HIVE CASE STUDY

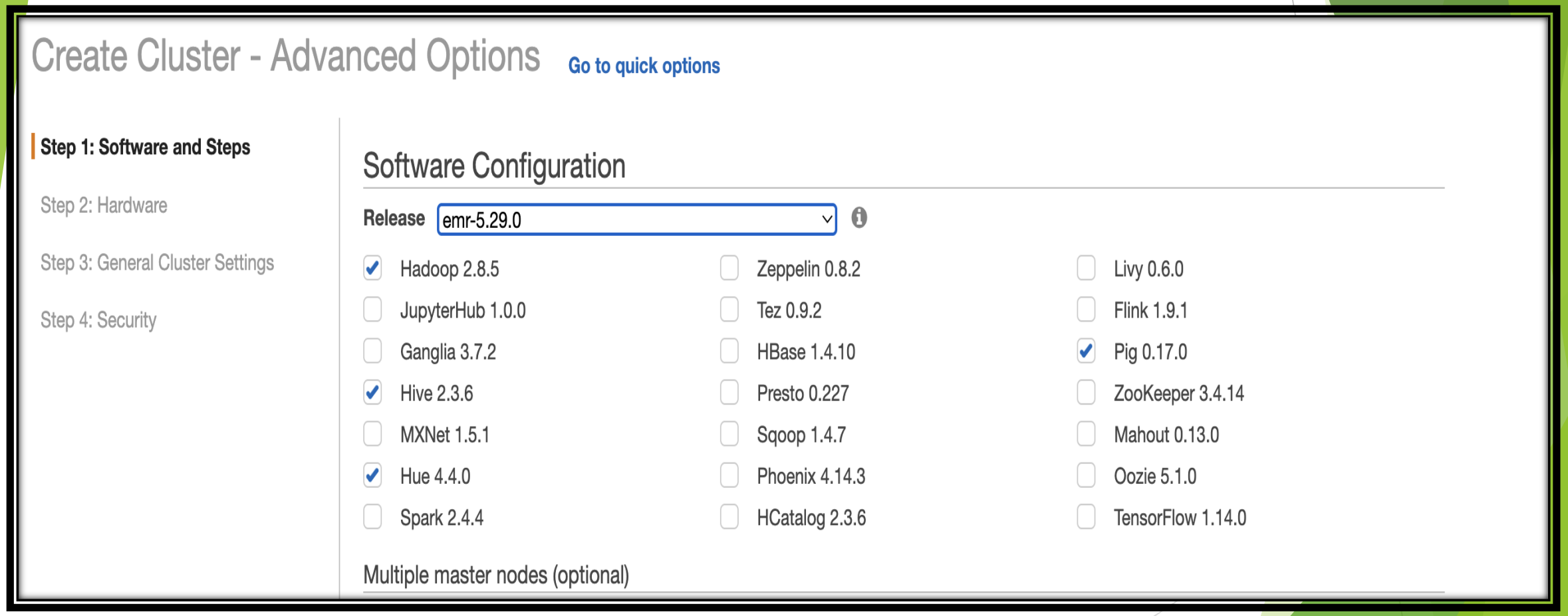
Created By: Sayeed Raheel
Debanjan Biswas

Batch : DS_B17_C37

- **Problem Synopsis:** 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 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.

- **Our Case Study Objective: To extract insights from a real time life data of an e commerce company , using AWS, EMR, S3 Hadoop and HIVE Systems.**

Creating EMR Cluster in AWS



Selecting Instance type for cluster Node

Cluster Nodes and Instances

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

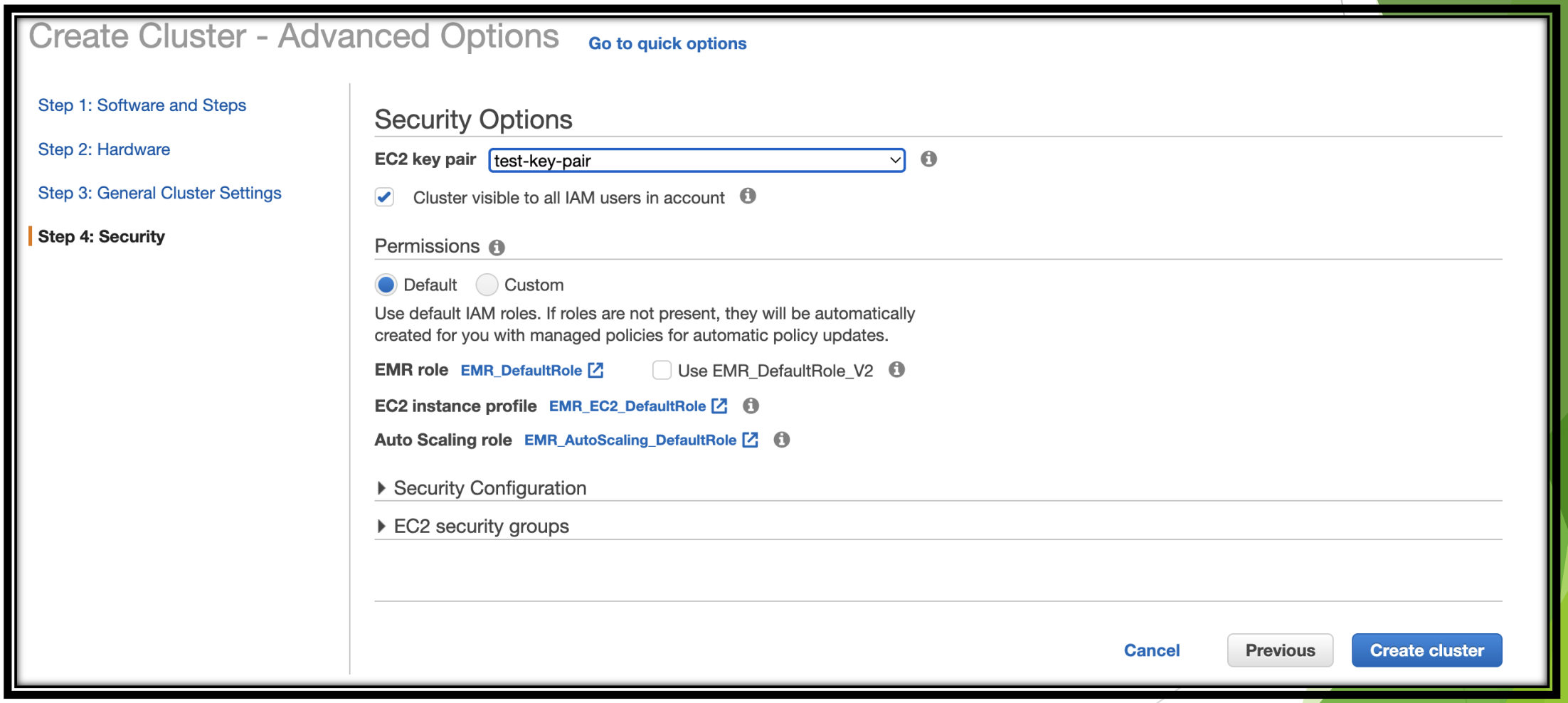
Console options for automatic scaling have changed. Learn more

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: 32 GiB Add configuration settings	2 Instances	**On-demand** **Spot** Use on-demand as max price
Task Task - 3	**m5.xlarge** 4 vCore, 16 GiB memory, EBS only storage EBS Storage: 32 GiB Add configuration settings	0 Instances	**On-demand** **Spot** Use on-demand as max price

+ Add task instance group

Total core and task units 2 Total units

Giving Security as Key Value Pair



Clustering Starting

Amazon EMR

EMR Studio

EMR Serverless New

EMR on EC2

Clusters

Notebooks

Git repositories

Security configurations

Block public access

VPC subnets

Events

EMR on EKS

Virtual clusters

Help

What's new

EMR Serverless is now GA.
With EMR Serverless, get the benefits of Amazon EMR such as open source compatibility, latest versions and performance optimized runtime for popular frameworks along with easy provisioning, quick job startup, automatic capacity management, and simple cost controls. Get Started with EMR Serverless.

Clone | Terminate | AWS CLI export

Auto-termination is not available for this account when using this release of EMR.

Cluster: My case study cluster Starting

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

Summary

ID: j-2ZVDXD0GAWVA8
Creation date: 2022-07-07 00:07 (UTC+5:30)
Elapsed time: 0 seconds
After last step completes: Cluster waits
Termination protection: Off Change
Tags: -- View All / Edit
Master public DNS: --

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
Log URI: s3://aws-logs-286749191220-us-east-1/elasticmapreduce/
EMRFS consistent view: Disabled
Custom AMI ID: --

Application user interfaces

Persistent user interfaces: --
On-cluster user interfaces: --

Network and hardware

Availability zone: --
Subnet ID: subnet-0302f64f

Connecting Cluster to System Via SSH

Amazon EMR

EMR Studio

EMR Serverless New

EMR on EC2

Clusters

Notebooks

Git repositories

Security configurations

Block public access

VPC subnets

Events

EMR on EKS

Virtual clusters

Help

What's new

EMR Serverless is now GA.
With EMR Serverless, get the benefits of Amazon EMR such as open source compatibility, latest versions and performance optimized runtime for popular frameworks along with easy provisioning, quick job startup, automatic capacity management, and simple cost controls. Get Started with EMR Serverless.

Clone | Terminate | AWS CLI export

Auto-termination is not available for this account when using this release of EMR.

Cluster: My case study cluster Starting

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

1. Open a terminal window. On Mac OS X, choose Applications > Utilities > Terminal. On other Linux distributions, terminal is typically found at Applications > Accessories > Terminal.
2. To establish a connection to the master node, type the following command. Replace ~/test-key-pair.pem with the location and filename of the private key file (.pem) used to launch the cluster.

```
ssh -i ~/test-key-pair.pem hadoop@ec2-23-22-157-255.compute-1.amazonaws.com
```

3. Type yes to dismiss the security warning.

Close

EMRFS consistent view: Disabled

Custom AMI ID: --

Application user interfaces

Persistent user interfaces: --

On-cluster user interfaces: Not Enabled Enable an SSH Connection

Network and hardware

Initiating EMR Cluster On System

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

https://aws.amazon.com/amazon-linux-ami/2018.03-release-notes/
68 package(s) needed for security, out of 97 available
Run "sudo yum update" to apply all updates.

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 MMMMMMM             MMMMMMM RRRRRRR      RRRRRR

[hadoop@ip-172-31-29-91 ~]$
```

Loading DATA In HDFS

```
[hadoop@ip-172-31-29-91 ~]$ wget https://e-commerce-events-ml.s3.amazonaws.com/2
019-Nov.csv
--2022-07-05 18:54:27--  https://e-commerce-events-ml.s3.amazonaws.com/2019-Nov.
csv
Resolving e-commerce-events-ml.s3.amazonaws.com (e-commerce-events-ml.s3.amazona
ws.com)... 52.217.161.145
Connecting to e-commerce-events-ml.s3.amazonaws.com (e-commerce-events-ml.s3.ama
zonaws.com)|52.217.161.145|:443... connected.
HTTP request sent, awaiting response... 200 OK
Length: 545839412 (521M) [text/csv]
Saving to: '2019-Nov.csv'

2019-Nov.csv        100%[===================>] 520.55M  51.9MB/s    in 9.3s

2022-07-05 18:54:36 (55.9 MB/s) - '2019-Nov.csv' saved [545839412/545839412]

[hadoop@ip-172-31-29-91 ~]$
```

We have successfully loaded both the files into HDFS

Creating anew directory and pushing data into the directory

```
2022-07-06 18:49:58 (61.5 MB/s) - '2019-Nov.csv' saved [545839412/545839412]

[hadoop@ip-172-31-22-128 ~]$ hadoop fs -mkdir /hivest
[hadoop@ip-172-31-22-128 ~]$ hadoop fs -put ./2019-Oct.csv /hivest
[hadoop@ip-172-31-22-128 ~]$ hadoop fs -put ./2019-Nov.csv /hivest
[hadoop@ip-172-31-22-128 ~]$
```

Checking for Data in new Directory

```
[hadoop@ip-172-31-29-91 ~]$ hadoop fs -ls /hivecase
Found 2 items
-rw-r--r--   1 hadoop hadoop  545839412 2022-07-05 18:57 /hivecase/2019-Nov.csv
-rw-r--r--   1 hadoop hadoop  482542278 2022-07-05 18:56 /hivecase/2019-Oct.csv
[hadoop@ip-172-31-29-91 ~]$
```

Creating Database Named Retail

```
hive> create database if not exists retail ;
OK
Time taken: 0.771 seconds
hive> use retail ;
OK
Time taken: 0.069 seconds
hive> set hive.cli.print.header=true;
```

Creating External Table and Pushing Data by giving data location in HDFS

```
hive> create External table if not exists retailsales(event_time timestamp,event_type string,product_id string,categor
y_id string,category_code string,brand string,price float, user_id bigint,user_session string) ROW FORMAT SERDE 'org.a
pache.hadoop.hive.serde2.OpenCSVSerde' WITH SERDEPROPERTIES ("separatorChar"=",","quoteChar"="\"","escapeChar"="\\")st
ored as textfile Location '/hivecase' TBLPROPERTIES("skip.header.line.count"="1") ;
OK
Time taken: 0.64 seconds
hive> select * from retailsales limit 5 ;
OK
retailsales.event_time  retailsales.event_type  retailsales.product_id  retailsales.category_id retailsales.category_c
ode     retailsales.brand       retailsales.price       retailsales.user_id     retailsales.user_session
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     22.22   556138645       57ed222e-a54a-
4907-9944-5a875c2d7f4f
2019-11-01 00:00:11 UTC cart    5876812 1487580010100293687             jessnail        3.16    564506666       186c19
51-8052-4b37-adce-dd9644b1d5f7
2019-11-01 00:00:24 UTC remove_from_cart        5826182 1487580007483048900                     3.33    553329724    2
067216c-31b5-455d-a1cc-af0575a34ffb
Time taken: 3.494 seconds, Fetched: 5 row(s)
hive>
```

Creating another external table and Dynamic portioning on event_type and bucketing on user_id

```
hive> create External table if not exists dynamic(event_time timestamp, product_id str
ing, category_id string, category_code string, brand string, price float, user_id bigi
nt, user_session string) partitioned by (event_type string) clustered by (user_id) int
o 9 buckets ROW FORMAT SERDE 'org.apache.hadoop.hive.serde2.OpenCSVSerde' stored as te
xtfile LOCATION '/hivecase' TBLPROPERTIES ("skip.header.line.count"="1");
OK
Time taken: 0.075 seconds
```

Inserting data in partitioned table from previous external table named retail sales

```
hive> set hive.exec.dynamic.partition=true;
hive> set hive.exec.dynamic.partition.mode=nonstrict;
hive> INSERT INTO TABLE dynamic partition(event_type) SELECT event_time, product_id, category_id, category_code, bran
d, price, user_id, user_session, event_type FROM retailsales;
Query ID = hadoop_20220706101331_d4524977-e824-4f6d-905d-6d6224ccfd67
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_1657099224332_0002)
```

Checking partition on event type and viewing data

```
hive> show partitions dynamic;
OK
partition
event_type=cart
event_type=purchase
event_type=remove_from_cart
event_type=view
Time taken: 0.135 seconds, Fetched: 4 row(s)
hive> set hive.cli.print.header=true;
hive> select * from dynamic limit 5;
OK
dynamic.event_time      dynamic.product_id      dynamic.category_id     dynamic.category_code   dynamic.brand   dynam
ic.price        dynamic.user_id dynamic.user_session    dynamic.event_type
2019-10-08 06:46:30 UTC 5781356 1638456119066100510             pole    4.11    510742055       504c931c-a1a6-4b5b-9f
6e-50e8d92fac53 cart
2019-10-08 20:51:54 UTC 5585186 1487580009362096156                     1.90    190707972       bd38b8b6-969c-460b-8d
23-2feee1ad25b6 cart
2019-10-08 06:46:27 UTC 5782028 1638456119066100510             pole    4.11    510742055       504c931c-a1a6-4b5b-9f
6e-50e8d92fac53 cart
2019-10-10 08:05:58 UTC 5772285 1602943681873052386             grattol 6.03    539363027       0bd14f42-7b88-4356-a0
d1-97ac28f25bc7 cart
2019-10-09 08:52:43 UTC 5782030 1638456119066100510             pole    4.11    555509990       14d946ba-ae78-4f80-80
2c-7bdb4b716a9a cart
Time taken: 0.247 seconds, Fetched: 5 row(s)
hive>
```

Question 1:Find the total revenue generated due to purchases made in October

```
hive> select sum(price) from dynamic where event_type='purchase' and month(event_time)=10 ;
Query ID = hadoop_20220706101939_5b1be9f6-cd58-4181-9a33-c5991f0cb297
Total jobs = 1
Launching Job 1 out of 1
Status: Running (Executing on YARN cluster with App id application_1657099224332_0002)

--------------------------------------------------------------------------------
        VERTICES      MODE        STATUS  TOTAL  COMPLETED  RUNNING  PENDING  FAILED  KILLED
--------------------------------------------------------------------------------
Map 1 .......... container     SUCCEEDED      3          3        0        0       0       0
Reducer 2 ...... container     SUCCEEDED      1          1        0        0       0       0
--------------------------------------------------------------------------------
VERTICES: 02/02  [==========================>>] 100%  ELAPSED TIME: 19.14 s
--------------------------------------------------------------------------------
OK
_c0
1211526.4499999236
Time taken: 20.521 seconds, Fetched: 1 row(s)
```

We see that the Total Revenue in the month of ‘October 2019’ is Rs.1211507.19 /-.

Question 2:To yield the total sum of purchases per month in a single output.

```
hive> select month(event_time) as Month , sum(price) as MONTHWISE_Revenue from dynamic where event_type='purchase' gr
oup by month(event_time);
Query ID = hadoop_20220706102704_b484e7fa-82ba-43e2-a82d-4ae4451a34be
Total jobs = 1
Launching Job 1 out of 1
Status: Running (Executing on YARN cluster with App id application_1657099224332_0002)

--------------------------------------------------------------------------------
        VERTICES      MODE        STATUS  TOTAL  COMPLETED  RUNNING  PENDING  FAILED  KILLED
--------------------------------------------------------------------------------
Map 1 .......... container     SUCCEEDED      3          3        0        0       0       0
Reducer 2 ...... container     SUCCEEDED      1          1        0        0       0       0
--------------------------------------------------------------------------------
VERTICES: 02/02  [==========================>>] 100%  ELAPSED TIME: 17.13 s
--------------------------------------------------------------------------------
OK
month   monthwise_revenue
10      1211526.4499999236
11      1530911.7999998932
Time taken: 17.819 seconds, Fetched: 2 row(s)
hive>
```

- We observe that the sum of purchases (Revenue) in the month of ‘November 2019’ is higher than that of ‘October 2019’.
- We can infer that the month of November has performed better than October.

Question 3: To find the change in revenue generated due to purchases from October to November

```
hive> with new as (SELECT SUM (case when date_format(event_time,'MM')=10 then price else 0 end) AS Oct_Rev, SUM (case when d]
ate_format(event_time,'MM')=11 then price else 0 end) AS Nov_Rev FROM dynamic WHERE event_type = 'purchase' AND date_format(
event_time,'MM') IN (10,11)) select round((Nov_Rev-Oct_Rev)) AS Change_in_Revenue from new;
Query ID = hadoop_20220706110133_4e6348a7-d357-471f-b512-ffde82215084
Total jobs = 1
Launching Job 1 out of 1
Status: Running (Executing on YARN cluster with App id application_1657099224332_0005)

--------------------------------------------------------------------------------
        VERTICES      MODE        STATUS  TOTAL  COMPLETED  RUNNING  PENDING  FAILED  KILLED
--------------------------------------------------------------------------------
Map 1 .......... container     SUCCEEDED      3          3        0        0       0       0
Reduce r 2 ...... container     SUCCEEDED      1          1        0        0       0       0
--------------------------------------------------------------------------------
VERTICES: 02/02  [==========================>>] 100%  ELAPSED TIME: 31.15 s
--------------------------------------------------------------------------------
OK
319385.0
Time taken: 31.91 seconds, Fetched: 1 row(s)
hive>
```

In continuation with our previous inference, we can see that Revenue generated in November is higher than October by Rs. 3,19,497.78 /-

Question 4:To Find distinct categories of products. Categories with null category code can be ignored.

```
hive> select distinct split(category_code,'\\.')[0] as Category_List from dynamic;
Query ID = hadoop_20220706113106_0d585a4b-37f6-42c4-8d0c-4b46c0823331
Total jobs = 1
Launching Job 1 out of 1
Status: Running (Executing on YARN cluster with App id application_1657099224332_0009)

--------------------------------------------------------------------------------
        VERTICES      MODE        STATUS  TOTAL  COMPLETED  RUNNING  PENDING  FAILED  KILLED
--------------------------------------------------------------------------------
Map 1 ........... container     SUCCEEDED     13         13        0        0       0       0
Reduce 2 ...... container     SUCCEEDED      5          5        0        0       0       0
--------------------------------------------------------------------------------
VERTICES: 02/02  [==========================>>] 100%  ELAPSED TIME: 57.16 s
--------------------------------------------------------------------------------
OK

furniture
appliances
accessories
apparel
sport
stationery
Time taken: 58.158 seconds, Fetched: 7 row(s)
hive>
```

We observe 6 different categories present namely:

- accessories • apparel
- appliances • furniture
- sport • stationery

The category_code column contained values, which were delimited by '.'. We use the SPLIT command to split and located the first index alone, which contained the main Category.

Question 5:To Find the total number of products available under each category.

```
hive> select split(category_code,'\\.')[0] as category, count(product_id) as Total_products from dynamic group by split(category_code,'\\.')[0]
] order by Total_products desc;
Query ID = hadoop_20220706114213_6f79ac8e-1cd4-4fcc-8161-e95add4cd61d
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_1657099224332_0010)

--------------------------------------------------------------------------------
        VERTICES      MODE        STATUS  TOTAL  COMPLETED  RUNNING  PENDING  FAILED  KILLED
--------------------------------------------------------------------------------
Map 1 .......... container     SUCCEEDED     16         16        0        0       0       0
Reducer 2 ...... container     SUCCEEDED      5          5        0        0       0       0
Reducer 3 ...... container     SUCCEEDED      1          1        0        0       0       0
--------------------------------------------------------------------------------
VERTICES: 03/03  [==========================>>] 100%  ELAPSED TIME: 57.99 s
--------------------------------------------------------------------------------
OK
        8594861
appliances      61735
stationery      26722
furniture       23603
apparel 18232
accessories     12929
sport   2
Time taken: 68.165 seconds, Fetched: 7 row(s)
hive>
```

- We observe that, ‘appliances’ category has the highest number of cosmetic products available under it.
- We can see ‘sports’ category has the least cosmetic products under it. This make sense as sports category would not contain many cosmetic products.

Question 6: Which brand had the maximum sales in October and November combined

```
hive> SELECT brand, ROUND(SUM(price),2) as Total_sales FROM dynamic WHERE event_type = 'purchase' GROUP BY brand
ORDER BY Total_sales desc LIMIT 5;
Query ID = hadoop_20220706123641_66b69f4b-863e-4395-a584-98f4dd6a8dc7
Total jobs = 1
Launching Job 1 out of 1
Status: Running (Executing on YARN cluster with App id application_1657099224332_0012)

Map 1: 0/3      Reducer 2: 0/1  Reducer 3: 0/1
Map 1: 0/3      Reducer 2: 0/1  Reducer 3: 0/1
Map 1: 0(+1)/3  Reducer 2: 0/1  Reducer 3: 0/1
Map 1: 0(+2)/3  Reducer 2: 0/1  Reducer 3: 0/1
Map 1: 0(+3)/3  Reducer 2: 0/1  Reducer 3: 0/1
Map 1: 0(+3)/3  Reducer 2: 0/1  Reducer 3: 0/1
Map 1: 1(+2)/3  Reducer 2: 0(+1)/1      Reducer 3: 0/1
Map 1: 1(+2)/3  Reducer 2: 0(+1)/1      Reducer 3: 0/1
Map 1: 2(+1)/3  Reducer 2: 0(+1)/1      Reducer 3: 0/1
Map 1: 3/3      Reducer 2: 0(+1)/1      Reducer 3: 0/1
Map 1: 3/3      Reducer 2: 1/1  Reducer 3: 0(+1)/1
Map 1: 3/3      Reducer 2: 1/1  Reducer 3: 1/1
OK
        1094174.64
runail  148297.94
grattol 106918.25
irisk   92538.0
uno     86341.78
Time taken: 16.74 seconds, Fetched: 5 row(s)
hive>
```

- We observe that, ‘runail’ brand is a popular brand with high sales in both months.
- We see that the brand ‘gruttol’ is not far behind with a difference of hardly Rs 40,000/

Question 7: Which brands increased their sales from October to November

```
hive> with ko as (select brand, sum(case when month(event_time)=10 then price else 0 end) as oct_rev, sum(case when month(event_time)=11 then price else 0 end) as nov_rev fro
dynamic group by brand) select brand, round(nov_rev-oct_rev,2) as Increased_Sales  from ko where (nov_rev-oct_rev)>0 order by Increased_Sales desc;
Query ID = hadoop_20220706130801_acb8f474-cf01-4e7b-8def-2af988904c7a
Total jobs = 1
Launching Job 1 out of 1
Status: Running (Executing on YARN cluster with App id application_1657099224332_0015)

Map 1: 0/16     Reducer 2: 0/5  Reducer 3: 0/1
Map 1: 0/16     Reducer 2: 0/5  Reducer 3: 0/1
Map 1: 0/16     Reducer 2: 0/5  Reducer 3: 0/1
Map 1: 0(+1)/16 Reducer 2: 0/5  Reducer 3: 0/1
Map 1: 0(+2)/16 Reducer 2: 0/5  Reducer 3: 0/1
Map 1: 0(+3)/16 Reducer 2: 0/5  Reducer 3: 0/1
Map 1: 0(+4)/16 Reducer 2: 0/5  Reducer 3: 0/1
Map 1: 0(+7)/16 Reducer 2: 0/5  Reducer 3: 0/1
Map 1: 0(+7)/16 Reducer 2: 0/5  Reducer 3: 0/1
Map 1: 0(+7)/16 Reducer 2: 0/5  Reducer 3: 0/1
Map 1: 0(+7)/16 Reducer 2: 0/5  Reducer 3: 0/1
Map 1: 0(+7)/16 Reducer 2: 0/5  Reducer 3: 0/1
```

```
grattol 353039.3
jessnail        217137.38
strong  163467.68
lianail 144334.35
marathon        124160.6
polarus 121377.82
uno     113158.09
jas     80599.37
cnd     70410.84
shik    67108.72
ingarden        65672.45
max     49760.5
freedecor       46617.77
cosmoprofi      42547.34
staleks 33801.45
s.care  27526.71
lovely  26082.14
yoko    25271.18
missha  24733.7
benovy  23382.81
runail  23190.81
haruyama        22547.79
coifin  21890.04
naomi   21459.03
emil    20090.87
sanoto  19173.29
artex   16309.34
ecolab  14167.58
beauty-free     12569.43
vosev   11891.32
bpw.style       11057.38
milv    10712.36
italwax 10355.34
matreshka       10283.63
beautix 9201.21
browxenna       9124.02
concept 8904.98
swarovski       7882.67
nagaraku        7820.14
ecocraft        7659.64
f.o.x   7636.27
metzger 7437.95
yu-r    6686.4
roubloff        6586.18
severina        6440.59
```

```
severina        6440.59
art-visage      6124.18
de.lux  5762.35
levissime       5358.22
sophin  5126.78
markell 4835.99
limoni  4764.48
kapous  4732.63
candy   4725.73
zeitun  4582.52
refectocil      3748.23
gehwol  3489.1
beauugreen      2667.5
dewal   2028.91
lowence 2022.1
smart   1980.05
airnails        1898.01
biofollica      1891.11
lador   1873.98
nitrile 1821.87
balbcare        1736.41
koreatida       1727.94
entity  1630.59
batiste 1595.1
koelf   1590.22
fedua   1573.24
cristalinas     1498.9
dizao   1471.84
tertio  1279.87
shary   1212.99
finish  1082.49
happyfons       1046.63
ellips  1024.14
kosmekka        970.76
greymy  934.12
carmex  896.22
eos     762.27
jaguar  559.83
australis       557.98
mane    504.22
glysolid        498.36
plazan  445.58
kiss    426.72
levrana 400.89
```

```
levrana 400.89
opi     397.92
elskin  319.68
aura    313.7
igrobeauty      274.82
consly  203.54
fly     172.67
vilenta 137.04
ikoo    113.33
frozen  91.14
busch   82.52
kamill  81.74
barbie  68.19
skinlite        67.84
laiseven        58.24
enas    58.11
philips 38.73
inm     32.43
shifei  14.05
invisibobble    9.2
ibd     7.76
tazol   5.39
Time taken: 70.862 seconds, Fetched: 110 row(s)
hive>
```

- We observe that, ‘gruttol’ brand has seen the highest per month increase of Rs 36027.17/- . In the previous query we observed that it was a brand with second highest total sales.
- We see ‘runail’ brand is in the 9th position.
- The brand with lowest per month increase of ‘0.56’ is ‘ovale’; followed by ‘cosima’ with a ‘0.7’ difference in monthly sales.

Question 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.

```
hive> select user_id, sum(price) as amount_spend from dynamic where event_type='purchase' group by user_id order by amount_spend desc limit 10;
Query ID = hadoop_20220706132011_ccc5bb3f-0463-419c-a392-5c2a707bbbfe
Total jobs = 1
Launching Job 1 out of 1
Status: Running (Executing on YARN cluster with App id application_1657099224332_0016)

--------------------------------------------------------------------------------
        VERTICES      MODE        STATUS  TOTAL  COMPLETED  RUNNING  PENDING  FAILED  KILLED
--------------------------------------------------------------------------------
Map 1 .......... container     SUCCEEDED      3          3        0        0       0       0
Reduce 2 ...... container     SUCCEEDED      1          1        0        0       0       0
Reduce 3 ...... container     SUCCEEDED      1          1        0        0       0       0
--------------------------------------------------------------------------------
VERTICES: 03/03  [==========================>>] 100%  ELAPSED TIME: 17.28 s
--------------------------------------------------------------------------------
OK
user_id amount_spend
557790271       2715.869999999991
150318419       1645.9699999999998
562167663       1352.8500000000001
531900924       1329.4500000000003
557850743       1295.4800000000005
522130011       1185.3900000000003
561592095       1109.7000000000003
431950134       1097.5899999999992
566576008       1056.36
521347209       1040.91
Time taken: 18.11 seconds, Fetched: 10 row(s)
hive>
```

user_id	Total_money_spent
557790271	2715.87
150318419	1645.97
562167663	1352.85
531900924	1329.45
557850743	1295.48
522130011	1185.39
561592095	1109.7
431950134	1097.59
566576008	1056.36
521347209	1040.91

Insights From The case Study

- We see change in result obtained through both ways as the data used is the same
- We observed the query execution time has increased significantly by dynamic portioning on event_type and bucketing on user_id.
- We can also conclude that partioning and bucketing are essential to reduce query execution time.