Database Exercises

Søren Hougaard Mulvad
Rober Wan Bengini

June 20, 2019

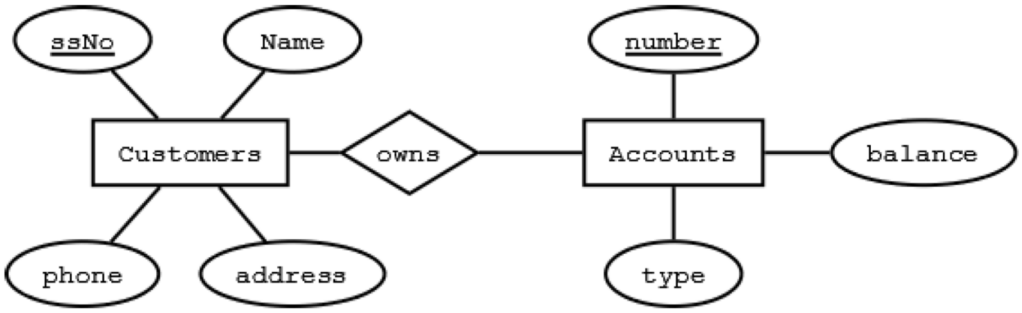
Contents

E/R Modelling

4.1.1 - Designing E/R for bank with customers and accounts

Design a database for a bank, including information about customers and their accounts. Information about a customer includes their name, address, phone, and customer ID. Accounts have numbers, types (e.g., savings, checking) and balances. Also record the customer(s) who own an account. Draw the E/R diagram for this database. Be sure to include arrows where appropriate, to indicate the multiplicity of a relationship.

Solution:

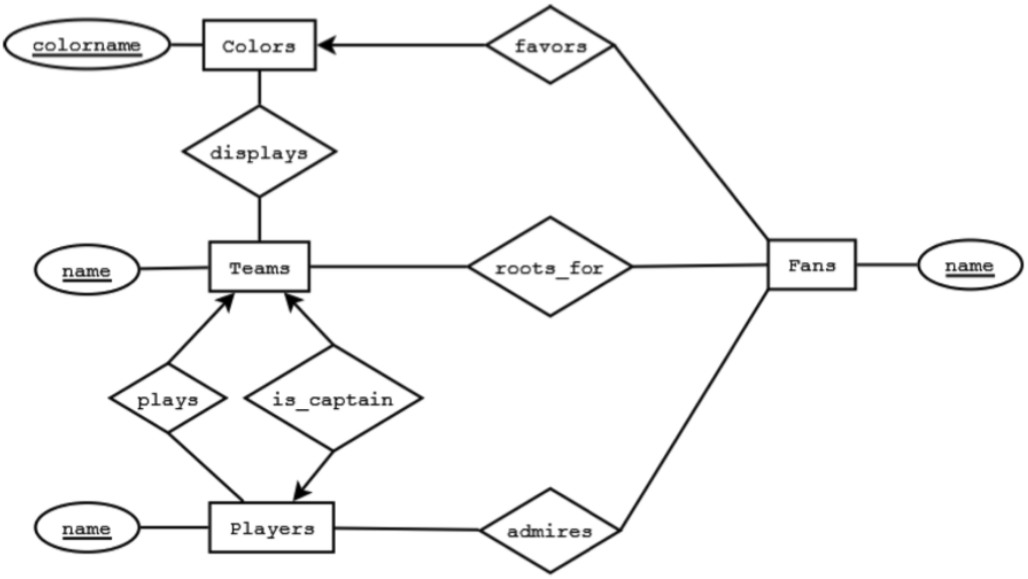


4.1.3 - Designing E/R for teams, players and fans

Give an ER diagram for a database recording information about teams, players, and their fans, including:

1. For each team, its name, its players, its team captain (one of its players), and the color of its uniform
2. For each player, his/her name
3. For each fan, his/her name. favorite teams, favorite players, and favorite colors. Remember that a set of colors is not a suitable attribute type for teams. How can you get around this restriction?

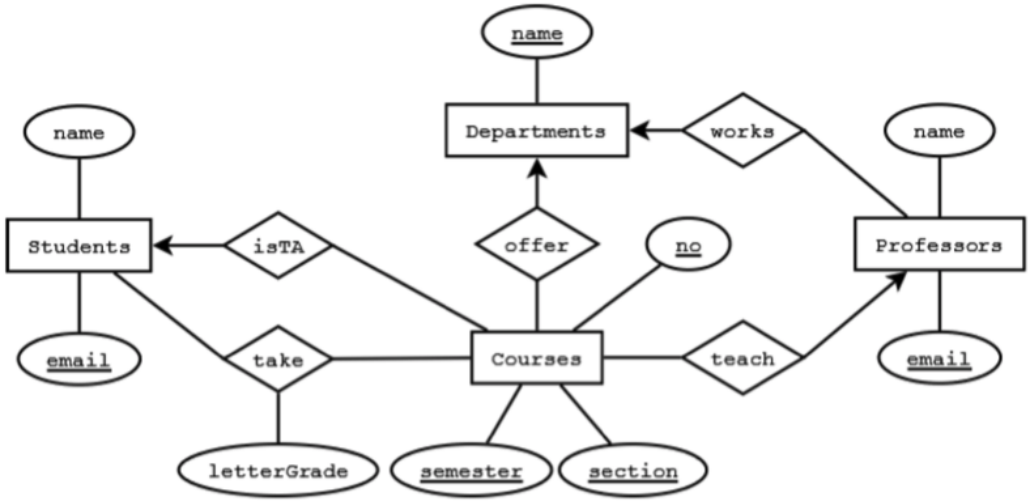
Solution:



4.1.9 - Designing E/R for university registrar

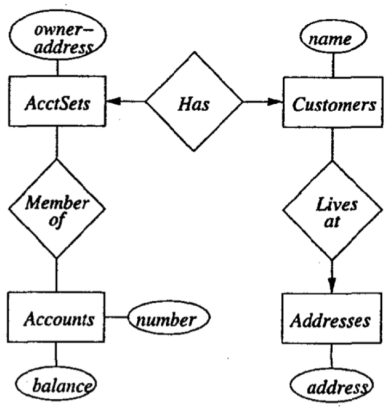
Design a database suitable for a university registrar. This database should include information about students, departments, professors, courses, which students are enrolled in which courses, which professors are teaching which courses, student grades, TAs for a course (TAs are students), which courses a department offers, and any other information you deem appropriate. Note that this question is more free-form than the other questions, and you need to make some decisions about multiplicities of relationships, appropriate types, and even what information needs to be represented.

Solution:
Assumptions: **i)** A Professor only works in at most one department **ii)** A course has at most one TA **iii)** A course is only taught by one professor and offered by one department **iv)** Students and professors have been assigned unique email ids **v)** A course is uniquely identified by the course no, section no, and semester



4.2.1 - Optimizing poor bank E/R diagram

The ER diagram for a bank database involving customers and accounts is given. Since customers may have several accounts, and accounts may be held jointly by several customers, we associate with each customer an "account set," and accounts are members of one or more account sets. Assuming the meaning of the various relationships and attributes are as expected given their names, criticize the design. What design rules are violated? Why? What modifications would you suggest?



Solution:

1. Redundancy: The owner address is repeated in AccSets and Addresses entity sets.

2. Simplicity: AccSets does not serve any useful purpose and the design can be more simply represented by creating many-to-many relationship between Customers and Accounts.

3. Right kind of element: The entity set Addresses has a single attribute address. A customer cannot have more than one address.

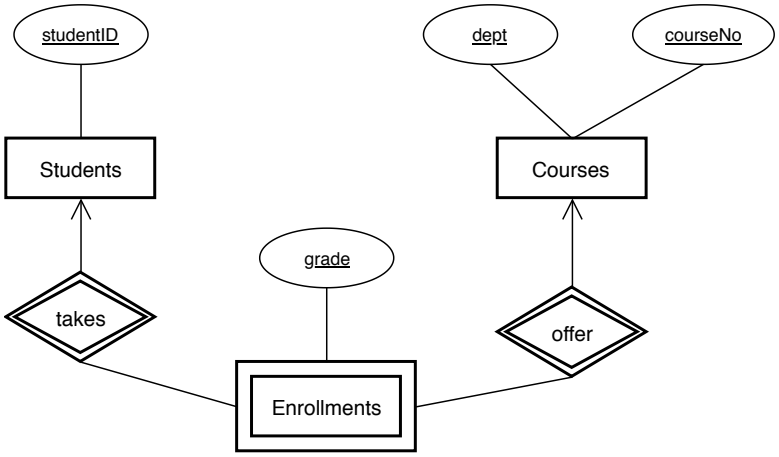
4. Hence address should be an attribute of entity set Customers.

5. Faithfulness: Customers cannot be uniquely identified by their names. In real world Customers would have a unique attribute such as ssNo or customerNo.

4.4.2 - Designing E/R for students, courses and grades using weak entity sets

One way to represent students and the grades they get in courses is to use entity sets corresponding to students, to courses, and to "enrollments." Enrollment entities form a "connecting" entity set between students and courses and can be used to represent not only the fact that a student is taking a certain course, but the grade of the student in the course. Draw an E/R diagram for this situation, indicating weak entity sets and the keys for the entity sets:

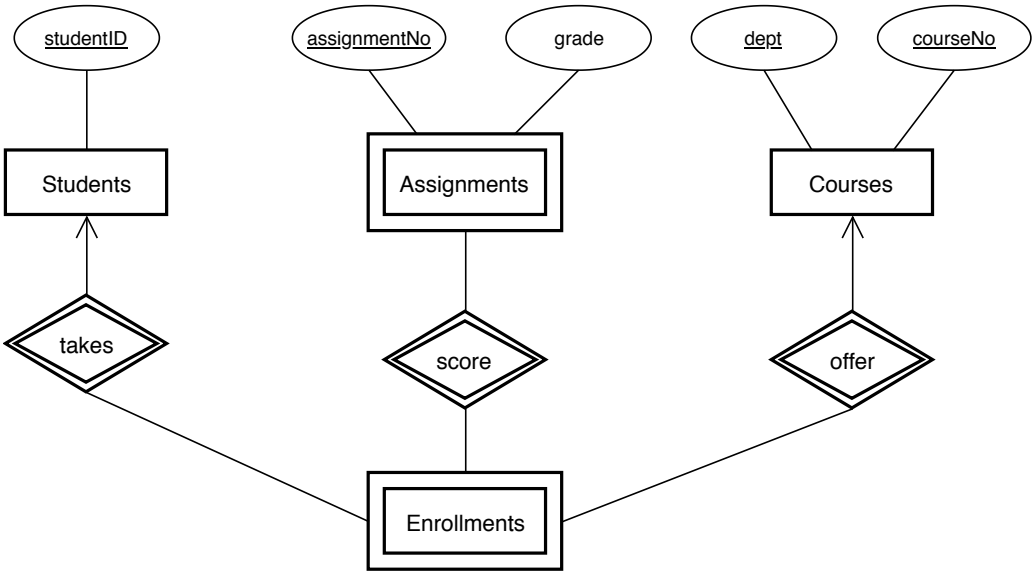
Solution:



4.4.3 - Designing E/R for students, courses and assignment grades using weak entity sets

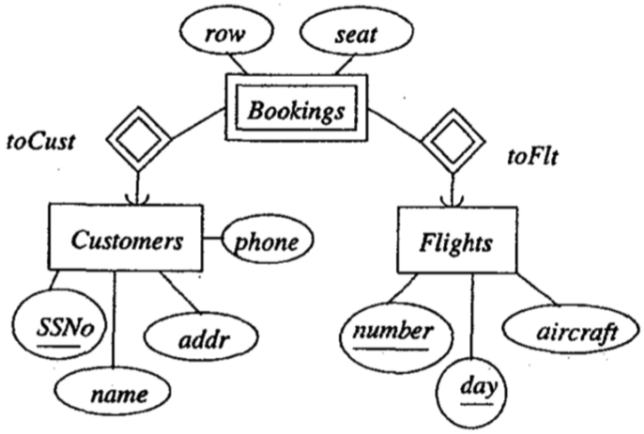
Modify your solution to the previous exercise so that we can record grades of the student for each of several assignments within a course. Again, indicate weak entity sets and keys.

Solution:
It is possible to make assignmentNo a weak key of enrollments but this is not good design (redundancy since multiple assignments correspond to a course). A new entity set Assignment is created and it is also a weak entity set. Hence the key attributes of Assignment will come from the strong entity sets to which enrollments is connected i.e. studentID, dept, and courseNo.



4.5.1 - Converting flight-bookings E/R to relational database schema

Convert the E/R diagram of the flight-booking to a relational database schema.



Solution:

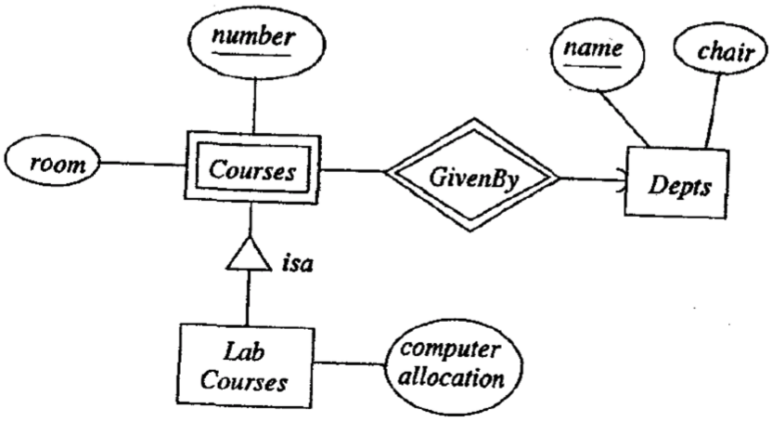
```
Customers(SSNo, name, addr, phone)
Flights(number, day, aircraft)
Bookings(custSSNo, flightNo, flightDay, row, seat)
```

Relations for `toCust` and `toFlt` relationships are not required since the weak entity set Bookings already contains the keys of Customers and Flights.

4.6.1 - Converting E/R using nulls-, OO- and straight E/R-method

Convert the E/R diagram of the figure to a relational database schema, using each of the following approaches:

1. The straight-E/R method.
2. The object-oriented method.
3. The nulls method.



Solution:
The weak relation Courses has the key from `Depts` along with number. Hence there is no relation for `GivenBy` relationship.

1. **Straight E/R:**

   `Depts(`<u>`name`</u>`, chair)`
   `Courses(`<u>`number`</u>`, `<u>`deptName`</u>`, room)`
   `LabCourses(`<u>`number`</u>`, `<u>`deptName`</u>`, alloc)`

2. **Object-oriented method:**

   `Depts(`<u>`name`</u>`, chair)`
   `Courses(`<u>`number`</u>`, `<u>`deptName`</u>`, room)`
   `LabCourses(`<u>`number`</u>`, `<u>`deptName`</u>`, room, alloc)`

3. **Nulls-method:**

   `Depts(`<u>`name`</u>`, chair)`
   `Courses(`<u>`number`</u>`, `<u>`deptName`</u>`, room, alloc)`

Relational Algebra

2.4.1 - Expressions for PC-database

Write expressions answering the questions below for the PC-database. Some sample data is shown:

`Product`

maker	model	type
A	1001	pc
A	1002	pc
D	3004	printer

`PC`

model	speed	ram	hd	price
1001	2.66	1024	250	2114
1002	2.10	512	250	995
1013	3.06	512	80	529

`Laptop`

model	speed	ram	hd	screen	price
2001	2.00	2048	240	20.1	3673
2002	1.73	1024	80	17.0	949
2010	2.00	2048	160	15.4	2300

`Printer`

model	color	type	price
3004	true	ink-jet	120
3005	false	laser	120
3007	true	laser	200

a) Find those manufacturers that sell printers, but not PC's:

$$R1 := \pi_{\text{maker}}(\sigma_{type=\text{'printer'}}(\texttt{Product}))$$
$$R2 := \pi_{\text{maker}}(\sigma_{type=\text{'PC'}}(\texttt{Product}))$$
$$R3 := R1 - R2$$

b) What PC models have a speed of at least 2.50:

$$R1 := \pi_{model}(\sigma_{speed \geq 2.50}(\texttt{PC}))$$

c) Which manufacturers make laptops with a hard disk of at least 120GB:

$$R1 := \pi_{model}(\sigma_{hd \geq 120}(\texttt{Laptop}))$$
$$R2 := \sigma_{type=\text{'laptop'}}(\texttt{Product})$$
$$R3 := \pi_{maker}(R1 \bowtie R2)$$

d) Find the model number and price of all products (of any type) made by manufacturer C:

$$R1 := \pi_{model}(\sigma_{maker=\text{'C'}}(\texttt{Product}))$$
$$R2 := \pi_{model,price}(\texttt{PC})$$
$$R3 := \pi_{model,price}(\texttt{Laptop})$$
$$R4 := \pi_{model,price}(\texttt{Printer})$$
$$R5 := (R1 \bowtie R2) \cup (R1 \bowtie R3) \cup (R1 \bowtie R4)$$

e) Find the model numbers of all black-and-white laser printers:

$$R1 := \pi_{model}(\sigma_{color=false}(\texttt{Printer}))$$

!f) Find those hard-disk sizes that occur in two or more PC's:

$$PC1 := PC$$
$$PC2 := PC$$
$$R1 := PC1 \bowtie_{PC1.hd=PC2.hd \textbf{ AND } PC1.model \neq PC2.model} PC2$$
$$R2 := \pi_{hd}(R1)$$

!g) Find those pairs of PC models that have both the same speed and RAM. A pair should only be listed once, i.e. list *(i, j)* but not *(j, i)*.

$$
\begin{aligned}
PC1 &:= \pi_{model,speed,ram}(PC) \\
PC2 &:= \pi_{model,speed,ram}(PC) \\
R1 &:= PC1 \bowtie_{PC1.speed=PC2.speed \textbf{ AND } PC1.ram=PC2.ram \textbf{ AND } PC1.model>PC2.model} PC2 \\
R2 &:= \rho_{R2(model1,model2)}(\pi_{PC1.model,PC2.model}(R1))
\end{aligned}
$$

!!h) Find those manufacturers of at least two different computers (PC's or Laptops) with speeds of at least 2.20:

$$
\begin{aligned}
R1 &:= \pi_{model}(\sigma_{speed\geq 2.20}(PC)) \cup \pi_{model}(\sigma_{speed\geq 2.20}(Laptop)) \\
Com1 &:= R1 \bowtie Product \\
Com2 &:= R1 \bowtie Product \\
R2 &:= Com1 \bowtie_{Com1.maker=Com2.maker \textbf{ AND } Com1.model>Com2.model} Com2 \\
R3 &:= \pi_{maker}(R2)
\end{aligned}
$$

!!i) Find the manufacturers who sell exactly three different models of PC:

$$
\begin{aligned}
PC1 &:= \pi_{maker,model}(Product \bowtie PC) \\
PC2 &:= PC1 \\
PC3 &:= PC1 \\
PC4 &:= PC1 \\
R1 &:= PC1 \bowtie_{PC1.maker=PC2.maker \textbf{ AND } PC1.model>PC2.model} PC2 \\
R2 &:= R1 \bowtie_{R1.PC1.maker=PC3.maker \textbf{ AND } PC3.model>R1.PC1.model} PC3 \\
R3 &:= R2 \bowtie_{R2.PC3.maker=PC4.maker \textbf{ AND } PC4.model>R2.PC3.model} PC4 \\
R4 &:= \rho_{R4(maker)}(\pi_{PC3.maker}(R2)) \\
R5 &:= \rho_{R5(maker)}(\pi_{PC4.maker}(R3)) \\
R6 &:= R5 - R4
\end{aligned}
$$

!!j) Find the manufacturer(s) of the computer (PC or Laptop) with the highest available speed:

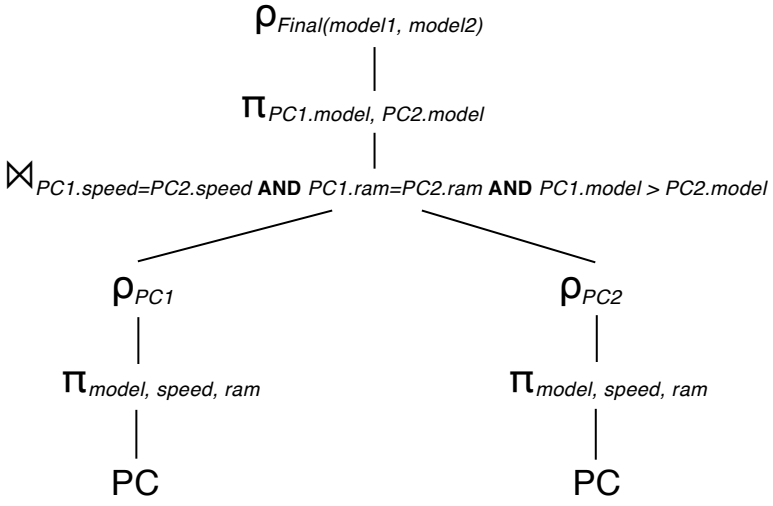
$$
\begin{aligned}
R1 &:= \pi_{model,speed}(PC) \cup \pi_{model,speed}(Laptop) \\
R2 &:= R1 \\
AllExceptMax &:= \rho_{AllExceptMax(model,speed)}(\pi_{R1.model,R1.speed}(R1 \bowtie_{R1.speed<R2.speed} R2)) \\
Max &:= R1 - AllExceptMax \\
R3 &:= \pi_{maker}(Product \bowtie Max)
\end{aligned}
$$

!!k) Find the manufacturers of PC's with at least three different speeds:

$$
\begin{aligned}
R1 &:= \pi_{maker,speed}(Product \bowtie PC) \\
R2 &:= \rho_{R2(maker,s2)}(R1) \\
R3 &:= \rho_{R3(maker,s3)}(R1) \\
R4 &:= (R1 \bowtie R2) \bowtie R3 \\
R5 &:= \pi_{maker}(\sigma_{s3>s2 \textbf{ AND } s2>speed}(R4))
\end{aligned}
$$

2.4.2 - Expression tree for PC-expression

In the following is an expression tree for expression 2.4.1 g)



2.4.3 - Expressions for ship-database

Write expressions answering the questions below for the ship-database. Some sample data is shown:

`Classes`

class	type	country	numGuns	bore	displacement
Bismarck	bb	Germany	8	15	42000
Iowa	bb	USA	9	16	46000
Kongo	bc	Japan	8	14	32000

`Battles`

name	date
Denmark Strait	5/24-27/41
Guadalcanal	11/15/42
North Cape	12/26/42

`Outcomes`

ship	battle	result
Arizona	Pearl Habor	sunk
Bismarck	Denmark Strait	sunk
California	Surigao Strait	ok

`Ships`

name	class	launched
Alabama	South Dakota	1942
Haruna	Kongo	1915
Hiei	Kongo	1914

a) Find the ships launched prior to 1917:

$$R1 := \pi_{ships}(\sigma_{launched<1917}(Ships))$$

b) Find the ships sunk in the battle of Surigao Strait:

$$R1 := \pi_{ship}(\sigma_{result=\text{'sunk'}\ \textbf{AND}\ battle=\text{'Surigao Strait'}}(Outcomes))$$

d) List the name, displacement, and number of guns of the ships engaged in the battle of North Cape:

$$R1 := \sigma_{battle=\text{'North Cape'}}(Outcomes)$$
$$R2 := Ships \bowtie \rho_{R1(name,battle,result)}(R1)$$
$$R3 := \pi_{name,displacement,numGuns}(Classes \bowtie R2)$$

f) Give the class names and countries of the classes that carried guns of at least 16-inch bore:

$$R1 := \pi_{class,country}(\sigma_{bore\geq 16}(Classes))$$

!g) Find those countries that had both battleships and battlecruisers:

$$R1 := \pi_{country}(\sigma_{type=\text{'bb'}}(Classes)) \cap \pi_{country}(\sigma_{type=\text{'bc'}}(Classes))$$

!h) Find those ships that "lived to fight another day"; they were damaged in one battle, but later fought in another:

$$R1 := \rho_{Battles(battle,date)}(Battles) \bowtie Outcomes$$
$$R2 := \sigma_{result=\text{'damaged'}}(R1)$$
$$R3 := R1 \bowtie_{R1.ship=R2.ship\ \textbf{AND}\ R1.date>R2.date} R2$$
$$R4 := \pi_{ship}(R3)$$

!i) Find the classes that had only one ship as a member of that class:

$$R1 := Ships \bowtie_{Ships.class=Ships2.class\ \textbf{AND}\ Ships.name<>Ships2.name} \rho_{Ships2}(Ships)$$
$$R2 := \pi_{class}(Ships) - \rho_{Ships(class)}(\pi_{Ships.class}(R1))$$

2.4.7 - Number of tuples after expression

Suppose relations R and S have n tuples and m tuples, respectively. Give the minimum and maximum numbers of tuples that the results of the following expressions can have:

(a) $R \cup S$
Min is $min(n, m)$
max is $n + m$.

(b) $R \bowtie S$
Joins only where they are equal. The maximum should then be $min(n, m)$.
The minimum is zero as all tuples might be different.

(c) $\sigma_C(R) \times S$ for some condition C
If the condition is true for all tuples $\sigma_C(R) = R$. Thus the maximum of the expression is $R \times S$ - which is $n \cdot m$.
If all evaluate to false $\sigma_C(R) = 0$ (no tuples), then $R \times S$ will also evaluate to 0 as $0 \cdot m = 0$.

(d) $\pi_L(R) - S$ for some list of attributes L.
Maximum is n and happens when $\pi_L(R)$ is not in S.
Minimum is $max(n - m, 0)$ and happens when $\pi_L(R)$ tuples are in S.

5.2.1 - Extended relational algebra

Here are two relations:

$$R(A, B) : \{(1, 2), (3, 4), (1, 2), (3, 5), (4, 5)\}$$
$$S(B, C) : \{(1, 2), (3, 5), (3, 6), (4, 5), (1, 3), (4, 5)\}$$

Compute the following:

a) $\pi_{A^2, B^2 A+B}(R) = \{(1, 4, 3), (9, 16, 7), (1, 4, 3), (9, 25, 8), (16, 15, 9)\}$

b) $\pi_{B-1, C+1}(S) = \{(0, 3), (2, 6), (2, 7), (3, 6), (0, 4), (3, 6)\}$

c) $\tau_{A,B}(R) = \{(1, 2), (1, 2), (3, 4), (3, 5), (4, 5)\}$

d) $\tau_{C,B}(S) = \{(1, 2), (1, 3), (3, 5), (4, 5), (4, 5), (3, 6)\}$

e) $\delta(R) = \{(1, 2), (3, 4), (3, 5), (4, 5)\}$

f) $\delta(S) = \{(1, 2), (3, 5), (3, 6), (4, 5), (1, 3)\}$

g) $\gamma_{A, AVG(B)}(R) = \{(1, 2), (3, 4.5), (4, 5)\}$

h) $\gamma_{B, SUM(C)}(S) = \{(1, 5), (3, 11), (4, 10)\}$

¡i) $\gamma_A(R) = \{(1), (3), (1), (3), (4)\}$
or maybe: $\gamma_A(R) = \{((1, 2), (1, 2)), ((3, 4), (3, 5)), ((4, 5))\}$

¡j) $\gamma_{A, MAX(C)}(R \bowtie S) = \{(3, 5)\}$

k) $R \overset{\circ}{\bowtie}_R S = \{(\perp, 1, 2), (\perp, 3, 5), (\perp, 3, 6), (3, 4, 5), (\perp, 1, 3), (3, 4, 5)\}$

Functional Dependency

3.3.1 - Functional dependency and Boye-Codd normal form

For the following relation schemas and set of FD's,

i) Indicate all the BCNF violations:

ii) Decompose the relations, as necessary, into collections of relations that are in BCNF:

Solution:

a) $R(A, B, C, D)$ with FD's $B \to A, C \to B, D \to C$, and $A \to D$:

We know that for BCNF to hold, the left side of every nontrivial FD must contain a key. This is actually already the case and therefore this relation doesn't violate the BCNF.

b) $R(A, B, C, D)$ with FD's $BC \to D, D \to A$, and $A \to B$:

Decomposing we i.e. get: $R_1(B, C, D), R_2(D, A)$

other a) $R(A, B, C, D)$ with FD's $AB \to C, C \to D, D \to A$:
Start by computing closures:
$\{AB\}^+ = \{A, B, C, D\}$
$\{D\}^+ = \{D, A\}$
$\{C\}^+ = \{C, D, A\}$

Last two violates BCNF.
Now choosing $C \to D$:
$R_1(C, D, A)$ and $R_2(C, B)$
R_2 is in BCNF.

Projected FDs for R_1: $C \to D$, $D \to A$.
$\{C\}^+ = \{C, D, A\}$
$\{D\}^+ = \{D, A\}$

Last one gives violation. Now choosing $D \to A$:
$R_3(D, A)$ and $R_4(D, C)$

Both are in BCNF.

Hence we get: $R_2(C, B)$, $R_3(D, A)$ and $R_4(D, C)$.

other b) $R(A, B, C, D)$ with FD's $B \to C, B \to D, D \to CD$:
We have: $\{B\}^+ = \{B, C, D\}$
And we get a violation.
$R_1(B, C, D)$ and $R_2(B, A)$
$R_2(B, A)$ is in BCNF.

From R_1, we get $B \to C$ and $B \to D$
$\{B\}^+ = \{B, C, D\}$
We have no violations left.

Hence we get: $R_1(B, C, D)$, $R_2(B, A)$

SQL

2.3.1 - Writing simple CREATE and ALTER declarations

Write suitable schemas for the following relations:

```
Product(maker, model, type)
PC(model, speed, ram, hd, price)
Laptop(model, speed, ram, hd, screen, price)
Printer(model, color, type, price)
```

```sql
-- a) A suitable schema for Product
CREATE TABLE Product (
  maker CHAR(100),
  model INT,
  type  CHAR(16),
  PRIMARY KEY (maker, model, type)
);

-- b) A suitable schema for Laptop
CREATE TABLE Laptop (
  model  INT PRIMARY KEY,
  speed  FLOAT,
  ram    INT,
  hd     INT,
  screen FLOAT,
  price  INT
);

-- c) A suitable schema for Printer
CREATE TABLE Printer (
  model INT PRIMARY KEY,
  color BOOLEAN,
  type  CHAR(16),
  price INT
);

-- d) A suitable schema for PC
CREATE TABLE PC (
  model INT PRIMARY KEY,
  speed FLOAT,
  ram   INT,
  hd    INT,
  price INT
);

-- e) Deleting color from Printer
ALTER TABLE Printer DROP color;

-- f) Alter Laptop to add 'od' (optical disk type) and letting the default be 'none'
ALTER TABLE Laptop ADD od CHAR(16) DEFAULT 'none'
```

6.1.2 - Writing simple SELECT statements on Movie-database

Remembering that the Movie-database has the following schema, write the queries described in the comments:

```
Movies(title, year, length, genre, studioName, producerC#)
StarsIn(movieTitle, movieYear, starName)
MovieStar(name, address, gender, birthdate)
MovieExec(name, address, cert#, netWorth)
Studio(name, address, presC#)
```

```sql
-- a) Find Aishwara Rai's birthdate
SELECT birthdate
FROM moviestar
WHERE name = 'Aishwara Rai';


-- b) Find the address of Film City
SELECT address
FROM studio
WHERE name = 'Film City';


-- c) Find all stars that appeared either
-- in a movie made in 2000 or a movie with
-- ''Story'' in the title.
SELECT starname
FROM starsin
WHERE movieyear = 2000
   OR movietitle LIKE '%Story%';


-- d) Find all the stars who are either female
-- or live in Mumbai (have string Mumbai as part
-- of their address)
SELECT name
FROM moviestar
WHERE gender = 'F'
   OR address LIKE '%Mumbai%';


-- e) Find all executives worth at least $20,000,000
SELECT name
FROM movieexec
WHERE networth >= 20000000;
```

6.1.4 - Writing simple SELECT statements on Ships-database

Remembering that the Ships-database has the following schema, write the queries described in the comments:

```
Classes(class, type, country, numGuns, bore, displacement)
Ships(name, class, launched)
Battles(name, date)
Outcomes(ship, battle, result)
```

```sql
-- a) Find the class name and country for
-- all classes with at least 12 guns
SELECT class, country
FROM classes
WHERE numguns >= 12;


-- b) Find the names of all ships launched
-- prior to 1915, but call the resulting
-- column shipName
SELECT name AS shipName
FROM ships
WHERE launched <= 1915;


-- c) Find the names of all ships that
-- begin with the letter M
SELECT name
FROM ships
WHERE name LIKE 'M%'
   OR name LIKE 'm%';


-- d) Find the names of ships sunk in battle
-- and the name of the battle in which they
-- were sunk
SELECT ship, battle
FROM outcomes
WHERE result = 'sunk';


-- e) Find all ships that have the same name
-- as their class
SELECT name
FROM ships
WHERE class = name;


-- !f) Find the names of all ships whose name
-- consists of three or more words (e.g., King George IV)
SELECT S.name
FROM ships S
WHERE S.name LIKE '% % %';
```

6.2.1 - Writing simple SELECT statements on Movies-database

Write the queries described in the comments:

```sql
-- a) Who is the president of Film City
SELECT M.name
FROM studio S, movieexec M
WHERE S.name = 'Film City'
  AND S.presc = M.cert;


-- b) Who were the male stars in MonSoon Wedding
SELECT name
FROM starsin S
  INNER JOIN moviestar MS ON S.starname = MS.name
WHERE S.movietitle = 'Moonsoon Wedding'
  AND MS.gender = 'M';


-- c) Which stars appeared in movies produced by
-- Sony in 2005?
SELECT starname
FROM starsin S
  INNER JOIN movies M ON (S.movietitle = M.title AND S.movieyear = M.year)
WHERE M.studioname = 'Sony';


-- !d) Which executives are worth more than Subhash Ghai:
SELECT name
FROM movieexec
WHERE networth >
  (SELECT networth FROM movieexec WHERE name = 'Subhash Ghai');


-- !e) Which movies are longer than Bride and Prejudice?
SELECT title
FROM movies
WHERE length >
  (SELECT length FROM movies WHERE title = 'Bride and Prejudice');
```

6.2.2 - Writing simple SELECT statements on PC-database

Write the queries described in the comments:

```sql
-- a) Find those manufacturers that sell PC's but not Laptops
(SELECT DISTINCT maker
FROM product
WHERE type = 'pc')
  EXCEPT
(SELECT DISTINCT maker
FROM product
WHERE type = 'laptop');


-- b) Give the manufacturer and speed of laptops with a hard disk of at least 100 gigabytes
SELECT maker, speed
FROM product P
  INNER JOIN laptop L ON P.model = L.model
WHERE L.hd > 100;


-- c) Find the model number and price of all products (of any type) made by manufacturer C
SELECT P.model, price
FROM product P,
((SELECT model, price FROM printer)
  UNION
 (SELECT model, price FROM pc)
  UNION
 (SELECT model, price FROM laptop)) AS prods
WHERE P.maker = 'C'
  AND P.model = prods.model;


-- !d) Find those pairs of PC models that have both the same RAM and hard disk
SELECT pc1.model AS model1, pc2.model AS model2
FROM pc AS pc1, pc AS pc2
WHERE pc1.ram = pc2.ram
  AND pc1.hd = pc2.hd
  AND pc1.model > pc2.model;


-- !e) Find those processor speeds that occur in two or more PC's
SELECT pc1.speed
FROM pc AS pc1, pc AS pc2
WHERE pc1.speed = pc2.speed
  AND pc1.model > pc2.model;


-- !!f) Find those manufacturers of at least two different
-- computers (PC's or laptops) with speeds of at least 2.0
SELECT maker
FROM product P
WHERE P.model IN
((SELECT model FROM pc WHERE speed >= 2.0)
  UNION
 (SELECT model FROM laptop WHERE speed >= 2.0))
```

```
GROUP BY maker
HAVING COUNT(maker) >= 2;
```

6.3.2 - Writing SELECT statements on Ships-database

Remembering that the Ships-database has the following schema, write the queries described in the comments:

```
Classes(class, type, country, numGuns, bore, displacement)
Ships(name, class, launched)
Battles(name, date)
Outcomes(ship, battle, result)
```

```
-- b) Find the names of ships with 9 guns
SELECT name
FROM ships S
WHERE class IN
(SELECT C.class FROM classes C WHERE C.numguns = 9);


-- c) Find the battles in which ships of
-- the South Dakota class participated
SELECT battle
FROM outcomes
WHERE ship IN
(SELECT name FROM ships WHERE class = 'South Dakota');


-- !d) Find the classes of ships, at least
-- one of which was sunk in battle
SELECT DISTINCT S.class
FROM ships S
WHERE S.name IN
    (SELECT O.ship
    FROM outcomes O
    WHERE O.result = 'sunk');
```

6.5.1 - INSERT, UPDATE and DELETE in the PC-database

Remembering that the PC-database has the following schema, write the queries described in the comments:

```
Product(maker, model, type)
PC(model, speed, ram, hd, price)
Laptop(model, speed, ram, hd, screen, price)
Printer(model, color, type, price)
```

```sql
-- a) Delete all PC's with less than 200 GB of hard disk
DELETE FROM pc
WHERE hd < 200;


-- b) Using two INSERT statements, store in the database
-- the fact that PC model 1500 is made by manufacturer A,
-- has speed 3.1, RAM 1024, hd 300 and sells for $2499
INSERT INTO pc VALUES(1500, 3.1, 1024, 300, 2499);
INSERT INTO product VALUES('A', 1500, 'pc');


-- c) Delete all laptops made by a manufacturer that
-- doesn't sell PC's
DELETE FROM product
WHERE type = 'laptop' AND maker IN
  ((SELECT DISTINCT maker
    FROM product)
  EXCEPT
   (SELECT DISTINCT maker
    FROM product
    WHERE type = 'pc'));


-- d) Manufacturer B buys manufacturer C. Change all products
-- made by C so they are now made by B
UPDATE product
SET maker = 'B'
WHERE maker = 'C';


-- e) For each PC, double the amount of hard disk and add 1024
-- megabytes to the RAM
UPDATE pc
SET hd = hd * 2,
    ram = ram + 1024;


-- !f) For each laptop made by manufacturer D, add one inch to
-- the screen size and subtract $200 from the price
UPDATE laptop
SET screen = screen + 1,
    price = price - 200
WHERE model IN (SELECT model FROM product WHERE maker = 'D');
```

7.1.5 - CREATE Ships/Outcomes tables with constraints

Create the tables with the described constraints:

```
-- Referential integrity constraints in Ships database
-- a) Every battle mentioned in Outcomes must be mentioned in Battles
-- b) Every ship mentioned in Outcomes must be mentioned in Ships
CREATE TABLE Outcomes (
    ship   CHAR(32) REFERENCES Ships(name),
    battle CHAR(32) REFERENCES Battles(name),
    result CHAR(16),
    PRIMARY KEY(ship, battle)
);


-- c) Every class mentioned in Ships must be mentioned in Classes
CREATE TABLE Ships (
    name     CHAR(32),
    class    CHAR(32) REFERENCES Classes(class),
    launched TIMESTAMP
);
```

7.2.1 - CREATE Movies table with constraints

```
-- a,b,c) The length cannot be less than 30 nor more than 500,
-- the year cannot be before 1909 and the genre can only be
-- drama, comedy, sciFi or teen
CREATE TABLE Movies(
    title      CHAR(32),
    year       INT CHECK(year >= 1909),
    length     INT CHECK(length >= 30 AND length <= 500),
    genre      CHAR(32) CHECK(genre IN ('drama', 'comedy', 'sciFi', 'teen')),
    studioName CHAR(32),
    producerC  INT,
    PRIMARY KEY(title, year)
);
```

7.2.3 - CREATE Movies table with complex constraints

```
-- NB: Subqueries don't work in 'Check' in PostgreSQL, but syntax follows book

-- a) A star may not appear in a movie made before they were born
CREATE TABLE StarsIn2(
    movieTitle VARCHAR(64),
    movieYear INT,
    starName VARCHAR(64)
      CHECK(movieYear > (SELECT EXTRACT(YEAR FROM birthdate) FROM moviestar WHERE name = starName)),
    PRIMARY KEY(movieTitle, movieYear, starName)
);


-- !b) A studio name that appears in Studio must also
-- appear in at at least one Movies tuple
CREATE TABLE Studio (
```

```
    name    VARCHAR(32) CHECK(name IN (SELECT studioname FROM movies)),
    address VARCHAR(64),
    presc   INT,
    PRIMARY KEY(name)
);
```

7.5.2 - Creating TRIGGERs for pc-database

Create the triggers described. If a trigger is met, the changes shouldn't take effect:

```
-- a) When inserting a new laptop, check
-- that the model exists in Product
CREATE TRIGGER LaptopModelNumExists
BEFORE INSERT ON Laptop
REFERENCING
    NEW ROW AS NewRow
    NEW TABLE AS NewStuff
FOR EACH ROW
WHEN NewRow.model NOT IN (SELECT Model FROM Product)
BEGIN
    DELETE FROM Laptop
    WHERE (model, speed, ram, hd, screen, price) IN NewStuff;
END;


-- b) When updating the price of a printer, check that there
-- is no lower priced printer of the same type
CREATE TRIGGER NoLowerPrinterPrice
BEFORE UPDATE OF price on Printer
REFERENCING
    OLD ROW AS OldRow
    NEW ROW AS NewRow
    OLD TABLE AS OldStuff
    NEW TABLE AS NewStuff
FOR EACH ROW
WHEN NewRow.price > (SELECT MIN(price) FROM OldStuff WHERE OldStuff.type = NewRow.type)
UPDATE NewStuff SET price = OldRow.price;
-- Not quite sure with this one...
```

8.1.1 - CREATE VIEWs in Movie-database

Create the following VIEWs:

```
-- a) A view StudioPress giving the name, address,
-- and certificate number of all executives who are
-- studio presidents
CREATE VIEW StudioPress(name, address, cert) AS
    SELECT ME.name, ME.address, ME.cert
    FROM movieexec ME
        INNER JOIN studio S ON ME.cert = S.presc;


-- b) A view ExecutiveStar giving the name, address,
-- gender, birth date, certificate number, and net worth
-- of all individuals who are both executives and stars
CREATE VIEW ExecutiveStar(name, address, gender, birthdate, cert, networth) AS
    SELECT MS.name, MS.address, MS.gender, MS.birthdate, ME.cert, ME.networth
    FROM movieexec ME, moviestar MS
    WHERE ME.name = MS.name
      AND ME.address = MS.address;

-- c) A view RichExec giving giving the name, address,
-- certificate number and net worth of all executives
-- with a net worth of at least $5,000,000
CREATE VIEW RichExec(name, address, cert, networth) AS
    SELECT *
    FROM movieexec
    WHERE networth >= 5000000;
```