Courier Management System

Task 1 - Database Design

Design a SQL schema for a Courier Management System with tables for Customers, Couriers, Orders, and Parcels. Define the relationships between these tables using appropriate foreign keys.

Requirements:

• Define the Database Schema • Create SQL tables for entities such as **User, Courier, Employee, Location,Payment**

• Define relationships between these tables **(one-to-many, many-to-many, etc.).**

• **Populate Sample Data** • Insert sample data into the tables to simulate real-world scenarios.

User Table:

User

(UserID INT PRIMARY KEY, Name VARCHAR(255),

Email VARCHAR(255) UNIQUE, Password VARCHAR(255),

ContactNumber VARCHAR(20), Address TEXT);

Courier

(CourierID INT PRIMARY KEY,

SenderName VARCHAR(255), SenderAddress TEXT,

ReceiverName VARCHAR(255),

ReceiverAddress TEXT,

Weight DECIMAL(5, 2), Status VARCHAR(50),

TrackingNumber VARCHAR(20) UNIQUE,

DeliveryDate DATE);

CourierServices

(ServiceID INT PRIMARY KEY,

ServiceName VARCHAR(100), Cost DECIMAL(8, 2));

Employee Table:

(EmployeeID INT PRIMARY KEY,

Name VARCHAR(255),

Email VARCHAR(255) UNIQUE,

ContactNumber VARCHAR(20), Role VARCHAR(50),

Salary DECIMAL(10, 2));

Location Table:

(LocationID INT PRIMARY KEY,

LocationName VARCHAR(100),

Address TEXT);

Payment Table:

(PaymentID INT PRIMARY KEY,

CourierID INT, LocationId INT,

Amount DECIMAL(10, 2), PaymentDate DATE,

FOREIGN KEY (CourierID) REFERENCES Couriers(CourierID),

FOREIGN KEY (LocationID) REFERENCES Location(LocationID));

Introduction

The Courier Management System requires a well-structured database to manage customer, courier, order, and parcel information. The database schema is designed to ensure data integrity and efficiency through proper table relationships and constraints.

Database Schema

The database consists of multiple tables, each serving a specific function in the system. Below is the schema design:

Tables and Relationships:

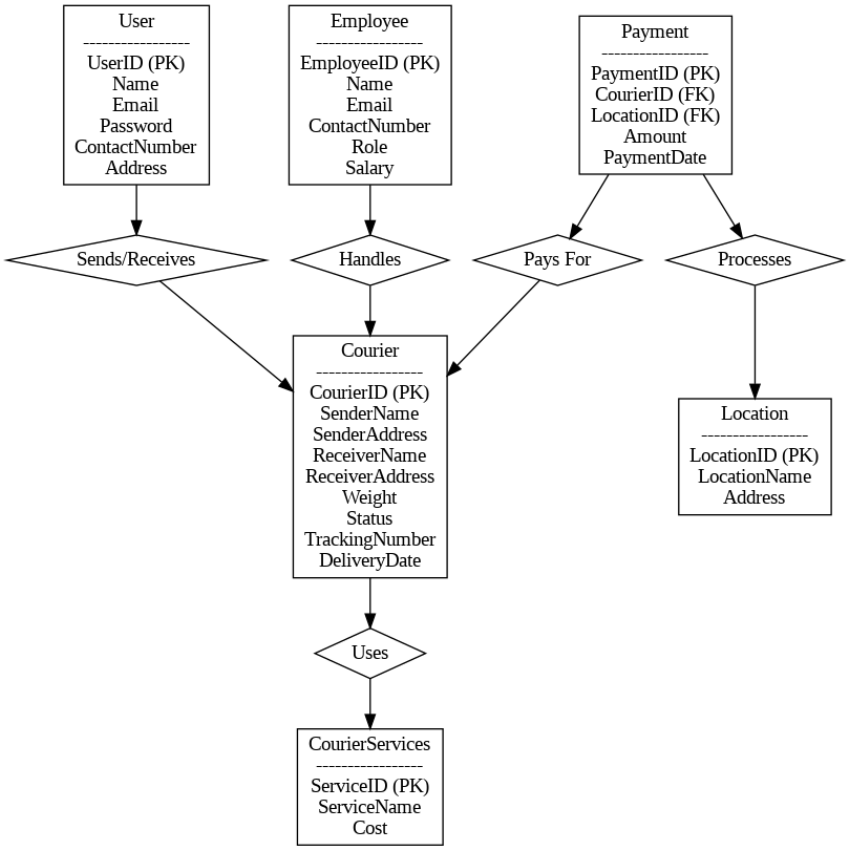
1. **User Table:**
    - Stores user details such as name, email, password, contact number, and address.
    - UserID serves as the primary key.
2. **Courier Table:**
    - Stores information about each courier, including sender and receiver details, weight, status, tracking number, and delivery date.
    - CourierID serves as the primary key.
3. **CourierServices Table:**
    - Stores the available courier services with their respective costs.
    - ServiceID serves as the primary key.
4. **Employee Table:**
    - Maintains employee details such as name, email, contact number, role, and salary.
    - EmployeeID serves as the primary key.
5. **Location Table:**
    - Stores details of different locations involved in the courier process.
    - LocationID serves as the primary key.
6. **Payment Table:**
    - Stores payment transactions related to courier services.
    - PaymentID serves as the primary key.
    - Foreign keys:
        - CourierID references Courier(CourierID)
        - LocationID references Location(LocationID)

3. Relationships Between Tables

- **One-to-Many:**
    - One user can send multiple couriers.
    - One courier service can be used for multiple couriers.
    - One employee can handle multiple couriers.
    - One location can have multiple payments associated with it.
- **Many-to-Many:**
    - If needed, a separate junction table can be created to manage many-to-many relationships (e.g., between employees and couriers).

4. ER Diagram

The Courier Management System ER Diagram represents the relationship between different entities involved in handling, processing, and tracking couriers. It ensures smooth tracking and management of courier deliveries, payments, and employee responsibilities.



- **User**: Represents individuals who send or receive packages. Attributes include UserID, Name, Email, Password, ContactNumber, and Address.
- **Employee**: Employees are responsible for managing courier operations. Attributes include EmployeeID, Name, Email, ContactNumber, Role, and Salary.
- **Courier**: This entity tracks the details of shipments. Attributes include CourierID, SenderName, SenderAddress, ReceiverName, ReceiverAddress, Weight, Status, TrackingNumber, and DeliveryDate.

- **CourierServices**: Defines different courier services available with respective costs. Attributes include ServiceID, ServiceName, and Cost.
- **Payment**: Manages financial transactions for courier services. Attributes include PaymentID, CourierID (FK), LocationID (FK), Amount, and PaymentDate.
- **Location**: Represents processing centers or delivery hubs. Attributes include LocationID, LocationName, and Address.

5. Sample Data Insertion

- Each table was populated with at least 10 sample rows to facilitate testing and ensure meaningful data relationships.
- SQL queries were executed to verify data integrity and successful foreign key constraints.

.

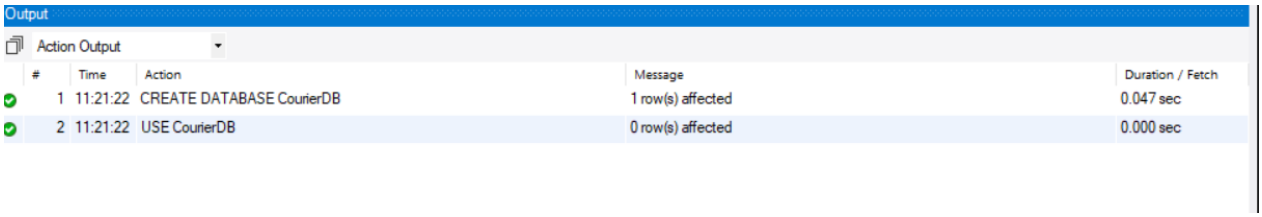
Steps :

Step 1: Create the Database

Open MySQL Workbench and connect to your database server.

CREATE DATABASE CourierDB;

USE CourierDB;



Step 2: Creating the Tables.

Run the following SQL queries to create the necessary tables in **CourierDB**

-- User Table

CREATE TABLE User (

UserID INT PRIMARY KEY AUTO_INCREMENT,

Name VARCHAR(255) NOT NULL,

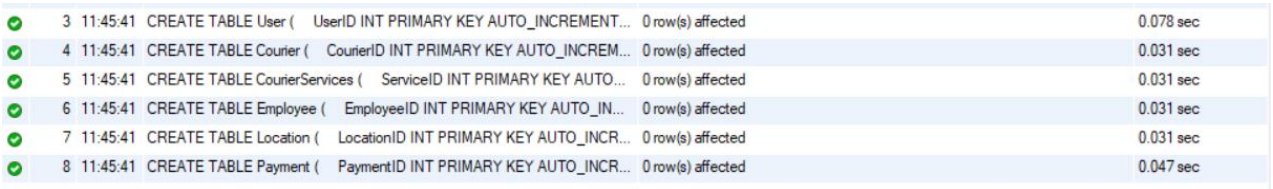
Email VARCHAR(255) UNIQUE NOT NULL,

Password VARCHAR(255) NOT NULL,

ContactNumber VARCHAR(20),

```
    Address TEXT
);

-- Courier Table
CREATE TABLE Courier (
    CourierID INT PRIMARY KEY AUTO_INCREMENT,
    SenderName VARCHAR(255) NOT NULL,
    SenderAddress TEXT NOT NULL,
    ReceiverName VARCHAR(255) NOT NULL,
    ReceiverAddress TEXT NOT NULL,
    Weight DECIMAL(5,2) NOT NULL,
    Status VARCHAR(50) NOT NULL,
    TrackingNumber VARCHAR(20) UNIQUE NOT NULL,
    DeliveryDate DATE
);

-- Courier Services Table
CREATE TABLE CourierServices (
    ServiceID INT PRIMARY KEY AUTO_INCREMENT,
    ServiceName VARCHAR(100) NOT NULL,
    Cost DECIMAL(8,2) NOT NULL
);

-- Employee Table
CREATE TABLE Employee (
    EmployeeID INT PRIMARY KEY AUTO_INCREMENT,
    Name VARCHAR(255) NOT NULL,
    Email VARCHAR(255) UNIQUE NOT NULL,
    ContactNumber VARCHAR(20) NOT NULL,
    Role VARCHAR(50) NOT NULL,
```

```
    Salary DECIMAL(10,2) NOT NULL
);

-- Location Table
CREATE TABLE Location (
    LocationID INT PRIMARY KEY AUTO_INCREMENT,
    LocationName VARCHAR(100) NOT NULL,
    Address TEXT NOT NULL
);

-- Payment Table
CREATE TABLE Payment (
    PaymentID INT PRIMARY KEY AUTO_INCREMENT,
    CourierID INT,
    LocationID INT,
    Amount DECIMAL(10,2) NOT NULL,
    PaymentDate DATE NOT NULL,
    FOREIGN KEY (CourierID) REFERENCES Courier(CourierID),
    FOREIGN KEY (LocationID) REFERENCES Location(LocationID)
);
```



Step 3: Insert Sample Data

Now, let's populate the tables with sample data. Run the following SQL queries:

```
-- Insert data into User table
INSERT INTO User (Name, Email, Password, ContactNumber, Address)
VALUES
('John Doe', 'john@example.com', 'password123', '9876543210', '123 Main St'),
```

('Alice Smith', 'alice@example.com', 'alicepass', '8765432109', '456 Elm St');

-- Insert data into Courier table

INSERT INTO Courier (SenderName, SenderAddress, ReceiverName, ReceiverAddress, Weight, Status, TrackingNumber, DeliveryDate)

VALUES

('John Doe', '123 Main St', 'Alice Smith', '456 Elm St', 2.5, 'In Transit', 'TRK12345', '2024-03-20'),

('Alice Smith', '456 Elm St', 'John Doe', '123 Main St', 1.2, 'Delivered', 'TRK54321', '2024-03-18');

-- Insert data into CourierServices table

INSERT INTO CourierServices (ServiceName, Cost)

VALUES

('Standard Delivery', 100.00),

('Express Delivery', 250.00);

-- Insert data into Employee table

INSERT INTO Employee (Name, Email, ContactNumber, Role, Salary)

VALUES

('David Wilson', 'david@example.com', '9876123456', 'Delivery Agent', 25000.00),

('Emma Brown', 'emma@example.com', '8765123498', 'Manager', 50000.00);

-- Insert data into Location table

INSERT INTO Location (LocationName, Address)

VALUES

('Warehouse A', '789 Industrial Road'),

('Warehouse B', '321 Commercial St');

-- Insert data into Payment table

INSERT INTO Payment (CourierID, LocationID, Amount, PaymentDate)

VALUES

(1, 1, 100.00, '2024-03-21'),

(2, 2, 250.00, '2024-03-19');

9	12:02:36	INSERT INTO User (Name, Email, Password, ContactNumber, Address) VA...	10 row(s) affected Records: 10	Duplicates: 0	Warnings: 0	0.031 sec
10	12:02:50	INSERT INTO Courier (SenderName, SenderAddress, ReceiverName, Rece...	10 row(s) affected Records: 10	Duplicates: 0	Warnings: 0	0.015 sec
11	12:02:50	INSERT INTO CourierServices (ServiceName, Cost) VALUES ('Standard D...	10 row(s) affected Records: 10	Duplicates: 0	Warnings: 0	0.016 sec
12	12:02:58	INSERT INTO Employee (Name, Email, ContactNumber, Role, Salary) VAL...	10 row(s) affected Records: 10	Duplicates: 0	Warnings: 0	0.015 sec
13	12:02:58	INSERT INTO Location (LocationName, Address) VALUES ('Warehouse A...	10 row(s) affected Records: 10	Duplicates: 0	Warnings: 0	0.000 sec
14	12:02:58	INSERT INTO Payment (CourierID, LocationID, Amount, PaymentDate) VA...	10 row(s) affected Records: 10	Duplicates: 0	Warnings: 0	0.015 sec

---

Verification

1. **Table Creation** - Screenshot of SHOW TABLES;

```
247
248 •  SHOW TABLES;
249
250
251
252
```

Result Grid | Filter Rows: | Export:

Tables_in_courierdb
courier
courierservices
employee
location
payment
user

2. **Data Insertion**

```
229
230    -- View all records from the User table
231 •  SELECT * FROM User;
232
233    -- View all records from the Courier table
234 •  SELECT * FROM Courier;
235
236    -- View all records from the CourierServices table
237 •  SELECT * FROM CourierServices;
```

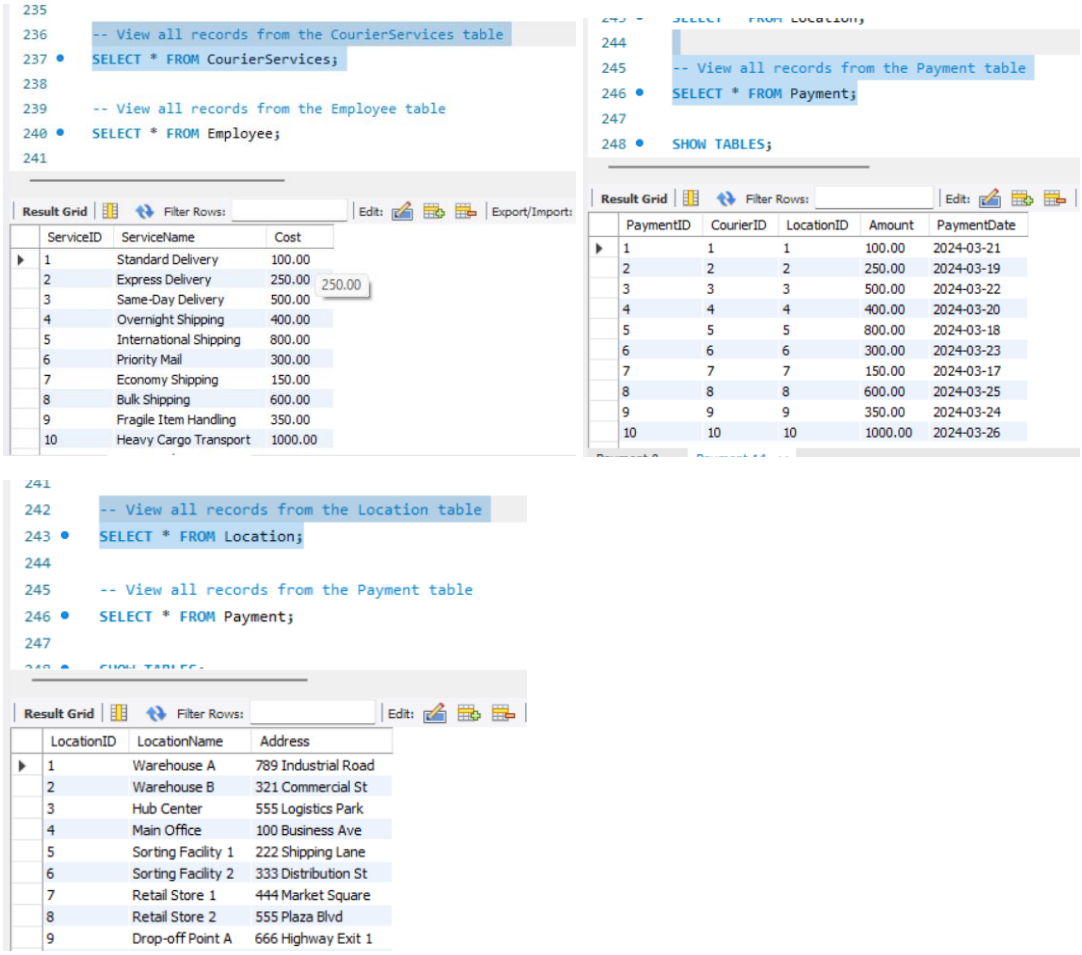
Result Grid | Filter Rows: | Edit: | Export/Import: | Wrap Cell Content:

UserID	Name	Email	Password	ContactNumber	Address
1	John Doe	john@example.com	password123	9876543210	123 Main St
2	Alice Smith	alice@example.com	alicepass	8765432109	456 Elm St
3	Michael Brown	michael@example.com	mikepass	7654321098	789 Oak St
4	Emma Wilson	emma@example.com	emmapass	6543210987	321 Pine St
5	Daniel Lee	daniel@example.com	danpass	5432109876	654 Cedar St
6	Sophia Green	sophia@example.com	sophiapass	4321098765	987 Maple St
7	William Clark	william@example.com	willpass	3210987654	258 Birch St
8	Olivia Adams	olivia@example.com	oliviapass	2109876543	369 Walnut St
9	James Turner	james@example.com	jamespass	1098765432	741 Cherry St
10	Charlotte Evans	charlotte@example.com	charpass	1987654321	852 Willow St

```
235
236    -- View all records from the CourierServices table
237 •  SELECT * FROM CourierServices;
238
239    -- View all records from the Employee table
```

Result Grid | Filter Rows: | Edit: | Export

ServiceID	ServiceName	Cost
1	Standard Delivery	100.00
2	Express Delivery	250.00
3	Same-Day Delivery	500.00
4	Overnight Shipping	400.00
5	International Shipping	800.00
6	Priority Mail	300.00
7	Economy Shipping	150.00
8	Bulk Shipping	600.00
9	Fragile Item Handling	350.00
10	Heavy Cargo Transport	1000.00



```
235
236     -- View all records from the CourierServices table
237 •   SELECT * FROM CourierServices;
238
239     -- View all records from the Employee table
240 •   SELECT * FROM Employee;
241
```

Result Grid | Filter Rows: | Edit: | Export/Import:

ServiceID	ServiceName	Cost
1	Standard Delivery	100.00
2	Express Delivery	250.00
3	Same-Day Delivery	500.00
4	Overnight Shipping	400.00
5	International Shipping	800.00
6	Priority Mail	300.00
7	Economy Shipping	150.00
8	Bulk Shipping	600.00
9	Fragile Item Handling	350.00
10	Heavy Cargo Transport	1000.00

250.00

```
244
245     -- View all records from the Payment table
246 •   SELECT * FROM Payment;
247
248 •   SHOW TABLES;
```

Result Grid | Filter Rows: | Edit:

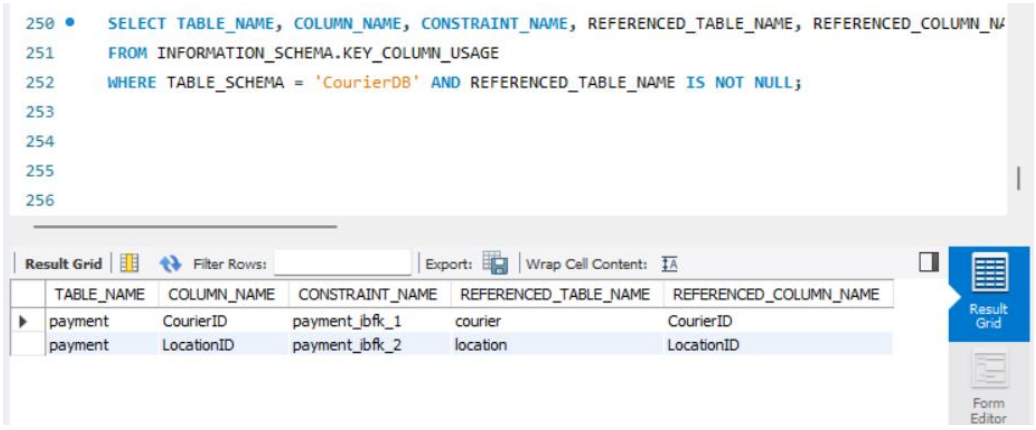
PaymentID	CourierID	LocationID	Amount	PaymentDate
1	1	1	100.00	2024-03-21
2	2	2	250.00	2024-03-19
3	3	3	500.00	2024-03-22
4	4	4	400.00	2024-03-20
5	5	5	800.00	2024-03-18
6	6	6	300.00	2024-03-23
7	7	7	150.00	2024-03-17
8	8	8	600.00	2024-03-25
9	9	9	350.00	2024-03-24
10	10	10	1000.00	2024-03-26

```
241
242     -- View all records from the Location table
243 •   SELECT * FROM Location;
244
245     -- View all records from the Payment table
246 •   SELECT * FROM Payment;
247
```

Result Grid | Filter Rows: | Edit:

LocationID	LocationName	Address
1	Warehouse A	789 Industrial Road
2	Warehouse B	321 Commercial St
3	Hub Center	555 Logistics Park
4	Main Office	100 Business Ave
5	Sorting Facility 1	222 Shipping Lane
6	Sorting Facility 2	333 Distribution St
7	Retail Store 1	444 Market Square
8	Retail Store 2	555 Plaza Blvd
9	Drop-off Point A	666 Highway Exit 1

3. **Relationships** - Checking foreign keys (screenshot attached)



```
250 •   SELECT TABLE_NAME, COLUMN_NAME, CONSTRAINT_NAME, REFERENCED_TABLE_NAME, REFERENCED_COLUMN_NA
251     FROM INFORMATION_SCHEMA.KEY_COLUMN_USAGE
252     WHERE TABLE_SCHEMA = 'CourierDB' AND REFERENCED_TABLE_NAME IS NOT NULL;
253
254
255
256
```

Result Grid | Filter Rows: | Export: | Wrap Cell Content:

TABLE_NAME	COLUMN_NAME	CONSTRAINT_NAME	REFERENCED_TABLE_NAME	REFERENCED_COLUMN_NAME
payment	CourierID	payment_ibfk_1	courier	CourierID
payment	LocationID	payment_ibfk_2	location	LocationID

Result Grid | Form Editor

Conclusion

The database schema for the Courier Management System has been successfully designed, implemented, and validated. The structure ensures efficient data management, maintains integrity through foreign keys, and allows smooth operations for future enhancements.