SOMAIYA
VIDYAVIHAR UNIVERSITY
K J Somaiya College of Engineering

Full Functional Dependency

(X→Y)

DMS - IA2

Class : SY - IT (B3)

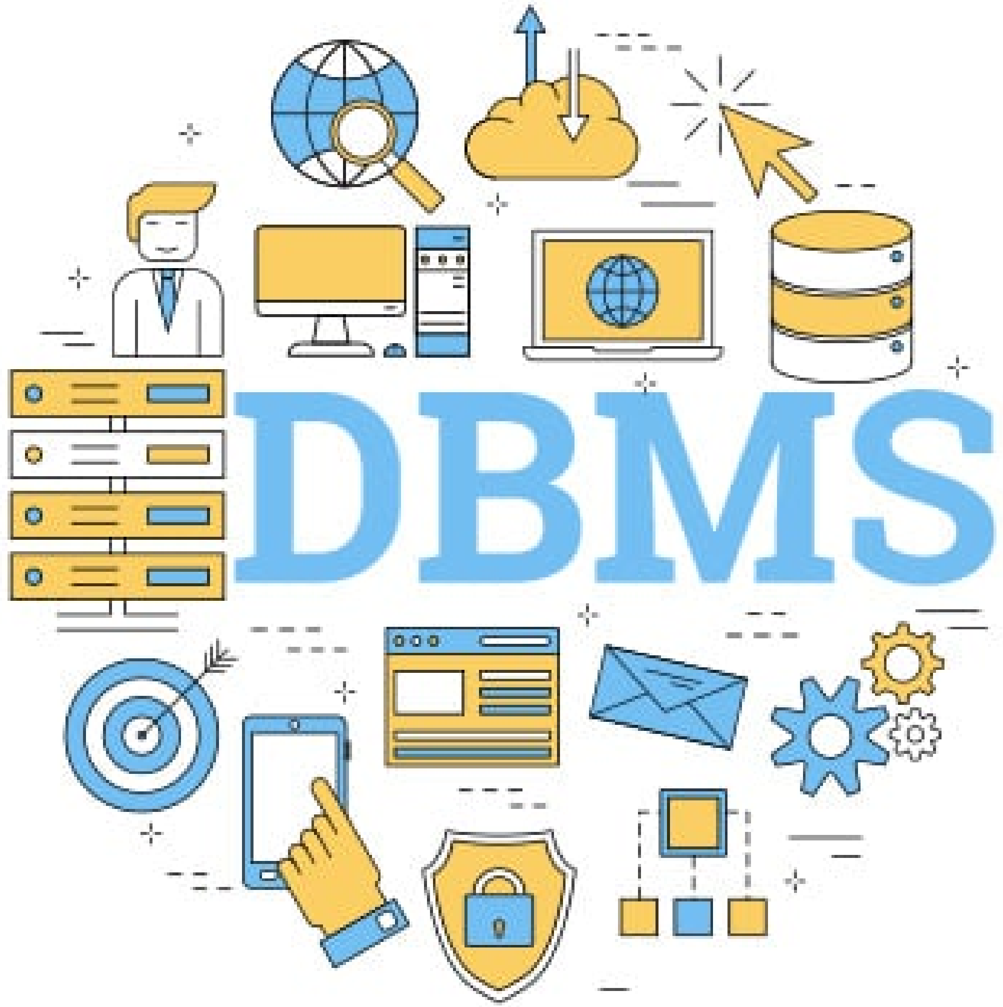
Ritesh Jha
16010423076

Introduction to Functional Dependency (FD)

Functional Dependency (FD) is a constraint that describes the relationship between attributes in a relational database.

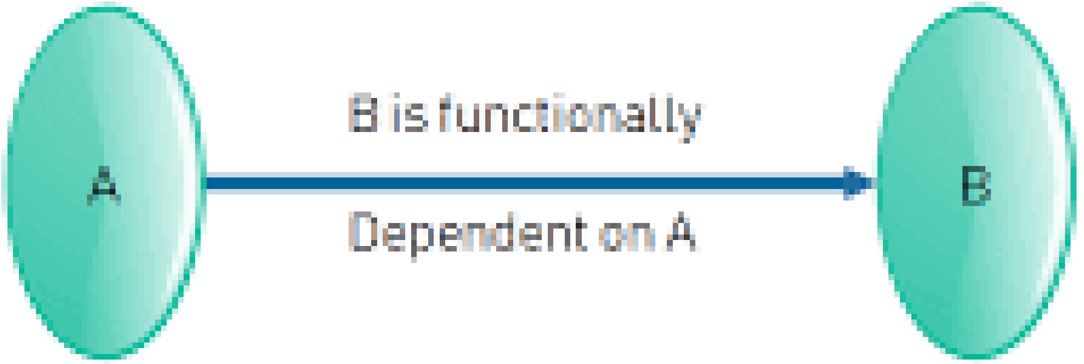
Definition: If **X** is an attribute or a set of attributes, and **Y** is another attribute, then **X -> Y** signifies that Y is functionally dependent on X.

Example: In a student database, **Student_ID -> Student_Name** indicates that the student name depends on the student ID.



Understanding Full Functional Dependency (FFD)

- **Full Functional Dependency (FFD)** is a type of dependency where an attribute is fully dependent on a primary key.
- **Definition:** An attribute B is fully functionally dependent on an attribute A if **A -> B** and removing any part of A makes B no longer dependent on it.
- It is crucial for maintaining data integrity and avoiding redundancy.



KEY TERMINOLOGY IN FUNCTIONAL DEPENDENCIES

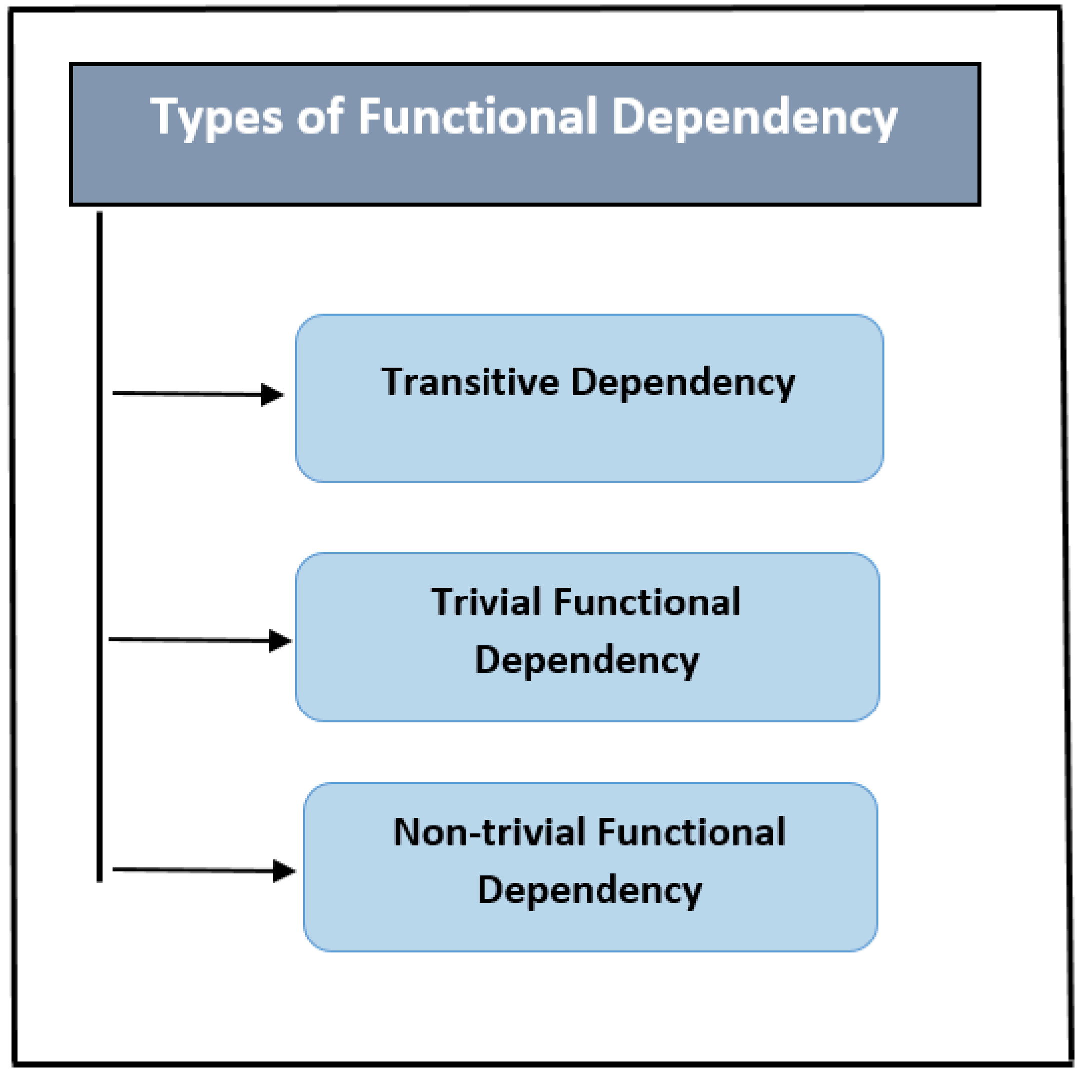
- **Primary Key:** Unique identifier for a record in a table.
- **Composite Key:** A key that consists of two or more attributes.
- **Partial Dependency:** Dependency where a non-key attribute depends on part of a composite key.
- **Transitive Dependency:** Dependency where an attribute depends indirectly on a primary key.
- **Candidate Key:** A minimal superkey that can uniquely identify a record in a table, where no subset of attributes within it can uniquely identify a record on its own.
- **Super Key:** Any combination of attributes that can uniquely identify a record in a table, including primary keys, candidate keys, and composite keys.

Importance of Full Functional Dependency

- **Ensures Data Integrity:** Avoids anomalies and ensures each attribute correctly reflects the entity.
- **Aids Normalization:** Helps achieve Second Normal Form (2NF) in database normalization.
- **Reduces Redundancy:** Prevents unnecessary duplication of data.
- **Enhances Query Efficiency:** By ensuring each attribute is dependent only on the primary key, FFD simplifies data retrieval and reduces the time needed for complex queries.
- **Facilitates Database Maintenance:** With FFD in place, the structure is clearer and easier to maintain, as modifications or updates affect fewer parts of the database, minimizing potential issues.

Types of Functional Dependencies

- **Full Functional Dependency:** A non-key attribute is fully dependent on the primary key.
- **Partial Dependency:** A non-key attribute is dependent on part of a composite key.
- **Transitive Dependency:** A non-key attribute depends indirectly on the primary key.
- **Trivial Dependency:** An attribute is dependent on itself or a subset of itself.


Types of Functional Dependency
Transitive Dependency
Trivial Functional Dependency
Non-trivial Functional Dependency

Example of Full Functional Dependency (Explained)

Scenario: An "Employee" table with attributes

Employee_ID, Department_ID, Employee_Name, Department_Name.

Here:

Employee_ID -> Employee_Name

Employee name depends solely on the employee ID.

Employee_ID, Department_ID -> Department_Name

Department name is fully dependent on the composite key (Employee_ID and Department_ID).

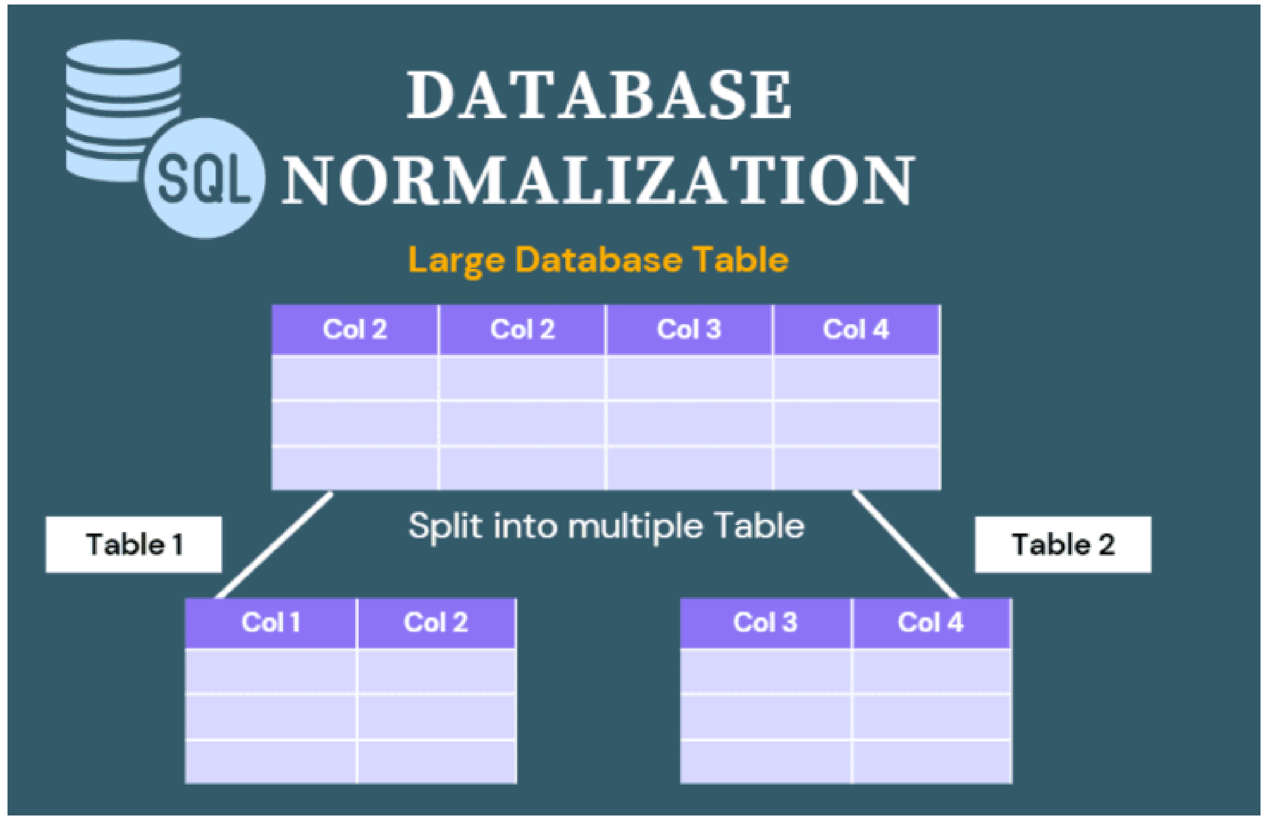
Identifying FFD in Database Normalization

FFD is essential in achieving Second Normal Form (2NF):

1NF: Ensures atomicity.

2NF: Removes partial dependencies.

Example: If a table has partial dependencies, separating them into new tables can achieve 2NF and ensure full functional dependency.



How to Identify Full Functional Dependency

- **Check the Dependency:** Verify if each non-key attribute is dependent on the primary key.
- **Confirm Composite Keys:** For composite keys, ensure no subset of the key determines the non-key attribute.
- **Normalize if Needed:** Remove any partial dependencies by moving attributes to new tables.

- **Analyze Functional Dependencies:** List all functional dependencies within the table to see how attributes relate to each other and identify which are fully dependent on the primary key.
- **Apply Test Cases:** Try inserting, updating, or deleting records to observe if any anomalies occur. If anomalies arise, it may indicate the need for further normalization.

Benefits of Full Functional Dependency

- **Data Integrity:** Ensures each attribute in a table reflects only relevant information.
- **Reduction in Anomalies:** Prevents update, insert, and delete anomalies.
- **Optimized Storage:** Reduces redundancy, which helps save storage space by avoiding duplicate data entries.
- **Improved Query Performance:** Minimizes the need for complex joins or subqueries by ensuring tables are more efficiently structured.
- **Enhanced Data Consistency:** Ensures that data modifications are uniformly applied across the database, reducing discrepancies and inconsistencies.

Limitations of Full Functional Dependency

- **Complexity in Large Databases:** Maintaining FFD for complex databases can be challenging.
- **Maintenance Overhead:** Additional tables created in normalization can increase overhead.
- **Higher Initial Setup Time:** Defining and implementing FFDs requires careful planning and can take significant time in the initial stages of database design.
- **Potential for Over-normalization:** Excessive focus on FFD may lead to over-normalization, creating too many small tables that complicate data retrieval.
- **Reduced Flexibility:** Strict adherence to FFD may limit the ability to store unstructured or semi-structured data, making it harder to adapt to evolving data requirements.

Conclusion

Full Functional Dependency is fundamental to effective database design, ensuring that each attribute within a table is fully dependent on its primary key, promoting data integrity and reducing redundancy.

In practical applications, FFD helps to streamline data management, providing a structured and reliable approach to handling complex datasets. Ultimately, understanding and leveraging FFD is essential for building efficient, scalable, and reliable database systems.

REFERENCES

https://www.geeksforgeeks.org/differentiate-between-partial-dependency-and-fully-functional-dependency/
https://www.javatpoint.com/fully-functional-dependency-in-dbms
https://byjus.com/gate/partial-dependency-in-dbms-notes/
https://www.youtube.com/watch?v=2JQ0iLcFNDY
https://www.bauer.uh.edu/hpanahi/db/MIS-3376-7373-Chapter7-8-9.pdf