JENSON USA

Presented by Amruta Bankar

BACKGROUND

As a data analyst at Jensen's, craft SQL queries to derive insights on customer behavior, staff performance, inventory management, and store operations.

QUESTIONS

1.FIND THE TOTAL NUMBER OF PRODUCTS SOLD BY EACH STORE ALONG WITH THE STORE NAME

```sql
# 1.Find the total number of products sold by each store along with the store name

Select
stores.store_name, Sum(order_items.quantity) total_ordered_products
from stores
join orders
on stores.store_id = orders.store_id
join order_items
on order_items.order_id = orders.order_id
group by stores.store_id;
```

2.CALCULATE THE CUMULATIVE SUM OF QUANTITIES SOLD FOR EACH PRODUCT OVER TIME

```sql
# 2.Calculate the cumulative sum of quantities sold for each product over time.

with a as (select products.product_name,orders.order_date,sum(order_items.quantity) total_orders
from products
join order_items
on products.product_id = order_items.product_id
join orders
on orders.order_id = order_items.order_id
group by products.product_name,orders.order_date)

select *, sum(total_orders)
over (partition by product_name order by order_date) total_quantity from a;
```

3.FIND THE PRODUCT WITH THE HIGHEST TOTAL SALES (QUANTITY * PRICE) FOR EACH CATEGORY.

```
with a as (select categories.category_name,products.product_name,
sum(order_items.quantity*order_items.list_price)Sales
from order_items join products on order_items.product_id = products.product_id
join categories on categories.category_id = products.product_id
group by categories.category_name,products.product_name)

select * from (select *,rank() over(partition by category_name order by Sales desc)rnk from a)
where rnk = 1;
```

4.FIND THE CUSTOMER WHO SPENT THE MOST MONEY ON ORDERS

```sql
select orders.customer_id,
sum(order_items.quantity*order_items.list_price)sales
from orders join order_items on orders.order_id = order_items.order_id
group by orders.customer_id
order by sales desc
limit 1;
```

5.FIND THE HIGHEST-PRICED PRODUCT FOR EACH CATEGORY NAME

```sql
with product_rnk as (select c.category_id,c.category_name,p.product_id,p.product_name,p.list_price,
rank() over(partition by c.category_id order by p.list_price desc) as product_rnk
From products p
join categories c using(category_id))

select category_id,category_name,product_id,product_name,list_price
from product_rnk
where product_rnk = 1;
```

6.FIND THE TOTAL NUMBER OF ORDERS PLACED BY EACH CUSTOMER PER STORE.

```
59 •    select customers.customer_id,
60      customers.first_name,
61      customers.last_name,
62      stores.store_id,
63      stores.store_name,
64      count(*) as num_orders
65      from orders join customers using (customer_id)
66      join stores on orders.store_id = stores.store_id
67      group by customers.customer_id,customers.first_name,customers.last_name,
68      stores.store_id,stores.store_name
69      order by customer_id;
```

7.FIND THE NAMES OF STAFF MEMBERS WHO HAVE NOT MADE ANY SALES.

```
select staff_id,first_name,last_name
from staffs
where staff_id not in (select distinct staff_id
from orders);
```

8.FIND THE TOP 3 MOST SOLD PRODUCTS IN TERMS OF QUANTITY.

```
78 •    select p.product_id,product_name, sum(quantity) total_quantity
79      from order_items o
80      join products p using (product_id)
81      group by product_id
82      order by total_quantity DESC
83      limit 3;
```

9.FIND THE MEDIAN VALUE OF THE PRICE LIST.

```sql
with it_price as (select list_price,row_number() over(order by list_price) as price_rank,
count(*) over() as total_count from products)
select case when total_count % 2 = 0 then
(select avg(list_price)
from it_price where price_rank in (total_count/2,total_count/2+1))
else(select list_price
from it_price where price_rank = (total_count + 1)/2)
end as median
from it_price
limit 1;
```

10.LIST ALL PRODUCTS THAT HAVE NEVER BEEN ORDERED.(USE EXISTS)

```
100 •   select p.product_id,product_name
101     from products p
102     where
103     not exists(select 1
104     from order_items oi
105     where oi.product_id = p.product_id);
```

11.LIST THE NAMES OF STAFF MEMBERS WHO HAVE MADE MORE SALES THAN THE AVERAGE NUMBER OF SALES BY ALL STAFF MEMBERS.

```
109 • with staff_sales as (select staff_id, sum(quantity * list_price) as sales
110     from orders o
111     join order_items as oi using(order_id)
112     group by staff_id order by sales desc)
113     select staffs.staff_id,first_name,last_name,sales
114     from staff_sales
115     join staffs using(staff_id)
116   where sales > (select avg(sales)
117     from staff_sales);
```

12.THE CUSTOMERS WHO HAVE ORDERED ALL TYPES OF PRODUCTS (I.E., FROM EVERY CATEGORY IDENTIFY).

```
select orders.customer_id, customers.first_name,customers.last_name
from order_items join orders using(order_id)
join products on order_items.product_id = products.product_id
join customers on orders.customer_id = customers.customer_id
group by orders.customer_id,customers.first_name,customers.last_name
having count(distinct products.category_id) = (select count(category_id)
from categories) order by orders.customer_id;
```

THANK YOU