Database

Final Task

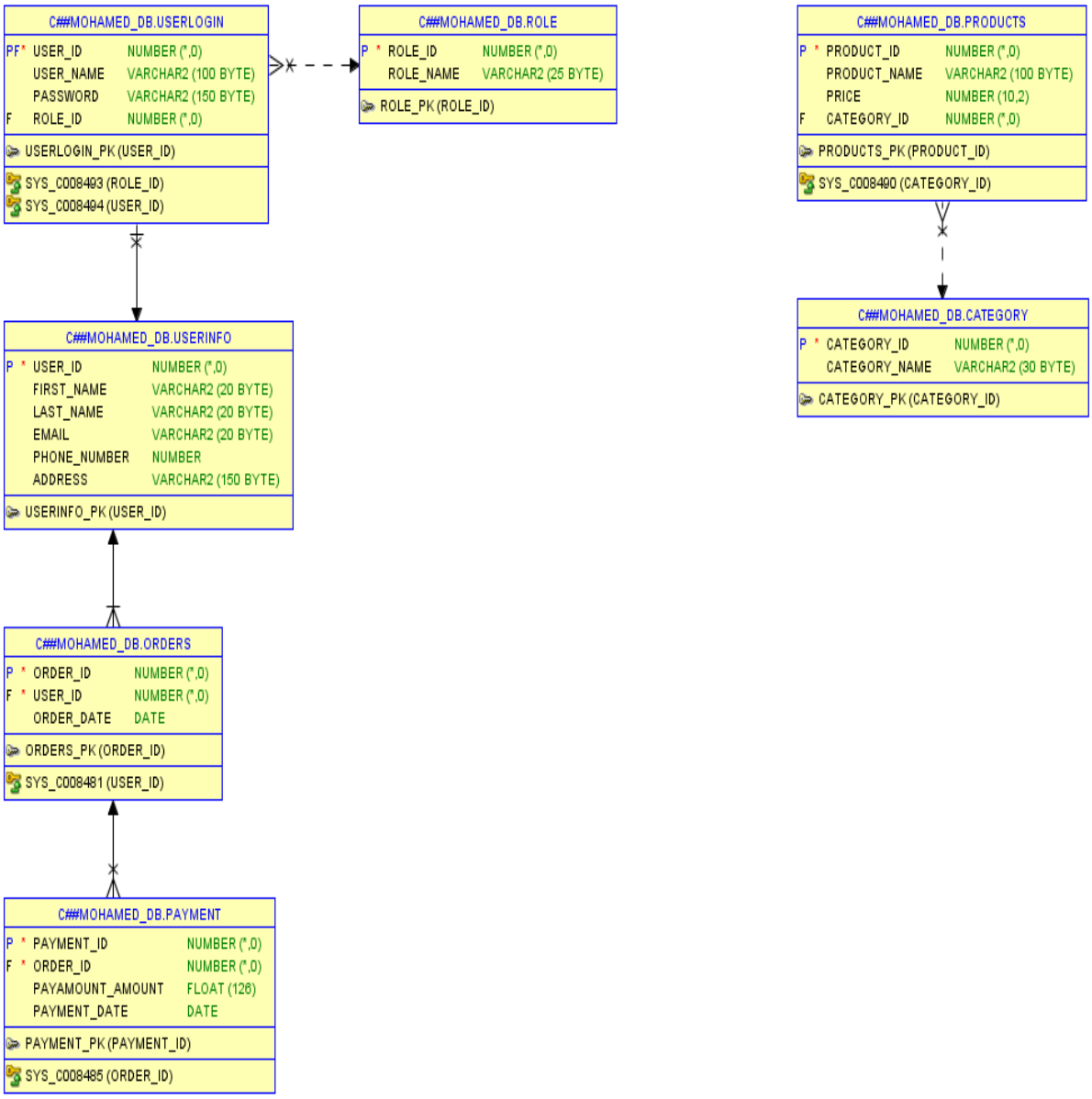
Ecommerce System

Student: Mohamed Fathi Ibrahim

Instructor: Raghad Al-Quran

1- Generate the class diagram

Class Diagram



2- Create a package for order, Product, and User table

- Order Package

```
-- Create a package header
CREATE OR REPLACE PACKAGE OrderPackage is
-- Define the procedure to display orders at intervals
 PROCEDURE GetOrdersByDateRange(start_date IN DATE, end_date IN
DATE);
 PROCEDURE GetOrdersByStartDate(start_date IN DATE);
END OrderPackage;
 -- Create a package body to implement the procedure
CREATE OR REPLACE PACKAGE BODY OrderPackage is
 --Create a stored procedure to display orders at intervals in
the database

  PROCEDURE GetOrdersByDateRange(start_date IN DATE, end_date IN
DATE) is
  C_order SYS_REFCURSOR;
  BEGIN
   OPEN C_order FOR SELECT * FROM orders WHERE ORDER_DATE
BETWEEN start_date AND end_date ;
        DBMS_SQL.RETURN_RESULT(C_order);

    END GetOrdersByDateRange;
-- Create a stored procedure to display orders by start date in
the database
 PROCEDURE GetOrdersByStartDate(start_date IN DATE) is
  C_order SYS_REFCURSOR;

  BEGIN
  OPEN C_order FOR SELECT * FROM orders  WHERE ORDER_DATE >=
start_date;
  DBMS_SQL.RETURN_RESULT(C_order);
  END GetOrdersByStartDate;


END OrderPackage;

BEGIN
  OrderPackage.GetOrdersByDateRange(TO_DATE('08-SEP-23', 'DD-
MON-YY'), TO_DATE('12-SEP-23', 'DD-MON-YY'));
END;
BEGIN
  OrderPackage.GetOrdersByStartDate(TO_DATE('08-SEP-23', 'DD-
MON-YY'));
END;
```

- Product Package

```
-- Create A package header

CREATE OR REPLACE PACKAGE ProductPackage IS
  -- Procedure to display all product by names and prices
  PROCEDURE DisplayAllProducts;

  -- Procedure to display products by product name
  PROCEDURE DisplayProductsByName(p_name IN products.product_name%type);

  -- Procedure to update a product
  PROCEDURE UpdateProduct(
    p_id IN products.product_id%type,
    new_product_name IN products.product_name%type,
    new_price IN products.price%type
  );

  -- Procedure to delete a product
  PROCEDURE DeleteProduct(p_id IN products.product_id%type);

  -- Procedure to display products by price
  PROCEDURE DisplayProductsByPrice(price IN products.price%type);

  -- Procedure to display the top three cheapest products
  PROCEDURE DisplayTopThreeCheapestProducts;
END ProductPackage;

-- Create package body
CREATE OR REPLACE PACKAGE BODY ProductPackage IS
  -- Procedure to display all product names and prices
  PROCEDURE DisplayAllProducts IS
    C_PRODUCT SYS_REFCURSOR;
    product_rec PRODUCTS%ROWTYPE;
  BEGIN
    OPEN C_PRODUCT FOR SELECT * FROM PRODUCTS;
    DBMS_SQL.RETURN_RESULT(C_PRODUCT);
  END DisplayAllProducts;

  -- Procedure to display products by product name
  PROCEDURE DisplayProductsByName(p_name IN products.product_name%type) IS
    C_PRODUCT SYS_REFCURSOR;
    product_rec PRODUCTS%ROWTYPE;
  BEGIN
    OPEN C_PRODUCT FOR SELECT * FROM PRODUCTS WHERE PRODUCT_NAME = p_name;
    DBMS_SQL.RETURN_RESULT(C_PRODUCT);
  END DisplayProductsByName;

  -- Procedure to update a product
  PROCEDURE UpdateProduct(
    p_id IN products.product_id%type,
    new_product_name IN products.product_name%type,
    new_price IN products.price%type) IS
  BEGIN
```

```
    UPDATE PRODUCTS
    SET PRODUCT_NAME = new_product_name, PRICE = new_price
    WHERE PRODUCT_ID = p_id;
    COMMIT;
  END UpdateProduct;

  -- Procedure to delete a product
  PROCEDURE DeleteProduct(p_id IN products.product_id%type) IS
  BEGIN
    DELETE FROM PRODUCTS WHERE PRODUCT_ID = p_id;
    COMMIT;
  END DeleteProduct;

  -- Procedure to display products by price
  PROCEDURE DisplayProductsByPrice(price IN products.price%type) IS
    C_PRODUCT SYS_REFCURSOR;
    product_rec PRODUCTS%ROWTYPE;
  BEGIN
    OPEN C_PRODUCT FOR SELECT * FROM PRODUCTS WHERE PRICE = price;
    DBMS_SQL.RETURN_RESULT(C_PRODUCT);
  END DisplayProductsByPrice;

  -- Procedure to display the top three cheapest products
  PROCEDURE DisplayTopThreeCheapestProducts IS
    C_PRODUCT SYS_REFCURSOR;
    product_rec PRODUCTS%ROWTYPE;
  BEGIN
    OPEN C_PRODUCT FOR
      SELECT PRODUCT_NAME, PRICE
      FROM PRODUCTS
      ORDER BY PRICE
      FETCH FIRST 3 ROWS ONLY;
    DBMS_SQL.RETURN_RESULT(C_PRODUCT);
  END DisplayTopThreeCheapestProducts;
END ProductPackage;


-- Display all product names and prices
BEGIN
  ProductPackage.DisplayAllProducts;
END;


-- Display products by product name
BEGIN
  ProductPackage.DisplayProductsByName('Product1');
END;


-- Update a product
BEGIN
  ProductPackage.UpdateProduct(1, 'New Product Name', 19.99);
END;


-- Delete a product
BEGIN
```

```
  ProductPackage.DeleteProduct(2);
END;


-- Display products by price
BEGIN
  ProductPackage.DisplayProductsByPrice(100.99);
END;


-- Display the top three cheapest products
BEGIN
  ProductPackage.DisplayTopThreeCheapestProducts;
END;
```

- User PACKAGE

```
-- Create A package header

CREATE OR REPLACE PACKAGE UserPackage AS
    PROCEDURE GetUserInformation(
        p_username IN USERLOGIN.USER_NAME%TYPE,
        p_password IN USERLOGIN.PASSWORD%TYPE,
        C_USER OUT SYS_REFCURSOR
    );

  PROCEDURE GetUserRoles(
        p_username IN USERLOGIN.USER_NAME%TYPE,
        C_USER_R OUT SYS_REFCURSOR
    );
     PROCEDURE IncrementSalaryForDeliveryRole(p_username IN
USERLOGIN.USER_NAME%TYPE);
END UserPackage;

-- Create A package BODY


CREATE OR REPLACE PACKAGE BODY UserPackage IS
-- PROCEDURE FOR GET USER INFORAMTION


    PROCEDURE GetUserInformation(
        p_username IN USERLOGIN.USER_NAME%TYPE,
        p_password IN USERLOGIN.PASSWORD%TYPE,
        C_USER OUT SYS_REFCURSOR
    )
    IS
    BEGIN
      OPEN C_USER FOR
        SELECT UL.USER_ID, UI.First_Name, UI.Last_Name, UI.Email,
UI.Phone_Number, UI.Address
        FROM USERLOGIN UL
        JOIN USERINFO UI ON UL.USER_ID = UI.USER_ID
        WHERE UL.USER_NAME = p_username AND UL.PASSWORD = p_password;
         DBMS_SQL.RETURN_RESULT(C_USER);
    EXCEPTION
        WHEN NO_DATA_FOUND THEN

            NULL;
        WHEN OTHERS THEN

            NULL;
    END GetUserInformation;

-- PROCEDURE FOR GET USER ROLES


    PROCEDURE GetUserRoles(
        p_username IN USERLOGIN.USER_NAME%TYPE,
        C_USER_R OUT SYS_REFCURSOR
```

```
    )
    IS
    BEGIN
        OPEN C_USER_R FOR
        SELECT R.ROLE_NAME
        FROM USERLOGIN UL
        JOIN ROLE R ON UL.ROLE_ID = R.ROLE_ID
        WHERE UL.USER_NAME = p_username;

    END GetUserRoles;

-- PROCEDURE INCREMENT SALARY



    PROCEDURE IncrementSalaryForDeliveryRole(p_username IN
USERLOGIN.USER_NAME%TYPE) IS

    BEGIN
UPDATE USERINFO
        SET Salary = Salary + 50
        WHERE USER_ID IN (
            SELECT UL.USER_ID
            FROM USERLOGIN UL
            JOIN ROLE R ON UL.ROLE_ID = R.ROLE_ID
            WHERE UL.USER_ID = (
                SELECT USER_ID
                FROM USERLOGIN
                WHERE USER_NAME = p_username
            )
            AND R.ROLE_NAME = 'DELIVERY'
        );

        COMMIT;
    END IncrementSalaryForDeliveryRole;
END UserPackage;


-- DISPLAY USER AFTER INCREMENET



BEGIN
    UserPackage.IncrementSalaryForDeliveryRole('NOUR_4758');
END;



-- DISPLAY  ROLES


DECLARE
    C_USER_R SYS_REFCURSOR;
    p_username USERLOGIN.USER_NAME%TYPE := 'MOHAMED_2022';
    role_name ROLE.ROLE_NAME%TYPE; BEGIN
    UserPackage.GetUserRoles(p_username, C_USER_R);
```

```
    LOOP
        FETCH C_USER_R INTO role_name;
        EXIT WHEN C_USER_R%NOTFOUND;

        DBMS_OUTPUT.PUT_LINE('User ' || p_username || ' has role: ' ||
role_name);
    END LOOP;


    CLOSE C_USER_R;
END;

-- DISPLAY USERINFO


DECLARE
    C_USER SYS_REFCURSOR;
BEGIN
    UserPackage.GetUserInformation('MOHAMED_2022', 12345678, C_USER);

END;
```

A. Create a stored procedure to display the user information by the username and password.

```
PROCEDURE GetUserInformation(
    p_username IN USERLOGIN.USER_NAME%TYPE,
    p_password IN USERLOGIN.PASSWORD%TYPE,
    C_USER OUT SYS_REFCURSOR
  )
  IS
  BEGIN
   OPEN C_USER FOR
    SELECT UL.USER_ID, UI.First_Name, UI.Last_Name, UI.Email, UI.Phone_Number,
UI.Address
    FROM USERLOGIN UL
    JOIN USERINFO UI ON UL.USER_ID = UI.USER_ID
    WHERE UL.USER_NAME = p_username AND UL.PASSWORD = p_password;
     DBMS_SQL.RETURN_RESULT(C_USER);
  EXCEPTION
    WHEN NO_DATA_FOUND THEN

      NULL;
    WHEN OTHERS THEN

      NULL;
  END GetUserInformation;
```

USER_ID	FIRST_NAME	LAST_NAME	EMAIL	PHONE_NUMBER	ADDRESS	SALARY
3	NOUR	MOSTAFA	NOUR@GAMIL.COM	2125656556	ASWAN	250

B. Create a stored procedure to display orders at intervals in the database.

- PROCEDURE GetOrdersByDateRange(start_date IN DATE, end_date IN DATE) is
  C_order SYS_REFCURSOR;
  BEGIN
  OPEN C_order FOR SELECT * FROM orders WHERE ORDER_DATE BETWEEN start_date AND end_date ;
  DBMS_SQL.RETURN_RESULT(C_order);

  END GetOrdersByDateRange;

ORDER_ID	USER_ID	ORDER_DAT
1	2	08-SEP-23
3	4	10-SEP-23

C. Create a stored procedure to display orders by start date in the database.

- PROCEDURE GetOrdersByStartDate(start_date IN DATE) is
  C_order SYS_REFCURSOR;

  BEGIN
  OPEN C_order FOR SELECT * FROM orders WHERE ORDER_DATE >= start_date;
  DBMS_SQL.RETURN_RESULT(C_order);
  END GetOrdersByStartDate;

ORDER_ID	USER_ID	ORDER_DAT
1	2	08-SEP-23
2	3	12-SEP-23
3	4	10-SEP-23
4	2	20-SEP-23

D. Create a stored procedure to display all product names and prices in the database.

- PROCEDURE DisplayAllProducts IS
  C_PRODUCT SYS_REFCURSOR;
  product_rec PRODUCTS%ROWTYPE;
  BEGIN
  OPEN C_PRODUCT FOR SELECT * FROM PRODUCTS;
  DBMS_SQL.RETURN_RESULT(C_PRODUCT);
  END DisplayAllProducts;

```
PRODUCT_ID PRODUCT_NAME
---------- --------------------
         1 Smartphone
         2 T-SHIRT
         3 OFFICE-CHAIR
         4 MEAT
```

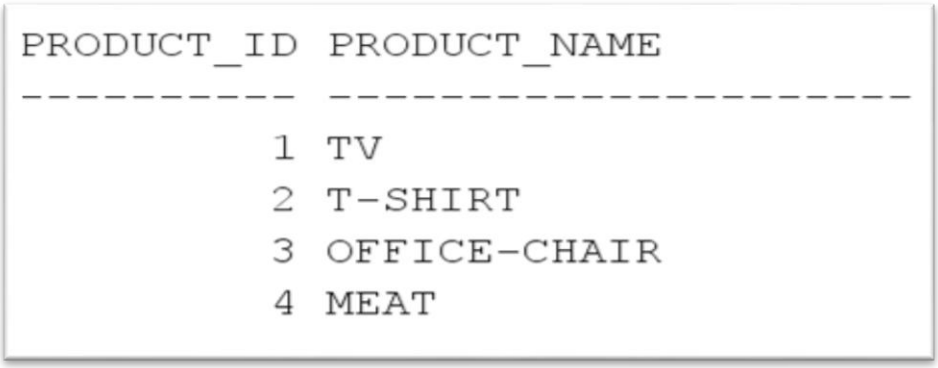
E. Create a stored procedure to display products by product name in the database.

- PROCEDURE DisplayProductsByName(p_name IN products.product_name%type) IS
  C_PRODUCT SYS_REFCURSOR;
  product_rec PRODUCTS%ROWTYPE;
  BEGIN
  OPEN C_PRODUCT FOR SELECT * FROM PRODUCTS WHERE PRODUCT_NAME = p_name;
  DBMS_SQL.RETURN_RESULT(C_PRODUCT);
  END DisplayProductsByName;

```
PRODUCT_ID PRODUCT_NAME
---------- --------------------
         1 Smartphone
```

F. Create a stored procedure to update a product.

- PROCEDURE UpdateProduct(
  p_id IN products.product_id%type,
  new_product_name IN products.product_name%type,
  new_price IN products.price%type) IS
  BEGIN
  UPDATE PRODUCTS
  SET PRODUCT_NAME = new_product_name, PRICE = new_price
  WHERE PRODUCT_ID = p_id;
  COMMIT;
  END UpdateProduct;



PRODUCT_ID	PRODUCT_NAME
1	TV
2	T-SHIRT
3	OFFICE-CHAIR
4	MEAT

G. Create a stored procedure to delete product from the database.

- PROCEDURE DeleteProduct(p_id IN products.product_id%type) IS
  BEGIN
  DELETE FROM PRODUCTS WHERE PRODUCT_ID = p_id;
  COMMIT;
  END DeleteProduct;

PRODUCT_ID	PRODUCT_NAME
2	T-SHIRT
3	OFFICE-CHAIR
4	MEAT

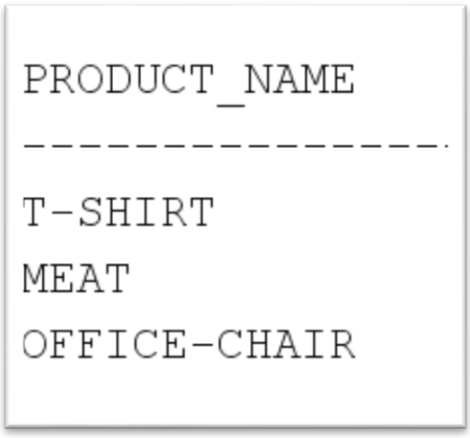
H. Create **a** stored procedure to display products by price in the database.

- PROCEDURE DisplayProductsByPrice(price IN products.price%type) IS
  C_PRODUCT SYS_REFCURSOR;
  product_rec PRODUCTS%ROWTYPE;
  BEGIN
  OPEN C_PRODUCT FOR SELECT * FROM PRODUCTS WHERE PRICE = price;
  DBMS_SQL.RETURN_RESULT(C_PRODUCT);
  END DisplayProductsByPrice;

PRODUCT_ID	PRODUCT_NAME	PRICE	CATEGORY_ID
2	T-SHIRT	20	2
3	OFFICE-CHAIR	200	3
4	MEAT	100	4

I. Create a stored procedure to display the top three of the cheapest products in terms of price in the database.

- PROCEDURE DisplayTopThreeCheapestProducts IS
  C_PRODUCT SYS_REFCURSOR;
  product_rec PRODUCTS%ROWTYPE;
  BEGIN
  OPEN C_PRODUCT FOR
  SELECT PRODUCT_NAME, PRICE
  FROM PRODUCTS
  ORDER BY PRICE
  FETCH FIRST 3 ROWS ONLY;
  DBMS_SQL.RETURN_RESULT(C_PRODUCT);
  END DisplayTopThreeCheapestProducts;



J. Create a stored procedure to display users with their roles in the database.

- PROCEDURE GetUserRoles(
    p_username IN USERLOGIN.USER_NAME%TYPE,
    C_USER_R OUT SYS_REFCURSOR
)
  IS
  BEGIN
    OPEN C_USER_R FOR
    SELECT R.ROLE_NAME
    FROM USERLOGIN UL
    JOIN ROLE R ON UL.ROLE_ID = R.ROLE_ID -- Corrected the join condition
    WHERE UL.USER_NAME = p_username;

  END GetUserRoles;

```
User MOHAMED_2022 has role: ADMIN
```

K. Create a stored procedure to Increment the user's salary with 50 JD for the users who have a delivery role.

- PROCEDURE IncrementSalaryForDeliveryRole(p_username IN USERLOGIN.USER_NAME%TYPE) IS

```
BEGIN
  -- Update the salary for users with a delivery role
  UPDATE USERINFO
  SET Salary = Salary + 50
  WHERE USER_ID IN (
    SELECT UL.USER_ID
    FROM USERLOGIN UL
    JOIN ROLE R ON UL.ROLE_ID = R.ROLE_ID
   WHERE UL.USER_ID = (
      SELECT USER_ID
      FROM USERLOGIN
      WHERE USER_NAME = p_username
    )
    AND R.ROLE_NAME = 'DELIVERY'
  );

  COMMIT;
END IncrementSalaryForDeliveryRole;
```

USER_ID	FIRST_NAME	LAST_NAME	EMAIL	PHONE_NUMBER	ADDRESS	SALARY
3	NOUR	MOSTAFA	NOUR@GAMIL.COM	2125656556	ASWAN	300

Thank you for your efforts with us ❤