Week 02: SQL Practice Tasks

Online IDE for practice: http://www.sqlfiddle.com/
Practice document:
https://github.com/NYU-DataScienceBootCamp/Week-2-SQL/blob/main/SQL_Practice.pdf

NOTE: Make sure you answer the queries in the boxes given and paste screenshots in the output box.

The solution queries will be posted on June 24th before the session

Input Data

Use the database which was discussed during the session and feel free to change the attributes of the tables. Make sure that the following conditions are satisfied:

- There are three “tables”. One for storing Employee Details, One for Bonus, and One for Employee Title.
- There are at least 12 employees in the table which stores Employee Details.

NOTE: Make sure that you paste your input data in the box given below

Used the same input table as the one from the session

Tasks

SELECTing data

- Display the entire table containing the details of all the Employees

  QUERY:

  ```
  SELECT * FROM Employee
  ```

  OUTPUT:

- Write a query to fetch “FIRST_NAME” from the Employees table in the UPPER CASE

  QUERY:

  ```
  SELECT upper(FIRST_NAME) FROM Employee
  ```

  OUTPUT:

GROUPing them together

- Write a query to fetch the number of Employees for each department in the descending order

  QUERY:

  ```
  SELECT DEPARTMENT, count(EMPLOYEE_ID) No_Of_Employees
  FROM Employee
  GROUP BY DEPARTMENT
  ORDER BY No_Of_Employees DESC;
  ```

  OUTPUT:

Using WHERE somewhere

- Write a query to fetch the names of the Employees with salaries >= 90000 and <= 200000

QUERY:

```
SELECT CONCAT(FIRST_NAME, ' ', LAST_NAME) As Employee_Name, Salary
FROM Employee
WHERE EMPLOYEE_ID IN
(SELECT EMPLOYEE_ID FROM Employee
WHERE Salary BETWEEN 90000 AND 200000);
```

OUTPUT:

JOINing the tables

- Write a query to print details of Employees who are also “Managers”

QUERY:

```
SELECT DISTINCT E.FIRST_NAME, T.EMPLOYEE_TITLE
FROM Employee E
INNER JOIN Title T
ON E.EMPLOYEE_ID = T.EMPLOYEE_REF_ID
AND T.EMPLOYEE_TITLE in ('Manager');
```

OUTPUT:

COPYing

- Write an SQL query to clone a new table from another table

QUERY:

```
SELECT * INTO EmployeeClone FROM Employee
```

OUTPUT:

Aliasing

- Find the average salary of employees in each department and name the AVG(SALARY) column as “AverageSalary”

QUERY:

```
SELECT DEPARTMENT, AVG(SALARY) AS AverageSalary FROM  Employee
GROUP BY (DEPARTMENT);
```

OUTPUT:

Some other stuff

- Write an SQL query to show the second-highest salary from a table

QUERY:

```
SELECT max(SALARY) FROM Employee
WHERE SALARY NOT IN (SELECT max(SALARY) FROM Employee);
```

OUTPUT:

- Write an SQL query to show one row twice in results from a table

QUERY:

```
SELECT FIRST_NAME, DEPARTMENT from Employee E where
E.DEPARTMENT='HR'
UNION all
SELECT FIRST_NAME, DEPARTMENT from Employee E1 where
E1.DEPARTMENT='HR';
```

OUTPUT:

- Write an SQL query to fetch the departments that have less than five people in it

QUERY:

```
SELECT DEPARTMENT, COUNT(EMPLOYEE_ID) as ‘Number of Employees’
```

```
FROM Employee
GROUP BY DEPARTMENT HAVING COUNT(EMPLOYEE_ID) < 5;
```

OUTPUT:

- Write an SQL query to fetch the last five records from a table

QUERY:

```
SELECT * FROM Employee WHERE EMPLOYEE_ID <=5
UNION
SELECT * FROM (SELECT * FROM Employee E ORDER BY E.EMPLOYEE_ID
DESC) AS E1 WHERE E1.EMPLOYEE_ID <=5;
```

OUTPUT:

END OF FILE