Final Project Report

CISC 660 Database Management Systems

Winter 2016

Instructor: Dr. Junping Sun

Nova Southeastern University

College of Engineering and Computing

Master of Science in Computer Science

Submitted by:

Ranon Martin

rm1905@nova.edu

1. Partition the sentences

General Sentence:

In a university, we represent data about both students and employees.

Sentences about students:

The university keeps track of each student's name, student number, social security number, address, phone, birth date, sex, class (freshman, sophomore,..., graduate), major department, minor department (if any), and degree program (B.A., B.S., M.A., M.S., ..., Ph.D.). Some user applications need to access the city, state, and zip code of the student's address and the student last name. Both social security number and student number have unique values for each student. Each student has a study plan that shows list of required courses to be taken.

Students may have a transcript for all the courses they have taken. For graduate students, the student’s advisor should be included in the database.

Sentences about department:

Each department is described by a name, department number, office number, office phone, and college. Both department name and department number have unique values for each department. Each department has a Chairperson or a Dean in charge of that department.

Sentences about courses:

Each course has a course name, course number, number of semester hours (credit), and offering department. Some courses have prerequisites (please pay attention here).

Each section has an instructor, semester, year, course, and section number. The section number distinguishes different sections of the same course that is taught during the same semester/year (may be at the same time), its values are 1, 2, 3, ..., up to the number of sections taught during each semester. Each course has the day, meeting time, place where the class is held. A grade report for a course has a student name, section number, and grades.

Sentences about employees:

Employees are classified into faculty and staff, both of them have dependents, the database stores the information of employees' dependents for the insurance and benefit purposes. Faculty could be full-time or part-time employees. Professors have ranks (Lecturer, Assistant Professor, Associate Professor, Full Professor) and salaries. Faculties (Professors) may hold different degree (highest degree is only considered here). Each professor belongs to at least one department. Professors may have joint appointments from other department(s).

Staff are secretaries, program coordinators, assistant directors, directors, deans, vice presidents, and president.

2. Conceptual schema design

ERRD: Please see the attached document for the Enhanced-Entity Relation Diagram

CSDL:

Schema:	UNIVERSITY		
Entity:	PERSON		
	Attributes:	SSN:	text (9)
		Phone:	text (10)
		DOB:	text (9)
		Sex:	enumeration [M, F]
	Composite attributes: Name of		
		Fname:	text (20)
		Minit:	text (1)
		Lname:	text (20)
	Composite attributes: Address of		
		Street:	text (30)
		City:	text (30)
		State:	text (2)
		Zipcode:	text (5)
	Identifiers:	SSN	

Entity:	STUDENT		
	Attributes:	StudentNo:	text (12)
		Class:	text (20)
		DegreeProg:	text (3)
	Identifiers:	StudentNo	
Entity:	GRAD_STUDENT		
Entity:	EMPLOYEE		
Entity:	STAFF		
	Attributes:	Title:	text (20)
Entity:	FACULTY		
	Attributes:	Rank:	text (20)
	Attributes:	Degree:	text (10)
	Attributes:	Salary:	real
Entity:	FULL_TIME		
Entity:	PART_TIME		

Entity:	DEPENDENT		
	Attributes:	Relationship:	text (20)
		DOB:	text (9)
		Sex:	enumeration [M, F]
		Name:	text (40)

Entity:	DEPARTMENT		
	Attributes:	OfficeNo:	text (5)
		Dname:	text (50)
		Dno:	integer
		College:	text (50)
		OfficePh:	text (10)
	Identifiers:	Dno, Dname	

Entity:	COURSE		
	Attributes:	Cno:	text (10)
		Credits:	integer

		Cname:	text (50)
	Identifiers:	Cno	
Entity:	SECTION		
	Attributes:	Sno:	integer
		Year:	integer
		Semester:	text (6)
	Composite attributes: Meeting of		
		Time:	text (7)
		Day:	text (9)
		Room:	text (5)
	Identifiers:	Sno, Year, Semester	

Generalization:	PERSON_TYPE	
	Father:	PERSON
	Sons:	STUDENT, EMPLOYEE

Generalization: EMPLOYMENT_ROLE

	Father:	EMPLOYEE
	Sons:	STAFF, FACULTY
Generalization:	EMPLOYMENT_TYPE	
	Father:	FACULTY
	Sons:	FULL_TIME, PART_TIME
Generalization:	STUDENT_CLASS	
	Father:	STUDENT
	Sons:	GRAD_STUDENT

Relationship:	MINOR
Connected entities:	(0, N) STUDENT
	(0, 1) DEPARTMENT
Relationship:	MAJOR
Connected entities:	(0, N) STUDENT
	(1, 1) DEPARTMENT
Relationship:	STUDY_PLAN
Connected entities:	(0, N) STUDENT
	(1, N) COURSE

Relationship:	TRANSCRIPT	
Connected entities:	(0, N) STUDENT	
	(0, N) COURSE	
Attributes:	Grade:	text (2)

Relationship:	GRADE_REPORT	
Connected entities:	(0, N) STUDENT	
	(0, N) SECTION	
Attributes:	Grade:	text (2)
	StudentName:	text (43)

Relationship:	ADVICES
Connected entities:	(1, 1) FACULTY
	(0, N) GRAD_STUDENT

Relationship:	OFFERS
Connected entities:	(1, 1) DEPARTMENT
	(0, N) COURSE

Relationship:	PRODUCES
Connected entities:	(1, 1) COURSE

	(0, N) SECTION
Relationship:	WORKS_FOR
Connected entities:	(1, N) STAFF
	(1, 1) DEPARTMENT
Relationship:	CHAIRS
Connected entities:	(1, 1) STAFF
	(0, 1) DEPARTMENT
Relationship:	REQUIRES
Connected entities:	(0, N) COURSE
	(0, N) COURSE
Relationship:	DEPENDENT_OF
Connected entities:	(0, N) DEPENDENT
	(1, 1) EMPLOYEE
Relationship:	APPOINTED_TO
Connected entities:	(0, N) FACULTY
	(1, N) DEPARTMENT

Relationship:	TEACHES
Connected entities:	(1, 1) FACULTY
	(0, N) SECTION

3. ER schema to relational database schema transformation

Key

FD: Functional dependency

PK: Primary key

FK: Foreign key

CK: Candidate key

PERSON (**<u>ssn:</u>** text (9), fname: text (20), minit: text (1), lname: text (20), sex: enum, dob: text (9), street: text (30), city: text (30), state: text (2), zipcode: text (5), phone: text (10))

FD1: ssn -> fname, minit, lname, sex, dob, street, city, state, zipcode, phone

FD2: zipcode -> city, state

FD3: street, city, state -> zipcode

PK: ssn

EMPLOYEE (**<u>ssn</u>:** text (9))

PK: ssn

FK: ssn

Referential integrity constraint:

- The foreign key ssn in the EMPLOYEE table references to the primary key ssn in the PERSON table.

DEPENDENT (**<u>name</u>**: text (40), **<u>essn</u>**: text (9), relationship: text (20), dob: text (9), sex: enum)

PK: essn, name

FK: essn

Referential integrity constraint:

- The foreign key essn in the DEPENDENT table references to the primary key ssn in the EMPLOYEE table.

STUDENT (**ssn**: text (9), student_no: text (12), class: text (20), degree_prog: text (3), minor_dept: integer, major_dept: integer)

FD1: ssn -> student_no, class, degree_prog, minor_dept, major_dept

FD2: student_no -> ssn, class, degree_prog, minor_dept, major_dept

FD3: ssn, student_no -> class, degree_prog, minor_dept, major_dept

CKs: ssn, student_no

PK: ssn

FK: ssn, minor_dept, major_dept

Referential integrity constraint:

- The foreign key ssn in the STUDENT table references to the primary key ssn in the PERSON table.
- The foreign key minor_dept in the STUDENT table references to the primary key dno in the DEPARTMENT table.
- The foreign key major_dept in the STUDENT table references to the primary key dno in the DEPARTMENT table.

GRAD_STUDENT (**student_ssn**: text (9), advisor_ssn: text (9))

FD1: student_no -> advisor_ssn

PK: student_ssn

FK: student_ssn, advisor_ssn

Referential integrity constraint:

- The foreign key student_ssn in the GRAD_STUDENT table references to the primary key ssn in the STUDENT table.
- The foreign key advisor_ssn in the GRAD_STUDENT table references to the primary key ssn of the FACULTY table.

STAFF (<u>**ssn**</u>: text (9), title: text (20), dno: integer)

FD1: ssn -> title, dno

PK: ssn

FK: ssn

Referential integrity constraint:

- The foreign key ssn in the STAFF table references to the primary key ssn in the EMPLOYEE table.

FACULTY (**ssn**: text (9), rank: text (20), degree: text (10), salary: decimal, employment_type: text (9))

FD1: ssn -> rank, degree, salary, employment_type

FD2: rank, employment_type -> salary

PK: ssn

FK: ssn

Referential integrity constraint:

- The foreign key ssn in the FACULTY table references to the primary key ssn in the EMPLOYEE table.

DEPARTMENT (**dno**: integer, dname: text (50), college: text (50), office_ph: text (10), office_no: text (5), chair_ssn: text (9))

FD1: dno -> dname, college, office_ph, office_no, chair_ssn

FD2: dname -> dno, college, office_ph, office_no, chair_ssn

FD3: chair_ssn -> dno

FD4: office_ph -> office_no

FD5: office_no -> dno

CKs: dno, dname

PK: dno

FK: chair_ssn

Referential integrity constraint:

- The foreign key chair_ssn in the DEPARTMENT table references to the primary key ssn in the STAFF table.

APPOINTED_TO (**<u>fssn</u>**: text (9), **<u>dno</u>**: text (9))

PK: fssn, dno

FKs: fssn, dno

Referential integrity constraint:

- The foreign key fssn in the APPOINTED_TO table references to the primary key ssn in the FACULTY table.

- The foreign key dno in the APPOINTED_TO table references to the primary key dno in the DEPARTMENT table.

COURSE (**<u>cno</u>**: text (10), cname: text (50), credits: integer, dno: integer)

FD1: cno -> cname, credits, dno

PK: cno

FK: dno

Referential integrity constraint:

- The foreign key dno in the COURSE table references to the primary key dno in the DEPARTMENT table.

COURSE_PREREQUISITE (**<u>cno</u>**: text (10), **<u>prerequisite_cno</u>**: text (10))

PK: cno, prerequisite_cno

FK: cno, prerequisite_cno

Referential integrity constraint:

- The foreign key cno in the COURSE_PREREQUISITE table references to the primary key cno in the COURSE table.
- The foreign key prerequisite_cno in the COURSE_PREREQUISITE table references to the primary key cno in the COURSE table.

STUDY_PLAN (**student_ssn**: text (9), **cno**: text (10))

PK: student_ssn, cno

FK: student_ssn, cno

Referential integrity constraint:

- The foreign key student_ssn in the STUDY_PLAN table references to the primary key ssn in the STUDENT table.
- The foreign key cno in the STUDY_PLAN table references to the primary key cno in the COURSE table.

TRANSCRIPT (**student_ssn**: text (9), **cno**: text (10), grade: text (2))

FD1: student_ssn, cno -> grade

PK: student_ssn, cno

FK: student_ssn, cno

Referential integrity constraint:

- The foreign key student_ssn in the TRANSCRIPT table references to the primary key ssn in the STUDENT table.
- The foreign key cno in the TRANSCRIPT table references to the primary key cno in the COURSE table.

SECTION (**<u>sno</u>**: integer, **<u>semester</u>**: text (6), **<u>year</u>**: integer, meeting_day: text (9), meeting_time: text (7), meeting_room: text (5), instructor: text (9), cno: text (10))

FD1: sno, semester, year, cno -> meeting_day, meeting_time, meeting_room, instructor

FD2: cno, semester, year, meeting_time, meeting_day, meeting_room -> sno, instructor

PK: sno, semester, year, cno

FK: instructor, cno

Referential integrity constraint:

- The foreign key instructor in the SECTION table references to the primary key ssn in the FACULTY table.

- The foreign key cno in the SECTION table references to the primary key cno in the COURSE table.

GRADE_REPORT (**student_ssn**: text (9), **sno**: integer, **semester**: text (6), **year**: integer, **cno**: text (10), grade: text (2))

FD1: student_ssn, sno, semester, year, cno -> grade

PK: student_ssn, sno, semester, year, cno

FK: student_ssn, {sno, semester, year, cno}

Referential integrity constraint:

- The foreign key student_ssn in the GRADE_REPORT table references to the primary key ssn in the STUDENT table.
- The foreign key {sno, semester, year, cno} in the GRADE_REPORT table references to the primary key {sno, semester, year, cno} in the SECTION table.

4. Join paths

a. The join condition that exists between EMPLOYEE (e) and PERSON (p) is

   e.ssn = p.ssn

b. The join condition that exists between DEPENDENT(d) and EMPLOYEE(e) is

   d.essn = e.ssn

c. The join condition that exists between STUDENT(s) and PERSON(p) is

   s.ssn = p.ssn

d. The join condition that exists between STUDENT(s) and DEPARTMENT(d) is

   s.minor_dept = d.dno

e. The join condition that exists between STUDENT(s) and DEPARTMENT(d) is

   s.major_dept = d.dno

f. The join condition that exists between GRAD_STUDENT(g) and STUDENT(s) is

   g.student_ssn = s.ssn

g. The join condition that exists between GRAD_STUDENT(g) and FACULTY(f) is

   g.advisor_ssn = f.ssn

h. The join condition that exists between STAFF(s) and EMPLOYEE(e) is

   s.ssn = e.ssn

i. The join condition that exists between FACULTY(f) and EMPLOYEE(e) is

   f.ssn = e.ssn

j. The join condition that exists between DEPARTMENT(d) and STAFF(s) is

   d.chair_ssn = s.ssn

k. The join condition that exists between APPOINTED_TO(a) and FACULTY(f) is

   a.fssn = f.ssn

l. The join condition that exists between COURSE(c) and DEPARTMENT(d) is

   c.dno = d.dno

m. The join condition that exists between COURSE_PREREQUISITE(cp) and COURSE(c) is

   cp.cno = c.cno

n. The join condition that exists between COURSE_PREREQUISITE(cp) and COURSE(c) is

   cp.prerequisite_cno = c.cno

o. The join condition that exists between STUDY_PLAN(sp) and STUDENT(s) is

   sp.student_ssn = s.ssn

p. The join condition that exists between STUDY_PLAN(sp) and COURSE (c) is

   sp.cno = c.cno

q. The join condition that exists between TRANSCRIPT(t) and STUDENT(s) is

   t.student_ssn = s.ssn

r. The join condition that exists between TRANSCRIPT(t) and COURSE(c) is

   t.cno = c.cno

s. The join condition that exists between SECTION(s) and FACULTY(f) is

   s.instructor = f.ssn

t. The join condition that exists between SECTION(s) and COURSE(c) is

   s.cno = c.cno

u. The join condition that exists between GRADE_REPORT(g) and STUDENT(s) is

   g.student_ssn = s.ssn

v. The join condition that exists between GRADE_REPORT(g) and SECTION(s) is g.sno = s.sno and g.semester = s.semester and g.year = s.year and g.cno = s.cno

5. Normalized relation schema in 3NF

PERSON (**<u>ssn</u>**: text (9), fname: text (20), minit: text (1), lname: text (20), sex: enum, dob: text (9), street: text (30), zipcode: text (5), phone: text (10))

PK: ssn

FK: zipcode

Referential integrity constraint:

- The foreign key zipcode in the PERSON table references to the primary key zipcode in the CITY_STATE table.

CITY_STATE (**<u>zipcode</u>**: text (5), city: text (30), state: text (2))

PK: zipcode

Note: Normalizing the derived PERSON relation to 3NF results in the above PERSON and CITY_STATE relations.

EMPLOYEE (**<u>ssn</u>**: text (9))

PK: ssn

FK: ssn

Referential integrity constraint:

- The foreign key ssn in the EMPLOYEE table references to the primary key ssn in the PERSON table.

DEPENDENT (**<u>name</u>**: text (40), **<u>essn</u>**: text (9), relationship: text (20), dob: text (9), sex: enum)

PK: name, essn

FK: essn

Referential integrity constraint:

- The foreign key essn in the DEPENDENT table references to the primary key ssn in the EMPLOYEE table.

STUDENT (**<u>ssn</u>**: text (9), <u>student_no</u>: text (12), degree_prog: text (3), minor_dept: integer, major_dept: integer, class: text (20))

CKs: ssn, student_no

PK: ssn

FK: ssn, minor_dept, major_dept

Referential integrity constraint:

- The foreign key ssn in the STUDENT table references to the primary key ssn in the PERSON table.
- The foreign key minor_dept in the STUDENT table references to the primary key dno in the DEPARTMENT table.
- The foreign key major_dept in the STUDENT table references to the primary key dno in the DEPARTMENT table.

GRAD_STUDENT (**<u>student_ssn</u>**: text (9), advisor_ssn: text (9))

PK: student_ssn

FK: student_ssn, advisor_ssn

Referential integrity constraint:

- The foreign key student_ssn in the GRAD_STUDENT table references to the primary key ssn in the STUDENT table.
- The foreign key advisor_ssn in the GRAD_STUDENT table references to the primary key ssn in the FACULTY table.

STAFF (**<u>ssn</u>**: text (9), title: text (20), dno: integer)

PK: ssn

FK: ssn, dno

Referential integrity constraint:

- The foreign key ssn in the STAFF table references to the primary key ssn in the EMPLOYEE table.
- The foreign key dno in the STAFF table references to the primary key dno in the DEPARTMENT table.

FACULTY (**<u>ssn</u>**: text (9), degree: text (10), rank: text (20), employment_type: text (9))

PK: ssn

FK: ssn, {rank, employment_type}

Referential integrity constraint:

- The foreign key ssn in the FACULTY table references to the primary key ssn in the EMPLOYEE table.
- The foreign key {rank, employment_type} in the FACULTY table references to the primary key {rank, employment_type} in the SALARY_SCALE table.

SALARY_SCALE (**rank**: text (20), **employment_type**: text (9), salary: decimal)

PK: rank, employment_type

Note: Normalizing the derived FACULTY relation to 3NF results in the above FACULTY and SALARY_SCALE relations.

DEPARTMENT (**dno**: integer, dname: text (50), college: text (50), office_no: text (5), chair_ssn: text (9))

CKs: dno, dname

PK: dno

FKs: office_no, chair_ssn

Referential integrity constraint:

- The foreign key office_no in the DEPARTMENT table references to the primary key office_no in the OFFICE table.
- The foreign key chair_ssn in the DEPARTMENT table references to the primary key ssn in the STAFF table.

OFFICE (**office_no**: text (5), office_ph: text (10))

PK: office_no

Note: Normalizing the derived DEPARTMENT relation to 3NF results in the above DEPARTMENT and OFFICE relations.

APPOINTED_TO (**fssn**: text (9), **dno:** integer)

PK: fssn, dno

FK: fssn, dno

Referential integrity constraint:

- The foreign key fssn in the APPOINTED_TO table references to the primary key ssn in the FACULTY table.
- The foreign key dno in the APPOINTED_TO table references to the primary key dno in the DEPARTMENT table.

COURSE (**cno**: text (10), cname: text (50), credits: integer, dno: integer)

PK: cno

FK: dno

Referential integrity constraint:

- The foreign key dno in the COURSE table references to the primary key dno in the DEPARTMENT table.

COURSE_PREREQUISITE (**<u>cno</u>**: text (10), **<u>prerequisite_cno</u>**: text (10))

PK: cno, prerequisite_cno

FK: cno, prerequisite_cno

Referential integrity constraint:

- The foreign key cno in the COURSE_PREREQUISITE table references to the primary key cno in the COURSE table.
- The foreign key prerequisite_cno in the COURSE_PREREQUISITE table references to the primary key cno in the COURSE table.

STUDY_PLAN (**<u>student_ssn</u>**: text (9), **<u>cno</u>**: text (10))

PK: student_ssn, cno

FK: student_ssn, cno

Referential integrity constraint:

- The foreign key student_ssn in the STUDY_PLAN table references to the primary key ssn in the STUDENT table.
- The foreign key cno in the STUDY_PLAN table references to the primary key cno in the COURSE table.

TRANSCRIPT (**<u>student_ssn</u>**: text (9), **<u>cno</u>**: text (10), grade: text (2))

PK: student_ssn, cno

FK: student_ssn, cno

Referential integrity constraint:

- The foreign key student_ssn in the TRANSCRIPT table references to the primary key ssn in the STUDENT table.
- The foreign key cno in the TRANSCRIPT table references to the primary key cno in the COURSE table.

SECTION (**<u>sno</u>**: integer, **<u>semester</u>**: text (6), **<u>year</u>**: integer, **<u>cno</u>**: text (10), meeting_day: text (9), meeting_time: text (7), meeting_room: text (5), instructor: text (9))

PK: sno, semester, year, cno

FK: instructor, cno

Referential integrity constraint:

- The foreign key instructor in the SECTION table references to the primary key ssn in the FACULTY table.
- The foreign key cno in the SECTION table references to the primary key cno in the COURSE table.

GRADE_REPORT (**<u>student_ssn</u>**: text (9), **<u>sno</u>**: integer, **<u>semester</u>**: text (6), **<u>year</u>**: integer, **<u>cno</u>**: text (10), grade: text (2))

PK: student_ssn, sno, semester, year, cno

FK: student_ssn, {sno, semester, year, cno}

Referential integrity constraint:

- The foreign key student_ssn in the GRADE_REPORT table references to the primary key ssn in the STUDENT table.
- The foreign key {sno, semester, year, cno} in the GRADE_REPORT table references to the primary key {sno, semester, year, cno} in the SECTION table.

6. Relational database implementation

Database schema creation SQL statements:

```
create table CITY_STATE

(

  zipcode char(5) not null,

  city varchar2(30),

  state char(2),

  constraint cityPK primary key (zipcode)

);


create table PERSON

(

  ssn char(9) not null,

  fname varchar2(20) not null,

  minit char(1),

  lname varchar2(20) not null,

  sex char(1) check (sex in ('M', 'F')),

  dob date,

  street varchar2(30),

  zipcode char(5),
  phone char(10),
  constraint personPK primary key (ssn)
);

create table EMPLOYEE
(
  ssn char(9) not null,
  constraint empPK primary key (ssn),
  constraint emppersonFRK foreign key (ssn) references person(ssn) on delete cascade
);

create table DEPENDENT
(
  name varchar2(40) not null,
  essn char(9) not null,
  relationship varchar2(20),
  dob date,
  sex char(1) check (sex in ('M', 'F')),
  constraint dependentPK primary key (name, essn)
);
```

```
create table STUDENT

(

  ssn char(9) not null,

  student_no char(12) not null,

  degree_prog varchar2(3),

  minor_dept number(5,0),

  major_dept number(5,0) not null,

  class varchar2(20),

  constraint studentPK primary key (ssn),

  constraint personstuFRK foreign key (ssn) references person(ssn) on delete cascade,

  constraint stunumUK unique (student_no)

);


create table STAFF

(

  ssn char(9) not null,

  title varchar2(20),

  dno number(5,0),

  constraint staffPK primary key (ssn),

  constraint empstaffFRK foreign key (ssn) references employee(ssn) on delete cascade

);
```

```
create table FACULTY

(

  ssn char(9) not null,

  degree varchar2(10),

  rank varchar2(20),

  employment_type varchar2(9),

  constraint facultyPK primary key (ssn),

  constraint empfacultyFRK foreign key (ssn) references employee(ssn) on delete

cascade

);


create table SALARY_SCALE

(

  rank varchar2(20) not null,

  employment_type varchar2(9) not null,

  salary number(8,2),

  constraint salscalePK primary key (rank, employment_type)

);


create table GRAD_STUDENT

(

  student_ssn char(9) not null,
  advisor_ssn char(9) not null,
  constraint gradstudentPK primary key (student_ssn),
  constraint advisorFRK foreign key (advisor_ssn) references faculty(ssn) on delete set
null,
  constraint gradstudentFRK foreign key (student_ssn) references student(ssn) on delete
cascade
);

create table DEPARTMENT
(
  dno number(5,0) not null,
  dname varchar2(50) not null,
  college varchar2(50),
  office_no varchar2(5),
  chair_ssn char(9),
  constraint dnameUK unique (dname),
  constraint deptPK primary key (dno),
  constraint deptchairFRK foreign key (chair_ssn) references staff(ssn) on delete set null
);

create table OFFICE
(
  office_no varchar2(5) not null,

  office_ph char(10),

  constraint officenoPK primary key (office_no)

);


create table APPOINTED_TO

(

  fssn char(9) not null,

  dno number(5,0) not null,

  constraint appointedPK primary key (fssn, dno),

  constraint appfssnFRK foreign key (fssn) references faculty(ssn) on delete cascade,

  constraint appdnoFRK foreign key (dno) references department(dno) on delete cascade

);


create table COURSE

(

  cno varchar2(10) not null,

  cname varchar2(50),

  credits number(5,0),

  dno number(5,0),

  constraint coursePK primary key (cno),
  constraint coursednoFRK foreign key (dno) references department(dno) on delete
cascade
);

create table COURSE_PREREQUISITE
(
  cno varchar2(10) not null,
  prerequisite_cno varchar2(10) not null,
  constraint courseprereqPK primary key (cno, prerequisite_cno),
  constraint cno1FRK foreign key (cno) references course(cno) on delete cascade,
  constraint cno2FRK foreign key (prerequisite_cno) references course(cno) on delete
cascade
);

create table STUDY_PLAN
(
  student_ssn char(9) not null,
  cno varchar2(10) not null,
  constraint studyplanPK primary key (student_ssn, cno),
  constraint spstuFRK foreign key (student_ssn) references student(ssn) on delete
cascade,
  constraint spcnoFRK foreign key (cno) references course(cno) on delete cascade
```

```
);

create table TRANSCRIPT
(
  student_ssn char(9) not null,
  cno varchar2(10) not null,
  grade varchar2(2) not null,
  constraint transcriptPK primary key (student_ssn, cno),
  constraint tstuFRK foreign key (student_ssn) references student(ssn) on delete
cascade,
  constraint tcnoFRK foreign key (cno) references course(cno) on delete cascade
);

create table SECTION
(
  sno number(5,0) not null,
  semester varchar2(6) not null,
  year number(5,0) not null,
  meeting_day varchar2(9),
  meeting_time varchar2(7),
  meeting_room varchar2(5),
  instructor char(9),
  cno varchar2(10) not null,
  constraint sectionPK primary key (sno, semester, year, cno),
  constraint instructorFRK foreign key (instructor) references faculty(ssn) on delete
cascade,
  constraint sectioncnoFRK foreign key (cno) references course(cno) on delete cascade
);

create table GRADE_REPORT
(
  student_ssn char(9) not null,
  sno number(5,0) not null,
  semester varchar2(6) not null,
  year number(4,0) not null,
  cno varchar2(10) not null,
  grade varchar2(2),
  constraint greportPK primary key (student_ssn, sno, semester, year, cno),
  constraint grssnFRK foreign key (student_ssn) references student(ssn) on delete
cascade,
  constraint grsectionFRK foreign key (sno, semester, year, cno) references section(sno,
semester, year, cno) on delete cascade
);
```

```
alter table PERSON add

(

  constraint zipcodeFRK foreign key (zipcode) references city_state(zipcode) on delete

set null

);


alter table DEPENDENT add

(

  constraint deppersonFRK foreign key (essn) references employee(ssn) on delete

cascade

);


alter table FACULTY add

(

  constraint salaryFRK foreign key (employment_type, rank) references

salary_scale(employment_type, rank) on delete cascade

);


alter table DEPARTMENT add

(

  constraint deptofficeFRK foreign key (office_no) references office(office_no) on delete

set null
);


alter table STAFF add
(
  constraint staffdnoFRK foreign key (dno) references department(dno) on delete
cascade
);


alter table STUDENT add
(
  constraint mindeptFRK foreign key (minor_dept) references department(dno) on
delete set null,
  constraint majdeptFRK foreign key (major_dept) references department(dno) on
delete cascade
);
```

Data loading SQL statements:

Populate persons' table

```
insert into PERSON (ssn, fname, minit, lname, sex, dob, street, zipcode, phone) values
('961705899', 'Fred', 'R', 'Castillo', 'M', '13-JAN-45', '1558 Mandrake Hill', '33430',
'7505900484');

insert into PERSON (ssn, fname, minit, lname, sex, dob, street, zipcode, phone) values
('755215319', 'Melissa', 'K', 'Larson', 'F', '08-JAN-65', '04434 Buell Street', '33431',
'8568427137');

insert into PERSON (ssn, fname, minit, lname, sex, dob, street, zipcode, phone) values
('505329915', 'Shirley', 'M', 'Woods', 'F', '20-JAN-48', '2 Westridge Street', '33063',
'6483545531');

insert into PERSON (ssn, fname, minit, lname, sex, dob, street, zipcode, phone) values
('861648651', 'Carolyn', 'B', 'Ross', 'F', '05-JUL-51', '2211 Orin Alley', '33133',
'1809101554');

insert into PERSON (ssn, fname, minit, lname, sex, dob, street, zipcode, phone) values
('119292173', 'Christina', 'C', 'Carroll', 'F', '02-JUN-75', '89 Merry Street', '33430',
'1764417503');
```

```
insert into PERSON (ssn, fname, minit, lname, sex, dob, street, zipcode, phone) values
('858270649', 'Roy', 'T', 'Peters', 'M', '05-FEB-48', '3 Washington Terrace', '33054',
'1158723225');

insert into PERSON (ssn, fname, minit, lname, sex, dob, street, zipcode, phone) values
('158506058', 'Shirley', 'C', 'Gilbert', 'F', '04-JUL-72', '4 Morningstar Plaza', '33146',
'3622284510');

insert into PERSON (ssn, fname, minit, lname, sex, dob, street, zipcode, phone) values
('639785819', 'Terry', 'J', 'Smith', 'M', '06-DEC-60', '3 Cordelia Hill', '33056',
'5632067474');

insert into PERSON (ssn, fname, minit, lname, sex, dob, street, zipcode, phone) values
('183338056', 'Antonio', 'D', 'Wright', 'M', '23-DEC-46', '136 Prairie Rose Pass', '33056',
'1064106818');

insert into PERSON (ssn, fname, minit, lname, sex, dob, street, zipcode, phone) values
('816694343', 'Katherine', 'J', 'Holmes', 'F', '12-JUL-59', '6274 Dexter Avenue', '33426',
'9178038274');
insert into PERSON (ssn, fname, minit, lname, sex, dob, street, zipcode, phone) values
('477685256', 'Willie', 'H', 'Hernandez', 'M', '16-JUN-84', '74 Manitowish Avenue',
'33430', '3925356210');
```

```
insert into PERSON (ssn, fname, minit, lname, sex, dob, street, zipcode, phone) values
('973893430', 'Roger', 'D', 'Butler', 'M', '22-JUN-93', '56858 Namekagon Terrace',
'33133', '9925777693');

insert into PERSON (ssn, fname, minit, lname, sex, dob, street, zipcode, phone) values
('495056486', 'Franklin', 'S', 'Hawkins', 'M', '10-JAN-89', '893 Cherokee Park', '33026',
'7161433608');

insert into PERSON (ssn, fname, minit, lname, sex, dob, street, zipcode, phone) values
('318716310', 'Kathy', 'H', 'Ford', 'F', '26-APR-80', '3696 Maryland Pass', '33076',
'4725192945');

insert into PERSON (ssn, fname, minit, lname, sex, dob, street, zipcode, phone) values
('188104101', 'Helen', 'R', 'Shaw', 'F', '26-MAY-92', '663 Golf View Pass', '33054',
'2878995418');

insert into PERSON (ssn, fname, minit, lname, sex, dob, street, zipcode, phone) values
('707023425', 'Jimmy', 'T', 'Henderson', 'M', '12-DEC-80', '5934 Schurz Place', '33063',
'5909463165');
```

```
insert into PERSON (ssn, fname, minit, lname, sex, dob, street, zipcode, phone) values
('812073369', 'Jean', 'S', 'Vasquez', 'F', '26-OCT-89', '3 Myrtle Plaza', '33324',
'9165744375');

insert into PERSON (ssn, fname, minit, lname, sex, dob, street, zipcode, phone) values
('877115419', 'Shawn', 'K', 'Garcia', 'M', '19-MAY-85', '788 Chinook Drive', '33430',
'4738694709');

insert into PERSON (ssn, fname, minit, lname, sex, dob, street, zipcode, phone) values
('157783174', 'Evelyn', 'A', 'Coleman', 'F', '17-SEP-93', '09 Springs Drive', '33426',
'1502245289');

insert into PERSON (ssn, fname, minit, lname, sex, dob, street, zipcode, phone) values
('841135835', 'Rebecca', 'C', 'Watson', 'F', '12-JAN-83', '4678 Myrtle Junction', '33441',
'4958048422');
```

Populate city_state table

```
insert into CITY_STATE (zipcode, city, state) values ('33076', 'Parkland', 'FL');
insert into CITY_STATE (zipcode, city, state) values ('33065', 'Coral Springs', 'FL');
insert into CITY_STATE (zipcode, city, state) values ('33063', 'Margate', 'FL');
insert into CITY_STATE (zipcode, city, state) values ('33026', 'Pembroke Pines', 'FL');

insert into CITY_STATE (zipcode, city, state) values ('33324', 'Plantation', 'FL');

insert into CITY_STATE (zipcode, city, state) values ('33146', 'Coral Gables', 'FL');

insert into CITY_STATE (zipcode, city, state) values ('33010', 'Hialeah', 'FL');

insert into CITY_STATE (zipcode, city, state) values ('33054', 'Opa-locka', 'FL');

insert into CITY_STATE (zipcode, city, state) values ('33133', 'Coconut Grove', 'FL');

insert into CITY_STATE (zipcode, city, state) values ('33056', 'Homestead', 'FL');

insert into CITY_STATE (zipcode, city, state) values ('33431', 'Boca Raton', 'FL');

insert into CITY_STATE (zipcode, city, state) values ('33426', 'Boynton Beach', 'FL');

insert into CITY_STATE (zipcode, city, state) values ('33430', 'Belle Glade', 'FL');

insert into CITY_STATE (zipcode, city, state) values ('33441', 'Deerfield Beach', 'FL');

insert into CITY_STATE (zipcode, city, state) values ('33073', 'Pompano Beach', 'FL');
```

Populate Employee table

```
insert into EMPLOYEE (ssn) values ('961705899');

insert into EMPLOYEE (ssn) values ('755215319');

insert into EMPLOYEE (ssn) values ('505329915');

insert into EMPLOYEE (ssn) values ('861648651');

insert into EMPLOYEE (ssn) values ('119292173');

insert into EMPLOYEE (ssn) values ('858270649');

insert into EMPLOYEE (ssn) values ('158506058');

insert into EMPLOYEE (ssn) values ('639785819');
insert into EMPLOYEE (ssn) values ('183338056');

insert into EMPLOYEE (ssn) values ('816694343');
```

Populate Staff table

```
insert into STAFF (ssn, title, dno) values ('961705899', 'Program coordinator', 3);

insert into STAFF (ssn, title, dno) values ('755215319', 'Dean', 3);

insert into STAFF (ssn, title, dno) values ('505329915', 'Dean', 1);

insert into STAFF (ssn, title, dno) values ('861648651', 'Dean', 2);

insert into STAFF (ssn, title, dno) values ('119292173', 'Program coordinator', 2);
```

Populate Faculty table

```
insert into FACULTY (ssn, degree, rank, employment_type) values ('858270649', 'PHD',
'Associate Professor', 'part-time');

insert into FACULTY (ssn, degree, rank, employment_type) values ('158506058', 'PHD',
'Associate Professor', 'part-time');

insert into FACULTY (ssn, degree, rank, employment_type) values ('639785819', 'PHD',
'Full Professor', 'full-time');

insert into FACULTY (ssn, degree, rank, employment_type) values ('183338056', 'PHD',
'Full Professor', 'full-time');
```

```
insert into FACULTY (ssn, degree, rank, employment_type) values ('816694343', 'MSC',
'Lecturer', 'full-time');
```

Populate Students table

```
insert into STUDENT (ssn, student_no, degree_prog, minor_dept, major_dept, class)
values ('477685256', 'N219730702', 'PHD', null, 2, 'graduate');

insert into STUDENT (ssn, student_no, degree_prog, minor_dept, major_dept, class)
values ('973893430', 'N392184993', 'PHD', 3, 2, 'graduate');

insert into STUDENT (ssn, student_no, degree_prog, minor_dept, major_dept, class)
values ('495056486', 'N911927207', 'BSC', 1, 2, 'freshman');

insert into STUDENT (ssn, student_no, degree_prog, minor_dept, major_dept, class)
values ('318716310', 'N291733899', 'MSC', 3, 1, 'graduate');
insert into STUDENT (ssn, student_no, degree_prog, minor_dept, major_dept, class)
values ('188104101', 'N519487360', 'MSC', null, 1, 'graduate');

insert into STUDENT (ssn, student_no, degree_prog, minor_dept, major_dept, class)
values ('707023425', 'N109124265', 'PHD', null, 1,'graduate');
```

```
insert into STUDENT (ssn, student_no, degree_prog, minor_dept, major_dept, class)
values ('812073369', 'N287857573', 'BSC', 3, 2, 'graduate');


insert into STUDENT (ssn, student_no, degree_prog, minor_dept, major_dept, class)
values ('877115419', 'N911443513', 'MSC', 2, 3, 'graduate');


insert into STUDENT (ssn, student_no, degree_prog, minor_dept, major_dept, class)
values ('157783174', 'N459664859', 'PHD', null, 3, 'graduate');


insert into STUDENT (ssn, student_no, degree_prog, minor_dept, major_dept, class)
values ('841135835', 'N884110281', 'BSC', 2, 1, 'sophomore');
```

Populate Dependents table

```
insert into DEPENDENT (name, essn, relationship, dob, sex) values ('Ryan Burton',
'858270649', 'Son', '08-DEC-83', 'M');
insert into DEPENDENT (name, essn, relationship, dob, sex) values ('Norma Reed',
'119292173', 'Daughter', '12-DEC-03', 'F');


insert into DEPENDENT (name, essn, relationship, dob, sex) values ('Sharon Powell',
'183338056', 'Spouse', '27-JUN-88', 'F');
```

insert into DEPENDENT (name, essn, relationship, dob, sex) values ('Richard Griffin', '158506058', 'Son', '28-JUL-03', 'M');

insert into DEPENDENT (name, essn, relationship, dob, sex) values ('Phillip Torres', '158506058', 'Spouse', '14-JUL-70', 'M');

insert into DEPENDENT (name, essn, relationship, dob, sex) values ('Amanda Cunningham', '755215319', 'Daughter', '20-DEC-94', 'F');

Populate Salary scale table

insert into SALARY_SCALE (rank, employment_type, salary) values ('Lecturer', 'part-time', 107774.84);

insert into SALARY_SCALE (rank, employment_type, salary) values ('Lecturer', 'full-time', 111458.48);

insert into SALARY_SCALE (rank, employment_type, salary) values ('Associate Professor', 'part-time', 140522.46);

insert into SALARY_SCALE (rank, employment_type, salary) values ('Associate Professor', 'full-time', 146775.44);

insert into SALARY_SCALE (rank, employment_type, salary) values ('Full Professor', 'part-time', 149189.60);

insert into SALARY_SCALE (rank, employment_type, salary) values ('Full Professor', 'full-time', 184191.01);

Populate Departments table

insert into DEPARTMENT (dno, dname, college, office_no, chair_ssn) values (1, 'Engineering and Technology', 'CEC', 'CD460', '505329915');

insert into DEPARTMENT (dno, dname, college, office_no, chair_ssn) values (2, 'Computer Science', 'CEC', 'CD452', '861648651');

insert into DEPARTMENT (dno, dname, college, office_no, chair_ssn) values (3, 'Information Systems and Cybersecurity', 'CEC', 'HD345', '755215319');

Populate Office table

insert into OFFICE (office_no, office_ph) values ('CD452', '5061091326');

insert into OFFICE (office_no, office_ph) values ('CD460', '9728049818');

insert into OFFICE (office_no, office_ph) values ('HD345', '2165356413');

insert into OFFICE (office_no, office_ph) values ('HD456', '2634291696');

insert into OFFICE (office_no, office_ph) values ('TA442', '6689476658');

```
insert into OFFICE (office_no, office_ph) values ('PB235', '7054881559');
```

Populate Courses table

```
insert into COURSE (cno, cname, credits, dno) values ('CSIS3400', 'Data Structures', 4, 2);

insert into COURSE (cno, cname, credits, dno) values ('CSIS3460', 'Object Oriented
Design', 3, 2);

insert into COURSE (cno, cname, credits, dno) values ('CSIS3500', 'Networks and Data
Communication', 3, 2);

insert into COURSE (cno, cname, credits, dno) values ('CISC610', 'Programming
Languages', 3, 2);

insert into COURSE (cno, cname, credits, dno) values ('CISC615', 'Design and Analysis of
Algorithms', 3, 2);
insert into COURSE (cno, cname, credits, dno) values ('CISC630', 'Compilers', 3, 2);

insert into COURSE (cno, cname, credits, dno) values ('CISC631', 'Theory of
Computation', 3, 2);
```

```
insert into COURSE (cno, cname, credits, dno) values ('CISC650', 'Computer Networks', 3,
2);

insert into COURSE (cno, cname, credits, dno) values ('CENG3720', 'Computer Systems
Engineering', 3, 1);

insert into COURSE (cno, cname, credits, dno) values ('CENG4710', 'Embedded Systems',
4, 1);

insert into COURSE (cno, cname, credits, dno) values ('EENG2710', 'Electrical
Circuits/Lab', 4, 1);

insert into COURSE (cno, cname, credits, dno) values ('EENG3310', 'Signals and Systems',
3, 1);

insert into COURSE (cno, cname, credits, dno) values ('MATH2100', 'Calculus I', 4, 1);

insert into COURSE (cno, cname, credits, dno) values ('CISC680', 'Software Engineering',
3, 1);

insert into COURSE (cno, cname, credits, dno) values ('CISC682', 'Software Requirements
Engineering', 3, 1);
```

insert into COURSE (cno, cname, credits, dno) values ('MMIS621', 'Information Systems Project Management', 3, 1);

insert into COURSE (cno, cname, credits, dno) values ('ISEC600', 'Secure Computer Systems', 3, 1);

insert into COURSE (cno, cname, credits, dno) values ('ISEC620', 'Applied Cryptography', 3, 1);

insert into COURSE (cno, cname, credits, dno) values ('ISEC660', 'Advanced Network Security', 3, 1);

insert into COURSE (cno, cname, credits, dno) values ('MSIT630', 'Database Systems', 3, 1);

Populate courses prerequisites table

insert into COURSE_PREREQUISITE (cno, prerequisite_cno) values ('CISC630', 'CISC610');

insert into COURSE_PREREQUISITE (cno, prerequisite_cno) values ('CISC631', 'CISC610');

```
insert into COURSE_PREREQUISITE (cno, prerequisite_cno) values ('CENG3720',
'EENG2710');

insert into COURSE_PREREQUISITE (cno, prerequisite_cno) values ('CENG3720',
'EENG3310');

insert into COURSE_PREREQUISITE (cno, prerequisite_cno) values ('CENG3720',
'MATH2100');

insert into COURSE_PREREQUISITE (cno, prerequisite_cno) values ('CENG4710',
'MATH2100');

insert into COURSE_PREREQUISITE (cno, prerequisite_cno) values ('EENG2710',
'MATH2100');

insert into COURSE_PREREQUISITE (cno, prerequisite_cno) values ('EENG3310',
'MATH2100');
insert into COURSE_PREREQUISITE (cno, prerequisite_cno) values ('CISC682', 'CISC680');

insert into COURSE_PREREQUISITE (cno, prerequisite_cno) values ('ISEC620', 'ISEC600');

insert into COURSE_PREREQUISITE (cno, prerequisite_cno) values ('ISEC660', 'CISC650');
```

Appoint faculty members to department

```
insert into APPOINTED_TO (fssn, dno) values ('858270649', 2);

insert into APPOINTED_TO (fssn, dno) values ('858270649', 3);

insert into APPOINTED_TO (fssn, dno) values ('158506058', 1);

insert into APPOINTED_TO (fssn, dno) values ('158506058', 3);

insert into APPOINTED_TO (fssn, dno) values ('639785819', 1);

insert into APPOINTED_TO (fssn, dno) values ('639785819', 2);

insert into APPOINTED_TO (fssn, dno) values ('639785819', 3);

insert into APPOINTED_TO (fssn, dno) values ('183338056', 1);

insert into APPOINTED_TO (fssn, dno) values ('183338056', 2);

insert into APPOINTED_TO (fssn, dno) values ('183338056', 3);

insert into APPOINTED_TO (fssn, dno) values ('816694343', 1);

insert into APPOINTED_TO (fssn, dno) values ('816694343', 2);
```

Add sections

```
insert into SECTION (sno, semester, year, meeting_day, meeting_time, meeting_room,

instructor, cno) values (1, 'summer', 2016, 'wednesday', '06:00PM', 'CD330',

'639785819', 'CSIS3500');
```

```
insert into SECTION (sno, semester, year, meeting_day, meeting_time, meeting_room,
instructor, cno) values (1, 'summer', 2016, 'wednesday', '04:00PM', 'CD330',
'158506058', 'CENG3720');

insert into SECTION (sno, semester, year, meeting_day, meeting_time, meeting_room,
instructor, cno) values (2, 'summer', 2016, 'wednesday', '04:00PM', 'CD378',
'183338056', 'CENG3720');

insert into SECTION (sno, semester, year, meeting_day, meeting_time, meeting_room,
instructor, cno) values (1, 'winter', 2016, 'wednesday', '04:00PM', 'PB345', '639785819',
'ISEC660');

insert into SECTION (sno, semester, year, meeting_day, meeting_time, meeting_room,
instructor, cno) values (1, 'summer', 2016, 'monday', '04:00PM', 'PB345', '858270649',
'CISC615');
insert into SECTION (sno, semester, year, meeting_day, meeting_time, meeting_room,
instructor, cno) values (1, 'summer', 2016, 'tuesday', '06:00PM', 'PB345', '858270649',
'ISEC620');

insert into SECTION (sno, semester, year, meeting_day, meeting_time, meeting_room,
instructor, cno) values (1, 'winter', 2016, 'friday', '06:00PM', 'CD378', '183338056',
'CISC631');
```

insert into SECTION (sno, semester, year, meeting_day, meeting_time, meeting_room, instructor, cno) values (1, 'summer', 2016, 'thursday', '04:00PM', 'PB345', '183338056', 'CENG4710');

Assign study plan

insert into STUDY_PLAN (student_ssn, cno) values ('157783174', 'CISC610');

insert into STUDY_PLAN (student_ssn, cno) values ('157783174', 'CISC631');

insert into STUDY_PLAN (student_ssn, cno) values ('157783174', 'CISC682');

insert into STUDY_PLAN (student_ssn, cno) values ('157783174', 'CSIS3460');

insert into STUDY_PLAN (student_ssn, cno) values ('157783174', 'EENG3310');

insert into STUDY_PLAN (student_ssn, cno) values ('157783174', 'ISEC660');

insert into STUDY_PLAN (student_ssn, cno) values ('157783174', 'MMIS621');

insert into STUDY_PLAN (student_ssn, cno) values ('157783174', 'MSIT630');

insert into STUDY_PLAN (student_ssn, cno) values ('188104101', 'CENG3720');

insert into STUDY_PLAN (student_ssn, cno) values ('188104101', 'CISC631');

insert into STUDY_PLAN (student_ssn, cno) values ('188104101', 'CISC650');

insert into STUDY_PLAN (student_ssn, cno) values ('188104101', 'CISC680');

insert into STUDY_PLAN (student_ssn, cno) values ('188104101', 'CISC615');

insert into STUDY_PLAN (student_ssn, cno) values ('188104101', 'CSIS3500');

insert into STUDY_PLAN (student_ssn, cno) values ('188104101', 'ISEC660');

insert into STUDY_PLAN (student_ssn, cno) values ('188104101', 'MATH2100');

```
insert into STUDY_PLAN (student_ssn, cno) values ('318716310', 'CISC680');

insert into STUDY_PLAN (student_ssn, cno) values ('318716310', 'CSIS3460');

insert into STUDY_PLAN (student_ssn, cno) values ('318716310', 'CISC630');

insert into STUDY_PLAN (student_ssn, cno) values ('318716310', 'CSIS3500');

insert into STUDY_PLAN (student_ssn, cno) values ('318716310', 'EENG2710');

insert into STUDY_PLAN (student_ssn, cno) values ('318716310', 'CENG4710');

insert into STUDY_PLAN (student_ssn, cno) values ('318716310', 'EENG3310');

insert into STUDY_PLAN (student_ssn, cno) values ('318716310', 'ISEC620');

insert into STUDY_PLAN (student_ssn, cno) values ('477685256', 'CENG3720');

insert into STUDY_PLAN (student_ssn, cno) values ('477685256', 'CISC610');

insert into STUDY_PLAN (student_ssn, cno) values ('477685256', 'ISEC600');

insert into STUDY_PLAN (student_ssn, cno) values ('477685256', 'CISC650');

insert into STUDY_PLAN (student_ssn, cno) values ('477685256', 'ISEC660');

insert into STUDY_PLAN (student_ssn, cno) values ('477685256', 'CISC682');

insert into STUDY_PLAN (student_ssn, cno) values ('477685256', 'EENG3310');

insert into STUDY_PLAN (student_ssn, cno) values ('477685256', 'MSIT630');

insert into STUDY_PLAN (student_ssn, cno) values ('495056486', 'CENG3720');

insert into STUDY_PLAN (student_ssn, cno) values ('495056486', 'CISC610');

insert into STUDY_PLAN (student_ssn, cno) values ('495056486', 'CISC615');

insert into STUDY_PLAN (student_ssn, cno) values ('495056486', 'CSIS3460');

insert into STUDY_PLAN (student_ssn, cno) values ('495056486', 'EENG2710');

insert into STUDY_PLAN (student_ssn, cno) values ('495056486', 'EENG3310');
insert into STUDY_PLAN (student_ssn, cno) values ('495056486', 'ISEC620');

insert into STUDY_PLAN (student_ssn, cno) values ('495056486', 'ISEC660');

insert into STUDY_PLAN (student_ssn, cno) values ('707023425', 'CENG3720');

insert into STUDY_PLAN (student_ssn, cno) values ('707023425', 'CISC610');

insert into STUDY_PLAN (student_ssn, cno) values ('707023425', 'CISC630');

insert into STUDY_PLAN (student_ssn, cno) values ('707023425', 'CSIS3400');

insert into STUDY_PLAN (student_ssn, cno) values ('707023425', 'CISC631');

insert into STUDY_PLAN (student_ssn, cno) values ('707023425', 'CSIS3460');

insert into STUDY_PLAN (student_ssn, cno) values ('707023425', 'CISC680');

insert into STUDY_PLAN (student_ssn, cno) values ('707023425', 'MATH2100');

insert into STUDY_PLAN (student_ssn, cno) values ('812073369', 'CISC610');

insert into STUDY_PLAN (student_ssn, cno) values ('812073369', 'CISC630');

insert into STUDY_PLAN (student_ssn, cno) values ('812073369', 'CISC650');

insert into STUDY_PLAN (student_ssn, cno) values ('812073369', 'EENG3310');

insert into STUDY_PLAN (student_ssn, cno) values ('812073369', 'ISEC600');

insert into STUDY_PLAN (student_ssn, cno) values ('812073369', 'ISEC620');

insert into STUDY_PLAN (student_ssn, cno) values ('812073369', 'CISC631');

insert into STUDY_PLAN (student_ssn, cno) values ('812073369', 'ISEC660');

insert into STUDY_PLAN (student_ssn, cno) values ('841135835', 'CENG3720');

insert into STUDY_PLAN (student_ssn, cno) values ('841135835', 'CENG4710');

insert into STUDY_PLAN (student_ssn, cno) values ('841135835', 'CISC615');

insert into STUDY_PLAN (student_ssn, cno) values ('841135835', 'CISC630');
insert into STUDY_PLAN (student_ssn, cno) values ('841135835', 'CISC631');

insert into STUDY_PLAN (student_ssn, cno) values ('841135835', 'CISC680');

insert into STUDY_PLAN (student_ssn, cno) values ('841135835', 'MATH2100');

insert into STUDY_PLAN (student_ssn, cno) values ('841135835', 'MMIS621');

insert into STUDY_PLAN (student_ssn, cno) values ('877115419', 'CENG3720');

insert into STUDY_PLAN (student_ssn, cno) values ('877115419', 'CISC610');

insert into STUDY_PLAN (student_ssn, cno) values ('877115419', 'CISC615');

insert into STUDY_PLAN (student_ssn, cno) values ('877115419', 'CISC682');

insert into STUDY_PLAN (student_ssn, cno) values ('877115419', 'EENG2710');

insert into STUDY_PLAN (student_ssn, cno) values ('877115419', 'ISEC600');

insert into STUDY_PLAN (student_ssn, cno) values ('877115419', 'ISEC660');

insert into STUDY_PLAN (student_ssn, cno) values ('877115419', 'ISEC620');

insert into STUDY_PLAN (student_ssn, cno) values ('973893430', 'CENG3720');

insert into STUDY_PLAN (student_ssn, cno) values ('973893430', 'CENG4710');

insert into STUDY_PLAN (student_ssn, cno) values ('973893430', 'EENG3310');

insert into STUDY_PLAN (student_ssn, cno) values ('973893430', 'CISC610');

insert into STUDY_PLAN (student_ssn, cno) values ('973893430', 'CISC630');

insert into STUDY_PLAN (student_ssn, cno) values ('973893430', 'CISC631');

insert into STUDY_PLAN (student_ssn, cno) values ('973893430', 'CISC680');

insert into STUDY_PLAN (student_ssn, cno) values ('973893430', 'CISC682');
```

Populate Transcript

```
insert into TRANSCRIPT (student_ssn, cno, grade) values ('157783174', 'EENG3310', 'A');

insert into TRANSCRIPT (student_ssn, cno, grade) values ('157783174', 'ISEC660', 'B');

insert into TRANSCRIPT (student_ssn, cno, grade) values ('157783174', 'MMIS621', 'C');

insert into TRANSCRIPT (student_ssn, cno, grade) values ('188104101', 'CISC631', 'A');

insert into TRANSCRIPT (student_ssn, cno, grade) values ('188104101', 'CISC650', 'B');

insert into TRANSCRIPT (student_ssn, cno, grade) values ('188104101', 'CSIS3500', 'C');

insert into TRANSCRIPT (student_ssn, cno, grade) values ('188104101', 'ISEC660', 'A');

insert into TRANSCRIPT (student_ssn, cno, grade) values ('318716310', 'CISC680', 'B');

insert into TRANSCRIPT (student_ssn, cno, grade) values ('477685256', 'CISC610', 'C');

insert into TRANSCRIPT (student_ssn, cno, grade) values ('477685256', 'ISEC600', 'A');

insert into TRANSCRIPT (student_ssn, cno, grade) values ('477685256', 'CISC650', 'B');

insert into TRANSCRIPT (student_ssn, cno, grade) values ('477685256', 'ISEC660', 'C');

insert into TRANSCRIPT (student_ssn, cno, grade) values ('477685256', 'CISC682', 'A');

insert into TRANSCRIPT (student_ssn, cno, grade) values ('707023425', 'CENG3720', 'B');

insert into TRANSCRIPT (student_ssn, cno, grade) values ('707023425', 'CISC610', 'C');

insert into TRANSCRIPT (student_ssn, cno, grade) values ('707023425', 'CISC630', 'A');

insert into TRANSCRIPT (student_ssn, cno, grade) values ('707023425', 'CSIS3400', 'B');

insert into TRANSCRIPT (student_ssn, cno, grade) values ('707023425', 'CISC631', 'C');

insert into TRANSCRIPT (student_ssn, cno, grade) values ('707023425', 'CSIS3460', 'A');

insert into TRANSCRIPT (student_ssn, cno, grade) values ('812073369', 'CISC610', 'B');

insert into TRANSCRIPT (student_ssn, cno, grade) values ('812073369', 'CISC630', 'C');
insert into TRANSCRIPT (student_ssn, cno, grade) values ('812073369', 'CISC650', 'A');

insert into TRANSCRIPT (student_ssn, cno, grade) values ('841135835', 'CISC615', 'B');

insert into TRANSCRIPT (student_ssn, cno, grade) values ('841135835', 'CISC630', 'C');

insert into TRANSCRIPT (student_ssn, cno, grade) values ('841135835', 'CISC631', 'A');

insert into TRANSCRIPT (student_ssn, cno, grade) values ('841135835', 'CISC680', 'B');

insert into TRANSCRIPT (student_ssn, cno, grade) values ('841135835', 'MATH2100', 'C');

insert into TRANSCRIPT (student_ssn, cno, grade) values ('841135835', 'MMIS621', 'A');

insert into TRANSCRIPT (student_ssn, cno, grade) values ('877115419', 'CENG3720', 'B');

insert into TRANSCRIPT (student_ssn, cno, grade) values ('877115419', 'CISC610', 'C');

insert into TRANSCRIPT (student_ssn, cno, grade) values ('877115419', 'CISC615', 'A');

insert into TRANSCRIPT (student_ssn, cno, grade) values ('877115419', 'CISC682', 'B');

insert into TRANSCRIPT (student_ssn, cno, grade) values ('877115419', 'EENG2710', 'C');

insert into TRANSCRIPT (student_ssn, cno, grade) values ('877115419', 'ISEC600', 'A');

insert into TRANSCRIPT (student_ssn, cno, grade) values ('877115419', 'ISEC620', 'B');

insert into TRANSCRIPT (student_ssn, cno, grade) values ('973893430', 'CENG3720', 'C');

insert into TRANSCRIPT (student_ssn, cno, grade) values ('973893430', 'CENG4710', 'A');

insert into TRANSCRIPT (student_ssn, cno, grade) values ('973893430', 'EENG3310', 'B');

insert into TRANSCRIPT (student_ssn, cno, grade) values ('973893430', 'CISC610', 'C');

insert into TRANSCRIPT (student_ssn, cno, grade) values ('973893430', 'CISC680', 'A');

insert into TRANSCRIPT (student_ssn, cno, grade) values ('973893430', 'CISC682', 'B');

insert into TRANSCRIPT (student_ssn, cno, grade) values ('495056486', 'CSIS3500', 'B+');

insert into TRANSCRIPT (student_ssn, cno, grade) values ('495056486', 'ISEC660', 'A-');
```

Assign advisors

```
insert into GRAD_STUDENT (student_ssn, advisor_ssn) values ('157783174',
'158506058');

insert into GRAD_STUDENT (student_ssn, advisor_ssn) values ('188104101',
'158506058');

insert into GRAD_STUDENT (student_ssn, advisor_ssn) values ('318716310',
'816694343');

insert into GRAD_STUDENT (student_ssn, advisor_ssn) values ('477685256',
'816694343');

insert into GRAD_STUDENT (student_ssn, advisor_ssn) values ('707023425',
'858270649');
insert into GRAD_STUDENT (student_ssn, advisor_ssn) values ('877115419',
'816694343');

insert into GRAD_STUDENT (student_ssn, advisor_ssn) values ('973893430',
'183338056');
```

Populate grade report

```
insert into GRADE_REPORT (student_ssn, sno, semester, year, cno, grade) values
('477685256', 1, 'summer', 2016, 'CSIS3500', null);


insert into GRADE_REPORT (student_ssn, sno, semester, year, cno, grade) values
('707023425', 1, 'summer', 2016, 'CSIS3500', null);


insert into GRADE_REPORT (student_ssn, sno, semester, year, cno, grade) values
('157783174', 2, 'summer', 2016, 'CENG3720', null);


insert into GRADE_REPORT (student_ssn, sno, semester, year, cno, grade) values
('318716310', 1, 'summer', 2016, 'CISC615', null);


insert into GRADE_REPORT (student_ssn, sno, semester, year, cno, grade) values
('188104101', 1, 'summer', 2016, 'ISEC620', null);
insert into GRADE_REPORT (student_ssn, sno, semester, year, cno, grade) values
('157783174', 1, 'summer', 2016, 'CENG4710', null);


insert into GRADE_REPORT (student_ssn, sno, semester, year, cno, grade) values
('973893430', 1, 'winter', 2016, 'ISEC660', 'A');
```

```
insert into GRADE_REPORT (student_ssn, sno, semester, year, cno, grade) values
('973893430', 1, 'winter', 2016, 'CISC631', 'B');

insert into GRADE_REPORT (student_ssn, sno, semester, year, cno, grade) values
('477685256', 1, 'summer', 2016, 'CENG3720', null);

insert into GRADE_REPORT (student_ssn, sno, semester, year, cno, grade) values
('707023425', 1, 'summer', 2016, 'CENG3720', null);

insert into GRADE_REPORT (student_ssn, sno, semester, year, cno, grade) values
('841135835', 2, 'summer', 2016, 'CENG3720', null);

insert into GRADE_REPORT (student_ssn, sno, semester, year, cno, grade) values
('812073369', 1, 'summer', 2016, 'CISC615', null);

insert into GRADE_REPORT (student_ssn, sno, semester, year, cno, grade) values
('318716310', 1, 'summer', 2016, 'ISEC620', null);

insert into GRADE_REPORT (student_ssn, sno, semester, year, cno, grade) values
('707023425', 1, 'summer', 2016, 'CENG4710', null);
```

```
insert into GRADE_REPORT (student_ssn, sno, semester, year, cno, grade) values
('841135835', 1, 'winter', 2016, 'ISEC660', 'B');

insert into GRADE_REPORT (student_ssn, sno, semester, year, cno, grade) values
('318716310', 1, 'winter', 2016, 'CISC631', 'C');

insert into GRADE_REPORT (student_ssn, sno, semester, year, cno, grade) values
('495056486', 1, 'summer', 2016, 'CSIS3500', null);

insert into GRADE_REPORT (student_ssn, sno, semester, year, cno, grade) values
('841135835', 1, 'summer', 2016, 'CENG3720', null);

insert into GRADE_REPORT (student_ssn, sno, semester, year, cno, grade) values
('812073369', 2, 'summer', 2016, 'CENG3720', null);

insert into GRADE_REPORT (student_ssn, sno, semester, year, cno, grade) values
('841135835', 1, 'summer', 2016, 'CISC615', null);
```

7. Query implementation

1) For each department, list the numbers of major students and minor students.

select dname, count(*)

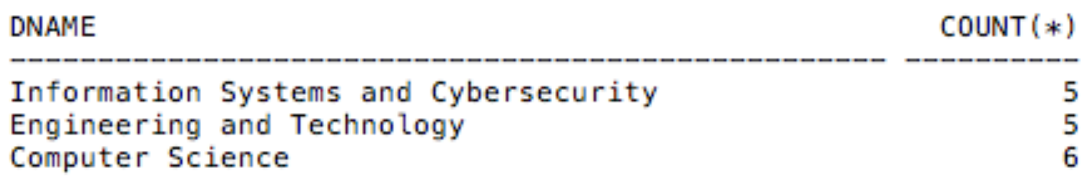
from department

left join student

on dno = major_dept or dno = minor_dept

group by dname;

Result:



```
DNAME                                                COUNT(*)
---------------------------------------------------- ----------
Information Systems and Cybersecurity                         5
Engineering and Technology                                    5
Computer Science                                              6
```

2) For each department, list all the instructors along with the number of courses he/she teaches.

select d.dname, a.fssn as ssn, count(s.sno) as num_of_courses

from department d

left join appointed_to a

on d.dno = a.dno

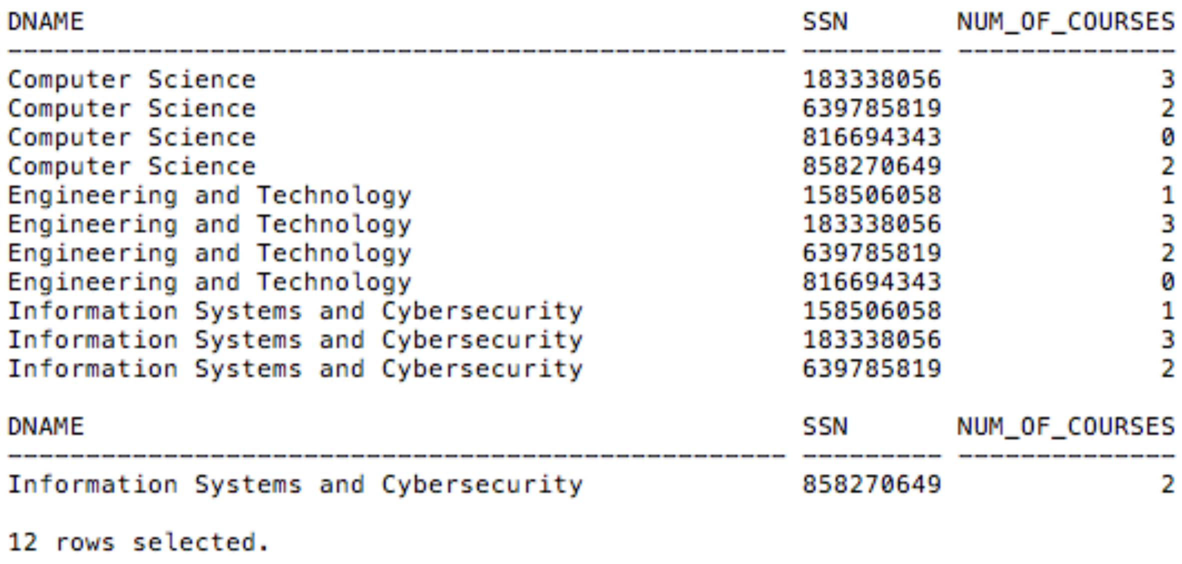
left join section s

on a.fssn = s.instructor

group by d.dname, a.fssn

order by d.dname asc;

Result:



```
DNAME                                              SSN       NUM_OF_COURSES
-------------------------------------------------- --------- --------------
Computer Science                                   183338056              3
Computer Science                                   639785819              2
Computer Science                                   816694343              0
Computer Science                                   858270649              2
Engineering and Technology                         158506058              1
Engineering and Technology                         183338056              3
Engineering and Technology                         639785819              2
Engineering and Technology                         816694343              0
Information Systems and Cybersecurity              158506058              1
Information Systems and Cybersecurity              183338056              3
Information Systems and Cybersecurity              639785819              2

DNAME                                              SSN       NUM_OF_COURSES
-------------------------------------------------- --------- --------------
Information Systems and Cybersecurity              858270649              2

12 rows selected.
```

3) For each department, list all the courses which it offers.

select d.dname, c.cno

from department d

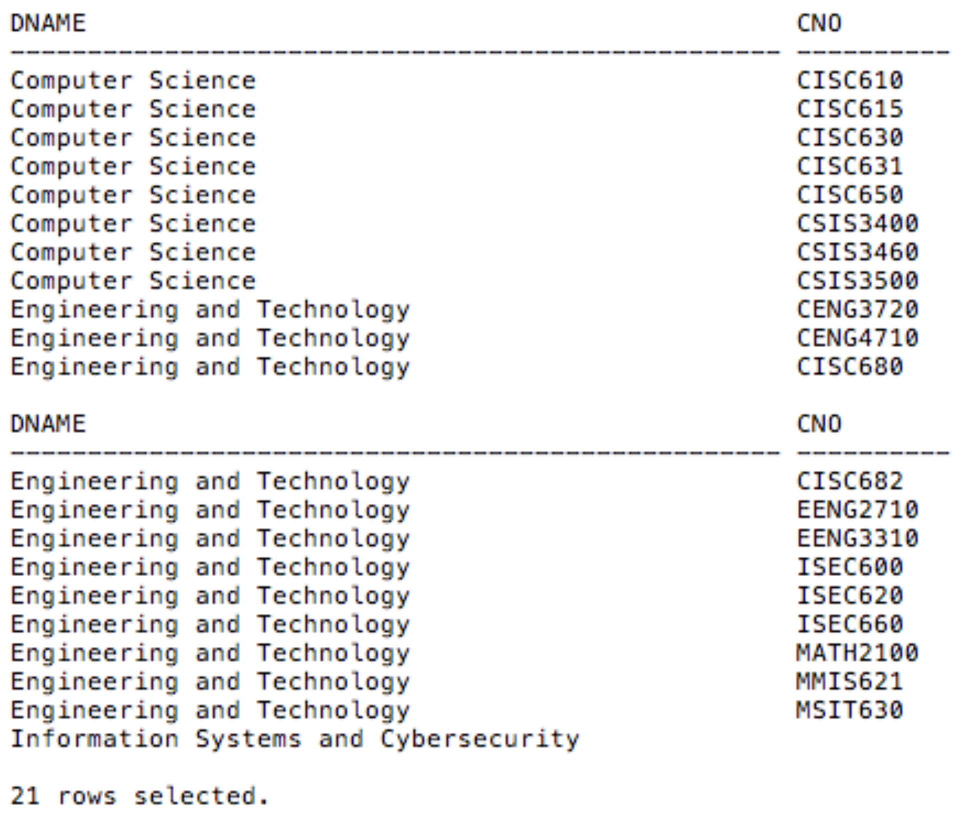
left join course c

on d.dno = c.dno

group by d.dname, c.cno

order by d.dname asc;

Result:

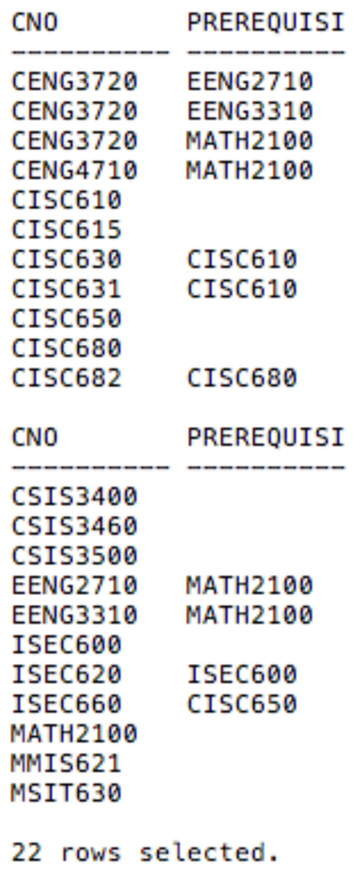


```
DNAME                                              CNO
-------------------------------------------------- ----------
Computer Science                                   CISC610
Computer Science                                   CISC615
Computer Science                                   CISC630
Computer Science                                   CISC631
Computer Science                                   CISC650
Computer Science                                   CSIS3400
Computer Science                                   CSIS3460
Computer Science                                   CSIS3500
Engineering and Technology                         CENG3720
Engineering and Technology                         CENG4710
Engineering and Technology                         CISC680

DNAME                                              CNO
-------------------------------------------------- ----------
Engineering and Technology                         CISC682
Engineering and Technology                         EENG2710
Engineering and Technology                         EENG3310
Engineering and Technology                         ISEC600
Engineering and Technology                         ISEC620
Engineering and Technology                         ISEC660
Engineering and Technology                         MATH2100
Engineering and Technology                         MMIS621
Engineering and Technology                         MSIT630
Information Systems and Cybersecurity

21 rows selected.
```

4) For each course, list all the prerequisites of that course.

```
select c.cno, p.prerequisite_cno

from course c

left join course_prerequisite p

on c.cno = p.cno

group by c.cno, p.prerequisite_cno

order by c.cno asc;
```

Result:

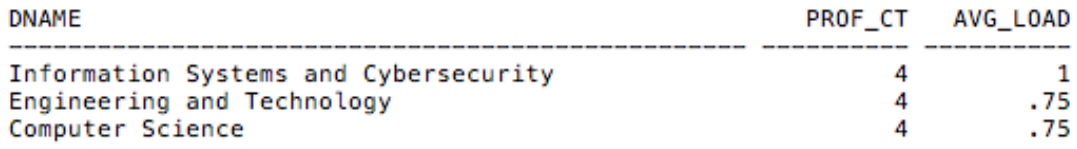

```
CNO        PREREQUISI
---------- ----------
CENG3720   EENG2710
CENG3720   EENG3310
CENG3720   MATH2100
CENG4710   MATH2100
CISC610
CISC615
CISC630    CISC610
CISC631    CISC610
CISC650
CISC680
CISC682    CISC680

CNO        PREREQUISI
---------- ----------
CSIS3400
CSIS3460
CSIS3500
EENG2710   MATH2100
EENG3310   MATH2100
ISEC600
ISEC620    ISEC600
ISEC660    CISC650
MATH2100
MMIS621
MSIT630

22 rows selected.
```

5) For each department, list the total number of professors and average teaching load.

```
select d.dname, count(distinct t1.fssn) as prof_ct, count(load)/count(distinct
t1.fssn) as avg_load
from department d
   left join
   (select dno, fssn from appointed_to) t1
```

```
    on t1.dno = d.dno

    left join

    (select count(*) as load, instructor from section group by instructor) t2

    on t1.fssn = t2.instructor

group by d.dname;
```

Result:

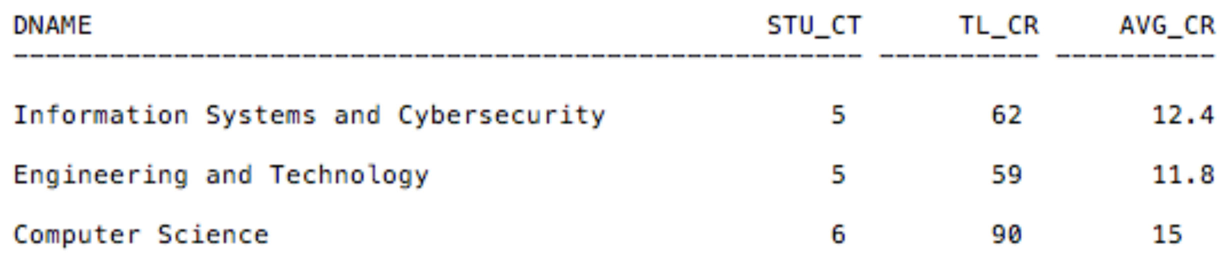


```
DNAME                                             PROF_CT   AVG_LOAD
------------------------------------------------- ---------- ----------
Information Systems and Cybersecurity                      4          1
Engineering and Technology                                 4        .75
Computer Science                                           4        .75
```

6) For each department, list the total numbers of students, total number of credit hours taken by these students, and the average credit hour per student.

```
select d.dname, count(distinct t1.ssn) as stu_ct, sum(t2.credits) as tl_cr,

sum(t2.credits) / count(distinct t1.ssn) as avg_cr

from department d,

     (select ssn, major_dept, minor_dept from student) t1,

     (select c.credits, t.student_ssn as ssn from transcript t, course c where t.cno =

c.cno) t2

where (t1.major_dept = d.dno or t1.minor_dept = d.dno) and (t2.ssn = t1.ssn)

group by d.dname;
```

Result:



```
DNAME                                                  STU_CT      TL_CR     AVG_CR
-------------------------------------------------- ---------- ---------- ----------
Information Systems and Cybersecurity                       5         62       12.4
Engineering and Technology                                  5         59       11.8
Computer Science                                            6         90         15
```

7) For each instructor, list the number of all students who register in the sections that the instructor teaches.

select f.ssn, stu.student_no

from faculty f

left join section s

on f.ssn = s.instructor

left join grade_report g

on s.sno = g.sno and s.semester = g.semester and s.year = g.year and s.cno = g.cno

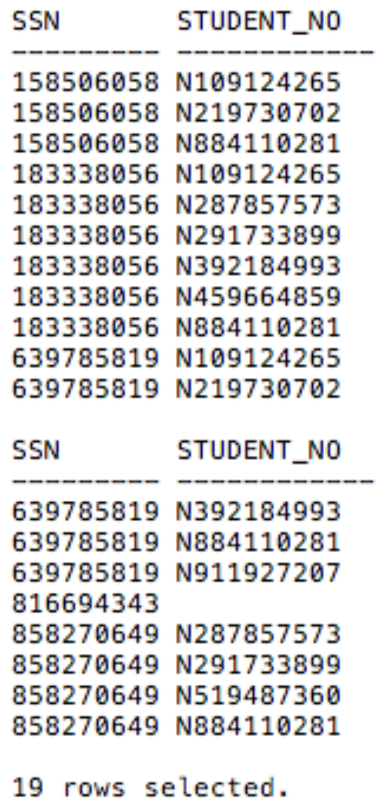
left join student stu

on stu.ssn = student_ssn

group by f.ssn, stu.student_no

order by f.ssn asc;

Results:



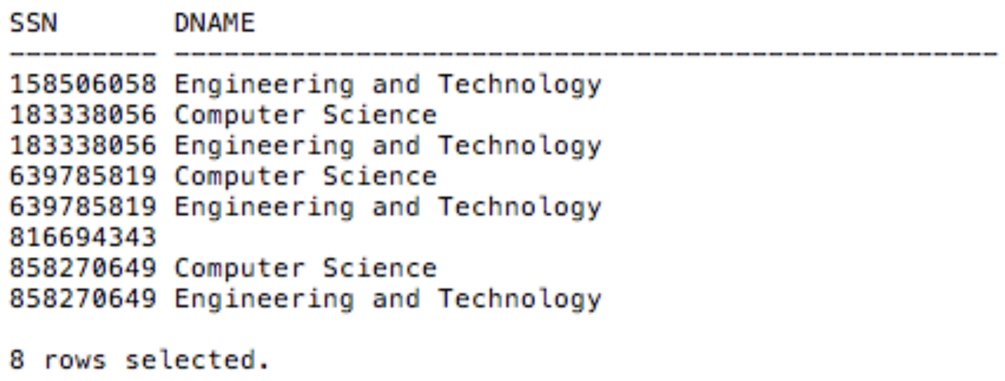
```
SSN       STUDENT_NO
--------- ------------
158506058 N109124265
158506058 N219730702
158506058 N884110281
183338056 N109124265
183338056 N287857573
183338056 N291733899
183338056 N392184993
183338056 N459664859
183338056 N884110281
639785819 N109124265
639785819 N219730702

SSN       STUDENT_NO
--------- ------------
639785819 N392184993
639785819 N884110281
639785819 N911927207
816694343
858270649 N287857573
858270649 N291733899
858270649 N519487360
858270649 N884110281

19 rows selected.
```

8) For each instructor, list all the departments which offer the courses that the instructor teaches.

```
select f.ssn, d.dname
from faculty f
left join department d
on exists ((select cno from course where dno = d.dno) intersect (select cno from section where instructor = f.ssn))
group by f.ssn, d.dname
```

order by f.ssn asc;

Results:



```
SSN       DNAME
--------- --------------------------------------------------
158506058 Engineering and Technology
183338056 Computer Science
183338056 Engineering and Technology
639785819 Computer Science
639785819 Engineering and Technology
816694343
858270649 Computer Science
858270649 Engineering and Technology

8 rows selected.
```

9) For each department, list all the professors who teach more than two courses, and make the salaries less than the average salary of the professors in their department.

select d.dname, f.ssn

from department d

left join appointed_to a

on d.dno = a.dno

left join faculty f

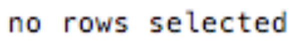
on a.fssn = f.ssn

left join salary_scale ss

on f.rank = ss.rank and f.employment_type = ss.employment_type

```
where (select count(*) from section where instructor = f.ssn) > 2

and ss.salary < (select avg(sal.salary)

        from faculty fac, salary_scale sal, appointed_to at

        where fac.rank = sal.rank and fac.employment_type =

sal.employment_type and at.fssn = fac.ssn and at.dno = d.dno)

group by d.dname, f.ssn;
```

Results:

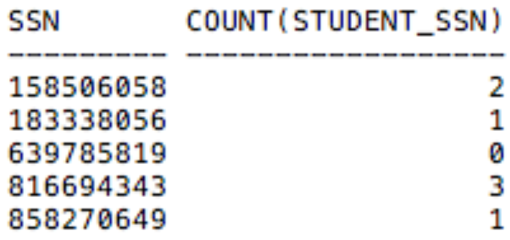


```
no rows selected
```

10) Show how many students that each professor advises.

```
select ssn, count(student_ssn)

from faculty

left join grad_student

on ssn = advisor_ssn

group by ssn;
```

Results:

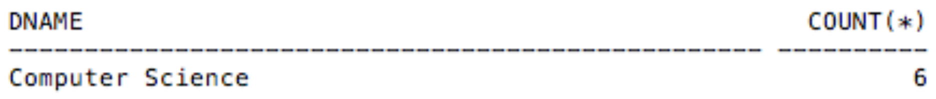


```
SSN       COUNT(STUDENT_SSN)
--------- ------------------
158506058                  2
183338056                  1
639785819                  0
816694343                  3
858270649                  1
```

11) Find the departments which have more students than the average students per department.

```
select dname, count(*)
from department
inner join student
on dno = major_dept or dno = minor_dept
group by dname
having count(*) > (select avg(count)
                from (select count(*) as count
                    from department
                    inner join student
                    on dno = major_dept or dno = minor_dept
                    group by dno));
```

Results:

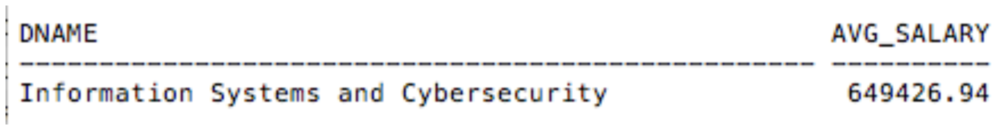


```
DNAME                                                COUNT(*)
--------------------------------------------------- ----------
Computer Science                                              6
```

12) Find the departments whose total salary is greater than the average salary per department.

```
select d.dname, sum(s.salary) as avg_salary
from department d
inner join appointed_to a
on d.dno = a.dno
inner join faculty f
on a.fssn = f.ssn
inner join salary_scale s
on f.rank = s.rank and f.employment_type = s.employment_type
group by d.dname
having sum(s.salary) > (select avg(avg_salary)
            from (select d.dname, sum(s.salary) as avg_salary
               from department d
               inner join appointed_to a
               on d.dno = a.dno
               inner join faculty f
               on a.fssn = f.ssn
               inner join salary_scale s
               on f.rank = s.rank and f.employment_type = s.employment_type
               group by d.dname));
```

Results:

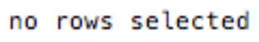

```
DNAME                                              AVG_SALARY
-------------------------------------------------- ----------
Information Systems and Cybersecurity               649426.94
```

13) For each department, list the professors who have the number of Ph.D. students he/she advises more than the average number of Ph.D. students these professors advise in their department.

```
select d.dname, a.fssn, count(g.advisor_ssn) as count
from department d
left join appointed_to a
on d.dno = a.dno
inner join (grad_student g
inner join student s
on g.student_ssn = s.ssn and s.degree_prog = 'PHD')
on a.fssn = g.advisor_ssn
group by d.dname, a.fssn, d.dno
having count(g.advisor_ssn) > (select avg(phd_count)
                    from (select count(*) as phd_count
                        from appointed_to at, grad_student gs, student stu
                        where at.dno = d.dno
                          and at.fssn = gs.advisor_ssn
                          and gs.student_ssn = stu.ssn
```

```
                              and stu.degree_prog = 'PHD'))
order by d.dname asc;
```

Results:

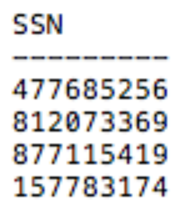


```
no rows selected
```

14) List the students who have completed all the prerequisite courses for their major.

```
select s.ssn
from student s
where not exists
   ((select distinct c.cno
    from course c, course_prerequisite p
    where c.dno = s.major_dept and c.cno = p.prerequisite_cno)
    minus
    (select cno from transcript where student_ssn = s.ssn));
```

Results:

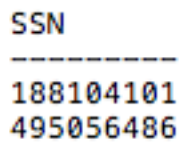


```
SSN
---------
477685256
812073369
877115419
157783174
```

15) List the students who have taken all the courses offered by Professor Smith.

```
select stu.ssn
from student stu
where not exists
   ((select s.cno
     from person p, section s
     where p.ssn = s.instructor and p.lname = 'Smith')
     minus
     (select cno from transcript where student_ssn = stu.ssn));
```

Results:

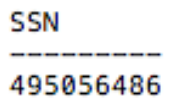


```
SSN
---------
188104101
495056486
```

16) List the students who have only taken the courses taught by Professor Smith.

```
select distinct stu.ssn
from student stu, transcript t
where stu.ssn = t.student_ssn and not exists
                    ((select cno from transcript where student_ssn = stu.ssn)
                     minus
                     (select s.cno
                     from person p, section s
                     where p.ssn = s.instructor and p.lname = 'Smith'));
```

Results:

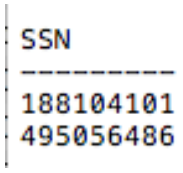


```
SSN
---------
495056486
```

17) List the students who have taken all the courses that the student Franklin has taken.

```
select stu.ssn
from student stu
where not exists ((select t.cno
          from person p, transcript t
          where p.ssn = t.student_ssn and p.fname = 'Franklin')
          minus
          (select cno from transcript where student_ssn = stu.ssn));
```

Results:

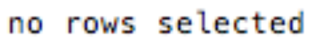


```
SSN
---------
188104101
495056486
```

18) List the students who passed all the exams required by their respective study plan.

```
select stu.ssn
from student stu
where not exists ((select cno
          from study_plan
          where student_ssn = stu.ssn)
          minus
          (select cno from transcript where student_ssn = stu.ssn and grade in
('A', 'A-', 'B+', 'B', 'B-', 'C+')));
```

Results:

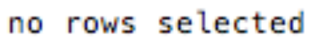


```
no rows selected
```

19) List the students who had taken the courses required by their study plan.

```
select stu.ssn

from student stu

where not exists ((select cno

            from study_plan

            where student_ssn = stu.ssn)

            minus

            (select cno from transcript where student_ssn = stu.ssn));
```

Results:



```
no rows selected
```