UNIT - IV

NORMALIZATION

Problems Of Redundancy:

a. Wastage of Storage Space

b. Anomalies

- Insert Anomalies
- Delete Anomailies
- Update Anomalies.

Normalization is used to reduce the redundancy

Functional Dependency: $R(x, y, z)$ $fD1: x \rightarrow y$

It is a constrain b/w two sets of attributes

A functional dependency is a constraion b/w two sets of attributes in relation R.

if $t_1[x] = t_2[x]$ then $t_1[y] = t_2[y]$

if $x \rightarrow y$

Inference Rules (or) Amstrongs Inference Rules

IR1: Reflexive if $y \subseteq x$, then $x \rightarrow y$

IR2: Augmentation if $x \rightarrow y$, then $xz \rightarrow yz$

IR3: Transitive if $x \rightarrow y$, $y \rightarrow z$ then $x \rightarrow z$

IR1, IR2, IR3 from a sound & complete set of inference

Decomposition, if $x \to yz$ then $x \to z$ & $x \to y$
(or) projection rule.

Union, if $x \to y$ & $x \to z$ then $x \to yz$
(or) additive rule.

By sound means for a given set of functional dependencies F on relation R, any functional dependency can infere from F by using IR1, IR2, IR3 holds in every relation state r(R). that satisfies the relation state F.

By complete means we can infere the complete set of all possible dependencies F.

Solve R(A, B, C, D, E) with following functional depen's

$F = \{ A \to B, CD \to E, E \to A, B \to D \}$

Specify all minimal key of R.

$A^+ = \{A, B, D\}$

$B^+ = \{B, D\}$

$C^+ = \{C\}$

$D^+ = \{D\}$

$E^+ = \{E, A, B, D\}$

$AB^+ = \{A, B, D\}$

$AC^+ = \{A, B, C, D, E\}$ ✓

$AD^+ = \{A, D, B\}$

$AE^+ = \{A, E, B, D\}$

$BC^+ = \{A, B, C, D, E\}$ ✓

$BD^+ = \{B, D\}$

$BE^+ = \{B, E, D, A\}$ ✗

$CD^+ = \{C, D, E, A, B\}$ ✓

$CE^+ = \{C, E, A, B, D\}$ ✓

$DE^+ = \{D, E, A, B\}$

R (A, B, C, D, E, F, G, H) with following FD's.

$F = \{ A \to BCD$ Specify all minimal keys

$AD \to E$

$EFG \to H$

$F \to GH \}$

$AB^+ = \{A, B, C, D, E\}$

$AC^+ = \{A, C, B, D, E\}$

$AD^+ = \{A, B, C, D, E\}$

$AE^+ = \{A, B, C, D, E\}$

✓ $AF^+ = \{A, B, C, D, F, G, H\}$

$A^+ = \{A, B, C, D, E\}$

$B^+ = \{B\}$

$C^+ = \{C\}$

$D^+ = \{D\}$

$E^+ = \{E\}$

$F^+ = \{F, G, H\}$

$G^+ = \{G\}$

$H^+ = \{H\}$

<u>Normalization</u>: Used to reduce data redundancy

Normalization is the process of organizing data in database.

Normalization is used to min the redundancy from a relation or set of relations. It also used to elemenate the undesirable charac like insertion, updation.

Normalization divides the larger table into the smaller table & links them using relationship

1st Normal Form:

It disallows composetition attributes, multivalue attributes, nested relations.

A relation R is in 1st NF if all the values are atomic.

1	credit	1234	{bangl, Hyd, delhi}
D.no	D.name	D.mgrId	D.location

X ↳ multivalued attribute

↓

D.no	D.name	D.mgrId

D.no	D.location

$$f : \{ D.no \to D.name,\ D.no \to D.mgrId,\ D.no \to D.location \}$$

As D.location is Multivalued attribute, dept is not in Normal form.
As D.location violates the rule so it is seperated from Dept Table.

2nd Normal Form:

1st nf ↑ R. ×

A relational scheme R is in 2nd N.F if every (non-frame) non-prime attribute A is fully functionally dependentn on key (R).

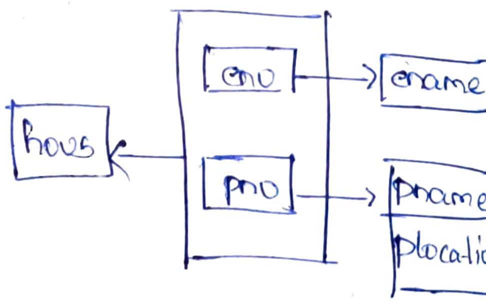
A fuctional Dependency $x \to y$ is a fully functional Dependency, if removal of any attribute a from x means that the dependency does not hold any more i.e.., for any attrible $A \in x$, $(x - \{A\})$ does not functionally determine y. A functional Dependency is a partial FD if some attributes
$a \to x$ can be removed from x & the dependency still holds

Ex:

emp-pro (eno, pno , ename , pname , plocation, hours)

key (eno, pno)

F = { eno → ename
pno → pname , plocation
eno, pno → hours }



Decompsed:

emp

eno	ename

eno → ename

proj

pno	pname	plocation

pno → pname, plocation

emp-pro (eno, pno, hours)

Properties Of decomposition:

A relational schema $e = \{A_1, A_2 \dots A_n\}$ includes all the attributes of the database

* A set of functional dependencies that should hold on attributes of 'e'
* Using functional dependencies (fd's) decompose R into a set of relational schema

$D = \{R_1, R_2 \dots R_n\}$; 'D' is called the Decompositio

Attribute Preservation Property:

Make sure that each attribute in 'R' will appear in atleast one relational schema in the decomposition so that no attributes are lost.

$$\boxed{\bigcup_{i=1}^{n} R_i = R}$$

Dependency Preservation Property:

Each functional dependency $x \to y$.

Specified in 'F' either appear direactly in one of the R_i (or) could be inforced from the dependencies that appear in 'R'.

$$(\pi_{R_1}(F) \cup \pi_{R_2}(F) \dots \cup \pi_{R_n}(F))^+ = F^+$$

π_{R_1} - projection of functional dependency in R_1

The projection of functional dependencies in decomposition D & their closure must be equal to the closure set of functional dependencies is relational R.

Lossless Join Property:

In ensures that no spurous (wrong) tuples are generated when a natural join operation is applied to the relations in decomposition.

→ A decomposition $D = \{R_1, R_2 \ldots R_n\}$ of 'R' has the lossless join property w.r.t the set fd's on R if for every relation state r(R) that satisfies f then the following holds

$$* \left(\pi_{R_1}(r), \pi_{R_2}(r) \ldots \pi_{R_n}(r)\right) = r.$$

1. Create a Matrix S, where rows are $R_1, R_2 \ldots R_n$ & columns are attributes of R.
2. Set $S(i,j) = b$ if $j \notin i$ &
   $S(i,j) = a$ if $j \in i$.
3. Repeat for each FD in F ∀ rows having 'a' in columns corresponding to 'x' if one of row is having 'a' for 'y' make others also 'a'.
4. If row is having all a's, then it is loss less else it is lossy

$R(A, B, C, D, E, F)$

$$F = \{ A \to BC,\ C \to A,\ D \to E,\ F \to A,\ E \to D \}$$

is decomposition of R into
$R_1(A, C, D)$
$R_2(B, C, D)$
$R_3(E, F, D)$

	A	B	C	D	E	F
R_1	a	~~b~~a	a	a	~~b~~a	b
R_2	b	~~a~~b	~~a~~b	a	~~b~~a	b
R_3	b	b	b	a	a	a

	A	B	C	D	E	F
R_1	a	~~b~~a	a	a	~~b~~a	b
R_2	~~b~~a	a	a	a	~~b~~a	b
R_3	b	b	b	a	a	a

(2) R = { Eno, ename, pno, pname, plocation, hours }

R_1 = { ename, plocation } = emp_locs

R_2 = emp_proj1 = { eno, pno, hours, pname, plocation }

F = { eno → ename,
pno → { pname, plocation },
{ eno, pno } → hours }

i:

	Eno	ename	pno	pname	plocation	hours
R_1	b	a	b	b	a	b
R_2	a	b	a	a	a	a

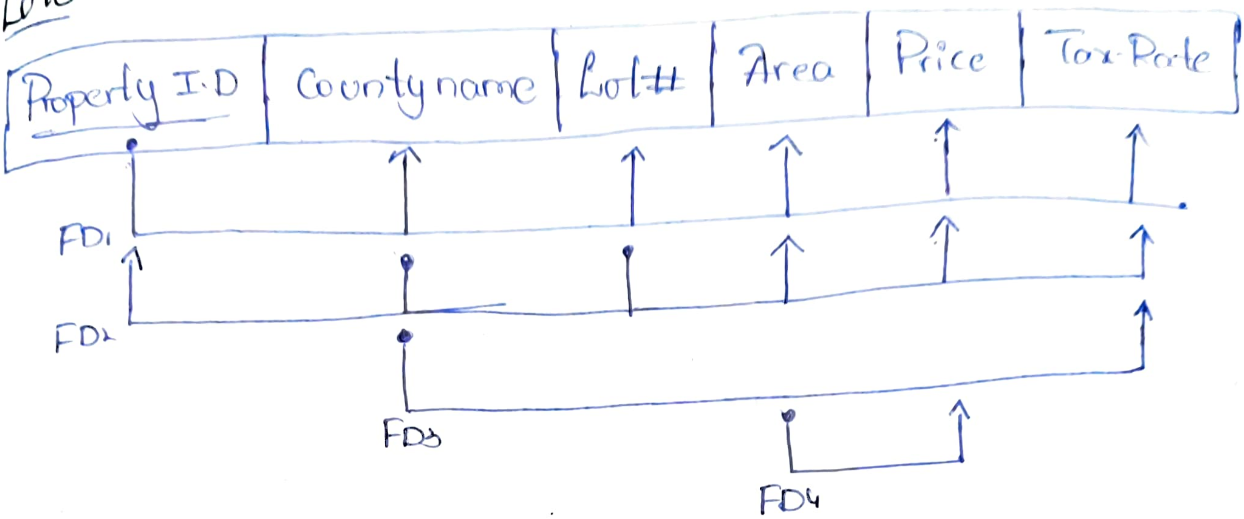
Lossy decomposition.
No single row consists of all a's so it is L.D

ii: R_1 = { Eno, ename }, R_2 = { pno, pname, plocation }
R_3 = { eno, pno, hours }

	Eno	ename	pno	pname	plocation	hours
R_1	a	a	b	b	b	~~a~~ b
R_2	b	b	a	a	a	b
R_3	a	~~b~~ a	a	~~b~~ a	~~b~~ a	a

Multi valued Dependencies:

LOTS:



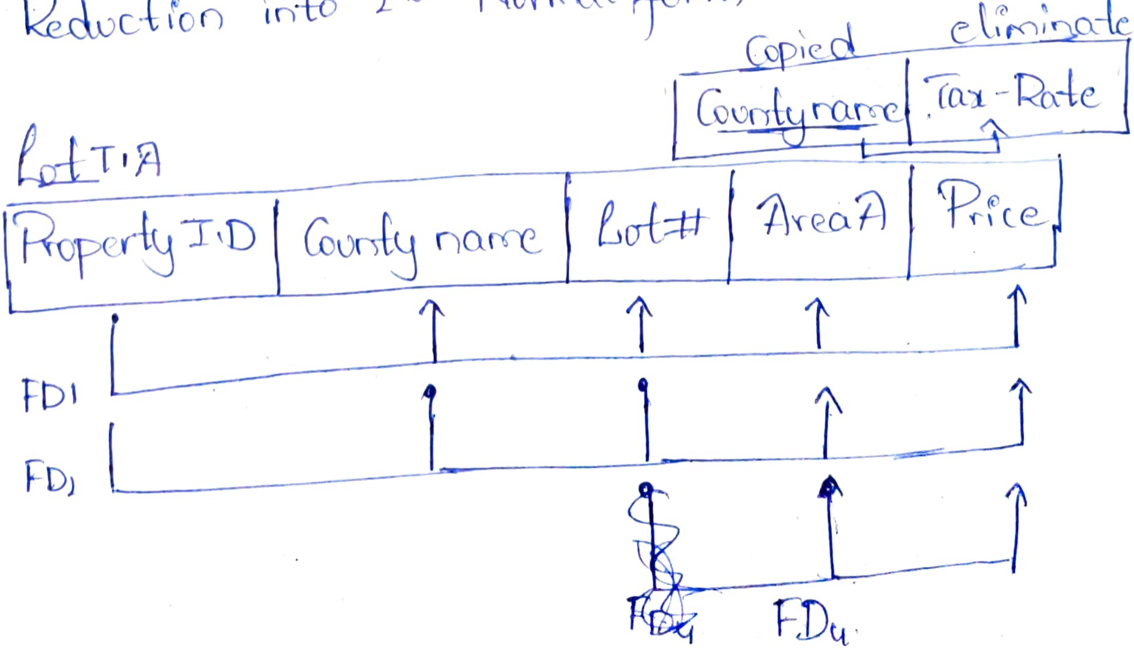
Candidate key:

C_1 = property I.D

C_2 = {County name, Lot#}

Clearly data is in 1st Normal form & not in 2nd N.F becoz of FD_3 (partial function)

Reduction into 2nd Normal form



(Again LoT1 is not in 3rd Normal form becoz of FD4.

(3rd N.F if $x \rightarrow y$.

(i) x is super key

(ii) y is a prime attribute).

To make it satisfy 3rd N.F

Lots1B

Property ID	County name	Lot#	Area

FD_1

FD_2

County-name	Tax-Rate

Area	Price

These are also in BCNF

(BCNF if $x \rightarrow y$)

(i) x is super key.

Trivial functional Dependency:

For a functional dependency $x \rightarrow y$ if $y \subseteq x$ then it is Trivial functional Dependency.

if $y \not\subseteq x$ then it is Non-Trivial functional Dependency.

3rd N.F

Def: A relational schema 'R' is in 3rd N.F if it is in 2nd N.F & no-nonprime attribute of 'R' is transitively is dependent on primary key

Def2:
A relational schema 'R' is in 3rd N.F if whenever a functional dependency $x \to A$ holds 'R' then either

(i) x is a super key of R

(or)

(ii) A is a prime attribute.

Ex: emp_dept (<u>eno</u>, ename, dob, address, Dno, Dname, Dmgrno)

eno → ename
eno → dob
eno → address
eno → Dno

dno → dname
dno → dmgrno.

In 2nd fD <u>dno</u> is not a super key as wall as dname, dmgrno is not a prime attributes

→ Dname, Dmgrno are transitively dependent on eno through a non-key attribute the number so emp_dept relation is not in 3rd Normal form.

So remove dname, dmgrno from this relational & keep it in a new relational dept.

To preserves fd copy dno & make it as a primary key in dept relation.

BCNF

Boyce-CODD Normal form:

A relational schema 'R' is in BCNF if whenever a non-trivial functional dependcy $x \rightarrow A$ holds in R then x is a super key of R.

All BCNF relations are in third normal form but the 3rd N.F relations may not neccessarily in B.C.N.F

4th N.F:

A multivalues fd $x \rightarrow\rightarrow y$ specified on Reln R

$Z = R - (x \cup y)$

- Where x, y are subsets of R under the condition

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

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

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

Trivial & Non-Trivial ~~Form~~ MVD:

$y \subseteq x$ as wells as $x \cup y = R$ - either of Them (or) Both - Trivial MVD

R(a,b,c)

$a, b \rightarrow\rightarrow c$

$x \cup y = a, b \cup c = R.$

if $y \not\subseteq x$ & $x \cup y \neq R$ then Non-Trivial MVD

4th Normal Form:

For every non-trivial MVD x is a Super Key

A reln schema R said to be in 4NF w.r.t set of dependencies (fd is MVD) for every non-trivial MVD ($\rightarrow\!\!\!\rightarrow$) is FT, x is super key for R

EMP

Ename	Pname	Dname
Smith	x	John
Smith	y	
Smith	x	
Smith	y	John

! The EMP reln with two MVD's Ename $\rightarrow\!\!\!\rightarrow$ Pname & Ename $\rightarrow\!\!\!\rightarrow$ Dname

emp (ename, pname, dname)

Ename $\rightarrow\!\!\!\rightarrow$ Pname

x y

* not superkey Non-trival.
* Union $\neq$ R

* Decompose emp reln into emp-proj & emp_dependents

→

ename	pname

ename	dname

Now MVD1 becomes trivial.

MVD2 : Ename $\rightarrow\!\!\!\rightarrow$ Dname

Join Dependency:

A join dependency (JD), denoted by JD(R_1, R_2, ... specified on relation schema R, specifies a constraint on the stats r(R)

$$* \left(\pi_{R_1}(r), \pi_{R_2}(r), \ldots \pi_{R_n}(r)\right) = r \quad \text{———} \quad (*)$$

The constraint state that every legal state r(R) should have a non-additive join decompos^n into R_1, R_2 ... R_n that is for every such r we have the above — (*)

5th NF:

R_i = Decomposed rel^n R_i must be a super key

A rel^n schema R is in 5th NF ore projecty join Normal form wrt a set F of functional, multivalued & join dependencies if,

for every non-trivial join dependency JD (R_1, R_2...) in F^+ (i.e., ⇒ by F), every R_i is a superkey of R.

Note:

A join dipendency JD (R_1.... R_n) specified on rel^n schema R, is a Trivial JD if one of the rel^n schema R_i in J.D (R_1, R_2 ..) is equal to equal to R.