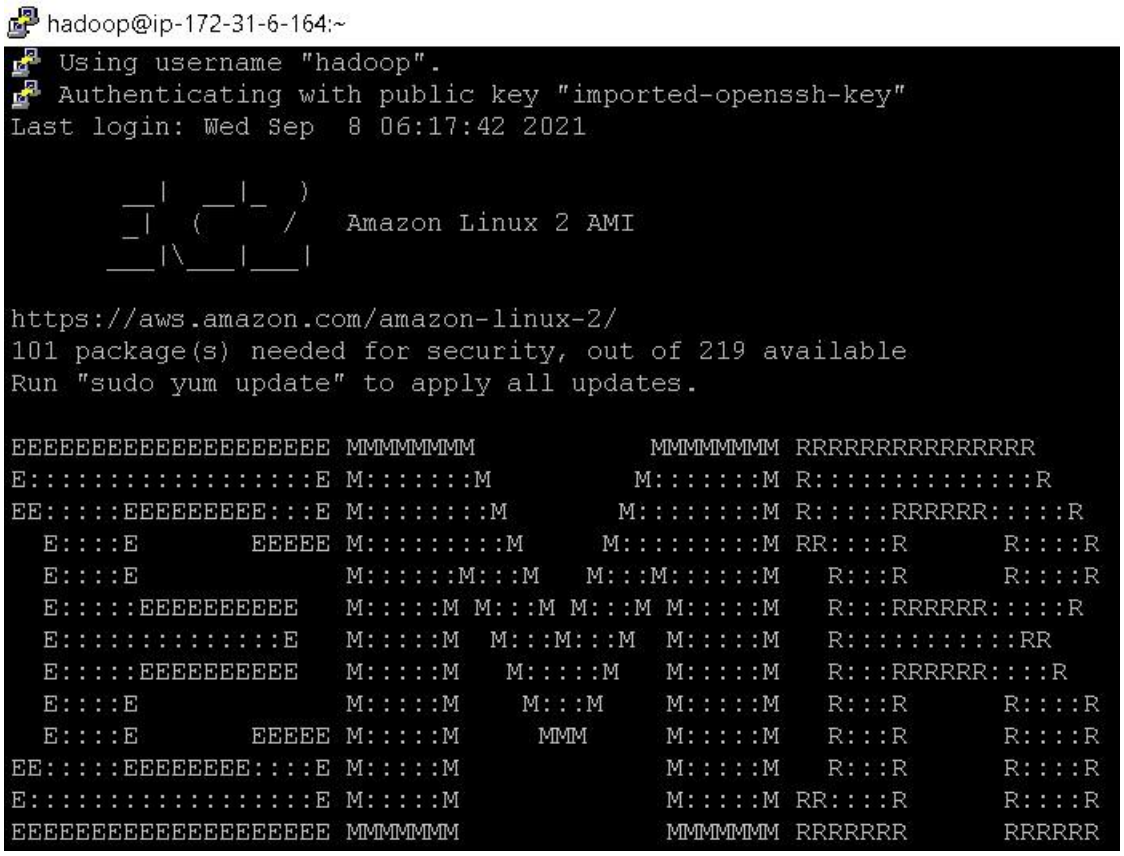
Hive Case Study

The following steps are performed in the hive case study :

1) Connect the local machine to the master node using SSH



2) **Create a folder named 'nilam' in the HDFS using the following command** :
hadoop fs -mkdir /nilam

3) **To check the created folder in the HDFS use the following command** :
hadoop fs –ls /

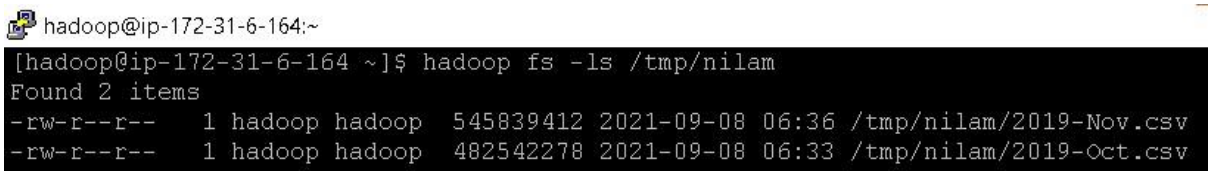
```
[hadoop@ip-172-31-6-164 ~]$ hadoop fs -mkdir /nilam
[hadoop@ip-172-31-6-164 ~]$ hadoop fs -ls /
Found 5 items
drwxr-xr-x   - hdfs   hadoop          0 2021-09-08 06:11 /apps
drwxr-xr-x   - hadoop hadoop          0 2021-09-08 06:25 /nilam
drwxrwxrwt   - hdfs   hadoop          0 2021-09-08 06:13 /tmp
drwxr-xr-x   - hdfs   hadoop          0 2021-09-08 06:11 /user
drwxr-xr-x   - hdfs   hadoop          0 2021-09-08 06:11 /var
```

We can see in the screenshot that the folder named 'nilam' is created in the HDFS.

4) **Import the data to the folder 'nilam' in the HDFS using the following two commands** :

i) hadoop distcp s3://nilamhivecs/2019-Oct.csv /tmp/nilam/2019-Oct.csv
ii) hadoop distcp s3://nilamhivecs/2019-Nov.csv /tmp/nilam/2019-Nov.csv

5) **Now we have imported the data in the HDFS. To see the imported data run the following command** :
hadoop fs –ls /tmp/nilam



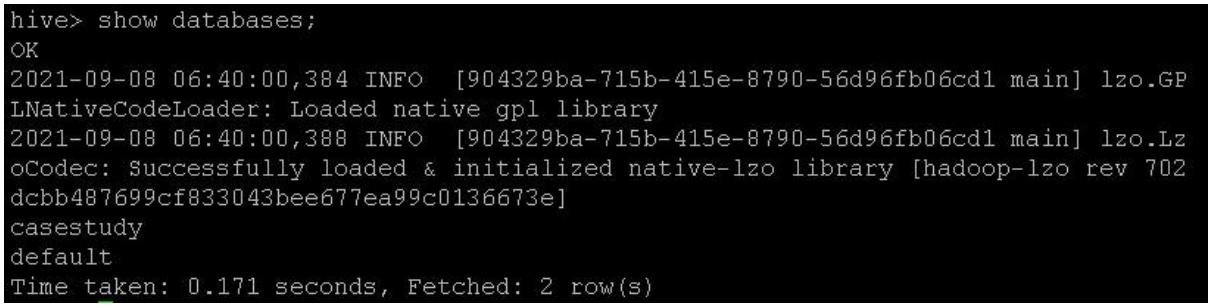
We can see both the files uploaded in the HDFS.

6) **Launch the hive service. For this run the command** 'hive'.

7) Create the database named 'casestudy' using the following query :

create database if not exists casestudy ;

8) To see the created database run the following query :
show databases ;



We can see in the screenshot above that the database named 'casestudy' has created.

9) To use the database 'casestudy' run the command '*use casestudy* ;' .

10) Create the external table by using the following query

create table if not exists table2019 (event_time int , event_type string , product_id int , category_id int , category_code string , brand string , price float , user_id int , user_session string) row format serde 'org.apache.hadoop.hive.serde2.OpenCSVSerde' stored as textfile location '/tmp/nilam/' tblproperties ("skip.header.line.count"="1");

```
hive> create table if not exists table2019(event_time int, event_type string, pr
oduct_id int, category_id int, category_code string, brand string, price float,
user_id int, user_session string) row format serde 'org.apache.hadoop.hive.serde
2.OpenCSVSerde' stored as textfile location '/tmp/nilam/' tblproperties("skip.he
ader.line.count"="1");
OK
Time taken: 0.338 seconds
```

11) **Describe the table 'table2019' by using the following query :**

describe table2019;

```
hive> describe table2019;
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: 0.101 seconds, Fetched: 9 row(s)
```

12) To show the headers for all the queries use the following query :

set hive.cli.print.header=true ;

13**) Create the partitioning and bucketing using the following command :**

create external table if not exists ext_table2019 (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 (category_code) into 12 buckets row format serde 'org.apache.hadoop.hive.serde2.OpenCSVSerde' stored as textfile;

14) **Set dynamic partitioning mode to nonstrict using the following command :**

set hive.exec.dynamic.partition.mode=nonstrict ;

15) **Load the data in the partitioned and bucketed table named 'ext_table2019' using the following command :**

insert into ext_table2019 partition (event_type) select event_time, product_id, category_id, category_code, brand, price, user_id, user_session, event_type from table2019

```
hive> set hive.exec.dynamic.partition.mode=nonstrict;
hive> insert into ext_table2019 partition(event_type) select event_time, product
_id, category_id, category_code, brand, price, user_id, user_session, event_type
 from table2019;
2021-09-08 10:35:04,103 INFO  [8404347b-3b97-46fa-8a5b-4c2cdddb1557 main] reducesink.VectorReduceSinkObjectHashOperator: VectorReduceSinkObjectHashOperator constructor vectorReduceSinkInfo
org.apache.hadoop.hive.ql.plan.VectorReduceSinkInfo@333e01c6
Query ID = hadoop_20210908103503_bc4eaf17-9046-472a-b259-45db68ad3e65
Total jobs = 1
Launching Job 1 out of 1
Status: Running (Executing on YARN cluster with App id application_1631081627075_0020)

----------------------------------------------------------------------------------------------
        VERTICES      MODE        STATUS  TOTAL  COMPLETED  RUNNING  PENDING  FAILED  KILLED
----------------------------------------------------------------------------------------------
Map 1 .......... container     SUCCEEDED      2          2        0        0       0       0
Reducer 2 ...... container     SUCCEEDED     11         11        0        0       0       0
----------------------------------------------------------------------------------------------
VERTICES: 02/02  [==========================>>] 100%  ELAPSED TIME: 116.57 s
----------------------------------------------------------------------------------------------
Loading data to table casestudy.ext_table2019 partition (event_type=null)

Loaded : 4/4 partitions.
        Time taken to load dynamic partitions: 1.93 seconds
        Time taken for adding to write entity : 0.003 seconds
OK
Time taken: 120.95 seconds
hive>
```

Query Optimization

We have created the partitioned and bucketed table named 'ext_table2019' to optimize the queries. Let's see how we can optimize queries through the example.

Q. Fetch the first 10 rows.

First we will fetch the first 10 rows of the table 'table2019'.

```
hadoop@ip-172-31-6-164:~
hive> select * from table2019 limit 10;
OK
table2019.event_time    table2019.event_type    table2019.product_id    table201
9.category_id   table2019.category_code table2019.brand table2019.price table201
9.user_id       table2019.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     2
2.22    556138645       57ed222e-a54a-4907-9944-5a875c2d7f4f
2019-11-01 00:00:11 UTC cart    5876812 1487580010100293687             jessnail
3.16    564506666       186c1951-8052-4b37-adce-dd9644b1d5f7
2019-11-01 00:00:24 UTC remove_from_cart        5826182 1487580007483048900     3
.33     553329724       2067216c-31b5-455d-a1cc-af0575a34ffb
2019-11-01 00:00:24 UTC remove_from_cart        5826182 1487580007483048900     3
.33     553329724       2067216c-31b5-455d-a1cc-af0575a34ffb
2019-11-01 00:00:25 UTC view    5856189 1487580009026551821             runail 1
5.71    562076640       09fafd6c-6c99-46b1-834f-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     m
ilv     0.79    429913900       2f0bff3c-252f-4fe6-afcd-5d8a6a92839a
2019-11-01 00:00:37 UTC view    5870803 1487580007675986893             milv    0
.79     429913900       2f0bff3c-252f-4fe6-afcd-5d8a6a92839a
Time taken: 0.267 seconds, Fetched: 10 row(s)
```

Here we can see that the time taken to execute the query in the table ‘table2019’ is 0.267 seconds.

Now, we will fetch the first 10 rows of the partitioned and bucketed table ‘ext_table2019’ .

```
hive> select * from ext_table2019 limit 10;
OK
ext_table2019.event_time        ext_table2019.product_id        ext_table2019.ca
tegory_id       ext_table2019.category_code     ext_table2019.brand     ext_tabl
e2019.price     ext_table2019.user_id   ext_table2019.user_session      ext_tabl
e2019.event_type
2019-10-10 14:45:40 UTC 5858495 1487580005754995573                    4.60    5
51819168        7e09bf98-aadb-4f3d-98dd-470be176cbc7    cart
2019-10-07 20:53:08 UTC 5635080 1487580005754995573                    4.44    5
27827629        b5f0f964-9457-4dfd-bade-239a9cde9c5d    cart
2019-10-10 14:45:41 UTC 5762728 1487580013950664926                    4.44    5
49261009        723a7bea-e6a4-c6b6-63e6-c7076529b77d    cart
2019-10-01 00:00:03 UTC 5773353 1487580005134238553             runail  2.62    4
63240011        26dd6e6e-4dac-4778-8d2c-92e149dab885    cart
2019-10-09 04:44:26 UTC 5861326 1487580008145748965                    0.70    5
53450358        ae5665f5-110f-46ca-99d1-8c81355a8b07    cart
2019-10-01 00:00:07 UTC 5881589 2151191071051219817             lovely  13.48   4
29681830        49e8d843-adf3-428b-a2c3-fe8bc6a307c9    cart
2019-10-08 10:53:30 UTC 5850320 1487580005754995573                    4.44    4
17575049        dbbdfb29-4006-42a1-a8a5-f9a6d33f1c4f    cart
2019-10-01 00:00:07 UTC 5723490 1487580005134238553             runail  2.62    4
63240011        26dd6e6e-4dac-4778-8d2c-92e149dab885    cart
2019-10-09 04:44:27 UTC 5670326 1487580005754995573                    4.44    5
57576916        01782ae8-e912-4eeb-88a2-369195c20147    cart
2019-10-01 00:00:15 UTC 5881449 1487580013522845895             lovely  0.56    4
29681830        49e8d843-adf3-428b-a2c3-fe8bc6a307c9    cart
Time taken: 0.233 seconds, Fetched: 10 row(s)
```

Here we can see that the time taken to execute the query in the partitioned and bucketed table ‘ext_table2019’ is 0.233 seconds.

From this example, we can conclude that the partitioned and bucketed table ‘ext_table2019’ takes less time as compared to table ‘table2019’ to execute the query. So now onwards, we will perform all the queries on the partitioned and bucketed table ‘ext_table2019’.

Questions and Answers

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

Answer : *select sum(price) as revenue from ext_table2019 where month(event_time)=10 and event_type = 'purchase';*

```
hive> select sum(price) as revenue from ext_table2019 where month(event_time)=10
 and event_type='purchase';
Query ID = hadoop_20210908094126_031a87c4-e35e-468d-ac27-97a1831ce8d6
Total jobs = 1
Launching Job 1 out of 1
Status: Running (Executing on YARN cluster with App id application_1631081627075_0016)

--------------------------------------------------------------------------------
        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: 20.84 s
--------------------------------------------------------------------------------
OK
1211538.4299996966
Time taken: 22.311 seconds, Fetched: 1 row(s)
hive> -
```

The total revenue generated due to the purchases made in October month is 1211538.4299996966.

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

Answer : *select month(event_time), sum(price) from ext_table2019 where year(event_time)=2019 and event_type='purchase' group by month(event_time);*

```
hive> select month(event_time), sum(price) from ext_table2019 where year(event_t
ime)=2019 and event_type='purchase' group by month(event_time);
Query ID = hadoop_20210908094451_01a7db19-2679-48d0-8ad0-456b57db05ae
Total jobs = 1
Launching Job 1 out of 1
Status: Running (Executing on YARN cluster with App id application_1631081627075_0016)

--------------------------------------------------------------------------------
        VERTICES      MODE        STATUS  TOTAL  COMPLETED  RUNNING  PENDING  FAILED  KILLED
--------------------------------------------------------------------------------
Map 1 .......... container     SUCCEEDED      3          3        0        0       0       0
Reducer 2 ...... container     SUCCEEDED      4          4        0        0       0       0
--------------------------------------------------------------------------------
VERTICES: 02/02  [==========================>>] 100%  ELAPSED TIME: 25.29 s
--------------------------------------------------------------------------------
OK
10      1211538.4299996966
11      1531016.8999998341
Time taken: 26.286 seconds, Fetched: 2 row(s)
hive>
```

The total sum of purchases in the month of October is 1211538.4299996966 and the total sum of purchases in the month of November is 1531016.8999998341.

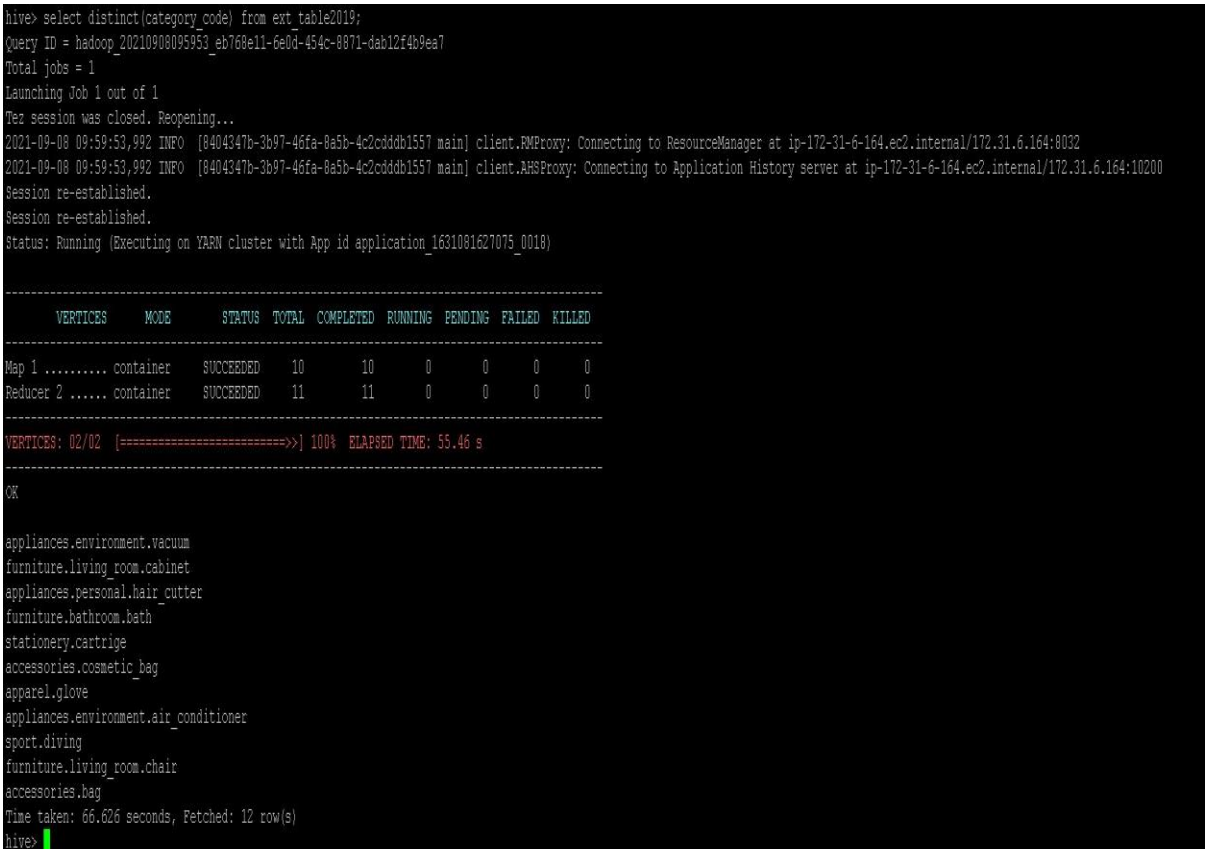
3) Write a query to find the change in revenue generated due to purchases from October to November.

Answer : *select sum(case when month(from_unixtime(unix_timestamp(event_time,'yyyy-MM-dd HH:mm:ss'))) = 10 then cast (price as float) else - 1 * cast(price as float)end) as change_in_revenue from ext_table2019 where month(from_unixtime(unix_timestamp(event_time, 'yyyy-MMdd HH:mm:ss'))) IN (10, 11) and event_type = 'purchase';*

The change in revenue generated due to purchases from October to November is 319478.469592195 . The negative sign indicates that the revenue of November is less than the revenue of October .

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

Answer : *select distinct(category_code) from ext_table2019*



The distinct categories of the products are as follows :

i) Appliances.environment.vacuum
ii) Furniture.living_room.cabinet
iii) Appliances.person.hair_cutter
iv) Furniture.bathroom.bath
v) Stationary.cartrige
vi) Accessories.cosmetic_bag
vii) Apparel.glove
viii) Appliances.environment.air_conditioner
ix) Sport.diving
x) Furniture.living_room.chair
xi) Accessories.bag

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

Answer : *select category_code ,count(product_id) as total_order from ext_table2019 group by category_code;*

```
hive> select category_code, count(product_id) as total_order from ext_table2019
group by category_code ;
Query ID = hadoop_20210908100316_9ab00aff-d5e7-4d56-95df-3bd487f7fee5
Total jobs = 1
Launching Job 1 out of 1
Status: Running (Executing on YARN cluster with App id application_1631081627075_0018)

--------------------------------------------------------------------------------
        VERTICES      MODE        STATUS  TOTAL  COMPLETED  RUNNING  PENDING  FAILED  KILLED
--------------------------------------------------------------------------------
Map 1 .......... container     SUCCEEDED     10         10        0        0       0       0
Reducer 2 ...... container     SUCCEEDED     11         11        0        0       0       0
--------------------------------------------------------------------------------
VERTICES: 02/02  [==========================>>] 100%  ELAPSED TIME: 48.71 s
--------------------------------------------------------------------------------
OK
        8594895
appliances.environment.vacuum   59761
furniture.living_room.cabinet   13439
appliances.personal.hair_cutter 1643
furniture.bathroom.bath 9857
stationery.cartrige     26722
accessories.cosmetic_bag        1248
apparel.glove   18232
appliances.environment.air_conditioner  332
sport.diving    2
furniture.living_room.chair     308
accessories.bag 11681
Time taken: 49.717 seconds, Fetched: 12 row(s)
```

i) Appliances.environment.vacuum -59761
ii) Furniture.living_room.cabinet – 13439
iii) Appliances.person.hair_cutter - 1643
iv) Furniture.bathroom.bath – 9857
v) Stationary.cartrige – 26722
vi) Accessories.cosmetic_bag – 1248
vii) Apparel.glove – 18232
viii) Appliances.environment.air_conditioner -332
ix) Sport.diving – 2
x) Furniture.living_room.chair – 308
xi) Accessories.bag -11681

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

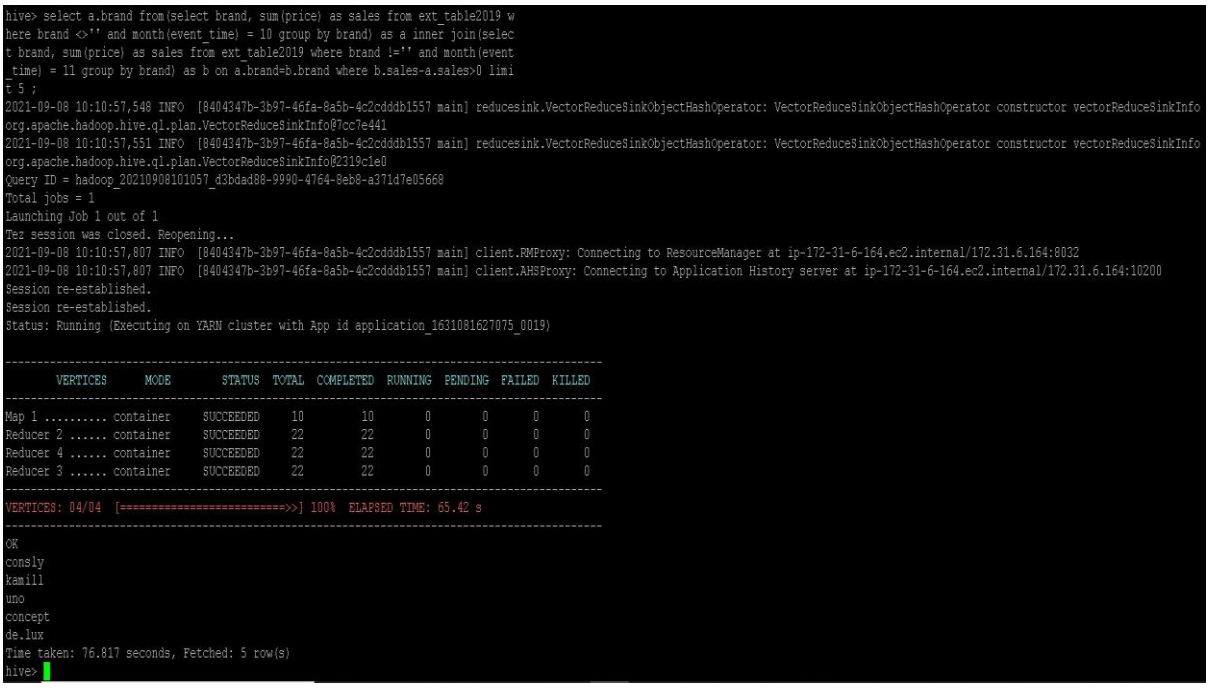
Answer : *select brand, sum(price)as sales from ext_table2019 where brand is not null and event_type='purchase' group by brand order by sales desc limit 1;*

```
hive> select brand, sum(price) as sales from ext_table2019 where brand is not nu
ll and event_type = 'purchase' group by brand order by sales desc limit 1;
2021-09-08 10:15:24,820 INFO  [8404347b-3b97-46fa-8a5b-4c2cdddb1557 main] reducesink.VectorReduceSinkObjectHashOperator: VectorReduceSinkObjectHashOperator constructor vectorReduceSinkInfo
org.apache.hadoop.hive.ql.plan.VectorReduceSinkInfo@285ac29
Query ID = hadoop_20210908101524_110416eb-915d-4da6-9f68-872de08a9a9a
Total jobs = 1
Launching Job 1 out of 1
Status: Running (Executing on YARN cluster with App id application_1631081627075_0019)

--------------------------------------------------------------------------------
        VERTICES      MODE        STATUS  TOTAL  COMPLETED  RUNNING  PENDING  FAILED  KILLED
--------------------------------------------------------------------------------
Map 1 .......... container     SUCCEEDED      3          3        0        0       0       0
Reducer 2 ...... container     SUCCEEDED      6          6        0        0       0       0
Reducer 3 ...... container     SUCCEEDED      1          1        0        0       0       0
--------------------------------------------------------------------------------
VERTICES: 03/03  [==========================>>] 100%  ELAPSED TIME: 24.92 s
--------------------------------------------------------------------------------
OK
        1094188.300000023
Time taken: 26.467 seconds, Fetched: 1 row(s)
```

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

Answer : *select a.brand from(select brand, sum(price) as sales from ext_table2019 where brand <>'' and month(event_time)=10 group by brand) as a inner join (select brand,sum(price) as sales from ext_table2019 where brand !='' and month(event_time)=11 group by brand) as b on a.brand=b.brand where b.salesa.sales>0 limit 5;*

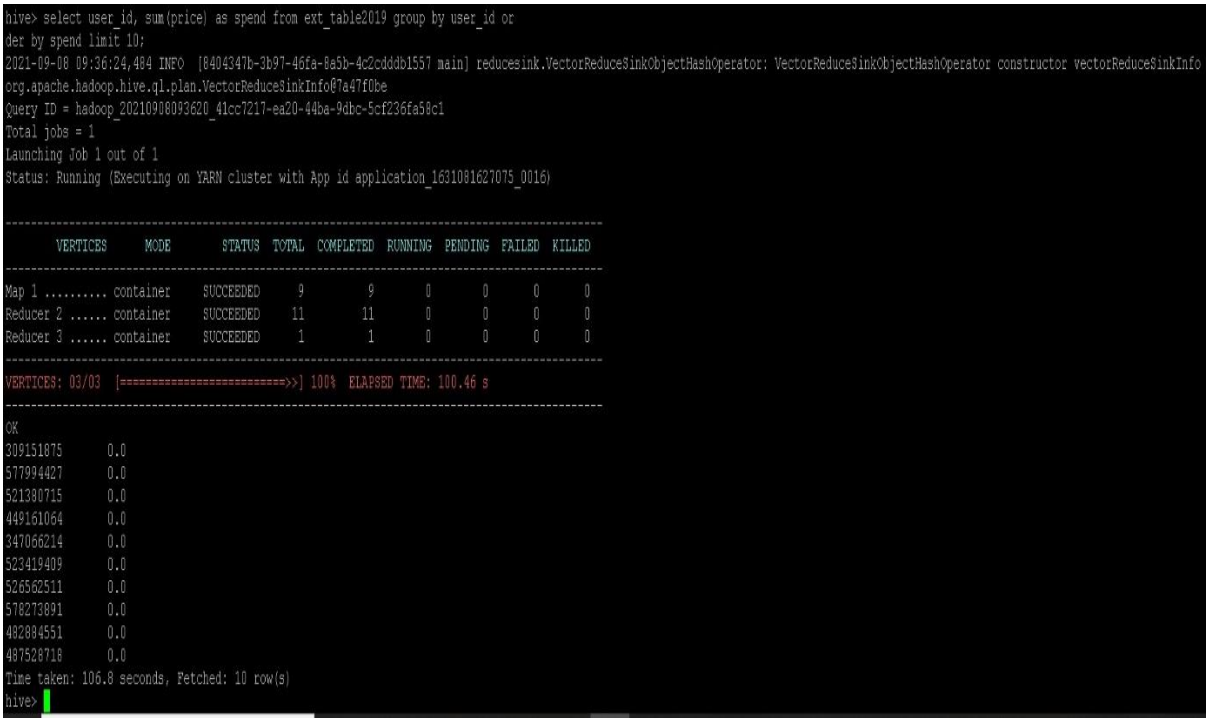


The following top 5 brands increased their sales from October to November:

1) Consly
2) Kamill
3) Uno
4) Concept
5) De.lux

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**.

Answer : *select user_id, sum(price) as spend from ext_table2019 group by user_id order by spend limit 10;*


hive> select user_id, sum(price) as spend from ext_table2019 group by user_id or
der by spend limit 10;
2021-09-08 09:36:24,484 INFO [0404347b-3b97-46fa-8a5b-4c2cdddb1557 main] reducesink.VectorReduceSinkObjectHashOperator: VectorReduceSinkObjectHashOperator constructor vectorReduceSinkInfo
org.apache.hadoop.hive.ql.plan.VectorReduceSinkInfo@7a47f0be
Query ID = hadoop_20210908093620_41cc7217-ea20-44ba-9dbc-5cf236fa58c1
Total jobs = 1
Launching Job 1 out of 1
Status: Running (Executing on YARN cluster with App id application_1631081627075_0016)
VERTICES MODE STATUS TOTAL COMPLETED RUNNING PENDING FAILED KILLED
Map 1 ..... container SUCCEEDED 9 9 0 0 0 0
Reducer 2 ..... container SUCCEEDED 11 11 0 0 0 0
Reducer 3 ..... container SUCCEEDED 1 1 0 0 0 0
VERTICES: 03/03 [==========>>] 100% ELAPSED TIME: 100.46 s
OK
309151875 0.0
577994427 0.0
521380715 0.0
449161064 0.0
347066214 0.0
523419409 0.0
526562511 0.0
578273891 0.0
482884551 0.0
487528718 0.0
Time taken: 106.8 seconds, Fetched: 10 row(s)
hive>