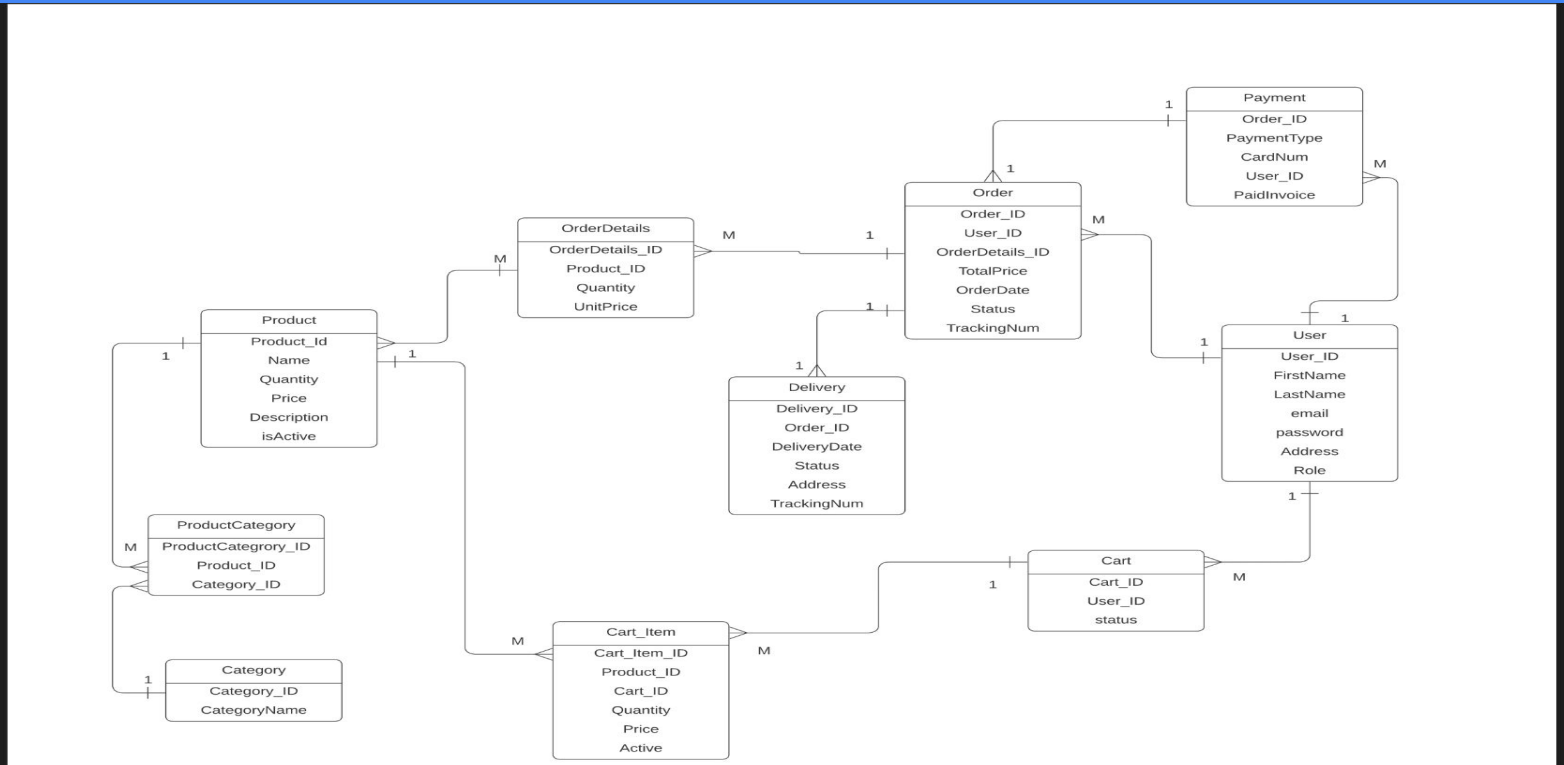
PL/SQl MIDTERM

OUR TEAM:Ibrayev AIkyn, Shyngys Lapidenov, Berik Yerman, Abzal Slamkozha, Ibrakhim Namdar

Introduction

Our database consists of such tables as: Product, Category, Product Category, User, Shopping Cart, Product in the cart, Order, Order Item, Delivery, Payment. Each table has its own purpose.

ER DIAGRAM OF ONLINE SHOP



ABOUT NORMALIZATION

EACH TABLE IN 3NF FORM

It’s just intro to normalization, all explanation in report

The Product table has attributes: Product_Id, Name, quantity, price, description, active, the table is in 1 normal form since it does not contain a composite or multi-valued attribute.
It is also in 2 normal form since the table does not contain a partial dependency, that is, all attributes depend only on the primary key Product_id and all attributes of the relationship are the main attribute, then the relationship in 3nf

FUNCTIONS

Function which counts the number of records

```
CREATE OR REPLACE FUNCTION
count_records(nameOfTable IN
VARCHAR2)
 RETURN NUMBER IS
  countOfRecord NUMBER := 0;
BEGIN
  EXECUTE IMMEDIATE 'SELECT COUNT(*)
FROM ' || nameOfTable INTO
countOfRecord;
  RETURN countOfRecord;
END;
```

In this code we create a function that returns the number of rows in the table, first we create the function itself, we specify that we will return the number, and initialize the number itself, and in the main part of the code we write a regular sql query that returns the number of rows in the table, but instead of the name of the table we take the function parameter nameOfTable, and we write the answers in the variable countOfRecords, which we opened in the beginning of the code, and write EXECUTE IMMEDIATE so the query was dynamic. (we borrowed this from the Internet), and at the end we return the answer.

PROCEDURES(It is first procedure)

- **Procedure which uses SQL%ROWCOUNT to determine the number of rows affected**

```
CREATE OR REPLACE PROCEDURE
update_iphone_quantity AS
BEGIN
  UPDATE Product
  SET Quantity = Quantity + 30
  WHERE Name LIKE 'Iphone%';
  DBMS_OUTPUT.PUT_LINE(SQL%ROWCOUNT || '
rows updated successfully.');
END;
```

Procedure that updates the Quantity field in the Product table by adding 30 to it for all products whose name starts with "Iphone". The procedure then outputs the number of rows that were successfully updated using the DBMS_OUTPUT.PUT_LINE statement. The purpose of this code is to automate the process of updating the inventory of iPhones in the Product table.

PROCEDURE 2

- **Procedure which does group by information**

```
CREATE OR REPLACE PROCEDURE User_role
AS
  cursor user_group is
    select ROLE, count(*) as user_count
    from Users
    group by ROLE;
BEGIN
  for role_group in user_group
  loop
    DBMS_OUTPUT.PUT_LINE(role_group.Role  || '|
:' ||  role_group.user_count);
```

This is a PL/SQL code block that defines a stored procedure named "User_role". The procedure uses a cursor named "user_group" to fetch data from the "Users" table, grouping the users by their role and counting the number of users for each role.
The "for loop" iterates through the cursor and for each role, it prints out the role name and the number of users with that role using the DBMS_OUTPUT.PUT_LINE statement.
Overall, the procedure generates a report of the number of users for each role in the Users table.

EXCEPTIONS and TRIGGERS

```
CREATE OR REPLACE TRIGGER check_password_length
BEFORE INSERT ON Users
FOR EACH ROW
DECLARE
  too_short exception;
BEGIN
  IF LENGTH(:new.password) < 5 THEN
    RAISE too_short;
  END IF;
EXCEPTION
  WHEN too_short THEN
    raise_application_error(-20001, 'Password
length should be at least 5 characters');
END;
```

- Add user-defined exception which disallows to enter title of item (e.g. book) to be less than 5 characters

This trigger and exception which the length of the password field before a new row is inserted into the Users table. If the length of the password is less than 5 characters, the trigger raises a custom exception called "too_short". Then, an exception handler catches the "too_short" exception and raises an application error with the message "Password length should be at least 5 characters". This trigger is designed to enforce a password length requirement for new users to ensure that their passwords are secure.

Triggers

```
CREATE OR REPLACE TRIGGER
row_count_trigger
BEFORE INSERT ON USERS
DECLARE
    row_count INT;
BEGIN
    SELECT COUNT(*) INTO row_count FROM
USERS;
    DBMS_OUTPUT.PUT_LINE('Current row
count: ' || row_count);
END;
```

- Create a trigger before insert on any entity which will show the current number of rows in the table

This trigger was created with an aim to show the number of rows that existed before the insertion of any new data into the table

Conclusion

In this project we have learned how to work in pl/sql with triggers, procedures, functions e.t.c. and how to change to normalization forms.