**Module-1**

**CHAPTER 1.4**

**(Functional dependencies )**

**What is Functional Dependency?**

Functional dependency in DBMS, as the name suggests is a relationship between attributes of a table dependent on each other. Introduced by E. F. Codd, it helps in preventing data redundancy and gets to know about bad designs.

To understand the concept thoroughly, let us consider P is a relation with attributes A and B. Functional Dependency is represented by **-> (arrow sign)**

Then the following will represent the functional dependency between attributes with an arrow sign −

**A -> B**

A functional dependency is a constraint between two sets of attributes from the database. Suppose that our relational database schema has n attributes **A1, A2, ..., An.** If we think of the whole database as being described by a single universal relation schema **R = {A1, A2, ... , An}.**

A functional dependency, denoted by **X ->Y,** between two sets of attributes X and Y that are subsets of R, such that any two tuples t1 and t2 in r that have **t1[X] = t2[X]**, they must also have **t1[Y] = t2[Y]**.

This means that the values of the Y component of a tuple in r depend on, or are determined by, the values of the X component; We say that the values of the X component of a tuple uniquely (or functionally) determine the values of the Y component. We say that there is a functional dependency from X to Y, or that Y is functionally dependent on X.

**Example**

The following is an example that would make it easier to understand functional dependency −

We have a **<Department>** table with two attributes − **DeptId**and **DeptName**.

**DeptId = Department ID**

**DeptName = Department Name**

The **DeptId**is our primary key. Here, **DeptId**uniquely identifies the **DeptName**attribute. This is because if you want to know the department name, then at first you need to have the **DeptId**.

**DeptId DeptName**

001 Finance

002 Marketing

003 HR

Therefore, the above functional dependency between **DeptId**and **DeptName**can be determined as **DeptId**is functionally dependent on **DeptName** −

**DeptId -> DeptName**

**Advantages of Functional Dependency**

* Functional Dependency avoids data redundancy. Therefore same data do not repeat at multiple locations in that database
* It helps you to maintain the quality of data in the database
* It helps you to defined meanings and constraints of databases
* It helps you to identify bad designs
* It helps you to find the facts regarding the database design

**Types of Functional Dependency**

Functional Dependency has three forms −

* Trivial Functional Dependency
* Non-Trivial Functional Dependency
* Completely Non-Trivial Functional Dependency

**Trivial Functional Dependency**

It occurs when B is a subset of A in −

A ->B

**Example**

We are considering the same **<Department>** table with two attributes to understand the concept of trivial dependency.

The following is a trivial functional dependency since **DeptId**is a subset of **DeptId**and **DeptName**

**{ DeptId, DeptName } -> Dept Id**

**Non –Trivial Functional Dependency**

It occurs when B is not a subset of A in −

**A ->B**

**Example**

**DeptId -> DeptName**

The above is a non-trivial functional dependency since DeptName is a not a subset of DeptId.

**Completely Non - Trivial Functional Dependency**

It occurs when A intersection B is null in −

**A ->B**

**Armstrong’s Axioms Property of Functional Dependency**

Armstrong’s Axioms property was developed by William Armstrong in 1974 to reason about functional dependencies.

The property suggests rules that hold true if the following are satisfied:

* **Transitivity**
* If A->B and B->C, then A->C i.e. a transitive relation.
* **Reflexivity**
* A-> B, if B is a subset of A.
* **Augmentation**
* The last rule suggests: AC->BC, if A->B

**Transitive dependency**

A functional dependency is said to be transitive if it is indirectly formed by two functional dependencies.

For e.g.

**X -> Z** is a transitive dependency if the following three functional dependencies hold true:

* **X->Y**
* **Y does not ->X**
* **Y->Z**

**Note:** A transitive dependency can only occur in a relation of three of more attributes. This dependency helps us normalizing the database in 3NF (3rd Normal Form).

**Example**: Let’s take an example to understand it better:

Book Author Author\_age

Game of Thrones George R. R. Martin 66

Harry Potter J. K. Rowling 49

Dying of the Light George R. R. Martin 66

{**Book} ->{Author} (if we know the book, we knows the author name)**

**{Author} does not ->{Book}**

**{Author} -> {Author\_age}**

Therefore as per the rule of **transitive dependency**: **{Book} -> {Author\_age}** should hold, that makes sense because if we know the book name we can know the author’s age.

**Multi-valued dependency**

When existence of one or more rows in a table implies one or more other rows in the same table, then the Multi-valued dependencies occur.

If a table has attributes P, Q and R, then Q and R are multi-valued facts of P.

It is represented by double arrow −

**->->**

For our example:

**P->->QP->->R**

In the above case, Multivalued Dependency exists only if Q and R are independent attributes.

A table with multivalued dependency violates the 4NF.

**Example**

Let us see an example &mins;

**<Student>**

**StudentName CourseDiscipline Activities**

Amit Mathematics Singing

Amit Mathematics Dancing

Yuvraj Computers Cricket

Akash Literature Dancing

Akash Literature Cricket

Akash Literature Singing

In the above table, we can see Students **Amit**and **Akash**have interest in more than one activity.

This is multivalued dependency because **CourseDiscipline**of a student are independent of Activities, but are dependent on the student.

Therefore, multivalued dependency −

**StudentName ->-> CourseDisciplineStudentName ->-> Activities**

The above relation violates Fourth Normal Form in Normalization.

To correct it, divide the table into two separate tables and break Multivalued Dependency −

**<StudentCourse>**

**StudentName CourseDiscipline**

Amit Mathematics

Amit Mathematics

Yuvraj Computers

Akash Literature

Akash Literature

Akash Literature

**<StudentActivities>**

**StudentName Activities**

Amit Singing

Amit Dancing

Yuvraj Cricket

Akash Dancing

Akash Cricket

Akash Singing

This breaks the multivalued dependency and now we have two functional dependencies −

StudentName -> CourseDisciplineStudentName - > Activities

**Other References**

[Transitive dependency in DBMS (beginnersbook.com)](https://beginnersbook.com/2015/04/transitive-dependency-in-dbms/)

[Functional dependency in DBMS (tutorialspoint.com)](https://www.tutorialspoint.com/Functional-dependency-in-DBMS)

[Join dependency in DBMS (tutorialspoint.com)](https://www.tutorialspoint.com/Join-dependency-in-DBMS)

[Multivalued dependency in DBMS (tutorialspoint.com)](https://www.tutorialspoint.com/Multivalued-dependency-in-DBMS)

**Suggested Book References**

1. J. Date, “An Introduction to Database Systems”,Addison Wesley.
2. Navathe,“Fundamentals of Database System”, The Benjamin / Cummings Publishing Co.