Student Name: Husanpreet Kaur **UID: 24BCY70138**
Branch: AIT-CSE (Cyber Security) **Section/Group: 24AIT_KRG2**
Semester: 4 **Date of Performance: 13/01/2026**
Subject Name: Database Management System **Subject Code: 24CSH-298**

WORKSHEET 1

AIM: To design and implement a sample database system using DDL, DML, and DCL commands, including database creation, data manipulation, schema modification, and role-based access control to ensure data integrity and secure, read-only access for authorized users.

S/W Requirement: Oracle Database Express Edition and 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.

Given:

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, 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.

1. **Database Design**

Create multiple tables such as Books, Library_Visitor, Book_Issue

Define appropriate **PRIMARY KEY** and **FOREIGN KEY** constraints.

Enforce **NOT NULL**, **UNIQUE**, and **CHECK** constraints where necessary.

Query:

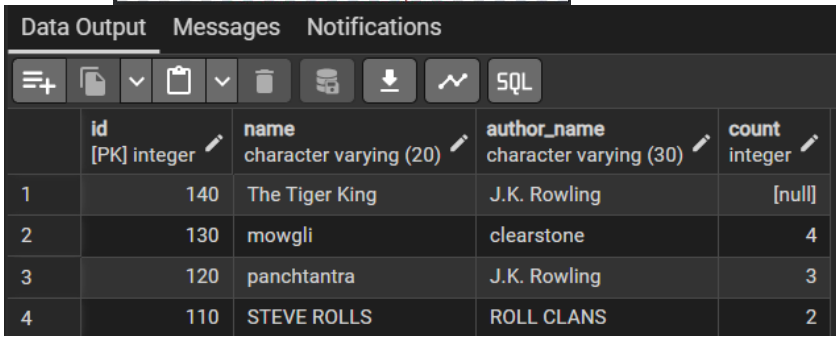
```
CREATE TABLE books (
   id INT PRIMARY KEY,
   name VARCHAR(20) NOT NULL,
   author_name VARCHAR(30) NOT NULL
);

CREATE TABLE library_visitor(
USER_ID INT PRIMARY KEY,
USER_NAME VARCHAR(20),
AGE INT CHECK(AGE>=17) NOT NULL,
EMAIL_ID VARCHAR(20) UNIQUE NOT NULL
);

CREATE TABLE book_issue(
BOOK_ISSUE_ID INT PRIMARY KEY,
BOOK_ID INT REFERENCES books(ID),
USER_ID INT REFERENCES library_visitor(USER_ID) NOT NULL,
BOOK_ISSUE DATE NOT NULL
);
```

2. **Data Manipulation**

Insert sample records into all tables.

Query:

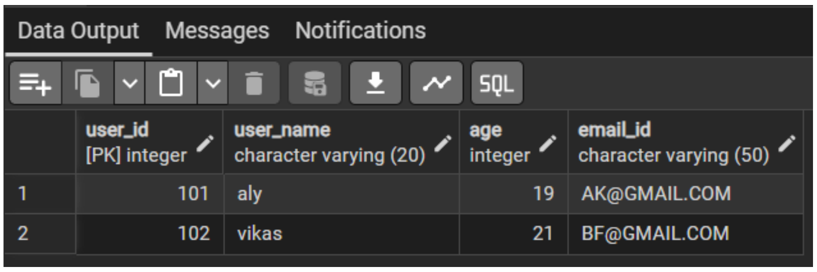
```
INSERT INTO books (id, name, author_name)
VALUES (140, ‘The Tiger King’, ‘J.K. Rowling’);
VALUES(130,’mowgli’,’clearstone’);
VALUES(120,’panchtantra’,’J.K.Rowling’);
VALUES(110,’STEVE ROLLS’,’ROLL CLANS’);
SELECT * FROM books;
```


Data Output | Messages | Notifications

	id [PK] integer	name character varying (20)	author_name character varying (30)	count integer
1	140	The Tiger King	J.K. Rowling	[null]
2	130	mowgli	clearstone	4
3	120	panchtantra	J.K. Rowling	3
4	110	STEVE ROLLS	ROLL CLANS	2

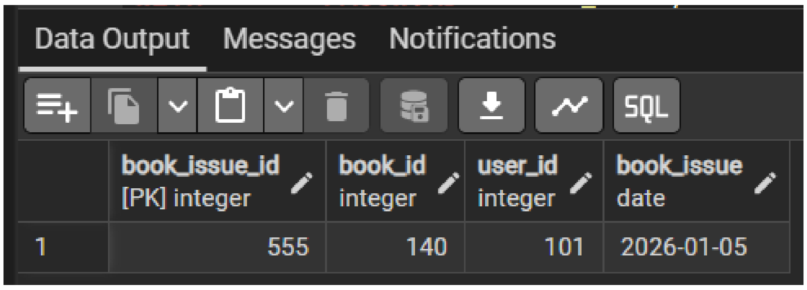
INSERT INTO library_visitor(USER_ID,USER_NAME,AGE,EMAIL_ID) VALUES(101,'aly',19,'abcd@gmail.com');

INSERT INTO library_visitor(USER_ID,USER_NAME,AGE,EMAIL_ID) VALUES(102,'aman',21,'efgh@gmail.com');


Data Output | Messages | Notifications

	user_id [PK] integer	user_name character varying (20)	age integer	email_id character varying (50)
1	101	aly	19	AK@GMAIL.COM
2	102	vikas	21	BF@GMAIL.COM

INSERT INTO book_issue VALUES(555,140,101,'05-01-2026');


Data Output | Messages | Notifications

	book_issue_id [PK] integer	book_id integer	user_id integer	book_issue date
1	555	140	101	2026-01-05

Perform **UPDATE** operations to modify existing records.

Query:

UPDATE books

SET COUNT=3 WHERE ID=120;

UPDATE library_visitor

SET USER_NAME='vikas'

WHERE USER_ID=102;

UPDATE library_visitor

SET EMAIL_ID='AK@GMAIL.COM'

WHERE USER_ID=101;

UPDATE library_visitor

SET EMAIL_ID='BF@GMAIL.COM'

WHERE USER_ID=102;

3. **Access Control & Security**

   Query:

Grant **ONLY SELECT privilege** on required tables to this role/user.

Querry:

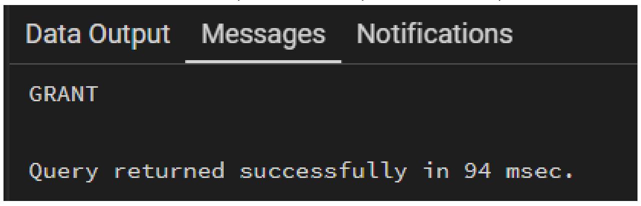
CREATE ROLE librarian

WITH LOGIN PASSWORD 'Husan_123';

GRANT SELECT, INSERT, DELETE, UPDATE ON books TO librarian;

GRANT SELECT, INSERT, DELETE, UPDATE ON book_issue TO librarian;

GRANT SELECT, INSERT, DELETE, UPDATE ON library_visitor TO librarian;

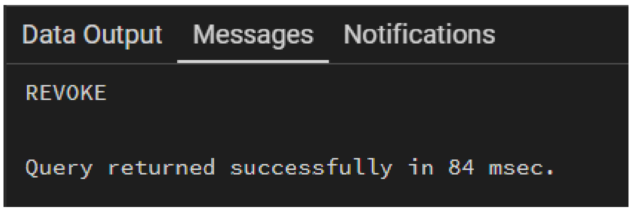


```
GRANT

Query returned successfully in 94 msec.
```

Explicitly **REVOKE CREATE privilege** so that the user cannot create any database objects.

Query:

REVOKE SELECT, INSERT, DELETE, UPDATE ON library_visitor FROM librarian;



```
REVOKE

Query returned successfully in 84 msec.
```

4. **Schema Modification**

Use **ALTER TABLE** to add or modify a column.

Query:

ALTER TABLE books

```
ADD COUNT INT CHECK(COUNT>=1);


ALTER TABLE library_visitor
ALTER COLUMN EMAIL_ID TYPE VARCHAR(50);


ALTER TABLE library_visitor
DROP COLUMN EMAIL_ID;


ALTER TABLE library_visitor
ADD COLUMN EMAIL_ID VARCHAR(30) UNIQUE;
```

Learning Outcomes:

1. Understood the basics of **relational database design** using tables, keys, and relationships.
2. Learned to apply **primary key and foreign key constraints** to maintain data integrity.
3. Gained hands-on experience with **INSERT, UPDATE, and DELETE** operations safely.
4. Understood how **roles and privileges** control access to database objects.
5. Learned to use **GRANT and REVOKE** for implementing **read-only users**.
6. Practiced **ALTER TABLE and DROP TABLE** for managing database changes.