E-COMMERCE DATABASE MANAGEMENT

A Course Based Project Submitted in Partial Fulfilment of the Requirements for the Award of the degree of

BACHELOR OF TECHNOLOGY

COMPUTER SCIENCE AND ENGINEERING-CYBER SECURITY

Submitted by

21071A6249 - R. AVINASH

21071A6260 - U. VAMSHI KRISHNA

22075A6202 - D. SHRAVYA

Under the Guidance of

Mrs.N. Sunanda

(Assistant prof. Dept of CSE-CYS, DS & (AI &DS))

DEPARTMENT OF CSE-CYS, DS & (AI &DS)

VALLURUPALLI NAGESWARARAO VIGNANA JYOTHI

INSTITUTE OF ENGINEERING AND TECHNOLOGY

(An Autonomous Institute, NAAC Accredited With ‘A++’ Grade, NBA Accredited, Approved by AICTE, New Delhi, Affiliated to JNTUH

VALLURUPALLI NAGESWARARAO VIGNANA JYOTHI

INSTITUTE OF ENGINEERING AND TECHNOLOGY

(An Autonomous Institute)

CERTIFICATE

This is to Certify that

R. AVINASH (21071A6249), U. VAMSHI KRISHNA (21071A6260), D. SHRAVYA (22075A6202) have successfully completed their project work at CSE CYS, DS & (AI & DS) Department of VNRVJIET, Hyderabad entitled

"E-COMMERCE DATABASE MANAGEMENT" in partial fulfilment of the requirements for the award of the Bachelor of Technology degree during the Academic year 2022-2023

Project Guide	Head of Department
Mrs.N. Sunanda	Dr.M. Rajasekhar
Assistant Prof. and Internal Guide	Prof. and Head
Dept. of CSE-CYS, DS and AI&DS	Dept. of CSE-CYS, DS and AI&DS
VNRVJIET	VNRVJIET

DECLARATION

This is to certify that the project work entitled **" E-COMMERCE DATABASE MANAGEMENT SYSTEM "** submitted in VNR Vignana Jyothi Institute of Engineering & Technology in partial fulfilment of requirement for the award of Bachelor of Technology in Computer Science and Engineering. It is a Bonafide report of the work carried out by us under the guidance and supervision of Mrs.N.Sunanda(Assistant Professor), Department of CSE-CYS,DS,AI&DS, VNRVJIET. To the best of our knowledge, this report has not been submitted in any form to any university or institution for the award of any degree or diploma.

R. AVINASH	U. VAMSHI KRISHNA	D. SHRAVYA
(21071A6249)	(21071A6260)	(22075A6202)
II B. TECH - CSE-CYS	II B. TECH – CSE-CYS	II B. TECH – CSE-CYS
VNRVJIET	VNRVJIET	VNRVJIET

ACKNOWLEDGEMNET

Behind every achievement lies the heartfelt gratitude to those who activated in completing the project. To them we lay the words of gratitude within us.

We are indebted to our venerable principal **Dr. C.D. NAIDU** for this inflicting devotion, which led us to complete this project. The support, encouragement given by him and his motivation led us to complete the project.

We express our sincere thanks to internal guide **Mrs.N. Sunanda** and also Head of the Department **Dr. M. RAJA SHEKHAR** for having provided us a lot of facilities to undertake the project work and guide us to complete the project.

We take the opportunity to express thanks to our faculty of the Dept. of **COMPUTER SCIENCE AND ENGINEERING-CYBER SECURITY** and remaining members of our college **VNR VIGNANA JYOTHI INSTITUTE OF ENGINEERING AND TECHNOLOGY** who extended their valuable support in helping us to complete the project in time.

R. AVINASH (21071A6249)
U. VAMSHI KRISHNA (21071A6260)
D.SHRAVYA (22075A6202)

ABSTRACT

An ecommerce database is a database that is specifically designed to store and manage data for an ecommerce business. An ecommerce database is an important tool for any ecommerce business, as it allows you to efficiently store, manage, and retrieve data related to your customers, products, and orders.

For example, it might include tables for storing customer information, product information, and order information, as well as relationships between these tables to allow for easy linking of data.

Some of the key components of an ecommerce database might include:

Product information: This can include details about the products being sold, such as the name, price and any other relevant information.

Customer information : This can include details about the customers who are making purchases, such as their name, contact information.

Order information: This can include details about the orders that have been placed, such as the products that were purchased, the quantities, the prices, the shipping details, and any other relevant information.

CONTENTS

INTRODUCTION

E-commerce database management involves creating and maintaining a database to manage an online store's data. This includes managing customer information, product information, orders, and other related data.

An ecommerce database for orders and customers is a crucial aspect of any online retail business. This database serves as the foundation of an ecommerce site, tracking and managing all the essential customer and order data. The database includes customer information such as names, contact details, and addresses, along with order details, such as order date, payment method, and shipping information.
The ecommerce database provides a seamless shopping experience for customers, allowing them to easily browse products, place orders, and track shipments.

To create an e-commerce database, you need to start by identifying the data entities that need to be stored. This typically includes customers, products, orders, and order items. We will then need to create tables for each of these entities, with columns to store relevant information.

Once the tables are created, we can begin to insert data into them. This can be done manually or via an automated process. It's important to ensure that data is entered accurately and consistently to ensure the database functions correctly.
As the database is used, it will need to be maintained and optimized to ensure it performs efficiently. This includes tasks such as regularly backing up the database, optimizing queries, and monitoring for any issues or errors.

Overall, effective e-commerce database management is critical for the success of an online store. By ensuring that customer, product, and order data is properly managed and maintained, businesses can streamline their operations, improve customer satisfaction, and ultimately increase sales.

DATABASE SCHEMA

1.Customers Table

customer_id	(number, primary key)
first_name	(varchar2)
last_name	(varchar2)
email	(varchar2)
billing_address	(varchar2)
shipping_address	(varchar2)

In this schema, customer_id is used as the primary key to uniquely identify each customer. The first_name and last_name columns store the customer's name, and the email column stores the customer's email address.
The billing_address and shipping_address columns store the customer's billing and shipping addresses, respectively.

2.Products Table

product_id	(number, primary key)
name	(varchar2)
description	(varcha2)
price	(number)

In this schema, product_id is used as the primary key to uniquely identify each product. The name column stores the product name, the description column stores a more detailed description of the product, and the price column stores the price of the product.

We may even include additional columns like image_url, category_id, or in_stock depending on the needs of e-commerce store.

3.Orders Table

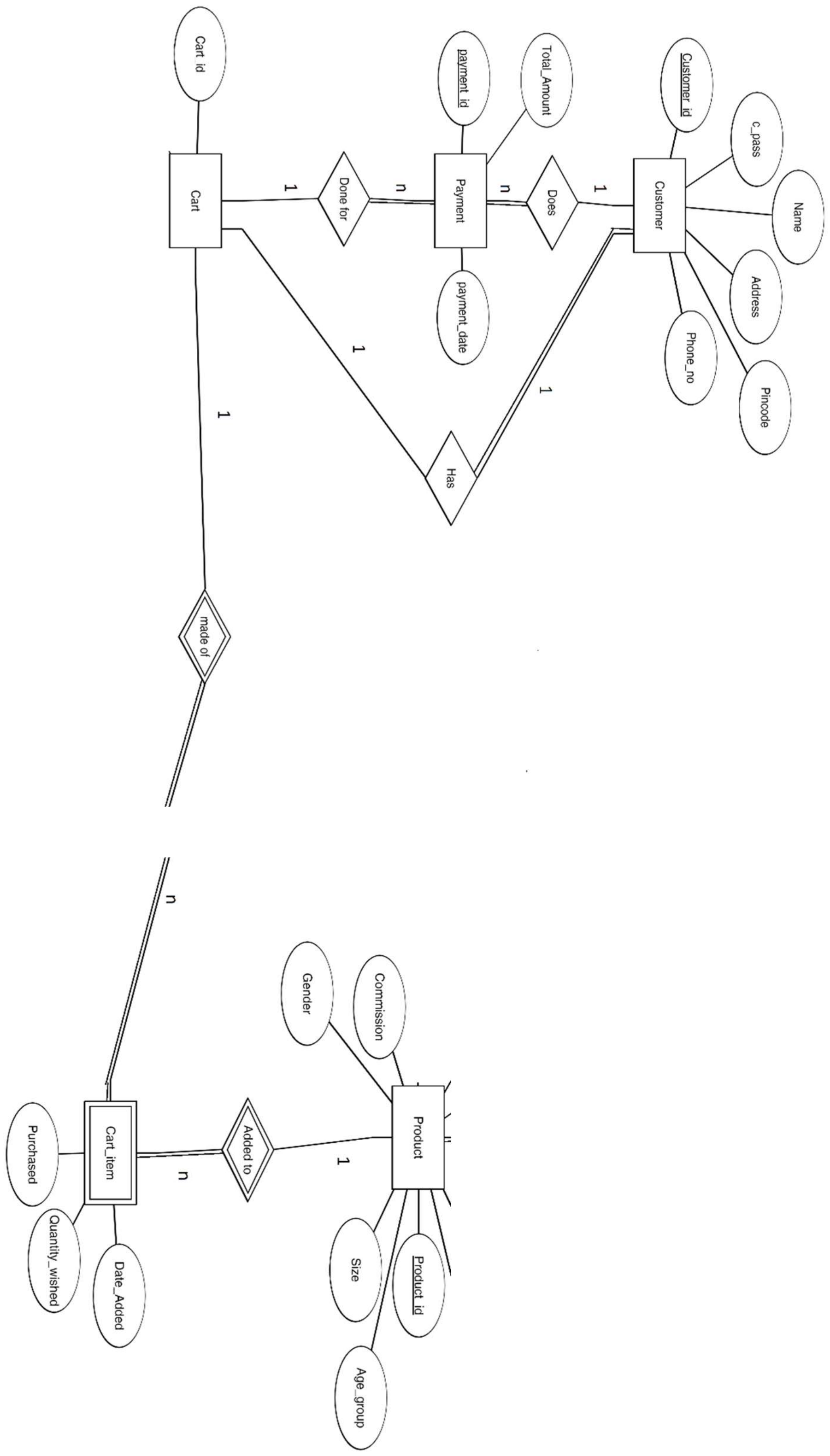
order_id	(number, primary key)
customer_id	(number, foreign key references customers(customer_id))
date	(date)

In this schema, order_id is used as the primary key to uniquely identify each order. The customer_id column stores the foreign key reference to the customer_id column in the customers table, which allows you to link each order to the customer who placed it. The date column stores the date when the order was placed.

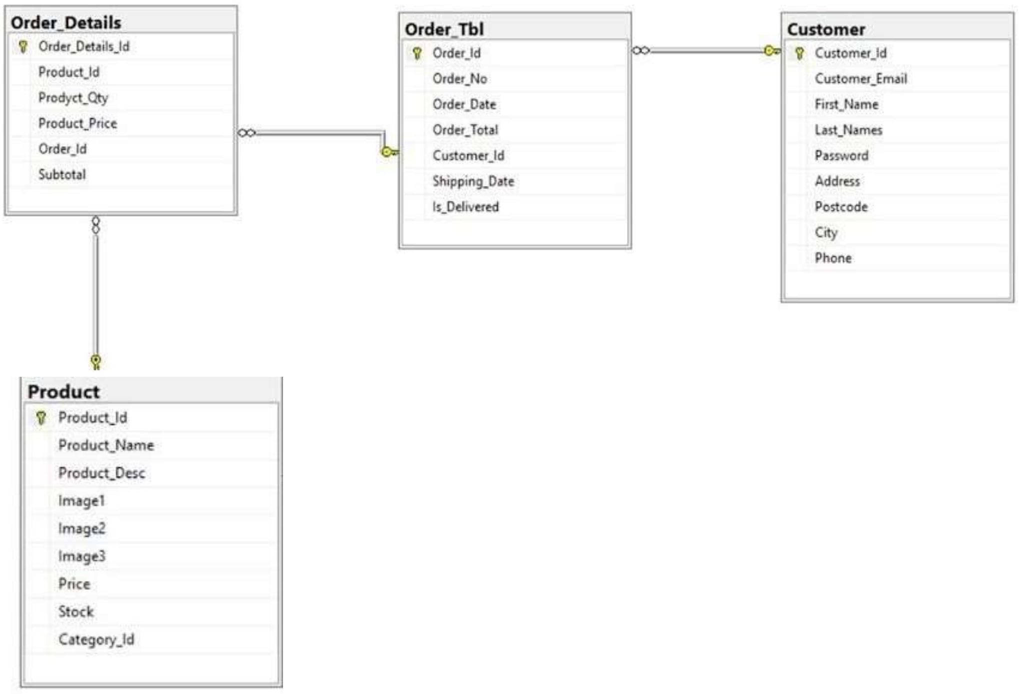
4.Order_items Table

Order_item_id	(number,primary key)
Order_id	(number,foreign key references order(order_id))
Product_id	(number,foreign key references products(product_id))
Quantity	(number)
Price	(number)

ER DIAGRAM

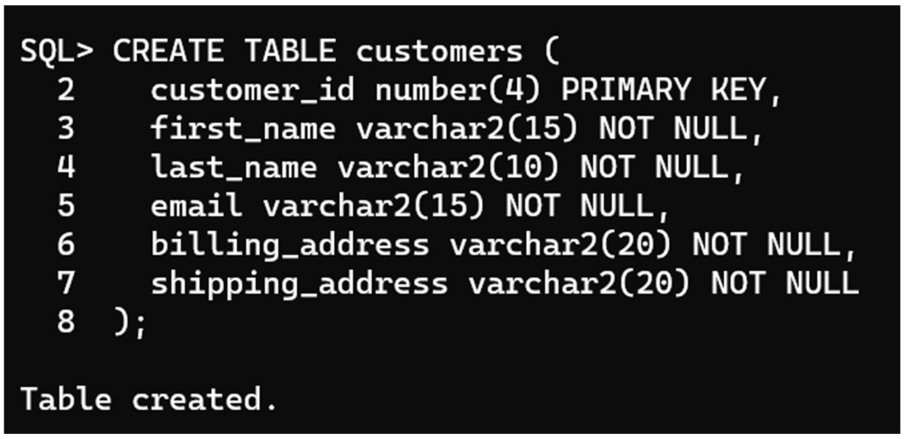

Cart id
Cart
Done for
Payment
payment_id
Total_Amount
payment_date
Does
Customer
Customer_id
c_pass
Name
Address
Phone_no
Pincode
Has
made of
Cart_item
Purchased
Quantity_wished
Date_Added
Added to
Product
Gender
Commission
Size
Product_id
Age_group
1
n
1
n
1
1
1
1
n
n
1

REPRESENTATION

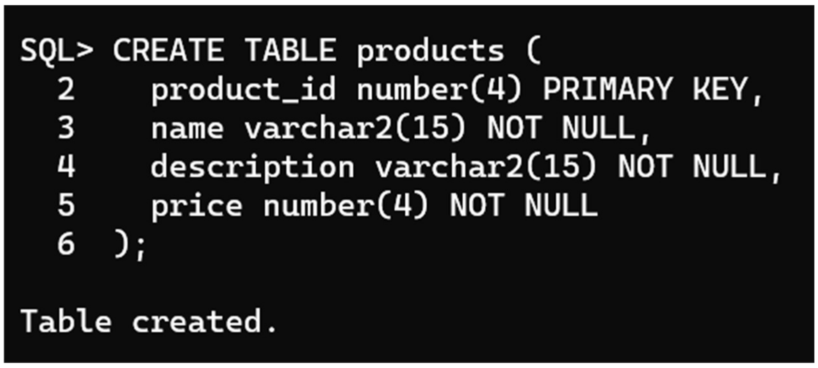


SQL IMPLEMETATION

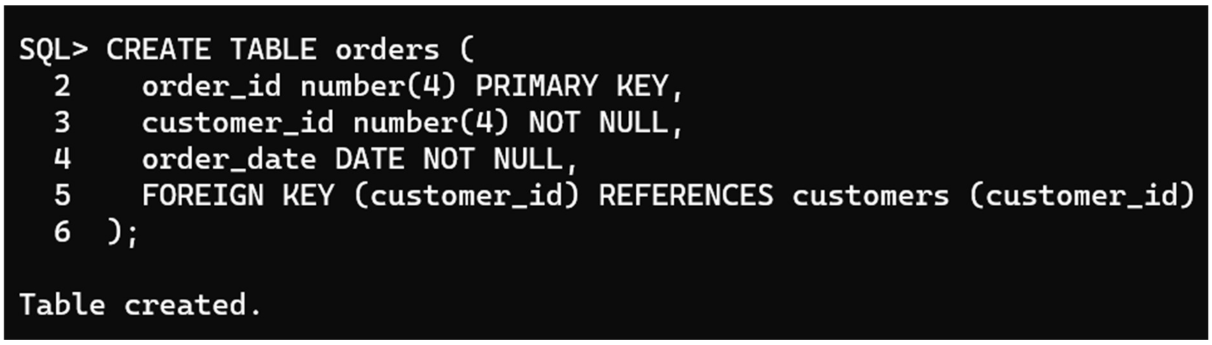
Creation of Tables



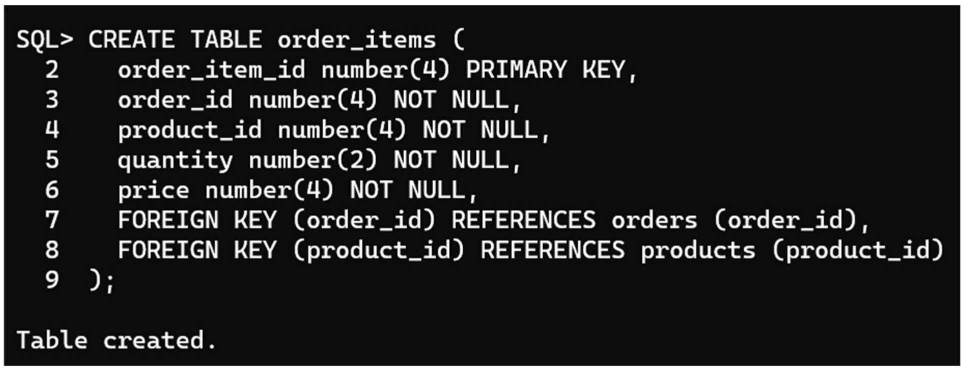
```
SQL> CREATE TABLE customers (
  2    customer_id number(4) PRIMARY KEY,
  3    first_name varchar2(15) NOT NULL,
  4    last_name varchar2(10) NOT NULL,
  5    email varchar2(15) NOT NULL,
  6    billing_address varchar2(20) NOT NULL,
  7    shipping_address varchar2(20) NOT NULL
  8  );

Table created.
```



```
SQL> CREATE TABLE products (
  2    product_id number(4) PRIMARY KEY,
  3    name varchar2(15) NOT NULL,
  4    description varchar2(15) NOT NULL,
  5    price number(4) NOT NULL
  6  );

Table created.
```

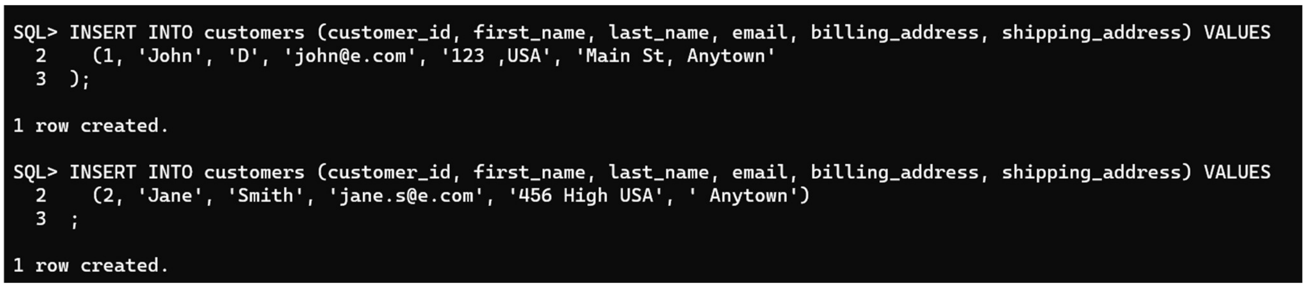


```
SQL> CREATE TABLE orders (
  2    order_id number(4) PRIMARY KEY,
  3    customer_id number(4) NOT NULL,
  4    order_date DATE NOT NULL,
  5    FOREIGN KEY (customer_id) REFERENCES customers (customer_id)
  6  );

Table created.
```

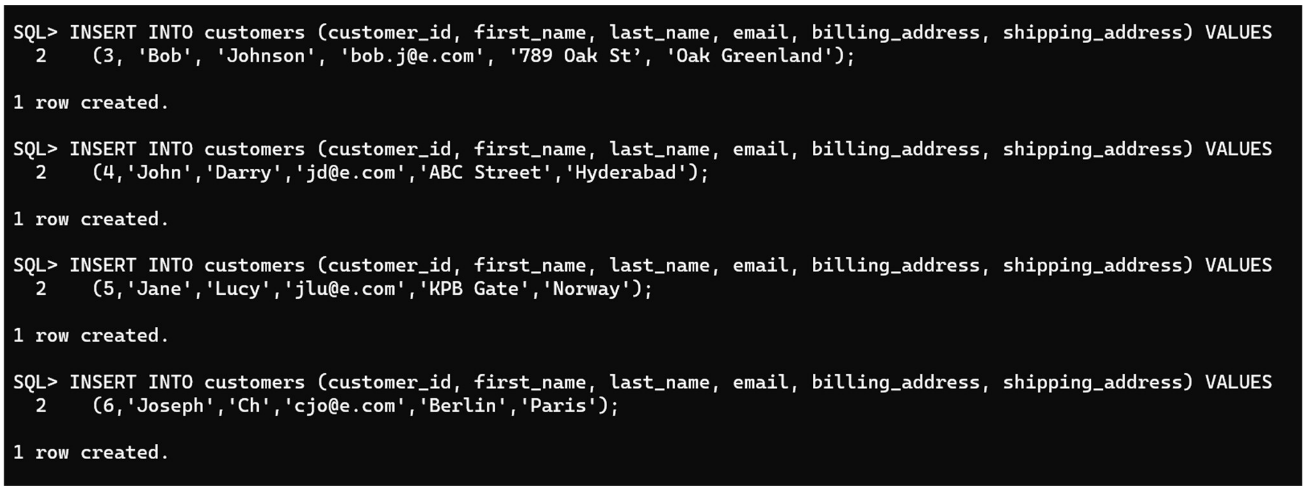


```
SQL> CREATE TABLE order_items (
  2    order_item_id number(4) PRIMARY KEY,
  3    order_id number(4) NOT NULL,
  4    product_id number(4) NOT NULL,
  5    quantity number(2) NOT NULL,
  6    price number(4) NOT NULL,
  7    FOREIGN KEY (order_id) REFERENCES orders (order_id),
  8    FOREIGN KEY (product_id) REFERENCES products (product_id)
  9  );

Table created.
```

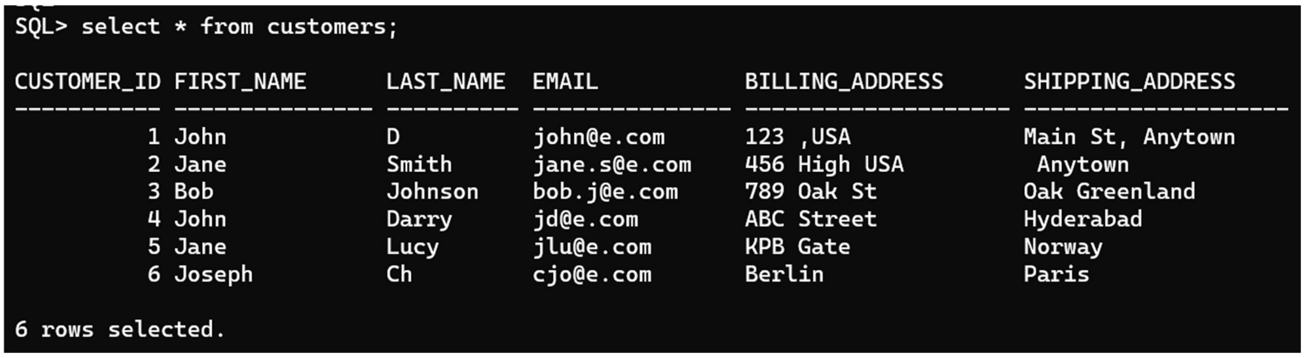
Insertion of Values



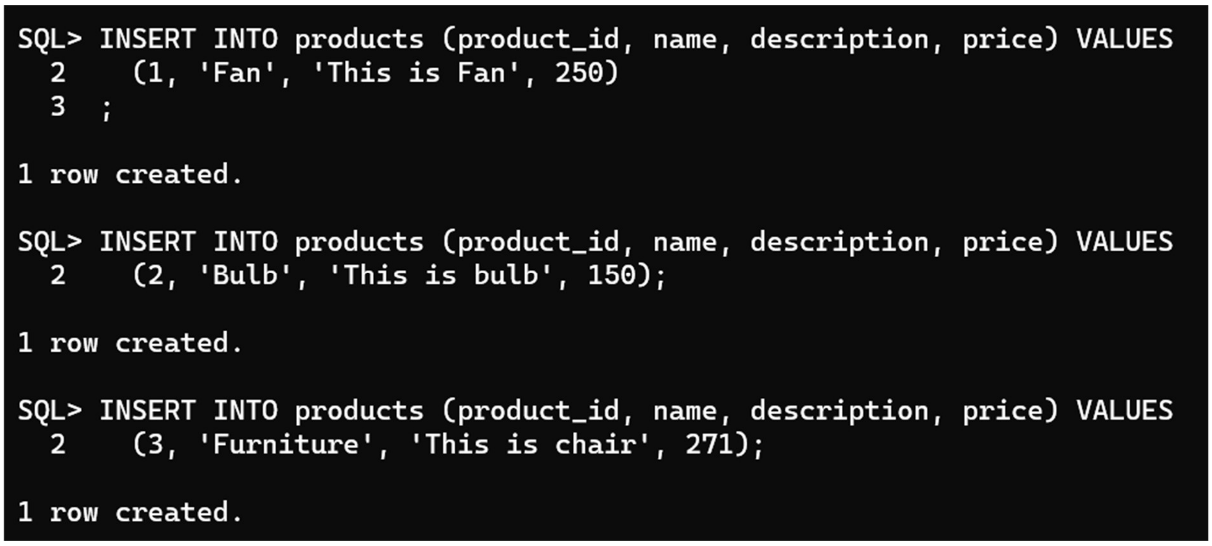
```
SQL> INSERT INTO customers (customer_id, first_name, last_name, email, billing_address, shipping_address) VALUES
  2    (1, 'John', 'D', 'john@e.com', '123 ,USA', 'Main St, Anytown'
  3  );

1 row created.

SQL> INSERT INTO customers (customer_id, first_name, last_name, email, billing_address, shipping_address) VALUES
  2    (2, 'Jane', 'Smith', 'jane.s@e.com', '456 High USA', ' Anytown')
  3  ;

1 row created.
```



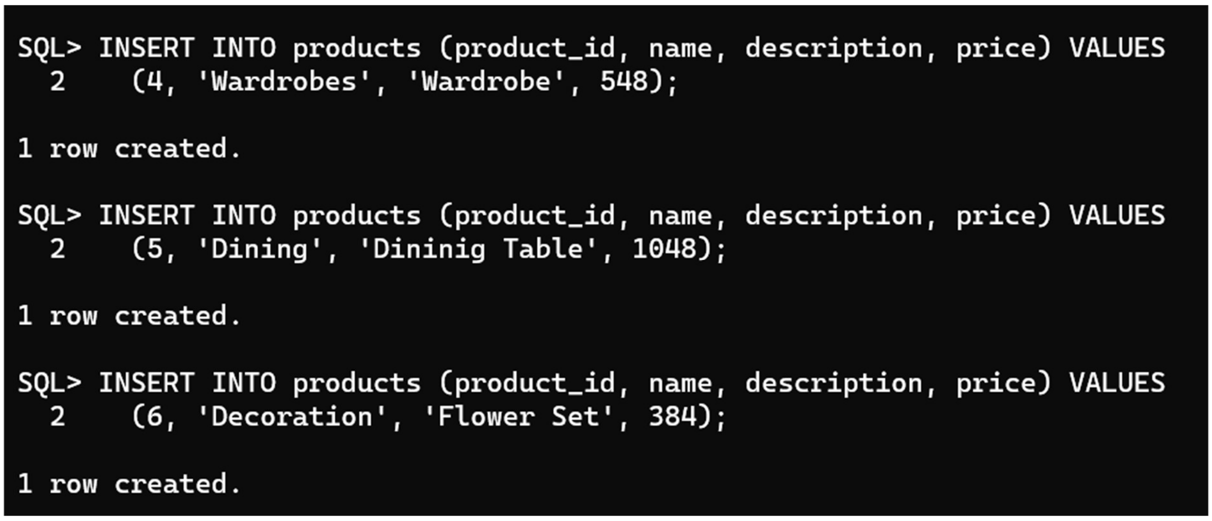
```
SQL> INSERT INTO customers (customer_id, first_name, last_name, email, billing_address, shipping_address) VALUES
  2    (3, 'Bob', 'Johnson', 'bob.j@e.com', '789 Oak St’, 'Oak Greenland');

1 row created.

SQL> INSERT INTO customers (customer_id, first_name, last_name, email, billing_address, shipping_address) VALUES
  2    (4,'John','Darry','jd@e.com','ABC Street','Hyderabad');

1 row created.

SQL> INSERT INTO customers (customer_id, first_name, last_name, email, billing_address, shipping_address) VALUES
  2    (5,'Jane','Lucy','jlu@e.com','KPB Gate','Norway');

1 row created.

SQL> INSERT INTO customers (customer_id, first_name, last_name, email, billing_address, shipping_address) VALUES
  2    (6,'Joseph','Ch','cjo@e.com','Berlin','Paris');

1 row created.
```



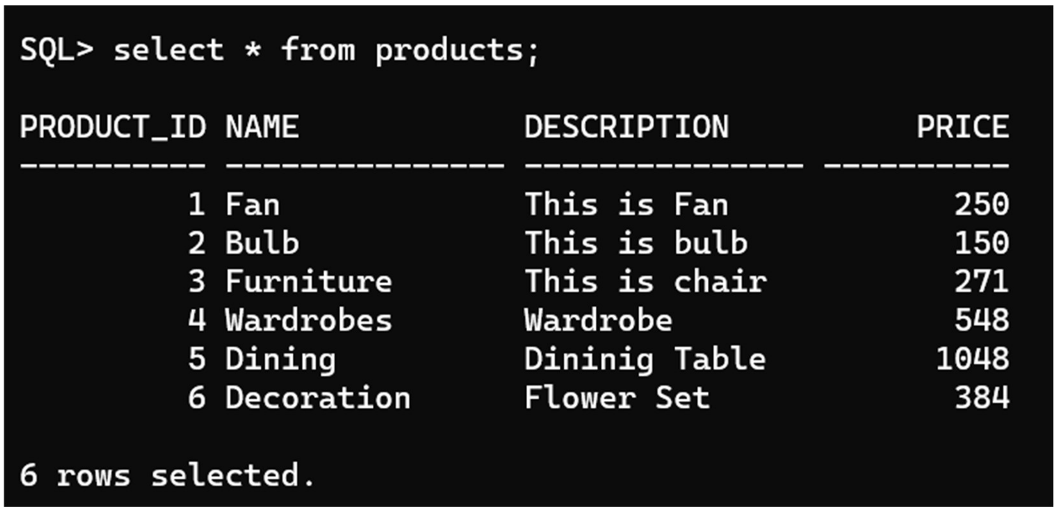
```
SQL> select * from customers;

CUSTOMER_ID FIRST_NAME      LAST_NAME  EMAIL           BILLING_ADDRESS      SHIPPING_ADDRESS
----------- --------------- ---------- --------------- -------------------- --------------------
          1 John            D          john@e.com      123 ,USA             Main St, Anytown
          2 Jane            Smith      jane.s@e.com    456 High USA          Anytown
          3 Bob             Johnson    bob.j@e.com     789 Oak St           Oak Greenland
          4 John            Darry      jd@e.com        ABC Street           Hyderabad
          5 Jane            Lucy       jlu@e.com       KPB Gate             Norway
          6 Joseph          Ch         cjo@e.com       Berlin               Paris

6 rows selected.
```



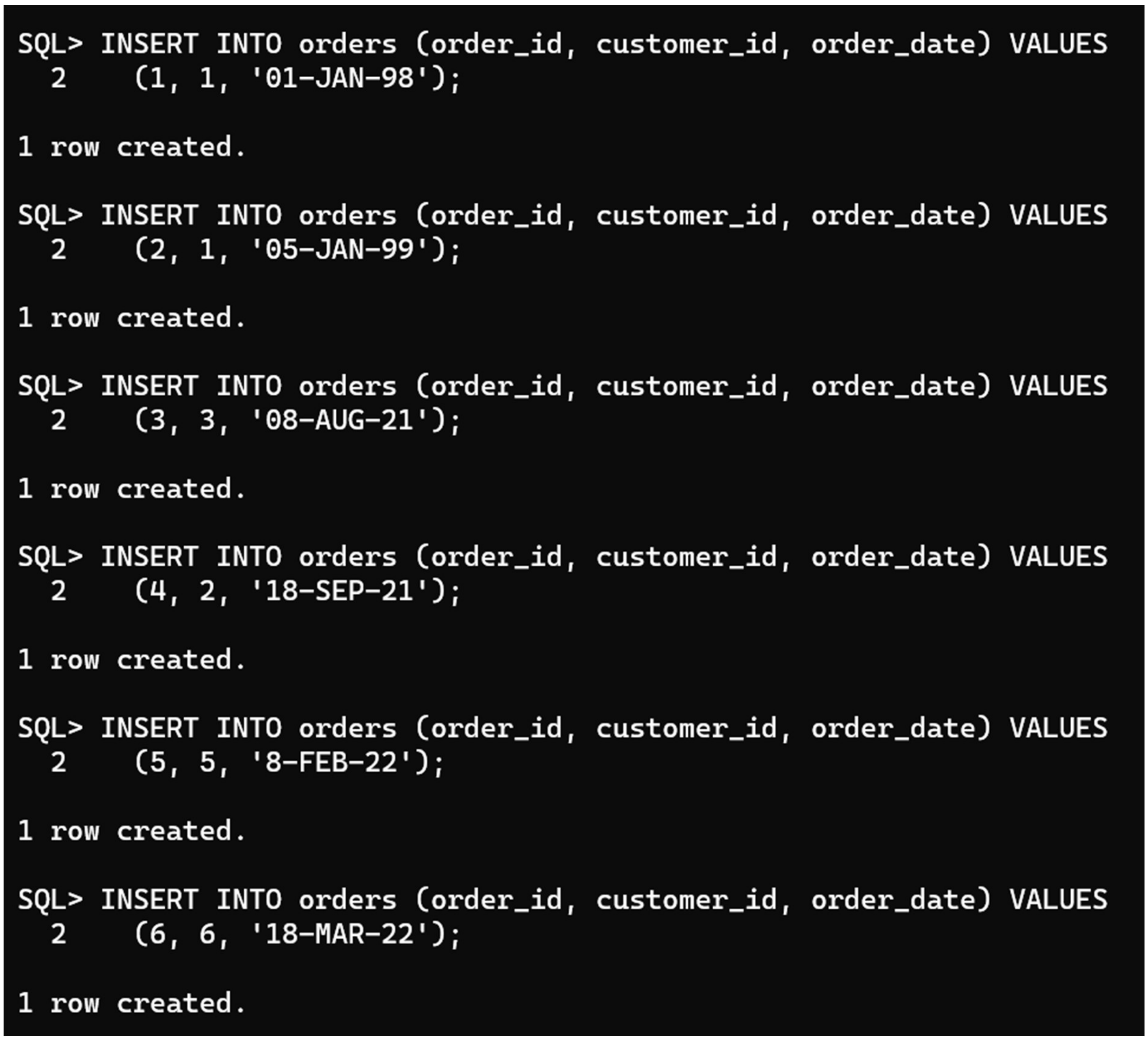
```
SQL> INSERT INTO products (product_id, name, description, price) VALUES
  2    (1, 'Fan', 'This is Fan', 250)
  3  ;

1 row created.

SQL> INSERT INTO products (product_id, name, description, price) VALUES
  2    (2, 'Bulb', 'This is bulb', 150);

1 row created.

SQL> INSERT INTO products (product_id, name, description, price) VALUES
  2    (3, 'Furniture', 'This is chair', 271);

1 row created.
```



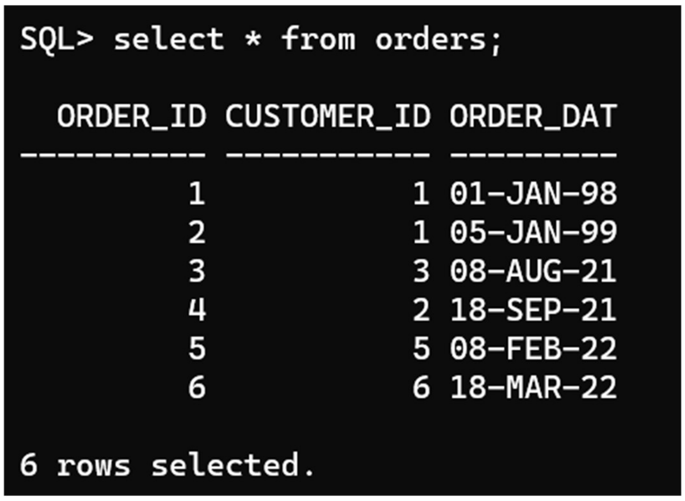
```
SQL> INSERT INTO products (product_id, name, description, price) VALUES
  2    (4, 'Wardrobes', 'Wardrobe', 548);

1 row created.

SQL> INSERT INTO products (product_id, name, description, price) VALUES
  2    (5, 'Dining', 'Dininig Table', 1048);

1 row created.

SQL> INSERT INTO products (product_id, name, description, price) VALUES
  2    (6, 'Decoration', 'Flower Set', 384);

1 row created.
```



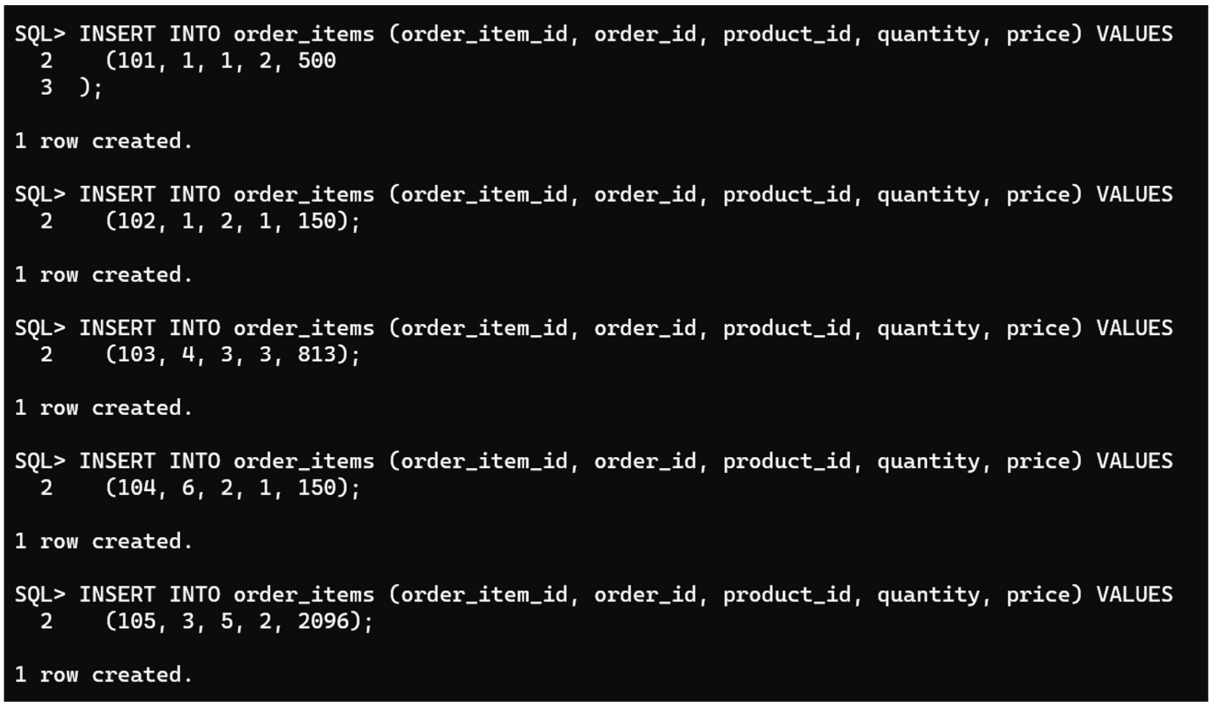
```
SQL> select * from products;

PRODUCT_ID NAME            DESCRIPTION          PRICE
---------- --------------- --------------- ----------
         1 Fan             This is Fan            250
         2 Bulb            This is bulb           150
         3 Furniture       This is chair          271
         4 Wardrobes       Wardrobe               548
         5 Dining          Dininig Table         1048
         6 Decoration      Flower Set             384

6 rows selected.
```



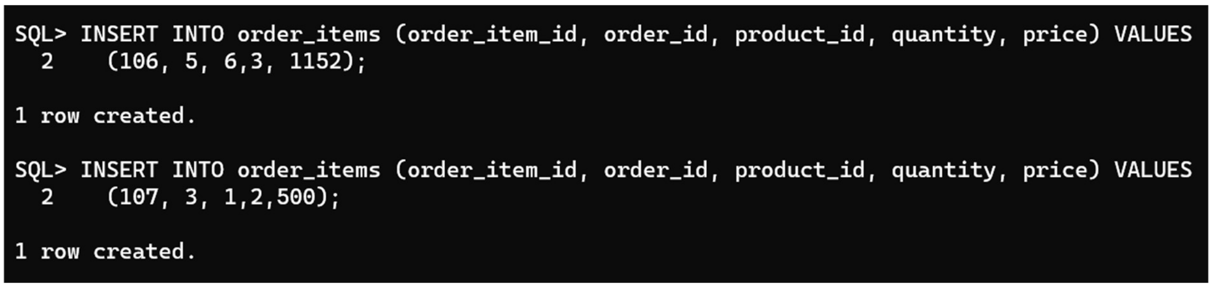
```
SQL> INSERT INTO orders (order_id, customer_id, order_date) VALUES
  2    (1, 1, '01-JAN-98');

1 row created.

SQL> INSERT INTO orders (order_id, customer_id, order_date) VALUES
  2    (2, 1, '05-JAN-99');

1 row created.

SQL> INSERT INTO orders (order_id, customer_id, order_date) VALUES
  2    (3, 3, '08-AUG-21');

1 row created.

SQL> INSERT INTO orders (order_id, customer_id, order_date) VALUES
  2    (4, 2, '18-SEP-21');

1 row created.

SQL> INSERT INTO orders (order_id, customer_id, order_date) VALUES
  2    (5, 5, '8-FEB-22');

1 row created.

SQL> INSERT INTO orders (order_id, customer_id, order_date) VALUES
  2    (6, 6, '18-MAR-22');

1 row created.
```



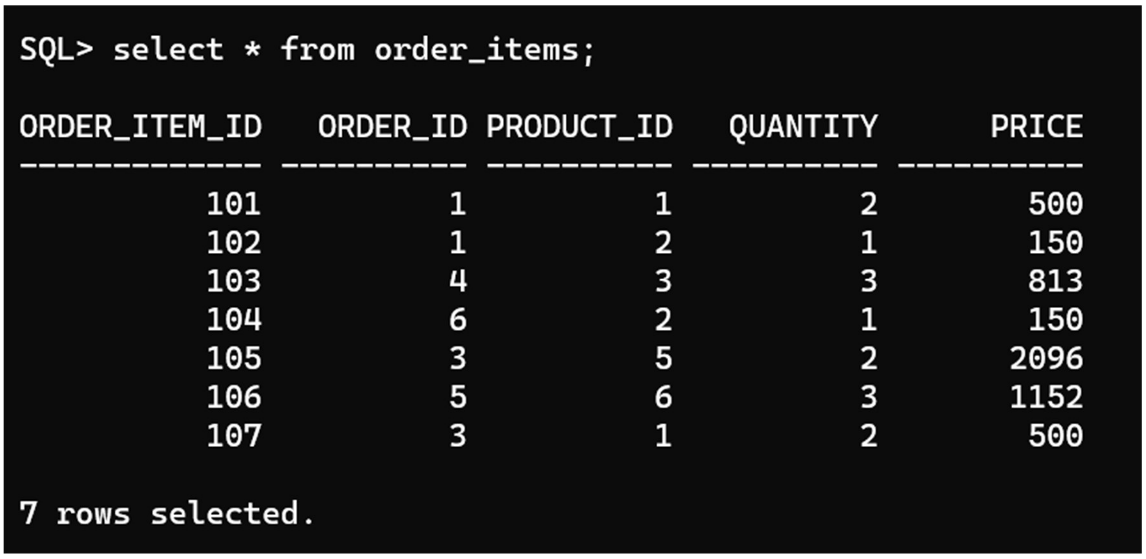
```
SQL> select * from orders;

  ORDER_ID CUSTOMER_ID ORDER_DAT
---------- ----------- ---------
         1           1 01-JAN-98
         2           1 05-JAN-99
         3           3 08-AUG-21
         4           2 18-SEP-21
         5           5 08-FEB-22
         6           6 18-MAR-22

6 rows selected.
```



```
SQL> INSERT INTO order_items (order_item_id, order_id, product_id, quantity, price) VALUES
  2    (101, 1, 1, 2, 500
  3  );

1 row created.

SQL> INSERT INTO order_items (order_item_id, order_id, product_id, quantity, price) VALUES
  2    (102, 1, 2, 1, 150);

1 row created.

SQL> INSERT INTO order_items (order_item_id, order_id, product_id, quantity, price) VALUES
  2    (103, 4, 3, 3, 813);

1 row created.

SQL> INSERT INTO order_items (order_item_id, order_id, product_id, quantity, price) VALUES
  2    (104, 6, 2, 1, 150);

1 row created.

SQL> INSERT INTO order_items (order_item_id, order_id, product_id, quantity, price) VALUES
  2    (105, 3, 5, 2, 2096);

1 row created.
```



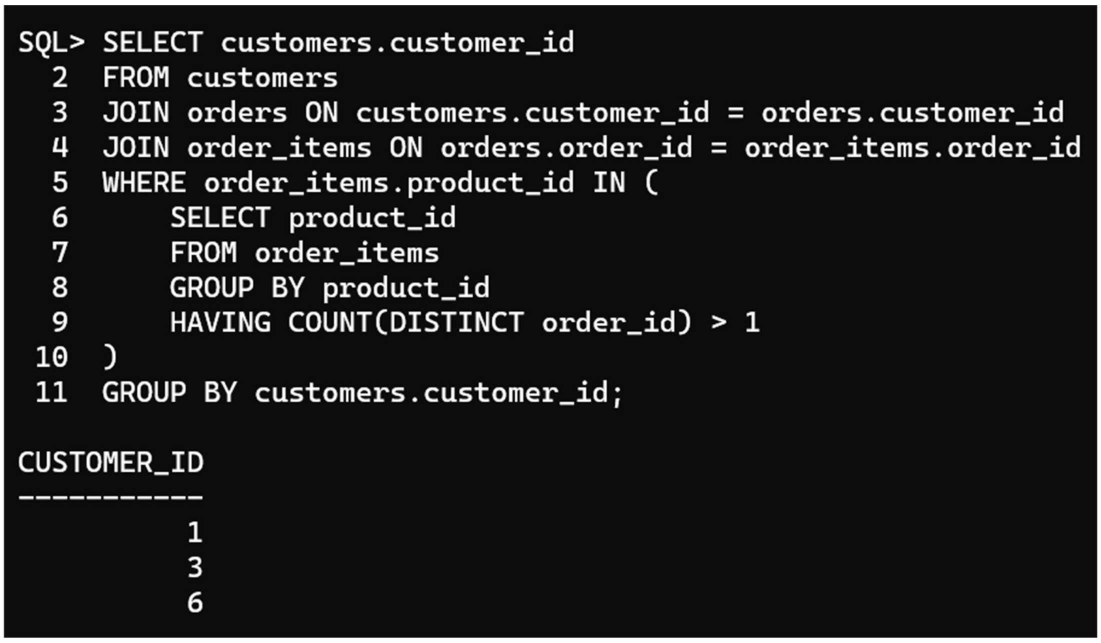
```
SQL> INSERT INTO order_items (order_item_id, order_id, product_id, quantity, price) VALUES
  2    (106, 5, 6,3, 1152);

1 row created.

SQL> INSERT INTO order_items (order_item_id, order_id, product_id, quantity, price) VALUES
  2    (107, 3, 1,2,500);

1 row created.
```



```
SQL> select * from order_items;

ORDER_ITEM_ID   ORDER_ID PRODUCT_ID   QUANTITY      PRICE
------------- ---------- ---------- ---------- ----------
          101          1          1          2        500
          102          1          2          1        150
          103          4          3          3        813
          104          6          2          1        150
          105          3          5          2       2096
          106          5          6          3       1152
          107          3          1          2        500

7 rows selected.
```

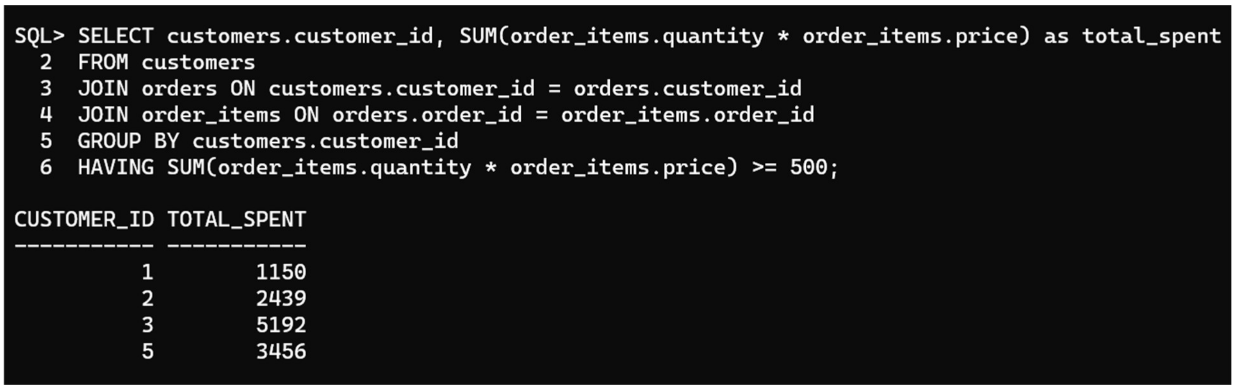
SAMPLE SQL QUERIES

1. Get the customers who have ordered the same product more than once



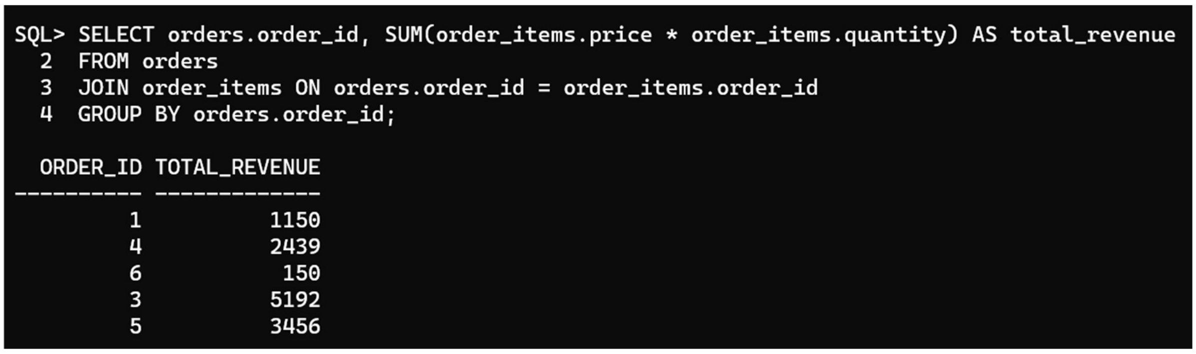
```
SQL> SELECT customers.customer_id
     FROM customers
     JOIN orders ON customers.customer_id = orders.customer_id
     JOIN order_items ON orders.order_id = order_items.order_id
     WHERE order_items.product_id IN (
         SELECT product_id
         FROM order_items
         GROUP BY product_id
         HAVING COUNT(DISTINCT order_id) > 1
     )
     GROUP BY customers.customer_id;

CUSTOMER_ID
-----------
          1
          3
          6
```

2. Get the customers who have placed orders for products with a total value of at least 500



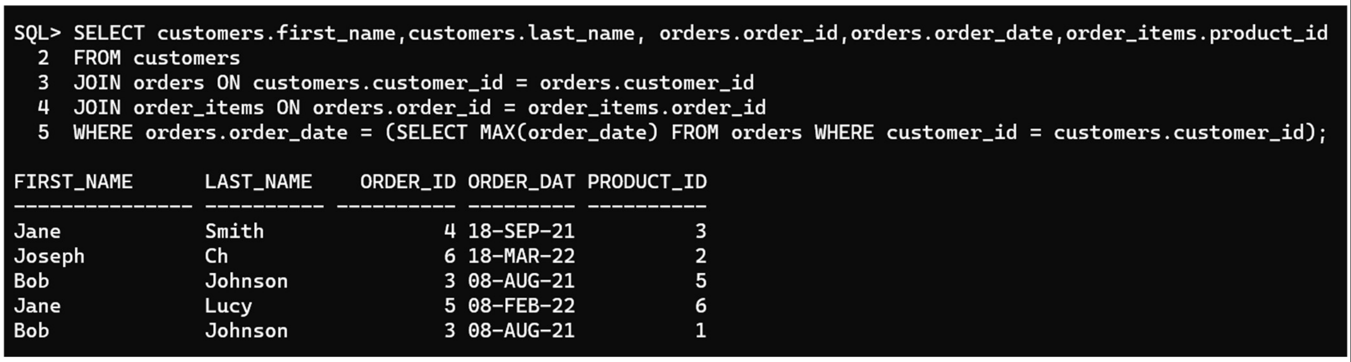
```
SQL> SELECT customers.customer_id, SUM(order_items.quantity * order_items.price) as total_spent
     FROM customers
     JOIN orders ON customers.customer_id = orders.customer_id
     JOIN order_items ON orders.order_id = order_items.order_id
     GROUP BY customers.customer_id
     HAVING SUM(order_items.quantity * order_items.price) >= 500;

CUSTOMER_ID TOTAL_SPENT
----------- -----------
          1        1150
          2        2439
          3        5192
          5        3456
```

3. Retrieve the total revenue for each order



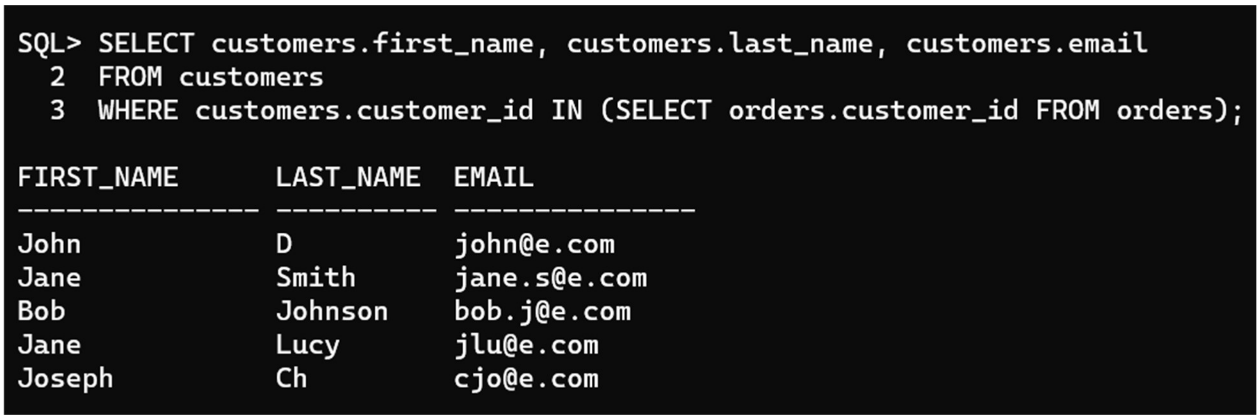
```
SQL> SELECT orders.order_id, SUM(order_items.price * order_items.quantity) AS total_revenue
     FROM orders
     JOIN order_items ON orders.order_id = order_items.order_id
     GROUP BY orders.order_id;

  ORDER_ID TOTAL_REVENUE
---------- -------------
         1          1150
         4          2439
         6           150
         3          5192
         5          3456
```

4. Retrieve the details of the most recent order for each customer



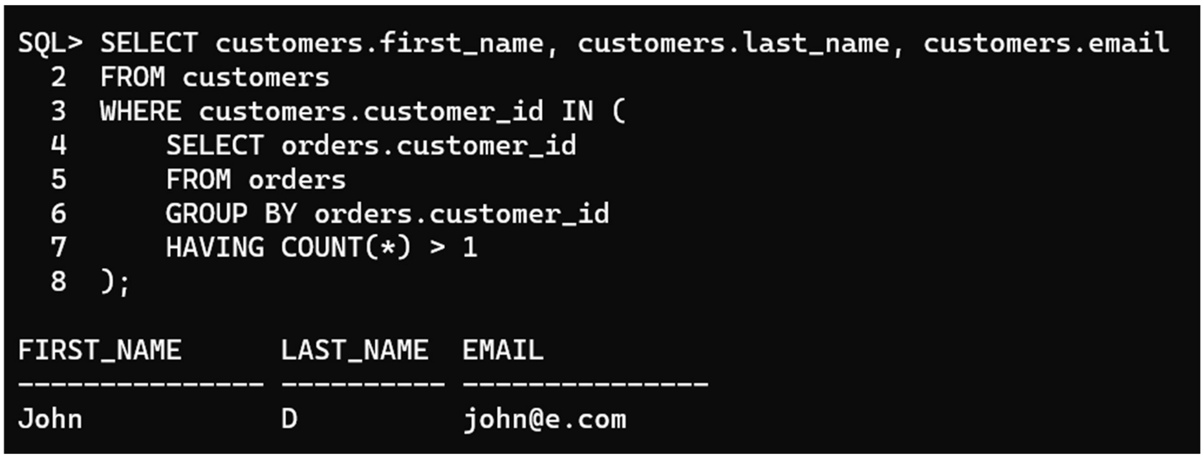
```
SQL> SELECT customers.first_name,customers.last_name, orders.order_id,orders.order_date,order_items.product_id
FROM customers
JOIN orders ON customers.customer_id = orders.customer_id
JOIN order_items ON orders.order_id = order_items.order_id
WHERE orders.order_date = (SELECT MAX(order_date) FROM orders WHERE customer_id = customers.customer_id);

FIRST_NAME      LAST_NAME    ORDER_ID ORDER_DAT PRODUCT_ID
--------------- ---------- ---------- --------- ----------
Jane            Smith               4 18-SEP-21          3
Joseph          Ch                  6 18-MAR-22          2
Bob             Johnson             3 08-AUG-21          5
Jane            Lucy                5 08-FEB-22          6
Bob             Johnson             3 08-AUG-21          1
```

5. Retrieve the name and email of customers who have placed orders



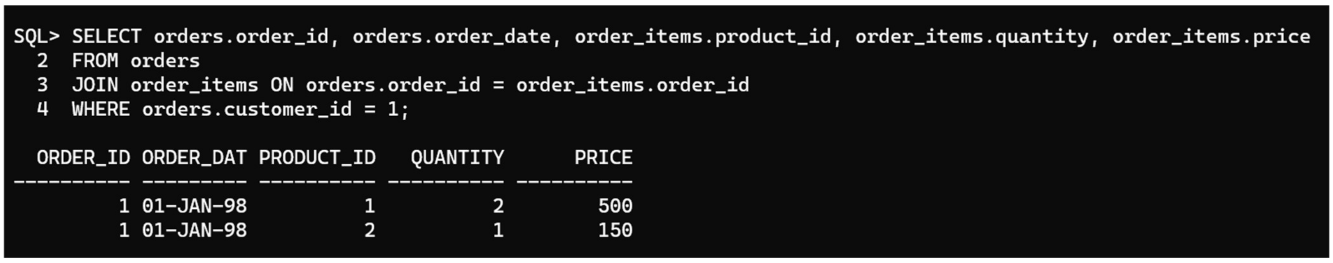
```
SQL> SELECT customers.first_name, customers.last_name, customers.email
FROM customers
WHERE customers.customer_id IN (SELECT orders.customer_id FROM orders);

FIRST_NAME      LAST_NAME  EMAIL
--------------- ---------- ---------------
John            D          john@e.com
Jane            Smith      jane.s@e.com
Bob             Johnson    bob.j@e.com
Jane            Lucy       jlu@e.com
Joseph          Ch         cjo@e.com
```

6. Retrieve the name and email of customers who have placed more than one order



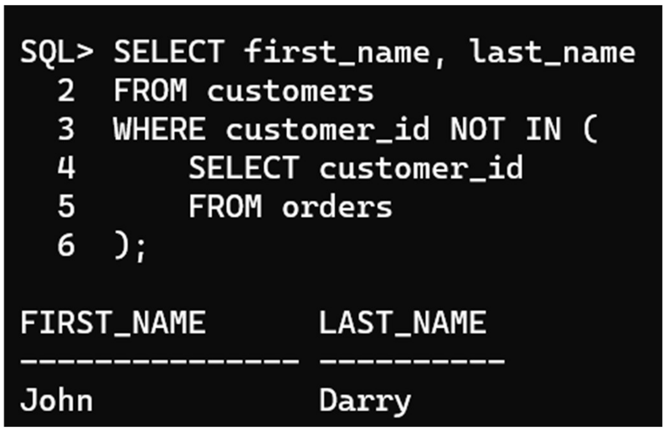
```
SQL> SELECT customers.first_name, customers.last_name, customers.email
FROM customers
WHERE customers.customer_id IN (
    SELECT orders.customer_id
    FROM orders
    GROUP BY orders.customer_id
    HAVING COUNT(*) > 1
);

FIRST_NAME      LAST_NAME  EMAIL
--------------- ---------- ---------------
John            D          john@e.com
```

7. Retrieve the details of all the orders for a specific customer with ID 1



```
SQL> SELECT orders.order_id, orders.order_date, order_items.product_id, order_items.quantity, order_items.price
  FROM orders
  JOIN order_items ON orders.order_id = order_items.order_id
  WHERE orders.customer_id = 1;

  ORDER_ID ORDER_DAT PRODUCT_ID   QUANTITY      PRICE
---------- --------- ---------- ---------- ----------
         1 01-JAN-98          1          2        500
         1 01-JAN-98          2          1        150
```

8. Select the first and last name of all customers who have not made an order



```
SQL> SELECT first_name, last_name
  FROM customers
  WHERE customer_id NOT IN (
      SELECT customer_id
      FROM orders
  );

FIRST_NAME      LAST_NAME
--------------- ----------
John            Darry
```

CONCLUSION

In conclusion, an ecommerce database project is an essential aspect of any online retail business. The project report focuses on designing and maintaining a well-structured database that tracks and manages all essential customer and order data, along with product inventory.

The database plays a crucial role in providing customers with a seamless shopping experience, making it easy for them to browse products, place orders, and track shipments. For businesses, the database helps to manage inventory, track sales, and improve marketing strategies by analyzing customer data.

The model provides a strong foundation for building an efficient and reliable database system, but businesses must continuously strive to improve and adapt to the changing market demands.