21BLC1059  
Govind Sankar H

**E-Commerce Database**

**Code**

-- Create tables

CREATE TABLE users (

user\_id INT PRIMARY KEY,

name VARCHAR(50),

email VARCHAR(50),

age INT,

address VARCHAR(100)

);

CREATE TABLE payments (

payment\_id INT PRIMARY KEY,

user\_id INT,

amount DECIMAL(10,2),

payment\_date DATE,

FOREIGN KEY (user\_id) REFERENCES users(user\_id)

);

CREATE TABLE categories (

category\_id INT PRIMARY KEY,

name VARCHAR(50)

);

CREATE TABLE products (

product\_id INT PRIMARY KEY,

name VARCHAR(50),

price DECIMAL(10,2),

category\_id INT,

FOREIGN KEY (category\_id) REFERENCES categories(category\_id)

);

CREATE TABLE cart\_items (

item\_id INT PRIMARY KEY,

user\_id INT,

product\_id INT,

quantity INT,

FOREIGN KEY (user\_id) REFERENCES users(user\_id),

FOREIGN KEY (product\_id) REFERENCES products(product\_id)

);

-- Insert data into users table

INSERT INTO users (user\_id, name, email, age, address) VALUES (1, 'Rajesh Patel', 'rajesh.patel@gmail.com', 30, '123 Dance St');

INSERT INTO users (user\_id, name, email, age, address) VALUES (2, 'Neha Gupta', 'neha.gupta@gmail.com', 25, '420 Green St');

INSERT INTO users (user\_id, name, email, age, address) VALUES (3, 'Amit Sharma', 'amit.sharma@gmail.com', 35, '690 Wood St');

INSERT INTO users (user\_id, name, email, age, address) VALUES (4, 'Priya Singh', 'priya.singh@gmail.com', 28, '911 Nehru St');

INSERT INTO users (user\_id, name, email, age, address) VALUES (5, 'Anil Verma', 'anil.verma@gmail.com', 32, '764 Cedar St');

-- Insert data into payments table

INSERT INTO payments (payment\_id, user\_id, amount, payment\_date) VALUES (1, 1, 100.00, TO\_DATE('2023-07-15', 'YYYY-MM-DD'));

INSERT INTO payments (payment\_id, user\_id, amount, payment\_date) VALUES (2, 2, 50.00, TO\_DATE('2023-07-14', 'YYYY-MM-DD'));

INSERT INTO payments (payment\_id, user\_id, amount, payment\_date) VALUES (3, 3, 75.00, TO\_DATE('2023-07-13', 'YYYY-MM-DD'));

INSERT INTO payments (payment\_id, user\_id, amount, payment\_date) VALUES (4, 1, 150.00, TO\_DATE('2023-07-12', 'YYYY-MM-DD'));

INSERT INTO payments (payment\_id, user\_id, amount, payment\_date) VALUES (5, 3, 200.00, TO\_DATE('2023-07-16', 'YYYY-MM-DD'));

INSERT INTO payments (payment\_id, user\_id, amount, payment\_date) VALUES (6, 5, 75.00, TO\_DATE('2023-07-17', 'YYYY-MM-DD'));

-- Insert data into categories table

INSERT INTO categories (category\_id, name) VALUES (1, 'Electronics');

INSERT INTO categories (category\_id, name) VALUES (2, 'Grocery');

INSERT INTO categories (category\_id, name) VALUES (3, 'Clothing');

-- Insert data into products table

INSERT INTO products (product\_id, name, price, category\_id) VALUES (1, 'iPhone', 299999.99, 1);

INSERT INTO products (product\_id, name, price, category\_id) VALUES (2, 'Chocolate', 42.53, 2);

INSERT INTO products (product\_id, name, price, category\_id) VALUES (3, 'shirt', 1250.57, 3);

-- Insert data into cart\_items table

INSERT INTO cart\_items (item\_id, user\_id, product\_id, quantity) VALUES (1, 1, 2, 2);

INSERT INTO cart\_items (item\_id, user\_id, product\_id, quantity) VALUES (2, 2, 2, 1);

INSERT INTO cart\_items (item\_id, user\_id, product\_id, quantity) VALUES (3, 3, 3, 3);

INSERT INTO cart\_items (item\_id, user\_id, product\_id, quantity) VALUES (4, 1, 2, 3);

INSERT INTO cart\_items (item\_id, user\_id, product\_id, quantity) VALUES (5, 3, 3, 1);

INSERT INTO cart\_items (item\_id, user\_id, product\_id, quantity) VALUES (6, 4, 1, 2);

INSERT INTO cart\_items (item\_id, user\_id, product\_id, quantity) VALUES (7, 3, 2, 1);

INSERT INTO cart\_items (item\_id, user\_id, product\_id, quantity) VALUES (8, 5, 3, 2);

-- Display all users

SELECT 'User ID: ' || user\_id || ', Name: ' || name || ', Email: ' || email || ', Age: ' || age || ', Address: ' || address AS user\_info FROM users;

-- Update user's email using PL/SQL block

BEGIN

UPDATE users SET email = 'rajesh@gmail.com' WHERE user\_id = 1;

COMMIT;

END;

/

-- Delete a user using PL/SQL block

DECLARE

user\_id\_to\_delete INT := 3;

BEGIN

-- Delete related child records

DELETE FROM cart\_items WHERE user\_id = user\_id\_to\_delete;

DELETE FROM payments WHERE user\_id = user\_id\_to\_delete;

-- Delete the user

DELETE FROM users WHERE user\_id = user\_id\_to\_delete;

DBMS\_OUTPUT.PUT\_LINE('User with ID ' || user\_id\_to\_delete || ' and related records have been deleted.');

END;

/

-- Calculate total payment amount for each user

SELECT 'User ID: ' || u.user\_id || ', Name: ' || u.name || ', Total Amount: ' || COALESCE(SUM(p.amount), 0) AS payment\_info

FROM users u

LEFT JOIN payments p ON u.user\_id = p.user\_id

GROUP BY u.user\_id, u.name;

-- Display the average price of all products

SELECT 'Average Price: ' || AVG(price) AS avg\_price FROM products;

-- Count the number of products in each category

SELECT 'Category: ' || c.name || ', Product Count: ' || COUNT(\*) AS category\_info

FROM categories c

JOIN products p ON c.category\_id = p.category\_id

GROUP BY c.name;

-- To display what each user has in their cart

SELECT u.name AS user\_name, p.name AS product\_name, ci.quantity

FROM users u

JOIN cart\_items ci ON u.user\_id = ci.user\_id

JOIN products p ON ci.product\_id = p.product\_id;

-- Find the most expensive product

SELECT 'Most Expensive Product - ID: ' || product\_id || ', Name: ' || name || ', Price: ' || price AS most\_expensive\_product

FROM products

ORDER BY price DESC

FETCH FIRST ROW ONLY;

-- Calculate the total quantity of a specific product in the cart

SELECT SUM(quantity) AS total\_quantity

FROM cart\_items ci

JOIN products p ON ci.product\_id = p.product\_id

WHERE p.name = 'iPhone';

-- Find the user with the highest total payment amount using PL/SQL block

DECLARE

highest\_payment\_user\_id INT;

user\_info VARCHAR2(200);

BEGIN

SELECT user\_id INTO highest\_payment\_user\_id

FROM (

SELECT user\_id, SUM(amount) AS total\_amount

FROM payments

GROUP BY user\_id

ORDER BY total\_amount DESC

)

WHERE ROWNUM = 1;

SELECT 'User with Highest Total Payment Amount - ID: ' || u.user\_id || ', Name: ' || u.name

INTO user\_info

FROM users u

WHERE u.user\_id = highest\_payment\_user\_id;

DBMS\_OUTPUT.PUT\_LINE(user\_info);

END;

/

DECLARE

average\_order\_value NUMBER;

-- Calculate the average order value using PL/SQL

BEGIN

SELECT AVG(amount)

INTO average\_order\_value

FROM payments;

DBMS\_OUTPUT.PUT\_LINE('Average Order Value: ' || average\_order\_value);

END;

/

**Explanation**

The DBMS is for an ecommerce website. The database schema implemented for our ecommerce website forms a fundamental part of the backend infrastructure. It consists of tables specifically designed to store and organize essential data related to the online store.

These tables include the "users" table, which stores user information such as names, email addresses, ages, and addresses. The "payments" table stores payment details like payment IDs, user IDs, payment amounts, and payment dates. The "categories" table categorizes the products, while the "products" table contains information about each product such as names, prices, and the corresponding category ID. Lastly, the "cart\_items" table contains the user ID, product ID, and quantities.

Sample inputs are then added to the tables using Insert function.

* Display all users:

This query displays all user records from the "users" table and combines the information into a single string. The resultincludes user ID, name, email, age, and address.

* Update user's email:

This query updates the email address of a specific user, identified by their user ID. Here, it sets the email address of user ID 1 to '[rajesh1.patel@gmail.com](mailto:rajesh1.patel@gmail.com)'.

* Deleting:

This PL/SQL block deletes a user with user\_id = 3. It first deletes related child records from the cart\_items and payments tables, and then deletes the user from the users table and a message is displayed after deletion

* Calculate the total payment amount for each user:

This query calculates the total payment amount for each user by joining the "users" and "payments" tables based on the primary key user ID. It adds the payment amounts and groups the results by user ID and name, providing a look of each user's total payment amount. The COALESCE function is used to handle cases where there are no payments for a user. If the sum of payment amounts (SUM(p.amount)) is NULL, it is replaced with 0.

* Display the average price of all products:

This query calculates the average price of all products in the "products" table and returns it.

* Count the number of products in each category:

This counts the number of products in each category by joining the "categories" and "products" tables based on the category ID. It groups the results by category name and returns the count of products in each category.

* Displaying User Cart Items:

This query displays the products in each user's cart by joining the users, cart\_items, and products tables. It shows the user's name, product name, and quantity.

* Finding the most expensive product:

The SELECT statement finds the most expensive product by ordering the products table based on price in descending order and fetching the first row.

* Calculating total quantity of a specific product:

This calculates the total quantity of a specific product 'iPhone' in the cart\_items table by joining the cart\_items and products tables.

* Finding the user with highest total payment amount:

The PL/SQL block retrieves the user\_id with the highest total payment amount by selecting the user\_id from the payments table and ordering the results by the total amount. The result is stored in the highest\_payment\_user\_id variable. Then, the user information is retrieved from the users table based on the highest\_payment\_user\_id, and it is displayed.

* Calculating average order value:

This PL/SQL block calculates the average order value by taking the average amount from the payments table and storing it in the average\_order\_value variable.

**Output**

  


  








