FINAL CAPSTONE PROJECT

BY ANSHUL MOURYA

1.Create a data dictionary for the table with the best of your understanding

Column Name	Data Type	Description
Order_id	Integer	Unique identifier for each order
Cuatomer_id	Integer	Identifier for the user who placed the order
Order_date	Varchar	Date when the order was placed
Order_status	Varchar	Status of the order (e.g., Cancelled, Delivered)
Payment_method	Varchar	Method used for payment (e.g., Debit Card, Credit Card)
Quantity	Integer	Quantity of products ordered
Order_rating	Integer	Rating given by the user for the order
Product_category	Varchar	Category of the ordered product (e.g., Clothing, Electronics)
Product_portfolio	Varchar	Portfolio of the ordered product (e.g., Gadgets, Women Fashion)
Product_id	Integer	Identifier for the ordered product
Gender	Varchar	Gender of the user (e.g., Female, Male)
Supplier_id	Integer	Identifier for the supplier of the product
Order_timestamp	Varchar	Date and time when the order was placed
Discount_percentage	Varchar	Percentage of discount applied to the order
Offer_name	Varchar	Name of the offer applied to the order
List_price	Integer	Original price of the product before any discount
Sale_price	Integer	Final price of the product after applying discounts
Delivery_date	Date	Expected delivery date of the order
GMV	Integer	is the total amount of sales a company makes over a specified period of time,

1.Write an SQL query to find the count of users in each order bucket based on their order history. The order buckets are categorized as 'No_orders' for customers with no delivered orders, '1_to_5_orders' for those with 1 to 5 delivered orders, '6_to_10_orders' for 6 to 10 delivered orders, and '>=10_orders' for customers with more than 10 delivered orders. The result should display the order bucket and the corresponding count of users.

```
with bucket as
(
        select  customer_id,
        case when count(distinct order_id) filter (where order_status_name = 'delivered') = 0 then 'No_orders'
        when count(distinct order_id) filter (where order_status_name = 'delivered') <= 5 then '1_to_5_orders'
        when count(distinct order_id) filter (where order_status_name = 'delivered') <=10 then '6_to_10_orders'
        else '>=10_orders' end as order_bucket
from    orders
group   by 1
)
select   order_bucket,
         count(distinct customer_id) as count_of_users
from     bucket b
group   by 1
```

order_bucket	count_of_users
1_to_5_orders	942
6_to_10_orders	318

Fetch the below metrics for month on month comparison of:
1. Total Revenue,2.Avg Order Value,3. Total Orders,4.Avg ProcessingTime

```
1 Select    extract(month from Order_date :: date) as month,
2           sum(Sale_price) AS total_revenue,
3           round(avg(Sale_price),2) as avg_order_value,
4           count(Order_id) as total_orders,
5           round(avg(delivery_date :: date-order_date :: date),2) as
            avg_processing_time
6 FROM     orders
7 GROUP    BY 1
8 ORDER    BY 1
```

month	total_revenue	avg_order_value	total_orders	avg_processing_time
1	403562	236.00	1710	6.93
2	378364	237.22	1595	6.98

2(5). Top 10 highest selling products

```
with top10 as
(
select  extract(month from order_date :: date)as month,product_id,sum(quantity) as total_sold_product
from    orders
where   order_status_name = 'delivered'
group   by 1,2
),
ranking as
(
  select  product_id,month,total_sold_product,
          row_number() over(partition by month order by total_sold_product desc) as rank
  from   top10
)
select  a.product_id,a.month,a.total_sold_product
from    ranking a
where   rank<=10
order   by 2
```

product_id	month	total_sold_product
8360	1	10
8700	1	10
3874	1	10

2.(6)Bucket DIstribution of different ratings and count of orders in it

```sql
select  order_rating,
        extract(month from order_date :: date) as month,
        count(customer_id) as total_ratings
from    orders
group   by 1,2
order   by 1,2
```

order_rating	month	total_ratings
1	1	346
1	2	329
1	3	333
1	4	353
1	5	319
1	6	311
2	1	363
2	2	304

2.(7)Sales by Product Category

```
select  extract(month from order_date :: date) as month,
        product_category,sum(sale_price) as total_sales
from    orders
group   by 1,2
order   by 1
```

month	product_category	total_sales
1	Electronics	67242
1	Home Goods	68300
1	Books	58988
1	Sports Gear	69194
1	Toys	66116
1	Apparel	73722
2	Apparel	63779
2	Electronics	59178

2(8).Sales and Orders by Gender

```sql
with male as(
select  extract(month from order_date :: date) as month,
        count(order_id) as male_order,
        sum(sale_price) as total_price
from    orders
where   Gender ilike 'male'
group   by 1
),
female as(
  select  extract( month from order_date :: date) as month,
          count(order_id) as female_order,
          sum(sale_price) as total_price
  from    orders
  where   Gender ilike 'female'
    group   by 1
)
select *
from   female f
full outer  join male m
on      f.month = m.month
```

month	female_order	total_price	month	male_order	total_price
1	786	182133	1	924	221429
2	757	181697	2	838	196667

5.Calculate the return rate for each product category. The return rate is defined as the percentage of orders that have been returned out of the total delivered orders for each product category.

```
WITH DeliveredOrders as (
                        select  product_category,
                                count(Customer_id) AS total_delivered
                        from   orders
                        where  order_status_name ilike 'delivered'
                        group  by 1

),
ReturnedOrders as (
                 select   product_category,
                          count(Customer_id) AS total_returned
                 from    orders
                 where   order_status_name ilike 'return%'
                 group   by 1
)
select  r.product_category,
          round(r.total_returned*100/d.total_delivered,2) as return_rate
from   DeliveredOrders d,ReturnedOrders r
limit   6
```

product_category	return_rate
Apparel	30.00
Books	25.00

6.Write a SQL query to fetch only those customers who placed an order but cancelled the same orders more than once.

```
Select  customer_id,
        count(distinct order_id) as cancelled_orders_count
From    orders
Where   order_status_name ilike '%cancel%'
group   by customer_id
having  count(distinct order_id) > 1;
```

customer_id	cancelled_orders_count
1	3
4	2
15	3

7.Write a query to find what is the Average delivery time for orders placed on weekdays versus weekends

```
with week as( Select   o.Order_date,extract(dow from order_date :: date)in (0,6)as days,o.delivery_date
              from   orders o
              limit  7000
 ),
weekend as(   select  a.Order_date,a.delivery_date
              from    week a
              where   days=true
),
weekdays as( select  e.Order_date,e.delivery_date
                from    week e
                where   days=false
)
select  distinct round(avg(c.delivery_date :: date-c.Order_date::date),2)as avg_delivery_time_weekend,
        round(avg(b.delivery_date :: date-b.Order_date ::date),2)as avg_delivery_time_weekdays
from    weekend c,weekdays b
```

Results

Query #1 **Execution time: 19085.61ms**

avg_delivery_time_weekend	avg_delivery_time_weekdays
7.10	6.99

8.Write a query to Determine the percentage contribution of each supplier to the total quantity of products sold in different product categories

```
with each_supplier as (
    select  supplier_id, product_category, sum(quantity) as total_quantity
    from        orders
    group       by 1,2
    ),
total_quant as (
          select sum(quantity) as total
           from orders
           )

select  supplier_id, product_category, (total_quantity*100/total) as   total_percentage_contribution
from            each_supplier, total_quant;
```

9.Write a query to Calculate the running total of orders delivered over time to identify trends in the order delivery volume.

```
select  order_date ::date,order_id,sale_price,
        sum(sale_price) over (partition by order_date order  by order_id) as running_total
From    orders
Where   order_status_name = 'delivered'
Order   by 1
```

order_date	order_id	sale_price	running_total
2022-01-01T00:00:00.000Z	696	413	413
2022-01-01T00:00:00.000Z	1147	468	881
2022-01-01T00:00:00.000Z	1671	321	1202
2022-01-01T00:00:00.000Z	1821	194	1396
2022-01-01T00:00:00.000Z	2045	181	1577
2022-01-01T00:00:00.000Z	2503	108	1685
2022-01-01T00:00:00.000Z	3147	224	1909
2022-01-01T00:00:00.000Z	3482	447	2356
2022-01-01T00:00:00.000Z	3716	378	2734
2022-01-01T00:00:00.000Z	3819	222	2956
2022-01-01T00:00:00.000Z	3851	106	3062

10.(A)Evaluate the distribution of orders delivered based on gender to understand any gender-related trends

```
select  count(order_id) as total_orders,gender
from    orders
where   order_status_name ='delivered'
group   by 2
```

total_orders	gender
2708	Female
3241	Male

10.(B)Day over Day trend of GMV

```
select  extract(month from order_date :: date) as
        extract(day from order_date :: date) as da
        sum(GMV) as GMV
from    orders
group   by 1,2
order   by 1,2
```

month	day	gmv
1	1	37135
1	2	40228
1	3	32333
1	4	26517