



Hive Case Study

Samanyu Ghose



Problem Statement

With online sales gaining popularity, tech companies are exploring ways to improve their sales by analyzing customer behavior and gaining insights about product trends. Furthermore, the websites make it easier for customers to find the products that they require without much scavenging. Goal is to, extract data and gather insights from a real-life data set of an e-commerce company.

1. Find the total revenue generated due to purchases made in October.
2. Write a query to yield the total sum of purchases per month in a single output.
3. Write a query to find the change in revenue generated due to purchases from October to November.
4. Find distinct categories of products.
5. Find the total number of products available under each category.
6. Which brand had the maximum sales in October and November combined?
7. Which brands increased their sales from October to November?
8. 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.

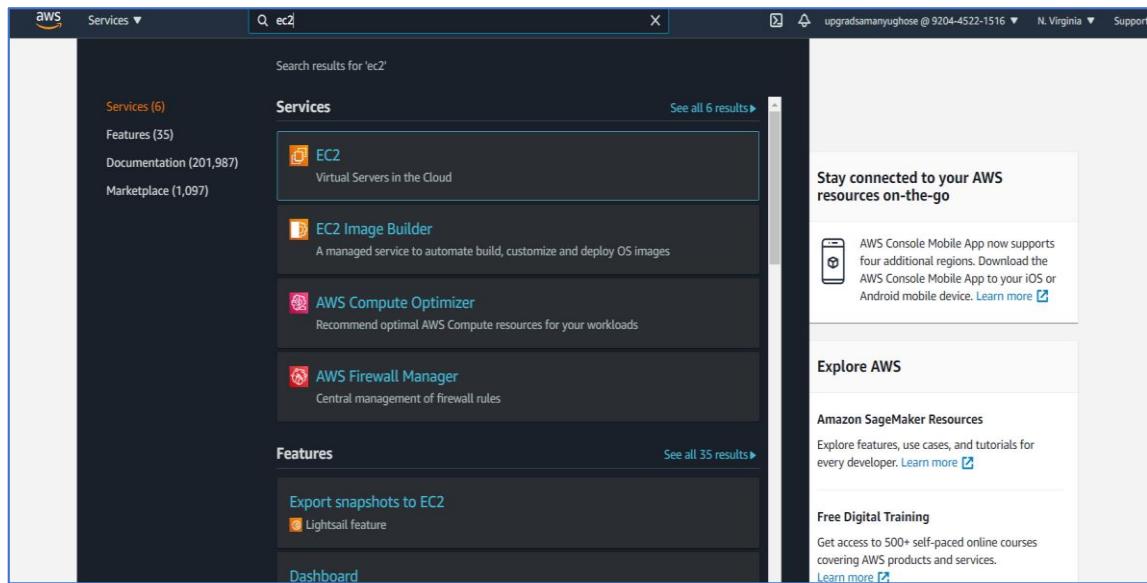
Launching EMR Cluster

Created a 2 node cluster which comprises of following configurations:

1. EMR version – 5.29.0
2. Services selected – Hadoop 2.8.5, Hue 4.4.0 and Hive 2.3.6
3. Selected 1 instance of each, master node & core node of instance type m4.large
4. Selected an appropriate key – pair to SSH into the master node of Amazon EMR cluster

Steps to create EC2, key-pair, Create cluster, load PuttyGen and open Putty command prompt :-

1. First open AWS console and click on EC2



2. Click on key-pair to create a key-pair

	Name	Fingerprint	ID
<input type="checkbox"/>	samanyu_920445221516_2021-01-28...	4af8:b1:56:81:d0:0:c:28:f5:52:09:39:3...	key-0b1a11508bd7b8a56
<input type="checkbox"/>	test	33:4b:86:87:27:0d:bf:22:55:78:e5:37:c...	key-0dd1a5bce92b1c8f9
<input type="checkbox"/>	test_emr	fc:4e:3c:23:ca:2d:43:e5:0:c:00:f7:bc:0:a...	key-0f57799b1ca797e5a

Created a key-pair by the name `hive_emr` and the file format is `.pem` file and the file will be downloaded

Create key pair

Key pair
A key pair, consisting of a private key and a public key, is a set of security credentials that you use to prove your identity when connecting to an instance.

Name
The name can include up to 255 ASCII characters. It can't include leading or trailing spaces.

File format
 pem
For use with OpenSSH
 ppk
For use with PuTTY

Tags (Optional)
No tags associated with the resource.
[Add tag](#)
You can add 50 more tags.

[Cancel](#) **Create key pair**

Click on 'Create key-pair'

Now we can see that key-pair has been created

	Name	Fingerprint	ID
<input type="checkbox"/>	hive_emr	98:2b:cb:49:8a:10:cc:b2:cf:7f:67:24:81...	key-08cc6b6c7df2a7447
<input type="checkbox"/>	hive_emr1	2c:e3:c3:53:50:07:09:03:12:ab:bf:38:1...	key-0459c09aef68a5334
<input checked="" type="checkbox"/>	hive_emr2	77:07:31:32:c8:3:cd2:59:b1:6:c79:7dc...	key-0f3a213ac6f28e97d
<input type="checkbox"/>	samanyu_920445221516_2021-01-28...	4af8:b1:56:81:d0:0:c:28:f5:52:09:39:3...	key-0b1a11508bd7b8a56
<input type="checkbox"/>	test	33:4b:86:87:27:0:bf:22:55:78:e5:37:c...	key-0dd1a5bce92b1c8f9
<input type="checkbox"/>	test_emr	fc:4e:3c:23:ca:2d:43:e5:0:c:00:f7:bc:0:a...	key-0f57799b1ca797e5a

1. Now we will goto AWS console and type EMR and create EMR cluster

The screenshot shows the AWS search interface with the query 'emr'. The search results are displayed under the 'Services' section, with 'EMR' being the top result. Below it are 'AWS Glue DataBrew' and 'Documentation' sections. The 'Documentation' section lists three code samples: 'emr_basics.py - AWS Code Sample', 'emr_usage_demo.py - AWS Code Sample', and 'test_emr_basics.py - AWS Code Sample'. On the right side of the screen, there is a separate window or tab showing a list of items, likely related to key pairs.

Then we will create cluster by clicking on 'Create cluster'

The screenshot shows the 'Amazon EMR' service page. The left sidebar includes sections for 'Clusters', 'Notebooks', 'Git repositories', 'Security configurations', 'Block public access', 'VPC subnets', and 'Events'. Under 'EMR on EC2', there is a 'Clusters' section with two entries: 'emr_hive' and 'democlus_sam'. Both clusters are listed as 'Terminated' with their respective creation times, elapsed times, and normalized instance hours.

After clicking on 'Create Cluster', goto advanced options

Create Cluster - Quick Options [Go to advanced options](#)

General Configuration

Cluster name

Logging [i](#)
S3 folder [i](#)

Launch mode Cluster [i](#) Step execution [i](#)

Software configuration

Release [i](#)

Applications

Hardware configuration

Instance type [i](#) The selected instance type adds 32 GiB of GP2 EBS storage per instance by default. [Learn more](#) [i](#)

Number of instances (1 master and 2 core nodes)

Security and access

EC2 key pair [i](#) [Learn how to create an EC2 key pair.](#)

Choose emr-5.29.0 and click on Next

demo_sam

Multiple master nodes (optional)
 Use multiple master nodes to improve cluster availability. [Learn more](#) [i](#)

AWS Glue Data Catalog settings (optional)
 Use for Hive table metadata [i](#)
 Use for Spark table metadata [i](#)

Edit software settings [i](#)
 Enter configuration Load JSON from S3
`classification=config-file-name,properties=[myKey1=myValue1,myKey2=myValue2]`

Steps (optional)
A step is a unit of work you submit to the cluster. For instance, a step might contain one or more Hadoop or Spark jobs. You can also submit additional steps to a cluster after it is running. [Learn more](#) [i](#)

Concurrency: Run multiple steps at the same time to improve cluster utilization

After last step completes: Clusters enters waiting state Cluster auto-terminates

Step type [Add step](#)

[Cancel](#) [Next](#)

Choose Master and Core instance as 1 and instance type as m4.large in the next screen

demo Sam

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	1 Instances	<input checked="" type="radio"/> On-demand <input type="radio"/> Spot Use on-demand as max price
Core Core - 2	m4.large	1 Instances	<input checked="" type="radio"/> On-demand <input type="radio"/> Spot Use on-demand as max price
Task Task - 3	m5.xlarge	0 Instances	<input checked="" type="radio"/> On-demand <input type="radio"/> Spot Use on-demand as max price
+ Add task instance group			

Feedback English (US) © 2008 - 2021, Amazon Internet Services Private Ltd. or its affiliates. All rights reserved. Privacy Policy Terms of Use Cookie preferences

demo Sam

Task
Task - 3

m5.xlarge
4 vCore, 16 GiB memory, EBS only storage
EBS Storage: 64 GiB
Add configuration settings

0 Instances On-demand
 Spot
Use on-demand as max price

[+ Add task instance group](#)

Total core and task units 1 Total units

Cluster scaling
Adjust the number of Amazon EC2 instances available to an EMR cluster via EMR-managed scaling or a custom automatic scaling policy. [Learn more](#)

Cluster scaling Enable Cluster Scaling

EBS Root Volume
Specify the root device volume size up to 100 GiB. This sizing applies to all instances in the cluster. [Learn more](#)

Root device EBS volume size GiB

[Cancel](#) [Previous](#) [Next](#)

Feedback English (US) © 2008 - 2021, Amazon Internet Services Private Ltd. or its affiliates. All rights reserved. Privacy Policy Terms of Use Cookie preferences

Type cluster name as 'emr_hivesam'

Cluster name

Logging i

S3 folder

Debugging i

Termination protection i

Tags i

Key	Value (optional)
<input type="text" value="Add a key to create a tag"/>	<input type="text"/>

Additional Options

EMRFS consistent view i

Custom AMI ID

▶ Bootstrap Actions

Cancel **Previous** **Next**

Select EC2 key pair as 'hive_emr1' which we have just created

EC2 key pair

Cluster visible to all IAM users in account i

Permissions i

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 i

EC2 instance profile i

Auto Scaling role i

▶ Security Configuration
▶ EC2 security groups

Create cluster

First we will set security for newly created EMR ('emr_hivesam') and setting inbound rules for SSH as 'anywhere' so that we can use at any number of times when we will accessing the EMR cluster for the 2nd time.

As soon as the cluster status shows 'waiting', we will start with the inbound security setting of security rules

Cluster: emr_hivesam Waiting Cluster ready after last step completed.

Summary

ID: j-DUWRSRWZEOYYB
Creation date: 2021-02-23 21:49 (UTC+5:30)
Elapsed time: 41 minutes
After last step completes: Cluster waits
Termination protection: On Change
Tags: -- View All / Edit
Master public DNS: ec2-18-212-10-54.compute-1.amazonaws.com Connect to the Master Node Using SSH

Configuration details

Release label: emr-5.29.0
Hadoop distribution: Amazon 2.8.5
Applications: Hive 2.3.6, Pig 0.17.0, Hue 4.4.0, Spark 2.4.4
Log URI: s3://aws-logs-920445221516-us-east-1/elastictmapreduce/

EMRFS consistent view: Disabled
Custom AMI ID: --

Application user interfaces

Persistent user interfaces [2]: Spark history server
On-cluster user Not Enabled Enable an SSH Connection
interfaces [2]

Network and hardware

Availability zone: us-east-1a
Subnet ID: [subnet-46ab3367](#)
Master: [Running](#) 1 m4.large
Core: [Running](#) 1 m4.large
Task: --
Cluster scaling: Not enabled

Security and access

Key name: hive_emr1

Click on security group for Master

Security and access

Key name: hive_emr1
EC2 instance profile: EMR_EC2_DefaultRole
EMR role: EMR_DefaultRole
Auto Scaling role: EMR_AutoScaling_DefaultRole
Visible to all users: All Change
Security groups for Master: [sg-0cf13256f84f7d8ed](#) (ElasticMapReduce-master)
Security groups for Core & [sg-0a2c22e925ed4ab6a](#) (ElasticMapReduce-Task: slave)

Click on security groups for Master and click on security ID for Master and edit inbound rules

The screenshot shows the AWS EC2 Security Groups page. The left sidebar includes options like EC2 Dashboard, Instances, and Images. The main area displays two security groups: 'ElasticMapReduce-slave' and 'ElasticMapReduce-master'. The 'ElasticMapReduce-master' group is selected, indicated by a checked checkbox. The table columns include Name, Security group ID, Security group name, VPC ID, Description, and Owner. The 'Description' column for the master group notes it as the 'Master group for Elastic MapReduce'.

This screenshot shows the detailed view for the 'ElasticMapReduce-master' security group. It includes fields for Security group name (ElasticMapReduce-master), Security group ID (sg-0cf13256f84f7d8e4), Description (Master group for Elastic MapReduce created on 2021-02-05T02:00:28.525Z), Owner (920445221516), Inbound rules count (20 Permission entries), and Outbound rules count (1 Permission entry). The 'Inbound rules' tab is selected, showing two existing rules: one for All TCP traffic from the slave security group and another for All TCP traffic from the master security group itself. The right sidebar contains links for Actions, Details, Inbound rules, Outbound rules, and Tags.

Click on Add rule, select SSH from drop down list and rule as ‘anywhere’ so that every time we clone a cluster, we won’t have to select ‘My IP’ and for testing purpose, it is a common practice to choose the IP as ‘Anywhere’

This screenshot shows the 'Inbound rules' configuration page for the 'ElasticMapReduce-master' security group. It features a table with columns for Type, Protocol, Port range, Source, and Description - optional. Two rules are listed: one for All TCP traffic from the slave security group and another for All TCP traffic from the master security group itself. Below the table is an 'Edit inbound rules' button. At the bottom, there is a note about edits and a warning about traffic drops during rule creation. Buttons for Cancel, Preview changes, and Save rules are at the very bottom.

Now inbound security rules has been successfully modified

The screenshot shows the AWS EC2 Security Groups page. A green banner at the top indicates that the 'Inbound security group rules successfully modified on security group (sg-0cf13256f84f7d8e4 | ElasticMapReduce-master)'.

The main section displays the details for the security group 'sg-0cf13256f84f7d8e4 - ElasticMapReduce-master'. It shows the following information:

- Security group name:** ElasticMapReduce-master
- Security group ID:** sg-0cf13256f84f7d8e4
- Description:** Master group for Elastic MapReduce created on 2021-02-05T02:00:28.52Z
- VPC ID:** vpc-8e5df3f3
- Owner:** 920445221516
- Inbound rules count:** 20 Permission entries
- Outbound rules count:** 1 Permission entry

The 'Inbound rules' tab is selected. A button labeled 'Edit inbound rules' is visible.

Now goto EMR , Select EMR cluster 'emr_hivesam' and select 'connect to master node using SSH' under Master public DNS and we will copy the Host name field and paste it in PuttY gen

The screenshot shows the AWS EMR Cluster Summary page for the cluster 'emr_hivesam' (Status: Waiting). A modal window titled 'SSH' provides instructions for connecting to the master node using SSH:

Connect to the Master Node Using SSH

You can connect to the Amazon EMR master node using SSH to run interactive queries, examine log files, submit Linux commands, and so on.

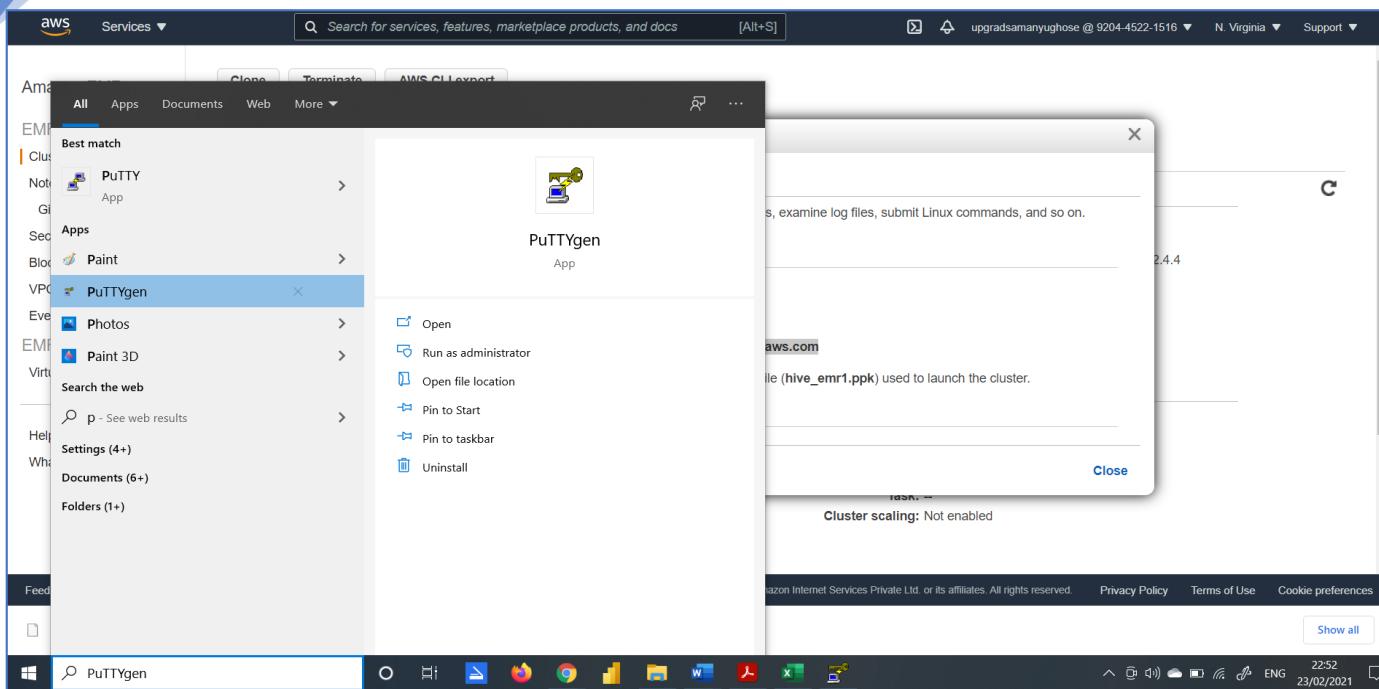
[Learn more](#)

Windows **Mac / Linux**

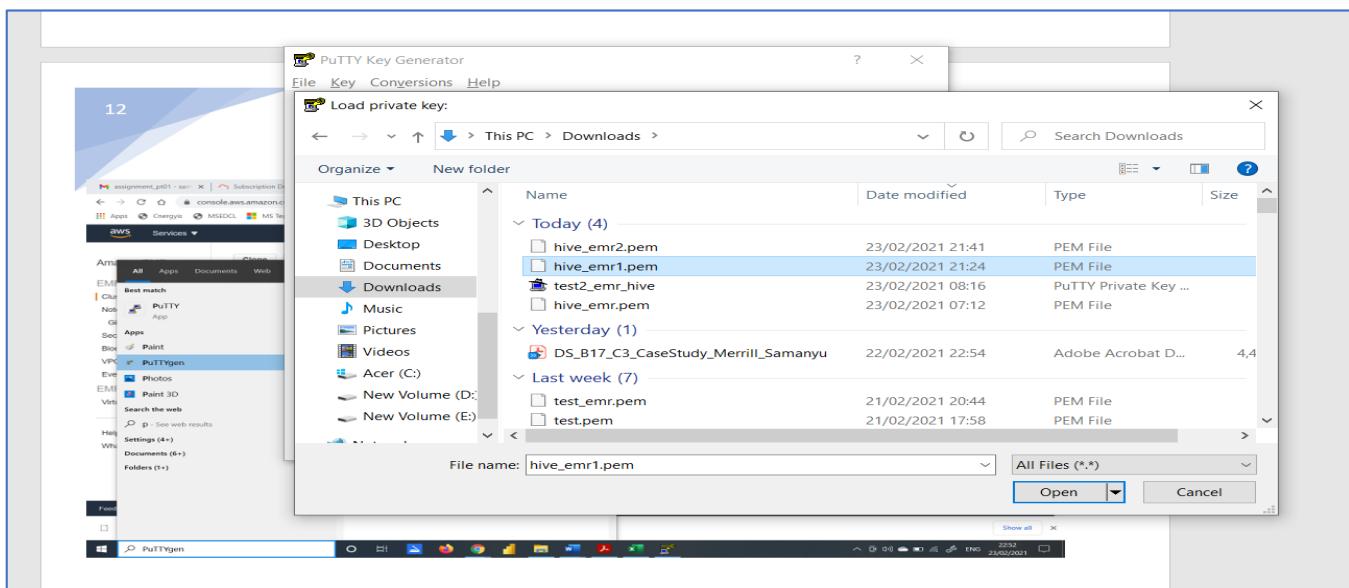
- Download PuTTY.exe to your computer from: <http://www.chiark.greenend.org.uk/~sgtatham/putty/download.html>
- Start PuTTY.
- In the Category list, click Session.
- In the Host Name field, type **hadoop@ec2-18-212-10-54.compute-1.amazonaws.com**.
- In the Category list, expand Connection > SSH, and then click Auth.
- For Private key file for authentication, click Browse and select the private key file (**hive_emr1.ppk**) used to launch the cluster.
- Click Open.
- Click Yes to dismiss the security alert.

Security and access

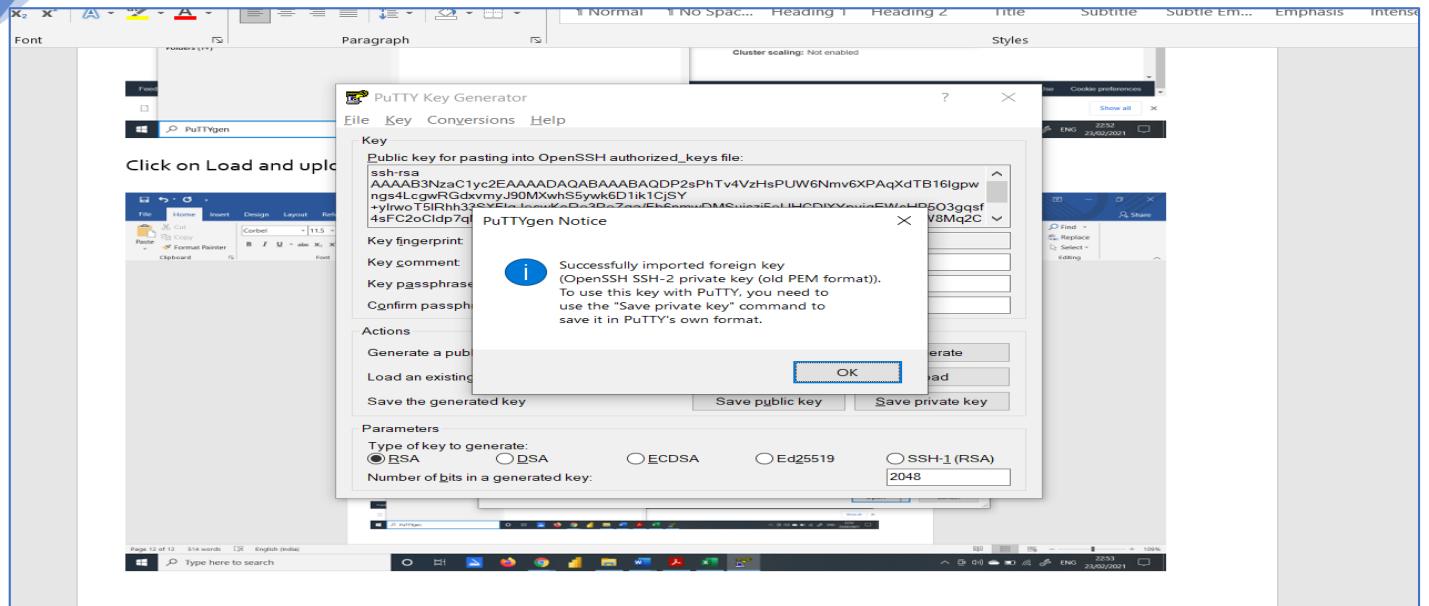
Key name: **hive_emr1**



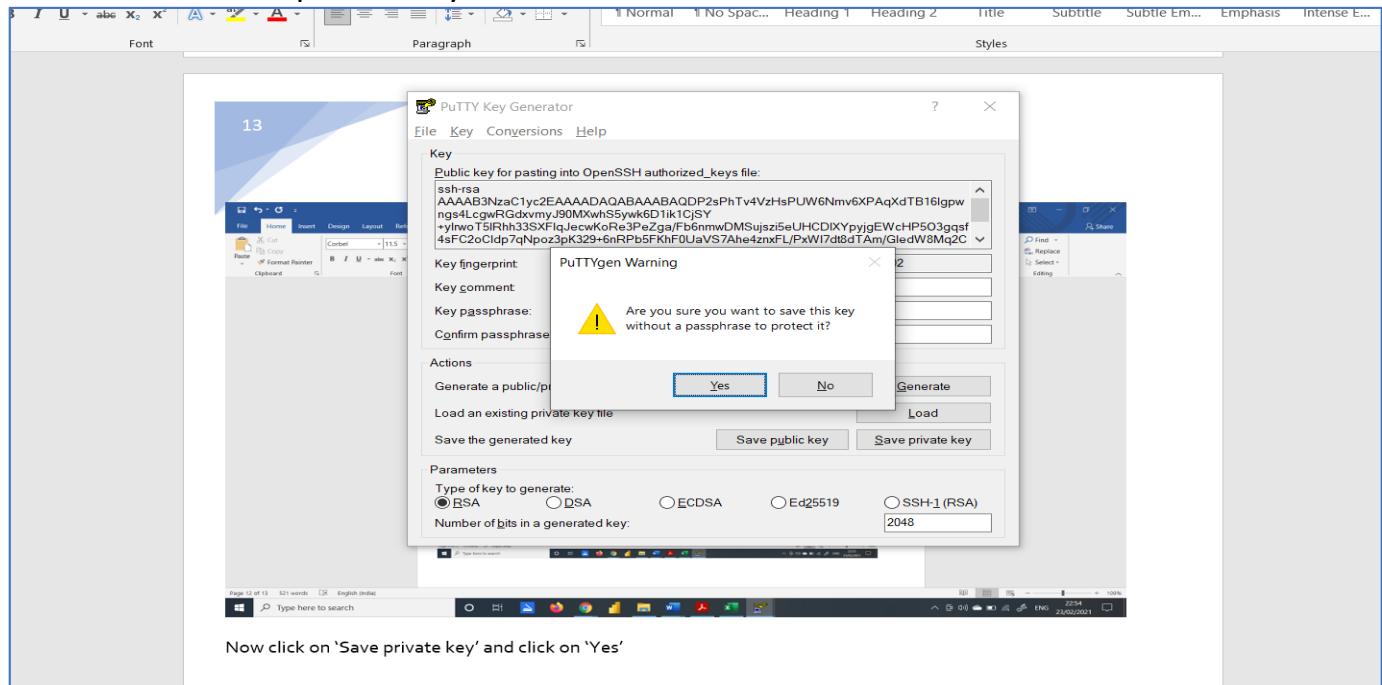
Click on Load and upload 'hive_emr1.pem' file



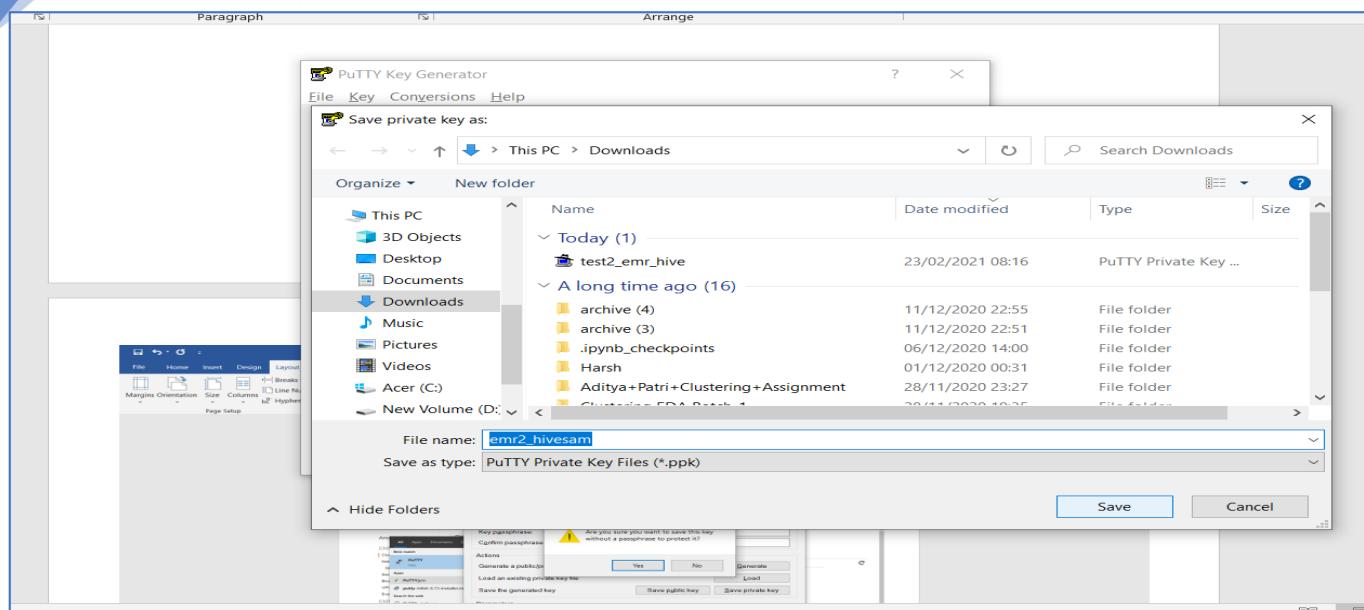
We will get successful message



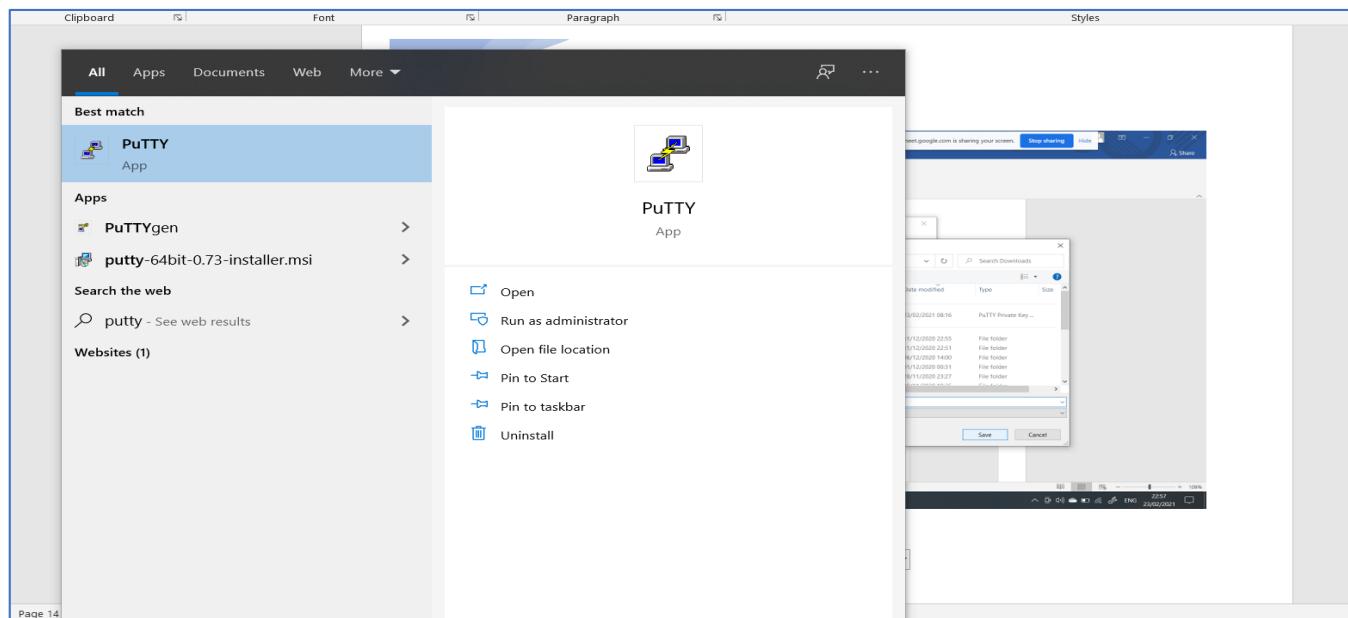
Now click on 'Save private key' and click on 'Yes'

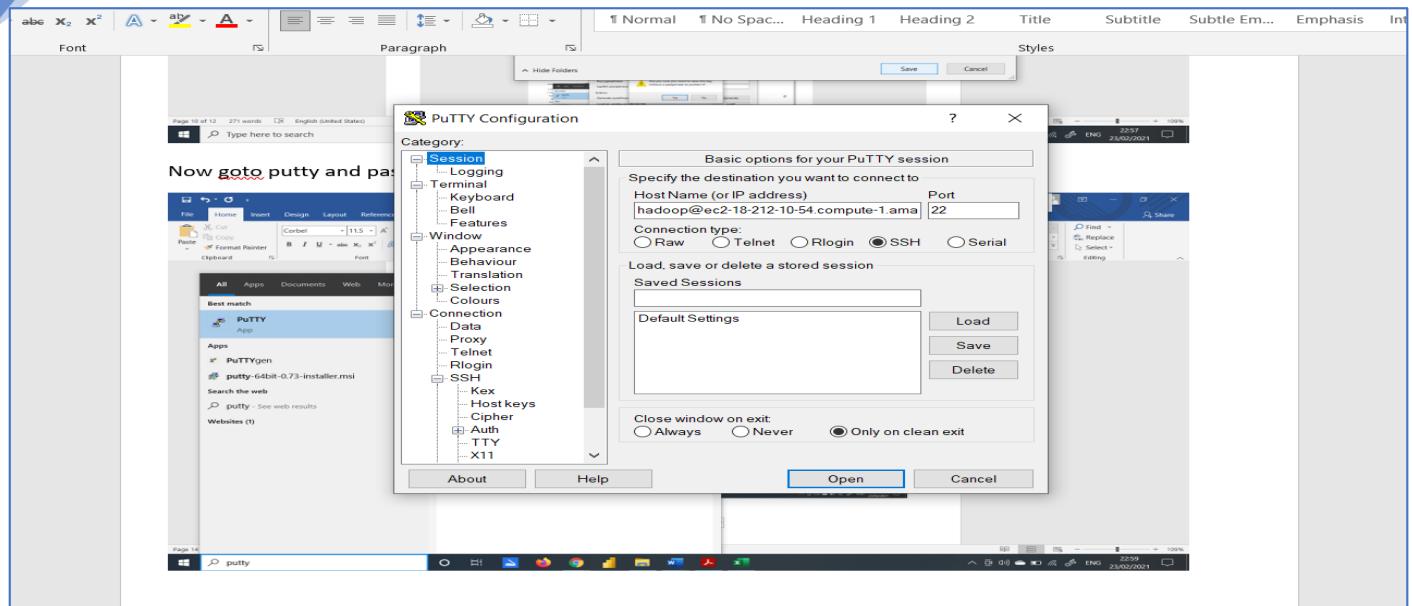


Type the name of .ppk file as 'emr2_hivesam'



Now goto putty and paste the Master DNS SSH host name





And click on 'open' to open Putty terminal and click on 'Yes'

```

[hadoop@ip-172-31-85-4 ~]$

```

Snapshot: Master node, when connected to EMR cluster

1. Create Directory

First created another folder name `hivestudy` and placed all related data needed for this assignment in that folder.

Command:

`hadoop fs -mkdir /hivestudy`

```

hadoop@ip-172-31-85-4:~$ include in the classpath.
archives <comma separated list of archives>      specify comma separated archives
to be unarchived on the compute machines.

The general command line syntax is
command [genericOptions] [commandOptions]

[hadoop@ip-172-31-85-4 ~]$ fs -ls /
-bash: fs: command not found
[hadoop@ip-172-31-85-4 ~]$ hadoop fs -ls /
Found 4 items
drwxr-xr-x  - hdfs hadoop          0 2021-02-23 16:26 /apps
drwxrwxrwt  - hdfs hadoop          0 2021-02-23 16:28 /tmp
drwxr-xr-x  - hdfs hadoop          0 2021-02-23 16:26 /user
drwxr-xr-x  - hdfs hadoop          0 2021-02-23 16:26 /var
[hadoop@ip-172-31-85-4 ~]$ hadoop fs -mkdir /hivestudy
[hadoop@ip-172-31-85-4 ~]$ hadoop fs -ls /
Found 5 items
drwxr-xr-x  - hdfs hadoop          0 2021-02-23 16:26 /apps
drwxr-xr-x  - hadoop hadoop         0 2021-02-23 17:44 /hivestudy
drwxrwxrwt  - hdfs hadoop          0 2021-02-23 16:28 /tmp
drwxr-xr-x  - hdfs hadoop          0 2021-02-23 16:26 /user
drwxr-xr-x  - hdfs hadoop          0 2021-02-23 16:26 /var
[hadoop@ip-172-31-85-4 ~]$ 

```

Snapshot: Directory creation

2. Loading data into HDFS

Case study data is present in s3 in two files.

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

Both of the files loaded into HDFS under hivestudy directory with file names 2019-Oct.csv & 2019-Nov.csv

Commands:

hadoop distcp s3n://e-commerce-events-ml/2019-Oct.csv /hivestudy/2019-Oct.csv
 hadoop distcp s3n://e-commerce-events-ml/2019-Nov.csv /hivestudy/2019-Nov.csv

```

hadoop@ip-172-31-85-4:~$ hadoop distcp s3n://e-commerce-events-ml/2019-Oct.csv /hivestudy/2019-Oct.csv
21/02/23 17:52:52 INFO tools.DistCp: Input Options: DistCpOptions(atomicCommit=false, syncFolder=false, deleteMissing=false, ignoreFailures=false, overwrite=false, skipCRC=false, blocking=true, numListStatusThreads=0, maxMaps=20, mapBandwidth=100, ssConfigurationFile=null, copyStrategy='uniformsize', preserveStatus=[], preserveRawAttrs=false, atomicWorkPath=null, logPath=null, sourceFileListin
g=null, sourcePaths=[s3n://e-commerce-events-ml/2019-Oct.csv], targetPath=/hives
study/2019-Oct.csv, targetPathExists=false, filtersFile='null')
21/02/23 17:52:52 INFO client.RMProxy: Connecting to ResourceManager at ip-172-3
1-85-4.ec2.internal/172.31.85.4:8032
21/02/23 17:52:56 INFO tools.SimpleCopyListing: Paths (files+dirs) cnt = 1; dirC
nt = 0
21/02/23 17:52:56 INFO tools.SimpleCopyListing: Build file listing completed.
21/02/23 17:52:56 INFO Configuration.deprecation: io.sort.mb is deprecated. Inst
ead, use mapreduce.task.io.sort.mb
21/02/23 17:52:56 INFO Configuration.deprecation: io.sort.factor is deprecated.
Instead, use mapreduce.task.io.sort.factor
21/02/23 17:52:56 INFO tools.DistCp: Number of paths in the copy list: 1
21/02/23 17:52:56 INFO tools.DistCp: Number of paths in the copy list: 1
21/02/23 17:52:56 INFO client.RMProxy: Connecting to ResourceManager at ip-172-3
1-85-4.ec2.internal/172.31.85.4:8032
21/02/23 17:52:57 INFO mapreduce.JobSubmitter: number of splits:1
21/02/23 17:52:57 INFO mapreduce.JobSubmitter: Submitting tokens for job: job_16
14097634219_0001
21/02/23 17:52:57 INFO impl.YarnClientImpl: Submitted application application_16
14097634219_0001
21/02/23 17:52:57 INFO mapreduce.Job: The url to track the job: http://ip-172-31
-85-4.ec2.internal:2088/proxy/application_1614097634219_0001/
21/02/23 17:52:57 INFO tools.DistCp: DistCp job-id: job_1614097634219_0001
21/02/23 17:52:57 INFO mapreduce.Job: Running job: job_1614097634219_0001
21/02/23 17:53:08 INFO mapreduce.Job: Job job_1614097634219_0001 running in uber
mode : false
21/02/23 17:53:08 INFO mapreduce.Job: map 0% reduce 0%
21/02/23 17:53:25 INFO mapreduce.Job: map 100% reduce 0%
21/02/23 17:53:26 INFO mapreduce.Job: Job job_1614097634219_0001 completed succe
ssfully
21/02/23 17:53:26 INFO mapreduce.Job: Counters: 38
  File System Counters
    FILE: Number of bytes read=0
    FILE: Number of bytes written=172802
    FILE: Number of read operations=0
    FILE: Number of large read operations=0
    FILE: Number of write operations=0
    HDFS: Number of bytes read=360
    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
    S3N: Number of bytes read=482542278

```

Snapshot: Loading Oct file into HDFS

```

hadoop@ip-172-31-85-4:~$ hadoop distcp s3n://e-commerce-events-ml/2019-Nov.csv /hivestudy/2019-Nov.csv
21/02/23 17:58:47 INFO tools.DistCp: Input Options: DistCpOptions(atomicCommit=false, syncFolder=false, deleteMissing=false, ignoreFailures=false, overwrite=false, skipCRC=false, blocking=true, numListStatusThreads=0, maxMaps=20, mapBandwidth=100, ssConfigurationFile=null, copyStrategy='uniformsize', preserveStatus=[], preserveRawAttrs=false, atomicWorkPath=null, logPath=null, sourceFileListin
g=null, sourcePaths=[s3n://e-commerce-events-ml/2019-Nov.csv], targetPath=/hives
study/2019-Nov.csv, targetPathExists=false, filtersFile='null')
21/02/23 17:58:47 INFO client.RMProxy: Connecting to ResourceManager at ip-172-31-85-4.ec2.internal/172.31.85.4:8032
21/02/23 17:58:51 INFO tools.SimpleCopyListing: Paths (files+dirs) cnt = 1; dirC
nt = 0
21/02/23 17:58:51 INFO tools.SimpleCopyListing: Build file listing completed.
21/02/23 17:58:51 INFO Configuration.deprecation: io.sort.mb is deprecated. Instead, use mapreduce.task.io.sort.mb
21/02/23 17:58:51 INFO Configuration.deprecation: io.sort.factor is deprecated. Instead, use mapreduce.task.io.sort.factor
21/02/23 17:58:51 INFO tools.DistCp: Number of paths in the copy list: 1
21/02/23 17:58:51 INFO tools.DistCp: Number of paths in the copy list: 1
21/02/23 17:58:52 INFO client.RMProxy: Connecting to ResourceManager at ip-172-31-85-4.ec2.internal/172.31.85.4:8032
21/02/23 17:58:52 INFO mapreduce.JobSubmitter: number of splits:1
21/02/23 17:58:52 INFO mapreduce.JobSubmitter: Submitting tokens for job: job_1614097634219_0002
21/02/23 17:58:52 INFO impl.YarnClientImpl: Submitted application application_1614097634219_0002
21/02/23 17:58:53 INFO mapreduce.Job: The url to track the job: http://ip-172-31-85-4.ec2.internal:2088/proxy/application_1614097634219_0002/
21/02/23 17:58:53 INFO tools.DistCp: DistCp job-id: job_1614097634219_0002
21/02/23 17:58:53 INFO mapreduce.Job: Running job: job_1614097634219_0002
21/02/23 17:59:01 INFO mapreduce.Job: Job job_1614097634219_0002 running in uber mode : false
21/02/23 17:59:01 INFO mapreduce.Job: map 0% reduce 0%
21/02/23 17:59:19 INFO mapreduce.Job: map 100% reduce 0%
21/02/23 17:59:20 INFO mapreduce.Job: Job job_1614097634219_0002 completed successfully
21/02/23 17:59:20 INFO mapreduce.Job: Counters: 38
  File System Counters
    FILE: Number of bytes read=0
    FILE: Number of bytes written=172802
    FILE: Number of read operations=0
    FILE: Number of large read operations=0
    FILE: Number of write operations=0
    HDFS: Number of bytes read=360
    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
    S3N: Number of bytes read=545839412
    S3N: Number of bytes written=0
    S3N: Number of read operations=0
    S3N: Number of large read operations=0
    S3N: 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)=545440
    Total time spent by all reduces in occupied slots (ms)=0
    Total time spent by all map tasks (ms)=17045
    Total vcore-milliseconds taken by all map tasks=17045
    Total megabyte-milliseconds taken by all map tasks=17454080
  Map-Reduce Framework
    Map input records=1

```

Snapshot: Loading Nov file into HDFS

3. Quick view

Taking a quick view of few rows to check whether data has correctly loaded into the file.

Commands:

```
hadoop fs -cat /hivestudy/2019-Nov.csv | head
```

hadoop fs -cat /hivestudy/2019-Oct.csv | head

```
[hadoop@ip-172-31-85-4:~]$ hadoop fs -cat /caseStudy/2019-Nov.csv | head
cat: '/caseStudy/2019-Nov.csv': No such file or directory
[hadoop@ip-172-31-85-4:~]$ hadoop fs -cat /hivestudy/2019-Nov.csv | head
event_type,product_id,category_id,category_code,brand,price,user_id,user_session
2019-11-01 00:00:02 UTC,view,5802432,1487580009286598691,,0,32,562076640,09fafdfc-6c99-4eb1-834f-33527f4de241
2019-11-01 00:00:03 UTC,cart,5844397,1487580006317032337,,2,38,553329724,2067216c-31b5-455d-alcc-af0575a34ffb
2019-11-01 00:00:10 UTC,view,5837166,1783999064103190764,,pnb,22,22,556138645,57ed222e-a54a-4907-9944-5a875c2d7f4f
2019-11-01 00:00:11 UTC,cart,5876112,14875800010100293687,,jessmail,3,16,564506666,186c1951-0052-4b37-adce-dd9644bd5f7
2019-11-01 00:00:24 UTC,remove_from_cart,5826182,1487580007483048900,,3,33,553329724,2067216c-31b5-455d-alcc-af0575a34ffb
2019-11-01 00:00:24 UTC,remove_from_cart,5826182,1487580007483048900,,3,33,553329724,2067216c-31b5-455d-alcc-af0575a34ffb
2019-11-01 00:00:25 UTC,view,5856189,148758000902651821,,runail,15,71,562076640,09fafdfc-6c99-4eb1-834f-33527f4de241
2019-11-01 00:00:32 UTC,view,5837835,1933472286753424063,,3,49,514649199,432a4e95-375c-4b40-bd36-0fc039e77580
2019-11-01 00:00:34 UTC,remove_from_cart,5870838,1487580007675986893,,milv,0,79,429913900,2f0bff3c-252f-4fe6-afcd-5d8a6a92839a
cat: Unable to write to output stream.
[hadoop@ip-172-31-85-4:~]$ hadoop fs -cat /hivestudy/sdataset.csv | head
cat: '/hivestudy/sdataset.csv': No such file or directory
[hadoop@ip-172-31-85-4:~]$ hadoop fs -cat /hivestudy/2019-Nov.csv | head
event_type,product_id,category_id,category_code,brand,price,user_id,user_session
2019-11-01 00:00:02 UTC,view,5802432,1487580005134238553,,runail,2,62,463240011,26dd66ee-4dac-4778-8d2c-92e149dab885
2019-11-01 00:00:03 UTC,cart,5773353,1487580005134238553,,runail,2,62,463240011,26dd66ee-4dac-4778-8d2c-92e149dab885
2019-11-01 00:00:07 UTC,cart,5723490,1487580005134238553,,runail,2,62,463240011,26dd66ee-4dac-4778-8d2c-92e149dab885
2019-10-01 00:00:07 UTC,view,5881589,14875800013522845895,,lovelly,0,56,429681830,49e8d843-adf3-42b8-a2c3-fe8bc6a307c9
2019-10-01 00:00:11 UTC,cart,5857269,1487580005134238553,,runail,2,62,430174032,73deale7-664e-43f4-b30-3d2b9d5af04f
2019-10-01 00:00:24 UTC,remove_from_cart,5826182,1487580007483048900,,3,33,553329724,2067216c-31b5-455d-alcc-af0575a34ffb
2019-10-01 00:00:25 UTC,view,5856189,148758000902651821,,runail,15,71,562076640,09fafdfc-6c99-4eb1-834f-33527f4de241
2019-10-01 00:00:32 UTC,view,5837835,1933472286753424063,,3,49,514649199,432a4e95-375c-4b40-bd36-0fc039e77580
2019-10-01 00:00:34 UTC,remove_from_cart,5870838,1487580007675986893,,milv,0,79,429913900,2f0bff3c-252f-4fe6-afcd-5d8a6a92839a
cat: Unable to write to output stream.
[hadoop@ip-172-31-85-4:~]$ hadoop fs -cat /hivestudy/2019-Oct.csv | head
event_type,product_id,category_id,category_code,brand,price,user_id,user_session
2019-10-01 00:00:02 UTC,view,5802432,1487580005134238553,,runail,2,62,463240011,26dd66ee-4dac-4778-8d2c-92e149dab885
2019-10-01 00:00:03 UTC,cart,5773353,1487580005134238553,,runail,2,62,463240011,26dd66ee-4dac-4778-8d2c-92e149dab885
2019-10-01 00:00:07 UTC,cart,5723490,1487580005134238553,,runail,2,62,463240011,26dd66ee-4dac-4778-8d2c-92e149dab885
2019-10-01 00:00:07 UTC,view,5881589,14875800013522845895,,lovelly,0,56,429681830,49e8d843-adf3-42b8-a2c3-fe8bc6a307c9
2019-10-01 00:00:11 UTC,cart,5857269,1487580005134238553,,runail,2,62,430174032,73deale7-664e-43f4-b30-3d2b9d5af04f
2019-10-01 00:00:24 UTC,remove_from_cart,5826182,1487580008246412266,,kapous,4,75,3776767011,81326ac6-daa4-4f0a-b488-fd0956a78733
2019-10-01 00:00:25 UTC,view,5825598,1487580009445982239,,0,56,467916806,2f5b5546-b8cb-9ee7-7ecd-84276f80ef486
2019-10-01 00:00:25 UTC,cart,5698998,1487580006317032337,,1,27,385985999,d30965e8-1101-44ab-b45d-cc1bb9fae694
cat: Unable to write to output stream.
[hadoop@ip-172-31-85-4:~]$
```

Screenshot: Quick view of loaded data

Hive DDL Querying

1. Enter Hive CLI

Now, that data has been loaded into HDFS, entering into HIVE command line input to perform analysis.

Command: hive

```
[hadoop@ip-172-31-85-4:~]$ hadoop fs -cat /caseStudy/2019-Nov.csv | head
cat: '/caseStudy/2019-Nov.csv': No such file or directory
[hadoop@ip-172-31-85-4:~]$ hadoop fs -cat /hivestudy/2019-Nov.csv | head
event_type,product_id,category_id,category_code,brand,price,user_id,user_session
2019-11-01 00:00:02 UTC,view,5802432,1487580009286598691,,0,32,562076640,09fafdfc-6c99-4eb1-834f-33527f4de241
2019-11-01 00:00:03 UTC,cart,5844397,1487580006317032337,,2,38,553329724,2067216c-31b5-455d-alcc-af0575a34ffb
2019-11-01 00:00:10 UTC,view,5837166,1783999064103190764,,pnb,22,22,556138645,57ed222e-a54a-4907-9944-5a875c2d7f4f
2019-11-01 00:00:11 UTC,cart,5876112,14875800010100293687,,jessmail,3,16,564506666,186c1951-0052-4b37-adce-dd9644bd5f7
2019-11-01 00:00:24 UTC,remove_from_cart,5826182,1487580007483048900,,3,33,553329724,2067216c-31b5-455d-alcc-af0575a34ffb
2019-11-01 00:00:24 UTC,remove_from_cart,5826182,1487580007483048900,,3,33,553329724,2067216c-31b5-455d-alcc-af0575a34ffb
2019-11-01 00:00:25 UTC,view,5856189,148758000902651821,,runail,15,71,562076640,09fafdfc-6c99-4eb1-834f-33527f4de241
2019-11-01 00:00:32 UTC,view,5837835,1933472286753424063,,3,49,514649199,432a4e95-375c-4b40-bd36-0fc039e77580
2019-11-01 00:00:34 UTC,remove_from_cart,5870838,1487580007675986893,,milv,0,79,429913900,2f0bff3c-252f-4fe6-afcd-5d8a6a92839a
cat: Unable to write to output stream.
[hadoop@ip-172-31-85-4:~]$ hadoop fs -cat /hivestudy/sdataset.csv | head
cat: '/hivestudy/sdataset.csv': No such file or directory
[hadoop@ip-172-31-85-4:~]$ hadoop fs -cat /hivestudy/2019-Nov.csv | head
event_type,product_id,category_id,category_code,brand,price,user_id,user_session
2019-11-01 00:00:02 UTC,view,5802432,1487580005134238553,,runail,2,62,463240011,26dd66ee-4dac-4778-8d2c-92e149dab885
2019-11-01 00:00:03 UTC,cart,5773353,1487580005134238553,,runail,2,62,463240011,26dd66ee-4dac-4778-8d2c-92e149dab885
2019-11-01 00:00:07 UTC,cart,5723490,1487580005134238553,,runail,2,62,463240011,26dd66ee-4dac-4778-8d2c-92e149dab885
2019-10-01 00:00:07 UTC,view,5881589,14875800013522845895,,lovelly,0,56,429681830,49e8d843-adf3-42b8-a2c3-fe8bc6a307c9
2019-10-01 00:00:11 UTC,cart,5857269,1487580005134238553,,runail,2,62,430174032,73deale7-664e-43f4-b30-3d2b9d5af04f
2019-10-01 00:00:24 UTC,remove_from_cart,5826182,1487580008246412266,,kapous,4,75,3776767011,81326ac6-daa4-4f0a-b488-fd0956a78733
2019-10-01 00:00:25 UTC,view,5825598,1487580009445982239,,0,56,467916806,2f5b5546-b8cb-9ee7-7ecd-84276f80ef486
2019-10-01 00:00:25 UTC,cart,5698998,1487580006317032337,,1,27,385985999,d30965e8-1101-44ab-b45d-cc1bb9fae694
cat: Unable to write to output stream.
[hadoop@ip-172-31-85-4:~]$ hadoop fs -cat /hivestudy/2019-Oct.csv | head
event_type,product_id,category_id,category_code,brand,price,user_id,user_session
2019-10-01 00:00:02 UTC,view,5802432,1487580005134238553,,runail,2,62,463240011,26dd66ee-4dac-4778-8d2c-92e149dab885
2019-10-01 00:00:03 UTC,cart,5773353,1487580005134238553,,runail,2,62,463240011,26dd66ee-4dac-4778-8d2c-92e149dab885
2019-10-01 00:00:07 UTC,cart,5723490,1487580005134238553,,runail,2,62,463240011,26dd66ee-4dac-4778-8d2c-92e149dab885
2019-10-01 00:00:07 UTC,view,5881589,14875800013522845895,,lovelly,0,56,429681830,49e8d843-adf3-42b8-a2c3-fe8bc6a307c9
2019-10-01 00:00:11 UTC,cart,5857269,1487580005134238553,,runail,2,62,430174032,73deale7-664e-43f4-b30-3d2b9d5af04f
2019-10-01 00:00:24 UTC,remove_from_cart,5826182,1487580008246412266,,kapous,4,75,3776767011,81326ac6-daa4-4f0a-b488-fd0956a78733
2019-10-01 00:00:25 UTC,view,5825598,1487580009445982239,,0,56,467916806,2f5b5546-b8cb-9ee7-7ecd-84276f80ef486
2019-10-01 00:00:25 UTC,cart,5698998,1487580006317032337,,1,27,385985999,d30965e8-1101-44ab-b45d-cc1bb9fae694
cat: Unable to write to output stream.
[hadoop@ip-172-31-85-4:~]$
```

Screenshot: Entering hive CLI

2. Enable headers in CLI

Command:

```
set hive.cli.print.header = true ;
```

```
Logging initialized using configuration in file:/etc/hive/conf.dist/hive-log4j2.properties Async: false
hive> set hive.cli.print.header=true;
hive>
```

Snapshot: Enabling headers in hive

3. Create & use database

Command :-

```
CREATE DATABASE IF NOT EXISTS hivecs WITH DBPROPERTIES('Creator'= 'ecom','Purpose'='Hive Case Study');
```

```
SHOW DATABASES;
```

```
USE hivecs;
```

```
hive> CREATE DATABASE IF NOT EXISTS hivecs WITH DBPROPERTIES('Creator'= 'ecom', 'Purpose'='Hive Case Study');
OK
Time taken: 0.791 seconds
hive> SHOW DATABASES;
OK
database_name
default
hivecs
Time taken: 0.15 seconds, Fetched: 2 row(s)
hive>
```

```
hive> USE hivecs;
OK
Time taken: 0.548 seconds
hive>
```

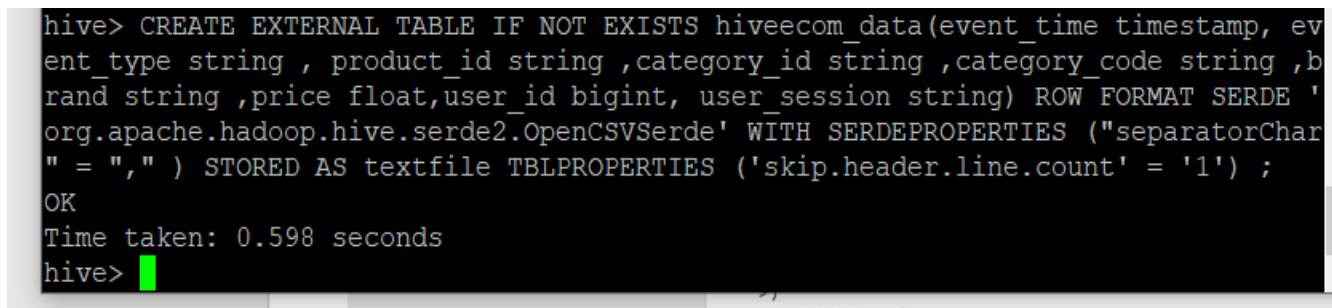
Snapshot: Creating, showing & using databases

4. Create Table Schema

In order to perform analysis on loaded data, a schema needs to be created. Since our files have an extension of .csv, used csvSerde format. By using this format, data types for all the columns get over written to string data type by default

Command :

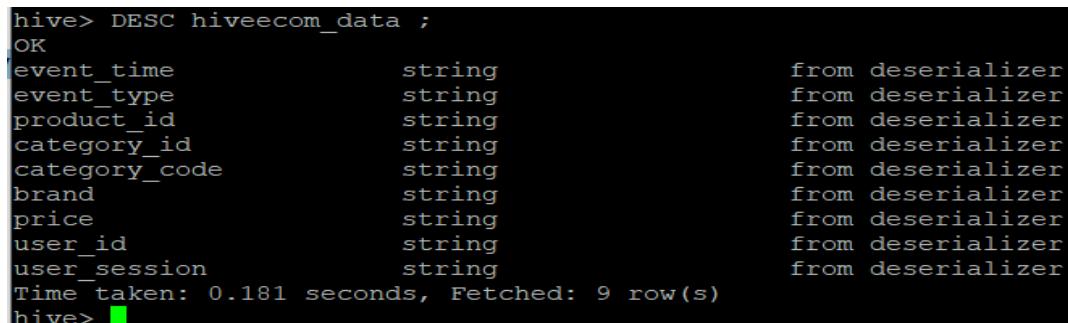
```
CREATE EXTERNAL TABLE IF NOT EXISTS hiveecom_data(event_time timestamp,
event_type string , product_id string ,category_id string ,category_code string ,brand
string ,price float,user_id bigint, user_session string) ROW FORMAT SERDE
'org.apache.hadoop.hive.serde2.OpenCSVSerde' WITH SERDEPROPERTIES
("separatorChar" = "," ) STORED AS textfile TBLPROPERTIES ('skip.header.line.count' = '1') ;
```



```
hive> CREATE EXTERNAL TABLE IF NOT EXISTS hiveecom_data(event_time timestamp, ev
   ent_type string , product_id string ,category_id string ,category_code string ,b
   rand string ,price float,user_id bigint, user_session string) ROW FORMAT SERDE
   'org.apache.hadoop.hive.serde2.OpenCSVSerde' WITH SERDEPROPERTIES ("separatorChar
   " = "," ) STORED AS textfile TBLPROPERTIES ('skip.header.line.count' = '1') ;
OK
Time taken: 0.598 seconds
hive>
```

Checking the data base

```
DESCRIBE hiveecom_data ;
```



```
hive> DESCRIBE hiveecom_data ;
OK
event_time          string          from deserializer
event_type          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
Time taken: 0.181 seconds, Fetched: 9 row(s)
hive>
```

5. Load Data into table

Now that the schema is created, loaded data into table from HDFS location. In HDFS data is present in two files. Combining the data from those files into a single table.

Commands:

```
LOAD DATA INPATH '/hivestudy/2019-Oct.csv' into table hiveecom_data ;
LOAD DATA INPATH '/hivestudy/2019-Nov.csv' into table hiveecom_data ;
```

```

hive> LOAD DATA INPATH '/hivestudy/2019-Oct.csv' into table hiveecom_data ;
Loading data to table hivecs.hiveecom_data
OK
Time taken: 1.186 seconds
hive> LOAD DATA INPATH '/hivestudy/2019-Nov.csv' into table hiveecom_data ;
Loading data to table hivecs.hiveecom_data
OK
Time taken: 0.659 seconds
hive>

```

Checking first few rows of the table:

Command:

```
SELECT * FROM hiveecom_data limit 3 ;
```

```

hive> SELECT * FROM hiveecom_data limit 3 ;
OK
2019-11-01 00:00:02 UTC view      5802432 1487580009286598681          0
.32      562076640          09fafd6c-6c99-46b1-834f-33527f4de241
2019-11-01 00:00:09 UTC cart      5844397 1487580006317032337          2
.38      553329724          2067216c-31b5-455d-a1cc-af0575a34ffb
2019-11-01 00:00:10 UTC view      5837166 1783999064103190764          pnb    2
2.22      556138645          57ed222e-a54a-4907-9944-5a875c2d7f4f
Time taken: 1.956 seconds, Fetched: 3 row(s)
hive>

```

6. Create another table schema & load data

In the above data types for all columns of the above table is string type, CsvSerde format gives an error while trying to alter data types in that table. Due to this reason, created a new table with required data types for each column and also added another column for simplicity of our analysis. Once table is created, loaded data from previously created table (hiveecom_data) into the newly created table by doing modifications to its data types

A new column has been added i.e. event_month which is derived from event_time column

```

CREATE TABLE IF NOT EXISTS hiveecom_data1
(event_time timestamp,
event_month string,
event_type string,
product_id string ,
category_id string ,
category_code string ,
brand string ,
price float,
user_id bigint,
user_session string) ;

```

```
hive> CREATE TABLE IF NOT EXISTS hiveecom_data1
  > (event_time timestamp,
  > event_month string,
  > event_type string,
  > product_id string ,
  > category_id string ,
  > category_code string ,
  > brand string ,
  > price float,
  > user_id bigint,
  > user_session string)
  > ;
OK
Time taken: 0.101 seconds
hive>
```

Snapshot: Create table hiveecom_data1

```
DESCRIBE hiveecom_data1 ;
```

```
hive> DESCRIBE hiveecom_data1 ;
OK
col_name      data_type      comment
event_time      timestamp
event_month     string
event_type      string
product_id      string
category_id     string
category_code   string
brand           string
price            float
user_id          bigint
user_session    string
Time taken: 0.045 seconds, Fetched: 10 row(s)
hive>
```

Snapshot: DESCRIBE hiveecom_data1

```
INSERT INTO TABLE hiveecom_data1
SELECT substr(event_time,1,19) AS event_time,
(CASE WHEN event_time REGEXP '-10-' THEN 'Oct' WHEN event_time REGEXP '-11-' THEN
'Nov' ELSE 'Other_months' END) AS event_month,
event_type,
product_id,
category_id,
category_code,
brand,
CAST (price AS float) AS price,
CAST(user_id AS bigint) AS user_id,
user_session
FROM hiveecom_data;
```

Snapshot: Inserting data into the new table

7. Data Quality Checks :-

- a. Checked Month & year to make sure data only belongs to 2019 Oct & Nov months.

No discrepancies have been found for this.

```
SELECT DISTINCT(month(from_unixtime(unix_timestamp(event_time, 'yyyy-MM-dd HH:mm:ss')))) AS event_month FROM hiveecom_data1;
```

```

hive> SELECT DISTINCT(month(from_unixtime(unix_timestamp(event_time, 'yyyy-MM-dd HH:mm:ss')))) AS event_month FROM hivecom_data ;
Query ID = hadoop_20210225163326_d2bcd045-d494-485f-9150-a38655798fc8
Total jobs = 1
Launching Job 1 out of 1
Status: Running (Executing on YARN cluster with App id application_1614268971294_0004)

-----  

      VERTICES    MODE      STATUS  TOTAL  COMPLETED  RUNNING  PENDING  FAILED  KILLED  

Map 1 ..... container  SUCCEEDED   8       8       0       0       0       0       0  

Reducer 2 ..... container  SUCCEEDED   4       4       0       0       0       0       0  

-----  

VERTICES: 02/02  [=====>>>] 100%  ELAPSED TIME: 55.01 s  

-----  

OK  

event_month  

11  

10  

Time taken: 56.041 seconds, Fetched: 2 row(s)
hive>

```

Snapshot: Month data quality check

```
SELECT DISTINCT(year(from_unixtime(unix_timestamp(event_time, 'yyyy-MM-dd HH:mm:ss')))) AS event_year FROM hiveecom_data1 ;
```

```
hive> SELECT DISTINCT(year(from_unixtime(unix_timestamp(event_time, 'yyyy-MM-dd HH:mm:ss')))) AS event_year FROM hiveecom_data1 ;
Query ID = hadoop_20210225163502_5f27d75d-1c18-4456-9ced-f131efef696d
Total jobs = 1
Launching Job 1 out of 1
Status: Running (Executing on YARN cluster with App id application_1614268971294_0004)

-----  

      VERTICES    MODE     STATUS  TOTAL  COMPLETED  RUNNING  PENDING  FAILED  KILLED  

-----  

Map 1 ..... container  SUCCEEDED    8        8        0        0        0        0  

Reducer 2 ..... container  SUCCEEDED    4        4        0        0        0        0  

-----  

VERTICES: 02/02  [=====>>] 100%  ELAPSED TIME: 53.27 s  

-----  

OK  

event_year  

2019  

Time taken: 54.006 seconds, Fetched: 1 row(s)
hive>
```

Snapshot: Year data quality check

- b. Checked event_type column if there are more than 4 distinct entries present apart from the ones mentioned in data description table.
 No discrepancies were found.

```
SELECT DISTINCT(event_type) AS event_type FROM hiveecom_data1 ;
```

```
hive> SELECT DISTINCT(event_type) AS event_type FROM hiveecom_data1 ;
Query ID = hadoop_20210225163627_ec01fa71-4797-4434-8bd2-31091f833744
Total jobs = 1
Launching Job 1 out of 1
Status: Running (Executing on YARN cluster with App id application_1614268971294_0004)

-----  

      VERTICES    MODE     STATUS  TOTAL  COMPLETED  RUNNING  PENDING  FAILED  KILLED  

-----  

Map 1 ..... container  SUCCEEDED    8        8        0        0        0        0  

Reducer 2 ..... container  SUCCEEDED    4        4        0        0        0        0  

-----  

VERTICES: 02/02  [=====>>] 100%  ELAPSED TIME: 28.85 s  

-----  

OK  

event_type  

view  

purchase  

cart  

remove_from_cart  

Time taken: 29.517 seconds, Fetched: 4 row(s)
hive>
```

Snapshot: Event_type data quality check

- c. General statistics regarding data which include how many files are present, rows in the table, raw size & compressed size of the table etc.

```
ANALYZE TABLE hiveecom_data1 COMPUTE STATISTICS NOSCAN;
```

```
hive> ANALYZE TABLE hiveecom_data1 COMPUTE STATISTICS NOSCAN;
OK
hiveecom_data1.event_time    hiveecom_data1.event_month    hiveecom_data1.event_type    hiveecom_data1.product_id    hiveecom_data1.category_id    hiveecom_data1.category_code
hiveecom_data1.brand    hiveecom_data1.price    hiveecom_data1.user_id    hiveecom_data1.user_session
Time taken: 0.329 seconds
hive>
```

Snapshot: Analyze table hiveecom_data1

8.Optimization in Hive

a. Enabling bucketing & vectorized Execution

```
set hive.enforce.bucketing= true;
```

```
set hive.vectorized.execution.enabled = true;
```

```
hive> set hive.enforce.bucketing= true;
hive> set hive.vectorized.execution.enabled = true;
```

Snapshot: enable bucket and vectorized execution

9. Create partitioned & bucketed table

To get insights stated in problem statement, most queries require categorization of event_time as October & November months. Also, few queries required frequent group by operations on brand column. Since the cardinality of event_month (derived column) and event_type is low, therefore we have performed two level static partitioning on these columns as below.

All insights except for query 4 and 5 had to be drawn when the event_type was ‘purchase’, hence performed static partitioning

On the other hand, brand column has high cardinalities, chose to perform bucketing (10 buckets) over this column

```
CREATE TABLE IF NOT EXISTS hive_h1
(event_time timestamp , product_id string, category_id string, brand string, price float,
user_id bigint)
PARTITIONED BY (event_month string, event_type string)
CLUSTERED BY (brand) INTO 10 BUCKETS ;
```

```
hive> CREATE TABLE IF NOT EXISTS hive_h1
> (event_time timestamp , product_id string, category_id string, brand string, price float, user_id bigint)
> PARTITIONED BY (event_month string, event_type string)
> CLUSTERED BY (brand) INTO 10 BUCKETS ;
```

OK

Time taken: 0.077 seconds

hive>

Snapshot: Create table hive_h1

```
DESC hive_h1 ;
```

```

hive> DESC hive_h1 ;
OK
col_name          data_type      comment
event_time        timestamp
product_id        string
category_id       string
brand             string
price             float
user_id           bigint
event_month       string
event_type        string

# Partition Information
# col_name          data_type      comment
event_month       string
event_type        string
Time taken: 0.074 seconds, Fetched: 14 row(s)

```

Snapshot: desc hive_h1

10. Load data into partitioned & bucketed table

Loading data twice into the partitioned & bucketed table. Since its static partitioning, first inserted data where month was October and event_type was purchase.

After the first set of data was inserted, we have added data where month was November and event_type was purchase

```

INSERT INTO TABLE hive_h1
PARTITION (event_month = 'Oct', event_type = 'purchase')
SELECT event_time, product_id ,category_id ,brand ,price , user_id
FROM hiveecom_data1 WHERE event_month = 'Oct' AND event_type = 'purchase' ;

```

```

hadoop@ip-172-31-81-235:-
Time taken: 0.03 Seconds
hive> INSERT INTO TABLE hive_h1
> PARTITION (event_month = 'Oct', event_type = 'purchase')
> SELECT event_time, product_id ,category_id ,brand ,price , user_id
> FROM hiveecom_data1 WHERE event_month = 'oct' AND event_type = 'purchase'
;
Query ID = hadoop_20210225165146_e2f26b9b-8172-4ae0-bf8e-b5914dec9d4d
Total jobs = 1
Launching Job 1 out of 1
Tez session was closed. Reopening...
Session re-established.
Status: Running (Executing on YARN cluster with App id application_1614268971294
-0006)
Map 1: 0/8          Reducer 2: 0/10
Map 1: 0/8          Reducer 2: 0/10
Map 1: 0/8          Reducer 2: 0/10
Map 1: 0(+1)/8     Reducer 2: 0/10
Map 1: 0(+2)/8     Reducer 2: 0/10
Map 1: 0(+3)/8     Reducer 2: 0/10
Map 1: 1(+2)/8     Reducer 2: 0/10
Map 1: 1(+3)/8     Reducer 2: 0/10
Map 1: 3(+3)/8     Reducer 2: 0/10
Map 1: 3(+3)/8     Reducer 2: 0/10
Map 1: 5(+1)/8     Reducer 2: 0/10
Map 1: 5(+1)/8     Reducer 2: 0/10
Map 1: 6(+0)/8     Reducer 2: 0/10
Map 1: 6(+1)/8     Reducer 2: 0/10
Map 1: 6(+2)/8     Reducer 2: 0(+1)/10
Map 1: 6(+2)/8     Reducer 2: 0(+2)/10
Map 1: 7(+1)/8     Reducer 2: 0(+2)/10
Map 1: 8/8          Reducer 2: 0(+3)/10
Map 1: 8/8          Reducer 2: 1(+3)/10
Map 1: 8/8          Reducer 2: 2(+3)/10
Map 1: 8/8          Reducer 2: 3(+2)/10
Map 1: 8/8          Reducer 2: 4(+2)/10
Map 1: 8/8          Reducer 2: 5(+3)/10
Map 1: 8/8          Reducer 2: 6(+2)/10
Map 1: 8/8          Reducer 2: 6(+2)/10
Map 1: 8/8          Reducer 2: 7(+2)/10
Map 1: 8/8          Reducer 2: 7(+3)/10
Map 1: 8/8          Reducer 2: 8(+2)/10
Map 1: 8/8          Reducer 2: 10/10
Loading data to table hivecom.hive_h1 partition (event_month=Oct, event_type=purchase)
OK
Time taken: 52.358 seconds

```

Snapshot: Insert table hive_h1

```

INSERT INTO TABLE hive_h1
PARTITION (event_month = 'Nov', event_type = 'purchase')
SELECT event_time, product_id ,category_id ,brand ,price , user_id
FROM hiveecom_data1 WHERE event_month = 'Nov' AND event_type = 'purchase' ;

```

```

hadoop@ip-172-31-81-235:~ 
Map 1: 8/8 Reducer 2: 10/10
Loading data to table hivecs.hive_h1 partition (event_month=Oct, event_type=purchase)
OK
Time taken: 52.358 seconds
hive> INSERT INTO TABLE hive_h1
> PARTITION (event_month = 'Nov', event_type = 'purchase')
> SELECT event_time, product_id, category_id, brand, price, user_id
> FROM hiveecom_data1 WHERE event_month = 'Nov' AND event_type = 'purchase';
Query ID = hadoop_20210225165440_630ed109-cc3e-4310-8ad3-ef80c3713234
Total jobs = 1
Launching Job 1 out of 1
Status: Running (Executing on YARN cluster with App id application_1614268971294
_0006)
Map 1: 0/8 Reducer 2: 0/10
Map 1: 0/8 Reducer 2: 0/10
Map 1: 0/8 Reducer 2: 0/10
Map 1: 0(+1)/8 Reducer 2: 0/10
Map 1: 0(+2)/8 Reducer 2: 0/10
Map 1: 0(+3)/8 Reducer 2: 0/10
Map 1: 0(+4)/8 Reducer 2: 0/10
Map 1: 0(+5)/8 Reducer 2: 0/10
Map 1: 0(+6)/8 Reducer 2: 0/10
Map 1: 0(+7)/8 Reducer 2: 0/10
Map 1: 0(+8)/8 Reducer 2: 0/10
Map 1: 0(+9)/8 Reducer 2: 0/10
Map 1: 0(+10)/8 Reducer 2: 0/10
Map 1: 1(+1)/8 Reducer 2: 0/10
Map 1: 1(+2)/8 Reducer 2: 0/10
Map 1: 1(+3)/8 Reducer 2: 0/10
Map 1: 2(+1)/8 Reducer 2: 0/10
Map 1: 2(+2)/8 Reducer 2: 0/10
Map 1: 2(+3)/8 Reducer 2: 0/10
Map 1: 3(+1)/8 Reducer 2: 0/10
Map 1: 3(+2)/8 Reducer 2: 0/10
Map 1: 4(+1)/8 Reducer 2: 0/10
Map 1: 4(+2)/8 Reducer 2: 0/10
Map 1: 6(+0)/8 Reducer 2: 0/10
Map 1: 6(+1)/8 Reducer 2: 0/10
Map 1: 6(+2)/8 Reducer 2: 0/10
Map 1: 6(+3)/8 Reducer 2: 0(+1)/10
Map 1: 7(+1)/8 Reducer 2: 0(+2)/10
Map 1: 8/8 Reducer 2: 0(+2)/10
Map 1: 8/8 Reducer 2: 0(+3)/10
Map 1: 8/8 Reducer 2: 1(+3)/10
Map 1: 8/8 Reducer 2: 2(+3)/10
Map 1: 8/8 Reducer 2: 3(+3)/10
Map 1: 8/8 Reducer 2: 5(+3)/10
Map 1: 8/8 Reducer 2: 6(+3)/10
Map 1: 8/8 Reducer 2: 7(+2)/10
Map 1: 8/8 Reducer 2: 7(+3)/10
Map 1: 8/8 Reducer 2: 9(+1)/10
Map 1: 8/8 Reducer 2: 10/10
Loading data to table hivecs.hive_h1 partition (event_month=Nov, event_type=purchase)
OK
Time taken: 38.556 seconds
hive>

```

Verifying data load:

Command:

`SELECT *`

`FROM hive_h1 AS h`

`LIMIT 10;`

```

hive> SELECT *
> FROM hive_h1 AS h
> LIMIT 10 ;
OK
2019-11-17 21:41:15      5875408 2084144451428549153      2.05    464392142    Nov    purchase
2019-11-17 21:41:15      5875407 2084144451428549153      2.05    464392142    Nov    purchase
2019-11-17 21:41:15      5875376 2084144451428549153      2.05    464392142    Nov    purchase
2019-11-17 21:41:15      5875366 2084144451428549153      2.05    464392142    Nov    purchase
2019-11-17 21:41:15      5875364 2084144451428549153      2.05    464392142    Nov    purchase
2019-11-17 21:41:15      5875361 2084144451428549153      2.05    464392142    Nov    purchase
2019-11-17 21:41:15      5875360 2084144451428549153      2.05    464392142    Nov    purchase
2019-11-17 21:41:15      5875359 2084144451428549153      2.05    464392142    Nov    purchase
2019-11-17 21:41:15      5875334 2084144451428549153      2.05    464392142    Nov    purchase
2019-11-17 21:41:15      5754898 1487580005092295511      4.76    464392142    Nov    purchase
Time taken: 0.251 seconds, Fetched: 10 row(s)
hive>

```

Snapshot: Select * from hive_h1

NEED TO TAKE SCREENSHOT OF COMPARISON DATA FOR EXECUTION TIME FOR BOTH TABLE as given below

SELECT * FROM hiveecom_data1 limit 5;

SELECT * FROM hive_h1 limit 5;

Query 1

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

Command before partitioning:

// summing the price column when month was October & event_type was purchase

```
SELECT ROUND(SUM(price),2) AS Oct_Rev
FROM hivecom_data1
WHERE event_month = 'Oct' AND event_type = 'purchase';
```

```
hadoop@ip-172-31-81-235:~
```

Time	Event Month	Event Type	Price	Partition	File
2019-11-17 21:41:15	2084144451428549153	2.05	464392142	Nov	purchase
2019-11-17 21:41:15	5875407	2.05	464392142	Nov	purchase
2019-11-17 21:41:15	5875376	2.05	464392142	Nov	purchase
2019-11-17 21:41:15	5875366	2.05	464392142	Nov	purchase
2019-11-17 21:41:15	5875364	2.05	464392142	Nov	purchase
2019-11-17 21:41:15	5875361	2.05	464392142	Nov	purchase
2019-11-17 21:41:15	5875360	2.05	464392142	Nov	purchase
2019-11-17 21:41:15	5875359	2.05	464392142	Nov	purchase
2019-11-17 21:41:15	5875334	2.05	464392142	Nov	purchase
2019-11-17 21:41:15	5754898	4.76	464392142	Nov	purchase

Time taken: 0.251 seconds, Fetched: 10 row(s)

```
hive> SELECT ROUND(SUM(price),2) AS Oct_Rev
> FROM hivecom_data1
> WHERE event_month = 'Oct' AND event_type = 'purchase';
Query ID = hadoop_20210225170415_989c5d55-cff7-489b-984f-ef615dc61748
Total jobs = 1
Launching Job 1 out of 1
Tez session was closed. Reopening...
Session re-established.
Status: Running (Executing on YARN cluster with App id application_1614268971294_0007)
Map 1: 0/8 Reducer 2: 0/1
Map 1: 0/8 Reducer 2: 0/1
Map 1: 0/8 Reducer 2: 0/1
Map 1: 0(+1)/8 Reducer 2: 0/1
Map 1: 0(+2)/8 Reducer 2: 0/1
Map 1: 0(+3)/8 Reducer 2: 0/1
Map 1: 0(+2)/8 Reducer 2: 0/1
Map 1: 2(+2)/8 Reducer 2: 0/1
Map 1: 2(+3)/8 Reducer 2: 0/1
Map 1: 3(+2)/8 Reducer 2: 0/1
Map 1: 3(+3)/8 Reducer 2: 0/1
Map 1: 3(+3)/8 Reducer 2: 0/1
Map 1: 4(+2)/8 Reducer 2: 0/1
Map 1: 5(+1)/8 Reducer 2: 0/1
Map 1: 6(+0)/8 Reducer 2: 0/1
Map 1: 6(+1)/8 Reducer 2: 0/1
Map 1: 6(+2)/8 Reducer 2: 0/1
Map 1: 6(+2)/8 Reducer 2: 0(+1)/1
Map 1: 7(+1)/8 Reducer 2: 0(+1)/1
Map 1: 8/8 Reducer 2: 0(+1)/1
Map 1: 8/8 Reducer 2: 1/1
OK
1211538.43
Time taken: 38.632 seconds, Fetched: 1 row(s)
```

Snapshot: Query 1 results (before optimization)

Command after partitioning:

// summing the price column when month was October & event_type was purchase

```
SELECT ROUND(SUM(price),2) AS Oct_Rev
FROM hive_h1
WHERE event_month = 'Oct' AND event_type = 'purchase';
```

```

hadoop@ip-172-31-81-235:~
Map 1: 7(+1)/8 Reducer 2: 0(+1)/1
Map 1: 8/8 Reducer 2: 0(+1)/1
Map 1: 8/8 Reducer 2: 1/1
OK
1211538.43
Time taken: 38.632 seconds, Fetched: 1 row(s)
hive> SELECT ROUND(SUM(price),2) AS Oct_Rev
>   FROM hive_h1
> WHERE event_month = 'Oct' AND event_type = 'purchase';
Query ID = hadoop_20210225170710_a6d6c77d-af32-4b24-b9b3-ff56985e7897
Total jobs = 1
Launching Job 1 out of 1
Status: Running (Executing on YARN cluster with App id application_1614268971294_0007)
Map 1: 0/1 Reducer 2: 0/1
Map 1: 0/1 Reducer 2: 0/1
Map 1: 0(+1)/1 Reducer 2: 0/1
Map 1: 1/1 Reducer 2: 0(+1)/1
Map 1: 1/1 Reducer 2: 1/1
OK
1211538.43
Time taken: 8.341 seconds, Fetched: 1 row(s)
hive>

```

Snapshot: Query 1 results (after optimization)

Here without partitioning, it took 38.632 seconds where as with partition, it took 8.341 seconds.

So obviously with partition, it is taking less time for execution so we will use partition table to run rest of queries.

Here the total revenue generated due to purchases made in the month of October is Rs 1211538.43

Query 2

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

// For each month, aggregating the price where event_type was purchase

```

SELECT event_month, ROUND(SUM(price),2) AS Total_purchases
FROM hive_h1
WHERE event_type = 'purchase'
GROUP BY event_month ;

```

```

hadoop@ip-172-31-81-235:~ 
OK
1211538.43
Time taken: 8.341 seconds, Fetched: 1 row(s)
hive> SELECT event_month, ROUND(SUM(price),2) AS Total_purchases
    > FROM hive_h1
    > WHERE event_type = 'purchase'
    > GROUP BY event_month ;
Query ID = hadoop_20210225170905_a8c9814e-1d68-4517-b369-a3bf3f9874fa
Total jobs = 1
Launching Job 1 out of 1
Status: Running (Executing on YARN cluster with App id application_161426897
_0007)

Map 1: 0/1      Reducer 2: 0/1
Map 1: 0/1      Reducer 2: 0/1
Map 1: 0(+1)/1  Reducer 2: 0/1
Map 1: 0(+1)/1  Reducer 2: 0/1
Map 1: 1/1      Reducer 2: 0(+1)/1
Map 1: 1/1      Reducer 2: 1/1
OK
Nov      1531016.9
Oct      1211538.43
Time taken: 8.627 seconds, Fetched: 2 row(s)
hive>

```

The total sum of purchases in the month of Nov is Rs 1531016.9 and in Oct is Rs 1211538.43

Query 3

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

// Aggregating prices where event type was purchased based on month of the event

```

WITH month_wise_agg AS (
SELECT event_month,SUM(price) AS current_month_sales
FROM hive_h1
WHERE event_type = 'purchase'
GROUP BY event_month
),

```

// Using lead function making a new column which displays the sales for Nov month in the previous row.

```

next_month_sales AS (
SELECT *, LEAD(current_month_sales,1) OVER (ORDER BY event_month desc) AS
next_month_sales
FROM month_wise_agg
),

```

// Calculating the difference in revenue from Oct month to Nov month

```
change_in_rev AS (
SELECT *, ROUND((next_month_sales - current_month_sales),2) AS Sales_difference
FROM next_month_sales )
```

// Change in revenue from October to November

```
SELECT Sales_difference AS revenue_change_from_October_to_November
FROM change_in_rev
WHERE Sales_difference IS NOT NULL ;
```

```
hadoop@ip-172-31-81-235:~ 
> FROM next_month_sales )
> SELECT Sales_difference AS revenue_change_from_October_to_November
> FROM change_in_rev
> WHERE Sales_difference IS NOT NULL
> ;
FAILED: SemanticException [Error 10001]: Line 9:5 Table not found 'month_wise_ag
g'
SELECT *, ROUND((next_month_sales - current_month_sales),2) AS Sales_diffe
rence
> hive
> hive;
FAILED: ParseException line 2:0 missing EOF at 'hive' near 'Sales_difference'
hive> WITH month_wise_agg AS (
> SELECT event_month,SUM(price) AS current_month_sales
> FROM hive_h1
> WHERE event_type = 'purchase'
> GROUP BY event_month
> ),
> next_month_sales AS (
> SELECT *, LEAD(current_month_sales,1) OVER (ORDER BY event_month desc) AS
next_month_sales
> FROM month_wise_agg
> ),
> change_in_rev AS (
> SELECT *, ROUND((next_month_sales - current_month_sales),2) AS Sales_diffe
rence
> FROM next_month_sales )
> SELECT Sales_difference AS revenue_change_from_October_to_November
> FROM change_in_rev
> WHERE Sales_difference IS NOT NULL ;
Query ID = hadoop_20210225172218_9f6fac10-ff99-406b-9fa5-42b6027d372d
Total jobs = 1
Launching Job 1 out of 1
Tez session was closed. Reopening...
Session re-established.
Status: Running (Executing on YARN cluster with App id application_1614268971294
_0008)
Map 1: -- Reducer 2: 0/1 Reducer 3: 0/1
Map 1: 0/1 Reducer 2: 0/1 Reducer 3: 0/1
Map 1: 0/1 Reducer 2: 0/1 Reducer 3: 0/1
Map 1: 0(+1)/1 Reducer 2: 0/1 Reducer 3: 0/1
Map 1: 0(+1)/1 Reducer 2: 0/1 Reducer 3: 0/1
Map 1: 1/1 Reducer 2: 0/1 Reducer 3: 0/1
Map 1: 1/1 Reducer 2: 0(+1)/1 Reducer 3: 0/1
Map 1: 1/1 Reducer 2: 1/1 Reducer 3: 0(+1)/1
Map 1: 1/1 Reducer 2: 1/1 Reducer 3: 1/1
OK
319478.47
Time taken: 17.678 seconds, Fetched: 1 row(s)
hive>
```

There is an increase in revenue of amount Rs 319478.47 /-

Query 4

Find distinct categories of products.

For this query, we have not used partitioned & bucketed table because that table contains information for event_type = purchase only.

```
SELECT category_code FROM hiveecom_data1 GROUP BY category_code;
```

```

hive> SELECT category_code FROM hiveecom_data1 GROUP BY category_code;
Query ID = hadoop_20210225181833_4a71bf02-53c1-4074-ac5c-16b7522eb14e
Total jobs = 1
Launching Job 1 out of 1
Status: Running (Executing on YARN cluster with App id application_1614268971294_0010)

-----  

      VERTICES    MODE     STATUS  TOTAL  COMPLETED  RUNNING  PENDING  FAILED  KILLED  

-----  

Map 1 ..... container  SUCCEEDED    4        4        0        0        0        0  

Reducer 2 ..... container  SUCCEEDED    4        4        0        0        0        0  

-----  

VERTICES: 02/02  [=====>>] 100%  ELAPSED TIME: 25.27 s
-----  

OK  

accessories.bag  

appliances.environment.vacuum  

appliances.personal.hair_cutter  

sport.diving  

apparel.glove  

furniture.bathroom.bath  

furniture.living_room.cabinet  

stationery.cartidge  

accessories.cosmetic_bag  

appliances.environment.air_conditioner  

furniture.living_room.chair  

Time taken: 26.274 seconds, Fetched: 12 row(s)
hive> [hadoop@ip-172-31-81-235 ~]$ █

```

Query 5

Find the total number of products available under each category. For this query, didn't use the partitioned & bucketed table because that table contains information for event_type = purchase only.

Command:

// Count of products under each category

```

SELECT category_code, COUNT(product_id) AS product_counts
FROM hivecom_data1
GROUP BY category_code
ORDER BY product_counts DESC;

```

```

hadoop@ip-172-31-81-235:~ 
Map 1: 1/1      Reducer 2: 1/1  Reducer 3: 1/1
OK
319478.47
Time taken: 17.678 seconds, Fetched: 1 row(s)
hive> SELECT category_id, COUNT(product_id) AS product_counts
> FROM hivecom_data1
> GROUP BY category_id
> ORDER BY product_counts DESC;
Query ID = hadoop_20210225172455_91ad607e-f00b-4ad9-8e52-effdd630a6a6
Total jobs = 1
Launching Job 1 out of 1
Status: Running (Executing on YARN cluster with App id application_1614268971294
_0008)

Map 1: 0/8      Reducer 2: 0/4  Reducer 3: 0/1
Map 1: 0/8      Reducer 2: 0/4  Reducer 3: 0/1
Map 1: 0/8      Reducer 2: 0/4  Reducer 3: 0/1
Map 1: 0(+2)/8 Reducer 2: 0/4  Reducer 3: 0/1
Map 1: 0(+3)/8 Reducer 2: 0/4  Reducer 3: 0/1
Map 1: 0(+3)/8 Reducer 2: 0/4  Reducer 3: 0/1
Map 1: 0(+3)/8 Reducer 2: 0/4  Reducer 3: 0/1
Map 1: 0(+3)/8 Reducer 2: 0/4  Reducer 3: 0/1
Map 1: 0(+3)/8 Reducer 2: 0/4  Reducer 3: 0/1
Map 1: 0(+3)/8 Reducer 2: 0/4  Reducer 3: 0/1
Map 1: 1(+3)/8 Reducer 2: 0/4  Reducer 3: 0/1
Map 1: 2(+3)/8 Reducer 2: 0/4  Reducer 3: 0/1
Map 1: 3(+3)/8 Reducer 2: 0/4  Reducer 3: 0/1
Map 1: 3(+3)/8 Reducer 2: 0/4  Reducer 3: 0/1
Map 1: 3(+2)/8 Reducer 2: 0/4  Reducer 3: 0/1
Map 1: 4(+2)/8 Reducer 2: 0/4  Reducer 3: 0/1
Map 1: 6(+1)/8 Reducer 2: 0/4  Reducer 3: 0/1
Map 1: 6(+2)/8 Reducer 2: 0/4  Reducer 3: 0/1
Map 1: 6(+2)/8 Reducer 2: 0(+1)/4 Reducer 3: 0/1
Map 1: 7(+1)/8 Reducer 2: 0(+1)/4 Reducer 3: 0/1
Map 1: 7(+0)/8 Reducer 2: 0(+2)/4 Reducer 3: 0/1
Map 1: 8/8      Reducer 2: 0(+3)/4 Reducer 3: 0/1
Map 1: 8/8      Reducer 2: 1(+3)/4 Reducer 3: 0/1
Map 1: 8/8      Reducer 2: 2(+2)/4 Reducer 3: 0(+1)/1
Map 1: 8/8      Reducer 2: 4/4   Reducer 3: 0(+1)/1
Map 1: 8/8      Reducer 2: 4/4   Reducer 3: 1/1
OK
1487580007675986893 497756
1487580005595612013 322269
1487580005092295511 321824
1487580005671109489 300570
1487580006317032337 283387
1602943681873052386 245975
1487580005268456287 194193
1487580005134238553 163722
1487580008246412266 157042
1487580009286598681 151845
1487580013841613016 148361

```

Note: The actual result is quite huge, snapshot only contains few rows of categories with high product counts.

Query 6

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

// Displaying brand which has most sales in both months combined

```

SELECT brand, ROUND(SUM(price),2) AS sales
FROM hive_h1
WHERE brand != "" AND event_type = 'purchase'
GROUP BY brand
ORDER BY sales DESC
LIMIT 1;

```

```

hadoop@ip-172-31-81-235:~ 
Time taken: 29.376 seconds, Fetched: 500 row(s)
hive> SELECT brand, ROUND(SUM(price),2) AS sales
> FROM hive_h1
> WHERE brand != "" AND event_type = 'purchase'
> GROUP BY brand
> ORDER BY sales DESC
> LIMIT 1;
Query ID = hadoop_20210225172712_2b151cd4-4b13-4d0c-8ffe-eb0866df9e41
Total jobs = 1
Launching Job 1 out of 1
Status: Running (Executing on YARN cluster with App id application_1614268971294_0008)
Map 1: 0/1      Reducer 2: 0/1  Reducer 3: 0/1
Map 1: 0/1      Reducer 2: 0/1  Reducer 3: 0/1
Map 1: 0(+1)/1  Reducer 2: 0/1  Reducer 3: 0/1
Map 1: 0(+1)/1  Reducer 2: 0/1  Reducer 3: 0/1
Map 1: 1/1      Reducer 2: 0(+1)/1  Reducer 3: 0/1
Map 1: 1/1      Reducer 2: 1/1  Reducer 3: 0(+1)/1
Map 1: 1/1      Reducer 2: 1/1  Reducer 3: 1/1
OK
runail 148297.94
Time taken: 9.025 seconds, Fetched: 1 row(s)
hive> 

```

Here 'runail' brand has most sales combining both months with a value of 148297.94

Query 7

Which brands increased their sales from October to November?

// Sales in Oct for each brand

```

WITH Oct_table AS (
select brand, ROUND(SUM(price),2) AS Oct_sales
FROM hive_h1
WHERE event_month = 'Oct' AND brand != "" AND event_type = 'purchase'
group by brand ),

```

// Sales in Nov for each brand

```

Nov_table AS (
select brand, ROUND(SUM(price),2) AS Nov_sales
FROM hive_h1
WHERE event_month = 'Nov' AND brand != "" AND event_type = 'purchase'
group by brand
ORDER BY Nov_sales DESC ),

```

// Combining the sales from Oct and Nov month for each brand and calculating the percentage increase in sales from October to November

```

Oct_Nov AS(
SELECT /*+ MAPJOIN (Oct_table) */ *, (((Nov_sales - Oct_sales)/Oct_sales)*100) AS
percent_increase_sales

```

```
FROM Oct_table INNER JOIN Nov_table ON Oct_table.brand = Nov_table.brand)
```

// Displaying only those brand, whose sales have increased

```
SELECT brand, percent_increase_sales
```

```
FROM Oct_Nov
```

```
WHERE percent_increase_sales > 0;
```

```
hadoop@ip-172-31-81-235:~
```

```
OK
runail 148297.94
Time taken: 9.025 seconds, Fetched: 1 row(s)
hive> WITH Oct_table AS (
    > select brand, ROUND(SUM(price),2) AS Oct_sales
    > FROM hive_hl
    > WHERE event_month = 'Oct' AND brand != "" AND event_type = 'purchase'
    > group by brand ,
    > Nov_table AS (
    > select brand, ROUND(SUM(price),2) AS Nov_sales
    > FROM hive_hl
    > WHERE event_month = 'Nov' AND brand != "" AND event_type = 'purchase'
    > group by brand
    > ORDER BY Nov_sales DESC ),
    > Oct_Nov AS(
    > SELECT /*+ MAPJOIN (Oct_table) */ *, (((Nov_sales - Oct_sales)/Oct_sales)*
100) AS percent_increase_sales
    > FROM Oct_table INNER JOIN Nov_table ON Oct_table.brand = Nov_table.brand)
    > SELECT brand, percent_increase_sales
    > FROM Oct_Nov
    > WHERE percent_increase_sales > 0;
No Stats for hivecs@hive_hl, Columns: price, brand
No Stats for hivecs@hive_hl, Columns: price, brand
Query ID = hadoop_20210225173206_8fb998fa-c8be-44fe-8834-c91653a0837f
Total jobs = 1
Launching Job 1 out of 1
Status: Running (Executing on YARN cluster with App id application_1614268971294
_0008)

Map 1: 0/1      Map 3: 0/1      Reducer 2: 0/1      Reducer 4: 0/1      Reducer 5: 0/1
Map 1: 0/1      Map 3: 0/1      Reducer 2: 0/1      Reducer 4: 0/1      Reducer 5: 0/1
Map 1: 0/1      Map 3: 0(+1)/1  Reducer 2: 0/1      Reducer 4: 0/1      Reducer 5: 0/1
Map 1: 0(+1)/1 Map 3: 0(+1)/1  Reducer 2: 0/1      Reducer 4: 0/1      Reducer 5: 0/1
Map 1: 0(+1)/1 Map 3: 0(+1)/1  Reducer 2: 0/1      Reducer 4: 0/1      Reducer 5: 0/1
Map 1: 0(+1)/1 Map 3: 0(+1)/1  Reducer 2: 0/1      Reducer 4: 0/1      Reducer 5: 0/1
Map 1: 0(+1)/1 Map 3: 1/1      Reducer 2: 0/1      Reducer 4: 0(+1)/1      Reducer
5: 0/1
Map 1: 1/1      Map 3: 1/1      Reducer 2: 0(+1)/1      Reducer 4: 0(+1)/1      R
educer 5: 0/1
Map 1: 1/1      Map 3: 1/1      Reducer 2: 0(+1)/1      Reducer 4: 1/1      Reducer
5: 0(+1)/1
Map 1: 1/1      Map 3: 1/1      Reducer 2: 1/1      Reducer 4: 1/1      Reducer 5: 0(+1)
/
Map 1: 1/1      Map 3: 1/1      Reducer 2: 1/1      Reducer 4: 1/1      Reducer 5: 1/1
OK
uno      44.58021252602188
strong   32.45114247744845
jessnail  26.84659523186387
italwax  13.031443594053652
cosmoprofi 74.6644462627406
```



```

hadoop@ip-172-31-81-235:~ 
strong 32.45114247744845
jessnail 26.84659523186387
italwax 13.031443594053652
cosmoprofi 74.6644462627406
domix 14.678310359480722
staleks 39.389511169954936
marathon 41.09947464203551
f.o.x 29.483426752996216
smart 32.416327519597246
uskusi 10.657550070299692
nagaraku 21.922128090000793
kaaral 15.266871089173073
roublöff 40.74085743091518
rosi 24.84595585367756
levrana 63.316336536575804
depilflax 3.5724972017716587
lador 18.61768757109056
misssha 66.19494060270672
zeitun 183.58168938560104
kosmekka 53.488116197183075
insight 19.274087414282747
gehwl 43.02845547118185
farmavita 54.289023967899496
sanoto 669.8103601883671
jaguar 0.7748772808522009
happyfons 36.12205706304868
scare 121.25375593680334
batiste 13.17581563956499
kiss 93.88684616296999
konad 9.57517267480366
osmo 18.081415161560134
entity 49.936419920368564
yu-r 148.2259312479275
igrobeauty 25.583070513569307
cristalinias 36.788812758693275
lowence 133.7959150057651
kerasys 21.881599405908432
freshbubble 57.62158770003138
mavala 9.11402307842753
cutrin 22.797875538631125
im 21.93944864939935
deoproce 3.8915540967049744
laboratorium 26.78296146044624
miskin 85.4403948367502
mane 289.6691121425362
tertio 4.081978319783204
ecocraft 487.8279883381924
vilenta 17.009109311740897
soleo 4.079333986287959
elizavecca 189.6639727775415

```

Type here to search

```

hadoop@ip-172-31-81-235:~ 
soleo 4.079333986287959
elizavecca 189.6639727775415
plazan 91.38798461083158
aura 111.44729005360333
enjoy 230.2781136638452
profepl 26.413881748071976
marutaka-foot 122.12515237708249
grace 1.6745937376139493
glysolid 31.34949089344615
kamill 29.32867798762101
foamie 129.70890410958904
supertan 32.04288266825493
neoleor 19.09698226215159
fly 58.518086347724626
moyou 80.03502626970227
masura 5.732698182823044
lianail 178.2060941753043
shik 44.84975457919312
artex 58.47017548999502
nitrile 37.22500236049478
provoc 28.482228046232432
orly 3.181586471331372
greymy 1575.7617254364945
nefertiti 57.005823912298716
beautyblender 38.95097790195581
blixx 62.772785622593055
grattol 101.64091166335739
kapous 18.159561873907958
oniq 16.809152314249392
freedecor 124.20494596379663
jas 10.19807409549979
reflectocil 27.958382728685145
benovy 695.852253307944
solomeya 41.38021792914671
shary 34.92476719115555
kims 91.50406011392556
estelare 6.083496324273286
godefroy 5.956831663426543
ladykin 35.750099482690004
browxenna 4.0844664536607365
concept 21.285622545435428
haruyama 31.544220925192924
beautix 16.476160073185024
yoko 33.69875903714894
markell 60.25045936395759
skinlite 36.584655029603944
keen 84.31140258091814
fedua 403.64642993508977
eos 180.84284136915716
biore 48.903544929925815

```

Type here to search

```

hadoop@ip-172-31-81-235:~ 
eos      180.84284136915716
biore    48.903544929925815
irisk    2.9699973416365557
ingarden   44.92312421663812
metzger   20.167862360308554
severina   28.1539737179326
airnails   11.186387700482541
de.lux    67.2296198108092
farmona  8.920151731798684
beauugreen 50.21211706516002
kocostar   91.38812932282448
balbcare   36.72825597115816
rasyan    53.93617021276596
skinity   40.09009009009007
lovely    37.161520981390986
kaypro    270.8784351101731
art-visage 43.249661921623165
joico     185.61911781381107
sophin    41.92122562882776
ecolab    361.9745101769069
profhenna  8.483135314989031
trind     82.15855335994902
elskin    22.525787566211307
nirvel    43.72546614327735
egomania   88.51168194139666
ovale     22.047244094488192
runail    7.295824056378544
milv      44.483910124099225
levissime  38.50998877665544
limoni    37.26029490411795
biaqua   48.280287202112646
ellips    146.5080333536709
protokeratin 126.97639751552796
likato    15.169222454907796
carmex    67.74193548387096
polarus   89.09975855211084
bluesky   2.505908468222345
kinetics   9.646130165370804
swarovski  61.19029836911325
koelf     20.003311806590496
finish    134.1736125228705
treaclemoon 11.091387647670933
koelcia   103.15315315315314
estel     10.966343778367625
bpw.style   28.21679636022693
matrix    14.907577275880668
beauty-free 221.71716260353324
coifin    58.1937984496124
bodyton   0.31242280250520815
dizao     15.428564452528903

```

Type here to search

```

hadoop@ip-172-31-81-235:~ 
de.lux    67.2296198108092
farmona  8.920151731798684
beauugreen 50.21211706516002
kocostar   91.38812932282448
balbcare   36.72825597115816
rasyan    53.93617021276596
skinity   40.09009009009007
lovely    37.161520981390986
kaypro    270.8784351101731
art-visage 43.249661921623165
joico     185.61911781381107
sophin    41.92122562882776
ecolab    361.9745101769069
profhenna  8.483135314989031
trind     82.15855335994902
elskin    22.525787566211307
nirvel    43.72546614327735
egomania   88.51168194139666
ovale     22.047244094488192
runail    7.295824056378544
milv      44.483910124099225
levissime  38.50998877665544
limoni    37.26029490411795
biaqua   48.280287202112646
ellips    146.5080333536709
protokeratin 126.97639751552796
likato    15.169222454907796
carmex    67.74193548387096
polarus   89.09975855211084
bluesky   2.505908468222345
kinetics   9.646130165370804
swarovski  61.19029836911325
koelf     20.003311806590496
finish    134.1736125228705
treaclemoon 11.091387647670933
koelcia   103.15315315315314
estel     10.966343778367625
bpw.style   28.21679636022693
matrix    14.907577275880668
beauty-free 221.71716260353324
coifin    58.1937984496124
bodyton   0.31242280250520815
dizao     15.428564452528903
Time taken: 15.068 seconds, Fetched: 152 row(s)
hive> 

```

Type here to search

Query 8

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

// Top 10 users which have spent the most

```

SELECT user_id,
ROUND(SUM(price),2) AS amt_spent ,
ROW_NUMBER() OVER (ORDER BY SUM(price) DESC) AS top_users
FROM hive_h1
WHERE event_type = 'purchase'
GROUP BY user_id
LIMIT 10;

```

```

hadoop@ip-172-31-81-235:~$ koelf 20.003311806590496
finish 134.1736125228705
treaclemoon 11.091387647670933
koelcia 103.15315315315314
estel 10.96634378367625
bpw.style 28.21679636022693
matrix 14.907577275880668
beauty-free 221.71716260353324
cofin 58.1937984496124
bodyton 0.31242280250520815
dizao 15.428564452528903
candy 49.42799461641991
chi 50.055719618877816
latinoil 54.131933311958946
veraclara 42.10736379964078
cosima 3.4602076124567436
Time taken: 15.068 seconds, Fetched: 152 row(s)
hive> SELECT user_id,
    > ROUND(SUM(price),2) AS amt_spent ,
    > ROW_NUMBER() OVER (ORDER BY SUM(price) DESC) AS top_users
    > FROM hive_h1
    > WHERE event_type = 'purchase'
    > GROUP BY user_id
    > LIMIT 10;
Query ID = hadoop_20210225173715_686feba6-ab3e-420f-8e16-ba93ef59d047
Total jobs = 1
Launching Job 1 out of 1
Status: Running (Executing on YARN cluster with App id application_1614268971294
_0008)
Map 1: 0/1 Reducer 2: 0/1 Reducer 3: 0/1
Map 1: 0/1 Reducer 2: 0/1 Reducer 3: 0/1
Map 1: 0(+1)/1 Reducer 2: 0/1 Reducer 3: 0/1
Map 1: 0(+1)/1 Reducer 2: 0/1 Reducer 3: 0/1
Map 1: 1/1 Reducer 2: 0(+1)/1 Reducer 3: 0/1
Map 1: 1/1 Reducer 2: 1/1 Reducer 3: 0(+1)/1
Map 1: 1/1 Reducer 2: 1/1 Reducer 3: 1/1
OK
557790271 2715.87 1
150318419 1645.97 2
562167663 1352.85 3
531900924 1329.45 4
557850743 1295.48 5
522130011 1185.39 6
561592095 1109.7 7
431950134 1097.59 8
566576008 1056.36 9
521347209 1040.91 10
Time taken: 11.177 seconds, Fetched: 10 row(s)
hive>

```

Clearing Up Resources

1. Dropping tables & database

Finally, after getting all the insights needed, dropped the database & its tables in one go by using cascade keyword with drop databases statement.

Command:

```
DROP DATABASE IF EXISTS hivecs CASCADE;
```

```

hadoop@ip-172-31-81-235:~ 
      VERTICES    MODE     STATUS   TOTAL  COMPLETED  RUNNING  PENDING  FAILED  KILLED
Map 1 ..... container  SUCCEEDED   4       4        0        0        0        0        0
Reducer 2 ..... container  SUCCEEDED   4       4        0        0        0        0        0
Reducer 3 ..... container  SUCCEEDED   1       1        0        0        0        0        0
VERTICES: 03/03 [=====>>] 100% ELAPSED TIME: 25.17 s
OK
12
Time taken: 25.939 seconds, Fetched: 1 row(s)
hive> SELECT category_code FROM hiveecom_datal GROUP BY category_code;
Query ID = hadoop_20210225181833_4a71bf02-53c1-4074-ac5c-16b7522eb14e
Total jobs = 1
Launching Job 1 out of 1
Status: Running (Executing on YARN cluster with App id application_1614268971294_0010)

      VERTICES    MODE     STATUS   TOTAL  COMPLETED  RUNNING  PENDING  FAILED  KILLED
Map 1 ..... container  SUCCEEDED   4       4        0        0        0        0        0
Reducer 2 ..... container  SUCCEEDED   4       4        0        0        0        0        0
VERTICES: 02/02 [=====>>] 100% ELAPSED TIME: 25.27 s
OK
accessories.bag
appliances.environment.vacuum
appliances.personal.hair_cutter
sport.diving

apparel.glove
furniture.bathroom.bath
furniture.living_room.cabinet
stationery.cartige
accessories.cosmetic_bag
appliances.environment.air_conditioner
furniture.living_room.chair
Time taken: 26.274 seconds, Fetched: 12 row(s)
hive> [hadoop@ip-172-31-81-235 ~]$ hive

Logging initialized using configuration in file:/etc/hive/conf.dist/hive-log4j2.properties Async: false
hive> DROP DATABASE IF EXISTS hivescs CASCADE;
OK
Time taken: 1.101 seconds
hive> SHOW databases ;
OK
default
Time taken: 0.204 seconds, Fetched: 1 row(s)
hive> 

```

2. Exiting hive

Command:

EXIT ;

```

hadoop@ip-172-31-81-235:~ 
      at org.apache.hadoop.hive.cli.CliDriver.processLine(CliDriver.java:403)
      at org.apache.hadoop.hive.cli.CliDriver.executeDriver(CliDriver.java:821)
)
      at org.apache.hadoop.hive.cli.CliDriver.run(CliDriver.java:759)
      at org.apache.hadoop.hive.cli.CliDriver.main(CliDriver.java:686)
      at sun.reflect.NativeMethodAccessorImpl.invoke0(Native Method)
      at sun.reflect.NativeMethodAccessorImpl.invoke(NativeMethodAccessorImpl.
java:62)
      at sun.reflect.DelegatingMethodAccessorImpl.invoke(DelegatingMethodAcces
sorImpl.java:43)
      at java.lang.reflect.Method.invoke(Method.java:498)
      at org.apache.hadoop.util.RunJar.run(RunJar.java:239)
      at org.apache.hadoop.util.RunJar.main(RunJar.java:153)
FAILED: ParseException line 1:0 cannot recognize input near 'hadoop' 'fs' '-'
hive> EXIT hive ;
NoViableAltException (24@[])
      at org.apache.hadoop.hive.ql.parse.HiveParser.statement(HiveParser.java:1300)
      at org.apache.hadoop.hive.ql.parse.ParseDriver.parse(ParseDriver.java:208)
      at org.apache.hadoop.hive.ql.parse.ParseUtils.parse(ParseUtils.java:77)
      at org.apache.hadoop.hive.ql.parse.ParseUtils.parse(ParseUtils.java:70)
      at org.apache.hadoop.hive.ql.Driver.compile(Driver.java:468)
      at org.apache.hadoop.hive.ql.Driver.compileInternal(Driver.java:1317)
      at org.apache.hadoop.hive.ql.Driver.runInternal(Driver.java:1457)
      at org.apache.hadoop.hive.ql.Driver.run(Driver.java:1237)
      at org.apache.hadoop.hive.ql.Driver.run(Driver.java:1227)
      at org.apache.hadoop.hive.ql.CliDriver.processLocalCmd(CliDriver.java:233)
      at org.apache.hadoop.hive.ql.CliDriver.processCmd(CliDriver.java:184)
      at org.apache.hadoop.hive.ql.CliDriver.processLine(CliDriver.java:403)
      at org.apache.hadoop.hive.ql.CliDriver.executeDriver(CliDriver.java:821)
      at org.apache.hadoop.hive.ql.CliDriver.run(CliDriver.java:759)
      at org.apache.hadoop.hive.ql.CliDriver.main(CliDriver.java:686)
      at sun.reflect.NativeMethodAccessorImpl.invoke0(Native Method)
      at sun.reflect.NativeMethodAccessorImpl.invoke(NativeMethodAccessorImpl.java:62)
      at sun.reflect.DelegatingMethodAccessorImpl.invoke(DelegatingMethodAccessorImpl.java:43)
      at java.lang.reflect.Method.invoke(Method.java:498)
      at org.apache.hadoop.util.RunJar.run(RunJar.java:239)
      at org.apache.hadoop.util.RunJar.main(RunJar.java:153)
FAILED: ParseException line 1:0 cannot recognize input near 'EXIT' 'hive' '<EOF>'.
hive> EXIT ;
[hadoop@ip-172-31-81-235 ~]$ hadoop fs -rm -r /hivestudy
Deleted /hivestudy
[hadoop@ip-172-31-81-235 ~]$ hadoop fs -ls /
-bash: hadoop: command not found
[hadoop@ip-172-31-81-235 ~]$ hadoop fs -ls /
Found 4 items
drwxr-xr-x  - hdfs hadoop          0 2021-02-25 16:01 /apps
drwxrwxrwt  - hdfs hadoop          0 2021-02-25 16:03 /tmp
drwxr-xr-x  - hdfs hadoop          0 2021-02-25 16:01 /user
drwxr-xr-x  - hdfs hadoop          0 2021-02-25 16:01 /var
[hadoop@ip-172-31-81-235 ~]$ 

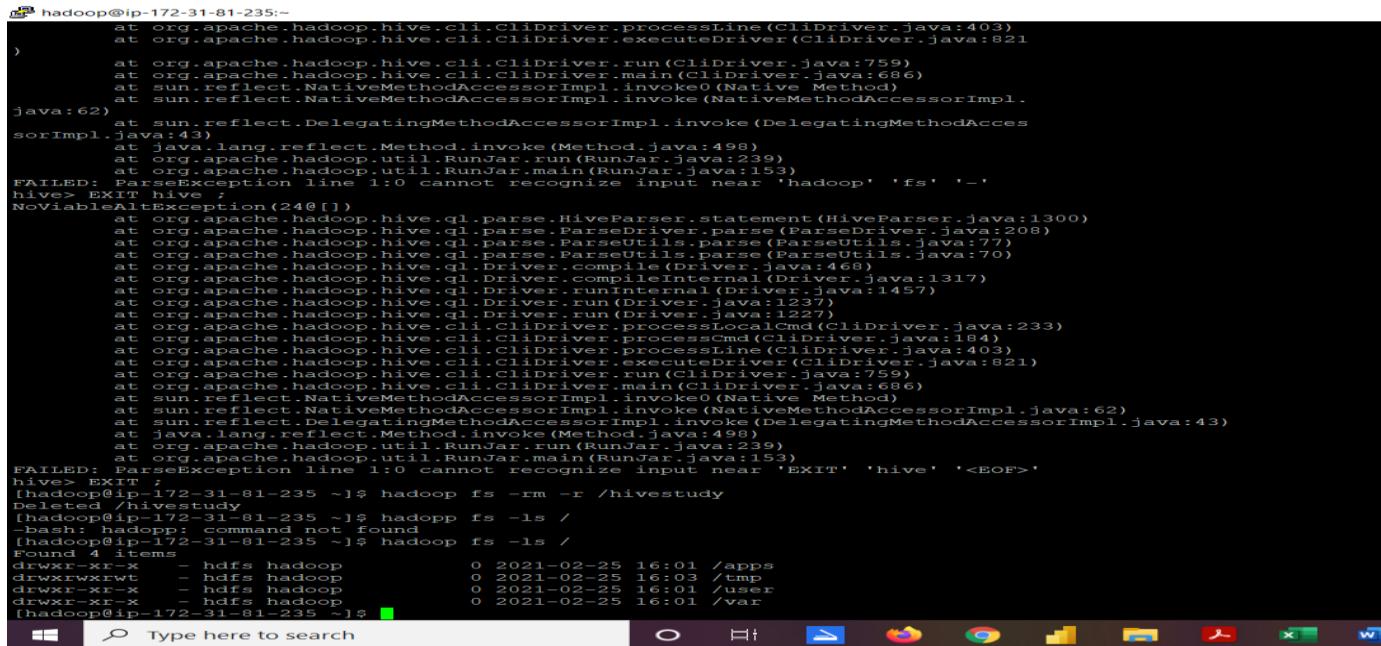
```

3. Deleting directory

Cleaned up HDFS storage as well by deleting the folder created in which all the data was stored.

Command:

```
hadoop fs -rm -r /hivestudy
```

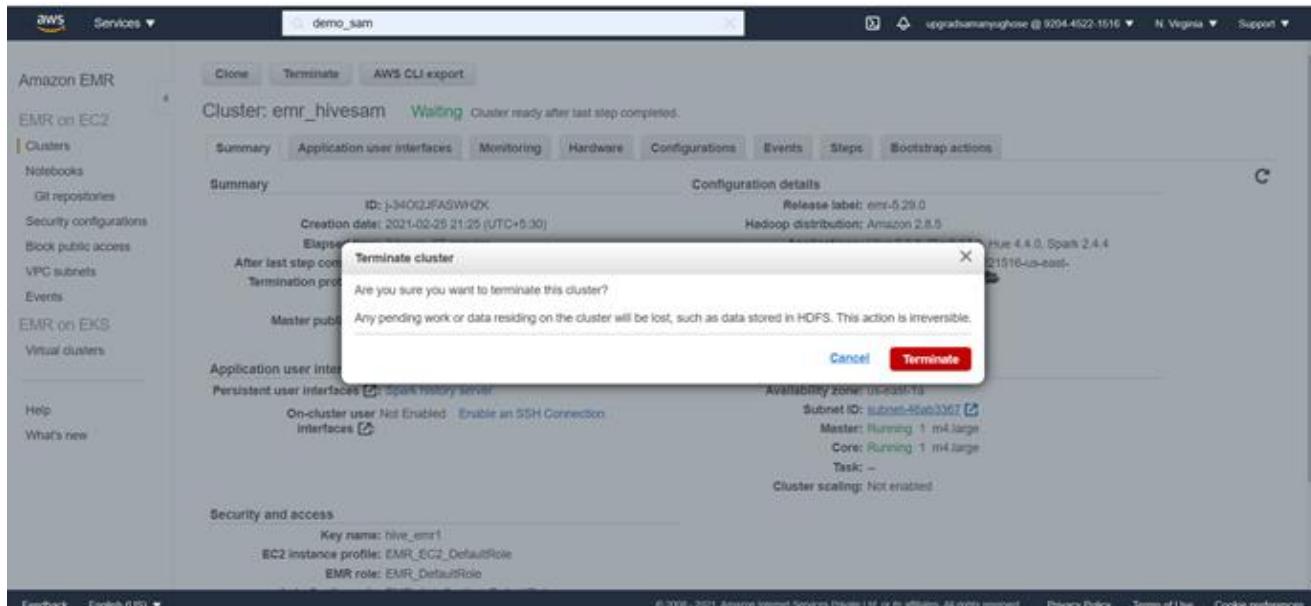


```

hadoop@ip-172-31-81-235:~$ hadoop fs -rm -r /hivestudy
Deleted /hivestudy
hadoop@ip-172-31-81-235:~$ hadoop fs -ls /
Found 4 items
drwxr-xr-x  - hdfs  hadoop   0 2021-02-25 16:01 /apps
drwxr-xr-x  - hdfs  hadoop   0 2021-02-25 16:03 /tmp
drwxr-xr-x  - hdfs  hadoop   0 2021-02-25 16:01 /user
drwxr-xr-x  - hdfs  hadoop   0 2021-02-25 16:01 /var
hadoop@ip-172-31-81-235:~$
```

4. Cluster Terminated

Once exited from the SSH terminal, terminated the EMR cluster which was created.



The screenshot shows the AWS EMR console with the cluster `emr_hivesam` in the `Waiting` state. A modal dialog box is open, asking for confirmation to terminate the cluster. The dialog states: "Are you sure you want to terminate this cluster? Any pending work or data residing on the cluster will be lost, such as data stored in HDFS. This action is irreversible." It includes a `Cancel` button and a prominent `Terminate` button.