Experiment 1 – Library Management System Database

Experiment

Experiment 1: Design and implementation of a Library Management System using PostgreSQL with DDL, DML and DCL commands.

Aim

The aim of this experiment is to design and implement a Library Management System database using PostgreSQL. The database is created using appropriate tables, primary keys, foreign keys and constraints. Data manipulation operations are performed and database security is implemented using roles and privileges.

Objective

The objective of this experiment is to understand the use of DDL, DML and DCL commands in PostgreSQL. It also helps in learning role-based access control by creating roles and assigning permissions.

Practical / Experiment Steps

1. Create tables for book details, library users and book transactions.
2. Apply primary keys, foreign keys and check constraints.
3. Insert sample records into the tables.
4. Display records using SELECT queries.
5. Create a database role named **LIBRARIAN_ROLE**.
6. Grant required permissions to the role.

7. Revoke permissions when required to ensure database security.

Procedure of the Experiment

1. Open pgAdmin and connect to the PostgreSQL server.
2. Create a new database for the Library Management System.
3. Create tables **BOOK_DETAILS**, **LIBRARY_USERS**, and **BOOK_TRANSACTIONS** using CREATE TABLE command.
4. Define primary keys and foreign keys while creating tables.
5. Insert records into tables using INSERT command.
6. Retrieve data using SELECT command.
7. Create a role named **LIBRARIAN_ROLE** using CREATE ROLE command.
8. Grant SELECT, INSERT, UPDATE and DELETE permissions to the role.
9. Revoke INSERT, UPDATE and DELETE permissions from BOOK_DETAILS table.
10. Execute all queries and verify the output.

Input / Output Details

Input

SQL commands for table creation, insertion of records, selection of data, role creation, grant and revoke permissions.

Output

Successful creation of tables, insertion and display of records, creation of role, granting and revoking of permissions.

Screenshots

Table Creation

```
class=# CREATE TABLE BOOK_DETAILS (
class(#     BOOK_ID INT PRIMARY KEY,
class(#     TITLE VARCHAR(30) NOT NULL,
class(#     AUTHOR VARCHAR(30) NOT NULL,
class(#     COPIES_AVAILABLE INT CHECK (COPIES_AVAILABLE >= 1)
class(# );
CREATE TABLE
class=#
class=# INSERT INTO BOOK_DETAILS (BOOK_ID, TITLE, AUTHOR, COPIES_AVAILABLE)
class-# VALUES (1, 'Harry Potter', 'J K Rowling', 3);
INSERT 0 1
class=#
class=# SELECT * FROM BOOK_DETAILS;
 book_id |    title     |   author    | copies_available
---------+--------------+-------------+------------------
       1 | Harry Potter | J K Rowling |                3
(1 row)
```

DML Operations

```
class=# CREATE TABLE LIBRARY_USERS (
class(#     USER_ID INT PRIMARY KEY,
class(#     FULL_NAME VARCHAR(25) NOT NULL,
class(#     AGE INT NOT NULL CHECK (AGE >= 17),
class(#     EMAIL_ID VARCHAR(40) UNIQUE
class(# );
CREATE TABLE
class=#
class=# INSERT INTO LIBRARY_USERS
class-# VALUES (101, 'Akash Sharma', 17, 'a@gmail.com');
INSERT 0 1
class=#
class=# SELECT * FROM LIBRARY_USERS;
 user_id |  full_name   | age |  email_id
---------+--------------+-----+-------------
     101 | Akash Sharma |  17 | a@gmail.com
(1 row)
```

SELECT Output

```
class=# CREATE TABLE BOOK_TRANSACTIONS (
class(#     ISSUE_ID INT PRIMARY KEY,
class(#     BOOK_ID INT NOT NULL,
class(#     USER_ID INT NOT NULL,
class(#     ISSUE_DATE DATE NOT NULL,
class(#     CONSTRAINT FK_BOOK_ID
class(#         FOREIGN KEY (BOOK_ID) REFERENCES BOOK_DETAILS(BOOK_ID),
class(#     CONSTRAINT FK_USER_ID
class(#         FOREIGN KEY (USER_ID) REFERENCES LIBRARY_USERS(USER_ID)
class(# );
CREATE TABLE
class=#
class=# INSERT INTO BOOK_TRANSACTIONS
class-# VALUES (5552, 1, 101, '2026-01-09');
INSERT 0 1
class=#
class=# SELECT * FROM BOOK_TRANSACTIONS;
 issue_id | book_id | user_id | issue_date
----------+---------+---------+------------
     5552 |       1 |     101 | 2026-01-09
(1 row)
```

Role Creation and Grant

```
class=# GRANT SELECT, INSERT, UPDATE, DELETE
class-# ON BOOK_DETAILS, BOOK_TRANSACTIONS, LIBRARY_USERS
class-# TO LIBRARIAN_ROLE;
GRANT
class=#
class=# REVOKE INSERT, UPDATE, DELETE
class-# ON BOOK_DETAILS
class-# FROM LIBRARIAN_ROLE;
REVOKE
class=#
```

Revoke Permissions

```
class=# GRANT SELECT, INSERT, UPDATE, DELETE
class-# ON BOOK_DETAILS, BOOK_TRANSACTIONS, LIBRARY_USERS
class-# TO LIBRARIAN_ROLE;
GRANT
class=#
class=# REVOKE INSERT, UPDATE, DELETE
class-# ON BOOK_DETAILS
class-# FROM LIBRARIAN_ROLE;
REVOKE
class=#
```

Learning Outcome

After completing this experiment, the student learned how to design a database using tables and constraints. The student also understood the use of DDL, DML and DCL commands in PostgreSQL and gained practical knowledge of role-based access control.