CSE 4308 - DBMS LAB

DBMS LAB 04 TASKS AND SOLUTIONS

Prepared by:

Mohammad Anas Jawad

Lecturer, IUT CSE

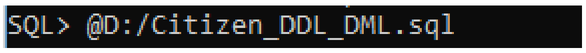
Department of Computer Science and Engineering

Islamic University of Technology

February 9, 2020

Note: Write down your commands and errors encountered in a notepad file to be evaluated.

Open your sql console, log in to your user account and run the attached .sql file. You can run the file as follows: @{path}{file}



```
SQL> @D:/Citizen_DDL_DML.sql
```

1. Show all the values of citizen table.

   ```
   SELECT * FROM CITIZEN;
   ```

2. Show only the c_name,age and occupation from the table.

   ```
   SELECT C_NAME, AGE, OCCUPATION FROM CITIZEN;
   ```

3. Show the name of the citizens who are living in Dhaka.

   ```
   SELECT NAME FROM CITIZEN WHERE C_HOME IN ('Dhaka');
   ```

4. Make a list of people whose income is more than 50,000/-

   ```
   SELECT * FROM CITIZEN WHERE SALARY>50000;
   ```

5. Make a list of people whose income is less than 90,000/-

   ```
   SELECT * FROM CITIZEN WHERE SALARY < 90000;
   ```

6. Make a list of people whose income is more than 45,000/-

   ```
   SELECT * FROM CITIZEN WHERE SALARY>45000;
   ```

7. Show the names and hometowns of those whose age is less than 45.

   ```
   SELECT C_NAME, C_HOME FROM CITIZEN WHERE AGE<45;
   ```

8. Make a list (all attributes) of the female citizens.

   ```
   SELECT * FROM CITIZEN WHERE GENDER IN ('Female');
   ```

9. Make a list (all attributes) of engineers, doctors and retired citizens.

   ```
   SELECT * FROM CITIZEN WHERE OCCUPATION IN ('Engineer','Doctor','Retired');
   ```

10. Show the id, name and salary of musicians and businessmen.

    ```
    SELECT C_ID, C_NAME, SALARY FROM CITIZEN WHERE OCCUPATION IN ('Musician','Business');
    ```

11. Show the list of citizens whose occupations might be either doctor or engineer.

```
SELECT * FROM CITIZEN WHERE OCCUPATION IN ('Engineer','Doctor');
```

12. Make an ordered list of engineers according to the salary.

```
SELECT * FROM CITIZEN WHERE OCCUPATION IN 'Engineer'ORDER BY SALARY ASC;
```

13. Make an descending ordered list based on the age and show the names and age only. If the ages are same, the names should be displayed in ascending order.

```
SELECT C_NAME, AGE FROM CITIZEN ORDER BY AGE DESC, C_NAME ASC;
```

14. Make a list of all the distinct c_home values in the tables. Rename the output column as 'Unique_District'

```
SELECT DISTINCT C_HOME AS Unique_District FROM CITIZEN;
```

15. Make a list of all the distinct age values in the tables. Rename the output column as 'Unique_Age'

```
SELECT DISTINCT AGE AS Unique_Age FROM CITIZEN;
```

16. What will be the age of all working citizens after 5 years?

```
SELECT AGE+5 FROM CITIZEN OCCUPATION NOT IN ('Student','Retired');
```

Note: A lot of you used the UPDATE statement to do this. But doing so will result in modifying the data in the original table. By using the SELECT query in this way, you can query whatever you want (including mathematical operations like this.)

17. What will be the salary of all working citizens if it is increased by 40%?

```
SELECT SALARY*1.4 FROM CITIZEN WHERE OCCUPATION NOT IN ('Student','Retired');
```

Note: Same as the previous one.

18. Make a list of male citizens who earn more than 50000 per month.

```
SELECT * FROM CITIZEN WHERE GENDER IN 'Male'AND SALARY>50000;
```

19. Show the details of citizens whose salary is within 30000/- to 50000/-.

```
SELECT * FROM CITIZEN WHERE SALARY BETWEEN 30000 AND 50000
```

20. Show the salary of citizens who aren't in the age group from 30 to 55.

    `SELECT SALARY FROM CITIZEN WHERE AGE NOT BETWEEN 30 AND 55;`

21. Create a new table –

    Updated_Citizen(C_ID, Name, C_Home, Age, Occupation, Gender, New_Salary).

    Populate the table with the same entries as the 'Citizen' table. Now, your task is to change the value of C_Home from 'Ctg' to 'Chittagong' wherever applicable.

    Create the table using your usual `CREATE TABLE` statement. You can copy the entries from CITIZEN table to UPDATED_CITIZEN using the syntax:

    `INSERT INTO UPDATED_CITIZEN SELECT * FROM CITIZEN;`

    Then,

    `UPDATE CITIZEN SET C_HOME='Chittagong'WHERE C_HOME IN 'Ctg';`

22. Delete the bottom 13 rows from the 'Updated_Citizen' table.

    `DELETE FROM UPDATED_CITIZEN WHERE C_ID>10`

23. Delete the top 10 rows from the 'Updated_Citizen' table.

    `DELETE FROM UPDATED_CITIZEN WHERE C_ID<11;`

24. Increase the age by 10 years and salary by 30% in the 'Updated_Citizen' table.

    `UPDATE UPDATED_CITIZEN SET AGE=AGE+10, NEW_SALARY=1.3*NEW_SALARY;`

25. Show a salary comparison between the citizens present in both the tables.

    ```
    SELECT CITIZEN.C_ID, SALARY, NEW_SALARY FROM CITIZEN, UPDATED_CITIZEN WHERE
    CITIZEN.C_ID = UPDATED_CITIZEN.C_ID;
    ```

    Or,

    ```
    SELECT C_NAME, SALARY, NEW_SALARY FROM CITIZEN NATURAL JOIN (SELECT C_ID, C_NAME,
    OCCUPATION, GENDER, NEW_SALARY FROM UPDATED_CITIZEN);
    ```

    Note: Notice how simply writing

    `SELECT C_NAME, SALARY, NEW_SALARY FROM CITIZEN NATURAL JOIN UPDATED_CITIZEN;`

    doesn't work in this case. The reason behind this is that the common attributes (columns with same names) between the tables must have the same value for natural join to work. In our case, we have changed the value of the Age attribute and C_Home in the UPDATED_CITIZEN table in a previous task. Since Age and C_Home are common attributes in both the tables but have different values in each, simply natural joining does not work in this case. Instead, what can be done is we can natural join the two tables by excluding the Age and C_Home attributes from any one of the table as we have done in our solution.