Yu Si

COMP 440

Assignment 2: SQL

Problem 1)

1) Write SQL statement to create the above tables in MySQL DBMS.

```
create table department(
DepartmentID int not null primary key,
DepartmentName varchar(25) not null unique);

create table employee(
EmployeeID int not null primary key,
LastName varchar(25) not null,
DeptId int,
foreign key(DeptId) references department(DepartmentId));
```

2) Write SQL Statement to insert the values into each table (5 pts).

```
insert into department (DepartmentID, DepartmentName)
values(31,'Sales'),
    (33,'Engineering'),
    (34,'Clerical'),
    (35,'Marketing');

insert into employee (EmployeeID, LastName, DeptID)
values(1,'Rafferty',31),
    (2,'Jones',33 ),
    (3,'Heisenberg',33),
    (4,'Robinson',34),
    (5,'Smith',34),
```

```
(6,'Williams',null),
(7,'Brown',null);
```

3) Write SQL statement to add the FirstName column into the Employee table and add the following first names. The structure of the FirstName is similar to the LastName column (varchar(25), not null) (5 pts).

```
alter table employee
add column FirstName varchar(25) not null after EmployeeID;

update employee
set FirstName='John'
where EmployeeID=1;

update employee
set FirstName='Mary'
where EmployeeID=2;

update employee
set FirstName='David'
where EmployeeID=3;

update employee
set FirstName='Bob'
where EmployeeID=4;

update employee
set FirstName='Peter'
where EmployeeID=5;
```

```
update employee
set FirstName='Alice'
where EmployeeID=6;

update employee
set FirstName='Heather'
where EmployeeID=7;
```

4) Write the following join for them (deliver both SQL statements as well as the table result) (5 pts):

a. Cross Join

```
select * from employee
cross join department
where employee.DeptId = department.DepartmentID;
```

EmployeeID	FirstName	LastName	DeptId	DepartmentID	DepartmentName
1	John	Rafferty	31	31	Sales
2	Mary	Jones	33	33	Engineering
3	David	Heisenberg	33	33	Engineering
4	Bob	Robinson	34	34	Clerical
5	Peter	Smith	34	34	Clerical

b. Inner Join

```
select * from employee
inner join department on employee.DeptId = department.DepartmentID;
```

EmployeeID	FirstName	LastName	DeptId	DepartmentID	DepartmentName
1	John	Rafferty	31	31	Sales
2	Mary	Jones	33	33	Engineering
3	David	Heisenberg	33	33	Engineering
4	Bob	Robinson	34	34	Clerical
5	Peter	Smith	34	34	Clerical

c. Left Join

select * from employee

left join department on employee.DeptId = department.DepartmentID;

EmployeeID	FirstName	LastName	DeptId	DepartmentID	DepartmentName
1	John	Rafferty	31	31	Sales
2	Mary	Jones	33	33	Engineering
3	David	Heisenberg	33	33	Engineering
4	Bob	Robinson	34	34	Clerical
5	Peter	Smith	34	34	Clerical
6	Alice	Williams	NULL	NULL	NULL
7	Heather	Brown	NULL	NULL	NULL

d. Right Join

select * from employee

right join department on employee.DeptId = department.DepartmentID;

EmployeeID	FirstName	LastName	DeptId	DepartmentID	DepartmentName
NULL	NULL	NULL	NULL	35	Marketing
1	John	Rafferty	31	31	Sales
2	Mary	Jones	33	33	Engineering
3	David	Heisenberg	33	33	Engineering
4	Bob	Robinson	34	34	Clerical
5	Peter	Smith	34	34	Clerical

5) Delete the employee(s) with no department (Use only ONE SQL statement) (5 pts).

delete from employee where DeptId is null;

6) Delete the Sales department. If you are not able to delete this record, explain why? And how you can solve the problem (5 pts).

Can't delete this record, because table employee has a foreign key(DeptId) references to table department(DepartmentId)). To solve this problem, we can

a. temporarily disable the foreign keys:

SET FOREIGN_KEY_CHECKS=0;

b. then, delete the sales department:

delete from department where DepartmentName='Sales';

c. when we need, we can enable the foreign key:

```
SET FOREIGN_KEY_CHECKS=1;
```

Problem 2)

Query 1) Find all instructors earning the salary higher than the average salary (10 pts).

```
select *
from instructor
where salary > (select avg(salary)
                from instructor);
```

Equivalent Query:

```
select *
from instructor
having salary > (select avg(salary)
                from instructor);
```

Query 2) Find the minimum, maximum, and average salary for each department (10 pts).

```
select *,min(salary),max(salary),avg(salary)
from department as d
join instructor as i on d.dept_name = i.dept_name
group by dept_name;
```

Equivalent Query:

```
select *,min(salary),max(salary),avg(salary)
from department as d
left join instructor as i on d.dept_name = i.dept_name
where salary is not null
group by dept_name;
```

Query 3) Find all the students who take credits between 30 and 100 and order them alphabetically by name (10 pts).

```
select *
from student
where tot_cred between 30 and 100
order by name;
```

Equivalent Query:

```
select *
from student
where tot_cred >29 and tot_cred<101
order by name;
```

Query 4) Find all the instructors with their department name and department building (10 pts).

```
select i.ID, i.name, i.salary, i.dept_name, d.bulding
from instructor as i
join department as d on i.dept_name=d.dept_name;
```

Equivalent Query:

```
select i.ID, i.name, i.salary, i.dept_name, d.bulding
from instructor as i
right join department as d on i.dept_name=d.dept_name
where i.ID is not null;
```

Query 5) Find all the students with their taken courses and grades (10 pts).

```
select s.ID, s.name, s.dept_name, s.tot_cred, t.course_id, t.grade
from student as s
join takes as t on s.ID=t.ID;
```

Equivalent Query:

```
select s.ID, s.name, s.dept_name, s.tot_cred, t.course_id, t.grade
from student as s
right join takes as t on s.ID=t.ID
where s.ID is not null;
```

Query 6) Find the instructor(s) who earns the second highest salary (10 pts).

```
select *, max(salary) as msalary
from instructor
where msalary < (select max(salary)
                 from instructor);
```

Equivalent Query:

```
select *,max(salary) as msalary
from instructor
where msalary not in(select msalary
                     from instructor);
```

Query 7) Increase all credits by 1 for those courses that are taught in semester Fall 2010 (10 pts).

```
update course as c
inner join section as s on c.course_id=s.course_id
set credits = credits + 1
where s.semester = "Fall 2010";
```

Query 8) Delete those instructors who have never taught (10 pts).

```
delete instructor, teaches from instructor
inner join teaches on teaches.ID=instructor.ID
where course_id is null;
```

Equivalent Query:

```
delete instructor from instructor
left join teaches on instructor.ID=teaches.ID
where course_id is null;
```