***Database Design***

**Project Name: -** **SareeOrderingApp**

**Description:-** In this project admin can login and add the number of items in the stock and

user can also login and select the item to add to cart and place order on his/her delivery

location. This project is done by using Core Java and JDBC.

**What is JDBC?**

[JDBC](https://www.geeksforgeeks.org/introduction-to-jdbc/) is an acronym for Java Database Connectivity. It’s an advancement for ODBC ( Open Database Connectivity ). JDBC is a standard API specification developed in order to move data from the front end to the back end. This API consists of classes and interfaces written in Java. It basically acts as an interface (not the one we use in Java) or channel between your Java program and databases i.e. it establishes a link between the two so that a programmer can send data from Java code and store it in the database for future use.

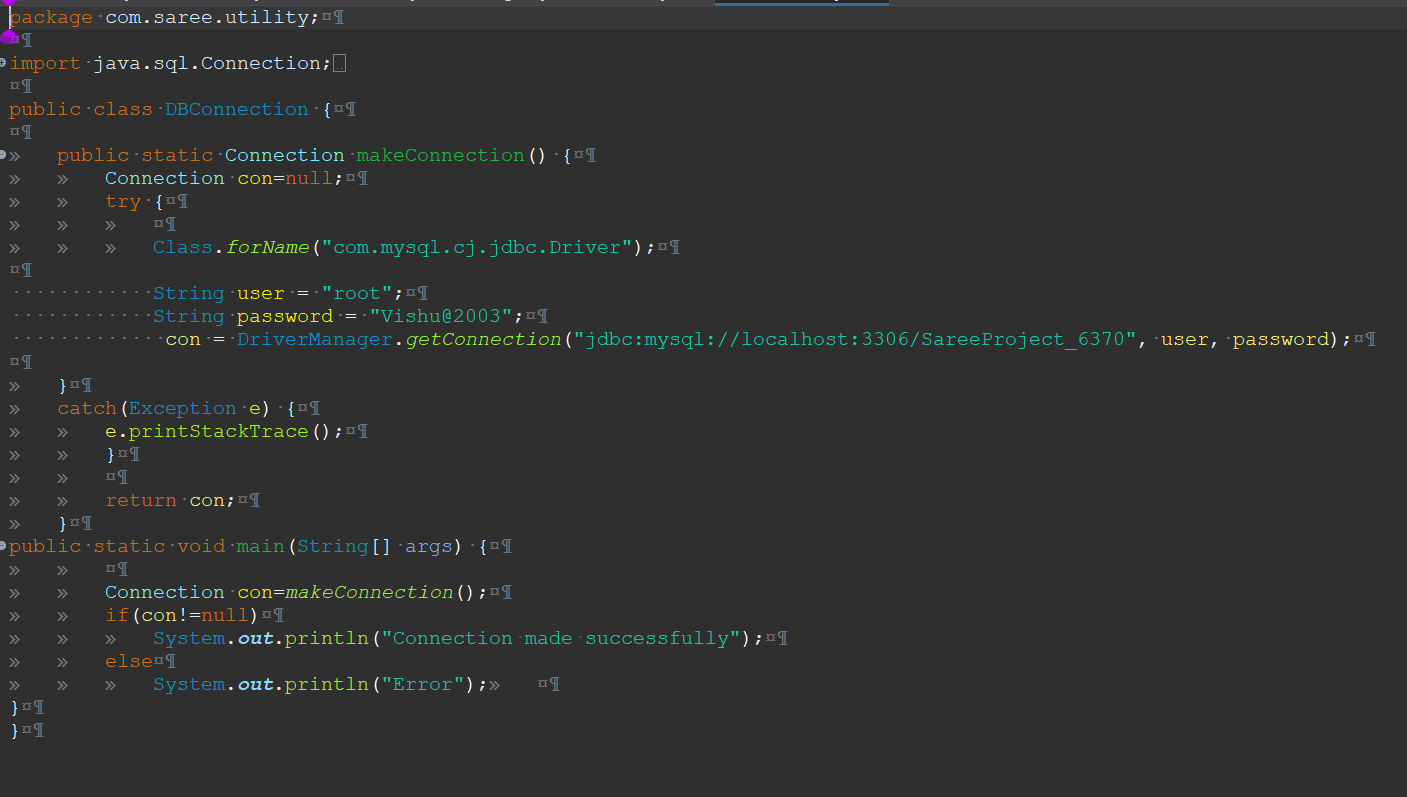
**Steps to Connect Java Application with Database**

**Below are the steps that explains how to connect to Database in Java:**

**Step 1** – Import the Packages  
**Step 2** – Load the drivers using the *forName() method*  
**Step 3** – Register the drivers *using DriverManager*  
**Step 4** – Establish a connection*using the Connection class object*  
**Step 5** – Create a statement  
**Step 6** – Execute the query  
**Step 7** – Close the connections

**Java Database Connectivity**

we load the driver’s class file into memory at the runtime. No need of using new or create objects.

****

### **Step 3:**Establish a connection using the Connection class object

After loading the driver, establish connections as shown below as follows:

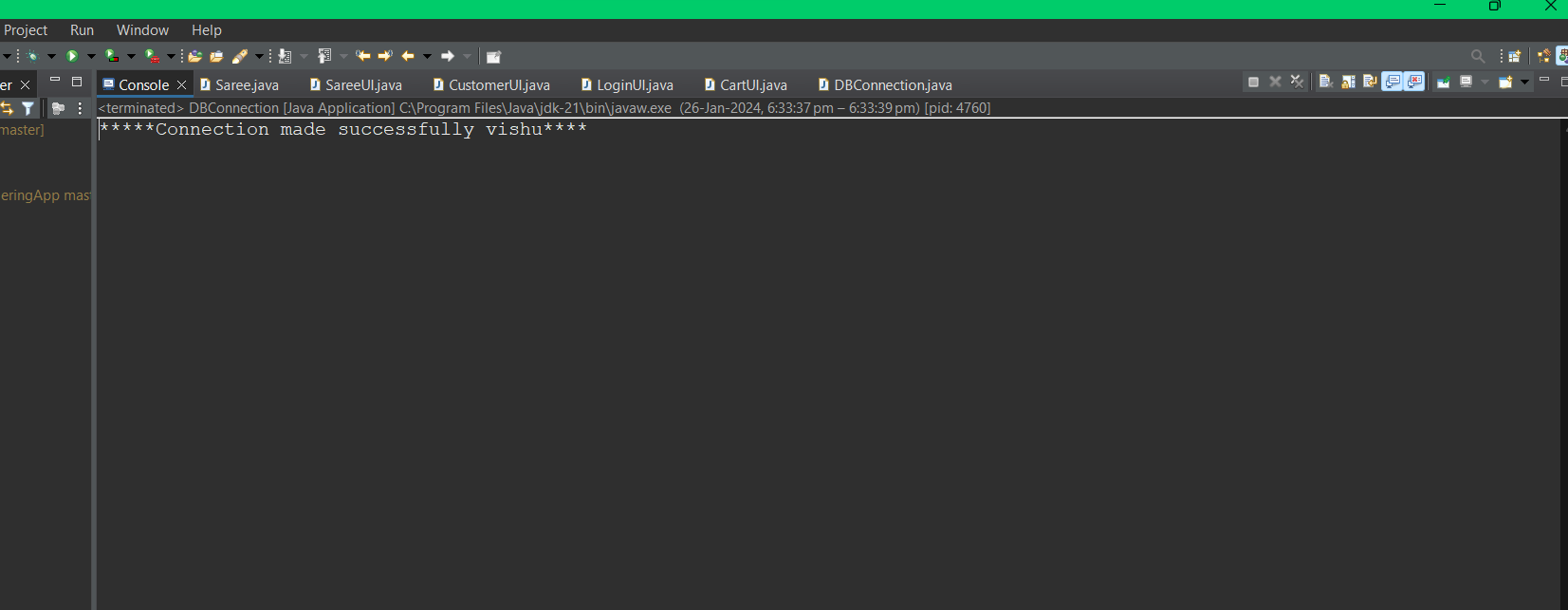
Connection con =

String user = "root";

String password = "Vishu@2003";

con = DriverManager.*getConnection*("jdbc:mysql://localhost:3306/SareeProject\_6370", user, password);

* **user: U**sername from which your SQL command prompt can be accessed.
* **password:**password from which the SQL command prompt can be accessed.
* **con:** It is a reference to the Connection interface.
* **Url**: Uniform Resource Locator which is created as shown above

**After apply all steps you get result:-**

Now, we will use MySQL as the DBMS to create the database and its related operation

1. **Introduction to MySQL and How to installing:-**

MySQL is an open-source relational database management system(RDBMS) that uses structured query language

(SQL) to manage and manipulate data in a database. It is widely used for various applications, from small web applications to large enterprise systems.

**Installation of MySQL**

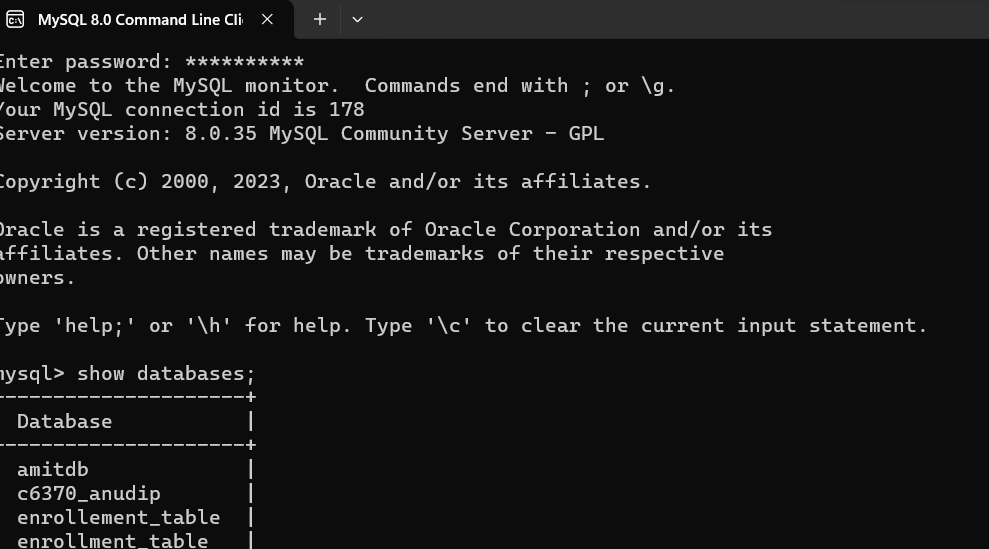
MySQL can be installed on various operating systems, including Windows, macOS, and Linux. Here are the general steps to install MySQL:

**Windows:**

* Download the MySQL installer from the official website.

<https://dev.mysql.com/downloads/installer/>

* Run the installer and follow the on-screen instructions.
* Choose the installation type (Typical, Complete, or Custom). Recommended Custom.
* Set a root password for the MySQL server.



Now Let’s see E-R Diagram….

**2. E-R Diagram :-**

An Entity-Relationship Diagram(ERD) is a visual representation of the data model that shows the entities,

attributes, relationships between entities and cardinality. ERDs are commonly used in database design to

help developers and stakeholders understand the structure and relationships within a database.Now let’s identify the attributes of each entity POJO Class for the SareeOrderingApp:-

**Saree**

* Attributes: -

1. sareeId(primarykey)
2. sareeName
3. type; //Regular, Partywear, wedding, Girls Wear
4. category; //Banarasi, Mysore silk, Kanchipuram, maduvaniprint...
5. price
6. quantityInstock;
7. color;//red blue green...

**Customer: -**

• Attributes: -

1. Customer Id(Primary Key)
2. Customer Name
3. Customer Password
4. Customer email Id
5. Customer Contact Number
6. Customer DropingAddress
7. Customer Address

**Order: -**

• Attributes: -

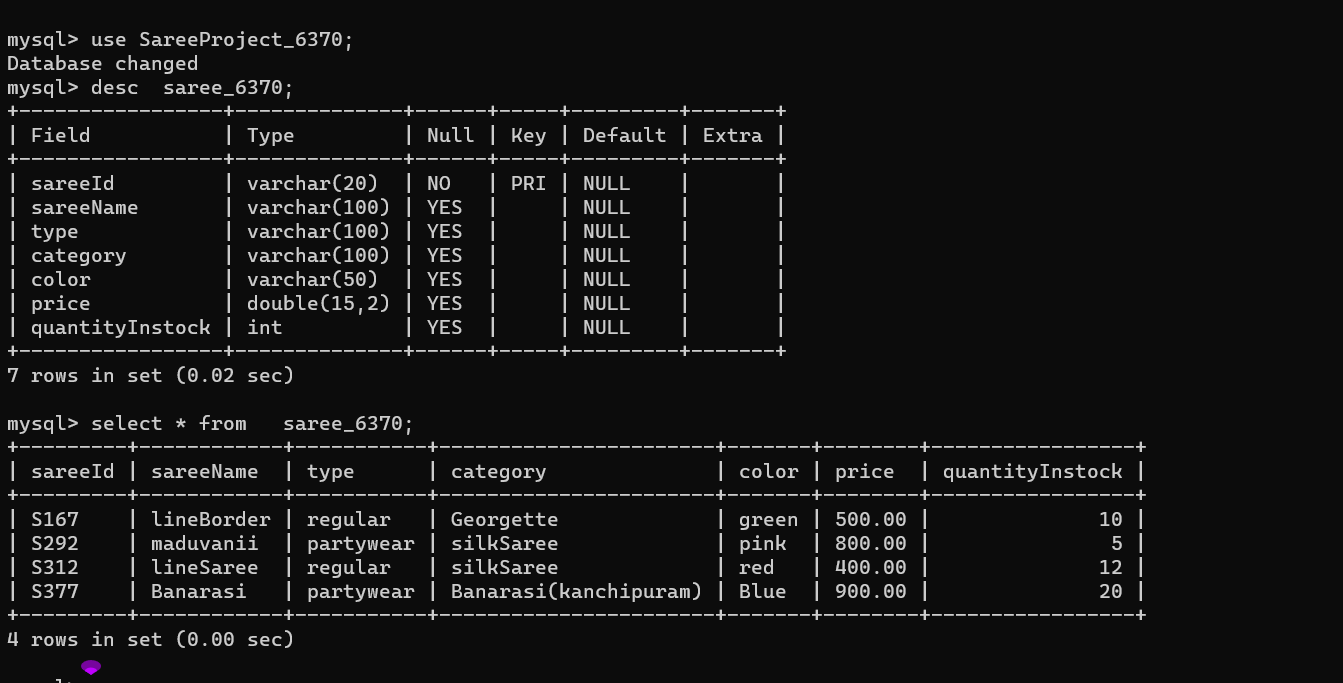
1. Order Id(Primary Key)
2. Order Date
3. Drop Location
4. Expected Delivery
5. Customer Id
6. Billing Amount
7. Status

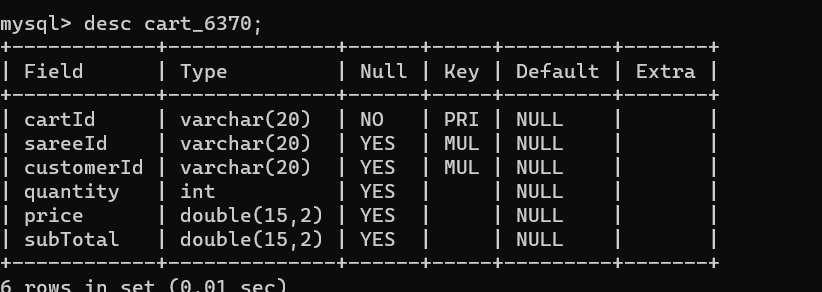
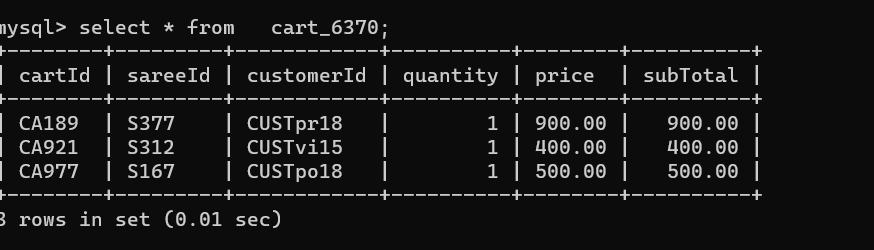
**Cart: -**

• Attributes: -

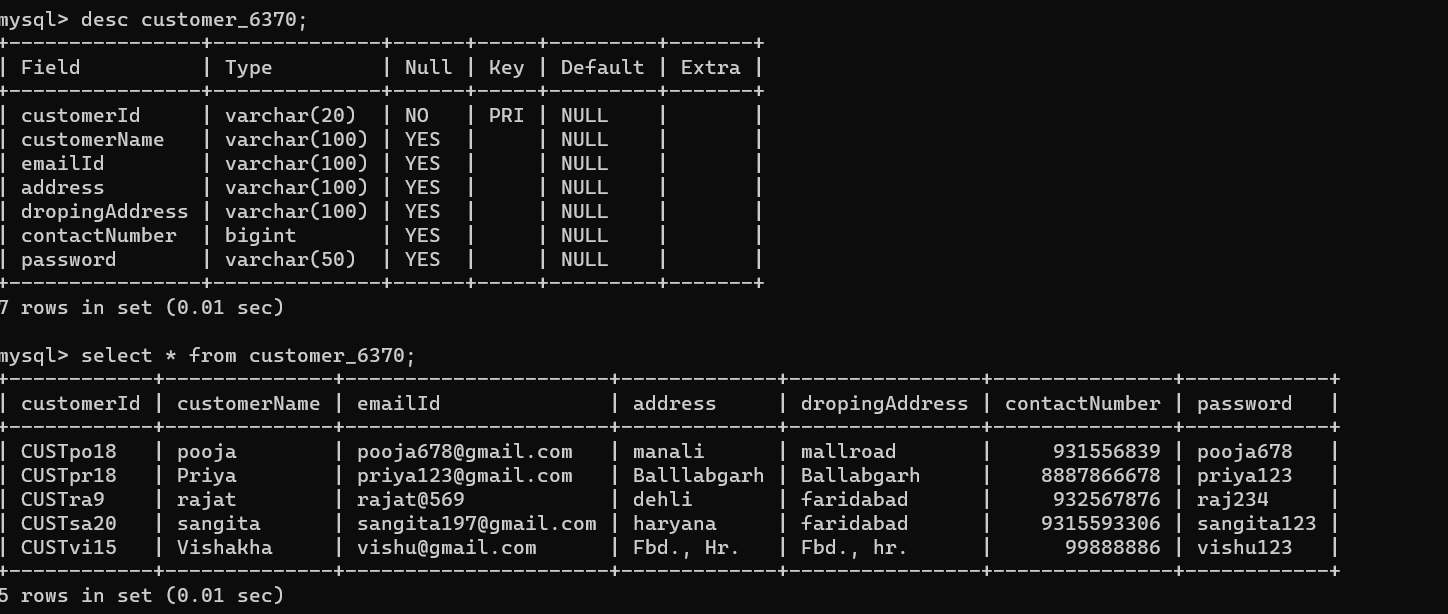
1. cartId;
2. sareeId;
3. email Id;
4. quantity;
5. customerId;
6. subtotal;//price \* quantity
7. price;//saree price

* Table Structures

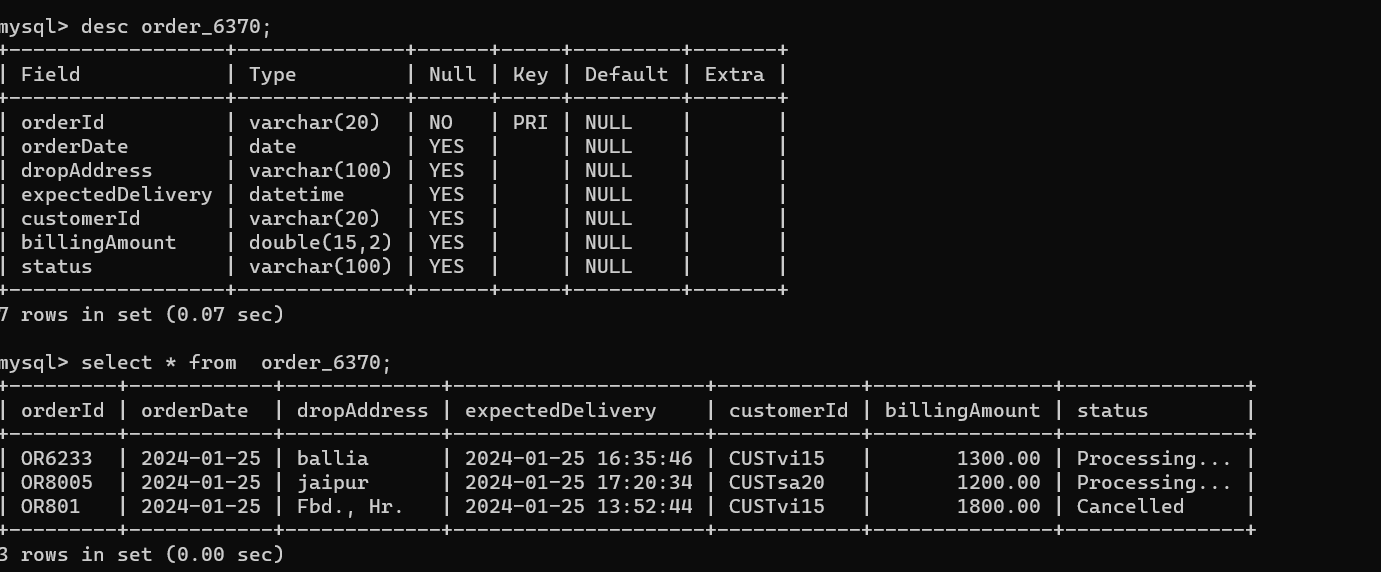
1. Saree:-
2. Cart: -



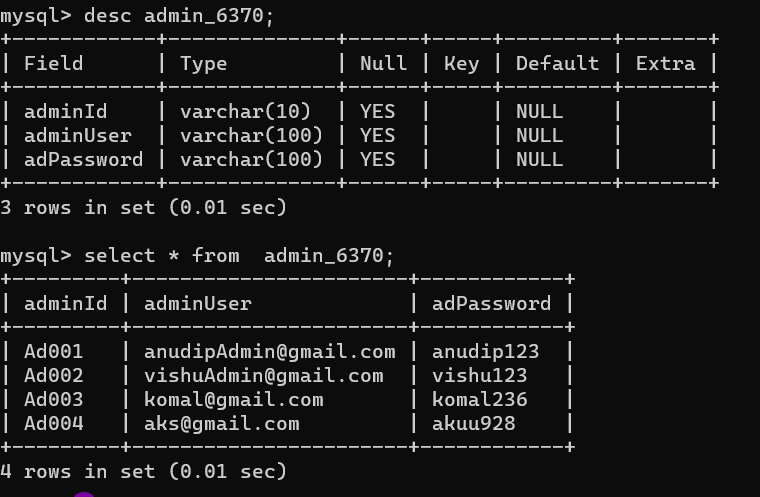
3)Customer: -



**4)Order: -**

****

**5) Admin: -**

****

ERD Diagram: -

Saree

Cart

Changes

Login By

buying

Order

Customer

Add Cart

by

**Admin:-**

* **Attributes**

1. **Admin Id (Primary Key)**
2. **Admin user**
3. **•Admin password**

• Relationships Between Entities:

1. Customer and Order:

Description: A customer places one or more orders (one to many relation).

2. Customer and Cart:

Relationship Name: Add cart

Description: A Customer can add multiple items to their cart

(one to many relation).

3. Cart and saree:

Relationship: changes

Description: A cart contains multiple items’ saree and a saree can be in

multiple carts (many to many relation)

5. Admin and Cart:

Relationship: Manages

Description: An admin manages carts, overseeing and possibly

modifying cart contents. This relationship indicates the administrative

control or oversight over the shopping carts within the system (many to

many).

6. Admin and Order:

Relationship: manages

Description: An admin manages orders, overseeing and possibly

modifying order details (many to many).

7. Admin and saree:

Relationship: by login

Description: An admin, as a manager, oversees the list of

available plants. This relationship signifies the administrative control

or oversight over the plant catalog within the system.