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

|  |
| --- |
| CREATE TABLE Employee (  EMPLOYEE\_ID INT NOT NULL PRIMARY KEY AUTO\_INCREMENT,  FIRST\_NAME CHAR(25),  LAST\_NAME CHAR(25),  SALARY INT(15),  JOINING\_DATE DATETIME,  DEPARTMENT CHAR(25)  );  INSERT INTO Employee  (EMPLOYEE\_ID, FIRST\_NAME, LAST\_NAME, SALARY, JOINING\_DATE,  DEPARTMENT) VALUES  (001, 'Neville', 'Longbottom', 100000, '14-02-20 09.00.00', 'HR'),  (002, 'Ronald', 'Weasley', 80000, '14-06-11 09.00.00', 'Admin'),  (003, 'Hermoine', 'Granger', 300000, '14-02-20 09.00.00', 'HR'),  (004, 'Harry', 'Potter', 500000, '14-02-20 09.00.00', 'Admin'),  (005, 'Severus', 'Snape', 500000, '14-06-11 09.00.00', 'Admin'),  (006, 'Luna', 'Lovegood', 200000, '14-06-11 09.00.00', 'Account'),  (007, 'Draco', 'Malfoy', 75000, '14-01-20 09.00.00', 'Account'),  (008, 'Minerva', 'Mcgonagall', 90000, '14-04-11 09.00.00', 'Admin'),  (009, 'James', 'Potter', 700000, '14-02-11 09.00.00', 'Account'),  (010, 'Lilly', 'Evans', 300000, '14-06-11 09.00.00', 'Admin'),  (011, 'Remus', 'Lupin', 600000, '14-02-11 09.00.00', 'HR'),  (012, 'Sirius', 'Black', 700000, '14-06-11 09.00.00', 'Account');  CREATE TABLE Bonus (  EMPLOYEE\_REF\_ID INT,  BONUS\_AMOUNT INT(10),  BONUS\_DATE DATETIME,  FOREIGN KEY (EMPLOYEE\_REF\_ID)  REFERENCES Employee(EMPLOYEE\_ID)  ON DELETE CASCADE  );  INSERT INTO Bonus  (EMPLOYEE\_REF\_ID, BONUS\_AMOUNT, BONUS\_DATE) VALUES  (001, 5000, '16-02-20'),  (002, 3000, '16-06-11'),  (003, 4000, '16-02-20'),  (001, 4500, '16-02-20'),  (002, 3500, '16-06-11');  CREATE TABLE Title (  EMPLOYEE\_REF\_ID INT,  EMPLOYEE\_TITLE CHAR(25),  AFFECTED\_FROM DATETIME,  FOREIGN KEY (EMPLOYEE\_REF\_ID)  REFERENCES Employee(EMPLOYEE\_ID)  ON DELETE CASCADE  );  INSERT INTO Title  (EMPLOYEE\_REF\_ID, EMPLOYEE\_TITLE, AFFECTED\_FROM) VALUES  (001, 'Manager', '2016-02-20 00:00:00'),  (002, 'Executive', '2016-06-11 00:00:00'),  (008, 'Executive', '2016-06-11 00:00:00'),  (005, 'Manager', '2016-06-11 00:00:00'),  (004, 'Assistant Manager', '2016-06-11 00:00:00'),  (007, 'Executive', '2016-06-11 00:00:00'),  (006, 'Lead', '2016-06-11 00:00:00'),  (003, 'Lead', '2016-06-11 00:00:00'),  (009, 'Executive', '2016-06-11 00:00:00'),  (010, 'Executive', '2016-06-11 00:00:00'),  (011, 'Executive', '2016-06-11 00:00:00'),  (012, 'Executive', '2016-06-11 00:00:00'); |

# 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) as 'first name' from Employee; |

**OUTPUT:**

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| |  | | --- | | **first name** | | NEVILLE | | RONALD | | HERMOINE | | HARRY | | SEVERUS | | LUNA | | DRACO | | MINERVA | | JAMES | | LILLY | | REMUS | | SIRIUS | |

## 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) as count from Employee group by DEPARTMENT ORDER BY count DESC; |

**OUTPUT:**

|  |
| --- |
|  |

## Using WHERE somewhere

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

|  |
| --- |
| select FIRST\_NAME from Employee where SALARY >= 90000 AND SALARY <= 200000; |

**OUTPUT:**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| |  | | --- | | **FIRST\_NAME** | | Neville | | Luna | | Minerva | |

## JOINing the tables

* Write a query to print details of Employees who are also “Managers”  
    
  **QUERY:**

|  |
| --- |
| select E.FIRST\_NAME, T.EMPLOYEE\_TITLE from Employee E INNER JOIN Title T where E.EMPLOYEE\_ID = T.EMPLOYEE\_REF\_ID AND T.EMPLOYEE\_TITLE = 'Manager'; |

**OUTPUT:**

|  |
| --- |
|  |

## COPYing

* Write an SQL query to clone a new table from another table  
    
  **QUERY:**

|  |
| --- |
| * CREATE TABLE EmployeeClone LIKE Employee; * INSERT INTO EmployeeClone SELECT \* 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) as SecondHighestSalary from Employee where SALARY NOT IN (Select MAX(Salary) from Employee); |

**OUTPUT:**

|  |  |  |
| --- | --- | --- |
| |  | | --- | | **SecondHighestSalary** | | 600000 | |

* 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) 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:**

|  |
| --- |
|  |