JENSON USA

JENSON USA

Data Analysis using Advanced SQL

Fauziya Malik

23 September, 2025

Introduction

Jenson USA is a U.S.-based bicycle retailer specializing in bikes, parts, apparel, and accessories, selling through its online store and retail locations in California. Founded in 1994 as a rider-owned business, it has grown into a well-known mid-sized company serving cyclists across the country.

JENSON USA

Background Project

Current scenario:

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

JENSON USA

SQL Essentials

Foundational Insight Queries

Core metrics and essential business overviews

Strategic Performance Queries

Trend analysis, rankings, efficiency and performance measurement

Advanced logic & optimization Queries

Existence checks, set coverage, and complex relational logic

Foundational Insight Queries

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

```sql
select stores.store_id , sum(order_items.quantity), stores.store_name
from stores left join orders using(store_id) inner join order_items using(order_id)
group by stores.store_id, stores.store_name;
```

2. Find the highest-priced product for each category name.

```sql
#5. Find the highest-priced product for each category name.
with ranked_products as (select c.category_id, c.category_name, p.product_name, p.list_price,
row_number() over (partition by c.category_id order by p.list_price desc) as rn
from categories c join products p on c.category_id = p.category_id)
select category_id, category_name, product_name, list_price from ranked_products where rn = 1;
```

Foundational Insight Queries

3. Find the median value of the price list.

```sql
with a as(select list_price, row_number() over(order by list_price) as rn, count(*) over() as 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;
```

These queries provide foundational business insights by summarizing store sales, identifying top-priced products by category, and determining the median product price.

Strategic Performance Queries

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

```
select  orders.order_date, products.product_id, products.product_name, order_items.quantity,
sum(order_items.quantity) over(partition by products.product_id order by orders.order_date) as cumulative_sum
from products left join order_items using(product_id) inner join orders using(order_id);
```

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

```
from (select c.category_id, c.category_name, p.product_id, p.product_name, sum(oi.quantity * oi.list_price) as total_sales
row_number() over (partition by c.category_id order by sum(oi.quantity * oi.list_price) desc) as rn
from products p join categories c on p.category_id = c.category_id join order_items oi on p.product_id = oi.product_id
group by c.category_id, c.category_name, p.product_id, p.product_name
) x
```

Strategic Performance Queries

3. Find the customer who spent the most money on orders.

```sql
select c.customer_id, c.first_name, c.last_name, sum(oi.quantity * oi.list_price) as total_sales
from customers c join orders o on c.customer_id = o.customer_id join order_items oi on o.order_id = oi.order_id
group by c.customer_id, c.first_name, c.last_name order by total_sales desc limit 1;
```

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

```sql
select c.customer_id, c.first_name, c.last_name, s.store_id, s.store_name, count(o.order_id) as total_orders
from customers c join orders o on c.customer_id = o.customer_id join stores s on o.store_id = s.store_id
group by c.customer_id, c.first_name, c.last_name, s.store_id, s.store_name order by c.customer_id, s.store_id;
```

5. Find the top 3 most sold products in terms of quantity.

```sql
select p.product_name, sum(oi.quantity) as total_sales
from products p join order_items oi on p.product_id = oi.product_id
group by p.product_name order by total_sales desc limit 3
```

Strategic Performance Queries

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

```
with staff_sales as (select s.staff_id, concat(s.first_name,'', s.last_name) as full_name,
coalesce(sum(oi.quantity * oi.list_price),0) as total_sales
from staffs s left join orders o using(staff_id) left join order_items oi using(order_id)
group by s.staff_id, s.first_name, s.last_name)
select * from staff_sales where total_sales > (select avg(total_sales) from staff_sales);
```

These queries deliver strategic performance insights by tracking sales trends, identifying top-selling products, analyzing customer spending, evaluating order activity, and comparing staff performance against organizational averages.

Advanced logic & Optimization Queries

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

```
select p.product_name, p.product_id
from products p
where not exists (select oi.order_id from order_items oi where oi.product_id = p.product_id);
```

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

```
select s.staff_id, s.first_name, s.last_name, o.order_id
from staffs s left join orders o using(staff_id) where o.order_id is null;
```

Advanced logic & Optimization Queries

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

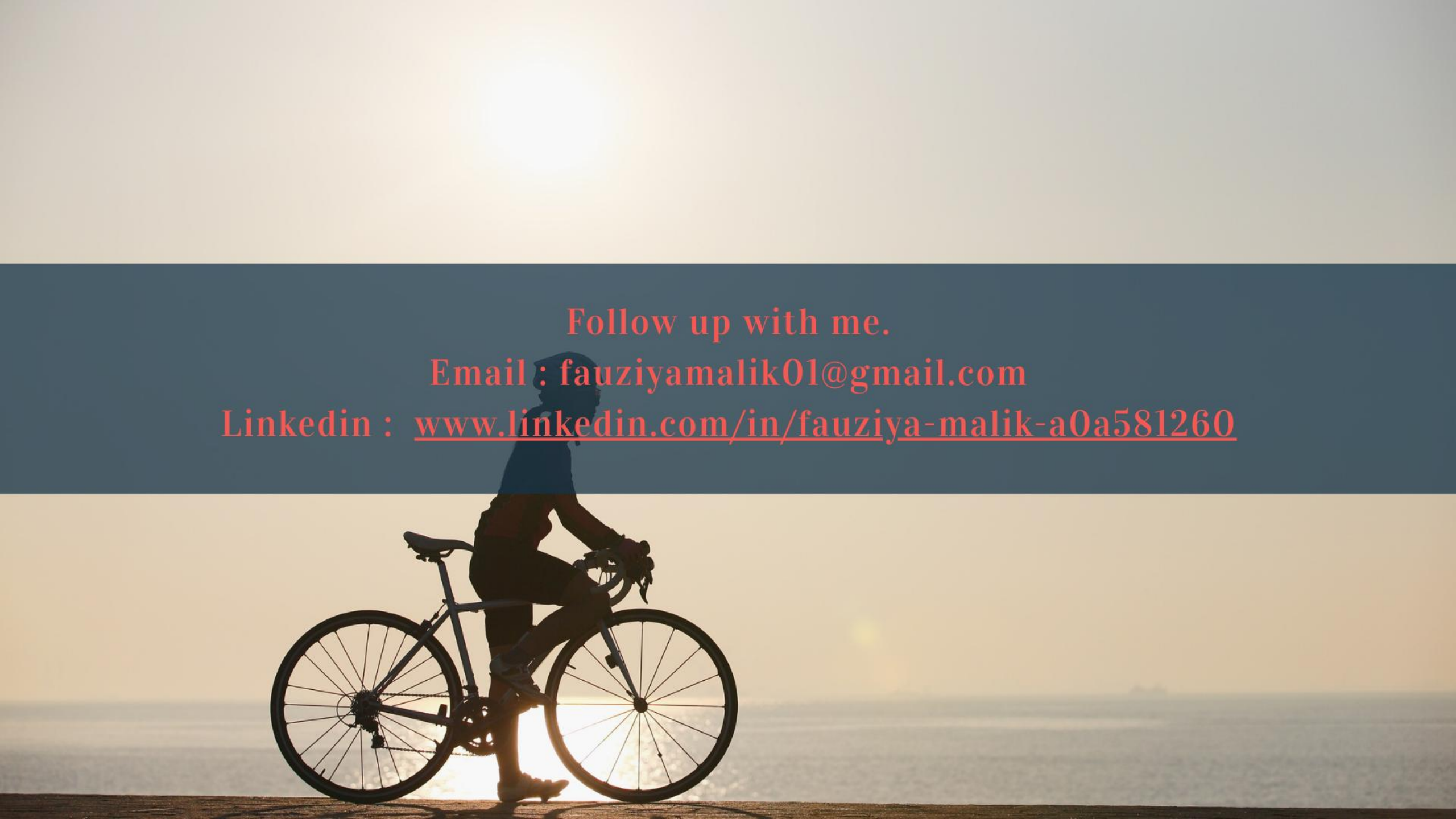
```
select c.customer_id, concat(c.first_name, c.last_name) as full_name,
count(p.category_id) as product_count_per_category
from customers c join orders o using(customer_id) join order_items oi
using(order_id) join products p using(product_id)
group by c.customer_id
having count(distinct p.category_id) = (select count(category_id) from categories);
```

These queries uncover gaps in sales activity and identify products, staff, and customers based on advanced existence and coverage checks.

Conclusion

Together, these queries provide a complete view of business performance—from basic insights to advanced analysis—highlighting sales trends, top performers, activity gaps, and overall operational patterns.

JENSON USA



Follow up with me.

Email : fauziyamalik01@gmail.com

Linkedin : www.linkedin.com/in/fauziya-malik-a0a581260