E-COMMERCE PROJECT

DATA INTRODUCTION:

This Project consist of one Table of E-commerce website for 5 months October 2020-February 2021.
The table consisted of details of users using the website .
The abbreviations used are below:

UTC-Universal Time Coordinated
IST – Indian Time Standard

PROBLEM STATEMENT:

Despite the huge usage of ecommerce website a section of the society still prefers offline stores. Why?

Objective:

- To analyze the usage of ecommerce website.
- To provide suggestions for improving the business on basis of Findings.

The data for the analysis has been attached below:

events.csv.zip

The data in CSV file suggested that it had more than 8 Lakhs records or rows and nine fields or columns namely .

My Work File:

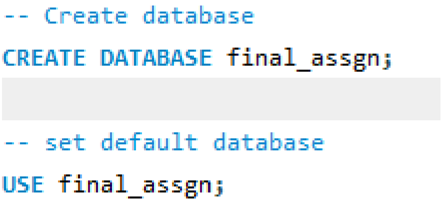
final_assgn.sql

Here is my work data

<u>MY WORK ON DATA</u>

Data Uploading and Data Cleaning and Modifying

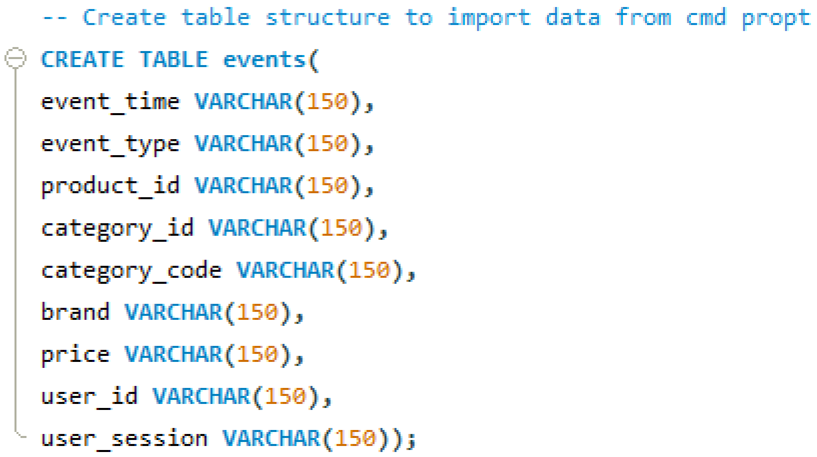
- Firstly we'll need to create a database so that we can upload our date into it.
  So we'll create database 'final_assgn' and set it as default.



```
-- Create database
CREATE DATABASE final_assgn;

-- set default database
USE final_assgn;
```

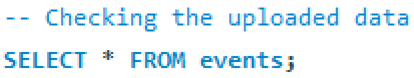
- Now We'll Create our table structure in simple datatype 'VARCHAR()' So that we can easily upload our data from command promt and change it later according to our columns datatype.



```
-- Create table structure to import data from cmd propt
CREATE TABLE events(
event_time VARCHAR(150),
event_type VARCHAR(150),
product_id VARCHAR(150),
category_id VARCHAR(150),
category_code VARCHAR(150),
brand VARCHAR(150),
price VARCHAR(150),
user_id VARCHAR(150),
user_session VARCHAR(150));
```

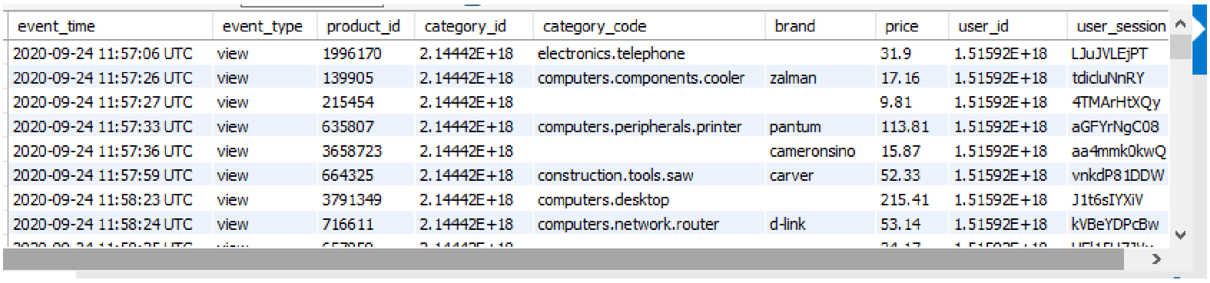
- Our table is now ready we'll upload our data using command prompt
- Checking our uploaded data

Query:



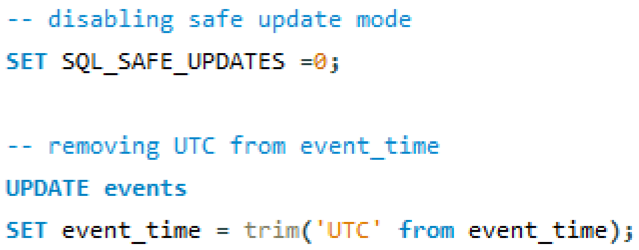
```
-- Checking the uploaded data
SELECT * FROM events;
```

Output:



event_time	event_type	product_id	category_id	category_code	brand	price	user_id	user_session
2020-09-24 11:57:06 UTC	view	1996170	2.14442E+18	electronics.telephone		31.9	1.51592E+18	LJuJVLEjPT
2020-09-24 11:57:26 UTC	view	139905	2.14442E+18	computers.components.cooler	zalman	17.16	1.51592E+18	tdicluNnRY
2020-09-24 11:57:27 UTC	view	215454	2.14442E+18			9.81	1.51592E+18	4TMArHtXQy
2020-09-24 11:57:33 UTC	view	635807	2.14442E+18	computers.peripherals.printer	pantum	113.81	1.51592E+18	aGFYrNgC08
2020-09-24 11:57:36 UTC	view	3658723	2.14442E+18		cameronsino	15.87	1.51592E+18	aa4mmk0kwQ
2020-09-24 11:57:59 UTC	view	664325	2.14442E+18	construction.tools.saw	carver	52.33	1.51592E+18	vnkdP81DDW
2020-09-24 11:58:23 UTC	view	3791349	2.14442E+18	computers.desktop		215.41	1.51592E+18	J1t6sIYXiV
2020-09-24 11:58:24 UTC	view	716611	2.14442E+18	computers.network.router	d-link	53.14	1.51592E+18	kVBeYDPcBw

- Disabling Safe update and removing “UTC” from event time column



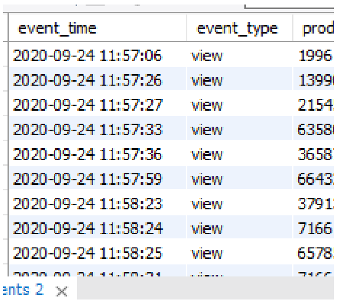
```
-- disabling safe update mode
SET SQL_SAFE_UPDATES =0;

-- removing UTC from event_time
UPDATE events
SET event_time = trim('UTC' from event_time);
```

- Checking updated Data

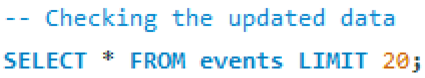
Query:

Output:



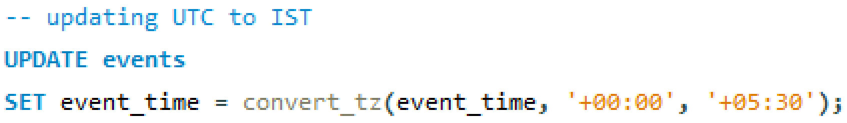
event_time	event_type	prod
2020-09-24 11:57:06	view	1996
2020-09-24 11:57:26	view	1399
2020-09-24 11:57:27	view	2154
2020-09-24 11:57:33	view	6358
2020-09-24 11:57:36	view	3658
2020-09-24 11:57:59	view	6643
2020-09-24 11:58:23	view	3791
2020-09-24 11:58:24	view	7166
2020-09-24 11:58:25	view	6578

ents 2



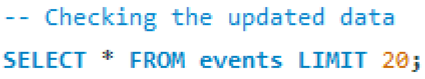
```
-- Checking the updated data
SELECT * FROM events LIMIT 20;
```

- Updating UTC time IST time



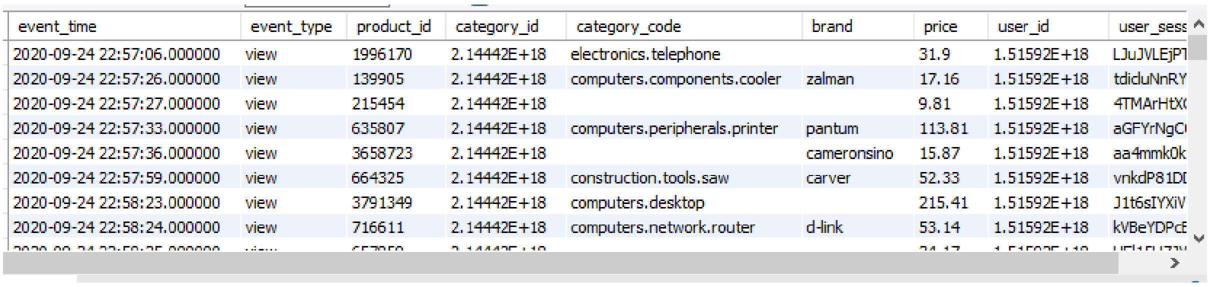
```
-- updating UTC to IST
UPDATE events
SET event_time = convert_tz(event_time, '+00:00', '+05:30');
```

- Checking updated Data



```
-- Checking the updated data
SELECT * FROM events LIMIT 20;
```

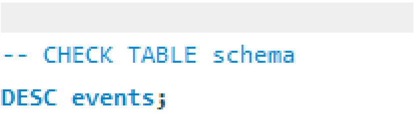
Output:



event_time	event_type	product_id	category_id	category_code	brand	price	user_id	user_sess
2020-09-24 22:57:06.000000	view	1996170	2.14442E+18	electronics.telephone		31.9	1.51592E+18	LJuJVLEjPT
2020-09-24 22:57:26.000000	view	139905	2.14442E+18	computers.components.cooler	zalman	17.16	1.51592E+18	tdiduNnRY
2020-09-24 22:57:27.000000	view	215454	2.14442E+18			9.81	1.51592E+18	4TMArHtX
2020-09-24 22:57:33.000000	view	635807	2.14442E+18	computers.peripherals.printer	pantum	113.81	1.51592E+18	aGFYrNgC
2020-09-24 22:57:36.000000	view	3658723	2.14442E+18		cameronsino	15.87	1.51592E+18	aa4mmk0k
2020-09-24 22:57:59.000000	view	664325	2.14442E+18	construction.tools.saw	carver	52.33	1.51592E+18	vnkdP81D
2020-09-24 22:58:23.000000	view	3791349	2.14442E+18	computers.desktop		215.41	1.51592E+18	J1t6sIYXiV
2020-09-24 22:58:24.000000	view	716611	2.14442E+18	computers.network.router	d-link	53.14	1.51592E+18	kVBeYDPcE

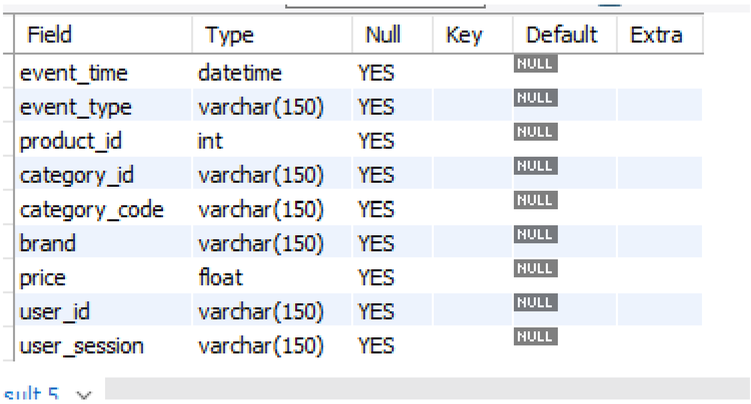
- Now we’ll change our datatypes of the & check the updated table schema

Query:



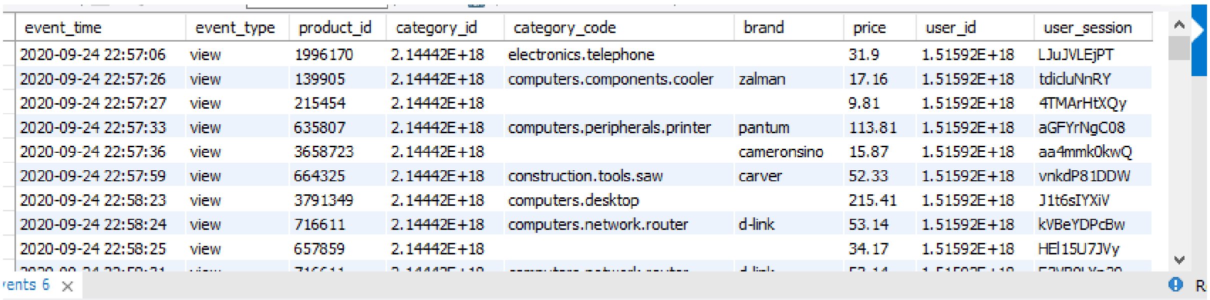
```
-- CHECK TABLE schema
DESC events;
```

Output:



Field	Type	Null	Key	Default	Extra
event_time	datetime	YES		NULL	
event_type	varchar(150)	YES		NULL	
product_id	int	YES		NULL	
category_id	varchar(150)	YES		NULL	
category_code	varchar(150)	YES		NULL	
brand	varchar(150)	YES		NULL	
price	float	YES		NULL	
user_id	varchar(150)	YES		NULL	
user_session	varchar(150)	YES		NULL	

- This is how our Updated Data looks like now



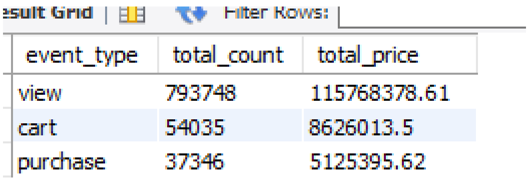
event_time	event_type	product_id	category_id	category_code	brand	price	user_id	user_session
2020-09-24 22:57:06	view	1996170	2.14442E+18	electronics.telephone		31.9	1.51592E+18	LJuJVLEjPT
2020-09-24 22:57:26	view	139905	2.14442E+18	computers.components.cooler	zalman	17.16	1.51592E+18	tdicluNnRY
2020-09-24 22:57:27	view	215454	2.14442E+18			9.81	1.51592E+18	4TMArHtXQy
2020-09-24 22:57:33	view	635807	2.14442E+18	computers.peripherals.printer	pantum	113.81	1.51592E+18	aGFYrNgC08
2020-09-24 22:57:36	view	3658723	2.14442E+18		cameronsino	15.87	1.51592E+18	aa4mmk0kwQ
2020-09-24 22:57:59	view	664325	2.14442E+18	construction.tools.saw	carver	52.33	1.51592E+18	vnkdP81DDW
2020-09-24 22:58:23	view	3791349	2.14442E+18	computers.desktop		215.41	1.51592E+18	J1t6sIYXiV
2020-09-24 22:58:24	view	716611	2.14442E+18	computers.network.router	d-link	53.14	1.51592E+18	kVBeYDPcBw
2020-09-24 22:58:25	view	657859	2.14442E+18			34.17	1.51592E+18	HEl15U7JVy

Data Understanding

- Now we’ll have look at each column like what kind of data it has stored and

1. event_type

```
-- event_type
-- cheking total counts of event_type and its total_price
SELECT event_type, COUNT(*) total_count, round(SUM(price),2) total_price
FROM events GROUP BY event_type;
```

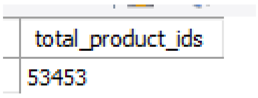


event_type	total_count	total_price
view	793748	115768378.61
cart	54035	8626013.5
purchase	37346	5125395.62

So as we can see that we have total 3 event type and total counts and total price

2. Product_id

```
-- product_id
SELECT COUNT(DISTINCT product_id) as total_product_ids FROM events;
```

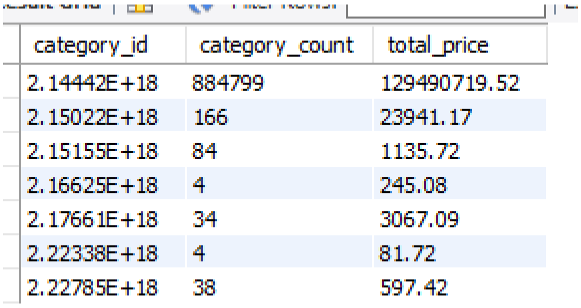


total_product_ids
53453

We have total 53453 unique product ids

3. Category_id

```
-- category_id
SELECT DISTINCT category_id, COUNT(*) category_count,round(SUM(price),2) total_price
FROM events GROUP BY category_id;
```

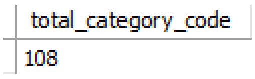


category_id	category_count	total_price
2.14442E+18	884799	129490719.52
2.15022E+18	166	23941.17
2.15155E+18	84	1135.72
2.16625E+18	4	245.08
2.17661E+18	34	3067.09
2.22338E+18	4	81.72
2.22785E+18	38	597.42

There are 7 unique categories, we can see their total_sales and total occurance

4. Category_code

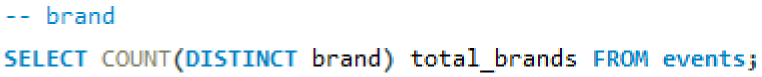
```
-- category_code
SELECT COUNT( DISTINCT category_code) total_category_code FROM events;
```



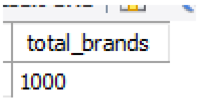
total_category_code
108

There are total 108 unique categories.

5. brands



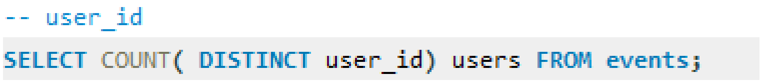
```
-- brand
SELECT COUNT(DISTINCT brand) total_brands FROM events;
```



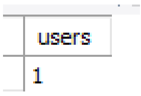
total_brands
1000

There are total 1000 brands.

6. User_id



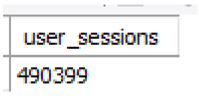
```
-- user_id
SELECT COUNT( DISTINCT user_id) users FROM events;
```



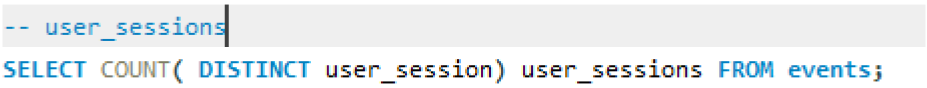
users
1

As we can see we have data of one user only.

7. User_sessions



user_sessions
490399

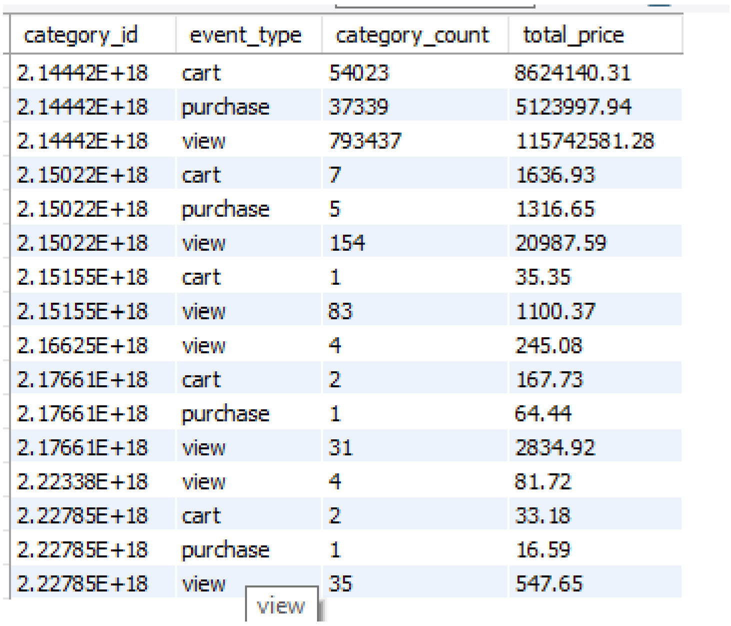


```
-- user_sessions
SELECT COUNT( DISTINCT user_session) user_sessions FROM events;
```

There are total 490399 unique sessions,

8. Category id and Event type

```
-- category id and event type
SELECT DISTINCT category_id, event_type, COUNT(*) category_count, round(SUM(price),2) total_price
FROM events GROUP BY category_id, event_type;
```



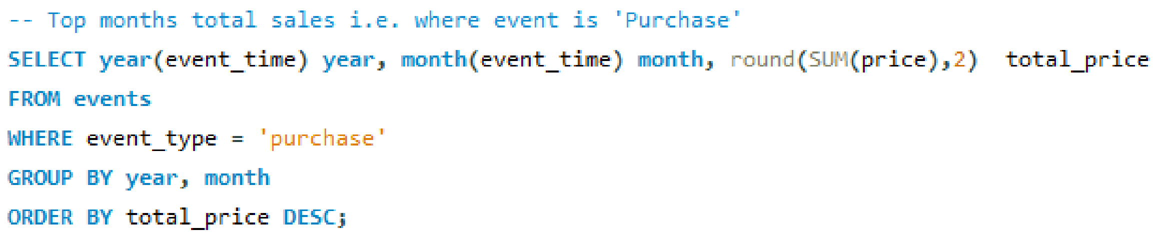
category_id	event_type	category_count	total_price
2.14442E+18	cart	54023	8624140.31
2.14442E+18	purchase	37339	5123997.94
2.14442E+18	view	793437	115742581.28
2.15022E+18	cart	7	1636.93
2.15022E+18	purchase	5	1316.65
2.15022E+18	view	154	20987.59
2.15155E+18	cart	1	35.35
2.15155E+18	view	83	1100.37
2.16625E+18	view	4	245.08
2.17661E+18	cart	2	167.73
2.17661E+18	purchase	1	64.44
2.17661E+18	view	31	2834.92
2.22338E+18	view	4	81.72
2.22785E+18	cart	2	33.18
2.22785E+18	purchase	1	16.59
2.22785E+18	view	35	547.65

Here we can see categories, event types and there total price and their total occurances.

Below are the factors which are affecting business basically:

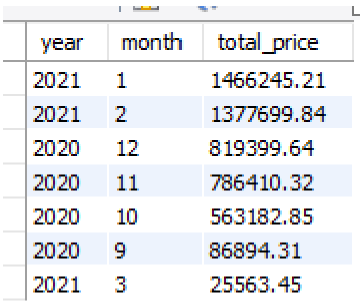
1)**Month of Sales** *: Here students are required to find months in decreasing order of sales and also highest sales in which month and year and lowest sales in which month and year with the help of SQL query*

Query:



```
-- Top months total sales i.e. where event is 'Purchase'
SELECT year(event_time) year, month(event_time) month, round(SUM(price),2)  total_price
FROM events
WHERE event_type = 'purchase'
GROUP BY year, month
ORDER BY total_price DESC;
```

Output:



year	month	total_price
2021	1	1466245.21
2021	2	1377699.84
2020	12	819399.64
2020	11	786410.32
2020	10	563182.85
2020	9	86894.31
2021	3	25563.45

Observations:

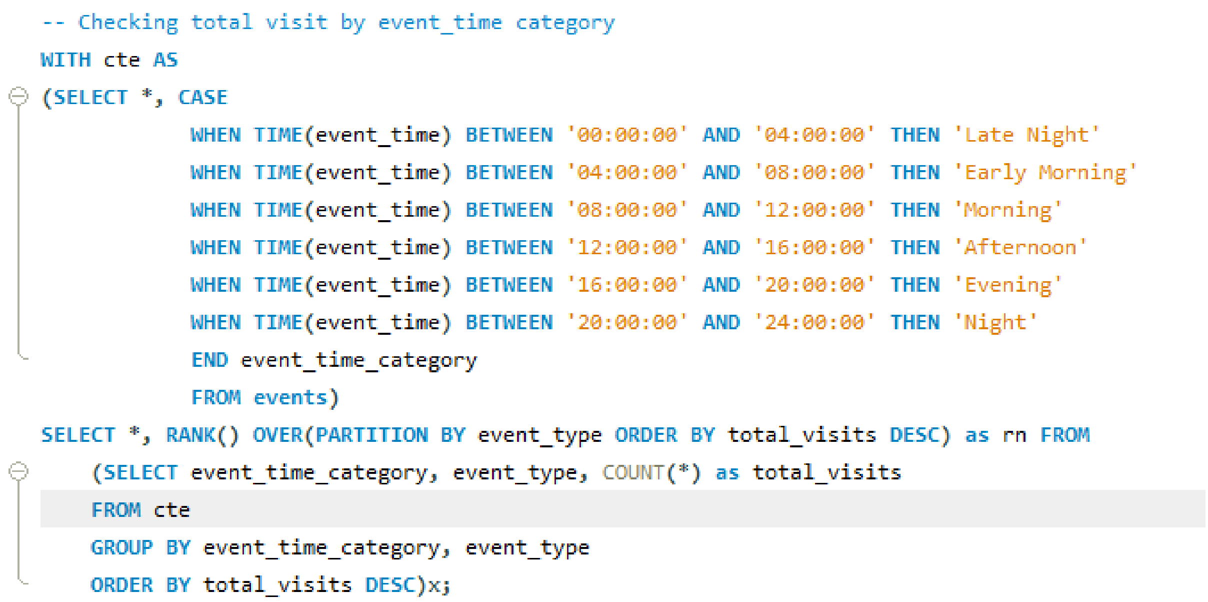
1. We have calculated total sales i.e. where user actually purchased the products
2. We have total 7 months of data
3. Highest sales is in 'January' of year '2021' followed by 'February' of year '2021
4. Lowest sales is in 'March' of year '2021' followed by 'september' of year '2020'

2)**Top Time of Visit**: Here Students are required to calculate the time for viewing, adding to cart and Purchase time with help of SQL query and need to confirm if mostly viewed and added to cart time is similar to Purchase time or not.

Here we need to check top time to for viewing , adding to cart, and purchase in short. We'll calculation top time where user is viewing , adding to cart, and purchase the product. We can calculation it with hour wise break down but it will not give us any proper insights so I have created bins/buckets of times with 4 hours each like follows.

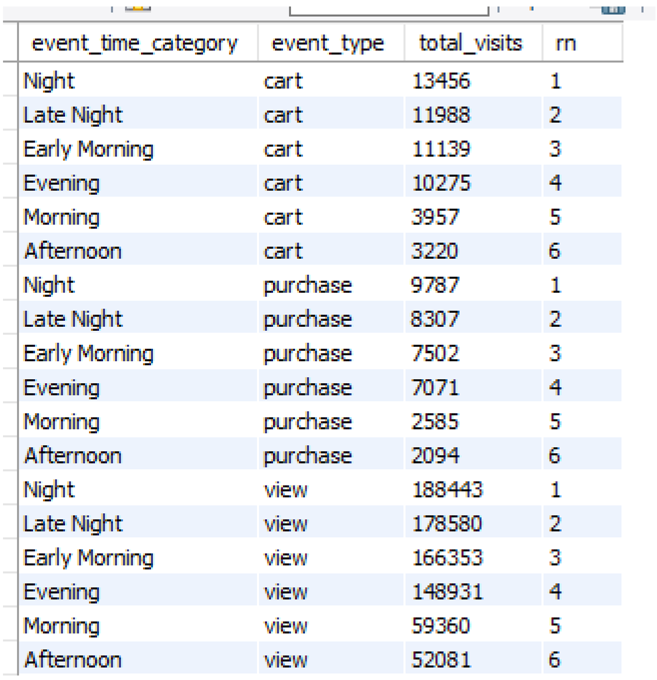
12am - 4am --> 'Late Night'
4am - 8am --> 'Early Morning'
8am - 12pm --> 'Morning'
12pm - 4pm --> 'Afternoon'
4pm - 8pm --> 'Evening'
8pm - 12am --> 'Night'

Query1: Here we are checking the total visit by event type in each time frame



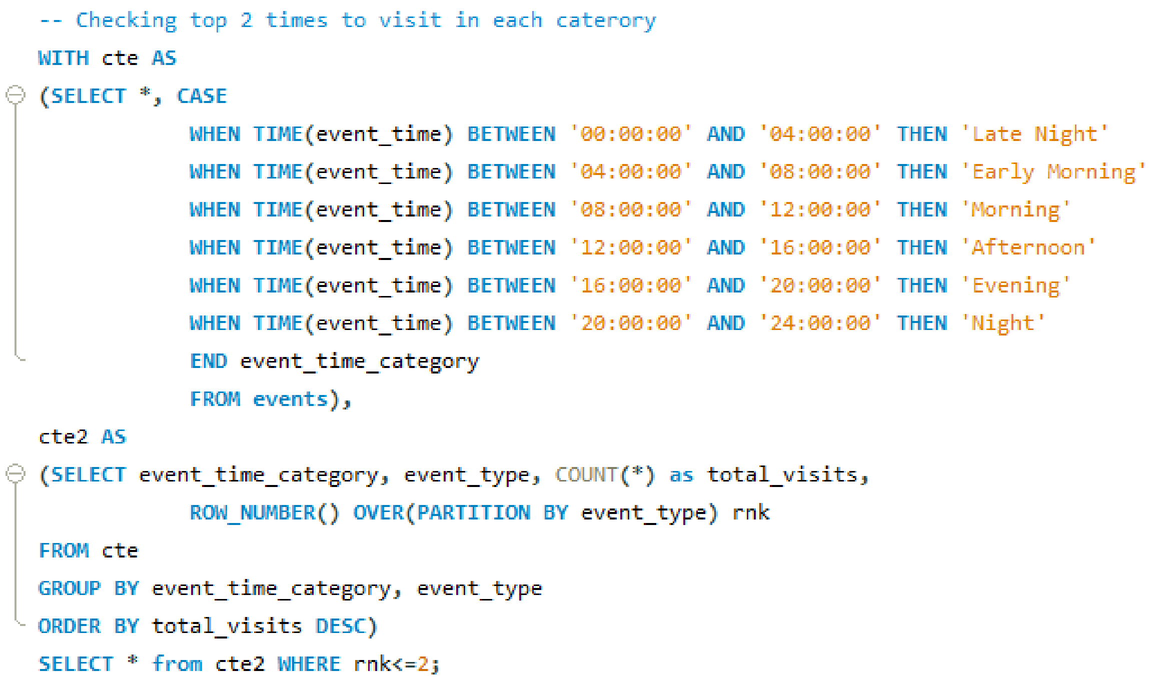
```
-- Checking total visit by event_time category
WITH cte AS
(SELECT *, CASE
            WHEN TIME(event_time) BETWEEN '00:00:00' AND '04:00:00' THEN 'Late Night'
            WHEN TIME(event_time) BETWEEN '04:00:00' AND '08:00:00' THEN 'Early Morning'
            WHEN TIME(event_time) BETWEEN '08:00:00' AND '12:00:00' THEN 'Morning'
            WHEN TIME(event_time) BETWEEN '12:00:00' AND '16:00:00' THEN 'Afternoon'
            WHEN TIME(event_time) BETWEEN '16:00:00' AND '20:00:00' THEN 'Evening'
            WHEN TIME(event_time) BETWEEN '20:00:00' AND '24:00:00' THEN 'Night'
            END event_time_category
            FROM events)
SELECT *, RANK() OVER(PARTITION BY event_type ORDER BY total_visits DESC) as rn FROM
    (SELECT event_time_category, event_type, COUNT(*) as total_visits
    FROM cte
    GROUP BY event_time_category, event_type
    ORDER BY total_visits DESC)x;
```

Output:



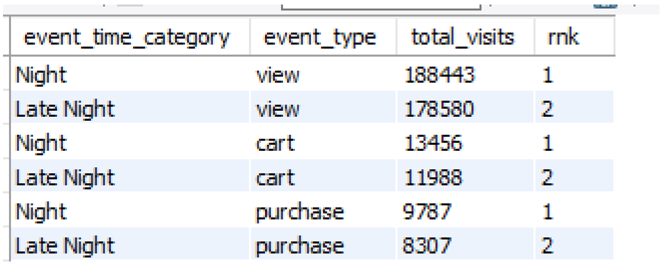
event_time_category	event_type	total_visits	rn
Night	cart	13456	1
Late Night	cart	11988	2
Early Morning	cart	11139	3
Evening	cart	10275	4
Morning	cart	3957	5
Afternoon	cart	3220	6
Night	purchase	9787	1
Late Night	purchase	8307	2
Early Morning	purchase	7502	3
Evening	purchase	7071	4
Morning	purchase	2585	5
Afternoon	purchase	2094	6
Night	view	188443	1
Late Night	view	178580	2
Early Morning	view	166353	3
Evening	view	148931	4
Morning	view	59360	5
Afternoon	view	52081	6

Query2: Here is the main Query where we have checked the top 2 event time category in each event type.



```
-- Checking top 2 times to visit in each caterory
WITH cte AS
(SELECT *, CASE
            WHEN TIME(event_time) BETWEEN '00:00:00' AND '04:00:00' THEN 'Late Night'
            WHEN TIME(event_time) BETWEEN '04:00:00' AND '08:00:00' THEN 'Early Morning'
            WHEN TIME(event_time) BETWEEN '08:00:00' AND '12:00:00' THEN 'Morning'
            WHEN TIME(event_time) BETWEEN '12:00:00' AND '16:00:00' THEN 'Afternoon'
            WHEN TIME(event_time) BETWEEN '16:00:00' AND '20:00:00' THEN 'Evening'
            WHEN TIME(event_time) BETWEEN '20:00:00' AND '24:00:00' THEN 'Night'
            END event_time_category
            FROM events),
cte2 AS
(SELECT event_time_category, event_type, COUNT(*) as total_visits,
            ROW_NUMBER() OVER(PARTITION BY event_type) rnk
FROM cte
GROUP BY event_time_category, event_type
ORDER BY total_visits DESC)
SELECT * from cte2 WHERE rnk<=2;
```

Output:



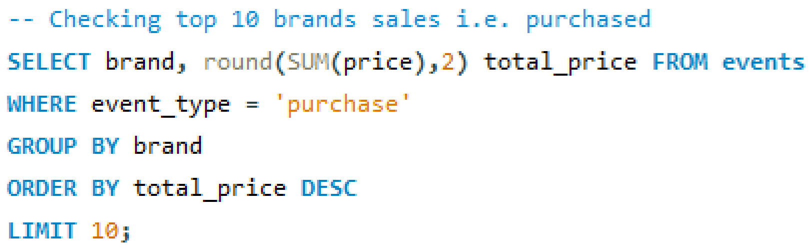
event_time_category	event_type	total_visits	rnk
Night	view	188443	1
Late Night	view	178580	2
Night	cart	13456	1
Late Night	cart	11988	2
Night	purchase	9787	1
Late Night	purchase	8307	2

Observations:

1. **Top visited of event time category** for every **event type** is **Night time** which is **8PM – 12AM**
2. **Second most visited event time category** for every **event type** is **Late Nigh**t which is **12AM – 4AM**
3. So Night and Late Night time category where we could target our users.

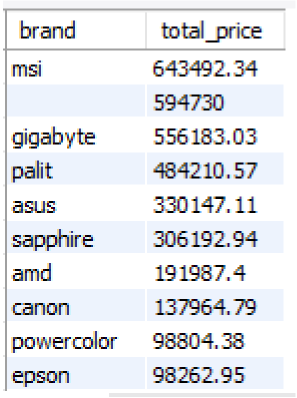
3)**Top brands by Sale**: Here Students are required to find Top 6 brand w.r.t Sales which is the Top Brand followed by other brand with the help of SQL query.

Query: Here we'll check top 10 brands by sales i.e. **event_type** is **purchase**



```
-- Checking top 10 brands sales i.e. purchased
SELECT brand, round(SUM(price),2) total_price FROM events
WHERE event_type = 'purchase'
GROUP BY brand
ORDER BY total_price DESC
LIMIT 10;
```

Output:



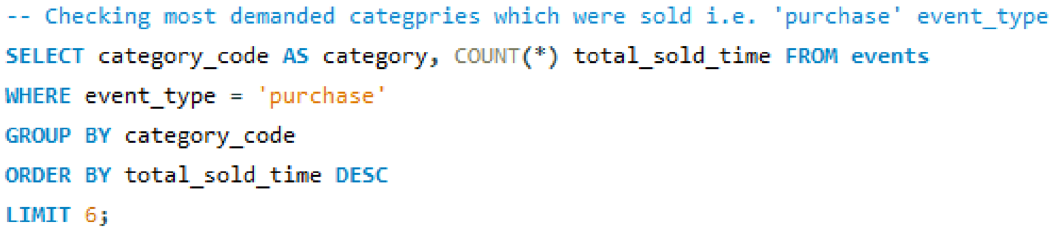
brand	total_price
msi	643492.34
	594730
gigabyte	556183.03
palit	484210.57
asus	330147.11
sapphire	306192.94
amd	191987.4
canon	137964.79
powercolor	98804.38
epson	98262.95

Observations:

1. Top selling brand is MSI.
2. Some of the selling products brand is not updated.
3. Most of the brand are of Electronics and Electronics Gadgets category.

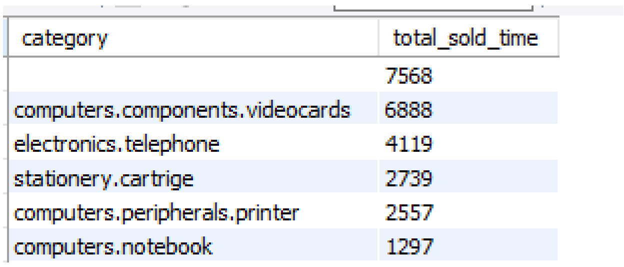
4)**Demand for Items:** Here Students are required to find Top 6 Category which was sold most number of times and should also show the count as well.

Query: Here We'll check top 6 selling Category codes



```
-- Checking most demanded categpries which were sold i.e. 'purchase' event_type
SELECT category_code AS category, COUNT(*) total_sold_time FROM events
WHERE event_type = 'purchase'
GROUP BY category_code
ORDER BY total_sold_time DESC
LIMIT 6;
```

Output:



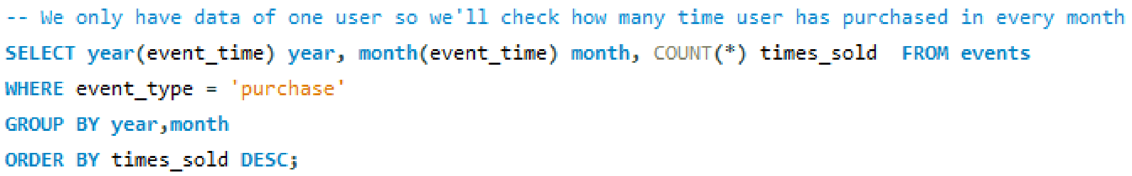
category	total_sold_time
	7568
computers.components.videocards	6888
electronics.telephone	4119
stationery.cartrige	2739
computers.peripherals.printer	2557
computers.notebook	1297

Observations:

1. Top selling category is code is not updated for most of the products.
2. Computer products are top selling.

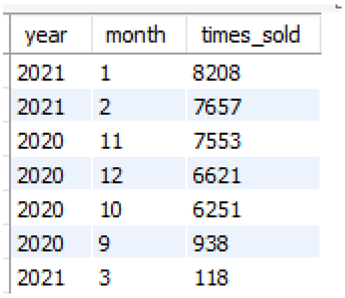
5)**Frequency of Purchase**- Here Students are required to find count of Users who has purchase at each time i.e. number of users purchasing once,twice,thrice etc once in a given time span of 5 months and also Maximum number of Times one person has purchased in a given time span of 5 months.

Query: The data is of six months only and user_id is also one. So here we'll check the Top purchased months. i.e. Monthly purchased count.



```
-- We only have data of one user so we'll check how many time user has purchased in every month
SELECT year(event_time) year, month(event_time) month, COUNT(*) times_sold  FROM events
WHERE event_type = 'purchase'
GROUP BY year,month
ORDER BY times_sold DESC;
```

Output:



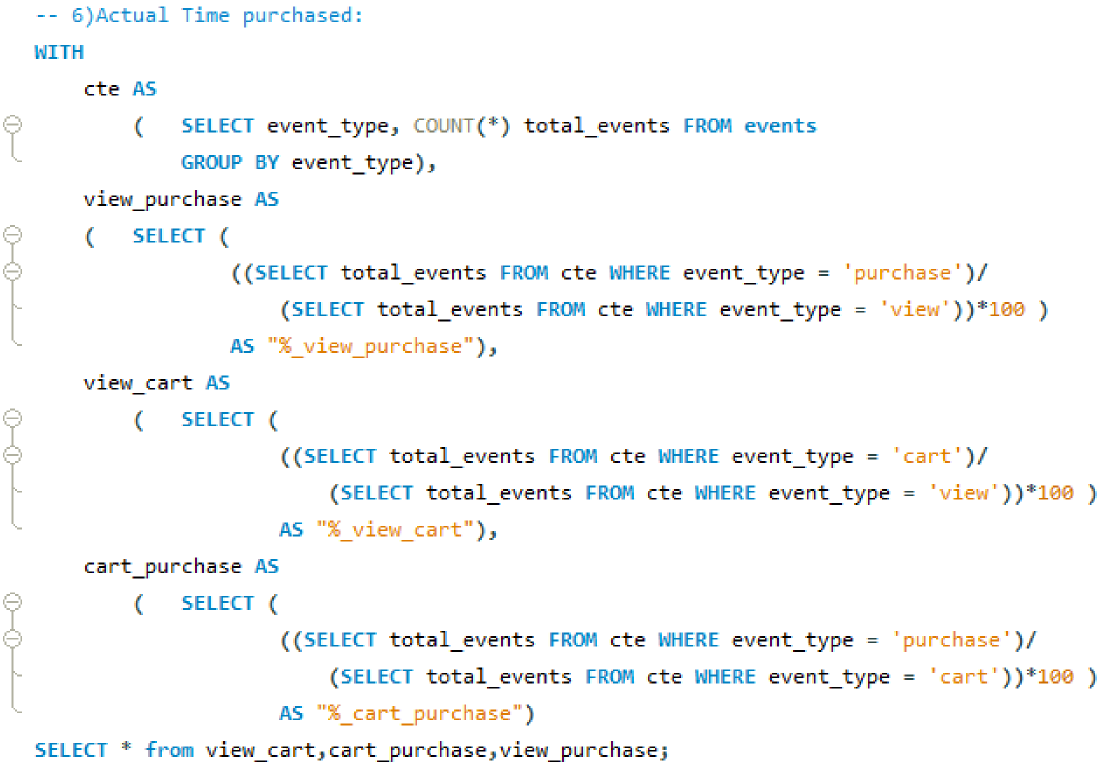
year	month	times_sold
2021	1	8208
2021	2	7657
2020	11	7553
2020	12	6621
2020	10	6251
2020	9	938
2021	3	118

Observations:

1. Mostly purchased month is January followed by February in year 2021
2. Least purchased month is March of 2021

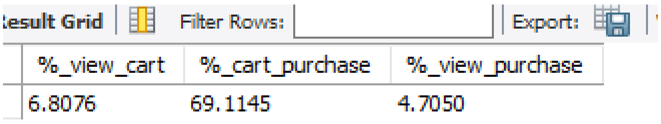
6)**Actual Time purchased** – Here Students have to find the number of times that the item has actually been purchased after the users have viewed the items i.e. the query should give result for actual number of times purchased.
Also query should be written to find the number of times the item has been viewed.

Query: Here we are considering that every product purchased is viewed added cart as well. So we'll check the percentage of product which were viewed are purchased and percentage of products which were purchased which were added to cart with percentage of products added to cart are also viewed.



```
-- 6)Actual Time purchased:
WITH
    cte AS
        (   SELECT event_type, COUNT(*) total_events FROM events
            GROUP BY event_type),
    view_purchase AS
    (   SELECT (
                ((SELECT total_events FROM cte WHERE event_type = 'purchase')/
                    (SELECT total_events FROM cte WHERE event_type = 'view'))*100 )
                AS "%_view_purchase"),
    view_cart AS
        (   SELECT (
                ((SELECT total_events FROM cte WHERE event_type = 'cart')/
                    (SELECT total_events FROM cte WHERE event_type = 'view'))*100 )
                AS "%_view_cart"),
    cart_purchase AS
        (   SELECT (
                ((SELECT total_events FROM cte WHERE event_type = 'purchase')/
                    (SELECT total_events FROM cte WHERE event_type = 'cart'))*100 )
                AS "%_cart_purchase")
SELECT * from view_cart,cart_purchase,view_purchase;
```

Output:



%_view_cart	%_cart_purchase	%_view_purchase
6.8076	69.1145	4.7050

Observations:

1. 6.81% of products which were viewed are added to cart.
2. 69.11% added to cart products are purchased by users.
3. 4.70% viewed products are purchased by users.

Overall Observations and Conclusions:

1. Top selling month is January followed by February so we can say that users tends buy new things at start of the year.
2. Least selling month is March which further proves our point.
3. So start or end of year good time for us to target users/customers.
4. Top time visit is Night followed by Late Night time which is 8PM to 4AM
5. Users tends to order more at relaxed time.
6. This is the time where we could really target our users/customers.
7. Most of top selling product have not updated there brand names, so it needs to updated to avoid future errors and to give convenience to users.
8. Most top selling products have not updated there category code, so it need to be updated so that users could search there required products easily and We could also organise our products more easily.
9. Most times sold month is January followed by February, so we can say that users tends to buy new things at the start of the year.
10. Least times sold month is March which further proves our point.
11. So start or end of year good time for us to target users/customers.
12. 6.81% of products which were viewed are added to cart and 69.11% added to cart products are purchased by users so users to buys most of times when they add it to cart.
13. Only 4.70% viewed products are purchased by users.
14. So we need to motivated users to add items to the cart buy giving some offers and deals.

Answer to problem_statement:

Despite the huge usage of ecommerce website a section of the society still prefers offline stores. Why?

Answer:

We Indians still prefers offline mode. India a developing country, which possess a large number of youth power & tremendous opportunity for world market . Recently mobile revolution have taken place in India & continuous growing in larger pace. In couple of few years things have changed many much , now these days one can see a low end or even a high end smartphone in the hand of a normal indian layman. A lots of telecom companies like Airtel, VI, JIO etc have covered almost every
corner of india with a net of telecommunication tower & cables. Now in these days of 4g era with 5g coming soon, online payments, shopping etc are not remain alien words for a normal Indian layman. These days lots of people from India making online transactions with the help of phone or laptop etc. But still a lots of people actually a very big segments of Indian populations doesn't believe in online world. They by no doubt prefer offline mode for financial & other transactions than online mode. Main reason behind this is lack of proper knowledge about internet & also fear of scam &

scam like sites etc. Reason behind same is lack of trust on online world & service providers. People from India still prefer to purchase from local market a local brand instead from a reputed online shopping site who may be offering a leading brand in affordable price with doorstep delivery. Indian prefer offline activity in place of online but things are changing .We are growing & learning. Later or sooner situation will improve & we will able to see a lots of Indian trusting online world.

Indians usually buy those things which they can first **see** and **touch** and then **bargain** according to their perception about the product. While buying online, one cannot do these things. Although the mindset is changing now and more and more people are trying online buying but still majority believes that it is better to get a feel before they buy anything.

Even if the market is now changing.