

**Name:** Kamal Sharma

**UID:** 24BAI70380

**Course:** BE-CSE (AI&ML)

**Subject:** Database Management System

---

# Experiment: Library Management System Implementation

## 1. Aim of the Session

The aim of this lab session was to design and implement a relational database for a Library Management System. The task involved building structured tables, defining relationships among entities, and enforcing role-based security to ensure controlled access.

## 2. Objective of the Session

By the end of the session, the following objectives were achieved:

- Developed relational tables with **Primary Keys, Foreign Keys, and Check Constraints** to ensure data accuracy.
- Practiced **DML operations** (INSERT, UPDATE, DELETE, SELECT) for managing records.
- Applied **DCL commands** to configure role-based permissions for secure access.
- Maintained **referential integrity** across tables such as BOOKS, LIBRARY\_VISITORS, and BOOK\_ISSUE.

## 3. Practical / Experiment Steps

The work was carried out through the following activities:

- 1. Schema Design:** Created tables for books and visitors with constraints like NOT NULL, UNIQUE, and CHECK (e.g., minimum age requirement).
- 2. Relationship Setup:** Built the BOOK\_issue table linking BOOKS and LIBRARY\_VISITORS using foreign keys.
- 3. Data Insertion:** Added sample records to validate schema and constraints.

**4.Functional Testing:** Performed update and delete operations to check cascading and referential rules.

**5.Security Setup:** Created a role with login privileges and controlled access using GRANT and REVOKE.

## 4. Procedure of the Practical

Execution was performed in the following order:

1. **Environment Setup:** Logged into DBMS interface and accessed the server instance.
2. **Database Setup:** Created a dedicated database for the library system.
3. **Schema Execution:** Executed CREATE TABLE commands ensuring parent tables were defined first.
4. **Data Entry Phase:** Inserted records into BOOKS and LIBRARY\_VISITORS.
5. **Verification Queries:** Verified data using SELECT queries
6. **Update/Delete Checks:** Tested mutability with UPDATE and DELETE.
7. **Role Creation:** Created a librarian role and assigned relevant operations through DCL.
8. **Permission Testing:** Validated access control by revoking permissions and checking role privileges.
9. **Documentation:** Saved final SQL script and captured outputs for reporting.

## 5. I/O Analysis (Input / Output Analysis)

### Input Queries

SQL

```
CREATE TABLE BOOKS (  
  
BOOK_ID INT PRIMARY KEY,  
  
NAME VARCHAR(20) NOT NULL,  
  
AUTHOR_NAME VARCHAR(20) ,  
  
BOOK_COUNT INT CHECK(BOOK_COUNT>0) NOT NULL  
  
)  
  
CREATE TABLE LIBRARY_VISITORS (  
  
USER_ID INT PRIMARY KEY,  
  
NAME VARCHAR(20) NOT NULL,  
  
AGE INT CHECK(AGE>=17) NOT NULL,
```

EMAIL VARCHAR(30) NOT NULL UNIQUE

)

CREATE TABLE BOOK\_ISSUE(

BOOK\_ISSUE\_ID INT PRIMARY KEY,

USER\_ID INT NOT NULL,

BOOK\_ID INT NOT NULL,

ISSUE\_DATE DATE NOT NULL,

FOREIGN KEY(USER\_ID) REFERENCES LIBRARY\_VISITORS(USER\_ID),

FOREIGN KEY(BOOK\_ID) REFERENCES BOOKS(BOOK\_ID)

)

INSERT INTO BOOKS VALUES(101, 'STAR WARS', 'DAVID', 5)

INSERT INTO BOOKS VALUES(102, 'DEMON', 'JAMES', 8)

SELECT \* FROM BOOKS

INSERT INTO LIBRARY\_VISITORS(USER\_ID, NAME, AGE, EMAIL)

VALUES(501, 'SHIVAM SINGH', 20, 'SHIVAM18@GMAIL.COM')

INSERT INTO LIBRARY\_VISITORS(USER\_ID, NAME, AGE, EMAIL)

VALUES(502, 'SAM SHARMA', 20, 'SAM18@GMAIL.COM')

SELECT \* FROM LIBRARY\_VISITORS

INSERT INTO BOOK\_ISSUE VALUES(10001, 501, 101, '2026-01-08')

INSERT INTO BOOK\_ISSUE VALUES(10002, 502, 102, '2026-01-09')

SELECT \* FROM BOOK\_ISSUE

UPDATE BOOK\_ISSUE

SET ISSUE\_DATE='2026-02-09'

```
WHERE BOOK_ISSUE_ID=10001
```

```
SELECT * FROM BOOK_ISSUE
```

```
DELETE FROM BOOK_ISSUE WHERE BOOK_ISSUE_ID=10002
```

```
SELECT * FROM BOOK_ISSUE
```

```
CREATE ROLE LIBRARIAN WITH LOGIN PASSWORD 'PASSWORD'
```

```
GRANT SELECT, INSERT, DELETE, UPDATE ON BOOKS TO LIBRARIAN
```

```
GRANT SELECT, INSERT, DELETE, UPDATE ON LIBRARY_VISITORS TO LIBRARIAN
```

```
GRANT SELECT, INSERT, DELETE, UPDATE ON BOOK_ISSUE TO LIBRARIAN
```

```
REVOKE SELECT, INSERT, DELETE, UPDATE ON BOOKS, LIBRARY_VISITORS, BOOK_ISSUE  
FROM LIBRARIAN
```

## Output Details

### 1. Schema Creation

- All three tables (BOOKS, LIBRARY\_VISITORS, and BOOK\_ISSUE) were successfully created.
- The PRIMARY KEY constraints ensured unique identification of books and visitors.
- The CHECK(age>=18) constraint prevented entries of visitors below 18 years of age.
- The CHECK(count>0) constraint disallowed non-positive values for book count.
- FOREIGN KEY constraints ensured that book issue entries could only reference existing books and visitors.

✓ Result: Schema creation completed without errors.

### 2. DML Outputs

The following SQL commands executed successfully:

visitor insertion :

```
INSERT INTO LIBRARY_VISITORS(USER_ID,NAME,AGE,EMAIL)
VALUES(501,'SHIVAM SINGH',20,'SHIVAM18@GMAIL.COM')
```

```
INSERT INTO LIBRARY_VISITORS(USER_ID,NAME,AGE,EMAIL)
VALUES(502,'SAM SHARMA',20,'SAM18@GMAIL.COM')
```

	user_id [PK] integer	name character varying (20)	age integer	email character varying (30)
1	501	SHIVAM SINGH	20	SHIVAM18@GMAIL.CO...
2	502	SAM SHARMA	20	SAM18@GMAIL.COM

Book Issue Entry :

```
INSERT INTO BOOK_ISSUE VALUES(10001,501,101,'2026-01-08')
```

```
INSERT INTO BOOK_ISSUE VALUES(10002,502,102,'2026-01-09')
```

	book_issue_id [PK] integer	user_id integer	book_id integer	issue_date date
1	10001	501	101	2026-02-08
2	10002	502	102	2026-01-09

Update Book Issue Entry :

```
UPDATE BOOK_ISSUE
SET ISSUE_DATE='2026-02-09'
WHERE BOOK_ISSUE_ID=10001
```

	book_issue_id [PK] integer	user_id integer	book_id integer	issue_date date
1	10002	502	102	2026-01-09
2	10001	501	101	2026-02-09

### 3. DELETE Operation Result

```
DELETE FROM BOOK_ISSUE WHERE BOOK_ISSUE_ID=10002
```

	book_issue_id [PK] integer	user_id integer	book_id integer	issue_date date
1	10001	501	101	2026-02-09

### 4. DCL (Security / Role-Based Access Control) Output

- **DCL Verification:** The `librarian` role was successfully created and assigned the necessary privileges for library management tasks.

```
CREATE ROLE LIBRARIAN WITH LOGIN PASSWORD 'PASSWORD'

GRANT SELECT, INSERT, DELETE, UPDATE ON BOOKS TO LIBRARIAN

GRANT SELECT, INSERT, DELETE, UPDATE ON LIBRARY_VISITORS TO LIBRARIAN

GRANT SELECT, INSERT, DELETE, UPDATE ON BOOK_ISSUE TO LIBRARIAN

REVOKE SELECT, INSERT, DELETE, UPDATE ON BOOKS,LIBRARY_VISITORS,BOOK_ISSUE FROM LIBRARIAN
```

Data Output Messages Notifications

GRANT

Query returned successfully in 38 msec.

- **Validation:** Testing confirmed that after the `REVOKE` command, the `librarian` could no longer perform operations on the `books` table, ensuring the security policy is functional.

```
ERROR:  permission denied for table book_issue  
  
SQL state: 42501
```

## 6. Learning Outcome

From this practical, the following knowledge and skills were gained:

- **Schema Design Insight:** Learned how relational constraints like `CHECK`, `UNIQUE`, and `FOREIGN KEY` contribute to logical data consistency.
- **Database Security Skills:** Understood the advantage of assigning roles instead of individual user permissions for scalable security.
- **Real-world Contextualization:** Saw how SQL is applied in real applications (such as library systems) where multiple entities interact systematically.