1. **Normalize all of your tables to third normal form . Make any necessary changes to the EER . Explain why these changes needed to be made.**

OFFICE(room-number, building-abbreviation)

In OFFICE table , primary key contains all attributes. Any functional dependency in this table will fallows the condition in 3NF .Thus OFFICE is in 3NF.

PEOPLE(net-id ,phone-number, DOB, email, last-name, middle-name, first-name, zip-code, state, city, street)

In PEOPLE table , net-id 🡪all the other attributes, and net-id is primary key . It follows the definition of 3NF.Then PEOPLE is in 3NF.

PROFESSOR(net-id , rank, office-roomnumber, office-building-abbreviation , office-hour)

In PROFESSSOR table, net-id🡪all the other attributes, and net-id is primary key. It follows the definition of 3NF. Then PROFESSOR is in 3NF .

RA(net-id)

In RA table, primary key contains all attributes .Any functional dependency in this table will follows the conditions in 3NF,(The attributes dependent on will be part of primary key) . Thus RA is in 3NF.

RA\_WORK\_ASSIGNMENT(workload , prof-net-id , ra-net-id , room-number , building-abbreviation)

In RA\_WORK\_ASSIGNMENT , no-prime attribute workload dependent on primary key. It follows conditions in 3NF. There are no other functional dependencies . Thus RA\_WORK\_ASSIGNMENT is in 3NF.

ROOM(room-number, building-abbreviation)

In ROOM table , primary key contains all attributes. Any functional dependency in this table will fallows the condition in 3NF .Thus ROOM is in 3NF.

SECTION(course number , section-number , year , semester , class-time , capacity, instructor-net-id, building-abbreviation , room-number)

1In SECTION table, no-prime attributes dependent on primary key. It follows conditions in 3NF. There are no other functional dependencies . Thus SECTION is in 3NF.

STUDENT(net-id , track-name)

In STUDENT table, no-prime attribute track-name dependent on primary key. It follows conditions in 3NF. There are no other functional dependencies . Thus STUDENT is in 3NF.

TA(net-id , office-roomnumber , office-building-abbreviation, office-hour)

In TA table, no-prime attributes dependent on primary key. It follows conditions in 3NF. There are no other functional dependencies . Thus TA is in 3NF.

# Write SQL statements to create database, tables and all other structures. Primary keys and foreign keys must be defined appropriately. The quantity CONSTRAINTts of the relation between the entities, which should be described in EER diagram, are not required.

OFFICE(room-number, building-abbreviation)

CREAT TABLE OFFICE

(

room-number INTEGER NOT NULL,

building-abbreviation VARCHAR(255) NOT NULL,

CONSTRAINT pk\_office PRIMARY KEY (room-number,building-abbreviation)

CONSTRAINT chk\_office\_room\_number CHECK(room-number>=1000 AND room-number<=9999)

);

PEOPLE(net-id ,phone-number, DOB, email, last-name, middle-name, first-name, zip-code, state, city, street)

CREAT TABLE PEOPLE

(

net-id VARCHAR(255) NOT NULL,

phone-number INTEGER NOT NULL,

DOB DATE NOT NULL,

email VARCHAR(255) ,

last-name VARCHAR(255) NOT NULL,

middle-name TEXT,

first-name VARCHAR(255) NOT NULL,

zip-code INTEGER NOT NULL,

state VARCHAR(255) NOT NULL,

city VARCHAR(255) NOT NULL,

sreet VARCHAR(255) NOT NULL,

CONSTRAINTT pk\_people PRIMARY KEY (net-id),

CONSTRAINTT chk\_phone\_number CHECK (phone\_number>=10000000000 AND phone\_number<=9999999999),

CONSTRAINT chk\_people\_phone\_number CHECK (zip-code>=10000 AND zip-code<=99999),

);

PROFESSOR(net-id , rank, office-roomnumber, office-building-abbreviation , office-hour)

CREAT TABLE PROCEFESSOR

(

net-id VARCHAR(255) NOT NULL,

rank VRACHAR(10) NOT NULL,

office-roomnumber INTEGER NOT NULL,

office-building-abbreviation VARCHAR(255) NOT NULL,

office-hour DECIMAL NOT NULL,

CONSTRAINT pk\_professor PRIMARY KEY (net-id),

CONSTRAINT chk\_pro\_office\_roomnumber CHECK (office\_roomnumber>=1000 AND office\_roomnumber<=9999)

COSTRAIN chk\_prof\_rank CHECK (rank in (“assistant”,”associate”,”full”))

);

RA(net-id)

CREAT TABLE RA

(

net id VARCHAR(255) NOT NULL,

CONSTRAINT pk\_ra PRIMARY KEY (net-id),

);

RA\_WORK\_ASSIGNMENT(workload , prof-net-id , ra-net-id , room-number , building-abbreviation)

CREAT TABLE RA\_WORK\_ASSIGNMENT

(

workload DECIMAL NOT NULL,

pro-net-id VARCHAR(255) NOT NULL,

ra-net-id VARCHAR(255) NOT NULL,

room-number INTEGER NOT NULL,

building-abbreviaton VARCHAR(255) NOT NULL,

CONSTRAINT pk\_raw PRIMARY KEY (prof-net-id, room-number, building-abbreviation),

CONSTRAINT chk\_raw\_room\_number CHECK (room-number>=1000 AND room-number<=9999),

);

ROOM(room-number, building-abbreviation)

CREAT TABLE ROOM

(

room-number INTEGER NOT NULL,

building-abbreviation VARCHAR(255),

CONSTRAINT pk\_room PRIMARY KEY(room\_number, building\_abbreviation)

CONSTRAINT chk\_room\_toom\_bumber CHECK (room\_number>=1000 AND room\_number<=9999)

);

SECTION(course number , section-number , year , semester , class-time , capacity, instructor-net-id, building-abbreviation , room-number)

CREAT TABLE SECTION

(

course\_number INTEGER NOT NULL,

section\_number INTEGER NOT NULL,

year INTEGER NOT NULL,

semester VAECHAR(255) NOT NULL,

class\_time DECIMAL NOT NULL,

capacity INTEGER NOT NULL,

instructor\_net\_id VARCHAR(255),

building\_abbreviation VARCHAR(255),

room\_number INTEGER NOT NULL,

CONSTRAINT pk\_section PRIMARY KEY (course\_number, section\_number ,year, semester),

CONSTRAINT chk\_sec\_course\_number CHECK(course\_number>=1000 AND course\_number<=9999 )

CONSTRAINT chk\_sec\_year CHECK (year>=1000 AND year<=9999),

CONSTRAINT chk\_sec\_room\_number(room\_number>=1000 AND room\_number<=9999),

);

STUDENT(net-id , track-name)

CREAT TABLE STUDENT

(

net\_id VARCHAR(255) NOT NULL,

track\_name VARCHAR(255) NOT NULL,

CONSTRAINT pk\_student PRIMARY KEY (net\_id),

);

TA(net-id , office-roomnumber , office-building-abbreviation, office-hour)

CREAT TABLE TA

(

net\_id VARCHAR(255) NOT NULL,

office\_number INTEGER NOT NULL,

office\_building\_abbreviation VARCHAR(255) NOT NULL,

office\_hour DECIMAL NOT NULL,

CONSTRAINT pk\_ta PRIMARY KEY (net\_id),

CONSTRAINT chk\_ta\_office\_number CHECK (office\_number>=1000 AND office\_number<=9999),

);

# Answer the following Queries. Feel free to use any of the views that you created in part d.

# 3. Current courses: List name and department of courses that have section in current semester.

CREATE VIEW [Current course] AS

SELECT c.name,d.full\_name

FROM COURSE c, DEPARTMENT d, SECTION s

WHERE (c.course\_number=s.course\_number, d. abbreviation=c. dept\_abbreviation)

AND (ear=2014)

AND (semester=’fall’);

# Answer the following Queries. Feel free to use any of the views that you created in part d.

1. **Retrieve name and phone number of students living in Richardson.**

SELECT p.phone\_number,p.last\_name,p.middle\_name,p.first\_name

FROM PEOPLE P, STUDENT s

WHERE (p.net\_id=s.net\_id)

AND(city=’richardson’);

1. **Retrieve the SSN and name of lecturers and TA's working for CS department.**

SELECT e.ssn,p.last\_name,p.middle\_name,p.first\_name

FROM LECTURER l, EMPLOYEE e, TA t,PEOPLE p

WHERE(l.net\_id=e.net\_id,t.net\_id=e.net\_id,l.net\_id=p.net\_id,t.net\_id=p.net\_id)

AND(t.office\_building\_abbreviation=cs)

AND(l.office\_building\_abbreviation=cs);

**3. Retrieve the name and web site address of departments which have the most number of buildings.**

SELECT d.website\_address,d.full\_name

FROM BUILDING b, DEPARTMENT d,

WHERE (b.dept\_abbreviation=d.abbreviation)

HAVING COUNT(\*)=(

SELECT MAX(num)

FROM(

SELECT Count(\*) AS num

FROM b.dept\_abbreviation

GROUP BY d.abbreviation

)

)

**4. Retrieve the name and total capacity of all courses.**

SELECT c.name,s.capacity

FROM COURSE c, SECTION s

WHERE(c.course\_number=s.course\_number)

HAVING COUNT(\*)=(

SELECT SUM(num)

FROM(

SELECT Count(\*) AS num

FROM S.capacity

GROUP BY s.cours­\_number

)

)

**5. For students who work as both TA and RA, retrieve their name, address, and course sections they work for.**

SELECT p.last\_name,p.middle\_name,p.first\_name,p.state,p.city,p.street,p.zip\_code,c.name

FROM TA t, RA r, PEOPLE p, SECTIION\_HAS\_TA s, COURSE c

WHERE(t.net\_id=r.net\_id)

AND (t.net\_id=p.net\_id)

AND(t.net\_id=s.ta\_ney\_id)

AND(s.course\_number=c.course\_number)