Unit:-5 Relational Database Design

Normalization

1) Functional dependencies, keys.
2) Normalization
3) Indexing, physical structures
4) Transaction.

1) Functional dependencies

α	β
a	1
b	2
c	3
d	4

$f : \alpha \rightarrow \beta.$

→ On value of α you will compute β that is not true.

→ it means fd $\alpha \rightarrow \beta$, on the value of α you can search the value of β.

→ in above table all the value of of α is diff.

$$\alpha \rightarrow \beta$$
$$a \rightarrow 1$$

So that y we called as functional dependency

but a is multiple time ~~but~~ then if i say α value is a so what is β?

So we can't say the α value.
that's y we called as fd.

if i say $f(x) \rightarrow y$

x ki value pe y value

$1 \longrightarrow a$ (true)

$1 \longrightarrow b$ (false)

$2 \longrightarrow a$ (true)

two diff x value pe y ki same value mil sakti he that is true.
but α ki same value pe β ki same value hi honi chahiye.
So every time on you must have it.

α	β
a	1
a	2
c	3
d	4

$t_1 \rightarrow$ a 1, $t_2 \rightarrow$ a 2

if i do then for both β value this is true.

→ if i say fd : $\alpha \rightarrow \beta$ in R it means if i tell you value of α then you can search the value of β.

Date
Page

$\alpha \subseteq R, \ \beta \subseteq R.$

	α	β
$t_1 \rightarrow$	a	b
$t_2 \rightarrow$	a	b

R

$\alpha \longrightarrow \beta.$

determinant (α) → dependent (β).

Method: if $t_1[\alpha] = t_2[\alpha]$ ↳ $t_1[\beta] = t_2[\beta]$.

→ $\alpha \longrightarrow \beta$.

Trival | Non-trival.

(valid) $AB \longrightarrow A$. | $B \not\subseteq A$.

Ex Accountant

if $\beta \subseteq \alpha$

Which Fd holds good?

Ex
- ✓ (a) $A \longrightarrow BC$
- ✓ (b) $DE \longrightarrow C$.
- ✗ (c) $C \longrightarrow DE$
- ✓ (d) $BC \longrightarrow A$.

R

A	B	C	D	E
a	2	3	4	5
2	a	3	4	5
a	2	3	6	5
a	2	3	6	6

→ if $\alpha \rightarrow \beta$. What are the Condition? it is not hold good when on $a \longrightarrow 1$
on $a \longrightarrow 2$.

Date
Page

→ First you have to see if all α values are diff. than it all good
→ Second you have to check β value is ~~d~~ Same.

→ Ex

A	B	C	D	E
a				
b				
c				
d				

$A \to B$

$A \to BD$

$A \to BCDE$.

Can i say

$A \to C$

$BD \to C$

$ABDE \to C$.

all value of β same.

$\alpha \to \beta$

Step-1 :- Check value of α
α value is diff ✓

Step 2 - β value is Same than No need to check.

Step :-3 Step by step check α & β.

Note → Data depend on dependency, dependency Not depend on Data.

Example :-

f.d. valid or Not valid.

Q.

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

× (a) $XY \rightarrow Z$ && $Z \rightarrow Y$
✓ (b) $YZ \rightarrow X$ && $Y \rightarrow Z$
× (c) $YZ \rightarrow X$ && $X \rightarrow Z$
× (d) $XZ \rightarrow Y$ && $Y \rightarrow Z$

$\alpha \rightarrow \beta$
a
b
d.

Q-2

A	B	C
1	2	4
3	5	4
3	7	2
1	4	2

$\alpha \rightarrow \beta$

× (a) $A \rightarrow B$ && $BC \rightarrow A$.
× (b) $C \rightarrow B$ && $CA \rightarrow B$.
✓ (c) $B \rightarrow C$ && $AB \rightarrow C$
× (d) $A \rightarrow C$ && $BC \rightarrow A$.

* Closure of set of Attributes | Attribute closure
Closure on Attribute set.

→ Attribute closure of an attribute set 'A' can be defined as set of attribute which can be functionally determined from it denoted by F^+

Bindal Ukeka

Binaal

Date
Page

→ We can ~~to~~ work on Functional Dependency
→ We can Minimize it, Compare of functional Dependency Set. or we can Identify Candidate Key also.

→ Even Normalization we need Attribute Closure.
→ Then after you have to Learn some Rule on Which FD holds. that is known as Armstrong Axiom Rule.

Q-1 $R(ABC)$ $(A)^+ \rightarrow A$ ↗ directly or indirectly We can Find A Attributes

$A \rightarrow B$ AB

$B \rightarrow C$ ABC.

Q-3 $R(ABCDEF)$

$A \rightarrow B$ $(D)^+ \rightarrow ADF$

$C \rightarrow DE$ $\rightarrow ABDF$

$AC \rightarrow F$

$D \rightarrow AF$ $(DE)^+ \rightarrow ADF$

$E \rightarrow CF$ $\rightarrow ACDEF$

$\rightarrow ABCDEF$

→ Attribut closuer, Set of attribute closure.

Ad:- We ~~need~~ can check the all FD.

This is direct method, Here We have Armstrong Rule. that is ~~depend~~ holds on all the FD

Date ______
Page ______

* "Armstrong" axiom/rule :-

→ axiom is a statement that is taken to be true and serve as premise or starting point for further arguments.

(kabhi galat nahi hota)

→ Armstrong axioms holds on every relational database can be used to generate closure set.

→ For many analysis it is used.

Primary Rule.

→ Let's understand we can say that Functionally Complete hona.

in DE, NAND or NOR gate we can say functionally complete why we can say that using this two component we can make any circuit.

→ Koi bhi boolean function implement using NAND & NOR gate.

→ So using two we can implement anything the why we use another gate that is bcz few energy use

→ bcz only NAND ~~se~~ gate se banayege to it becomes more complex, costly & all.

→ For that every functionality ke liye diff. gate available he.

→ This same concept work with the primary rule.

→ Primary Rule (RAT) (Basic framework of FD)

1) Reflexivity :-

→ if $y \subseteq x$. $xy \subseteq R$.

then, $x \rightarrow y$

if y is set of attribute of x then there always FD from $x \rightarrow y$

Ex $X = ABC$ $Y \subset X$.
$Y = AB$. $ABC \longrightarrow AB$ (trival FD)

So if i know ABC the i defineatly know the AB.

2) Augmentation :- if $X \rightarrow Y$
then $XZ \rightarrow YZ$

We can augment anything both size.
You can Add

3) Transitivity

if $X \rightarrow Y$ && $Y \rightarrow Z$

$X \rightarrow Z$.

* Secondary Rules

1) Union :- if $X \rightarrow Y$ && $X \rightarrow Z$

$X \rightarrow YZ$

2) Decomposition — if $X \rightarrow YZ$

then, $X \rightarrow Y$ && $X \rightarrow Z$

assumem $AB \rightarrow CD$.

$AB \rightarrow C$ ✓ Decomposition

$AB \rightarrow D$

different ko eksath likte he then Union., So we do only right side

$AB \rightarrow C$

$A \rightarrow C$ X.

$B \rightarrow C$

you can write $AB \rightarrow C$

but only A & B se you can find C.

→ So aap yaha FD ka power decree hota he So on left side we can't do decomposition

3) pseudo transitivity

if $X \to Y$ && $WY \to Z$
then $WX \to Z$

$WX \to WY$ (Augmentation).
$WY \to Z$ (Transitivity).

4) Composition if $X \to Y$ && $Z \to W$.
$XZ \to YW$.

$ZX \to ZY$. (Augment).
$ZX \to XW$.

$ZX \to ZYXW$ (union)
decomposition

$XZ \to YW$
$XZ \to XZ$.

⇒

Date
Page

* **Closure Set of Attribute :-**

Suppose i have table R (A B C)

A → B.

B → C.

To do Normalization We need FD. So should i go partial FD or should i go entire FD.

→ if i tell you A → B , A → C.

A → BC

it means what

Let me Say

Dicompositn

F (FD holds in table) = $F_1 + F_2$ (which not visible directly but holds good)

(directly visible)

A → C

A → B.

(Total FD) A → BC.

before Normalization

We need total FD. You have entire set of FD. For that We need Colsure Set of Attribute.

* Identify Candidate ke, Number of key So For that you have to knowlege about the keys.

→ $A^+ = \{A, B, C\}$.
$B^+ = \{B, C\}$.
$C^+ = \{C\}$.

→ 1) R (ABCDEFG)

1) A → B
2) BC → DE
3) AEG → G

$(AC)^+ = AC$
$= ABC$.
$\boxed{(AC)^+ = ABCDE}$

entire set of FD.

2) R (ABCDE)

A → BC
~~E~~D → E
B → D
E → A

$(B^+) \to B$
$= \boxed{BD.}$

3) R(ABCDEF)

AB → C
BC → AD
D → E
CF → B.

$(AB)^+ = ABC$.
$= ABCD$.
$= ABCDE$

~~4)~~

Date ____
Page ____

4) R(ABCDEFGH)

A → BC

CD → E

E → C

D → AEH

ABH → BD

DH → BC

BCD → H ? (valid or not) Not valid

$(BCD)^+$ → BCDE

→ BCDAE

→ ABCDEH

→ ABCDEH

→ ~~AB~~

* Equivalence of Functional Dependency.

→ We have How to compare two diff. set of FD.

Ex R(ACDEH)

F:	G:
A → C	A → CD
[illegible]	E → AH.
E → AD	
E → H	

a) F ⊆ G

b) F ⊇ G

✓ c) F = G

d) F ≠ G

You have to find that what is the subset of others or Equal or Not equal.

F:
left side | Now i compute with G

$(A)^+ = ACD.$
$(AC)^+ = ACD.$ → This is power of G.
$(E)^+ = \underline{AEHCD}.$

→ All those work which done by F that is also done by G. $F \subseteq G$.

$A \to CD$
$E \to AH$

$(A)^+ = ACD$
$(E)^+ = EAD.$
$= EADHCD.$

$G \subseteq F.$

$\boxed{F = G}$

Q.d) R(PQRS)

X: $P \to Q$ ✓
$Q \to R$ ×
$R \to S$ ✓

Y: $P \to QR$
$R \to S$

(a) $X \subseteq Y$
(b) $Y \subseteq X$
(c) $X = Y$
(d) $X \neq Y$

$(P)^+ = PQRS.$
$(Q)^+ = Q.$
$(R)^+ = RS.$

~~$Y \subseteq X$~~.
$X \not\subseteq Y.$

$(P)^+ = PQRS$
$(R)^+ = RS$

$Y \subseteq X.$

Q-3 R(ABC)

F: $A \to B$
$B \to C$
$C \to A$

G: $A \to BC$
$B \to A$
$C \to A$

$(A)^+ = ABC$
F⊆G $(B)^+ = BAC$
G⊆F $(C)^+ = CAB$

$(A)^+ = ABC$
$(B)^+ = BCA$
$(C)^+ = CAB$

F=G Ans F=G

Q-4 R(~~V~~ VWXYZ)

F: $W \to X$
$WX \to Y$
$Z \to WY$
$Z \to V$

G: $W \to XY$
$Z \to WX$

$(W)^+ = WXY$
$(WX)^+ = WXY$
$(Z)^+ = ZWXY$

F⊄G

$(W)^+ = WXY$
$(Z)^+ = ZWYVX$

G⊆F

(Minimal FD)

Date
Page

* Introduction Set of FD (Canonical form)

→ Any given FD we have to Find any Reduddant element is avilable or not or, if any Redudant element avilable then we must have to Remove it.

→ So basically We have to minimize FD.

Q.1) R(Wxyz)

$x \to W$

$Wz \to xy$

$y \to WXZ$

$\underset{\text{Redudancy}}{\alpha} \to \underset{\text{Reduency}}{\beta}.$ → or entire Form Redudent

Extra

⇒ decomposition.

$\alpha \to \beta\gamma$.

$\alpha \to \beta$

$\alpha \to \gamma$

step :- 1 Apply decompusit

1) $X \to W$

$W2 \to X$ (Remove it)

$W2 \to Y$

$Y \to W$ X

$Y \to X$

$Y \to 2$

Here no Redudancy on write side.

→ How to Find it is accetial or it is Reduandant. iske rahne ya rahne se Fark nahi pad raha

→ if your existance Which not affect the system that means you are Reduandant.

$(x^+) = xw$

$\hookrightarrow x^+ = x$. (ignore x. So if i ~~[illegible]~~ ignore x FD & then i get same value then it is Redudent but i get diff. value then it is essential)

one more time.

$(wz)^+ = wzxy$; if i ignore $wz \to x$ then also we can find it.

$(wz)^+ = ywxz$

$\hookrightarrow (wz)^+ = wz$ (impotent)

$y^+ = ywxz$

$y^+ = yxzw$ (Redunant)

$y^+ = \cancel{ywxz}\ yz$ $(y \to x)$ essential

$y^+ = yxw$ (essential) $y \to z$.

$\Rightarrow$ After this filter $x \to w$

$wz \to y$

$y \to \cancel{x}$

$y \to z$.

How check wz me se ek Redudent nahi he.

$(wz)^+ = wzyx$.

Single closer $\begin{cases} w^+ = \\ z^+ = \end{cases}$

if both are same that it implies that
2 is Redudant Why?

$w^+ = w$. (ignore $wy \to 2$)

$2^+ = 2$.

↳ both essential.

So
$X \to W$
$WZ \to Y$
$Y \to XZ$

this set of dependency is cannonical form of the Redundant def.

Q-2) $R(ABCD)$

$A \to B$

$C \to B$

$D \to ABC$

$AC \to D$.

Step: 1

$A \to B$

$C \to B$

$D \to A$.

$D \to B$ (X)

$D \to C$ (X)

$AC \to D$.

⇒

$A \to B$

$C \to B$

$D \to A$.

$AC \to D$

$D \to C$

Final

$A^+ = AB$. (ignore $A \to B$).

$A^+ = A$

$C^+ = CB$. (ignore $C \to B$)

$C^+ = C$

$D^+ \Rightarrow DABC$

$D^+ = DBC$ (ignore $D \to A$)

$D^+ = DABC$. (Redundant)

$D^+ \to DAB$. (ignore $D \to C$)

Redundant

$AC^+ = ACDB$

$AC^+ = ACB$. (ignore $AC \to D$)

Now check for

$(AC)^+ = ACDB$.

$A^+ = AB$.

$C^+ = CB$.

ignore $AC \to D$

Q-3) R(VWXYZ)

$V \to W$

$VW \to X$

$Y \to VXZ$

$V \to W$

$VW \to X$

$Y \to V$

$Y \to X$ (x)

$Y \to Z$

$V^+ = VWX$. (ignore $V \to W$)

$V^+ = V$.

$(VW)^+ = VWX$. &

$(VW)^+ = VW$. (igme $VW \to X$)

$Y^+ = YVWXZ$ (

$Y_V^+ = YXZ$ (ignore $Y \to V$)

$Y_X^+ = YVWZX$. (igm $Y \to X$)

$Y_Z^+ = YVXW$.. (igm $Y \to Z$)

Step 1 - 2

$V \to W$

$VW \to X$

$Y \to V$

$Y \to Z$

$(VW)^+ = VWX$.

$V^+ = VWX$.

$W^+ = W$ (Redunant)

$V \to W$

$V \to X$

$Y \to V$

$Y \to Z$

Final

* :- Types of keys :-

A	B	C
1	a	p
2	b	q
3	c	q
4	c	r.

=) For finding Data from database We need keys.

→ keys is very important in case of Database & Normalization.

Ex → if Normalization is war then We need gun for protection & FD is gun then keys are bullets.

→ Without key We can do Normalization.

→ So if you want some data from database then you have to tell from where you have to access the data.

→ So basically We divide table into Number Columns & Rows.

→ For that you need to Learn Addressing for accessing data from database.

→ So if you have 10 columns then You have to tell me that 10 me se ek particular column data i need.

Date ____
Page ____

→ To database me columns ke name available he.

→ if i say i want to access data from A then i can give the value.

→ but if there are so many Rows available & i want fetch data from Rows, so you can see that there is no name for row.

→ So key say that every column has name but every row has no name.

→ So, we need some information from Rows then we need hint & whatever hint you are using that is key.

→ Assume that we have one table student in that we have some column Roll-no, name, age, branch.

Student

R-roll	name	age	branch
1	A	19	101
2	B	20	101
3	A	19	101

→ So from this table if you learn How to Access data from row then we can say you are able to fetch it.

=) key ki value ke base pe we can find rows value

Date
Page

Def^n => key is set of Attribute which can uniqly identify a row or tuple.

* R key = $A \to BC$ Apart from A, BC (key)

$(A)^+ = R$

(ABC)

$(key)^+ = R$

↳ when we find closure for key then we get whole Relation.
if it not find out whole R then it is not a key.

→ ek system me ek se jyda key ho sakti he, single Attribute or set of Attribute then you can identify it uniquley.

=> you have to learn super key, foriegn key, Primary key.

R(ABCD)

(1) $A \to BC$.
$A^+ = ABC$
A is not key bcz C missing.

(2) R(ABCD)

✓ $ABC \to D$. $(ABC)^+ = ABCD$

✓ $AB \to CD$ $(AB)^+ = ABCD$

✓ $A \to BCD$ $(A)^+ = ABCD$

which one is key? => All three according definition key

* Super key :-

$(SK)^+ = R.$

S.K is set of key using that you can uniquelly identify all the Attributes.

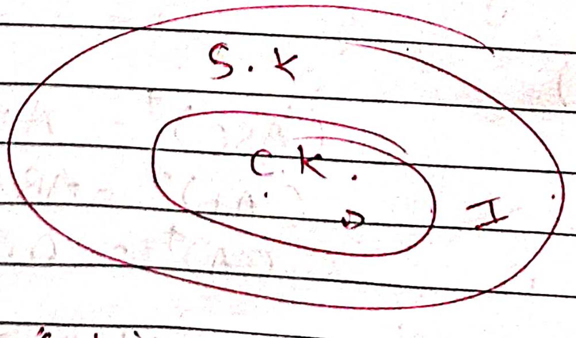
* Valid but not efficient ABC & AB.

* Candidate key :-

Super key k efficient way ko Candidate key. So A is here Candidate key.

⇒ Us super key ko Candidate key kehte jiska koi proper subset super key na ho

SK $(ABC)^+$ proper subset.
SK. $(AB)^+$. proper subset.
Sk./C.k. $(A)^+$ → No proper subset key.
↳ Minimal.



All S.k is not C.K.
but all C.k is S.K.

Date
Page

(3) R(ABCD)

B → ACD ✓ S.K / C.K / P.K.

ACD → B ✓ S.K / C.K. → AC, AD, CD, A, C, D.

$(B)^+$ = ABCD.

$(ACD)^+$ = ABCD.

(4) R(ABCD)

AB → C. $(AB)^+$ = ABCD / S.K / C.K

C → BD. $(C)^+$ = ACBD. / S.K / C.K.

D → A $(D)^+$ = DA.

No logicall diff between key & S.key.

S.K → C.K → P.K.

⇒ ek se jyda S.k & C.k. possible but P.k. is only one.

⇒ technically no diff between C.k & P.k.

PK ⇒ V/s PK :- is that C.K which is selected by DBA as primary mean to identify.

① R(ABCD)

A → B

B → C

C → A.

~~A^+~~ =

~~B^+~~ =

~~C^+~~ =

Check attribute on write side.

D is essistenal no one is Find it.

$(D)^+ = D$. (No key)

$(AD)^+ = ABCD$. | S.K. | C.K.

$(BD)^+ = BCAD$. | S.K. | C.K.

$(CD)^+ = ABCD$. | S.K. / C.K.

② R(ABCD)

AB → CD.

D → A.

R(ABCD)

B has no incoming age.

$(B)^+ = B$.

$(AB)^+ = ABCD$ ✓ C.K.

$(BC)^+ = BC$ ×

$(BD)^+ = B\cancel{A}AD$ ✓ C.K.

(3) $R(ABCDEF)$.

$AB \to C$

$C \to D$

$B \to AE$

$R(ABCDEF)$

$\cancel{(ABF)^+}$ $\quad$ $(BF)^+ = ABCDEF$

$\cancel{(CBF)^+}$

$\cancel{(DBF)^+}$ No, need to find the Combination.

~~BFA~~.

(4) $R(ABCD)$

$AB \to CD$

$C \to A$

$D \to B$

$R(ABCD)$

$\cancel{(AB)^+}$ ✗

$A^+ = A$

$B^+ = B$

$C^+ = AC$.

$D^+ = DB$

$AB^+ = ABCD$ ✓ C.K $\quad$ ~~ACB~~ ✗

$AC^+ = AC$ ✗ $\quad$ ACD ✗

$AD^+ = ADBC$ ✓ C.K $\quad$ BDA ✗

$BC^+ = BCAD$ ✓ C.K $\quad$ BDC ✗

$BD^+ = BD$ ✗

$CD^+ = ACDB$ ✓ C.K

(5) R(ABCDE)

$AB \to CD$

$D \to A$

$BC \to DE$.

R(ABCDE)

~~A^+~~ $AB^+ = ABCDE$ ✓

~~B^+~~ $BC^+ = ABCDE$ ✓

~~C^+~~ $BD^+ = BDACE$ ✓

~~D~~ $BE^+ = BE$ ×

(6) R(WXYZ)

$Z \to W$

$Y \to XZ$.

$\cancel{X}W \to Y$

R(WXYZ)

$Z^+ = WZ$ ×

$Y^+ = XZYW$. ✓

$X^+ = X$ ×

$W^+ = W\cancel{Y}$. ×

No need to apply Y in Combination

$WX^+ = WXYZ$ ✓

$XZ^+ = XZWY$ ✓

$WZ^+ = WZ$ ×

Three keys = Y, WX, XZ.

(7) R(ABCDEF)

AB → C

DC → AE

E → F

R(ABCDEF)

$(BD)^+ = BD$.

$(ABD)^+ = ABCDEF$

$(BCD)^+ = BCDAEF$

$(BDE)^+ = BDE$ ✗

$(BDF)^+ =$ ✗

$(BDEF)^+ =$ ✗

key = ABD^+, BCD^+.

(8) R(ABCDE)

CE → D

D → B

C → A

R(ABCDE)

$(CE)^+ = CEDAB = R$.

(9) R(ABCDEFGHIJ)

AB → C

AD → GH

BD → EF

A → I

H → J.

R(ABCDEFGHIJ)

$(ABD)^+ = ABCDEFGHIJ$

(3) R(ABCDE)

A → B

BC → E

DE → A.

R(ABCDE)

~~A~~: $(CD\cancel{E})^+$ = CD~~EAB~~

$(ACD)^+$ = ACDBE.
$(BCD)^+$ = BCDEA.
$(CDE)^+$ = CDEAB
} C.K.

(4) R(ABCDE)

BC → ADE.

D → B.

R(ABCDE)

$(C)^+$ = C.

$(AC)^+$ = AC. ✗

$(BC)^+$ = ABCDE ✓

$(CD)^+$ = CDBAE ✓

$(CE)^+$ = CE ✗

only two key BC^+ & CD^+.

(5) R(ABCDEF)

AB → C

C → D.

D → BE

E → F

F → A

R(ABCDEF)

Date
Page

(A~~CF~~)⁺

$A^+ = A$ ×
$B^+ = B$ ×
$C^+ = CDBEFA$ ✓
$D^+ = BDEFAC$. ✓
$E^+ = EFA$ ×
$F^+ = AF$ ×.

No need to check any Combination with C & D. How many failure 4. Now make Combination of this.

$(AB)^+ = ABCDEF$ ✓
$(AE)^+ = AEF\cancel{A}$ ×.
$(AF)^+ = AF$ ×
$(BE)^+ = BEFACD$. ✓
$(BF)^+ = BFACDE$ ✓
$(EF)^+ = EFA$. ×.

11000
6000 - grade pay
21000 basics

Keys :- C^+, D^+, AB^+, BE^+, BF^+.

$(AEF)^+$ ×.

(6) R(ABCDEFGH)

CH → G
A → BC
B → CFH.
E → A
F → EG.

R(ABCDEFGH)

$(\cancel{DE})$ = ~~DEABCFHG~~

$(D)^+ = D$.

$AD^+ = ABCFHDGE$ ✓
BD^+ ✓
CD^+ ×
DE^+ ✓
DF^+ ✓
DG^+ ×
DH^+ ×.

$(CDG)^+ = $ ×
$(CDGH)^+ = $ ×.

4.1 Insertion, Deletion & update anomalies

→ student_info

S_id	name	age	Branchcode	Br_name	HOD_name
1	A	18	101	CS	xyz
2	B	19	101	CS	xyz
3	C	18	101	CS	xyz
4	D	21	102	EC	PQR
5	E	20	102	EC	PQR
6	F	19	102	ME	KLM

* Same infor. we store multiple time that is known as Redudancy.

→ This is Fundamental problem.

Date ____
Page ____

Idea :- In the table student info we have tried to store entire data about student.

Result :- Entire branch data of a branch must be Repeated for every student of the branch.

Redundancy :- When same data is stored multiple times unnessarly in database

Disadvantage

(i) Insertion, deletation & modification Anomalies.
(ii) Inconsistency (data)
(iii) Increase in database size & increase in time (slow)

* Insertion Anomalies :- When certain data (Attributes) cannot be inserted into Data base, Without the presence of other data.

* Deletion Anomalies :- i we delete some data (unwanted), it cause deletion of some other data which is wanted.

Date
Page

* Updation / modification Annomalies :-

When We want to update a Single piece of data, but it must be done at all the Copies.

4.2 Normalization

Student info (Refer privious table)

→ Decomposition of table

* Decomposition of table jab tak table ek logic pe na ajye.

* ~~THis use by~~ In this We need Functional dependancies.

Student_info

S.id	Name	age	B.code
1	A	18	101
2	B	19	101
3	C	18	101
4	D	21	102
5	E	20	102
6	F	19	103

Branch info.

Br code	B name	HOD Name
101	C.S	xyz
102	E.C	PQR
103	M.E	KLM

Date ______
Page ______

4.3 Normalization is Process Which We use to Reduce the Redundancy.

To do Normalization We will use Fd.

⇒ 1NF :-

Rollno.	name	Course
101	Modi	CN/OS
102	Sonia	OS/DBMS.
101	Modi	~~DBMS~~ CN.
101	Modi	Os
102	Sonia	DBMS
102	Sonia	Co.

Number of implecation

⇒ A table Set to be 1NF, if ~~table Contai~~ every Shell Contain Automic value.

→ For Every multivalued values you need to maintain a individual table.

(Domain Same)

→ Every column Should Contain values of domain.

→ They must be belongs to from Same domain.

→ The orders of Colum & orders row is insignificant no matter

→ Every column Should have unique Name.

→ Maxeless trivall value.

2NF

→ R(ABCD)

A & B essential. $(AB)^+ = ABCD = C.K.$

When you understand What is C.K key then you are able to define Prime attribute & Not prime Attribute.

* if i Say A & B is C.K then $A, B \in$ Prime Attribute
  & thoes Who are not part of C.K that is non prime attributes.

* For 2 NF implications.
1) you must able to identify C.K.
2) if Attribute is part of C.K then it set to be Prime attribute
3) Not part of C.K that is non prime.

So here D is dependent Attribute
here using $AB \rightarrow D$ Finding.
but in case of C it is Like $B \rightarrow C$.
So he C is dependent but it is not dependent whole C.K. it is depend on part of C.K. So this kind of dependency
$B \rightarrow C$ (known as partial Dependency)

→ ~~Persila~~ Partial dependency :- When Non prime is dependent on part of C.K rather then depending on whole C.K.

→ 2NF :- if table is 2NF then it must be in 1NF.
→ it must be independant on partial depedancy
(partial dependancy nahi honi chahiye)
→ if it is contain partial dependancy then we can say it is not in 2NF.

What is actual Problem ~~Wh~~ With this partial dependancy. you have ?

⇒ We know a primary key value can not be a Null.
⇒ but here you see A & B. both a Primary key
(Primary key ki ~~total~~ value Null nahi ho sakti but kya ek value Null ho sakti he. Yes)

Now

A	B	
1	-	✓
-	2	✓
-	-	X
3	4	✓

→ Out of 4 Cases which values are permissible you see. C.K is Not null A have value B have value, that is ok, 3 case Not ok

→ S. key Null nahi ho sakta So sare value Null nahi ho sakte.

→ AB ka value Null nahi ho sakta & in case of AB → D, D have advantage then is A Null or B Null still we can find D.

→ but in case of C. it is depend on B if B becomes Null run time you can not find C. and you promise then B is part of C.K & you can find every thing Now it becomes wrong

→ C is Non prime C ko pure C.K pe depend hona chahiye that but it is only depend on B.

→ And promise you did will not hold.

→ Now this is the practical problem ~~with~~ with partial dependency.

R(A B C D) Now How to translate in 2NF

⇒ R_1(A B D) (First table i have always key) (those value we put which is depend on entire C.K)

↓

Non partially dependency.

R_2(B C)

i have decompose R in two table R_1 & R_2 & Now there is no problem with Non Prime Attribute.

Date
Page

in 1NF there is partial dependency but for 2NF we have to remove that.

* 3NF

→ When table is already 2NF then How to convert in 3NF.

→ if table is in already in 2NF that means there is no concept of partial dependency.

→ from 2N to 3N. you need to deal with transitive dependency.

→ R(ABC)

A→B, B→C.

key A^+ = ABC.

B→C

(Transitive dependency)

R_1(BC)

R_2(AB)

A	B	C
a	1	x
b	1	x
c	1	x
d	2	y
e	2	y
f	3	z
g	3	z

A

A	B
a	1
b	1
c	1
d	2
e	2
f	3
g	3

B.

B	C
1	x
2	y
3	z

Transitive Dependency :- A F.D from $\alpha \to \beta$ called transitive if α, β Non prime.

3NF:- A relation R. is in 3NF if it is in 2NF & there is no transitive dependency.

Every dependency from $\alpha \to \beta$.
(i) either α is super key
(ii) or β is a prime attribute.

P.D : P→NP.
TD : NP→NP.
S.K → β.
$\alpha \to \beta$(prime)

Date
Page

Decomposition in 3NF.

1. R(ABCDE)

AB → C

B → D

D → E.

* R(ABCDE)

× A → B

B → E $(AC)^+ = ABCDE$.

(s.k) C → D.

α → β (Prime)

No. 3 NF.

* ABCDE
  - R1 (ABE)
    - R11 (AB)
    - R12 (BE)
  - R2 (CD)
  - R3 (AC)

2. R(ABCDEFGHIJ)

AB → C.

× A → DE. (AB).

B → F.

F → GH

D → IJ.

- R1 (ADEIJ)
  - R11 (ADE)
  - R12 (DIJ)
- R2 (BFGH)
  - R21 (BF)
  - R22 (FGH)
- R3 (ABC)

How to Normalize or decompose a Relation into 2NF.

Date
Page

* **2NF :- Example :-**

(1) R(ABCDE)

AB → C (PD)

D → E (PD)

R(ABCDE)

→ $(ABD)^+$ = ABCDE. = C.K. = R.

C & E Non prime.

⇒ No in 2NF. Now Convert in 2NF.

- R_1 (ABC)
- R_2 (DE)
- R_3 (ABD)

(2) R(ABCDE)

A → B (PD)

B → E

C → D (PD)

R (ABCDE)

$(AC)^+$ - ABECD. = C.K. = R

- R_1 (ABE) (This is transitivity but Now only check PD)
- R_2 (CD)
- R_3 (AC)

③ R(ABCDEFGHIJ)

(PD) AB → C

(PD) AD → GH

(PD) BD → EF

(PD) A → I

H → J.

R(ABCDEFGHIJ)

$(ABD)^+$ = ABCEFGHIJ = R = C.K.

R1 (ABC)

R2 (AI)

R3 (ADGHJ) (Again transitive)

R4 (BDEF)

R5 (ABD)

3NF

3) R(ABCDE)

AB → C $(AB)^+$ = ABCDE

B → D.

D → E.

R1 (BDE)

R11 (BD)

R12 (DE)

R2 (ABC)