January 24, 2018

Tutorial 2

Exercise 2.3.2

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

a) schema for relation Classes
give type of each variable

```
CREATE TABLE Classes(
        class VARCHAR(10) PRIMARY KEY,
        type CHAR(2),
        country VARCHAR(30),
        numGuns INTEGER,
        bore INTEGER,
        displacement INTEGER
);
```

Union:

2 tables:

stud_info		
st_id	name	courseno
1	sharan	353
2	ABC	249

student_info		
st_id	name	courseno
2	ABC	249

STUD_INFO union STUDENT_INFO (**union**)

stud_info		
st_id	name	courseno
1	sharan	353
2	ABC	249

STUD_INFO **difference** STUDENT_INFO

stud_info		
st_id	name	courseno
1	sharan	353

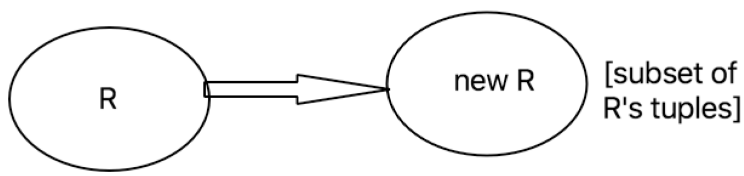
Projection:

stud_id	first_name	last_name	courseno	credits
1	sharan	ABC	352	4
2	xyz	DEC	249	4s

π first_name, last_name,, student_info

first	last
sharan	ABC
XYZ	DEC

Selection



January 31, 2018

□ entity

○ attributes

◇ Relations

— m : m

→ m : 1

↔ 1 : 1

→ referential attribute

Tutorial 3 (Faubourg)

Schema:

Student(SID, SName, dob, iNTERNATIONAL, CourseID, GPA)

Courses(CourseID, CName, Instructor_ID, Credits, Term)

Instructors(InsID, IName, Course_ID)

CourseID	**Cname**	**Instructor_ID**	**Credits**	**Term**
COMP101	DBMS	1001	3	WINTER
COMP201	OOPS	1001	2	WINTER
COMP301	DATA STRC	1001	3	FALL
COMP401	OS	1003	4	SUMMER
COMP501	Compiler Des	1002	3	WINTER
COMP601	COMP Network	1002	3	Fall

Q1) Give winter course names with at least 3 credits

σ : select   π : projection

One way to answer:

$\pi_{\text{Cname}} (\sigma_{\text{Credits>=3 AND term = 'winter'}} (\text{Courses}))$

Another way using intersection:

R1 := $\sigma_{\text{Credits>= 3}}$ (Courses)

R2:= $\sigma_{\text{term ='winter'}}$ (Courses)

Ans:= π_{CName}(R1 $\cup$ R2)

Q2) Give the student names and IDs of international students having GPA>3

SID	***SName***	***International***	***CourseID***	***GPA***
101	Harry	N	comp101	3.75
201	Ron	N	comp101	3.50
301	Hermione	Y	comp101	4.00
401	Malfoy	N	COMP201	3.00
501	Neville	Y	COMP201	2.50
101	Harry	N	COMP301	2.70
201	Ron	N	COMP301	2.70
301	Hermione	Y	COMP401	3.00

Answer:

R1:= $\sigma_{\text{GPA>3}}$ (Students)

R2:= $\sigma_{\text{International ='Y'}}$ (Students)

Ans:= $\pi_{\text{Cname,IDs}}$ (Students)

Q3) Which students are taking the courses instructed by Prof Tom or Prob Bob?
(same students list as Q2)

InsID	***IName***	***CourseID***
1001	Tom	COMP101
1001	tom	COMP201
1001	Tom	COMP301
1002	Bob	COMP401
1003	George	COMP501

$\Gamma\prod$

R1:= $\sigma_{\text{CourseID, IName = 'Tom' OR IName = 'Bob'}}$ (Instructors)

R2:= π_{Sname} (R1) (Students)

If we use join:

R1 := Students ⋈ Instructors

It will associate the two table together by using the courseID as unique key

R1 := Students ⋈ Instructors

R2:= π_{SName} ($\sigma_{\text{IName ='Bob' Or IName='Tom}}$ (R1))

Q4) Which students are taking the courses instructed by Prof Tom and Prob Bob?

R1 := Students ⋈ Instructors

R2:= π_{SName} ($\sigma_{\text{IName ='Bob' } \textbf{AND} \text{ IName='Tom}}$ (R1))

It will not work because IName cannot get 2 different names. (Like a variable in Java can't have 2 meanings)

R1:= Students ⋈ Instructors

R2:= $\sigma_{\text{IName = 'Bob'}}$ (R1)

R3:= $\sigma_{\text{IName = 'Tom'}}$ (R1)

R4:= π_{CName} ∩ (R2 AND R3)

Q3) Do it in SQL

SELECT SName

FROM Students, Instructors

WHERE Students.CourseID = Instructors.CourseID **AND** (IName = 'Tom' **AND** IName='Bob')

In this question you need to use intersection

(voir au-dessus)

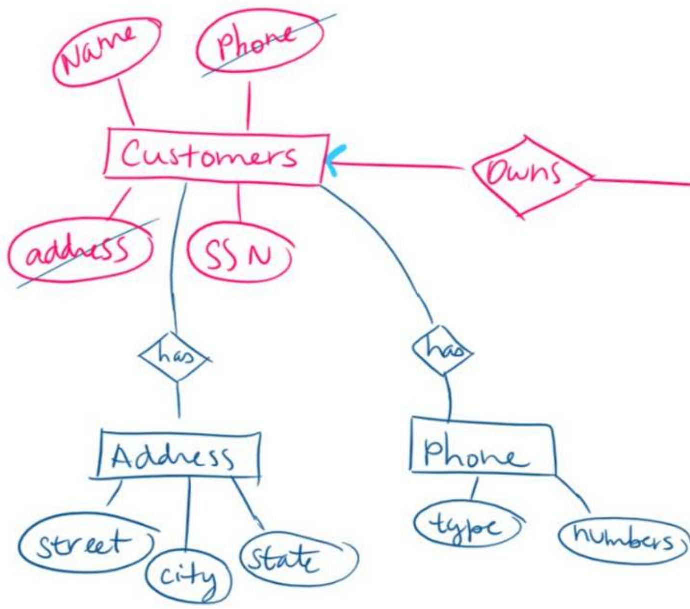
February 7, 2018

Exercise 4.1.1

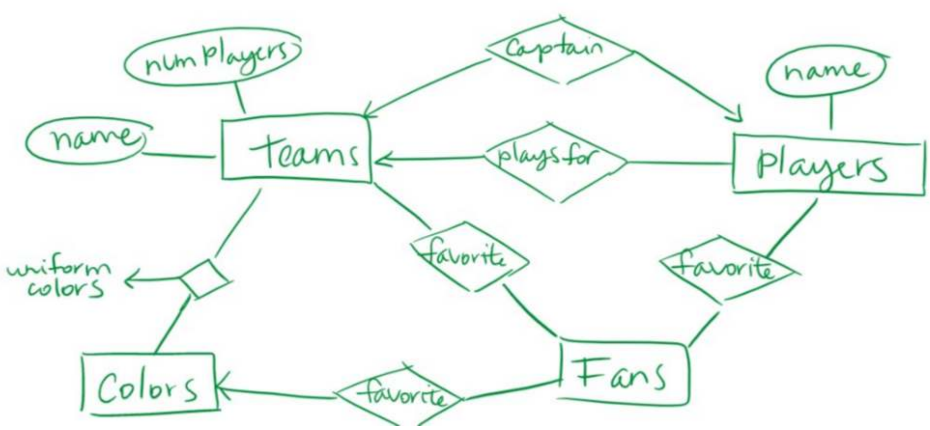
Q1: Design a DB for a bank, including information about customers and their accounts. information about customer includes their name, address, phone and social security number. Accounts have numbers, types (savings, checking) and balances. Also record the customer(s) who own an account.

Draw the E/R diagram for this DB. Be sure to include the arrows where appropriate, to indicate the multiplicity of a relationship.

Exercise 4.1 .1 .2 .3 .2c)



Exercise 4.3 ?



Q5: Given basic ER and relational models, which of this is incorrect

A. Attribute of entity can have more than 1 value

B. Attribute of entity can be composite

C. In a row of relational table, an attribute can have more than 1 value. → because you can have many values in one column but for a row, one value for a row

D. In a row of relational table, an attribute can have exactly 1 value or NULL value. → true

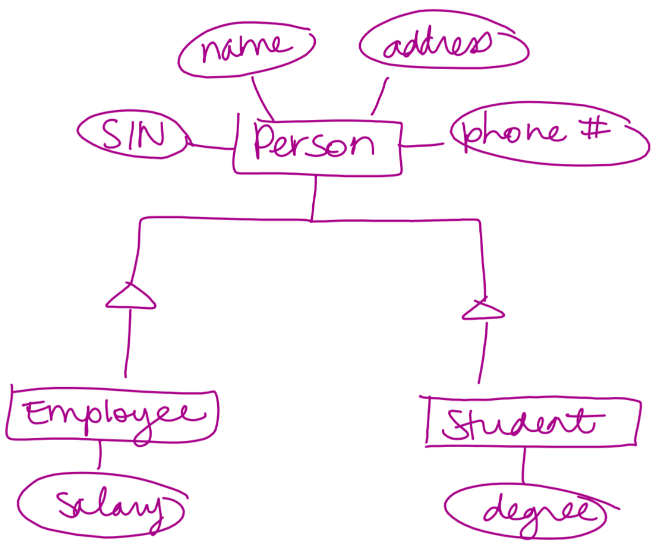
February 14, 2018

Question 6:

E/R diagram recording name, addresses, phone numbers of two type of people

Employee along with their salaries

Students along with their degree



Schema for this diagram

Person (SIN, phone, name, address)

Employee (SIN, salary)

Student (SIN, degree)

If constraint: person either Employee or Student

solution ①: only person table, will have null values

Person (SIN, phone, name, address, type, degree, salary)

solution ②: 2 tables

Employee (SIN, phone, name, address, salary)

Student (SIN, phone, name, address, degree)

Question 7 :

Create schema (S) for this E/R DIAGRAM

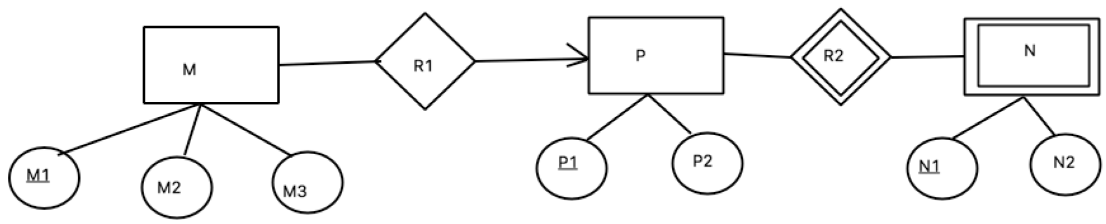
Person(SIN, address, phone, name, type of degree, salary)

Employee(SIN, salary)

Student(SIN, degree)

1 table for Person and 1 for employee/student if they are joint.

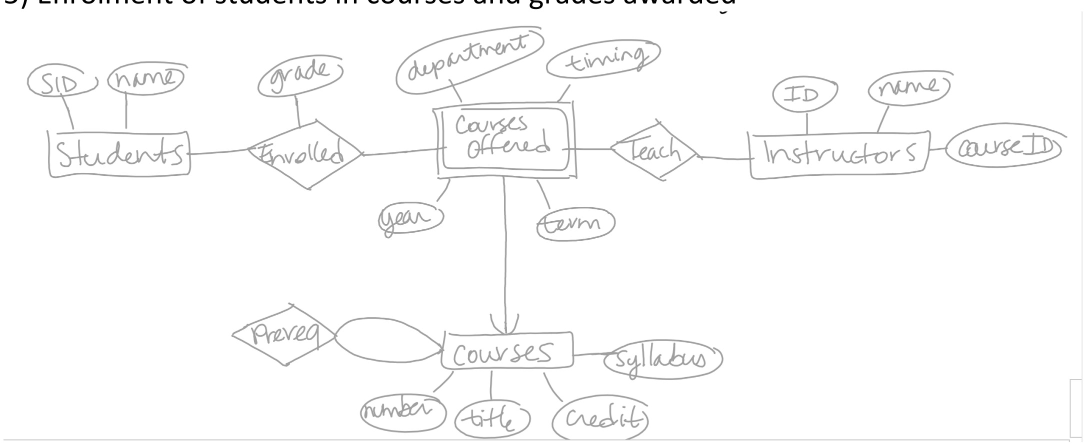
Q: How many tables?



3 tables because even the weak entity has its own attributes
M(M1,M2,M3)
P(P1,P2)
N(P1,N1,N2)

Question 8: E/R diagram for University Register

1) Courses, including number, title, credit, syllabus, prerequisites
2) Students, SID, name, courseID
3) Instructors, ID, name, course ID
4) Courses offered, course title, year, term, department, instructor, time
5) Enrolment of students in courses and grades awarded



February 28, 2018

Question 1: Consider R(X,Y,Z,W) with FDs F={W $\rightarrow$ Y, X $\rightarrow$ Z}

Prove or disprove F |= WX $\rightarrow$ Y

W $\rightarrow$ Y (given)
WX $\rightarrow$ YX (augmentation)
WX $\rightarrow$ X (decomposition)

Question 2: Consider R(X,Y,Z,W) with FDs F={X $\rightarrow$ Y, X $\rightarrow$ W, WY $\rightarrow$ Z}

Prove or disprove F |= X $\rightarrow$ Z

$X \rightarrow Y$, $X \rightarrow W$ (given)
$WY \rightarrow Z$ (union)
$X \rightarrow Z$ (transitivity)

Question 3: Consider R(X,Y,Z,W) with FDs F={$W \rightarrow Y$, $X \rightarrow Z$}
Prove or disprove F |= $WX \rightarrow Y$ with closure test
WX^+= {W, X}
WX^+= {W, X, Y} | $W \rightarrow Y$
WX^+= {W, X, Y, Z} | $X \rightarrow Z$

Question 4: Consider R(X,Y,Z,W) with FDs F={$X \rightarrow Y$, $X \rightarrow W$, $WY \rightarrow Z$}
Prove or disprove F |= $X \rightarrow Z$ with closure test Closure Test

Closure Test	Rules
X+ = {X}	$X \rightarrow W$, $X \rightarrow Y$ (given)
X+ = {X, w} $X \rightarrow W$	$X \rightarrow WY$ (union)
X+ = {X,W, Y} $X \rightarrow Y$	$WY \rightarrow Z$ (given)
X+ = {X,W,Y,Z} $X \rightarrow Z$	$X \rightarrow Z$ (transitivity)
It is true.	

Question 6: Consider a relation with schema R(A,B,C,D) and FD = {$AB \rightarrow C$, $C \rightarrow D$, $D \rightarrow A$}
a) What are all nontrivial FDs from current FD?

** Start with calculating all transitive closures*

A^+ = {A}
B^+ = {B}
C^+ = {C, D, A}
$C \rightarrow A$ non trivial
D^+ = {D, A}

AB^+ = {A, B, C, D}
$AB \rightarrow B$
AC^+ = {A, C, D}
$AC \rightarrow D$
AD^+ = {A, D}
BC^+ = {B, C, D, A}
$BC \rightarrow D$, $BC \rightarrow A$
BD^+ = {B, D, A, C}
$BD \rightarrow A$, $BD \rightarrow C$
CD^+ = {C, D, A}
$CD \rightarrow A$

ABC^+ = {A,B,C,D}
$ABC \rightarrow D$
ACD^+ = {A,C,D}
BCD^+ = {B,C,D,A}
$BCD \rightarrow A$
ABD^+ = {A,B,D,C}
$ABD \rightarrow C$
$ABCD^+$ = {A,B,C,D}

b) what are all candidate keys of R?
Any FDs that give all the transitive closure that are A,B,C,D can be candidate keys. BUT they need to be minimal which can only includes: (in blue) AB,BC,BD.

c) What are all super keys of R that are not candidate keys?
ABC,BCD,ABD,ABCD (in yellow)

Question 7: R=(A,B,C,D,E,F) be relation scheme with dependencies: $C \rightarrow F$, $E \rightarrow A$, $EC \rightarrow D$, $A \rightarrow B$. Which is a key for R?
a) CD **b) EC** c)AE d)AC
CD+ ={C,D, F}
EC+ = {E,C, A, F, D, B}

Question 8: given following relation instance:

X	Y	Z
1	4	2
1	5	3
1	6	3
3	2	2

Which is true?

a) XY→Z and Z→Y
b) YZ→X and Y→Z
c) YZ→X and X→Z
d) XZ→Y and Y→X

Question 8: given following relation instance:

A	B	C
1	1	1
1	1	0
2	3	2
2	3	2

a) A→ B and B → C: incorrect
b) A→ B and B not → C: correct

Question 13: X(P,Q,R,S,T,U)
F = {
{P,R]→ {S,T}
{P,S,U]→{Q,R}
}
Which is trivial functional dependency in F+ is closure F?

a) {P, R] → {S,T}
b) {P, R} → {R,T}
c) {P, S} → {S}
d) {P, S, U} → {Q}

March 7, 2018

Functional dependencies

Question 1:

R=(A,B,C,D,E,H) FD {A→B, BC→D, E→C, D→A}. What are the candidate keys of R?

a. AE, BE
B. AE,BE,DE
C. AEH, BEH, BCH
D. AEH,BEH,DEH

Solution:

AE+ = A,E,C,B,D not H included
BE+= B,E,C,D,A not H included
DE+= D,E,C,A,B not H included

Technique:

we have all these attributes: A B C D E H
by looking at the FD

A	B	C	D	E	H
x	x	x	x	ESSENTIAL	ESSENTIAL

Question 2:

R has attributes A,B,C,D,E,F,G,H FD {CH→G, A→BC, B→CFH, E→AM F→EG}. What are the candidate keys of R?

a. 3
b. 4
c. 5
d. 6

keys candidate: DA, BD, DE, DF

A	B	C	D	E	F	G	H
x	x	x		x	x	x	x

DA^+ = {D,A,B,C,F,H,E,G}
DB^+ = {D,A,B,C,F,H,E,G}
DE^+ = {D,E,A,B,C,F,H,G}
DF^+= {D,E,A,B,C,F,H,G}
DG^+= {D,E,A,B,C,F,H,G}
DC^+={}

Question 3:
R(A,B,C,D) FD{A→B, B→C, C→D}. Which functional dependencies will be projected onto relation S(A,C,D)

1) get closure of all attributes:

A+ = ABCD
C+ = CD
D+ = D
AC+ = ACDB
AD+ = ADB
CD+ = CD
ACD+ = {A, B, C, D}

2) projection

A → C
AC→ D
C → D
AD→ C

3)

C→ D, AD→ C, A→C, AC→D

Question 4:
R(A,B,C,D) FD{AB→C, C→D, D→A}. R is decomposed into R1(A,B,C) and R2(C,D). Check whether decomposition is dependency preserving or not.

A+ = A
B+=B
C+= C,D,A
AB+= A,B,C, D
BC+= B,C,D,A
AC+=A,C,D
D+=D,A
CD+= C,D,A

AB→C, C→A, D→A

Q5: R(A,B,C,D) FD{A→B, B→C, C→D, D→B}. The decomposition of R into (A,B), (B,C), (B,D)

A) gives lossless join, and is dependency preserving
B) gives lossless join, and is not dependency preserving
C) gives not lossless join, and is dependency preserving
D) gives not lossless join, and is not dependency preserving

R1(A,B) A+=A,B,C,D
B+= B,C,D
R2(B,C) B+= B,C,D
C+= C,D,B
R3(B,D) B+= B,C,D
D+= B,C,D

It preserves dependency preserving as you can obtain all FD with R1 U R2 U R3.
Is it lossless?
R1 intersection R2 = B^+
R2 intersection R3 = B^+
IT IS LOSSLESS join.

March 14, 2018

Find canonical form/minimal basis
1) Simplify RHS of each FD
2) Check LHS for redundancy
3) Find redundancy in FDs and remove

Check what type of normal form is it?
1NF: normal relations (most basic form) (automatically)
2NF: (does **not** have partial dependencies)
partial dependency: AB candidate key it means you can have AB→C and not B→D All attributes should be derived by candidate key or superkey.
How to check it? A→B so A should be Candidate key/superkey and B non key attribute
3NF: does not have transitivity AB→C,C→D any non-key attribute cannot derive any non-key attribute.
BCNF: AB→C,C→B non-key attribute derived a key attribute. (LHS superkey or candidate key)

Question 1: Canonical Form/Minimal Basis
R(X,Y,N,Z) and FD = {X→W, WZ→XY, Y→WZX}
Steps:

1) Simplify RHS of each FD
FD = {X→W, WZ→X, WZ→Y, Y→W, Y→Z, Y→X}
2) Check LHS for redundancy
FD = {X→W, Y→W, Y→Z, Y→X}
WZ→X, WZ→Y need to check enclosure can give us the same dependency of attribute
W+ = W and Z+= Z since we cannot derive X or Y so we can say that these are in simple form.
3) Find redundancy and remove
X→W FD = {WZ→X, WZ→Y, Y→W, Y→Z, Y→X}
X^+= (X) This cannot be derive if we remove this dependency
WZ→X, FD = {X→W, WZ→Y, Y→W, Y→Z, Y→X}
WZ+=(WZYX) so this is redundant
WZ→Y FD = {X→W, WZ→X, Y→W, Y→Z, Y→X}
WZ+=(WZX) This cannot be derive if we remove this dependency

Y→W FD = {X→W, WZ→X, WZ→Y, Y→Z, Y→X}

Y+= (Y,Z,X,W) so this is redundant

Y→Z FD{X→W, WZ→X, WZ→Y, Y→W, Y→X}

Y+= (Y,W,X) This cannot be derive if we remove this dependency

Y→X FD{X→W, WZ→X, WZ→Y, Y→W, Y→Z }

Y+= (Y,Z) This cannot be derive if we remove this dependency

FD={X→W, WZ→Y, Y→Z, Y→X}

So minimal FD is: FD={X→W, WZ→Y, Y→ZX}

Question 2: Normal form

R(A,B,C,D,E,F,G,H,I)

FD={AB→C, BD→EF, AD→GH, A→I}

a) Find candidate keys

A	B	C	D	E	F	G	H	I
KEY	KEY	x	KEY	x	x	x	x	x

CANDIDATE KEY: A,B,D

so any superkey should have ABD

BNCF: No because no superkey

3NF: No because AB→C 1)AB is not a candidate key or superkey. or 2) C should be a subset of AB (in their closure)

2NF: AB→C partial dependency

1NF: YES.

Question3:: Normal form

R(A,B,C,D,E)

FD={AB→CD, D→A, BC→DE}

1) find candidate keys

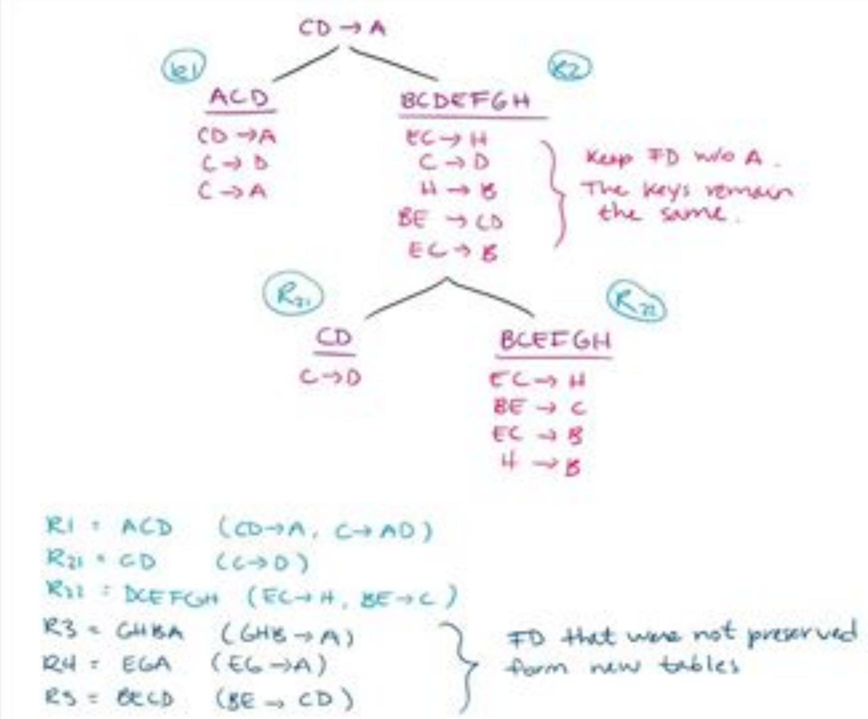
A	B	C	D	E
x	KEY	x	x	x

AB+ = A,B,C,D,E

BC+= B,C,D,E,A,

D+=D,A

keys: ABCD, AB, BC, BD



2)

	BCNF	3NF
AB→CD	Yes	Yes
D→A	No	Yes
BC→DE		Yes

Question4:: Normal form

R(A,B,C,D,E,F)

ABC→ D
ABD→E
CD→ F
CDF→ B
BF→ D

A	B	C	D	E	F
	x		x	x	x

ABC+= a,b,c,d,f,e
ACD+= a,c,d,c,b,f,e
ACE+=A,C,E
ACF+=A,C,F
ABD+= a,b,d,e
CD+= c,d,f,b,d
CDF+= c,d,f,b,d
BF+= b,f,d
candidate keys: ABC, ACD

	BCNF	3NF	2NF	1NF
ABC→ D	Yes	Yes	Yes	Yes
ABD→E	No	No	No	Yes
CD→ F				Yes
CDF→ B				Yes
BF→ D				Yes

ABD→E where ABD is not a superkey or candidate key
1NF

March 21, 2018

<u>**normal forms**</u>

Question 1:

R(,A,B,C,D,E,F)
FD= AB→C, DC→ AE, E→F
Keys: ABD, BCD

	BCNF	3NF	2NF	1NF
AB→C	No	Yes		
DC→ AE		No	No	Yes
E→F				Yes

AB→C is 3NF because C is part of the key candidate
1NF

Question 2:

R(A,B,C,D,E)
BC→ADE, D→B
Keys: BC, CD

	BCNF	3NF	2NF	1NF
BC→ADE	Yes	Yes	Yes	Yes
D→B	No	Yes	Yes	Yes

3NF

Question 3:

R(A,B,C,D)
A→B, A→C, C→BD
Keys: A,B,C,D

	BCNF	3NF	2NF	1NF
A→B	Yes	YES	Yes	YES
A→C	YES	Yes	YES	YES
C→BD	NO	NO	YES	YES

2NF

Question 4:
R(A,B,C,D,E,F)
AB→CD, CD→EF, BC→DEF, D→B, CE→F
Keys: AB,AD

	BCNF	3NF	2NF	1NF
AB→CD	yes			
CD→EF	no	no	yes	
BC→DEF		no	yes	
D→B			yes	
CE→F			yes	

2NF

Question 5:
R(A,B,C)
A→B B→C C→A
keys: ABC
It's **BCNF**

<u>Decomposition</u>
Question 6:
R(A,B,C,D,E,F,G,H)
F= {CD→A, EC→H, GHB→AB, C→D, EG→AM H→B, BE→CD, EC→B}
KEYS: BEFG, CEFG, EFGH
CD→A decomposed into
- ACD (CD→A, C→AD) and
- BCDEGH (EC→H, C→D, BE→CD EC→B, H→B)

March 28, 2018
Question 1: SQL & relational algebra

Treatment(disease, medication)
Doctor(name, specification)
Treated(doct_Name, patient_Name, date, procedure, diagnostic)
Procedure(consultations, intervention)
Intervention (surgery)

A) Give names of doctors not suffering from any disease
<u>**Relational algebra:**</u>
$\rho_{doctpatients}(\pi_{name}(\text{Doctors} \bowtie_{doctor.name= patient.Name} \text{Treated}))$
$\pi_{name}(\text{Doctors}) - \text{doctpatients})$

SQL:

```
SELECT name
FROM Doctor, Treated
WHERE name NOT IN  (SELECT patient_name FROM Treated);

SELECT name
FROM Doctor d
WHERE exists
(
        SELECT patient_Name
        FROM treated t
        WHERE t.patient_name = d.name
)
```

B) Give names of patients who had operation done by a doctor who has HIV

Relational algebra:

$\rho_{DHIV}(\pi_{name}(\text{Doctors}) - \pi_{name}(\sigma_{diagnostic <> HIV}(\text{Treated}))$

$\pi_{pnames}(\text{Treated} \bowtie_{docnames = name} \text{DHIV})$

$\pi\ \text{pname}((\sigma_{docname = name}(\text{treated}\ \ x\ \text{DHIV}))$

SQL:

```
SELECT patient_name
FROM Treated
WHERE doc_name IN(
        SELECT name
        FROM Doctor d, Treated t
        WHERE d.name = patient_name AND diagnostic = HIV
)
```

Question 2

A) Increase the price of beers manufactured by Mary by 10%

Beer(name, manuf)
Sells(bar, beer, price)

```
UPDATE Sells
SET price = price*1.1
WHERE beer IN (
        SELECT name
        FROM Beer
        WHERE manuf = 'Mary'
)
```

Format
Update t1, t2
Set _______
Where ________

```
UPDATE Sells, Beers
SET price = price*1.1
WHERE beer = name AND manuf = 'Mary'
```

Two constraints to add:

1. Relation Sells shouldn't have any beer that is not in Beers
2. Prices should be <= 5 $

What type of constraints are these?
attribute, tuple, assertions → They are **attribute constraints**

```
CREATE TABLE Sells(
        bar  Char(10)
        beer Char(10) REFERENCES Beer(name)
        price REAL/FLOAT CHECK(price <=5)
)
```

Or can be written as

```
CREATE TABLE Sells(
        bar  Char(10)
        beer Char(10) CHECK(Beer IN (
                                        SELECT DISTINCT name
                                        FROM Beer
                                    )
        price REAL/FLOAT CHECK(price <=5)
)
```

3. Only 'Mary' can sell beer > 5$

```
CREATE TABLE Sells(
        bar  Char(10)
        beer Char(10) REFERENCES Beer(name)
        price REAL CHECK(price <=5 OR Beer IN (
                                                SELECT DISTINCT name
                                                FROM Beer
                                                WHERE manuf = 'Mary'
                                                )
```

April 4, 2018

Format:
CREATE TRIGGGER < trigger NAME >
(After/Before) (update/insert/delete) on (Table anme/attribute)
REFERENCING {Old ROW as oldrow}
For each
< Trigger action>

Question 1:
Employees(empID, dept, name, salary)
Find employees getting higher salary than anyone in dept='5'

ID	Dept	name	Salary
101	1	a	2000
102	2	b	4000
…	…	…	…

SELECT empID
FROM Employee
WHERE salary > **ANY** (
SELECT salary **FROM** Employees **WHERE** dept ='5'
);

Question 2:
Projects(project#, pjname, city)
Parts(supplier#, part#, project#, qty)
Express constraint: No project can use m ore than 100 units of part#='P65'
CREATE ASSERTION PartQty **CHECK**
NOT EXIST(
SELECT project#, **SUM**(qty) total
FROM Parts
WHERE part#='P65' **AND** Project.project#= Parts.project#
HAVING total > 100
GROUP BY project#);

Question 3:
Trigger to add any bar that raises to price more than 1$ to relation Ripoff
Sells(bar, price)
Ripoff(bar)

CREATE TRIGGGER Price
AFTER UPDATE of price **ON** Sells
REFERENCING OLD **AS** oldval
NEW **AS** newval
FOR EACH
WHEN(newval.price > oldval price+1)
INSERT INTO Ripoff **values**(oldval.bar)

Question 4:
Trigger to keep track of no of employees in the company
Employee(emplid, dept, name, salary)
Stats(#emp, #prod, revenue)
Employee hired #emplID = #emplID+1 INSERT
Employee leaves #emplID = #emplID-1 DELETE

CREATE TRIGGGER newEmp
After INSERT ON Employee
FOR EACH ROW
Update Stats **SET** #emp=#emp+1

CREATE TRIGGGER empLeave
After DELETE **ON** Employee
FOR EACH ROW
Update Stats **SET** #emp=#emp-1

Question 5:
MovieStar(name, add, gender, dob)
MovieExec(name, add, cert#, networth)
Create a view ExecStar with movie exdecutives that are also stars

CREATE VIEW ExecStar **AS**
Select s.name, s.add, s.dob, s.gender, e.cert#, e.networth
FROM MovieStar s, movieExec e
WHERE s.name = e.name **AND** s.add = e.add

April 11, 2018

SQL, Datalog, Views

Updatable views:
- no join (multiple relations are not allowed)
- not contain group by, aggregate (count, sum,…), distinct
- single relation (contain primary key, and all the not null constraints attributes that don't have a default value)

Question 1:

Treatment(disease, medication)

Query is to Find all the diseases for which there is only one medication.

ρ_{T1}(Treatment)
ρ_{T2}(Treatment)
ρ_{D1} ($\pi_{disease}$($\sigma_{T1.disease=T2.disease}$ (T1 $\bowtie_{T1.medication <> T2.medication}$ T2))
Treatment(disease) – D1

T1(d1, m1) ← Treatment(disease, medication)
T2(d2, m2) ← Treatment(disease, medication)
D1(d1) ← T1(d1, m1) **AND** T2(d2, m2) **AND** m1 <> m2 **AND** d1=d2
D2(d) ← T(d) **AND NOT** D1(D)

Question 2:

MOVIE(title, year, length, studio, genre, producer#)
CREATE VIEW DisneyComedies **AS**
SELECT * FROM movies
WHERE studio='Disney' **AND** genre = 'comedy'

a) Is this updatable? Yes

b) If it is, write a trigger to write insertion.

CREATE TRIGGER DisneyComedies **AS**
INSTEAD OF INSERT ON DisneyComedies
REFERENCING NEW ROW **AS** newRow
FOR EACH ROW
INSERT INTO MOVIES **VALUES**(newRow.title, newRow.year, newRow.length, 'Disney', 'Comedy')

UPDATE DisneyComedies
SET length = 120
WHERE title = 'Tangled'
UPDATE Movies
SET length = 120
WHERE title='Tangled' **AND** studio = 'Disney' **AND** genre = 'Comedy'

Question 3:
Ship(name, class, launched)
Write SQL names with 3 words (King George V)
SELECT name
FROM Ship
WHERE name **LIKE** (% % %)