Universidad Autónoma de Madrid

EDAT

# Exercises

Roberto Marabini Ruiz

# Contents

# 1 Chess Championship

```
Player (player_id, name)
Referee (referee_id, name, birthdate)
Room (room_id, name)
Game (time, white^, black,
      referee_id^, room_id^, winner^, finished ) -- players do not play
                                                 -- simultaneous games
Move (move_id^, piece, origin, target,
      white^, time^)
-- another possible schema  is
-- Game (game_id, white^, black^,
--      referee_id^, room_id^, move_id^, winner^, finished, time )
-- Move(move_id^, piece, origin, target, game_id)
```

## 1.1 Total number of games

```
SELECT COUNT(*)
FROM game;
```

## 1.2 Number of games played by each player (give player_id)

```
CREATE VIEW playerlist AS
    SELECT white AS player_id
    FROM game
    UNION
    SELECT black AS player_id
    FROM game;

SELECT count(*),
       player_id
FROM playerlist
GROUP BY player_id;
```

## 1.3 Number of games played by each player (give player's name)

```
-- playerlist  was defined in the previous exercise

SELECT count(*), name
       player_id
FROM playerlist NJ player
GROUP BY player_id, name;
```

## 1.4 Last played game

```
--time at which the last game was played
CREATE VIEW lastgame AS
SELECT max(time) as time
FROM game;

SELECT white, black, time             -- or game_id
FROM lastgame  NJ game;
```

## 1.5 Players (names) who have played more than 5 games

```
--time at which the last game was played
CREATE VIEW playerIDs AS
SELECT player_id
FROM playerlist     -- defined in a previous query
GROUP BY player_id
HAVING count(*)> 5


SELECT name
FROM playerIDs NJ player;
```

## 1.6 More popular movement

A movement is defined by the triplet (piece, origin, target)

```
-- number of times each movement has been made
CREATE VIEW movements AS
SELECT count (*) as times,  piece, origin, target
FROM move
GROUP BY  piece, origin, target

-- number of times the more popular movement has been done
CREATE VIEW maxMovements AS
SELECT max(times) as times
FROM movements

-- more popular movement/s
SELECT piece, origin, target, times
FROM maxMovements NJ movements
```

## 1.7 Players that have never played against each other

```
-- all possible player pairs
CREATE VIEW playerPairs AS
SELECT p1.player_id as firstPlayer,
       p2.player_id as secondplayer
FROM player p1, player p2
WHERE p1.player_id < p2.player_id


-- players that have played against each  other
CREATE VIEW actualPairs AS
SELECT white as firstPlayer,
       black as secondplayer
```

```
FROM game
WHERE white < black
UNION
SELECT black firstPlayer,
       white as secondplayer
FROM game
WHERE white > black

--played that have never played against each other
SELECT firstPlayer, secondplayer
FROM playerPairs
EXCEPT
SELECT firstPlayer, secondplayer
FROM actualPairs
```

## 1.8 First Movement of each Play

```
-- move_id of the first moves
CREATE VIEW firtMovement AS
SELECT min(move_id) as move_id
FROM move
GROUP BY white, time

SELECT move_id, white,
FROM move NJ firtMovement
```

## 1.9 Most popular first movement

```
-- all possible player pairs
CREATE VIEW firtMovement2 AS
SELECT count(*) as times, piece, origin, target
FROM move NJ firtMovement -- view from previous exercise
GROUP BY piece, origin, target
```

```
--number of times the most popular first movement has been made
CREATE VIEW firtMovementTimes AS
SELECT max(times) as times
FROM firtMovement2;

--recover movement
SELECT piece, origin, target
FROM firtMovementTimes NJ firtMovement2
```

# 2 Social Network - Likes

Given the following schema use SQL to answer the queries

```
MEN   (NameM, age)
WOMEN (NameW, age)
MlikesW (NameM^, NameW^) -- Man NameM likes woman NameW
WlikesM (NameW^, NameM^) -- Woman NameW likes man NameN
MARRIAGE (NameM^, NameW^)
```

## 2.1 Find pairs of men and women that (1) they like each other, (2) are between 30 and 40 years old and (3) are not married to each other

```
-- They like each other
CREATE VIEW likeeachother AS
SELECT m.NameM, m.NameW
FROM MlikesW m, WlikesM w
WHERE m.NameM=w.NameM AND m.NameW=w.NameM;

--Between 30 and 40 years
SELECT m.NameM, m.NameW
FROM likeeachother NJ MEN m NJ WOMEN w
WHERE m.age > 30 and w.age < 40  -- another alternative
 AND  w.age > 30 and w.age < 40  -- age BETWEEN 30 AND 40
EXCEPT -- remove married couples
SELECT m.NameM, m.NameW
FROM  MARRIAGE
```

## 2.2 Married women that do not like her husband

```
-- All Married women
```

```
SELECT NameW
FROM MARRIAGE

EXCEPT

-- Women that like their husband
SELECT ma.NameW
FROM MARRIAGE ma, WlikesM l
WHERE ma.NameM = l.NameM
  AND ma.NameW = l.NameW
```

## 2.3 Men that do not like any woman

```
--All men
SELECT NameM
FROM MEN
EXCEPT
--men that like at least one woman
SELECT NameM
FROM MlikesWn
```

## 2.4 Married women that do not like any married men

```
--All married women
SELECT NameW
FROM MARRIAGE
EXCEPT
--women that like at least one married man
SELECT w.NameM
FROM MARRIAGE m, WlikesM w
WHRE m.NameM = w.NameM
```

# 3 ER Modeling Concepts

## 3.1 Keys

Explain the distinctions among the terms primary key, candidate key, and superkey

A superkey is a set of one or more attributes that uniquely identifies an entity in the entity set. A candidate key is a superkey from which you cannot remove any fields. The primary key is one of the candidate keys that is chosen by the database designer as the principal means of identifying entities within an entity set.

## 3.2 Entities

Explain the difference between a weak and a strong entity set

A strong entity set has a primary key. All tuples in the set are distinguishable by that key. A weak entity set has no primary key unless the PK of the strong entity set on which it depends are included.

## 3.3 Attributes

What is an atomic attribute.

Atomic attributes are attributes that can not be further divided

# 4 E-R diagram for a car-insurance company

Construct an E-R diagram for a car-insurance company whose customers own one or more cars each. Each car has associated with it zero to any number of recorded accidents. The car owner may be different from the car driver (both persons are needed in each accident report).

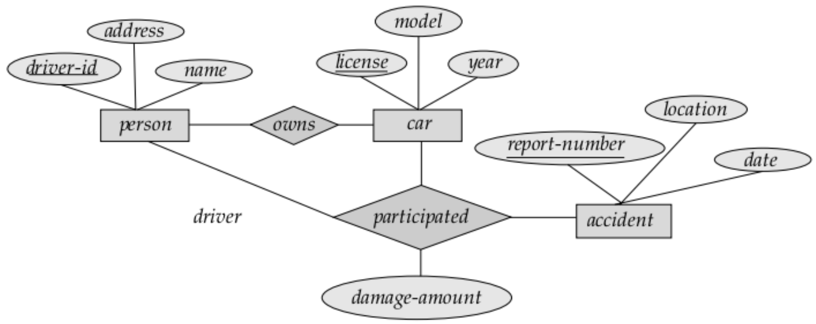



Figure 1: E-R diagram for a car-insurance company

# 5 Arrange Meeting

An application that satisfies the following requirements must be implemented. The application will persist the data using a database. Model the database using an ER diagram.

a. The user chooses the option to arrange a meeting.

b. The system prompts user for the names of attendees.

c. The user types in a list of names.

d. The system checks that the list is valid.

e. The system prompts the user for meeting constraints.

f. The user types in meeting constraints.

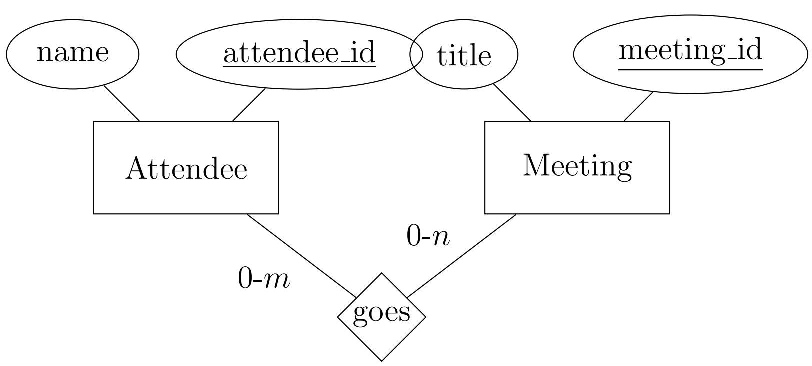
g. The system searches the calendars for a date that satisfies the constraints.

h. The system displays a list of potential dates.

i. The user chooses one of the dates.

j. The system writes the meeting into the calendar.

k. The system emails all the meeting participants informing them of them appointment




PK(goes)=(attendee_id, meeting_id)

# 6 Normalization

## 6.1 Exercise 1

Given a relation schema with one single relation $R(A, B, C, D)$ and a functional dependency $B \rightarrow CD$. Let us assume that $A, B$ is a PK of $R$ then decompose $R$ into two relations $R1$ and $R2$ using the given functional dependency.

```
Answer:

R1(A,B)
R2(B,C,D)
```

## 6.2 Exercise 2

Given a relation $R(A, B, C, D, E)$ as well as some data:

| A | B | C | D | E |
|---|---|---|---|---|
| 1 | 4 | 4 | 3 | 2 |
| 4 | 1 | 2 | 1 | 2 |
| 2 | 5 | 5 | 4 | 3 |
| 1 | 4 | 2 | 1 | 2 |
| 2 | 5 | 4 | 3 | 2 |
| 4 | 1 | 4 | 3 | 2 |
| 3 | 4 | 3 | 3 | 3 |

a. Find at least 3 independent, non-trivial functional dependencies that do not conflict with the given data.

b. Can you be sure, if the functional dependencies found in the previous exercise are the "real" functional dependencies, defined on the schema?

Answer:

$A \rightarrow B$

$C \rightarrow D$

$AC \rightarrow E$
No, without further information there is no way to know if the proposed FD are real. A tuple such as (1, 5, 1, 1, 1) would not satisfy $A \rightarrow B$

## 6.3 Exercise 3

Given following scheme $R(A, B, C, D, E)$. Which normal form is violated in the according schema? What exactly causes the violation?

**Case 1**

$A \rightarrow BC$
$D \rightarrow E$

**Case 2**

$A \rightarrow BCD$
$D \rightarrow E$

Answer:
Case 1: No 2NF. $E$ is not prime and depends on part of the PK $(A, D)$. Case 2: No 3NF. $E$ is not prime and depends on $D$ that is nor candidate key.

## 6.4 Exercise 4

Given a Schema with one single relation $R(A, B, C, D, E.F)$ and a set of functional dependencies

$BC \rightarrow DC$
$B \rightarrow E$
$D \rightarrow EF$
$FC \rightarrow E$
$C \rightarrow A$
$F \rightarrow E$

- find PK
- Find a minimal equivalent set (closure) of dependencies
- Normalize the schema into BCNF.

Answer:
PK = $BC$

closure
$BC \to DC$, $BC \to D$
$B \to E$
$D \to EF$
$C \to A$
$F \to E$

BCNF
$R1(BCD)$
$R2(B, E)$
$R3(CA)$
$R3(DF)$

## 6.5 Exercise 5

Given the relation schema $R = (A, B, C, D, E)$ and the set of functional dependencies:

$A \to BC$
$CD \to E$
$B \to D$
$E \to A$

Decompose the relation $R$ so that the resulting set of relations satisfy Boyce-Codd Normal.

Answer:

PK = $A$ or $E$

$R$ is not Boyce Codd since we have $B \rightarrow D$ and $B$ is not a candidate key

So we decompose $R$ in:

$R1 = ABCE$

$R2 = BD$

The remaining dependencies are:

for $R1$

$A \rightarrow BC$

$E \rightarrow A$

for $R2$

$B \rightarrow D$.

Since $A$ and $E$ are candidate keys $R1$ and $R2$ satisfy Boyce-Codd NF. $PK(R2) = B$

## 6.6 Exercise 6

Suppose you are given the following functional dependencies:

$name \rightarrow address, gender\ address \rightarrow rank\ rank, gender \rightarrow salary$

Then:

a. Give a primary key for the relation $R(name, address, gender, rank, salary)$

b. Normalize the relation r(name, address, gender, rank, salary) to 3rd normal form. Specify the primary keys in the normalized relations.

Answer:

$name \rightarrow address, gender \rightarrow rank \rightarrow rank\ name$ is a primary keys

Since length(PK)=1 $R$ must satisfy first normal form. $R$ does not satisfy 3FN for $rank$. We may break $R$ in $R1 = addres, rank$ and $R2 = (name, address, gender, salary$. $R2$ does not satisfy 3FN since $rank, gender \rightarrow salary$. There we split $R2$ as follows

$R21(name, address, gender)PK = name$
$R22(rank, gender, salary), PK = rank, gender$
$R1(address, rank), PK = address$

# 7 Relational Algebra and Calculus

## 7.1 Exercise 1

Consider the following schema:
$Suppliers(\underline{sid}, sname, address)$
$Parts(\underline{pid}, pname, color)$
$Catalog(\underline{sid}\hat{}, \underline{pid}\hat{}, cost)$

Write the following queries in relational algebra (RA), tuple relational calculus (RC), and SQL:

### 7.1.1 Find the names of suppliers who supply some red part

SQL:

SELECT sname
FROM Parts NJ Catalog NJ Suppliers
WHERE color = 'red'

RA

$$\Pi_{sname}\sigma_{color='red'}Parts \bowtie Catalog \bowtie Suppliers$$

RC

$$\{S.sname \mid Suppliers(S) \wedge (\exists C \exists P$$
$$(Catalog(C) \wedge Parts(P) \wedge$$
$$(P.color =' red') \wedge$$
$$(P.pid = C.pid) \wedge$$
$$(C.sid = S.did)))\}$$

### 7.1.2 Find the sids of suppliers who supply some red or green part

SQL:

SELECT sid FROM Catalog NJ Parts
WHERE color = 'red' OR color = 'green'

RA

$\Pi_{sid}\sigma_{color='red' \vee color='green'} Parts \bowtie Catalog$

RC

$\{C.sid \mid Catalog(C) \wedge (\exists P$
$(Parts(P) \wedge$
$((P.color =' red') \vee (P.color =' green')) \wedge$
$(P.pid = C.pid)$
$)\}$

#### 7.1.3 Find the sids of suppliers who supply some red part and some green part.

SQL_1:

```
SELECT sid
FROM Catalog C1, Catalog C2, Parts P1, Parts P2
WHERE C1.pid = P1.pid AND C2.pid = P2.pid
AND C1.sid = C2.sid AND P1.color='red' AND P2.color='green'
```

SQL_2:

```
SELECT sid FROM Catalog C, Parts P WHERE C.pid = P.pid AND
P.color='red'
INTERSECTS
SELECT sid FROM Catalog C, Parts P WHERE C.pid = P.pid AND
P.color='green'
```

RA

$A \leftarrow \Pi_{sid}\sigma_{color='red'}Parts \bowtie Catalog$
$B \leftarrow \Pi_{sid}\sigma_{color='gree'}Parts \bowtie Catalog$
$A \cap B$

RC_1

$\{C1.sid \mid Catalog(C1) \wedge (\exists P1 \exists P2 \exists C2($
$Parts(P1) \wedge Parts(P2) \wedge Catalog(C2)$
$\wedge(P1.color =' red') \wedge (P2.color =' green')$
$\wedge(P1.pid = C1.pid) \wedge (P2.pid = C2.pid)$
$\wedge(C1.sid = C2.sid)$
$))\}$

RC_2

$\{C1.sid \mid Catalog(C1) \wedge (\exists P1($
$Parts(P1) \wedge (P1.color =' red' \wedge (P1.pid = C1.pid) \wedge (\exists C2($
$Catalog(C2) \wedge (C1.sid = C2.sid) \wedge (exists P2($
$Parts(P2) \wedge (P2.color =' green' \wedge (P2.pid = C2.pid)$
$))))))))\}$

### 7.1.4 Find the sids of suppliers who supply every part

SQL_1:

```
CREATE view parts_per_supplier as
SELECT sid, count(*) as number
FROM Catalog
GROUP BY sid


CREATE view parts_number as
SELECT count(*) as number
FROM Part
```

SELECT sid
FROM parts_per_supplier NJ parts_number

SQL_2:

```
SELECT C1.sid
FROM Catalog C1
WHERE NOT EXISTS (SELECT P.pid
                  FROM Parts P
                  WHERE NOT EXISTS (SELECT C2.sid
                                    FROM Catalog C2
                                    WHERE C2.sid = C1.sid
                                      AND C2.pid = P.pid))
```

RA

$\Pi_{sid,pid}Catalog/\Pi_{pid}Parts$

RC

$\{C1.sid \mid Catalog(C1) \wedge (\forall P$
$(Parts(P) \wedge (\exists C2$
$(Catalog(C2) \wedge (C1.sid = C2.sid)$
$\wedge(P.pid = C2.pid)$
$)))\}$

#### 7.1.5 Find pairs of sids such that the supplier with the first sid charges more for some part than the supplier with the second sid

SQL:

SELECT c1.sid, c2.sid
FROM Catalog c1, Catalog c2

WHERE c1.cost >= c2.cost
AND c1.sid <> c2.sid
AND c1.pid = c2.pid

RA

$C1 \leftarrow Catalog$
$C2 \leftarrow Catalog$
$\Pi_{C1.sid,C2.sid}\sigma_{C1.pid=C2.pid \wedge C1.sid \neq C2.sid \wedge C1.cost>=C2.cost} SC1 \times C2$

RC_1

$\{C1.sid \mid Catalog(C1) \wedge (\exists C2($
$Catalog(C2) \wedge (C1.sid \neq C2.sid)$
$\wedge(C1.pid = C2.pid)$
$\wedge(C1.cost >= C2.cost)$
$))\}$

>= cover the "patological" case in which all parts have the same cost.

#### 7.1.6 Find the pids of the most expensive parts supplied by supplier named Yosemite Sham (may be 2 parts have the same price)

SQL_1:

```
CREATE VIEW maxCost AS
SELECT max(cost) as cost
FROM Suppliers NJ Parts NJ Catalog
WHERE sname = 'Yosemite Sham';

SELECT *
FROM maxCost NJ Catalog;
```

SQL_2:

```
SELECT C.pid
FROM   Catalog C NJ Suppliers S
WHERE  S.sname = 'Yosemite Sham'
  AND  C.cost >= ALL (Select C2.cost
                      FROM Catalog C2 NJ Suppliers S2
                      WHERE S2.sname = 'Yosemite Sham'
                      )
```

RA:

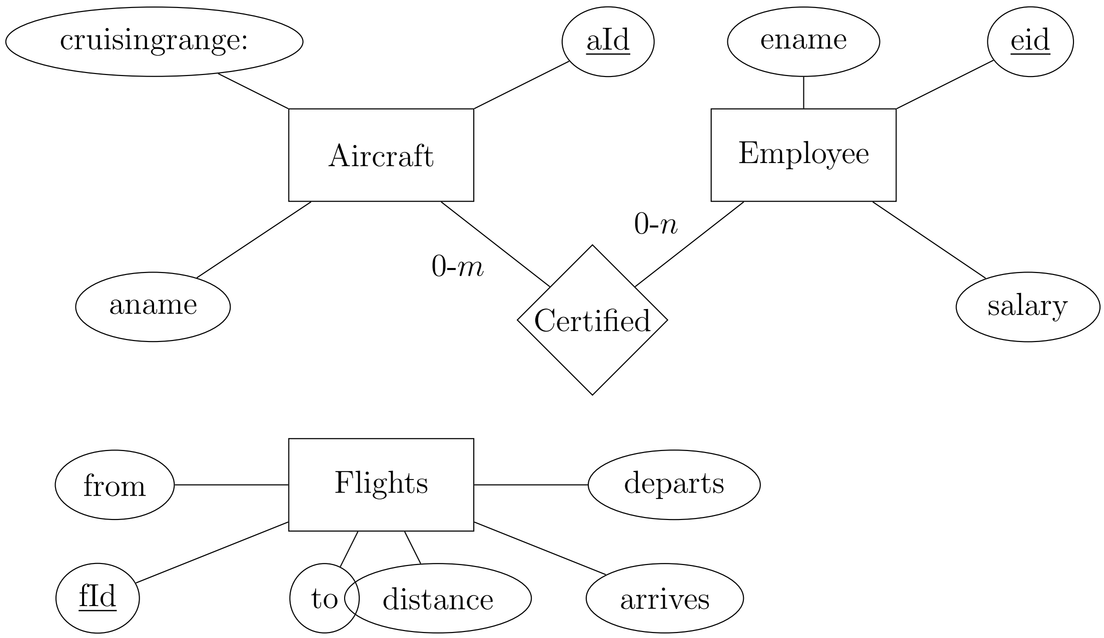
$maxCost \leftarrow G_{MAX(cost)}\sigma_{sname='YosemiteSham'}Suppliers \bowtie Parts \bowtie Catalog$
$\rho(maxCost(cost))maxCost$
$maxCost \bowtie Catalog$

RC_1

$\{C.pid \mid Catalog(C) \wedge (\exists S($
$Supplier(S) \wedge (S.name =' YosemiteSham') \wedge (S.sid = C.sid)$
$\wedge(NOT \exists C2(Catalog(C2)$
$\wedge(C2.sid = S.sid) \wedge (C2.cost > C1.cost)$
$))))\}$

## 7.2 Exercise 2

### 7.2.1 Convert the following E-R diagram to a relational model. Identify explicitly primary and foreign keys




Flights(<u>flno</u>, from, to, distance, departs, arrives)
Aircraft(<u>aid</u>, aname, cruisingrange)
Employees(<u>eid</u>, ename, salary)
Certified(<u>eid</u>∧, <u>aid</u>∧)

Write the following queries in relational algebra (RA), tuple relational calculus (RC), and SQL:

### 7.2.2 Identify the flights that can be piloted by every pilot whose salary is more than $100,000. Note: use 'distance' and 'cruisingrange'

SQL:

```
SELECT FId
FROM Aircraft NJ Certified C NJ  Employees E, Flights F
WHERE distance < cruisingrange
  AND salary > 100,000
```

RA

$\Pi_{fId}\sigma_{distance<cruisingrange \land salary>100,000} Aircraft \bowtie Certified \bowtie Employees \times Flights$

RC

$\{F.pid \mid Flights(F) \land (\exists A($
$Aircrafts(A) \land (F.distance < A.cruisingrange)$
$\exists C(Certified(C) \land (C.aid = A.aid)$
$\exists E(employee(E) \land (E.eid = C.eid) \land (E.salary > 100000)$
$))))\}$

## 7.3 Find the eids of employees who are certified for exactly two aircrafts.

SQL:

SELECT eid FROM Certified GROUP BY eids HAVING count(*) = 2

RA

$\rho(A(number)) \leftarrow eid G_{count(*)} Certified$
$\Pi_{eid}\sigma_{number=2} A$

RC

$\{C1.eid \mid Certified(C1) \land (\exists C2($
$Certified(C2) \land (C1.eid = C2.eid) \land (C1.aid \neq C2.aid)$
$NOT \exists C3(Certified(C3) \land (C1.eid = C3.eid) \land ((C3.aid \neq C1.aid) \land (C3.aid \neq C2.aid))$
$)))\}$

# 8 Functional Dependencies

## 8.1 Exercise 1

The database of a company that buys and shells shares in the stock market contains a relation $R(b, o, a, q, c, d)$ with the following attributes:
b: broker
o: broker's office
a: share
q: number of shares
c: client
d: dividend paid by share

We know that the following functional dependencies are satisfied:
$a \to d$
$c \to b$
$c, a \to q$
$b \to o$

a. Provide a possible primary key for $R$.

b. Does $R$ satisfy the 2NF? If it does not decompose $R$ so it does satisfy it.

c. Does $R$ satisfy the 3NF? If it does not decompose $R$ so it does satisfy it.

Let us assume that $R$ is decomposed in the following two relations:
$R1(c, a, q, d)$
$R2(c, b, o)$

d) Does $R1$ and $R2$ satisfy the 3NF? If it does not decompose $R$ so it does satisfy it.

e) Describe a redundancy that is present in the original relation $R$ but not in de decomposition obtained in exercise (c).

ANSWERS:

a) possible PK $\{a, c\}$

b) No 2NF $d$ depends only on $a$

c) NO 3NF since R does not satisfy 2NF. $r1(c, a, q)$, $r2(a, d)$, $r3(c, b)$, $r4(b, o)$

d) No, in $R2$, $o$ (non-prime attribute) depends only on $b$ (no candidate key)

e) If a brokers has several clients his office information appears many times in $R$

## 8.2 Exercise 2

Answer to the following questions. Please, explain your answers:

A) Given a relation $R1(A, B, C, D, E, F, G)$ with primary key $B$ and the following functional dependencies:

$DF1 : B \to A, C, D, E, F, G$

$DF2 : E \to F, G$

What is the highest normal form satisfied by $R1$?

B) Given a relation $R2(A, B, C, D)$ with primary key $\{A, B\}$ and the following functional dependencies:

$DF1 : A, B \to C, D$

$DF2 : C \to B$

What is the highest normal form satisfied by $R2$?

Decompose $R2$ so it does satisfy Boyce-Codd normal form. Provide the functional dependencies for the decomposition.

ANSWERS:

A) $R1$ satisfies the 1FN since (a) $R1$ has a primary key, (b) attributes are atomic and are not multi-valued. $R1$ satisfies the 2NF because all non-prime attributes fully depend on the primary Key. $R1$ does not satisfy the 3NF because the non-prime attribute $G$ depend on $E$ that is not a candidate key. $R1a(B, A, C, D, E)$ and $R1b(E, F, G)$ satisfy the 3FN.

B) $R2$ satisfies the 1FN since (a) $R2$ has a primary key, (b) attributes are atomic and are not multi-valued. $R2$ satisfies the 2NF because all non-prime attributes fully depend on the primary Key. $R2$ satisfies the 3NF because the is non-prime attribute do that depends on another attribute that are not part of a candidate key. $R2$ does not satisfy the BCNF because there is one attribute $B$ that depends on $C$ and $C$ is neither a primary key or a candidate key.

Decomposition: $R2a(A, C, D)$ and $R2b(C, B)$ with dependencies $AC \to D$ and $C \to B$ satisfy the BCNF.