E-COMMERCE SALES ANALYSIS

“PRESENTATION BY TISHA GANDHI”

HELLO!

MY NAME IS TISHA GANDHI,
IN THIS PROJECT I HAVE UTILISED SQL QUERRIES TO ANSWER QUESTIONS RELATED TO E-COMMERCE SALES.

DATASET

e-Commerce (Target) Sales Dataset

Detailed Overview of Target's Brazilian Operations and Customer Data

kaggle.com

LIST ALL UNIQUE CITIES WHERE CUSTOMERS ARE LOCATED

```
select distinct customer_city from customers;
```

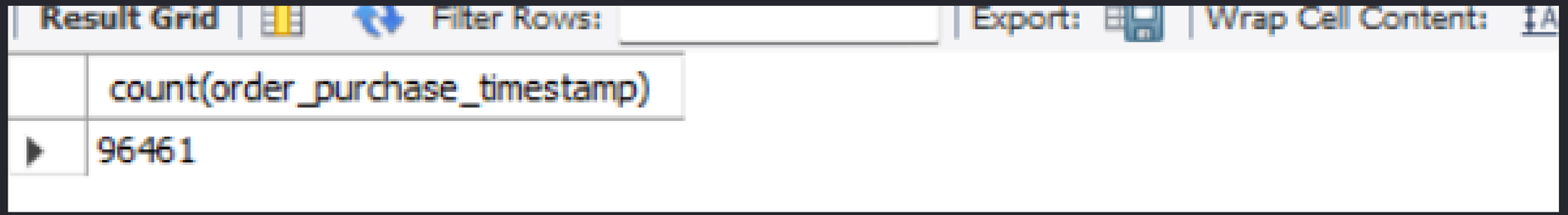
Result Grid | Filter

customer_city
franca
sao bernardo do campo
sao paulo
mogi das cruzes
campinas
jaragua do sul
timoteo
curitiba
belo horizonte
montes claros
rio de janeiro
lencois paulista
caxias do sul

customers 1

COUNT THE NUMBER OF ORDERS PLACED IN 2017

```
select count(order_purchase_timestamp) from orders;
```



FIND THE TOTAL SALES PER CATEGORY

```sql
SELECT
    p.product_category,
    ROUND(SUM(s.payment_value), 0) AS Total_sales
FROM
    products AS p
        JOIN
    order_items AS o ON o.product_id = p.product_id
        JOIN
    payments AS s ON o.order_id = s.order_id
GROUP BY p.product_category;
```

Result Grid | Filter Rows:

product_category	Total_sales
perfumery	506739
Furniture Decoration	1430176
telephony	486882
bed table bath	1712554
automotive	852294
computer accessories	1585330
housewares	1094758
babies	537885
toys	619038
Furniture office	646826
Cool Stuff	779698
HEALTH BEAUTY	1657373
pet Shop	311269

CALCULATE THE PERCENTAGE OF ORDERS THAT WERE PAID IN INSTALLMENTS

```sql
SELECT
    SUM(CASE
        WHEN payment_installments > 1 THEN 1
        ELSE 0
    END) / COUNT(payment_installments) * 100 AS installments
FROM
    payments;
```

Result Grid

	installments
▶	49.4176

COUNT THE NUMBER OF CUSTOMERS FROM EACH STATE

```
select customer_state,count(customer_id) as Total_Customers
from customers
group by customer_state;
```

Result Grid | Filter Rows:

customer_state	Total_Customers
SP	41746
SC	3637
MG	11635
PR	5045
RJ	12852
RS	5466
PA	975
GO	2020
ES	2033
BA	3380
MA	747
MS	715
CE	1336

CALCULATE THE NUMBER OF ORDERS PER MONTH IN 2018

```sql
SELECT
    MONTHNAME(order_purchase_timestamp) AS Month,
    COUNT(order_id) AS Order_count
FROM
    orders
WHERE
    YEAR(order_purchase_timestamp) = 2018
GROUP BY Month
ORDER BY Order_count;
```

Result Grid | Filter Rows:

Month	Order_count
June	6096
July	6156
August	6351
February	6556
May	6749
April	6798
March	7003
January	7069

FIND THE AVERAGE NUMBER OF PRODUCTS PER ORDER, GROUPED BY CUSTOMER CITY

```sql
with count_per_order as
(select o.customer_id,o.order_id,count(i.order_id) as OC
from orders as o join order_items as i
on o.order_id=i.order_id
GROUP BY o.order_id,o.customer_id)
SELECT
    c.customer_city, ROUND(AVG(co.OC), 2) AS Average_Order
FROM
    customers AS c
        JOIN
    count_per_order AS co ON co.customer_id = c.customer_id
GROUP BY c.customer_city;
```

Result Grid | Filter Rows:

customer_city	Average_Order
sao paulo	1.16
sao jose dos campos	1.14
indaial	1.12
treze tilias	1.27
rio de janeiro	1.15
mario campos	1.33
guariba	1.00
cuiaba	1.20
franca	1.25
tocos	1.00
januaria	1.18
campinas	1.16
embu-guacu	1.24

CALCULATE THE PERCENTAGE OF TOTAL REVENUE CONTRIBUTED BY EACH PRODUCT CATEGORY

```sql
SELECT
    products.product_category AS product_category,
    ROUND((SUM(payments.payment_value) / (SELECT
                    SUM(payment_value)
                FROM
                    payments)) * 100,2)
                AS total_revenue
FROM
    products
        JOIN
    order_items ON order_items.product_id = products.product_id
        JOIN
    payments ON payments.order_id = order_items.order_id
GROUP BY product_category
order by total_revenue desc;
```

Result Grid | Filter Rows:

product_category	total_revenue
bed table bath	10.7
HEALTH BEAUTY	10.35
computer accessories	9.9
Furniture Decoration	8.93
Watches present	8.93
sport leisure	8.7
housewares	6.84
automotive	5.32
Garden tools	5.24
Cool Stuff	4.87
Furniture office	4.04
toys	3.87
babies	3.36

IDENTIFY THE CORRELATION BETWEEN PRODUCT PRICE AND THE NUMBER OF TIMES A PRODUCT HAS BEEN PURCHASED

```sql
SELECT
    p.product_category,
    COUNT(o.product_id) AS Count_of_Orders,
    ROUND(AVG(o.price), 2) AS Avg_Price
FROM
    products AS p
        JOIN
    order_items AS o ON p.product_id = o.product_id
GROUP BY p.product_category
ORDER BY Count_of_Orders DESC;
```

Result Grid | Filter Rows: | Expo

product_category	Count_of_Orders	Avg_Price
bed table bath	11115	93.3
HEALTH BEAUTY	9670	130.16
sport leisure	8641	114.34
Furniture Decoration	8334	87.56
computer accessories	7827	116.51
housewares	6964	90.79
Watches present	5991	201.14
telephony	4545	71.21
Garden tools	4347	111.63
automotive	4235	139.96
toys	4117	117.55
Cool Stuff	3796	167.36
perfumery	3419	116.74

CALCULATE THE TOTAL REVENUE GENERATED BY EACH SELLER, AND RANK THEM BY REVENUE

```sql
select * ,
dense_rank() over (order by Total_Revenue desc) as 'Dense_Rank'
from
(select o.seller_id,round(sum(p.payment_value),0) as Total_Revenue
from
order_items as o join payments as p
on o.order_id = p.order_id
group by o.seller_id) as a;
```

Result Grid | Filter Rows: | Export: | Wrap

seller_id	Total_Revenue	Dense_Rank
7c67e1448b00f6e969d365cea6b010ab	507167	1
1025f0e2d44d7041d6cf58b6550e0bfa	308222	2
4a3ca9315b744ce9f8e9374361493884	301245	3
1f50f920176fa81dab994f9023523100	290253	4
53243585a1d6dc2643021fd1853d8905	284903	5
da8622b14eb17ae2831f4ac5b9dab84a	272219	6
4869f7a5dfa277a7dca6462dcf3b52b2	264166	7
955fee9216a65b617aa5c0531780ce60	236322	8
fa1c13f2614d7b5c4749cbc52fecda94	206513	9
7e93a43ef30c4f03f38b393420bc753a	185134	10
6560211a19b47992c3666cc44a7e94c0	179658	11
7a67c85e85bb2ce8582c35f2203ad736	169031	12
25c5c91f63607446a97b143d2d535d31	160535	13

CALCULATE THE MOVING AVERAGE OF ORDER VALUES FOR EACH CUSTOMER OVER THEIR ORDER HISTORY

```sql
select customer_id,order_purchase_timestamp,payment,
Avg(payment) over(partition by customer_id
order by order_purchase_timestamp
rows between 2 preceding and current row) as Moving_Avg
from
(select o.customer_id,o.order_purchase_timestamp,p.payment_value as payment
from orders as o join payments as p
on o.order_id=p.order_id) as a;
```

Result Grid | Filter Rows: | Export: | Wrap Cell Content:

customer_id	order_purchase_timestamp	payment	Moving_Avg
00012a2ce6f8dcda20d059ce98491703	2017-11-14 16:08:00	114.74	114.73999786376953
000161a058600d5901f007fab4c27140	2017-07-16 09:40:00	67.41	67.41000366210938
0001fd6190edaaf884bcaf3d49edf079	2017-02-28 11:06:00	195.42	195.4199981689453
0002414f95344307404f0ace7a26f1d5	2017-08-16 13:09:00	179.35	179.35000610351562
000379cdec625522490c315e70c7a9fb	2018-04-02 13:42:00	107.01	107.01000213623047
0004164d20a9e969af783496f3408652	2017-04-12 08:35:00	71.8	71.80000305175781
000419c5494106c306a97b5635748086	2018-03-02 17:47:00	49.4	49.400001525878906
00046a560d407e99b969756e0b10f282	2017-12-18 11:08:00	166.59	166.58999633789062
00050bf6e01e69d5c0fd612f1bcfb69c	2017-09-17 16:04:00	85.23	85.2300033569336
000598caf2ef4117407665ac33275130	2018-08-11 12:14:00	1255.71	1255.7099609375
0005aefbb696d34b3424dccd0a0e9fd0	2018-06-20 09:46:00	147.33	147.3300018310547
00062b33cb9f6fe976afdcff967ea74d	2017-03-15 23:44:00	58.95	58.95000076293945
00066ccbe787a588c52bd5ff404590e3	2018-02-06 16:10:00	270	270

CALCULATE THE CUMULATIVE SALES PER MONTH FOR EACH YEAR

```
select years,months,sum(payment)
over (order by years,months) as Cumulative_sales
from
(select year(o.order_purchase_timestamp) as Years,
monthname( o.order_purchase_timestamp) as Months,
round(sum(p.payment_value),2) as payment
from orders as o join payments as p on o.order_id=p.order_id
group by years,months ) as a;
```

Result Grid | Filter Rows:

years	months	Cumulative_sales
2016	December	19.62
2016	October	47290.82
2017	April	438243
2017	August	1084243.6099999999
2017	December	1927442.7799999998
2017	February	2196901.76
2017	January	2324332.5
2017	July	2890736.43
2017	June	3380962.0300000003
2017	March	3795331.4200000004
2017	May	4362204.15
2017	November	5515597.37
2017	October	6266737.640000001

CALCULATE THE YEAR-OVER-YEAR GROWTH RATE OF TOTAL SALES

```sql
with a as (select year(o.order_purchase_timestamp) as Years,
round(sum(p.payment_value),2) as payment
from orders as o join payments as p
on o.order_id=p.order_id
group by years)
select years, ((payment - lag(payment) over(order by years))/
lag(payment) over(order by years)) * 100 as YOY_Growth_Rate from a;
```

Result Grid | Filter Rows:

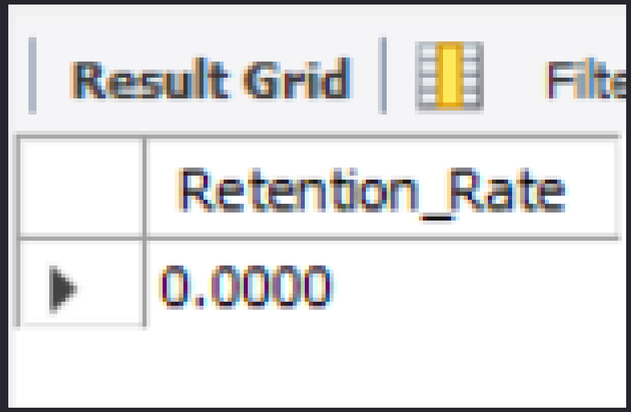
years	YOY_Growth_Rate
2016	NULL
2017	14533.755198154737
2018	22.130820726166515

CALCULATE THE RETENTION RATE OF CUSTOMERS, DEFINED AS THE PERCENTAGE OF CUSTOMERS #WHO MAKE ANOTHER PURCHASE WITHIN 6 MONTHS OF THEIR FIRST PURCHASE

```sql
# step 1 ---- find first_order date of each customer
with a as (select c.customer_id,min(o.order_purchase_timestamp) as first_order
from orders as o
join customers as c on o.customer_id=c.customer_id group by c.customer_id),

# step 2 ---- find repeat customers from a who placed another order
# after their first, but within 6 month.
b as (select a.customer_id,count(distinct o.order_purchase_timestamp) as next_order
from a join orders as o
on a.customer_id=o.customer_id
where o.order_purchase_timestamp > first_order
and o.order_purchase_timestamp < date_add(first_order,interval 6 month)
group by a.customer_id)

# step 3 ---- calculate retention rate of customer
select 100*(count(distinct b.customer_id)/count(distinct a.customer_id)) Retention_Rate
# returning customers (b) / total customers (a) * 100
from a left join b
on a.customer_id=b.customer_id;
```



IDENTIFY THE TOP 3 CUSTOMERS WHO SPENT THE MOST MONEY IN EACH YEAR

```
select years,customer_id,Most_Money_Spent,d_Rank
from (select year(o.order_purchase_timestamp) as Years,o.customer_id,
round(sum(p.payment_value),0) as Most_Money_Spent,
dense_rank() over (partition by year(o.order_purchase_timestamp)
order by sum(p.payment_value) desc) as 'd_Rank'
from orders as o join payments as p
on  o.order_id = p.order_id
group by year(o.order_purchase_timestamp),o.customer_id) as d
where d_rank <= 3 ;
```

Result Grid | Filter Rows: | Export: | Wrap Cell Content

years	customer_id	Most_Money_Spent	d_Rank
2016	a9dc96b027d1252bbac0a9b72d837fc6	1424	1
2016	1fc56719b52f82c03caddc5faf531fbb	982	2
2016	85f0e92957e9fb9c5f72ba5378f492a0	980	3
2017	1617b1357756262bfa56ab541c47bc16	13664	1
2017	c6e2731c5b391845f6800c97401a43a9	6929	2
2017	3fd6777bbce08a352fddd04e4a7cc8f6	6727	3
2018	ec5b2ba62e574342386871631fafd3fc	7275	1
2018	f48d464a0baaea338cb25f816991ab1f	6922	2
2018	3d979689f636322c62418b6346b1c6d2	4682	3

THANK YOU