Date:
Page:

∴ B cannot be a C.K.

&

$C^+ = CD$

$D^+ = D$.

Prime att. = {A}

Non prime att. = {B, C, D}.

∴ only A is C.K.

$\boxed{CK = \{A\}}$ Ans.

Note! →

$(AB)^+ = ABCD$ (AB → itself, B → C, C → D.)

Here,

AB can be a C.K.

But,

It is not a C.K.

bcz

C.K. is always minimal

&

in AB only A is C.K.

& rest is super key (S.K.)

$\underbrace{A\,B}_{\text{Super key}}$

(A ke saath adde any thing, becomes S.K.)

∴ AB is a Super key.

Ex! →

R(ABCD).

FD = {A → B, B → C, C → D, D → A}.

$A^+ = ABCD$

$B^+ = BCDA$

$C^+ = CDAB$

$D^+ = DABC$

∴ C.K. = {A, B, C, D}

all 4.

prime Attribute :- are attribute which is used in making of the C.K.

So, prime att. = {A, B, C, D} . { bcz, all of are C.K. }

& Non-prime att. = { ϕ } . → NULL .

Q :- R (ABCDE).

FD = {A → B, BC → D, E → C, D → A } .

⇒ Now, we have to check that which attribute are coming on the right side. bcz, Attributes on the Right side, वो सब determine हो रहे हैं।

= BDCA (E नहीं आया).

E = BDCAE

Note :- Each & Every candidate key must contain E. bcz, E if present on left side, then only it is written in right side also.

जो Att. Right Side में नहीं आ रहा। यानि वो left side में होना ही चाहिए। वो candidate key बनाने में use होगा ही होगा।

Now,

$E^+ = EC$ → (E alone is not Candidate key but, it is used in making C.K.)

Now, Start :- With A :-

$AE^+ = ABECD$ ∴ A ∈ C.K.

$BE^+ = BECDA$
$CE^+ = CE$ ×
$DE^+ = DEABC$

$\boxed{C.K. : \{AE, BE, DE\}}$ Ans

* Trick! - First, we got AE as C.K. So, Check either (A or E) on the right side of FD.

FD = {A → B, BC → D, E → C, D → A}.

So,
directly DE is also becomes the C.K.,
now check # D, it is depend on 2. So, check ~~them~~ both

prime Att. = {A, B, D, E}
non-prime att. = {C}.

(used in making C.K.). Ans.

× ×

(23) Functional Dependency! → (F.D.)

is the method which describes the rela'ship b/w the attributes.

X → Y

Determinant → X; Y → Dependent Att.

X determines Y (or)
Y is determined by X.

Ex!-

S-id		Sname
1	→	Ranjit
2	→	Ranjit

→ valid. ∵ These 2 are diff.

Ex!-

1	→	Ranjit
1	→	Ranjit

Same Student

Valid case

Ex:-
1 → Ranjit
2 → Varun } Valid.

Ex:-
1 → Ranjit
1 → Varun. } Not Valid.

(4) F. D. are of 2 types:→

1.) Trivial F.D.
2.) Non - Trivial F.D.

1.) Trivial F.D.:→

If $x \to y$

then,

y is subset of x.

These Trivial F.D. are valid. (Always True).

Ex:- $\underset{x}{\text{Sid}} \to \underset{y}{\text{Sid}}$ ✓.

NOTE:→ $x \to y$

L.HS ∩ RHS ≠ ϕ (Never NULL).

Ex:- Sid Sname → Sid.

∩ Sid. ✓ valid.

2.) Non-Trivial F.D.:→

If $x \to y$

then,

y is not a subset of x.

i.e. $\boxed{X \cap Y = \phi}$ (NULL)

ex! -

Sid → S.name
Sid → phone no.
Sid → Loca"

Now, In this we have to check cases. & find which is valid or not.

Properties of F.D! →

1.) **Reflexivity:** If Y is subset of X, then Trivial

$X \to Y$. (Sid → Sid).

2.) **Augmentation:** If $X \to Y$, then

$XZ \to YZ$

Sid → Sname
Sid phone → Sname phone.

3.) Transitive: If

$X \to Y$ & $Y \to Z$

then,

$X \to Z$

Sid → Sname & Sname → City
Sid → City.

4.) Union! - If $X \to Y$ & $X \to Z$

then $X \to YZ$

5.) **Decomposition!** If

$X \rightarrow YZ$ then

$X \rightarrow Y$ and $X \rightarrow Z$.

But,

$XY \rightarrow Z$ ~~$X \rightarrow Z$ & $Y \rightarrow Z$~~ ✗

6.) **Pseudo Transitive!**

If

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

$WX \rightarrow Z$

7.) **Composition!** -

If $X \rightarrow Y$ & $Z \rightarrow W$

then

$XZ \rightarrow YW$.

---

24. 2nd Normal Form ! → (2nd NF).

2 Rules

→ Table or Relaⁿ must be in 1st NF.

→ All the non-prime attributes should be fully functional dependent on Candidate key (C.K. or CP.K.)

(or)

[There should be no partial dependency in the Relaⁿ].

Ex:-

Part of CK ← Non prime

AB
C.K.

A → C
(a part) (non-prime)

If it's partial depend → not 2nd NF

Ex! →

Customer

Customer-Id	Store-Id	Locaⁿ
1	1	Delhi
1	3	Mumbai
2	1	Delhi
3	2	Bangalore
4	3	Mumbai

C.K. : Customer-Id Store Id

Prime attributes: C-Id
Store-Id.

Non prime : locaⁿ

Here, locaⁿ is only depend on Store-Id.
i.e. partial dependency.

bcz,
(to be in 2nd NF it should depend on the both C-id & S-id (bcz both are C.K.)).

Now →

Convert it into 2nd NF : →

Here, we make 2 Table.

C-Id	Store-Id
1	1
1	3
2	1
3	2
4	3

Store-Id	Locaⁿ
1	Delhi
2	Bangalore
3	Mumbai

(here, it is fully dependent bcz only 1 C.K. is here).

2nd NF

Q:- R (ABCDEF)

FD: {C → F, E → A, EC → D, A → B}

Solⁿ:- [CK: ?]

First, check Right hand side

Step 1:- F A D B

(means Now, these att are determined by some of the values.)

So,

[On LHS, there must be CE].

(C.K. जो भी बनेगी, उसमें CE तो होगा ही)

CE = FADB

Now:

EC^+ = ECFADB (all 6 are present)

∴, EC is C.K.

Now, use Trick

Either E or C must be present at the RHS of any F.D

but

not, neither E nor C, no one is present.

So,

there is only 1 C.K. in this table.

i.e,

[C.K. = {EC}]

Step 1 is comp. here,

after finding C.K. = {EC}

Proof:- check

A^+ = AB
B^+ = B
C^+ = CF

∴, proved.

proper subset is always less than a set.

$X \subset XY$ → proper subset
$X \subseteq XY$ → subset

Date:
Page:

Step 2 :- prime Attributes : {E, C}

non-prime attributes : {A, B, D, F}.

Step 3 :- C.K. = {EC}

What is proper subset of EC
↳ Either 'E' or 'C'.

Now check :-

F.D. = {C → F, E → A, EC → D, A → B}.

Rule

~~(Either proper subset of C.K.)~~

for partial dependency i.e., not in 2nd NF:
- Check on LHS → Either 'E' or 'C' [redacted]).
  $\boxed{\text{AND}}$
- [redacted] check on RHS → whether it is non-prime att.

We get,

C → F
↳ partial dependency.

So,

$\boxed{\text{Table is not in 2nd NF}}$ Ans.

C → F	{PD}	partial
E → A	{PD}	
EC → D	{FD}	fully.
A → B	{FD}.	

→ 1 भी P.D. मिलती है, Table is not in 2nd NF.

25. 3rd NF :→

- Table or Rel^n must be in 2nd NF.

&

→ There should be no transitive dependency in table.

Non prime as Non-unique
prime as unique.

Date :
Page:

→ not sufficient cond'n.

(NPA)

* Mean, Non-prime att. को कोई Non-prime att. determine नहीं कर सकता।

(C.K. & Prime att. (P.A.) तो कर सकते हैं determine NPA को।).

Ex:-

Rollno.	State	City
1	punjab	Mohali
2	Haryana	Ambala
3	punjab	Mohali
4	haryana	Ambala
5	Bihar	patna

C.K. = {Roll no.}
F.D → {Roll no → state,
State → city}

⇒ PA = {Roll no.}

⇒ NPA = {State, City}.

So,

(NPA → NPA).

here, Roll → State and State → City.

It is Transitive dependency & we don't want that.

So,

It is not in 3rd NF.

Ex:-

R (ABCD)

FD : {AB → C, C → D}

⇒ C.K. = {AB}

PA = {A, B}

NPA = {C, D}

Transitive

AB^+ = ABCD.

C → D
NPA NPA

It is not in 3rd NF.

C.K. + any thing = S.K.

Super key

Date:
Page:

Ex: R (ABCD)
FD: (AB → CD , D → A).

Soln: C.K. : {AB, DB}.
PA : {A, B, D}.
NPA : {C}.

(B not on RHS)

$B^+ = B$

$\boxed{AB^+ = ABCD}$

Now, A on RHS.
so,
$\boxed{DB^+ = DBAC}$
is also C.K.

Imp: Now, for each F.D.

Then, Table is in 3rd NF.

LHS → C.K. or S.K.
OR
RHS → is a P.A.

→ check only 1 bcz OR.

FD: (AB → CD, D → A).

∴ Table is in 3rd NF.

bcz, (NPA → NPA) is not present here.

26. BCNF. (Boyce Codd Normal Form) :-

⇒ also called as special case of 3rd NF.

Student

Roll-no	Name	Voter-Id	Age
1	Ravi	K0123	20
2	Varun	M034	21
3	Ravi	K786	23
4	Rahul	D286	21

Table is in 3rd NF already.

C.K. = {Roll no, voter-Id}.

F.D. : = { Roll no → name
Roll no → voterid
voterid → age
voterid → Roll no. }

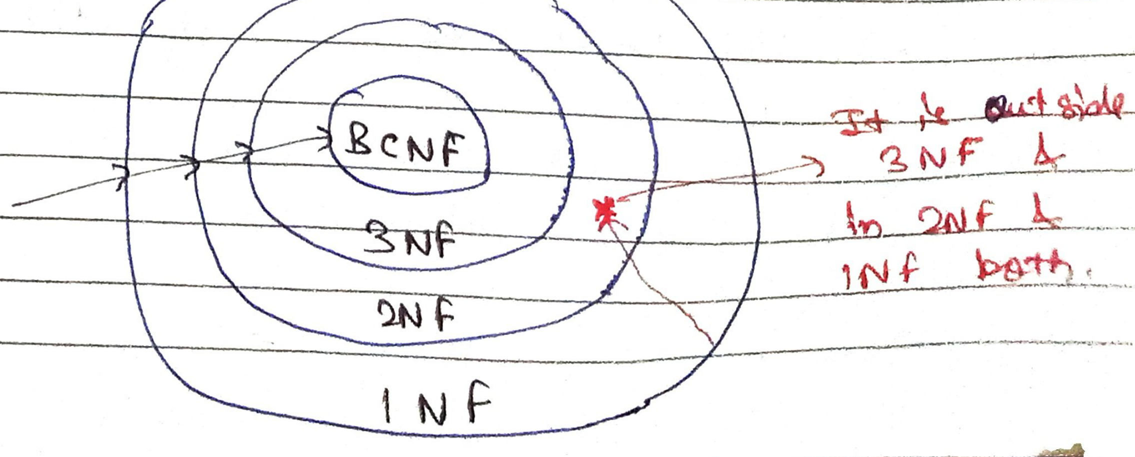
Note! → LHS of each FD should be C.K. or S.K.

Here, 3NF से की (OR) वाली condiⁿ भी हटा दी, जिसमें RHS के P.A. होने से भी काम चल जाता था).

Here,
we only want C.K. or S.K. in LHS. & RHS से कोई लेना देना नहीं।

Solⁿ! so, check all the f.D. one-by-one.
& in all the 4 f.D, the LHS is C.K.
so, It is in BCNF form.

Compare! - 1NF, 2NF, 3NF, BCNF ! →



1NF.
2NF = 1NF + condi^n
3NF = 2NF + condi^n
BCNF = 3NF + condi^n

27 Lossless & Lossy Join Decomposi^n :→

We normalise table → or we decompose table into 1NF, ... forms.

R

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

→ R_1 (AB)
→ R_2 (BC).

We divide this table into R_1 & R_2
& B is common in both the Table.

R_1

A	B
1	2
2	2
3	3

R_2

B	C
2	1
2	2
3	2

Q) find the value of C if the value of A='1'?
→ Now, for this we have to join R_1 & R_2 tables.

So,

Select $R_2.C$ from R_2 Natural Join R_1 where $R_1.A$ = '1'.

(1st row को Multiply all rows से होगा of Table 2).

Cross product :- If R_1 has x rows & R_2 has y rows then their join has x.y rows.

&

Condⁿ :- Common ~~element~~ colᵐ of both Tables (R_1 & R_2), here (B) has the same value in join Table.

&

Natural Join = Cross product + condⁿ.

Now,

R_1 A	R_1 B	R_2 B	R_2 C	
1	2	2	1	✓
1	2	2	2	✓
~~1~~	~~2~~	~~3~~	~~2~~	
2	2	2	1	✓
2	2	2	2	✓
~~2~~	~~2~~	~~3~~	~~2~~	
~~3~~	~~3~~	~~2~~	~~1~~	
~~3~~	~~3~~	~~2~~	~~2~~	
3	3	3	2	✓

Now.

R'

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

→ table after Joining.

(take). Spurious tuples.

* In original Table (R), we have only 3 tuples (rows).
but,
after Joining, in R', we have 5 tuples.

(कमी)
It is a flaw. It is called the Lossy Decomposition.

* Why lossy :-
Here, we get 2 extra rows, then, why lossy.
Here,
we don't talk about rows. We call it lossy because of inconsistency. There is a problem in Database.

⇒ In original, for A=1, C = 1.
but
In join table, for A=1 < C = 1, C = 2 }.

(#) Why we get Error :- (2 tuples more)

⇒ bcz, we take B as common in both Table.
but,

Criteria for Common :- Common attribute should be C.K. or S.K. of either R_1 or R_2 or both.
So, we have to C.K. or S.K. of original Table.

5) ▷ B has duplicacy in Table, We have to choose attri. 'A' for Right Ans.

bcz, A is unique. <1, 2, 3>.

R_1 (<u>A</u> B)
R_2 (<u>A</u> C).

→ We get 3 tuples also in joining table.

Condi^n for lossless Joining Decompos^n! →

1.) $R_1 \cup R_2 = R$.

~~AB ∪ AC = ABC.~~

2.) $R_1 \cap R_2 \neq \phi$

AB ∩ AC

A ≠ ϕ

3.) <u>R_1 C.K.</u> (or) <u>R_2 C.K.</u> (or) <u>Both</u> (C.K. of original Table)

<u>To take common attribute</u>

(28.) All normal forms with Real life Examples →

~~6 forms here~~

1st NF	2nd NF	3rd NF.
→ No multivalued attribute. → only single valued.	→ In 1st NF + → No partial Dependency. * only full dependency. $\boxed{AB \rightarrow C}$ (If A & B are 2 head of a Co., then both will use the Emplo. C)	→ In 2nd NF. + → No Transitive dependency → No, NP A. should determine N.P.A.

BCNF	4th N.F.	5th N.F.
→ In 3rd NF	→ In BCNF	→ In 4th NF
+	+	+
→ LHS must be C.K. or S.K.	→ No multivalued Dependency.	→ lossless Decomposition.
$X \rightarrow Y$	$X \rightarrow\rightarrow Y$	

4th N.F.:

Varun → 3 Phone no.
→ 3 Mail Id.

(Varun depends on multiple att. i.e. phone & mail.)

table

Varun	M_1	E_1
Varun	M_1	E_2
|	M_1	E_3
|	M_2	E_1
|	M_2	E_2
|	|	

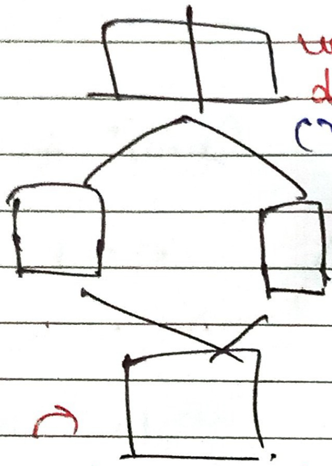
→ Very long.
→ We also don't able to form a key.

So, Make 2 table.

[Varun — phone] [Varun — Mail]

(Now, no multivalued dependency).

5th N.F.:



vertically divide (rows same)

May be xtra tuples come. So, make C.K. as common attribute in both tables.

29. Minimal Cover :- (Irreducible) :-

Q:- For the following functional dependencies, find the correct Minimal Cover →

$\{A \rightarrow B,\ C \rightarrow B,\ D \rightarrow ABC,\ AC \rightarrow D\}$.

a) A → B , C → B , D → A , AC → D .
b) A → B , C → B , D → C , AC → D .
c) A → BC , D → CA , AC → D .
d) A → B , C → B , D → AC , AC → D Ans

Solⁿ:- Our RHS in F.D. must be single.

Step 1 {A → B , C → B , D → ABC , AC → D}.

By decompos" prop, separate them.

A → B , C → B , D → A , D → B , D → C , AC → D .

Step 2 Remove the Redundant F.D. :-

A → B , C → B , D → A , D → B , D → C , AC → D .

let, ✗ ✗ ✓ ✓ ✓ ✓

→ let, A → B को हटा दिया , Now check the closure of A,

$\boxed{A^+ = A}$ (not all +).

So, A → B is not Redundant. We can't remove it.

→ & Same check this for every F.D. :→

D → B is redundant,

bcz

let, D → B ✗ then, $D^+ = DABC$ (all +).

So, ↓

$\boxed{\text{remove, } D \to B}$

अब जब आगे check करेंगे, तो इसको नहीं लेंगे, इसको तो यहाँ ही remove कर दिया।

Now, for $AC \to D$ ×

$$AC^+ = ACB$$

So, also include $AC \to D$ ✓

Now, we get

$\{A \to B, C \to B, D \to A, D \to C, AC \to D\}$.

Step 3. Now, we only want 1 Att. in LHS.

Here, $AC \to D$

Now, check by remove A & then check closure of C

$$C^+ = CB$$

• अगर C^+ में हमें 'A' भी मिल जाता, तो हम A को हटा सकते थे। लेकिन नहीं मिला तो,

$AC \to D$ ही रहेगा।

&

Same check for A, by removing C

&

$$A^+ = AB.$$

So, can't remove C.

So, $AC \to D$ can't be reduced.

So;

$\{A \to B, C \to B, \boxed{D \to A, D \to C}, AC \to D\}$.

compose

So,

$\boxed{\{A \to B, C \to B, D \to AC, AC \to D\}}$. Ans:

30 Question on Normalization :→

Q! R (ABCDEF), check the highest normal form ?

F.D. ! {AB→C, C→DE, E→F, F→A}.

Sol^n! Step1: Find all CKs in Rela^n !→

By closure Method !→

(B is not on RHS.) So, B compulso^n.

$B^+ = B$, $A^+ = A$

$AB^+ = ABCDEF$ (all +)

∴ $\boxed{\text{(AB is C.K.)}}$

Now, check ∀ A on RHS

we get, F→A

∴ $\boxed{\text{(FB is also CK.)}}$

Now, check F on RHS !→

E→F

So, $\boxed{\text{EB is also C.K.}}$

Now, check E on RHS.

C→DE

So, $\boxed{\text{CB is also C.K.}}$

Now check ∀ C on RHS.

AB → C.

&

AB is already our C.K.
So, we get all C.K.

$\boxed{C.K. = \{AB, FB, EB, CB\}}$ ∴ 4 CK.

Step 2 :- Write all prime att. & NPA :-

P.A. = {A, B, C, E, F}.

NPA = {D}.

Step 3 :- Now, check FD :-

{AB → C, C → DE, E → F, F → A}.

Now,
Check 1-by-1.

Highest NF :- 1NF, 2NF, 3NF, BCNF,
Redundancy decreases.

* mean, when table is in BCNF, then redundancy is lowest. & in 1NF redundancy is highest.

* So, to check highest NF, we start from BCNF :-

→ In BCNF, we know, all LHS of all FD's should be CK or SK.

{ AB → C, | C → DE, | E → F, | F → A }
✓ | × | × | ×

↓

It is not in BCNF form.

→ Now, 3NF :→

Check :→ transitive dependency (NPA → NPA) ×

LHS → C.K. or S.K. OR RHS → is a P.A.	→ then it is 3NF.

Now, 1st Condi'n is already checked in BCNF.
So, here check only 2nd condi'n that whether RHS is P.A. or not.

	{ AB → C	C → DE	E → F	F → A. }
BCNF	✓	×	×	×
3NF	✓	×	✓	✓

∴ not in 3NF :→

→ Now, 2NF :→
Same thing → if there is already a tick in 3NF, then we don't have to check that for 2NF. वो already 2NF में होगा ही।
Check only for 'x' tick.