Database Management Systems Lab
Lab 5
CSE 4308

A Z Hasnain Kabir
200042102
Software Engineering

9 October 2022

Given Databases and Files

DDL+drop.sql and *smallRelationsInsertFile.sql* are given. A picture of the database schema are given below.

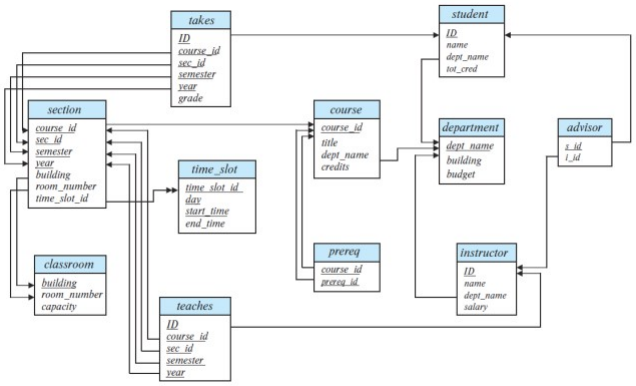

Figure 1: Here, the underlined attributes denote the primary keys and the arcs denote the foreign key relationships.

Task 1

Create a view named Advisor_Selection that shows the ID, name and department name of instructors.

Analysis of the Problem

View is a virtual table which behaves like a table but does not store any data. It derives its data from the table on which it is based on.

Code

```
CREATE OR REPLACE VIEW
Advisor_Selection AS
SELECT ID, name, dept_name FROM instructor;
```

Explanation of the solution

The CREATE OR REPLACE VIEW in Oracle generates a view which we can work with later. We name our table Advisor_selection and select three attributes from instructor table.

Task 2

Create another view named Student_Count using Advisor_Selection and advisor to show the name of the instructor and the number students assigned under them.

Analysis of the problem

We are to generate another view from the existing view which we created earlier called Advisor_selection and show number of students assigned under each instructor.

Code

```
CREATE OR REPLACE VIEW Student_Count AS
SELECT max(name) AS max_name, count(s_ID) AS num_S_ID
 FROM Advisor_Selection,
advisor WHERE i_ID = ID GROUP BY i_ID;
```

Explanation of the solution

We can create one view from another using the same SQL keywords CREATE OR REPLACE VIEW and select maximum number of names, number of student IDs under Advisor_selection.

Task 3

Four categories of users have been identified in the database:

Task 3(a)

Students should be able to view information regarding advisors and courses.

Analysis of the Problem

We are to generate a Role called Student where someone of the role will be given the privilege of being able to view Advisor_selection view and course table.

Code

```
CREATE ROLE STUDENT;
GRANT SELECT ON Advisor_Selection TO STUDENT;
GRANT SELECT ON course TO STUDENT;
```

Task 3(b)

Course teachers should be able to view information about the students and courses.

Analysis of the Problem

We are to generate a Role called course_teacher where someone of the role will be given the privilege of being able to view student table and course table.

Code

```
CREATE ROLE course_teacher;
GRANT SELECT ON student TO course_teacher;
GRANT SELECT ON course TO course_teacher;
```

Task 3(c)

Head of the Departments should have all the privileges that a course teacher has. Additionally, s/he should be able to add new instructors.

Analysis of the Problem

We are to generate a Role called Head where someone of the role will be given the privilege of being a course_teacher and insert into the instructor table.

Code

```
CREATE ROLE HEAD;
GRANT course_teacher to HEAD;
GRANT INSERT ON instructor TO HEAD;
```

Task 3(d)

Administrator should be able to see information about department and instructors. They should also be able to update the department budget.

Analysis of the Problem

We are to generate a Role called Administrator where someone of the role will be given the privilege of viewing the department as well as instructor table.

Code

```
CREATE ROLE ADMINISTRATOR;
GRANT SELECT ON department TO ADMINISTRATOR;
GRANT SELECT ON instructor TO ADMINISTRATOR;
```

Explanation of the solution

The CREATE ROLE keyword on SQL creates a role which is used for access control and privileges. Users of different roles can access the database differently.

Task 4

Create users under these roles and write relevant SQL queries to demonstrate that the imposed access control works.

Analysis of the problem

We are to create individual users to which we will grant roles we created and then test our users using some queries which will execute our operations.

Code

```
CREATE USER ABC IDENTIFIED BY cse_200042102;
GRANT STUDENT TO ABC;

CREATE USER X_LECTURER IDENTIFIED BY cse_17_109;
GRANT course_teacher TO X_LECTURER;

CREATE USER Prof_X IDENTIFIED BY cse_head_2022;
GRANT HEAD TO Prof_X;

CREATE USER X_operator IDENTIFIED BY admins;
GRANT ADMINISTRATOR TO X_operator;



GRANT CREATE SESSION TO ABC;
GRANT CREATE SESSION TO X_LECTURER;
GRANT CREATE SESSION TO Prof_X;
GRANT CREATE SESSION TO X_operator;

conn ABC/cse_200042102

SELECT ID, name, dept_name FROM
NIBIR_200042102.Advisor_Selection;
SELECT * FROM
NIBIR_200042102.course;

conn X_LECTURER/cse_17_109

SELECT * FROM
NIBIR_200042102.student;
SELECT * FROM
NIBIR_200042102.course;

conn Prof_X/cse_head_2022
```

```
INSERT INTO NIBIR_200042102.instructor
values('12345', 'Gilbert', 'Comp. Sci.', '50000');

conn X_operator/admins

SELECT * FROM
NIBIR_200042102.department;
SELECT * FROM
NIBIR_200042102.instructor;

UPDATE NIBIR_200042102.department
SET budget = '80000' WHERE dept_name = 'Biology';
```

Explanation of the Solution

Our solution is based on the ability to create users in Oracle technology. We create four users ABC, X_LECTURER, Prof_X and X_operator. We grant these users the privilege to create session. We connect using our username and password. Then we try some Data Manipulation techniques as connected user to check if our roles work as expected.