Experiment No.: 03

Title: To implement database for relational model in Experiment no. 2 using DDL statements.

Batch:A3 **Roll No.:16010423099** **Experiment No.: 03**

Aim: To implement database for relational model in experiment no. 2 using DDL statements (Virtual Lab).

Resources needed: PostgreSQL PgAdmin3

Theory:

The Data Definition Language (DDL) is used to create and modify the relational schema. Also it is used to add various constraints to the table like the primary key, foreign key, check constraint, not null constraint and unique constraint.

The DDL statements are:

CREATE

DROP

ALTER

PostgreSQL supports the standard SQL types int, smallint, real, double precision, char(N), varchar(N), date, time, timestamp, and interval for creating tables.

Procedure:

Create Database and use it:

$ createdb mydb

$ psql mydb

Delete a database: $

dropdb mydb

Create table:

```
CREATE TABLE my_first_table (
first_column text,
second_column integer
);

CREATE TABLE products (
product_no integer,
name text, price numeric);
```

Drop Table:

```
DROP TABLE my_first_table;
DROP TABLE products;
```

Default Value:

```
CREATE TABLE products (
product_no integer,
name text, price numeric DEFAULT 9.99 );
```

Constraints:

1. **Primary Key**

```
CREATE TABLE products (
product_no integer PRIMARY KEY,
name text,
price numeric );
```

Primary keys can also constrain more than one column.

```
CREATE TABLE example (
a integer,
b integer,
c integer,
PRIMARY KEY (a, c)
);
```

2. **Check Constraint**

```
CREATE TABLE products (
product_no integer,
name text,
price numeric CHECK (price > 0) );
```

3. **Not Null Constraint**

```
CREATE TABLE products (
product_no integer NOT NULL,
name text NOT NULL,
price numeric );
```

4. **Unique Constraint**

```
CREATE TABLE products (
product_no integer UNIQUE,
name text,
price numeric );
```

5. **Foreign Key Constarint**

```
CREATE TABLE products (
product_no integer PRIMARY KEY,
name text,
price numeric );

CREATE TABLE orders (
order_id integer PRIMARY KEY,
product_no integer REFERENCES products (product_no),
quantity integer );
```

Here a foreign key constraint in the order table references the products table.

Modifying table:

Adding column
ALTER TABLE products ADD COLUMN description text;

Removing column
ALTER TABLE products DROP COLUMN description;

Adding Constraint
ALTER TABLE products ADD CONSTRAINT some_name UNIQUE (product_no); ALTER TABLE products ADD FOREIGN KEY (product_group_id) REFERENCES product_groups;

Removing Constraint
ALTER TABLE products DROP CONSTRAINT some_name;

Adding Not Null Constraint
ALTER TABLE products ALTER COLUMN product_no SET NOT NULL;

Removing Not Null Constraint
ALTER TABLE products ALTER COLUMN product_no DROP NOT NULL;

Results: (Queries printout with output)

CREATING TABLES:

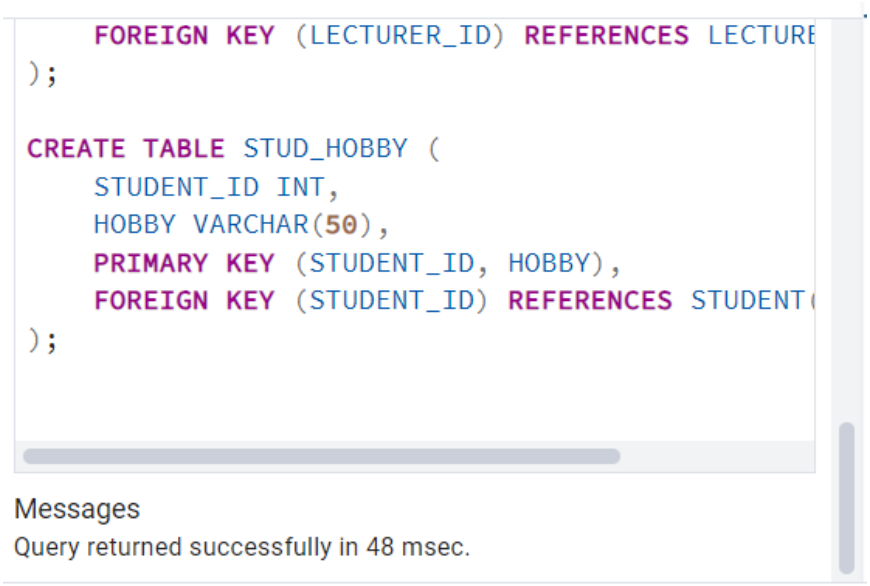
```
CREATE TABLE COURSE (
   COURSE_ID INT PRIMARY KEY,
   COURSE_NAME VARCHAR(100) NOT NULL
);

CREATE TABLE STUDENT (
   STUDENT_ID INT PRIMARY KEY,
   STUDENT_NAME VARCHAR(100) NOT NULL,
   DOB DATE,
   DOOR_NO VARCHAR(10),
   STREET VARCHAR(100),
   CITY VARCHAR(50),
   STATE VARCHAR(50),
   PIN VARCHAR(10),
   COURSE_ID INT,
   FOREIGN KEY (COURSE_ID) REFERENCES COURSE(COURSE_ID)
);

CREATE TABLE LECTURER (
   LECTURER_ID INT PRIMARY KEY,
   LECTURER_NAME VARCHAR(100) NOT NULL,
   COURSE_ID INT,
   FOREIGN KEY (COURSE_ID) REFERENCES COURSE(COURSE_ID)
);
```

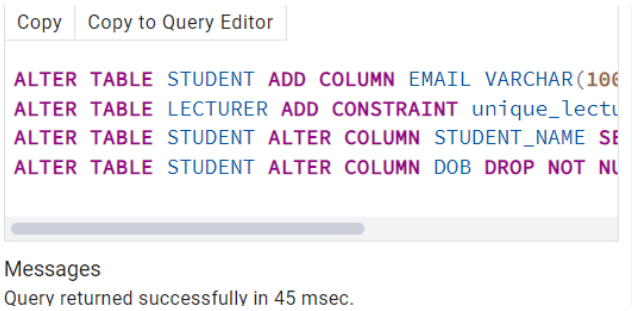
```
CREATE TABLE SUBJECT (
   SUBJECT_ID INT PRIMARY KEY,
   SUBJECT_NAME VARCHAR(100) NOT NULL,
   LECTURER_ID INT,
   FOREIGN KEY (LECTURER_ID) REFERENCES LECTURER(LECTURER_ID)
);

CREATE TABLE STUD_HOBBY (
   STUDENT_ID INT,
   HOBBY VARCHAR(50),
   PRIMARY KEY (STUDENT_ID, HOBBY),
   FOREIGN KEY (STUDENT_ID) REFERENCES STUDENT(STUDENT_ID)
);
```



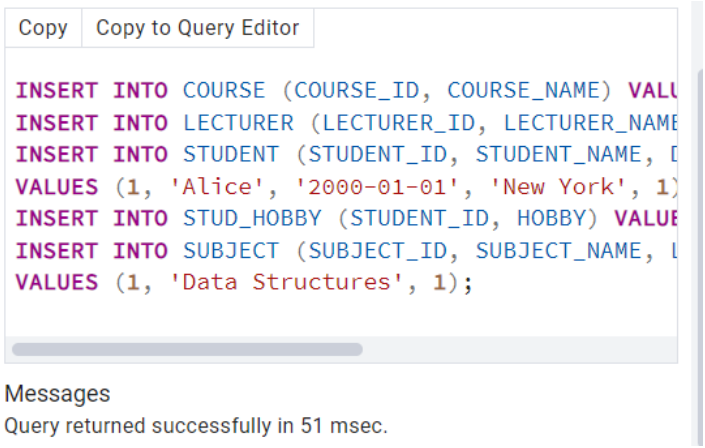
ALTERING TABLES:

```
ALTER TABLE STUDENT ADD COLUMN EMAIL VARCHAR(100);
ALTER TABLE LECTURER ADD CONSTRAINT unique_lecturer_name UNIQUE (LECTURER_NAME);
ALTER TABLE STUDENT ALTER COLUMN STUDENT_NAME SET NOT NULL;
ALTER TABLE STUDENT ALTER COLUMN DOB DROP NOT NULL;
```



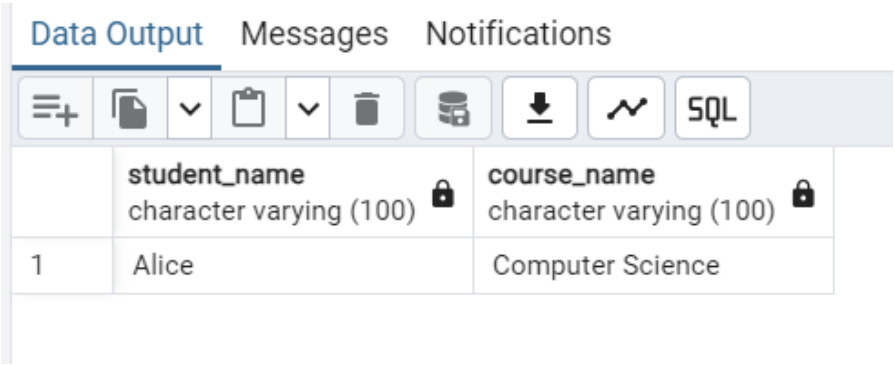
INSERTING SAMPLE DATA INTO TABLES:

INSERT INTO COURSE (COURSE_ID, COURSE_NAME) VALUES (1, 'Computer Science');
INSERT INTO LECTURER (LECTURER_ID, LECTURER_NAME, COURSE_ID) VALUES (1, 'Dr. Smith', 1);
INSERT INTO STUDENT (STUDENT_ID, STUDENT_NAME, DOB, CITY, COURSE_ID) VALUES (1, 'Alice', '2000-01-01', 'New York', 1);
INSERT INTO STUD_HOBBY (STUDENT_ID, HOBBY) VALUES (1, 'Reading');
INSERT INTO SUBJECT (SUBJECT_ID, SUBJECT_NAME, LECTURER_ID) VALUES (1, 'Data Structures', 1);



QUERYING THE DATABASE:

SELECT STUDENT_NAME, COURSE_NAME
FROM STUDENT
JOIN COURSE ON STUDENT.COURSE_ID = COURSE.COURSE_ID;



Outcomes:

CO1: Realize the features of Relational database management systems

Questions:

Q1 what is the difference between Truncate, Drop and delete? Explain with example
TRUNCATE removes all rows from a table but keeps the structure, making it faster and irreversible. DROP deletes the entire table, including its structure and data. DELETE removes specific rows based on a condition and can be rolled back.

Conclusion:
Successfully created the database in Postgres and create tables, altered tables and queried the database.

Grade: AA / AB / BB / BC / CC / CD /DD

Signature of faculty in-charge with date
