**Assignment Chương 2. Constraint, Trigger và Views**

**Câu 1 Ràng buộc khóa ngoại là gì? Tại sao các ràng buôc như vậy là quan trọng? Tính toàn vẹn tham chiếu là gì?**

* Ràng buộc khóa ngoại (ràng buộc tham chiếu): Các giá trị trong cột khóa ngoại ở bảng tham chiếu phải có trong cột tương ứng ở bảng được tham chiếu. Các ràng buộc như vậy là quan trọng để đảm bảo tính toàn vẹn của cơ sở dữ liệu
* Tính toàn vẹn tham chiếu là sự nhất quán về CSDL giữa 2 quan hệ tham chiếu với nhau.

**Câu 2. Cho lược đồ CSDL gồm các lược đồ quan hệ sau:**



1. **Liệt kê tất cả các ràng buộc khóa ngoại giữa các quan hệ này.**

Enrolled(*sid*) REFERENCES Students(sid)

Enrolled(*cid*) REFERENCES Cources(*cid*)

Teaches(*fid*) REFERENCES Faculty(*fid*)

Teaches(*cid*) REFERENCES Cources(*cid*)

Meets\_In(*cid*) REFERENCES Cources(*cid*)

Meets\_In(*rno*) REFERENCES Rooms(*rno*)

1. **Cho một ví dụ về một ràng buộc (hợp lý) bao gồm một hoặc nhiều quan hệ trên nhưng không phải là ràng buộc khóa chính hay khóa ngoại.**

VD: -2,147,483,648 <Rooms.capacity< 2,147,483,647 là ràng buộc miền giá trị của thuộc tính *capacity* của bảng Rooms

1. **Viết các câu lệnh tạo CSDL trên bao gồm tất cả ràng buộc có thể có.**

create database QLTH

go

use QLTH

go

create table Students(

sid char(10) primary key,

name varchar(50),

login varchar(50),

age int,

gpa real

)

go

create table Faculty(

fid char(10) primary key,

fname varchar(50),

sal real

)

go

create table Courses(

cid char(10) primary key,

cname varchar(50),

credits int

)

go

create table Rooms(

rno int primary key,

address varchar(50),

capacity int,

)

go

create table Enrolled(

sid char(10) references Students(sid),

cid char(10) references Courses(cid),

grand varchar(50),

primary key(sid, cid)

)

go

create table Teachers(

fid char(10) references Faculty(fid),

cid char(10) references Courses(cid)

primary key(fid, cid)

)

go

create table Meets\_in(

cid char(10) references Courses(cid),

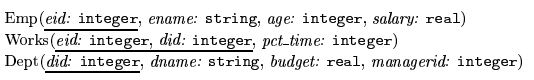
rno int references Rooms(rno),

time varchar(50),

primary key (cid,rno)

)

**Câu 3. Cho lược đồ CSDL gồm các lược đồ quan hệ sau:**



Hãy trả lời các câu hỏi sau :

1. **Cho một ví dụ về ràng buộc khóa ngoại có liên quan tới quan hệ Dept. Các lựa chọn nào có thể thực hiện để tuân thủ ràng buôc này khi người dùng xóa một bộ trong quan hệ Dept.**

- Works(*did*) có ràng buộc khóa ngoại với Dept(*did*)

- Không có lựa chọn nào có thể thực hiện để tuân thủ ràng buộc này khi người dùng xóa một bộ trong quan hệ Dept vì:

+ Nếu để Default Policy thì sẽ gây lỗi vì Works(*did*) có ràng buộc khóa ngoại với Dept(*did*)

+ Nếu để Set-null Policy thì sẽ gây lỗi vì Works(*did*) có ràng buộc khóa ngoại với Dept(*did*) nhưng Works(*did*) có ràng buộc not null.

1. **Viết các câu lệnh SQL để tạo các quan hệ trên bao gồm tất cả các RBTV khóa chính và khóa ngoại.**

create database QLNV

go

use QLNV

go

create table Emp(

eid int primary key,

ename varchar(50),

age int,

salary real

)

go

create table Dept(

did int primary key,

dname varchar(50),

budget real,

managerid int references Emp(eid)

)

go

create table Works(

eid int references Emp(eid),

did int references Dept(did),

pct\_time int

primary key (eid, did)

)

1. Dùng SQL định nghĩa lại quan hệ Dept sao cho mọi department được đảm bảo có một người quản lý.

ALTER TABLE Dept ADD CONSTRAINT managerid\_dept FOREIGN KEY (managerid) REFERNCES Emp.eid;

1. Viết một câu lệnh SQL để thêm nhân viên ‘John Doe’ với eid = 101, age = 32, và salary = 15.000

Insert into Emp(eid,ename,age,sallary) values (101, ‘Jhon Doe’,32,15000)

1. Viết một câu lệnh SQL để tăng 10% lương cho mọi nhân viên.

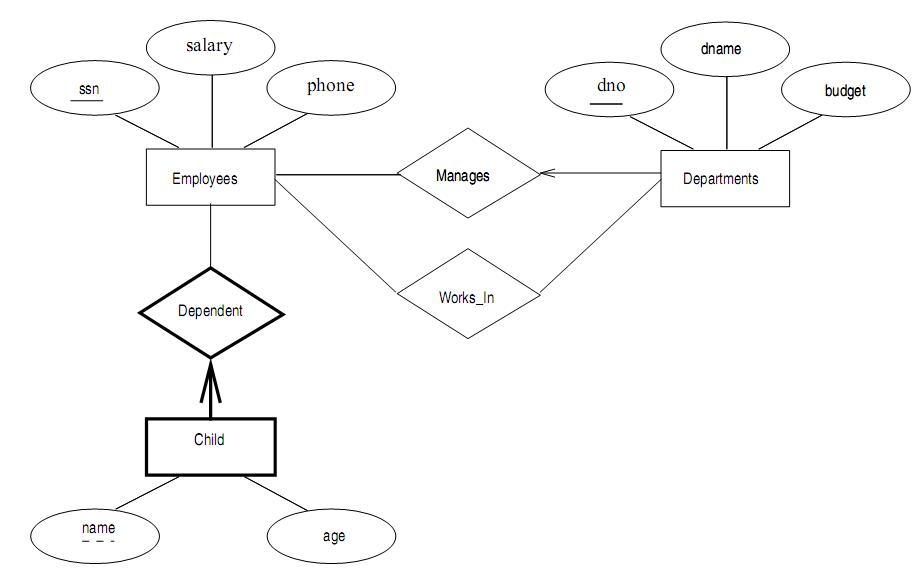
Update emp set sallary=sallary\*110/100;

1. Viết câu lệnh SQL để xóa department ‘Toy’. Với ràng buộc toàn vẹn tham chiếu bạn đã chọn cho lược đồ này, hãy giải thích điều gì xảy ra khi câu lệnh xóa này được thực hiện.

Delete from Dept where dname=’Toy’;

* **Giải thích:** Thuộc tính ***did*** trong quan hệ ***Works*** là một khóa ngoại và tham chiếu đến quan hệ ***Dept***. Đây là ràng buộc toàn vẹn tham chiếu được chọn. Bằng cách thêm hành động ON DELETE CASCADE vào hành động này, khi một hồ sơ bộ phận bị xóa, hồ sơ ***Works*** liên quan đến Department đó cũng bị xóa. Truy vấn hoạt động như sau: Quan hệ ***Dept*** được tìm kiếm một bản ghi có tên = 'Toy' và bản ghi đó sẽ bị xóa. Sau đó, trường ***did*** của bản ghi đó được sử dụng để tìm kiếm các bản ghi có giá trị did phù hợp trong quan hệ ***Works***. Tất cả các bản ghi như vậy sau đó sẽ bị xóa khỏi quan hệ ***Works***.

**Câu 4. Giả sử bạn đã thiết kế lược đồ ER cho CSDL một công ty như sau. Hãy viết các câu lệnh SQL để tạo các quan hệ tương ứng bao gồm nhiều nhất các ràng buộc có thể. Nếu có ràng buộc nào bạn không định nghĩa được, hãy giải thích tại sao.**



CREATE TABLE Like Group (

name CHAR(20),

cust\_name CHAR(20),

PRIMARY KEY (name,

cust\_name

),

FOREIGN KEY (name)

REFERENCES Group,

FOREIGN KEY (

cust\_name)

REFERENCES Customer)

CREATE TABLE Artwork Paints(

title CHAR(20),

artist name CHAR(20),

type CHAR(20),

price INTEGER,

year INTEGER,

PRIMARY KEY (title),

FOREIGN KEY (artist name)

REFERENCES Artist)

CREATE TABLE Classify (

title CHAR(20),

name CHAR(20),

PRIMARY KEY (title, name),

FOREIGN KEY (title) REFERENCES

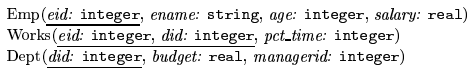
Artwork\_Paints

,

FOREIGN KEY (name) REFERENCES Group )

**Câu 5.**

**Cho lược đồ quan hệ sau:**



Hãy trả lời các câu hỏi sau.

1. **Định nghĩa một ràng buộc mức bảng trên Emp sao cho đảm bảo mọi nhân viên đều có lương ít nhất là 10.000**

CREATE TABLE Emp (eid INTEGER, ename CHAR(10), age INTEGER, salary REAL, PRIMARY KEY (eid), CHECK (salary >= 10000) )

1. **Định nghĩa một ràng buộc mức bảng trên Dept sao cho đảm bảo tất cả người quản lý đều có tuổi lớn hơn 30.**

CREATE TABLE Dept (did INTEGER, buget REAL, managerid INTEGER, PRIMARY KEY (did), FOREIGN KEY (managerid) REFERENCES Emp, CHECK((SELECT E.age FROM Emp E, Dept D) WHERE E.eid=D.managerid)>30)

1. **Định nghĩa một assertion trên Dept sao cho đảm bảo là mọi người quản lý đều có tuổi lớn hơn 30. So sánh assertion này với ràng buộc tạo ở câu 2. Hãy giải thích cái nào tốt hơn.**

***Lựa chọn 1:***

CREATE TABLE Dept (did INTEGER, budget REAL, managerid INTEGER, PRIMARY KEY (did))

Điều kiện chỉ được kiểm tra khi quan hệ Dept đang được cập nhật.

***Lựa chọn 2:***

CREATE ASSERTION managerAge CHECK ((SELECT E.age FROM Emp E, Dept D WHERE E.eid = D.managerid ) > 30 )

Nó kiểm tra khả năng vi phạm xác nhận bất cứ khi nào một trong các quan hệ được cập nhật 🡪 tốt hơn.

1. **Viết câu lệnh SQL để xóa tất cả thông tin về những Employees có lương cao hơn lương của người quản lý của họ. Phải đảm bảo là tất cả các ràng buộc toàn vẹn liên quan phải được thỏa mãn sau khi cập nhật.**

* Để có thể xóa mà các ràng buộc toàn vẹn liên quan phải được thỏa mãn sau khi cập nhật, cần phải xem xét các ràng buộc được xác định qua các bảng.

CREATE TABLE Emp ( eid INTEGER,

ename CHAR(10),

age INTEGER,

salary REAL,

PRIMARY KEY (eid) )

CREATE TABLE Works (eid INTEGER,

did INTEGER,

pcttime INTEGER,

PRIMARY KEY(eid, did),

FOREIGN KEY(did) REFERENCES Dept,

FOREIGN KEY(eid) REFERENCES Emp,

ON DELETE CASCADE)

CREATE TABLE Dept ( did INTEGER,

budget REAL,

managerid INTEGER,

PRIMARY KEY(did),

FOREIGN KEY(managerid) REFERENCES Emp,

ON DELETE SET NULL)

***Xóa Employees có lương cao hơn lương của người quản lý của họ:***

Gán A🡨(SELECT Emp.eid, salary, managerid

FROM Emp, Works, Dept

WHERE Emp.eid=Works.eid AND Works.did=Dept.did)

DELETE

FROM (SELECT nv.salary, ql.salary

FROM A nv, A ql

WHERE ql.eid=nv.managerid

),

WHERE(nv.salary > ql.salary)

**Câu 6. Hãy cho biết điểm mạnh và điểm yếu của cơ chế trigger. Trình bày sự tương phản giữa trigger với các ràng buộc toàn vẹn khác được hỗ trợ bởi SQL.**

***Điểm mạnh của Trigger:***

* Tính toán cập nhật dữ liệu tự động
* Kiểm tra dữ liệu nhập
* Kiểm tra ràng buộc phức tạp
* Bẫy lỗi dể hiểu

***Điểm yếu của trigger:***

* Không tạo và tham chiếu bảng
* Không tạo,thay đổi xóa cấu trúc đối tượng sẵn có trong CSDL
* Không gán cấp quyền cho người dùng.
* Trigger có thể tạo ra 1 bảng ảo mà khi thao tác trên CSDL nó sẽ tự động cập nhật

**Câu 7. Cho lược đồ quan hệ dưới đây. Một nhân viên có thể làm viêc ờ nhiều phòng ban. Cột pct\_time trong quan hê Works chỉ tỉ lệ % thời gian nhân viên đó làm việc cho phòng ban tương ứng.**

**Viết các RBTV, assertions hoặc triggers để đảm bảo các yêu cầu sau:**

1. **Mỗi nhân viên phải có lương tối thiểu là 1000**

CREATE TABLE Emp ( eid INTEGER,

ename CHAR(20),

age INTEGER, salary REAL, PRIMARY KEY (eid),

CHECK ( salary > 1000))

1. **Mọi người quản lý cũng là một nhân viên**

CREATE ASSERTION ManagerIsEmployee

CHECK ( ( SELECT COUNT (\*)

FROM Dept D

WHERE D.managerid NOT IN (SELECT \* FROM Emp))

= 0)

1. **Tổng tỉ lệ % thời gian làm việc cho các phòng ban của một nhân viên phải dưới 100%.**

CREATE TABLE Works ( eid INTEGER,

did INTEGER,

pct time INTEGER,

PRIMARY KEY (eid, did),

CHECK ( (SELECT COUNT (W.eid)

FROM Works W

GROUP BY W.eid

HAVING Sum(pct time) > 100) = 0))

1. **Một người quản lý phải luôn có lương cao hơn bất kỳ một nhân viên nào mà người đó quản lý.**

CREATE ASSERTION ManagerHigherSalary

CHECK ( SELECT E.eid

FROM Emp E, Emp M, Works W, Dept D

WHERE E.eid = W.eid

AND W.did = D.did

AND D.managerid = M.eid

AND E.salary > M.salary)

1. **Bất cứ khi nào một nhân viên được tăng lương, lương người quản lý cũng phải được tăng tương ứng.**

CREATE TRIGGER GiveRaise AFTER UPDATE ON Emp

WHEN old.salary < new.salary

FOR EACH ROW

BEGIN

UPDATE Emp M

SET M.Salary = new.salary

WHERE M.salary < new.salary

AND M.eid IN ( SELECT D.mangerid

FROM Emp E, Works W, Dept D

WHERE E.eid = new.eid

AND E.eid = W.eid

AND W.did = D.did);

END

1. **Bất cứ khi nào một nhân viên được tăng lương, lương người quản lý cũng phải được tăng tương ứng. Hơn nữa, bất cứ khi nào một nhân viên được tăng lương, ngân sách của phòng ban tương ứng cũng phải được tăng lớn hơn tổng lương của tất cả nhân viên thuộc phòng đó.**

CREATE TRIGGER GiveRaise AFTER UPDATE ON Emp

WHEN old.salary < new.salary

FOR EACH ROW

DECLARE

raise REAL;

BEGIN

raise := new.salary - old.salary;

UPDATE Emp M

SET M.Salary = new.salary

WHERE M.salary < new.salary

AND M.eid IN ( SELECT D.mangerid

FROM Emp E, Works W, Dept D

WHERE E.eid = new.eid AND E.eid = W.eid

AND W.did = D.did);

UPDATE Dept D

SET D.budget = D.budget + raise

WHERE D.did IN ( SELECT W.did

FROM Emp E, Works W, Dept D

WHERE E.eid = new.eid

AND E.eid = W.eid AND D.did = W.did

AND D.budget <

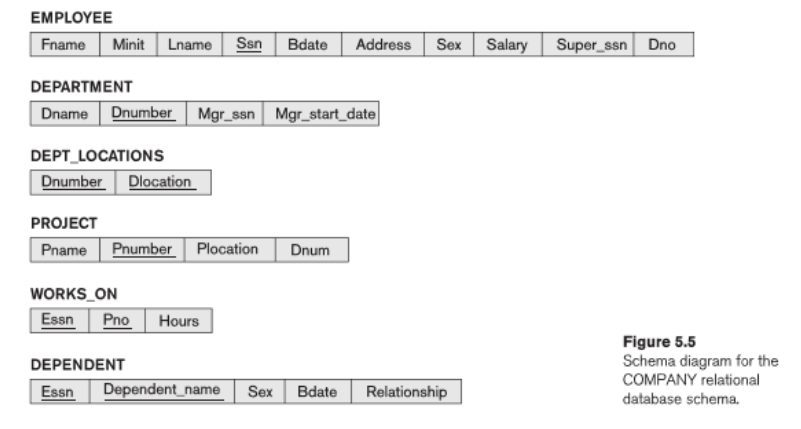
(SELECT Sum(E2.salary)

FROM Emp E2, Works W2

WHERE E2.eid = W2.eid AND W2.dept = D.did));

END

**Câu 8. Cho CSDL quan hệ của một công ty như dưới đây.**

****

***Dùng SQL để xác định các views sau :***

1. **Một view có department name, manager name và manager salary của mọi phòng ban.**

CREATE VIEW V1 AS

SELECT Dname, Fname, Lname, Salary

FROM Department, Employee

WHERE D. Mgrssn=E.Ssn

1. **Một view có employee name, supervisor name và employee salary của mỗi nhân viên thuộc phòng ‘Research’**

CREATE VIEW Research AS

SELECT eid, managerid, salary

FROM Dept, Emp

WHERE Dept.did = Emp.eid And Dept.dname = ‘Research’;

1. **Một view có project name, controlling department name, number of employees và tổng số giờ được làm việc mỗi tuần của mỗi dự án.**

CREATE VIEW V3 AS

SELECT Pname, Dname, (SELECT COUNT(Ssn) FROM WORKS\_ON W1 WHERE W1.Pno=P.Pno) AS Num\_Employees,

(SELECT SUM(Hours) FROM WORKS\_ON W2 WHERE W2.Pno=P.Pno) AS Total\_Hours

FROM Project P, Department D

WHERE P.Dno=D.Dno

1. **Một view có project name, controlling department name, number of employees và tổng số giờ được làm việc mỗi tuần của mỗi dự án với những dự án có nhiều hơn một nhân viên làm việc cho dự án đó.**

CREATE VIEW V4 AS

SELECT L.Pname AS Project\_Name, L.Dname AS ControllingDeptName, COUNT(P.Ssn) AS NumberOfEmp, SUM(P.Hours) AS HoursOfWeek

FROM ( SELECT PROJECT.Pname, DEPARTMENT.Dname, PROJECT.Pnumber FROM DEPARTMENT INNER JOIN PROJECT

ON DEPARTMENT.Dnumber = PROJECT.Dnum) AS L

INNER JOIN

(SELECT WORKS\_ON.Hours, EMPLOYEE.Ssn, WORKS\_ON.Pno

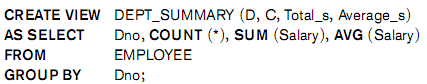
FROM WORKS\_ON INNER JOIN EMPLOYEE

ON WORKS\_ON.Essn = EMPLOYEE.Ssn) AS P

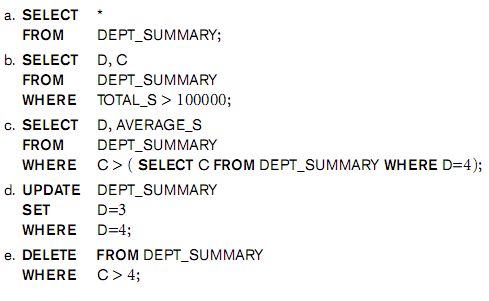
ON L.Pnumber = P.PnoGroup by L.Pname, L.Dname

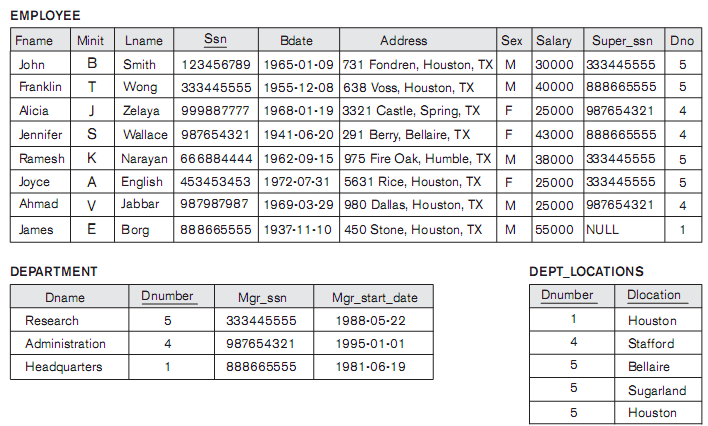
Having COUNT(P.Ssn)>1

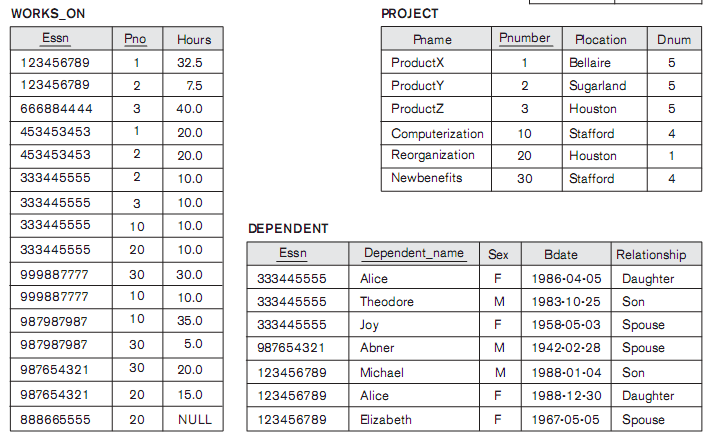
**Câu 9. Cho view DEPT\_SUMMARY được định nghĩa trên CSDL một công ty như sau.**



Hãy cho biết các truy vấn và thao tác cập nhật nào sau đây được phép trên view. Nếu một query hoặc update nào được phép, hãy cho biết câu truy vấn hoặc cập nhật tương ứng được thực hiện trên các quan hệ cơ sở tương ứng với view.





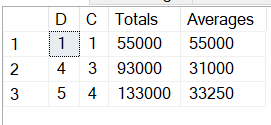


a. Chạy được

Hàm tương ứng

Select Dno as D,Count(\*) as C,sum(Salary) as Total\_s,AVG(Salary) as Average\_s

from EMPLOYEE Group by Dno;

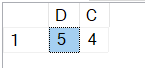


b. Chạy được

Hàm tương ứng

Select Dno as D,Count(\*) as C

from EMPLOYEE Group by Dno having sum(Salary)>100000;

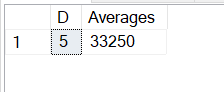


c. Chạy được

Hàm tương ứng

Select Dno as D,AVG(Salary) as Average\_s

from EMPLOYEE Group by Dno having Count(\*)>(select Count(\*) from EMPLOYEE where Dno=4) ;



d. Không chạy được

Không thể cập nhật dạng xem hoặc hàm 'DEPT\_SUMMARY' vì nó chứa tổng hợp hoặc mệnh đề DISTINCT hoặc GROUP BY hoặc toán tử PIVOT hoặc UNPIVOT.

e. Không chạy được

Không thể cập nhật dạng xem hoặc hàm 'DEPT\_SUMMARY' vì nó chứa tổng hợp hoặc mệnh đề DISTINCT hoặc GROUP BY hoặc toán tử PIVOT hoặc UNPIVOT.