Computer Science Department
Michigan State University
CSE480 Database Systems
Lab Week #4 More on EER Model, Relation schemas and Constraints

(45 MINUTES)

1. Designer of the University Student database found that the concept of instructor was not appropriately captured in the EER diagram of lab 3 because of the following:

   (a) University requires that all those who teach courses must pass the English test and be paid at the same rate as that for the instructors. Therefore, those graduate students who teach courses must also be included in the instructor category.

   (b) University also allows people from local industries to teach courses.

Because of the new requirements, instructor becomes an important data on its own and we make instructor as an entity type. Answer the following questions related to incorporating these new requirements to the EER diagram.

   (a) Can Graduate be a subclass of Instructor? NO
   (b) Can Graduate be a super class of Instructor? NO
   (c) Can NonUniversityInstructor be a subclass of Instructor? YES
   (d) Can we make a separate entity type GradInstructor a subclass of both Graduate and Instructor? YES
   (e) Can we make a separate entity type FacultyInstructor a subclass of both Faculty and Instructor? YES

2. Make appropriate changes to the EER diagram given to you today in the lab to incorporate the above changes in requirements.
3. Give relation schemas for the above changes in the EER diagram.

(30 MINUTES)
In this part of the lab you will create a few relationship types in Oracle where the order in which you create them is important.

1. Implement the relationships **chair** and **works for** in the EER diagram in Oracle as follows (note: foreign keys are referencing each other):

Add the following into a file lab4.sql

```sql
Drop table Department cascade constraints;
Drop table Faculty cascade constraints;
create table Department(
Did number(7) CONSTRAINT PK_Department PRIMARY KEY,
DeptName varchar2(30),
CONSTRAINT FK_chair_Department FOREIGN KEY (chair) 
REFERENCES Faculty(FacSSNo));

create table Faculty(
FacSSNo varchar2(9) CONSTRAINT PK_Faculty PRIMARY KEY,
FacName varchar2(30),
worksFor number(7),
CONSTRAINT FK_worksFor_Faculty FOREIGN KEY (worksFor) 
REFERENCES Department(Did));

Start lab4.sql in sqlplus
What does the message say? Why is there an error?
Cannot create Department with FK reference to Faculty table for Chair, which is not created yet.

Remove the constraints **FK_chair_Department** from Department table above and add the following at the end of lab4.sql file.

```sql
Alter table Department
Add chair varchar2(9) UNIQUE
Add CONSTRAINT FK_chair_Department FOREIGN KEY (chair) 
REFERENCES Faculty(FacSSNo);

Describe Faculty;
Describe Department;
```
start lab4.sql
Why does it work now?
FK reference to Faculty has been removed.
Now check the following integrity constraint violation due to insert:
Add the following to the lab4.sql file

```sql
insert into Faculty values('000405555', 'John Doe', 01);
```

run lab4.sql
What do you see? What is the name of the constraint that is being violated?
Now remove the above insert statement and add the following into the lab4.sql file.

```sql
insert into Faculty values('000405555', 'John Doe', null);
Select * from Faculty;
insert into Department values(01, 'CSE', '000405555');
Select * from Department;
update Faculty set worksFor=01 where FacSSNo='000405555';
Select * from Faculty;
```

run lab4.sql
Does it work?

2. Now you implement the recursive 1:N relationship type mentor
the same way as above.
Create a separate table with name mentor. The pk is < Sid, MentorSid >. MentorId is a foreign key to student table.

3. For N:M (recursive) relationship type there is no ordering problem. Implement the recursive N:M relationship type prerequisite.
A separate table prerequisite with < cno, Pcno > as PK with foreign key referencing to CourseDescription.
(30 MINUTES)
You saw referential integrity violations due to inserts. In this part you will investigate how Oracle handles referential integrity violations due to deletes.

1. Add the following to your lab4.sql file:

```sql
Select * from Department;
Delete from Department where Did=01;
Select * from Department;
Select * from Faculty;
```

start lab4.sql
Why it does not work?
Referential integrity constraints violation.
Add the following two statements before the above Delete statement in the lab4.sql file:

```sql
Select * from Faculty;
Delete from Faculty where FacSSNo='000405555';
```

start lab4.sql
Does it work? Why it still does not work? Make appropriate modifications so that it works.
Still has referential integrity constraints violation.

2. Now you try delete cascade option (note cascade implies chain effect):
Add delete cascade option to the FK_chair_Department constraint defined earlier by Alter Table command, as follows:

```sql
Add CONSTRAINT FK_chair_Department FOREIGN KEY (chair)
REFERENCES Faculty(FacSSNo)
on Delete Cascade;
```

Remove the following two statements from the lab4.sql file.
Select * from Department;
Delete from Department where Did=01;

run lab4.sql
Does deleting Department tuple deletes all corresponding Faculty tuples? YES

3. Instead of deleting the corresponding faculty tuples you may want to set the Did attribute in the faculty tuple to null by using the Delete set null option instead of delete cascade. This is shown for FK_chair_Department constraint below:

Add CONSTRAINT FK_chair_Department FOREIGN KEY (chair)
    REFERENCES Faculty(FacSSNo)
    on Delete set null;

Make appropriate changes to lab4.sql file and see the effect of system actions for Delete set null option.

4. So you have three levels of actions by the system:

(a) Maximally restricted (no delete cascade, no delete set null): delete is rejected
(b) Less restricted (delete set null): delete is allowed, children (tuples in the referencing relation) are not deleted but the foreign key attribute values in the children are set to null.
(c) Least restricted (delete cascade): delete is allowed, all children, their children, and so on are also deleted by the system (could be dangerous but allows automatic maintenance of deletes).

Which option should you be choosing for FK_chair_Department and FK_worksFor_Faculty constraints, and why?

If we assume that a department must have a chair all the time, it should be maximally restricted. If we assume a new faculty may not have a dept assigned yet, we may assume less restricted (set null).