

## FUNCTIONAL DEPENDENCY

The functional dependency is a relationship that exists between two attributes. It typically exists between the primary key and non-key attribute within a table.

$$1. X \rightarrow Y$$

The left side of FD is known as a determinant, the right side of the production is known as a dependent.

### For example:

Assume we have an employee table with attributes: Emp\_Id, Emp\_Name, Emp\_Address.

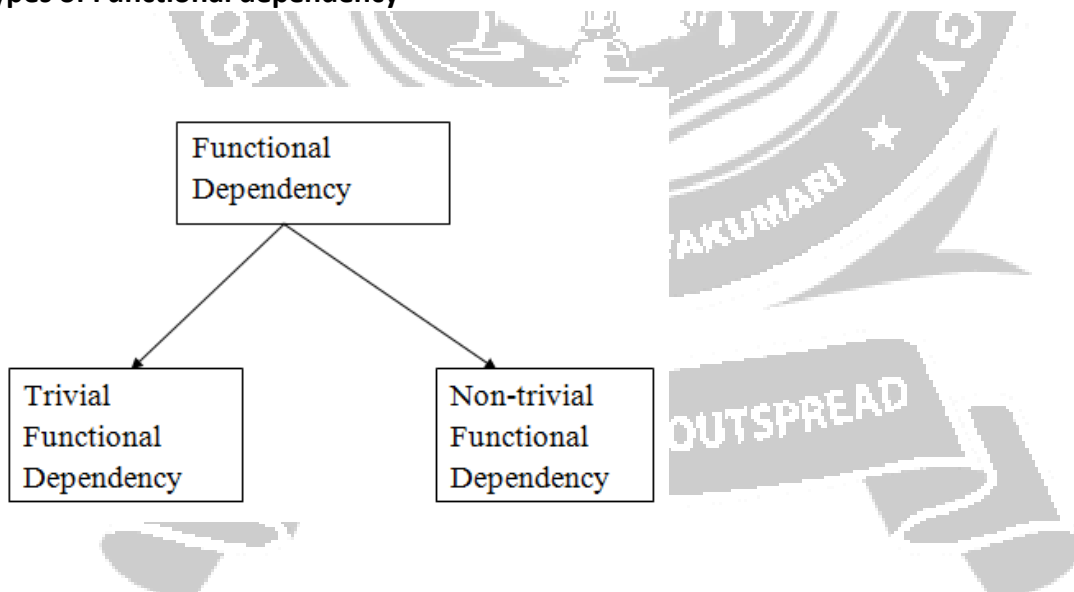
Here Emp\_Id attribute can uniquely identify the Emp\_Name attribute of employee table because if we know the Emp\_Id, we can tell that employee name associated with it.

Functional dependency can be written as:

$$\text{Emp\_Id} \rightarrow \text{Emp\_Name}$$

We can say that Emp\_Name is functionally dependent on Emp\_Id.

### Types of Functional dependency



#### 1. Trivial functional dependency

- $A \rightarrow B$  has trivial functional dependency if B is a subset of A.
- The following dependencies are also trivial like:  $A \rightarrow A$ ,  $B \rightarrow B$

#### Example:

1. Consider a table with two columns Employee\_Id and Employee\_Name.
2.  $\{Employee\_id, Employee\_Name\} \rightarrow Employee\_Id$  is a trivial functional dependency as
3. Employee\_Id is a subset of  $\{Employee\_Id, Employee\_Name\}$ .
4. Also,  $Employee\_Id \rightarrow Employee\_Id$  and  $Employee\_Name \rightarrow Employee\_Name$  are trivial dependencies too.

**2. Non-trivial functional dependency**

- $A \rightarrow B$  has a non-trivial functional dependency if B is not a subset of A.
- When  $A \cap B$  is NULL, then  $A \rightarrow B$  is called as complete non-trivial.

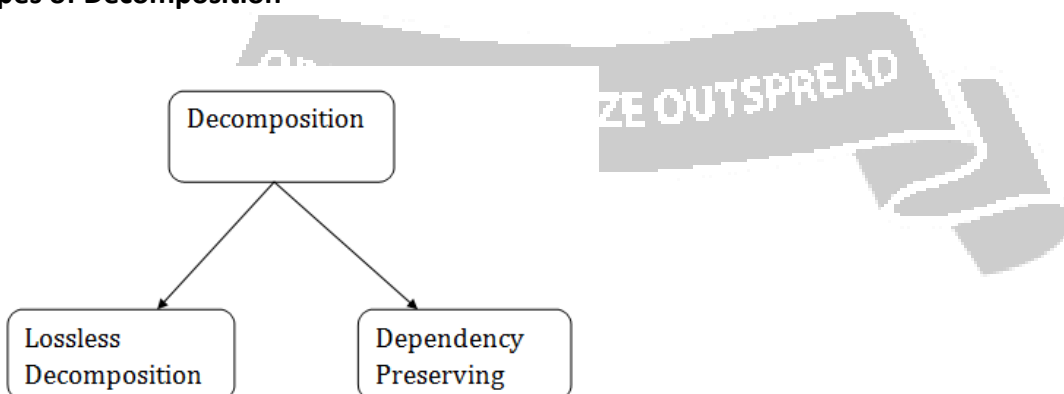
**Example:**

1. ID  $\rightarrow$  Name,
2. Name  $\rightarrow$  DOB

**RELATIONAL DECOMPOSITION**

- When a relation in the relational model is not in appropriate normal form then the decomposition of a relation is required.
- In a database, it breaks the table into multiple tables.
- If the relation has no proper decomposition, then it may lead to problems like loss of information.
- Decomposition is used to eliminate some of the problems of bad design like anomalies, inconsistencies, and redundancy.

**Types of Decomposition**



**NONLOSS DECOMPOSITION Or (Lossless Decomposition)**

- If the information is not lost from the relation that is decomposed, then the decomposition will be lossless.
- The lossless decomposition guarantees that the join of relations will result in the same relation as it was decomposed.
- The relation is said to be lossless decomposition if natural joins of all the decomposition give the original relation.

**Example:****EMPLOYEE\_DEPARTMENT table:**

EMP_ID	EMP_NAME	EMP_AGE	EMP_CITY	DEPT_ID	DEPT_NAME
22	Denim	28	Mumbai	827	Sales
33	Alina	25	Delhi	438	Marketing
46	Stephan	30	Bangalore	869	Finance
52	Katherine	36	Mumbai	575	Production
60	Jack	40	Noida	678	Testing

The above relation is decomposed into two relations EMPLOYEE and DEPARTMENT

**EMPLOYEE table:**

EMP_ID	EMP_NAME	EMP_AGE	EMP_CITY
22	Denim	28	Mumbai
33	Alina	25	Delhi
46	Stephan	30	Bangalore
52	Katherine	36	Mumbai
60	Jack	40	Noida

**DEPARTMENT table**

DEPT_ID	EMP_ID	DEPT_NAME
827	22	Sales
438	33	Marketing
869	46	Finance
575	52	Production
678	60	Testing

Now, when these two relations are joined on the common column "EMP\_ID", then the resultant relation will look like:

#### Employee ⋈ Department

EMP_ID	EMP_NAME	EMP_AGE	EMP_CITY	DEPT_ID	DEPT_NAME
22	Denim	28	Mumbai	827	Sales
33	Alina	25	Delhi	438	Marketing
46	Stephan	30	Bangalore	869	Finance
52	Katherine	36	Mumbai	575	Production
60	Jack	40	Noida	678	Testing

Hence, the decomposition is Lossless join decomposition.