

**Experiment No.: 04**

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

Populate the database

**Batch: B-1** **Roll No.: 16010422234 Name: Chandana Ramesh Galgali**

**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’ clauses.

INSERT INTO Aviation\_Services VALUES (1001, 'Yes', 'No', 'None', 'Yes', 'Yes', 'None');

INSERT INTO Aviation\_Services VALUES (1002, 'Yes', 'No', 'Refreshments', 'Yes', 'No', 'None');

INSERT INTO Aviation\_Services VALUES (1003, 'Yes', 'No', 'None', 'No', 'Yes', 'None');

INSERT INTO Aviation\_Services VALUES (1004, 'No', 'Yes', 'None', 'Yes', 'No', 'None');

INSERT INTO Aviation\_Services VALUES (1005, 'No', 'No', 'Refreshments', 'No', 'No', 'None');

INSERT INTO Airport VALUES (1001, 'John F. Kennedy', 'New York', 4);

INSERT INTO Airport VALUES (1002, 'Heathrow', 'London', 2);

INSERT INTO Airport VALUES (1003, 'Changi', 'Singapore', 3);

INSERT INTO Airport VALUES (1004, 'Los Angeles', 'Los Angeles', 5);

INSERT INTO Airport VALUES (1005, 'Dubai', 'Dubai', 3);

INSERT INTO Airline VALUES (1001, 'Delta Air Lines', 'United States', 200);

INSERT INTO Airline VALUES (1002, 'British Airways', 'United Kingdom', 150);

INSERT INTO Airline VALUES (1003, 'Singapore Airlines', 'Singapore', 100);

INSERT INTO Airline VALUES (1004, 'American Airlines', 'United States', 180);

INSERT INTO Airline VALUES (1005, 'Emirates', 'United Arab Emirates', 250);

INSERT INTO Aircraft VALUES (1, 'Jet', 'Boeing');

INSERT INTO Aircraft VALUES (2, 'Propeller', 'Airbus');

INSERT INTO Aircraft VALUES (3, 'Jet', 'Bombardier');

INSERT INTO Aircraft VALUES (4, 'Propeller', 'Cessna');

INSERT INTO Aircraft VALUES (5, 'Jet', 'Embraer');

INSERT INTO Operated\_By VALUES (1, 1001, 'Delta Air Lines');

INSERT INTO Operated\_By VALUES (2, 1002, 'British Airways');

INSERT INTO Operated\_By VALUES (3, 1003, 'Singapore Airlines');

INSERT INTO Operated\_By VALUES (4, 1004, 'American Airlines');

INSERT INTO Operated\_By VALUES (5, 1005, 'Emirates');

INSERT INTO Aircraft\_Model (Aircraft\_ID, Model\_ID, Aircraft\_Model) VALUES (1, 1, 'Boeing 737');

INSERT INTO Aircraft\_Model (Aircraft\_ID, Model\_ID, Aircraft\_Model) VALUES (2, 2, 'Airbus A320');

INSERT INTO Aircraft\_Model (Aircraft\_ID, Model\_ID, Aircraft\_Model) VALUES (3, 3, 'Bombardier Global 6000');

INSERT INTO Aircraft\_Model (Aircraft\_ID, Model\_ID, Aircraft\_Model) VALUES (4, 4, 'Cessna Citation X');

INSERT INTO Aircraft\_Model (Aircraft\_ID, Model\_ID, Aircraft\_Model) VALUES (5, 5, 'Embraer E190');

INSERT INTO Flights VALUES (1, 1, '10:00', '08:00', 'New York', 'London', 'Boeing 737');

INSERT INTO Flights VALUES (2, 2, '15:30', '13:00', 'London', 'Singapore', 'Airbus A320');

INSERT INTO Flights VALUES (3, 3, '20:45', '18:30', 'Singapore', 'Los Angeles', 'Bombardier Global 6000');

INSERT INTO Flights VALUES (4, 4, '05:15', '03:00', 'Los Angeles', 'Dubai', 'Cessna Citation X');

INSERT INTO Flights VALUES (5, 5, '12:30', '10:00', 'Dubai', 'New York', 'Embraer E190');

INSERT INTO Passenger VALUES (1, 'pass123', 'John Smith', '1990-05-15', 'johnsmith', '123 Main St', 1234567890, 'Savings', 'ABC Bank', 'Yes');

INSERT INTO Passenger VALUES (2, 'pass456', 'Emma Johnson', '1985-09-20', 'emmajohnson', '456 Elm St', 1876543210, 'Checking', 'XYZ Bank', 'No');

INSERT INTO Passenger VALUES (3, 'pass789', 'Michael Davis', '1992-12-10', 'michaeldavis', '789 Oak St', 468135790, 'Savings', 'PQR Bank', 'Yes');

INSERT INTO Passenger VALUES (4, 'passabc', 'Sophia Wilson', '1988-07-05', 'sophiawilson', '321 Pine St', 1357924680, 'Checking', 'LMN Bank', 'No');

INSERT INTO Passenger VALUES (5, 'passxyz', 'Oliver Thompson', '1995-03-25', 'oliverthompson', '654 Cedar St', 187563210, 'Savings', 'XYZ Bank', 'Yes');

INSERT INTO Used\_By VALUES (1, 1001, 1);

INSERT INTO Used\_By VALUES (2, 1002, 2);

INSERT INTO Used\_By VALUES (3, 1003, 3);

INSERT INTO Used\_By VALUES (4, 1004, 4);

INSERT INTO Used\_By VALUES (5, 1005, 5);

INSERT INTO Mobile\_Number VALUES (1, 1234567890);

INSERT INTO Mobile\_Number VALUES (2, 1876543210);

INSERT INTO Mobile\_Number VALUES (3, 468135790);

INSERT INTO Mobile\_Number VALUES (4, 1357924680);

INSERT INTO Mobile\_Number VALUES (5, 187563210);

INSERT INTO Tickets VALUES (1, 1, 123, 10, 'Economy', 500);

INSERT INTO Tickets VALUES (2, 2, 456, 20, 'Business', 1000);

INSERT INTO Tickets VALUES (3, 3, 789, 30, 'Economy', 600);

INSERT INTO Tickets VALUES (4, 4, 321, 40, 'First Class', 1500);

INSERT INTO Tickets VALUES (5, 5, 654, 50, 'Economy', 550);

select \* from Aviation\_Services;

select \* from Airport;

select \* from Airline;

select \* from Aircraft;

select \* from Operated\_By;

select \* from Aircraft\_Model;

select \* from Flights;

select \* from Passenger;

select \* from Used\_By;

select \* from Mobile\_Number;

select \* from Tickets;

select Airport\_Name, Airport\_Location, Number\_Of\_Runways from Airport where Number\_Of\_Runways=3;

select Airline\_Name, Country\_Of\_Origin, Fleet\_Size from Airline where Fleet\_Size>=150;

select Manufacturer, Engine\_Design from Aircraft where Engine\_Design='Jet';

select Name, Account\_Number, Account\_Type, Bank\_Name from Passenger where Bank\_Name='XYZ Bank';

select Flight\_Number, Seat\_Number, Price, Seat\_Type from Tickets where Seat\_Type='Economy';

select \* from Flights where Destination='New York';

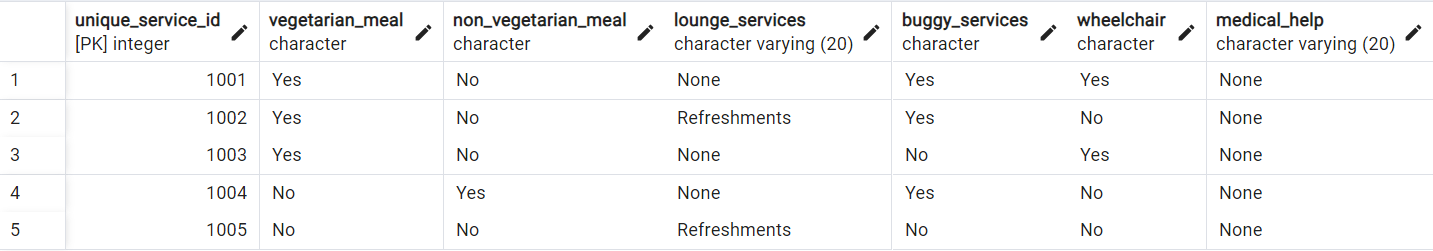
select Name, Account\_Number, Account\_Type, Bank\_Name from Passenger where Account\_Type='Savings';

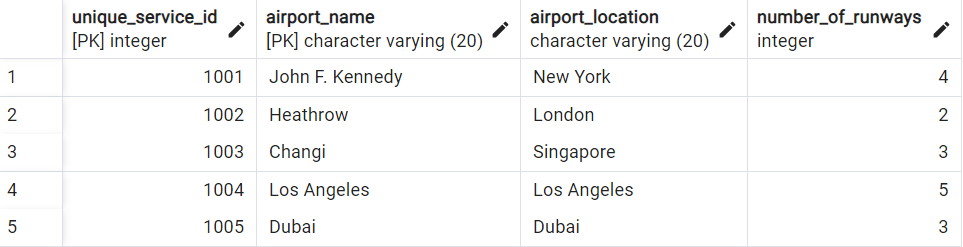
select \* from Airline where Country\_Of\_Origin='United States';

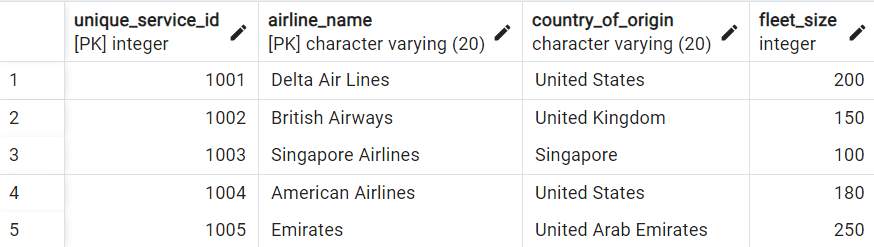
update Aviation\_Services set Medical\_Help='Onboard assistance' where Unique\_Service\_ID=1003;

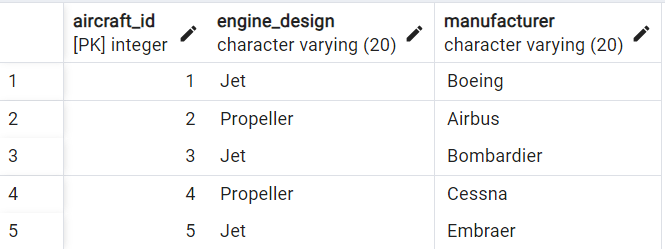
select Unique\_Service\_ID, Medical\_Help from Aviation\_Services;

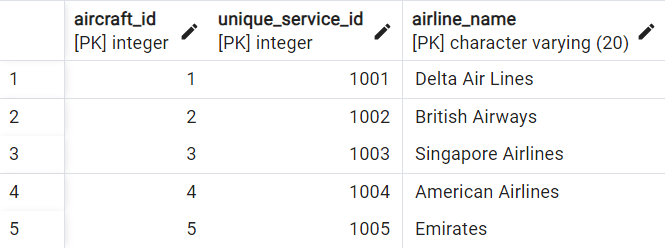
**Output:**

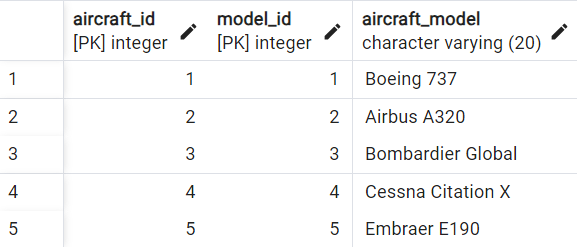


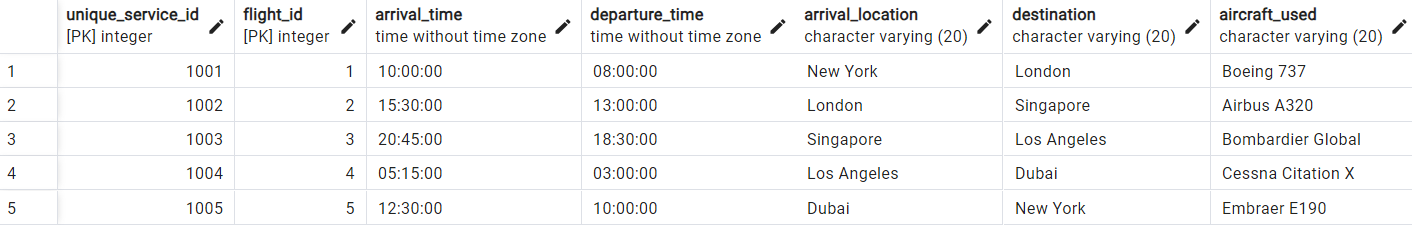


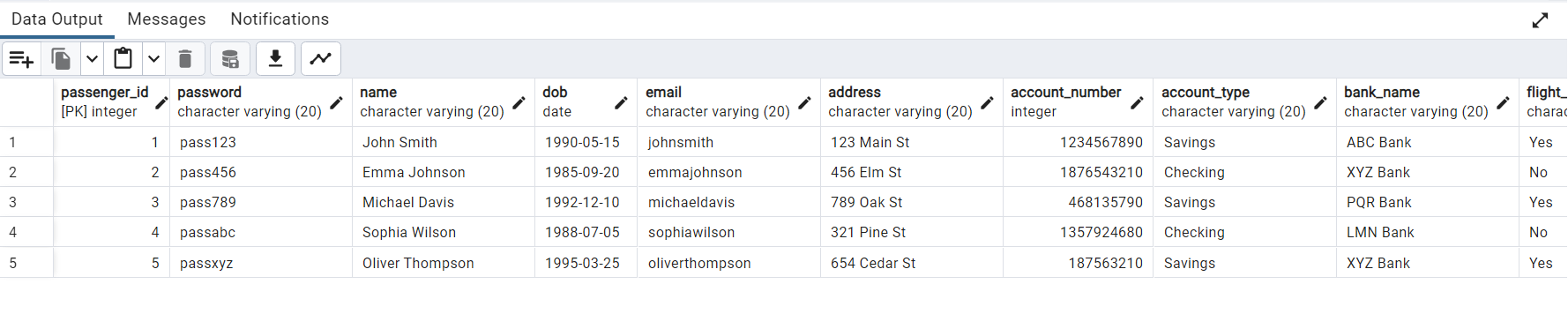


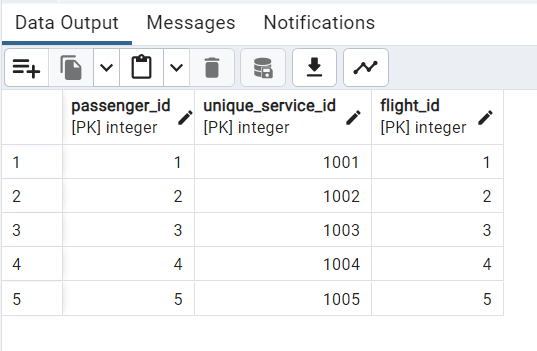


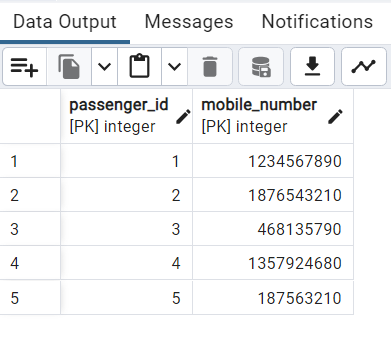


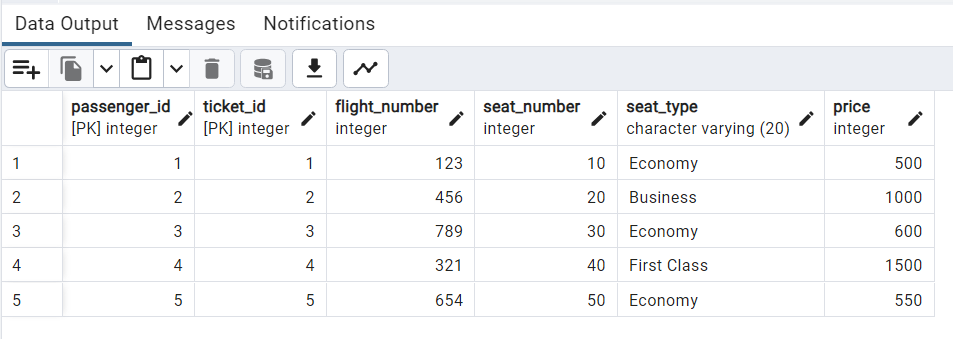


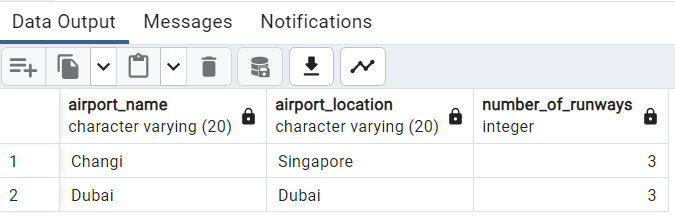


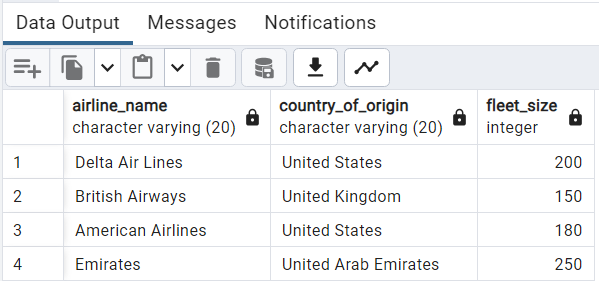


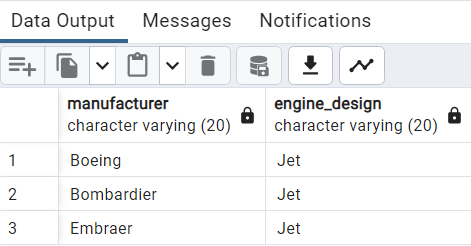


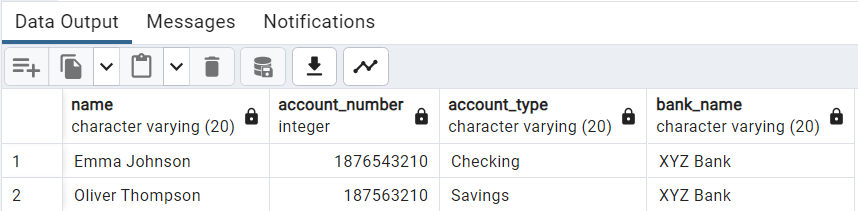


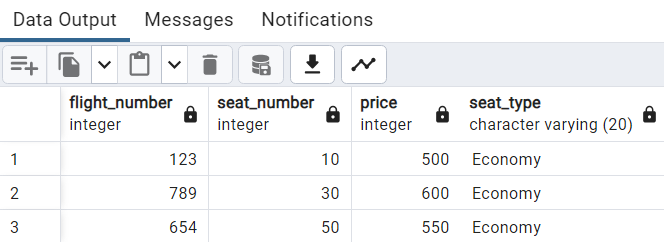


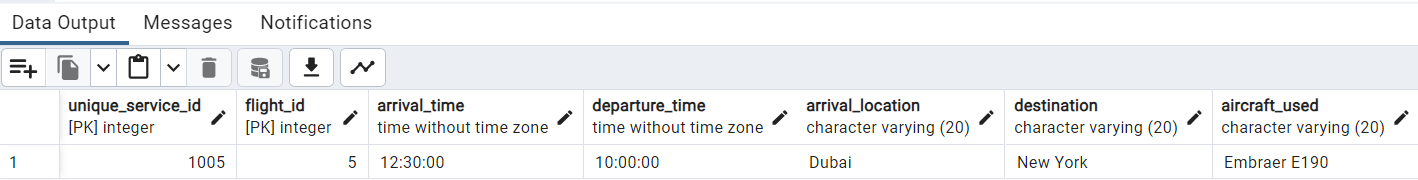


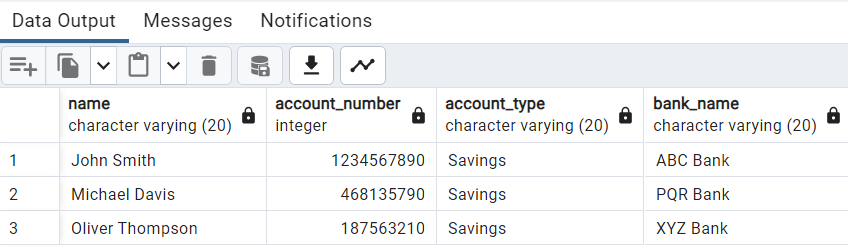


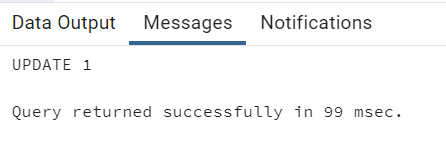


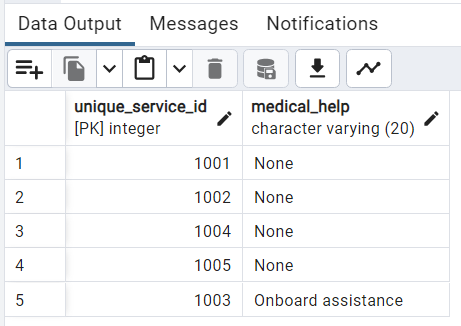












**Example:**

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: Apply data models to real world scenarios.**

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

**Questions:**

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

**Ans:** In SQL (Structured Query Language), there are several data types that are commonly used to define the type of data stored in a database. Here are some of the most commonly used data types in SQL:

1. Numeric Data Types:

- INT: Represents whole numbers (e.g., 1, 2, 3).

- FLOAT: Represents floating-point numbers with decimal places (e.g., 3.14, 2.718).

- DECIMAL: Represents fixed-point numbers with a specified precision and scale (e.g., 10.50, 3.14159).

2. Character Data Types:

- CHAR: Represents fixed-length character strings (e.g., 'Hello', 'OpenAI').

- VARCHAR: Represents variable-length character strings (e.g., 'Hello', 'OpenAI').

3. Date and Time Data Types:

- DATE: Represents a date value (e.g., '2023-10-18').

- TIME: Represents a time value (e.g., '12:30:45').

- DATETIME: Represents a combination of date and time values (e.g., '2023-10-18 12:30:45').

4. Boolean Data Type:

- BOOL or BOOLEAN: Represents a boolean value, which can be either true or false.

5. Binary Data Types:

- BLOB: Represents binary large objects, used to store large amounts of binary data (e.g., images, audio files).

- BINARY: Represents fixed-length binary data.

- VARBINARY: Represents variable-length binary data.

6. Other Data Types:

- ENUM: Represents a predefined list of values that a column can take.

- JSON: Represents JSON (JavaScript Object Notation) data.

These are just a few examples of the data types available in SQL. The specific data types supported may vary depending on the database management system (DBMS) you are using. It's important to choose the appropriate data type for each column in your database to ensure data integrity and efficient storage.

**Q2 What is outer JOIN and why is it used? Explain its type with examples.**

**Ans:** In SQL, an outer join is used to combine rows from two or more tables based on a related column, including unmatched rows from one or both tables. It includes all the rows from one table and the matching rows from the other table(s), and if there are no matches, it includes NULL values for the columns of the table with no match.

There are three types of outer joins:

1. Left Outer Join (or Left Join):

- Returns all the rows from the left table and the matching rows from the right table(s).

- If there is no match, NULL values are returned for the columns of the right table(s).

- Syntax: SELECT \* FROM table1 LEFT JOIN table2 ON table1.column = table2.column;

Example:

Consider two tables, "Customers" and "Orders":

Customers table:

+----+----------+

| ID | Name |

+----+----------+

| 1 | John |

| 2 | Mary |

| 3 | David |

+----+----------+

Orders table:

+----+--------------+

| ID | OrderDate |

+----+--------------+

| 1 | 2023-01-01 |

| 3 | 2023-02-01 |

+----+--------------+

Query: SELECT Customers.Name, Orders.OrderDate

FROM Customers

LEFT JOIN Orders ON Customers.ID = Orders.ID;

Result:

+-------+---------------+

| Name | OrderDate |

+-------+---------------+

| John | 2023-01-01 |

| Mary | NULL |

| David | 2023-02-01 |

+-------+---------------+

2. Right Outer Join (or Right Join):

- Returns all the rows from the right table(s) and the matching rows from the left table.

- If there is no match, NULL values are returned for the columns of the left table.

- Syntax: SELECT \* FROM table1 RIGHT JOIN table2 ON table1.column = table2.column;

3. Full Outer Join (or Full Join):

- Returns all the rows from both tables, including unmatched rows from both tables.

- If there is no match, NULL values are returned for the columns of the table(s) with no match.

- Syntax: SELECT \* FROM table1 FULL JOIN table2 ON table1.column = table2.column;

It's important to note that the availability of different types of outer joins may vary depending on the database management system (DBMS) you are using. Outer joins are useful when you want to retrieve data from multiple tables, including unmatched rows, and perform analysis or comparisons between the data in those tables.

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

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

The use of DML operations and SQL queries to populate the database has proven to be a successful approach, enabling us to effectively manage and utilize the data stored in 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/