COM S 461: ASSIGNMENT I

Date Assigned: Aug. 30, 2004
Due: Sep. 15, 2004 in class
Percentage in your final grade: 10%
Maximum score for this assignment: 100 points

Objectives:

1. Practice conceptual database design.
2. Practice SQL.

Questions

1. (50 points): Consider the following set of requirements for a university database that is used to keep track of students' transcripts.

   - For each student, the university maintains student name, student number, social security number, current address and phone, permanent address and phone, birthdate, gender, class (freshman, sophomore, ..., graduate), major department, minor department (if any) and current degree program (B.A., B.S., ..., Ph.D.). Some user applications need to refer to the city, state, and zip code of the student's current address, and to the student's last name. Both social security number and student number have unique values for each student. Each student must have exactly one major department. Each student can have at most one minor department.
   - Each department is described by a name, department code, office phone, and college. Both name and code have unique values for each department.
   - Each course has a course name, description, course number, credit hours, level, and offering department. The value of course number is unique for each course.
   - Each section is associated with an instructor, semester, year, course, and section number. The section number distinguishes different sections of the same course that are taught during the same semester/year. The possible values for the section number are 1, 2, 3, ...; up to the number of sections taught during each semester.
   - A grade report has a student name, section, letter grade, and numeric grade (0, 1, 2, 3, 4 for F, D, C, B, A, respectively).

   Draw an ER diagram that captures all the above requirements. Specify key attribute(s) of each entity set. For each relationship set, specify structural constraints and participation constraints. Write down any other assumptions that are not given, but are used by you to make the ER diagram complete.

   Answer: See Figure 1.

2. (25 points): Given the ER diagram in Figure 2, create the corresponding relations with constraints under your account in the Oracle database. Enter at least five records of your choice into each relation.

   - Put SQL DML commands to create these relations in a file named *createtbl.sql*, followed by the DDL commands to insert at least five rows into each of the relations. Ensure that when executing createtbl.sql, all relations with constraints and data are correctly created.
   - Put DML commands to drop the tables you created using createtbl.sql in another file named *droptbl.sql*. Ensure that all tables created by createtbl.sql are dropped when droptbl.sql is executed. DROP TABLE command with the cascade option is not allowed.

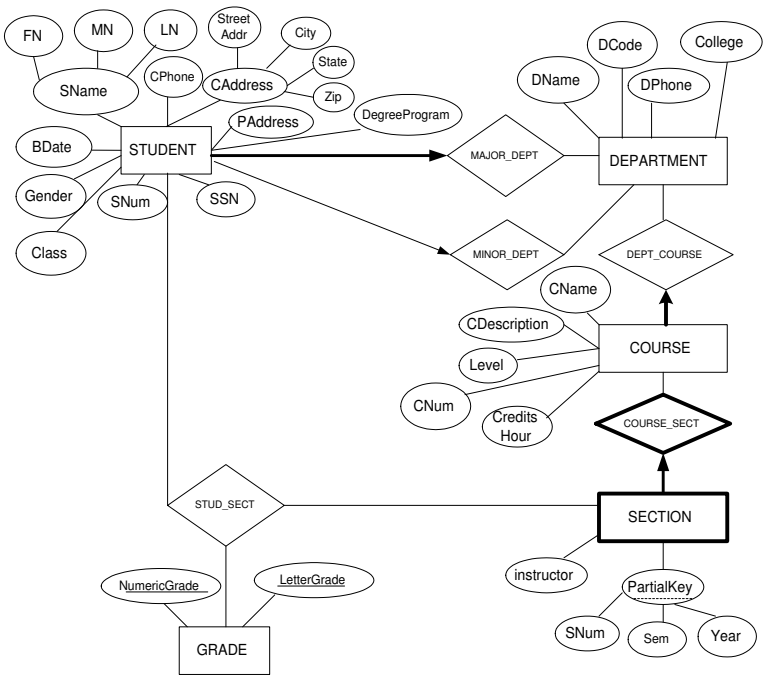

Figure 1: ER diagram of a database to keep track of students' information.

Answer:

One possible set of CREATE TABLE statements to define the database in createtbl.sql is as follows.

```
CREATE TABLE AIRPORT
(AIRPORT_CODE CHAR(3) NOT NULL,
NAME VARCHAR2(30) NOT NULL,
CITY VARCHAR2(30) NOT NULL,
STATE VARCHAR2(30),
PRIMARY KEY (AIRPORT_CODE));

CREATE TABLE FLIGHT
( FNO VARCHAR2(6) NOT NULL,
 AIRLINE VARCHAR2(20) NOT NULL,
 PRIMARY KEY (FNO));

CREATE TABLE FLIGHT_LEG
(FNO VARCHAR2(6) NOT NULL,
 LEG_NO INTEGER NOT NULL,
 DEPT_AIR_CODE CHAR(3) NOT NULL,
 SCHED_DEPT_TIME DATE,
 ARR_AIR_CODE CHAR(3) NOT NULL,
 SCHED_ARR_TIME DATE,
 PRIMARY KEY (FNO, LEG_NO),
 FOREIGN KEY (FNO) REFERENCES FLIGHT,
 FOREIGN KEY (DEPT_AIR_CODE) REFERENCES AIRPORT,
 FOREIGN KEY (ARR_AIR_CODE) REFERENCES AIRPORT);

CREATE TABLE AIRPLANE_TYPE
(TYPE_NAME VARCHAR2(20) NOT NULL,
```

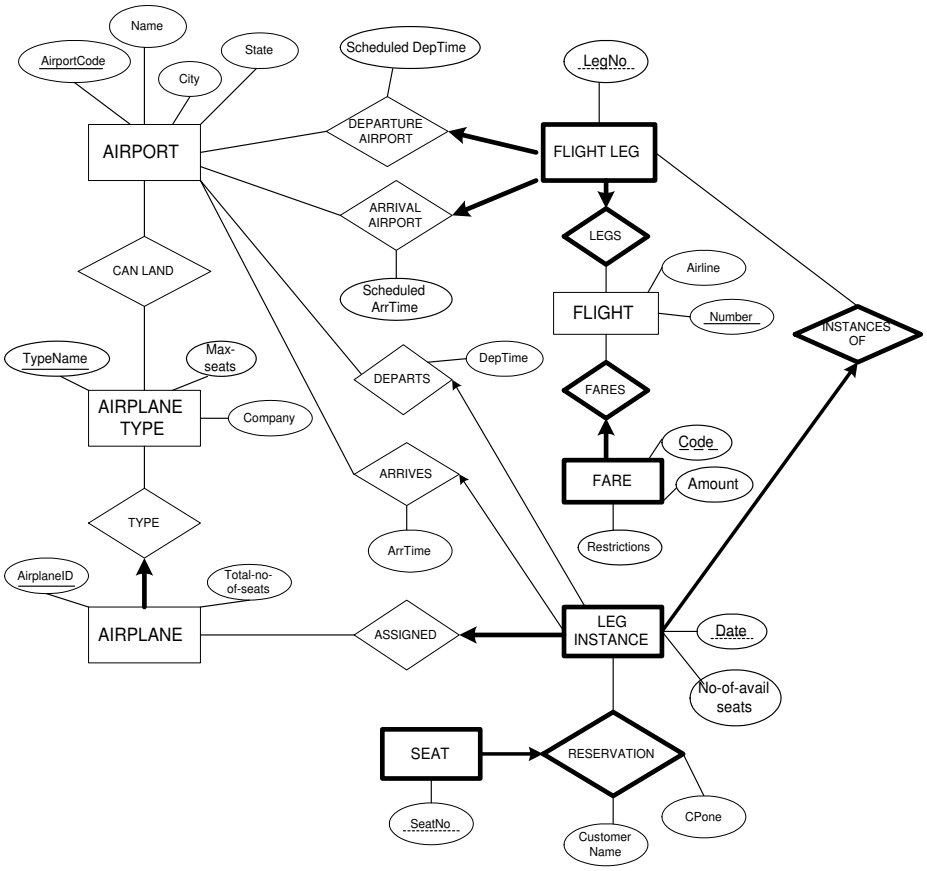

Figure 2: ER diagram of a database to keep track of flight information

```
 MAX_SEATS INTEGER NOT NULL,
 COMPANY VARCHAR2(15) NOT NULL,
 PRIMARY KEY (TYPE_NAME));

CREATE TABLE AIRPLANE
(AIRPLANE_ID INTEGER NOT NULL,
TOTAL_SEATS INTEGER NOT NULL,
AIRPLANE_TYPE VARCHAR2(20) NOT NULL,
PRIMARY KEY (AIRPLANE_ID),
FOREIGN KEY (AIRPLANE_TYPE) REFERENCES AIRPLANE_TYPE);

CREATE TABLE LEG_INSTANCE
(FNO VARCHAR2(6) NOT NULL,
LEG_NO INTEGER NOT NULL,
LEG_DATE DATE NOT NULL,
NO_OF_AVAIL_SEATS INTEGER,
AIRPLANE_ID INTEGER,
DEPT_AIR_CODE CHAR(3),
DEPT_TIME DATE,
ARR_AIR_CODE CHAR(3),
ARR_TIME DATE,
PRIMARY KEY (FNO, LEG_NO, LEG_DATE),
FOREIGN KEY (FNO, LEG_NO) REFERENCES FLIGHT_LEG,
```

```
FOREIGN KEY (AIRPLANE_ID) REFERENCES AIRPLANE,
FOREIGN KEY (DEPT_AIR_CODE) REFERENCES AIRPORT,
FOREIGN KEY (ARR_AIR_CODE) REFERENCES AIRPORT);

CREATE TABLE FARES
(FNO VARCHAR2(6) NOT NULL,
FARE_CODE VARCHAR2(10) NOT NULL,
AMOUNT DECIMAL(8,2) NOT NULL,
RESTRICTIONS VARCHAR2(200),
PRIMARY KEY (FNO, FARE_CODE),
FOREIGN KEY (FNO) REFERENCES FLIGHT);


CREATE TABLE CAN_LAND
(AIRPLANE_TYPE_NAME VARCHAR2(20) NOT NULL,
AIRPORT_CODE CHAR(3) NOT NULL,
PRIMARY KEY (AIRPLANE_TYPE_NAME, AIRPORT_CODE),
FOREIGN KEY (AIRPLANE_TYPE_NAME) REFERENCES AIRPLANE_TYPE,
FOREIGN KEY (AIRPORT_CODE) REFERENCES AIRPORT);


CREATE TABLE SEAT_RESERVATION
(FNO VARCHAR2(6) NOT NULL,
LEG_NO INTEGER NOT NULL,
LEG_DATE DATE NOT NULL,
SEAT_NO VARCHAR2(4),
CUSTOMER_NAME VARCHAR2(30) NOT NULL,
CUSTOMER_PHONE CHAR(12),
PRIMARY KEY (FNO, LEG_NO, LEG_DATE, SEAT_NO),
FOREIGN KEY (FNO, LEG_NO, LEG_DATE) REFERENCES LEG_INSTANCE);
```

For droptbl.sql, we have

```
DROP TABLE SEAT_RESERVATION;
DROP TABLE CAN_LAND;
DROP TABLE FARES;
DROP TABLE LEG_INSTANCE;
DROP TABLE AIRPLANE;
DROP TABLE AIRPLANE_TYPE;
DROP TABLE FLIGHT_LEG;
DROP TABLE FLIGHT;
DROP TABLE AIRPORT;
```

3. (25 points): Consider the following relational schemas.

   Emp(eid:integer, lname:string, fname:string, age:integer, salary:real)
   Dept(dname:char(40), budget:real, managerid:integer)
   Managerid is a foreign key to Emp.
   Works(eid:integer, dname:char(40), *pct_time*:integer)
   Eid is a foreign key to Emp. Dname is a foreign key to Dept.

   An employee can work more than one department; the *pct_time* field of the Works relation shows the percentage of time that a given employee works in a given department.

Write the following queries in SQL. Other column names that are not specified in the questions must not appear in the answers.

(a) Print the last name and age of each employee who works in both the Hardware department and the Software department.

(b) Find the *managerid* of each manager who manages only the departments with budgets greater than $1,000,000.

(c) Find the last name of each manager who manages the departments with the largest budget. The IN operator must be used in the query.

(d) If a manager manages one or more departments, he or she *controls* the sum of all the budgets for those departments. Find the *managerid* of managers who controls more than $5,000,000.

Answer:

(a)
```
SELECT E.lname, E.age
FROM EMP E, Works W1, Works W2,
WHERE E.eid = W1.eid AND W1.dname  = 'Hardware' AND
      E.eid = W2. eid AND W2.dname = 'Software'
```

(b)
```
SELECT D.managerid
FROM Dept D
WHERE 1000000 < ALL (SELECT D2.budget
                     FROM Dept D2
                     WHERE D2.managerid = D.managerid)
```

(c)
```
SELECT E.lname
FROM EMP E
WHERE E.eid IN (SELECT D.managerid
                FROM Dept D
                WHERE D.budget = (SELECT MAX (D2.budget)
                                  FROM Dept D2))
```

(d)
```
SELECT D.mangerid
FROM Dept D
WHERE 5,000,000 < (SELECT SUM (D2.budget)
                   FROM Dept D2
                   WHERE D2.managerid = D.managerid)
```

4. (0 points) You are encouraged to do the odd numbered exercises in Chapter 3 to sharpen your skills as a system analyst. Solutions for the odd numbered questions are provided by the authors of the textbook and accessible through our course Web site under the "Documents" link.

Grading Considerations

- Use the turn in script with the last argument as "hw1" to submit all your files. The script will log the time of the submission. You can submit the same homework several times. Only the latest one will be graded and the time of the latest submission will be used to determine whether the homework is late. The late policy as indicated in the course Web site (www.cs.iastate.edu/∼cs461) will be applied accordingly. Instructions on how to turnin your homework can be found on the website.
- The homework must be done individually. You may discuss with your classmates about the problems, but the solutions must be individual. You are welcome to see the TA or the instructor if the problems are not clear to you.
- Scores will be deducted if the indicated requirements or constraints specified by each question are not provided in your answers.
- Scores will be deducted if submitted scripts cannot be executed successfully from SQL*Plus.