Normal Forms

Normalization is the property of minimizing the redundancy & minimizing the insertion, deletion and updation anomalies.

These would include two properties :-

(1) The lossless join or non additive join property which generates that the spurious tuples does not occur.

(2) Dependencies presentation which ensure that each FD is presented in some individual reln resulting after decomposition.

Normal forms are used to reduce redundancy in table. Normalization rules are divided into following normal forms.

- → First Normal form
- → Second " "
- → Third " "
- → BCNF
- → Fourth " "
- → Fifth " "

(1) First Normal Form (1NF)

A table is in 1NF if the values in the domain of each attribute of the relⁿ are atomic ie only one value is associated with each attribute. It disallow multivalued and composite attribute.

For example Department table is given –

Dname	Dno	Dmngr	Dloc
CS	1	AS	Delhi, Kanpur
IT	2	AP	Delhi
MCA	3	MS	Gzb

disallow

There are three solutions to convert it into 1NF –

(i) Expand Key as –

Dname	Dno	Dmngr	Dloc
CS	1	AS	Delhi
CS	1	AS	Kanpur
IT	2	AP	Delhi
MCA	3	MS	Gzb

Now key is (Dno, Dloc)

Disadvantage –
→ Redundancy.

(1) First Normal Form (1NF)

A table is in 1NF if the values in the domain of each attribute of the relⁿ are atomic ie only one value is associated with each attribute. It disallow multivalued and composite attribute.

For example Department table is given —

Dname	Dno	Dmngr	Dloc
CS	1	AS	Delhi, Kanpur
IT	2	AP	Delhi
MCA	3	MS	Gzb

disallow

There are three solutions to convert it into 1NF —

(i) Expand Key as —

Dname	Dno	Dmngr	Dloc
CS	1	AS	Delhi
CS	1	AS	Kanpur
IT	2	AP	Delhi
MCA	3	MS	Gzb

Now key is (Dno, Dloc)

Disadvantage —
→ Redundancy.

(ii) Add extra column –

Dname	Dno	Dmngr	Dloc 1	Dloc 2
CS	1	AS	Delhi	Kanpur
IT	2	AP	Delhi	NULL
MCA	3	MS	Gzb	NULL

Disadvantage – Increasing NULL values.

(iii) Decomposition of table (Best Solution)

Dname	Dno	Mngr
CS	1	AS
IT	2	AP
MCA	3	MS

Dno	Dloc
1	Delhi
1	Kanpur
2	Delhi
3	Gzb

② 2nd Normal Form (2 NF)

To be in 2NF, a relⁿ must be in –

- → First Normal form
- → Relation must not contain any partial dependency ie no nonprime attribute (attribute which are not the part of any candidate key) is dependent on proper subset of any candidate key

of table.

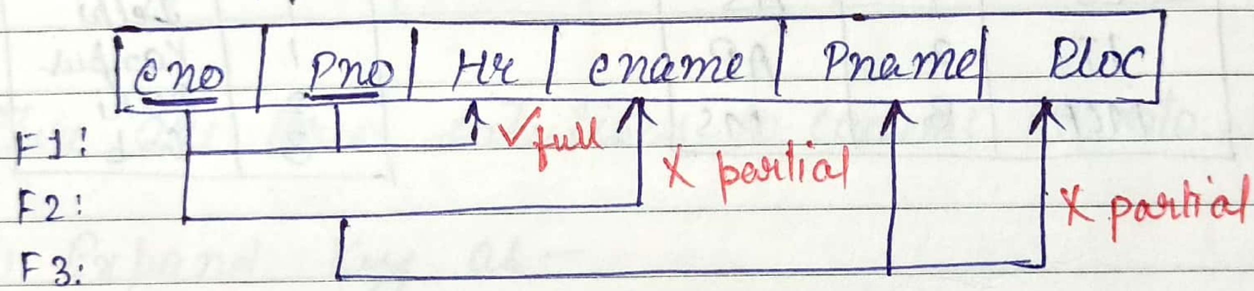
OR

Every non prime attribute is fully functional dependent on key of R.

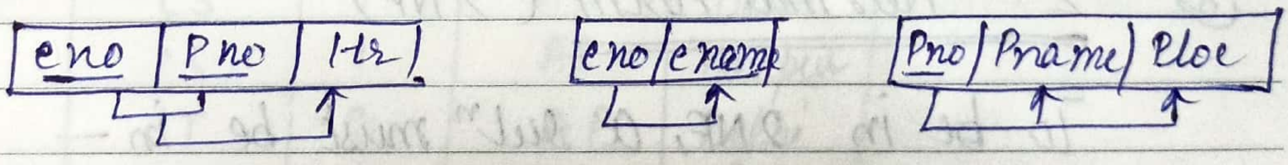
Full Functional Dependency :-

$X \rightarrow Y$ is full FD if removal of any attribute A from X means that dependency does not hold any more.

For example :-



Decompose :



Remedy : Decompose & set up new rel^n for each partial dependency (partial key with its dependent). Make sure to keep a rel^n with original key & attribute that fully fnl dependent on it.

(3) Third Normal Form (3NF)

Transitive Dependency —

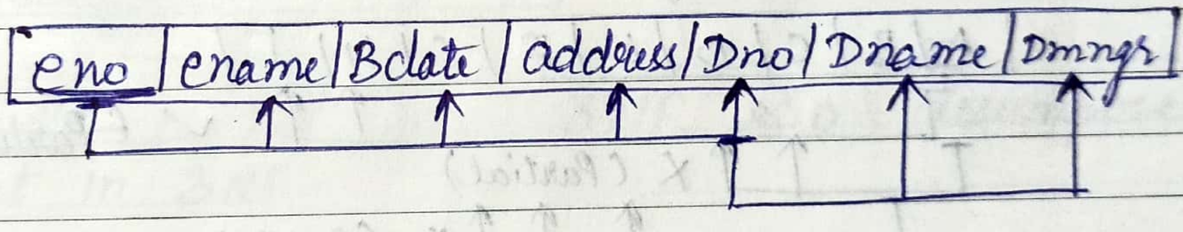
A FD $X \rightarrow Y$ is transitive dependency if there is a set of attributes Z that is neither a candidate key nor a subset of any key of R and both $X \rightarrow Z$ and $Z \rightarrow Y$ hold.

Rel^n R is in 3NF if —

→ It is in 2NF.

→ No transitive dependency. (No Nonprime attribute transitively dependent.)

eg:-



Here $eno \rightarrow Dno$ and $Dno \rightarrow Dname, Dmngr$ so Dname, Dmngr is transitively dependent on eno and Dno is not a key and not a subset of key.

Remedy :-

Decompose and set up relation that includes the nonkey attributes that functionaly determine non key attribute.

eg:- R(ABCDE)

$AB \to C$

$D \to E$

$ABD^+ = R$ so key is ABD

⇓ decompose

A	B	C

D	E

eg:- $R = (A, B, C, D, E, F, G, H, I)$

$A \to CD$, $B \to EFG$, $AB \to HI$, $F \to G$

{AB} is key.

A	B	C	D	E	F	G	H	I

✓ (~~Partial dependency~~)

× (Partial)

× (Partial)

✓

⇒

A	C	D

A	B	H	I

B	E	F	G

Now All tables are in 2NF.

eg:- R = (Name, Phno, course, major, prof, grade, major_ele)

Name → Phno, major (Partial dep.)
course → Prof (" ")
major → major_elective
Name, course → grade

Key is {Name, course}

2 NF Decomposition :-

R_1 [name | Phno | major | major_ele] R_2 [course | Prof]

R_3 [name | course | grade]

Now R_1 is not in 3NF so decompose it convert in 3NF.

3 NF Decomposition :-

1 [name | Phno | major | ~~major_ele~~]   [major | major_ele]

[course | Prof]   [name | course | grade]

Another Approach

There is one another approach to normalize in 3NF. First of all understand what is non trivial dep.

Non Trivial Dependency :-

X → Y : If Y is not a subset of X then This FD is called non trivial FD.

Relation is in 3NF if, every non trivial dependency of reln R follow one of the condition :-

For every : X → Y

→ X is a superkey (OR)
→ Y is prime attribute

④ BCNF (Boyce - Codd Normal form) :-

R is in BCNF if for all fnl dependency - X → Y , X is a super key (ie X holds candidate key).

eg:-

sid	sname	sub	grade

{sid, sub} → grade

{sname, sub} → grade
sname → sid
sid → sname

R is in which highest form?

Soln :- There are two keys in Reln (sid, sub) & (sname, sub)
So grade is the only non prime attribute.
in FD 1,2 left is superkey and in 3,4 right is prime attribute so
R is in 3NF but not in BCNF.

To decompose in BCNF –

sid	sub	grade

sid	sname

eg:-

A	B	C	D	E

A → B, C → D

Normalize in BCNF.

Soln :- (A,C,E) is key. Both FD create problem in BCNF. so firstly remove A → B

⇒

A	C	E	D

A	B

then ~~se~~ C → D

⇒

A	C	E

C	D

A	B

(5) Fourth Normal Form (4NF) :-

A relation is in 4NF if it is in –

→ Relation is in BCNF.
→ Contain No MVD (multivalued dependencies)

Multivalued Dependencies (MVDs)

Let R be a relⁿ schema and X and Y both are subset of R.

$X \twoheadrightarrow Y$ (x multidetermines y) if in any relⁿ r(R) for all pair of tuple t_1 & t_2 in r such that $t_1[X] = t_2[X]$, there exist some tuple t_3 & t_4 in r such that :

$$t_1(x) = t_2(x) = t_3(x) = t_4(x)$$

$$t_1(y) = t_3(y) \; \& \; t_2(y) = t_4(y)$$

$$t_2(z) = t_3(z) \; \& \; t_1(z) = t_4(z)$$

where $z = R - (X \cup Y)$

Trivial & Non trivial MVD :- A MVD $X \rightarrow Y$ in R is called trivial
if (a) Y is subset of X
or (b) $X \cup Y = R$.
And if it does not satisfies any of the condⁿ then it is called non trivial MVD.

eg:-

	Faculty	Sub	Committee
t_1	John	DBMS	Placement
t_2	John	N/W	Placement
t_3	John	DBMS	Registration
t_4	John	N/W	Registration

Decomposition - (Faculty, sub) & (Faculty, committee)

⑥ 5th Normal Form or Project Join Normal Form

The lossless join property refers to the facts that whenever we decompose reln using normalization we can rejoin the reln to produce the original reln.

A reln R is in 5NF iff for all

→ R is in 4NF
→ Not having any join dependency (& joining should be lossless)

A relation that has ~~no~~ a join dependency can not be devided into two (or more) reln such that the resulting table can be combined to form the original table.

Dependency Preserving Decomposition

Decomposition of R into R_1 and R_2 is a dependency preserving if closure of functional dependencies after decomposition is same as closure of FDs before decomposition.

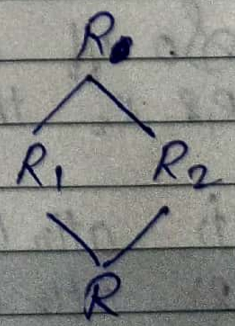
A simple way is to just check whether we can derive all the original FD from FD present after decomposition.

Lossless Join / Nonadditive Decomposition

This property guarantees that the extra or less tuple generation problem does not occur after decomposition.

eg:- if R decomposed into R_1 & R_2:
For lossless decomposition R will be generated after combining R_1 & R_2.

A	B	C	D
1	a	p	x
2	b	q	y



Case 2:-
If R_1 (AB) & R_2 (CD)
then it is lossless because for combining these two we will have to perform cartesian, which will give extra rows. (because join not possible).

Case 3 :-

A	B	C
1	a	p
2	b	q
3	a	r

R_1

A	B
1	a
2	b
3	a

R_2

B	C
a	p
b	q
a	r

$R_1 \bowtie R_2$

A	B	C	
1	a	p	
1	a	r	→ extra
2	b	q	
3	a	p	→ extra
3	a	r	

extra tuple occur because common attribute should be distinct.

So iff a rel^n R is decomposed into two rel^n R_1 & R_2 then it will be lossless iff -

(1) $attr(R_1) \cup attr(R_2) = attr(R)$

(2) $attr(R_1) \cap attr(R_2) \neq \phi$

(3) $attr(R_1) \cap attr(R_2) \rightarrow attr(R_1)$

or

$attr(R_1) \cap attr(R_2) \rightarrow attr(R_2)$

Scanned with CamScanner