DESIGN AND IMPLEMENTATION OF A LIBRARY MANAGEMENT SYSTEM DATABASE

SUBMITTED BY

NIKHIL SANKAR

X Nikhil

30/08/2024

Table of Contents

1. INTRODUCTION

Library management systems are essential tools for efficiently managing the vast array of resources and services offered by libraries. These systems streamline the processes of cataloging, tracking, and lending books, allowing librarians to focus on service rather than administrative tasks. A well-designed database is crucial in this context, as it organizes and manages data related to books, members, and transactions.

The objectives of this report are to design and implement a robust database for a library management system, ensuring it meets the functional requirements of managing book inventories, member information, and lending processes effectively.

2. SYSTEM DESIGN

2.1. **REQUIREMENTS**

The library management system aims to efficiently manage library operations, focusing on the management of books, branches, employees, and customer interactions, including the issue and return of books. Below are the key requirements:

2.1.1. FUNCTIONAL REQUIREMENTS:

- **Branch Management:** The system must manage multiple library branches, including details like branch number, manager, address, and contact information.
- **Employee Management:** The system should track employee information, including their position, salary, and the branch they are associated with.
- **Book Management:** The system should maintain a catalog of books, including details like ISBN, title, category, rental price, availability status, author, and publisher.

- **Customer Management:** The system should manage customer information, including their registration details, address, and the date they registered.
- **Issue and Return Management:** The system should track the issue and return of books by customers, linking them to specific books and recording dates of transactions.

2.1.2. NON-FUNCTIONAL REQUIREMENTS:

- **Performance:** The system should efficiently handle transactions such as issuing and returning books, even as the number of records increases.
- **Scalability:** The database design should allow easy expansion, such as adding more branches, employees, or books.
- **Security:** Customer and employee information must be protected, with restricted access to sensitive data.
- **Usability:** The system interface should be straightforward, enabling users to perform tasks with minimal effort.

2.2. DATABASE DESIGN

The database design for this library management system revolves around six key tables: *Branch*, *Employee*, *Books*, *Customer*, *IssueStatus*, and *ReturnStatus*. The relationships between these tables are essential for maintaining data integrity and supporting the system's functionality.

2.2.1. Tables and Relationships

1. **Branch Table:**
    - Branch_no (Primary Key): Unique identifier for each branch.
    - Manager_Id: ID of the manager in charge of the branch.
    - Branch_address: Address of the branch.
    - Contact_no: Contact number of the branch.

2. **Employee Table:**
    - Emp_Id (Primary Key): Unique identifier for each employee.
    - Emp_name: Name of the employee.
    - Position: Position or job title of the employee.
    - Salary: Salary of the employee.
    - Branch_no (Foreign Key): References Branch_no in the *Branch* table, indicating which branch the employee is associated with.
3. **Books Table:**
    - ISBN (Primary Key): Unique identifier for each book.
    - Book_title: Title of the book.
    - Category: Genre or category of the book (e.g., Fiction, Non-fiction).
    - Rental_Price: Price to rent the book.
    - Status: Availability status of the book (Yes if available, No if not).
    - Author: Author of the book.
    - Publisher: Publisher of the book.
4. **Customer Table:**
    - Customer_Id (Primary Key): Unique identifier for each customer.
    - Customer_name: Name of the customer.
    - Customer_address: Address of the customer.
    - Reg_date: Date the customer registered with the library.

5. **IssueStatus Table:**
    - Issue_Id (Primary Key): Unique identifier for each book issue transaction.
    - Issued_cust (Foreign Key): References Customer_Id in the *Customer* table, indicating which customer borrowed the book.
    - Issued_book_name: Name of the book issued.
    - Issue_date: Date the book was issued.
    - Isbn_book (Foreign Key): References ISBN in the *Books* table, indicating which book was issued.
6. **ReturnStatus Table:**
    - Return_Id (Primary Key): Unique identifier for each book return transaction.
    - Return_cust: References the customer who returned the book.
    - Return_book_name: Name of the book returned.
    - Return_date: Date the book was returned.
    - Isbn_book2 (Foreign Key): References ISBN in the *Books* table, indicating which book was returned.

2.2.2. Data Types

- Primary keys like Branch_no, Emp_Id, ISBN, Customer_Id, Issue_Id, and Return_Id are generally integers or strings, depending on the desired format.
- Text fields such as Branch_address, Emp_name, Book_title, Customer_name, and Author use VARCHAR for variable-length character storage.
- Dates like Reg_date, Issue_date, and Return_date use the DATE data type.

This design ensures that all interactions within the library system are accurately recorded and managed, providing a robust framework for library operations.

3. IMPLEMENTATION

3.1. TOOLS AND TECHNOLOGIES

For the implementation of the library management system database, the following tools and technologies are utilized:

3.1.1. SOFTWARE

MySQL Workbench 8.0 CE: This integrated development environment (IDE) is employed for designing, modelling, and managing the MySQL database. It provides essential tools for SQL query execution, database design, and data visualization, which are crucial for developing and demonstrating the library management system.

3.1.2. ADDITIONAL NOTES

MySQL Workbench 8.0 CE facilitates the creation and management of the database schema, execution of SQL queries, and analysis of results, providing a practical platform for showcasing SQL expertise.

3.2. SQL QUERIES

3.2.1. SQL SCRIPTS TO CREATE TABLES

1. BRANCH

```
CREATE TABLE BRANCH (
BRANCH_NO INT PRIMARY KEY,
MANAGER_ID INT,
BRANCH_ADDRESS VARCHAR (30),
CONTACT_NO INT);
```

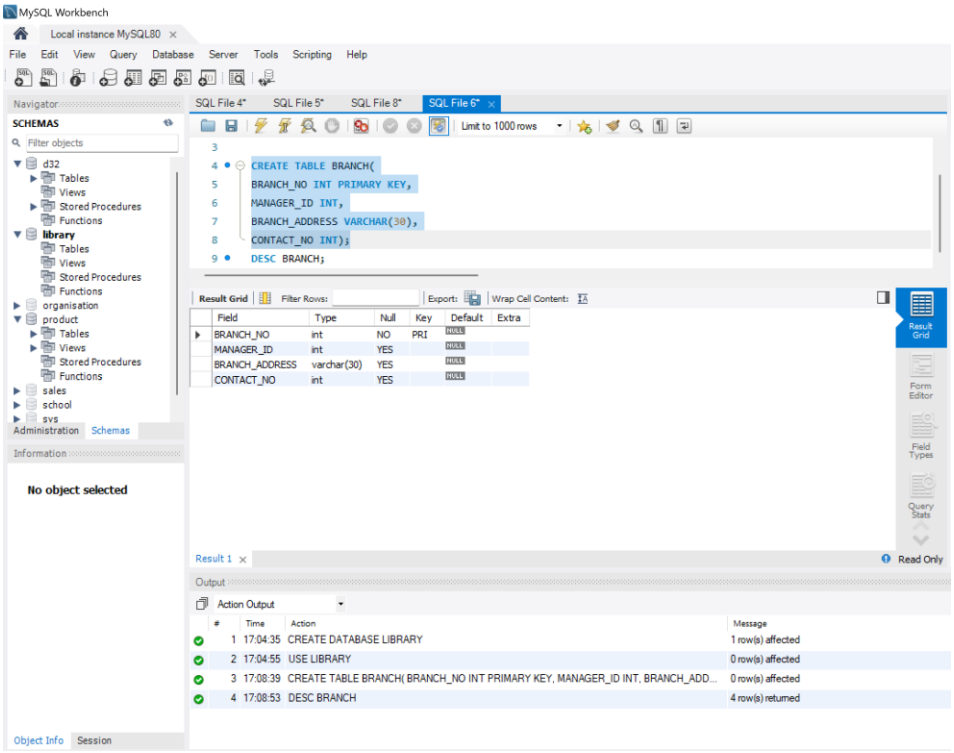

Figure 1 : Table named BRANCH with fields and description

2. EMPLOYEE

```
CREATE TABLE EMPLOYEE (
EMP_ID INT PRIMARY KEY,
EMP_NAME VARCHAR (20),
POSITION VARCHAR (30),
SALARY INT,
BRANCH_NO INT,
FOREIGN KEY (BRANCH_NO) REFERENCES BRANCH (BRANCH_NO));
```

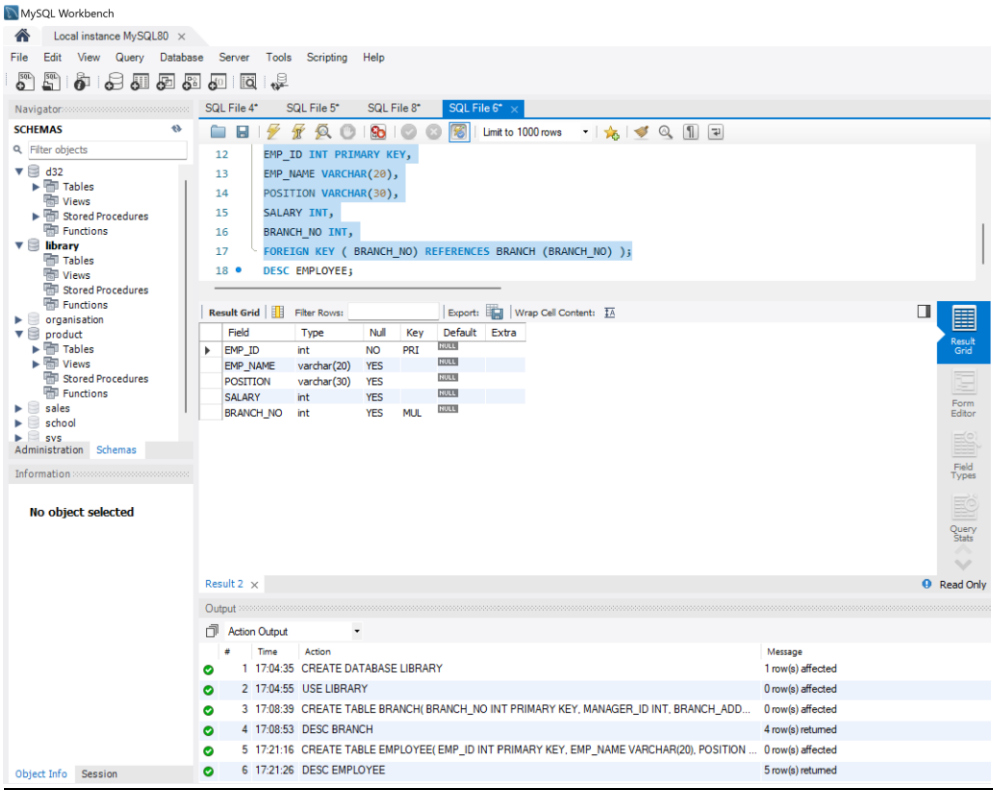

Figure 2: Table named EMPLOYEE with fields and description

3. BOOKS

```
CREATE TABLE BOOKS (
ISBN INT PRIMARY KEY,
BOOK_TITLE VARCHAR (50),
CATEGORY VARCHAR (20),
RENTAL_PRICE INT,
STATUS VARCHAR (5),
AUTHOR VARCHAR (20),
PUBLISHER VARCHAR (20));
```

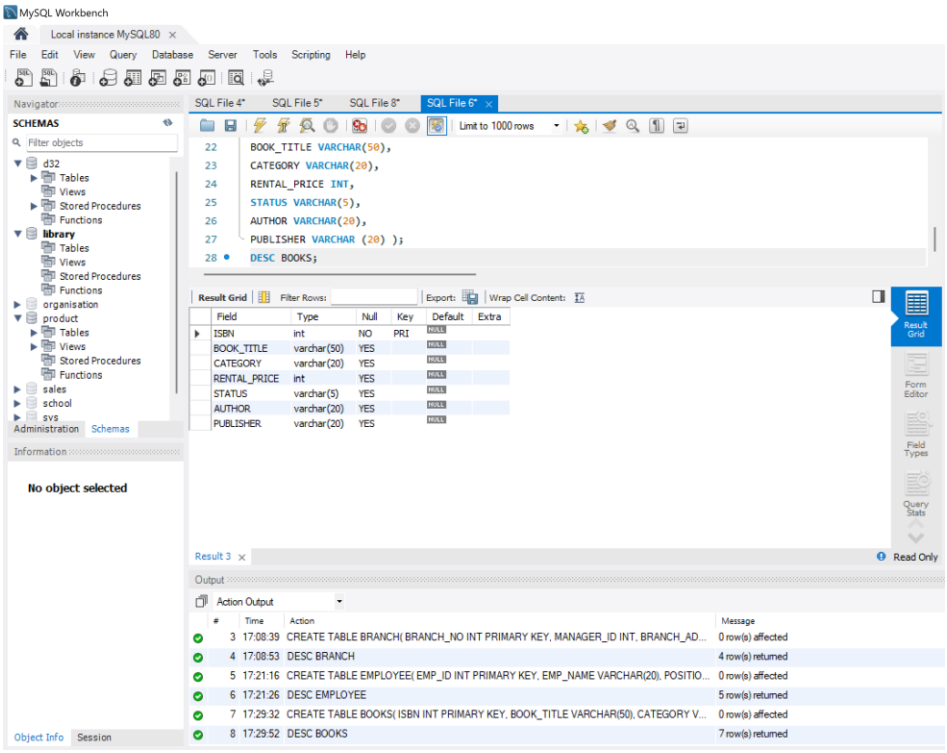

Figure 3: Table named BOOKS with fields and description

4. CUSTOMER

```
CREATE TABLE CUSTOMER (
CUSTOMER_ID INT PRIMARY KEY,
CUSTOMER_NAME VARCHAR (30),
CUSTOMER_ADDRESS VARCHAR (50),
REG_DATE DATE);
```

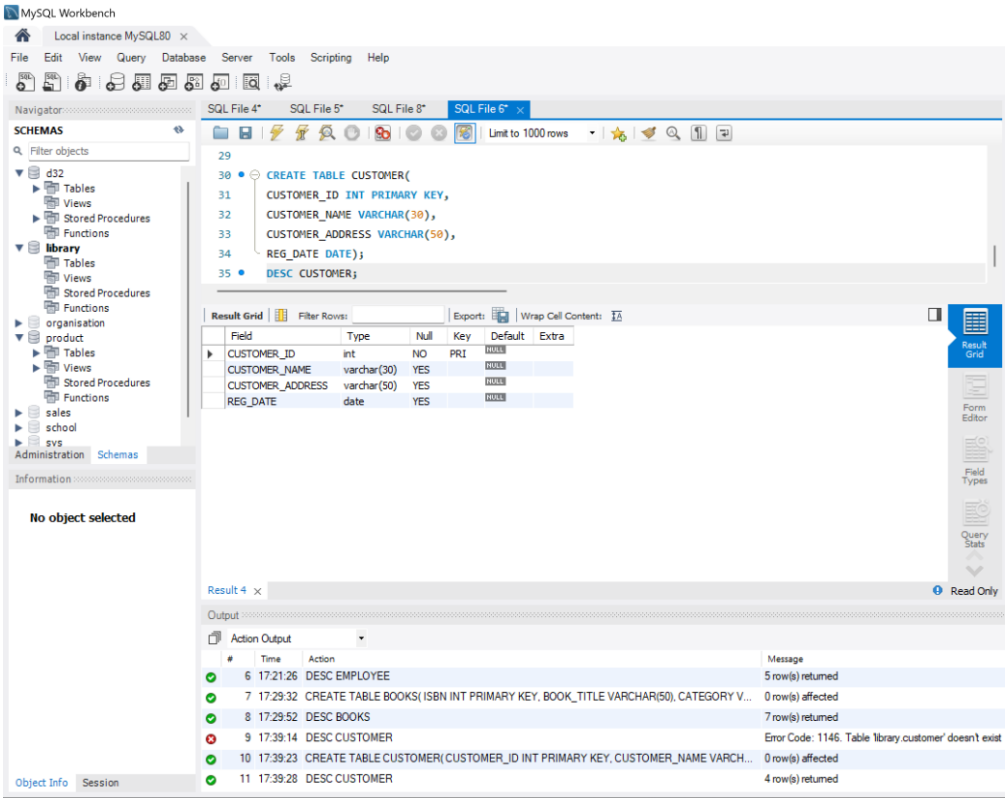

Figure 4: Table named CUSTOMER with fields and description

5. ISSUESTATUS

```
CREATE TABLE ISSUESTATUS (
ISSUE_ID INT PRIMARY KEY,
ISSUED_CUST INT,
ISSUED_BOOK_NAME VARCHAR (30),
ISSUE_DATE DATE,
ISBN_BOOK INT,
FOREIGN KEY(ISSUED_CUST) REFERENCES CUSTOMER(CUSTOMER_ID),
FOREIGN KEY (ISBN_BOOK) REFERENCES BOOKS(ISBN));
```

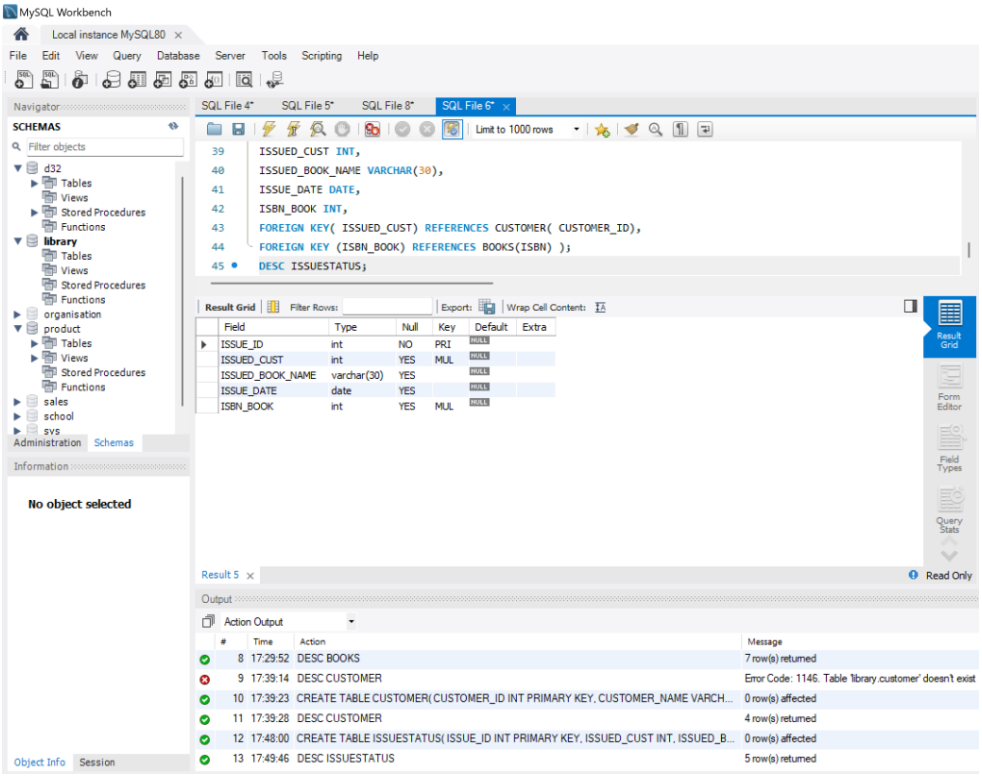

Figure 5: Table named ISSUESTATUS with fields and description

6. RETURNSTATUS

```
CREATE TABLE RETURNSTATUS (
RETURN_ID INT PRIMARY KEY,
RETURN_CUST INT,
RETURN_BOOK_NAME VARCHAR (30),
RETURN_DATE DATE,
ISBN_BOOK2 INT,
FOREIGN KEY (ISBN_BOOK2) REFERENCES BOOKS(ISBN));
```

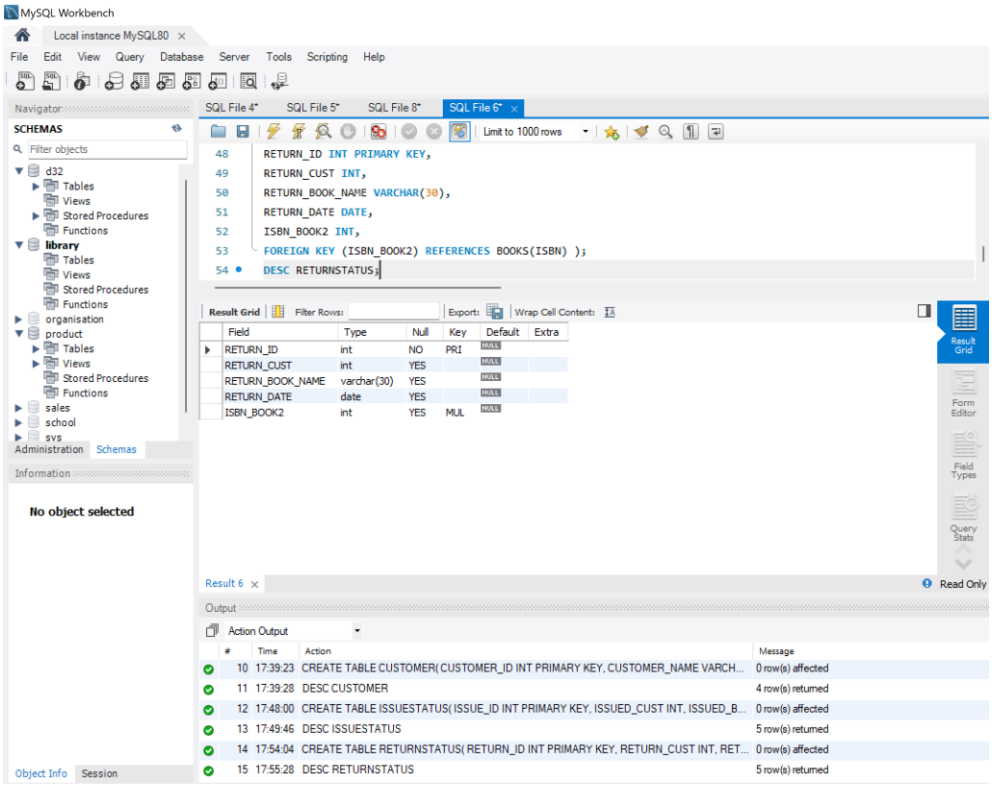

Figure 6: Table named RETURNSTATUS with fields and description

3.2.2. SQL SCRIPTS TO INSERT SAMPLE DATA

The following query was used to insert data into each table:

INSERT INTO tablename VALUES (column1 datatype1, column2 datatype2, …column n datatype n);

1. BRANCH

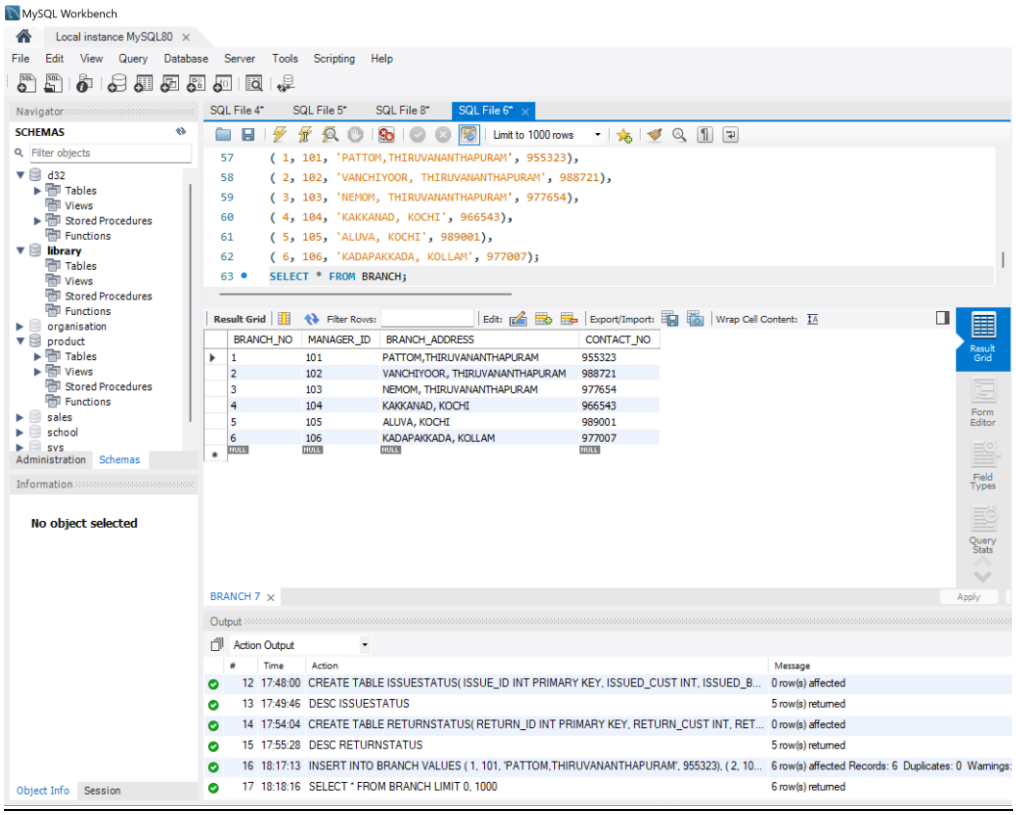

Figure 7: Values inserted to the table named BRANCH

2. EMPLOYEE

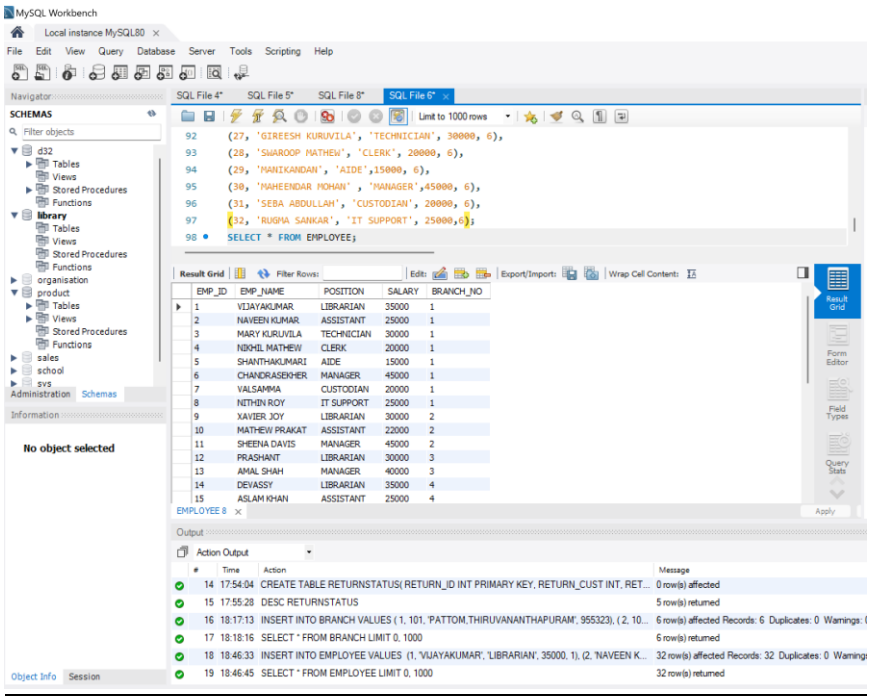

Figure 8: Values inserted to the table named EMPLOYEE

3. BOOKS

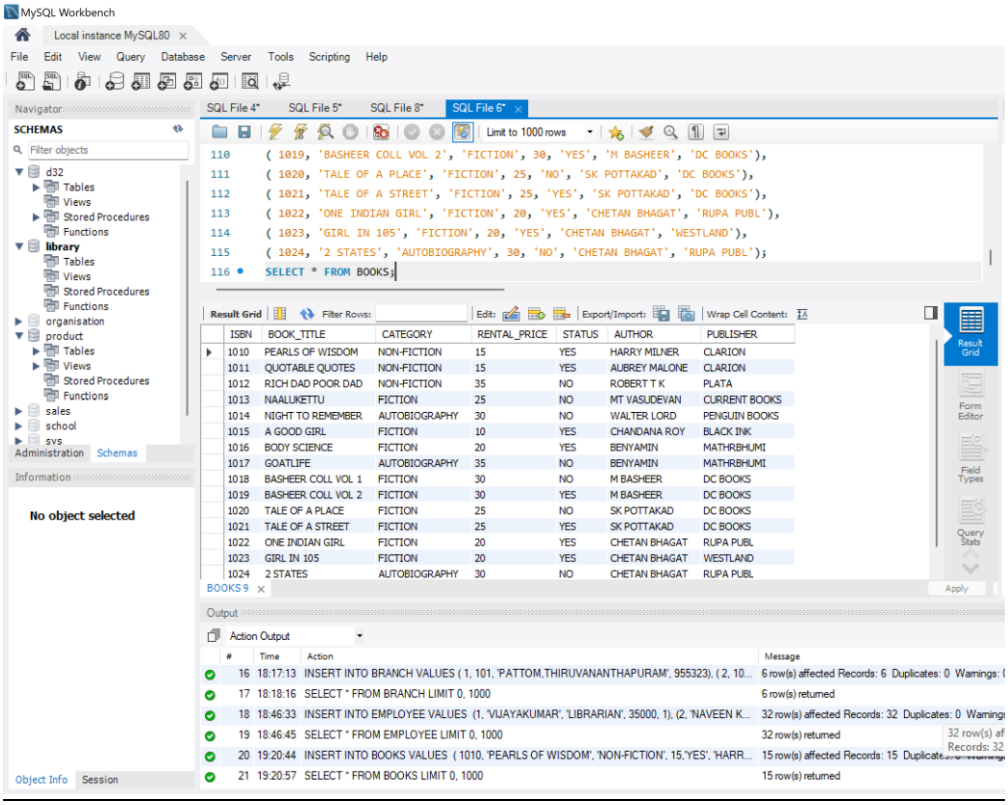

Figure 9: Values inserted to the Table named BOOKS

4. CUSTOMER

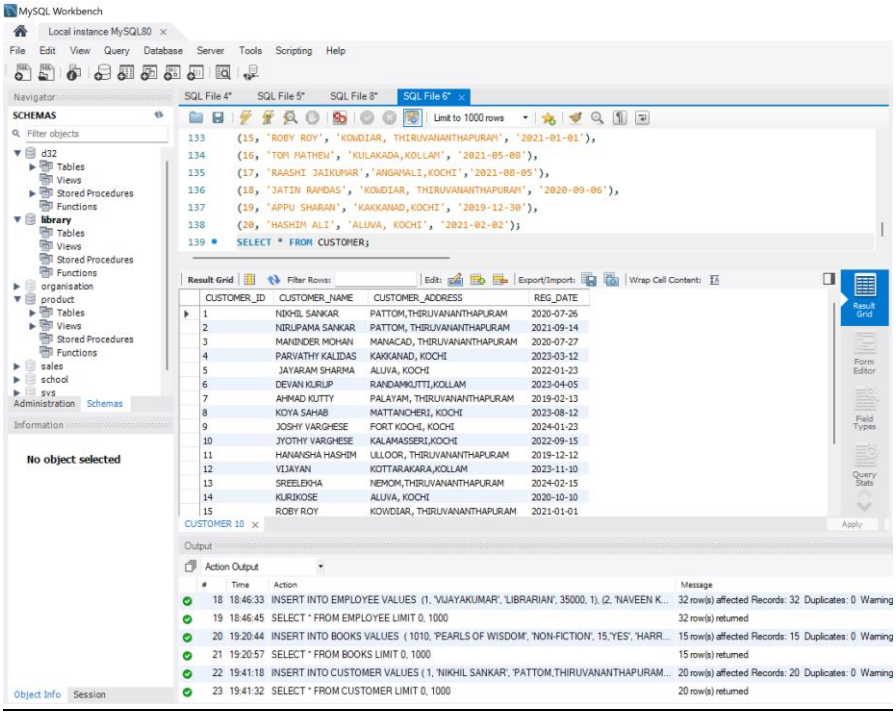

Figure 10: Values inserted to the table named CUSTOMER

5. ISSUESTATUS

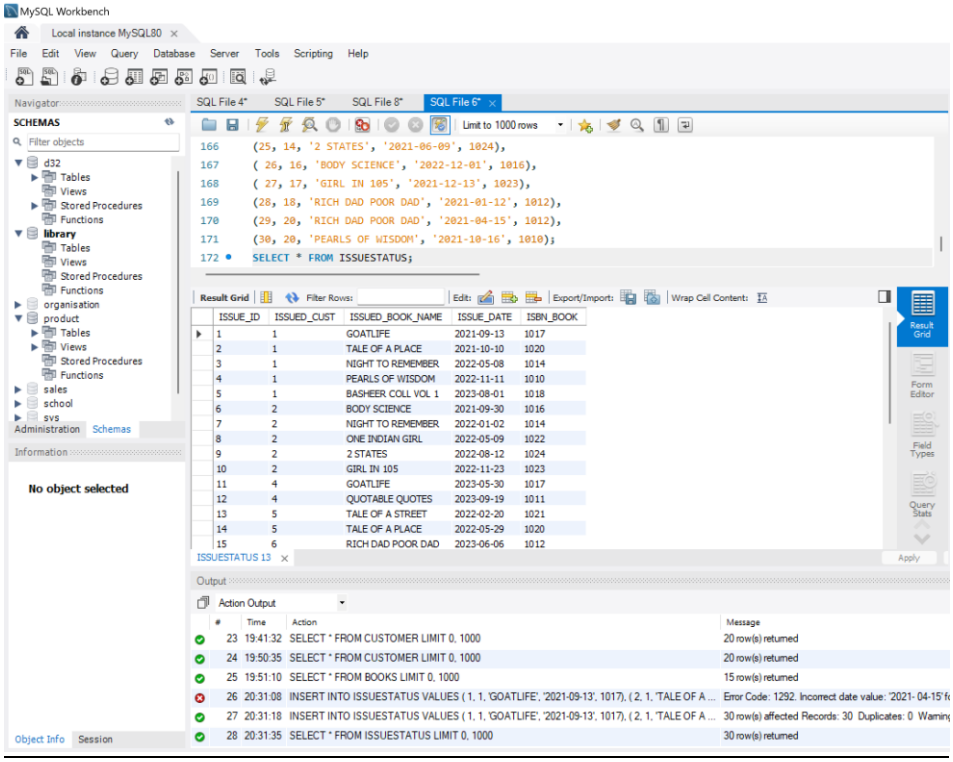

Figure 11: Values inserted to the table named ISSUESTATUS

6. RETURNSTATUS

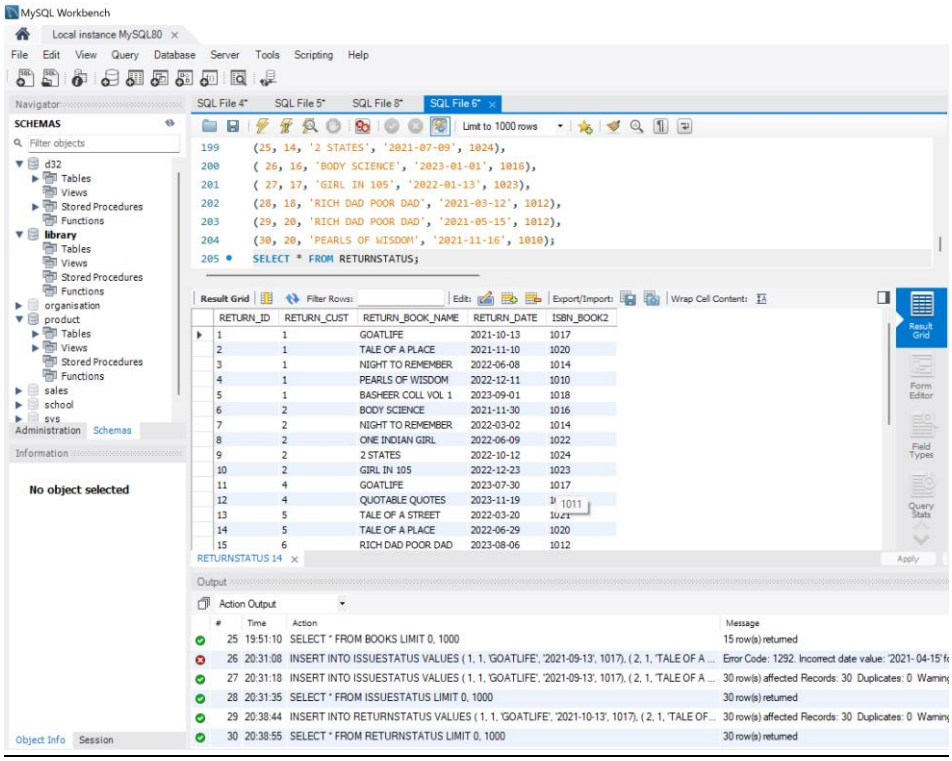

Figure 12: Values inserted to the table named RETURNSTATUS

3.2.3. MENTIONED QUERIES

1. RETRIEVE THE BOOK TITLE, CATEGORY, AND RENTAL PRICE OF ALL AVAILABLE BOOKS.

"SELECT BOOK_TITLE, CATEGORY, RENTAL_PRICE FROM BOOKS WHERE STATUS = 'YES';"

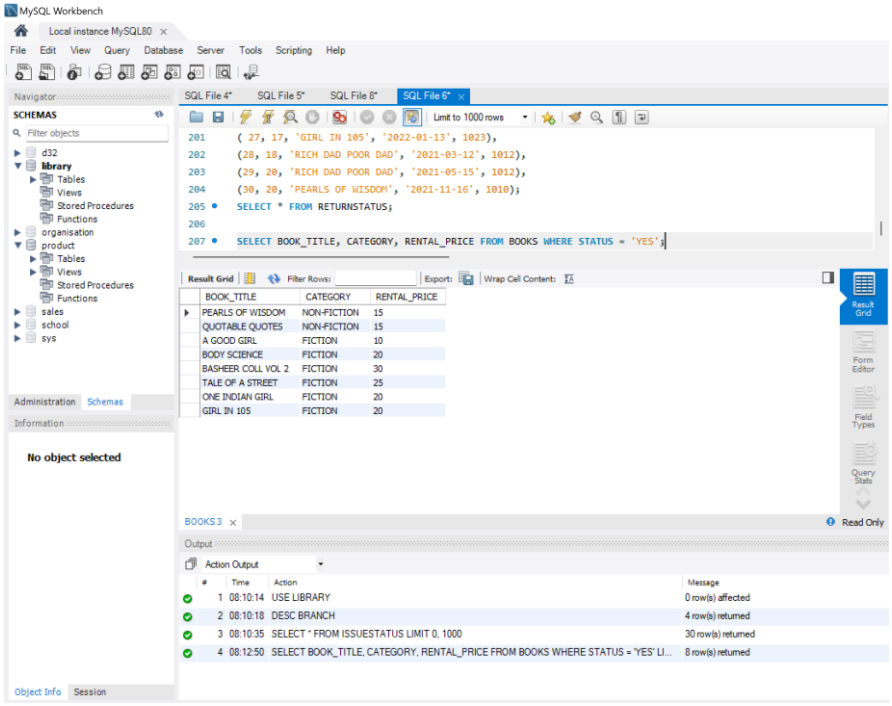

Figure 13: Result displaying book titles and their respective categories and rental prices

2. LIST THE EMPLOYEE NAMES AND THEIR RESPECTIVE SALARIES IN DESCENDING ORDER OF SALARY.

"SELECT EMP_NAME, SALARY FROM EMPLOYEE ORDER BY SALARY DESC;"

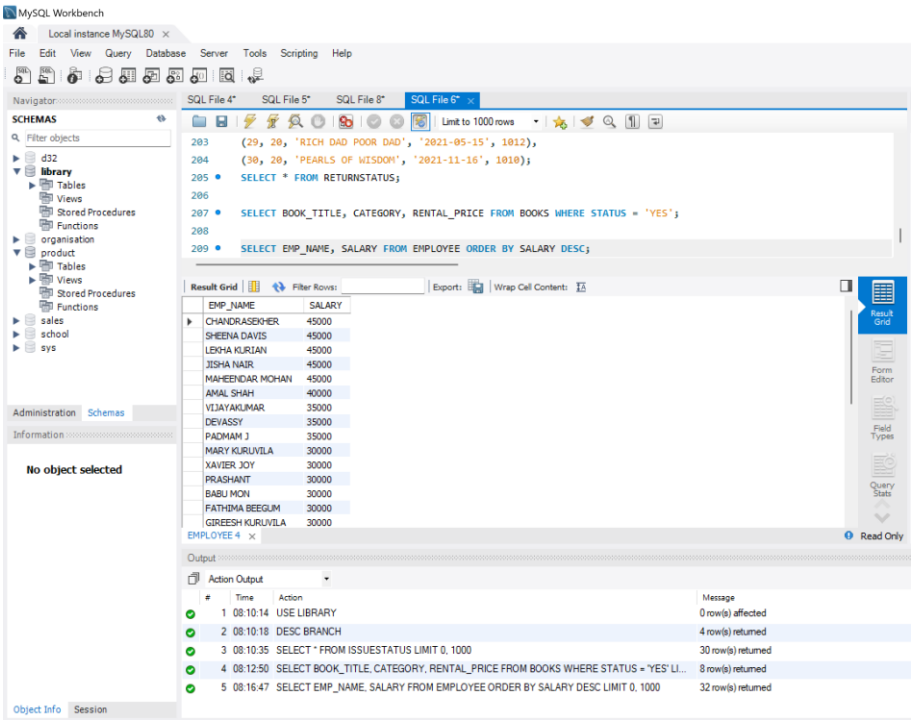

Figure 14: Results displaying the employees and their respective salaries in descending order

3. RETRIEVE THE BOOK TITLES AND THE CORRESPONDING CUSTOMERS WHO HAVE ISSUED THOSE BOOKS.

SELECT ISSUESTATUS.ISSUED_BOOK_NAME AS BOOK_TITLE, CUSTOMER.CUSTOMER_NAME FROM

ISSUESTATUS INNER JOIN CUSTOMER ON

ISSUESTATUS.ISSUED_CUST = CUSTOMER.CUSTOMER_ID;

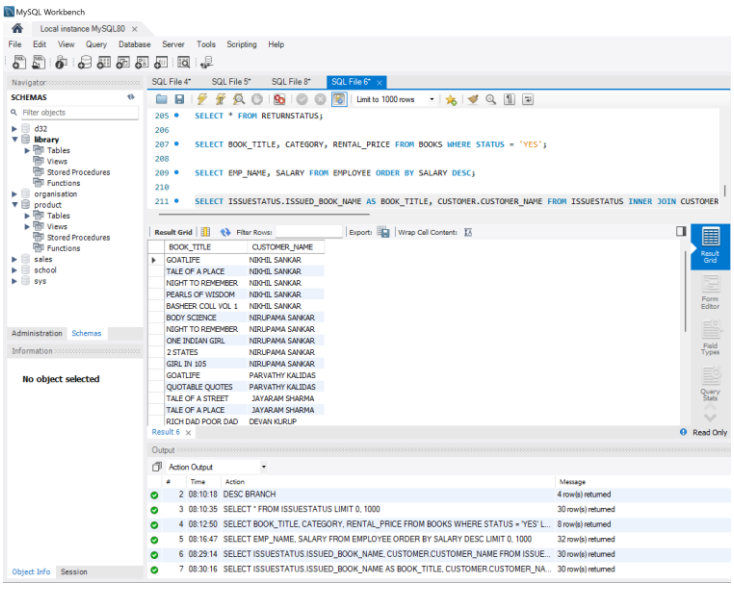

Figure 15: Results displaying the book titles and the customers who issued the book

4. DISPLAY THE TOTAL COUNT OF BOOKS IN EACH CATEGORY.

"SELECT CATEGORY, COUNT (*) AS NUMBER_OF_BOOKS FROM BOOKS GROUP BY CATEGORY;"

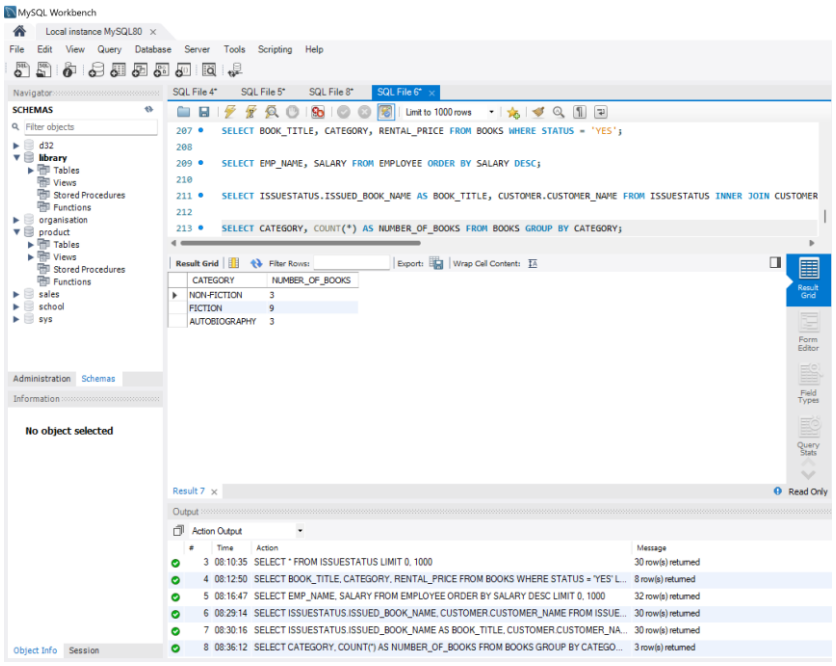

Figure 16: Results displaying the total number of books in each available genre

5. RETRIEVE THE EMPLOYEE NAMES AND THEIR POSITIONS FOR THE EMPLOYEES WHOSE SALARIES ARE ABOVE RS.30,000.

"SELECT EMP_NAME, POSITION FROM EMPLOYEE WHERE SALARY > 30000;"

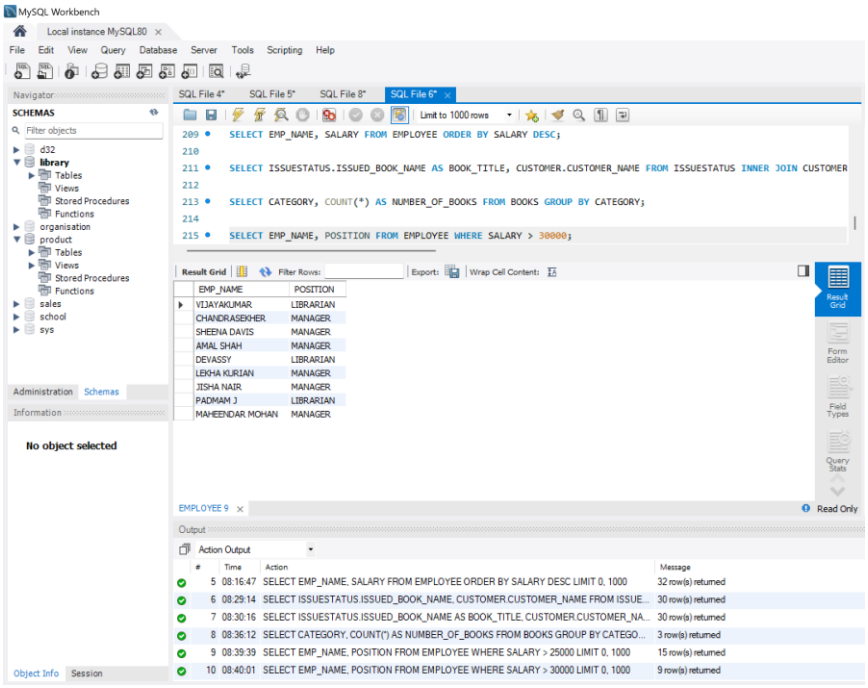

Figure 17: Results displaying the names and positions of employees having salaries greater than Rs.30,000

6. LIST THE CUSTOMER NAMES WHO REGISTERED BEFORE 2022-01-01 AND HAVE NOT ISSUED ANY BOOKS YET.

"SELECT CUSTOMER.CUSTOMER_NAME FROM

CUSTOMER LEFT JOIN ISSUESTATUS ON

CUSTOMER.CUSTOMER_ID = ISSUESTATUS.ISSUED_CUST WHERE CUSTOMER.REG_DATE < '2022-01-01' AND ISSUESTATUS.ISSUED_CUST IS NULL;"

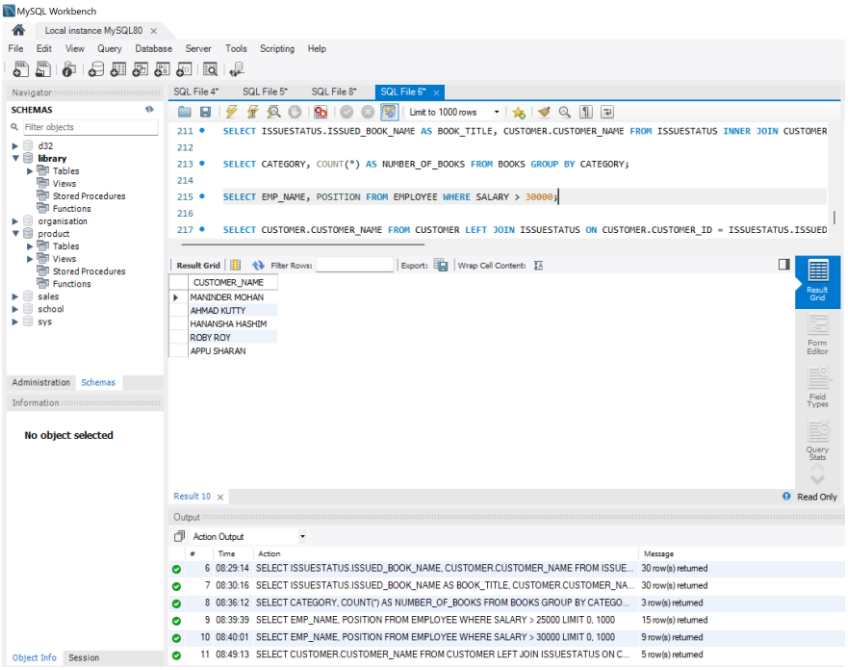

Figure 18: Results displaying the customers who have registered before 01/01/2022 and have not issued any books

7. DISPLAY THE BRANCH NUMBERS AND THE TOTAL COUNT OF EMPLOYEES IN EACH BRANCH.

"SELECT BRANCH_NO, COUNT (*) AS TOTAL_EMPLOYEES FROM EMPLOYEE GROUP BY BRANCH_NO;"

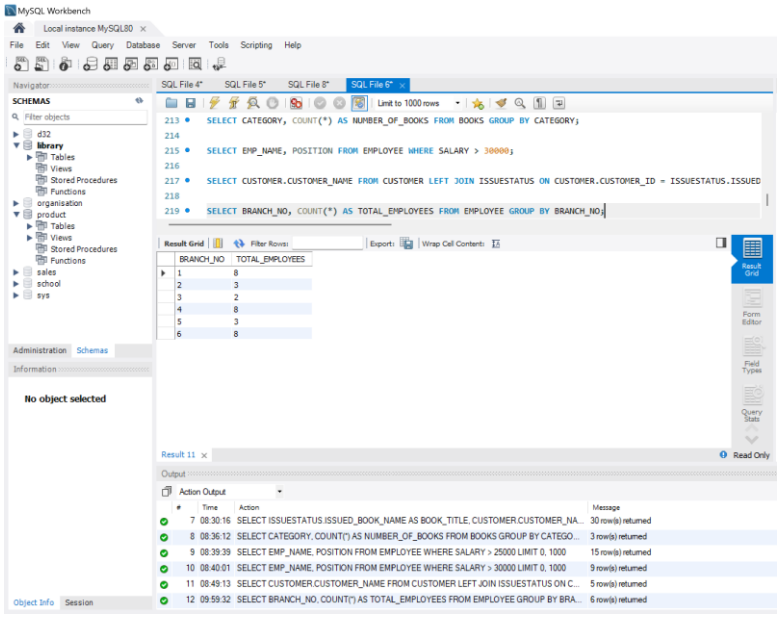

Figure 19: Results displaying the total number of employees in each branch

8. DISPLAY THE NAMES OF CUSTOMERS WHO HAVE ISSUED BOOKS IN THE MONTH OF JUNE 2023.

"SELECT CUSTOMER.CUSTOMER_NAME FROM

CUSTOMER INNER JOIN ISSUESTATUS ON

CUSTOMER.CUSTOMER_ID = ISSUESTATUS.ISSUED_CUST WHERE

MONTH (ISSUESTATUS.ISSUE_DATE) = 6 AND YEAR (ISSUESTATUS.ISSUE_DATE) = 2023;"

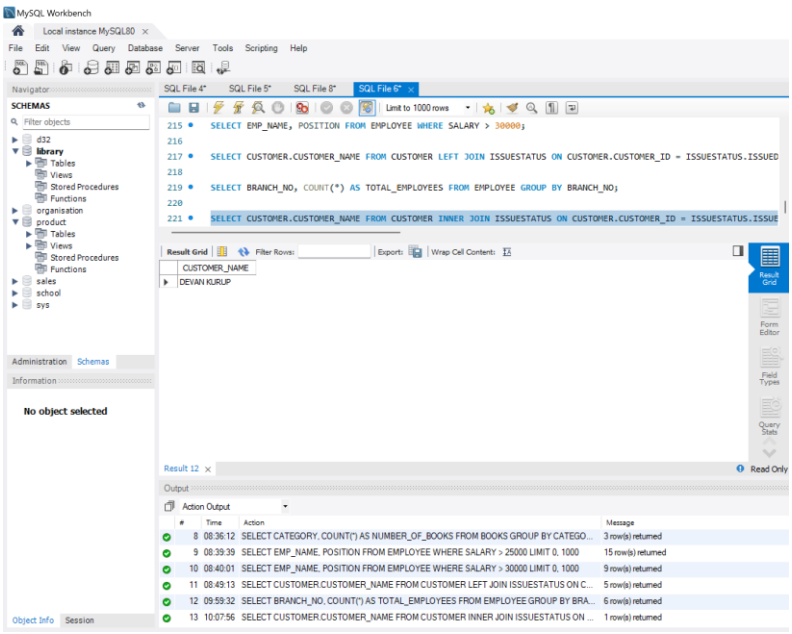

Figure 20: Results displaying the customers who issued books in JUNE 2023

9. RETRIEVE BOOK_TITLE FROM BOOK TABLE CONTAINING HISTORY.

"SELECT BOOK_TITLE FROM BOOKS WHERE BOOK_TITLE LIKE 'TALE%';"

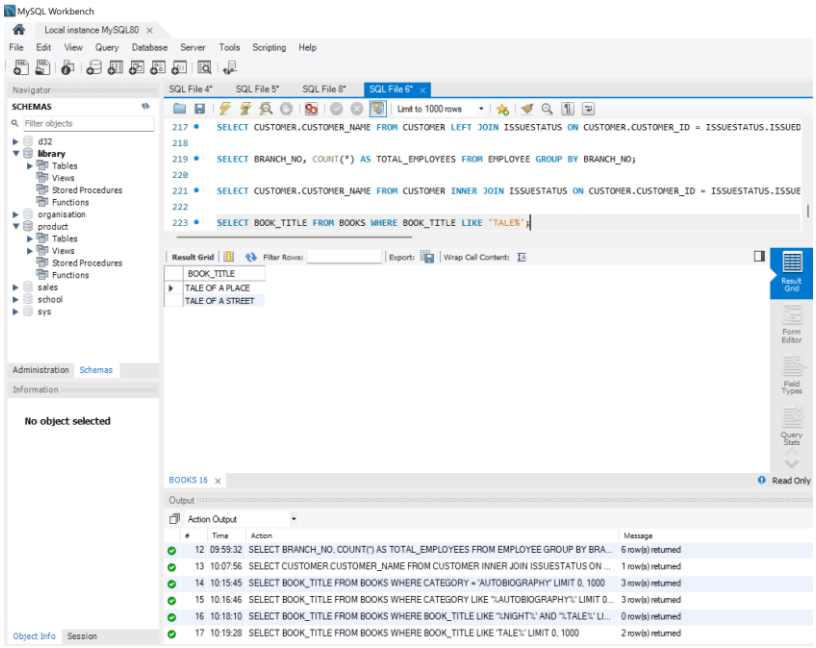

Figure 21: Results displaying book titles having history

10. RETRIEVE THE BRANCH NUMBERS ALONG WITH THE COUNT OF EMPLOYEES FOR BRANCHES HAVING MORE THAN 5 EMPLOYEES

"SELECT BRANCH_NO, COUNT (*) AS NUMBER_OF_EMPLOYEES FROM EMPLOYEE GROUP BY BRANCH_NO HAVING COUNT (*)>5;"

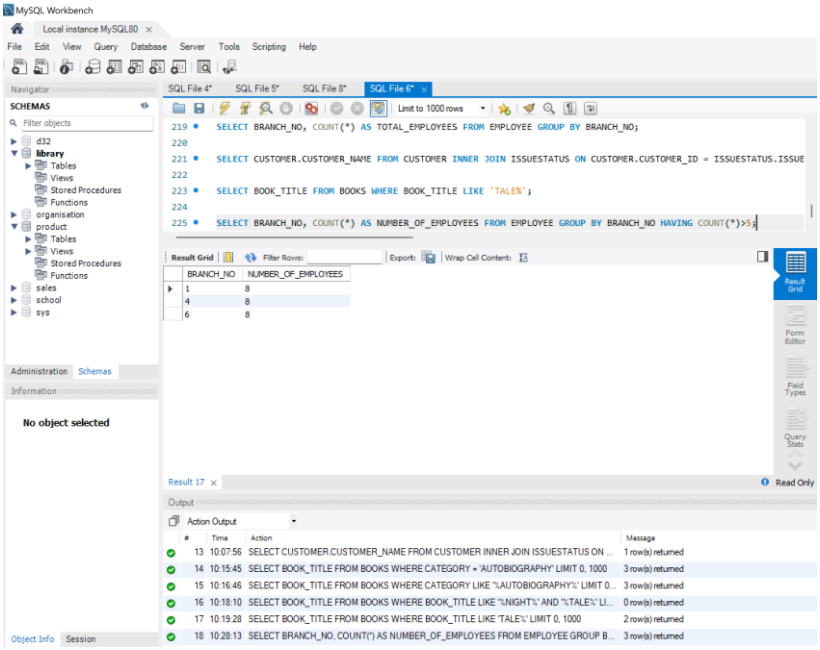

Figure 22: Results displaying branches having more than 5 employees

11. RETRIEVE THE NAMES OF EMPLOYEES WHO MANAGE BRANCHES AND THEIR RESPECTIVE BRANCH ADDRESSES.

"SELECT EMPLOYEE.EMP_NAME AS BRANCHMANAGER, BRANCH.BRANCH_ADDRESS FROM

EMPLOYEE INNER JOIN BRANCH ON

EMPLOYEE.BRANCH_NO = BRANCH.BRANCH_NO WHERE

EMPLOYEE.POSITION = 'MANAGER';"

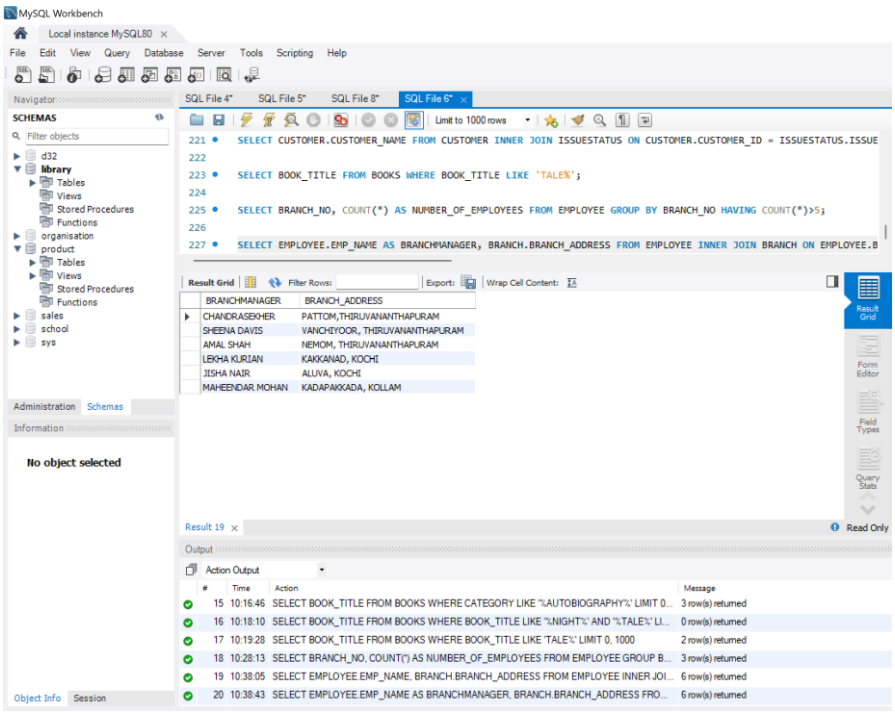

Figure 23: Results displaying the Managers and their respective branch addresses

12. DISPLAY THE NAMES OF CUSTOMERS WHO HAVE ISSUED BOOKS WITH A RENTAL PRICE HIGHER THAN RS. 25.

"SELECT DISTINCT CUSTOMER.CUSTOMER_NAME FROM

ISSUESTATUS INNER JOIN CUSTOMER ON

ISSUESTATUS.ISSUED_CUST = CUSTOMER.CUSTOMER_ID

INNER JOIN BOOKS ON

ISSUESTATUS.ISBN_BOOK = BOOKS.ISBN WHERE

BOOKS.RENTAL_PRICE > 25;"

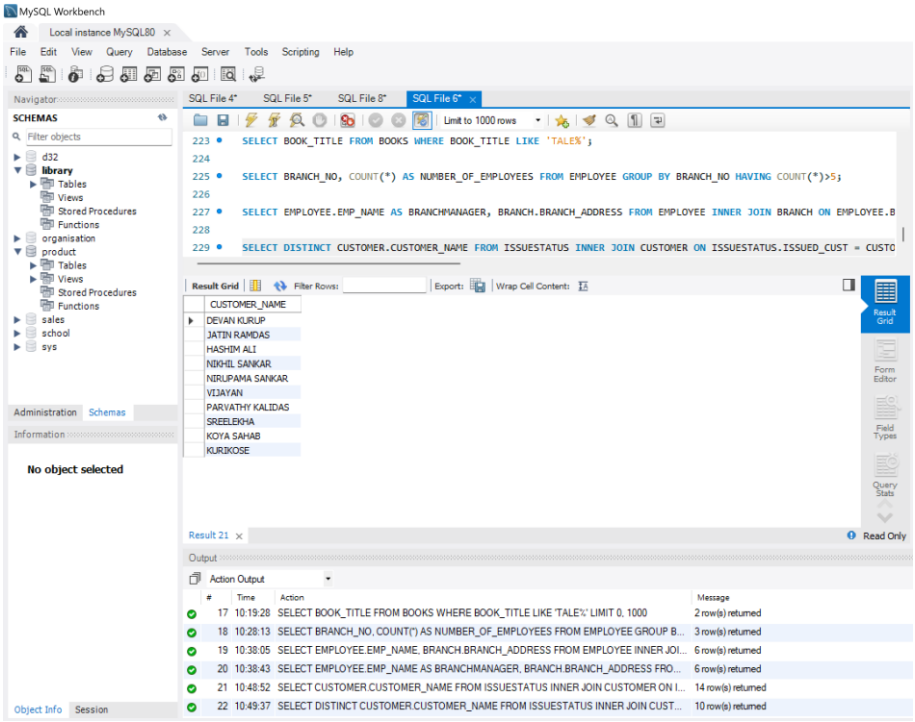

Figure 24: Results displaying customers who issued books with rent higher than Rs.25

4. CONCLUSION

This project successfully designed and implemented a database for a small-scale library using MySQL Workbench 8.0 CE. The database, structured into six tables—Branch, Employee, Books, Customer, IssueStatus, and ReturnStatus—effectively manages essential library operations such as book inventory, employee records, and customer transactions.

The SQL queries demonstrated practical applications in retrieving and manipulating data, highlighting SQL's efficiency in handling complex library datasets. These queries effectively addressed key aspects of library management, including tracking book availability, employee roles, and customer activities.

The project underscores the importance of a well-organized database in improving the efficiency of library operations and ensuring data integrity. While the current system meets the primary requirements, potential improvements include adding a more detailed loan tracking system and enhanced reporting features. These upgrades could further streamline operations and provide greater value to library staff and patrons.

Overall, this project showcases the capabilities of SQL in managing a library database and serves as a strong foundation for future enhancements.