SQL codes for Chapter 4: Intermediate SQL

Dr. Abu Raihan Mostofa Kamal
Professor, CSE Department
Islamic University of Technology

July 9, 2020

Listing 1: Joins (Inner and Outer)

```
    create table passports
(pid number primary key,
 name varchar2(10),
 issued_on date,
 expired_on date
);

insert into passports values(301,'a',sysdate-100,sysdate+360);
insert into passports values(302,'b',sysdate-10,sysdate+265);
insert into passports values(303,'c',sysdate-35,sysdate+163);

create table driving_lic
(did number primary key,
 name varchar2(10),
 issued_on date,
 expired_on date,
 vehicle varchar2(10),
 passport number,
 constraints fk_driv foreign key(passport) references passports
);


---now insert data into it

insert into driving_lic values(801,'a',sysdate-23,sysdate+230,'Heavy',301);
insert into driving_lic values(802,'c',sysdate-24,sysdate+30,'Light',303);

insert into driving_lic values(803,'n1',sysdate-24,sysdate+30,'Light',null);
insert into driving_lic values(804,'n2',sysdate-224,sysdate+432,'Light',null);
insert into driving_lic values(805,'n3',sysdate-56,sysdate+238,'Light',null);

---Natural join---
select p.pid,p.name,p.issued_on,d.did,d.issued_on,d.vehicle
from passports p, driving_lic d
where p.pid=d.passport;


---left outer ---

select p.pid,p.name,p.issued_on,d.did,d.issued_on,d.vehicle
from passports p, driving_lic d
where p.pid=d.passport(+);

---right outer---
select p.pid,p.name,p.issued_on,d.did,d.issued_on,d.vehicle
from passports p, driving_lic d
where p.pid(+)=d.passport;

---full outer
--new syntax

select p.pid,p.name,p.issued_on,d.did,d.issued_on,d.vehicle
from passports p
full outer join driving_lic d
on p.pid=d.passport;
```

Listing 2: Cascading Delete

```
alter session set "_ORACLE_SCRIPT"=true;

create user rps identified by test123;
grant dba to rps;

--now connect as RPS

create table depts
(id number primary key,
 name varchar2(10)
);

---insert data--

insert into depts values(2,'EEE');
insert into depts values(1,'MCE');
insert into depts values(3,'TVE');
insert into depts values(4,'CSE');
insert into depts values(5,'CEE');

create table students
(sid number primary key,
 name varchar2(10),
 cgpa number(4,2),
 dept number,
 constraints fk_stu foreign key(dept) references depts on delete cascade
);


----insert data---

insert into students values(401,'a1', 3.5,4);
insert into students values(402,'a2', 3.7,4);
insert into students values(403,'a3', 3.2,4);

insert into students values(501,'b1', 3.8,5);
insert into students values(502,'b2', 3.2,5);
insert into students values(503,'b3', 3.3,5);



create table grades
(cid varchar2(10),
 sid number,
 grade varchar2(6),
 constraints fk_grades foreign key(sid) references students on delete cascade
);

---now insert data--

insert into grades values('CSE 4101',401,'A+');
insert into grades values('CSE 4101',402,'A');
insert into grades values('CSE 4101',403,'A-');


insert into grades values('CEE 4101',501,'A');
insert into grades values('CEE 4201',503,'A-');
insert into grades values('CEE 4103',501,'B');
```

```
---NOW LETS SEE THE CASCADING EFFECTS---

--lets delete CEE dept--

delete depts
where id=5;

--you will find there is no students for CEE dept and no grades for those
   students
```

This example is the continuation of the listing 2 except the delete record part. So, run the previous script without the last delete record part.

Listing 3: Role-based Access Control

```
 ---scenario: a simple result processing system (rps)

--objects we have:

 -- 1: depts
 -- 2: students
--- 3: grades

--Lets create another 2 necessary relations

create table employees
(eid number primary key,
 name varchar2(20),
 designation number(1,0), -- 1:Lecturer, 2: Asst. Professor, 3: Assoc. Prof. 4:
   Professor
 RoomNo varchar2(10),
 CounselHour varchar2(15),
 Dept number,
 Salary number,
 PersonalFileNo varchar2(15),
 constraints fk_emp foreign key(Dept) references depts on delete cascade
);

---Lets put some recods

insert into employees values(90001,'Abdul Karim',3,'AB2-301','14:30-15:30'
   ,4,1000,'xyz');
insert into employees values(90002,'Abdur Rahim',1,'AB2-301','14:30-15:30',4,500,
   'xyz1');
insert into employees values(90003,'Abdul Halim',4,'AB2-301','16:30-17:30'
   ,4,2000,'xyz2');
insert into employees values(90004,'Abdullah',4,'AB2-301','15:30-17:30',4,1500,'
   xyz3');

---now another table--

create table course_info
(cid varchar2(15) primary key,
 title varchar2(30),
 credit number(4,2)
);

---insert some date--

insert into course_info values('CSE 4101','Fundamental of Computing',3.0);

insert into course_info values('CEE 4101','Fundamental of CEE',2.0);


---Now create a view for emp--

create or replace view emp_v
as
select name, designation, RoomNo,CounselHour
from employees readonly;
```

```
----followings are the steps to distribute application among different users--

---step 1: role create and customize it
alter session set "_ORACLE_SCRIPT"=true;

create role role_readonly;

-- summary of objects
--COURSE_INFO     TABLE
--DEPTS           TABLE
--EMPLOYEES       TABLE
--EMP_V           VIEW
--GRADES          TABLE
--STUDENTS        TABLE

grant select on course_info to role_readonly;
grant select on depts  to role_readonly;
grant select on emp_v to role_readonly;
grant select on grades to role_readonly;
grant select on students to role_readonly;


---2: role to insert and update values in grades

drop role role_marks_entry;
create role role_marks_entry;

grant insert,update on grades to role_marks_entry;

---role can also be granted to another role

grant role_readonly to role_marks_entry;


---3: role to update course info

drop role role_update_course;
create role role_update_course;

grant insert,update,delete on course_info to role_update_course;

---now I will create few more users--

drop user hod;
drop user reg;
drop user c_teacher;
create user hod identified by test123;
create user reg identified by test123;
create user c_teacher identified by test123;

---dont do in real-life: I am doing to save time!!

grant dba to hod;
grant dba to reg;
grant dba to c_teacher;

grant role_readonly,role_marks_entry to hod;
grant role_readonly,role_update_course to reg;
grant role_readonly,role_marks_entry to c_teacher;
```

```
---now connect to any of these newly created user---
 connect reg/test123;

---reg has no objects, but it can access rps's object by role-based access
   control

select *
from rps.students;
```