Normalization :-

⇒ Normal Forms are used in the process of normalization used in database.
⇒ Normalization generally used in to avoid redundancy in database tables.
⇒ Normalforms are set of rules to perform decomposition
⇒ For normalization firstly you should have to know about functional dependencies -

Functional Dependencies :-

$f : \alpha \rightarrow B$

$f(\alpha) \rightarrow B$ {in math}

α	β
a	1
b	2
c	3
d	4

set of attributes

⇒ Functions in dbms does not mean function in math. That if you have the value of α you should have to find the value of β by using it.
⇒ In DBMS functional dependency means if have the value of α you can search the value of β. Here you have not to compute the value of β.

α	β
a	1
b	2
c	3
d	4

Here if you have value of α is a can you find the value of β ?
Answer is yes Here a is only one time and you can find the value of β that is 1.

α	β
a	1
a	2
c	3
d	4

Here in this table here if value of α is a you can't find the value of β because here occurance of a is two times.

In math $f(x) \rightarrow y$

if $1 \rightarrow a$

then $1 \rightarrow b$ ✗

$2 \rightarrow a$ can be true

because 2 different values of α can have same value of β but with same value of α can't have two different values. Same Logic is used in FD in DBMS.

Scanned with CamScanner

(2)

R :

	α	B
t1 →	a	1
t2 →	a	2
	c	3
	d	4

X

On same value of α. there can't different values of B

α	B
a	1
b	1
c	3
d	4

✓

On different values of B there can be different value of B

Age
18
9

Functional Dependency :-

$\alpha \subseteq R$, $B \subseteq R$

$\alpha \rightarrow B$

R

	α	B
t1 →	a	
t2 →	a	

α & B are set of Attributes

if $t_1[\alpha] = t_2[\alpha]$

then $t_1[B] = t_2[B]$

$\alpha \rightarrow B$

Determinant — Dependent

⇒ If there is a functional dependency $\alpha \rightarrow B$, if for some tuples the value of α is same then the B value of B for these tuples also will be same for ~~both~~ all those tuples.

⇒ α is called determinant

⇒ B is called Dependent

⇒ Functional dependencies can be of two types.

$\alpha \rightarrow B$

Trivial

$AB \rightarrow A$

if $B \subseteq \alpha$

Functional Dependency is valid but you will not find anything new in the value of B

Non-trivial

$B \not\subseteq \alpha$

eg $AB \rightarrow ABC$ or $AB \rightarrow C$

In non-trivial Dependency you will find the value of B as a new value. or having something new than value of α.

Scanned with CamScanner

Functional Dependencies Basic Problems :-

$\alpha \rightarrow \beta$

(a) $A \rightarrow BC$ ✓

(b) $DE \rightarrow C$ ✓

(c) $C \rightarrow DE$ ✗

(d) $BC \rightarrow A$ ✓

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

⇒ We logically its not correct never find out functional dependency with the help of data

⇒ Data depends on FD & FD does not depends on Data.

⇒ If there is $\alpha \rightarrow B$, and if you get two different values of B on the same value of α, then the dependency is not valid.

$\alpha \rightarrow B$
$a \rightarrow 1$
$a \rightarrow 2$

This is wrong

⇒ If the value of α is unique the irrespective the value of B the dependency hold good.

e.g.

Here unique value of A →

A	B	C	D	E
a	2	2	4	5
b	a	3	4	5
c	2	3	6	5
d	2	3	6	6

$\alpha \rightarrow B$

$A \rightarrow B$

$A \rightarrow CD$

$A \rightarrow BCDE$ ✓

$\alpha \rightarrow B$

$A \rightarrow C$

$BD \rightarrow C$

$ABDE \rightarrow C$ ✓

(1) Example :- Find which dependencies are valid.

(a) $XY \rightarrow Z$ && $Z \rightarrow Y$ - invalid

(b) $YZ \rightarrow X$ && $Y \rightarrow Z$ - valid

(c) $YZ \rightarrow X$ && $X \rightarrow Z$ - invalid

(d) $XZ \rightarrow Y$ && $Y \rightarrow Z$ - invalid

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

(2)

(a) $A \rightarrow B$ && $BC \rightarrow A$ - invalid

(b) $C \rightarrow B$ && $CA \rightarrow B$ - invalid

(c) $B \rightarrow C$ && $AB \rightarrow C$ - valid

(d) $A \rightarrow C$ && $BC \rightarrow A$ - invalid

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

Scanned with CamScanner

Attribute closure / closure on attribute set / closure of attribute set :-

Attribute closure of an attribute set 'A' can be defined as a set of attributes which can be functionally determined from it.
Denoted by F^+

$(A)^+ = \overset{A}{\overset{AB}{ABC}}$

Directly or indirectly which attributes, we can find is called closure set of a attribute

R(ABC)
$A \to B$
$B \to C$
$\boxed{A \to BC}$

R(ABCDEF)
$A \to B$
$C \to DE$
$AC \to F$
$D \to AF$
$E \to CF$

$(D)^+ = D$
$= DAF$
$= \boxed{ABDF}$

$(DE)^+ = DE$
$= ADEF$
$= ACDEF$
$= \boxed{ABCDEF}$

Armstrong's Axioms / Rules :-

⇒ Axiom is a statement that is taken to be true, and serve as a premise or starting point for further arguments.

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

Primary Rule (RAT Rules)

Reflexivity :- if $Y \subseteq X$ then, $X \to Y$.

$X = ABC$
$Y = AB$
$X \to Y$
$ABC \to AB$

Augmentation :- if $X \to Y$ then $XZ \to YZ$

Transitivity :- if $X \to Y$ && $Y \to Z$ then $X \to Z$

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

$X \to Y$
$WY \to Z$
$WX \to WY$ (using Augmentation)
$WY \to Z$
$\Rightarrow WX \to Z$ ✓

Secondary Rules

Union :- if $X \to Y$ && $X \to Z$ then $X \to YZ$

Decomposition :- if $X \to YZ$ then $X \to Y$ && $X \to Z$

These both can be applied only on Right hand side eg.
$\boxed{AB \to CD}$
$AB \to C$
$AB \to C$ ✓

Pseudo Transitivity :- if $X \to Y$ && $WY \to Z$ then $WX \to Z$

$\boxed{AB \to C}$
$A \to C$
$B \to C$ ✗
$AB \to C$ will make FD weak

$ZX \to Y$
$ZX \to WX$
$XZ \to YWXZ$
$XZ \to YW$ ✓

Composition :- if $X \to Y$ && $Z \to W$ $XZ \to YW$

Scanned with CamScanner

Closure set of attribute:-

$F \to F_1 + F_2$

Total set of functional dependency equal direct & indirect dependency

$F \to F_1 + F_2$

$A \to B$ $\quad$ $A \to C$

$A \to BC$

R(ABC)

$A \to B$

$B \to C$

$A^+ \to BC$

> Closure set is a set of all attributes which can be computed by given attribute

Eg.

R(ABC)

$A \to B$

$B \to C$

$\Rightarrow$ Closure set of A, B & C

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

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

$C^+ = \{C\}$

(1) R(ABCDEFG)

$A \to B$

$BC \to DE$

$AEG \to G$

$(AC)^+ = ?$

$(AC)^+ = AC$

$= ABC$

$= \boxed{ABCDE}$

(2) R(ABCDE)

$A \to BC$

$CD \to E$

$B \to D$

$E \to A$

$B^+ = ?$

$B^+ = B$

$= \boxed{BD}$

(3) R(ABCDEF)

$AB \to C$

$BC \to AD$

$D \to E$

$CF \to B$

$(AB)^+ = ?$

$(AB)^+ = AB$

$= ABC$

$= ABCD$

$= \boxed{ABCDE}$

(4) R(ABCDEFGH)

$A \to BC$

$CD \to E$

$E \to C$

$D \to AEH$

$ABH \to BD$

$DH \to BC$

$BCD \to H$?

LHS

$(BCD)^+ = BCD$

$= BCDE$

$= ABCDEH$

it means

$\boxed{BCD \to H \text{ is valid}}$

$\Rightarrow$ Equivalence of functional dependency:-

$\Rightarrow$ If you have two sets of functional dependencies then check whether two dependencies are actually mean the same or one is a subset of other, or other is the subset of one or they are actually different

(1) R(ACDEH)

F:	G:
$A \to C$	$A \to CD$
$AC \to D$	$E \to AH$
$E \to AD$	
$E \to H$	
$F \subseteq G$	$G \subseteq F$
$(A)^+ = ACD$	$(A)^+ = ACD$
$(AC)^+ = ACD$	$(E)^+ = EADHC$
$(E)^+ = EAH$	
$= EAHCD$	

$\Rightarrow$ Find closure of F using FD of G and find closure of G using FD of F.

$\Rightarrow$ Then check these closure set with its own closure.

it means $F = G$

Scanned with CamScanner

6

(2) R(PQRS)

X: $P \to Q$, $Q \to R$, $R \to S$

$X \not\subseteq Y$

$P^+ = PQRS$
$Q^+ = Q$
$R^+ = RS$

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

$Y \subseteq X$

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

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

(3) R(ABC)

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

$F \subseteq G$

$A^+ = ABC$
$B^+ = BAC$
$C^+ = CAB$

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

$G \subseteq F$

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

(a) $F \subseteq G$
(b) $G \subseteq F$
(c) $F = G$ ✓
(d) $F \neq G$

(4) R(VWXYZ)

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

$F \not\subseteq G$

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

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

$G \subseteq F$

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

(a) $F \subseteq G$
(b) $G \subseteq F$ ✓
(c) $F = G$
(d) $F \neq G$

⇒ Irreducible set of FD (Canonical form)

⇒ Canonical form or irreducible set of FD, means whether in any given functional dependency there exist some redundant elements or not; and if there exist some redundant elements we must remove it.

⇒ Redundant elements means, if the presence or absent of elements do not effect the capability of FD.

⇒ We have to minimise FD.

Scanned with CamScanner

Irreducible set of FD (Canonical Form) :-

FD: $\alpha \to \beta$ $\boxed{\alpha \text{ and } \beta \text{ are set of attributes}}$

⇒ There can be three cases of redundancy in FD: $\alpha \to \beta$.
⇒ There could be redundancy on the side of α.
⇒ There could be some redundancy on the side of β.
⇒ There could be redundancy in the entire case of FD. ($\alpha \to \beta$)

R (WXYZ)

$X \to W$
$WZ \to XY$
$Y \to WXZ$

we can write like :-

$X \to W$
$WZ \to X$ (Step 1)
$WZ \to Y$
$Y \to W$
$Y \to X$
$Y \to Z$

Now Redundancy can't be on rightside

Rightside

$\alpha \to \beta\gamma$
$\alpha \to \beta$
$\alpha \to \gamma$

Decomposition

Leftside

$\alpha\beta \to \gamma$
$\alpha \to \gamma$
$\beta \to \gamma$

This is wrong

Decomposition can be applied only on rightside

⇒ If your presence or absent do not effect the behavior of the system, it means you are redundant.

⇒ To find whether the attribute is essential or redundant find closure of all attributes.

(Step 2)

① Find closure of X
$(X)^+ = XW$
~~$(WZ)^+$~~

② Again find closure of X by ignoring $X \to W$
$(X)^+ = X$

both the X^+ value is different, it means $X \to W$ is essential.

[If $X \to W$ dependency is redundant its absent will not effect the X^+ in second go]

[If it is essential then the power of X will be different from 1st X^+]

⇒ Now check for other dependencies

① $(WZ)^+ = WZXY$

$(WZ)^+$ by ignoring $WZ \to X$ is

② $(WZ)^+ = WZYX$

by ignoring $WZ \to X$ the result is same for $(WZ)^+$, it means $WZ \to X$ is redundant. So we can remove $WZ \to X$

⇒ Now check for $WZ \to Y$, ignore this and again find $(WZ)^+$

$(WZ)^+ = WZ$

ignorance of $WZ \to Y$ effect the power of WZ it means $WZ \to Y$ is important.

Scanned with CamScanner

(8) ⇒ Now find y^+

$(y)^+ = y w x z$

Now find y^+ 3 times again by ignoring each case once, which are $y \to w$, $y \to x$, $y \to z$

(a) by ignoring $y \to w$

$y^+ = y x z w$

Result for y^+ is same so we can remove $y \to w$, because is resundant.

Note :- When we find any redundancy we have to remove that dependency immediately, and we will not consider it for the next time.

(b) Now ignore $y \to x$

$y^+ = y z$

It effect the value of y^+ it means $y \to x$ is issential

(c) Now ignore $y \to z$

$y^+ \to y x w$

It is not same as $y^+ \to y x z w$ so it means $y \to z$ is issential.

Step (3) New Dependencies are

$x \to w$
$wz \to y$
$y \to x$
$y \to z$

Note :- There could be different minimal forms of FD. It depends on the what order you check the logic. In reverse can it is possible that any other rule will redundant. But every result will be minimal. All minimal forms will be correct on same FD.

⇒ Now Check redundancy on Left hand side :-

(1) Firstly find $(wz)^+$

(2) Then find w^+ and z^+ by ignoring $wz \to y$

(3) If the result of w^+ or z^+ is same as $(wz)^+$ it means another attribute is redundant.

$(wz)^+ = wzyx$

$w^+ = w$
$z^+ = z$. both results are different, it means w & z in wz both are issential

⇒ Finally Merge FD again

$x \to w$
$wz \to y$
$y \to xz$ [by using union rule]

This is the canonical form

Scanned with CamScanner

Minimal set of Functional Dependency:-

Example :- R(ABCD)

$A \to B$
$C \to B$
$D \to ABC$
$AC \to D$

Step-1 (Use decomposition at right side of FD)

$A \to B$
$C \to B$
$D \to A$
$D \to B$ ×
$D \to C$
$AC \to D$

⇒ Logically we have removed redundancy on the right hand side.

⇒ Now redundancy can be on ~~so~~ left hand side or the complete rule can be redundant

Step-2 :- To check the whole rule is extra or not.
Find closure of each rule. In first go find closure including the rule and in second go by ignoring the same rule. If rule is redundant immediately ignore the rule.

① $(A)^+ = AB$
After ignoring $A \to B$
$A^+ = A$
Closure of A^+ is different it means $A \to B$ rule is essential

② $(C)^+ = CB$
After ignoring $C \to B$
$(C)^+ = C$
closure of C^+ is different, it means $C \to B$ is also essential.

③ $(D)^+ = DABC$
After ignoring $D \to A$
$(D)^+ = DBC$
closure of D^+ is different, it means $D \to A$ is also essential.
Now ignore $D \to B$
$(D)^+ = DABC$
There is no difference b/w D^+ results, it means we can remove $D \to B$, because it is redundant
Now ignore $D \to C$
$D^+ = DAB$
There is difference in D^+ closure so ~~d~~ $D \to C$ is essential.

Scanned with CamScanner

(10) $(AC)^+ = ACBD$

Now ignore $AC \to D$

$(AC)^+ = ACB$

Now closure of $(AC)^+$ are different, so $AC \to D$ is essential.

⇒ <u>New dependencies are :-</u>

$A \to B$
$C \to B$
$D \to A$
$D \to C$
$AC \to D$

<u>Step-3</u> :- Check redundancy at left hand side.

⇒ Firstly find the closure set of AC

⇒ Then find closure set of A and C individually by ignoring $AC \to D$

$(AC)^+ = ACBD$

$(A)^+ = AB$ [Result of AC^+ & A^+ is not same, it means C is not redundant]

$(C)^+ = CB$ [Result of AC^+ & C^+ is not same it means A is not redundant]

<u>Step-4</u> :- Finally Merge FD again using Union Rule.

$A \to B$
$C \to B$
$D \to AC$
$AC \to D$ This is the canonical form

<u>Example 2</u> :-

$R(VWXYZ)$

$V \to W$
$VW \to X$
$Y \to VXZ$

Step 1
$V \to W$
$VW \to X$
$Y \to V$
$Y \to Z$

Step 2
$V^+ = VWX$
$V^+ = V$ (ignore $V \to W$)
$VW^+ = VWX$
$VW^+ = VW$ (ignore $VW \to X$)
$Y^+ = YVXZW$
$Y^+ = VXZ$ (ignore $Y \to V$)
$Y^+ = YVZWX$ (ignore $Y \to X$)
$Y^+ = YVWX$ (ignore $Y \to Z$)

Step 3
$V \to W$
$VW \to X$
$Y \to V$
$Y \to Z$
$(VW)^+ = VWX$
$V^+ = VWX$
$W^+ = W$
Result is same for VW^+ and V^+ it means W is redundant in VW we can ignore it

<u>Step 4</u> :- Finally merge FD again and remove W in VW at left side

$V \to W$
$V \to X$
$Y \to V$
$Y \to Z$
⟹
$V \to WX$
$Y \to VZ$
} Minimal Form

Scanned with CamScanner

Key:-

A Key is a set of attributes, which can uniquely identify a row or tuple in a table (Relation).

or

Key is a set of attributes with the help of which we can identify the value of other attributes.

⇒ Functional dependency and Keys are important concepts for normalization.

⇒ Without these FD & Keys we can't find normal forms.

R

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

⇒ To retrieve data from a database we need a key.

⇒ In a table we have name for each column but we can't have any name for any row of the table, So we can easily retrieve the data of column by using its name but can't retrieve the data of a row without any identification of that row.

⇒ There can be more than 1 ways to identify a row in a table.

Student

B-n	N	age	branch
1	A	19	101
2	B	21	101
3	A	19	101

⇒ In relation R, The values of A attribute are unique so it can be used as a key to identify the rows of the table.

$\boxed{Key = A}$

⇒ By using the value of A we can find the values of B and C attribute for a particular row.

Here is FD $A \rightarrow BC$

$$(A)^+ = ABC = R$$

$$(Key)^+ = R$$

⇒ If closure set of key can't find the whole relation, it means its not a key.

⇒ By using key value we can find all other values of the table

⇒ Apart from A attribute BC can also be a key in R relation because values of combination of B and C are unique.

$\boxed{Key = A, BC}$

⇒ It means a single table can have more than one key

⇒ Key can be a single attribute or can be a set of attributes

Scanned with CamScanner

(12) ⇒ There can be more than one tables in a database. And key is not used for database, its used for individual table.

⇒ If there are four tables in a database. Then every table should have a unique individual key.

⇒ Check the following FD where these are keys or not:-

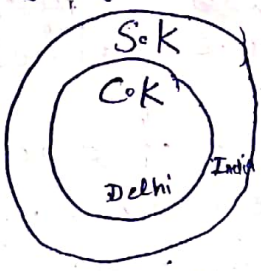
Eg (1). R(ABCD)

$A \to BC$

$A^+ = ABC$

A^+ can't search D, A can be key if $A^+ = R$

So we can say A is no a key



Eg (2) R(ABCD)

$ABC \to D$

$AB \to CD$

$A \to BCD$

Superkey { $(ABC)^+ = ABCD$, $(AB)^+ = ABCD$, $(A)^+ = ABCD$ } → Candidate key

Here ABC, AB, A all are Superkey because they can find R

⇒ Superkey :- Super key is the basic key used in DBMS. Super key is a set of attributes with the help of which you can uniquely identify a row or a tuple in a relation or you can uniquely identify all the other attributes

⇒ Candidate Key :- Efficient version of SK is called candidate key or we can say minimal super key is called candidate key. A super key is called candidate key if its proper sub-set is not a superkey. eg. if we have two superkey ABC and AB ABC key have its proper sub-set AB which is also a superkey, so it can't be a candidate key.

⇒ Every candidate key is a super key but every super key can't be candidate key

Scanned with CamScanner

Eg 3 R(ABCD)

$B \to ACD$

$ACD \to B$

$(B)^+ = ABCDB$

$(ACD)^+ = ACDB$

Here B and ACD both are super keys and candidate key. ACD is also candidate key because its proper-subset AC, AD, CD, A, B, C is not a superkey.

⇒ **Primary Key :-** A primary key is that candidate key which is selected by the database administrator as a primary mean to identify tuples. There can be multiple candidate keys in a table but there will be only single primary key, choosed by the database administrator.

Eg 4 R(ABCD)

$AB \to C$ $\quad (AB)^+ = ABCD$

$C \to BD$ $\quad (C)^+ = CBDA$

$D \to A$ $\quad (D)^+ = DA$

Here AB and C both are super & candidate keys.

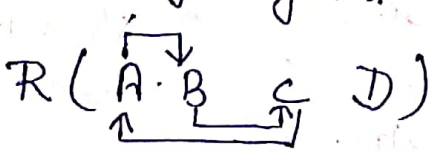
⇒ We can select AB or C candidate key as a primary key.

⇒ **How to find number of candidate keys :-**

Eg 1 R(ABCD)

FD: $A \to B$, $B \to C$, $C \to A$

Make edge diagram.



⇒ Here D attribute is not participating in Functional Dependency. Means D attribute is not finding any other attribute and anyother attribute cant find D

⇒ For Candidate key $(CK)^+ = R$

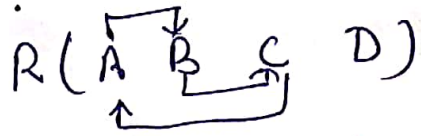
⇒ Attribute which don't have any incoming edge will be a essential attribute in candidate key, because nobody (no other attribute) can find it.

⇒ Now find its closure, it cant find R so make its combination with other attributes & find their closure sets.

Scanned with CamScanner

(14)

$A \to B$
$B \to C$
$C \to A$



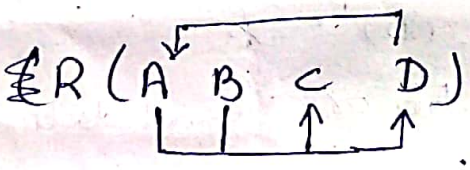
$(AD)^+ = ADBC$
$(BD)^+ = BDCA$
$(CD)^+ = CDAB$

} Here all combinations are candidate keys because they can find R which is ABCD

⇒ Now check can Attributes set of 3 attributes can be a candidate key?
like ABD, BCD

Ans :- ~~More than 2~~
Here combination of any other attribute with AD, BD or CD can be a super key but can't be a candidate key
⇒ because ABD etc will be a super set of AD etc and for candidate key is a minimal.

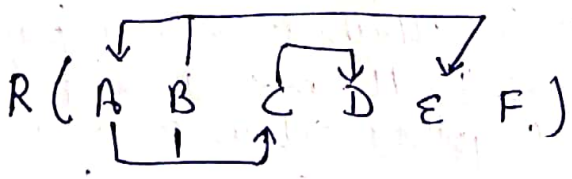
Eg (2) R(ABCD)
$AB \to CD$
$D \to A$



⇒ Here B attribute don't have incoming edge So it is essential for candidate key

$(B)^+$ — X
$(AB)^+ = ABCD$ ✓ Here AB and BD are candidate keys.
$(BC)^+ = BC$ X
$(BD)^+ = BDAC$ ✓

Eg (3) R(ABCDEF)
$AB \to C$
$C \to D$
$B \to AE$



⇒ Here essential attributes are BF

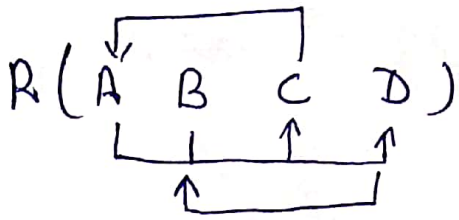
$(BF)^+ = BFAECD$

Here BF is only candidate key, because any combination with BF will become a super set of BF.

Scanned with CamScanner

Eg. 4 R(A B C D)

$AB \to CD$
$C \to A$
$D \to B$



⇒ Here every attribute have incoming edges, so here we don't find any essential attribute

⇒ Now we have to check closure set for every combination.

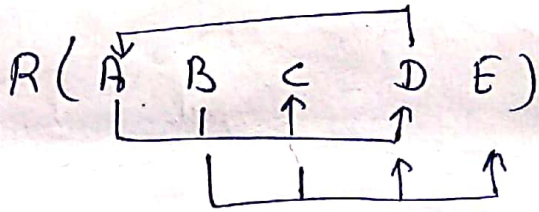
$A^+ = A$ ✗	$AB^+ = ABCD$ ✓	Combination with falin AC & BD
$B^+ = B$ ✗	$AC^+ = AC$ ✗	ACB ✗
$C^+ = CA$ ✗	$AD^+ = ADBC$ ✓	ACD ✗
$D^+ = DB$ ✗	$BC^+ = BCAD$ ✓	BDA ✗
	$BD^+ = BD$ ✗	BDC ✗
	$CD^+ = CDAB$ ✓	

because these all are super set of CK

Here candidate keys are AB, AD, BD, CD

Eg. 5 R(A B C D E)

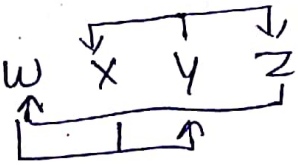
$AB \to CD$
$D \to A$
$BC \to DE$



Here essential element is B

$(B)^+ = X$	Any combination with falin BE will become super set of candidate key.
$(BA)^+ = ABCDE$ ✓	So here AB, BC, BD are only 3 candidate keys.
$(BC)^+ = BCDEA$ ✓	
$(BD)^+ = BDACE$ ✓	
$(BE)^+ = BE$ ✗	

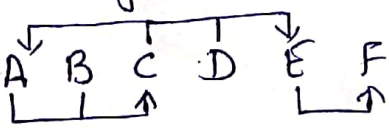
Eg 6 R(W X Y Z)



$Z \to W$
$Y \to XZ$
$XW \to Y$

$W^+ = W \cdot X$	$WX^+ = WXYZ$ ✓
$X^+ = X \cdot X$	$XZ^+ = XZWY$ ✓
$Y^+ = YXZW$ ✓	$ZW^+ = ZW$ ✗
$Z^+ = ZW \cdot X$	So CK are Y, WX, XZ Ans.

Eg 7 R(A B C D E F)



$AB \to C$
$DC \to AE$
$E \to F$

Essential BD

$(BD)^+ = BDX$	$(BDEF)^+ =$ BDEF ✗
$(BDA)^+ = ABCDEF$ ✓	
$(BDC)^+ = ABCDEF$ ✓	
$(BDE)^+ = BDEFX$	
$(BDF)^+ = BDFX$	

Scanned with CamScanner