Experiment 1.1

Student Name: Diwansh
UID: 24BAI70294
Branch: CSE - AIML
Section/Group: 24AIT_KRG G1
Semester: 4
Date of Performance: 9/1/26
Subject Name: DBMS
Subject Code: 24CSH-298

Aim

To design and implement a Library Management System database using appropriate tables, primary keys, foreign keys, and constraints, and to perform DML operations along with DCL commands such as role creation, privilege granting, and revoking to ensure database security.

Software Requirements

- Database Management System:
  - PostgreSQL
- Database Administration Tool:
  - pgAdmin

Objectives

To gain practical experience in implementing Data Definition Language (DDL), Data Manipulation Language (DML), and Data Control Language (DCL) operations in a real database environment. This will also include implementing role-based privileges to secure data.

Problem Statement

- A Library wants to develop a Library Management System database to manage information about books, members, and book issue records efficiently. The database should be designed using appropriate tables, primary keys, foreign keys, and constraints to ensure data integrity.

- The system must support basic database operations such as inserting records, updating existing data, and deleting obsolete entries. To ensure database security.
- To ensure database security, a database role named Librarian must be created. This role should be password protected and granted SELECT, INSERT, and DELETE permissions on the required tables. The system administrator (pgAdmin) should also have the ability to revoke these permissions when required using role-based access control.

Code

```
CREATE TABLE BOOKS(
BOOK_ID INT PRIMARY KEY,
BOOK_NAME VARCHAR(20) NOT NULL,
AUTHOR_NAME VARCHAR(20) NOT NULL
)

SELECT * FROM BOOKS

ALTER TABLE BOOKS
ADD BOOK_COUNT INT CHECK(BOOK_COUNT>0) NOT NULL

SELECT * FROM BOOKS

INSERT INTO BOOKS VALUES(101, 'Harry Potter', 'JK Rowling', 3)
INSERT INTO BOOKS VALUES(102, '1984', 'George Orwell', 5)

SELECT * FROM BOOKS

CREATE TABLE LIBRARY_VISITORS(
USER_ID INT PRIMARY KEY,
NAME VARCHAR(20) NOT NULL,
AGE INT CHECK(AGE>=17) NOT NULL,
EMAIL VARCHAR(20) NOT NULL UNIQUE
)

SELECT * FROM LIBRARY_VISITORS

INSERT INTO LIBRARY_VISITORS(USER_ID, NAME, AGE, EMAIL)
VALUES(101, 'Vansh Sharma', 18, 'vansh12@gmail.com')
```

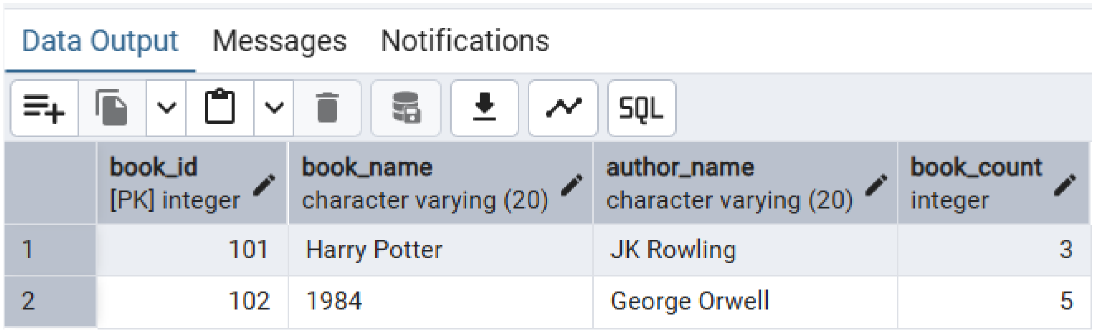
```
DROP TABLE BOOK_ISSUE
CREATE TABLE BOOK_ISSUE(
BOOK_ISSUE_ID INT PRIMARY KEY,
USER_ID INT NOT NULL,
BOOK_ID INT NOT NULL,
FOREIGN KEY(USER_ID) REFERENCES LIBRARY_VISITORS(USER_ID),  -
-THIS WORKS IN ORACLE
FOREIGN KEY(BOOK_ID) REFERENCES BOOKS(BOOK_ID)
)

INSERT INTO BOOK_ISSUE VALUES(10001, 501, 101)

ALTER TABLE BOOK_ISSUE
ADD ISSUE_DATE DATE

SELECT * FROM BOOK_ISSUE

INSERT INTO BOOK_ISSUE VALUES(1001, 501, 101, '2026-01-09')

UPDATE BOOK_ISSUE
SET ISSUE_DATE='2026-01-08'
WHERE BOOK_ISSUE_ID=1001


DELETE FROM BOOKS WHERE BOOK_ID=102

SELECT * FROM BOOKS

CREATE ROLE LIBRARIAN
WITH LOGIN PASSWORD 'PASSWORD12'

SELECT CURRENT_USER

GRANT SELECT, INSERT, DELETE, UPDATE ON BOOKS TO LIBRARIAN2
GRANT SELECT, INSERT, DELETE, UPDATE ON BOOK_ISSUE TO
LIBRARIAN2
```

GRANT SELECT, INSERT, DELETE, UPDATE ON LIBRARY_VISITORS TO LIBRARIAN2

REVOKE SELECT, INSERT, DELETE, UPDATE ON BOOKS, BOOK_ISSUE, LIBRARY_VISITORS FROM LIBRARIAN2

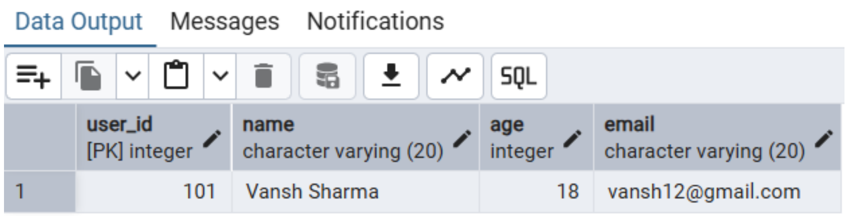
Output

Table books:



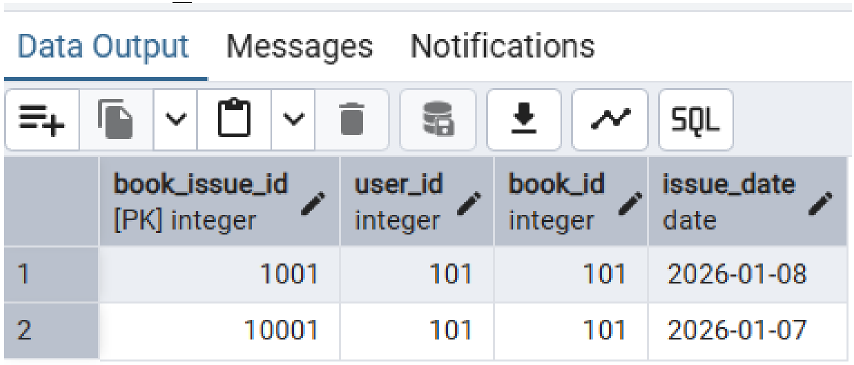
	book_id [PK] integer	book_name character varying (20)	author_name character varying (20)	book_count integer
1	101	Harry Potter	JK Rowling	3
2	102	1984	George Orwell	5

Table library_visitors:



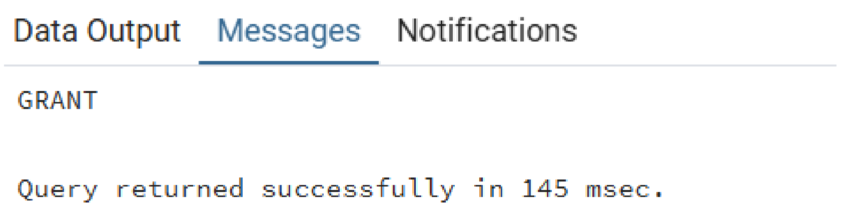
	user_id [PK] integer	name character varying (20)	age integer	email character varying (20)
1	101	Vansh Sharma	18	vansh12@gmail.com

Table book_issue:



	book_issue_id [PK] integer	user_id integer	book_id integer	issue_date date
1	1001	101	101	2026-01-08
2	10001	101	101	2026-01-07

Access granted to role – librarian:



```
GRANT

Query returned successfully in 145 msec.
```

Learning Outcomes

- Gained hands-on experience to work with PostgreSQL and pgAdmin
- Writing queries to create and delete tables
- Learnt to alter tables, view tables, create roles, granting and revoking access to the roles
- Primary and foreign keys implementations and roles