

# **DESIGN AND IMPLEMENTATION OF A LIBRARY MANAGEMENT SYSTEM DATABASE**

**SUBMITTED BY**  
**NIKHIL SANKAR**

Nikhil

30/08/2024

# **Table of Contents**

<b>1. INTRODUCTION .....</b>	<b>3</b>
<b>2. SYSTEM DESIGN .....</b>	<b>3</b>
<b>2.1. REQUIREMENTS.....</b>	<b>3</b>
<b>2.1.1. FUNCTIONAL REQUIREMENTS:.....</b>	<b>3</b>
<b>2.1.2. NON-FUNCTIONAL REQUIREMENTS:.....</b>	<b>4</b>
<b>2.2. DATABASE DESIGN.....</b>	<b>4</b>
<b>2.2.1. Tables and Relationships .....</b>	<b>4</b>
<b>2.2.2. Data Types .....</b>	<b>6</b>
<b>3. IMPLEMENTATION.....</b>	<b>7</b>
<b>3.1. TOOLS AND TECHNOLOGIES.....</b>	<b>7</b>
<b>3.1.1. SOFTWARE .....</b>	<b>7</b>
<b>3.1.2. ADDITIONAL NOTES .....</b>	<b>7</b>
<b>3.2. SQL QUERIES .....</b>	<b>7</b>
<b>3.2.1. SQL SCRIPTS TO CREATE TABLES .....</b>	<b>7</b>
<b>3.2.2. SQL SCRIPTS TO INSERT SAMPLE DATA .....</b>	<b>13</b>
<b>3.2.3. MENTIONED QUERIES.....</b>	<b>18</b>
<b>4. CONCLUSION .....</b>	<b>27</b>

# 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);
```

The screenshot shows the MySQL Workbench interface with the following details:

- Schemas:** The left sidebar shows the database structure with Schemas like d32, library, organization, product, sales, school, and sys.
- SQL Editor:** The main area displays the SQL code for creating the BRANCH table:
 

```

CREATE TABLE BRANCH(
  BRANCH_NO INT PRIMARY KEY,
  MANAGER_ID INT,
  BRANCH_ADDRESS VARCHAR(30),
  CONTACT_NO INT);
DESC BRANCH;
      
```
- Result Grid:** Below the SQL editor, the result grid shows the table structure with four columns: Field, Type, Key, and Default. The data is:
 

Field	Type	Key	Default
BRANCH_NO	int	NO	PRI NULL
MANAGER_ID	int	YES	NULL
BRANCH_ADDRESS	varchar(30)	YES	NULL
CONTACT_NO	int	YES	NULL
- Action Output:** The bottom section shows the log of actions taken:
 

#	Time	Action	Message
1	17:04:35	CREATE DATABASE LIBRARY	1 row(s) affected
2	17:04:55	USE LIBRARY	0 row(s) affected
3	17:08:39	CREATE TABLE BRANCH(BRANCH_NO INT PRIMARY KEY, MANAGER_ID INT, BRANCH_ADD...	0 row(s) affected
4	17:08:53	DESC BRANCH	4 row(s) returned

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));
      
```

The screenshot shows the MySQL Workbench interface with the following details:

- Schemas:** The current schema is "d32".
- SQL File 6\*** tab: Contains the SQL code for creating the EMPLOYEE table.
 

```

12 EMP_ID INT PRIMARY KEY,
13 EMP_NAME VARCHAR(20),
14 POSITION VARCHAR(30),
15 SALARY INT,
16 BRANCH_NO INT,
17 FOREIGN KEY ( BRANCH_NO ) REFERENCES BRANCH ( BRANCH_NO );
18 DESC EMPLOYEE;
      
```
- Result Grid:** Shows the structure of the EMPLOYEE table with four columns: Field, Type, Null, and Key.
 

Field	Type	Null	Key
EMP_ID	int	NO	PRI
EMP_NAME	varchar(20)	YES	
POSITION	varchar(30)	YES	
SALARY	int	YES	
BRANCH_NO	int	YES	MUL
- Result 2** tab: Shows the history of database operations with the following log entries:
 

#	Time	Action	Message
1	17:04:35	CREATE DATABASE LIBRARY	1 row(s) affected
2	17:04:55	USE LIBRARY	0 row(s) affected
3	17:08:39	CREATE TABLE BRANCH( BRANCH_NO INT PRIMARY KEY, MANAGER_ID INT, BRANCH_ADD...	0 row(s) affected
4	17:08:53	DESC BRANCH	4 row(s) returned
5	17:21:16	CREATE TABLE EMPLOYEE( EMP_ID INT PRIMARY KEY, EMP_NAME VARCHAR(20), POSITION ...	0 row(s) affected
6	17:21:26	DESC EMPLOYEE	5 row(s) returned

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));
  
```

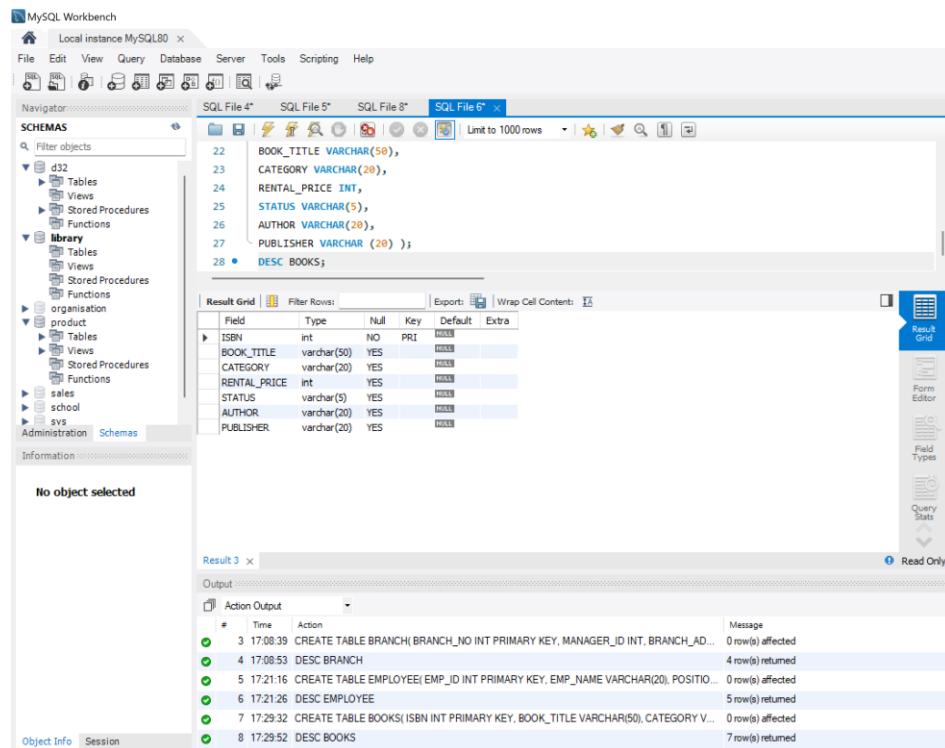


Figure 3: Table named *BOOKS* with fields and description

#### 4. CUSTOMER

```

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

```

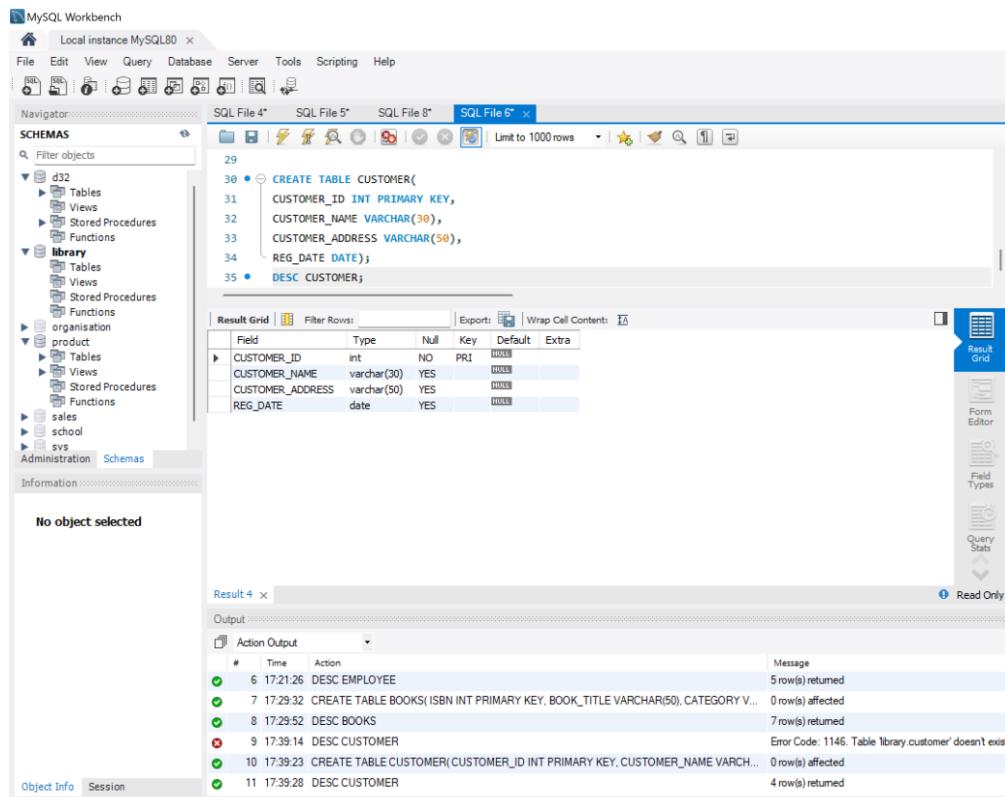


Figure 4: Table named CUSTOMER with fields and description

## 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));

```

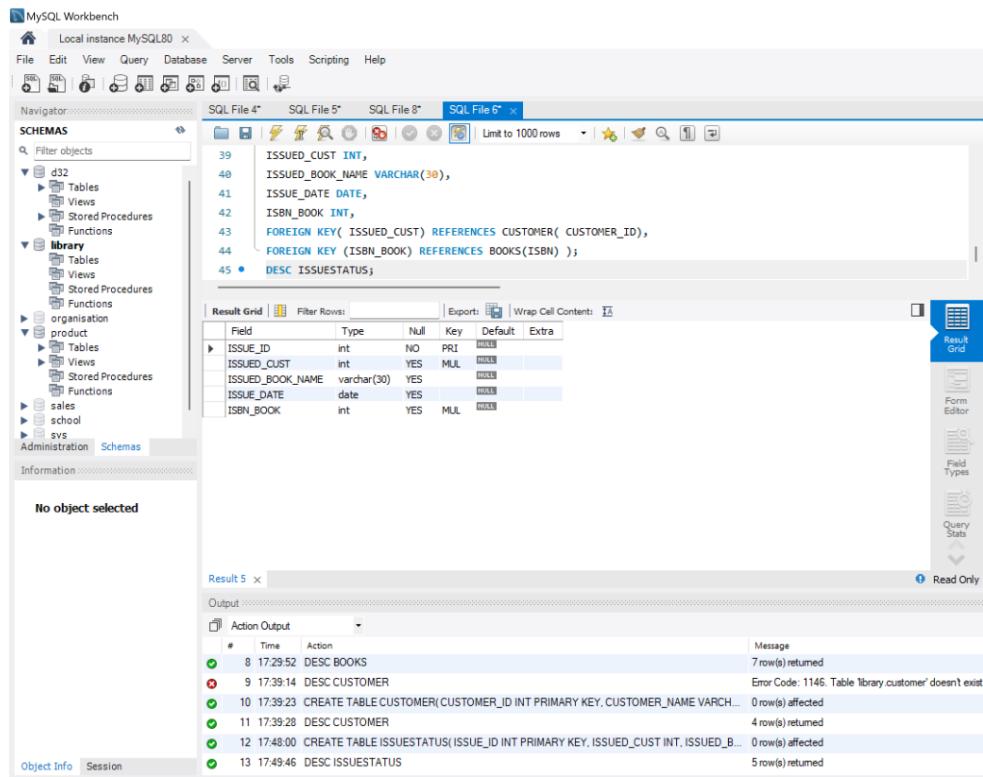


Figure 5: Table named ISSUESTATUS with fields and description

## 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));

```

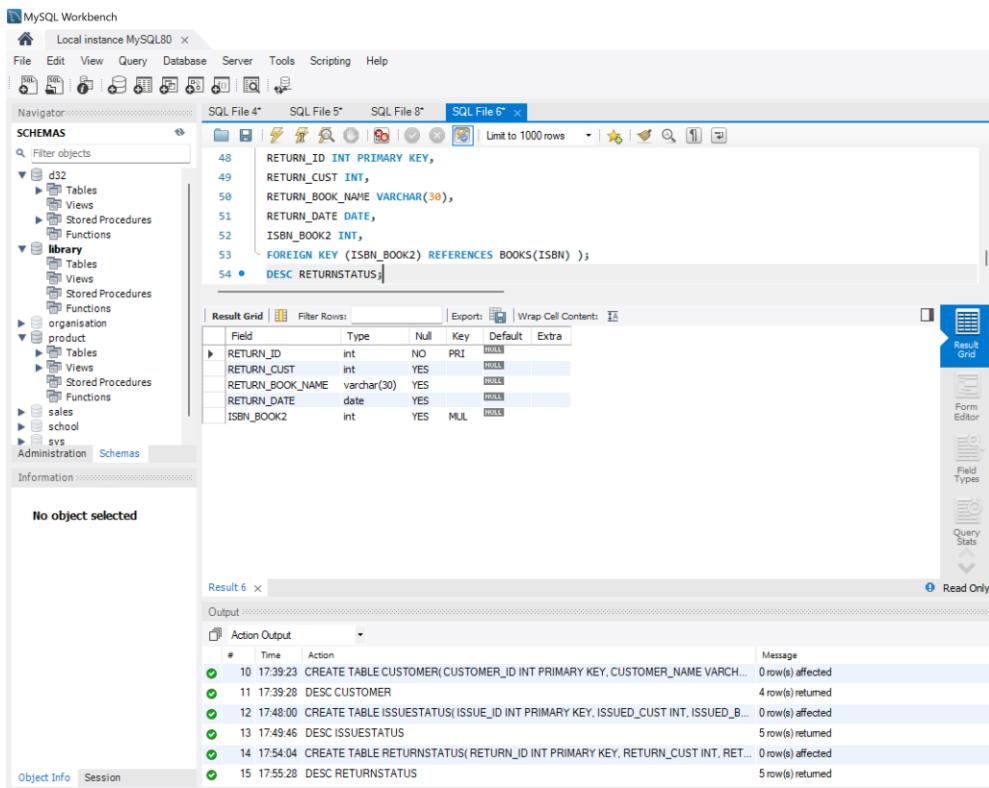


Figure 6: Table named RETURNSTATUS with fields and description

### **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

The screenshot shows the MySQL Workbench interface with the 'Local instance MySQL80' connection selected. In the Navigator pane, the 'Schemas' section is expanded, showing the 'library' schema which contains the 'BRANCH' table. The 'Tables' section is also visible. The main workspace displays the 'Result Grid' of the 'BRANCH' table, which has six rows of data. Below the grid is the SQL query: `SELECT * FROM BRANCH;`. To the right of the grid, there are various tools and tabs like 'Form Editor', 'Field Types', and 'Query Stats'. Below the grid, the 'Output' pane shows the 'Action Output' log with several entries, including table creation, descriptions, and data insertions. The log ends with a SELECT statement.

BRANCH_NO	MANAGER_ID	BRANCH_ADDRESS	CONTACT_NO
1	101	PATTOM,THIRUVANANTHAPURAM	955323
2	102	VANCHIYOOR, THIRUVANANTHAPURAM	988721
3	103	NEMOM, THIRUVANANTHAPURAM	977654
4	104	KAKKANAD, KOCHI	966543
5	105	ALUVA, KOCHI	989001
6	106	KADAPPAKKADA, KOLLAM	977007
NULL	NULL	NULL	NULL

Figure 7: Values inserted to the table named *BRANCH*

## 2. EMPLOYEE

The screenshot shows the MySQL Workbench interface with the 'Local instance MySQL80' connection selected. In the Navigator pane, the 'Schemas' section is expanded, showing the 'library' schema which contains the 'EMPLOYEE' table. The 'Tables' section is also visible. The main workspace displays the 'Result Grid' of the 'EMPLOYEE' table, which has 15 rows of data. Below the grid is the SQL query: `SELECT * FROM EMPLOYEE;`. To the right of the grid, there are various tools and tabs like 'Form Editor', 'Field Types', and 'Query Stats'. Below the grid, the 'Output' pane shows the 'Action Output' log with several entries, including table creation, descriptions, and data insertions. The log ends with a SELECT statement.

EMP_ID	EMP_NAME	POSITION	SALARY	BRANCH_NO
1	VIJAYAKUMAR	LIBRARIAN	35000	1
2	NAVEEN KUMAR	ASSISTANT	25000	1
3	MARY KURUVILA	TECHNICIAN	30000	1
4	NIKIL MATHEW	CLERK	20000	1
5	SHANTHAKUMARI	AIDE	15000	1
6	CHANDRASEKHER	MANAGER	45000	1
7	VALSAMMA	CUSTODIAN	20000	1
8	NITHIN ROY	IT SUPPORT	25000	1
9	XAVIER JOY	LIBRARIAN	30000	2
10	MATHEW PRAKAT	ASSISTANT	22000	2
11	SEENA DAVIS	MANAGER	45000	2
12	PRASHANT	LIBRARIAN	30000	3
13	AMAL SHAH	MANAGER	40000	3
14	DEVASVI	LIBRARIAN	35000	4
15	ASLAM KHAN	ASSISTANT	25000	4

Figure 8: Values inserted to the table named *EMPLOYEE*

### 3. BOOKS

The screenshot shows the MySQL Workbench interface with the 'BOOKS' table selected. The table has columns: ISBN, BOOK\_TITLE, CATEGORY, RENTAL\_PRICE, STATUS, AUTHOR, and PUBLISHER. The data includes various books like 'PEARLS OF WISDOM', 'QUOTEABLE QUOTES', and 'NIGHT TO REMEMBER'. The 'Output' pane shows the history of actions taken on the table.

ISBN	BOOK_TITLE	CATEGORY	RENTAL_PRICE	STATUS	AUTHOR	PUBLISHER
1010	PEARLS OF WISDOM	NON-FICTION	15	YES	HARRY MILNER	CLARION
1011	QUOTEABLE QUOTES	NON-FICTION	15	YES	AUBREY MALONE	CLARION
1012	RICH DAD POOR DAD	NON-FICTION	35	NO	ROBERT T K	PLATA
1013	NAALIKETTU	FICTION	25	NO	MT VASILDEVAN	CURRENT BOOKS
1014	NIGHT TO REMEMBER	AUTOBIOGRAPHY	30	NO	WALTER LORD	PENGUIN BOOKS
1015	A GOOD GIRL	FICTION	10	YES	CHANDANA ROY	BLACK INK
1016	BODY SCIENCE	FICTION	20	YES	BENYAMIN	MATHRSHUMI
1017	GOATLIFE	AUTOBIOGRAPHY	35	NO	BENYAMIN	MATHRSHUMI
1018	BASHEER COLL VOL 1	FICTION	30	NO	M BASHEER	DC BOOKS
1019	BASHEER COLL VOL 2	FICTION	30	YES	M BASHEER	DC BOOKS
1020	TALE OF A PLACE	FICTION	25	NO	SK POTTAKAD	DC BOOKS
1021	TALE OF A STREET	FICTION	25	YES	SK POTTAKAD	DC BOOKS
1022	ONE INDIAN GIRL	FICTION	20	YES	CHETAN BHAGAT	RUPA PUBL
1023	GIRL IN 105	FICTION	20	YES	CHETAN BHAGAT	WESTLAND
1024	2 STATES	AUTOBIOGRAPHY	30	NO	CHETAN BHAGAT	RUPA PUBL

Figure 9: Values inserted to the Table named BOOKS

### 4. CUSTOMER

The screenshot shows the MySQL Workbench interface with the 'CUSTOMER' table selected. The table has columns: CUSTOMER\_ID, CUSTOMER\_NAME, CUSTOMER\_ADDRESS, and REG\_DATE. The data includes various customers from different locations like Kondiar, Thiruvananthapuram, and Aluva. The 'Output' pane shows the history of actions taken on the table.

CUSTOMER_ID	CUSTOMER_NAME	CUSTOMER_ADDRESS	REG_DATE
1	NIKHIL SANKAR	PATTOM,THIRUVANANTHAPURAM	2020-07-26
2	NIRUPAMA SANKAR	PATTOM, THIRUVANANTHAPURAM	2021-09-14
3	MANINDER MOHAN	MANACAD, THIRUVANANTHAPURAM	2020-07-27
4	PARVATHY KALIDAS	KAKKANAD, KOCHI	2023-03-12
5	JAYARAM SHARMA	ALUVA, KOCHI	2022-01-23
6	DEVAN KURUP	RANDAMKUTTU,KOLLAM	2023-04-05
7	AHMAD KUTTY	PALAYAM,THIRUVANANTHAPURAM	2019-02-13
8	KOYA SAHAB	MATTANCHERY, KOCHI	2023-08-12
9	JOSEPH VARGHESE	FORT KOCHI, KOCHI	2024-01-23
10	JOYTHON VARGHESE	KALAMASERI,KOCHI	2022-09-15
11	HANANSHA HASHIM	ULLOR, THIRUVANANTHAPURAM	2019-12-12
12	VIJAYAN	KOTTARAKARA,KOLLAM	2023-11-10
13	SREELAKHA	NEMOM,THIRUVANANTHAPURAM	2024-02-15
14	KURIKOSE	ALUVA, KOCHI	2020-10-10
15	ROBY ROY	KOWDIAR, THIRUVANANTHAPURAM	2021-01-01

Figure 10: Values inserted to the table named CUSTOMER

## 5. ISSUESTATUS

The screenshot shows the MySQL Workbench interface with the 'ISSUESTATUS' table selected. The table has four columns: ISSUE\_ID, ISSUED\_CUST, ISSUED\_BOOK\_NAME, and ISSUE\_DATE. The data grid displays 15 rows of inserted values. The 'Output' pane at the bottom shows the history of actions taken on this table.

ISSUE_ID	ISSUED_CUST	ISSUED_BOOK_NAME	ISSUE_DATE
1	1	GOATLIFE	2021-09-13
2	1	TALE OF A PLACE	2021-10-10
3	1	NIGHT TO REMEMBER	2022-05-08
4	1	PEARLS OF WISDOM	2022-11-11
5	1	BASHEER COLL VOL 1	2023-08-01
6	2	BODY SCIENCE	2021-09-30
7	2	NIGHT TO REMEMBER	2022-01-02
8	2	ONE INDIAN GIRL	2022-05-09
9	2	2 STATES	2022-08-12
10	2	GIRL IN 105	2022-11-23
11	4	GOATLIFE	2023-05-30
12	4	QUOTEABLE QUOTES	2023-09-19
13	5	TALE OF A STREET	2022-02-20
14	5	TALE OF A PLACE	2022-05-29
15	6	RICH DAD POOR DAD	2023-06-06

Action Output:

#	Time	Action	Message
23	19:41:32	SELECT * FROM CUSTOMER LIMIT 0, 1000	20 row(s) returned
24	19:50:35	SELECT * FROM CUSTOMER LIMIT 0, 1000	20 row(s) returned
25	19:51:10	SELECT * FROM BOOKS LIMIT 0, 1000	15 row(s) returned
26	20:31:08	INSERT INTO ISSUESTATUS VALUES (1, 1, 'GOATLIFE', '2021-09-13', 1017), (2, 1, 'TALE OF A ...	Error Code: 1292. Incorrect date value: '2021-04-15' for column 'ISSUE_DATE' at row 1
27	20:31:18	INSERT INTO ISSUESTATUS VALUES (1, 1, 'GOATLIFE', '2021-09-13', 1017), (2, 1, 'TALE OF A ...	30 row(s) affected Records: 30 Duplicates: 0 Warnings: 0
28	20:31:35	SELECT * FROM ISSUESTATUS LIMIT 0, 1000	30 row(s) returned

Figure 11: Values inserted to the table named ISSUESTATUS

## 6. RETURNSTATUS

MySQL Workbench

Local instance MySQL80

File Edit View Query Database Server Tools Scripting Help

Navigator

SCHEMAS

- Filter objects
- d32
  - Tables
  - Views
  - Stored Procedures
  - Functions
- library
  - Tables
  - Views
  - Stored Procedures
  - Functions
- organisation
  - product
    - Tables
    - Views
    - Stored Procedures
    - Functions
  - sales
  - school
  - svs
- Administration Schemas

No object selected

RETURNSTATUS 14

Result Grid | Filter Rows: | Edit: | Export/Import: | Wrap Cell Content: | Result Grid | Form Editor | Field Types | Query Stats |

RETURN_ID	RETURN_CUST	RETURN_BOOK_NAME	RETURN_DATE	ISBN_BOOK2
1	1	GOATLIFE	2021-10-13	1017
2	1	TALE OF A PLACE	2021-11-10	1020
3	1	NIGHT TO REMEMBER	2022-06-08	1014
4	1	PEARLS OF WISDOM	2022-12-11	1010
5	1	BASHEER COLL VOL 1	2023-09-01	1018
6	2	BODY SCIENCE	2021-11-30	1016
7	2	NIGHT TO REMEMBER	2022-03-02	1014
8	2	ONE INDIAN GIRL	2022-06-09	1022
9	2	2 STATES	2022-10-12	1024
10	2	GIRL IN 105	2022-12-23	1023
11	4	GOATLIFE	2023-07-30	1017
12	4	QUOTEABLE QUOTES	2023-11-19	1011
13	5	TALE OF A STREET	2022-03-20	1021
14	5	TALE OF A PLACE	2022-06-29	1020
15	6	RICH DAD POOR DAD	2023-08-06	1012

Action Output

#	Time	Action	Message
25	19:51:10	SELECT * FROM BOOKS LIMIT 0, 1000	15 row(s) returned
26	20:31:08	INSERT INTO ISSUESTATUS VALUES (1, 1, 'GOATLIFE', '2021-09-13', 1017), (2, 1, 'TALE OF A ...	Error Code: 1232. Incorrect date value: '2021-04-15' for column 'RETURN_DATE' at row 1
27	20:31:18	INSERT INTO ISSUESTATUS VALUES (1, 1, 'GOATLIFE', '2021-09-13', 1017), (2, 1, 'TALE OF A ...	30 row(s) affected Records: 30 Duplicates: 0 Warnings: 0
28	20:31:35	SELECT * FROM ISSUESTATUS LIMIT 0, 1000	30 row(s) returned
29	20:38:44	INSERT INTO RETURNSTATUS VALUES (1, 1, 'GOATLIFE', '2021-10-13', 1017), (2, 1, 'TALE OF A ...	30 row(s) affected Records: 30 Duplicates: 0 Warnings: 0
30	20:38:55	SELECT * FROM RETURNSTATUS LIMIT 0, 1000	30 row(s) returned

Object Info Session

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';"*

The screenshot shows the MySQL Workbench interface. In the top navigation bar, 'Local instance MySQL80' is selected. The left sidebar displays the database schema with 'library' as the current database, containing tables like Books, Authors, and Publishers. The main area shows a SQL editor with the following code:

```
281   ( 27, 17, 'GIRL IN 105', '2022-01-13', 1023),
282   ( 28, 18, 'RICH DAD POOR DAD', '2021-03-12', 1012),
283   ( 29, 20, 'RICH DAD POOR DAD', '2021-05-15', 1012),
284   ( 30, 20, 'PEARLS OF WISDOM', '2021-11-16', 1010),
285 •  SELECT * FROM RETURNSTATUS;
286
287 •  SELECT BOOK_TITLE, CATEGORY, RENTAL_PRICE FROM BOOKS WHERE STATUS = 'YES';
```

The results grid displays the following data:

BOOK_TITLE	CATEGORY	RENTAL_PRICE
PEARLS OF WISDOM	NON-FICTION	15
QUOTEABLE QUOTES	NON-FICTION	15
A GOOD GIRL	FICTION	10
BODY SCIENCE	FICTION	20
BASH-KER COLL VOL 2	FICTION	30
TALE OF A STREET	FICTION	25
ONE INDIAN GIRL	FICTION	20
GIRL IN 105	FICTION	20

The bottom pane shows the 'Action Output' log with the following entries:

#	Time	Action	Message
1	08:10:14	USE LIBRARY	0 row(s) affected
2	08:10:18	DESC BRANCH	4 row(s) returned
3	08:10:35	SELECT * FROM ISSUESTATUS LIMIT 0, 1000	30 row(s) returned
4	08:12:50	SELECT BOOK_TITLE, CATEGORY, RENTAL_PRICE FROM BOOKS WHERE STATUS = 'YES' LI...	8 row(s) returned

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;"*

The screenshot shows the MySQL Workbench interface with the following details:

- Schemas:** d32, library, organization, product, sales, school, sys.
- Query Editor:** Contains three SQL statements:
  - Statement 1: `SELECT \* FROM RETURNSTATUS;` (rows 203-204)
  - Statement 2: `SELECT BOOK\_TITLE, CATEGORY, RENTAL\_PRICE FROM BOOKS WHERE STATUS = 'YES';` (rows 205-206)
  - Statement 3: `SELECT EMP\_NAME, SALARY FROM EMPLOYEE ORDER BY SALARY DESC;` (rows 207-209)
- Result Grid:** Displays the results of the third query, showing employee names and salaries in descending order.
- Action Output:** Shows the history of actions taken during the session, including database usage and query execution.

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;
```

The screenshot shows the MySQL Workbench interface with the following details:

- Schemas:** d32, library, organization, product, sales, school, sys.
- Query Editor:** Contains five SQL statements:
  - Statement 1: `SELECT \* FROM RETURNSTATUS;` (rows 205-206)
  - Statement 2: `SELECT BOOK\_TITLE, CATEGORY, RENTAL\_PRICE FROM BOOKS WHERE STATUS = 'YES';` (rows 207-208)
  - Statement 3: `SELECT EMP\_NAME, SALARY FROM EMPLOYEE ORDER BY SALARY DESC;` (rows 209-210)
  - Statement 4: `SELECT ISSUESTATUS.ISSUED\_BOOK\_NAME AS BOOK\_TITLE, CUSTOMER.CUSTOMER\_NAME FROM ISSUESTATUS INNER JOIN CUSTOMER ON ISSUESTATUS.ISSUED\_CUST = CUSTOMER.CUSTOMER\_ID;` (rows 211-212)
- Result Grid:** Displays the results of the fourth query, showing book titles and customer names.
- Action Output:** Shows the history of actions taken during the session, including database usage and query execution.

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;”**

The screenshot shows the MySQL Workbench interface. In the top navigation bar, 'Local instance MySQL80' is selected. The 'Navigator' panel on the left lists databases like 'library', 'organisation', 'product', 'sales', 'school', and 'sys'. The main area displays a SQL editor with several queries. The fourth query is:

```
207 • SELECT BOOK_TITLE, CATEGORY, RENTAL_PRICE FROM BOOKS WHERE STATUS = 'YES';  
208  
209 • SELECT EMP_NAME, SALARY FROM EMPLOYEE ORDER BY SALARY DESC;  
210  
211 • SELECT ISSUESTATUS.ISSUED_BOOK_NAME AS BOOK_TITLE, CUSTOMER.CUSTOMER_NAME FROM ISSUESTATUS INNER JOIN CUSTOMER  
212  
213 • SELECT CATEGORY, COUNT(*) AS NUMBER_OF_BOOKS FROM BOOKS GROUP BY CATEGORY;
```

The results grid shows the following data:

CATEGORY	NUMBER_OF_BOOKS
NON-FICTION	3
FICTION	9
AUTOBIOGRAPHY	3

The 'Output' pane at the bottom shows the session history with the following entries:

#	Time	Action	Message
3	08:10:35	SELECT * FROM ISSUESTATUS LIMIT 0, 1000	30 row(s) returned
4	08:12:50	SELECT BOOK_TITLE, CATEGORY, RENTAL_PRICE FROM BOOKS WHERE STATUS = 'YES' L...	8 row(s) returned
5	08:16:47	SELECT EMP_NAME, SALARY FROM EMPLOYEE ORDER BY SALARY DESC LIMIT 0, 1000	32 row(s) returned
6	08:29:14	SELECT ISSUESTATUS.ISSUED_BOOK_NAME AS BOOK_TITLE, CUSTOMER.CUSTOMER_NAME FROM ISSUE...	30 row(s) returned
7	08:30:16	SELECT ISSUESTATUS.ISSUED_BOOK_NAME AS BOOK_TITLE, CUSTOMER.CUSTOMER_NAME...	30 row(s) returned
8	08:36:12	SELECT CATEGORY, COUNT(*) AS NUMBER_OF_BOOKS FROM BOOKS GROUP BY CATEG...	3 row(s) returned

*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;”**

The screenshot shows the MySQL Workbench interface with several tabs open. The 'Navigator' tab displays the database schema with tables like 'EMPLOYEE', 'ISSUESTATUS', 'BOOKS', and 'CUSTOMER'. The 'SQL File 6\*' tab contains the following SQL code:

```

209 • SELECT EMP_NAME, SALARY FROM EMPLOYEE ORDER BY SALARY DESC
210
211 • SELECT ISSUESTATUS.ISSUED_BOOK_NAME AS BOOK_TITLE, CUSTOMER.CUSTOMER_NAME FROM ISSUESTATUS INNER JOIN CUSTOMER
212
213 • SELECT CATEGORY, COUNT(*) AS NUMBER_OF_BOOKS FROM BOOKS GROUP BY CATEGORY;
214
215 • SELECT EMP_NAME, POSITION FROM EMPLOYEE WHERE SALARY > 30000;

```

The 'Result Grid' tab shows the results of the last query, listing employee names and positions:

EMP_NAME	POSITION
VIDAYAKUMAR	LIBRARIAN
CHANDRASHEKHAR	MANAGER
SHEENA DAVIS	MANAGER
AMALA CHAH	MANAGER
DEVASSI	LIBRARIAN
LEENA KURJAN	MANAGER
ISHA NAIR	MANAGER
PADMAM J	LIBRARIAN
MAHEENDAR MOHAN	MANAGER

The 'Object Info' and 'Session' tabs are also visible at the bottom.

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;”*

The screenshot shows the MySQL Workbench interface with the following details:

- Schemas:** library, organisation, sales, school, sys
- SQL Editor:**

```

211 • SELECT ISSUESTATUS.ISSUED_BOOK_NAME AS BOOK_TITLE, CUSTOMER.CUSTOMER_NAME FROM ISSUESTATUS INNER JOIN CUSTOMER
212
213 • SELECT CATEGORY, COUNT(*) AS NUMBER_OF_BOOKS FROM BOOKS GROUP BY CATEGORY;
214
215 • SELECT EMP_NAME, POSITION FROM EMPLOYEE WHERE SALARY > 30000;
216
217 • SELECT CUSTOMER.CUSTOMER_NAME FROM CUSTOMER LEFT JOIN ISSUESTATUS ON CUSTOMER.CUSTOMER_ID = ISSUESTATUS.ISSUED
    
```
- Result Grid:**

CUSTOMER_NAME
MANINDER MOHAN
AHMAD KUTTY
HANANISHA HASHIM
ROBY ROY
APPU SHARAN
- Action Output:**

#	Time	Action	Message
6	08:29:14	SELECT ISSUESTATUS.ISSUED_BOOK_NAME, CUSTOMER.CUSTOMER_NAME FROM ISSUESTATUS INNER JOIN CUSTOMER	30 row(s) returned
7	08:30:16	SELECT ISSUESTATUS.ISSUED_BOOK_NAME AS BOOK_TITLE, CUSTOMER.CUSTOMER_NAME FROM ISSUESTATUS INNER JOIN CUSTOMER	30 row(s) returned
8	08:36:12	SELECT CATEGORY, COUNT(*) AS NUMBER_OF_BOOKS FROM BOOKS GROUP BY CATEGORY;	3 row(s) returned
9	08:39:39	SELECT EMP_NAME, POSITION FROM EMPLOYEE WHERE SALARY > 25000 LIMIT 0, 1000	15 row(s) returned
10	08:40:01	SELECT EMP_NAME, POSITION FROM EMPLOYEE WHERE SALARY > 30000 LIMIT 0, 1000	9 row(s) returned
11	08:49:13	SELECT CUSTOMER.CUSTOMER_NAME FROM CUSTOMER LEFT JOIN ISSUESTATUS ON CUSTOMER.CUSTOMER_ID = ISSUESTATUS.ISSUED	5 row(s) returned

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;*”

The screenshot shows the MySQL Workbench interface with the following details:

- Schemas:** library, organisation, sales, school, sys
- SQL Editor:**

```

213 • SELECT CATEGORY, COUNT(*) AS NUMBER_OF_BOOKS FROM BOOKS GROUP BY CATEGORY;
214
215 • SELECT EMP_NAME, POSITION FROM EMPLOYEE WHERE SALARY > 30000;
216
217 • SELECT CUSTOMER.CUSTOMER_NAME FROM CUSTOMER LEFT JOIN ISSUESTATUS ON CUSTOMER.CUSTOMER_ID = ISSUESTATUS.ISSUED
218
219 • SELECT BRANCH_NO, COUNT(*) AS TOTAL_EMPLOYEES FROM EMPLOYEE GROUP BY BRANCH_NO;
    
```
- Result Grid:**

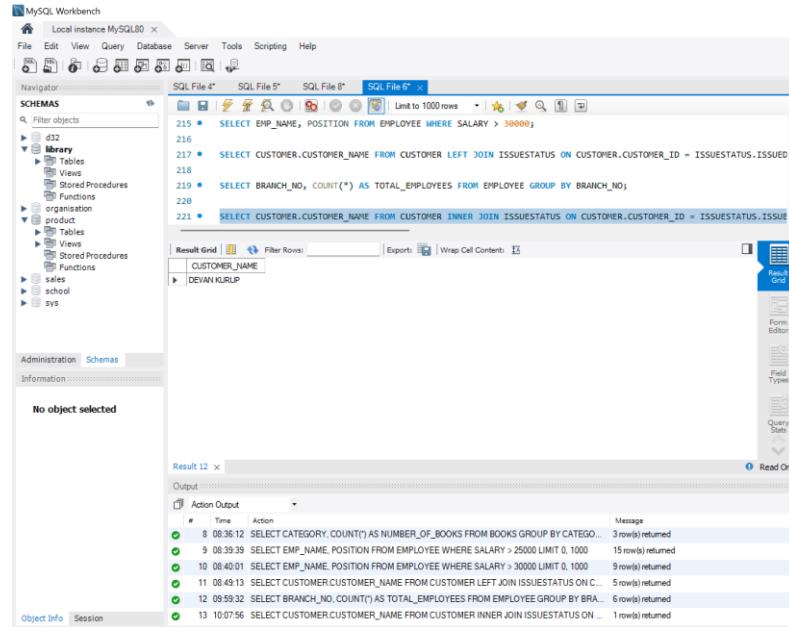
BRANCH_NO	TOTAL_EMPLOYEES
1	8
2	3
3	2
4	8
5	3
6	8
- Action Output:**

#	Time	Action	Message
7	08:30:16	SELECT ISSUESTATUS.ISSUED_BOOK_NAME AS BOOK_TITLE, CUSTOMER.CUSTOMER_NAME FROM ISSUESTATUS INNER JOIN CUSTOMER	30 row(s) returned
8	08:36:12	SELECT CATEGORY, COUNT(*) AS NUMBER_OF_BOOKS FROM BOOKS GROUP BY CATEGORY;	3 row(s) returned
9	08:39:39	SELECT EMP_NAME, POSITION FROM EMPLOYEE WHERE SALARY > 25000 LIMIT 0, 1000	15 row(s) returned
10	08:40:01	SELECT EMP_NAME, POSITION FROM EMPLOYEE WHERE SALARY > 30000 LIMIT 0, 1000	9 row(s) returned
11	08:49:13	SELECT CUSTOMER.CUSTOMER_NAME FROM CUSTOMER LEFT JOIN ISSUESTATUS ON CUSTOMER.CUSTOMER_ID = ISSUESTATUS.ISSUED	5 row(s) returned
12	09:59:32	SELECT BRANCH_NO, COUNT(*) AS TOTAL_EMPLOYEES FROM EMPLOYEE GROUP BY BRANCH_NO;	6 row(s) returned

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;”*



The screenshot shows the MySQL Workbench interface. In the top navigation bar, 'Local instance MySQL80' is selected. The 'Query' tab is active. The 'Result Grid' pane displays the query results:

CUSTOMER_NAME
DEVAN KURUP

The 'Output' pane at the bottom shows the execution history:

Action	Time	Action	Message
8 08:36:12	SELECT CATEGORY,COUNT(*) AS NUMBER_OF_BOOKS FROM BOOKS GROUP BY CATEG...		3 rows(s) returned
9 08:39:39	SELECT EMP_NAME, POSITION FROM EMPLOYEE WHERE SALARY > 25000 LIMIT 0, 1000		15 rows(s) returned
10 08:40:01	SELECT EMP_NAME, POSITION FROM EMPLOYEE WHERE SALARY > 30000 LIMIT 0, 1000		9 rows(s) returned
11 08:49:13	SELECT CUSTOMER.CUSTOMER_NAME FROM CUSTOMER LEFT JOIN ISSUESTATUS ON C...		5 rows(s) returned
12 09:59:32	SELECT BRANCH_NO,COUNT(*) AS TOTAL_EMPLOYEES FROM EMPLOYEE GROUP BY BRA...		6 rows(s) returned
13 10:07:56	SELECT CUSTOMER.CUSTOMER_NAME FROM CUSTOMER INNER JOIN ISSUESTATUS ON ...		1 row(s) returned

*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%';”*

The screenshot shows the MySQL Workbench interface with several tabs open. The main SQL editor tab contains the following code:

```

217 •    SELECT CUSTOMER.CUSTOMER_NAME FROM CUSTOMER LEFT JOIN ISSUESTATUS ON CUSTOMER.CUSTOMER_ID = ISSUESTATUS.ISSUED
218
219 •    SELECT BRANCH_NO, COUNT(*) AS TOTAL_EMPLOYEES FROM EMPLOYEE GROUP BY BRANCH_NO;
220
221 •    SELECT CUSTOMER.CUSTOMER_NAME FROM CUSTOMER INNER JOIN ISSUESTATUS ON CUSTOMER.CUSTOMER_ID = ISSUESTATUS.ISSUED
222
223 •    SELECT BOOK_TITLE FROM BOOKS WHERE BOOK_TITLE LIKE '%TALEN%';

```

The Result Grid shows the output for the last query:

BOOK_TITLE
TALE OF A PLACE
TALE OF A STREET

The bottom pane shows the history of actions:

Action	Time	Message
SELECT BRANCH_NO, COUNT(*) AS TOTAL_EMPLOYEES FROM EMPLOYEE GROUP BY BRANCH_NO;	12 09:59:32	6 row(s) returned
SELECT CUSTOMER.CUSTOMER_NAME FROM CUSTOMER INNER JOIN ISSUESTATUS ON CUSTOMER.CUSTOMER_ID = ISSUESTATUS.ISSUED	13 10:07:56	1 row(s) returned
SELECT BOOK_TITLE FROM BOOKS WHERE BOOK_TITLE LIKE '%TALEN%'	14 10:15:45	3 row(s) returned
SELECT BRANCH_NO, COUNT(*) AS TOTAL_EMPLOYEES FROM EMPLOYEE GROUP BY BRANCH_NO;	15 10:16:46	3 row(s) returned
SELECT BOOK_TITLE FROM BOOKS WHERE BOOK_TITLE LIKE '%NIGHT%' AND '%TALES%' LI..	16 10:18:10	0 row(s) returned
SELECT BOOK_TITLE FROM BOOKS WHERE BOOK_TITLE LIKE '%TALEN%'	17 10:19:28	2 row(s) returned
SELECT BRANCH_NO, COUNT(*) AS NUMBER_OF_EMPLOYEES FROM EMPLOYEE GROUP BY BRANCH_NO HAVING COUNT(*)>5;	18 10:28:13	3 row(s) returned

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;”*

The screenshot shows the MySQL Workbench interface with several tabs open. The main SQL editor tab contains the following code:

```

219 •    SELECT BRANCH_NO, COUNT(*) AS TOTAL_EMPLOYEES FROM EMPLOYEE GROUP BY BRANCH_NO;
220
221 •    SELECT CUSTOMER.CUSTOMER_NAME FROM CUSTOMER INNER JOIN ISSUESTATUS ON CUSTOMER.CUSTOMER_ID = ISSUESTATUS.ISSUED
222
223 •    SELECT BOOK_TITLE FROM BOOKS WHERE BOOK_TITLE LIKE '%TALEN%';
224
225 •    SELECT BRANCH_NO, COUNT(*) AS NUMBER_OF_EMPLOYEES FROM EMPLOYEE GROUP BY BRANCH_NO HAVING COUNT(*)>5;

```

The Result Grid shows the output for the last query:

BRANCH_NO	NUMBER_OF_EMPLOYEES
1	8
4	8
6	8

The bottom pane shows the history of actions:

Action	Time	Message
SELECT CUSTOMER.CUSTOMER_NAME FROM CUSTOMER INNER JOIN ISSUESTATUS ON ...	13 10:07:56	1 row(s) returned
SELECT BOOK_TITLE FROM BOOKS WHERE BOOK_TITLE LIKE '%TALEN%'	14 10:15:45	3 row(s) returned
SELECT BRANCH_NO, COUNT(*) AS TOTAL_EMPLOYEES FROM EMPLOYEE GROUP BY BRANCH_NO;	15 10:16:46	3 row(s) returned
SELECT BOOK_TITLE FROM BOOKS WHERE BOOK_TITLE LIKE '%NIGHT%' AND '%TALES%' LI..	16 10:18:10	0 row(s) returned
SELECT BOOK_TITLE FROM BOOKS WHERE BOOK_TITLE LIKE '%TALEN%'	17 10:19:28	2 row(s) returned
SELECT BRANCH_NO, COUNT(*) AS NUMBER_OF_EMPLOYEES FROM EMPLOYEE GROUP BY BRANCH_NO HAVING COUNT(*)>5;	18 10:28:13	3 row(s) returned

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';*”

BRANCHMANAGER	BRANCH_ADDRESS
CHANDRASEKHER	PATTOM,THIRUVANANTHAPRAM
SHEENA DAVIS	VANCYFOOR, THIRUVANANTHAPURAM
AMAL SHAH	NEMOM, THIRUVANANTHAPRAM
LEKHA KURJAN	KAKKANAD, KOCHI
JISHA NAIR	ALUVA, KOCHI
MAHEENDAR MOHAN	KADAPPAKKADA, KOLLAM

Action Output

#	Time	Action	Message
15	10:16:46	SELECT BOOK_TITLE FROM BOOKS WHERE CATEGORY LIKE '%AUTOBIOGRAPHY%' LIMIT 0, 3	3 row(s) returned
16	10:18:10	SELECT BOOK_TITLE FROM BOOKS WHERE BOOK_TITLE LIKE '%NIGHT%' AND '%TALE%' LI...	0 row(s) returned
17	10:19:28	SELECT BOOK_TITLE FROM BOOKS WHERE BOOK_TITLE LIKE '%TALE%' LIMIT 0, 1000	2 row(s) returned
18	10:28:13	SELECT BRANCH_NO,COUNT(*)AS NUMBER_OF_EMPLOYEES FROM EMPLOYEE GROUP BY BR...	3 row(s) returned
19	10:38:05	SELECT EMPLOYEE.EMP_NAME, BRANCH.BRANCH_ADDRESS FROM EMPLOYEE INNER JOI...	6 row(s) returned
20	10:38:43	SELECT EMPLOYEE.EMP_NAME AS BRANCHMANAGER, BRANCH.BRANCH_ADDRESS FRO...	6 row(s) returned

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;*”

The screenshot shows the MySQL Workbench interface with the following details:

- Navigator:** Shows the schema tree with databases like d32, library, organisation, product, sales, school, and sys.
- SQL Editor:** Contains several SQL queries numbered 223 to 229. The visible part of the code includes:
 

```

223 • SELECT BOOK_TITLE FROM BOOKS WHERE BOOK_TITLE LIKE 'TALE%';
224
225 • SELECT BRANCH_NO, COUNT(*) AS NUMBER_OF_EMPLOYEES FROM EMPLOYEE GROUP BY BRANCH_NO HAVING COUNT(*)>5;
226
227 • SELECT EMPLOYEE.EMP_NAME AS BRANCHMANAGER, BRANCH.BRANCH_ADDRESS FROM EMPLOYEE INNER JOIN BRANCH ON EMPLOYEE.BRANCH_NO = BRANCH.BRANCH_ID;
228
229 • SELECT DISTINCT CUSTOMER.CUSTOMER_NAME FROM ISSUETRACK INNER JOIN CUSTOMER ON ISSUETRACK.ISSUED_CUST = CUSTOMER.CUSTOMER_ID WHERE ISSUETRACK.RENT > 25;
      
```
- Result Grid:** Displays the results of the last query, listing customer names:
 

CUSTOMER_NAME
DEVAN KURUP
JATIN RAMDAS
HASHIM ALI
NIDHIL SANKAR
NIRUPAMA SANKAR
VIJAYAN
PRAVATHY KALIDAS
SREELAKSHMI
KOYA SAHAB
KURUKOSE
- Action Output:** Shows the history of actions taken in the session:
 

#	Time	Action	Message
17	10:19:28	SELECT BOOK_TITLE FROM BOOKS WHERE BOOK_TITLE LIKE 'TALE%' LIMIT 0, 1000	2 row(s) returned
18	10:28:13	SELECT BRANCH_NO, COUNT(*) AS NUMBER_OF_EMPLOYEES FROM EMPLOYEE GROUP BY BRANCH_NO HAVING COUNT(*)>5;	3 row(s) returned
19	10:38:05	SELECT EMPLOYEE.EMP_NAME AS BRANCHMANAGER, BRANCH.BRANCH_ADDRESS FROM EMPLOYEE INNER JOIN BRANCH ON EMPLOYEE.BRANCH_NO = BRANCH.BRANCH_ID;	6 row(s) returned
20	10:38:43	SELECT CUSTOMER.CUSTOMER_NAME FROM ISSUETRACK INNER JOIN CUSTOMER ON ISSUETRACK.ISSUED_CUST = CUSTOMER.CUSTOMER_ID WHERE ISSUETRACK.RENT > 25;	14 row(s) returned
21	10:48:52	SELECT DISTINCT CUSTOMER.CUSTOMER_NAME FROM ISSUETRACK INNER JOIN CUSTOMER ON ISSUETRACK.ISSUED_CUST = CUSTOMER.CUSTOMER_ID WHERE ISSUETRACK.RENT > 25;	10 row(s) returned
22	10:49:37	SELECT DISTINCT CUSTOMER.CUSTOMER_NAME FROM ISSUETRACK INNER JOIN CUSTOMER ON ISSUETRACK.ISSUED_CUST = CUSTOMER.CUSTOMER_ID WHERE ISSUETRACK.RENT > 25;	10 row(s) returned

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.