

**PG DIPLOMA IN DATA SCIENCE**

**NOVEMBER 2020**

**BATCH - C26**

**BUSINESS INTELLIGENCE / DATA ANALYTICS**

**HIVE CASE STUDY**

**Submitted by**

**Shivanshi Tiwari**

## Problem statement

For this assignment, you will be working with a public clickstream dataset of a cosmetics store. Using this dataset, your job is to extract valuable insights which generally data engineers come up with within an e-retail company.

## Summary of EMR Cluster used:

The screenshot displays the AWS EMR console interface for a cluster named 'j-2H7FMQ3AMECGR'. The 'Summary' tab is selected, showing the cluster's lifecycle: it was created on 2021-08-15 at 18:05 (UTC+5:30) and ended at 18:33 (UTC+5:30), having run for 28 minutes. The cluster is currently in a 'waiting' state after the last step completed. Termination protection is off, and there are no tags. The master public DNS is 'ec2-100-26-236-108.compute-1.amazonaws.com'. The 'Configuration details' tab shows the release label 'emr-5.29.0', Hadoop distribution 'Amazon 2.8.5', and applications including Hive 2.3.6, Pig 0.17.0, and Hue 4.4.0. The log URI is 's3://aws-logs-184017499986-us-east-1/elasticmapreduce/'. EMRFS consistent view is disabled, and the custom AMI ID is empty. The 'Application user interfaces' section shows no persistent or on-cluster interfaces. The 'Network and hardware' section shows the cluster is in the 'us-east-1e' availability zone, using subnet 'subnet-0a22443b', with one terminated master node and one terminated core node, both of type 'm4.large'. Cluster scaling is not enabled. The 'Security and access' section shows the key name 'hive-case-study', EC2 instance profile 'EMR\_EC2\_DefaultRole', EMR role 'EMR\_DefaultRole', and Auto Scaling role 'EMR\_AutoScaling\_DefaultRole'. It is visible to all users. Security groups for the master node are 'sg-0f0054a6a11509cc6' (ElasticMapReduce-master) and for the core & task nodes are 'sg-03d5983da745c0e81' (ElasticMapReduce-slave).

Summary	Configuration details
<b>ID:</b> j-2H7FMQ3AMECGR	<b>Release label:</b> emr-5.29.0
<b>Creation date:</b> 2021-08-15 18:05 (UTC+5:30)	<b>Hadoop distribution:</b> Amazon 2.8.5
<b>End date:</b> 2021-08-15 18:33 (UTC+5:30)	<b>Applications:</b> Hive 2.3.6, Pig 0.17.0, Hue 4.4.0
<b>Elapsed time:</b> 28 minutes	<b>Log URI:</b> s3://aws-logs-184017499986-us-east-1/elasticmapreduce/
<b>After last step completes:</b> Cluster waits	<b>EMRFS consistent view:</b> Disabled
<b>Termination protection:</b> Off	<b>Custom AMI ID:</b> --
<b>Tags:</b> --	
<b>Master public DNS:</b> ec2-100-26-236-108.compute-1.amazonaws.com	
Connect to the Master Node Using SSH	

Application user interfaces	Network and hardware
<b>Persistent user interfaces:</b> --	<b>Availability zone:</b> us-east-1e
<b>On-cluster user interfaces:</b> --	<b>Subnet ID:</b> subnet-0a22443b
	<b>Master:</b> Terminated 1 m4.large
	<b>Core:</b> Terminated 1 m4.large
	<b>Task:</b> --
	<b>Cluster scaling:</b> Not enabled

Security and access
<b>Key name:</b> hive-case-study
<b>EC2 instance profile:</b> EMR_EC2_DefaultRole
<b>EMR role:</b> EMR_DefaultRole
<b>Auto Scaling role:</b> EMR_AutoScaling_DefaultRole
<b>Visible to all users:</b> All
<b>Security groups for Master:</b> sg-0f0054a6a11509cc6 (ElasticMapReduce-master)
<b>Security groups for Core &amp; Task:</b> sg-03d5983da745c0e81 (ElasticMapReduce-slave)

## Creating necessary directory inside HDFS

The following codes and screenshots show the code used and the output that was displayed inside the PuTTY environment.

***hadoop fs -ls /user/hive***

This command was used to look at current folders or directories that were present in the HDFS.

```
hadoop fs -mkdir /user/hive/e-commerce
```

Once it was seen that only a default folder was present, a new directory was created with the name 'e-commerce' for storing the files that will later be uploaded from the public S3 bucket.

```
hadoop fs -ls /user/hive
```

After creating a new folder, a sanity check was performed to ensure that the folder existed in the HDFS.

```

[~] hadoop@ip-172-31-50-9:~
Using username "hadoop".
Authenticating with public key "imported-openssh-key"
Last login: Sun Aug 15 06:39:55 2021

  _ _ _ _ _
 _ _ _ _ _ /
 _ _ _ _ _

Amazon Linux AMI

https://aws.amazon.com/amazon-linux-ami/2018.03-release-notes/
68 package(s) needed for security, out of 107 available
Run "sudo yum update" to apply all updates.

EEEEEEEEEEEEEEEEEEEE MMMMMMMM MMMMMMMM RRRRRRRRRRRRRRRR
E::::::::::::::::::::E M::::::::M M::::::::M R::::::::::::R
EE::::::::EEEEEEEE::::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::::::::EEEEEEEE M::::M M::::M M::::M R::::RRRRRR::::R
E::::::::::::E M::::M M::::M M::::M R::::::::::::RR
E::::::::EEEEEEEE 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 MMMMMMMM MMMMMMMM RRRRRRR RRRRRR

[hadoop@ip-172-31-50-9 ~]$ hadoop fs -ls /user/hive
Found 1 items
drwxrwxrwt - hdfs hadoop 0 2021-08-15 06:35 /user/hive/warehouse
[hadoop@ip-172-31-50-9 ~]$ hadoop fs -mkdir /user/hive
mkdir: '/user/hive': File exists
[hadoop@ip-172-31-50-9 ~]$ hadoop fs -mkdir /user/hive/e-commerce
[hadoop@ip-172-31-50-9 ~]$ hadoop fs -ls /user/hive
Found 2 items
drwxr-xr-x - hadoop hadoop 0 2021-08-15 06:42 /user/hive/e-commerce
drwxrwxrwt - hdfs hadoop 0 2021-08-15 06:35 /user/hive/warehouse
[hadoop@ip-172-31-50-9 ~]$

```

## Importing data from public S3 to HDFS

```
hadoop distcp 's3://e-commerce-events-ml/2019-Oct.csv'
'/user/hive/e-commerce/'
```

Now that the folder was created, the '2019-Oct.csv' file was imported from the public S3 bucket into the HDFS using the above piece of code.

```
[hadoop@ip-172-31-50-9 ~]$ hadoop distcp 's3://e-commerce-events-ml/2019-Oct.csv' '/user/hive/e-commerce/'
21/08/15 06:43:26 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, sslConfigurationFile='null', copyStrategy='uniformsize', preserveStatus=[], preserveRawXattrs=false, atomicWorkPath=null, logPath=null, sourceFileListing=null, sourcePaths=[s3://e-commerce-events-ml/2019-Oct.csv], targetPath=/user/hive/e-commerce, targetPathExists=true, filtersFile='null'}
```

***hadoop distcp 's3://e-commerce-events-ml/2019-Nov.csv' '/user/hive/e-commerce/'***

After the first file was imported, the '2019-Nov.csv' file was imported from the public S3 bucket into the HDFS using the above piece of code.

```
[hadoop@ip-172-31-50-9 ~]$ hadoop distcp 's3://e-commerce-events-ml/2019-Nov.csv' '/user/hive/e-commerce/'
21/08/15 06:44:30 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, sslConfigurationFile='null', copyStrategy='uniformsize', preserveStatus=[], preserveRawXattrs=false, atomicWorkPath=null, logPath=null, sourceFileListing=null, sourcePaths=[s3://e-commerce-events-ml/2019-Nov.csv], targetPath=/user/hive/e-commerce, targetPathExists=true, filtersFile='null'}
```

***hadoop fs -ls /user/hive/e-commerce***

After hadoop had finished running to import the files, the above code was run to ensure the two files have been successfully imported.

```
[hadoop@ip-172-31-50-9 ~]$ hadoop fs -ls /user/hive/e-commerce
Found 2 items
-rw-r--r-- 1 hadoop hadoop 545839412 2021-08-15 06:45 /user/hive/e-commerce/2019-Nov.csv
-rw-r--r-- 1 hadoop hadoop 482542278 2021-08-15 06:43 /user/hive/e-commerce/2019-Oct.csv
[hadoop@ip-172-31-50-9 ~]$
```

# Hive

## Creating a database

A new database was created for the purpose of performing analysis on the given dataset.

```
create database case_study ;  
use case_study ;
```

Output:

```
[hadoop@ip-172-31-50-9 ~]$ hive  
Logging initialized using configuration in file:/etc/hive/conf.dist/hive-log4j2.properties Async: false  
hive> create database case_study;  
OK  
Time taken: 0.787 seconds  
hive> use case_study;  
OK  
Time taken: 0.05 seconds  
hive> set hive.cli.print.header=true;  
hive> █
```

To display headers for the columns

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

Creating the base table with all the imported data

Code used:

```
create external table if not exists clickstream(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'  
stored as textfile  
location '/user/hive/e-commerce/'  
tblproperties("skip.header.line.count"="1");
```

The above code was used to create an external table along with the required schema and including the default properties of CSVSerde, the location of where the data is and a property value to skip the headers that exist in the dataset.

It loads data from both the csv files into the same table, thus allowing us to not have to use any joins when querying.

## Output:

```
hive> create external table if not exists clickstream(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'
> stored as textfile
> location '/user/hive/e-commerce/'
> tblproperties("skip.header.line.count"="1");
OK
Time taken: 0.373 seconds
hive> █
```

## Checking for correct schema in the table:

Once the data is loaded, a small query is run to look at the top 5 rows in the table that was created to validate the presence of the correct schema.

```
hive> SELECT *
> FROM clickstream
> LIMIT 5;
OK
clickstream.event_time clickstream.event_type clickstream.product_id clickstream.category_id clickstream.category_
code clickstream.brand clickstream.price clickstream.user_id clickstream.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-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 5876812 1487580010100293687 jessnail 3.16 564506666 186c1
951-8052-4b37-adce-dd9644b1d5f7
2019-11-01 00:00:24 UTC remove_from_cart 5826182 1487580007483048900 3.33 553329724 2
067216c-31b5-455d-alcc-af0575a34ffb
Time taken: 2.395 seconds, Fetched: 5 row(s)
hive> █
```

## Code used for measuring query runtime:

The following code was used to measure the time taken to run queries using the existing table.

```
SELECT round(sum(price),2) AS oct_total_revenue
FROM clickstream
WHERE date_format(event_time,'MM')=10 AND event_type='purchase';
```

Output:

```
hive> SELECT round(sum(price),2) AS oct_total_revenue
> FROM clickstream
> WHERE date_format(event_time,'MM')=10 AND event_type='purchase';
Query ID = hadoop_20210815065903_12d30072-85ba-49f9-93e9-ae04fdd29a1
Total jobs = 1
Launching Job 1 out of 1
Status: Running (Executing on YARN cluster with App id application_1629009426685_0003)

-----
VERTICES      MODE      STATUS  TOTAL  COMPLETED  RUNNING  PENDING  FAILED  KILLED
-----
Map 1 ..... container  SUCCEEDED    2          2          0          0          0          0
Reducer 2 ..... container  SUCCEEDED    1          1          0          0          0          0
-----
VERTICES: 02/02  [=====>>] 100%  ELAPSED TIME: 120.97 s
-----
OK
oct_total_revenue
1211538.43
Time taken: 122.338 seconds, Fetched: 1 row(s)
hive> █
```

Time taken: 122.338 seconds

## Optimizing the base table

Now that it is seen that it takes nearly 2 minutes to run a simple query, it becomes important to perform optimization techniques so that query times reduce significantly.

The following properties are enabled to allow us to perform necessary optimization techniques.

```
hive> set hive.exec.dynamic.partition.mode=nonrestrict;
hive> set hive.exec.dynamic.partition=true;
hive> set hive.enforce.bucketing=true;
hive> █
```

a. **Partition key** - event\_type, **bucketing key** - user\_id

In the first method, event\_type has been chosen as the partition key as it would be the most frequently accessed column based on the queries that need to be performed. It also has a very low cardinality of only 4 unique values.

The key for bucketing is chosen to be `user_id` as it has a high cardinality and would therefore be ideal to be used here. 16 buckets are chosen based on the large size of the data.

#### Code used:

```
create table if not exists clickstream_bucket(event_time timestamp,
product_id string, category_id string, category_code string, brand
string, price float, user_id bigint, user_session string)
partitioned by (event_type string) clustered by (user_id) into 16 buckets
row format serde 'org.apache.hadoop.hive.serde2.OpenCSVSerde'
stored as textfile;
```

#### Output:

```
hive>
> set hive.exec.dynamic.partition.mode=nonrestrict;
hive> set hive.exec.dynamic.partition=true;
hive> set hive.enforce.bucketing=true;
hive> create table if not exists clickstream_bucket(event_time timestamp, product_id string, category_id string, cate
gory_code string, brand string, price float, user_id bigint, user_session string)
> partitioned by (event_type string) clustered by (user_id) into 16 buckets
> row format serde 'org.apache.hadoop.hive.serde2.OpenCSVSerde'
> stored as textfile;
OK
Time taken: 0.146 seconds
```

#### Inserting data into the optimized table:

Once an optimized table is created, data is directly fed into it from the base table that was created earlier.

```
insert into table clickstream_bucket partition(event_type) select
event_time, product_id, category_id, category_code, brand, price,
user_id, user_session, event_type
from clickstream;
```



## Output:

```
hive> insert into table clickstream_bucket partition(event_type) select event_time, product_id, category_id, category
_code, brand, price, user_id, user_session, event_type
> from clickstream;
Query ID = hadoop_20210814140500_30738669-8513-43d8-b760-1fal3fe6c84f
Total jobs = 1
Launching Job 1 out of 1
Status: Running (Executing on YARN cluster with App id application_1628949322050_0003)

-----
VERTICES      MODE        STATUS  TOTAL  COMPLETED  RUNNING  PENDING  FAILED  KILLED
-----
Map 1 ..... container  SUCCEEDED    2         2         0         0         0         0
Reducer 2 ..... container  SUCCEEDED   16        16         0         0         0         0
-----
VERTICES: 02/02  [=====>>] 100% ELAPSED TIME: 133.33 s
-----
Loading data to table case_study.clickstream_bucket partition (event_type=null)

Loaded : 4/4 partitions.
Time taken to load dynamic partitions: 0.878 seconds
Time taken for adding to write entity : 0.004 seconds
OK
event_time      product_id      category_id      category_code    brand    price    user_id user_session    event_type
Time taken: 137.296 seconds
```

## Checking for correct schema in the table:

Once the data is loaded, a small query is run to look at the top 5 rows in the table that was created to validate the presence of the correct schema.

```
hive> select *
> from clickstream_bucket
> limit 5;
OK
clickstream_bucket.event_time  clickstream_bucket.product_id  clickstream_bucket.category_id  clickstream_bucket.ca
tegory_code    clickstream_bucket.brand    clickstream_bucket.price    clickstream_bucket.user_id    click
stream_bucket.user_session    clickstream_bucket.event_type
2019-10-31 23:26:17 UTC 5760334 1487580009286598681          zinger  0.44    565700388    c34d0241-ae83-40ad-b8
2f-5b4d991e90d1 cart
2019-10-31 23:26:03 UTC 5687743 1487580007852147670          zinger  8.57    565700388    c34d0241-ae83-40ad-b8
2f-5b4d991e90d1 cart
2019-10-31 23:25:39 UTC 5675552 1487580007936033754          zinger  4.21    565700388    c34d0241-ae83-40ad-b8
2f-5b4d991e90d1 cart
2019-10-31 23:21:55 UTC 5022    1487580007952810971          runail  4.84    565700388    c34d0241-ae83-40ad-b8
2f-5b4d991e90d1 cart
2019-10-31 23:21:46 UTC 5824810 1487580007936033754          domix   0.79    565700388    c34d0241-ae83-40ad-b8
2f-5b4d991e90d1 cart
Time taken: 0.25 seconds, Fetched: 5 row(s)
```

## Code used for measuring query runtime:

The following code was used to measure the time taken to run queries using the existing table.

```
SELECT round(sum(price),2) AS oct_total_revenue
FROM clickstream_bucket
WHERE date_format(event_time,'MM')=10 AND event_type='purchase';
```

Output:

```
hive> select round(sum(price),2) as total_revenue
> from clickstream_bucket
> where date_format(event_time,'MM')=10 and event_type='purchase';
Query ID = hadoop_20210814140808_ddeac9cc-a39b-4935-aeba-529cc53c5e03
Total jobs = 1
Launching Job 1 out of 1
Status: Running (Executing on YARN cluster with App id application_1628949322050_0003)

-----
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: 28.05 s
-----
OK
total_revenue
1211538.43
Time taken: 29.531 seconds, Fetched: 1 row(s)
```

Time taken: 29.531 seconds

b. **Partition key** - event\_type, **bucketing key** - product\_id

In the second method, event\_type has been chosen as the partition key as it would be the most frequently accessed column based on the queries that need to be performed. It also has a very low cardinality of only 4 unique values.

The key for bucketing is chosen to be product\_id as it has a high cardinality and would therefore be ideal to be used here. 16 buckets are chosen based on the large size of the data.

Code used:

```
create table if not exists clickstream_bucket(event_time timestamp,
product_id string, category_id string, category_code string, brand
string, price float, user_id bigint, user_session string)
partitioned by (event_type string) clustered by (product_id) into 16
buckets
row format serde 'org.apache.hadoop.hive.serde2.OpenCSVSerde'
stored as textfile;
```

Output:

```
hive> create table if not exists clickstream_bucket(event_time timestamp, product_id string, category_id string, category_code string, brand string, price float, user_id bigint, user_session string)
> partitioned by (event_type string) clustered by (product_id) into 16 buckets
> row format serde 'org.apache.hadoop.hive.serde2.OpenCSVSerde'
> stored as textfile;
OK
Time taken: 0.099 seconds
hive>
```

Inserting data into the optimized table:

```
insert into table clickstream_bucket partition(event_type) select
event_time, product_id, category_id, category_code, brand, price,
user_id, user_session, event_type
from clickstream;
```

Output:

```
hive> insert into table clickstream_bucket partition(event_type) select event_time, product_id, category_id, category_code, brand, price, user_id, user_session, event_type
> from clickstream;
Query ID = hadoop_20210815065454_95ffde6a-bb13-4962-8902-82219c1f9972
Total jobs = 1
Launching Job 1 out of 1
Status: Running (Executing on YARN cluster with App id application_1629009426685_0003)

-----
VERTICES      MODE        STATUS  TOTAL  COMPLETED  RUNNING  PENDING  FAILED  KILLED
-----
Map 1 ..... container    SUCCEEDED    2         2         0         0         0         0
Reducer 2 ..... container    SUCCEEDED   16        16         0         0         0         0
-----
VERTICES: 02/02 [=====>>>] 100% ELAPSED TIME: 132.81 s
-----
Loading data to table case_study.clickstream_bucket partition (event_type=null)

Loaded : 4/4 partitions.
Time taken to load dynamic partitions: 1.244 seconds
Time taken for adding to write entity : 0.005 seconds
OK
event_time      product_id      category_id      category_code    brand    price    user_id user_session    event_type
Time taken: 136.39 seconds
hive>
```

Checking for correct schema in the table

Once an optimized table is created, data is directly fed into it from the base table that was created earlier.

```
hive> SELECT *
> FROM clickstream_bucket
> LIMIT 5;
OK
clickstream_bucket.event_time  clickstream_bucket.product_id  clickstream_bucket.category_id  clickstream_bucket.category_code  clickstream_bucket.brand  clickstream_bucket.price  clickstream_bucket.user_id  clickstream_bucket.user_session  clickstream_bucket.event_type
2019-10-31 23:57:05 UTC 5850570 1998040852064109417 6.35 566272508 8f8a6160-24b9-47b3-88ld-75798a7d45ad cart
2019-10-31 23:56:54 UTC 4653 1487580011157258342 runail 0.37 562691482 9025c3a5-9c56-49c4-9d3d-95a6c15b69a3 cart
2019-10-31 23:52:22 UTC 5877597 1487580007675986893 bpw.style 0.79 562691482 9025c3a5-9c56-49c4-9d3d-95a6c15b69a3 cart
2019-10-31 23:47:14 UTC 5868467 1487580009445982239 0.95 566272508 8f8a6160-24b9-47b3-88ld-75798a7d45ad cart
2019-10-31 23:47:04 UTC 5735268 1487580005268456287 haruyama 3.97 232273004 dlac5983-a839-4938-9054-357f3ef7de62 cart
Time taken: 0.23 seconds, Fetched: 5 row(s)
hive>
```

### Code used for measuring query runtime:

The following code was used to measure the time taken to run queries using the existing table.

```
SELECT round(sum(price),2) AS oct_total_revenue
FROM clickstream_bucket
WHERE date_format(event_time,'MM')=10 AND event_type='purchase';
```

### Output:

```
hive> SELECT round(sum(price),2) AS oct_total_revenue
> FROM clickstream_bucket
> WHERE date_format(event_time,'MM')=10 AND event_type='purchase';
Query ID = hadoop_20210815072232_262d02e8-429a-4df6-961f-0e6857d615cd
Total jobs = 1
Launching Job 1 out of 1
Status: Running (Executing on YARN cluster with App id application_1629009426685_0005)

-----
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: 27.71 s
-----
OK
oct_total_revenue
1211538.43
Time taken: 28.462 seconds, Fetched: 1 row(s)
hive> █
```

**Time taken:** 28.462 seconds

### Performance comparison:

Two different sets of keys were used for optimization of the base table and the same query was used on all 3 tables to measure the time taken. The least time taken would be the most optimized and would be ideal for running the rest of the queries.

The time taken by the different tables are shown below:

**Base table :** 122.338 seconds

**Optimized table:**

- a. Partitioning key - event\_type & Bucketing key - user\_id :  
29.531 seconds

**b. Partitioning key - event\_type & Bucketing key - product\_id :**  
28.462 seconds

It can be seen that the table with the partitioning key as event\_type and bucketing key as product\_id is the most optimized of the lot.

**Improvement:**  $28.462 / 122.338 = 0.23$  i.e. the optimized table has taken only **23% of the time taken** by the base table.

For all the queries below, **event\_type is taken to be 'purchase'** since it is the only event where a monetary transaction takes place and hence would contribute to the overall revenue.

### Queries asked:

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

Code:

```
SELECT round(sum(price),2) AS oct_total_revenue
FROM clickstream_bucket
WHERE date_format(event_time,'MM')=10 AND event_type='purchase';
```

Output:

```
hive> SELECT round(sum(price),2) AS oct_total_revenue
> FROM clickstream_bucket
> WHERE date_format(event_time,'MM')=10 AND event_type='purchase';
Query ID = hadoop_20210815072232_262d02e8-429a-4df6-961f-0e6857d615cd
Total jobs = 1
Launching Job 1 out of 1
Status: Running (Executing on YARN cluster with App id application_1629009426685_0005)

-----
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: 27.71 s
-----
OK
oct_total_revenue
1211538.43
Time taken: 28.462 seconds, Fetched: 1 row(s)
hive> █
```

**Time taken:** 28.462 seconds



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

Code:

```
SELECT date_format(event_time,'MM') AS month, round(sum(price),2) AS
total_purchase
FROM clickstream
WHERE event_type = 'purchase'
GROUP BY date_format(event_time,'MM');
```

Output:

```
hive> SELECT date_format(event_time,'MM') AS month, round(sum(price),2) AS total_purchase
> FROM clickstream
> WHERE event_type = 'purchase'
> GROUP BY date_format(event_time,'MM');
Query ID = hadoop_20210815070221_651e7afc-dfc5-47cf-a0d9-1d21be45103c
Total jobs = 1
Launching Job 1 out of 1
Status: Running (Executing on YARN cluster with App id application_1629009426685_0003)

-----
VERTICES      MODE      STATUS  TOTAL  COMPLETED  RUNNING  PENDING  FAILED  KILLED
-----
Map 1 ..... container    SUCCEEDED    2          2          0          0          0          0
Reducer 2 ..... container    SUCCEEDED    3          3          0          0          0          0
-----
VERTICES: 02/02  [=====>>>] 100%  ELAPSED TIME: 59.54 s
-----
OK
month  total_purchase
10      1211538.43
11      1531016.9
Time taken: 60.147 seconds, Fetched: 2 row(s)
```

Time taken: 60.147 seconds

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

Code:

```
SELECT round(sum(case when date_format(event_time,'MM')=11 then price
else 0 end) - sum(case when date_format(event_time,'MM')=10 then price
else 0 end),2) AS diff_in_sales
FROM clickstream_bucket
WHERE event_type='purchase';
```

### Output:

```
hive> SELECT round(sum(case when date_format(event_time,'MM')=11 then price else 0 end) - sum(case when date_format(e
vent_time,'MM')=10 then price else 0 end),2) AS diff_in_sales
> FROM clickstream_bucket
> WHERE event_type='purchase';
Query ID = hadoop_20210815125343_d4ee2e4a-4705-4798-84ce-63a6a334453d
Total jobs = 1
Launching Job 1 out of 1
Status: Running (Executing on YARN cluster with App id application_1629031331370_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: 34.90 s
-----
OK
diff_in_sales
319478.47
Time taken: 36.463 seconds, Fetched: 1 row(s)
hive>
```

**Time taken:** 36.463 seconds

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

### Code:

```
SELECT distinct category_code
FROM clickstream_bucket
WHERE length(category_code) > 0;
```

### Comments:

**length(category\_code) > 0** is given because there exists one particular category which is a blank string and hence it doesn't give any meaningful insight to keep it in the result.

### Output:

```
hive> SELECT distinct category_code
> FROM clickstream_bucket
> WHERE length(category_code) > 0;
Query ID = hadoop_20210815070553_09f7d245-079e-424a-a7af-a3bcc8e4ef5b
Total jobs = 1
Launching Job 1 out of 1
Status: Running (Executing on YARN cluster with App id application_1629009426685_0003)

-----
VERTICES    MODE      STATUS  TOTAL  COMPLETED  RUNNING  PENDING  FAILED  KILLED
-----
Map 1 ..... container    SUCCEEDED    6         6         0         0         0         0
Reducer 2 ..... container    SUCCEEDED    2         2         0         0         0         0
-----
VERTICES: 02/02  [=====>>] 100%  ELAPSED TIME: 62.18 s
-----
OK
category_code
accessories.bag
apparel.glove
appliances.environment.vacuum
appliances.personal.hair_cutter
furniture.bathroom.bath
furniture.living_room.cabinet
sport.diving
stationery.cartridge
accessories.cosmetic_bag
appliances.environment.air_conditioner
furniture.living_room.chair
Time taken: 62.794 seconds, Fetched: 11 row(s)
hive> █
```

**Time taken:** 62.794 seconds

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

### Code:

```
SELECT category_code AS category, count(product_id) AS count_of_products
FROM clickstream_bucket
WHERE length(category_code) > 0
GROUP by category_code;
```

### Comments:

`length(category_code) > 0` is given because there exists one particular category which is a blank string and hence it doesn't give any meaningful insight to keep it in the result.



### Output:

```
hive> SELECT category_code AS category, count(product_id) AS count_of_products
> FROM clickstream_bucket
> WHERE length(category_code) > 0
> GROUP by category_code;
Query ID = hadoop_20210815070726_948a22f1-4a56-4c71-9475-86071dbab7ae
Total jobs = 1
Launching Job 1 out of 1
Status: Running (Executing on YARN cluster with App id application_1629009426685_0003)

-----
VERTICES    MODE        STATUS  TOTAL  COMPLETED  RUNNING  PENDING  FAILED  KILLED
-----
Map 1 ..... container    SUCCEEDED    6         6         0         0         0         0
Reducer 2 ..... container    SUCCEEDED    2         2         0         0         0         0
-----
VERTICES: 02/02  [=====>>] 100%  ELAPSED TIME: 62.38 s
-----
OK
category      count_of_products
accessories.bag 11681
apparel.glove  18232
appliances.environment.vacuum  59761
appliances.personal.hair_cutter 1643
furniture.bathroom.bath 9857
furniture.living_room.cabinet 13439
sport.diving    2
stationery.cartridge 26722
accessories.cosmetic_bag 1248
appliances.environment.air_conditioner 332
furniture.living_room.chair 308
Time taken: 62.944 seconds, Fetched: 11 row(s)
hive>
```

**Time taken:** 62.944 seconds

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

### Code:

```
SELECT brand AS brand, round(sum(price),2) AS max_sales
FROM clickstream_bucket
WHERE event_type='purchase' AND length(brand) > 0
GROUP BY brand
ORDER BY max_sales DESC
LIMIT 1;
```

### Comments:

**length(brand) > 0** is given because there exists one particular brand which is a blank string and hence it doesn't give any meaningful insight to keep it in the result.

Output:

```
hive> SELECT brand AS brand, round(sum(price),2) AS max_sales
> FROM clickstream_bucket
> WHERE event_type='purchase' AND length(brand) > 0
> GROUP BY brand
> ORDER BY max_sales DESC
> LIMIT 1;
Query ID = hadoop_20210815070933_acebef31-a8f3-4666-95c3-6elbf47abf99
Total jobs = 1
Launching Job 1 out of 1
Status: Running (Executing on YARN cluster with App id application_1629009426685_0003)

-----
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
Reducer 3 ..... container  SUCCEEDED   1         1         0         0         0         0
-----
VERTICES: 03/03  [=====>>] 100%  ELAPSED TIME: 20.85 s
-----
OK
brand    max_sales
runail  148297.94
Time taken: 21.554 seconds, Fetched: 1 row(s)
hive>
```

Time taken: 21.554 seconds

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

Code:

```
WITH sales AS
(
SELECT brand AS brand,
round(sum(case when date_format(event_time,'MM')=10 then price else 0
end),2) as oct_sales,
round(sum(case when date_format(event_time,'MM')=11 then price else 0
end),2) as nov_sales
FROM clickstream_bucket
WHERE event_type='purchase' and length(brand) > 0
GROUP BY brand
)
SELECT brand, round((nov_sales-oct_sales),2) AS diff_in_sales
FROM sales
WHERE nov_sales > oct_sales;
```

Comments:

**length(brand) > 0** is given because there exists one particular brand which is a blank string and hence it doesn't give any meaningful insight to keep it in the result.

## Output:

```
hive> WITH sales AS
> (
> SELECT brand AS brand,
> round(sum(case when date_format(event_time,'MM')=10 then price else 0 end),2) as oct_sales,
> round(sum(case when date_format(event_time,'MM')=11 then price else 0 end),2) as nov_sales
> FROM clickstream_bucket
> WHERE event_type='purchase' and length(brand) > 0
> GROUP BY brand
> )
> SELECT brand, round((nov_sales-oct_sales),2) AS diff_in_sales
> FROM sales
> WHERE nov_sales > oct_sales;
Query ID = hadoop_20210815125512_3da786c4-3aa0-436a-98b1-aefb19ealf06
Total jobs = 1
Launching Job 1 out of 1
Status: Running (Executing on YARN cluster with App id application_1629031331370_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: 32.07 s
-----
OK
brand diff_in_sales
airnails 572.62
art-visage 905.09
artex 1596.61
aura 93.56
balbicare 57.05
barbie 12.39
batiste 101.77
beautix 1729.0
beauty-free 1228.69
beautyblender 30.67
beauugreen 256.84
benovy 2850.35
binacil 24.26
bioagua 455.23
biore 29.66
blixz 24.45
bluesky 258.29
bodyton 4.3
bpw.style 3265.29
browxenna 585.36
candy 264.42
carmex 98.28
chi 179.67
coifin 525.49
concept 2348.26
cosima 0.7
cosmoprofi 6214.18
cristalinas 157.32
cutrin 68.25
de.lux 1115.81
deoproce 12.33
depilflax 96.71
dewal 61.29
dizao 126.38
domix 1537.12
```

ecocraft	200.79
ecolab	951.45
egomania	68.57
elizavecca	133.77
ellips	360.19
elskin	56.56
enjoy	95.22
entity	239.55
eos	98.27
estel	2385.92
estelare	27.06
f.o.x	1953.05
farmavita	454.6
farmona	150.97
fedua	211.43
finish	132.0
fly	10.03
foamie	45.45
freedecor	4250.02
freshbubble	183.64
gehwol	468.61
glysolid	21.86
godefroy	23.9
grace	1.69
grattol	36027.17
greymy	460.28
happyfons	289.67
haruyama	2962.22
helloganic	3.1
igrobeauty	131.41
ingarden	10404.82
inm	63.19
insight	278.26
irisk	1354.08
italwax	2859.13
jaguar	8.54
jas	338.47
jessnail	7057.39
joico	1309.58
juno	21.08
kaaral	673.64
kamill	18.48
kapous	2165.92
kares	59.45
kaypro	2387.36
keen	199.27
kerasys	94.29
kims	302.0
kinetics	611.01
kiss	395.78
kocostar	284.08
koelcia	57.25
koelf	84.56
konad	70.84
kosmekka	631.93
laboratorium	66.02
lador	387.92
ladykin	44.92
latinoil	135.07
levissime	857.81
levrana	1420.54

```

lianail 10501.4
likato 44.91
limoni 487.7
lovely 3234.68
lowence 324.91
mane 193.47
marathon 2992.35
markell 1065.68
marutaka-foot 60.11
masura 1792.39
matreshka 182.67
matrix 483.49
mavala 37.28
metzger 1083.71
milv 1737.07
miskin 135.03
missha 856.45
moyou 4.57
nagaraku 957.94
naomi 389.0
nefertiti 133.12
neoleor 8.29
nirvel 71.29
nitrile 315.4
onig 1416.24
orly 28.71
osmo 116.73
ovale 0.56
plazan 92.64
polarus 5358.21
profepil 24.66
profhenna 57.62
protokeratin 255.54
provoc 235.83
rasyan 10.14
refectocil 759.4
rosi 764.52
roubloff 1422.41
runail 5219.38
s.care 500.39
sanoto 1052.54
severina 1344.6
shary 304.53
shik 1498.52
skinity 3.56
skinlite 238.51
smart 1444.88

```

```

soleo 8.33
solomeya 786.1
sophin 447.66
staleks 3355.88
strong 9474.64
supertan 16.14
swarovski 1155.23
tertio 9.64
treaclemoon 18.12
trind 244.89
uno 15737.72
uskusi 548.04
veraclara 21.1
vilenta 33.61
yoko 2950.97
yu-r 402.3
zeitun 1300.97
Time taken: 32.789 seconds, Fetched: 160 row(s)
hive>

```

**Time taken:** 32.789 seconds

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.

**Code :**

```
SELECT user_id AS user_id, round(sum(price),2) AS total_spend,
dense_rank() over(order by sum(price) desc) AS user_rank
FROM clickstream_bucket
WHERE event_type='purchase'
GROUP BY user_id
LIMIT 10;
```

**Output:**

```
hive> SELECT user_id AS user_id, round(sum(price),2) AS total_spend, dense_rank() over(order by sum(price) desc) AS u
ser_rank
> FROM clickstream_bucket
> WHERE event_type='purchase'
> GROUP BY user_id
> LIMIT 10;
Query ID = hadoop_20210815071310_3314fc01-e6d0-4d2c-9b3b-7f88b80f937f
Total jobs = 1
Launching Job 1 out of 1
Status: Running (Executing on YARN cluster with App id application_1629009426685_0003)

-----
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
Reducer 3 ..... container  SUCCEEDED   1         1         0         0         0         0
-----
VERTICES: 03/03 [=====>>] 100% ELAPSED TIME: 25.03 s
-----
OK
user_id total_spend  user_rank
957790271      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: 26.158 seconds, Fetched: 10 row(s)
hive>
```

**Time taken:** 26.158 seconds

Thus the required analysis was performed on the dataset of the cosmetic store's data and the results are also displayed above.