-- Library Management System

-- It keeps track of all information about books in the library, their cost, status and total number of books available in the library.

-- Create a database named library

```
CREATE DATABASE library;
USE library;
```

-- create tables

```sql
CREATE TABLE Branch (
    Branch_no INT PRIMARY KEY,
    Manager_Id INT,
    Branch_address VARCHAR(255),
    Contact_no VARCHAR(15)
);

CREATE TABLE Employee (
    Emp_Id INT PRIMARY KEY,
    Emp_name VARCHAR(255),
    Position VARCHAR(100),
    Salary DECIMAL(10, 2),
    Branch_no INT,
    FOREIGN KEY (Branch_no) REFERENCES Branch(Branch_no)
);

CREATE TABLE Books (
    ISBN VARCHAR(20) PRIMARY KEY,
    Book_title VARCHAR(255),
    Category VARCHAR(100),
    Rental_Price DECIMAL(10, 2),
    Status ENUM('yes', 'no'),
    Author VARCHAR(255),
    Publisher VARCHAR(255)
);
```

```sql
CREATE TABLE Books (
    ISBN VARCHAR(20) PRIMARY KEY,
    Book_title VARCHAR(255),
    Category VARCHAR(100),
    Rental_Price DECIMAL(10, 2),
    Status ENUM('yes', 'no'),
    Author VARCHAR(255),
    Publisher VARCHAR(255)
);

CREATE TABLE Customer (
    Customer_Id INT PRIMARY KEY,
    Customer_name VARCHAR(255),
    Customer_address VARCHAR(255),
    Reg_date DATE
);

CREATE TABLE IssueStatus (
    Issue_Id INT PRIMARY KEY,
    Issued_cust INT,
    Issued_book_name VARCHAR(255),
    Issue_date DATE,
    ISBN_book VARCHAR(20),
    FOREIGN KEY (Issued_cust) REFERENCES Customer(Customer_Id),
    FOREIGN KEY (ISBN_book) REFERENCES Books(ISBN)
);
```

-- Insert values into tables

```sql
-- Insert values into Branch

INSERT INTO Branch (Branch_no, Manager_Id, Branch_address, Contact_no) VALUES
(1, 101, '123 Main St, Cityville', '555-1234'),
(2, 102, '456 Elm St, Townsville', '555-5678');

-- Insert values into Employee

INSERT INTO Employee (Emp_Id, Emp_name, Position, Salary, Branch_no) VALUES
(201, 'Alice Smith', 'Manager', 60000, 1),
(202, 'Bob Johnson', 'Librarian', 45000, 1),
(203, 'Charlie Brown', 'Assistant', 30000, 2),
(204, 'Diana Prince', 'Manager', 70000, 2);

-- Insert values into Books

INSERT INTO Books (ISBN, Book_title, Category, Rental_Price, Status, Author, Publisher) VALUES
('978-3-16-148410-0', 'The Great Gatsby', 'Fiction', 20.00, 'yes', 'F. Scott Fitzgerald', 'Scribner'),
('978-0-7432-7356-5', 'A Brief History of Time', 'Science', 30.00, 'yes', 'Stephen Hawking', 'Bantam'),
('978-1-4028-9462-6', '1984', 'Dystopian', 15.00, 'no', 'George Orwell', 'Secker & Warburg'),
('978-0-06-112008-4', 'To Kill a Mockingbird', 'Fiction', 25.00, 'yes', 'Harper Lee', 'J.B. Lippincott & Co.');


-- Insert values into Customer

INSERT INTO Customer (Customer_Id, Customer_name, Customer_address, Reg_date) VALUES
(301, 'John Doe', '789 Oak St, Cityville', '2021-05-10'),
(302, 'Jane Roe', '321 Pine St, Townsville', '2020-12-15'),
(303, 'Emily Clark', '654 Maple St, Cityville', '2023-01-20');

-- Insert values into IssueStatus

INSERT INTO IssueStatus (Issue_Id, Issued_cust, Issued_book_name, Issue_date, ISBN_book) VALUES
(401, 301, 'The Great Gatsby', '2023-06-15', '978-3-16-148410-0'),
(402, 302, 'A Brief History of Time', '2023-06-20', '978-0-7432-7356-5');

-- Insert values into ReturnStatus

INSERT INTO ReturnStatus (Return_Id, Return_cust, Return_book_name, Return_date, ISBN_book2) VALUES
(501, 301, 'The Great Gatsby', '2023-07-01', '978-3-16-148410-0'),
(502, 302, 'A Brief History of Time', '2023-07-05', '978-0-7432-7356-5');
```

-- 1. Retrieve the book title, category, and rental price of all available books.

```sql
SELECT Book_title, Category, Rental_Price
FROM Books
WHERE Status = 'yes';
```

-- 2. List the employee names and their respective salaries in descending order of salary.

```sql
SELECT Emp_name, Salary
FROM Employee
ORDER BY Salary DESC;
```

-- 3. Retrieve the book titles and the corresponding customers who have issued those books.

```sql
SELECT B.Book_title, C.Customer_name
FROM IssueStatus I
JOIN Books B ON I.ISBN_book = B.ISBN
JOIN Customer C ON I.Issued_cust = C.Customer_Id;
```

-- 4. Display the total count of books in each category.

```sql
SELECT Category, COUNT(*) AS Total_Books
FROM Books
GROUP BY Category;
```

-- 5. Retrieve the employee names and their positions for the employees whose salaries are above Rs.50,000.

```sql
SELECT Emp_name, Position
FROM Employee
WHERE Salary > 50000;
```

-- 6. List the customer names who registered before 2022-01-01 and have not issued any books yet.

```sql
SELECT C.Customer_name
FROM Customer C
LEFT JOIN IssueStatus I ON C.Customer_Id = I.Issued_cust
WHERE C.Reg_date < '2022-01-01' AND I.Issued_cust IS NULL;
```

-- 7. Display the branch numbers and the total count of employees in each branch.

```sql
SELECT Branch_no, COUNT(*) AS Total_Employees
FROM Employee
GROUP BY Branch_no;
```

-- 8. Display the names of customers who have issued books in the month of June 2023.

```sql
SELECT DISTINCT C.Customer_name
FROM Customer C
JOIN IssueStatus I ON C.Customer_Id = I.Issued_cust
WHERE I.Issue_date BETWEEN '2023-06-01' AND '2023-06-30';
```

-- 9. Retrieve book_title from book table containing history.

```sql
SELECT Book_title
FROM Books
WHERE Book_title LIKE '%history%';
```

-- 10.Retrieve the branch numbers along with the count of employees for branches having more than 5 employees

```sql
SELECT Branch_no, COUNT(*) AS Total_Employees
FROM Employee
GROUP BY Branch_no
HAVING COUNT(*) > 5;
```

-- 11. Retrieve the names of employees who manage branches and their respective branch addresses.

```sql
SELECT E.Emp_name, B.Branch_address
FROM Employee E
JOIN Branch B ON E.Branch_no = B.Branch_no
WHERE E.Position = 'Manager';
```

-- 12. Display the names of customers who have issued books with a rental price higher than Rs. 25.

```sql
SELECT DISTINCT C.Customer_name
FROM Customer C
JOIN IssueStatus I ON C.Customer_Id = I.Issued_cust
JOIN Books B ON I.ISBN_book = B.ISBN
WHERE B.Rental_Price > 25;
```