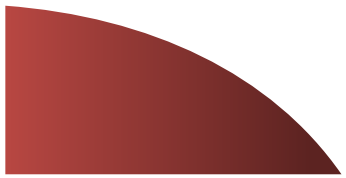
Relational Database Design

Outline

- Definition of FDs
- Armstrong's axioms OR Inference rules
- Types of FDs
- Closure of attribute sets



What is Functional Dependency (FD)?

- FD describes the relationship between the attributes in a relation.
- Let us assume A and B are the set of attribute(s) of the relation R.
- Attribute(s) A functionally (or uniquely) determines attribute(s) B (i.e., A → B) with respect to R if and only if any two tuples t1 and t2 of R, whenever

t1[A] = t2[A], then
t1[B] = t2[B]

- If the **values of the A component of a tuple uniquely** (or functionally) **determine the values of the B component**, then there is a **functional dependency from A to B**. This is also denoted by **A → B**.
- It is referred as: **B is functionally dependent on the A** or **A functionally determines B**.
- It deals with one-to-one relationship among attribute.

Uses of Functional Dependency (FD)?

- To identify the additional FDs.
- To identify the keys
- To identify the equivalence of FD
- To identify the standard form, canonical form or irreducible set of FD.

What is Functional Dependency (FD)?

Let attributes X and Y are two subsets of attributes of relation R.

Example: RollNo → Name, SPI, BL

Student			
RollNo	**Name**	**SPI**	**BL**
101	Raju	8	0
102	Mitesh	7	1
103	Jay	7	0

Exercise:

- Identify the valid and invalid dependency for the following relation R:

A	B	C
1	4	2
3	5	6
3	4	6
7	3	8
9	1	0

1.) A → B **2.) A → C** **3.) AB → C** **4.) C → A**

5.) BC → A **6.) AC → B** **7.) C → B**

How many valid dependencies are their for the following relation R:

A	B	C
f	e	e
d	e	e
b	c	e
a	c	b
a	b	c

1.) AB → C
2.) AC → B

Non-Trivial

3.) AB → A
4.) AB → B
5.) AC → C
6.) AC → A
....
....

Trivial

1.) ABC → C
2.) ABC → B
3.) ABC → A
4.) ABC → BC
5.) ABC → AB
6.) ABC → AC
....
....

Trivial

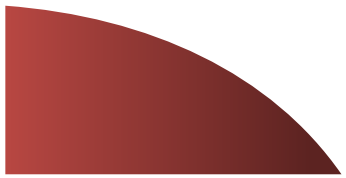
Note: *Keys of the table will always functionally determine every attribute of the table. Because keys are the set of attribute whose value must be unique or different in each tuple of the Relation.*

Properties and Types of FDs

FD must be defined on schema not on instance.

Types of FDs:

- Trivial FD
- Non-Trivial FD
- Completely Non-Trivial
- Transitive FD
- Full Functional Dependency
- Partial FD



Types of Functional Dependency (FD)

Trivial FD: RHS is subset of LHS

X ⟶ Y is trivial FD if **Y is a subset of X**

Eg. {Roll_No, Dept_Name, Semester} ⟶ Roll_No, Semester OR ABC →AB

Nontrivial FD: One of RHS attribute is not subset LHS.

X ⟶ Y is nontrivial FD if **Y is not a subset of X**

Eg. {Roll_No, Dept_Name, Semester} ⟶ Stud_Name, Semester OR ABC →AD

Completely Non-Trivial FD: None of RHS attribute is subset of LHS.

X ⟶ Y is Completely nontrivial FD if **all subset of Y is not a subset of X**

Eg. {Roll_No, Dept_Name, Semester} ⟶ Stud_Name, SPI OR ABC →DE

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**.

Example: {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 dependency.

Example: {Enrollment_No, Department_Name} → SPI

Enrollment_No is sufficient to find SPI, Department_Name is not required to find SPI.

Transitive Functional Dependency

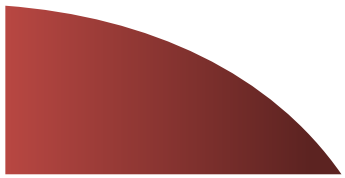
In a relation, if attribute(s) **A → B and B → C, then A → C (means C is transitively depends on A** via B).

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

Example: 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.

Armstrong's axioms OR Inference rules



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.

Self-determination

- If A → A

Reflexivity

- If B is a subset of A
  - then A → B

Transitivity

- If A → B and B → C
  - then A → C

Augmentation

- If A → B
  - then AC → BC

Pseudo Transitivity

- If A → B and BD → C
  - then AD → C

Decomposition

- If A → BC
  - then A → B and A → C

Union

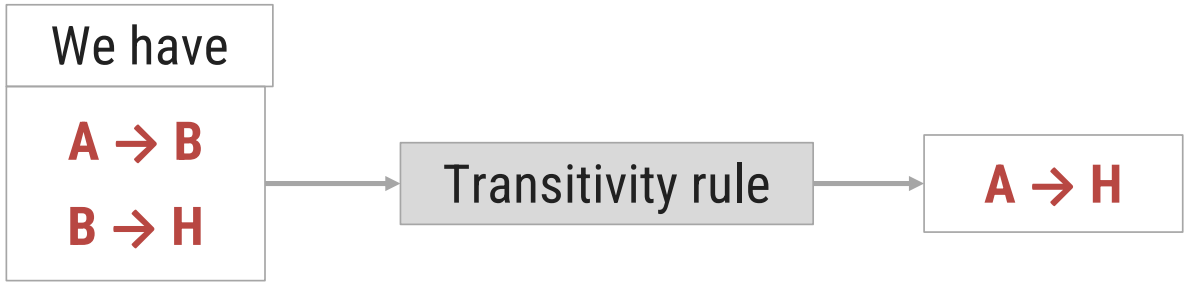
- If A → B and A → C
  - then A → BC

Composition

- If A → B and C → D
  - then AC → BD

[Example]

- 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.



[Example]

- 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.

[Example]

- 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.