Database Management Systems (COP 5725)

(Spring 2018)

Instructor: Dr. Markus Schneider

TA: Matin Kheirkhahan

Exam 2 Solutions

Name:	
UFID:	
Email Address:	

Pledge (Must be signed according to UF Honor Code)

On my honor, I have neither given nor received unauthorized aid in doing this assignment.

_____

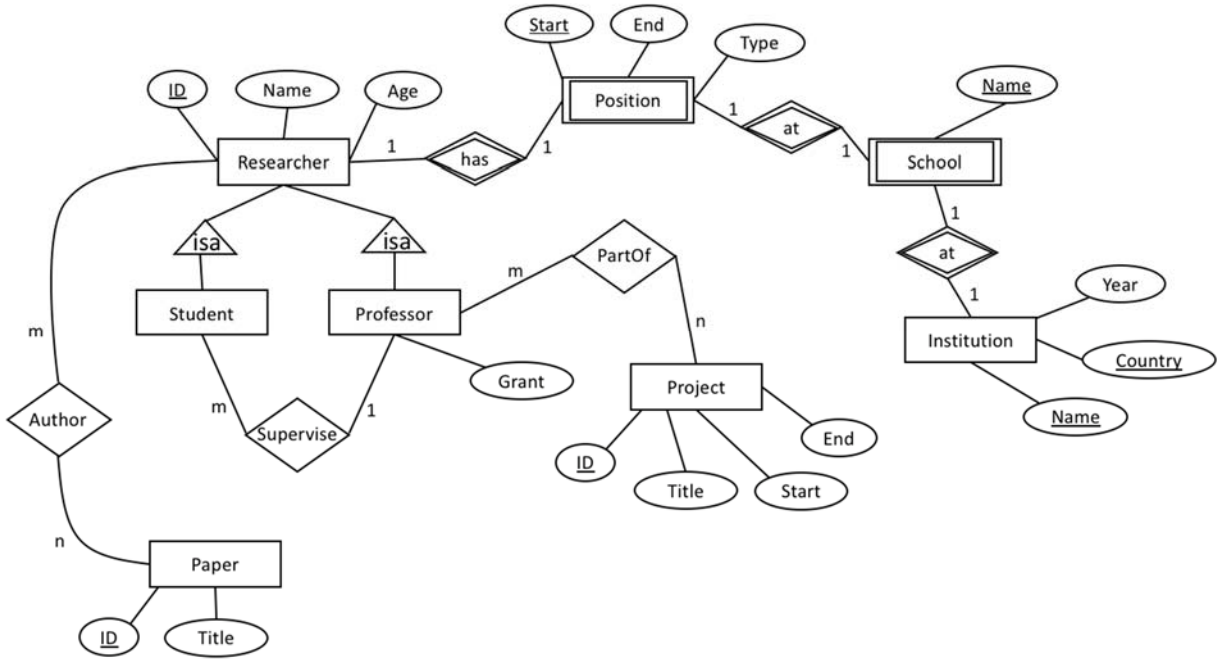
Signature

For scoring use only:

	Maximum	Received
Question 1	25	
Question 2	23	
Question 3	27	
Question 4	25	
Total	100	

Question 1 [25 points]

Consider the following ER diagram.



1. Transform the ER diagram above into a number of relation schemas of the form $R(A_1, A_2, \ldots, A_n)$. Underline all primary keys and list all foreign keys for each relation schema. [10 points]

   - Researcher(ID, name, age)
   - Professor(researcherID, grant)
   - Student(researcherID, professorID)
     - professorID is a foreign key to Professor
   - Paper(ID, title)
   - Position(researcherID, start, schoolName, end, type, institutionName)
     - (schoolName, institutionName) forms a foreign key to School
   - Project(ID, title, start, end)
   - School(name, institutionName, institutionCountry)
   - Institution(name, country, year)
   - Author(researcherID, paperID)
   - PartOf(professorID, projectID)

2. Write SQL commands to create tables for two entities of your choice as well as the relationship between them. Make sure to add the necessary constraints for primary key and foreign keys. [15 points]

Two entities: `Researcher` and `Paper`.

Relation between them: `Author`.

```
CREATE TABLE Researcher
(
    ID NUMBER CONSTRAINT ResearcherKey PRIMARY KEY,
    name VARCHAR(40),
    age NUMBER CONSTRAINT ResearcherAge
        CHECK (age > 0));
```

```
CREATE TABLE Paper
(
    ID NUMBER CONSTRAINT PaperKey PRIMARY KEY,
    Title VARCHAR(100)
);
```

```
CREATE TABLE Author
(
    ResearcherID NUMBER,
    PaperID NUMBER,
    CONSTRAINT AuthorKey PRIMARY KEY (ResearcherID, PaperID),
    CONSTRAINT FK_AuthorResearcher FOREIGN KEY (ResearcherID) REFERENCES Researcher(ID),
    CONSTRAINT FK_AuthorPaper FOREIGN KEY (PaperID) REFERENCES Paper(ID));
```

Question 2 (SQL) [28 points]

Consider the following relational schema:

```
Vehicle (model, manufacturer, type)
Sedan (model, color, year, horsepower, mpg)
SUV (model, color, year, horsepower, mpg, type)
Motorcycle (model, color, year, mpg, type)
```

The `Vehicle` relation has the information about model numbers, manufacturers and types (sedan, SUV, motorcycle) of vehicles. Model numbers (`model`) are unique for all manufacturers and vehicles. The `Sedan` relation has the information about color of the vehicle, the year it was manufactured, its horsepower and Miles Per Gallon (`mpg`). `SUV` has similar information, as well as the type, which can be either 'crossover' or 'sport'. The `Motorcycle` relation records the model number, color of the motorcycle, the year it was manufactured, its Miles Per Gallon (`mpg`) and type, which can be either 'scooter' or 'dirt bike'.

Write SQL statements to answer the following questions:

1. Find the manufacturers that made more than 10 models of motorcycles. [5 points]

```
select   manufacturer
from     Vehicle
where    type = 'motorcycle'
group by manufacturer
having   count(model) > 10;
```

2. Find the minimum and maximum Miles Per Gallon (mpg) of sedans for all manufacturers that also manufacture motorcycles. [5 points]

```
select min(sd.mpg), max(sd.mpg)
from   Vehicle v, Sedan sd
where  v.model = sd.model and
       v.manufacturer in (select manufacturer
                          from   Vehicle
                          where  type = `motorcycle');
```

3. Find the year that automobiles (Sedan or SUV) with the highest horsepower were manufactured. [5 points]

```
select year
from   (select model, horsepower from Sedan union
        select model, horsepower from SUV) am
where  am.horsepower >= all (select horsepower from Sedan union
                             select horsepower from SUV);
```

4. Delete all SUVs made by manufacturers that did not make blue sedans. [4 points]

```
with BlueSedanBuilder(manufacturer) as
(
  select distinct Vehicle.manufacturer
  from            Vehicle, Sedan
  where           Vehicle.model = Sedan.model and Sedan.color = `blue'
)
delete from SUV s
where       not exists
            (select *
             from   Vehicle v
             where  v.model = s.model and
                    v.manufacturer in (select manufacturer
                                       from   BlueSedanBuilder));
```

5. Delete the corresponding records from 4. from the `Vehicle` table. [4 points]

```
with BlueSedanBuilder(manufacturer) as
(
  select distinct Vehicle.manufacturer
  from            Vehicle, Sedan
  where           Vehicle.model = Sedan.model and Sedan.color = `blue'
)
delete from Vehicle v
where       v.type = 'SUV' and
            v.manufacturer not in (select manufacturer
                                   from   BlueSedanBuilder);
```

Question 3 (SQL) [27 points]

A) Consider the following table schemas (primary keys are underlined):

BOOK (BookId, Title, Publishername)
BOOK_AUTHORS (BookId, AuthorName)
PUBLISHER (Name, Address, Phone)
BOOK_COPIES (BookId, BranchId, No_Of_Copies)
BOOK_LOANS (BookId, BranchId, CardNo, DateOut, DueDate)
LIBRARY_BRANCH (BranchId, BranchName, Address)
BORROWER (CardNo, Name, Address, Phone)

Write SQL statements for the following queries:

1. List the names and addresses of all borrowers who have borrowed more than 5 books that are published by "ABC". [5 points]

```
select   br.name, br.address
from     Borrower br, Book_Loans bl, Book b
where    br.cardno=bl.cardno and bl.bookid=b.bookid and b.publishername='ABC'
group by br.cardno, br.name, br.address
having   count(*)>5;
```

2. List the library branch names where the number of copies of the book titled "The Spring" is greater than the number of copies of the book titled "The Pleasure of Coding". [6 points]

```
select a1.branch_name
from   (select lb.branch_id, lb.branch_name, bc.no_of_copies
        from   library_branch lb, book_copies bc, book bk
        where  lb.branch_id=bc.branch_id and bc.book_id=bk.book_id and
               bk.title='The Spring') a1,
       (select lb.branch_id, lb.branch_name, bc.no_of_copies
        from   library_branch lb, book_copies bc, book bk
        where  lb.branch_id=bc.branch_id and bc.book_id=bk.book_id and
               bk.title='The Pleasure of Codeing') a2
where  a1.branch_id=a2.branch_id and a1.no_of_copies>a2.no_of_copies;
```

B) Consider the following table schemas (primary keys are underlined):

Scientists(SSN, name)
AssignedTo(SSN, pcode)
Projects(pcode, name, hours)

Write SQL statements for the following queries:

3. List the name of scientists, the sum of hours he/she will spend on all the projects he/she participates, with the total time greater than the average working hours of all scientists. [5 points]

```
select   s.name, sum(p.hours)
from     scientists s, assignedto a, projects p
where    s.ssn=a.ssn and a.pcode=p.pcode
group by s.ssn, s.name
having   sum(p.hours)> (select avg(p.hours)
                        from   scientists s, assignedto a, projects p
                        where  s.ssn=a.ssn and a.pcode=p.pcode);
```

4. Find the project name that has the most number of scientists participated in and the corresponding number. [5 points]

```
select   p.name, count(*)
from     assignedto a, projects p
where    a.pcode=p.pcode
group by p.pcode, p.name
having   count(*) >= all (select   count(*)
                          from     assignedto a, projects p
                          where    a.pcode=p.pcode
                          group by p.pcode, p.name);
```

5. Find the scientists that have participated in every project. [6 points]

```
select s.name from scientists s
where  not exists ((select projects.pcode from projects)
                   minus
                   (select a.pcode from assignedto a where  a.ssn=s.ssn)
                  );
```

Question 4 (QBE) [25 points]

Assume we have the following schema for a ride-sharing system where passengers can rate their rides:

```
Passenger (pID, pname, age, rating)
Driver (dID, dname, age, cartype)
Ride (pID, dID, day, time, point, comment)
```

Passengers also receive rating from drivers through a different relation. The `rating` in `Passenger` relation is an integer attribute ranging from 1 to 10.

Draw Query-By-Example (QBE) tables for the following queries:

1. Find passengers' names who had rides on 7/1/2017. [5 points]

Passenger	pID	pname	age	rating
	_Id	P._N		

Ride	pID	dID	day	time	point	comment
	_Id		7/1/2017			

2. For each `rating` value in `Passenger` relation (1, 2, …, 10), print those for which the average age is less than 45. [5 points]

Passenger	pID	pname	age	rating
			_A	G.P.

Conditions
AVG._A < 45

3. Find drivers' car types (`cartype`) who had middle-aged passengers (44 < age < 70) on 7/4/2017. [5 points]

Passenger	pID	pname	age	rating
	_Id _Id		<70 >44	

Ride	pID	dID	day	time	point	comment
	_Id	_D	7/4/2017			

Driver	dID	dname	age	cartype
	_D			P.

4. Find names and ages of passengers who had rides with drivers that never gave ride to passenger with `pID` = 1. [5 points]

Passenger	pID	pname	age	rating
	_Id	P._N	P._A	

Ride	pID	dID	day	time	point	comment
¬	_Id 1	_D _D				

5. Delete drivers who had average point less than 2 or received comments with the word “terrible” in them. [5 points]

Driver	dID	dname	age	cartype
D.	_Id			

Ride	pID	dID	day	time	point	comment
		P.G._Id			_P	_C

Conditions
AVG._P < 2 OR _C LIKE '%terrible%'