CS482/CS502 – Test 1

8:55pm – 10:10 pm

October 12, 2005

Note:

- There are 5 questions (100 points for 25% of the total grade).
- Be short and precise in your answers.
- There are some questions that are different for **CS482** and **CS502**. Be sure that you do the correct one.

Name:

Signature:

1. (20 points)

(a) Design a database schema and supply the CREATE TABLE commands for a library system containing the following data.

i. the name, unique Id and number of books on loan for each patron;

ii. the unique isbn number, title, author (each book has a single author), current status (possible values are on-shelf or on-loan), borrower Id (if book is on-loan), and shelf-Id of each book;

iii. the unique shelf-Id and capacity (number of books) of each shelf.

Show all primary and foreign keys.

(b) add a constraint that enforces the restriction that a patron cannot borrow more than 5 books at a time.

NOTE: You should decide "which tables should be used to store the data" before you write down the CREATE TABLE commands; the order in which the data is listed should not influence your decision.

Solutions

The first question asks you to create three or four tables:

- Books(ISBN, Title, Author, Status, BorrowerID, ShelfID)
- Patrons(Id, Name, NumBorrowed)
- Shelves(ShelfId, Capacity)

or – you can move BorrowerID and ISBN (from Books) into a new table.

- Books(ISBN, Title, Author, Status, ShelfID)
- Patrons(Id, Name, NumBorrowed)
- Shelves(ShelfId, Capacity)
- BookOnLoan(BorrowerID, ISBN) – the key for this table will contain both attributes.

This is okay since one book is borrowed by only one patron. The commands are:

```
  CREATE TABLE Patrons ( Id INTEGER, Name CHAR(20),
    NumBorrowed INTEGER, PRIMARY KEY (Id) );
  CREATE TABLE Books ( Isbn CHAR(20), Title CHAR(40),
    Author CHAR(20), Status CHAR(5), BorrowerId INTEGER,
    ShelfId INTEGER,
    PRIMARY KEY (Isbn),
    CHECK (Status IN ('on-loan', 'on-shelf') ),
    FOREIGN KEY (ShelfId) REFERENCES Shelves (ShelfId),
    FOREIGN KEY (BorrowerId) REFERENCES Patrons (Id) );
CREATE TABLE Shelves ( ShelfId INTEGER, Capacity INTEGER
    PRIMARY KEY (ShelfId) )
```

(b) Add the following to Patrons: `CHECK (NumBorrowed` $\leq$ `5)`.

2. (40 points) Consider the following enterprise, which includes books, authors and publishers. Authors are people with normal attributes, like name, date of birth, and SSN, but in addition they wrote one or more books. A book has the usual attributes, such as title, ISBN, and publication date. Publishers are companies that publish books. They have an address, phone numbers (typically more than one), name, and stock code. A book can be written by more than one author, but it can be published by only one publisher. Books do not write themselves and do not publish themselves (**hint**: these are constraints). An author can write more than one book and to be called an author one, of course, has to write at least one book.

(a) Represent the above as an E-R diagram; include all relevant constraints.

(b) Translate the above diagram into the relational model by supplying the appropriate CREATE TABLE statements. Note that ISBN is a 10-digit string (which can have leading zeros), sex can have only two values, 'M' or 'F', and a phone number is a 10 digit number that never starts with a zero. Use domains to specify these constraints if you feel comfortable with it. Specify all the key and foreign key constraints. Try to preserve as many participation constraints as possible.

Undergraduate Students (CS482): List all the participation constraints that are present in the E-R diagram, but not in your translation to SQL.

Graduate Students (CS502): Specify SQL constraints that can be used to enforce all participation constraints.

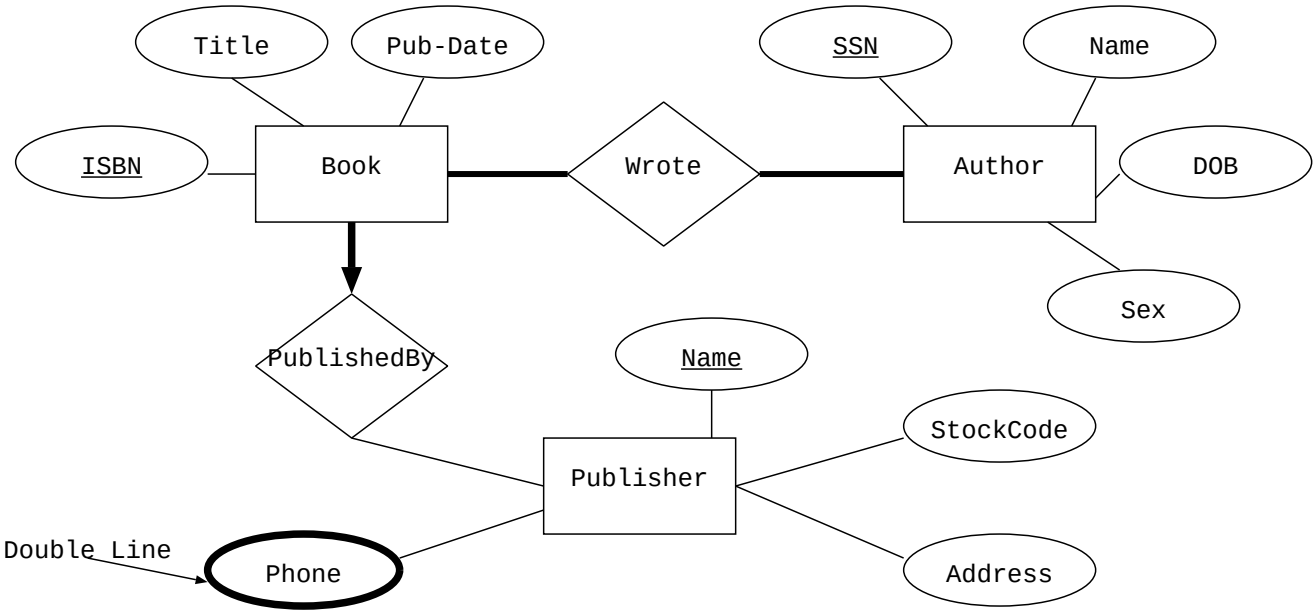

Figure 1: ER-Diagram for the database

```
CREATE DOMAIN ISBN-TYPE CHAR(10)
   CHECK( VALUE BETWEEN '0000000000' AND '9999999999');
```

```
CREATE DOMAIN SEX-TYPE CHAR(1)
   CHECK ( VALUE IN ('M','F'));
CREATE DOMAIN PHONE-TYPE INTEGER
   CHECK ( VALUE > 999999999 AND VALUE < 10000000000 );
CREATE TABLE Book (
   ISBN ISBN-TYPE,   Title CHAR(60), PublicationDate DATE,
   PName CHAR(60) NOT NULL,  PRIMARY KEY (ISBN),
   FOREIGN KEY (PName) REFERENCES Publisher(Pname)
)
CREATE TABLE Author (
   SSN INTEGER, AName CHAR(60), DOB DATE,
   Sex SEX-TYPE,
   PRIMARY KEY (SSN)
)
CREATE TABLE Publisher (
   PName CHAR(60), Address CHAR(60),  Scode   CHAR(10),
   PRIMARY KEY (PName)
)
CREATE TABLE PublisherPhone (
   PName CHAR(60), Phone PHONE-TYPE,
   PRIMARY KEY (PName, Phone),
   FOREIGN KEY (PName) REFERENCES Publisher(Pname)
)
CREATE TABLE Wrote (
   ISBN ISBN-TYPE, ASSN INTEGER,
   PRIMARY KEY (ISBN, ASSN),
   FOREIGN KEY (ISBN) REFERENCES Books(ISBN),
   FOREIGN KEY (ASSN) REFERENCES Author(SSN)
)
```

b) Author has to write at least one book; book has to be written by at least one author.

```
CREATE CONSTRAINT HasToWriteSome CHECK
   (NOT EXISTS (SELECT SSN A FROM Author A
                WHERE (NOT EXIST (SELECT * FROM WROTE W
                                  WHERE W.ASSN = A.SSN))))

CREATE CONSTRAINT BookBySomeOne CHECK
   (NOT EXISTS (SELECT ISBN B FROM BOOK B
                WHERE (NOT EXIST (SELECT * FROM WROTE W
                                  WHERE W.ISBN = B.ISBN))))
```

3. (10 points)

Given the following schema:

```
Student(Id,Name)
Transcript(StudId,CourseName,Semester,Grade)
```

Undergraduate Students (CS482): Formulate the following query in SQL: Create a list of all students (Id, Name) and, for each student, list the average grade for the courses taken in the S2002 semester. Note that there can be students who did not take any courses in S2002. You might or might not include these students. If you include them, the average grade should be listed as 0.

Graduate Students (CS482): Formulate the following query in SQL: Create a list of all students (Id, Name) and, for each student, list the average grade for the courses taken in the S2002 semester. Note that there can be students who did not take any courses in S2002. For these, the average grade should be listed as 0.

Undergraduate Students (CS482):

```
SELECT S.Id, S.Name, AVG(T.Grade) As GPAS2002
FROM   Student S, Transcript T
WHERE  S.Id = T.StudId AND T.Semester='S2002'
GROUP BY S.Id, S.Name
```

Graduate Students (CS482):

```
SELECT S.Id, S.Name, AVG(T.Grade) As GPAS2002
FROM   Student S, Transcript T
WHERE  S.Id = T.StudId AND T.Semester='S2002'
GROUP BY S.Id, S.Name
UNION
SELECT S.Id, S.Name, 0.0 As GPA
FROM   Student S
WHERE  S.Id NOT IN (
        SELECT T.StudId FROM Transcript T
        WHERE  T.Semester='S2002'
       )
```

4. (10 points) Consider an instance of the relation PIQ

ID	Name	IQ	Age
001	One	120	29
002	Two	10	89
003	Three	100	19
004	Four	NULL	9

Suppose that we would like to write a SQL query to *Find the names of all people with a higher IQ than all people wiht age* < 20. Two students supply the following two SQL queries as answer:

```
a)  SELECT I.Name
    FROM   PIQ I
    WHERE  NOT EXISTS (SELECT *
                       FROM PIQ I2
                       WHERE I2.Age < 21 AND I.IQ <= I2.IQ)

b)  SELECT I.Name
    FROM   PIQ I
    WHERE  I.IQ > ALL (SELECT I2.IQ
                       FROM PIQ I2
                       WHERE I2.Age < 21)
```

Compute the results of both queries. From the result, decide which query you would supply as answer if you were to write the SQL query.

For (a) we need to compute — for each row — the result of the subquery.

- For 'One', the subquery returns an empty relation: there is only two persons younger than 21, both have the condition $I2.Age < 21$ and $I.IQ \leq I2.IQ$ retunre false. So, 'One' belongs to the answer.
- For 'Two', the subquery returns the tuple 3.
- For 'Three', the subquery returns the tuple 3.
- For 'Four', the subquery returns an empty relation.

So, the answer is the list of 'One' and 'Four';

For (b) we need to compute the result of the subquery before we precess each row. This result in the the table

IQ
100
NULL

The comparison will fail for every rows. This means that the empty set is the result of the query.

To select the answer, we can see that potentially (a) will give wrong answer (anyone younger than 21 with unknown IQ, including new born baby, will be included in the answer).. Using (b) at least we do not get wrong answer. So, (b) is better then.

5. (20 points)

Consider the following schema:

```
Part(pID : integer, pName : string , color : string)
Supplier (sID : integer, sName : string, Address : string)
Catalog(sId : integer, pId : integer, cost : real)
```

The key of these relations are `pID`, `sID`, and `sID, pID` respectively.

(a) Describe in English the result of the following expression

$$\pi_{sName}(\sigma_{color='red'}(Part) \bowtie \sigma_{cost<100}(Catalog) \bowtie Supplier)$$

Answer: Name of suppliers who sale at least one red part for less than 100.

(b) Write a relational expression for the following queries:

(i) Find the `sIDs` of suppliers who supply red and greee part.

$$\pi_{sId}(\sigma_{color='red'}(Part) \bowtie (Catalog)) \cap \pi_{sId}(\sigma_{color='green'}(Part) \bowtie (Catalog))$$

(ii) Find the `pID` of parts supplied by at least two suppliers.

$$\pi_{pId}(\sigma_{sId \neq sId1}(Catalog \bowtie Catalog[Sid1, pId, Cost1]))$$

(iii) Find the `sIDs` of suppliers who supply every red part.

$$\pi_{sId,pId}(Catalog)/\pi_{pId}(\sigma_{color='red'}(Part))$$