Report
Online Toy Store

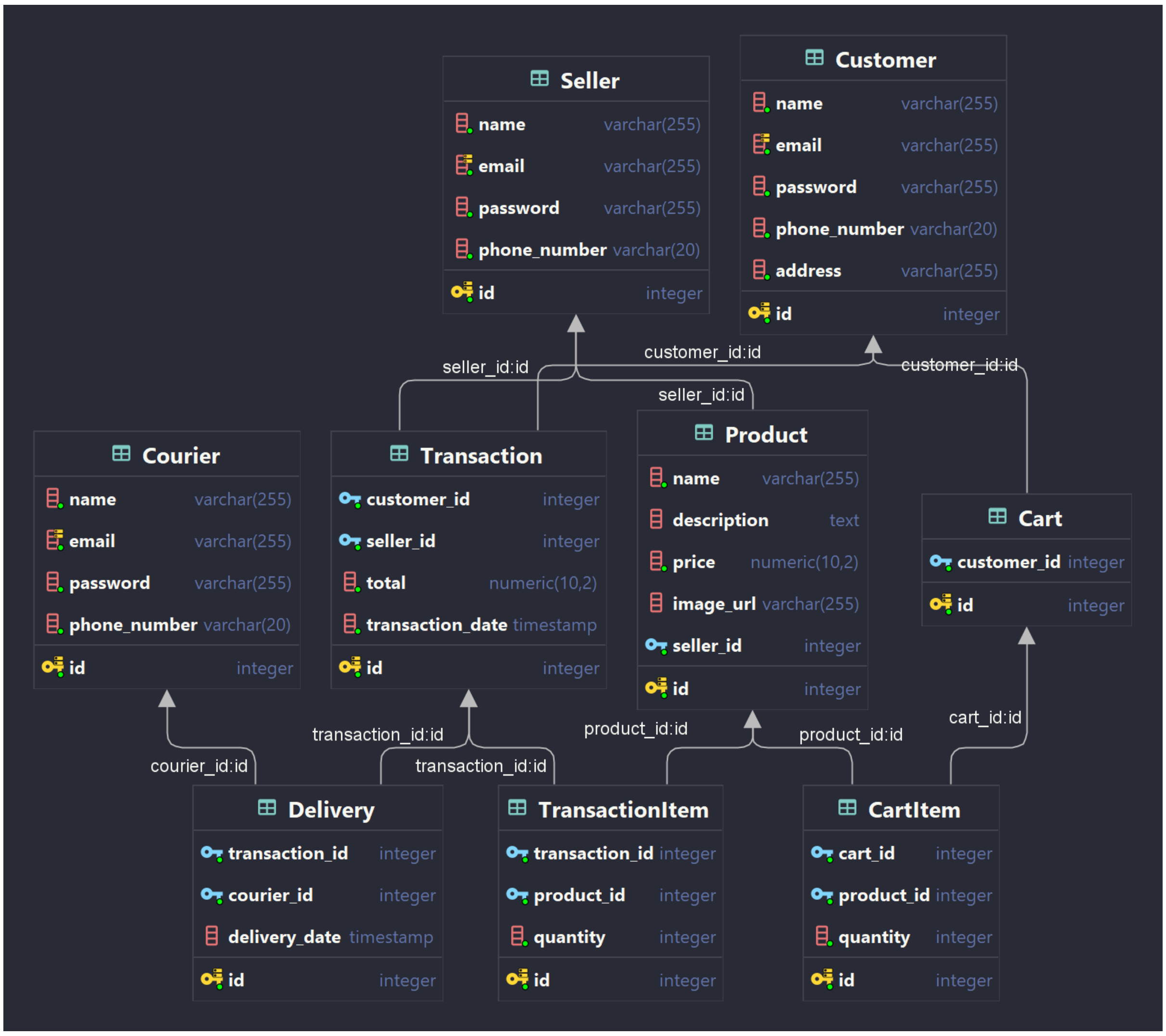
Requirements:

Write a report which will have the following structure:
- Introduction to the system
- ER diagram
- Explanation of why the structure follows normal forms
- Explanation and coding part of each item

1. Introduction to the system

The Online Toy Store is a database management system designed to handle the online transactions of a toy store. The database schema has been designed to capture all the necessary information required for processing transactions between customers, sellers, couriers, and the store. The database includes tables such as Customer, Seller, Product, Cart, Transaction, Transaction Item, Courier, and Delivery.

2. ERD:



3. Explanation of why the structure follows normal forms (1NF, 2NF, 3NF):

The database schema follows the rules of normalization to reduce data redundancy and maintain data integrity. The tables are structured in such a way that they follow the 1st, 2nd, and 3rd normal forms.

1NF (First Normal Form):
The database schema follows the 1st normal form, which ensures that each column in the table has atomic values. Each table has a primary key that uniquely identifies each row in the table. For example, the Customer table has a primary key 'id' that uniquely identifies each customer.

2NF (Second Normal Form):
The database schema follows the 2nd normal form, which ensures that all non-key attributes in a table are dependent on the primary key. The Product table, for instance, has a foreign key 'seller_id' that references the

Seller table's primary key 'id.' This way, the Product table's non-key attributes, such as 'name,' 'description,' 'price,' and 'image_url,' depend on the primary key of the Seller table.

3NF (Third Normal Form):
The database schema follows the 3rd normal form, which ensures that all non-key attributes in a table are independent of each other. The database schema avoids transitive dependencies, and each table is designed to store only one type of information. For instance, the Transaction table only stores information related to transactions and does not contain information related to Customers, Sellers, or Couriers.

4. Explanation and coding part of each item:

Tables Creation:

The first step in creating a database management system is to define the tables that will store the data. The following tables were created for this project:

Customer: This table contains information about the customers, such as their name, email, password, phone number, and address.

Seller: This table contains information about the sellers, such as their name, email, password, and phone number.

Product: This table contains information about the products, such as their name, description, price, image URL, and seller ID.

Cart: This table contains information about the carts, such as the customer ID.

CartItem: This table contains information about the items in the cart, such as the cart ID, product ID, and quantity.

Transaction: This table contains information about the transactions, such as the customer ID, seller ID, total price, and transaction date.

TransactionItem: This table contains information about the items in the transaction, such as the transaction ID, product ID, and quantity.

Courier: This table contains information about the couriers, such as their name, email, password, and phone number.

Delivery: This table contains information about the deliveries, such as the transaction ID, courier ID, and delivery date.

The following procedures were created for this project:

get_products_by_seller(): This procedure takes the name of a seller as input and returns the products sold by that seller. It uses a cursor to iterate through the products and a variable to count the number of products retrieved.

get_products_by_all_sellers(): This procedure calls the get_products_by_seller() procedure for each seller in the database.

group_by_info(): This procedure groups the sellers by name and counts the number of products sold by each seller. Initially, this procedure was causing an error due to the absence of a data result assignment, so we fixed it by using a record variable and looping through the query results.

The following functions were created for this project:

count_of_rows(): This function takes the name of a table as input and returns the number of rows in that table. It uses dynamic SQL to execute a SELECT COUNT(*) statement on the specified table and returns the count as an integer.

insert_product_name(): This function is designed to check and throw an exception when a new record is added to the Product table. If the length of the product name is less than 5, then the error will be: "Product name must be at least 5 characters". In other cases, as usual, data is added to the table.

The following trigger was created for this project:

log_table_size_trigger: This trigger is designed to display the amount of data in the table before adding data.