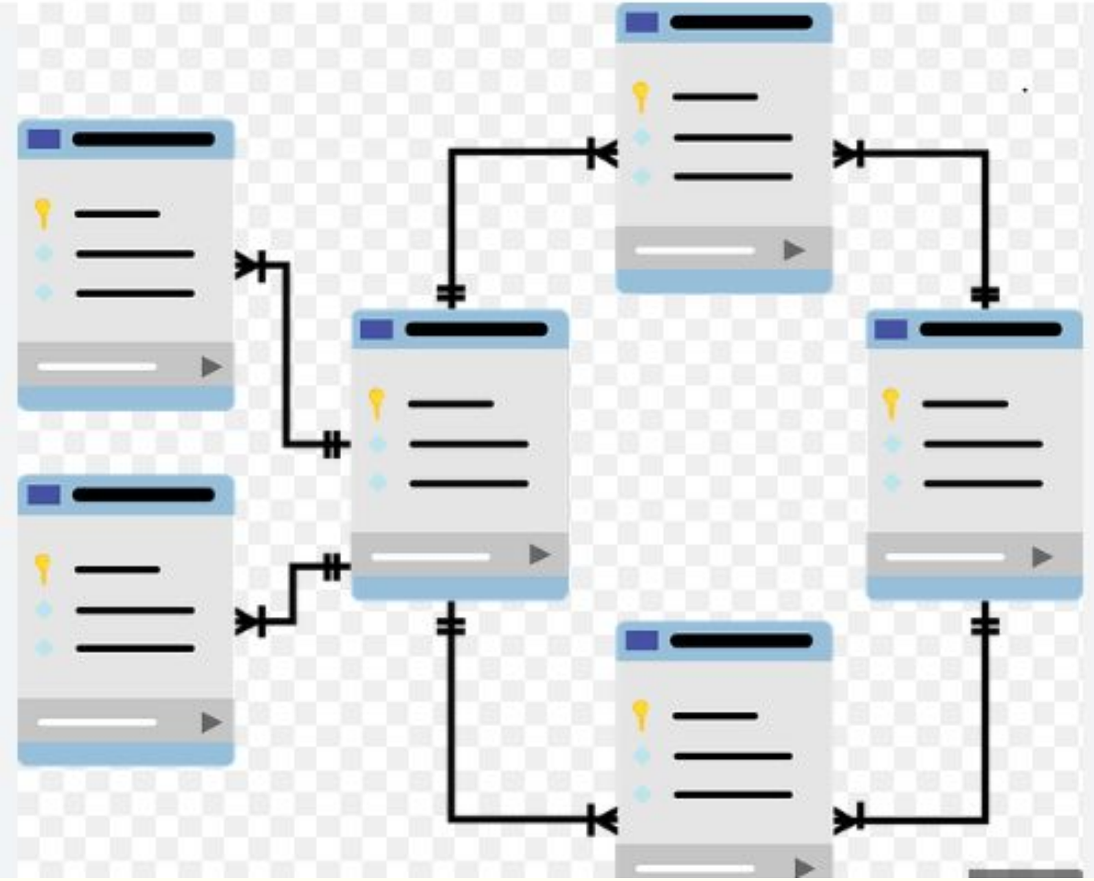
RELATIONAL DATABASE DESIGN

UNIT 4



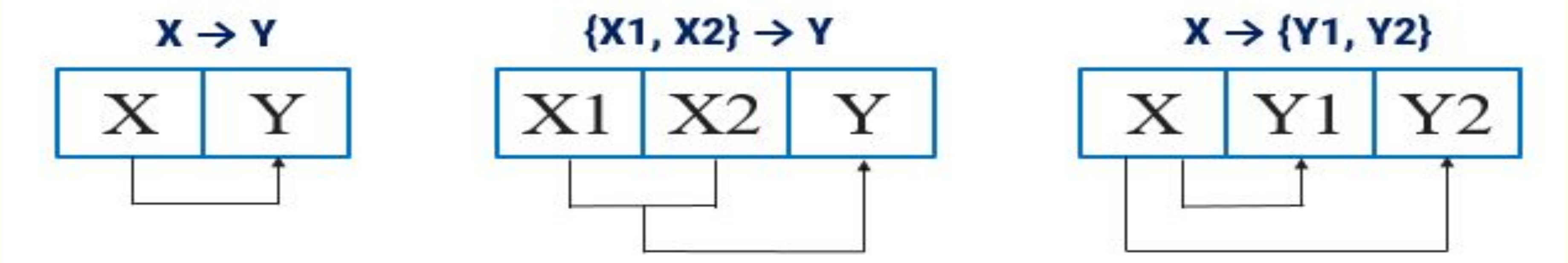
What is Functional Dependency (FD)?

- Let R be a relation schema having n attributes A1, A2, A3,…, An.

Student			
RollNo	**Name**	**SPI**	**BL**
101	Riva	8	0
102	Mitali	7	1
103	Jay	7	0

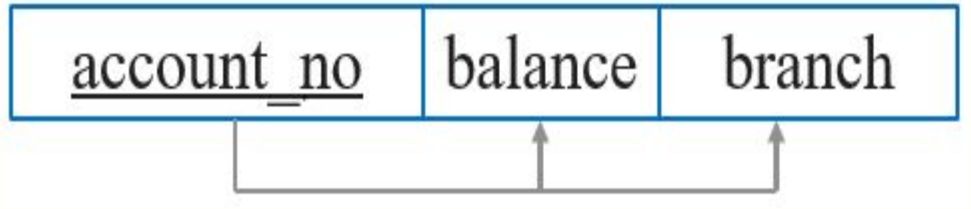
- Let attributes X and Y are two subsets of attributes of relation R.
- If the **values of the X component of a tuple uniquely** (or functionally) **determine the values of the Y component**, then there is a **functional dependency from X to Y**.
- This is denoted by **X** → **Y** (i.e RollNo → Name, SPI, BL).
- It is referred as: **Y is functionally dependent on the X** or **X functionally determines Y**.

Diagrammatic representation of Functional Dependency (FD)



Example

- Consider the relation Account(account_no, balance, branch).
- account_no can determine balance and branch.
- So, there is a functional dependency from account_no to balance and branch.
- This can be denoted by account_no → {balance, branch}.



Types of Functional Dependency (FD)

- **Full Functional Dependency**
  - In a relation, the attribute B is fully functional dependent on A if **B is functionally dependent on A, but not on any proper subset of A**.
  - Eg. {Roll_No, Semester, Department_Name} → SPI
  - We **need all three {Roll_No, Semester, Department_Name} to find SPI**.
- **Partial Functional Dependency**
  - In a relation, the attribute B is partial functional dependent on A if **B is functionally dependent on A as well as on any proper subset of A**.
  - If there is some attribute that can be removed from A and the still dependency holds then it is partial functional dependancy.
  - Eg. {Enrollment_No, Department_Name} → SPI
  - **Enrollment_No is sufficient to find SPI**, Department_Name is not required to find SPI.

Cont…

- **Transitive Functional Dependency**
  - In a relation, if attribute(s) **A → B and B → C, then A → C (means C is transitively depends on A** via B).

Sub_Fac		
Subject	**Faculty**	**Age**
DBMS	Patel	35
DS	Solanki	32
DF	Shah	35

  - Eg. Subject → Faculty & Faculty → Age then Subject → Age
  - Therefore as per the rule of transitive dependency: Subject → Age should hold, that makes sense because if we know the subject name we can know the faculty's age.

Cont…

- **Trivial Functional Dependency**
  - $X \rightarrow Y$ is trivial FD if Y is a subset of X
  - Eg. {Roll_No, Department_Name, Semester} → Roll_No

- **Nontrivial Functional Dependency**
  - $X \rightarrow Y$ is nontrivial FD if Y is not a subset of X
  - Eg. {Roll_No, Department_Name, Semester} → Student_Name

Armstrong's axioms OR Inference rules

- Armstrong's axioms are a set of rules used to infer (derive) all the functional dependencies on a relational database.

Reflexivity

If B is a subset of A then $A \rightarrow B$

Transitivity

If $A \rightarrow B$ and $B \rightarrow C$ then $A \rightarrow C$

Augmentation

If $A \rightarrow B$ then $AC \rightarrow BC$

Pseudo Transitivity

If $A \rightarrow B$ and $BD \rightarrow C$ then $AD \rightarrow C$

Union

If $A \rightarrow B$ and $A \rightarrow C$ then $A \rightarrow BC$

Self-determination

If $A \rightarrow A$

Composition

If $A \rightarrow B$ and $C \rightarrow D$ then $AC \rightarrow BD$

Decomposition

If $A \rightarrow BC$ then $A \rightarrow B$ and $A \rightarrow C$

Closure of a set of FDs

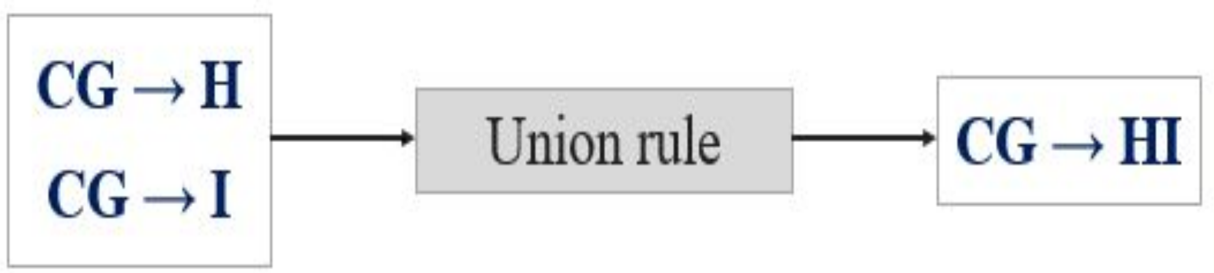
- **What is closure of a set of FDs?**
  - Given a set F set of functional dependencies, there are certain other **functional dependencies that are logically implied by F**.
  - E.g.: F = {A → B and B → C}, then we can infer that A → C (by transitivity rule)
  - The set of **functional dependencies (FDs) that is logically implied by F** is called the closure of F.
  - It is denoted by $\mathbf{F}^+$.

Examples

- Suppose we are given a relation schema **R(A,B,C,G,H,I)** and the set of functional dependencies are:
  - **F = (A → B, A → C, CG → H, CG → I, B → H)**
- The functional dependency **A → H** is logical implied.

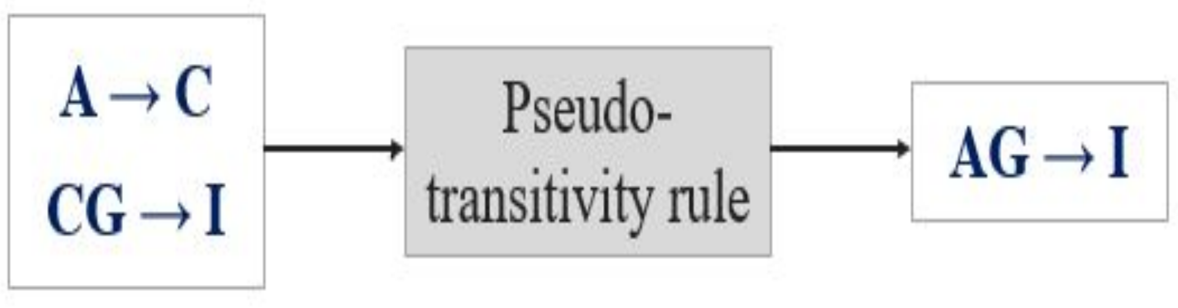
Cont…

- Suppose we are given a relation schema **R(A,B,C,G,H,I)** and the set of functional dependencies are:
  - **F = (A → B, A → C, CG → H, CG → I, B → H)**
- The functional dependency **CG → HI** is logical implied.



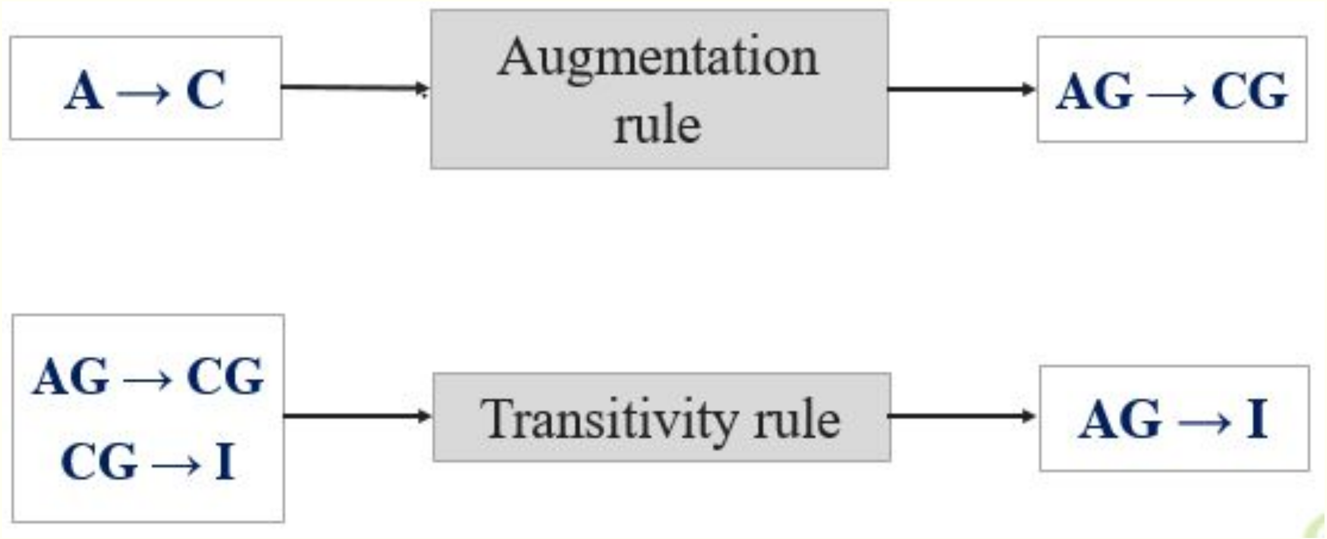
Cont…

- Suppose we are given a relation schema **R(A,B,C,G,H,I)** and the set of functional dependencies are:
  - **F = (A → B, A → C, CG → H, CG → I, B → H)**
- The functional dependency **AG → I** is logical implied.



Cont…

- Suppose we are given a relation schema **R(A,B,C,G,H,I)** and the set of functional dependencies are:
  - **F = (A → B, A → C, CG → H, CG → I, B → H)**
- The functional dependency **AG → I** is logical implied.



Cont…

- Suppose we are given a relation schema **R(A,B,C,G,H,I)** and the set of functional dependencies are:
    - $\mathbf{F = (A \rightarrow B,\ \ A \rightarrow C,\ \ CG \rightarrow H,\ \ CG \rightarrow I,\ \ B \rightarrow H)}$
- Find out the closure of F.

Several members of $\mathbf{F^+}$ are

$$\mathbf{F^+ = (A \rightarrow H,\ CG \rightarrow HI,\ AG \rightarrow I)}$$

Cont…

- Compute the closure of the following set F of functional dependencies FDs for relational schema **R = (A,B,C,D,E,F):**
  - **F = (A → B, A → C, CD → E, CD → F, B → E)**
- Find out the closure of F.

A → B & A → C	Union Rule	**A → BC**
CD → E & CD → F	Union Rule	**CD → EF**
A → B & B → E	Transitivity Rule	**A → E**
A → C & CD → E	Pseudo-transitivity Rule	**AD → E**
A → C & CD → F	Pseudo-transitivity Rule	**AD → F**

F^+ = (A → BC, CD → EF, A → E, AD → E, AD → F)

Cont…

- Compute the closure of the following set F of functional dependencies FDs for relational schema **R = (A,B,C,D,E):**
  - **F = (AB → C, D → AC, D → E)**
- Find out the closure of F.

D → AC	Decomposition Rule	**D → A & D → C**
D → AC & D → E	Union Rule	**D → ACE**

F^+ = (D → A, D → C, D → ACE)

Closure of attribute sets

What is a closure of attribute sets?

- Given a set of attributes α, the closure of α under F is the **set of attributes that are functionally determined by α under F**.
- It is denoted by $\boldsymbol{\alpha}^+$.

Closure of attribute sets [Exercise]

- Given functional dependencies (FDs) for relational schema $R = (A,B,C,D,E)$:
- $F = \{A \rightarrow BC,\ CD \rightarrow E,\ B \rightarrow D,\ E \rightarrow A\}$
  - Find Closure for A
  - Find Closure for CD
  - Find Closure for B
  - Find Closure for BC
  - Find Closure for E

Answer

$A^+ = ABCDE$

$CD^+ = ABCDE$

$B^+ = BD$

$BC^+ = ABCDE$

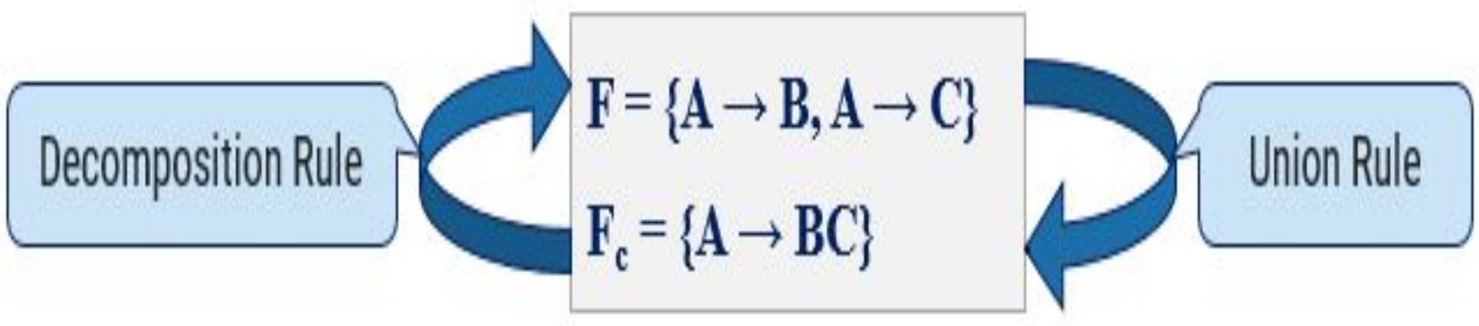
$E^+ = ABCDE$

What is extraneous attributes?

- Let us consider a relation R with schema R = (A, B, C) and set of functional dependencies FDs **F = { AB → C, A → C }**.
- In **AB → C, B is extraneous attribute**. The reason is, there is another FD **A → C**, which means when **A alone can determine C**, the use of B is unnecessary (extra).
- An attribute of a functional dependency is said to be extraneous if we can **remove it without changing the closure of the set of functional dependencies**.

What is canonical cover?

- A canonical cover of F is a **minimal set of functional dependencies** equivalent to F, having **no redundant dependencies or redundant parts of dependencies**.
- It is denoted by $\mathbf{F_c}$
- A canonical cover for F is a set of dependencies F_c such that
  - **F logically implies** all dependencies in $\mathbf{F_c}$ and
  - $\mathbf{F_c}$ **logically implies** all dependencies in **F** and
  - **No** functional dependency in $\mathbf{F_c}$ contains an **extraneous attribute** and
  - Each **left side** of functional dependency in $\mathbf{F_c}$ is **unique**.



Canonical cover [Example]

- Consider the relation schema R = (A, B, C) with FDs

  $F = \{A \rightarrow BC, B \rightarrow C, A \rightarrow B, AB \rightarrow C\}$
- Find canonical cover.

Combine $A \rightarrow BC$ and $A \rightarrow B$ into $A \rightarrow BC$ (Union Rule)

Set is $\{A \rightarrow BC, B \rightarrow C, AB \rightarrow C\}$

A is extraneous in $AB \rightarrow C$

Check if the result of deleting A from $AB \rightarrow C$ is implied by the other dependencies

Yes: in fact, $B \rightarrow C$ is already present

Set is $\{A \rightarrow BC, B \rightarrow C\}$

C is extraneous in $A \rightarrow BC$

Check if $A \rightarrow C$ is logically implied by $A \rightarrow B$ and the other dependencies

Yes: using transitivity on $A \rightarrow B$ and $B \rightarrow C$.

The canonical cover is: **$A \rightarrow B$, $B \rightarrow C$**

Cont…

- Consider the relation schema R = (A, B, C, D, E, F) with FDs

  $F = \{A \rightarrow BC, CD \rightarrow E, B \rightarrow D, E \rightarrow A\}$
- Find canonical cover.

The left side of each FD in F is unique.
Also none of the attributes in the left side or right side of any of the FDs is extraneous.
Therefore the canonical cover F_c is equal to F.
$F_c = \{A \rightarrow BC, CD \rightarrow E, B \rightarrow D, E \rightarrow A\}$

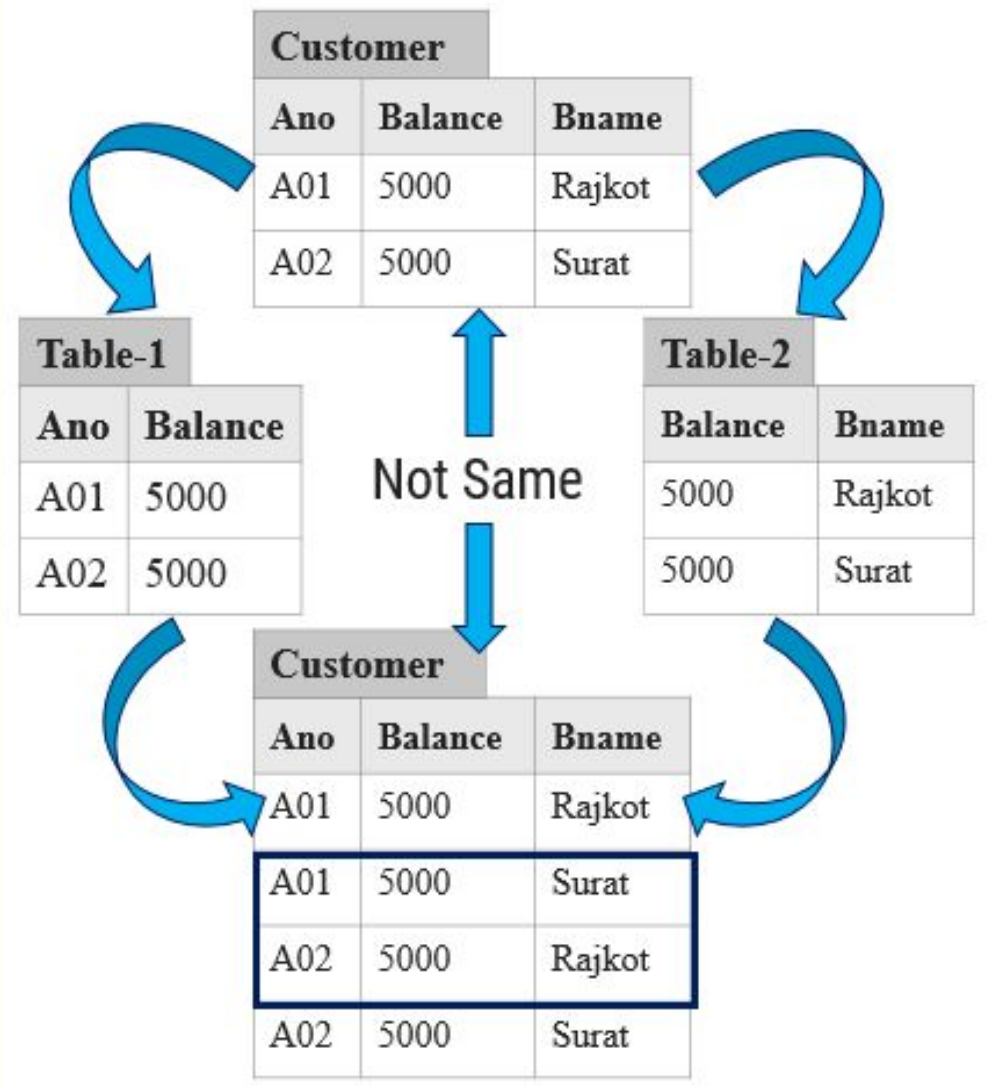
Decomposition

What is decomposition?

- Decomposition is the **process of breaking down given relation** into **two or more relations**.
- Relation R is replaced by two or more relations in such a way that:
  - Each new relation contains a **subset** of the **attributes of R**
  - Together, they all **include all tuples** and **attributes of R**
- Types of decomposition
  - Lossy decomposition
  - Lossless decomposition (non-loss decomposition)

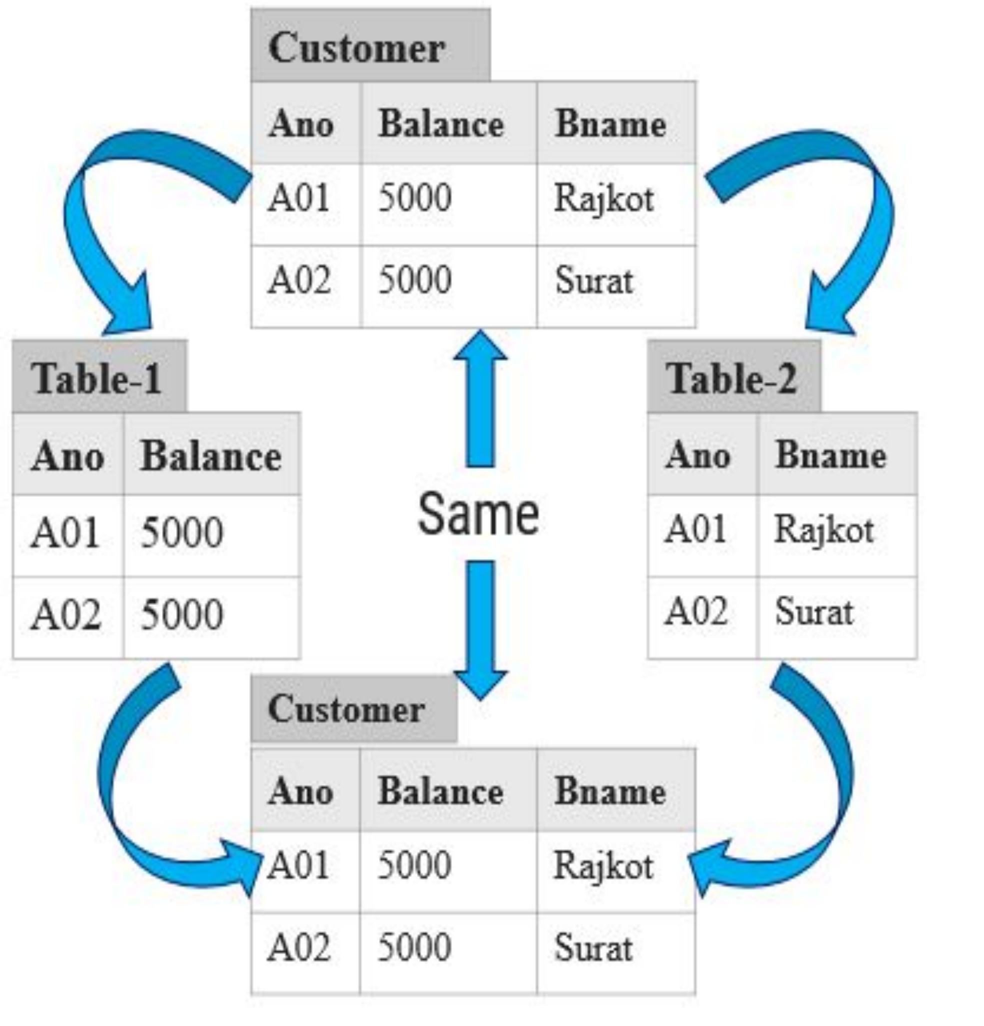
Lossy decomposition

- The decomposition of relation R into R1 and R2 is lossy when the join of R1 and R2 does not yield the same relation as in R.
- This is also referred as lossy-join decomposition.
- The disadvantage of such kind of decomposition is that some information is lost during retrieval of original relation.
- From practical point of view, decomposition should not be lossy decomposition.



Lossless decomposition

- The decomposition of relation R into R1 and R2 is lossless when the join of R1 and R2 produces the same relation as in R.
- This is also referred as a non-additive (non-loss) decomposition.
- All decompositions must be lossless.



Anomaly and its types

❑ What is an anomaly in database design?

- Anomalies are **problems that can occur in poorly planned, un-normalized database** where all the data are stored in one table.
- There are three types of anomalies that can arise in the database because of redundancy are
  - Insert anomaly
  - Delete anomaly
  - Update / Modification anomaly

Insert anomaly

- Consider a relation Emp_Dept(<u>EID</u>, Ename, City, DID, Dname, Manager) EID as a primary key

Emp_Dept

<u>EID</u>	Ename	City	DID	Dname	Manager
1	Raj	Rajkot	1	CE	Shah
2	Meet	Surat	1	CE	Shah
~~NULL~~	NULL	NULL	2	IT	NULL

An insert anomaly occurs when certain attributes cannot be inserted into the database without the presence of another attribute.

Want to insert new department detail (IT)

- Suppose a **new department (IT) has been started** by the organization but **initially there is no employee appointed** for that department.
- We **want to insert that department detail** in Emp_Dept table.
- But the **tuple for this department cannot be inserted** into this table as the **EID will have NULL value, which is not allowed because EID is primary key**.
- This kind of problem in the relation where some tuple cannot be inserted is known as insert anomaly.

Delete anomaly

- Consider a relation Emp_Dept(EID, Ename, City, DID, Dname, Manager) EID as a primary key

Emp_Dept

EID	Ename	City	DID	Dname	Manager
1	Raj	Rajkot	1	CE	Shah
2	Meet	Surat	1	CE	Shah
3	Jay	Baroda	2	IT	Dave

A delete anomaly exists when **certain attributes are lost because of the deletion of another attribute**.

Want to delete (Jay) employee's detail

- Now consider **there is only one employee in some department (IT)** and that **employee leaves the organization**.
- So we **need to delete tuple of that employee (Jay).**
- But in addition to that **information about the department also deleted**.
- This kind of problem in the relation where deletion of some tuples can lead to loss of some other data not intended to be removed is known as delete anomaly.

Update anomaly

- Consider a relation Emp_Dept(<u>EID</u>, Ename, City, Dname, Manager) EID as a primary key

Emp_Dept

<u>EID</u>	Ename	City	Dname	Manager
1	Raj	Rajkot	CE	Sah
2	Meet	Surat	C.E	Shah
3	Jay	Baroda	Computer	Shaah
4	Hari	Rajkot	IT	Dave

An update anomaly exists **when one or more records (instance) of duplicated data is updated, but not all**.

Want to update manager of CE department

- Suppose the **manager of a (CE) department has changed**, this requires that the **Manager in all the tuples corresponding to that department must be changed** to reflect the new status.

- If we **fail to update all the tuples of given department**, then **two different records of employee working in the same department might show different Manager lead to inconsistency** in the database.

Normalization and normal forms

What is normalization?

- Normalization is the **process of removing redundant data** from tables **to improve data integrity, scalability and storage efficiency**.
  - data integrity (completeness, accuracy and consistency of data)
  - scalability (ability of a system to continue to function well in a growing amount of work)
  - storage efficiency (ability to store and manage data that consumes the least amount of space)

- **What we do in normalization?**
  - Normalization generally involves **splitting an existing table into multiple (more than one) tables**, which can be **re-joined or linked** each time a query is issued (executed).

Cont…

How many normal forms are there?

- Normal forms:
  - 1NF (First normal form)
  - 2NF (Second normal form)
  - 3NF (Third normal form)
  - BCNF (Boyce–Codd normal form)
  - 4NF (Forth normal form)
  - 5NF (Fifth normal form)

As we move from 1NF to 5NF **number of tables** and **complexity increases** but **redundancy decreases**.

1NF (First Normal Form)

- Conditions for 1NF

Each **cells of a table should contain a single value**.

- A relation R is in first normal form (1NF) if and only if it **does not contain any composite attribute or multi-valued attributes or their combinations**.

OR

- A relation R is in first normal form (1NF) if and only if **all underlying domains contain atomic values only**.

1NF (First Normal Form) [Example - Composite attribute]

- In customer relation **address is composite attribute** which is further divided into sub-attributes as “Road” and “City”.
- So customer relation is not in 1NF.

Customer

CID	Name	Address
C01	Raju	Jamnagar Road, Rajkot
C02	Mitesh	Nehru Road, Jamnagar
C03	Jay	C.G Road, Ahmedabad

Problem: It is **difficult to retrieve the list of customers living in ’Jamnagar’ city** from customer table.
The reason is that **address attribute is composite attribute** which **contains road name as well as city name in single cell**.
It is possible that **city name word is also there in road name**.
In our example, ’Jamnagar’ word occurs in both records, in first record it is a part of road name and in second one it is the name of city.

Cont…

Customer

<u>CID</u>	Name	Address
C01	Raju	Jamnagar Road, Rajkot
C02	Mitesh	Nehru Road, Jamnagar
C03	Jay	C.G Road, Ahmedabad

Customer

<u>CID</u>	Name	Road	City
C01	Raju	Jamnagar Road	Rajkot
C02	Mitesh	Nehru Road	Jamnagar
C03	Jay	C.G Road	Ahmedabad

Solution: **Divide composite attributes** into **number of sub-attributes** and insert value in proper sub-attribute.

Exercise: Convert below relation into 1NF (First Normal Form)

Person

<u>PID</u>	Full_Name	City
P01	Raju Maheshbhai Patel	Rajkot

Cont…

Student		
Rno	Name	FailedinSubjects
101	Raju	DS, DBMs
102	Mitesh	DBMS, DS
103	Jay	DS, DBMS, DE
104	Jeet	DBMS, DE, DS
105	Harsh	DE, DBMS, DS
106	Neel	DE, DBMS

- In student relation **FailedinSubjects attribute is a multi-valued attribute** which can store more than one values.
- So above relation is not in 1NF.

Problem: It is difficult to retrieve the **list of students failed in 'DBMS' as well as 'DS' but not in other subjects** from student table.

The reason is that FailedinSubjects attribute is multi-valued attribute so it contains more than one value.

Cont…

Student

Rno	Name	FailedinSubjects
101	Raju	DS, DBMs
102	Mitesh	DBMS, DS
103	Jay	DS, DBMS, DE
104	Jeet	DBMS, DE, DS
105	Harsh	DE, DBMS, DS
106	Neel	DE, DBMS

Student

Rno	Name
101	Raju
102	Mitesh
103	Jay
104	Jeet
105	Harsh
106	Neel

Result

RID	Rno	Subject
1	101	DS
2	101	DBMS
3	102	DBMS
4	102	DS
5	103	DS
...	...	...

Solution: Split the table into two tables in such as way that

- the **first table contains all attributes except multi-valued attribute** with same primary key and
- **second table contains multi-valued attribute** and **place a primary key** in it.
- **insert the primary key of first table in the second table as a foreign key.**

2NF (Second Normal Form)

- Conditions for 2NF

> It is **in 1NF** and each **table should contain a single primary key**.

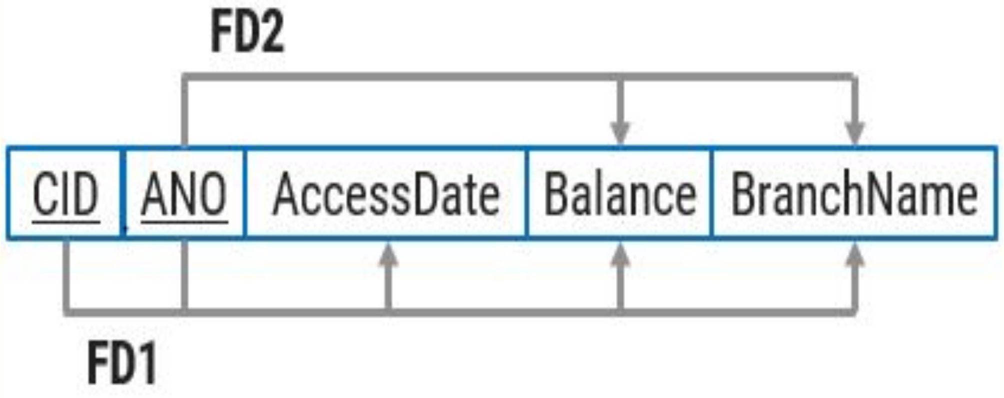
- A relation R is in second normal form (2NF)
  - if and only if it is in **1NF** and
  - **every non-primary key attribute is fully dependent on the primary key**

OR

- A relation R is in second normal form (2NF)
  - if and only if it is in **1NF** and
  - **no any non-primary key attribute is partially dependent on the primary key**

Cont…

Customer				
CID	ANO	AccessDate	Balance	BranchName
C01	A01	01-01-2017	50000	Rajkot
C02	A01	01-03-2017	50000	Rajkot
C01	A02	01-05-2017	25000	Surat
C03	A02	01-07-2017	25000	Surat



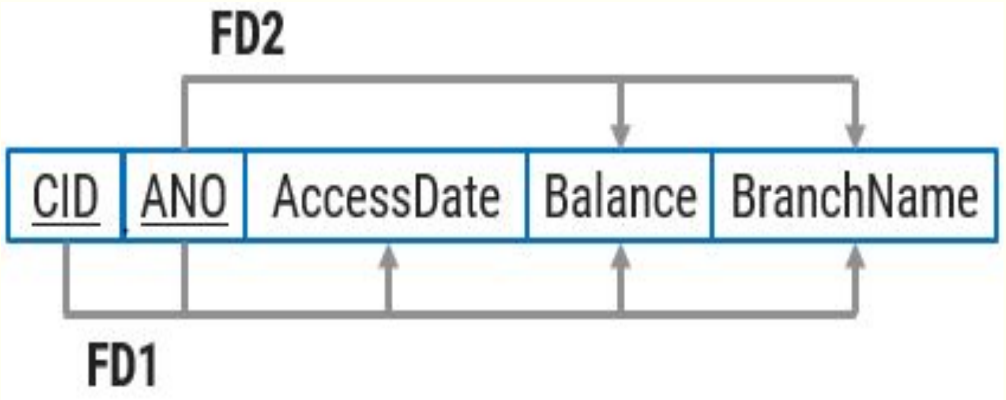
FD1: {CID, ANO} → {AccesssDate, Balance, BranchName}
FD2: ANO → {Balance, BranchName}

Balance and BranchName are partial dependent on primary key (CID + ANO). So customer relation is not in 2NF.

Cont…

Customer

CID	ANO	AccessDate	Balance	BranchName
C01	A01	01-01-2017	50000	Rajkot
C02	A01	01-03-2017	50000	Rajkot
C01	A02	01-05-2017	25000	Surat
C03	A02	01-07-2017	25000	Surat



Problem: For example, in case of a joint account multiple (more than one) customers have common (one) accounts.

If an account **'A01' is operated jointly by two customers** says **'C01' and 'C02'** then **data** values for attributes **Balance and BranchName** will be **duplicated in two different tuples** of customers 'C01' and 'C02'.

Cont…

Customer

CID	ANO	AccessDate	Balance	BranchName
C01	A01	01-01-2017	50000	Rajkot
C02	A01	01-03-2017	50000	Rajkot
C01	A02	01-05-2017	25000	Surat
C03	A02	01-07-2017	25000	Surat

Table-1

ANO	Balance	BranchName
A01	50000	Rajkot
A02	25000	Surat

Table-2

CID	ANO	AccessDate
C01	A01	01-01-2017
C02	A01	01-03-2017
C01	A02	01-05-2017
C03	A02	01-07-2017

Solution: Decompose relation in such a way that **resultant relations do not have any partial FD**.

- **Remove partial dependent attributes** from the relation that violets 2NF.
- **Place them in separate relation** along with the **prime attribute on which they are fully dependent**.
- The **primary key of new relation** will be the **attribute on which it is fully dependent**.
- **Keep other attributes same** as in that table with the **same primary key**.

3NF (Third Normal Form)

- **Conditions for 3NF**

It is in 2NF and there is no transitive dependency.

(Transitive dependency???) A $\rightarrow$ B & B $\rightarrow$ C then A $\rightarrow$ C

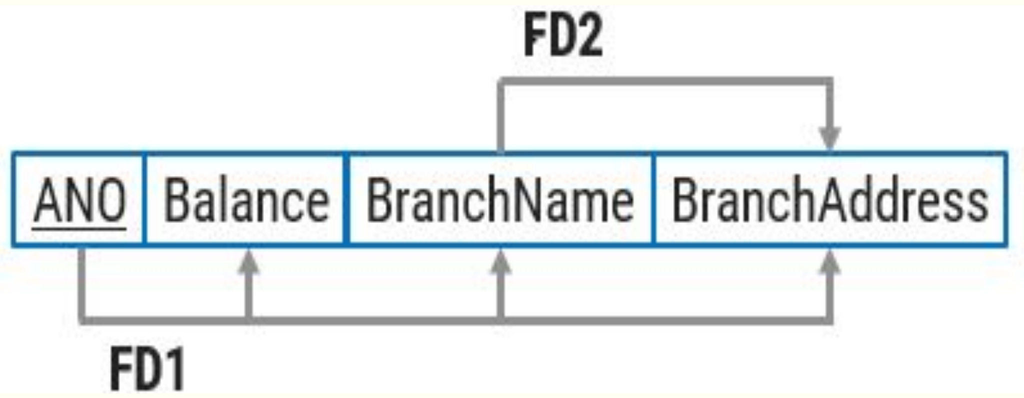
A relation R is in third normal form (3NF)
if and only if it is in **2NF** and
every non-key attribute is non-transitively dependent on the primary key

OR

A relation R is in third normal form (3NF)
if and only if it is in **2NF** and
no any non-key attribute is transitively dependent on the primary key

Cont…

Customer			
ANO	Balance	BranchName	BranchAddress
A01	50000	Rajkot	Kalawad road
A02	40000	Rajkot	Kalawad Road
A03	35000	Surat	C.G Road
A04	25000	Surat	C.G Road

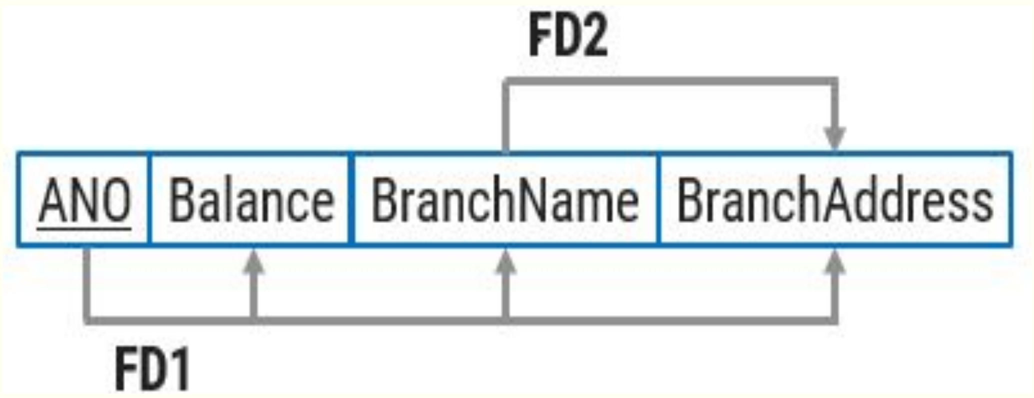


FD1: ANO → {Balance, BranchName, BranchAddress}
FD2: BranchName → BranchAddress
So AccountNO → BranchAddress (Using Transitivity rule)
BranchAddress is transitive depend on primary key (ANO). So customer relation is not in 3NF.

Cont…

Customer

<u>ANO</u>	Balance	BranchName	BranchAddress
A01	50000	Rajkot	Kalawad road
A02	40000	Rajkot	Kalawad Road
A03	35000	Surat	C.G Road
A04	25000	Surat	C.G Road



Problem: In this relation, **branch address will be stored repeatedly** for each account of the same branch which **occupies more space**.

Cont…

Customer			
ANO	Balance	BranchName	BranchAddress
A01	50000	Rajkot	Kalawad road
A02	40000	Rajkot	Kalawad Road
A03	35000	Surat	C.G Road
A04	25000	Surat	C.G Road

Table-1	
BranchName	BranchAddress
Rajkot	Kalawad road
Surat	C.G Road

Table-2		
ANO	Balance	BranchName
A01	50000	Rajkot
A02	40000	Rajkot
A03	35000	Surat
A04	25000	Surat

Solution: Decompose relation in such a way that **resultant relations do not have any transitive FD.**

- **Remove transitive dependent attributes** from the relation that violets 3NF.
- **Place them in a new relation along** with the **non-prime attributes due to which transitive dependency occurred**.
- The **primary key of the new relation** will be **non-prime attributes due to which transitive dependency occurred**.
- **Keep other attributes same as in the table** with **same primary key** and **add prime attributes of other relation into it as a foreign key**.

BCNF (Boyce-Codd Normal Form)

- **Conditions for BCNF**

BCNF is **based on the concept of a determinant**.

It is in **3NF** and **every determinant should be primary key**.

A relation R is in Boyce-Codd normal form (BCNF)
- if and only if it is in 3NF and
- for every functional dependency $X \rightarrow Y$, X should be the primary key of the table.

OR

A relation R is in Boyce-Codd normal form (BCNF)
- if and only if it is in 3NF and
- every prime key attribute is non-transitively dependent on the primary key **OR**

A relation R is in Boyce-Codd normal form (BCNF)
- if and only if it is in 3NF and
- no any prime key attribute is transitively dependent on the primary key

Cont…

Student		
RNO	Subject	Faculty
101	DS	Patel
102	DBMS	Shah
103	DS	Jadeja
104	DBMS	Dave
105	DBMS	Shah
102	DS	Patel
101	DBMS	Dave
105	DS	Jadeja

- **FD1**: RNO, Subject → Faculty
- **FD2**: Faculty → Subject
- So {RNO, Subject} → Subject (Transitivity rule)

In FD2, **determinant is Faculty which is not a primary key**. So student table is not in BCNF.

Problem: In this relation **one student can learn more than one subject with different faculty** then **records will be stored repeatedly for each student, language and faculty combination** which **occupies more space**.

- Here, one faculty teaches only one subject, but a subject may be taught by more than one faculty.
- A student can learn a subject from only one faculty.

Cont…

Student

RNO	Subject	Faculty
101	DS	Patel
102	DBMS	Shah
103	DS	Jadeja
104	DBMS	Dave
105	DBMS	Shah
102	DS	Patel
101	DBMS	Dave
105	DS	Jadeja

Table-1

Faculty	Subject
Patel	DS
Shah	DBMS
Jadeja	DS
Dave	DBMS

Table-2

RNO	Faculty
101	Patel
102	Shah
103	Jadeja
104	Dave
105	Shah
102	Patel
101	Dave
105	Jadeja

- **Solution**: Decompose relation in such a way that resultant relations do not have any transitive FD.
    - Remove transitive dependent prime attribute from relation that violets BCNF.
    - Place them in separate new relation along with the non-prime attribute due to which transitive dependency occurred.
    - The primary key of new relation will be this non-prime attribute due to which transitive dependency occurred.
    - Keep other attributes same as in that table with same primary key and add a prime attribute of other relation into it as a foreign key.

Multivalued dependency (MVD)

- For a dependency X → Y, if **for a single value of X, multiple values of Y exists**, then the **table may have multi-valued dependency**.

Student		
RNO	Subject	Faculty
101	DS	Patel
101	DBMS	Patel
101	DS	Shah
101	DBMS	Shah

- Multivalued dependency (MVD) is denoted by →→
- Multivalued dependency (MVD) is represented as **X** →→ **Y**

4NF (Forth Normal Form)

- Conditions for 4NF
- A relation R is in fourth normal form (4NF)
  - if and only if it is in **BCNF** and
  - **has no multivalued dependencies**

Student

RNO	Subject	Faculty
101	DS	Patel
101	DBMS	Patel
101	DS	Shah
101	DBMS	Shah

Subject

RNO	Subject
101	DS
101	DBMS

Faculty

RNO	Faculty
101	Patel
101	Shah

- Above student table **has multivalued dependency**. So student table is **not in 4NF**.

Functional dependency & Multivalued dependency

- A table can have both functional dependency as well as multi-valued dependency together.
  - RNO → Address
  - RNO →→ Subject
  - RNO →→ Faculty

Student

RNO	Address	Subject	Faculty
101	C. G. Road, Rajkot	DS	Patel
101	C. G. Road, Rajkot	DBMS	Patel
101	C. G. Road, Rajkot	DS	Shah
101	C. G. Road, Rajkot	DBMS	Shah

Subject

RNO	Subject
101	DS
101	DBMS

Faculty

RNO	Faculty
101	Patel
101	Shah

Address

RNO	Address
101	C. G. Road, Rajkot

5NF (Fifth Normal Form)

- Conditions for 5NF
- A relation R is in fifth normal form (5NF)
  - if and only if it is in **4NF** and
  - it **cannot have a lossless decomposition in to any number of smaller tables** (relations).

Student_Result

RID	RNO	Name	Subject	Result
1	101	Raj	DBMS	Pass
2	101	Raj	DS	Pass
3	101	Raj	DF	Pass
4	102	Meet	DBMS	Pass
5	102	Meet	DS	Fail
6	102	Meet	DF	Pass
7	103	Suresh	DBMS	Fail
8	103	Suresh	DS	Pass

Student_Result relation is **further decomposed** into sub-relations. So the above relation is **not in 5NF**.

Cont…

Student_Result

RID	RNO	Name	Subject	Result
1	101	Raj	DBMS	Pass
2	101	Raj	DS	Pass
3	101	Raj	DF	Pass
4	102	Meet	DBMS	Pass
5	102	Meet	DS	Fail
6	102	Meet	DF	Pass
7	103	Suresh	DBMS	Fail
8	103	Suresh	DS	Pass

Student

RNO	Name
101	Raj
102	Meet
103	Suresh

Subject

SID	Name
1	DBMS
2	DS
3	DF

Result

RID	RNO	SID	Result
1	101	1	Pass
2	101	2	Pass
3	101	3	Pass
4	102	1	Pass
5	102	2	Fail
6	102	3	Pass
7	103	1	Fail
8	103	2	Pass

None of the above relations can be further decomposed into sub-relations. So the above database is in 5NF.

Thank you…!!!