Question 1

1.1

```
CREATE TABLE Student ( SNUM INTEGER,
SName CHAR(20),
Major CHAR(20),
level CHAR(20),
age INTEGER,
PRIMARY KEY (SNUM));

CREATE TABLE Professor( PID INTEGER,
FName CHAR(20),
DeptID INTEGER,
PRIMARY KEY (PID));

CREATE TABLE Class ( Name CHAR(20),
meetsat TIME,
room CHAR(10),
fid INTEGER,
PRIMARY KEY (Name),
FOREIGN KEY (PID) REFERENCES Professor);

CREATE TABLE Enrolled ( SNUM INTEGER,
cname CHAR(20),
PRIMARY KEY (SNUM, cname),
FOREIGN KEY (SNUM) REFERENCES Student,
FOREIGN KEY (cname) REFERENCES Class);
```

1.2 A. CREATE TABLE Enrolled (SNUM INTEGER, cname CHAR (20) ,

PRIMARY KEY (SNUM, cname) ,

FOREIGN KEY (snum) REFERENCES Student),

FOREIGN KEY (cname) REFERENCES Class,),

CHECK ((SELECT COUNT (E.SNUM)

FROM Enrolled E

GROUP BY E.cname)>=15)

CHECK ((SELECT COUNT (E.SNUM)

FROM Enrolled E

GROUP BY E.cname)<=30));

B. This is already guaranteed because rooms are associated with classes, and a new room cannot be declared without an associated class in it.

C. CREATE ASSERTION Teaches

CHECK ((SELECT COUNT (*)

FROM Professor P, Class C

WHERE P.PID = C.PID

GROUP BY C.PID

HAVING COUNT (*) < 2) = 0)

D. CREATE TABLE Class (Name CHAR(20) , meetsat TIME, room CHAR(10), fid INTEGER, PRIMARY KEY (Name),

FOREIGN KEY (fid) REFERENCES Professor),

CHECK ((SELECT COUNT (*)

FROM (SELECT C.room, C.meetsat

FROM Class C

GROUP BY C.room, C.meetsat

HAVING COUNT (*) > 1))= 0))

E. CREATE ASSERTION DifferentRoom

CHECK ((SELECT COUNT (*)

FROM Professor Pl, Professor P2, Class Cl, Class C2

WHERE Pl.fid = Cl.fid

AND P2.fid = C2.fid

AND Cl.room = C2.room

AND Pl.DeptID!= P2.DeptID) = O)

2. A. CREATE TABLE Emp (

eid INTEGER,

ename CHAR(20),

age INTEGER,

salary REAL,

PRIMARY KEY (eid),

CHECK (salary > 5000)

);

B. CREATE ASSERTION ManagerIsEmployee

CHECK ((SELECT COUNT (*)

FROM DEPT D

WHERE D.managerID NOT IN(

SELECT * FROM EMP

))= 0);

C. CREATE TABLE Works (

eid INTEGER,

did INTEGER,

pct_time INTEGER,

PRIMARY KEY (eid, did),

CHECK ((SELECT COUNT (W.eid)

```
FROM Works W
GROUP BY W.eid
HAVING Sum(pct_time) > 100) = 0));

D. CREATE ASSERTION ManagerSalary
CHECK (SELECT E.EID
FROM EMP E, EMP M, WORKS W, DEPT D
WHERE E.EID = W.EID
AND W.DID = D.DID AND D.managerID = M.EID AND E.Salary > M.Salary
);

E. CREATE TRIGGER GiveRaise AFTER UPDATE ON Emp
WHEN old.salary < new.salary
FOR EACH ROW
BEGIN
UPDATE Emp M
SET M.Salary = new.salary
WHERE M.salary < new.salary AND M.EID IN (
SELECT D.mangerID
FROM Emp E, Works W, Dept D
WHERE E.eid = new.eid AND E.eid = W.eid AND W.did = D.did);

F. CREATE TRIGGER GiveRaise AFTER UPDATE ON EMP
WHEN old.salary < new.salary
FOR EACH ROW
        DECLARE raise REAL;
BEGIN
        Raise := new.salary – old.salary;
        UPDATE EMP M
```

```
        SET M.Salary = new.salary
        WHERE M.Salary < new.salary
        AND M.EID IN ( SELECT D.managerID
                        FROM EMP E, WORKS W, DEPT D
                        WHERE E.EID = new.EID AND E.EID = W.EID AND W.DID = D.DID);
        UPDATE DEPT D
        SET D.Budget = D.budget + raise
        WHERE D.DID IN (SELECT W.DID
                                FROM EMP E, WORKS W, DEPT D
                                WHERE E.EID = new.EID AND E.EID = W.EID AND W.DID = D.DID
                                AND D.Budget < ( SELECT SUM(E2.Salary)
                                FROM EMP E2. WORKS W2
                                WHERE E2.EID = W2.EID
                                AND W2.DID = E2.DID ));
END
```