E-COMMERCE

PROJECT USING SQL

I AM KAMAL AND IN THIS PROJECT I UTILIZED SQL QUERIES TO SOLVE THE SALES BASED QUESTIONS.

Questions

- List all unique cities where customers are located.
- Count the number of orders placed in 2017.
- Find the total sales per category.
- Calculate the percentage of orders that were paid in installments.
- Count the number of customers from each state.

Questions

- Calculate the number of orders per month in 2018.
- Find the average number of products per order, grouped by customer city.
- Calculate the percentage of total revenue contributed by each product category.
- Identify the correlation between product price and the number of times a product has been purchased.
- Calculate the total revenue generated by each seller, and rank them by revenue.

- **List all unique cities where customers are located.**

```sql
-- 1. List all unique cities where customers are located.

SELECT DISTINCT
    (customer_city) 'Unique_city'
FROM
    customers
```

output

Unique_city
franca
sao bernardo do campo
sao paulo
mogi das cruzes
campinas
jaragua do sul
timoteo
curitiba
belo horizonte

- Count the number of orders placed in 2017.

```sql
SELECT
    COUNT(*) AS order_count
FROM
    orders
WHERE
    YEAR(order_approved_at) = 2017;
```

output

Result Grid | Filter Rows:

order_count
44973

- **Find the total sales per category.**

```sql
-- Find the total sales per category

SELECT pd.product_category, round(sum(p.payment_value)) AS total_sales
FROM payments p
join order_items ot on ot.order_id = p.order_id
join products pd on pd.product_id = ot.product_id
group by pd.product_category
```

output

product_category	total_sales
perfumery	506739
Furniture Decoration	1430176
telephony	486882
bed table bath	1712554
automotive	852294
computer accessories	1585330
housewares	1094758
babies	539846
toys	619038
Furniture office	646826

- **Calculate the percentage of orders that were paid in installments.**

```sql
SELECT
    (COUNT(CASE
        WHEN payment_type = 'Installments' THEN 1
    END) * 100.0 / COUNT(*)) AS installment_percentage
FROM
    payments;
```

output

Result Grid | Filter Rows:

installment_percentage
0.00000

- **Count the number of customers from each state.**

```sql
-- Count the number of customers from each state.

select customer_state ,
count(customer_id) 'total customer'from customers
group by customer_state
```

output

customer_state	total customer
CE	1336
DF	2140
RN	485
PE	1652
MT	907
AM	148
AP	68
AL	413
RO	253
PB	536

- **Calculate the number of orders per month in 2018.**

```sql
-- Calculate the number of orders per month in 2018.

SELECT MONTH(order_approved_at) AS month,
 COUNT(*) AS order_count
FROM orders
WHERE YEAR(order_approved_at) = 2018
GROUP BY MONTH(order_approved_at)
ORDER BY month;
```

output

Result Grid | Filter Rows: | Expo

month	order_count
1	7187
2	6706
3	7288
4	6778
5	7066
6	6164
7	6176
8	6620

- **Find the average number of products per order, grouped by customer city.**

```sql
-- Find the average number of
-- products per order, grouped by customer city.

SELECT customer_city,
 AVG(od.order_id) AS avg_products_per_order
FROM customers c
join orders od on od.customer_id = c.customer_id
GROUP BY customer_city;
```

output

Result Grid | Filter Rows: | Export: | Wrap

customer_city	avg_products_per_order
vianopolis	15948.666666666666
ouro preto	0
goiania	0
feira de santana	3.297297297297297e267
sao paulo	0
cruz das almas	9.987184082568421e306
tocos	8.988465674311579e307
capelinha	2377.9
franca	0
guaruja	0
januaria	8.171332431192344e306

- **Calculate the percentage of total revenue contributed by each product category.**

```sql
-- Calculate the percentage of total revenue contributed by each product category

SELECT pd.product_category,
       round(SUM(p.payment_value))AS total_category_revenue,
       round((SUM(p.payment_value) * 100.0 / total_revenue.total)) AS revenue_percentage
FROM payments p
JOIN order_items ot ON p.order_id = ot.order_id
JOIN products pd ON pd.product_id = ot.product_id
CROSS JOIN (SELECT SUM(payment_value) AS total FROM payments) total_revenue
GROUP BY pd.product_category, total_revenue.total;
```

output

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

product_category	total_category_revenue	revenue_percentage
perfumery	506739	3
Furniture Decoration	1430176	9
telephony	486882	3
bed table bath	1712554	11
automotive	852294	5
computer accessories	1585330	10
housewares	1094758	7
babies	539846	3
toys	619038	4
Furniture office	646826	4

- **Identify the correlation between product price and the number of times a product has been purchased.**

```sql
-- Calculate the total revenue generated by each seller, and rank them by revenue

SELECT
    s.seller_id,
    SUM(p.payment_value) AS total_revenue,
    RANK() OVER (ORDER BY SUM(p.payment_value) DESC) AS revenue_rank
FROM payments p
join order_items ot on ot.order_id = p.order_id
join sellers s on s.seller_id = ot.seller_id
GROUP BY s.seller_id
ORDER BY total_revenue DESC;
```

output

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

seller_id	total_revenue	revenue_rank
7c67e1448b00f6e969d365cea6b010ab	507166.9073021412	1
1025f0e2d44d7041d6cf58b6550e0bfa	308222.0398402214	2
4a3ca9315b744ce9f8e9374361493884	301245.26976528764	3
1f50f920176fa81dab994f9023523100	290253.42012761533	4
53243585a1d6dc2643021fd1853d8905	284903.0804977417	5
da8622b14eb17ae2831f4ac5b9dab84a	272219.31931465864	6
4869f7a5dfa277a7dca6462dcf3b52b2	264166.1209387779	7
955fee9216a65b617aa5c0531780ce60	236322.30050226487	8
fa1c13f2614d7b5c4749cbc52fecda94	206513.22986984253	9
[illegible]	[illegible]	10

- **Calculate the total revenue generated by each seller, and rank them by revenue.**

```sql
-- Identify the correlation between product price and the number
-- of times a product has been purchased.

SELECT CORR(pd.price, purchase_count) AS price_purchase_correlation
FROM (
    SELECT
        oi.product_id,
        COUNT(oi.order_id) AS purchase_count
    FROM order_items oi
    GROUP BY oi.product_id
) AS purchase_data
JOIN products pd ON pd.product_id = purchase_data.product_id;
```