Efficient store management is critical for success in the retail industry. The Store Management System plays a vital role in achieving this goal by providing businesses with a digital framework to streamline their daily operations. This document aims to delve into the inner workings of the system, focusing on the essential elements required for its smooth operation. My project has two components: a database in the back-end, and a Java program in the front-end.

Here are the entities that I need for the system:

1. Supplier: Organizations that provide the store with the goods.
   1. Database: SUPPLIER Table with the following attributes:

* SupplierID: INT (Primary Key), range: 1 – 2,147,483,647
* SupplierName: VARCHAR(255), range: a string of up to 255 characters
* SupplierAddress: VARCHAR(255), range: a string of up to 255 characters
* SupplierPhone: VARCHAR(20), range: a string of up to 20 characters
  1. Memory: In the memory, each supplier is represented as an instance of the Supplier class model. The Supplier class encapsulates all relevant information about a supplier, including:
     1. Supplier ID
     2. Supplier name
     3. Supplier address
     4. Supplier phone number

Each instance of the Supplier class holds the specific details of a single supplier, allowing for easy access and manipulation within the application.

1. Product: Goods that the store currently has in-stock.
   1. Database: PRODUCT Table with the following attributes:

* ProductID: INT (Primary Key), range: 1 – 2,147,483,647
* SupplierID: INT (Foreign Key referring SupplierID in the SUPPLIER table), range: 1 – 2,147,483,647
* ProductName: VARCHAR(255), range: a string of up to 255 characters
* RetailPrice: DECIMAL(10, 2), range: 0.00 – 999,999,999.99
  1. Memory: Product entities are stored in the memory as instances of the Product class model. The Product class contains all relevant information about a product, including:
     1. Product ID
     2. Supplier ID
     3. Product name
     4. Retail price

Each instance of the Product class represents a unique product available in the store, enabling efficient management of product information and inventory.

1. Customer: Individuals who purchase goods from the store.
   1. Database: CUSTOMER Table with the following attributes:

* CustomerID: INT (Primary Key), range: 1 – 2,147,483,647
* CustomerName: VARCHAR(255), range: a string of up to 255 characters
* CustomerAddress: VARCHAR(300), range: a string of up to 300 characters
* CustomerPhone: VARCHAR(20), range: a string of up to 20 characters.
  1. Memory: Customer entities are stored in memory as instances of the Customer class model, which contains all relevant information about a customer, including:
     1. Customer’s ID
     2. Customer’s name
     3. Customer’s address,
     4. Customer’s phone number.

Instances of the Customer class hold individual customer details, facilitating customer management and order processing within the system.

1. Cust\_Order: Records customer orders placed in the store.
   1. Database: CUST\_ORDER Table with the following attributes:

* OrderID: INT (Primary Key), range: 1 – 2,147,483,647
* CustomerID: INT (Foreign Key referencing CustomerID in the CUSTOMER table), range: 1 – 2,147,483,647
* OrderDate: DATE, a string in DATE format recording the date and time when the order was placed.
  1. Memory: Customer orders are stored in memory as objects of the Cust\_Order class model. The Cust\_Order class includes attributes such as:
     1. Order ID
     2. Customer ID
     3. Order Date

The instance of the Cust\_Order class represents a single order placed by a customer, containing pertinent order information for order processing.

1. Order\_Detail: Records the details of a customer’s order.
   1. Database: ORDER\_DETAIL Table with the following attributes:

* OrderID: INT (Primary Key, Foreign Key referencing OrderID in the CUST\_ORDER table), range: 1 – 2,147,483,647
* ProductID: INT (Primary Key, Foreign Key referencing ProductID in PRODUCT table), range: 1 – 2,147,483,647
* Quantity: INT, range: 1 – 2,147,483,647
  1. Memory: Order details of customer orders are represented in memory by instances of the Order\_Detail class. The Order\_Detail class encapsulates details like:
     1. Order ID
     2. ProductID
     3. Quantity

Instances of the OrderDetail class contains details of a customer’s order, which helps facilitates ordering process and inventory management.

1. Payment: Manage order payments
   1. Database: PAYMENT table with the following attributes:

* PayID: INT (Primary Key, Foreign Key referencing OrderID in Cust\_Order table), range: 1 – 2,147,483,647
* CardNumber: VARCHAR(16), range: a string of up to 16 characters
* CardHolder: VARCHAR(255), range: a string of up to 255 characters
* ExpiryDate: DATE, range: a string in DATE format recording the expiration date of the card
* CVV: VARCHAR(3), range: a string of up to 3 characters
* Amount: DECIMAL(10, 2), range: 0.00 – 999,999,999.99
  1. Memory: Payments are stored in the memory as objects of the Payment class model. The Payment class includes attributes such as:
     1. Payment ID
     2. Card Number
     3. Card Holder name
     4. Expiry date
     5. CVV
     6. Payment Amount

Each instance of the Payment class represents a single payment transaction, containing relevant payment details for tracking and financial management within the system.

The back-end programs are developed in Java language. There are 21 class files, which handle the query and update of the tables in Customer, Order, Payment, Product, and Supplier entities, with the main program in the file “Application.java”. These are organized in a folder named Project01\_CSCE310, and compressed into a zip file named “Project01\_Final\_CSCE310.zip”.

To setup your testing environment, please download the zip file in a Windows environment and unzip it. You find the source code of the java “src” sub-folder. You may compile with xxx (Java compiler you used) to get the executables.