FUNCTIONAL DEPENDENCY & NORMALIZATION

Functional Dependency-

In any relation, a functional dependency $\alpha \rightarrow \beta$ holds if-

Two tuples having same value of attribute α also have same value for attribute β.

Mathematically,

If α and β are the two sets of attributes in a relational table R where-

- $\alpha \subseteq R$
- $\beta \subseteq R$

Then, for a functional dependency to exist from α to β,

If $t1[\alpha] = t2[\alpha]$, then $t1[\beta] = t2[\beta]$

α	β
t1[α]	t1[β]
t2[α]	t2[β]

$$f_d : \alpha \rightarrow \beta$$

α	β
1	a
2	b
3	c
4	d
5	e

α	β
1	a
2	a
3	c
4	d
5	e

α	β
1	a
1	a
3	c
4	d
5	e

α	β
1	a
1	b
3	c
4	d
5	e

Types Of Functional Dependencies-

Trivial Functional Dependencies

Non-trivial Functional Dependencies

Trivial Functional Dependencies-

- A functional dependency $X \rightarrow Y$ is said to be trivial if and only if $Y \subseteq X$.

Thus, if RHS of a functional dependency is a subset of LHS, then it is called as a trivial functional dependency.

Examples-

The examples of trivial functional dependencies are-

- $AB \rightarrow A$
- $AB \rightarrow B$
- $AB \rightarrow AB$

Non-Trivial Functional Dependencies-

- A functional dependency $X \rightarrow Y$ is said to be non-trivial if and only if $Y \not\subset X$.
- Thus, if there exists at least one attribute in the RHS of a functional dependency that is not a part of LHS, then it is called as a non-trivial functional dependency.

Examples-

The examples of non-trivial functional dependencies are-

- $AB \rightarrow BC$
- $AB \rightarrow ABC$
- $AB \rightarrow CD$

Which functional dependency is not valid AND which are valid explain for given table

a) A → BC

b) DE → C

c) C → DE

d) BC → A

A	B	C	D	E
a	2	3	4	5
2	a	3	4	5
a	2	3	6	5
a	2	3	6	5

a) XY → Z && Z →Y

b) YZ → X && Y →Z

c) YZ → X && X →Z

d) XZ → Y && Y →Z

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

a) A → B && BC → A

b) C → B && CA → B

c) B → C && AB → C

d) A → C && BC → A

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

Closure of Attributes set or Attribute closure

- The set of all those attributes which can be functionally determined from an attribute set is called as a closure of that attribute set.
- Closure of attribute set {X} is denoted as $\{X\}^+$.

R (A, B, C)

$A \rightarrow B$

$B \rightarrow C$

$A \rightarrow BC$

Then what will be closure

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

It means A can identify all the three attributes of the relation.

Consider a relation R (A , B , C , D , E , F , G) with the functional dependencies-

$A \rightarrow BC$

$BC \rightarrow DE$

$D \rightarrow F$

$CF \rightarrow G$

Closure of attribute A-

$A^+ = \{ A \}$

$= \{ A , B , C \}$ (Using $A \rightarrow BC$)

$= \{ A , B , C , D , E \}$ (Using $BC \rightarrow DE$)

$= \{ A , B , C , D , E , F \}$ (Using $D \rightarrow F$)

$= \{ A , B , C , D , E , F , G \}$ (Using $CF \rightarrow G$)

Thus,

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

Closure of attribute D-

$D^+ = \{ D \}$

$= \{ D , F \}$ (Using $D \rightarrow F$)

We can not determine any other attribute using attributes D and F contained in the result set.

Thus,

$$\mathbf{D^+ = \{ D , F \}}$$

Closure of attribute set {B, C}-

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

$= \{ B , C , D , E \}$ (Using $BC \rightarrow DE$)

$= \{ B , C , D , E , F \}$ (Using $D \rightarrow F$)

$= \{ B , C , D , E , F , G \}$ (Using $CF \rightarrow G$)

Thus,

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

Consider the given functional dependencies-

$AB \rightarrow CD$

$AF \rightarrow D$

$DE \rightarrow F$

$C \rightarrow G$

$F \rightarrow E$

$G \rightarrow A$

$\{ CF \}^+$
$\{ BG \}^+$
$\{ AF \}^+$
$\{ AB \}^+$

$\{ CF \}^+ = \{ C , F \}$

$= \{ C , F , G \}$ (Using $C \rightarrow G$)

$= \{ C , E , F , G \}$ (Using $F \rightarrow E$)

$= \{ A , C , E , E , F \}$ (Using $G \rightarrow A$)

$= \{ A , C , D , E , F , G \}$ (Using $AF \rightarrow D$)

$\{ BG \}^+ = \{ B , G \}$

$= \{ A , B , G \}$ (Using $G \rightarrow A$)

$= \{ A , B , C , D , G \}$ (Using $AB \rightarrow CD$)

$\{ AF \}^+ = \{ A , F \}$

$= \{ A , D , F \}$ (Using $AF \rightarrow D$)

$= \{ A , D , E , F \}$ (Using $F \rightarrow E$)

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

$= \{ A , B , C , D \}$ (Using $AB \rightarrow CD$)

$= \{ A , B , C , D , G \}$ (Using $C \rightarrow G$)

Consider a relation R (A, B, C, D, E, F) with the functional dependencies-

$A \rightarrow B$

$C \rightarrow DE$

$AC \rightarrow F$

$D \rightarrow AF$

$E \rightarrow CF$

Find these closures

- $\mathbf{D}^+$
- $\{\mathbf{DE}\}^+$
- $\{\mathbf{CD}\}^+$
- $\{\mathbf{BD}\}^+$

Q.1	Q.2	Q.3	Q.4
R (A, B, C, D, E, F, G)	**R (A, B, C, D, E)**	**R (A, B, C, D, E, F)**	**R (A, B, C, D, E, F, G)**
$A \rightarrow B$ $BC \rightarrow DE$ $AEG \rightarrow G$	$A \rightarrow BC$ $CD \rightarrow E$ $B \rightarrow D$ $E \rightarrow A$	$A\ B \rightarrow C$ $BC \rightarrow AD$ $D \rightarrow E$ $CF \rightarrow B$	$A \rightarrow BC$ $CD \rightarrow E$ $E \rightarrow C$ $D \rightarrow AEH$ $ABH \rightarrow BD$ $DH \rightarrow BC$
$\{A, C\}^+$ $\{A, G\}^+$	$\{B, C\}^+$ $\{A\}^+$	$\{A, B\}^+$ $\{C, F\}^+$	$\{B, C, D\}^+$ $\{A, D\}^+$ $\{D, H\}^+$

Inference Rules/ Armstrong's Axiom

Primary Rules (RAT)

Reflexivity-

If B is a subset of A

Then $A \rightarrow B$

Augmentation-

If $A \rightarrow B$,

Then $AC \rightarrow BC$

Transitivity-

If $A \rightarrow B$ && $B \rightarrow C$

Then $A \rightarrow C$

Secondary Rules

Additive/ Union rule

If $A \rightarrow B$ && $A \rightarrow C$

Then $A \rightarrow BC$

Decomposition-

If $A \rightarrow BC$,

Then $A \rightarrow B$ && $A \rightarrow C$

Composition-

If $A \rightarrow B$ && $C \rightarrow D$,

then $AC \rightarrow BD$

Pseudotransitivity rule.

If $A \rightarrow B$ holds && $CB \rightarrow D$

Then $AC \rightarrow D$

Equivalence on set of functional dependencies OR functional dependency comparison

- Two different sets of functional dependencies for a given relation may or may not be equivalent.
- If F and G are the two sets of functional dependencies, then following 4 cases are possible-

  $G \subseteq F$

  $F \subseteq G$

  $F \neq G$

  $F = G$

Q.1: **R (A, C, D, E, H)**

F:	G:
$A \rightarrow C$	$A \rightarrow CD$
$AC \rightarrow D$	$E \rightarrow AH$
$E \rightarrow AD$	
$E \rightarrow H$	

<u>Determining whether F covers G-</u>

<u>Step-01:</u>

- $(A)^+ = \{ A , C , D \}$ // closure of left side of $A \rightarrow CD$ using set G
- $(E)^+ = \{ A , C , D , E , H \}$ // closure of left side of $E \rightarrow AH$ using set G

<u>Step-02:</u>

- $(A)^+ = \{ A , C , D \}$ // closure of left side of $A \rightarrow CD$ using set F
- $(E)^+ = \{ A , C , D , E , H \}$ // closure of left side of $E \rightarrow AH$ using set F

<u>Step-03:</u>

Comparing the results of Step-01 and Step-02, we find-

- Functional dependencies of set F can determine all the attributes which have been determined by the functional dependencies of set G.
- Thus, we conclude F covers G i.e. $F \supseteq G$.

Q.1: R (A, C, D, E, H)

F:	G:
$A \to C$	$A \to CD$
$AC \to D$	$E \to AH$
$E \to AD$	
$E \to H$	

Determining whether G covers F-

Step-01:

- $(A)^+ = \{ A , C , D \}$ // closure of left side of $A \to C$ using set F
- $(AC)^+ = \{ A , C , D \}$ // closure of left side of $AC \to D$ using set F
- $(E)^+ = \{ A , C , D , E , H \}$ // closure of left side of $E \to AD$ and $E \to H$ using set F

Step-02:

- $(A)^+ = \{ A , C , D \}$ // closure of left side of $A \to C$ using set G
- $(AC)^+ = \{ A , C , D \}$ // closure of left side of $AC \to D$ using set G
- $(E)^+ = \{ A , C , D , E , H \}$ // closure of left side of $E \to AD$ and $E \to H$ using set G

Step-03:

Comparing the results of Step-01 and Step-02, we find-

- Functional dependencies of set G can determine all the attributes which have been determined by the functional dependencies of set F.
- Thus, we conclude G covers F i.e. $G \supseteq F$.

Determining whether both F and G cover each other-

- From Step-01, we conclude F covers G.
- From Step-02, we conclude G covers F.
- Thus, we conclude both F and G cover each other i.e. F = G.

R (A, C, D, E, H)

F:	G:
$A \rightarrow C$ $AC \rightarrow D$ $E \rightarrow AD$ $E \rightarrow H$	$A \rightarrow CD$ $E \rightarrow AH$
$\{A\}^+ = \{ A, C, D \}$ $\{A, C\}^+ = \{ A, C, D \}$ $\{E\}^+ = \{ A, C, D, E, H \}$	$\{A\}^+ = \{ A, C, D \}$ $\{E\}^+ = \{ A, C, D, E, H \}$

R (P, Q, R, S)

X: $P \rightarrow Q$ $Q \rightarrow R$ $R \rightarrow S$	Y: $P \rightarrow QR$ $R \rightarrow S$
$\{P\}^+ = \{P, Q, R, S\}$ $\{Q\}^+ = \{Q\}$ $\{R\}^+ = \{R, S\}$	$\{P\}^+ = \{P, Q, R, S\}$ $\{R\}^+ = \{R, S\}$

$Y \subseteq X$

Check the equivalence on set of functional dependencies in these questions

Q.1: R (A, B, C)

X:	Y:
$A \rightarrow B$	$A \rightarrow BC$
$B \rightarrow C$	$B \rightarrow A$
$C \rightarrow A$	$C \rightarrow A$

Q.2: R (V, W, X,Y,Z)

X:	Y:
$W \rightarrow X$	$W \rightarrow XY$
$WX \rightarrow Y$	$Z \rightarrow WX$
$Z \rightarrow WY$	
$Z \rightarrow V$	

Canonical Cover in DBMS-

- A canonical cover is a simplified and reduced version of the given set of functional dependencies.
- Since it is a reduced version, it is also called as **Irreducible set**.

$$\alpha \rightarrow \beta$$

Characteristics-

Canonical cover is free from all the unnecessary functional dependencies.

- The closure of canonical cover is same as that of the given set of functional dependencies.
- Canonical cover is not unique and may be more than one for a given set of functional dependencies.

Need-

Working with the set containing unnecessary functional dependencies increases the computation time.

- Therefore, the given set is reduced by eliminating the useless functional dependencies.
- This reduces the computation time and working with the irreducible set becomes easier.

Steps To Find Canonical Cover-

Step-01:

Write the given set of functional dependencies in such a way that each functional dependency contains exactly one attribute on its right side.

The functional dependency $X \rightarrow YZ$ will be written as-

$$X \rightarrow Y$$

$$X \rightarrow Z$$

Step-02:

Consider each functional dependency one by one from the set obtained in Step-01.

- Determine whether it is essential or non-essential.

To determine whether a functional dependency is essential or not, compute the closure of its left side-

- Once by considering that the particular functional dependency is present in the set
- Once by considering that the particular functional dependency is not present in the set

Then following two cases are possible-

Case-01: Results Come Out to be Same-

If results come out to be same,

- It means that the presence or absence of that functional dependency does not create any difference.
- Thus, it is non-essential.
- Eliminate that functional dependency from the set.

NOTE-

Eliminate the non-essential functional dependency from the set as soon as it is discovered.

- Do not consider it while checking the essentiality of other functional dependencies.

Case-01: Results Come Out to be Different-

If results come out to be different,

- It means that the presence or absence of that functional dependency creates a difference.
- Thus, it is essential.
- Do not eliminate that functional dependency from the set.
- Mark that functional dependency as essential.

Step-03:

Consider the newly obtained set of functional dependencies after performing Step-02.

- Check if there is any functional dependency that contains more than one attribute on its left side.

Then following two cases are possible-

Case-01: No-

There exists no functional dependency containing more than one attribute on its left side.

- In this case, the set obtained in Step-02 is the canonical cover.

Case-01: Yes-

There exists at least one functional dependency containing more than one attribute on its left side.

- In this case, consider all such functional dependencies one by one.
- Check if their left side can be reduced.

Use the following steps to perform a check-

- Consider a functional dependency.
- Compute the closure of all the possible subsets of the left side of that functional dependency.
- If any of the subsets produce the same closure result as produced by the entire left side, then replace the left side with that subset.

After this step is complete, the set obtained is the canonical cover.

Example:

The following functional dependencies hold true for the relational scheme R (W , X , Y , Z) –

$$X \rightarrow W$$

$$WZ \rightarrow XY$$

$$Y \rightarrow WXZ$$

Write the irreducible equivalent for this set of functional dependencies.

Step-01:

Write all the functional dependencies such that each contains exactly one attribute on its right side-

$$X \rightarrow W$$

$$WZ \rightarrow X$$

$$WZ \rightarrow Y$$

$$Y \rightarrow W$$

$$Y \rightarrow X$$

$$Y \rightarrow Z$$

Step-02:

Check the essentiality of each functional dependency one by one.

For $X \rightarrow W$:

- Considering $X \rightarrow W$, $(X)^+ = \{ X , W \}$
- Ignoring $X \rightarrow W$, $(X)^+ = \{ X \}$

Now,

- Clearly, the two results are different.
- Thus, we conclude that $X \rightarrow W$ is essential and can not be eliminated.

$X \rightarrow W$

$WZ \rightarrow X$

$WZ \rightarrow Y$

$Y \rightarrow W$

$Y \rightarrow X$

$Y \rightarrow Z$

For WZ → X:

- Considering WZ → X, $(WZ)^+$ = { W , X , Y , Z }
- Ignoring WZ → X, $(WZ)^+$ = { W , X , Y , Z }

Now,

- Clearly, the two results are same.
- Thus, we conclude that WZ → X is non-essential and can be eliminated.

Eliminating WZ → X, our set of functional dependencies reduces to-

X → W

WZ → Y

Y → W

Y → X

Y → Z

Now, we will consider this reduced set in further checks.

X → W

WZ → X

WZ → Y

Y → W

Y → X

Y → Z

For $WZ \rightarrow Y$:

- Considering $WZ \rightarrow Y$, $(WZ)^+ = \{ W , X , Y , Z \}$
- Ignoring $WZ \rightarrow Y$, $(WZ)^+ = \{ W , Z \}$

Now,

- Clearly, the two results are different.
- Thus, we conclude that $WZ \rightarrow Y$ is essential and can not be eliminated.

$X \rightarrow W$

$WZ \rightarrow Y$

$Y \rightarrow W$

$Y \rightarrow X$

$Y \rightarrow Z$

For $Y \rightarrow W$:

- Considering $Y \rightarrow W$, $(Y)^+ = \{ W , X , Y , Z \}$
- Ignoring $Y \rightarrow W$, $(Y)^+ = \{ W , X , Y , Z \}$

Now,

- Clearly, the two results are same.
- Thus, we conclude that $Y \rightarrow W$ is non-essential and can be eliminated.

Eliminating $Y \rightarrow W$, our set of functional dependencies reduces to-

$$X \rightarrow W$$
$$WZ \rightarrow Y$$
$$Y \rightarrow X$$
$$Y \rightarrow Z$$

$$X \rightarrow W$$
$$WZ \rightarrow Y$$
$$Y \rightarrow W$$
$$Y \rightarrow X$$
$$Y \rightarrow Z$$

For $Y \rightarrow X$:

- Considering $Y \rightarrow X$, $(Y)^+ = \{ W, X, Y, Z \}$
- Ignoring $Y \rightarrow X$, $(Y)^+ = \{ Y, Z \}$

Now,

- Clearly, the two results are different.
- Thus, we conclude that $Y \rightarrow X$ is essential and can not be eliminated.

$X \rightarrow W$

$WZ \rightarrow Y$

$Y \rightarrow X$

$Y \rightarrow Z$

For $Y \rightarrow Z$:

Considering $Y \rightarrow Z$, $(Y)^+ = \{ W, X, Y, Z \}$

- Ignoring $Y \rightarrow Z$, $(Y)^+ = \{ W, X, Y \}$

Now,

- Clearly, the two results are different.
- Thus, we conclude that $Y \rightarrow Z$ is essential and can not be eliminated.

From here, our essential functional dependencies are-

$X \rightarrow W$

$WZ \rightarrow Y$

$Y \rightarrow X$

$Y \rightarrow Z$

Step-03:

Consider the functional dependencies having more than one attribute on their left side.

- Check if their left side can be reduced.

In our set,

- Only WZ → Y contains more than one attribute on its left side.
- Considering WZ → Y, $(WZ)^+ = \{ W , X , Y , Z \}$

Now,

- Consider all the possible subsets of WZ.
- Check if the closure result of any subset matches to the closure result of WZ.

$$(W)^+ = \{ W \}$$

$$(Z)^+ = \{ Z \}$$

Clearly,

- None of the subsets have the same closure result same as that of the entire left side.
- Thus, we conclude that we can not write WZ → Y as W → Y or Z → Y.
- Thus, set of functional dependencies obtained in step-02 is the canonical cover.

X → W

WZ → Y

Y → X

Y → Z

Finally, the canonical cover is-

X → W

WZ → Y

Y → X

Y → Z

Q.1

R(A, B, C, D)

$$A \rightarrow B$$

$$C \rightarrow B$$

$$D \rightarrow ABC$$

$$AC \rightarrow D$$

Q.2

R(V, W, X, Y, Z)

$$V \rightarrow W$$

$$VW \rightarrow X$$

$$Y \rightarrow VXZ$$

Ans.1

R(A, B, C, D)

$$A \rightarrow B$$

$$C \rightarrow B$$

$$D \rightarrow AC$$

$$AC \rightarrow D$$

Ans. 2

R(V, W, X, Y, Z)

$$V \rightarrow WX$$

$$Y \rightarrow VZ$$

Super Key, Candidate Key, Primary Key

① $R(A,B,C,D)$

$A \rightarrow BC$

$A^+ = ABC$

what is key $\Rightarrow (key)^+ = R$

$(S.K)^+ = R$

② $R(A,B,C,D)$

$ABC \rightarrow D$

$AB \rightarrow CD$

$A \rightarrow BCD$

$(ABC)^+ = ABCD$

$(AB)^+ = ABCD$

$(A)^+ = ABCD$ C.K

③ $R(ABCD)$

$B \rightarrow ACD$

$ACD \rightarrow B$

$(B)^+ = ACDB$ SK.

$(ACD)^+ = ACDB$ S.K

CK B & ACD

④ $R(ABCD)$

$AB \rightarrow C$

$C \rightarrow BD$

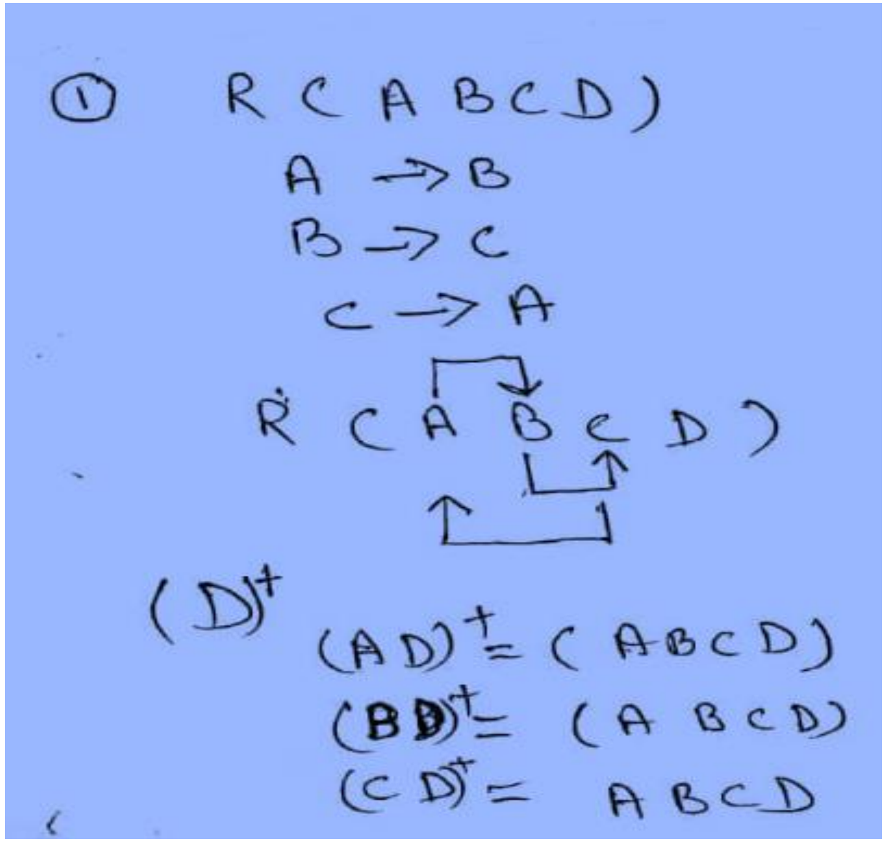
$D \rightarrow A$

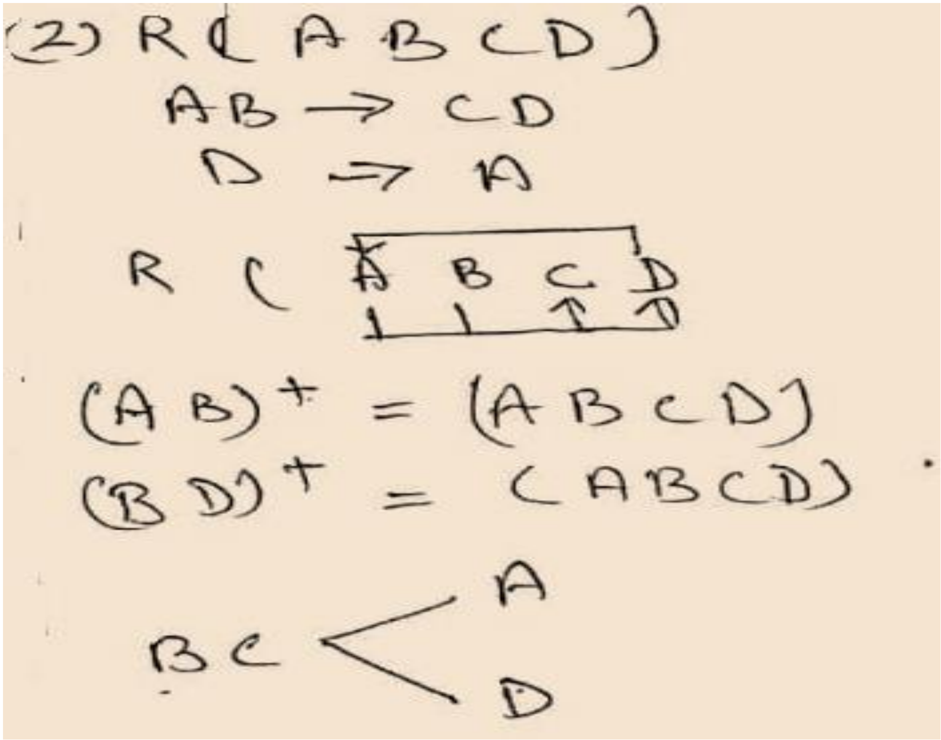
$(AB)^+ = ABCD$ {CK}

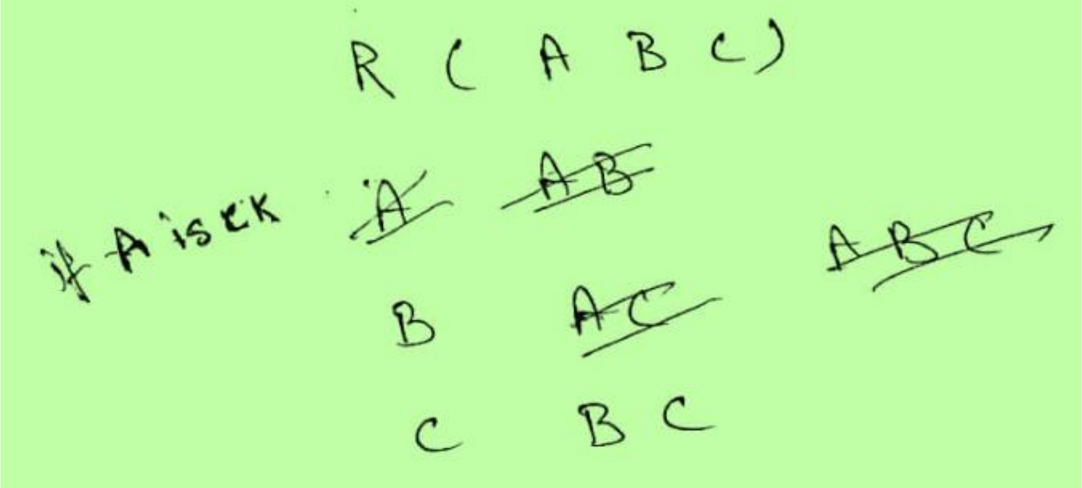
$(C)^+ = CBDA$ CK

$(D)^+ = DA$

Identifying Super Key and Candidate Key Based On Functional Dependencies







③ R(ABCDEF)

$AB \rightarrow C$

$C \rightarrow D$

$B \rightarrow AE$

R(A B C D E F)

$(BF)^+ = (ABCDEF)$

⑤ ~~ABC~~ R(ABCDE)

$AB \rightarrow CD$

$D \rightarrow A$

$BC \rightarrow DE$

R(A B C D E)

$(AB)^+ = (A, B, C, D, E)$ CK

$(BC)^+ = (B, C, D, E, A)$ CK

$(BD)^+ = (B, D, A, C, E)$ CK

$(BE)^+ = (B, E)$

④ R(A B C D)

$AB \rightarrow CD$

$C \rightarrow A$

$D \rightarrow B$

R(A B C D)

$(A)^+ = (A)$ ✗

$(B)^+ = (B)$ ✗

$(C)^+ = (CA)$ ✗

$D^+ = (DB)$ ✗

$AB = (A, B, C, D)$ ✓

$AC = (A, C)$

$AD = (A, D, B, C)$ ✓

$BC = (B, C, A, D)$ ✓

$BD = (B, D)$ ✗

$CD = (C, D, B, A)$

AC < ACB ✗, ACD ✗

BD < BDC ✗, BDA ✗

Q.6

R(W X Y Z)

Z → W

Y → X Z

X W → Y

R(W X Y Z)

$W^+ = (W X)$

$X = (X Y)$

$Y = (Y X Z W)$ CK

$Z = (Z W)$

$WX = (X W Y Z)$ CK

$WZ = (W Z)$

$XZ = (X Z W Y)$ CK

7

R(A B C D E F)

AB → C

DC → AE

E → F

R(A B C D E F)

$(ABD)^+ = (A B C D E F)$ CK

$(BCD)^+ = (A B C D E F)$ CK

$(BDE)^+ = (\cancel{A}\ B\ \cancel{C}\ D\ E\ F)$ ✗

$(BDF)^+ = (B D F)$ ✗

$(BDEF)^+ = (B D E F)$

R(A B C D E)

$CE \rightarrow D$

$D \rightarrow B$

$C \rightarrow A$

② R(A B C D E F G H I J)

$AB \rightarrow C$

$AD \rightarrow GH$

$BD \rightarrow EF$

$A \rightarrow I$

$H \rightarrow J$

③ R(A B C D E)

$A \rightarrow B$

$BC \rightarrow E$

$DE \rightarrow A$

R(A B C D E)

$BC \rightarrow ADE$

$D \rightarrow B$

R(A B C D E F)

$AB \rightarrow C$

$C \rightarrow D$

$D \rightarrow BE$

$E \rightarrow F$

$F \rightarrow A$

R(A B C D E F G H)

$CH \rightarrow G$

$A \rightarrow BC$

$B \rightarrow CFH$

$E \rightarrow A$

$F \rightarrow EG$

Database Normalization

Database Normalization is a technique of organizing the data in the database. Normalization is a systematic approach of decomposing tables to eliminate data redundancy(repetition) and undesirable characteristics like Insertion, Update and Deletion Anomalies. It is a multi-step process that puts data into tabular form, removing duplicated data from the relation tables.

Normalization is used for mainly two purposes,

- Eliminating redundant(useless) data.
- Ensuring data dependencies make sense i.e data is logically stored.

Exam rollno	name	Age	office_tel	Branch Code	Branch Name	hod
1	Ravi	18	94586	101	CSE	Mr. N
2	rohit	19	45677	101	CSE	Mr. N
3	mohit	18	24567	101	CSE	Mr. N
4	ashi	20	12458	101	CSE	Mr. N
5	Ram	18	87967	105	IT	Mr. P
6	sumit	19	45689	105	IT	Mr. P
7	Ayush	18	45789	103	EC	Mr. K
8	Amar	20	45489	103	SOC	Mr. A

Problems Without Normalization

- Insertion Updation and deletion anomalies
- Data Inconsistency
- Increase database size and slow data access

Insertion Anomaly

When certain data cannot be inserted into database without the presence of other data. Suppose for a new admission, until and unless a student opts for a branch, data of the student cannot be inserted, or else we will have to set the branch information as **NULL**.

Also, if we have to insert data of 100 students of same branch, then the branch information will be repeated for all those 100 students.

Exam rollno	name	Age	office_tel	Branch Code	Branch Name	hod
1	Ravi	18	94586	101	CSE	Mr. N
2	rohit	19	45677	101	CSE	Mr. N
3	mohit	18	24567	101	CSE	Mr. N
4	ashi	20	12458	101	CSE	Mr. N
5	Ram	18	87967	105	IT	Mr. P
6	sumit	19	45689	105	IT	Mr. P
7	Ayush	18	45789	103	EC	Mr. K
8	Amar	20	45489	103	SOC	Mr. A

Updation Anomaly

When we want to update a single piece of data, but it must be done at all its copies. What if Mr. N leaves the college? or is no longer the HOD of computer science department? In that case all the student records will have to be updated, and if by mistake we miss any record, it will lead to data inconsistency. This is Updation anomaly.

Deletion Anomaly

When we want to delete some unwanted data, it causes deletion of some wanted data. In our **Student** table, two different information are kept together, Student information and Branch information. Hence, at the end of the academic year, if student records are deleted, we will also lose the branch information. This is Deletion anomaly.

First Normal Form-

A given relation is called in First Normal Form (1NF) if each cell of the table contains only an atomic value.

OR

A given relation is called in First Normal Form (1NF) if the attribute of every tuple is either single valued or a null value.

- This can be done by rewriting the relation such that each cell of the table contains only one value.

Student_id	Name	Subjects
100	Akshay	Computer Networks, Designing
101	Aman	Database Management System
102	Anjali	Automata, Compiler Design

Student_id	Name	Subjects
100	Akshay	Computer Networks
100	Akshay	Designing
101	Aman	Database Management System
102	Anjali	Automata
102	Anjali	Compiler Design

Second Normal Form-

A given relation is called in Second Normal Form (2NF) if and only if-

- Relation already exists in 1NF.
- No partial dependency exists in the relation.

Partial Dependency

A partial dependency is a dependency where few attributes of the candidate key determines non-prime attribute(s).

OR

A partial dependency is a dependency where a portion of the candidate key or incomplete candidate key determines non-prime attribute(s).

In other words,

A → B is called a partial dependency if and only if-

A is a subset of some candidate key

B is a non-prime attribute.

If any one condition fails, then it will not be a partial dependency.

NOTE-

To avoid partial dependency, incomplete candidate key must not determine any non-prime attribute.

However, incomplete candidate key can determine prime attributes

Consider a relation- R (V , W , X , Y , Z) with functional dependencies-

$$VW \rightarrow XY$$

$$Y \rightarrow V$$

$$WX \rightarrow YZ$$

The possible candidate keys for this relation are-

VW , WX , WY

From here,

- Prime attributes = { V , W , X , Y }
- Non-prime attributes = { Z }

Now, if we observe the given dependencies-

- There is no partial dependency.
- This is because there exists no dependency where incomplete candidate key determines any non-prime attribute.

Thus, we conclude that the given relation is in 2NF.

① R (A, B, C, D) ①

$AB \rightarrow D$ $(AB)^+ = A, B, C, D$ CK

$B \rightarrow C$ So AB CK

(P → NP)

A,B will be prime attributes

C,D will be non prime attributes

$AB \rightarrow D$ holds good in 2NF

$B \rightarrow C$ here C is non prime attribute & B is part of C.K. So it is a P.D.

Therefore this relation is not in 2NF.

A	B	
x	—	✓
—	y	✓
—	—	✗
	Z	✓

If table is not in 2N

Then, How to translate it in 2 NF.

① R_1 (A B D) table for C.K and all the attributes which are entirly depend on C.K.

② R_2 (B C) for P.D.

(2) $R(ABCDE)$

$A\ B\ C\ D\ E$

PD $AB \rightarrow C$

PD $D \rightarrow E$

$(ABD)^{+} = (ABCDE)$

- $R_1(ABC)$
- $R_2(DE)$
- $R_3(ABD)$

(3) $R(ABCDE)$

PD ✓ $A \rightarrow B$

$B \rightarrow E$

PD ✓ $C \rightarrow D$

$A\ B\ C\ D\ E$

$(AC)^{+} = (ABCDE)$

- $R_1(ABE)$
- $R_2(C, D)$
- $R_3(AC)$

(f) R(A B C D E F G H I J)

PD AB → C

PD AD → G H

PD BD → E F

PD A → I

H → J

A B D

$(ABD)^+ = R$

- R_1 (A B C)
- R_2 (A, I)
- R_3 (A, D, G, H, J)
- R_4 (B, D, E, F)
- R_5 (A B D)

Third Normal Form-

A given relation is called in Third Normal Form (3NF) if and only if-

- Relation already exists in 2NF.
- No transitive dependency exists for non-prime attributes.

A relation is called in Third Normal Form (3NF) if and only if-

Any one condition holds for each non-trivial functional dependency $X \rightarrow Y$

- Either X is a super key
- Or Y is a prime attribute

A	B	C
a	1	x
b	1	x
c	1	x
d	2	y
e	2	y
f	3	z
g	3	z

A	B
a	1
b	1
c	1
d	2
e	2
f	3
g	3

B	C
1	x
2	y
3	z

A → B

B → C

Transitive Dependency

$X \rightarrow Y$ is called a transitive dependency if and only if-

X is not a super key.

Y is a non-prime attribute.

If any one condition fails, then it is not a transitive dependency.

Or

fd $X \rightarrow Y$ is called a transitive dependency if X and Y are non-prime

Example

Consider a relation- R (A , B , C , D , E) with functional dependencies-

$$A \rightarrow BC$$

$$CD \rightarrow E$$

$$B \rightarrow D$$

$$E \rightarrow A$$

The possible candidate keys for this relation are-

A , E , CD , BC

From here,

- Prime attributes = { A , B , C , D , E }
- There are no non-prime attributes

Now,

- It is clear that there are no non-prime attributes in the relation.
- In other words, all the attributes of relation are prime attributes.
- Thus, all the attributes on RHS of each functional dependency are prime attributes.

Thus, we conclude that the given relation is in 3NF.

① $R(ABCDE)$

$A \to B$

$B \to E$

$C \to D$

$AC^+ = (R)$

$(AC)^+ = (ABCDE)$

ABCDE

- $R_1(ABE)$
  - $R_{11}(A, B)$
  - $R_{12}(B, E)$
- $R_2(C, D)$
- $R_3(AC)$

② $R(ABCDEFGHIJ)$

$AB \to C$

$A \to DE$

$B \to F$

$F \to GH$

$D \to IJ$

$AB^+ = R$

- $R_1(ADEIJ)$
  - $R_{11}(ADE)$ ✓
  - $R_{12}(DIJ)$ ✓
- $R_2(BFGH)$
  - $R_{21}(BF)$ ✓
  - $R_{22}(FGH)$ ✓
- $R_3(A, B, C)$ ✓

③ $R(ABCDE)$

$AB \rightarrow C$

$B \rightarrow D$

$D \rightarrow E$ $\quad AB^{+} = (R)$

- $R_1(BDE)$
  - $R_{11}(BD)$
  - $R_{12}(DE)$
- $R_2(ABC)$

④ $R(ABCDEFGHIJ)$

$AB \rightarrow C$

$AD \rightarrow GH$

$BD \rightarrow EF$

$A \rightarrow I$

$H \rightarrow J$ $\quad (ABD)^{+} \doteq (R)$

- $R_1(ABCI)$
  - $R_{11}(ABC)$
  - $R_{12}(AI)$
- $R_2(ADGHJ)$
  - $R_{21}(ADGH)$
  - $R_{22}(HJ)$
- $R_3(BDEF)$
- $R_4(ABD)$

Boyce-Codd Normal Form-

A given relation is called in BCNF if and only if-

- Relation already exists in 3NF.
- For each non-trivial functional dependency $A \rightarrow B$, A is a super key of the relation.

Example-

Consider a relation- R (A , B , C) with the functional dependencies-

$$A \rightarrow B$$

$$B \rightarrow C$$

$$C \rightarrow A$$

The possible candidate keys for this relation are-

A , B , C

Now, we can observe that RHS of each given functional dependency is a candidate key.

Thus, we conclude that the given relation is in BCNF.

$$R(ABC)$$

$$AB \rightarrow C$$

$$C \rightarrow B$$

$$(AB)^+ = R$$

$$(AC)^+ = R$$

$$\rightarrow R_1(C, B)$$

$$\rightarrow R_2(AC)$$

$$\rightarrow \cancel{R_3(AB)}$$

1. R(ABCDEFGH)

$AB \rightarrow C$

$A \rightarrow DE$

$B \rightarrow F$

$F \rightarrow GH$

AB

1NF

2. R(ABCDE)

$CE \rightarrow D$

$D \rightarrow B$

$C \rightarrow A$

CE

1NF

3. R(ABCDEF)

$AB \rightarrow C$

$DC \rightarrow AE$

$E \rightarrow F$

1NF

ABD

BCD

4. R(ABCDE)

$BC \rightarrow ADE$

$D \rightarrow B$

3NF

BC CD

5. R(ABCDEGHI)

$AB \rightarrow C$

$BD \rightarrow EF$

$AD \rightarrow GH$

$A \rightarrow I$

1NF

ABD

6. R(ABCDE)

$AB \rightarrow CD$

$D \rightarrow A$

$BC \rightarrow DE$

3NF

AB BD BC

(1) $R(VWXYZ)$

$X \rightarrow YV$

$Y \rightarrow Z$

$Z \rightarrow Y$

$VW \rightarrow X$

(2) $R(ABCDEF)$

$ABC \rightarrow D$

$ABD \rightarrow E$

$CD \rightarrow F$

$CDF \rightarrow B$

$BF \rightarrow D$

(3) $R(ABC)$

$A \rightarrow B$

$B \rightarrow C$

$C \rightarrow A$

(4) $R(ABCDE)$

$A \rightarrow BC$

$CD \rightarrow E$

$B \rightarrow D$

$E \rightarrow A$

Answers

1. CK => VW, XW

   1NF

2. CK => ABC, ACD

   1NF

3. CK => A, B, C

   BCNF

4. CK => A,F, CD, BC

   3NF

(1) R (ABCDEF)

$A \to BCDEF$

$BC \to ADEF$

$DEF \to ABC$

(3) R (ABCDE)

$AB \to CD$

$D \to A$

$BC \to DE$

(5) R (ABCDE)

$A \to B$

$B \to E$

$C \to D$

(2) R (ABCDE)

$A \to B$

$BC \to E$

$DE \to A$

(4) R (WXYZ)

$Z \to W$

$Y \to XZ$

$XW \to Y$

Answers

(i) CK ⇒ A, BC, DEF (BCNF)

(ii) CK ⇒ ACD, BCD, CDE (3NF)

(iii) CK ⇒ AB, BC, BD (3NF)

(iv) CK ⇒ Y, XW, XZ (3NF)

(v) CK ⇒ AE (1NF)

① R(ABCDEF)

$C \to F$

$E \to A$

$EC \to D$

$A \to B$

③ R(ABCDEPG)

$AB \to CD$

$DE \to P$

$C \to E$

$P \to C$

$B \to G$

② R(ABCDEH)

$A \to B$

$BC \to D$

$E \to C$

$D \to A$

④ R(ABCDEFGH)

$CH \to G$

$A \to BC$

$B \to CFH$

$E \to A$

$F \to EG$

⑤ R(ABCD)

$A \to B$

$B \to C$

$C \to BD$

⑥ R(ABCDEF)

$AB \to CD$

$CD \to EF$

$BC \to DEF$

$D \to B$

$CE \to F$

Answers

(I) CK => (C E) 1NF

(II) CK => AEH, BEH, DEH 1NF

(III) CK => AB 1NF

(IV) CK => AD, BD, DE, DF, 1NF

(V) CK => A 2NF

(VI) CK => AB, AD, 2NF

4th Normal Form

For a table to satisfy the Fourth Normal Form, it should satisfy the following two conditions:

1. It should be in the **Boyce-Codd Normal Form**.
2. And, the table should not have any **Multi-valued Dependency**.

Multi-valued Dependency

A table is said to have multi-valued dependency, if the following conditions are true,

1. For a dependency A →→ B, if for a single value of A, multiple value of B exists, then the table may have multi-valued dependency.
2. Also, a table should have at-least 3 columns for it to have a multi-valued dependency.
3. And, for a relation R(A,B,C), if there is a multi-valued dependency between, A and B, then B and C should be independent of each other.
4. If all these conditions are true for any relation(table), it is said to have multi-valued dependency. In the EMP candidate key is (Ename, Pname, Dname) and here multivalued dependency exist. So we decompose the relation.

EMP

Ename	Pname	Dname
Smith	X	John
Smith	Y	Anna
Smith	X	Anna
Smith	Y	John

EMP_PROJECTS

Ename	Pname
Smith	X
Smith	Y

EMP_DEPENDENTS

Ename	Dname
Smith	John
Smith	Anna

Consider a relation- R (A, B, C) with functional dependencies and **multivalued dependency**

$A \rightarrow A$

$B \rightarrow B$

$C \rightarrow C$

$A \rightarrow\rightarrow B$

$A \rightarrow\rightarrow C$

The possible candidate key for this relation on the basis of FD is ABC

- **Never use the MVD to compute the CK.**

ABC

Now decompose the relation R (A, B, C)

R1(A, B)

R2(A, C)

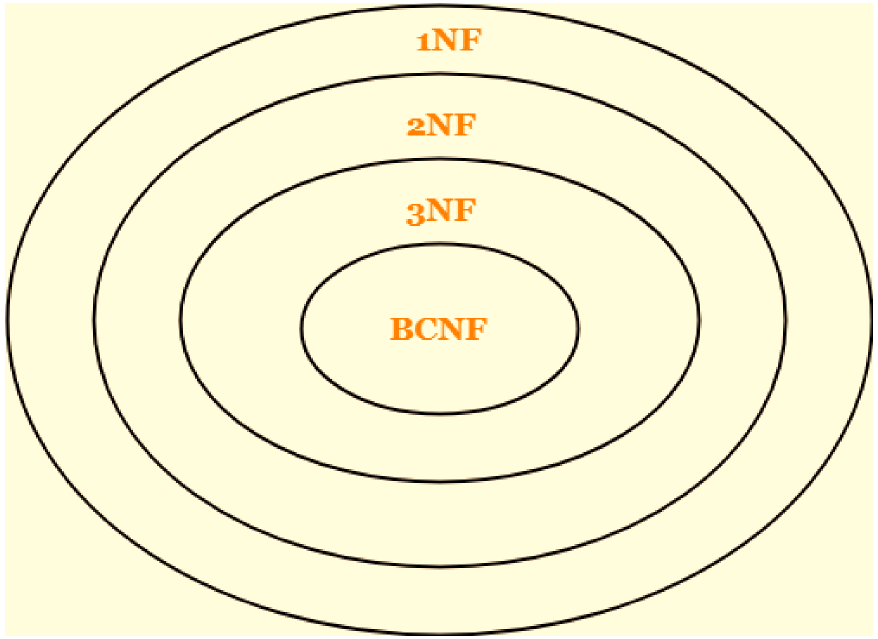
Now, we can observe that there is no multivalued dependency. So, the given relation is in 4NF.

Important Points About Normal Forms in Database

Point-01:

Remember the following diagram which implies-

- A relation in BCNF will surely be in all other normal forms.
- A relation in 3NF will surely be in 2NF and 1NF.
- A relation in 2NF will surely be in 1NF.



Point-02:

The diagram also implies-

- BCNF is stricter than 3NF.
- 3NF is stricter than 2NF.
- 2NF is stricter than 1NF.

Point-03:

While determining the normal form of any given relation,

- Start checking from BCNF.
- This is because if it is found to be in BCNF, then it will surely be in all other normal forms.
- If the relation is not in BCNF, then start moving towards the outer circles and check for other normal forms in the order they appear.

Point-04:

- In a relational database, a relation is always in First Normal Form (1NF) at least.

Point-05:

- Singleton keys are those that consist of only a single attribute.
- If all the candidate keys of a relation are singleton candidate keys, then it will always be in 2NF at least.
- This is because there will be no chances of existing any partial dependency.
- The candidate keys will either fully appear or fully disappear from the dependencies.
- Thus, an incomplete candidate key will never determine a non-prime attribute.

Point-06:

- If all the attributes of a relation are prime attributes, then it will always be in 2NF at least.
- This is because there will be no chances of existing any partial dependency.
- Since there are no non-prime attributes, there will be no **Functional Dependency** which determines a non-prime attribute.

Point-07:

- If all the attributes of a relation are prime attributes, then it will always be in 3NF at least.
- This is because there will be no chances of existing any transitive dependency for non-prime attributes.

Point-08:

- Third Normal Form (3NF) is considered adequate for normal relational database design.

Point-09:

- Every binary relation (a relation with only two attributes) is always in BCNF.

Point-10:

- BCNF is free from redundancies arising out of functional dependencies (zero redundancy).

Point-11:

- A relation with only trivial functional dependencies is always in BCNF.
- In other words, a relation with no non-trivial functional dependencies is always in BCNF.

Point-12:

- BCNF decomposition is always lossless but not always dependency preserving.

Point-13:

- Sometimes, going for BCNF may not preserve functional dependencies.
- So, go for BCNF only if the lost functional dependencies are not required else normalize till 3NF only.

Point-14:

- There exist many more normal forms even after BCNF like 4NF and more.
- But in the real world database systems, it is generally not required to go beyond BCNF.

Point-15:

- Lossy decomposition is not allowed in 2NF, 3NF and BCNF.
- So, if the decomposition of a relation has been done in such a way that it is lossy, then the decomposition will never be in 2NF, 3NF and BCNF.

Point-16:

- Unlike BCNF, Lossless and dependency preserving decomposition into 3NF and 2NF is always possible.

Point-17:

- A prime attribute can be transitively dependent on a key in a 3NF relation.
- A prime attribute can not be transitively dependent on a key in a BCNF relation.

Point-18:

- If a relation consists of only singleton candidate keys and it is in 3NF, then it must also be in BCNF.

Point-19:

- If a relation consists of only one candidate key and it is in 3NF, then the relation must also be in BCNF.

Lossless join decomposition / Nonadditive

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

If a relation R is decomposed into two relation R1 and R2, then it will be loss-less iff

(i) Attr(R1) U attr(R2) = attr(R)

(ii) Attr(R1) ∩ attr(R2) ≠ ф

(iii)Attr(R1) ∩ attr(R2) → attr(R1)

Or

Attr(R1) ∩ attr(R2) → attr(R2)

A	B	C
101	a	p
102	b	q
103	a	r

A	B
101	a
102	b
103	a

B	C
a	p
b	q
a	r

Based on table check decomposition is Lossless or lossy

A	B	C	D	E
a	122	1	p	w
b	234	2	q	x
a	568	1	γ	y
c	347	3	s	z

1. $R_1(A,B)$, $R_2(C\,D)$ ✗
2. $R_1(ABC)$, $R_2(DE)$ ✗
3. $R_1(A,B,C)$, $R_2(CDE)$ ✗
4. $R_1(A,B,C,D)$, $R_2(ACDE)$ ✓
5. $R_1(A,B,C,D)$, $R_2(DE)$ ✓
6. $R_1(A,B,C)$, $R_2(B,C,D)$, $R_3(DE)$ ✓

Based on FD check decomposition is Lossless or lossy

R (V W X Y Z)

$Z \rightarrow Y$

$Y \rightarrow Z$

$X \rightarrow YV$

$VW \rightarrow X$

1. R_1 (V W X) R_2 (X Y Z) ✓
2. R_1 (V W) R_2 (Y Z) ✗
3. R_1 (V W X) R_2 (Y Z) ✗
4. R_1 (V W) R_2 (W X Y Z) ✗

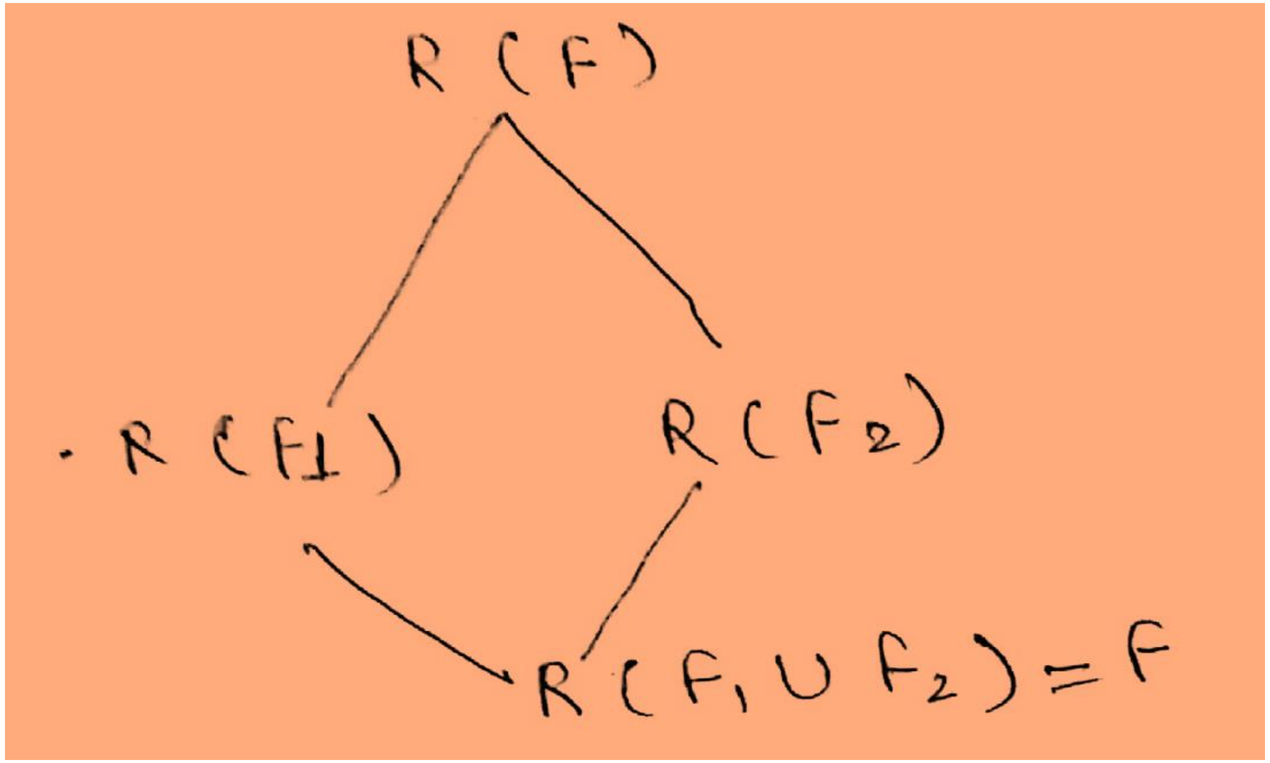
Dependency preserving decomposition

If a relation R is having Functional Dependencies F, is decomposed into two table R1 and R2 having Functional Dependencies set F1 and F2 then

$F1 \subseteq F^+$

$F2 \subseteq F^+$

$(F1 \cup F2)^+ = F^+$



Relation R (A, B, C, D)

AB –> C

C –> D

D –> A

Relation R is decomposed into R1(A, B, C) and R2(C, D)

Let us find closure of F1 and F2

To find closure of F1, consider all combination of ABC. i.e., find closure of A, B, C, AB, BC and AC Note ABC is not considered as it is always ABC

closure(A) = { A } // Trivial
closure(B) = { B } // Trivial
closure(C) = {C, A, D} but D can't be in closure as D is not present R1.
= {C, A}
C--> A // Removing C from right side as it is trivial attribute

closure(AB) = {A, B, C, D}
= {A, B, C}
AB --> C // Removing AB from right side as these are trivial attributes

closure(BC) = {B, C, D, A}
= {A, B, C}
BC --> A // Removing BC from right side as these are trivial attributes

closure(AC) = {A, C, D}
AC --> D // Removing AC from right side as these are trivial attributes

F1 {C--> A, AB --> C, BC --> A}.
Similarly F2 { C--> D }

In the original Relation Dependency { AB --> C , C --> D , D --> A}.
AB --> C is present in F1.
C --> D is present in F2.
D --> A is not preserved.

F1 U F2 is a subset of F. So given decomposition is not dependency preserving.

(1) $R(ABC)$

$A \rightarrow B$ ✓

$B \rightarrow C$ ✓

$C \rightarrow A$ ✓

$R_1(A, B)$	$R_2(B, C)$
F_1: ✓ $A \rightarrow B$	F_2: $B \rightarrow C$ ✓
$B \rightarrow A$	$C \rightarrow B$

$F_1 \cup F_2 = F$

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

So the decomposition is dependency preserving.

(2) $R(ABCD)$

$AB \rightarrow CD$

$D \rightarrow A$

F_1: $R(AD)$

$\cancel{A \rightarrow D}$

$D \rightarrow A$

$R_2(BCD)$

F_2:

$\cancel{B \rightarrow B}$

$\cancel{C \rightarrow C}$

$\cancel{D \rightarrow A}$

$\cancel{BC \rightarrow BC}$

$\cancel{CD \rightarrow A}$

$BD \rightarrow C$

$CD^+ = CDA$

$BD^+ = BDAC$

$(AB^+) = AB$

This is not dependency preserving decomposition

Normalize a relation table DBMS

R (A B C D E F G H I J)

AB → C

B → F

A → DE

D → IJ

F → GH

$(AB)^+ = R$ — CK

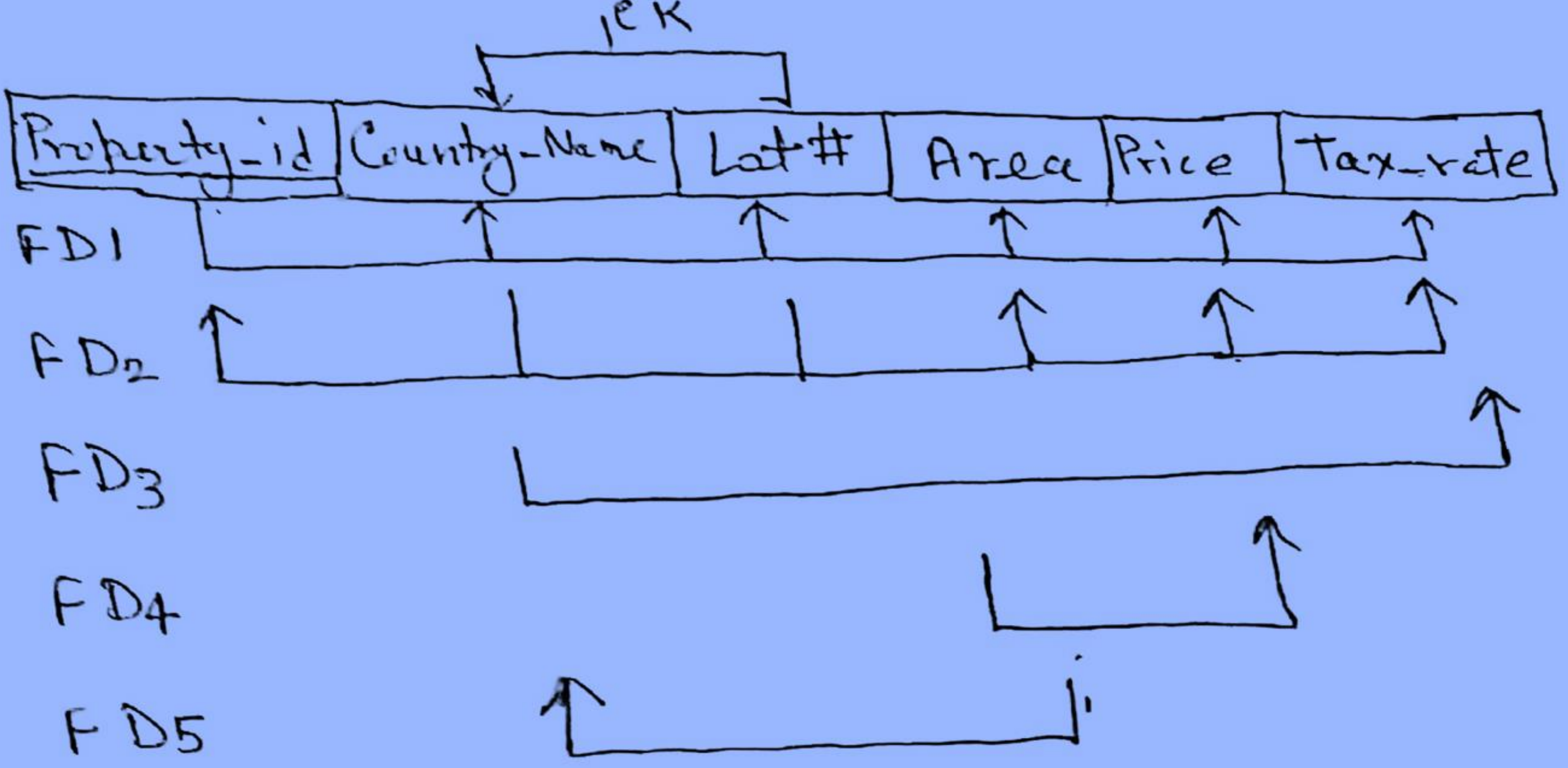
R1 (A B C) —— R1 (A B C)

R2 (A D E I J) → R21 (A D E), R22 (D I J)

R3 (B F G H) → R31 (B F), R32 (F G H)

2NF

3NF & BCNF



[redacted]

Normalize this relation into BCNF