- TY B10

```
-- ******************** 1) Department Table ********************
create table department(
        dept_name varchar(100),
        building varchar(100),
        budget numeric,
        primary key(dept_name)
);

insert into department (dept_name, building, budget) values
('Biology', 'Watson', 90000),
('Comp. Sci.', 'Taylor',100000),
('Elec. Eng.', 'Taylor', 85000),
('Finance', 'Painter', 120000),
('History', 'Painter', 50000),
('Music', 'Packard', 80000),
('Physics', 'Watson', 70000);

select * from department;


-- ******************** 2) Course Table ********************

create table course(
        course_id varchar(50),
        title varchar(50),
        dept_name varchar(100),
        credits int,
        primary key(course_id),
        foreign key (dept_name) references department on delete cascade on update cascade
```

```
);

insert into course (course_id, title, dept_name, credits) values
('BIO-101', 'Intro. to Biology', 'Biology', 4),
('BIO-301', 'Genetics', 'Biology', 4),
('BIO-399', 'Computational Biology', 'Biology', 3),
('CS-101', 'Intro. to Computer Science', 'Comp. Sci.', 4),
('CS-190', 'Intro. to Biology', 'Comp. Sci.', 4),
('CS-315', 'Intro. to Biology', 'Comp. Sci.', 4),
('CS-319', 'Intro. to Biology', 'Comp. Sci.', 3),
('CS-347', 'Intro. to Biology', 'Comp. Sci.', 3),
('EE-181', 'Intro. to Biology', 'Elec. Eng.', 3),
('FIN-201', 'Intro. to Biology', 'Finance', 3),
('HIS-351', 'Intro. to Biology', 'History', 3),
('MU-199', 'Intro. to Biology', 'Music', 3),
('PHY-101', 'Intro. to Biology', 'Physics', 4);

select * from course;

-- ********** 3) Instructor Table **********

create table instructor(
        i_id numeric,
        i_name varchar(50),
        dept_name varchar(100),
        salary numeric,
        primary key(i_id),
        foreign key (dept_name) references department on delete cascade on update cascade
);

insert into instructor (i_id, i_name, dept_name, salary) values
```

```
(10101, 'Srinivasan', 'Comp. Sci.', 65000),
(12121, 'Wu', 'Finance', 90000),
(15151, 'Mozart', 'Music', 40000),
(22222, 'Einstein', 'Physics', 95000),
(32343, 'El Said', 'History', 60000),
(33456, 'Gold', 'Physics', 87000),
(45565, 'Katz', 'Comp. Sci.', 75000),
(58583, 'Califieri', 'History', 62000),
(76543, 'Singh', 'Finance', 80000),
(76766, 'Crick', 'Biology', 72000),
(83821, 'Brandt', 'Comp. Sci.', 92000),
(98345, 'Kim', 'Elec. Eng.', 80000);

select * from instructor;

-- ********** 4) Student Table **********

create table student(
        s_id numeric,
        s_name varchar(50),
        dept_name varchar(100),
        tot_cred numeric,
        primary key(s_id),
        foreign key (dept_name) references department on delete cascade on update cascade
);

insert into student (s_id, s_name, dept_name, tot_cred) values
(00128, 'Zhang', 'Comp. Sci.', 102),
(12345, 'Shankar', 'Comp. Sci.', 32),
(19991, 'Brandt', 'History', 80),
(23121, 'Chavez', 'Finance', 110),
```

```
(44553, 'Peltier', 'Physics', 56),
(45678, 'Levy', 'Physics', 46),
(54321, 'Williams', 'Comp. Sci.', 54),
(55739, 'Sanchez', 'Music', 38),
(70557, 'Snow', 'Physics', 0),
(76543, 'Brown', 'Comp. Sci.', 58),
(76653, 'Aoi', 'Elec. Eng.', 60),
(98765, 'Bourikas', 'Elec. Eng.', 98),
(98988, 'Tanaka', 'Biology', 120);

select * from student;

-- ********** 5) Prereq Table **********

create table prereq(
        course_id varchar(50),
        prereq_id varchar(50),
        primary key(course_id),
        foreign key(course_id) references course on delete cascade on update cascade
);

insert into prereq (course_id, prereq_id) values
('BIO-301', 'BIO-101'),
('BIO-399', 'BIO-101'),
('CS-190', 'CS-101'),
('CS-315', 'CS-101'),
('CS-319', 'CS-101'),
('CS-347', 'CS-101'),
('EE-181', 'PHY-101');

select * from prereq;
```

```
-- ******************** 6) Creating advisor Table ********************
CREATE TABLE advisor(
        s_id int,
        i_id int,
        primary key (s_id, i_id),
        foreign key (s_id) references student on delete cascade  on update cascade,
        foreign key (i_id) references instructor on delete cascade  on update cascade
);

insert into advisor (s_id, i_id)
        select s_id, i_id
        from student, instructor;

SELECT * FROM advisor;

-- ******************** 7) Creating classroom Table ********************
CREATE TABLE classroom(
        building varchar(50),
        room_no int,
        capacity int,
        primary key (building, room_no)
);

insert into classroom (building, room_no, capacity) values
('Painter', 514, 60),
('Packard', 101, 60),
```

```
('Watson', 100, 60),
('Taylor', 3128, 60),
('Watson', 120, 60);

SELECT * FROM classroom;


-- ******************** 8) Creating time_slot Table ********************
CREATE TABLE time_slot(
        time_slot_id char(1),
        day varchar(50),
        start_time varchar(10),
        end_time varchar(10),
        primary key (time_slot_id)
);

insert into time_slot (time_slot_id, day, start_time, end_time) values
('A', 'Saturday', '09:30 am', '10:30 am'),
('B', 'Saturday', '09:30 am', '10:30 am'),
('C', 'Saturday', '09:30 am', '10:30 am'),
('D', 'Saturday', '09:30 am', '10:30 am'),
('E', 'Saturday', '09:30 am', '10:30 am'),
('F', 'Saturday', '09:30 am', '10:30 am'),
('G', 'Saturday', '09:30 am', '10:30 am'),
('H', 'Saturday', '09:30 am', '10:30 am');


SELECT * FROM time_slot;


-- ******************** 9) Creating section Table ********************
```

```
CREATE TABLE section(
        course_id varchar(50),
        sec_id int,
        sem varchar(50),
        year int,
        building varchar(100),
        room_no int,
        time_slot_id char(1),
        primary key (course_id, sec_id, sem, year),
        foreign key (course_id) references course on delete cascade  on update cascade,
        foreign key (building, room_no) references classroom(building, room_no) on delete cascade
on update cascade,
        foreign key (time_slot_id) references time_slot on delete cascade  on update cascade
);

insert into section(course_id, sec_id, sem, year, building, room_no, time_slot_id) values
('BIO-101', 1, 'Summer', 2009, 'Painter', 514, 'B'),
('CS-101', 1, 'Summer', 2008, 'Painter', 514, 'A'),
('EE-181', 2, 'Fall', 2006, 'Taylor', 3128, 'C'),
('FIN-201', 1, 'Fall', 2007, 'Watson', 120, 'D'),
('HIS-351', 2, 'Spring', 2011, 'Watson', 100, 'E'),
('PHY-101', 1, 'Spring', 2010, 'Packard', 101, 'F');

select * from section;

-- ******************** 10) Creating takes Table ********************

create table takes(
        s_id numeric,
        course_id varchar(50),
```

```
	sec_id int,
	sem varchar(50),
	year int,
	grade varchar(3),
	primary key(s_id, course_id, sec_id, sem, year),
	foreign key(s_id) references student on delete cascade on update cascade,
	foreign key(course_id, sec_id, sem, year) references section on delete cascade on update
cascade
);

insert into takes(s_id, course_id, sec_id, sem, year)
	select s_id, course_id, sec_id, sem, year
	from student, section;

update takes
set grade = 'F';

select * from takes;

-- ******************** 11) Creating teaches Table ********************

create table teaches(
	i_id numeric,
	course_id varchar(50),
	sec_id int,
	sem varchar(50),
	year int,
	primary key(i_id, course_id, sec_id, sem, year),
	foreign key(i_id) references instructor on delete cascade on update cascade,
	foreign key(course_id, sec_id, sem, year) references section on delete cascade on update
cascade
);
```

```
insert into teaches(i_id, course_id, sec_id, sem, year)
        select i_id, course_id, sec_id, sem, year
        from instructor, section;

select * from teaches;

-- -- -- -- -- -- -- -- -- -- -- -- -- -- -- -- -- -- -- -- -- -- -- -- -- -- -- -- -- -- -- --

-- 1. Find the department names of all instructors, and remove duplicates.
select distinct dept_name
from instructor;

-- 2. Find all instructors in Comp. Sci. dept with salary > 70000.
select *
from instructor
where dept_name = 'Comp. Sci.' and salary > 70000;

-- 3. Find the course ID, semester, year and title of each course offered by the Comp. Sci. department
select course_id, sem as semester, year, title
from course natural join section
where dept_name = 'Comp. Sci.';

-- 4. Find the names of all departments whose building name includes the substring ‘Watson’.
select dept_name
from department
where building like '%Watson%';

-- 5. Find all instructors who appear in the instructor relation with null values for salary.
select *
from instructor
```

```
where salary is null;

-- -- -- -- -- -- -- -- -- -- -- -- -- -- -- -- -- -- -- -- -- -- -- -- -- -- -- -- -- -- -- -- -- -- -- -- --

-- 1. Annual salary increases are being made, and salaries of all instructors are to be increased by 5 percent.

update instructor

set salary = salary * 1.05;

-- 2. Annual salary increases are being made, and salaries of all instructors are to be increased by 5 percent with salary of less than $70,000.

update instructor

set salary = salary * 1.05

where salary < 70000;

-- 3. Give a 5 percent salary raise to instructors whose salary is less than average.

update instructor

set salary = salary * 1.05

where salary < all(select avg(salary) from instructor);

-- 4. All instructors with salary over $100,000 receive a 3 percent raise, whereas all others receive a 5 percent raise (use case statement).

update instructor

set salary = case

                    when salary > 100000 then salary * 1.03

                    else salary * 1.05

                    end;

-- -- -- -- -- -- -- -- -- -- -- -- -- -- -- -- -- -- -- -- -- -- -- -- -- -- -- -- -- -- -- -- -- -- -- -- --

-- 1. Delete all tuples in the instructor relation pertaining to instructors in the Finance department.

delete from instructor
```

```
where dept_name = 'Finance';


-- 2. Delete all instructors with a salary between $13,000 and $15,000.

delete from instructor

where salary between 13000 and 15000;


-- 3. Delete all tuples in the instructor relation for those instructors associated with a department
located in the Watson building.

delete from instructor

where dept_name = some(select dept_name from department where building = 'Watson');


-- 4. Delete the records of all instructors with salary below the average at the university.

delete from instructor

where salary < all(select avg(salary) from instructor);
```