Unit-3

Database Design and Normalization

→ Relational - database design is generated a set of relation schemes to store info., without redundancy, and allow us to retrieve info. easily.

Goal →

→ To avoid redundant data.

→ Ensure that relationships among attributes are represented.

→ Facilitate the checking of updates for violation of db integrity constraints.

Functional Dependencies

Functional Dependencies play a key role in differentiating good db design from bad database design.

It is a method which describes the relationships b/w the attributes.

$$\boxed{X \longrightarrow Y}$$

determinant — X

dependent attribute — Y

{ X determines Y
or
Y is determined by X }

Notes:

eg →

$S_{id} \xrightarrow{} S_{name}$.

→ determinent (Sid) → Dependant (Sname)

Case 1: (valid case)

Sid →	Sname
1	Rohan
2	Rohan

01 or 01 → Rohan
01 or 02 → Rohan

Case 2: (valid case)

Sid →	Sname
1	Rohan
1	Rohan

Case 3: (valid Case)

Sid →	Sname
1	Rohan
2	Varun

Case 4: (Invalid case)

Sid →	Sname
1	Rohan
1	Varun

Notes:

Trivial FD → Always true (valid)

If $X \to Y$ is any functional Dependency then Y is subset of X.

$\therefore$ $X \to Y$ } Trivial.
if $X \subseteq Y$

$L.H.S \cap R.H.S \neq \phi$

eg → $S_{id} S_{name} \to S_{id}$

$\cap$

S_{id} valid.

Non trivial FD

$\boxed{X \to Y}$ $X \cap Y = \phi$

$S_{id} \to S_{name}$, $S_{id} \to Ph.no.$

$E_{id} \to E_{name}$, $E_{id} \to Location$.

We have to check all the four cases here for validation.

Notes:

ACC

__/__/____

Properties of FD

(1)** Reflexivity → if Y is Subset of X then $X \to Y$.

(2) Augmentation → if $X \to Y$, then $XZ \to YZ$

eg → if $\{S_{id} \to S_{name}\}$ then

$\{S_{id}\,Phno \to S_{name}\,Ph\,no.\}$ also valid.

(3)** Transitive → if $X \to Y$ and $Y \to Z$ then $X \to Z$.

eg → $\left\{\begin{array}{l} S_{id} \to S_{name} \\ \quad\text{and} \\ S_{name} \to City \end{array}\right\}$ then.

$S_{id} \to City$

(4) Union: if $X \to Y$ and $X \to Z$ then $X \to YZ$

(5) Decomposition: if $X \to YZ$ the $X \to Y$

Notes

and $X \to Z$.

$(XY \to Z\ ,\ X \to Z,\ Y \to Z)$??

Left hand Side can't be decomposed.

ACC

_/_/_

⑥ Psedutransitivity → if $X \rightarrow Y$ and $WY \rightarrow Z$ then $WX \rightarrow Z$.

⑦ Composition → If $X \rightarrow Y$ and $Z \rightarrow W$ then.

$XZ \rightarrow YW$

④ we can check the validity only for ~~of~~ Non-trivial FD because trivial is a kind of Reflexive which is always valid.

Closure Method

$\boxed{F = F_1 + F_2}$ → F_1 direct, F_2 indirect.

Clouse Set of Attribute

Helps to find all the candidate keys in a relation.

relation ← $R(ABCD)$ → Attributes.

FD $\{A \rightarrow B, B \rightarrow C, C \rightarrow D\}$

Closure of A ← $A^+ = BCDA$ (transitive property)

∴ A determines all the attributes of Relation R.

$\therefore$ A is a Candidate key.

$\Rightarrow$ $B^+ = BCD$

B can not be a candidate key as it can not determine A.

$\Rightarrow$ $C^+ = CD$ (not a candidate key)

$\Rightarrow$ $D^+ = D$ (")

$CK = \{A\}$ Prime attribute = A.

Candidate key.

non-Prime Attribute = $\{B, C, D\}$

$(AB)^+ = ABCD$.
But AB can't be Candidate key as it is not minimal. and by adding anything in Candidate key it become Super key

R(ABCD)

$FD = \{A \to B,\ B \to C,\ C \to D,\ D \to A\}$

$A^+ = ABCD$
$B^+ = BCDA$
$C^+ = CDAB$
$D^+ = DABC$

$CK = \{A, B, C, D\}$

What will be the prime attribute.

{used in the formation of candidate key}

Prime attribute = $\{A, B, C, D\}$

Non-Prime attribute = $\{\phi\}$ (none).

R(ABCDE)

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

$\{BDCA\} \to$ Determined as it comes on right side.

but

E is not determined yet.

E Should be on left side and used to form candidate key.

$E^+ = EC$. → E is not a candidate key but used to form candidate key.

$AE^+ \to ABECD$

$BE^+ \to BECDA$

$CE^+ \to$

$DE^+ \to DEABC$

CK = {AE, BE, DE

Another method to find second candidate key

Check if A or E present at the right hand side.

A, E → not present at right side

↓ replace with D.

check D on right hand side ← D, E^+ = DEABC

↓

BE^+ = BECDA

CE^+ = CE ✗

↓ not determined all the attributes.

AE

∴ CK = {AE, DE, BE}.

Prime attributes = {A, D, B, E}

non-prime attributes = {C}.

Minimal Cover

A → B, C → B, D → ABC, AC → D

Step 1

A → B, C → B, D → A, $\boxed{D \to B}$, D → C, AC → D

Redundant

Removed

Step 2

$A^+ = A$ ∴ A → B is not redundant.

$C^+ = C$ ∴ C → B also included.

$D^+ = DBC$

~~D→B~~ $D^+ = DABC$ → Removed.

$D^+ = DAB$ → not redundant.

$AC^+ = AC$ → not redundant.

Step 3

In left side only one attribute should be there.

$\check{A}C \rightarrow D$.

C^+ = if A comes then A can't be removed.

C^+ = A is not there so A can't be removed.

A^+ = Also C is not comming.

then A and C can't be removed.

$\therefore$ $AC \rightarrow D$ can not be further reduced.

$\therefore$ $A \rightarrow B$, $C \rightarrow B$, $\underbrace{D \rightarrow A \quad D \rightarrow C}_{D \rightarrow AC}$,

$AC \rightarrow D$.

Normalization

It is a technique to remove or reduce redundancy from a relation.

Row level redundancy

SID	Sname	Age
1	RAM	20
2	Varun	25
1	RAM	20

identical.

→ This should not be there.
→ To reduce this we have to set a primary key (unique and not null).

Column level redundancy

SID	Sname	Cid	Cname	Fid	Fname	Salary
1	RAM	C_1	DBMS	F_1	John	30000
2	RAVI	C_2	JAVA	F_2	BOB	40000
3	Nitin	C_1	DBMS	F_1	John	30000
4	Amrit	C_1	DBMS	F_1	John	30000
		C_{10}	MBBS			

When some columns are exactly same, three types of problem is there.

① Insertion anomaly.
② Deletion Anomaly.
③ Updation Anomaly.

①. Insertion Anomaly

When we insert a new course C_{10} in the scheme. it creates problem as we only have course id and name but no S_{id}. As S_{id} is a primary key it can't be null so this creates insertion anomaly.

② Deletion anomaly

→ If we want to remove the data of S_{id} 2.

Delete from student where $S_{id} = 2$.
done

Notes

→ But if only $S_{id} = 2$ is the only student who is studying a particular course. In that case we cannot recover that course and faculty name once we delete this information.

③ Updation Anomaly

→ We want to change Amrit to Amrit Pal.

⇒ Update Student
Set Sname = 'Amrit Pal'
where $S_{id} = 4$.
done.

→ change salary of F_1 from 30k to 40k.

→ it runs multiple time on the server. as F_1 comes multiple times in scheme due to column level duplicacy.

Solution

S_{ID}	Sname

C_{ID}	C_{name}

Notes

FID	Fname	Salary.

First Normal form

→ The table should not contain any mutivalved Attribute.

student

Roll no.	Name	Course.
1	Sonu	c/c++
2	Harsh	Java
3	omker.	c/DBMS

Here Course contain multiple value so this is not in 1NF.

Solution 1.

Rollno.	Name	Course.
1.	Sonu	C
1.	Sonu	C++
2	Harsh	Java
3	omkar	C
3	omkar	DBMS

Here, Primary key can't be Roll no. alone.
but we can have composite primary key i.e. Rollno. Course.

Solution 2.

Rollno.	Name	Course 1	Course 2
1	Sonu	C	C++
2	Harsh	Java	Null
3	omkar	C	DBMS

→ Primary key will be Roll no.

→ but if one student is ~~so~~ enrolled in multiple course and all others are enrolled in ~~do~~ only one course. then this will not be a good representation.

Solution 3 → good Representation

Seperate the multivalued attribute in another table

foreign key.

Rollno.	Name
1	Sonu
2	Harsh
3	omkar

Base table (Primary key exist)

Rollno	Course.
1	C
1	C++
2	Java
3	C
3	DBMS

Primary key → Rollno. Course
Foreign key → Roll no.

2nd Normal form (2NF)

① → The relation must be in 1st Normal form.

②(*) → All the non-prime attributes should be fully functional dependant on Candidate key.

~~Non-prime attributes → The attributes which are not participating~~ should be fully

(*) The attributes which are not participating in the formation of Candidate keys are Non-prime attributes.

~~→ All the non-prime~~

customer

Cust_ID	Store ID	location.
1	1	Delhi
1	3	Mumbai
2	1	Delhi
3	2	Banglore
4	3	Mumbai

→ Location is determined by a part of candidate key.

Notes:

Candidate key → Cust_ID, Store ID

Prime attribute → Cust_ID
Store ID

Non-prime → Location

The given table is not normalized.

First we have to divide it.

Cust_ID	Store_ID
1	1
1	3
2	1
3	2
4	3

↓
Here there is no non-prime attribute.

Store ID	Location
1	Delhi
2	Banglore
3	Mumbai

↓
Now a non-prime attribute (Location) is determined by a Candidate key (Store_ID) so it does not create a problem.

2NF

Notes:

ACC

(2) 2nd rule.

→ There should be no partial dependency in the table / relation.

Part of CK $\xleftarrow{\text{dependent on}}$ Non-prime

Fig: Partial dependency

eg ⇒ (A B) → candidate key.

subset → A or B

→ if this A is determining some attribute C which is a non prime attribute then is called partial dependency

→ This should not be there.

Q: R (A B C D E F)

FD { C → F, E → A, EC → D,
A → B }

ACC

/ /

CK: EC,

(1) Right side element → FADB.

∴ EC = FADB
↓
must be present at left side.

EC^+ = ECFADB { either E or C present on right hand side in FD }

∃ EC Not present in right hand side.

∴ Candidate key = EC (only)

(2) Prime attributes :→ (used to form candidate key)

E, C.

(3) Non-prime attributes = { A, B, D, F }

⇒ Now check whether a part of candidate key is determining a non prime attribute

Here, E and C

C → F → Partial Dependency.

determining non-prime.

∴ table is not in 2NF.

E → A → Partial dependency.

→ LHS should be proper subset of CK and RHS should be a non-prime attribute

Condition for Partial Dependency:

→ Like wise check all the conditions.

if any one is partial dependent then the relation is not in 2NF.

EC → D → Not partial Dependant

↓

not proper subset but only subset.

→ Here Two partial dependancy found So the given relation is not in 2NF.

EF Codd

3rd Normal form.

Rules

1. Table must be in 2NF.
2. There should be no transitive dependency in the table.

Transitive dependency → No non-prime attribute should determine a non-prime attribute.

Student

Roll no.	State	City.
1	Punjab	Moholi
2	Haryana	Ambala
3	Punjab	Moholi
4	Haryana	Ambala
5	Bihar	Patna.

CK = { Roll no. }

FD = Roll no. → State.
State → city.

1. Roll no. → State

Prime → non-prime { valid. }

→ State → City

non-Prime ← non-prime {not valid}

Also,
Roll no. → state and State → City.

∴ Roll no. → City {transitivity}.

· So, Roll no. determining city which is non-prime through another non-prime attribute (i.e. State). through transitivity. This is called <u>transitive dependency.</u>

<u>eg</u> →

R (ABCD)

FD → { AB → C, C → D }

CK : ~~{A, B}~~ {AB}
Prime attributes → {A, B}
non-Prime attributes → {C, D}

$\underset{\text{CK}}{\underline{\text{AB}}}$ → C
Prime → non-Prime {valid}

imp. $\boxed{\text{C} \rightarrow \text{D} \quad \text{non-Prime} \quad \text{non-Prime}}$ {not valid} according to 3NF.

Also.
∵ C → D
∴ AB → D through C which is non-prime attribute

if C is also a prime attribute then there is no problem or if D is prime then also it is valid.

eg
Prime → non prime → no issue
non-prime → non-prime (only condition to check)
non-prime → Prime → no issue

ACC

Q: R(ABCD)

FD = $AB \to CD$, $D \to A$

A: B must be present in CK. but B alone can't come

CK {AB, DB}

$AB^+ = ABCD$

$DB^+ = DBACD$

Prime attributes - {ABD}

Non-Prime Attribute - {C}.

→ Table in in 2NF

Now, check transitive dependency:

→ for Each FD LHS must be a candidate key or super key ~~and~~ or RHS must be prime attribute.

condition for transitive dependency.

① $AB \to CD$ → {valid FD}

↓

candidate key

② $D \to A$ → prime attribute → {valid FD}

false, true

↓

not a candidate

ACC

2nd

∴ This table is in 3NF

Summary:

Theory

① check that table should not have any transitive dependency.

& Non-prime ~~determining~~ Non-prime

Practical

A table will be in 3NF when it doest not contain any transitive dependency

LHS of all FD is candidate/super key

OR

RHS of all FD is prime attribute

Extra in 3NF

A	B	C
a	1	x
b	1	x
c	1	x
d	2	Y
e	2	Y
f	3	Z
g	3	Z

redundancy

$A \to B$, $B \to C$ } transitive

↓ Conversion in 3NF

$R_1(B, C)$

$R_2(A, B)$

A	B
a	1
b	1
c	1
d	2

B	C
1	x
2	Y

ACC

Just. To understand:

Partial → Part of CK → NP (non-Prime)

Transitive → NP → NP

1. if we find a prime at RHS then both the condition become false and the relation is in 3NF

or

2. If LHS is candidate/super key then it can't be a non-prime attribute. Also partial is not possible so again the relation is in 3NF.

Boyce Codd Normal form (BCFN)

⇒ Special Case of 3rd NF (3FN).

⇒ In BCNF every LHS should contain Candidate key for every function dependency.

⇒ No OR Condition as there ~~too~~ is in 3NF

example

Student

Roll no.	Name	voter id	age.
1	Ravi	K0123	20
2	Varun	M034	21
3	Ravi	K786	23
4	Rahul	D286	21

CK: { Roll no., Voter id }

FD: { Roll no. → name
Roll no → voter id
Voter id → age
Voter id → Roll no. }

Notes

~~LHS of~~ $\boxed{\text{LHS should be candidate / super key}}$

① Roll no. → name.

LHS is a candidate key.

∴ valid.

② Roll no. → voter id. } valid.
↓
candidate key

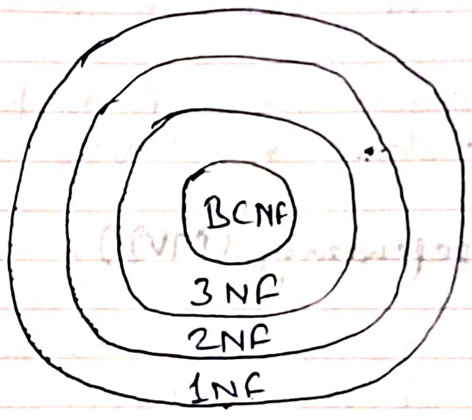
③ Voter id → age } valid.
↓
candidate key

④ voter id → Roll no. } valid.
↓
candidate key

∴ the given table is in BCNF.

Lossless → one should be common and it should be CK.

BCNF → Table should be in 3NF + LHS should be candidate key.



Q: if a table is in 2NF then it must be in 3NF? valid or Not.

<u>Comparision of BCNF and 3NF</u>

① 3NF design is always dependency preserving and lossless. But dependency preserving is difficult to achieve in BCNF sometimes.

② BCNF strictly removes transitive dependency.

③ BCNF relation is in 3NF, but reverse is not possible.

ACC

Multivalued Dependencies (MVDs) and Forth Normal form

A relation 'R' is in 4NF if and only if the following condition are satisfied:

i) 'R' is already in 3NF or BCNF.
ii) If it contains no MVDs.

Multivalued Dependency (MVD).
It is the dependency where one attribute value is potentially a 'multi-valued-fact' about another.

FD : $(\alpha \to \beta)$

(*) there must be three or more attributes
(*) Attributes must be independent of each other.

Student	Mobile no.	Fav. Book
S_1	M_1, M_2	F_1, F_2
S_2	M_3	F_2

ACC

	α Student (S)	β Mobile (M)	Fav.Book (B)
t_1	S_1	M_1	F_1
t_2	S_1	M_2	F_2
	S_2	M_1	F_2
	S_2	M_2	F_1

$(S \twoheadrightarrow M)$
$(S \twoheadrightarrow F)$

MVD:

~~$\alpha \twoheadrightarrow \beta$ if any legal relation $r(R)$ for all pairs of tuple t_1 and t_2 in r such that $t_1[\alpha] = t_2[\alpha]$ then exists tuples t_3 and t_4 in r such that:~~

~~$t_3[\alpha] = t_4[\alpha] = t_1[\alpha] = t_2[\alpha]$~~

$A \twoheadrightarrow B$ is Multivalued if -

$$A_1 < \begin{matrix} B_1 \\ B_2 \end{matrix}$$

Three conditions :

① $A \twoheadrightarrow B$, for a single value of A, more that one value of B exist.

② Table should have at least 3 columns.

③ for this, a table with A, B, C columns, B and C should be independent.

if all these are true then only MVD is there.

Sid	Course	hobby
1	Science	Cricket
1	Maths	Hockey
2	C#	Cricket
2	Php	Hockey

Two tables

Course opted (Sid & course)

1	Science
1	Maths
2	C#
2	Php

Hobbies (Sid, hobby)

1	cricket
1	Hockey
2	cricket
2	Hockey

Two independent table

→ if we add any independent attribute like address in the given table. again the table should split in 3 table to satisfy 4NF.

when we normalize the table, we need to decompose it. For decomposing a table we have to follow these two rules.

1. Decomposition should be lossless.
2. Decomposition should be dependency preserving.

⇒ when we join the table after decomposing it we check wether the join is lossless or lossy.

example:

R

A	B	C
1	2	1
2	2	2
3	3	2

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

A	B
1	2
2	2
3	3

R_1

B	C
2	1
2	2
3	2

R_2

* when we divide a table into two, there should be some attributes which are in common, (at least one), so that when we again join them, then it has to be done on the basis of that common attribute.

* To show a relation b/w two table, atleast one attribute must be there in common.

Q: find the value of C if the value of A = "1".

⇒ To find this we have to join R_1 and R_2 as A is in R_1 and C is in R_2.

Query:- Select R_2.C
from R_2 Natural join R_1
where R_1.A = '1';

A (R_1)	B (R_1)	B (R_2)	C (R_2)
1	2 ⟷	2	1
1	2 ⟷	2	2
1	2	3	2
2	2 ⟷	2	1
2	2 ⟷	2	2
2	2	3	2
3	3	2	1
3	3	2	2
3	3 ⟷	3	2

Natural join { Cross product + Some condition }

A	B	C
1	2	1
1	2	2
2	2	1
2	2	2
3	3	2

extra. tuples (spurious tuples)

Merging R_1 and R_2 after Cartesian.

Original table ~~rows~~ contains 3 rows but after joining the table again we obtain 5 rows. So, this is a problem. And it is called lossy decomposition.

↓

in terms of inconsistency

(*) There is certain criteria for selecting the common attribute. If we do not follow those criteria then these type of problem occurs.

criteria

→ Common attribute should be candidate/super key of either R_1, or R_2 or both.

In our example we are taking B as common attribute in which there is already some duplicacy. That is the reason, ~~of~~ ~~why~~ the whole database become inconsistent after join operation.

A is the appropriate attribute for join as it contains unique values and is a candidate key.

Correct decomposition should be:

$R_1(AB)$ and $R_2(AC)$

Conditions for lossless join Decomposition:

(1) $R_1 \cup R_2 \equiv R$ $[AB \cup AC \equiv ABC]$

(2) $R_1 \cap R_2 \neq \phi$ $[AB \cap AC \equiv A \neq \phi]$

(3) Common attribute must be ck of R_1 or R_2 or Both.

This same condition is used in 5NF.

ACC

Q: R(ABCDEF), check the highest Normal form?

FD : { AB → C, C → DE, E → F, F → A }

Ans Step 1: Find all CK in Relation R.

$AB^+ = ABCDEF$
$A^+ = A$
$B^+ = B$

CK = {AB, FB, EB, CB}

A is present in R.H.S of F → A.

∴ $FB^+ = FBACDE$

F is present in R.H.S of E → F.

$EB^+ = EBFACD$

E is present in R.H.S of C → DE

$CB^+ = CBDEFA$

C is present in R.H.S of AB i.e. CK.

Candidate keys:
{AB, FB, EB, CB}

Notes:

ACC

Step 2:
Write all prime attributes:

{A, B, C, E, F}

Step 3:
Non-prime attributes = {D}

Now we can check for highest Normal form. Starting from BCNF.

① BCNF.

(L.H.S of all FD's should be CK or S/C)

FD=	AB → C	C → DE	E → F	F → A
	AB = CK	C ≠ CK	E ≠ CK	F ≠ CK
	∴ valid	Not valid	Not valid	Not valid.

⇒ If any one condition of the given FD is not valid then it can't be in BCNF.

Notes:

(2) **3rd NF (3NF).**

(check for transitive dependency).

∴ L.H.S = CK **or**
R.H.S = Prime attribute.

AB → C	C → DE	E → F	F → A
valid (already checked)	D is non-prime ∴ Not valid.	F is prime ∴ valid.	A is prime ∴ valid.

C → DE is not valid.
∴ Given table is not in 3NF.

(3) **2NF.**

(Partial dependency should not be there)

{ if. L.H.S. proper subset of CK **and**
R.H.S. non-prime.

→ we have to check only one ~~condition~~ dependency, other 3 are already become valid.

∴ C → DE (DE: Non-prime attribute; C: Proper subset of CB)

→ both conditions are true for partial dependency.

∴ C → DE is not valid for 2NF.

(4) **1NF.**

No multivalue attribute, but it can't be checked. So by-default it should be in 1NF.

Q: How to find Normal form of a Relation.

R (ABCDEF)
CK = {AB, FB, EB, CB}
Prime attributes {A, B, C, E, F}
Non-Prime attributes {D}

FD's = {AB → C, C → D, C → E, E → F, F → A}

→ We assume that it is in 1NF. as ABCDEF are general attributes. we don't have any value for these.

ACC

2NF (No partial dependency).

Partial dependency { LHS = Proper subset of any CK
and
RHS = Non-prime attribute }

① $AB \to C$ (→ Prime) ∴ False and False.
↙ Not proper but only a subset.

② $C \to D$ (→ Non-Prime), ∴ True and True.
↙ Proper subset

$C \to D \Rightarrow$ Partial dependency.

③ $C \to E$ (→ Prime)
↙ Proper Subset

$C \to E \Rightarrow$ Not Partial dependancy

④ $E \to F$ (→ Prime) True and false
↙ Proper Subset

Notes: $E \to F \Rightarrow$ Full dependancy

ACC

⑤ $F \to A$ (→ Prime) True & false.
↙ Proper Subset

$F \to A \Rightarrow$ full dependancy.

⇒ $\underline{\underline{C \to D}}$ → Partial dependancy.

∴ It is not in 2NF.

We have to split the table & to achieve 2NF.

Conditions for split:

① Lossless decomposition
② Dependency should be preserve.

we have,
$R(AB\underline{CD}EF)$
and $C \to D$ is a problem.
So split
& $R_1(ABEF)$ $R_2(CD)$

Notes: But nothing is common in R_1 and R_2.

So we have to take some common attribute and it should be candidate of any one.

In $R_2(CD)$ we have

$C \to D$ $\quad C^+ = CD$

$\therefore$ C is the Candidate key in R_2.

We can add this C (candidate key) to R_1 for adding a common attribute.

Now, we have

$R_1(AB\underline{C}EF)$	$R_2(CD)$
$AB \to C$ $C \to E$ $E \to F$ $F \to A$	$C \to D$

$\downarrow$

2NF.

Now, Consider $R_2(CD)$

$CK = C$

$\boxed{3NF}$

L.H.S must be CK,
OR
R.H.S. prime attribute.

$R_1(ABCEF)$

check
$\{\overset{T}{AB} \to C, \overset{F}{C} \to \overset{T}{E}, E \to \overset{T}{F}, \overset{F}{F} \to \overset{T}{A}\}$

$\therefore$ R_1 is in 3NF.

$R_2(CD)$

$\overset{T}{C} \to D$

candidate key

$\therefore$ R_2 is in 3NF

BCNF

L.H.S must be a CK.

R_1 (ABCEF)

$\{AB \to C \checkmark, C \to E \times, E \to F \times, F \to A \times\}$

R_2 (CD)

$\{C \to D\} \checkmark$

R_2 is in BCNF.

$\therefore$ No redundancy present in R_2.

But in R_1 we have to convert it in BCNF.

$\Rightarrow$ R_1 (ABCEF)

$(AB \to C)$ R_3, $(C \to E)$ R_4, $(E \to F)$ R_5, $(F \to A)$ R_6

R_3 (ABC), R_4 (CE), R_5 (EF), R_6 (FA)

$\Rightarrow$ R_3 (ABC)

$AB^+ = ABC$

$\underline{\underline{AB}}$ = Candidate key.

$\Rightarrow$ R_4 (CE)

$CK = \underline{C}$

$\Rightarrow$ R_5 (EF)

$CK = \underline{E}$

R_6 (FA)

$CK = \underline{\underline{F}}$

Now all the tables are in BCNF as all have the candidate keys at there L.H.S.

Finally we have:

R_2	R_3	R_4	R_5	R_6
(CD)	(ABC)	(CE)	(EF)	(FA)
~~[illegible]~~ $C \to D$	$AB \to E$	$C \to E$	$E \to F$	$F \to A$

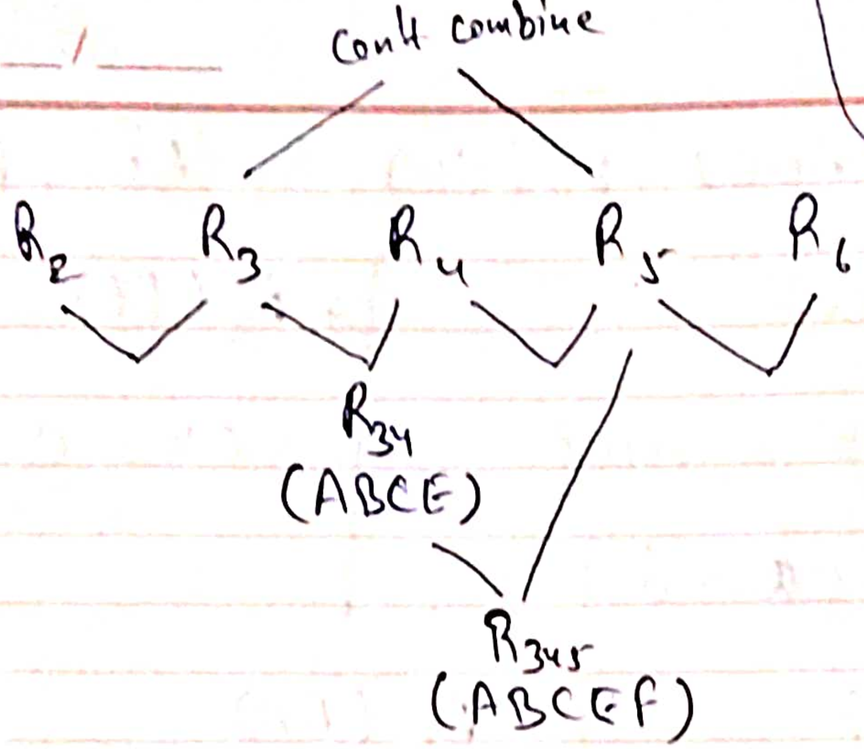
Can Combine as C is common

Can Combine

ABCE

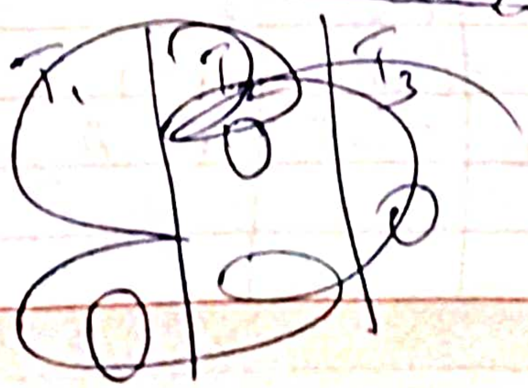
~~$R_3 R_4$~~ $\to R_5$

Precedence Graph
cycle / loop



when nothing is common in two table we can't combine directly but can take help for other table.

T_1	T_2
R(A)	
W(A)	
	R(A)
	W(A)
R(B)	



T_1	T_2
R(A)	
W(A)	
R(B)	
	R(A)
	W(A)