1)

```
CREATE TABLE Agent(ID varchar(32),
address varchar(64),
name varchar(32),
commission varchar (64),
PRIMARY KEY (ID));
```

```
CREATE TABLE Magazine(Title varchar(64),
publisher varchar(64),
PRIMARY KEY (Title));
```

```
CREATE TABLE EditorsOfMagazine(Title varchar(64),
Editor varchar(64),
PRIMARY KEY (Title,Editor),
FOREIGN KEY (Title) REFERENCES Magazine(Title));
```

```
CREATE TABLE PhonesOfMagazine(Title varchar(64),
Phone varchar(32),
PRIMARY KEY (Title,Phone),
FOREIGN KEY (Title) REFERENCES Magazine(Title));
```

```
CREATE TABLE Subscriber(email varchar(32),
name varchar(64),
address varchar(64),
PRIMARY KEY (email));
```

```
CREATE TABLE Manages(Title varchar(64),
ID varchar(32),
PRIMARY KEY (ID),
FOREIGN KEY (ID) REFERENCES Agent(ID),
FOREIGN KEY (Title) REFERENCES Magazine(Title));
```

```
CREATE TABLE Subscribes(email varchar(32),
Title varchar(64),
Since date,
Price double,
PRIMARY KEY (email, Title ),
FOREIGN KEY (email) REFERENCES Subscriber (email),
FOREIGN KEY (Title) REFERENCES Magazine(Title));
```

2. (20 points) Given relation R(A, B, C, D, E, F), give (i) the key(s) of R and (ii) the normal form of R if the FDs are

(a) AB → CD, C → E, D → F.

AB → C→ E

AB → D→ F

⇨ **(AB)+ ={A,B,C,D,E,F} ➔ AB is the Key**

Normal Form: R1(A,B,C,D) : AB is the key, C is a FK for R2, D is a FK for R3

R2(C,E): C is the Key

R3(D,F): D is the Key

(b) A → BCD, D → EF.

A → B, A → C, A → D → EF ➔ **A+={A,B,C,D,E,F} ➔ A is the key**

Normal Form: R1(A,B,C,D) : A is the key and D is a FK for R2

R2(D,E,F): D is the Key

(c) A → BC, D → EF.

AD+={A,B,C,D,E,F} ➔ AD is the key

Normal Form: R1(A,D) : AD is the key and A is a FK for R2, D is a FK for R3

R2(A,B,C): A is the Key

R3(D,E,F): D is the Key

(d) AB → C, CD → EF

ABD +={A,B,C,D,E,F} ➔ ABD is the key

Normal Form: R1(A,B,D) : ABD is the key and AB is a FK for R2 and D is a FK for R3

R2(A,B,C): AB is the Key

R3(C,D,E,F): CD is the Key

(e) AB → CD, D → EF, C → AB.

AB → C, AB → D→ EF ➔ **AB+ ={A,B,C,D,E,F} ➔ AB is a key**

C → AB→DEF ➔ **C+={A,B,C,D,E,F} ➔ C is a key**

Normal Form: R1(A,B,C,D) : AB is the key and D is a FK for R2

R2(D,E,F): D is the Key

3. (10 points) Consider the set of FDs {AB → C, C → DE, A → F, B → G, E → H}. Compute the following transitive closures:

(a) A+ = {A,F}

(b) B+ ={B,G}

(c) C + ={C,D,E,H}

(d) (A, B) + ={A,B,C,D,E,F,G,H}

(e) (A, C) + ={A,C,D,E,F,H}

4. (20 points) Consider the following table and FDs ACTIVITY(customerid, fname, lname, address, zip, status, partid, date, price, quantity, material, color)

customerid → fname,lname,address

address → zip

partid → price,material,color

customerid,partid → date,quantity,status

For each of the following decompositions, determine if lossless join holds. Use the matrix method; show only the matrix in the initial setup (before applying FDs) and the final result (after applying FDs). You can show the a's only, leave the b's empty.

(a) CUST(customerid,fname,lname,address), CUSTADD(address,zip), PART(partid,price,material,color), BUY(customerid,partid,date,quantity,status).

Initial Matrix:

	customerid	fname	lname	address	zip	status	partid	date	price	quantity	material	color
CUST	a1	a2	a3	a4								
CUSTADD				a4	a5							
PART							a7		a9		a11	a12
BUY	a1					a6	a7	a8		a10		

Final Matrix:

	customerid	fname	lname	address	zip	status	partid	date	price	quantity	material	color
CUST	a1	a2	a3	a4	a5							
CUSTADD				a4	a5							
PART							a7		a9		a11	a12
BUY	a1	a2	a3	a4	a5	a6	a7	a8	a9	a10	a11	a12

There is a row of a's So It is Lossless

(b) CUST(customerid,fname,lname,address,zip,status), PART(partid,price,material,quantity), BUY(customerid,partid,color,date).

Initial Matrix:

	customerid	fname	lname	address	zip	status	partid	date	price	quantity	material	color
CUST	a1	a2	a3	a4	a5	a6						
PART							a7		a9	a10	a11	
BUY	a1						a7	a8				a12

Final Matrix:

	customerid	fname	lname	address	zip	status	partid	date	price	quantity	material	color
CUST	a1	a2	a3	a4	a5	a6						
PART							a7		a9	a10	a11	a12
BUY	a1	a2	a3	a4	a5		a7	a8	a9		a11	a12

There is no a row of a's so it is Lossy

5. (10 points) Assume the following set of FDs: A → B, BC → D, D → E, B → C, A → D, B → E.

(a) calculate the canonical/minimal cover.

(b) For each simplified FD, show the reason it was simplified.

(c) For each eliminated FD, show the reason it was eliminated.

For eliminated FD:

		Result G
Remove: B → E from G ➔G: A → B, BC → D, D → E, B → C, A → D	B+={B,C,D,E} ➔ E ∈ B+ we can remove B → E	G: A → B, BC → D, D → E, B → C, A → D
Remove A → D from G ➔G: A → B, BC → D, D → E, B → C	A+={A,B,C,D,E} ➔ D ∈ A+ we can remove A → D	G: A → B, BC → D, D → E, B → C
Remove B → C from G ➔G: A → B, BC → D, D → E	B+={B} ➔ C ∉ B+ we cannot remove B → C	G: A → B, BC → D, D → E, B → C
Remove D → E from G ➔G: A → B, BC → D, B → C	D+={D} ➔ E ∉ D+ we cannot remove D → E	G: A → B, BC → D, D → E, B → C
Remove A → B from G ➔G: BC → D, D → E, B → C	A+={A}➔ B ∉ A+ we cannot remove A → B	G: A → B, BC → D, D → E, B → C
Remove BC → D from G ➔G: A → B, D → E, B → C	BC+={B,C}➔ BC+ ∉ D we cannot remove BC → D	G: A → B, BC → D, D → E, B → C

For simplified FDs:

		Result G
G: A → B, BC → D, D → E, B → C	B+={B,C,D,E}➔ D ∈ (BC-C)+ ➔replace BC → D by B → D	G: A → B, B → D, D → E, B → C Or G : A→B, B→DC , D→E

6. (10 points) Consider the following table and FDs:

PRODUCT(pid, pname, price, pmfr, c-o, discount, type)

pid → pname, type, pmfr

pmfr → c-o, discount

type → price

pmfr, pname → pid

(a) Give all the keys of PRODUCT and its normal form.

pid+={ pid, pname, price, pmfr, c-o, discount, type } ➔ **pid is a key**

(pmfr, pname) + ={pid, pname, price, pmfr, c-o, discount, type} ➔ **(pmfr, pname) is a key**

(b) Decompose PRODUCT into 3NF relations. Make sure the decomposition is lossless. Show only the final result of the decomposition

- ⇨ Fd is Minimal Cover
- ⇨ R1(pid, pname, type, pmfr) **pid** is the key , **pmfr** is FK for R2 and **type** is FK for R3
- ⇨ R2(Pmfr, c-o, discount) **pmfr** is the key
- ⇨ R3(type, price) **type** is the key

7. (10 points) Consider the following table and FDs:

USAGE(user-id, username, password, resource-id, type, name, location, access, permission)

user-id → username, password

resource-id → type, name, location

user-id,resource-id → permission

location → access

(a) Give all the keys of USAGE and its normal form.

Key: user-id, resource-id, location

(b) Decompose USAGE into 3NF relations. Make sure the decomposition is lossless. Show only the final result of the decomposition

- ⇨ Fd is Minimal Cover
- ⇨ R1(user-id,username, password)
- ⇨ R2(resource-id,type, name, location)
- ⇨ R3(user-id,resource-id, permission)
- ⇨ R4(location,access)
- ⇨ R5(user-id,resource-id, location) user-id is a Fk for R1, resource-id is a FK fo R2, user-id,resource-id is a FK for R3, location is a FK for R4