

**Experiment No.: 04**

**Title:** To use DML operations and SQL queries to

Populate the database

**Batch:** A-4 **Roll No.:** 16010422211 **Experiment No: 04**

**Aim:** To use DML operations and SQL queries to populate the database .

**\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_**

**Resources needed:** PostgreSQL PgAdmin4

**\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_**

**Theory:**

The Data Manipulation Language (DML) is used to populate the table with values, modify the table values and remove the rows of the table.

The DML statements are: SELECT

INSERT

UPDATE

DELETE

**\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_**

**Procedure:**

CREATE TABLE products ( product\_no integer,

name text, price numeric );

Let us consider the above products table

**Inserting rows:**

The INSERT command requires the table name and column values

INSERT INTO products VALUES (1, ’Cheese’, 9.99);

If we don’t have values for all the columns, you can omit some of them. In that case, the columns will be filled with their default values. For example:

INSERT INTO products (product\_no, name) VALUES (1, ’Cheese’)

**Updating the values:**

The UPDATE command requires three pieces of information:

1. The name of the table and column to update
2. The new value of the column
3. Which row(s) to update

UPDATE products SET price = 10 WHERE price = 5;

UPDATE products SET price = price \* 1.10;

**Deleting rows:**

The syntax of the DELETE command is similar to the UPDATE command. DELETE FROM products WHERE price = 10;

**Retrieving values:**

The general syntax of the SELECT command is SELECT select\_list FROM table\_expression SELECT \* FROM table1;

SELECT \* FROM products WHERE price=10;

SELECT product\_no, name FROM products WHERE price=10;

**Example:**

insert into department values('IT', 101, 'mumbai');

insert into department values('COMP', 102, 'mumbai');

insert into department values('ETRX', 103, 'delhi');

insert into department values('EXTC', 104, 'chennai');

insert into department values('account', 105, 'mumbai');

insert into employee values('anita','m','sharma','emp0001',20000,'mumbai',101);

insert into employee values('nita','g','patil','emp0004',10000,'mumbai',101);

insert into employee values('krupita','v','jetali','emp0003',20000,'delhi',103);

insert into employee values('juhi','r','verma','emp0002',15000,'delhi',104);

insert into employee values('anita','m','sharma', 'emp0005',20000,'mumbai',104);

insert into project values( 1, 'mumbai','website',101);

insert into project values( 2, 'chennai','coding',101);

insert into project values( 3, 'mumbai','testing',102);

insert into project values( 4, 'delhi','documentaion',103);

insert into works\_on values(1,'emp0001', 12);

insert into works\_on values(1,'emp0002', 10);

insert into works\_on values(2,'emp0001', 6);

insert into works\_on values(3,'emp0004', 2);

insert into dependent values(‘emp0001’, ‘sunita’,’sister’);

insert into dependent values(‘emp0001’, ‘nita’,’mother’);

insert into dependent values(‘emp0002’, ‘kamal’,’brother’);

insert into dependent values(‘emp0004’, ‘krishna’,’father’);

select \* from employee;

select \* from department;

select \* from project;

select \* from dependent;

select \* from works\_on;

1) employee

fnamemnamelnamessn salary ecitydno

---------------------------------------- ------------------------------ ----------------------------------

anita m sharma emp0001 20000 mumbai101

juhi r verma emp0002 15000 delhi 104

krupita v jetali emp0003 20000 delhi 103

nita g patil emp0004 10000 mumbai 101

anita m sharma emp0005 20000 mumbai104

2) department

dnamednodlocation

------------------------------ ----------- -----------------------------------

IT 101 mumbai

COMP 102 mumbai

ETRX 103 delhi

EXTC 104 chennai

account 105 mumbai

4) project



pnoplocationpnamedno

----------- ---------------------------------------- -------------------- -----------

1 mumbai website 101

2 chennai coding 101

3 mumbai testing 102

4 delhidocumentaion 103

5) dependents

ssndepname relation

-------------------- ------------------------------ ------------------------------

emp0001nita mother

emp0001sunita sister

emp0002kamal brother

emp0004krishna father

6) woks\_on

pnossnno\_of\_hrs

----------- -------------------- -----------

1 emp0001 12

1 emp0002 10

2 emp0001 6

3 emp0004 2

**\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_**

**Results: (Queries printout with output as per the format)**

1. Write 10 queries using ‘from’ and ‘where’ clause.

**Example:**

1. **To extract the name and ssn of all the employees:**

Select fname, mname, lname, ssn from employee;

fnamemnamelnamessn

---------------------------------------- ---------------------------------------- -----------------------------

anitasharmam emp0001

juhiverma r emp0002

krupitajetali v emp0003

nitapatil g emp0004

anitasharma m emp0005

1. **To select names and city of the employees earning salary more then 10000:**

Select fname, mname, lname, ecity from the employee where salary>10000;

fnamemnamelname ecity

---------------------------------------- ------------------------- ----------------------------------------

anitasharmam mumbai

juhivermar delhi

krupitajetaliv delhi

anitasharma m mumbai

1. **TO get the details of the cities of the employees in our company:**

select distinct ecity from employee;

ecity

------------

delhi

mumbai

1. **To find the name of the department located in Mumbai and with department number 101:**

select dname from department where dlocation=’Mumbai’ and dno=101;

dname

--------------

1. **To delete all dependent whose relation is mother with employee:**

delete form dependent where relation=’mother’;

ssndepname relation

-------------------- ------------------------------ ------------------------------

emp0001sunita sister

emp0002kamal brother

emp0004krishna father

1. **Update relation employee to increment salary of all employees working in Department 101 by Rs. 10000:**

update employee set salary=salary+10000 where dno=101;

fnamemnamelnamessn salary ecitydno

---------------------------------------- ------------------------------ ----------------------------------

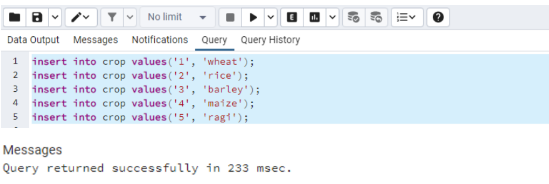
anita m sharma emp0001 30000 mumbai101

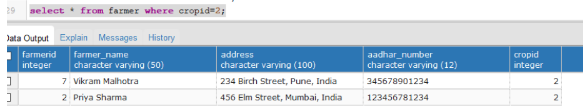
juhi r verma emp0002 15000 delhi 104

krupita v jetali emp0003 20000 delhi 103

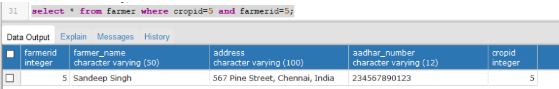
nita g patil emp0004 20000 mumbai 101

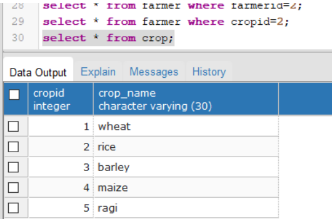
anita m sharma emp0005 20000 mumbai104

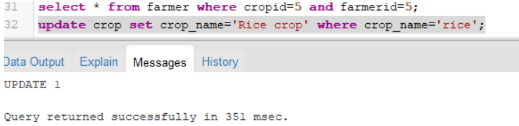


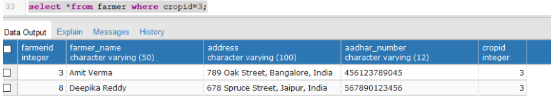


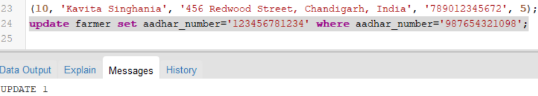


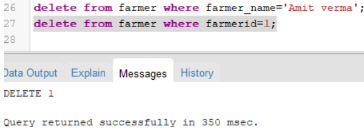




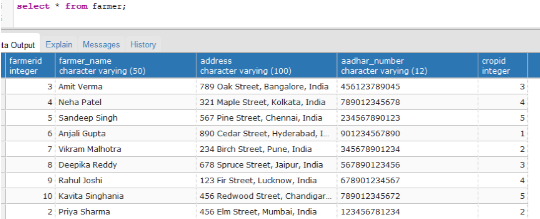












**\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_**

**Outcomes:**

**Illustrate the concept of security, Query processing, indexing and Normalization for Relational Database.**

**\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_**

**Questions:**

**Q1 Explain various data types used in SQL**

**Q2 what is outer JOIN and why it is used? Explain its type with example**

1. *SQL (Structured Query Language), data types are used to specify the type of data that can be stored in a column of a database table. The various different data types in SQL includes:*
2. *NUMERIC TYPE: Numeric types consist of two-byte, four-byte, and eight-byte integers, four-byte and eight-byte floating-point numbers, and selectable-precision decimals.*
3. *CHARACTER TYPE*
4. *MONETARY TYPE: The money type stores a currency amount with a fixed fractional precision.*
5. *BINARY DATA TYPE: The bytea data type allows storage of binary strings*
6. *DATA/TIME DATA TYPE: Used for date values, such as DD/MM/YYYY*
7. *BOOLEAN TYPE: Used for Boolean values which can either be TRUE, FALSE or NULL*
8. *GEOMETRIC TYPE: Geometric data types represent two-dimensional spatial objects.*
9. *BIT STRING TYPE: Bit String Types are used to store bit masks. They are either 0 or 1.*
10. *An OUTER JOIN in SQL is used to combine rows from two or more tables based on a related column between them. Unlike INNER JOIN, which only returns matching rows, OUTER JOIN returns all rows from at least one of the tables, along with the matching rows from the other table(s). If there is no match for a row in one table, the result will contain NULL values for columns from that table.*

*There are three main types of OUTER JOINs: LEFT OUTER JOIN, RIGHT OUTER JOIN, and FULL OUTER JOIN.*

*1. LEFT OUTER JOIN:*

*A LEFT OUTER JOIN returns all the rows from the left table (the first table specified in the query) and the matching rows from the right table (the second table specified in the query). If there's no match in the right table, NULL values are returned.*

*.*

*2. RIGHT OUTER JOIN:*

*A RIGHT OUTER JOIN is similar to a LEFT OUTER JOIN but returns all rows from the right table and matching rows from the left table.*

*3. FULL OUTER JOIN:*

*A FULL OUTER JOIN returns all rows when there is a match in either the left or right table. If there is no match in one of the tables, the result will contain NULL values for columns from that table. OUTER JOINs are useful when you want to retrieve data from multiple tables, including cases where there may not be a perfect one-to-one match between them. They help maintain data integrity and provide a more comprehensive view of your data.*

**Conclusion: (Conclusion to be based on the objectives and outcomes achieved)**

Learnt how to use DML operations and SQL queries to populate the database.

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

**Signature of faculty in-charge with date**

**\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_**

**References:**

**Books:**

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

Hill.

**WebSite:**

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