✅ **Functional Dependency**
✅ **Partial Dependency**
✅ **Transitive Dependency**
✅ **Multivalued Dependency**
✅ **Denormalization**

Each with **definition**, **example**, and **detailed explanation**.

✅ 1. Functional Dependency (FD)

🔹 **Definition:**

A **functional dependency** occurs when the value of one attribute (or a group of attributes) **uniquely determines** the value of another attribute in a relation.

📌 If A → B, then **B is functionally dependent on A**
(i.e., knowing A, you can uniquely determine B)

🔸 **Example:**

StudentID	Name	Course
101	Alice	Java
102	Bob	Python

Here:

- StudentID → Name
  (If you know StudentID, you can get the Name)
- StudentID → Course

So:

- StudentID functionally determines Name and Course

✅ **Summary:**

- **A → B**: means B depends on A
- A should be a **determinant**
- FD is the **foundation** of normalization and keys

✅ 2. Partial Dependency

🔹 **Definition:**

A **partial dependency** exists when a **non-prime attribute** is functionally dependent on **part of a composite primary key** and **not the whole key**.

🔸 Appears **only** when:

- The table has a **composite primary key**
- A column depends on only a **subset** of that key

🔸 **Example:**

StudentID	Course	StudentName
101	Java	Alice
101	DBMS	Alice

Assume:
Primary Key = (StudentID, Course)

Check:

- StudentName depends on StudentID only
- Not on full composite key → **Partial Dependency**

🧨 **Problem:**

- StudentName is repeated unnecessarily
- This breaks **2NF** rules

✅ **Fix:**

Split the table:

- Student(StudentID, StudentName)
- Enrollment(StudentID, Course)

✅ **3. Transitive Dependency**

🔹 **Definition:**

A **transitive dependency** occurs when a **non-key column depends on another non-key column**, which itself depends on the **primary key**.

📌 A → B and B → C ⇒ A → C
→ This is **indirect dependency**

🔸 **Example:**

EmpID	EmpName	DeptID	DeptName
E101	Alice	D1	HR
E102	Bob	D2	Sales

Assume:

- Primary Key = EmpID
- EmpID → DeptID
- DeptID → DeptName
  → So, **EmpID → DeptName** = transitive dependency

🧨 **Problem:**

- If you change DeptName for D1, it affects multiple rows
- Redundant storage of DeptName

✅ **Fix (in 3NF):**

- Employee(EmpID, EmpName, DeptID)
- Department(DeptID, DeptName)

✅ **4. Multivalued Dependency (4NF topic)**

🔹 **Definition:**

A **multivalued dependency** occurs when one attribute in a table **determines multiple values** of another attribute **independently** of other columns.

📌 A →→ B (multivalued dependency notation)

🔸 **Example:**

StudentID	Language	Hobby
S1	English	Cricket

StudentID	Language	Hobby
S1	English	Music
S1	French	Cricket
S1	French	Music

Explanation:

- StudentID →→ Language
- StudentID →→ Hobby
- Language and Hobby are **independent of each other**
  → This is a **multivalued dependency**

✅ **Fix (4NF):**

Split into two tables:

- Student_Language(StudentID, Language)
- Student_Hobby(StudentID, Hobby)

✅ **5. Denormalization**

🔹 **Definition:**

Denormalization is the process of **intentionally adding redundancy** into a database **to improve read performance**, especially when joins become expensive.

🔸 **When to Denormalize?**

- When the SELECT queries are too slow
- When joins across multiple tables cause overhead
- For reporting, analytics, dashboards

🔸 **Example:**

Instead of:

Order Table:

OrderID	CustomerID
O1	C101

Customer Table:

CustomerID	Name
C101	Alice

We create a denormalized table:

OrderID	CustomerID	CustomerName
O1	C101	Alice

This avoids JOIN at runtime.

⚠️ **Trade-off:**

Pros	Cons
Faster read performance	Data redundancy
Fewer joins	Higher risk of inconsistency
Great for reporting	Needs syncing if data updates