|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
|  | | | | | | | | | |
|  | Report on Lab-02  DATABASE MANAGEMENT SYSTEMS LAB | | | | | |  | |
|  |  | | | | | | |  | | |
| Submitted by  Adid-Al-Mahamud Shazid  Student Id: 210042172  Department: CSE  Programme: SWE  Course Title: CSE 4308  Submitted to  Zannatun Naim Sristy  Lecturer, Department of CSE  August 20, 2023 | |  | | | | |
|  |
|  | | |  |
|  | |  | | | |  | | | |

**Introduction**

In the lab class, we were given five tasks to solve using SQL command line to understand the basics of using Oracle. All the commands used were written in visual studio code which was then saved with .sql extension. The .sql file was then run through the SQL command line to execute all the commands.

**Task 1**

Create a user with username = <C\_student\_id> and password = cse4308 and grant necessary privileges to log in and execute DDL and DML statements. Then log in as that user.

* 1. **Solution**

CREATE USER c\_210042172 IDENTIFIED BY cse4308;

GRANT CREATE SESSION, RESOURCE, DBA to c\_210042172;

CONNECT c\_210042172/cse4308;

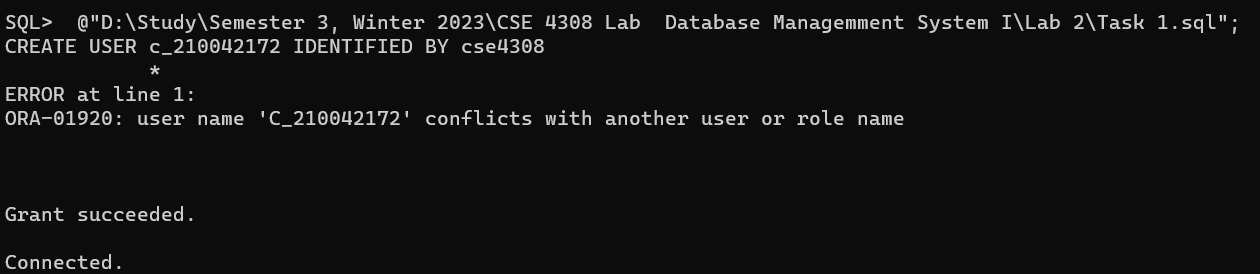
* 1. **Analysis and Explanation**

This problem was very straightforward and solved easily by following the instructions on the PDF document we were provided with.

* 1. **Difficulties**

I faced difficulty in figuring out that user name can’t start with numbers which restrained me creating the user name with my student id only.

* 1. **Output**

****

**Task 2**

Write SQL statement to create a table ‘INSTRUCTOR’ which has 4 attributes:

• ID (assign it as Primary Key)

• NAME

• DEPT\_NAME

• SALARY (ensure that SALARY is greater than 20000)

* 1. **Solution**

CREATE TABLE INSTRUCTOR

(

    ID NUMBER,

    NAME VARCHAR(50) NOT NULL,

    DEPT\_NAME VARCHAR(50) NOT NULL,

    SALARY INT NOT NULL,

    CONSTRAINT PK\_ID PRIMARY KEY (ID),

    CONSTRAINT SALARY\_CHECK CHECK (SALARY>20000)

);

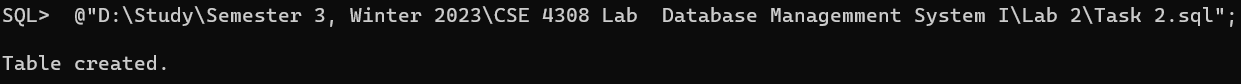
* 1. **Analysis and Explanation**

I created a table named INSTRUCTOR with 4 attributes. I learnt how to set the data type for different attributes and how to ensure that a field is not empty when inputting data into the table later (use of not null). I learnt to use constraint data type as well. PRIMARY KEY is such constraint which is used to uniquely identify a data. And CHECK constraint firstly checks a data whether it is true or false and only takes the input if it’s true.

* 1. **Difficulties**

I made the ID NUMBER null at the first run. But then I learnt that attributes having primary key shouldn’t be kept null as primary key must have a value.

* 1. **Output**

****

**Task 3**

Write SQL statements to insert the following records into ‘INSTRUCTOR’ table:

|  |  |  |  |
| --- | --- | --- | --- |
| ID | NAME | DEPT\_NAME | SALARY |
| 10101 | Srinivasan | Comp. Sci. | 65000 |
| 12121 | Wu | Finance | 90000 |
| 15151 | Mozart | Music | 40000 |
| 22222 | Einstein | Physics | 95000 |
| 32343 | El Said | History | 60000 |
| 00456 | Gold | Physics | 87000 |
| 45565 | Katz | Comp. Sci. | 75000 |
| 58583 | Califieri | History | 62000 |
| 76543 | Singh | Finance | 80000 |
| 76766 | Crick | Biology | 72000 |
| 03821 | Brandt | Comp. Sci. | 92000 |
| 98345 | Kim | Elec. Eng. | 80000 |

* 1. **Solution**

INSERT INTO INSTRUCTOR VALUES(10101, 'Srinivasan', 'Comp. Sci.', 65000);

INSERT INTO INSTRUCTOR VALUES(12121, 'Wu', 'Finance', 90000);

INSERT INTO INSTRUCTOR VALUES(15151, 'Mozart', 'Music', 40000);

INSERT INTO INSTRUCTOR VALUES(22222, 'Einstein', 'Physics', 95000);

INSERT INTO INSTRUCTOR VALUES(32343, 'El Said', 'History', 60000);

INSERT INTO INSTRUCTOR VALUES(00456, 'Gold', 'Physics', 87000);

INSERT INTO INSTRUCTOR VALUES(45565, 'Katz', 'Comp. Sci.', 75000);

INSERT INTO INSTRUCTOR VALUES(58583, 'Califieri', 'History', 62000);

INSERT INTO INSTRUCTOR VALUES(76543, 'Singh', 'Finance', 80000);

INSERT INTO INSTRUCTOR VALUES(76766, 'Crick', 'Biology', 72000);

INSERT INTO INSTRUCTOR VALUES(03821, 'Brandt', 'Comp. Sci.', 92000);

INSERT INTO INSTRUCTOR VALUES(98345, 'Kim', 'Elec. Eng.', 80000);

* 1. **Analysis and Explanation**

I inserted some records into the table I created in task 2. This task was also easy to complete.

* 1. **Difficulties**

I did not face any difficulties when doing this task.

* 1. **Output**

****

**Task 4**

Write SQL statements to perform the following queries:

(a) Display all records of ‘INSTRUCTOR’ table.

(b) Show instructor ID and name only.

(c) Find name and department of instructors who have salary more than 70000.

(d) Find name and department of instructors who have salary in between 80000 and 100000 (inclusive).

(e) Find ID and name of instructors of Comp. Sci. department.

(f) Find name and salary of instructors of Finance department.

(g) Find ID and name of instructors of Comp. Sci. department or instructors who are paid more than 75000.

(h) Find the names of the department.

* 1. **Solution**

SELECT \* FROM INSTRUCTOR;

SELECT ID, NAME FROM INSTRUCTOR;

SELECT NAME, DEPT\_NAME FROM INSTRUCTOR WHERE SALARY>70000;

SELECT NAME, DEPT\_NAME FROM INSTRUCTOR WHERE SALARY>=80000 AND SALARY<=100000;

SELECT ID, NAME FROM INSTRUCTOR WHERE DEPT\_NAME='Comp. Sci.';

SELECT NAME, SALARY FROM INSTRUCTOR WHERE DEPT\_NAME='Finance';

SELECT ID, NAME FROM INSTRUCTOR WHERE DEPT\_NAME='Comp. Sci.' OR SALARY>75000;

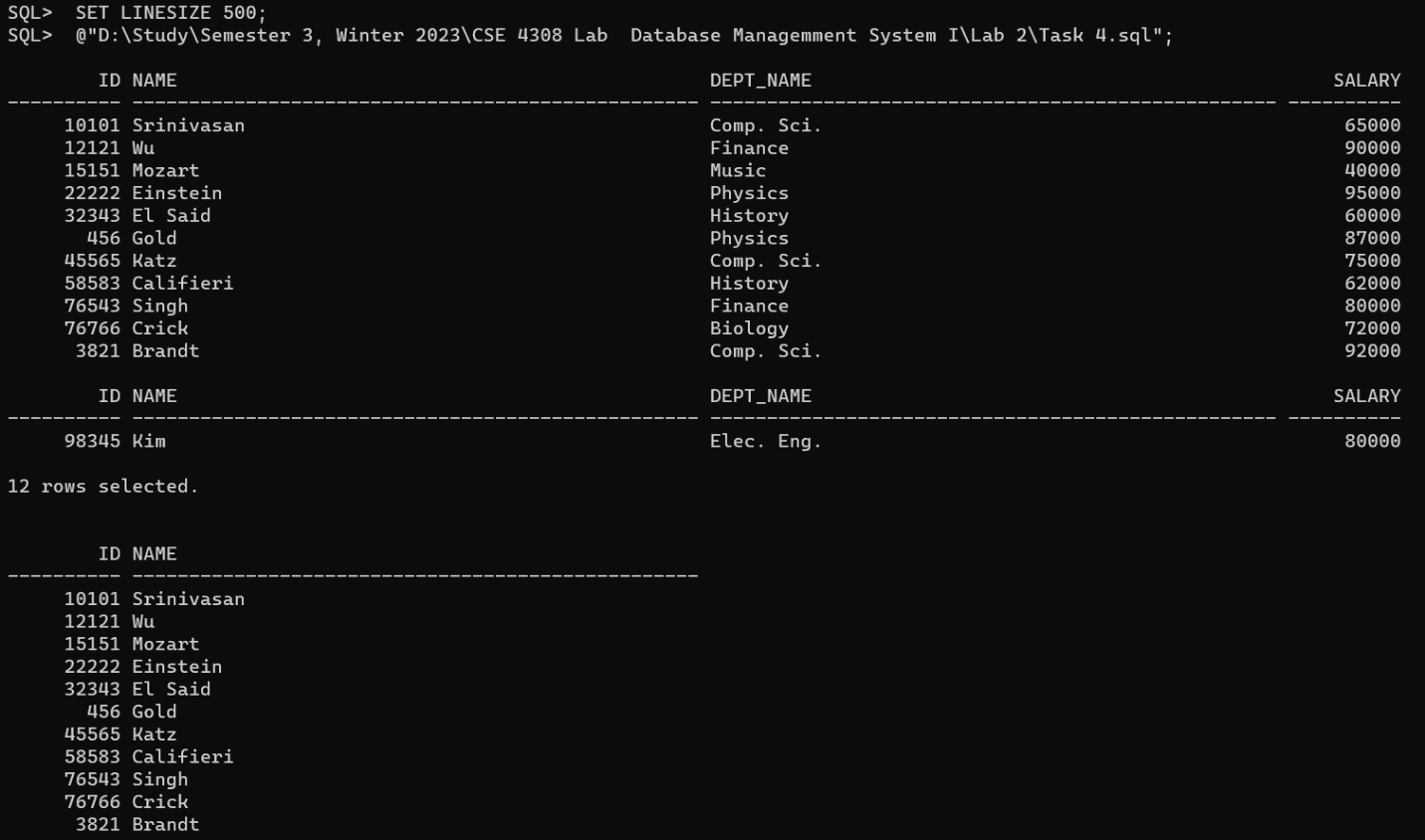
SELECT DEPT\_NAME FROM INSTRUCTOR;

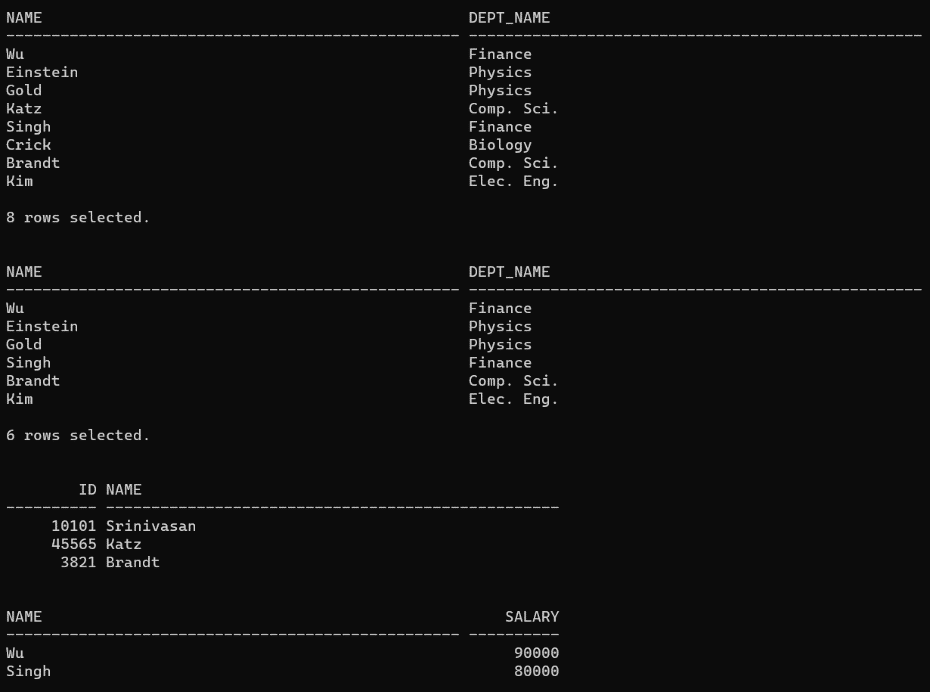
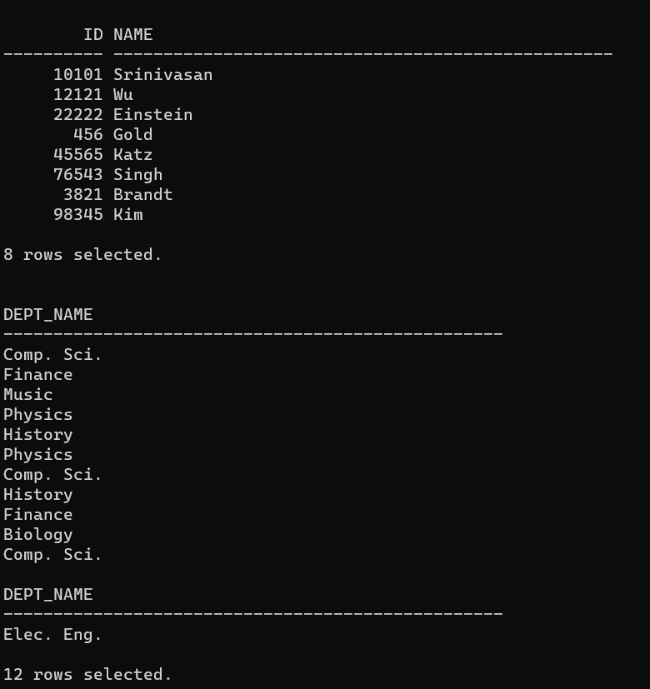
* 1. **Analysis and Explanation**

I learned about what the SELECT, FROM and WHERE commands do. The \* after SELECT shows all the columns available in the table that is why it was used for part (a) of the task. To show some selected attributes like only NAME, ID, SALARY and DEPT\_NAME in the other parts of the task, I listed them after writing the SELECT command. Comparison operators for SQL were similar to the operators in C++ programming language which made it easier to use for (c) to (h) parts of the task. The few differences were using = operator for equality and using the words and and or instead of any symbols. The last part (h) required printing all the different departments available in the table so the keyword distinct was used to avoid repetitions.

* 1. **Difficulties**

My output was not showing clearly during executing this task. So, I just added a command “SET LINESIZE 500” and could get the output properly.

* 1. **Output**

****

**Task 5**

Drop the ’INSTRUCTOR’ table with all its constraints.

* 1. **Solution**

DROP TABLE INSTRUCTOR CASCADE CONSTRAINTS;

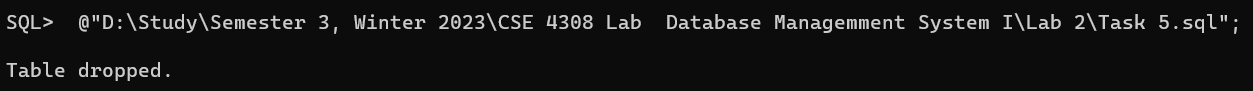
* 1. **Analysis and Explanation**

I just added CASCADE CONSTRAINTS so that the table is deleted with constraints.

* 1. **Difficulties**

I did not face any difficulties when doing this task.

* 1. **Output**

****