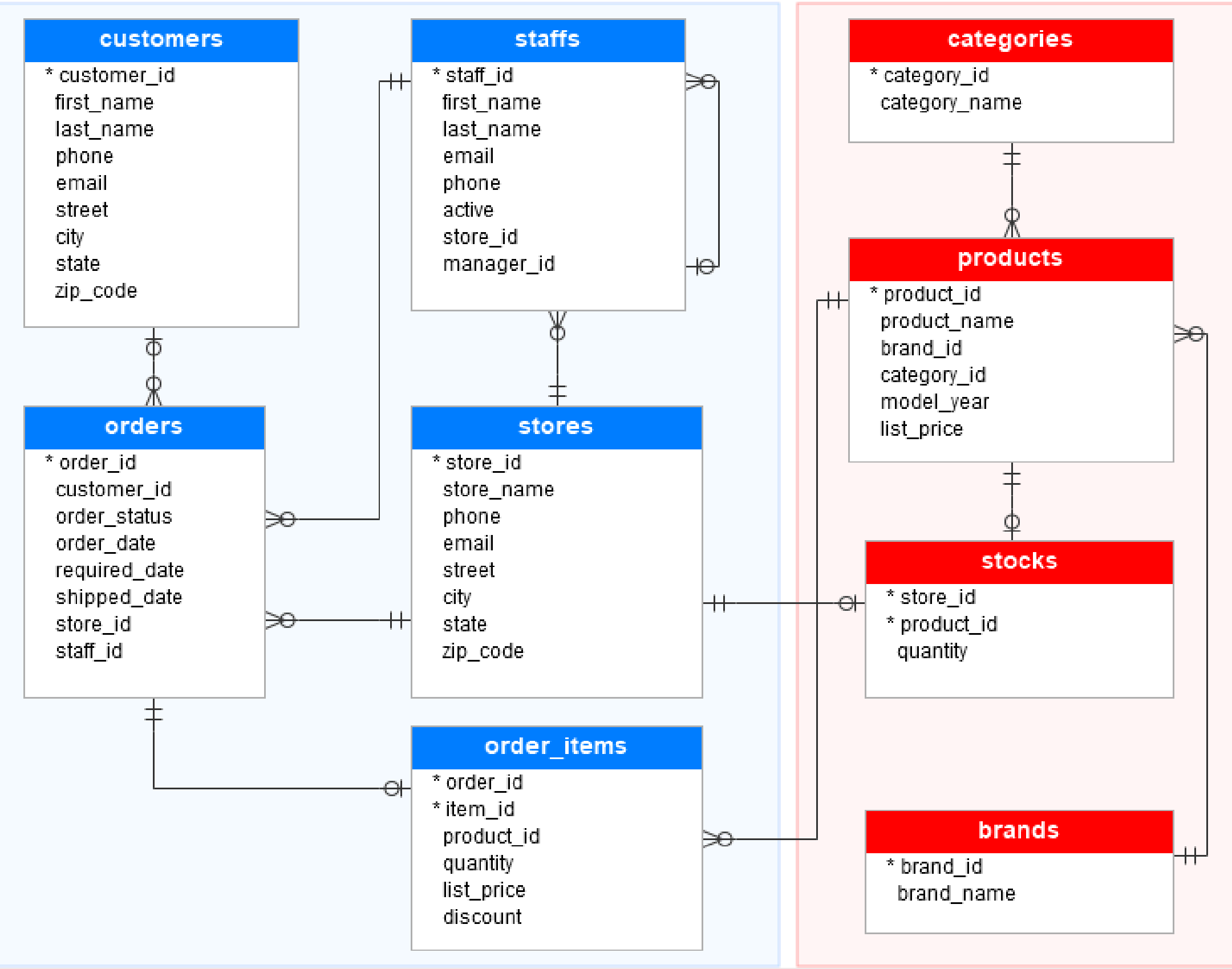
Bike Sales SQL Report

Project Overview

This project uses SQL to analyse the dataset of Bike Sales at different stores.The analysis covers the various aspects such as store detail, top customers ,total revenue generated ,staff details, brand details, order details etc.

Database Schema



Problems on which we are going to apply our SQL Queries:

Basic:

1. List all cities where customers live.
2. Retrieve the names of all categories.
3. Get the list of all active staff members.
4. the details of a specific product by its product_id (e.g., product_id = 1).
5. List all orders that have not been shipped yet.
6. Retrieve the phone numbers of all stores.
7. the first and last names of all customers living in 'New York'.
8. Get the total number of products available in the database.
9. List the email addresses of all staff members who work in a specific store (e.g., store_id = 1).
10. Retrieve the list price and model year of all products in a specific category (e.g., category_id = 1).

Moderate:

1. the total revenue generated by each product.
2. Retrieve the list of all customers along with their respective orders.
3. List the names and quantities of products in stock for each store.
4. the average list price of products for each category.
5. Retrieve the details of the staff members who are also managers.
6. List all products along with their brand names.

Hard:

1. the top 5 customers by total revenue generated from their orders.
2. Retrieve the products that have the highest total discount applied.
3. the store with the highest number of different products in stock.
4. Retrieve the average quantity of products ordered per order.
5. List the names and email addresses of customers who have placed an order in each month of the current year.
6. the staff members who have handled the most orders.

SQL Queries(Basic):

List all cities where customers live.

```
SELECT DISTINCT city FROM customers;
```

Retrieve the names of all categories.

```
SELECT category_name FROM categories;
```

Get the list of all active staff members.

```
SELECT first_name, last_name FROM staffs
WHERE active=1;
```

the details of a specific product by its product_id (e.g., product_id = 1).

```
SELECT * FROM products
WHERE product_id=1;
```

List all orders that have not been shipped yet.

```
SELECT * FROM orders
WHERE shipped_date IS NULL;
```

Retrieve the phone numbers of all stores.

```
SELECT phone FROM stores;
```

the first and last names of all customers living in 'New York'.

```
SELECT first_name,last_name from customers
WHERE city='New York'
```

Get the total number of products available in the database.

```
SELECT COUNT(*) AS total_products FROM products;
```

List the email addresses of all staff members who work in a specific store (e.g., store_id = 1).

```
SELECT email FROM staffs
WHERE store_id=1
```

Retrieve the list price and model year of all products in a specific category (e.g., category_id = 1).

```
SELECT list_price , model_year FROM products
WHERE category_id=1;
```

SQL Queries(Moderate):

the total revenue generated by each product.

```
SELECT p.product_name ,
ROUND(SUM(o.list_price*o.quantity-
o.discount),2) AS total_revenue FROM
products p
JOIN order_items o
ON p.product_id=o.product_id
GROUP BY p.product_name
ORDER BY total_revenue DESC;
```

Retrieve the list of all customers along with their respective orders.

```
SELECT
c.customer_id,c.first_name,c.last_name,o.order_id
FROM customers c
JOIN orders o
ON c.customer_id=o.customer_id
ORDER BY c.customer_id ASC;
```

List the names and quantities of products in stock for each store.

```
SELECT s.store_name, p.product_name, st.quantity
FROM stores s
JOIN stocks st ON s.store_id = st.store_id
JOIN products p ON st.product_id = p.product_id;
```

the average list price of products for each category.

```
SELECT
c.category_name,ROUND(AVG(p.list_price),2)
AS average_price FROM categories c
JOIN products p
ON c.category_id=p.category_id
GROUP BY c.category_name
```

Retrieve the details of the staff members who are also managers.

```
SELECT s1.staff_id,s1.first_name,s1.last_name FROM
staffs s1
JOIN staffs s2
ON s1.staff_id=s2.manager_id
```

List all products along with their brand names.

```
SELECT
 p.product_name,b.brand_name FROM products p
JOIN brands b
ON p.brand_id=b.brand_id
```

SQL Queries(Hard):

the top 5 customers by total revenue generated from their orders.

```
SELECT c.customer_id, c.first_name, c.last_name,
ROUND(SUM(oi.list_price*oi.quantity-oi.discount),2) AS
total_revenue
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_revenue DESC
LIMIT 5;
```

Retrieve the products that have the highest total discount applied.

```
SELECT p.product_name ,SUM(oi.discount*oi.quantity)
AS total_discount
FROM products p
JOIN order_items oi
ON p.product_id=oi.product_id
GROUP BY p.product_name
ORDER BY total_discount DESC
LIMIT 5;
```

the store with the highest number of different products in stock.

```
SELECT s.store_name, COUNT(DISTINCT st.product_id)
AS product_count
FROM stores s
JOIN stocks st ON s.store_id = st.store_id
GROUP BY s.store_name
ORDER BY product_count DESC
LIMIT 1;
```

Retrieve the average quantity of products ordered per order.

```
SELECT o.order_id, AVG(oi.quantity) AS
average_quantity
FROM orders o
JOIN order_items oi
ON o.order_id=oi.order_id
GROUP BY o.order_id
```

List the names and email addresses of customers who have placed an order in each month of the current year.

```
SELECT DISTINCT c.first_name, c.last_name, c.email
FROM customers c
JOIN orders o ON c.customer_id = o.customer_id
WHERE o.order_date BETWEEN '2024-01-01' AND '2024-12-31'
GROUP BY c.customer_id, c.first_name, c.last_name, c.email
HAVING  COUNT(DISTINCT  DATE_FORMAT(o.order_date,  '%Y-
%m')) = 12;
```

the staff members who have handled the most orders.

```
SELECT s.first_name, s.last_name, COUNT(o.order_id)
AS order_count
FROM staffs s
JOIN orders o ON s.staff_id = o.staff_id
GROUP BY s.staff_id, s.first_name, s.last_name
ORDER BY order_count DESC
LIMIT 5;
```

Thankyou

Rohit Kumawat