JENSON USA

MySQL

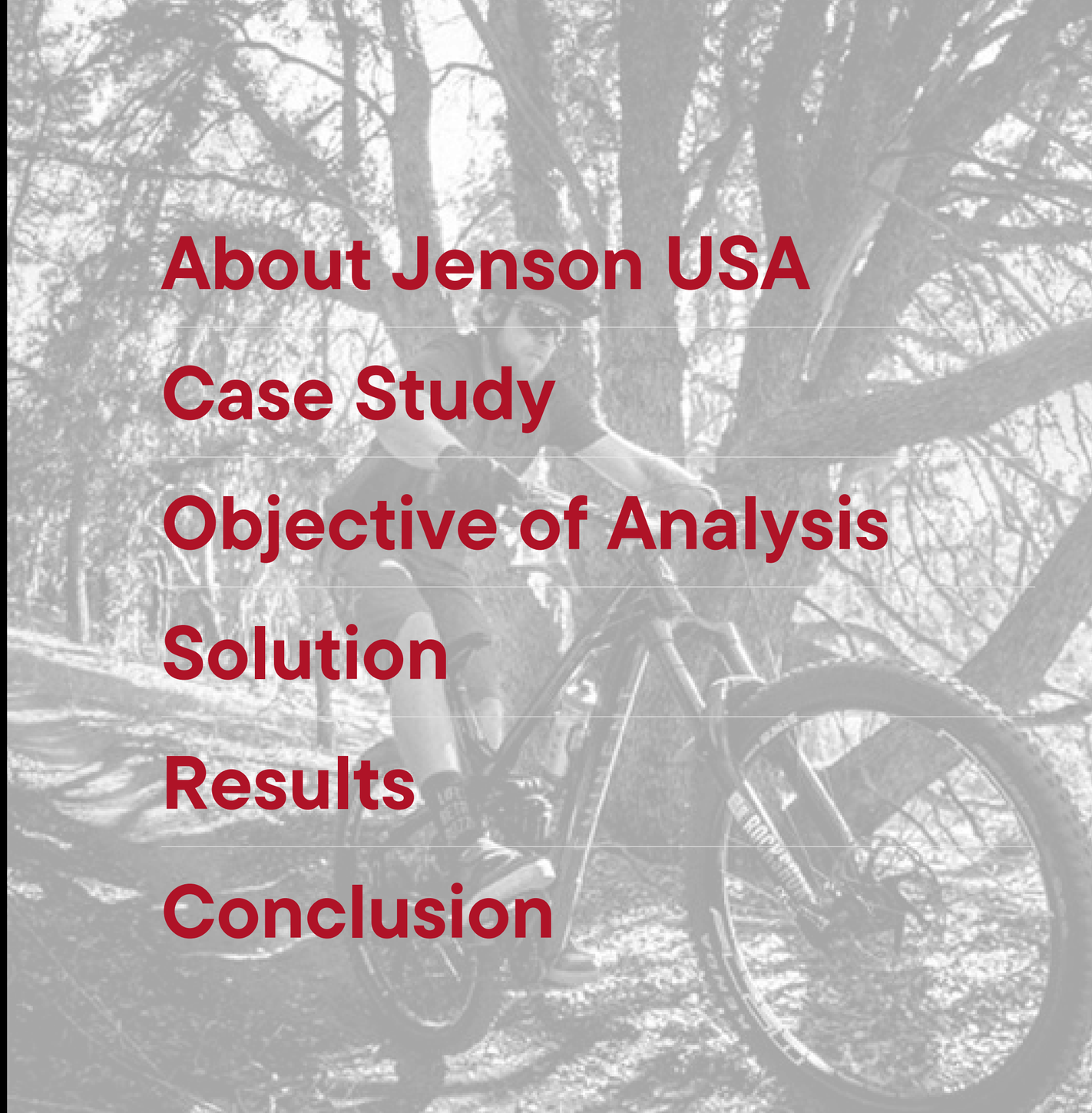
LEVERAGING SQL FOR

Operational Insights at Jenson USA

Presented by Keshab Karmakar

Report Overview

MAIN DISCUSSION POINTS



- About Jenson USA
- Case Study
- Objective of Analysis
- Solution
- Results
- Conclusion

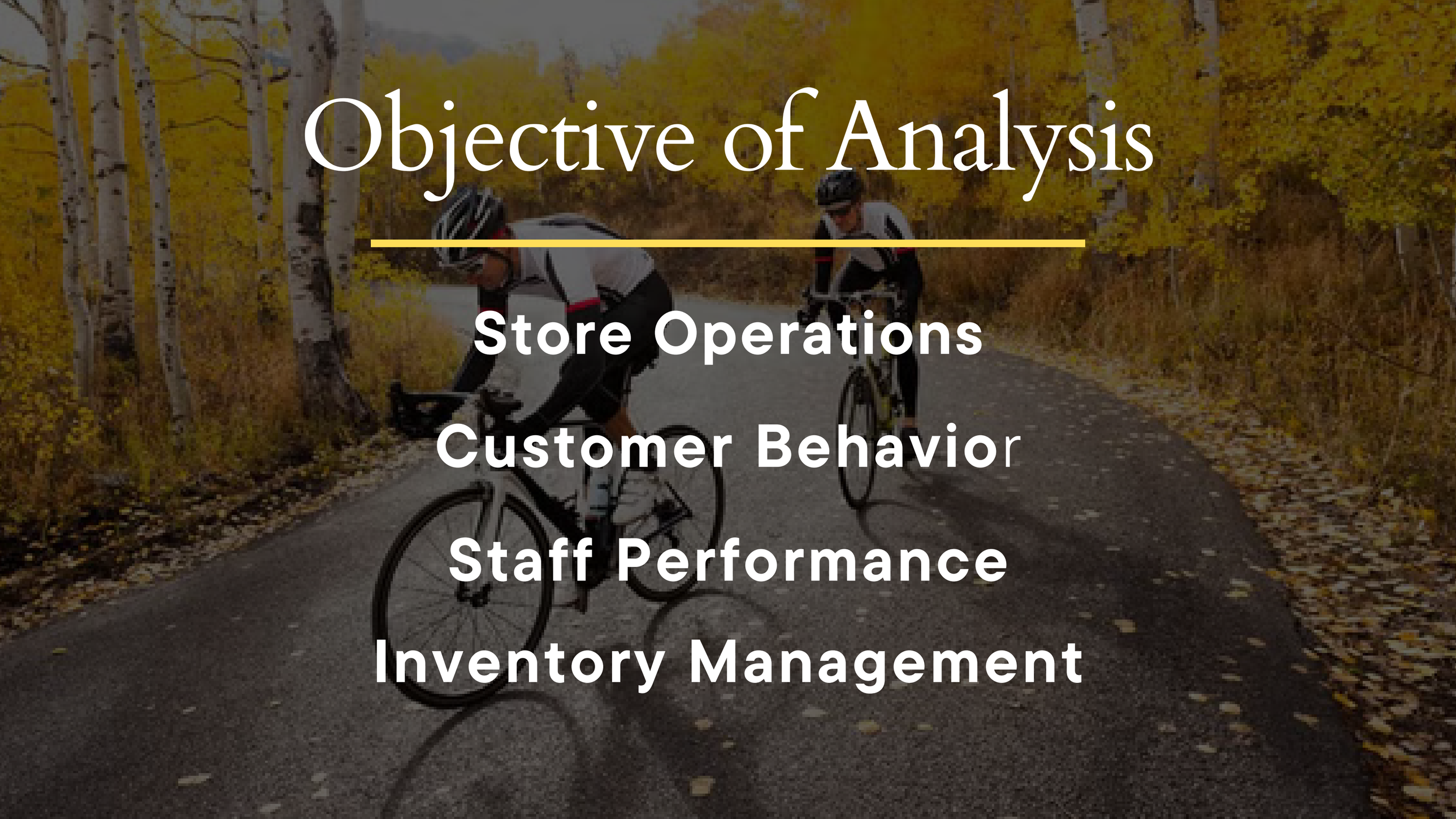
About

Jenson USA

Jenson USA is a leading retailer specializing in bicycles, bicycle parts, and cycling accessories. Established with a passion for cycling, the company is dedicated to providing a vast selection of high-quality products to enthusiasts of all skill levels. With multiple store locations and a strong online presence, Jenson USA is committed to delivering exceptional customer service, fostering a community of cycling enthusiasts, and promoting a healthy, active lifestyle

Case Study

Jenson USA, a prominent retailer specializing in bicycles and related products, sought to optimize its operations through data analysis. As a data analyst at Jenson USA, the goal was to craft SQL queries to gain actionable insights into customer behavior, staff performance, inventory management, and store operations



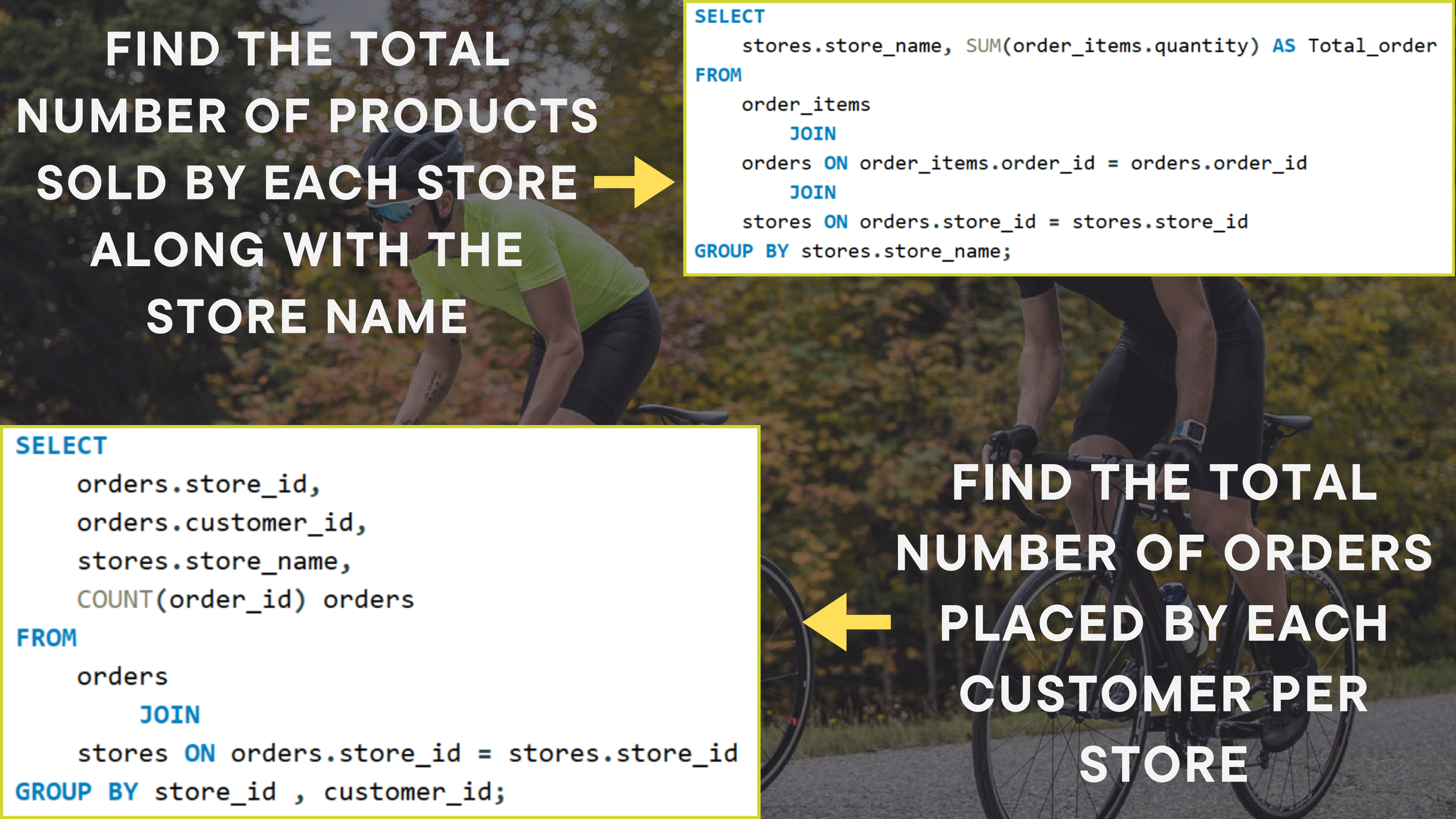
Objective of Analysis

Store Operations

Customer Behavior

Staff Performance

Inventory Management



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

```sql
SELECT
    stores.store_name, SUM(order_items.quantity) AS Total_order
FROM
    order_items
        JOIN
    orders ON order_items.order_id = orders.order_id
        JOIN
    stores ON orders.store_id = stores.store_id
GROUP BY stores.store_name;
```

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

```sql
SELECT
    orders.store_id,
    orders.customer_id,
    stores.store_name,
    COUNT(order_id) orders
FROM
    orders
        JOIN
    stores ON orders.store_id = stores.store_id
GROUP BY store_id , customer_id;
```

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

```sql
with a as (select order_items.product_id, orders.order_date, sum(order_items.quantity) as total_quantity
from order_items
join orders on
order_items.order_id = orders.order_id
group by order_items.product_id, orders.order_date)

select product_id, order_date,total_quantity, sum(total_quantity)
over(partition by product_id order by order_date) as Cum_total_quantity
from a;
```

Enabled tracking of product popularity and seasonal trends, aiding in demand forecasting

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

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

select * from
(select *, rank() over(partition by category_id order by total_sales desc) as rn from a) as B
where rn=1;
```

Highlighted key revenue-generating products within each category, informing inventory decisions

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

```sql
SELECT
    staffs.staff_id,
    CONCAT(staffs.first_name, ' ', staffs.last_name) AS full_name
FROM
    staffs
WHERE
    NOT EXISTS( SELECT
            staff_id
        FROM
            orders
        WHERE
            orders.staff_id = staffs.staff_id);
```

FIND THE MEDIAN VALUE OF THE PRICE LIST

```sql
with a as(select list_price, row_number() over(order by list_price) rn,
count(list_price) over() cn
from order_items)

SELECT CASE
WHEN cn % 2 = 0
THEN (SELECT AVG(list_price) FROM a
WHERE rn IN (cn / 2 , (cn / 2) + 1))
ELSE (SELECT list_price FROM a
WHERE rn = (cn + 1) / 2)
END AS median FROM a LIMIT 1;
```

FIND THE CUSTOMER WHO SPENT THE MOST MONEY ON ORDERS

```sql
with a as(select customers.customer_id, concat(customers.first_name, " ", customers.last_name) as full_name,
sum((order_items.quantity)* (order_items.list_price - order_items.discount)) as total_spent
from customers join orders
on customers.customer_id = orders.order_id
join order_items
on order_items.order_id = orders.order_id
group by customers.customer_id, concat(customers.first_name, " ", customers.last_name))

select * from (
select *, rank() over(order by total_spent desc) as rnk from a) as b
where rnk= 1;
```

Allowed for targeted marketing strategies to retain high-spending customers

FIND THE HIGHEST-PRICED PRODUCT FOR EACH CATEGORY NAME

```sql
with a as (select categories.category_id, categories.category_name, products.product_name,
products.list_price,
rank() over (partition by categories.category_id order by products.list_price desc) as rnk
from products join categories
on products.category_id = categories.category_id)

select * from a
where rnk=1;
```

Assisted in pricing strategy and focused marketing of premium products

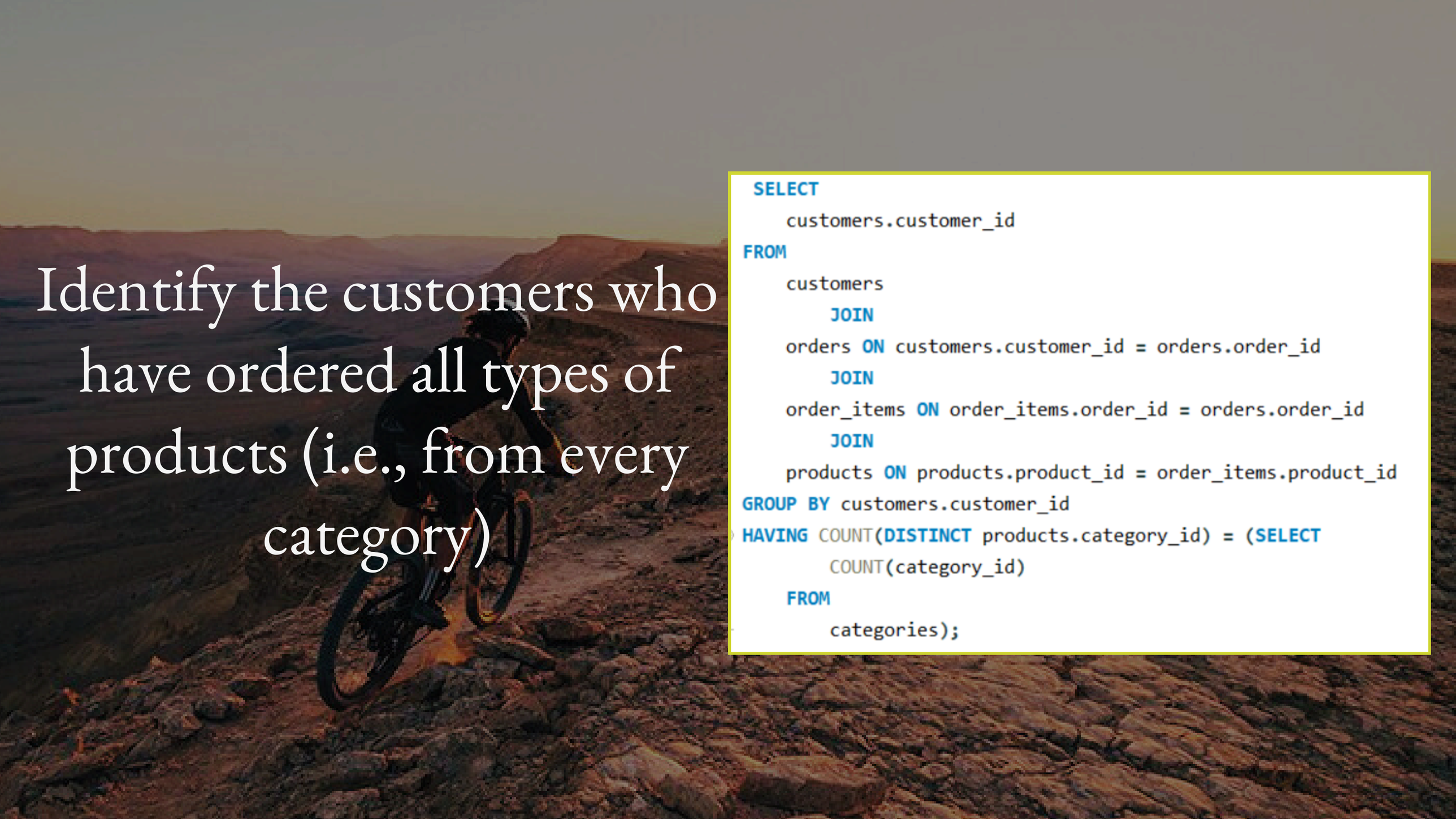
LIST THE NAMES OF STAFF MEMBERS WHO HAVE MADE MORE SALES THAN THE AVERAGE NUMBER OF SALES BY ALL STAFF MEMBERS

```sql
with a as (select staffs.staff_id, coalesce(
sum(order_items.quantity*(order_items.list_price-order_items.discount)),0) as sales
 from orders right join staffs
 on orders.staff_id = staffs.staff_id
 left join order_items
 on orders.order_id = order_items.order_id
 group by staffs.staff_id)

 select * from a where sales > (select avg(sales) from
 (select coalesce(
sum(order_items.quantity*(order_items.list_price-order_items.discount)),0) as sales
 from orders right join staffs
 on orders.staff_id = staffs.staff_id
 left join order_items
 on orders.order_id = order_items.order_id
 group by staffs.staff_id) as b);
```

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

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



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

```sql
SELECT
    customers.customer_id
FROM
    customers
        JOIN
    orders ON customers.customer_id = orders.order_id
        JOIN
    order_items ON order_items.order_id = orders.order_id
        JOIN
    products ON products.product_id = order_items.product_id
GROUP BY customers.customer_id
HAVING COUNT(DISTINCT products.category_id) = (SELECT
        COUNT(category_id)
    FROM
        categories);
```

FIND THE TOP 3 MOST SOLD PRODUCTS IN TERMS OF QUANTITY

```sql
with a as (select products.product_id, products.product_name, sum(order_items.quantity) as quantity,
rank() over(order by sum(order_items.quantity) desc) rnk
from products join order_items
on products.product_id = order_items.product_id
group by products.product_id, products.product_name)

select product_name from a
where rnk<= 3;
```

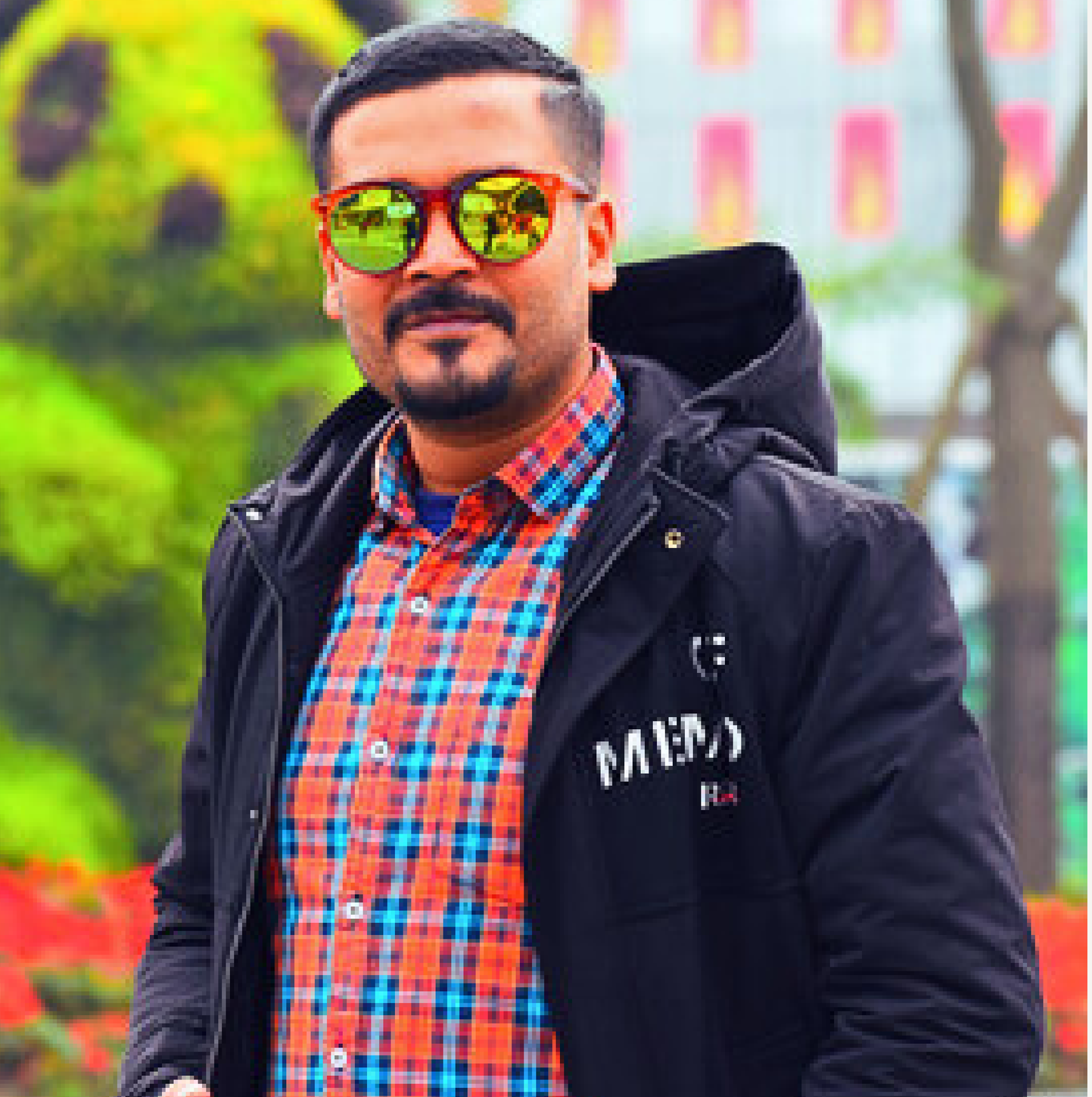
Focused inventory and marketing efforts on best-sellers, increasing sales efficiency

RESULTS

The insights gained from these SQL queries empowered Jenson USA to make data-driven decisions across various facets of the business. Store performance improved through targeted strategies, inventory management became more efficient, and customer satisfaction and retention increased.

Conclusion

Jenson USA successfully leveraged SQL queries to enhance customer understanding, optimize operations, and drive business growth. This case study underscores the value of data analysis in achieving operational excellence and strategic goals.



FOLLOW ME ON LINKEDIN TO GET MORE INFORMATION

THANK YOU