UNIT-3

Normalization.

The relational schema face the several undesireable problem

① **Redundancy** → meaning that the data is to be stored only once. storing the data or information many times leads to the wastage of storage space and an incresed in the total size of the data storage.

E.g,

Name	Course	Phone no	Major	Proffessor	Grade
vijay	160	99933777	CS	A	α
Sanjay	170	88002266	Physics	B	β
vijay	165	99933777	CS	C	β
Gopal	456	77213214	Maths	D	α
Santosh	491	88112244	Chemistry	E	γ
Santosh	356	88112244	Chemistry	F	α
vijay	168	99933777	CS	G	α

with respect to phone no. of vijay which has been saved several times, this decreses the performance of db.

② **Update Anomolies** :- Multiply copies of same fact may lead to update anomolies or inconsistency. when an update is made an only some of multiple copies are updated.

E.g,

suppose the phone no. of vijay is changed and we made a change only once in the list then it creates problems to db.

③ **Insertion Anomolies**

If this is the only relation in the data-base showing the association between the faculty members and the course he or she teaches, the fact that a given professore is teaching in a given course cannot be entered in a database unless a student is registered in the course.

④ **Deletion anomolies** :- If the only student registered in a given course discontinous the course then the information as to which professor is offering the course will be lost, if this is the only relationin the database showing the association between a faculty member and the course he or she teaches.

E.g;

Roll No.	Name	Course	Fees
10	Vijay	DBMS	15000
11	Santosh	C	5000
12	Gopal	C++	8000
13	Sanjay	JAVA	7000

If we delete Roll No. 11 then we lost all information name, course and fee so we cannot delete.

Decomposition The decomposition of a relation schema $R, \{A_1, A_2, A_3 ---- A_n\}$ is its replacement by a set of relation schema $\{R_1, R_2 ------ R_m\}$ such that

$R_i \leq R \quad 1 \leq i \leq m \quad$ and $R_1 \cup R_2 ---- \cup R_m = R$

A relation schema 'R' can be decompose into collection of relation schemas $(R_1, R_2, R_3 ---- R_n)$ to eliminate some of the anomalies contain in the original schema R

Functional dependency (FD):- A FD occurs when one attribute in a relation uniquely determine another attribute. This can be written as

A → B which would be the same as stating "B is functionally depended on A"

E.g. Emp (Emp-no, Emp-name, salary).

Emp-no → Emp-name.

" A functional dependency denoted by $X \rightarrow Y$ between two sets of attributes X and Y that are subset of R specifies a constraint on the possible touples that can form a relation state 'r' of R. The constraint is that, for any two touple t_1 and t_2 in 'r' that have $t_1[X] = t_2[X]$ they must also have $t_1[Y] = t_2[Y]$ ".

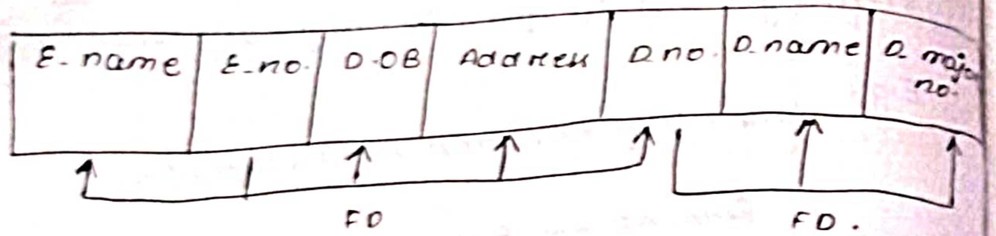
This means that the values of Y component of a touple in 'r' depend on, or determine by, the value of the X component; alternatively the values of the X component of a touple uniquely (or functionally) determine the value of the Y component. We also say that there is FD from $X \rightarrow Y$ or that Y is FD on X

X to Y

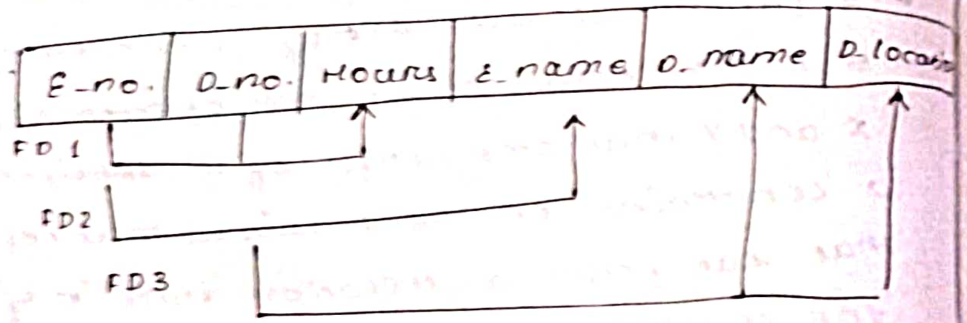
L.H.S FD R.H.S. FD.

E.g.,

(a)



(b)



Inference rules for F.D

Formally the set of all dependencies that include 'F' as well as dependencies that can be inferred from F is called the closure of F', it is denoted by 'F+'

Six rules IR1 through IR6 are well known inference rule for FD

IR1 (Reflexive rule): if $X \supseteq Y$ then $X \to Y$

IR2 (Argumentation rule): $\{x \to Y\} \models xz \to yz$

IR3 (Transitive rule): $\{x \to y, y \to z\} \models x \to z$

IR4 (Decomposition or projective rule): $\{x \to yz\} \models x \to y, x \to z$

IR5 (union or additive rule): $\{x \to y, x \to z\} \models x \to yz$

IR6 (~~Pseu~~ Pseudotransitive rule): $\{x \to y, wy \to z\} \models wx \to z$

The rules IR1, IR2 and IR3 are called as RAT Axioms or Armstrong's Axioms

Que) Given ~~Giving~~ the set of FD

$A \to B$, $ABCD \to E$ & $EF \to G$ if

~~AB~~ $ACDF \to G$ emplies by the set of given FD's

Steps to proof that two sets of FD are equivalent or not:

1. First check out the given FD
2. Now apply the inference rule (IR1 to IR6)
3. Obtain the required FD & write by applying which inference rule the relation is obtained.
4. Final obtain the result.

Sol:- $ACD \to E$ (By IR6)

$ACDF \to EF$ (By IR2).

$ACDF \to G$ (By IR3).

Hence proved.

NOTE :- The reflexivity rule (Rule 1) states that a set of attribute always determine itself or any of its element subsets, which is always true. Such type of obvious dependency is called trivial dependencies.

So a FD $X \to Y$ is trivial, if $X \subseteq Y$ otherwise it is non-trivial.

Ques) Given two sets F_1 & F_2 of FDs for a relation $F_1: A \to B, AB \to C, D \to AC, D \to E$

$F_2: A \to BC, D \to AE$ are the two sets equivalent

$A \to C$ (IR6). $\quad D \to AC$

$D \to ACE$ (IR5). $\quad D \to A, D \to C$ (IR4)

Given $D \to E$

So,

$\therefore D \to AE$ (IR5)

Again

$A \to B$ & $AB \to C$.

$\therefore A \to C$ (IR6)

$A \to BC$ (IR5).

To find closure

Algorithm to find the closure

1) $X^+ = X$;
2) Repeat
3) old $X^+ = X^+$;
4) For each FD $Y \to Z$ in F do;
5) If $X^+ \supseteq Y$ then $X^+ = X^+ \cup Z$;
6) until ($X^+ =$ old X^+)

Steps to find closure

Step-1 Find the set for which closure is to be find out

Step-2 Check for the FD of the element present in the set

Step 3 Add the element in the set. which is not present

Step 4 Repeat step 3 & 4 every element present in the set

5. If the next state element is equal to the previous state element then stop

Ques.

$A \to BC$, $E \to CF$, $B \to E$ and $C \to EF$

compute the closure X^+ of the set of attribute (A, B) under the given set of FD's

Initialize $X = [A, B]$.

$X^{(1)} = (A, B, C)$.

$X^{(2)} = (A, B, C, E)$

$X^{(3)} = (A, B, C, E, F)$

$X^{(4)} = (A, B, C, E, F)$

$X^{(3)} = X^{(4)}$

Hence the closure of given FD over (A, B) is $\{AB\}^+ = \{A, B, C, E, F\}$

Q) $A \to B$, $BC \to D$, $C \to E$, $CD \to F$

find the closure of $(A, C)^+$

Soln:- $X^0 = \{A, C\}$.

$X^{(1)} = \{A, B, C\}$

$X^{(2)} = \{A, B, C, E\}$

$X^{(3)} = \{A, B, C, E, D\}$

$X^{(4)} = \{A, B, C, D, E, F\}$

$X^{(5)} = \{A, B, C, D, E, F\}$

$X^4 = X^5$

To find super key, candidate key (irreducible set).

Steps to find super key & candidate key.

1. Check out for the F.D given
2. Now combine all the elements present on the L.H.S of the F.D without repeating an element.
3. This would be the set of superkey.
4. Now by applying the given F.D, reduce the set of the element
5. Repeat step 4 till the set of element is not functionally determined by F.D
6. This would be the candidatekey or irreducible set.

Q) $ABCD \to E$

$AB \to G$

$B \to F$

$C \to J$

$CJ \to I$

$G \to H$

S.K = {ABCDGJ}

ABCDG [∵ C → J].

ABCD [∵ AB → G].

Now the exp. can't be further reduced so it is the required candidate key. Because neither of ABCD is functionally determined by F.D so ABCD is not reducible set hence candidate key is {A, B, C, D}

Q) A → B, BC → D, BC → E, AEF → G, B → G

S.K = {A B C E F}

= {A, B, C, F} ∵ BC → E

= {A, C, F} ∵ A → B

↓

candidate key.

Q) AB → C

C → A

BC → D

ACD → B

BE → C

CE → B

CE → A

CF → B

CF → D

D → E

D → F

S.K = {A, B C D E F}

= {A B C D E} ∵ D → F

= {A B C D} ∵ D → E

= {A B C} ∵ BC → D

= {A B} ∵ AB → C

Q) A → BC

B → E

CD → EF

= {A B C D}

= {A D} ∵ A → BC

AB → C

A → DE

B → F

F → GH

D → H

{A B C D E F G H}

= {A B C D E F G} ∵ D → H

= {A B C D E} ∵ B → F

= {A B C} ∵ A → DE

= {AB} ∵ AB → C

Canonical cover

To reduce the functional dependencies to a minimum no. which covers all the dependencies

(Q) A → BC

B → C

A → B

AB → C.

A → BC

A → B

A → C , B → C , AB → C

Here A → C could be deleted because AB → C should require B → C.

$\boxed{\begin{array}{l} A \to B \\ B \to C \end{array}}$ Cannonical cover.

Lossless/Lossy Joint Decomposition

Let 'R' be a relation schema, and let 'F' be a set of F.D.s on 'R', let R_1 and R_2 form a decomposition of 'R'. This decomposition is a lossless joint decomposition of 'R' if atleast one of the following F.D.s are in 'F^+'

$R_1 \cap R_2 \rightarrow R_1$

$R_2 \cap R_2 \rightarrow R_2$

$R_1 \cap R_2 \rightarrow R_1 - R_2$

in other words, if $R_1 \cap R_2$ forms a super key of either R_1 or R_2 the decomposition of R is a lossless joint 'R' decomposition. For the general case of decomposition of a relation into multiple parts of one's, the test for lossless joint decomposition is more complicated

Example :- $S = (V, W, X, Y, Z)$

$Z \rightarrow V$

$W \rightarrow Y$

$XY \rightarrow Z$

$V \rightarrow WX$

(i) $S_1 = (V, W, X)$

$S_2 = (V, Y, Z)$

$S_1 \cap S_2 = \{V\}$

$\therefore V \rightarrow WX$

since (V, W, X) are present in S_1

$\therefore$ it is lossless joint decomposition.

(ii) $S_1 = (V, W, X)$

$S_2 = (X, Y, Z)$

$S_1 \cap S_2 = \{X\}$

there is no F.D.

$\therefore$ It is lossy decomposition

Steps to find lossless/lossy by joint decomposition matrix method:-

1. Develope mata matrix by the given set of the F.D.s
2. Put the symbol 'a' in the box for those element which are present in the given set.
3. Put the symbol 'b' in the box for those element which are not present in the given set.
4. Now apply the given FD and develope a new matrix
5. Change the symbol 'b' to 'a' if the

element is functionally dependent

(6) Now check whether any row is filled with all a's

(7) If any row is filled with all a's element then it is a lossless join decomposition otherwise it is lossy joint decomposition

Ques) Consider the relational schema

$S = (V, W, X, Y, Z)$

F.D $Z \to V$, $W \to Y$, $V \to WX$. Is the decomposition of 'S' into 'S_1' and 'S_2' is lossless?

$S_1 = (V, W, X)$, $S_2 = (V, Y, Z)$

	V	~~W~~ W	~~X~~ X	Y	Z
S_1	a_1	a_2	a_3	b_{14}	b_{15}
S_2	a_1	b_{22}	b_{23}	a_4	a_5

$V \to WX$

	V	W	X	Y	Z
S_1	a_1	a_2	a_3	b_{14}	b_{15}
S_2	a_1	a_2 ~~b_{22}~~	a_3 ~~b_{23}~~	a_4	a_5

S_2 row is all 'a' so it is lossless joint decomposition

Ques) Consider the following relation schema $S = (V, W, X, Y, Z)$ with the FD.

$Z \to V$, $W \to Y$, $XY \to Z$, $V \to WX$.

$S_1 = (V, W, X)$

$S_2 = (X, Y, Z)$ is the decomposition of S into S_1 and S_2 is lossless.

	V	W	X	Y	Z
S_1	a_1	a_2	a_3	b_{14}	b_{15}
S_2	b_{21}	b_{22}	a_3	a_4	a_5

	V	W	X	Y	Z
S_1	a_1	a_2	a_3	a_4 ~~b_{14}~~	b_{15}
S_2	a_1 ~~b_{21}~~	a_2 ~~b_{22}~~	a_3	a_4	a_5

Hence it is lossless.

Ques) Consider the relation schema $R = (A, B, C, D, E, F, G, H)$ $AB \to C$, $BC \to D$, $E \to F$, $G \to F$, $H \to A$, $FG \to H$.

$R_1 = \{ABCD\}$

$R_2 = (ABCEF)$

$R_3 = (ADFGH)$. $R_1 R_2 \& R_3$ is lossless.

	A	B	C	D	E	F	G	H
R_1	a_1	a_2	a_3	a_4	b_{15}	b_{16}	b_{17}	b_{18}
R_2	a_1	a_2	a_3	b_{24}	a_5	a_6	b_{27}	b_{28}
R_3	a_1	b_{32}	b_{33}	a_4	b_{35}	a_6	a_7	a_8

	A	B	C	D	E	F	G	H
R_1	a_1	a_2	a_3	a_4	b_{15}	$\cancel{b_{16}}$ a_6	b_{17}	b_{18} $\cancel{b_{18}}$
R_2	a_1	a_2	a_3	b_{24}	a_5	a_6	b_{27}	b_{28} $\cancel{b_{28}}$
R_3	a_1	b_{32}	b_{33}	a_4	b_{35}	a_6	a_7	a_8

It is lossy decomposition

Ques R = (E_no, E_name, P_no., P_name, P_location, Hours)

R_1 = {~~E_no~~ Employee_loan (E_name, P_location)}

R_2 = {Employee_project (E_no., P_no, Hours, P_name, P_location)}

E_no → E_name, P_no → (P_name, P_location)
(E_no, P_no) → Hours

Ques :- R = (E_no, E_name, P_no, P_name, P_location, Hours)

R_1 = Employee = (E_no, Em_name)
R_2 = Project = (P_no, P_name, P_location)
R_3 = Works_on = (E_no, P_no, Hours)

E_no → E_name, P_no → (P_name, P_location)
(E_no, P_no) → Hours.

1.

	E_no	E_name	P_no	P_name	P_location	Hours
R_1	a_{11}	a_2	b_{13}	$\cancel{b_{14}}$ a_4	a_5	b_{16}
R_2	a_1	b_{22}	a_3	a_4	a_5	a_6

It is lossy decomposition

2.

	E_no	E_name	P_no	P_name	P_location	Hours
R_1	a_1	a_2	b_{13}	$\cancel{b_{14}}$ a_4	$\cancel{b_{15}}$ a_5	b_{16}
R_2	b_{21}	a_2 $\cancel{b_{22}}$	a_3	a_4	a_5	b_{26}
R_3	a_1	a_2 $\cancel{b_{32}}$	a_3	a_4 $\cancel{b_{34}}$	a_5 $\cancel{b_{35}}$	a_6

loss-less
It is a ~~lossy~~ decomposition

Normalization :- The goal of a relational database is to generate a set of relation schema's that allow us to store information without any redundant (repeated data) it also allow us to retrieve information easily and more efficiently for this we use a approach normal form as the set of rules. These rules and regulations are known as normalization.

Database normalization is data design an organization process applied to data structure based on their FD. and primary keys that help build relational database.

Normalization of data can be looked upon a process of analysing the given relational schema based on their FD and primary key to achieve the desirable properties of :-

1. Minimizing redundancy
2. Minimizing the insertion, deletion and update anomolies

First normal form:-

A table (relation) is in 1NF if the values in the domain of each attribute of the relation are "atomic" another words only one value is associated with each attribute and the value is not a set of values or a list of values.

First normal form this allow multivalued attribute composite attribute and their combination. It states that the domain of an attribute must include only atomic values and that the value of any attribute in a touple must be a single value from the domain of that attribute. In another words 1NF this allows "Relations within relations" OR "Relations as attribute of touples". The only attribute values permitted by the 1NF are single atomic values

e.g.

Name	Designation	Address
P.K. Yadav	Manager	Paris
V.K. Yadav	Accountant	London
M.K. Yadav	Sales officer	India.

E.g

student_id	s.name	subject_1	subject_2

Here duplicacy can occur so to remove duplicacy we can decompose it into two parts

st_table

student_id	student_name

sub-table

st_id	sub_name.

Course
Course-Info

Faculty-dept	Professor	Course Preference.	
		course	Course-dept
C.S	Vijay	353 370 310	CS CS Physics
	Santosh	353 320 370	CS CS CS
Chemistry	Gopal	456 ~~370~~ 410 370	Chemistry Mathematics C.S

Professor	Course	Fac-dept	Course-dept
Vijay	353	CS	CS
Vijay	370	CS	CS
Vijay	310	CS	Physics
Santosh	353	CS	CS
Santosh	320	CS	CS
Santosh	370	CS	CS
Gopal	456	Chem	Chem
Gopal	410	Chem	Maths
Gopal	370	Chem	CS

Second Normal form:-

It depends on the concept of the primary key and functional dependency

A database is in 2NF if it is in 1NF and every attribute is fully functionaly dependent on the primary key

SSN	P_no	Hours	Ename	P.name	P.location

FD1

FD2

FD3

2NF

SSN	P.no.	Hrs

SSN	E.name

P.no.	P.name	P.location

Order no	Title	Qty	Unit price
1	CN	1	250
1	CG	1	275
1	DBMS	2	295
2	MM	1	300
2	DSC	1	190
3	DBMS	1	295
3	MM	2	300
3	CN	5	250

Primary key is order number and title. This order is already in 1NF but it is not in 2NF because unit price is not functionally dependent on order number but only on the title.

It can be seen that some book are repeated because a title can be ordered in multiple orders.

Order No.	Title	Qty
1	CN	1
1	CG	1
1	DBMS	2
2	MM	1
2	DSC	1
3	DBMS	2
3	MM	2
3	CN	5

Title	Unit price
CN	250
CG	275
DBMS	285
MM	300
DSC	190
DBMS	295
MM	300
CN	250

Course	Professor	Room	Room_cap	Enroll_unit
353	Vijay	A-532	45	40
351	Vijay	C-320	100	60
355	Santosh	H-940	50	45
456	Santosh	B-278	50	45
459	Gopal	D-110	300	200

Solⁿ :-

Course	Professor
353	Vijay
351	Vijay
355	Santosh
456	Santosh
459	Gopal

Course	Room	Room-cap	Enroll Unit
353	A-532	45	40
351	C-320	100	60
355	H-940	50	45
456	B-278	50	45
459	D-110	300	200

Third Normal Form :- A relation is in 3NF, if it is in 2NF and non prime attributes functional dependent on other non-prime attributes.

→ It is also define a relation is in 3NF, if it is in 2NF and no attributes of the table should be transitivity dependent on the primary key.

→ 3NF is based concept of transitive dependency

→ A FD $X \to Y$ in a relational schema R, in a transitive schema R, It is transitive dependency if there is a set of attribute Z that is neither a candidate key nor a subset of any key of R, and both $X \to Z$ and $Z \to Y$ hold

→ A relation schema R is in 3NF if whenever a non trivial FD

Ename	SSN	Bdate	Add	D.no.	D.name	D.mngr SSN

It is not in 3NF as it has transitivity with respect to primary key

E.name	SSN	Bdate	Add	D.no

D.no	D.name	D.mngr -SSN

Roll No.	Name	Dept	Year	Hostel Name
09	Ramesh	Physics	1	Ganga
09	Krishna	Chem	1	Ganga
10	Gopal	Maths	2	Kaveri
10	Raju	Botany	2	Kaveri
11	Maya	Geology	3	Krishna
12	Sam	CS	4	Godaveri

Transitive

Roll No. ⟶ Year.

Year ⟶ Hostel Name.

∴ Roll No ⟶ Hostel name

Roll No	Name	Dept	Year
09	Ramesh	Phy	1
09	Krishna	Chem	1
10	Gopal	Maths	2
10	Raja	Botany	2
11	Maya	Geology	3
12	Sam	CS	4

Year	Hostel Name
1	Ganga
1	Ganga
2	Kaveri
2	Kaveri
3	Krishna
4	Godaveri

course_name	Head department	Room no.	Room capacity
B.Tech(CS)	Prof. A	102	60
IT	B	107	50
EC	C	105	60
ME	D	103	100
MCA	E	111	40

course name → Room no. (Transitive

Room no. → Room capacity

course no. → Room capacity

course name	Head department	Room no.
B.Tech(CS)	Prof. A	102
IT	B	107
EC	C	105
ME	D	103
MCA	E	111

Room no.	Room capacity
102	60
107	50
105	60
103	100
111	40

Boyce - Codd Normal Form :-

It is slightly stronger version of the 3NF

⇒ A table is in BCNF if and only if

① It is in 3NF

② For every of its non-trivial F.D, X → Y 'X' is a super key OR

a relation schema 'R' is in BCNF if whenever a non-trivial FD ~~xa~~ X → A holds in R the X is the superkey of R.

NOTE:- every relation is in BCNF is also in 3NF, but a relation in 3NF is not necessary in BCNF

student

S.name	S_id	subject	Grade
Ajay	1001	Physics	A
Ajay	1001	chemistry	C
Ajay	1001	Maths	C
Anu	1002	Physics	A
Anu	1002	chemistry	A
Anu	1002	Maths	B

F.D

(i) s_name, subject → grade.

(ii) s_id, subject → grade

(iii) sid → s_name.

s-id	s_name
1001	Ajay
1002	Anu

s-id	subject	Grade

Example-2

Prof-code	Depment	HOD	% time
P_1	P	Ghosh	50
P_1	M	Krishna	50
P_2	C	Rao	25
P_2	P	Ghosh	75
P_3	M	Krishna	100
P_4	M	Krishna	30
P_4	P	Ghosh	70

F.D

(1) Prof-code , Dept ⟶ % time

(2) Dept → HOD

P

Prof. code	Dept	% time
P1	P	50
P1	M	50
P2	C	25
P2	P	75
P3	M	100
P4	M	30
P4	P	70

Dept.	HOD
P	Ghosh
M	Krishna
C	Rao

Example-3

Student	Course	Instructor
Ajay	DB	Pallav
Anu	DB	Nabaine
Anu	OS	Galvin
Anu	CN	Gopal
Anil	OS	Ahmed
Anil	DB	Pallav

F.D

① Student, course ⟶ Instructor.

② Instructor → course.

The above relation shows that the relation is in 3NF is not necessary in BCNF.

Fourth Normal Form

An entity type is in 4NF. If it is BCNF and there are non-multi value dependency between its attributes type

An entity is BCNF is transformed is 4NF.

① Direct may multivalue dependency
② Decompose: entity type.

→ " A relation schema 'R' is in w.r.to a set of dependencies 'F' (that include F.D and multi value dependencies) if fore every non trival multivalued dependencies $X \twoheadrightarrow Y$ in F^+ is a super key of fore R "

Author No.	Book-No.	Subject	Book Title	Author name
A1	B_1	CS.	Methods	Vijay
A1	B_1	Maths	Methods	Vijay
A2	B_1	CS	Methods	Gopal
A2	B_1	" Maths	Methods	Gopal
A1	B_2	Maths	Calculus	Vijay

In BCNF

✓ Author (Author_No., Author name)

✓ Book (Book_No., Book. Title).

* Author-Book subject (Author No, Book No, Subject)

Book No. $\twoheadrightarrow$ Author-No

Book No. $\twoheadrightarrow$ Subject

Author No.	Book No	Subject
A1	B_1	CS.
A1	B_2	Maths
A2	B_1	Maths.
A1	B_1	Maths

Author No	Book No.
A1	B1
A2	B1
A1	B2

Book No	Subject
B1	CS
B1	Maths
B2	Maths

A. name	A. no
Vijay	A1
Gopal	A2

B. title	B. no.
Methods	B1
Calculus	B2

Author - Book (Author No, Book No.)

Book - Subject (Book No, Subject).

Faculty relation (BCNF Forms).

Faculty	Subject	Institute
Vijay	DBMS	AIMT
Vijay	DS	AIMT
Vijay	C++	AIMT
Vijay	DBMS	GIMT
Vijay	DS	GIMT
Gopal	DS	GIMT
Gopal	JAVA	IILM

Faculty	Subject.
Vijay	DBMS
Vijay	C++
Vijay	DS
Gopal	DS
Gopal	JAVA

Faculty	Institute.
Vijay	AIMT
Vijay	GIMT
Gopal	GIMT
Gopal	IILM

Fifth Normal Form (5NF)

Join dependency. A join dependency (JD) denoted by (R_1, R_2, R_3) specified on relation scheme R, specifies a constraint all the states r of R should have a lossless join decomposition into $R_1, R_2 \ldots\ldots R_n$

That is for every such r, we have

$(\pi_{R_1}(r), \pi_{R_2}(r) \ldots\ldots \pi_{R_n}(r)) = r$

a notice that an MVD is a special case of a JD where n=2 i.e, a JD denoted as $JD(R_1, R_2)$ implies an MVD $(R_1 \cap R_2) \rightarrow R_1 - R_2$. A $JD(R_1, R_2 \ldots\ldots R_n)$ specified on relation schema R, is a ~~trivial~~ trival JD if one of the relation schema R_i in JD $(R_1, R_2 \ldots\ldots R_n)$ is equal to R such a dependency is called trival because it has a lossless joint property for any relation state r of R and hence does not specify any constraint on R

5NF A relation is in 5NF if it is in 4NF and cannot be further non-loss decomposed. More techniquely in 5NF we use the concept of JD which is generalised form of MVD.

→ A relation schema R is in 5NF or <u>project join Normal Form (PJNF)</u> with respect to a set of F of functional, MV and JD if for every non-trivial JD ($R_1, R_2, \ldots R_n$) in F^+ (i.e. implied by F) every R_i is a superkey of R

company	Product	supplier
Godrej	soap	Mr.X
Godrej	shampoo	Mr.X
Godrej	shampoo	Mr.Y
Godrej	Shampoo	Mr.Z
Godrej	soap	Mr.X
H.Lever	soap	Mr.Y
H.Lever	Shampoo	Mr.Y
H.Lever		

The above table is in 4NF (i.e there is no MVD) but it have lots of redundancy so we decompose

company	Product
Godrej	soap
Godrej	shampoo
H.Lever	soap
H.Lever	shampoo

company	supplier
Godrej	Mr.X
Godrej	Mr.Y
Godrej	Mr.Z
H.Lever	Mr.X
H.Lever	Mr.Y

the above has been eliminated redundancy but we have lost the information for e.g if we want to display the products and their supplier, then we will have to use the join based on the company attribute the result will display some spurious record Mr.Z it will display both the product soap and shampoo as the company for which Mr.Z is the supplier (Godrej) is producing soap & shampoo which is wrong. Therefore original table breaks into 3 parts.

company - product, company - supplier, product - supplier.

Product	supplier
Soap	Mr.X
shampoo	Mr.X
shampoo	Mr.Z
Soap	Mr.Y
Shampoo	Mr.Y

Supply

S_name	Part_name	Project_name
Smith	Bolt	X
Smith	Nut	Y
Ajay	Bolt	Y
Walton	Nut	Z
Ajay	Nail	X
Ajay	Bolt	X
Smith	Bolt	Y

Sol:-

S_name	P_name
Smith	Bolt
Smith	Nut
Ajay	Bolt
Ajay	Nail
Walton	Nut

S_name	Proj_name
Smith	X
Smith	Y
Ajay	X
Ajay	Y
Walton	Z

P_name	Proj_name
Bolt	X
Nut	Y
Bolt	Y
Nut	Z
Nail	X

Rough

Schedule A

T_1	T_2
$X = 90 - 3$	
$= 87$	
$Y = Y + N$	
$= 90 + 3$	
$= 93$	
	$X = X + M$
	$90 + 2$
	$= 92$
	$87 + 92$
	[illegible]

Schedule B

T_1	T_2
	82
82	
93	

Schedule C

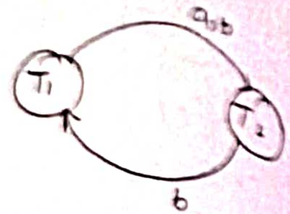
T_1	T_2
S	

Schedule D

T_1	T_2

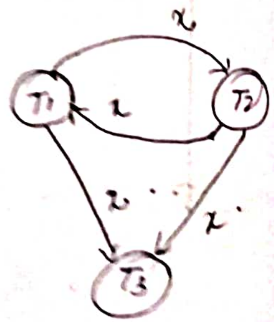
7/4/14

T_1	T_2
r(a)	
r(b)	
	r(a)
	a = a + ...
	w(a)
	r(b)
	b = b +
	w(b)

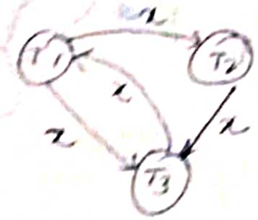


Sol:

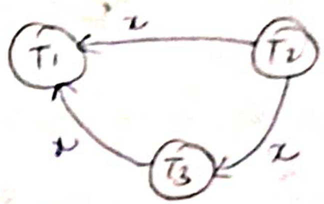
T_1	T_1	T_3
r(x)		
	w(x)	
w(x)		
		w(x)

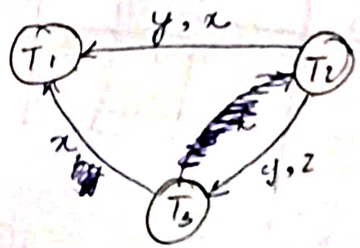


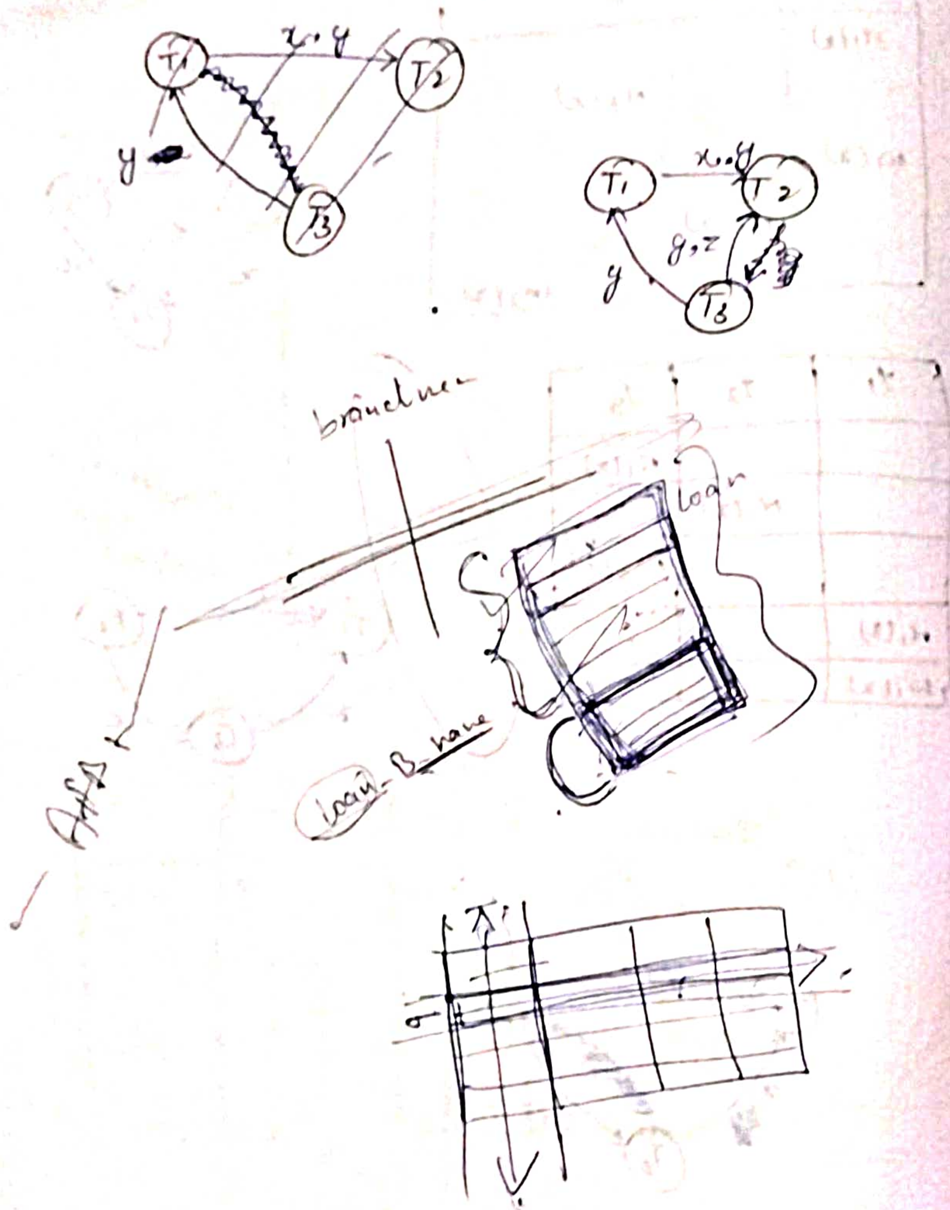
T_1	T_2	T_3
$r_1(x)$		
		$r_3(x)$
$w_1(x)$		
	$r_2(x)$	
		$w_3(x)$



T_1	T_2	T_3
		$r_1(x)$
	$r_2(x)$	
		$w_3(x)$
$r_1(x)$		
$w_1(x)$		







(Q) Union and set difference are fundamental operation but intersection is non-fundamental operation.

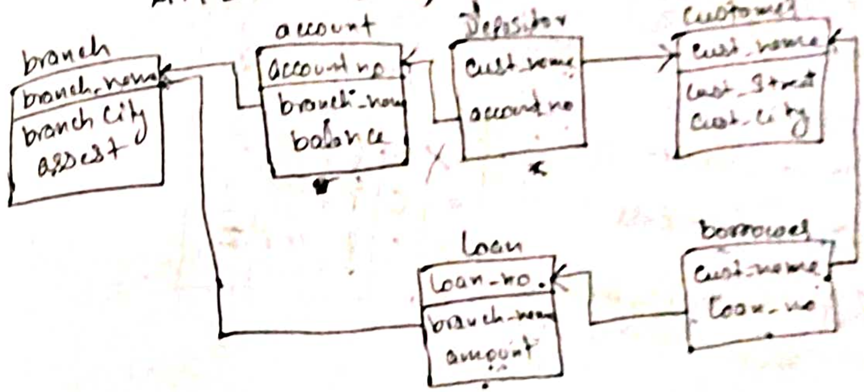
$A-(A-B)$

$(A-B) = 1,2$

$A-(1,2) = 2$

intersection

$A \cap B = A-(A-B)$



Q: Find name of customer who have a loan or an account or both.

$\pi_{cust\_name}(\text{Depositor})$

$\cup$

$\pi_{cust\_name}(\text{borrower})$

but it avoid duplication in ~~name~~ name.

Q: Find the customer who have both account and loan.

$\pi_{cust\_name}(\text{Depositor}) \wedge \pi_{cust\_name}(\text{borrower})$

Q: Find all the customers who has account but not loan.

$\pi_{cust\_name}(\text{Depositor}) - \pi_{cust\_name}(\text{borrower})$

Q: Find the name of branch who have account but not loan

$\pi_{branch\_name}(\text{account}) - \pi_{branch\_name}(\text{loan})$

Q: Find the name of customer who ~~have~~ neither have loan nor an account.

$\pi \ldots (\text{customer}) - (\pi_{cust\_name}(\text{Depositor}) \cup \pi_{cust\_name}(\text{borrower}))$