JENSON USA

Advanced SQL Project

Presented By :
Deepak Sharma

ABOUT JENSON

For over 25 years, we are America's best online cycling store at www.jensonusa.com and at retail locations in Corona and Riverside, California, featuring over 30,000+ products from all of the best brands for road bike, mountain bike, triathlon, BMX, gravel and commuter bikes, parts, apparel and accessories. Serving the bicycle community since 1994. Jenson USA was named to the Internet Retailers Top 500 list and nominated as a Top Workplace in the Inland Empire

PROJECT INTRODUCTION

This project involves analysing sales and customer data from Jenson USA using SQL to generate key business insights. By applying advanced SQL techniques such as Joins to combine data from multiple tables, Common Table Expressions (CTEs) for better query organization, and RANK functions for identifying top performing products and customers. The project aims to uncover patterns in purchasing behaviour, product popularity, and regional sales performance. The goal is to support data-driven decision making and improve customer engagement strategies.

Find the total number of products sold by each store along with the store name

```sql
SELECT
    orders.store_id,
    stores.store_name,
    SUM(order_items.quantity) Product_Sold
FROM
    orders
        JOIN
    order_items USING (order_id)
        JOIN
    stores USING (store_id)
GROUP BY store_id,store_name;
```

Calculate the cumulative sum of quantities sold for each product over time.

```sql
Select product_id, product_name, order_date, quantity,
sum(quantity) over (partition by product_id order by order_date) Cumulative_Sum
From (SELECT
    order_items.product_id, products.product_name, orders.order_date, SUM(order_items.quantity) quantity
FROM
    order_items JOIN orders USING (order_id)
        JOIN
    products USING (product_id)
GROUP BY product_id , order_date , product_name
) As A ;
```

Find the product with the highest total sales (quantity * price) for each category.

```sql
With A as (
Select categories.category_id, products.product_id,
Sum(order_items.quantity*order_items.list_price) Total_sales
from categories join products using (category_id) join order_items using (product_id)
group by category_id,product_id)

Select * from(
Select *,Row_number() over ( partition by category_id order by Total_sales Desc) Rating
from A) as B
Where B.Rating = "1";
```

Find the customer who spent the most money on orders

```sql
SELECT
    orders.customer_id,
    CONCAT(customers.first_name,' ',customers.last_name) As Customer_Name,
    SUM((order_items.quantity * order_items.list_price) - order_items.discount) AS Money_Spent
FROM order_items
        JOIN
    orders USING (order_id)
        JOIN
    customers USING (customer_id)
GROUP BY customer_id,customer_Name ORDER BY Money_spent DESC LIMIT 1;
```

Find the highest-priced product for each category name.

```
Select * from (
Select products.category_id,
        categories.category_name,
        products.product_name,
        products.list_price,
Rank() over(partition by category_id order by list_price Desc) Rating
from products join categories using (category_id) ) as A
Where Rating="1";
```

Find the total number of orders placed by each customer per store

```sql
SELECT
    orders.customer_id,
    CONCAT(customers.first_name,' ',customers.last_name) Customer_Name,
    orders.store_id, stores.store_name,
    COUNT(orders.order_id) order_Placed
FROM
    orders
        JOIN
    customers USING (customer_id)
        JOIN
    stores USING (store_id)
GROUP BY customer_id , store_id ORDER BY customer_id;
```

Find the names of staff members who have not made any sales

```sql
SELECT
    Staffs.staff_id,
    CONCAT(staffs.first_name, ' ', staffs.last_name) Staff_Name
FROM
    staffs
        LEFT JOIN
    orders USING (staff_id)
GROUP BY staff_id
HAVING SUM(orders.order_id) IS NULL ;
```

Find the top 3 most sold products in terms of quantity

```sql
Select A.product_id,products.product_name,A.Quantity_Sold
    from (
        Select order_items.product_id, sum(order_items.quantity) Quantity_Sold
        from order_items group by product_id
        ) as A Join products using (product_id)
order by A.quantity_Sold Desc limit 3;
```

Find the median value of the price list

```
With A as (
    select list_price,
    row_number() over(order by list_price ) rn,
    count(*) over () n from products
)
Select  case
            When n % 2 = 0 then (Select avg(list_price) from A
                where rn in ((n/2),(n/2)+1))
            else (Select list_price from A where rn = ((n+1)/2))
        End as median
from A Limit 1;
```

List all products that have never been ordered.(use Exists)

```sql
SELECT
    product_id, products.product_name
FROM
    products
WHERE
    NOT EXISTS( SELECT
            product_id
        FROM
            order_items
        WHERE
            order_items.product_id = products.product_id)
ORDER BY product_id;
```

List the names of staff members who have made more sales than the average number of sales by all staff members

```
With A as (
        Select distinct staffs.staff_id ,concat(staffs.first_name," ",staffs.last_name) Staff_Name,
            coalesce(Sum(list_price*quantity),0) As Sales
            from staffs left join orders using (staff_id)
                left join order_items using ( order_id)
                Group By staff_id
)
Select staff_id,Staff_Name,sales from  A
 where Sales >(Select avg(Sales) from A) ;
```

Identify the customers who have ordered all types of products (i.e., from every category)

```
Select orders.customer_id,concat(customers.first_name," ",customers.last_name) Customer_Name,
    count(distinct products.category_id) Category_ordered
    from orders join order_items using (order_id)
        join products using (product_id)
        Join customers using (customer_id)
    group by customer_id
    having count(distinct products.category_id) = (select count(categories.category_id) from categories);
```

JENSON USA

Thank You

Let’s Connect :

Deepak.official93@gmail.com

Deepak Sharma (deepakofficial93)