Islamic University of Technology

Lab 02

CSE 4308 - DBMS Lab

Submitted To :

Zannatun Naim Sristy
Lecturer, CSE Department
Islamic University of Technology

Submitted By :

Rhidwan Rashid
ID : 200042149
Prog. : SWE
Dept. : CSE

Creating a user:

Working code:

```
conn system/123
create user dihan200042149 identified by cse4308
grant all privileges to dihan200042149
conn dihan200042149/cse4308
```

Explanation:

First we log into system and then create a user using “create“ keyword as shown in code and set password after typing “identified by“. Now, we grant privileges to the user we created, for simplicity all the privileges are given here. Then we login into the user we created by providing the username we and password we gave.

Findings:

We have to create a user and give him/her privileges to access the oracle database. We can define what a user can do and can’t do by defining privileges.

Problems:

As we are leaning SQL as a new language, it was quite difficult to understand at first.

Creating student_info table:

Working code:

```sql
create table student_info(
      id varchar2(10) primary key,
      name varchar2(20),
      dept_name varchar2(20),
      tot_cred int
);
```

Explanation:

Best way to run this code snippet is writing it in a SQL format file and then execute it in SQL plus. As the title says, the code creates a student_info table which contains id, name and dept_name as an array of characters and tot_cred as integer.

Findings:

The coding style is new, parenthesis system is different. Other things like data types are more or less same as other languages.

Problems:

I got multiple errors because of the parenthesis issue. Took a while to find out.

Inserting records in student_info table:

Working code:

```sql
insert into student_info values('00128', 'Zhang', 'Comp. Sci', 102);
insert into student_info values('12345', 'Shankar', 'Comp. Sci', 32);
insert into student_info values('19991', 'Brandt', 'History', 80);
insert into student_info values('23121', 'Chavez', 'Finance', 110);
insert into student_info values('44553', 'Peltier', 'Physics', 56);
insert into student_info values('45678', 'Levy', 'Physics', 46);
insert into student_info values('54321', 'Williams', 'Comp. Sci', 5);
insert into student_info values('55739', 'Sanchez', 'Music', 38);
insert into student_info values('70557', 'Snow', 'Physics', 0);
insert into student_info values('76543', 'Brown', 'Comp. Sci', 58);
insert into student_info values('76653', 'Aoi', 'Elec. Eng', 60);
insert into student_info values('98765', 'Bourikas', 'Elec. Eng', 9);
insert into student_info values('98988', 'Tanaka', 'Biology', 120);
```

Explanation:

To insert records in the table we created we have to use the "insert" keyword as shown in the code. The values contain relative data we need to provide. It must be inserted in order when we created the table. So, we enter the id first, then name, department and lastly total credit. We have to do the same process for each row we need to insert in the table

Findings:

The values need to be inserted in order, otherwise it can break the code.

Problems:

No problem for short data list, but when the list gets bigger the problem will be bigger also if we type in every row and then insert in table.

SQL statements to perform queries:

Queries:

(a) Display all records of ‘STUDENT’ table.
(b) Show student ID and name only.
(c) Find name and department of students who have completed more than 100 credits.
(d) Find name and department of students who have completed in between 80 and 120
credits (inclusive).
(e) Find ID and name of students of Comp. Sci. department.
(f) Find name and total credit of students of Physics department.
(g) Find ID and name of students of Comp. Sci. department or students who have completed
less than 10 credits.
(h) Find the names of the department.

Solutions:

a.

Working code:

```
select * from student_info;
```

Explanation: Selects all records from student_info.

b.

Working code:

```
select id, name from student_info;
```

Explanation: selects only id and name attribute from all records.

c.

Working code:

```
select name, dept_name
from student_info
where tot_cred>100;
```

Explanation: this code snippet selects name and dept_name from table, and where is same as if else statement in other languages. This snippet find the students whose credit is above 100.

d.

Working code:

```
select name. dept_name
from student_info
where tot_cred>=80 and cred<=120;
```

Explanation: This snippet select name and department of those students whose credit is in between 80 t0 120. New keyword “and” was used.

e.

Working code:

```
select id, name
from student_info
where dept_name = 'Comp. Sci';
```

Explanation: This snippet prints id and name of Comp. Sci students.

f.

Working code:

```
select name, tot_cred
from student_info
where dept_name = 'Physics';
```

Explanation: This code snippet selects name and total credit of the students of physics department.

g.

Working code:

```
select id, name
from student_info
where tot_cred<10 or dept_name = 'Comp. Sci';
```

Explanation: This snippet prints id and name of those students who have less than 10 credit or they are from Comp. Sci department. New keyword “or” was used.

h.

Working code:

```
select distinct dept_name
from student_info;
```

Explanation: This code snippet prints all department name. “distinct” keyword was used to avoid duplication of department names.

Overall findings: The query commands are easy to use and powerful tools.

Problems: Not much problem faced before the last task. Use of “distinct” keyword to avoid duplication was a tricky part.