000047
Time taken: 91.431 seconds, Fetched: 1 row(s)
```

Answer: We can see that Runail is the brand with the maximum sales for October and November combined. The total sales is 148297.94

Q7. Which brands increased their sales from October to November?

```
with CTE2 as(select brand, sum(case when month(event_time)=10 then price else 0
end) as Oct,sum(case when month(event_time)=11 then price else 0 end) as Nov from
sales_dp where event_type='purchase' group by brand) select brand , Oct,Nov,(Nov-Oct)
as diff from CTE2 where (Nov-Oct)>0 ORDER BY diff;
```

```
hive> with CTE2 as(select brand, sum(case when month(event_time)=10 then price else 0 end) as Oct,sum(case when month(event_time)=11 then price else 0 end) as Nov from sales_dp where event_type='purchase' group by brand) select brand , O
ct,Nov,(Nov-Oct) as diff from CTE2 where (Nov-Oct)>0 ORDER BY diff;
Query ID = hadoop_20221206100109_bdd7b1e7-a176-4b37-b0c8-e7762b4e2153
Total jobs = 1
Launching Job 1 out of 1
Status: Running (Executing on YARN cluster with App id application_1670308675445_0016)

--------------------------------------------------------------------------------
        VERTICES      MODE        STATUS  TOTAL  COMPLETED  RUNNING  PENDING  FAILED  KILLED
--------------------------------------------------------------------------------
Map 1 ......... container     SUCCEEDED      4          4        0        0       0       0
Reducer 2 ...... container     SUCCEEDED      2          2        0        0       0       0
Reducer 3 ...... container     SUCCEEDED      1          1        0        0       0       0
--------------------------------------------------------------------------------
VERTICES: 03/03  [==========================>>] 100%  ELAPSED TIME: 32.60 s
--------------------------------------------------------------------------------
OK
brand   oct     nov     diff
ovale   2.54    3.1     0.56
cosima  20.229999999999997      20.929999999999996      0.6999999999999993
grace   100.91999999999997      102.61000000000001      1.6900000000000404
helloganic      0.0     3.1     3.1
skinity 8.88    12.440000000000001      3.5600000000000005
bodyton 1376.3400000000004      1380.6400000000003      4.2999999999999545
moyou   5.71    10.280000000000001      4.570000000000001
neoleor 43.41   51.7    8.290000000000006
soleo   204.1999999999997       212.5299999999998       8.330000000000098
jaguar  1102.1100000000001      1110.65 8.539999999999964
tertio  236.16000000000003      245.79999999999998      9.639999999999958
fly     17.14   27.17   10.030000000000001
rasyan  18.799999999999997      28.939999999999998      10.14
deoproce        316.84  329.17  12.330000000000041
barbie  0.0     12.39   12.39
supertan        50.370000000000005      66.50999999999999       16.139999999999986
treaclemoon     163.37  181.49  18.120000000000005
kamill  63.00999999999999       81.49000000000002       18.480000000000032
juno    0.0     21.08   21.08
veraclara       50.11   71.21   21.099999999999994
glysolid        69.72999999999999       91.58999999999999       21.86
godefroy        401.22  425.12  23.899999999999977
binacil 0.0     24.259999999999998      24.259999999999998
blixz   38.949999999999996      63.400000000000006      24.45000000000001
profepil        93.36000000000001       118.02000000000001      24.659999999999997
estelare        444.8100000000002       471.8700000000001       27.059999999999945
orly    902.38  931.0900000000001       28.71000000000015
biore   60.65   90.31   29.660000000000004
beautyblender   78.74000000000001       109.41  30.669999999999987
vilenta 197.59999999999997      231.20999999999998      33.610000000000014
mavala  409.0400000000001       446.32000000000005      37.27999999999997
likato  296.06  340.97  44.910000000000025
ladykin 125.65  170.57  44.91999999999999
foamie  35.04   80.49   45.449999999999996
elskin  251.09000000000003      307.65  56.559999999999945
balbcare        155.33000000000004      212.38000000000005      57.05000000000001
koelcia 55.49999999999999       112.75000000000003      57.250000000000036
profhenna       679.2299999999999       736.8499999999999       57.620000000000005
kares   0.0     59.449999999999996      59.449999999999996
marutaka-foot   49.22   109.33  60.11
dewal   0.0     61.28999999999999       61.28999999999999
inm     288.02  351.2099999999999       63.18999999999994
laboratorium    246.5   312.52000000000004      66.02000000000004
```

cutrin	299.37	367.62	68.25
egomania	77.47	146.04	68.57
konad	739.8299999999999	810.6699999999997	70.8399999999998
nirvel	163.04	234.32999999999998	71.28999999999999
koelf	422.7299999999999	507.2899999999999	84.56
plazan	101.36999999999999	194.00999999999996	92.63999999999997
aura	83.95	177.51	93.55999999999999
kerasys	430.9100000000001	525.1999999999999	94.28999999999985
enjoy	41.35	136.57	95.22
depilflax	2707.069999999998	2803.779999999999	96.71000000000095
eos	54.339999999999996	152.60999999999999	98.26999999999998
carmex	145.08	243.35999999999999	98.27999999999997
batiste	772.4	874.1699999999998	101.76999999999987
osmo	645.5799999999999	762.31	116.73000000000002
dizao	819.1300000000001	945.5100000000004	126.38000000000034
igrobeauty	513.6600000000001	645.0700000000002	131.41000000000008
finish	98.38	230.38	132.0
nefertiti	233.52000000000004	366.64	133.11999999999995
elizavecca	70.53	204.3	133.77
miskin	158.04000000000002	293.07	135.02999999999997
latinoil	249.51999999999998	384.59	135.07
farmona	1692.46	1843.4300000000003	150.97000000000025
cristalinas	427.63	584.9499999999998	157.31999999999982
chi	358.94	538.6100000000001	179.67000000000013
matreshka	0.0	182.66999999999996	182.66999999999996
freshbubble	318.69999999999993	502.34	183.64000000000004
mane	66.78999999999999	260.26	193.47
keen	236.35	435.62	199.27
ecocraft	41.160000000000004	241.95	200.79
fedua	52.38	263.81000000000006	211.43000000000006
provoc	827.9899999999996	1063.8199999999993	235.8299999999997
skinlite	651.9400000000004	890.45	238.50999999999965
entity	479.71000000000095	719.2600000000001	239.54999999999916
trind	298.07000000000005	542.96	244.89
protokeratin	201.25	456.79	255.54000000000002
beauugreen	511.5100000000001	768.3499999999999	256.8399999999998
bluesky	10307.239999999887	10565.529999999948	258.2900000000609
candy	534.9599999999999	799.3800000000001	264.4200000000002
insight	1443.7000000000003	1721.9600000000005	278.2600000000002
kocostar	310.84999999999997	594.9300000000001	284.0800000000001
happyfons	801.9200000000005	1091.5900000000004	289.66999999999985
kims	330.03999999999996	632.0400000000001	302.0000000000001
shary	871.9599999999997	1176.489999999999	304.5299999999994
nitrile	847.2799999999999	1162.6799999999998	315.4
lowence	242.84000000000003	567.75	324.90999999999997
jas	3318.9600000000023	3657.4300000000003	338.469999999998
ellips	245.85000000000002	606.0399999999998	360.1899999999998
lador	2083.6100000000015	2471.5300000000025	387.920000000001
naomi	0.0	389.0	389.0
kiss	421.55	817.3299999999999	395.7799999999999
yu-r	271.40999999999997	673.71	402.30000000000007
sophin	1067.8600000000006	1515.5200000000002	447.6599999999996
farmavita	837.3700000000001	1291.97	454.5999999999999
bioaqua	942.8900000000002	1398.1199999999994	455.2299999999992
greymy	29.21	489.49000000000007	460.2800000000001
gehwol	1089.0700000000002	1557.68	468.6099999999999
matrix	3243.25	3726.7400000000007	483.4900000000007
limoni	1308.9	1796.6000000000004	487.7000000000003
s.care	412.68	913.07	500.39000000000004
coifin	903.0	1428.4899999999998	525.4899999999998
uskusi	5142.270000000011	5690.3100000000095	548.0399999999981
airnails	5118.899999999989	5691.520000000004	572.6200000000154
browxenna	14331.36999999999	14916.73	585.3600000000097

```
kinetics        6334.250000000018       6945.260000000017       611.0099999999984
kosmekka        1181.44 1813.37 631.9299999999998
kaaral  4412.430000000002       5086.0700000000015      673.6399999999994
refectocil      2716.180000000002       3475.5800000000036      759.4000000000015
rosi    3077.040000000001       3841.560000000002       764.5200000000013
solomeya        1899.6999999999994      2685.799999999998       786.0999999999985
missha  1293.83 2150.28 856.4500000000003
levissime       2227.500000000004       3085.31000000001        857.8100000000059
art-visage      2092.7100000000023      2997.800000000002       905.0899999999997
ecolab  262.84999999999997      1214.3  951.45
nagaraku        4369.740000000009       5327.6799999999985      957.9399999999896
sanoto  157.14  1209.6799999999998      1052.54
markell 1768.749999999998       2834.4300000000003      1065.6800000000023
metzger 5373.450000000001       6457.159999999996       1083.7099999999955
de.lux  1659.6999999999987      2775.5099999999898      1115.809999999991
swarovski       1887.9299999999967      3043.1599999999944      1155.2299999999977
beauty-free     554.17  1782.8600000000008      1228.690000000001
zeitun  708.6600000000002       2009.6299999999999      1300.9699999999998
joico   705.52  2015.1  1309.58
severina        4775.879999999984       6120.479999999987       1344.600000000003
irisk   45591.9599999991       46946.03999999952       1354.0800000004238
oniq    8425.409999999998       9841.65 1416.2400000000016
levrana 2243.5600000000004      3664.100000000002       1420.5400000000018
roubloff        3491.3600000000024      4913.770000000005       1422.4100000000026
smart   4457.259999999995       5902.139999999994       1444.8799999999992
shik    3341.2000000000025      4839.720000000001       1498.5199999999986
domix   10472.04999999999       12009.169999999978      1537.119999999988
artex   2730.6399999999994      4327.250000000001       1596.6100000000015
beautix 10493.949999999992      12222.949999999995      1729.0000000000036
milv    3904.9399999999823      5642.009999999997       1737.0700000000143
masura  31266.079999999118      33058.46999999959       1792.3900000004687
f.o.x   6624.229999999996       8577.27999999999        1953.0499999999938
kapous  11927.159999999954      14093.080000000005      2165.920000000051
concept 11032.139999999989      13380.399999999949      2348.25999999996
estel   21756.749999999956      24142.669999999976      2385.92000000002
kaypro  881.34  3268.7  2387.3599999999997
benovy  409.6199999999999       3259.970000000001       2850.3500000000013
italwax 21940.239999999885      24799.3699999999        2859.1300000000156
yoko    8756.909999999996       11707.879999999997      2950.970000000001
haruyama        9390.68999999989        12352.909999999987      2962.2200000000976
marathon        7280.750000000002       10273.1 2992.3499999999985
lovely  8704.379999999994       11939.059999999978      3234.679999999984
bpw.style       11572.150000000041      14837.44000000022       3265.290000000179
staleks 8519.730000000016       11875.610000000008      3355.879999999992
freedecor       3421.7799999999825      7671.799999999975       4250.019999999992
runail  71539.28000000014       76758.66000000013       5219.37999999999
polarus 6013.72 11371.93        5358.21
cosmoprofi      8322.809999999994       14536.990000000034      6214.18000000004
jessnail        26287.84000000011       33345.23000000013       7057.390000000018
strong  29196.629999999997      38671.27000000002       9474.640000000021
ingarden        23161.38999999976       33566.209999999985      10404.820000000225
lianail 5892.839999999988       16394.239999999976      10501.399999999987
uno     35302.02999999996       51039.750000000095      15737.720000000132
grattol 35445.53999999991       71472.7100000031        36027.17000000319
        474679.06000001205      619509.240000016        144830.180000004
Time taken: 33.175 seconds, Fetched: 161 row(s)
hive>
```

Answer: From the output we can see that 161 brands were able to increase their sales from the month of October to November.

Q8. Your company wants to reward the top 10 users of its website with a Golden Customer plan. Write a query to generate a list of top 10 users who spend the most.

```
select user_id, sum(price) as Totalpurchases from sales_dp where event_type='purchase'
group by user_id order by Totalpurchases DESC limit 10;
```

```
hive> select user_id, sum(price) as Totalpurchases from sales_dp where event_type='purchase'group by user_id order by Totalpurchases DESC limit 10;
Query ID = hadoop_20221206100500_0fe3b821-d2dc-4d78-866f-365b0970698e
Total jobs = 1
Launching Job 1 out of 1
Status: Running (Executing on YARN cluster with App id application_1670308675445_0016)

--------------------------------------------------------------------------------------------------
        VERTICES      MODE        STATUS  TOTAL  COMPLETED  RUNNING  PENDING  FAILED  KILLED
--------------------------------------------------------------------------------------------------
Map 1 .......... container     SUCCEEDED      4          4        0        0       0       0
Reducer 2 ...... container     SUCCEEDED      2          2        0        0       0       0
Reducer 3 ...... container     SUCCEEDED      1          1        0        0       0       0
--------------------------------------------------------------------------------------------------
VERTICES: 03/03  [==========================>>] 100%  ELAPSED TIME: 27.91 s
--------------------------------------------------------------------------------------------------
OK
user_id totalpurchases
557790271       5431.739999999988
150318419       3291.939999999999
562167663       2705.6999999999994
531900924       2658.8999999999996
557850743       2590.959999999999
522130011       2370.7800000000025
561592095       2219.4000000000005
431950134       2195.1799999999985
566576008       2112.7200000000025
521347209       2081.8199999999993
Time taken: 28.651 seconds, Fetched: 10 row(s)
hive>
```

INSIGHTS OF QUERY 8

- We ran the same query from Question No. 8 on this table after constructing an optimised table by partitioning on the "event type" attribute and bucketing (clustering) on the "price."

- We may achieve the same outcome that we did when we executed on the Base table (Non-Optimized table).

- The execution time of the identical query has significantly decreased, falling from 69.753 seconds to 27.634 seconds—a difference of 42.119 seconds—which is the second and most crucial thing we can see.

- Consequently, by properly partitioning and bucketing the table, we can shorten the query's execution time.

Finishing up

Once we are done with the analysis, we can drop the databases and quit hive and then terminate the EMR Cluster.

```
hive> SHOW DATABASES;
OK
database_name
default
upgrad_assignment
Time taken: 0.022 seconds, Fetched: 2 row(s)
hive>
hive> drop DATABASE upgrad_assignment CASCADE;
OK
Time taken: 0.857 seconds
hive> SHOW DATABASES;
OK
database_name
default
Time taken: 0.009 seconds, Fetched: 1 row(s)
hive>
```

Create cluster | View details | Clone | Terminate

Filter: All clusters | Filter clusters ... | 20 clusters (all loaded)

Name	ID	Status	Creation time (UTC+5:30)	Elapsed time	Normalized instance hours
hive assignment	j-3TJV2HDU0Q2DO	Terminating User request	2022-12-06 11:57 (UTC+5:30)	3 hours, 42 minutes	36