## **ASSIGNMENT 4:**

- Download dataset from <a href="https://www.kaggle.com/datasets/krishujeniya/salary-prediction-of-data-professions?reso">https://www.kaggle.com/datasets/krishujeniya/salary-prediction-of-data-professions?reso</a> urce=download
- Ingest the dataset from your local machine storage into postgresQL database
   Hint: use copy command in sql editor which will copy your csv file to postgres DB
   For ingesting csv you might also need to create table according to the column
   structure of your CSV file ahead of executing copy command
- 3. Once the table is populated please complete following queries:
- -- creating table in db

```
CREATE TABLE
    employee_data (first_name VARCHAR(50), last_name VARCHAR(50), sex
CHAR(1), doj DATE, date_current DATE, designation VARCHAR(50), age FLOAT,
salary INT, unit VARCHAR(50), leaves_used FLOAT, leaves_remaining FLOAT,
ratings FLOAT, past_exp INT);
```

## -- COPY CSV TO DOCKER

```
docker cp data.csv postgres:/
```

-- copy from csv

```
\copy employee_data (first_name, last_name, sex, doj, date_current,
designation, age, salary, unit, leaves_used, leaves_remaining, ratings,
past_exp) FROM '/path/to/data.csv' DELIMITER ',' CSV HEADER;
```

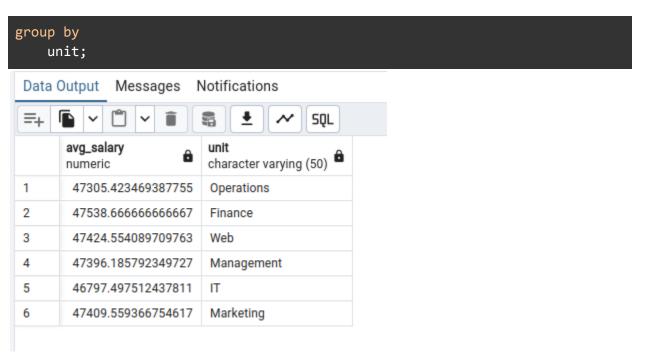
- -- Common Table Expressions (CTEs):
- -- Question 1: Calculate the average salary by department for all Analysts.

```
WITH

Analysts as (
select

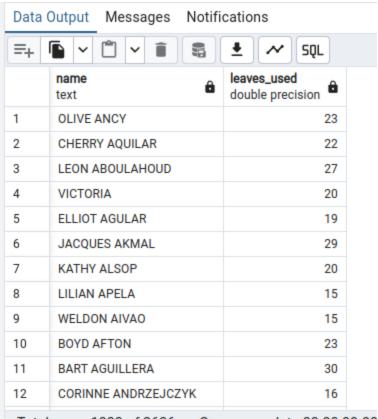
*
from
employee_data
where
designation LIKE '%Analyst'
)

SELECT
avg(salary) as avg_salary,
unit
from
Analysts
```



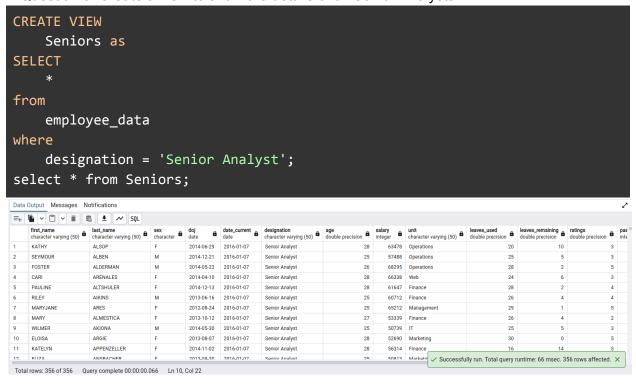
-- Question 2: List all employees who have used more than 10 leaves.

```
SELECT
    CONCAT (first_name, ' ', last_name) as Name,
    leaves_used
from
    employee_data
where
    leaves_used > 10;
```



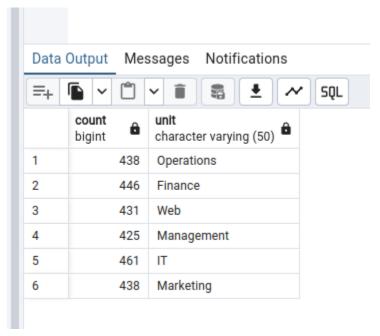
-- Views:

-- Question 3: Create a view to show the details of all Senior Analysts.



- -- Materialized Views:
- -- Question 4: Create a materialized view to store the count of employees by department.

```
CREATE MATERIALIZED VIEW Emp_count as
SELECT
      count(*),
      unit
from
      employee_data
group by
      Unit;
select * from Emp_count;
```



- -- Procedures (Stored Procedures):
- -- Question 6: Create a procedure to update an employee's salary by their first name and last name.

```
CREATE
OR REPLACE PROCEDURE update_salary (firstName varchar(50), lastName
varchar(50), newSalary INT) language plpgsql
as $$
BEGIN
Update employee_data
set
    salary = newSalary
where
```

```
first_name = firstName
   AND last_name = lastName;
-- commit;
END;
$$;
CALL update_salary ('BELLE', 'ARDS', 30000);
```

- -- Question 7: Create a procedure to calculate the total number of leaves used across all departments.
- -- drop procedure total\_leave;

