A.

BUILDING (abbreviation, full\_name, dept\_abbreviation)

In BUILDING table, abbreviation —> full\_name and dept-abbreviation, and abbreviation is primary key. It follows the definition of 3NF. Then BUILDING is in 3NF.

CLASSROOM (building\_abbreviation, room\_number, capacity, computer\_password)

In CLASSROOM table, {building\_abbreviation, room\_number} —> capacity and computer\_password, while {building\_abbreviation, room\_number} is the primary key. It follows the definition of 3NF. Then CLASSROOM is in 3NF.

COURSE\_TEXTBOOK (course\_number, textbook)

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

COURSE (course\_number, name, credit\_hour, dept\_abbreviation)

In COURSE table, course\_number —> name, credit\_hour and dept\_abbreviation. Considering course\_number is the primary key, it follows the definition of 3NF. Then COURSE is in 3NF.

DEPARTMENT (abbreviation, website\_address, full\_name, head\_prof\_net\_id)

In DEPARTMENT table, abbreviation —> website\_address, full\_name and head\_prof\_net\_id, while abbreviation is the primary key. It follows the definition of 3NF. Then DEPARTMENT is in 3NF.

EMPLOYEE (ssn, net\_id, salary)

In EMPLOYEE table, net\_id —> ssn and salary, and net\_id is primary key. It follows the definition of 3NF. Then EMPLOYEE is in 3NF.

INSTRUCTOR (net\_id)

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

LAB (room\_number, building\_abbreviation, name)

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

LECTURER (net\_id, office\_roomnumber, office\_building\_abbreviation, office\_hour)

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

C.

BUILDING (abbreviation, full\_name, dept\_abbreviation)

CREATE TABLE BUILDING

(

abbreviation VARCHAR(255) NOT NULL,

full\_name VARCHAR(255) NOT NULL,

dept\_abbreviation VERCHAR(255) NOT NULL,

CONSTRAINT pk\_building PRIMARY KEY

(abbreviation)

);

CLASSROOM (building\_abbreviation, room\_number, capacity, computer\_password)

CREATE TABLE CLASSROOM

(

building\_abbreviation VARCHAR(255) NOT NULL,

room\_number INTEGER NOT NULL,

capacity INTEGER NOT NULL,

computer\_password VARCHAR(255),

CONSTRAINT pk\_classroom PRIMARY KEY

(building\_abbreviation, room\_number)

CHECK (room\_number>=1000 AND room\_number<=9999)

);

COURSE\_TEXTBOOK (course\_number, textbook)

CREATE TABLE COURSE\_TEXTBOOK

(

course\_number INTEGER NOT NULL,

textbook VARCHAR(255) NOT NULL,

CONSTRAINT pk\_ct PRIMARY KEY

(course\_number, textbook),

CHECK (course\_number>=1000 AND course\_number<=9999)

);

COURSE (course\_number, name, credit\_hour, dept\_abbreviation)

CREATE TABLE COURSE

(

course\_number INTEGER NOT NULL,

name VARCHAR(255) NOT NULL,

credit\_hour INTEGER NOT NULL,

CONSTRAINT pk\_course PRIMARY KEY

(course\_number)

CHECK (credit\_hour>=1 AND credit\_hour<=6)

);

DEPARTMENT (abbreviation, website\_address, full\_name, head\_prof\_net\_id)

CREATE TABLE DEPARTMENT

(

abbreviation VARCHAR(255) NOT NULL,

website\_address VARCHAR(255),

full\_name VARCHAR(255) NOT NULL,

head\_prof\_net\_id VERCHAR(255) NOT NULL,

CONSTRAINT pk\_department PRIMARY KEY

(abbreviation)

);

EMPLOYEE (ssn, net\_id, salary)

CREATE TABLE EMPLOYEE

(

ssn INTEGER NOT NULL,

net\_id VARCHAR(255) NOT NULL,

salary DECIMAL NOT NULL,

CONSTRAINT pk\_employee PRIMARY KEY

(net\_id)

);

INSTRUCTOR (net\_id)

CREATE TABLE INSTRUCTOR

(

net\_id INTEGER NOT NULL,

CONSTRAINT pk\_instructor PRIMARY KEY

(net\_id)

);

LAB (room\_number, building\_abbreviation, name)

(

room\_number INTEGER NOT NULL,

building\_abbreviation VACHAR(255) NOT NULL,

name VACHAR(255) NOT NULL,

CONSTRAINT pk\_lab PRIMARY KEY

(room\_number, building\_abbreviation),

CHECK (room\_number>=1000 AND room\_number<=9999)

);

LECTURER (net\_id, office\_roomnumber, office\_building\_abbreviation, office\_hour)

(

net\_id VACHAR(255) NOT NULL

office\_roomnumber INTEGER NOT NULL,

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

office\_hour DECIMAL NOT NULL,

CONSTRAINT pk\_lecturer PRIMARY KEY

(net\_id),

CHECK (office\_roomnumber>=1000 AND office\_roomnumber<=9999)

)

D. Use the Create View statement to create the following views:

Student workers: List name and id of students who work as TA and/or RA, with their workloads. If a student work as both TA and RA, or if she work as TA for several course sections, show her total workload.

CREATE VIEW [Student work as TA and/or RA] AS

SELECT p.last\_name, p.middle\_name, p.first\_name, p.net\_id, SUM (rwa.workload, sht.workload)

FROM PEOPLE p, STUDENT s, TA ta, RA ra, RA\_WORK\_ASSIGNMENT rwa, SECTION\_HAS\_TA sht

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

OR (s.net\_id=ra.net\_id)

AND (ra.net\_id=rwa.ra\_net\_id)

AND (ta.net\_id=sht.ta\_net\_id)

GROUP BY s.net\_id;

E.

11. For each track of CS department, retrieve their name, number of core courses, and number of students.

SELECT t.name, COUNT (DISTINCT tcc.course\_number), COUNT (DISTINCT s.net\_id)

FROM STUDENT s, TRACK t, TRACK\_CORE\_COURSE tcc

WHERE (t.dept\_abbreviation=’CS’)

AND (t.name=s.track\_name)

AND (t.name=tcc.track\_name)

LEFT JOIN TRACK

ON t.name=s.track\_name

GROUP BY t.name;

12. Retrieve the average salary of lecturers who instruct at least 3 course sections.

SELECT AVG (e.salary)

FROM EMPLOYEE e, (

SELECT l.net\_id, COUNT (s.course\_number)

FROM LECTUROR l, SECTION s

LEFT JOIN LECTUROR

ON s.instructor\_net\_id=l.net\_id

GROUP BY l.net\_id

HAVING COUNT (s.course\_number)>=3)

WHERE (l.net\_id=e.net\_id);

13. Retrieve the name and id of professors who run exactly one lab and their lab and office are in the same building.

SELECT p.Fname, p.Lname, prof.net\_id

FROM PEOPLE p, PROFESSOR prof, RA\_WORK\_ASSIGNMENT rwa,

(SELECT rwa.prof\_net\_id, COUNT (rwa.room\_number)

FROM RA\_WORK\_ASSIGNMENT rwa

GROUP BY rwa.prof\_net\_id

HAVING COUNT (rwa.room\_number)=1)

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

AND (prof.net\_id=rwa.prof\_net\_id)

AND (rwa.building\_abbreviation=prof.office\_building\_abbreviation)

14. For each department, retrieve the name of the highest paid professor and the name of lab(s) she run.

SELECT p.Fname, p.Lname, l.name

FROM PEOPLE p, PROFESSOR prof, EMPLOYEE e, RA\_WORK\_ASSIGNMENT rwa, LAB l,

(SELECT e.net\_id, MAX (e.salary)

FROM EMPLOYEE e, PROFESSOR prof

WHERE (e.net\_id=prof.net\_id)

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

AND (prof.net\_id=rwa.prof\_net\_id)

AND (rwa.room\_number=l.room\_number); /\*这里key由room\_number和building\_abbreviation同时构成，不知道怎么表示合适\*/

/\*还有一个疑问，这里还需不需要LEFT JOIN PROFESSOR ON (prof.net\_id=rwa.prof\_net\_id) GROUP BY rwa.prof\_net\_id？什么时候需要用到JOIN还是有些疑惑\*/

15. Retrieve the name and email address of students with highest GPA.

SELECT p.Fname, p.Lname, p.email

FROM PEOPLE p, TAKE t,

(SELECT t.student\_net\_id, MAX (gpa)

FROM TAKE t,

(SELECT AVG (t.grade)

AS gpa /\*可以在这里这样定义gpa直接在MAX中引用吗？\*/

FROM TAKE t, STUDENT s

LEFT JOIN STUDENT

ON t.student\_net\_id=s.net\_id

GROUP BY t.student\_net\_id)

GROUP BY t.student\_net\_id)

WHERE (p.net\_id=t.student\_net\_id);