

**Experiment No.: 03**

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

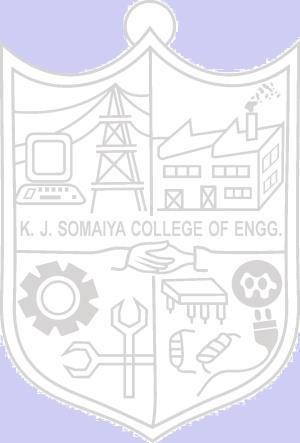
(Autonomous College Affiliated to University of Mumbai)

# Batch: B1 Roll No.: 1914078 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** );

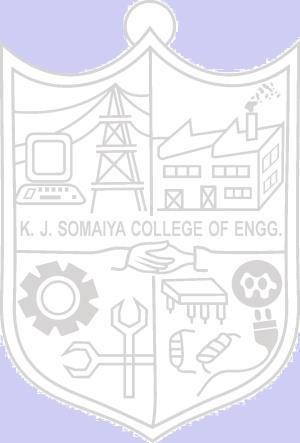
# Constraints:

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)

);

# Check Constraint

CREATE TABLE products ( product\_no integer,

name text,

price numeric **CHECK (price** > **0)** );

# Not Null Constraint

CREATE TABLE products ( product\_no integer **NOT NULL**, name text **NOT NULL**,

price numeric );

1. **Unique Constraint** CREATE TABLE products ( product\_no integer **UNIQUE**,

name text,

price numeric );

# Foreign Key Constarint

CREATE TABLE products ( product\_no integer PRIMARY KEY, name text,

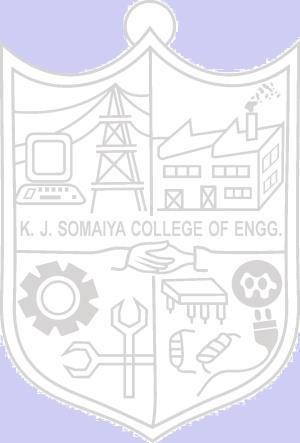
price numeric );

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)

**CONTRAINTS:**

1. **Primary key:**

CREATE TABLE employee (

ELoginID text PRIMARY KEY,

Name text UNIQUE,

Password text

);

INSERT into employee values('22', 'Devansh','abcdef');

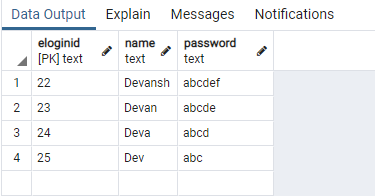
INSERT into employee values('23', 'Devan','abcde');

INSERT into employee values('24', 'Deva','abcd');

INSERT into employee values('25', 'Dev','abc');

SELECT \* from employee;

**Output:**



1. **Check constraint:**

CREATE TABLE building (

BName text PRIMARY KEY,

Photos text NOT NULL,

no\_of\_flats integer CHECK (no\_of\_flats > 200)

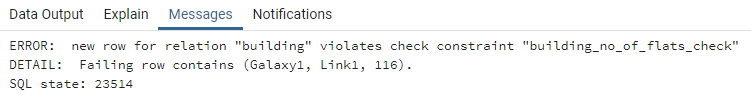
);

INSERT into building values('Galaxy2', 'Link2', 216);

INSERT into building values('Galaxy3', 'Link3', 316);

INSERT into building values('Galaxy1', 'Link1', 116);

**Output:**



1. **Not NULL constraint:**

CREATE TABLE building (

BName text PRIMARY KEY,

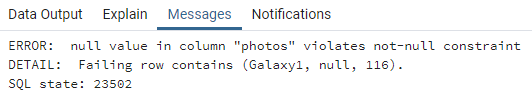
Photos text NOT NULL,

no\_of\_flats integer CHECK (no\_of\_flats > 200)

);

INSERT into building values('Galaxy1',NULL, 116);

**Output:**



1. **Unique constraint:**

CREATE TABLE employee (

ELoginID text PRIMARY KEY,

Name text UNIQUE,

Password text

);

INSERT into employee values('22', 'Devansh','abcdef');

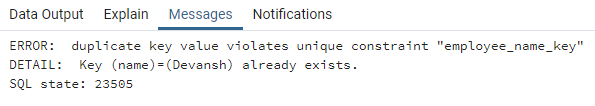
INSERT into employee values('23', 'Devan','abcde');

INSERT into employee values('24', 'Deva','abcd');

INSERT into employee values('25', 'Dev','abc');

INSERT into employee values('21', 'Devansh','abcdefg');

**Output:**



1. **Foreign key constraint:**

CREATE TABLE employee (

ELoginID text PRIMARY KEY,

Name text UNIQUE,

Password text

);

INSERT into employee values('22', 'Devansh','abcdef');

INSERT into employee values('23', 'Devan','abcde');

INSERT into employee values('24', 'Deva','abcd');

INSERT into employee values('25', 'Dev','abc');

CREATE TABLE building (

BName text PRIMARY KEY,

Photos text NOT NULL,

no\_of\_flats integer CHECK (no\_of\_flats > 200)

);

INSERT into building values('Galaxy2', 'Link2', 216);

INSERT into building values('Galaxy3', 'Link3', 316);

CREATE TABLE builder (

BLoginID text PRIMARY KEY,

Name text,

Password text,

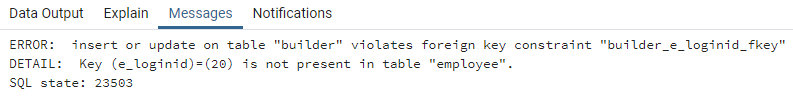
E\_LoginID text REFERENCES employee (ELoginID),

B\_name text REFERENCES building (Bname)

);

INSERT into builder values('11', 'Kevin','abcdefg', '20');

**Output:**



**MODIFYING TABLE:**

1. **Adding column:**

CREATE TABLE employee (

ELoginID text PRIMARY KEY,

Name text UNIQUE,

Password text

);

INSERT into employee values('22', 'Devansh','abcdef');

INSERT into employee values('23', 'Devan','abcde');

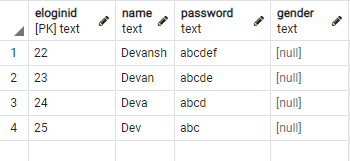
INSERT into employee values('24', 'Deva','abcd');

INSERT into employee values('25', 'Dev','abc');

ALTER TABLE employee ADD COLUMN Gender text;

SELECT \* from employee;

**Output:**

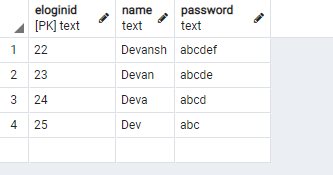


1. **Removing column:**

ALTER TABLE employee DROP COLUMN Gender;

SELECT \* from employee;

**Output:**



1. **Adding constraint:**

CREATE TABLE employee (

ELoginID text PRIMARY KEY,

Name text UNIQUE,

Password text

);

INSERT into employee values('22', 'Devansh','abcdef');

INSERT into employee values('23', 'Devan','abcde');

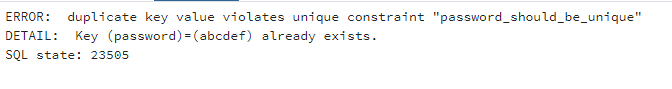
INSERT into employee values('24', 'Deva','abcd');

INSERT into employee values('25', 'Dev','abc');

ALTER TABLE employee ADD CONSTRAINT password\_should\_be\_unique UNIQUE (Password);

INSERT into employee values('26', 'Kevin','abcdef');

**Output:**



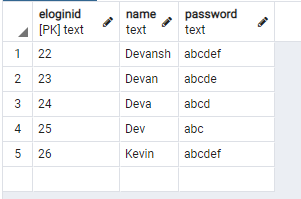
1. **Removing constraint:**

ALTER TABLE employee DROP CONSTRAINT password\_should\_be\_unique;

INSERT into employee values('26', 'Kevin','abcdef');

SELECT \* from employee;

**Output:**



1. **Adding Not Null constraint:**

CREATE TABLE employee (

ELoginID text PRIMARY KEY,

Name text UNIQUE,

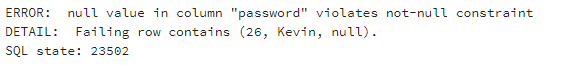
Password text

);

ALTER TABLE employee ALTER COLUMN Password SET NOT NULL;

INSERT into employee values('26', 'Kevin');

**Output:**



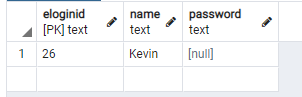
1. **Removing Not Null constraint:**

ALTER TABLE employee ALTER COLUMN Password DROP NOT NULL;

INSERT into employee values('26', 'Kevin');

SELECT \* from employee;

**Output:**



# 

**Questions:**

**Q1) What is difference between Truncate, Drop and delete? Explain with example**

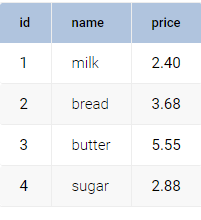
**Ans)**

The [**DELETE**](https://www.orafaq.com/wiki/DELETE)command is used to remove some or all rows from a table. A WHERE clause can be used to only remove some rows. If no WHERE condition is specified, all rows will be removed.

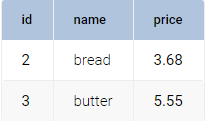
[**TRUNCATE**](https://www.orafaq.com/wiki/TRUNCATE) removes all rows from a table. The operation cannot be rolled back and no triggers will be fired. As such, TRUNCATE is faster and doesn't use as much undo space as a DELETE.

The [**DROP**](https://www.orafaq.com/wiki/DROP) command removes a table from the database. All the tables' rows, indexes and privileges will also be removed. The operation cannot be rolled back.

Initially:-









deletes all records stored in the table **product**



removes all data in the table **product** and the structure of the table.

**Outcomes:**

**CO3:** Illustrate the concept of security, Query processing, indexing and Normalization for Relational database.

**Conclusion:**

This experiment helps us convert relational model to tables in sql using postgresql software. All the commands given in the write up have been executed, attached a screenshot and thus we have a better understanding of these commands.

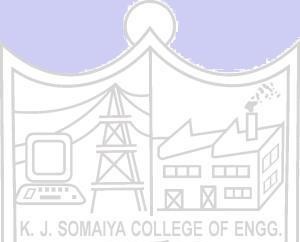
**Grade: AA / AB / BB / BC / CC / CD /DD Signature of faculty in-charge with date**

**Reference books:**

1. Elmasri and Navathe, “Fundamentals of Database Systems”, 6th Edition, Pearson Education
2. Korth, Slberchatz,Sudarshan, :”Database System Concepts”, 6th
3. Edition, McGraw – Hill.

# WebSite:

1. <http://www.tutorialspoint.com/postgresql/>
2. <http://sage.virtual-labs.ac.in/home/pub/21/>

(Autonomou of Mumbai)

s College Affiliated to University