Database Design Guide

This guide will help the student to create a database on the Online Electricity Billing System. It will help to manage the below functionalities.

1.Product

2.Product\_Category

3.Customer

4.Orders

5.Order\_item

6.State

7.country

8.payment

9.feedback

10. address

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

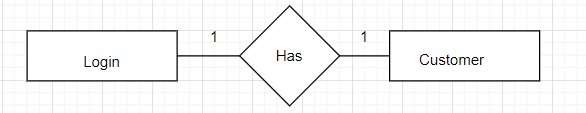
* For each entity, list the attributes (properties or fields) that describe it.
* These attributes will become columns in the corresponding database table.

# Identify Relationships

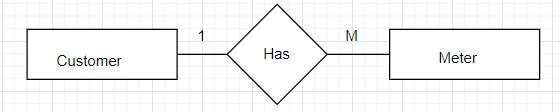
* Determine how entities are related to each other. There are three types of relationships: one-to-one (1:1), one-to-many (1:N), and many-to-many (N:M).
* Represent these relationships using lines connecting the entities.

Let’s see a few examples of relationships:

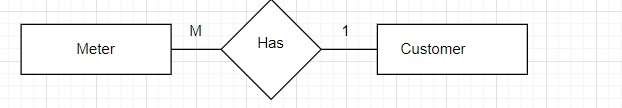
# One to One



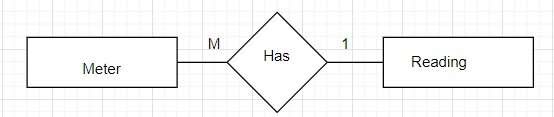
# One to Many



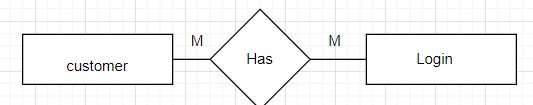
# Many to One



# Many to One



# Many to Many



# Cardinality Notation

Cardinality represents the number of times an entity of an entity set participates in a relationship set. Or we can say that the cardinality of a relationship is the number of tuples (rows) in a relationship.

* Use notation (such as Crow's Foot Notation or Chen Notation) to indicate the cardinality of each relationship.
* Cardinality describes how many instances of one entity are related to how many instances of another entity.
* Common notations include:
* One (1)
* Zero or one (0..1)
* Many (N)
* Zero or many (0..N)

Optional:

# Add Attributes and Constraints

 Include additional information in your ERD, such as primary keys, foreign keys, and constraints (e.g., unique constraints).

# 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.Product

2.Product\_Category

3.Customer

4.Orders

5.Order\_item

6.State

7.country

8.payment

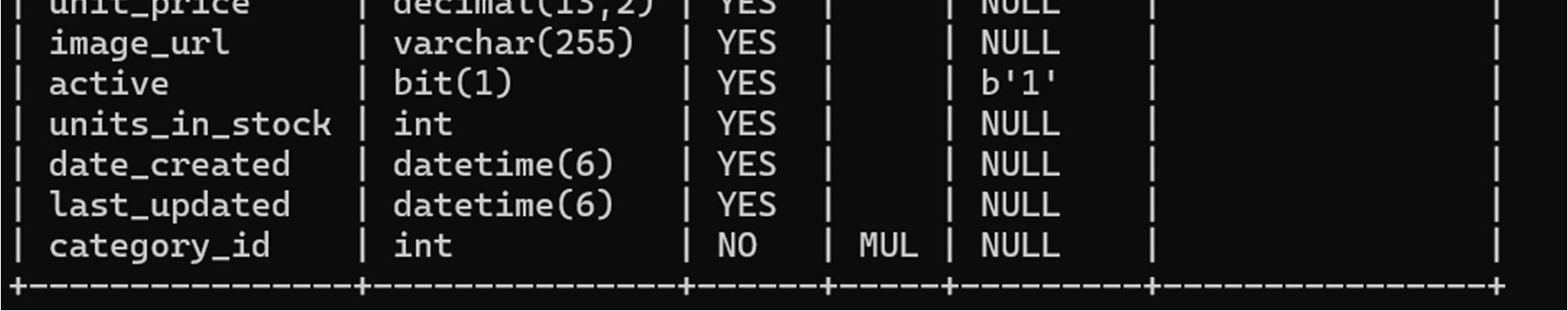
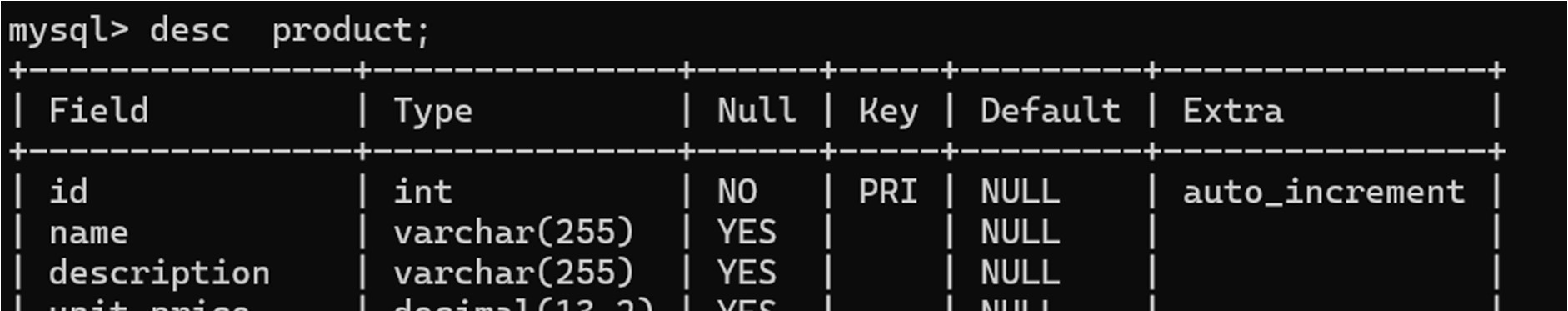
9.feedback

10. address

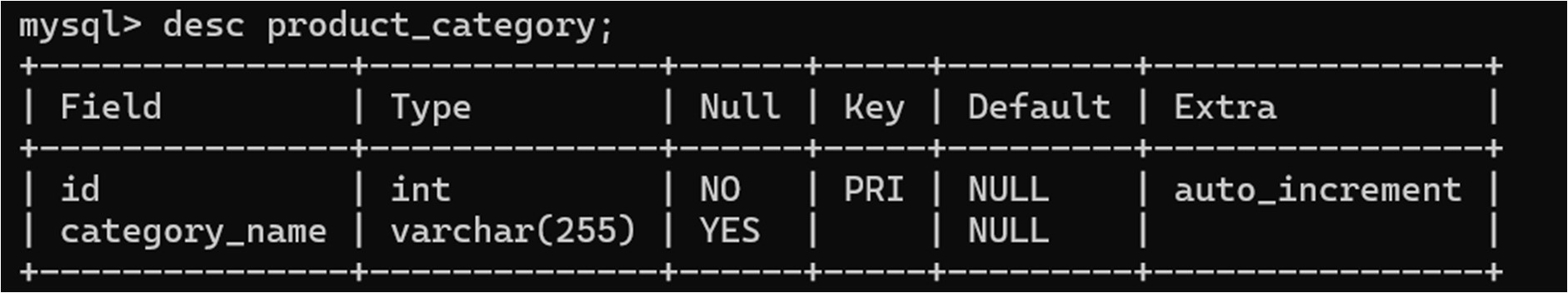
11.login

Table Structure

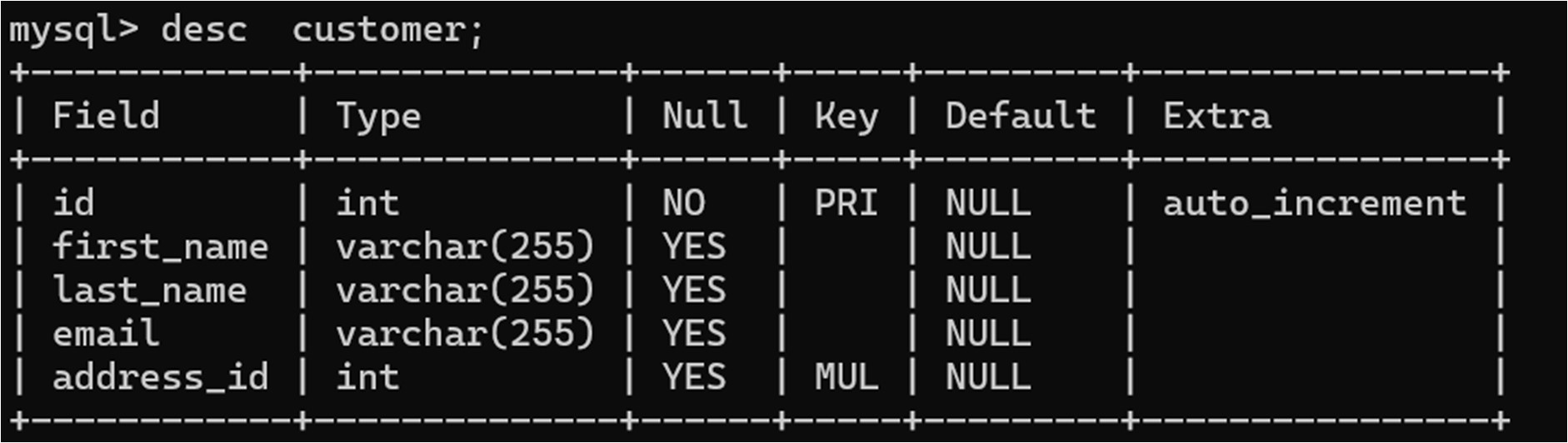
# 1.Product



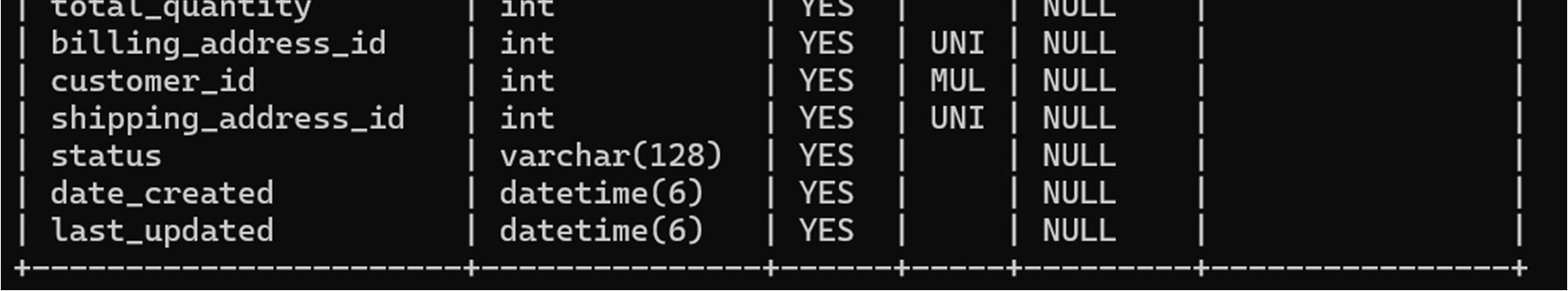
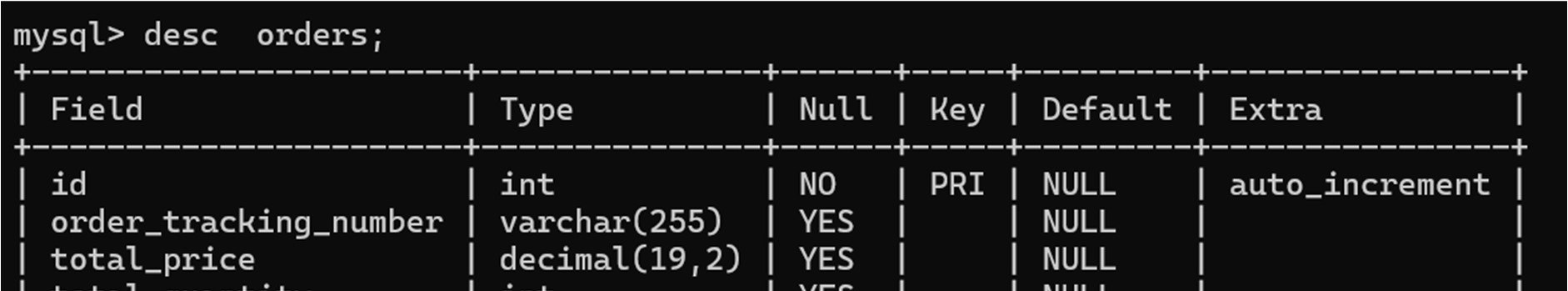
# 2.Product\_Category



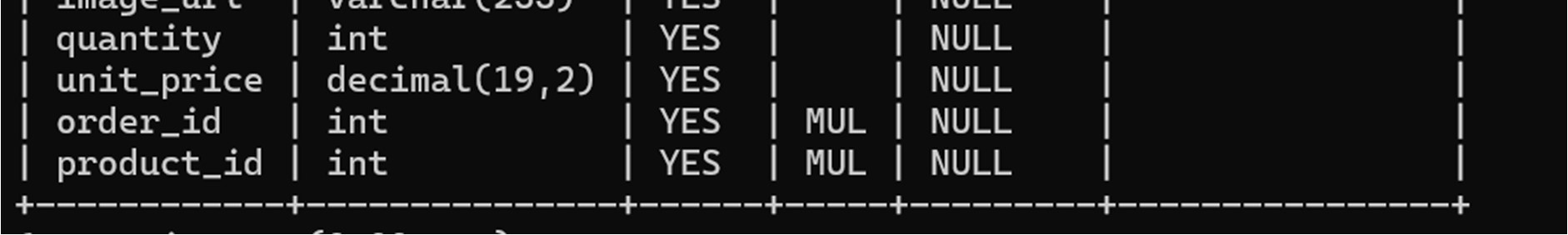
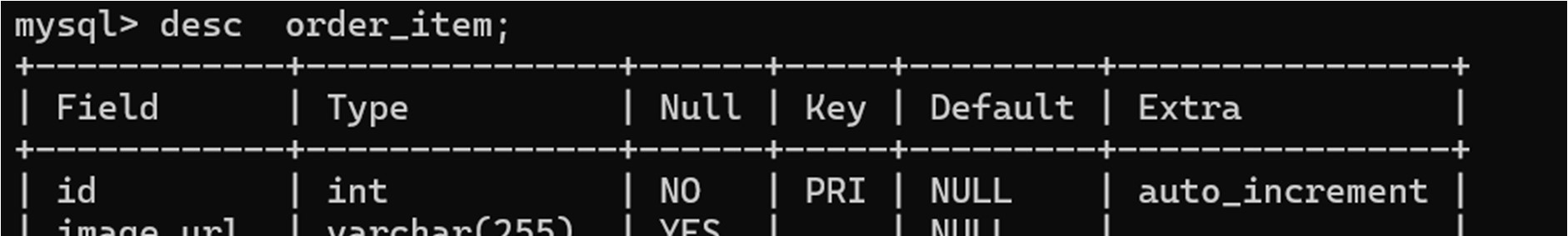
# 3.Customer



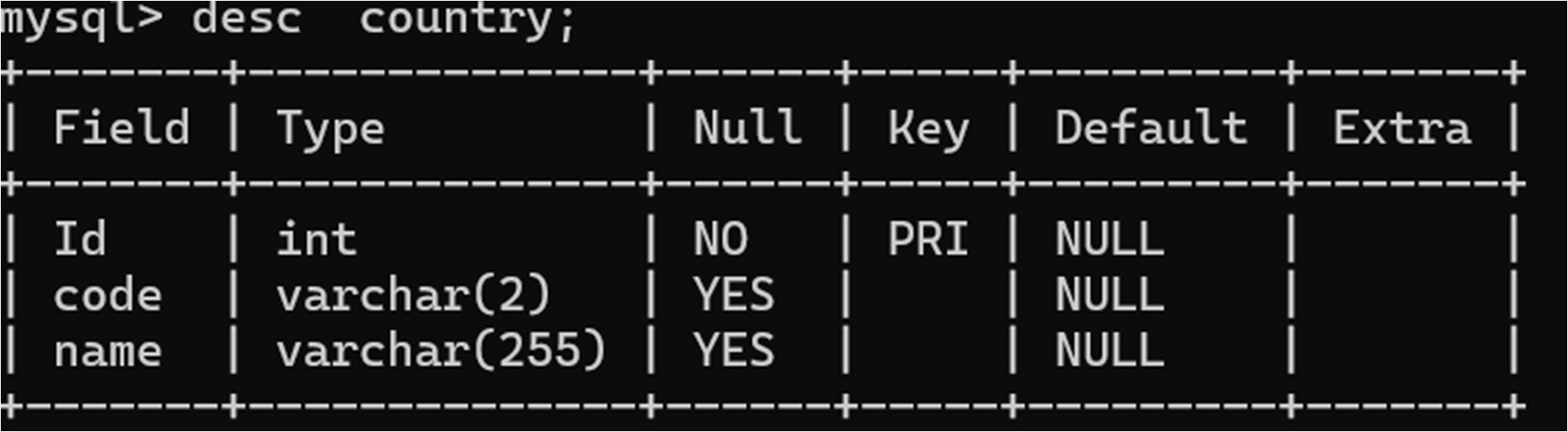
# 4.Orders



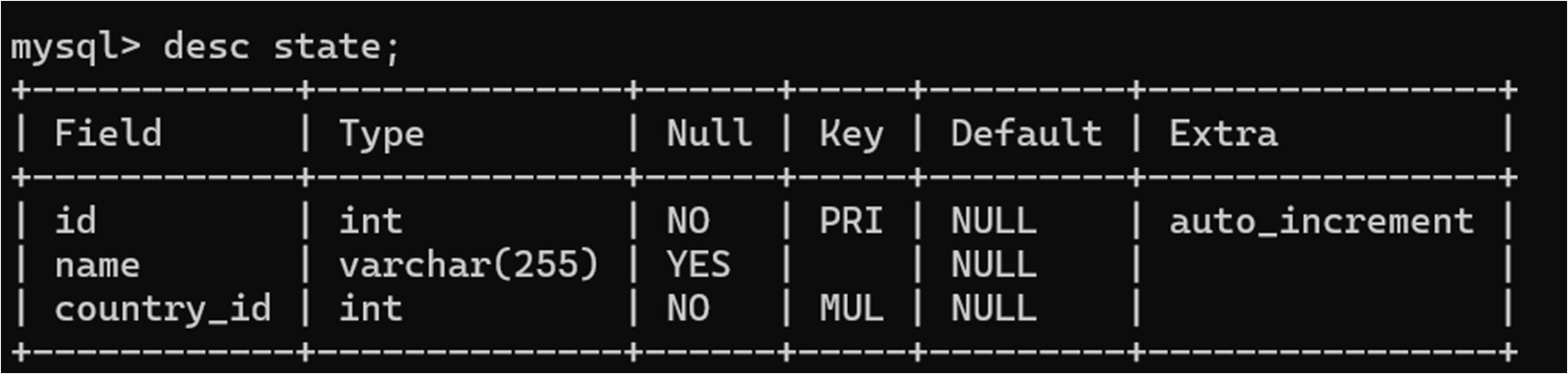
# 5.Order\_item



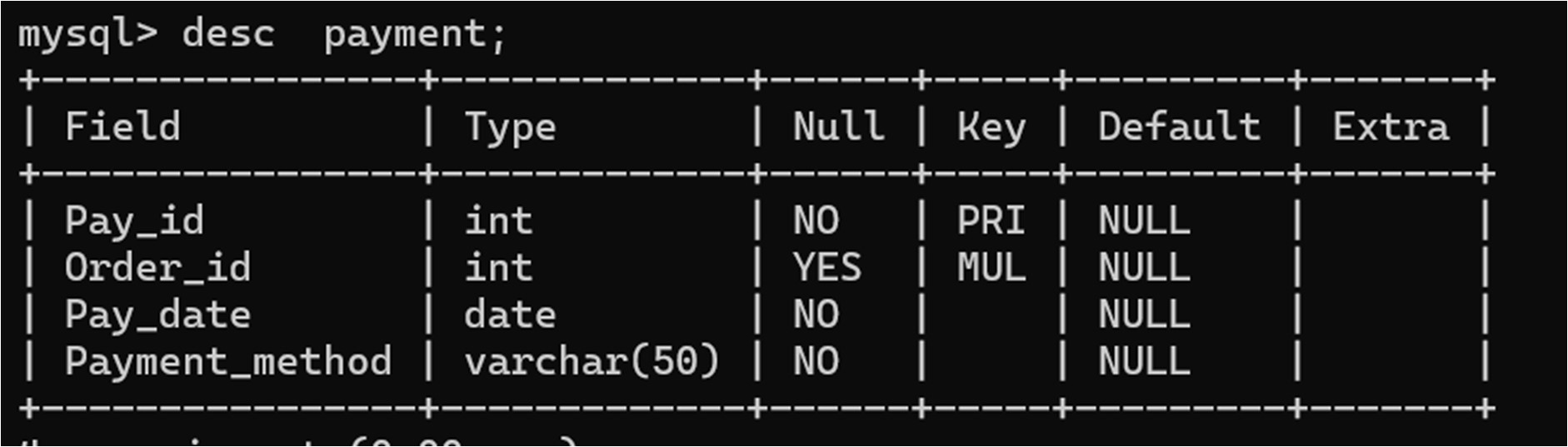
# 6.country



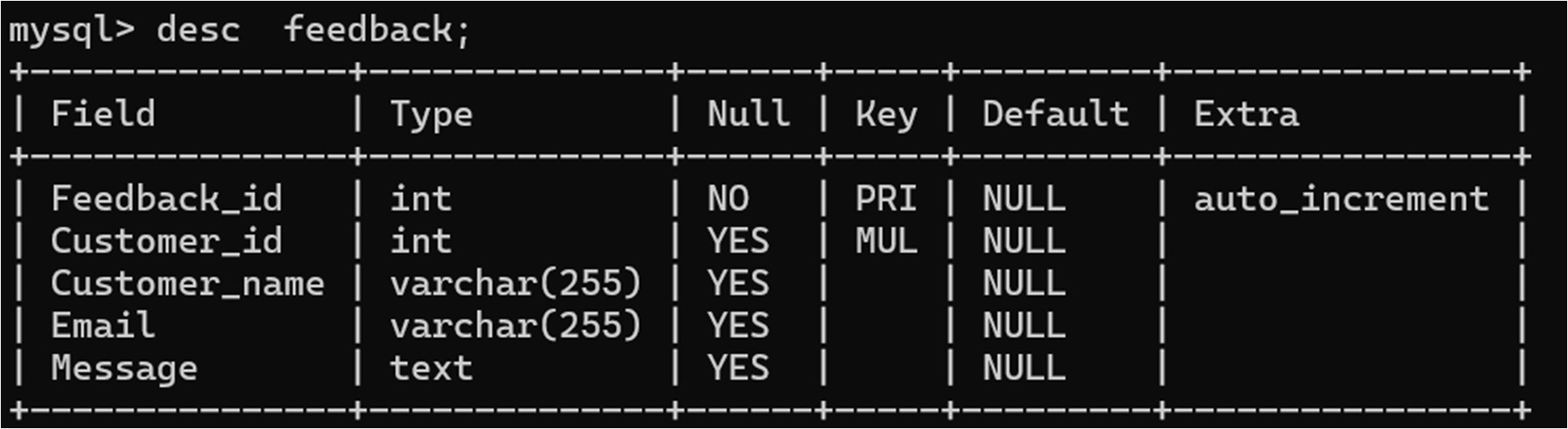
# 7.state



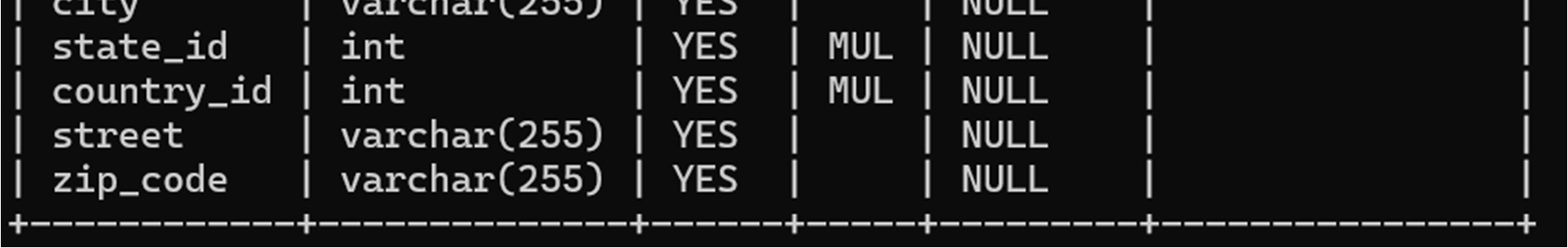
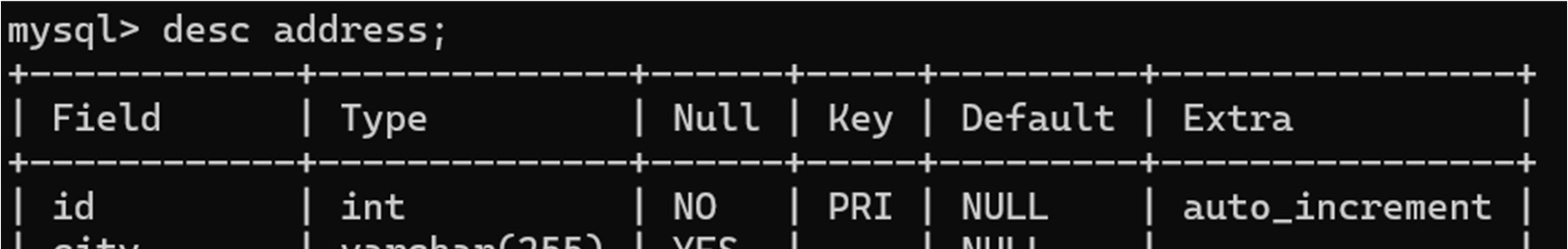
# 8.payment



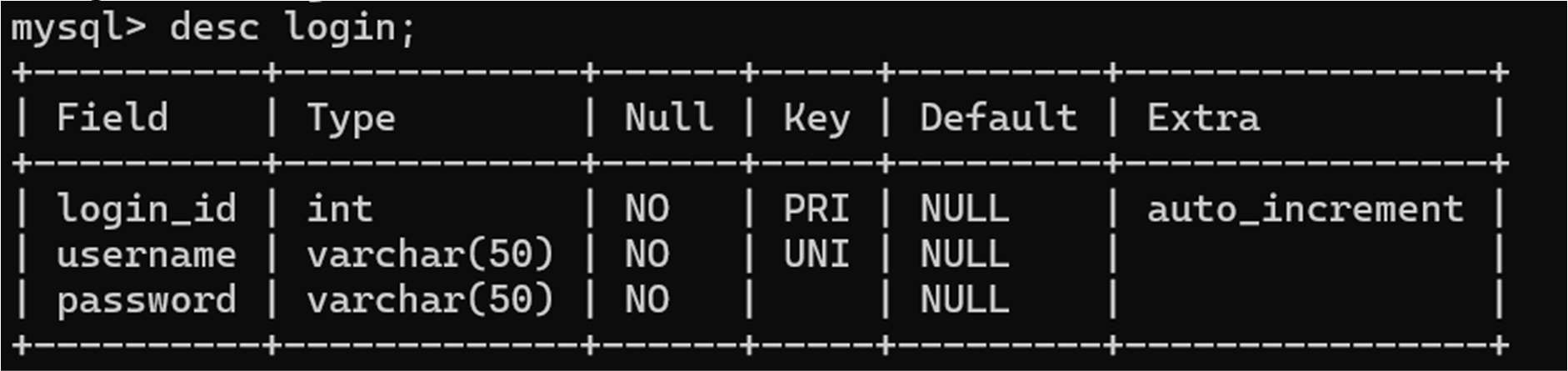
# 9.feedback



# 10. address



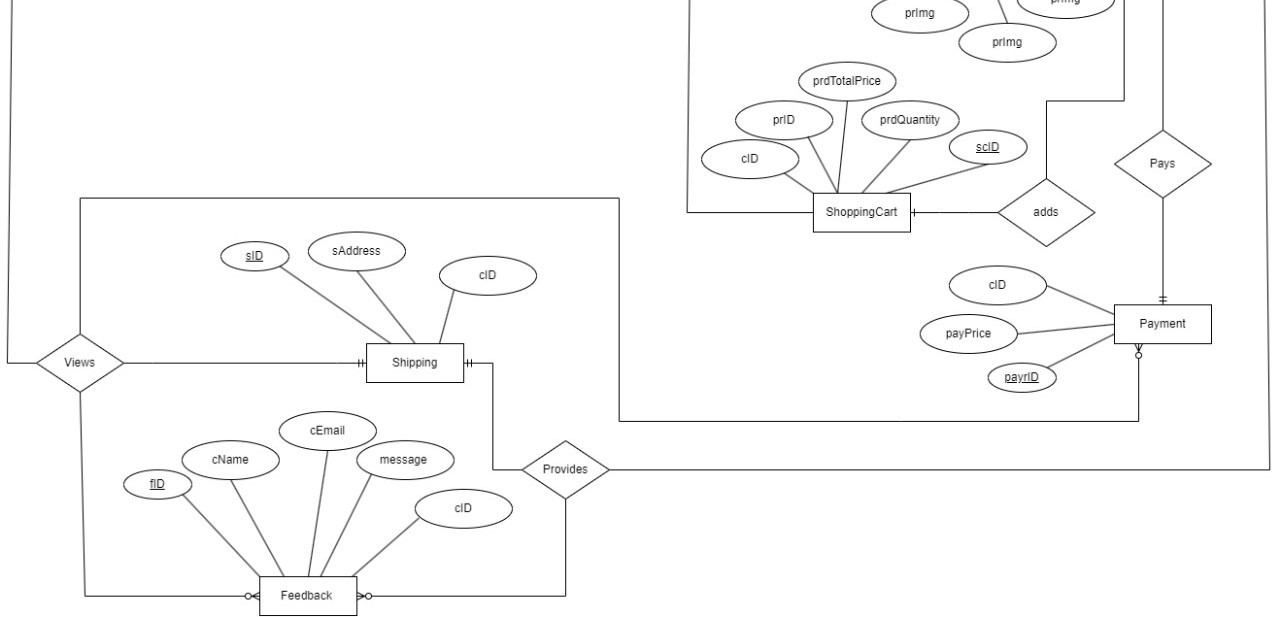
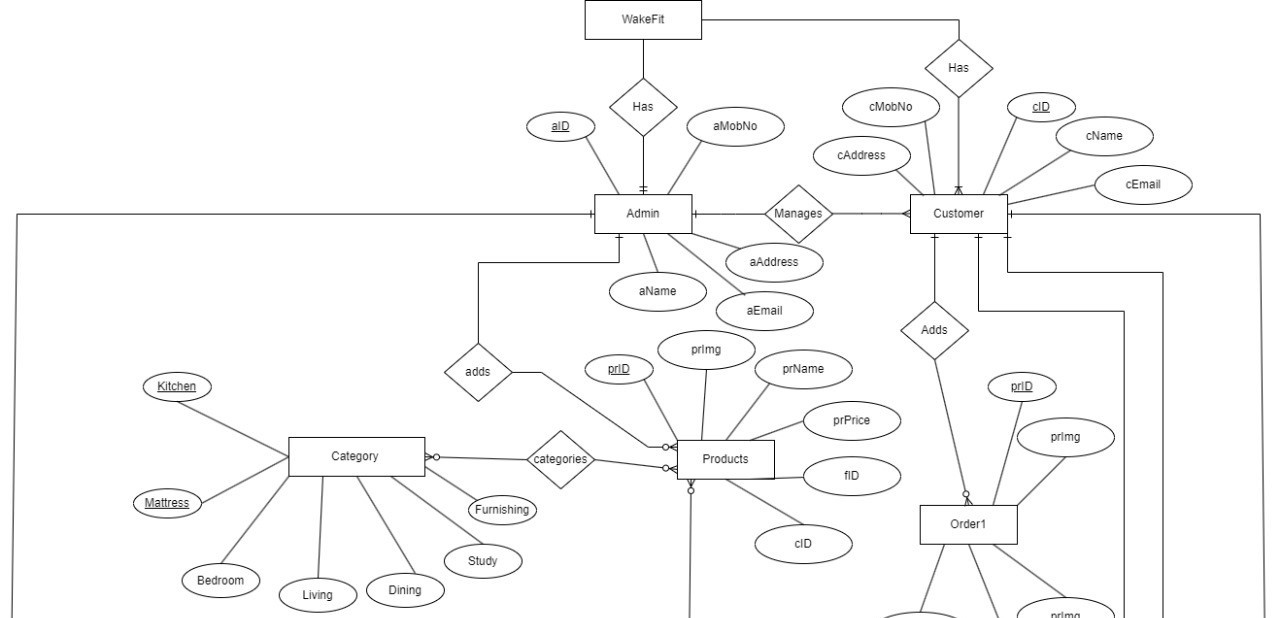
11.login



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

# ERD Diagram

In this ERD:



The Login table stores login credentials with a unique username and a corresponding password for authentication purposes.

The Customer table contains information about customers, including their names, email addresses, and foreign key reference to the address table for their location details.

The Product table holds details about various products available in the system, such as their name, description, price, availability, and category. It also includes foreign key references to the product\_category table for categorization.

Product\_Category table categorizes products into different categories, facilitating organization and navigation for users.

Orders table manages information related to orders, including order tracking numbers, total prices, quantities, and status. It also maintains foreign key references to the customer table for customer details and to the address table for billing and shipping addresses.

Order\_Item table represents individual items within orders, including details such as image URLs, quantities, and unit prices. It also includes foreign key references to the orders and product tables for order and product identification.

Country table contains information about countries, including their unique identifiers, ISO codes, and names.

State table stores details about states or regions within countries, with references to the country table to establish relationships between states and countries.

Payment table records payment details for orders, including payment IDs, associated order IDs, payment dates, and payment methods, with a foreign key reference to the orders table for order identification.

Feedback table captures feedback provided by customers, including customer IDs, names, email addresses, and feedback messages, with a foreign key reference to the customer table for customer identification.

Address table stores address information, including city, country, state, street, and zip code details, with a unique identifier for each address entry.

# 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 wakefit;

# 5. Using a Database

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

USE wakefit;

# 6. Creating the tables for each entity

USE wakefit;

1.Login CREATE TABLE Login ( login\_id INT AUTO\_INCREMENT PRIMARY KEY, username VARCHAR(50) UNIQUE NOT NULL, password VARCHAR(50) NOT NULL

);

2.Customer

CREATE TABLE customer ( id INT NOT NULL AUTO\_INCREMENT, first\_name VARCHAR(255) DEFAULT NULL, last\_name VARCHAR(255) DEFAULT NULL, email VARCHAR(255) DEFAULT NULL, address\_id INT,

PRIMARY KEY (id),

FOREIGN KEY (address\_id) REFERENCES address(id)

) AUTO\_INCREMENT=1;

# 3.Product

CREATE TABLE product ( id INT(20) NOT NULL AUTO\_INCREMENT, name VARCHAR(255) DEFAULT NULL, description VARCHAR(255) DEFAULT NULL, unit\_price DECIMAL(13,2) DEFAULT NULL, image\_url VARCHAR(255) DEFAULT NULL, active BIT DEFAULT 1, units\_in\_stock INT(11) DEFAULT NULL, date\_created DATETIME(6) DEFAULT NULL, last\_updated DATETIME(6) DEFAULT NULL, category\_id INT(20) NOT NULL,

PRIMARY KEY ( id),

KEY fk\_category ( category\_id),

CONSTRAINT fk\_category FOREIGN KEY ( category\_id) REFERENCES product\_category (id)

) AUTO\_INCREMENT = 1;

# 4.Product\_Category

CREATE TABLE product\_category ( id INT(20) NOT NULL AUTO\_INCREMENT, category\_name VARCHAR(255) NULL DEFAULT NULL,

PRIMARY KEY (id)) AUTO\_INCREMENT = 1;

# 5.Orders

CREATE TABLE orders( id int NOT NULL AUTO\_INCREMENT, order\_tracking\_number varchar(255) DEFAULT NULL, total\_price decimal(19,2) DEFAULT NULL, total\_quantity int DEFAULT NULL, billing\_address\_id int DEFAULT NULL, customer\_id int DEFAULT NULL, shipping\_address\_id int DEFAULT NULL, status varchar(128) DEFAULT NULL, date\_created datetime(6) DEFAULT NULL, last\_updated datetime(6) DEFAULT NULL,

PRIMARY KEY (id),

UNIQUE KEY UK\_billing\_address\_id (billing\_address\_id),

UNIQUE KEY UK\_shipping\_address\_id (shipping\_address\_id),

KEY K\_customer\_id (customer\_id),

CONSTRAINT FK\_customer\_id FOREIGN KEY (customer\_id) REFERENCES customer (id),

CONSTRAINT FK\_billing\_address\_id FOREIGN KEY (billing\_address\_id) REFERENCES address (id),

CONSTRAINT FK\_shipping\_address\_id FOREIGN KEY (shipping\_address\_id) REFERENCES address (id) ) AUTO\_INCREMENT=1;

# 6.Order\_item

CREATE TABLE order\_item ( id int NOT NULL AUTO\_INCREMENT, image\_url varchar(255) DEFAULT NULL, quantity int DEFAULT NULL, unit\_price decimal(19,2) DEFAULT NULL, order\_id int DEFAULT NULL, product\_id int DEFAULT NULL, PRIMARY KEY (id),

KEY K\_order\_id (order\_id),

CONSTRAINT FK\_order\_id FOREIGN KEY (order\_id) REFERENCES orders (id),

CONSTRAINT FK\_product\_id FOREIGN KEY (product\_id) REFERENCES product (id)

) AUTO\_INCREMENT=1;

# 7.country

CREATE TABLE country( Id int NOT NULL, code varchar(2) DEFAULT NULL, name varchar(255) DEFAULT NULL,

PRIMARY KEY (id)

);

# 8.state

CREATE TABLE state ( id int NOT NULL AUTO\_INCREMENT, name varchar(255) DEFAULT NULL, country\_id int NOT NULL, PRIMARY KEY (id),

KEY fk\_country (country\_id),

CONSTRAINT fk\_country FOREIGN KEY ( country\_id) REFERENCES country (id)

) AUTO\_INCREMENT=1;

# 9.payment

CREATE TABLE payment (

Pay\_id INT PRIMARY KEY,

Order\_id INT,

Pay\_date DATE NOT NULL,

Payment\_method VARCHAR(50) NOT NULL,

FOREIGN KEY (Order\_id) REFERENCES orders(OrderId)

);

# 9.feedback

CREATE TABLE feedback (

Feedback\_id INT PRIMARY KEY AUTO\_INCREMENT,

Customer\_id INT,

Customer\_name VARCHAR(255),

Email VARCHAR(255),

Message TEXT,

CONSTRAINT FK\_CustomerID FOREIGN KEY (Customer\_id) REFERENCES customer(id)

);

# 10. address

CREATE TABLE address ( id int NOT NULL AUTO\_INCREMENT, city varchar(255) DEFAULT NULL, country varchar(255) DEFAULT NULL, state varchar(255) DEFAULT NULL, street varchar(255) DEFAULT NULL, zip\_code varchar(255) DEFAULT NULL,

PRIMARY KEY ( id )

) AUTO\_INCREMENT=1;

7. Insert records

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

# -- Insert Product

INSERT INTO product (name, description, unit\_price, image\_url, active, units\_in\_stock, date\_created, last\_updated, category\_id) VALUES

('Orthopedic Memory Foam Mattress', 'Orthopedic memory foam mattress for comfortable sleep.',

10000.00, 'mattress1.jpg', 1, 100, NOW(), NOW(), 1),

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

2),

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

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

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

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

('Modular Kitchen Set', 'Modular kitchen set with cabinets and countertops.', 50000.00,

'kitchen1.jpg', 1, 10, NOW(), NOW(), 7),

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

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

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

# -- Insert Product\_Category

INSERT INTO product\_category (category\_name) VALUES

('Mattress'),

('Bedroom'),

('Living'),

('Dining'),

('Study'),

('Furnishing'),

('Kitchen'),

('Essentials'),

('Decor'),

('Kids'),

('Plus Series');

# -- Insert Customer

INSERT INTO customer (first\_name, last\_name, email, address\_id) VALUES

('Rohit', 'Musale', 'rohit@gmail.com', 1),

('Sagar', 'Kurde', 'sagar@gmail.com', 2),

('Kunal', 'Kadam', 'kk@gmail.com', 2),

('Rajan', 'Urkude', 'raju@gmail.com', 3),

('Shivraj','Sayar','ss@gmail.com',1);

# -- Insert Orders

INSERT INTO orders (order\_tracking\_number, total\_price, total\_quantity, billing\_address\_id, customer\_id, shipping\_address\_id, status, date\_created, last\_updated) VALUES

('WF1001', 15000.00, 2, 1, 1, 1, 'Processing', NOW(), NOW()),

('WF1002', 25000.00, 1, 2, 2, 2, 'Shipped', NOW(), NOW()),

('WF1003', 30000.00, 3, 3, 3, 2, 'Delivered', NOW(), NOW()),

('WF1004', 20000.00, 1, 4, 4, 3, 'Pending', NOW(), NOW()),

('WF1005', 22000.00, 1, 1, 5, 1, 'Pending', NOW(), NOW());

# -- Insert Order\_item

INSERT INTO order\_item (image\_url, quantity, unit\_price, order\_id, product\_id) VALUES

('mattress1.jpg', 2, 7500.00, 1, 1),

('bed1.jpg', 1, 20000.00, 2, 2),

('sofa1.jpg', 3, 10000.00, 3, 3),

('dining1.jpg', 1, 25000.00, 4, 4),

('chair1.jpg', 2, 5000.00, 5, 5);

# -- Insert State

INSERT INTO state (name, country\_id) VALUES

('Karnataka', 1),

('Maharashtra', 1),

('Delhi', 1),

('Tamil Nadu', 1),

('Uttar Pradesh', 1),

('Rajasthan', 1);

# -- Insert country

INSERT INTO country (Id, code, name) VALUES (1, 'IN', 'India');

# -- Insert payment

INSERT INTO payment (Pay\_id, Order\_id, Pay\_date, Payment\_method) VALUES

(1, 1, '2024-04-10', 'Credit Card'),

(2, 2, '2024-04-11', 'Debit Card'),

(3, 3, '2024-04-12', 'Net Banking'),

(4, 4, '2024-04-13', 'UPI'),

(5, 5, '2024-04-14', 'Cash');

# -- Insert feedback

INSERT INTO feedback (Customer\_id, Customer\_name, Email, Message) VALUES

(1, 'Rohit Musale', 'rohit@gmail.com', 'Excellent service and product quality.'),

(2, 'Sagar Kurde', 'sagar@gmail.com', 'Fast delivery and great customer support.'),

(3, 'Kunal Kadam', 'kk@gmail.com', 'Satisfied with the purchase. Will shop again.'),

(4, 'Rajan Urkude', 'raju@gmail.com', 'Good experience overall. Recommend to others.'), (5, 'Shivraj Sayar', 'ss@gmail.com', 'Impressive products and timely delivery.');

# -- Insert address

INSERT INTO address (city, state\_id, country\_id, street, zip\_code)

VALUES

('Mumbai', 1, 1, '123 ABC Street', '400001'),

('Pune', 2, 1, '456 XYZ Street', '411001'),

('Bangalore', 2, 1, '789 PQR Street', '560001'),

('Delhi', 3, 1, '101 GHI Street', '110001'),

('Mumbai', 1, 1, '222 JKL Street', '400002');

# -- Insert login

INSERT INTO Login (username, password) VALUES

('admin', 'admin'),

('rohit', 'rohit'),

('kunal', 'kunal');

# 8. Select records

Write SQL queries to retrieve and manage data.

For example:

Retrieve all orders:

Select \* FROM orders;

Retrieve a orders that are pending:

SELECT order\_tracking\_number, total\_price, total\_quantity, billing\_address\_id, customer\_id, shipping\_address\_id, status, date\_created, last\_updated

FROM orders

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:

UPDATE customer

SET email = 'RO@gmail.com.com'

WHERE id = 1;

# 10. Delete records

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

DELETE FROM customer

WHERE first\_name = 'Rajan' AND last\_name = 'Urkude';