```
AND     D.managerid = E2.eid
AND     E.salary > E2.salary )
```

**Exercise 5.8** Consider the following relations:

Student(*snum:* `integer`, *sname:* `string`, *major:* `string`,
*level:* `string`, *age:* `integer`)
Class(*name:* `string`, *meets_at:* `time`, *room:* `string`, *fid:* `integer`)
Enrolled(*snum:* `integer`, *cname:* `string`)
Faculty(*fid:* `integer`, *fname:* `string`, *deptid:* `integer`)

The meaning of these relations is straightforward; for example, Enrolled has one record per student-class pair such that the student is enrolled in the class.

1. Write the SQL statements required to create these relations, including appropriate versions of all primary and foreign key integrity constraints.
2. Express each of the following integrity constraints in SQL unless it is implied by the primary and foreign key constraint; if so, explain how it is implied. If the constraint cannot be expressed in SQL, say so. For each constraint, state what operations (inserts, deletes, and updates on specific relations) must be monitored to enforce the constraint.
   (a) Every class has a minimum enrollment of 5 students and a maximum enrollment of 30 students.
   (b) At least one class meets in each room.
   (c) Every faculty member must teach at least two courses.
   (d) Only faculty in the department with *deptid=33* teach more than three courses.
   (e) Every student must be enrolled in the course called Math101.
   (f) The room in which the earliest scheduled class (i.e., the class with the smallest *meets_at* value) meets should not be the same as the room in which the latest scheduled class meets.
   (g) Two classes cannot meet in the same room at the same time.
   (h) The department with the most faculty members must have fewer than twice the number of faculty members in the department with the fewest faculty members.
   (i) No department can have more than 10 faculty members.
   (j) A student cannot add more than two courses at a time (i.e., in a single update).
   (k) The number of CS majors must be more than the number of Math majors.

(l) The number of distinct courses in which CS majors are enrolled is greater than the number of distinct courses in which Math majors are enrolled.

(m) The total enrollment in courses taught by faculty in the department with *deptid=33* is greater than the number of Math majors.

(n) There must be at least one CS major if there are any students whatsoever.

(o) Faculty members from different departments cannot teach in the same room.

**Answer 5.8** Answers are given below.

1. The SQL statements needed to create the tables are given below:

```
CREATE TABLE Student ( snum    INTEGER,
                       sname   CHAR(20),
                       major   CHAR(20),
                       level   CHAR(20),
                       age     INTEGER,
                       PRIMARY KEY (snum))

CREATE TABLE Faculty ( fid     INTEGER,
                       fname   CHAR(20),
                       deptid  INTEGER,
                       PRIMARY KEY (fnum))

CREATE TABLE Class (   name    CHAR(20),
                       meets_atTIME,
                       room    CHAR(10),
                       fid     INTEGER,
                       PRIMARY KEY (name),
                       FOREIGN KEY (fid) REFERENCES Faculty)

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

2. The answer to each question is given below

   (a) The Enrolled table should be modified as follows:

```
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) >= 5),
                                CHECK (( SELECT   COUNT (E.snum)
                                         FROM     Enrolled E
                                         GROUP BY E.cname) <= 30))
```

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

(c) Create an assertion as follows:

```
CREATE ASSERTION TeachTwo
CHECK ( ( SELECT   COUNT (*)
          FROM     Facult F, Class C
          WHERE    F.fid = C.fid
          GROUP BY C.fid
          HAVING   COUNT (*) < 2) = 0)
```

(d) Create an assertion as follows:

```
CREATE ASSERTION NoTeachThree
CHECK ( ( SELECT   COUNT (*)
          FROM     Facult F, Class C
          WHERE    F.fid = C.fid AND F.deptid ≠ 33
          GROUP BY C.fid
          HAVING   COUNT (*) > 3) = 0)
```

(e) Create an assertion as follows:

```
CREATE ASSERTION InMath101
CHECK (( SELECT   COUNT (*)
         FROM     Student S
         WHERE    S.snum NOT IN ( SELECT   E.snum
                                  FROM     Enrolled E
                                  WHERE    E.cname = 'Math101')) = 0)
```

(f) The Class table should be modified as follows:

```
CREATE TABLE Class (   name      CHAR(20),
                       meets_at  TIME,
                       room      CHAR(10),
                       fid       INTEGER,
```

```
                    PRIMARY KEY (name),
                    FOREIGN KEY (fid) REFERENCES Faculty),
                    CHECK (   (SELECT   MIN (meets_at)
                              FROM      Class) <>
                              (SELECT   MAX (meets_at)
                              FROM      Class)))
```

(g) The Class table should be modified as follows:

```
CREATE TABLE Class (   name      CHAR(20),
                       meets_at  TIME,
                       room      CHAR(10),
                       fid       INTEGER,
                       PRIMARY KEY (name),
                       FOREIGN KEY (fid) REFERENCES Faculty),
                       CHECK ((SELECT COUNT (*)
                                 FROM (    SELECT C.room, C.meets
                                           FROM Class C
                                           GROUP BY C.room, C.meets
                                           HAVING COUNT (*) > 1)) = 0))
```

(h) The Faculty table should be modified as follows:

```
CREATE TABLE Faculty ( fid       INTEGER,
                       fname     CHAR(20),
                       deptid    INTEGER,
                       PRIMARY KEY (fnum),
                       CHECK (   (SELECT MAX (*)
                                 FROM ( SELECT COUNT (*)
                                        FROM Faculty F
                                        GROUP BY F.deptid))
                                 < 2 *
                                 (SELECT MIN (*)
                                 FROM ( SELECT COUNT (*)
                                        FROM Faculty F
                                        GROUP BY F.deptid))))
```

(i) The Faculty table should be modified as follows:

```
CREATE TABLE Faculty (fid        INTEGER,
                      fname      CHAR(20),
                      deptid     INTEGER,
                      PRIMARY KEY (fnum),
                      CHECK ( ( SELECT    COUNT (*)
                                FROM      Faculty F
                                GROUP BY  F.deptid
                                HAVING    COUNT (*) > 10) = 0))
```

(j) This constraint cannot be done because integratey constraints and assertions only affect the content of a table, not how that content is manipulated.

(k) The Student table should be modified as follows:

```
CREATE TABLE Student ( snum    INTEGER,
                       sname   CHAR(20),
                       major   CHAR(20),
                       level   CHAR(20),
                       age     INTEGER,
                       PRIMARY KEY (snum),
                       CHECK ((SELECT COUNT (*)
                               FROM Student S
                               WHERE S.major = 'CS') >
                               (SELECT COUNT (*)
                               FROM Student S
                               WHERE S.major = 'Math')))
```

(l) Create an assertion as follows:

```
CREATE ASSERTION MoreCSMajors
CHECK (    (SELECT   COUNT (E.cname)
            FROM     Enrolled E, Student S
            WHERE    S.snum = E.snum AND S.major = 'CS') >
            (SELECT  COUNT (E.cname)
            FROM     Enrolled E, Student S
            WHERE    S.snum = E.snum AND S.major = 'Math'))
```

(m) Create an assertion as follows:

```
CREATE ASSERTION MoreEnrolledThanMath
CHECK (    (SELECT   COUNT (E.snum)
            FROM     Enrolled E, Faculty F, Class C
            WHERE    E.cname = C.name
            AND C.fid = F.fid AND F.deptid = 33) >
            (SELECT  COUNT (E.snum)
            FROM     Student S
            WHERE    S.major = 'Math'))
```

(n) The Student table should be modified as follows:

```
CREATE TABLE Student ( snum    INTEGER,
                       sname   CHAR(20),
                       major   CHAR(20),
```