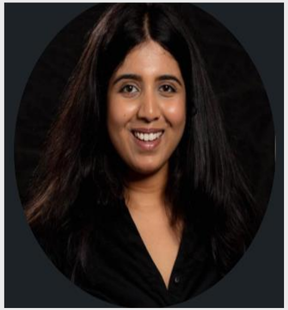
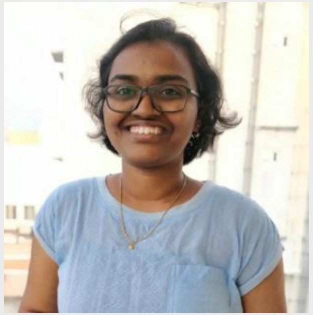
Team 23: Retail

MEET THE TEAM



SANCHITA GODSE



GEETHASREE NAGARAJU

AGENDA

Introduction	A little bit about the topic.
Our Problem	Why we chose this topic and what we hope to achieve.
ERD and Explanation	Entities and relationships we created.
Reports	5 Reports
Conclusion	Final Outcome of our Project.

01

Introduction

WHY RETAIL?

Retail refers to the sale of products to consumers for domestic or personal use. Since retail is an important part of the economy and retail business performance has a big impact on the development and prosperity of a region, hence we were interested in studying this topic.

02

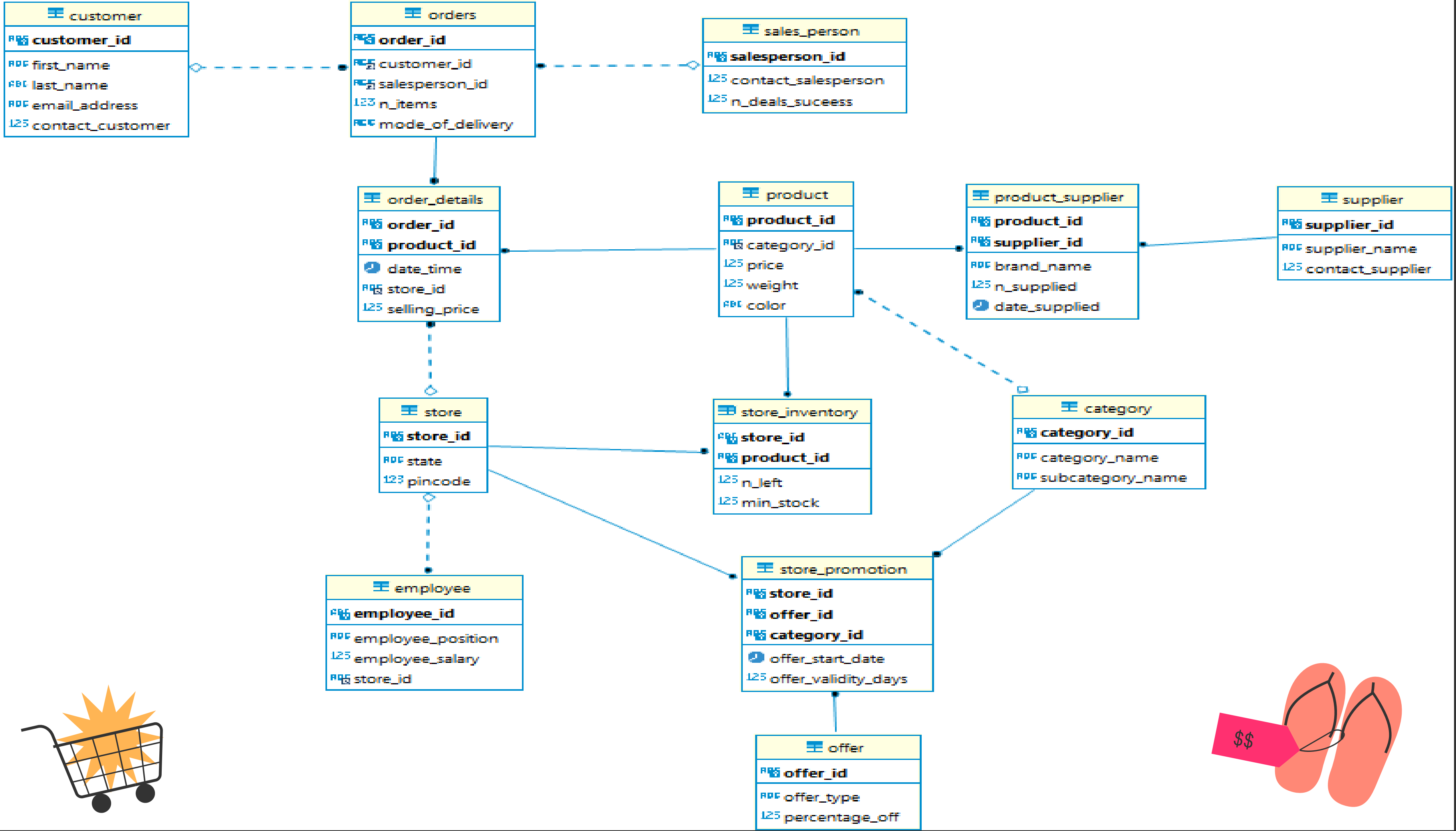
Our Problem

Retail Database Construction Challenge

The retail industry significantly impacts the economy and regional development, making it crucial to optimize operations and gain insights into customer behavior. To achieve this, a retail chain operating across the country aims to streamline its data management processes. However, constructing a comprehensive retail database is complex and challenging. It must accurately capture entity types such as customers, orders, products, suppliers, stores, inventory, employees, and promotions, while adhering to industry best practices and relevant regulations. The challenge is to construct an efficient and comprehensive retail database that provides valuable insights into customer behavior and operational efficiency.

03

ERD



customer
customer_id
first_name
last_name
email_address
contact_customer

orders
order_id
customer_id
salesperson_id
n_items
mode_of_delivery

sales_person
salesperson_id
contact_salesperson
n_deals_suceess

order_details
order_id
product_id
date_time
store_id
selling_price

product
product_id
category_id
price
weight
color

product_supplier
product_id
supplier_id
brand_name
n_supplied
date_supplied

supplier
supplier_id
supplier_name
contact_supplier

store
store_id
state
pincode

store_inventory
store_id
product_id
n_left
min_stock

category
category_id
category_name
subcategory_name

employee
employee_id
employee_position
employee_salary
store_id

store_promotion
store_id
offer_id
category_id
offer_start_date
offer_validity_days

offer
offer_id
offer_type
percentage_off

Explanations

Entity Explanation

Customer Entity

customer_id
first_name, last_name
email_address, phone_number

Orders Entity

order_id
n_items
mode_of_delivery
customer_id, salesperson_id

Order_details Entity

order_id, product_id
date_time
store_id
selling_price

Sales_person

salesperson_id
contact_salesperson
n_deals_success

Product

product_id
price
weight, color
category_id

Product_Supplier

product_id, supplier_id
brand_name
n_supplied
date_supplied

Entity Explanation

Supplier

supplier_id
supplier_name
contact_supplier

Store

store_id
state
pincode

Category

category_id
category_name
subcategory_name

Employee

employee_id
employee_position
employee_salary
store_id

Store_Inventory

store_id
product_id
n_left
min_stock

Offers

offer_id
offer_type
percentage_off

Store_Promotions

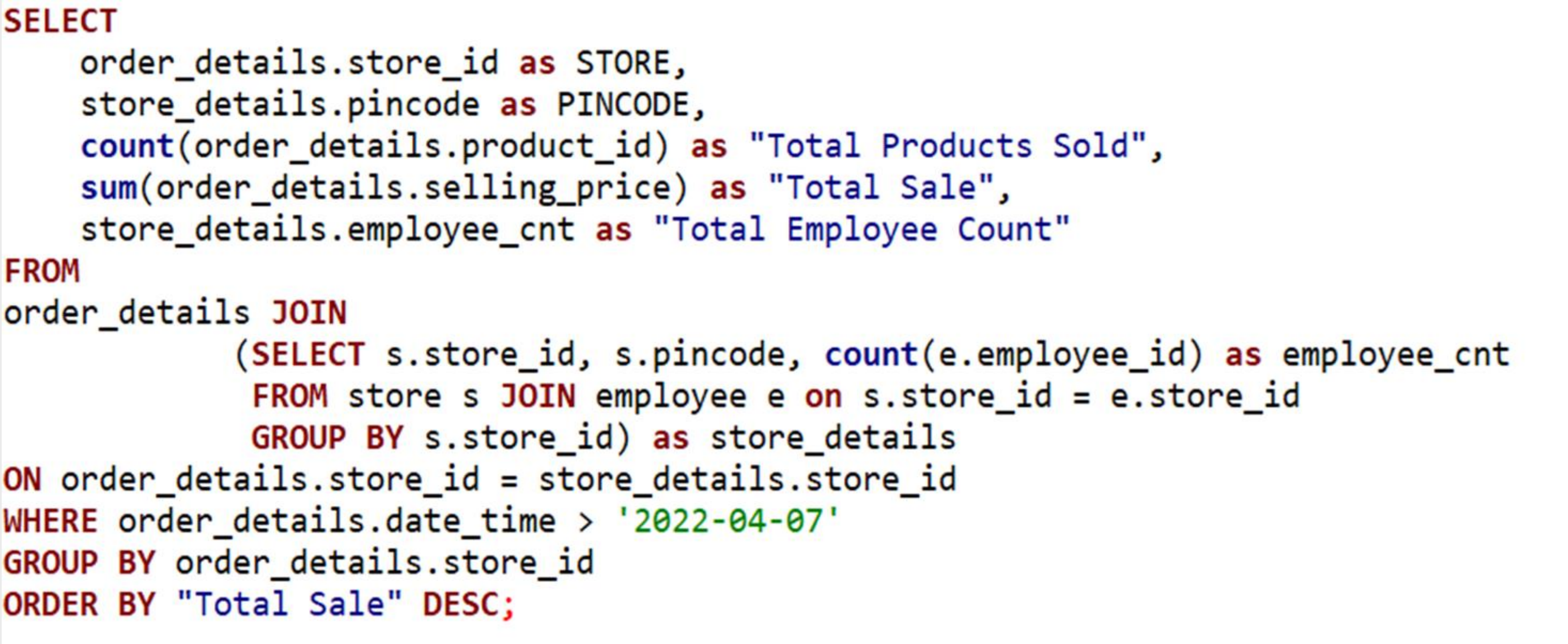
store_id
offer_id
category_id
offer_validity_days
offer_start_date

04

Reports

Report 1: Sales analysis by store

SQL Code:



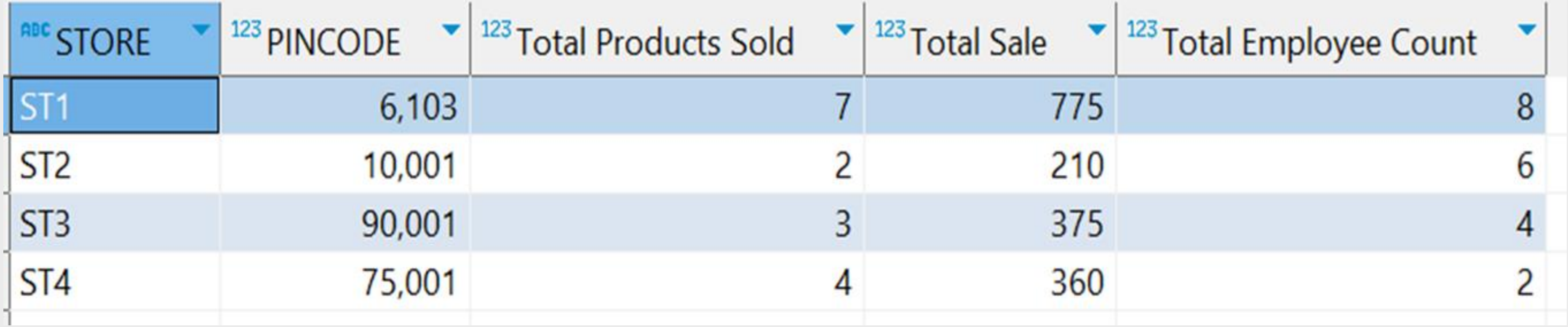
```sql
SELECT
    order_details.store_id as STORE,
    store_details.pincode as PINCODE,
    count(order_details.product_id) as "Total Products Sold",
    sum(order_details.selling_price) as "Total Sale",
    store_details.employee_cnt as "Total Employee Count"
FROM
order_details JOIN
            (SELECT s.store_id, s.pincode, count(e.employee_id) as employee_cnt
             FROM store s JOIN employee e on s.store_id = e.store_id
             GROUP BY s.store_id) as store_details
ON order_details.store_id = store_details.store_id
WHERE order_details.date_time > '2022-04-07'
GROUP BY order_details.store_id
ORDER BY "Total Sale" DESC;
```

Business value:

(i)which store is generating the highest sales?

(ii) is the employee count at each of the stores affecting the sales?

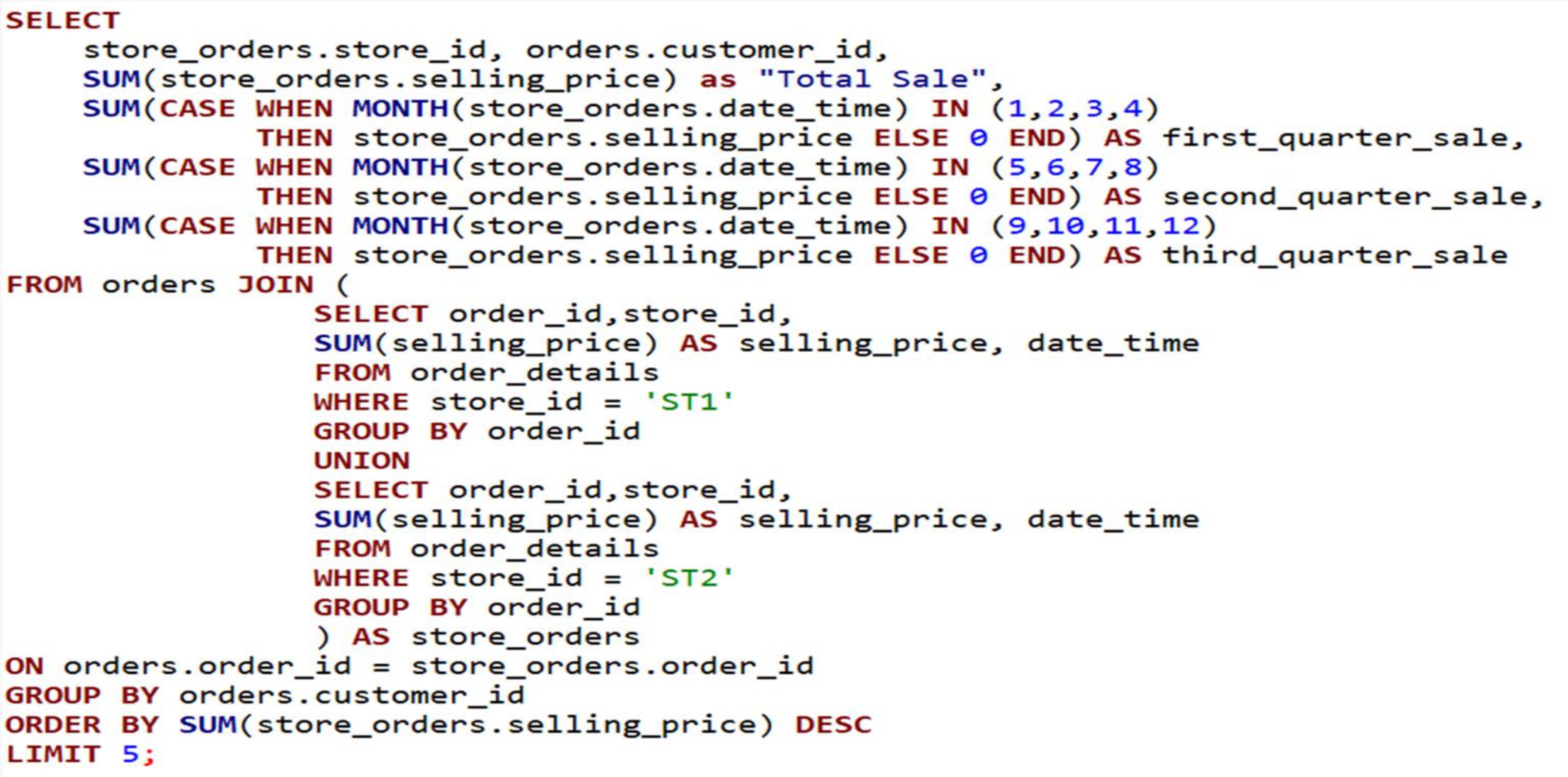
Output:



STORE	PINCODE	Total Products Sold	Total Sale	Total Employee Count
ST1	6,103	7	775	8
ST2	10,001	2	210	6
ST3	90,001	3	375	4
ST4	75,001	4	360	2

Report 2: Customer purchase behavior patterns

SQL Code:



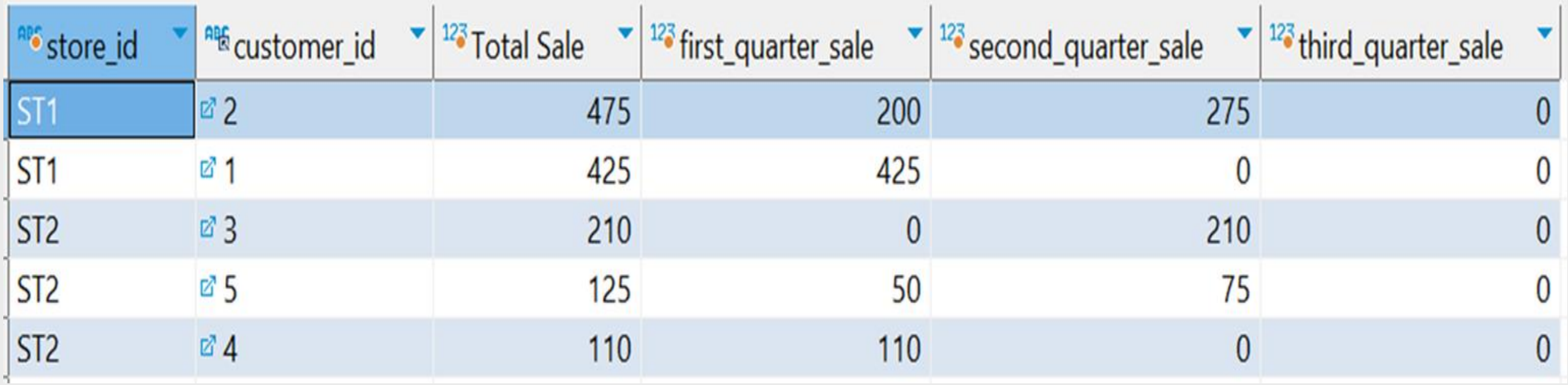
```
SELECT
    store_orders.store_id, orders.customer_id,
    SUM(store_orders.selling_price) as "Total Sale",
    SUM(CASE WHEN MONTH(store_orders.date_time) IN (1,2,3,4)
             THEN store_orders.selling_price ELSE 0 END) AS first_quarter_sale,
    SUM(CASE WHEN MONTH(store_orders.date_time) IN (5,6,7,8)
             THEN store_orders.selling_price ELSE 0 END) AS second_quarter_sale,
    SUM(CASE WHEN MONTH(store_orders.date_time) IN (9,10,11,12)
             THEN store_orders.selling_price ELSE 0 END) AS third_quarter_sale
FROM orders JOIN (
                SELECT order_id,store_id,
                SUM(selling_price) AS selling_price, date_time
                FROM order_details
                WHERE store_id = 'ST1'
                GROUP BY order_id
                UNION
                SELECT order_id,store_id,
                SUM(selling_price) AS selling_price, date_time
                FROM order_details
                WHERE store_id = 'ST2'
                GROUP BY order_id
                ) AS store_orders
ON orders.order_id = store_orders.order_id
GROUP BY orders.customer_id
ORDER BY SUM(store_orders.selling_price) DESC
LIMIT 5;
```

Business value:

(i) Is there a pattern in purchase behaviour of top customers in a year?

(ii) can business provide customized promotions to its valuable customers in particular season? (individual marketing decisions)

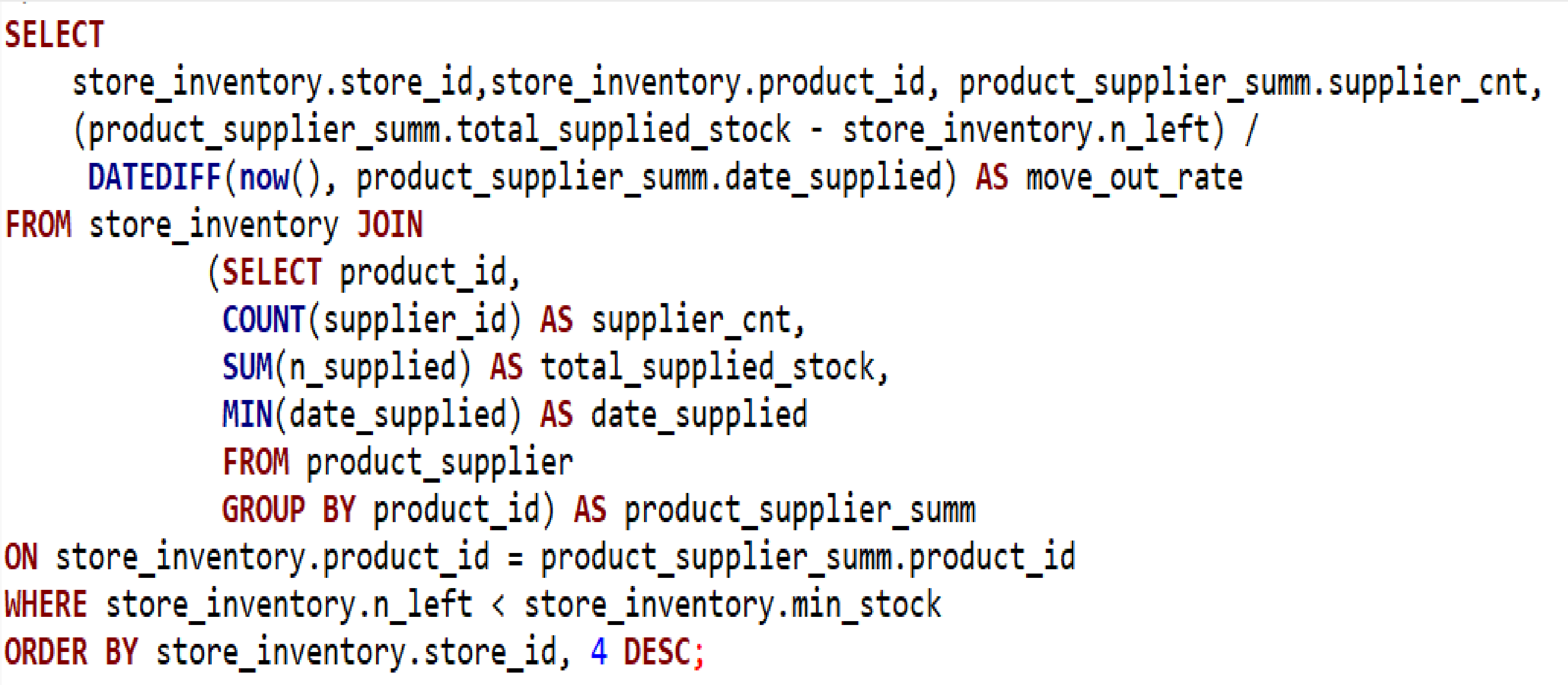
Output:



store_id	customer_id	Total Sale	first_quarter_sale	second_quarter_sale	third_quarter_sale
ST1	2	475	200	275	0
ST1	1	425	425	0	0
ST2	3	210	0	210	0
ST2	5	125	50	75	0
ST2	4	110	110	0	0

Report 3: Inventory management and tracking of product stock levels and reordering

SQL Code:



```sql
SELECT
    store_inventory.store_id,store_inventory.product_id, product_supplier_summ.supplier_cnt,
    (product_supplier_summ.total_supplied_stock - store_inventory.n_left) /
     DATEDIFF(now(), product_supplier_summ.date_supplied) AS move_out_rate
FROM store_inventory JOIN
            (SELECT product_id,
             COUNT(supplier_id) AS supplier_cnt,
             SUM(n_supplied) AS total_supplied_stock,
             MIN(date_supplied) AS date_supplied
             FROM product_supplier
             GROUP BY product_id) AS product_supplier_summ
ON store_inventory.product_id = product_supplier_summ.product_id
WHERE store_inventory.n_left < store_inventory.min_stock
ORDER BY store_inventory.store_id, 4 DESC;
```

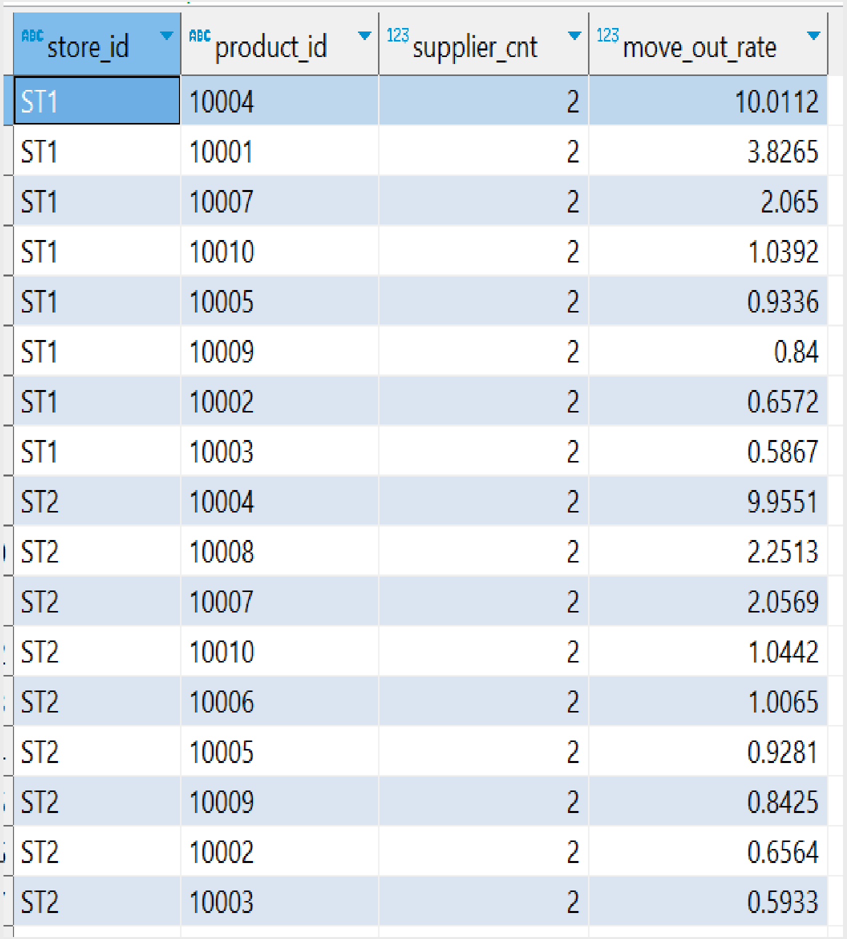
Output:

Business value:

(i) what products are currently out of stock

(ii) what is the move out rate of each of these products? which products are in high demand?

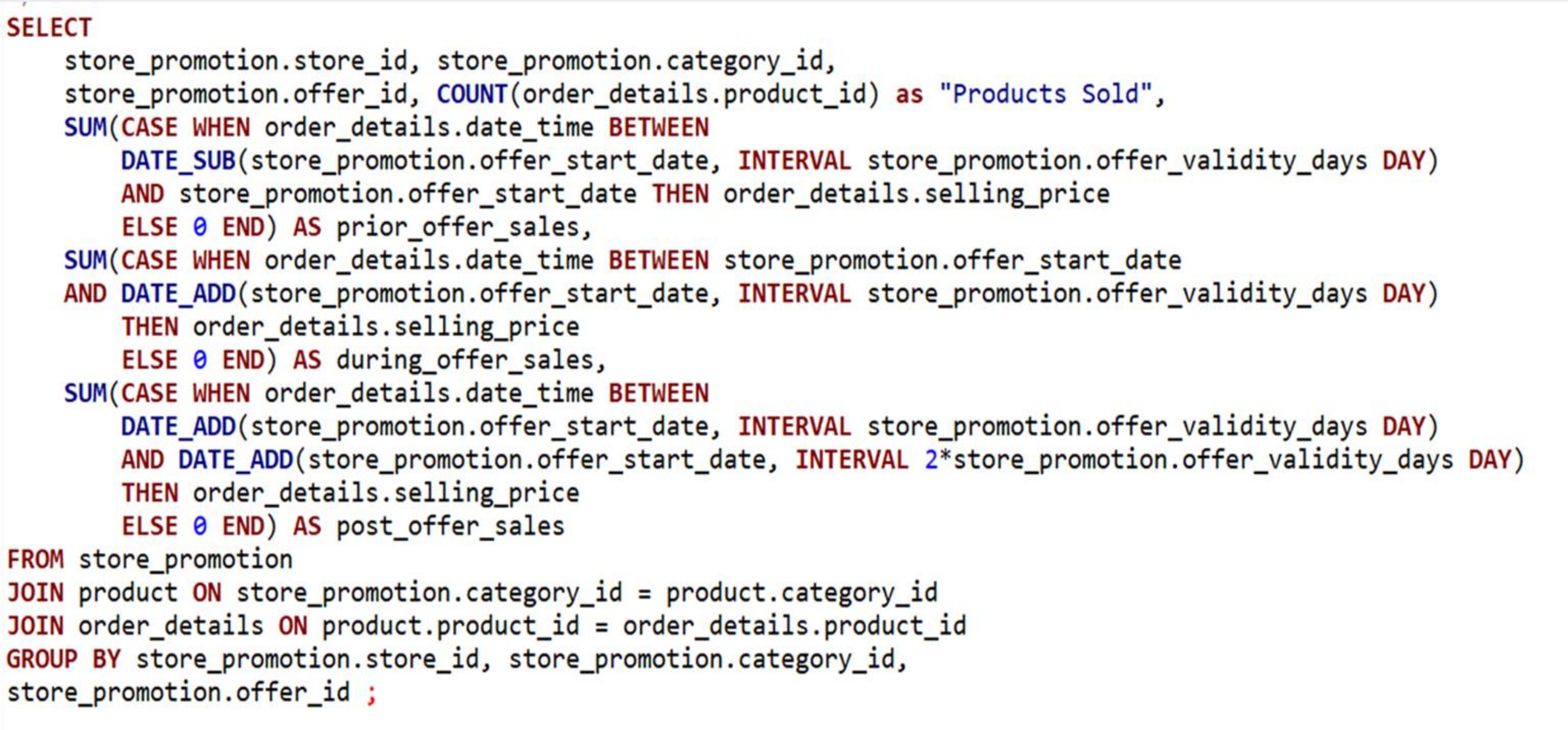
(iii) better inventory planning can reduce cost and lead to profits in long run



store_id	product_id	supplier_cnt	move_out_rate
ST1	10004	2	10.0112
ST1	10001	2	3.8265
ST1	10007	2	2.065
ST1	10010	2	1.0392
ST1	10005	2	0.9336
ST1	10009	2	0.84
ST1	10002	2	0.6572
ST1	10003	2	0.5867
ST2	10004	2	9.9551
ST2	10008	2	2.2513
ST2	10007	2	2.0569
ST2	10010	2	1.0442
ST2	10006	2	1.0065
ST2	10005	2	0.9281
ST2	10009	2	0.8425
ST2	10002	2	0.6564
ST2	10003	2	0.5933

Report 4: Impact of promotions on sales

SQL Code:



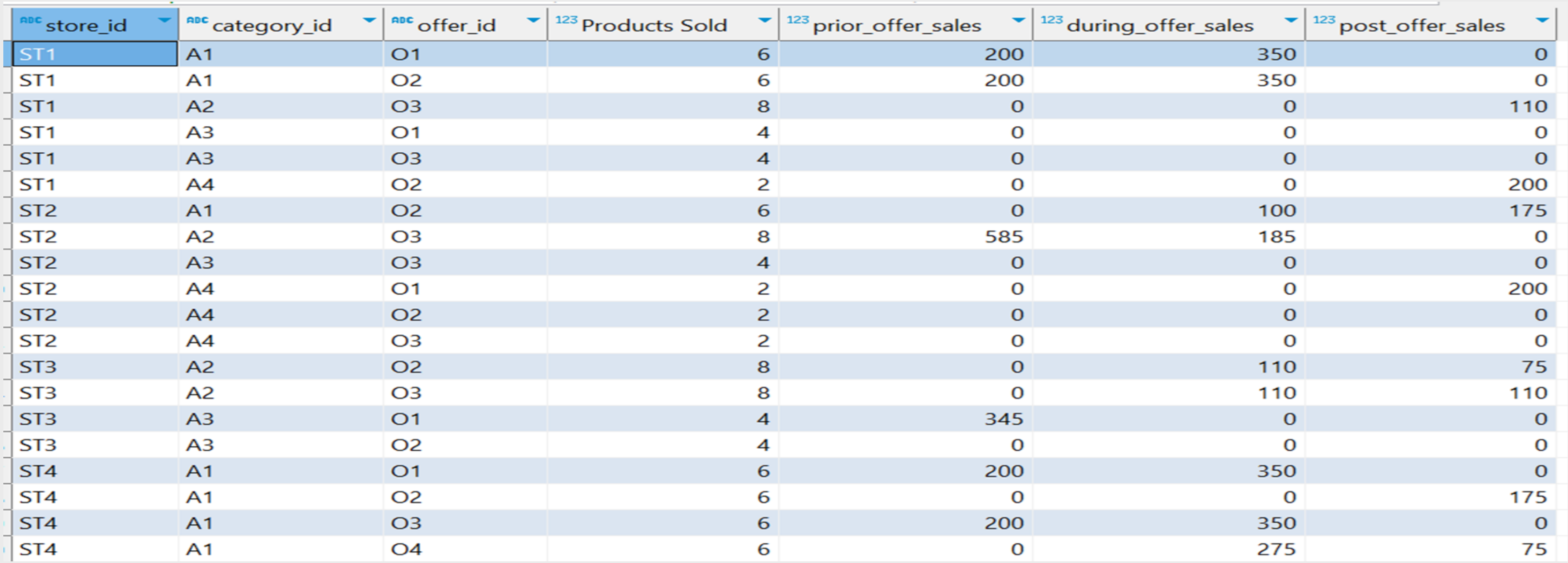
```
SELECT
    store_promotion.store_id, store_promotion.category_id,
    store_promotion.offer_id, COUNT(order_details.product_id) as "Products Sold",
    SUM(CASE WHEN order_details.date_time BETWEEN
        DATE_SUB(store_promotion.offer_start_date, INTERVAL store_promotion.offer_validity_days DAY)
        AND store_promotion.offer_start_date THEN order_details.selling_price
        ELSE 0 END) AS prior_offer_sales,
    SUM(CASE WHEN order_details.date_time BETWEEN store_promotion.offer_start_date
    AND DATE_ADD(store_promotion.offer_start_date, INTERVAL store_promotion.offer_validity_days DAY)
        THEN order_details.selling_price
        ELSE 0 END) AS during_offer_sales,
    SUM(CASE WHEN order_details.date_time BETWEEN
        DATE_ADD(store_promotion.offer_start_date, INTERVAL store_promotion.offer_validity_days DAY)
        AND DATE_ADD(store_promotion.offer_start_date, INTERVAL 2*store_promotion.offer_validity_days DAY)
        THEN order_details.selling_price
        ELSE 0 END) AS post_offer_sales
FROM store_promotion
JOIN product ON store_promotion.category_id = product.category_id
JOIN order_details ON product.product_id = order_details.product_id
GROUP BY store_promotion.store_id, store_promotion.category_id,
store_promotion.offer_id ;
```

Business value:

(i) Is an offer more effective than other on a specific category and a store?

(ii) are there any offers that are not effective?

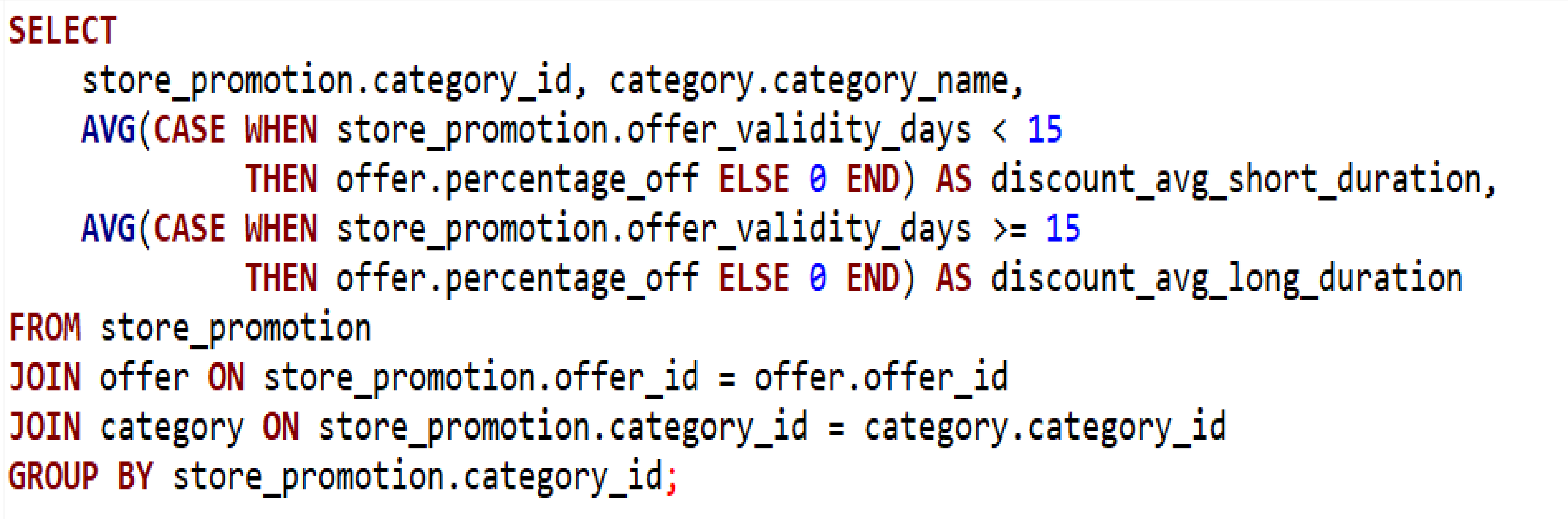
Output:



store_id	category_id	offer_id	Products Sold	prior_offer_sales	during_offer_sales	post_offer_sales
ST1	A1	O1	6	200	350	0
ST1	A1	O2	6	200	350	0
ST1	A2	O3	8	0	0	110
ST1	A3	O1	4	0	0	0
ST1	A3	O3	4	0	0	0
ST1	A4	O2	2	0	0	200
ST2	A1	O2	6	0	100	175
ST2	A2	O3	8	585	185	0
ST2	A3	O3	4	0	0	0
ST2	A4	O1	2	0	0	200
ST2	A4	O2	2	0	0	0
ST2	A4	O3	2	0	0	0
ST3	A2	O2	8	0	110	75
ST3	A2	O3	8	0	110	110
ST3	A3	O1	4	345	0	0
ST3	A3	O2	4	0	0	0
ST4	A1	O1	6	200	350	0
ST4	A1	O2	6	0	0	175
ST4	A1	O3	6	200	350	0
ST4	A1	O4	6	0	275	75

Report 5: Discount optimization for Short and Long Duration Offers by Category

SQL Code:



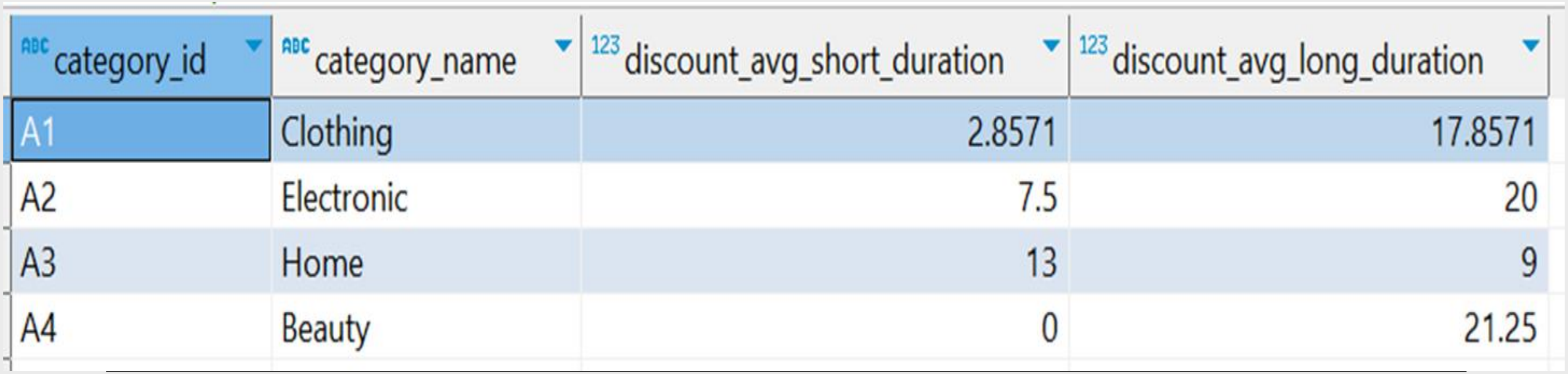
```sql
SELECT
    store_promotion.category_id, category.category_name,
    AVG(CASE WHEN store_promotion.offer_validity_days < 15
             THEN offer.percentage_off ELSE 0 END) AS discount_avg_short_duration,
    AVG(CASE WHEN store_promotion.offer_validity_days >= 15
             THEN offer.percentage_off ELSE 0 END) AS discount_avg_long_duration
FROM store_promotion
JOIN offer ON store_promotion.offer_id = offer.offer_id
JOIN category ON store_promotion.category_id = category.category_id
GROUP BY store_promotion.category_id;
```

Business value:

(i) what are the optimal discount percentages of short and long duration offers per category?

(ii) can business adjust its discounts as per its competetors to maximize benefit?

Output:



category_id	category_name	discount_avg_short_duration	discount_avg_long_duration
A1	Clothing	2.8571	17.8571
A2	Electronic	7.5	20
A3	Home	13	9
A4	Beauty	0	21.25

05

Conclusion

Business Recommendations

1. Implement Predictive Analytics to forecast demand, identify trends, and optimize inventory levels. This will help the business make data-driven decisions and improve supply chain management.

1. Monitor Performance: Track key performance indicators (KPIs), such as sales, customer retention, and inventory turnover, using the system. This will help the business measure its progress and identify areas for improvement.

Thank You!