

Experiment 1.1: Library Management System using SQL

AIM

To design and implement a Library Management System database using SQL with appropriate tables, constraints, and relationships, and to perform DDL, DML, and DCL operations for secure database management.

OBJECTIVE

- To understand database creation and table design
- To implement primary keys, foreign keys, and constraints
- To perform insert, select, and delete operations
- To create database users and manage privileges using GRANT and REVOKE

PRACTICAL / EXPERIMENT STEPS

1. Create and select the database
2. Create required tables with constraints
3. Insert records into tables
4. Retrieve data using SELECT queries
5. Create a database user
6. Grant and revoke privileges
7. Verify database operations

PROCEDURE OF THE PRACTICAL

- i. Start the system and open MySQL
- ii. Create and use the database
- iii. Write SQL commands for table creation
- iv. Insert sample records
- v. Execute queries to view data
- vi. Create a database user
- vii. Grant and revoke privileges
- viii. Verify the results

SQL PROGRAM

```
USE library_db;
DROP TABLE IF EXISTS BOOK_ISSUE;
DROP TABLE IF EXISTS LIBRARY_VISITORS;
DROP TABLE IF EXISTS BOOK_S;
CREATE TABLE BOOK_S (
    BOOK_ID INT PRIMARY KEY,
    BOOK_NAME VARCHAR(20) NOT NULL,
    AUTHOR_NAME VARCHAR(20) NOT NULL,
    BOOK_COUNT INT NOT NULL CHECK (BOOK_COUNT > 0)
```

```

);
INSERT INTO BOOK_S VALUES
(101, 'HARRY POTTER', 'DAVID', 3);

```

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

BOOK_ID	BOOK_NAME	AUTHOR_NAME	BOOK_COUNT
101	HARRY POTTER	DAVID	3

BOOK_S 1 LIBRARY_VISITORS 2 BOOK_ISSUE 3 Result 4 BOOK_S 5 BOOK_ISSUE 6 LIBRARY_VISITORS 7 Apply Revert

```

CREATE TABLE LIBRARY_VISITORS (
    USER_ID INT PRIMARY KEY,
    NAME VARCHAR(20) NOT NULL,
    AGE INT NOT NULL CHECK (AGE >= 17),
    EMAIL VARCHAR(50) NOT NULL UNIQUE
);
INSERT INTO LIBRARY_VISITORS VALUES
(501, 'VANSH SHARMA', 18, 'vansh08@gmail.com'),
(502, 'VANI', 19, 'vani08@gmail.com');

```

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

USER_ID	NAME	AGE	EMAIL
501	VANSH SHARMA	18	vansh08@gmail.com
502	VANI	19	vani08@gmail.com

BOOK_S 1 LIBRARY_VISITORS 2 BOOK_ISSUE 3 Result 4 BOOK_S 5 BOOK_ISSUE 6 LIBRARY_VISITORS 7 Apply Revert

```

CREATE TABLE BOOK_ISSUE (
    BOOK_ISSUE_ID INT PRIMARY KEY,
    USER_ID INT NOT NULL,
    BOOK_ID INT NOT NULL,
    ISSUE_DATE DATE,
    FOREIGN KEY (USER_ID) REFERENCES LIBRARY_VISITORS(USER_ID),
    FOREIGN KEY (BOOK_ID) REFERENCES BOOK_S(BOOK_ID)
);
INSERT INTO BOOK_ISSUE VALUES
(1001, 501, 101, '2026-01-08');

```

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

BOOK_ISSUE_ID	USER_ID	BOOK_ID	ISSUE_DATE
1001	501	101	2026-01-08
NULL	NULL	NULL	NULL

BOOK_S 1 LIBRARY_VISITORS 2 BOOK_ISSUE 3 Result 4 BOOK_S 5 BOOK_ISSUE 6 LIBRARY_VISITORS 7 Apply Revert

```
CREATE USER IF NOT EXISTS 'LIBRARIAN'@'localhost'
IDENTIFIED BY 'ShivamDBMS3819';
GRANT SELECT, INSERT, UPDATE, DELETE
ON library_db.BOOK_S TO 'LIBRARIAN'@'localhost';
GRANT SELECT, INSERT, UPDATE, DELETE
ON library_db.BOOK_ISSUE TO 'LIBRARIAN'@'localhost';
GRANT SELECT, INSERT, UPDATE, DELETE
ON library_db.LIBRARY_VISITORS TO 'LIBRARIAN'@'localhost';
```

Result Grid Filter Rows: Search Export: Result Grid

CURRENT_USER()
root@localhost

BOOK_S 1 LIBRARY_VISITORS 2 BOOK_ISSUE 3 Result 4 BOOK_S 5 BOOK_ISSUE 6 LIBRARY_VISITORS 7 Read Only

```
REVOKE SELECT, INSERT, UPDATE, DELETE
ON library_db.BOOK_S FROM 'LIBRARIAN'@'localhost';
```

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

BOOK_ID	BOOK_NAME	AUTHOR_NAME	BOOK_COUNT
101	HARRY POTTER	DAVID	3
NULL	NULL	NULL	NULL

BOOK_S 1 LIBRARY_VISITORS 2 BOOK_ISSUE 3 Result 4 BOOK_S 5 BOOK_ISSUE 6 LIBRARY_VISITORS 7 Apply Revert

```
REVOKE SELECT, INSERT, UPDATE, DELETE
ON library_db.BOOK_ISSUE FROM 'LIBRARIAN'@'localhost';
```

```
REVOKE SELECT, INSERT, UPDATE, DELETE  
ON library_db.LIBRARY_VISITORS FROM 'LIBRARIAN'@'localhost';
```

```
DROP TABLE BOOK_ISSUE;
```

INPUT / OUTPUT ANALYSIS

Input: SQL commands for creating tables, inserting records, and managing users.

Output: Successfully created tables, inserted records, and managed user privileges.

LEARNING OUTCOME

Through this experiment, students learned:

- Database schema design
 - Use of constraints and relationships
 - Execution of DDL, DML, and DCL commands
 - Role-based access control in databases