Functional Dependency:-

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

X	Y
X_1	Y_1
X_2	Y_2
X_3	Y_3
X_4	Y_1

$X \rightarrow Y$

$\Downarrow$

X determines Y

X => Left hand side set of attbutes.

Y => Right hand side set of ".

Thus X functionally determines Y in a relation schema R if and only if when ~~vo~~ ever any two tuples of r(R) ~~are~~ agree on their X-value, they must necessarily agree on their ~~vostra~~ Y-value.

Note the following:-

1) If a constraint on R states that there cannot be more than one tuple with a given X-value in any relation instance r(R) - that means X is a candiate key of R.

2) If $X \rightarrow Y$ in R this does not say whether or not $Y \rightarrow X$ in R.

Closure of Functional dependency:-

The set of all denpendencies that include F as well as all dependencies that can infered from F called closure of F; it is denoted by F^+

~~Arm~~ Armstrong Rule or Inference Rule:-

For a Given relation there may be man[y] Functionl dependencies let us denote F as a set o[f] functionl dependencies for a given relation R. The[re] can be many other Functional dependencies that can be derive from F using the inference Rule.

IR 1:- (Reflexive Rule):-

If $x \supseteq y$ then $x \to y$.

Example:- $\underbrace{\text{S\_name}, \text{Saddress}}_{y}$ → $\underbrace{\text{S\_name}}_{x}$ then ~~Sname~~

IR2 (augmentation Rule): $\{x \to y\} \models xz \to yz$.

Example: Sid → s-name, then

S_id, S-address → S-name, S-address.

IR3 : (Transitive Rule)

$\{X \to Y \quad Y \to Z\} \models X \to Z$

SSn → Dnum , Dnum → Dname , then SSn → Dname

IR4: (decomposition or projective rule) : $\{X \to YZ\}$

$\models X \to Y$

$\{\cancel{Y} \to \cancel{Y}Z\} \models \cancel{X}$

Ssn → Dnum → {Dname, Dmgr-ssn}

then

Dnum → Dname

and

D-num → Dmgr_ssn.

IR 5: (union or additive Rule):

$\{X \to Y, X \to Z\} \models X \to YZ$

SSn → Ename   Ssn → D-num

Ssn → {Ename, D-num}

IR6 (Pseudotransitive Rule):

$\{X \to Y, WY \to Z\} \models WX \to Z$

E-id → S-name and S-name, Ssn → Saddress
mail

then E-id , SSn → address
mail

~~Definition of closure of an attribute:-~~

Full Functional Dependency : (FFD)

The term FFD is used to indicate the minimum set of attributes in a determinant of a FD. In other words, attributes in a determinant of a FD. In other words, x will be fully functionally dependent on a set of attributes Y if ~~and~~ the following condition holds good:

1) x is functionally dependent on Y
2) x is not functionally dependent on any subset of Y.

$Y \rightarrow X$

Partial Functional dependency:-

{Emp-id, project, project-budget} → {years spent in a project}

{SSn, age} → {name} Not FDD

Fully Functional dependency:-

{Emp-id, Project} ~~Proj~~ → {years spent in a project}

{SSN} → {Name}

FDD.

Trivial Functionl dependency:-

The ~~Reflect~~ Reflexive rule (IR1) States that a Set of attributes always determines itself or any of its Subsets. The Dependencies generated by ~~refel~~ reflexive rule are always true, Such dependencies called trivial. A functional dependency said to be a trivial if $x \supseteq Y$ when $x \to Y$.

Algorithm to demostrate how the ~~constr~~ Armstrong rule computes the closure Of functional dependency:-

1. $F^+ = F$
2. Repeat for each functional dependency in F^+
   - 2.1 Apply reflexive and augmentation rules on f and add resulting functional dependencies to F^+.
   - 2.2 For each pair of functional dependencies F_1 and F_2, if F_1 and F_2 can be combined using transitive rule, then add the resulting FD.
   - 2.3 Continue the procedure until F^+ doesn't change any further.
3. End.

Emp id	Ename	Dept id	Deptname	Dept Loc
1	A	001	X	abc def
2	B	001	Y	def abc
3				
4	C	002	Z	cde X
	D	003	X	xyz
Null		004	A	def

$R(A, B, C, G, H, I)$ and $F = \{A \to B, A \to C, CG \to H, CG \to I, B \to H\}$

Find closure of Functional dependency F^+.

(1) By applying reflexive Rule :- No extra dependency found.

(2) By Transitive Rule :-

$A \to B$ $B \to H$ New $A \to H$

(3) By union :-

$CG \to H$ and $CG \to I$ New $CG \to HI$

(4) Pseudotransitive :-

$A \to C$ and $CG \to HI$

$AG \to HI$

(5) Augmentation :-

$A \to C$

$AG \to CG$

(6) Transitive :-

$AG \to CG$ $CG \to H$ New $AG \to H$

$AG \to CG$ $CG \to I$ $AG \to I$

$F^+ = \{A \to B, A \to C, CG \to H, CG \to I, B \to H, A \to H, CG \to HI, AG \to HI, AG \to CG, AG \to H, AG \to I\}$

For relation R (A, B, C, D, E, F, G) the functional dependencies given are A→B, BC→DE, AEF→G

Prove that ACF→DG

1. A→B 2. BC→DE 3. AEF→G

4. AC→BC (Augmentation, 1)

5. AC→DE (4, 2, Pseudotransitive)

6. ACF→DEF (5, Augmentation)

7. ~~ACF→~~ ACF→ACDEF (6, Augmentation)

8. ACF→AEF (7, Decomposition)

9. ACF→G (8, 3, Transitivity)

10. ACF→D (6, Decomposition)

11. ACF→DG (9, 10, Additive Rule)

It means a set of attributes that is dependent on X. If it contains all the other attributes of that relation then X becomes the key of the relation.

Algorithm to compute closure of set of attributes X^+

Result = X

while (change to result) do

for each FD $B \rightarrow \gamma$ in F do

Begin

if B ⊆ result then result = result ∪ γ

End

Q. Consider a Relation R with attributes A, B, C, D, E, and F which satisfies the following set of functional dependencies

$AB \rightarrow C$ $D \rightarrow E$

$BC \rightarrow AD$ $CF \rightarrow B$.

1) Find $\{A, B\}^+$

2) $\{B, C\}^+$ 3) Find $\{A, B, F\}^+$

$A^+ = \{A\}$

$B^+ = \{B\}$

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

① By $AB \rightarrow C$

② By $BC \rightarrow AD$

③ By $D \rightarrow E$

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

① By $BC \rightarrow AD$

② By $D \rightarrow E$

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

$\Downarrow$

candidate key.

Q. Find out the possible candidate keys from the following FD's of Relation R.

$A \rightarrow C$

$C \rightarrow D$

$D \rightarrow B$

$E \rightarrow F$

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

From the FD's it is seen that C, D, B and F determined from A, C, D and E respectively. However the attributes A and E cannot be determined by any other attributes and therefore, they are not dependent on any other attributes, so A and E should be present in candidate key.

Q. $R(A, B, C, D)$

$A \rightarrow B$, $B \rightarrow C$, $C \rightarrow D$, $D \rightarrow A$ Find candidate key?

Ans A, B, C, D.

Q. $R(A, B, C, D, E, F, G, H)$

$CH \rightarrow G$, $A \rightarrow BC$, $B \rightarrow CFH$, $E \rightarrow A$, $F \rightarrow EG$

Find candidate keys.

Ans :- D

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

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

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

Equivalence of Sets of Functional Dependencies

Cover of a FD:-

A Set of FD's F is said to cover another set of FD's E if every FD in E is also in F^+; that is, if every dependency in E can be inferred from F; alternatively, we can say E is covered by F.

Defn of Equivalence:-

Two sets of functional dependencies E and F are equivalent if $E^+ = F^+$. Therefore, equivalence means that every FD in E can be inferred from F, and every FD in F can be inferred from E; ~~that is~~

that is E is equivalent of F if both the conditions

E covers F and F covers E hold.

Consider the following set of FDs

$F = \{A \to C,\ AC \to D,\ E \to AD,\ E \to AH\}$

and

$G = \{A \to CD,\ E \to AH\}$

Prove that F and G are equivalent.

~~$F^+ = \{A \to C,\ AD \to CD\ AC \to D,$~~

$C.C \to D$ (By pseudotransitive)

(1) $C \to D$

(2) $A \to C,\ C \to D$ so that $A \to D$

(3) $A \to CD$ (1, 2, additive)

$F^+ = \{A \to CD,\ E \to AH,\ A \to C,\ AC \to D,\ E \to AD\}$

Covers G.

Prove G^+ covers F

~~$G^+ =$~~

$G^+ =$ (1) $A \to C$ (2) $A \to D$ (3) $E \to AH$

(4) $AC \to AD$ (augmentation, 2)

(5) $AC \to D$ (by decomposition, 4)

~~E → AH~~

For ~~A → D~~ E → AD

E → AH

(6) E → A (by decomposition, 3)

(7) E → D (by transitive 2, 6)

(8) E → AD (by union 6, 7)

$G^+ = \{ A \to C, AC \to D, E \to AD, E \to AH \}$

So F and G are equivalent.

Minimal set of FDs (Minimal Cover)

A Set of FDs is Said to be minimal cover if Satisfies the following properties:-

1) Single attribute in RHS of the FD i.e Singleton RHS

2) No extra attribute in LHS of FD

3) No Redundant FD's.

Algorithm for computing extra attributes in LHS of FD

Consider F to be a set of FDs and $\alpha \rightarrow \beta$ be a FD in F. To test if $A \in \alpha$ is extra in α :-

1. Compute $\{\alpha - A\}^+$ using dependencies in F.
2. Check that $\{\alpha - A\}^+$ contains A or not If it contains A then A is extraneous.

Find the minimal cover for the Relation R (A, B, C, D, E) and the set of FDs $F = \{A \rightarrow D, BC \rightarrow AD, C \rightarrow B, E \rightarrow A, E \rightarrow D\}$

1. $A \rightarrow D$

$\left.\begin{array}{l} BC \rightarrow A \\ BC \rightarrow D \end{array}\right\}$ Decomposition

$C \rightarrow B$

$E \rightarrow A$

$E \rightarrow D$

Making all R.H.S Single ton

2. Check for extraneous attributes in L.H.S

$\{\alpha - B\}^+ = \{C\}^+ = \{C, B\}$

B is extraneous.

$\{\alpha - C\}^+ = \{B\}^+ = \{B\}$

C is not extraneous.

Now FD's are:-

$A \to D$
$C \to A$
$C \to D$
$C \to B$
$E \to A$
$E \to D$

~~Removin~~

3) Removing Redundant FD's:-

① $A \to D \Rightarrow$ Excluding $A \to D$ Find $(A)^+ = \{A\}$ Not giving 'D' so Not Redundant.

② $C \to A$ Excluding $C \to A$ Find $(C)^+ = \{D, B, C\}$ Not giving 'A' so not Redundant.

③ $C \to D \Rightarrow \{C\}^+ = \{C, B, A, D\}$ Not Redundant

④ $C \to B \Rightarrow \{C\}^+ = \{C, D, A\}$ Not Redundant

⑤ $E \to A \Rightarrow \{E\}^+ = \{E, D\}$ Not Redundant

⑥ $E \to D \Rightarrow \{E\}^+ = \{E, A, D\}$ Redundant then Remove

Minimal cover $F_c = \{A \to D, C \to A, C \to D, C \to B, E \to A\}$

Find minimal cover for the given set of FD's

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

1) All are singleton in RHS

2) Check for extraneous attributes in L.H.S:-

$AB \to C$.

$\{\alpha - A\}^+ = B^+ = \{B\}$

A is not extraneous

$\{\alpha - B\}^+ = \{A\}^+ = \{A, B\}$

B is extraneous.

$A \to C$

ii) Check ~~Redundancy~~ of Redundancy:-

✓ $A \rightarrow B$ $(A)^+ = \{A, C\}$ Not Redundant

✓ $A \rightarrow C$ $(A)^+ = \{A, B\}$ "

✓ $D \rightarrow A$ $(D)^+ = \{D, C, E\}$ "

× $D \rightarrow C$ $(D)^+ = \{D, A, B, C\}$ Redundant

✓ $D \rightarrow E$ $(D)^+ = \{D, A, B, C\}$ Not Redundant.

FD' cover $F_c = \{A \rightarrow B, A \rightarrow C, D \rightarrow A, D \rightarrow E\}$

Decomposition:-

The Decomposition of Relational Schema $R = \{A_1, A_2 \ldots A_n\}$ is its replacement by a set of relational schema $R_1, R_2 \ldots R_n$ such that $1 \leq i \leq n$ and $R_1 \cup R_2 \ldots \cup R_n = R$

A Relational schema R can be decomposed into a collection of relational schemas to eliminate some anomalies in original Relation R. Here, relational schema ~~R~~ R_i is a subset of R and intersection schemas $R_i \cap R_j$ for $i \neq j$ should not be empty.

Lossless Decomposition:-

A Relation when decomposed into two ~~or~~ or more relations, the primary objective is to ensure that should be loss less. In order to achieve that some FD's needs to be checked. Let us assume that relation R is decomposed into two Relations R_1 and R_2, and F is a set of FDs on R. F^+ is a closure set of FDs of R holding any of the two following dependencies.

1. $R_1 \cap R_2 \rightarrow R_1$

$R_1 \cap R_2 \rightarrow R_2$

For the decomposition to be loss less, atleast one of two dependencies should hold true value.

2. R (A, B, C}

R_1(A, B) R_2 (A, C)

Recover

R' (A, B, C)

If R' = R, then decomposition is loss less.

R

A	B	C
A_1	B_1	C_1
A_2	B_1	C_2
A_3	B_2	C_1

R_1

A	B
A_1	B_1
A_2	B_1
A_3	B_3

R_2

A	C
A_1	C_1
A_2	C_2
A_3	C_1

Loss Less

$R_1 \bowtie R_2$

A	B	C
A_1	B_1	C_1
A_2	B_1	C_2
A_3	B_2	C_1

R_2

A	B
A_1	B_1
A_2	B_1
A_3	B_2

R_3

B	C
B_1	C_1
B_1	C_2
B_2	C_1

Los

$R_2 \bowtie R_3$

A	B	C
A_1	B_1	C_1
A_1	B_1	C_2
A_2	B_1	C_1
A_2	B_1	C_2
A_3	B_2	C_1

extra

R (A, B, C, D, E, G) Decomposed into

R_1 (A, B) R_2 (B, C) R_3 (A, B, D, E)

R_4 (E, G)

$AB \to C$ ✗ $B \to D$ ✓

$AC \to B$ ✗ $BC \to A$ ✗

$AD \to E$ $E \to G$

Rule :-

1) Two or more α in LHS of a FD.
2) One or more α at R.H.S of a FD.
3) then put α on blank corresponding position of R.H.S.

Dependency preserving?

R_1 (A, B)

R_2 (B, C)

R_3 (A, B, D, E) $AD \to E$, $B \to D \to F_3$

R_4 (E, G) $E \to G$ F_4

$F_3 + F_4 \neq F$

Not dependency preserving.

<u>Lossless Join:-</u>

$AD \to E$, $B \to D$, $E \to G$

	A	B	C	D	E	G
R_1	α	α		@	@	@
R_2		α	α	@		
R_3	α	α		α	α	@
R_4					α	α

<u>LOSSY</u>

$\alpha \Rightarrow$ Distinguishable variable.

R(A, B, C, D, E) Decomposed into $R_1(B, C, D)$
$R_2(A, C, E)$ $F = \{AB \to C,\ C \to E,\ B \to D,\ E \to A\}$

	A	B	C	D	E
R_1	α		α		α
R_2	@	α	α	α	@

Lossless

Normalization

why Normalization? [Data anomalies are unmatched or missing information caused by incorrect design of Database Schema]

1) Insertion Anomalies
2) Deletion "
3) updation "

Insertion anomalies :-
Insertion anomalies are issues that occures when we ~~are~~ try to insert information in data base for the first time Missing or incorrectly formatted entries are common insertion Error.

Deletion anomalies:-
Deletion anomalies occures when we delete data item from a flaw Schema. If we ~~write~~ delete a tuple that contains the last piece of information, then some of the important information gets lost.

Updation anomalies:-

updation anomalies are data inconsistencies that results from data redundancy on partial update. If a modification is not carried out in all relevant rows, the database become inconsistent.

Normalization:-

Example:-

EMP

Eid	E-name	Dept-id	Dept-name	Dept-loc
1	A	~~CSE~~ D1	CSE	L_1

Insertion Anomalies
→

Null	Null	D2	EC	L_2

Deletion anomalies (If delete following info) IF info Lost.

2	B	D_3	IT	L_3
3	C	D_1	CSE	L_1
4	D	D_1	CSE	L_1

Change

not Changed

upate the Location of D_1 to D2

then ~~updti~~ updation anomalies.

Normalization:-

Normalization of data can be considered as a Process of analizing the given schema on their FD's and primary keys to achieve desirable properties of minimizing redundancies, minimizing inconsistency, minimizing insertion, deletion and updation anomalies.

1NF:-

A table is said to be in 1NF if the domain of the attributes are atomic in nature. It does not allows to have a set of values for a particular attribute in any tuple of a relation.

Emp

Eid	E name	E-loc
1	A	{KOL, Mumbai}
2	B	{KOL}
3	C	{Siliguri, KOL}
4	D	{Chennai}

Techniques to convert a relation to 1NF:-

1) Introducing redundant values:-

E id	E-name	E-loc
1	A	KOL
1	A	Mumbai
2	B	KOL
3	C	Siliguri
3	C	KOL
4	4	Chennai

Introducing ~~d~~ Null values:-

E id	E name	E loc 1	E Loc 2	~~E loc~~
1	A	KOL	mumbai	
2	B	KOL	Null	
3	C	Siliguri	KOL	
4	D	chennai	Null	

Decomposing the table:-

E id	E-name
1	A
2	B
3	C
4	D

E-id	E-loc
1	KOL
1	Mumbai
2	KOL
3	Siliguri
3	KOL
4	Chennai

1st table : PK + all non-repeating attributes

2nd table :- PK + all repeating ~~+~~

Second Normal Form:- (2NF)

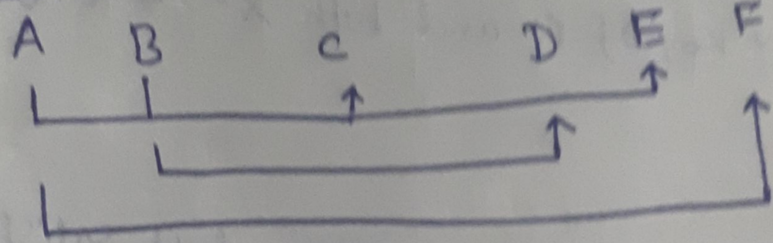
2NF is based on full functional dependency.

$X \to Y$ is a fully functionally dependent if removal of any attribute A from X means that ~~depend~~ dependency ~~does~~ does not hold any more that is: for any attribute $A \in X$, $(X - \{A\})$ does not determines Y.

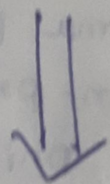
Definition:-

A Relational schema R is in 2NF if every non prime attribute fully functionally dependent on the primary key of R.

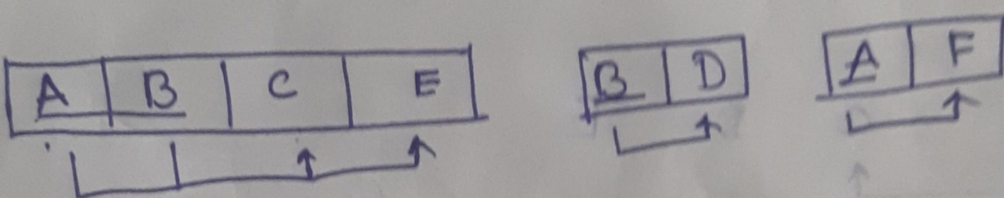
Example 1



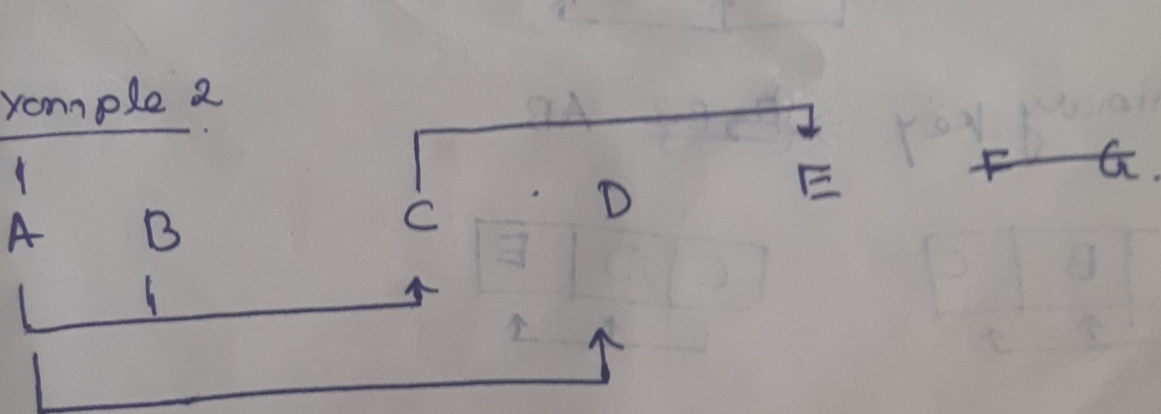
$\{A\ B\}^+ \rightarrow \{A, B, C, D, E, F\}$



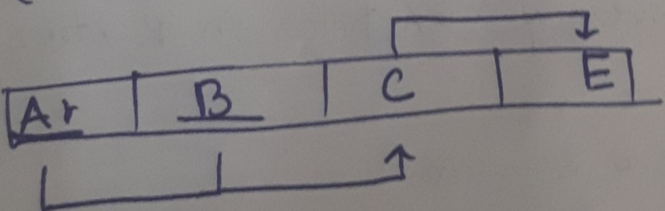
2NF Normalization

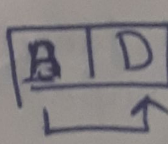


Example 2



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



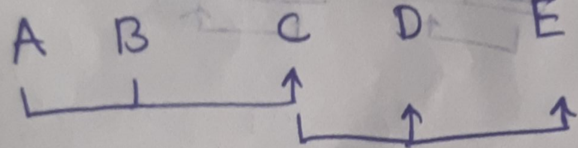


Third Normal form:-

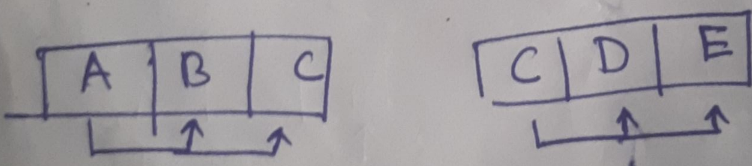
3NF is based on the concept of transitive dependency. A FD $X \rightarrow Y$ in a relation schema R is transitive if there is set of attributes Z that is neigther a candidate key nor a subset of any key of R and both $X \rightarrow Z$ and $Z \rightarrow Y$ hold.

Definition:-

A relational schema R is in 3NF if it satisfies 2NF and no non-prime attribute of R transitively dependent on the primary key



Primary key ⇒ ~~{A, B}~~ AB



Boyce - codd Normal Form:-

It is in

A relational schema R is in BCNF if whenever and a non-trivial functional dependency $X \rightarrow A$ holds in R, then X is a superkey of R.

FD1:- {Student, course} → instructor

Instructor → course.

Student Course Instructor

primary key : Student, Course

Student, Course → instructor ✓

Instructor → course ✗

AS Instructor is not a superkey.

A B C

Student	Course

Instructor	Course

A B C D

A D → B C D

B C → A D

D → B

A superkey ✓ P.K

B, C superkey ✓ Candidate key

D superkey ✗

Not in BCNF.

A	C	B

D	B

Given

R (A, B, C, D)

A, C → B, D

AD → B

A, B → C

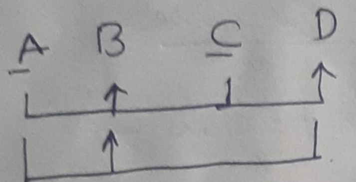
A → DE

B → F

F → GH

D → IJ

A, B, C, D, E, F, G, H, I, J



P.K :- AC   AC → B D

AD superkey ?

A, D → B

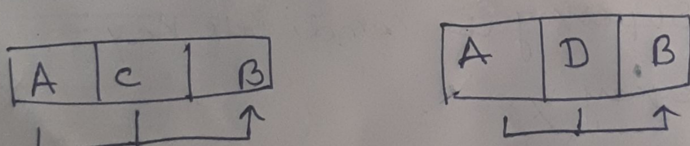
AD → AD

AD → A B D

AD can not determines C therefore

there fore AD can not be considered as key

The relation is not in BCNF.



find the emp name where empid = 5

$\Pi_{empname} (\sigma_{empid = 5} EMP)$