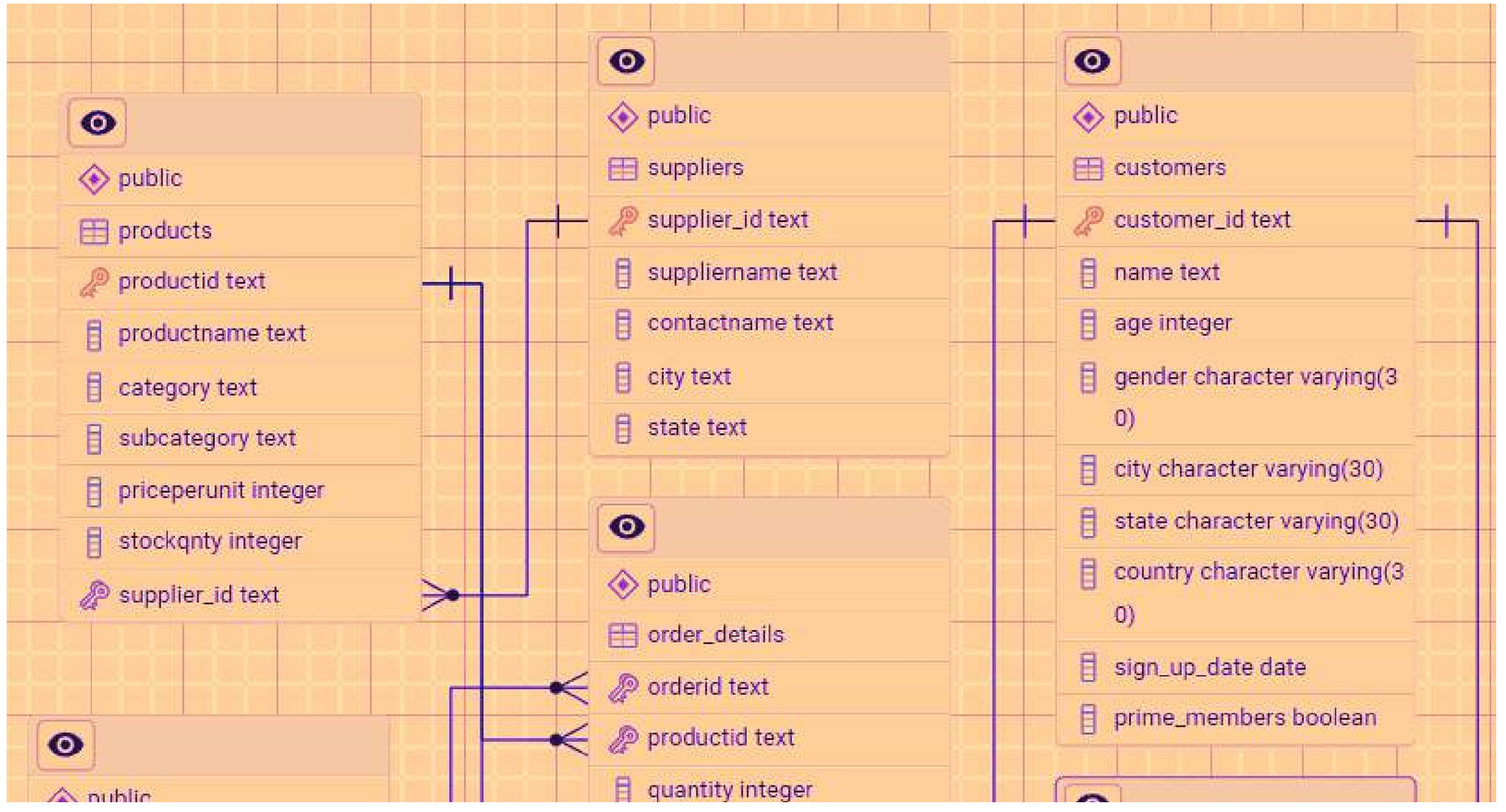
Amazon Fresh Analytics

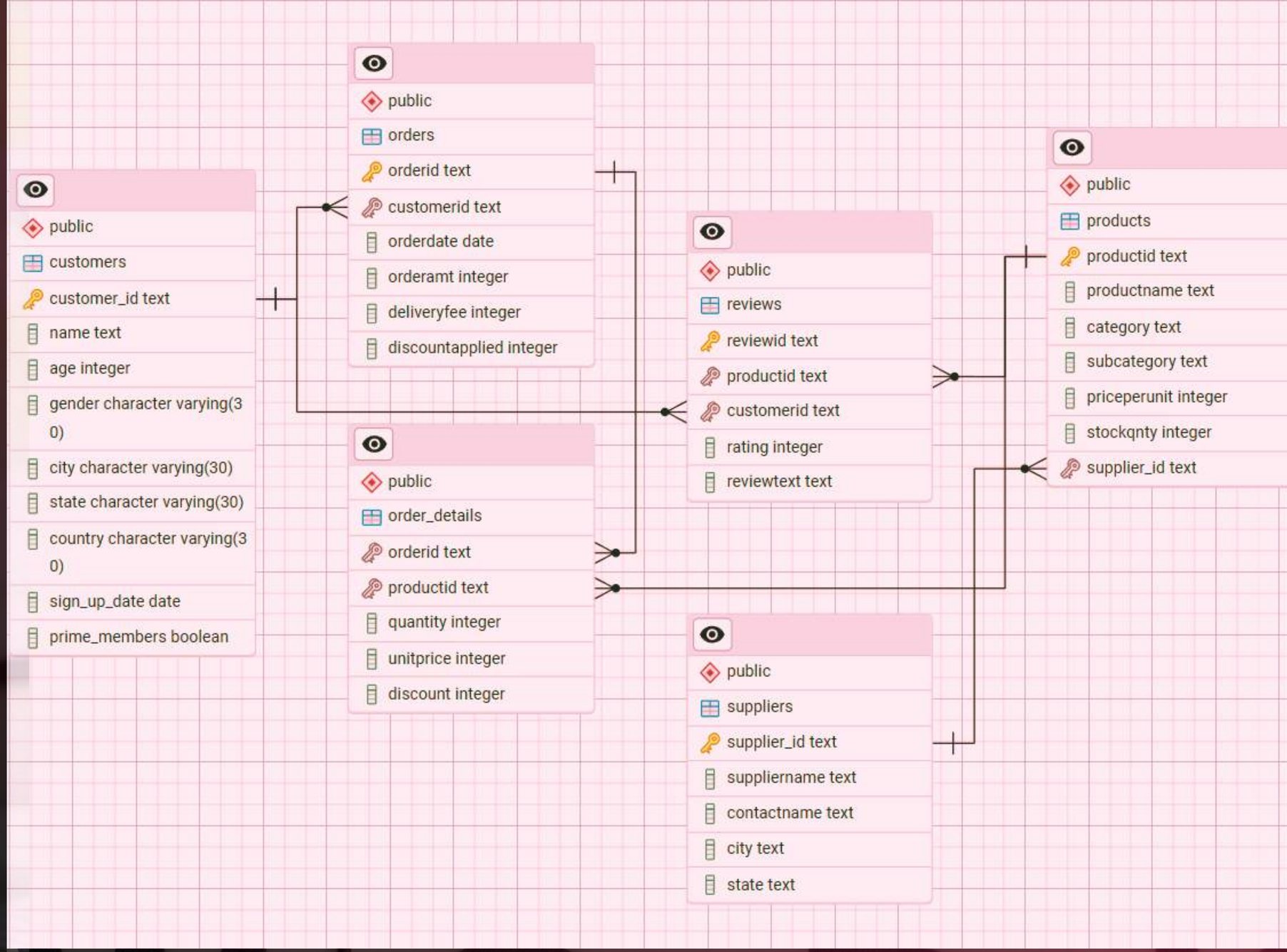
PostgreSQL

PROJECT

ER diagram for the Amazon Fresh database to understand the relationships between tables



PRIMARY KEYS AND FOREIGN KEYS FOR EACH TABLE AND THEIR RELATION SHIPS.

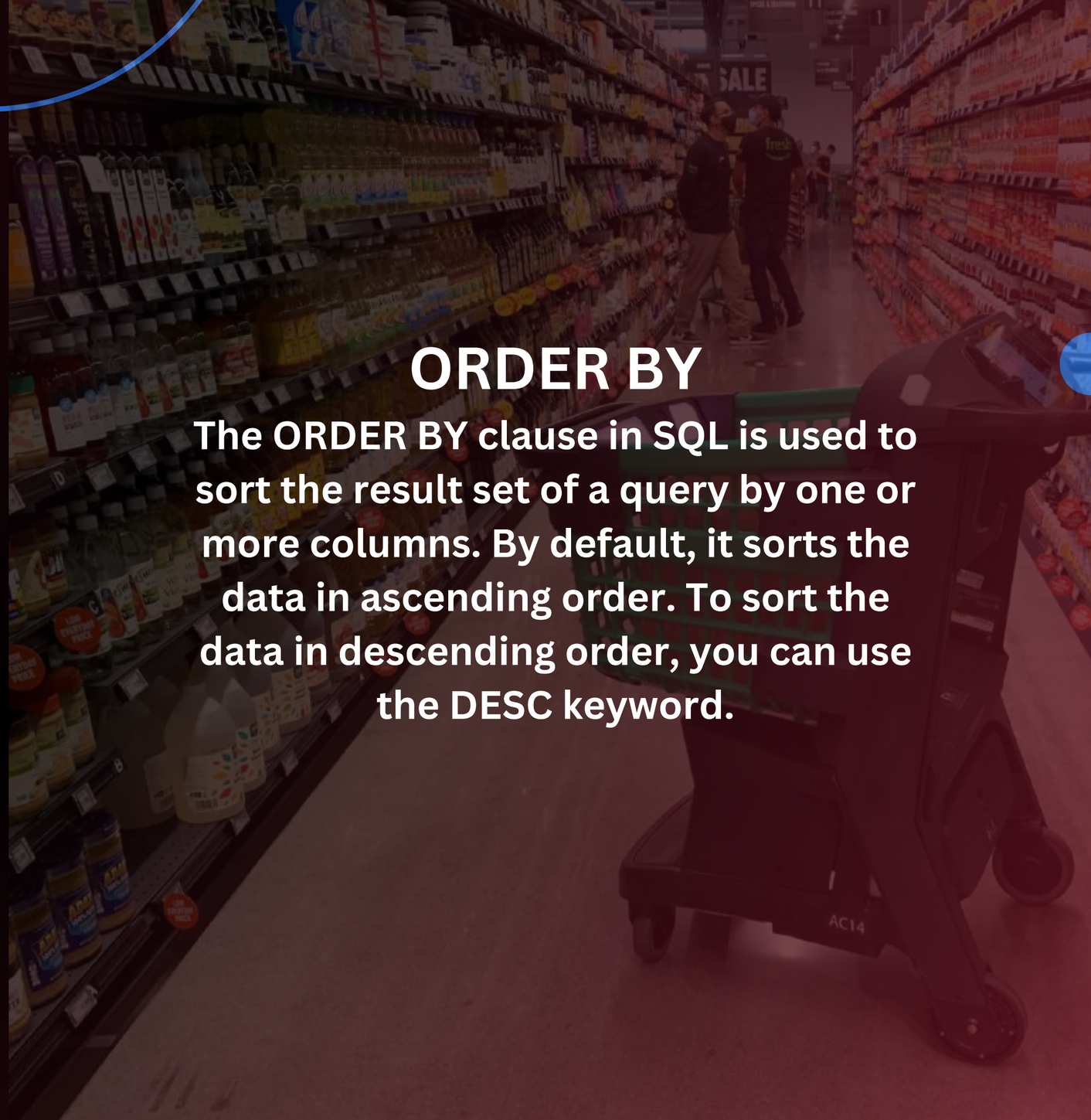


TASK 3: WRITE A QUERY TO: RETRIEVE ALL CUSTOMERS FROM A SPECIFIC CITY. FETCH ALL PRODUCTS UNDER THE "FRUITS" CATEGORY.

Selected customer’s name and their city from the customers table.

Pulled all the products name which comes under the Fruits category

In both the task retrieved data by using “ORDER BY” clause.



ORDER BY

The ORDER BY clause in SQL is used to sort the result set of a query by one or more columns. By default, it sorts the data in ascending order. To sort the data in descending order, you can use the DESC keyword.

PRIMARY KEY

The PRIMARY KEY constraint uniquely identifies each record in a table. Primary keys must contain UNIQUE values, and cannot contain NULL values.

UNIQUE CONSTRAINT

The UNIQUE constraint ensures that all values in a column are different. Both the UNIQUE and PRIMARY KEY constraints provide a guarantee for uniqueness for a column or set of columns.

TASK 4: WRITE DDL STATEMENTS TO RECREATE THE CUSTOMERS TABLE WITH THE FOLLOWING CONSTRAINTS: CUSTOMER_ID AS THE PRIMARY KEY. ENSURE AGE CANNOT BE NULL AND MUST BE GREATER THAN 18. ADD A UNIQUE CONSTRAINT FOR NAME.

Recreated the customers table with the mentioned constraint above.
Primary Key
Unique

Constraint:
SQL constraints are used to specify rules for the data in a table.

TASK 5: Insert 3 new rows into the Products table using INSERT statements.

Inserted three different rows in the products table.

Column names: Product ID, Product Name, Category, Subcategory, Price per unit, Stock quantity, Supplier ID.

TASK 6: Update the stock quantity of a product where Product_ID matches a specific ID.

Retrieved product name and stock quantity column from the products table. Found number of quantity in each products.

amazon fresh

TASK 8: ADD A CHECK CONSTRAINT TO ENSURE THAT RATINGS IN THE REVIEWS TABLE ARE BETWEEN 1 AND 5.

TASK 7: DELETE A SUPPLIER FROM THE SUPPLIERS TABLE WHERE THEIR CITY MATCHES A SPECIFIC VALUE.

PULLED THE NAME OF THE SUPPLIER AND THEIR ID FROM SUPPLIERS TABLE. AND DELETED THEIR NAME WHERE THEIR CITY MATCHES A SPECIFIC VALUE.

ADDED THE CHECK CONSTRAINT IN THE REVIEWS TABLE AND CHECKED WHETHER THE RATINGS ARE IN-BETWEEN OF VALUES 1 TO 5.

ADD A DEFAULT CONSTRAINT FOR THE PRIMEMEMBER COLUMN IN THE CUSTOMERS TABLE (DEFAULT VALUE: "NO").

ALTERED THE TABLE BY ADDING THE DEFAULT CONSTRAINT TO THE CUSTOMERS TABLE. AND SET THE DEFAULT VALUE AS ‘NO’.

TASK 9

WHERE clause to find orders placed after 2024-01-01.

RETRIEVED ORDER ID, CUSTOMERS ID AND ORDER DATE COULMNS FROM THE ORDERS TABLE AND USED WHERE CLAUSE WHICH US ORDER DATE SHOULD BE AFTER 2024-01-01.

HAVING clause to list products with average ratings greater than 4.

JOINED TWO DIFFERENT TABLES (PRODUCTS & REVIEWS) BY USING LEFT JOIN. USED AGGREGATE FUNCTION AVERAGE IN RATINGS COLUMN IN REVIEWS TABLE AND HAVING CLAUSE TO GET REVIEW RATINGS GREATER THAN 4.

GROUP BY and ORDER BY clauses to rank products by total sales.

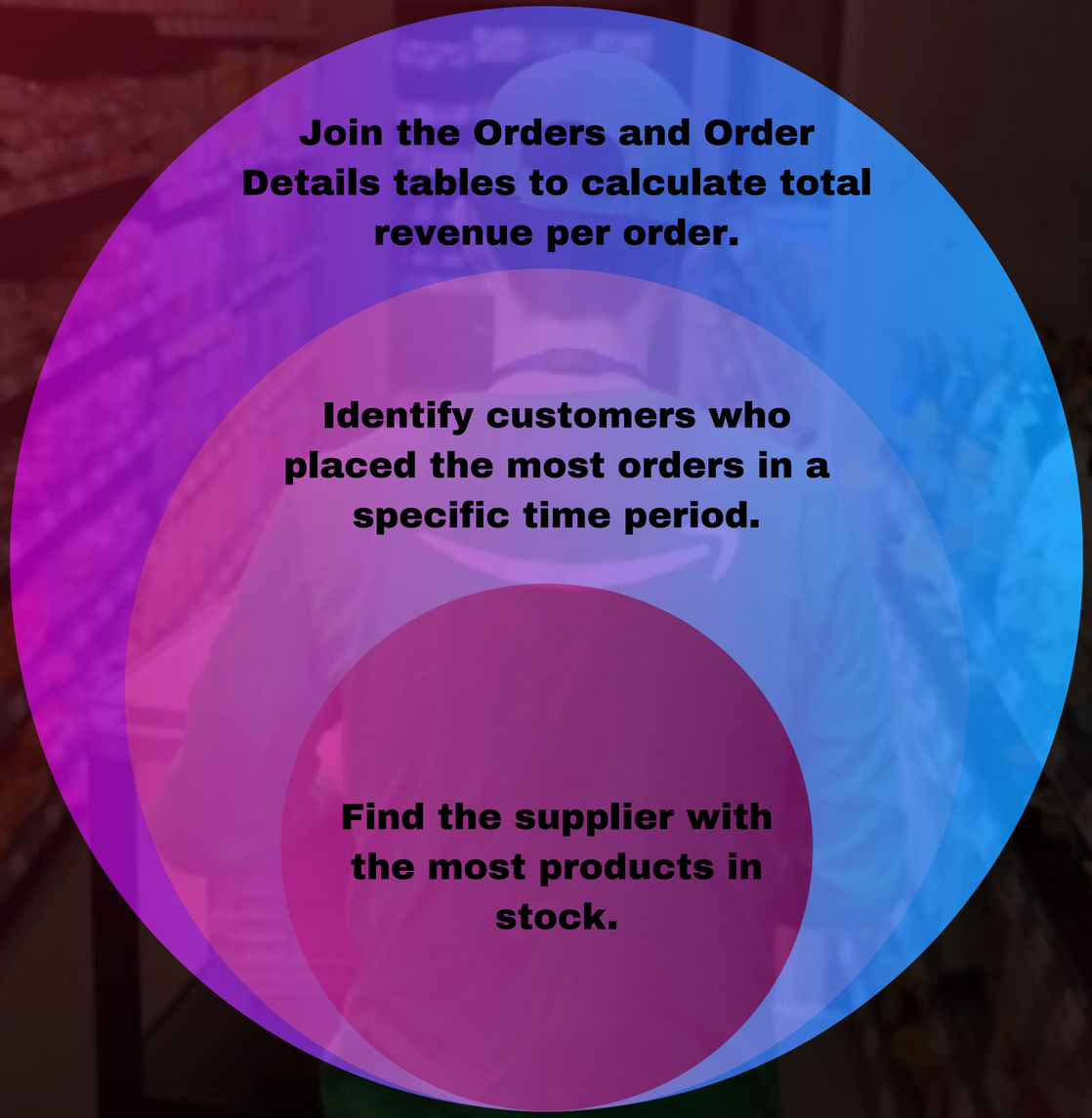
RANKED EACH PRODUCT BASED ON THE TOTAL SALES.

TASK 10

Calculate each customer's total spending.

Rank customers based on their spending.

Identify customers who have spent more than ₹5,000.



TASK 11

USED AGGREGATE FUNCTION SUM IN THE ORDER AMOUNT COLUMN FROM ORDERS TABLE AND PULLED TOTAL REVENUE PER ORDERS

USED AGGREGATE FUNCTION COUNT TO PULL THE COUNT OF THE CUSTOMERS WHO PLACED MOST ORDERS.

USED LEFT JOIN AND COMBINED TWO DIFFERENT TABLES TO RETRIEVE SUPPLIERS NAME WHO HAS MOST PRODUCT IN STOCK.

TASK 12

Separate product categories and subcategories into a new table.

CREATED A NEW TABLE IN THE NAME OF PRODUCT_CAT

Create foreign keys to maintain relationships.

IN THE PRODUCT_CAT TABLE, CREATED PRODUCT_ID COLUMN TO MAINTAIN A RELATIONSHIP WITH PRODUCTS TABLE. PRODUCT_ID COLUMN FROM PRODUCT_CAT IS A FOREIGN KEY.

TASK 13

Identify the top 3 products based on sales revenue.

JOINED THREE DIFFERENT TABLES BY USING FULL JOIN. IN ‘ORDER BY’ CLAUSE USED ‘FETCH FIRST 3 ROWS ONLY’ CONDITION TO PULL TOP 3 PRODUCTS.

Find customers who haven’t placed any orders yet.

JOINED CUSTOMERS AND ORDERS TABLES TO PULL USER WHO DOESN’T PLACED ANY ORDERS YET.

Which cities have the highest concentration of Prime members?

USED AGGREGATE FUNCTION COUNT IN THE PRIME_MEMBERS COLUMN. SO THAT WE CAN RETRIEVE DATA WHICH CITIES HAS MOST PRIME MEMBERS.

What are the top 3 most frequently ordered categories?

JOINED TWO TABLES BY USING FULL JOIN. USER AGGREGATE FUNCTION IN THE QUANTITY COLUMN TO GET MOST FREQUENTLY ORDERED CATEGORY. APPLIED ‘LIMIT’, ‘ORDER BY - DESC’ TO GET A PRECISE DATA.

END RESULTS

- STRUCTURED RESULTS OF EACH AND EVERY TASK.
- GENERATE SQL QUERIES TO ANSWER KEY BUSINESS QUESTIONS.
- CREATE ACTIONABLE REPORTS FOR STAKEHOLDERS.
- APPLY ACID PRINCIPLES TO ENSURE DATA INTEGRITY.
- NORMALIZE DATASETS FOR OPTIMIZED DATABASE STRUCTURE.

Thank You!

E-commerce and Retail Analytics