SQL PROJECT

text

JENSON USA

DRIVEN INSIGHTS USING ADVANCED SQL TECHNIQUES

BY PARUL KOKCHA

INTRODUCTiON TO JENSON USA

-Jenson USA is a leading online cycling store in the U.S,operating for over 25 years.

-Founded in 1994.

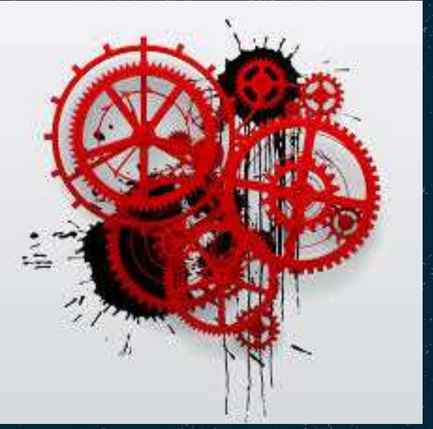
-Founder and CEO-Michael Cachat.

-Industry-Sporting Goods Manufacturing

-Headquarters-Riverside,California

PROJECT : JENSON USA

MY ROLE INVOLVES STUDYING AND ENHANCING DATABASE STRUCTURES TO IMPROVE DATA ANALYSIS, ENABLING DATA-DRIVEN DECISION-MAKING FOR BUSINESS OPTIMIZATION.



Here are the findings, actions taken for improvement, and insights :

DATA SOURCE :

Jenson USA Data

TOOLS USED :

MySQL

SKiLLS ENFORCED:

SQL Query
Crafting Data
Analysis
Insights
Derivation

FOCUS AREAS

STAFF PERFORMANCE

SALES PERFORMANCE

STORE OPERATIONS

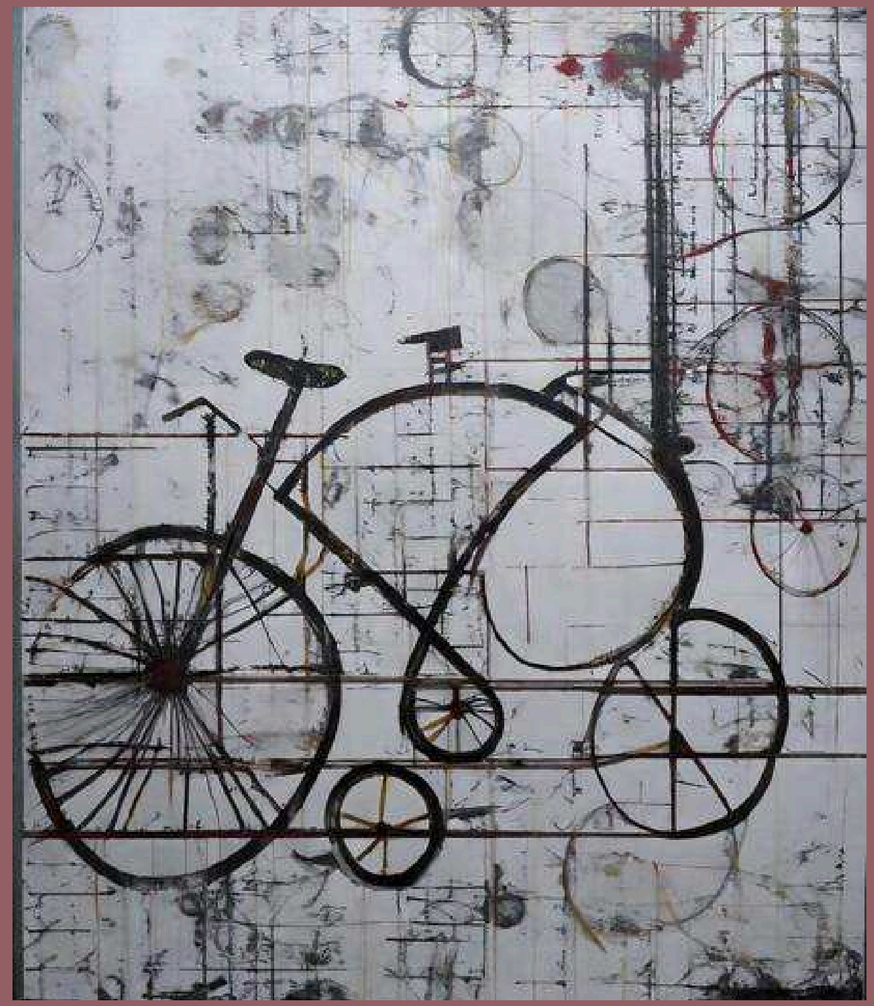
INVENTORY MANAGEMENT

CUSTOMER BEHAVIOR

DATASET OVERVIEW

STORES	CATEGORIES	BRANDS
PRODUCTS	CUSTOMERS	STAFFS
ORDERS	ORDER_ITEMS	STOCKS

KEY QUERiES :



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

```
2  select stores.store_name,
3  sum(order_items.quantity) as products_sold
4  from stores join orders
5  on stores.store_id = orders.store_id
6  join order_items
7  on order_items.order_id = orders.order_id
8  group by stores.store_name;
```

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

```sql
with a as (select products.product_name,orders.order_date,order_items.quantity
from products join order_items
on products.product_id = order_items.product_id
join orders
on order_items.order_id = orders.order_id)

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

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

```sql
with a as (select categories.category_name,
products.product_name,
sum(order_items.quantity*order_items.list_price) as 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 categories.category_name,
products.product_name)
select * from
(select *, dense_rank() over(partition by category_name order by sales desc) as rnk from a) as b where rnk=1
```

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

```sql
select customers.customer_id,
concat(customers.first_name , " ", customers.last_name) full_name,
sum(order_items.quantity * order_items.list_price) sales
from customers join orders
on customers.customer_id = orders.customer_id
join order_items
on order_items.order_id = orders.order_id
group by 1,2
order by sales desc
limit 1
```

5. Find the highest priced product for each category name.

```sql
select * from
(select categories.category_name,
products.product_name,
products.list_price,
dense_rank() over(partition by categories.category_name order by products.list_price desc) as rnk
from categories join products
on categories.category_id = products.category_id) a
where rnk = 1
```

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

```sql
select customers.customer_id, customers.first_name,
stores.store_name,
count(orders.order_id)
from customers left join orders
on customers.customer_id = orders.customer_id
join stores
on stores.store_id = orders.store_id
group by customers.customer_id, customers.first_name,
stores.store_name;
```

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

```sql
select staff_id, concat(first_name,last_name) fullname from staffs
where not exists (select * from orders
where staffs.staff_id = orders.staff_id);
```

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

```sql
select staffs.staff_id, concat(staffs.first_name,staffs.last_name) fullname,
orders.order_id from staffs
left join orders
on staffs.staff_id = orders.staff_id
where orders.order_id is null;
```

8. Find the top three most sold products in terms of quantity.

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

9. 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 mod(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;
```

10. List all products that have never been ordered.(Use exists)

```
select products.product_name from products
where not exists (select products.product_id from order_items
where order_items.product_id = products.product_id);
```

11. 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, staffs.first_name,
coalesce(sum(order_items.quantity * order_items.list_price),0) as sales
from staffs left join orders
on staffs.staff_id = orders.staff_id
left join order_items
on orders.order_id = order_items.order_id
group by staffs.staff_id,
staffs.first_name)

select * from a where sales > (select avg(sales) from a);
```

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

```sql
select customers.customer_id,
customers.first_name,
count(order_items.product_id) total_orders
from customers join orders
on customers.customer_id = orders.customer_id
join order_items
on orders.order_id = order_items.order_id
join products
on products.product_id = order_items.product_id
group by customers.customer_id,
customers.first_name
having count(distinct products.category_id) = (select count(*) from categories)
```

CONCLUSION

This project Jenson USA is focusing on SQL query-based problem-solving skills. It also challenges learners to analyze sales, customer behavior and staff performance using SQL. The project includes critical SQL queries like sales calculations, customer spending and staff performance analysis.


THANKS