Database Design Guide

This guide will help the student to create a database on the Wakefit . It will help to manage the below functionalities.

1. Customer
2. Admin
3. Product
4. Category
5. Order
6. Order_item /Shipping
7.ShippingCart
8. payment
9. feedback

We will use MySQL as the DBMS to create the database and its related operations.

1. Introduction to MySQL

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.

MySQL's key features include:

- Scalability: Capable of handling large amounts of data and concurrent connections.
- Flexibility: Supports various data types and storage engines.
- Performance: Optimized for speed and efficiency.
- Reliability: Known for its stability and robustness.

2. 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.

3. E-R Diagram (ERD)

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.

Identify Entities

- Start by identifying the main entities in your system. These are the objects or concepts about which you want to store data.

- Each entity should correspond to a table in your database.

Define Attributes

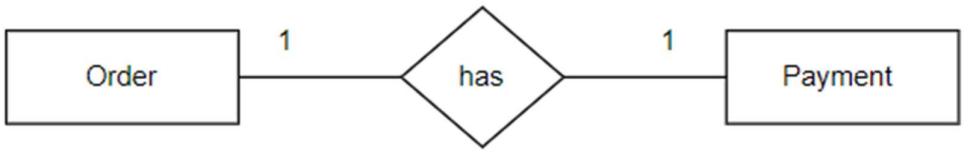
- For each entity, list the attributes (properties or fields) that describe it.
- These attributes will become columns in the corresponding database table.

Identify Relationships

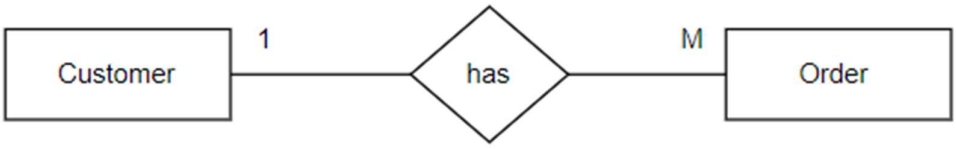
- Determine how entities are related to each other. There are three types of relationships: one-to-one (1:1), one-to-many (1:N), and many-to-many (N:M).
- Represent these relationships using lines connecting the entities.

Let's see a few examples of relationships:

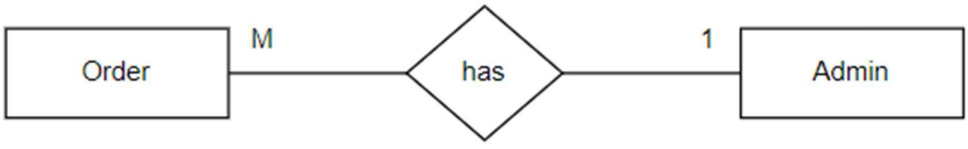
One to One



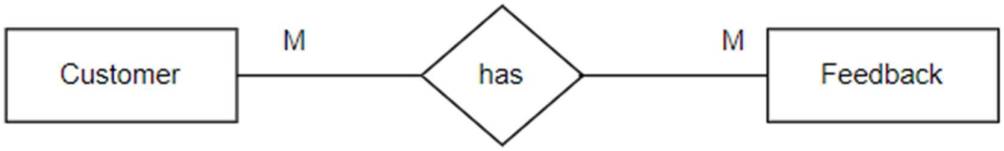
One to Many



Many to One



Many to Many



Create the Diagram

- Use specialized diagramming software or tools (e.g., Lucidchart, draw.io, or even pen and paper) to create your ERD.

Refine and Review:

- Review your ERD with stakeholders and team members to ensure it accurately represents the data model and relationships. Make any necessary refinements.

Let's identify the entities of the WakeFit

1. Customer
2. Admin
3. Product
4. Category
5. Order
6. Order_item /Shipping
7.ShippingCart
8. payment
9. feedback

Now let's identify the attributes and relationships of each entity for the Wakefit:

Category Table:

- **Attributes:**

catID: Primary key for the category.

categoryName: Name of the category.

- **Relationships:**

Each category can have multiple products (One-to-Many relationship with Product table).

Product Table:

- **Attributes:**

pID: Primary key for the product.

pName: Name of the product.

pDescription: Description of the product.

pUnitPrice: Price of the product.

pImageUrl: URL of the product image.

active: Indicates whether the product is active or not.

unitsInStock: Number of units in stock.

dateCreated: Date when the product was created.

lastUpdated: Date when the product was last updated.

catID: Foreign key referencing the catID in the Category table, indicating the category to which the product belongs.

- **Relationships:**

Each product belongs to one category (Many-to-One relationship with Category table).

- **Relationships:**

Each product can be purchased by multiple customers (One-to-Many relationship with Customer table).

Customer Table:

- **Attributes:**

cID: Primary key for the customer.

cName: Name of the customer.

cEmail: Email address of the customer.

cMobNo: Mobile number of the customer.

cAddress: Address of the customer.

- **Relationships:**

Each customer can purchase multiple products (One-to-Many relationship with Product table).

Feedback Table:

- **Attributes:**

fID: Primary key for the feedback.

message: Feedback message.

cID: Foreign key referencing the cID in the Customer table, indicating the customer who provided the feedback.

- **Relationships:**

Each feedback is provided by one customer (Many-to-One relationship with Customer table).

Order Table:

Attributes:

oID: Primary key for the order.

orderTrackingNumber: Tracking number of the order.

totalPrice: Total price of the order.

oQuantity: Quantity of items in the order.

status: Status of the order.

dateCreated: Date when the order was created.

lastUpdated: Date when the order was last updated.

cID: Foreign key referencing the cID in the Customer table, indicating the customer who placed the order.

- **Relationships:**

Each order is placed by one customer (Many-to-One relationship with Customer table).

Payment Table:

- **Attributes:**

pID: Primary key for the payment.

pMethod: Payment method used.

pDate: Date of the payment.

status: Status of the payment.

oID: Foreign key referencing the oID in the Order Table, indicating the order to which the payment corresponds.

- **Relationships:**

Each payment corresponds to one order (Many-to-One relationship with Order Table).

Shipping Table:

- **Attributes:**

sID: Primary key for the shipping.

oID: Foreign key referencing the oID in the Order Table, indicating the order for which the shipping is done.

- **Relationships:**

Each shipping operation is associated with one order (Many-to-One relationship with Order Table).

ShoppingCart Table:

- **Attributes:**

id: Primary key for the shopping cart.

imageUrl: URL of the product image in the shopping cart.

unitPrice: Unit price of the product in the shopping cart.

quantity: Quantity of the product in the shopping cart.

productId: Identifier of the product in the shopping cart.

cID: Foreign key referencing the cID in the Customer table, indicating the customer who owns the shopping cart.

- **Relationships:**

Each item in the shopping cart is owned by one customer (Many-to-One relationship with Customer table).

Table Structure

1.Admin

```
mysql> desc admin;
+--------+-------------+------+-----+---------+-------+
| Field  | Type        | Null | Key | Default | Extra |
+--------+-------------+------+-----+---------+-------+
| aID    | int         | NO   | PRI | NULL    |       |
| aName  | varchar(255) | YES  |     | NULL    |       |
| Email  | varchar(255) | YES  |     | NULL    |       |
| aMobNo | int         | YES  |     | NULL    |       |
+--------+-------------+------+-----+---------+-------+
```

2.Category

```
mysql> desc category;
+--------------+-------------+------+-----+---------+-------+
| Field        | Type        | Null | Key | Default | Extra |
+--------------+-------------+------+-----+---------+-------+
| catID        | int         | NO   | PRI | NULL    |       |
| categoryName | varchar(255) | YES  |     | NULL    |       |
+--------------+-------------+------+-----+---------+-------+
2 rows in set (0.00 sec)
```

3.Customer

```
mysql> desc customer;
+----------+-------------+------+-----+---------+-------+
| Field    | Type        | Null | Key | Default | Extra |
+----------+-------------+------+-----+---------+-------+
| cID      | int         | NO   | PRI | NULL    |       |
| cName    | varchar(255) | YES  |     | NULL    |       |
| cEmail   | varchar(255) | YES  |     | NULL    |       |
| cMobNo   | int         | YES  |     | NULL    |       |
| cAddress | text        | YES  |     | NULL    |       |
+----------+-------------+------+-----+---------+-------+
5 rows in set (0.00 sec)
```

4.Feedback

```
mysql> desc feedback;
+---------+--------------+------+-----+---------+-------+
| Field   | Type         | Null | Key | Default | Extra |
+---------+--------------+------+-----+---------+-------+
| fID     | int          | NO   | PRI | NULL    |       |
| message | text         | YES  |     | NULL    |       |
| cID     | int          | YES  | MUL | NULL    |       |
| cName   | varchar(255) | YES  |     | NULL    |       |
| cEmail  | varchar(255) | YES  |     | NULL    |       |
+---------+--------------+------+-----+---------+-------+
5 rows in set (0.00 sec)
```

5.Order_table

```
mysql> desc order_table;
+--------------------+-------------+------+-----+---------+-------+
| Field              | Type        | Null | Key | Default | Extra |
+--------------------+-------------+------+-----+---------+-------+
| oID                | int         | NO   | PRI | NULL    |       |
| orderTrackingNumber | varchar(255) | YES  |     | NULL    |       |
| totalPrice         | float       | YES  |     | NULL    |       |
| oQuantity          | int         | YES  |     | NULL    |       |
| status             | varchar(255) | YES  |     | NULL    |       |
| dateCreated        | date        | YES  |     | NULL    |       |
| lastUpdated        | date        | YES  |     | NULL    |       |
| cID                | int         | YES  | MUL | NULL    |       |
+--------------------+-------------+------+-----+---------+-------+
8 rows in set (0.00 sec)
```

6.Payment

```
mysql> desc payment;
+---------+--------------+------+-----+---------+-------+
| Field   | Type         | Null | Key | Default | Extra |
+---------+--------------+------+-----+---------+-------+
| pID     | int          | NO   | PRI | NULL    |       |
| pMethod | varchar(255) | YES  |     | NULL    |       |
| pDate   | date         | YES  |     | NULL    |       |
| status  | varchar(255) | YES  |     | NULL    |       |
| oID     | int          | YES  | MUL | NULL    |       |
+---------+--------------+------+-----+---------+-------+
5 rows in set (0.00 sec)
```

7.Product

```
mysql> desc  product;
+-------------+-------------+------+-----+---------+-------+
| Field       | Type        | Null | Key | Default | Extra |
+-------------+-------------+------+-----+---------+-------+
| pID         | int         | NO   | PRI | NULL    |       |
| pName       | varchar(255)| YES  |     | NULL    |       |
| pDescription| text        | YES  |     | NULL    |       |
| pUnitPrice  | float       | YES  |     | NULL    |       |
| pImageUrl   | varchar(255)| YES  |     | NULL    |       |
| active      | tinyint(1)  | YES  |     | NULL    |       |
| unitsInStock| int         | YES  |     | NULL    |       |
| dateCreated | date        | YES  |     | NULL    |       |
| lastUpdated | date        | YES  |     | NULL    |       |
| catID       | int         | YES  | MUL | NULL    |       |
+-------------+-------------+------+-----+---------+-------+
10 rows in set (0.00 sec)
```

8.Shipping

```
mysql> desc  shipping;
+-------+------+------+-----+---------+-------+
| Field | Type | Null | Key | Default | Extra |
+-------+------+------+-----+---------+-------+
| sID   | int  | NO   | PRI | NULL    |       |
| oID   | int  | YES  | MUL | NULL    |       |
+-------+------+------+-----+---------+-------+
2 rows in set (0.00 sec)
```

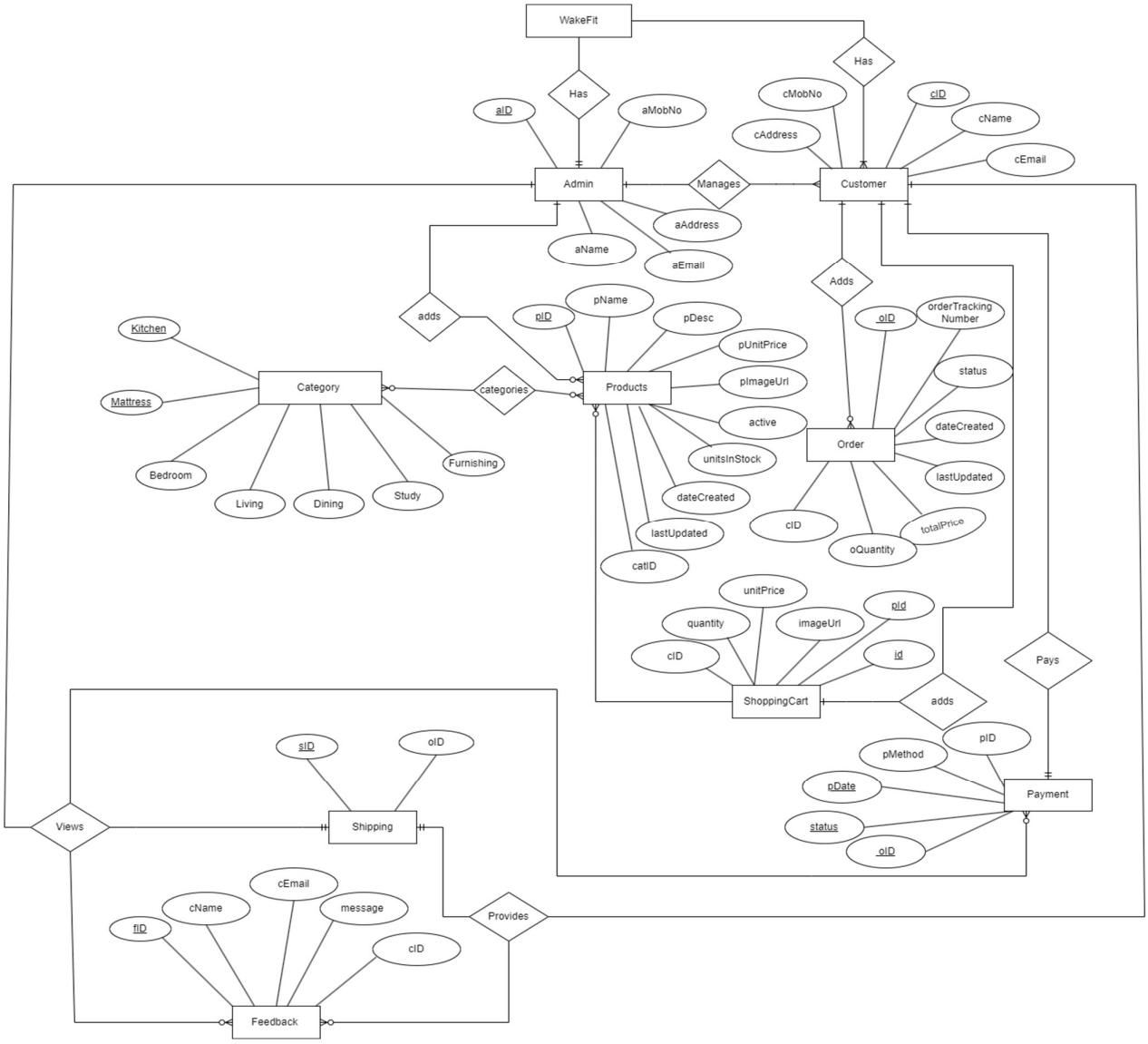
9. ShoppingCart

```
mysql> desc shoppingcart;
+-----------+-------------+------+-----+---------+-------+
| Field     | Type        | Null | Key | Default | Extra |
+-----------+-------------+------+-----+---------+-------+
| id        | int         | NO   | PRI | NULL    |       |
| imageUrl  | varchar(255)| YES  |     | NULL    |       |
| unitPrice | float       | YES  |     | NULL    |       |
| quantity  | int         | YES  |     | NULL    |       |
| productId | int         | YES  |     | NULL    |       |
| cID       | int         | YES  | MUL | NULL    |       |
+-----------+-------------+------+-----+---------+-------+
6 rows in set (0.00 sec)
```

Now, let’s create the ER diagram to visually represent the entities and relationships.

ERD Diagram

In this ERD:



In this ERD:

- One Customer can purchase more than one product create One-to-Many relationship
- Each category can have multiple products (One-to-Many relationship with Product table
- Each product can be purchased by multiple customers (One-to-Many relationship with Customer table).
- Each feedback is provided by one customer (Many-to-One relationship with Customer table).
- Each order is placed by one customer (Many-to-One relationship with Customer table).
- Each shipping operation is associated with one order (Many-to-One relationship with Order Table).
- Each item in the shopping cart is owned by one customer (Many-to-One relationship with Customer table).

4. Creating a Database
Using MySQL server, create a new database for your student management system. You can do this with SQL commands or through the graphical interface.

CREATE DATABASE wakefitFurniture;

5. Using a Database
Before performing any operations on a database, you need to select it using the USE statement:

USE wakefitFurniture;

6. Creating the tables for each entity

USE wakefitFurniture;

```
CREATE TABLE admin (
   aID INT PRIMARY KEY,
   aName VARCHAR(255),
   Email VARCHAR(255),
   aMobNo INT
);

Desc admin;

CREATE TABLE category (
   catID INT PRIMARY KEY,
   categoryName VARCHAR(255)
);

Desc category;
```

```
CREATE TABLE product (
   pID INT PRIMARY KEY,
   pName VARCHAR(255),
   pDescription TEXT,
   pUnitPrice FLOAT,
   pImageUrl VARCHAR(255),
   active BOOLEAN,
   unitsInStock INT,
   dateCreated DATE,
   lastUpdated DATE,
   catID INT,
   FOREIGN KEY (catID) REFERENCES category(catID)
);

Desc product;

CREATE TABLE customer (
   cID INT PRIMARY KEY,
   cName VARCHAR(255),
   cEmail VARCHAR(255),
   cMobNo INT,
   cAddress TEXT
);

Desc customer;

CREATE TABLE feedback (
   fID INT PRIMARY KEY,
   message TEXT,
   cID INT,
   FOREIGN KEY (cID) REFERENCES customer(cID)
);
ALTER TABLE feedback
ADD COLUMN customerName VARCHAR(255),
ADD COLUMN customerEmail VARCHAR(255);

ALTER TABLE feedback
CHANGE COLUMN customerName cName VARCHAR(255),
CHANGE COLUMN customerEmail cEmail VARCHAR(255);

desc feedback;


CREATE TABLE order_table (
   oID INT PRIMARY KEY,
   orderTrackingNumber VARCHAR(255),
   totalPrice FLOAT,
   oQuantity INT,
   status VARCHAR(255),
```

```
    dateCreated DATE,
    lastUpdated DATE,
    cID INT,
    FOREIGN KEY (cID) REFERENCES customer(cID)
);

Desc order_table;

CREATE TABLE payment (
    pID INT PRIMARY KEY,
    pMethod VARCHAR(255),
    pDate DATE,
    status VARCHAR(255),
    oID INT,
    FOREIGN KEY (oID) REFERENCES order_table(oID)
);

Desc payment;

CREATE TABLE shipping (
    sID INT PRIMARY KEY,
    oID INT,
    FOREIGN KEY (oID) REFERENCES order_table(oID)
);

Desc shipping;

CREATE TABLE shoppingcart (
    id INT PRIMARY KEY,
    imageUrl VARCHAR(255),
    unitPrice FLOAT,
    quantity INT,
    productId INT,
    cID INT,
    FOREIGN KEY (cID) REFERENCES customer(cID)
);

Desc shippingCart;
```

7. Insert records

Add data to your tables to work with. This step helps you test your database.

-- Insert Admin

```
INSERT INTO admin (aID, aName, Email, aMobNo)

VALUES (1, 'admin', 'admin@gmail.com', '987654321');
```

-- Insert Category

```
INSERT INTO category (catID, categoryName)
VALUES (1, 'Mattress'),
    (2, 'Bedroom'),
    (3, 'Living'),
    (4, 'Dining'),
    (5, 'Study'),
    (6, 'Furnishing'),
    (7, 'Kitchen'),
    (8, 'Essentials'),
    (9, 'Decor'),
    (10, 'Kids'),
    (11, 'Plus Series');
```

-- Insert Product

```
INSERT INTO product (pID, pName, pDescription, pUnitPrice, pImageUrl, active, unitsInStock,
dateCreated, lastUpdated, catID)

VALUES

(1, 'Orthopedic Memory Foam Mattress', 'Orthopedic memory foam mattress for comfortable sleep.',
10000.00, 'mattress1.jpg', 1, 100, NOW(), NOW(), 1),

(2, 'Queen Size Bed', 'Queen size bed made of solid wood.', 20000.00, 'bed1.jpg', 1, 50, NOW(),
NOW(), 2),

(3, 'L-shaped Sofa Set', 'L-shaped sofa set for your living room.', 30000.00, 'sofa1.jpg', 1, 30, NOW(),
NOW(), 3),

(4, 'Dining Table with Chairs', 'Wooden dining table with 4 chairs.', 25000.00, 'dining1.jpg', 1, 20,
NOW(), NOW(), 4),

(5, 'Study Table with Bookshelf', 'Study table with attached bookshelf for easy storage.', 15000.00,
'study1.jpg', 1, 40, NOW(), NOW(), 5),

(6, 'Curtains Set', 'Set of 2 curtains for your windows.', 5000.00, 'curtains1.jpg', 1, 60, NOW(),
NOW(), 6),

(7, 'Modular Kitchen Set', 'Modular kitchen set with cabinets and countertops.', 50000.00,
'kitchen1.jpg', 1, 10, NOW(), NOW(), 7),

(8, 'Home Decorative Vases', 'Set of decorative vases to enhance your home decor.', 8000.00,
'decor1.jpg', 1, 70, NOW(), NOW(), 9),

(9, 'Kids Bunk Bed', 'Bunk bed for kids with ladder and safety railings.', 18000.00, 'kids1.jpg', 1, 25,
NOW(), NOW(), 10),

(10, 'Plus Series Recliner Sofa', 'Luxurious recliner sofa from the Plus Series.', 40000.00, 'plus1.jpg',
1, 15, NOW(), NOW(), 11);
```

-- Insert Customer

```
INSERT INTO customer (cID, cName, cEmail, cMobNo, cAddress)

VALUES (1, 'Kunal Kadam', 'kunal@gmail.com', 987654321, 'satara, City'),

        (2, 'Sagar Kurde', 'Sagar@gmail.com', 987654321, 'indapur, Town');
```

-- Insert Order_Table

```
INSERT INTO order_table (oID, orderTrackingNumber, totalPrice, oQuantity, status, dateCreated,
lastUpdated, cID)VALUES (1, 'ORD123', 100.00, 2, 'Pending', NOW(), NOW(), 1),

                        (2, 'ORD456', 50.00, 1, 'Delivered', NOW(), NOW(), 2);
```

-- Insert payment

```
INSERT INTO payment (pID, pMethod, pDate, status, oID)
VALUES (1, 'Cash', NOW(), 'Success', 1),
         (2, 'Gpay', NOW(), 'Success', 2);
```

-- Insert feedback

```
INSERT INTO feedback (fID, message, cID)
VALUES (1, 'Great service!', 1),
     (2, 'Product arrived on time', 2);
 UPDATE feedback
SET customerName = 'sagar kurde', customerEmail = 'sagar@gmail.com'
WHERE fID = 2;
```

-- Insert shipping

```
INSERT INTO shipping (sID, oID)
VALUES (1, 1),
     (2, 2);
```

-- Insert shippingCart

```
INSERT INTO shoppingcart (id, imageUrl, unitPrice, quantity, productId, cID)
VALUES (1, 'product1.jpg', 25.00, 2, 1, 1),
         (2, 'product2.jpg', 30.00, 1, 2, 2);
```

8. Select records

Write SQL queries to retrieve and manage data.

For example:

Retrieve all orders:

*Select * FROM order_table;*

Retrieve a orders that are pending:

SELECT oID, orderTrackingNumber, totalPrice, oQuantity, status, dateCreated, lastUpdated, cID
FROM order_table
WHERE status = 'Pending';

***Now try similar Select queries with other tables**

9. Update records

Write SQL statements to update record(s) when needed. For example:

```
select * from customer;

UPDATE customer
SET cEmail = 'KK@gmail.com'
WHERE cID = 1;
```

10. Delete records

Write SQL statements to delete record(s) when needed.

```
DELETE FROM customer

WHERE cName = 'Rahul Deshmukh';
```