Report

DBMS EndTerm

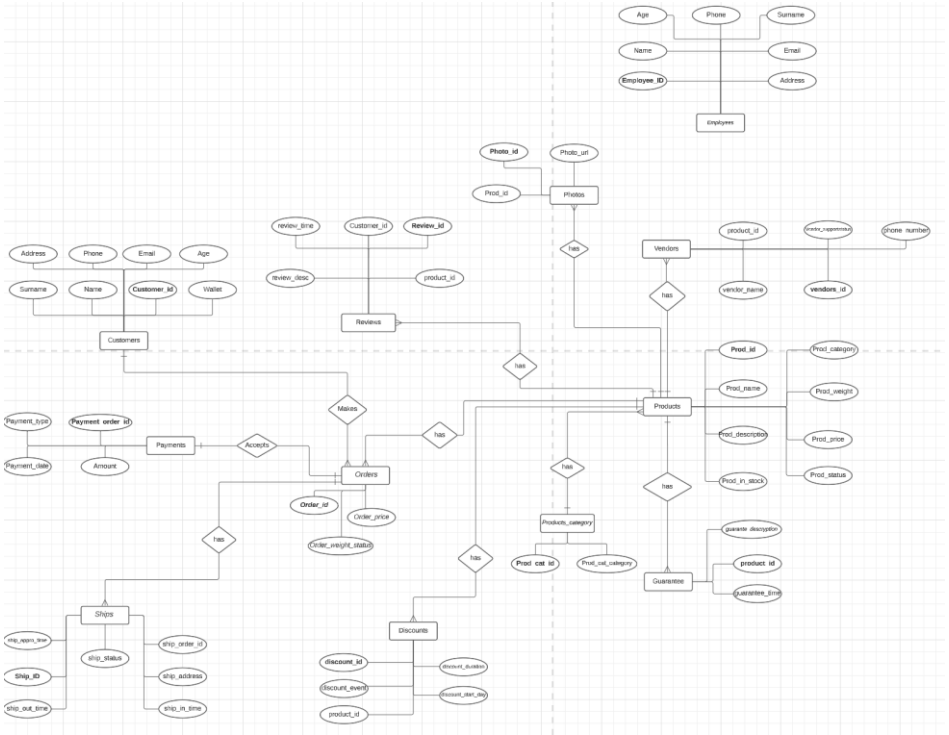
Introduction:

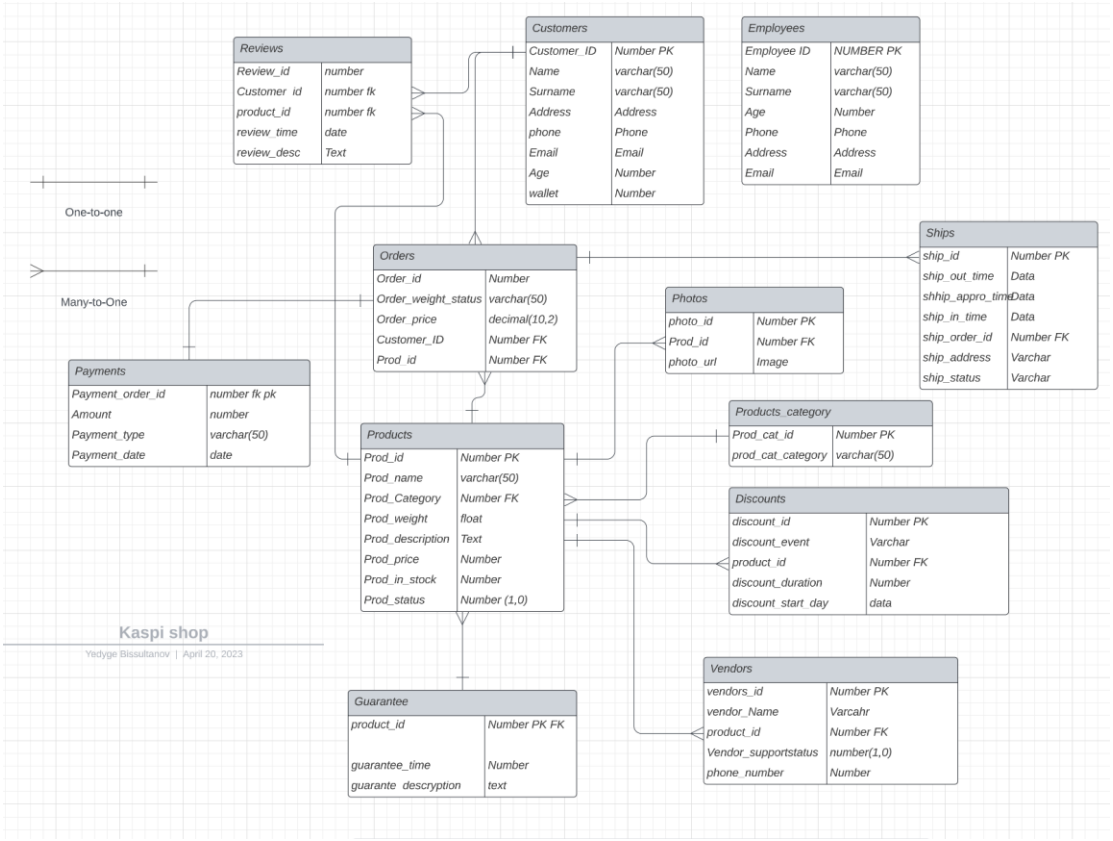
The system described in this report is a e-commerce platform for selling and purchasing products online. The
platform is designed to provide a seamless shopping experience for customers and a user-friendly interface for vendors to manage
their products and sales. The system includes product categorization, search functionality, customer
and vendor account management, payment processing, order management, and shipping and delivery options.

The system is built using a database management system to store and organize information about products, customers, vendors,
orders, discounts, reviews, photos, guarantees, and shipments. The data is structured according to the principles of relational
database management systems, normalized and satisfies the requirements of the third normal form.

E-commerce platform described in this report is a comprehensive system that provides a seamless and efficient shopping
experience for customers and a powerful tool for vendors to manage their products and sales.[Link to GitHub]

ER Diagram:





1.Let's analyze the MP2_Products_category table to determine whether it satisfies the third normal form (3NF).
In the MP2_Products_category table, the primary key is Prod_cat_id, and the only non-prime attribute is prod_cat_category. As
can be seen, prod_cat_category is dependent solely on the primary key, which means that the table satisfies the 3NF.
This ensures that data redundancy is minimized, and updates, deletions, and insertions are carried out without anomalies. It
also facilitates the querying of data and the development of efficient database management systems.

2.The MP2_Products table represents the products available for purchase on the e-commerce platform. It has the following
attributes:
The Prod_id attribute is the primary key of the table, ensuring that each product has a unique identifier. The Prod_Category
attribute is a foreign key that references the MP2_Products_category table, ensuring that each product belongs to a category
that exists in the category table.
The Prod_name, Prod_weight, Prod_description, Prod_price, Prod_in_stock, and Prod_status attributes are all atomic and are not
multi-valued. Therefore, the table satisfies the first normal form.
The table does not contain any repeating groups, as each product is represented by a single row. Therefore, the table satisfies
the second normal form.

The Prod_category attribute is functionally dependent on the Prod_id attribute and is not dependent on any other non-key
attributes. Therefore, the table satisfies the third normal form.

3.The MP2_Employees table stores information about the employees of the company. It contains the following attributes:
Employee_ID, Name, Surname, Age, Phone, Address, and Email.
The Employee_ID attribute is the primary key of the table, and it uniquely identifies each employee. The Name and Surname
attributes store the first and last name of each employee, respectively. The Age attribute stores the age of each employee. The
Phone attribute stores the phone number of each employee, while the Address attribute stores the physical address of each
employee. Finally, the Email attribute stores the email address of each employee.
This table satisfies the first normal form because it has a primary key (Employee_ID), and all attributes are atomic, meaning
they cannot be further divided. It also satisfies the second normal form because all non-key attributes (Name, Surname, Age,
Phone, Address, and Email) depend solely on the primary key (Employee_ID).
To demonstrate that the table satisfies the third normal form, we need to ensure that there are no transitive dependencies. A
transitive dependency occurs when a non-key attribute depends on another non-key attribute. In this table, we can see that there
are no such dependencies. All non-key attributes depend solely on the primary key (Employee_ID). Therefore, the MP2_Employees
table satisfies the third normal form.

4.The MP2_Customers table contains information about customers, such as their name, address, phone number, email address, age,
and wallet. This table satisfies the 3NF because it meets the requirements of 1NF and 2NF, and there is no transitive dependency
between non-key attributes.

For 1NF table has a primary key (Cust_ID) that uniquely identifies each customer. Each attribute contains only atomic values,
and there are no repeating groups.
For 2NF All non-key attributes (Cust_Name, Cust_Surname, Cust_Address, Cust_Phone, Cust_Email, Cust_Age, and Cust_Wallet) are
functionally dependent on the primary key (Cust_ID).
For 3NF There no transitive dependencies between non-key attributes. For example, Cust_Address is not dependent on Cust_Phone or
Cust_Email.
The MP2_Customers table satisfies the 3NF, which means it is well-designed and can avoid data inconsistencies and anomalies.

5.The Discount table has the following attributes:
Discounts_id: a unique identifier for each discount
Discount_event: the name or description of the discount event
Product_id: the ID of the product that the discount applies to
Discount_start_day: the date when the discount event starts
Discount_duration: the duration of the discount event

This table satisfies the first normal form (1NF) because it has a primary key and there are no repeating groups or arrays. Each
attribute has a single value and each row is unique.
It satisfies the second normal form (2NF) because all non-key attributes are functionally dependent on the primary key, which is
the Discounts_id. The Discount_event, Product_id, Discount_start_day, and Discount_duration are all dependent on the primary key.
It satisfies the third normal form (3NF) because there are no transitive dependencies. All non-key attributes are only dependent
on the primary key, and there are no dependencies between non-key attributes. Therefore, the Discount table is in third normal
form.

6.The MP2_Vendors table represents the vendors that supply the products in the MP2_Products table. It contains the following
columns:

Vendors_id: A unique identifier for each vendor.
Vendor_Name: The name of the vendor.
Product_id: A foreign key that references the Prod_id column in the MP2_Products table. It specifies which product the vendor
supplies.
Vendor_SupportStatus: A flag that indicates whether the vendor provides support for the product.
Phone_Number: The phone number of the vendor.
This table satisfies the first normal form because each column contains atomic values, and there are no repeating groups.
It satisfies the second normal form because it has a single-column primary key and all other columns are dependent on the
primary key. Specifically, the Product_id column depends on the Vendors_id column, and the remaining columns depend on both the
Vendors_id and Product_id columns.
It satisfies the third normal form because there are no transitive dependencies between non-key columns. The
Vendor_SupportStatus and Phone_Number columns are dependent only on the Vendors_id column and not on the Product_id column.
Thus, the table is in third normal form.

7.The MP2_Review table is used to store product reviews submitted by customers. It has the following attributes:

Review_id: a unique identifier for each review (primary key)
Cust_id: a foreign key referencing the customer who submitted the review
Prod_id: a foreign key referencing the product being reviewed
Review_time: the date and time the review was submitted
Review_desc: the description of the review
The table satisfies the first normal form because each attribute in each row of the table is atomic and cannot be further
decomposed.
The table satisfies the second normal form because it has a composite primary key (Review_id, Prod_id) that uniquely identifies

each row in the table. Additionally, all non-key attributes (Cust_id, Review_time, Review_desc) are fully dependent on the
entire primary key, meaning that no partial dependency exists.
The table satisfies the third normal form because there are no transitive dependencies between non-key attributes. All non-key
attributes are dependent only on the primary key, and no other attributes in the table. Therefore, this table is also in third
normal form.
The MP2_Review table is well-structured and satisfies the requirements of the first, second, and third normal forms.

8.The MP2_Orders table satisfies the third normal form (3NF) as it does not have any transitive dependencies.
The table has the following columns:
order_id (primary key)
order_weight_status
order_price
customer_id (foreign key referencing the customer_id column in MP2_Customers table)
product_id (foreign key referencing the Prod_id column in MP2_Products table)
There are no repeating groups in the table and each column is dependent on the primary key, order_id. There are no columns that
are dependent on non-primary key columns. Therefore, the table is in the first normal form (1NF).
The table is also in the second normal form (2NF) as there are no partial dependencies. All non-key columns depend on the
primary key, order_id.
Finally, the table satisfies the third normal form (3NF) as there are no transitive dependencies. All non-key columns depend
only on the primary key and not on any other non-key columns.

9.The MP2_Photos table in the database has three columns: Photo_id, Product_id, and Photo_url. This table stores the URLs of
product photos and associates them with their corresponding product IDs.
To satisfy the first normal form (1NF), the MP2_Photos table has a primary key, which is the Photo_id column. Each row in the
table has a unique identifier, and there are no repeating groups.
To satisfy the second normal form (2NF), the MP2_Photos table has a foreign key constraint on the Product_id column that
references the Prod_id column in the MP2_Products table. This ensures that each product in the MP2_Products table has a
corresponding photo in the MP2_Photos table.
To satisfy the third normal form (3NF), there are no transitive dependencies in the MP2_Photos table. All columns are
functionally dependent on the primary key, Photo_id, and there are no other dependencies between columns. Therefore, the
MP2_Photos table is in 3NF.

10.The Guarantee table has a primary key of Prod_id, which is a foreign key reference to the Prod_id column in the Products
table. This ensures that each guarantee is tied to a specific product.
The table has two other columns: Guarantee_time and Guarantee_description. Guarantee_time is a timestamp indicating when the

guarantee starts, and Guarantee_description is a description of the guarantee.
There are no repeating groups or multiple values in this table, as each guarantee is tied to a single product and has its own
unique ID. Therefore, the Guarantee table satisfies the first normal form.
There are no partial dependencies in this table, as both the Guarantee_time and Guarantee_description columns depend on the
entire primary key (Prod_id). Therefore, the Guarantee table satisfies the second normal form.
Finally, there are no transitive dependencies in this table, as there are only three columns: the primary key, the guarantee
time, and the guarantee description. Therefore, the Guarantee table satisfies the third normal form.

11.The Ships table contains information about the shipping of orders. The table has a primary key column, "ship_id", which is a
unique identifier for each shipping record. The table also has a foreign key column, "ship_order_id", which references the
order_id column in the Orders table, linking each shipping record to a specific order.
The table has columns for the "ship_out_time", "ship_appro_time", and "ship_in_time" which represent the time when the order was
shipped out, the time when the shipping was approved, and the time when the order was received by the customer respectively.
There is also a "ship_address" column which represents the address to which the order was shipped.

Finally, the table has a "ship_status" column which indicates the current status of the shipment. This column can have values
like "shipped", "in transit", "delivered", etc.
The Ships table satisfies the first normal form as each column has a unique name, and each row has a unique identifier in the
form of the "ship_id" column.
The table also satisfies the second normal form as it has a primary key, "ship_id", which uniquely identifies each row in the
table. Additionally, all other columns are dependent only on the "ship_id" column, and there are no partial dependencies on any
subset of columns.
Finally, the table satisfies the third normal form as there are no transitive dependencies between the columns. All columns
depend only on the "ship_id" column, which is the primary key of the table.

ERD
1. Customers and Orders have entity relation "Places"

2. Ships and Orders have entity relation "has"
The relationship between Ships and Orders is a "has" relationship, where each order can have one or more associated shipments.
This is because an order may be fulfilled in multiple shipments, especially if the ordered items are not immediately available
or are coming from different locations.
σ Orders.OrderID = Ships.OrderID (Orders ⋈Ships)
3. Reviews table and Products table have entity relation "has".
The relationship between the Reviews and Products tables can be expressed as "has," meaning that a product can have many
reviews, but a review can only belong to one product. This is a one-to-many relationship.
4. Discounts table and Products table have entity relation "has".

the relation between Discounts table and Products table is that a product may have multiple discounts, and a discount may apply
to multiple products. Therefore, the relation between them is "has".
5. Orders table and Products table have entity relation "has".
The relationship between the Orders table and the Products table can be expressed as "has," indicating that an order has one or
more products associated with it. This relationship can be represented in relational algebra using a join operation.
6. Guarantee table and Products table have entity relation "has".
The entity relation between Guarantee table and Products table is "has" because each product can have one or more guarantees
associated with it, and each guarantee can be associated with only one product. This is a one-to-many relationship between the
Products table and Guarantee table.
7. Products_Category table and Products table have entity relation "has".
Products_Category table has the attribute category_id which is a foreign key referencing the id attribute in the Categories
table, and Products table has the attribute category_id which is a foreign key referencing the id attribute in the
Products_Category table.
8. Photos table and Products table have entity relation "has".
where product_id is a foreign key that references the primary key of Products table. This relation indicates that a product can
have multiple photos associated with it.
9.Vendors table and Products table have entity relation "has".
vendor can have many products, and a product can belong to only one vendor. Therefore, we can represent this relationship using
the foreign key vendor_id in the Products table that references the primary key id in the Vendors table.

- Procedure which does group by information

```
6   CREATE OR REPLACE PROCEDURE group_by_info IS
7     cursor Products_info is
8       SELECT P.prod_category, PC.Prod_cat_category, COUNT(*) AS Number_of_Products
9       FROM MP2_Products P, MP2_Products_category PC
10      WHERE P.Prod_category = PC.Prod_cat_id
11      GROUP BY P.prod_category, PC.Prod_cat_category;
12  BEGIN
13    dbms_output.put_line('Product Category | Number of Products');
14    for Product_cat in Products_info loop
15      dbms_output.put_line(Product_cat.Prod_cat_category || ' | ' || Product_cat.Number_of_Products);
16    end loop;
17  END;
18
```

Results | Explain | Describe | Saved SQL | History

```
Product Category | Number of Products
Furniture | 1
Arts and crafts | 1
Watches | 1
Garden | 1
Books | 1
Outdoor | 1
Industrial | 1
Clothing | 1
Toys | 1
```

- Function which counts the number of records

```
19  -- Function which counts the number of records
20  CREATE OR REPLACE FUNCTION count_records RETURN NUMBER AS
21    num_records NUMBER;
22  BEGIN
23    SELECT COUNT(*) INTO num_records FROM MP2_Customers;
24    RETURN num_records;
25  END;
26
```

Results | Explain | Describe | Saved SQL | History

```
50

Statement processed.

0.07 seconds
```

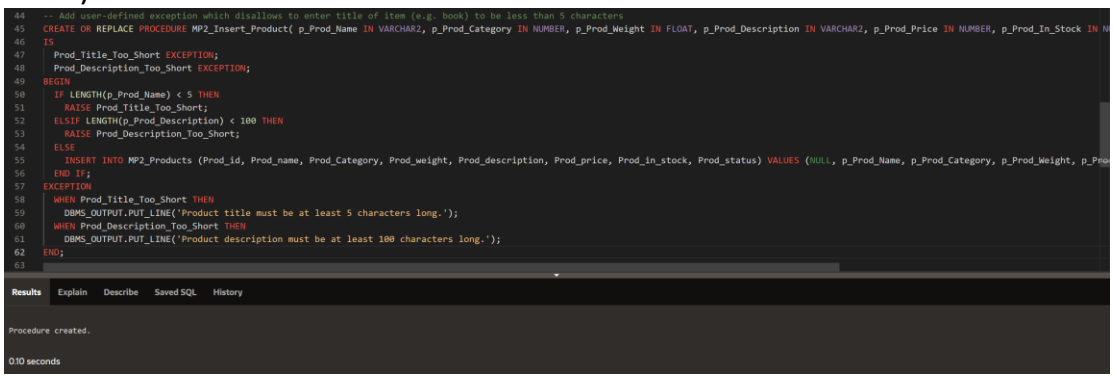
- Procedure which uses SQL%ROWCOUNT to determine the number of rows affected

```
32  -- Procedure which uses SQL%ROWCOUNT to determine the number of rows affected
33  CREATE OR REPLACE PROCEDURE update_employee_salary(emp_id IN NUMBER, emp_new_salary IN NUMBER) AS
34  BEGIN
35    UPDATE MP2_EMPLOYEES SET salary = emp_new_salary WHERE Employee_id = emp_id;
36    DBMS_OUTPUT.PUT_LINE('Number of rows affected: '  SQL%ROWCOUNT);
37  END;
38
```

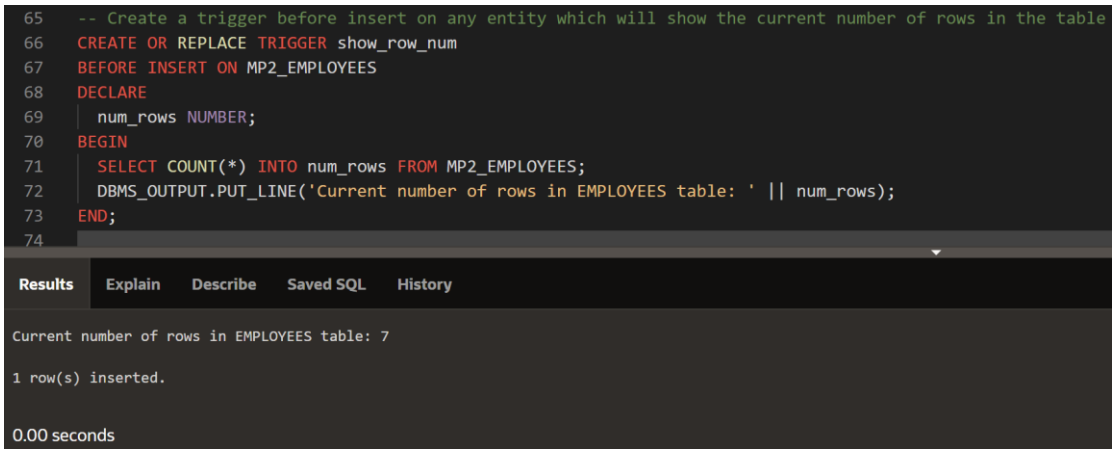
Results | Explain | Describe | Saved SQL | History

```
Number of rows affected: 1

Statement processed.

0.07 seconds
```

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



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



```
65  -- Create a trigger before insert on any entity which will show the current number of rows in the table
66  CREATE OR REPLACE TRIGGER show_row_num
67  BEFORE INSERT ON MP2_EMPLOYEES
68  DECLARE
69    num_rows NUMBER;
70  BEGIN
71    SELECT COUNT(*) INTO num_rows FROM MP2_EMPLOYEES;
72    DBMS_OUTPUT.PUT_LINE('Current number of rows in EMPLOYEES table: ' || num_rows);
73  END;
74
```

Results | Explain | Describe | Saved SQL | History

```
Current number of rows in EMPLOYEES table: 7

1 row(s) inserted.

0.00 seconds
```

Made By:
Bissultanov Yedyge
Abylay Akhmetov
Ramazan Khakimzhan
Chsheriyazdanov Rassul
Amirkhan Mamytbekov