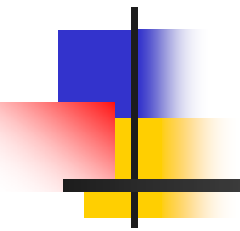
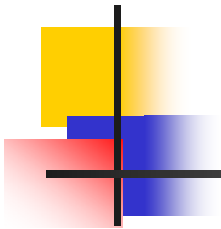
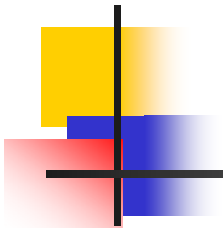


Normalization



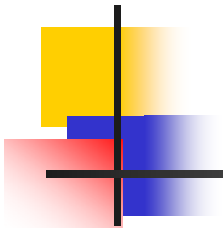
Objectives

- Introduction +
- Objectives of Normalization +
- Normal Forms +
- The Process of Normalization +
- Un Normalized Form (UNF) +
- Converting UNF to Normalized Form +
- Summary +



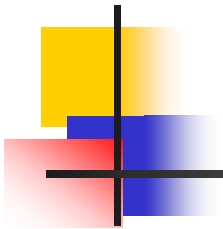
- Introduction

- A process to validate and improve logical design so that it satisfies certain constraints that avoid unnecessary duplication of the data.
- Normalization is a formal technique for analyzing a relation based on its primary key and the FDs between the attributes of that relation.
- The normalization process was first proposed by Codd 1972.
- The normalization process takes a relation schema through a series of tests to certify whether it satisfies a certain normal form.



- Objectives of Normalization

- Solve problems associated with redundant data.
- Identify various types of update anomalies such as insertion, deletion, and modification anomalies.
- Recognize the appropriateness or quality of the design of relations.
- Use FDs to group attributes into relations that are in a known normal form.

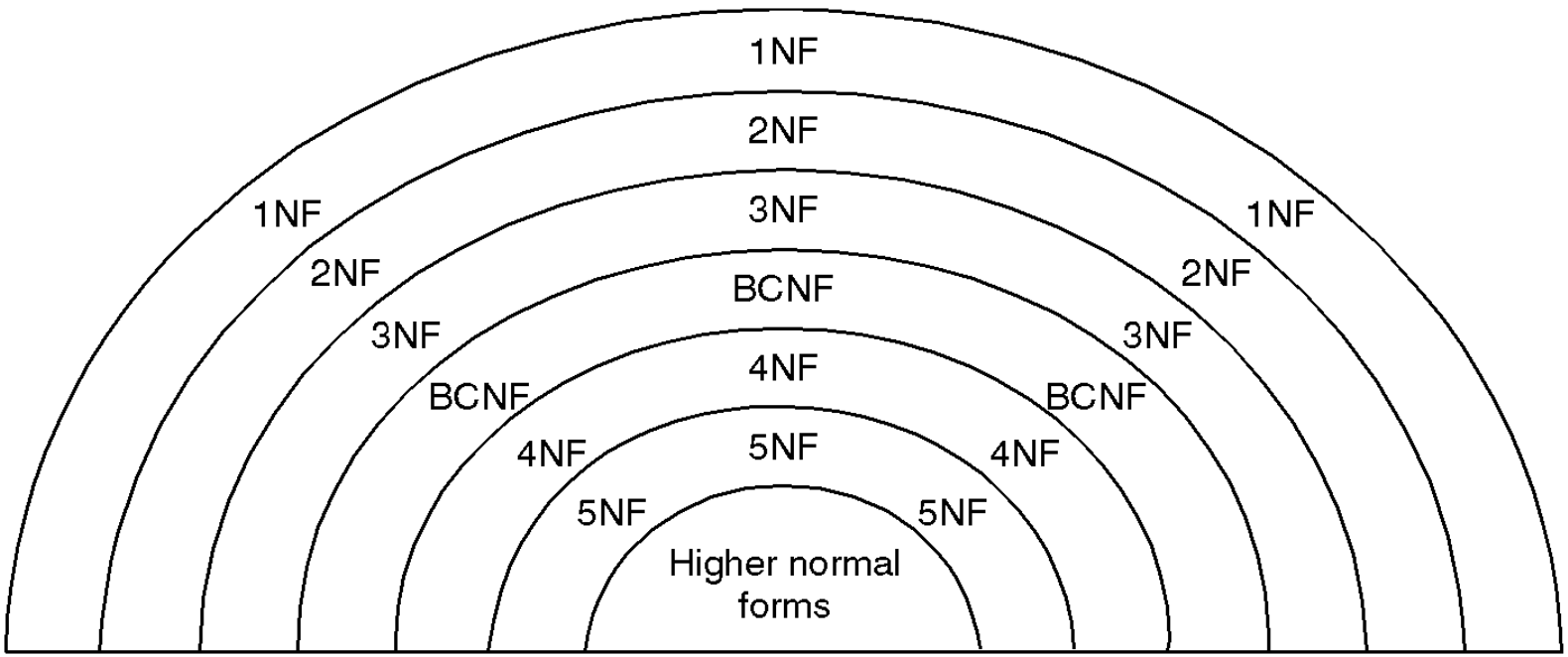


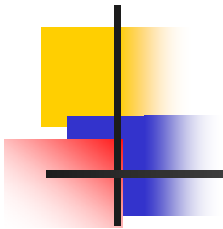
- Normal Forms

- The four most commonly used normal forms are:
  - First Normal Form (1NF)
  - Second Normal Form (2NF)
  - Third Normal Form (3NF)
  - Boyce-Codd Normal Form (BCNF)
- To convert un-normalized table to a normalized one, you first convert it to INF, then to 2NF, then to 3NF and then to BCNF. In other words:

  UNF → 1NF → 2NF → 3NF → BCNF

-- Relationship Between Normal Forms





- The Process of Normalization

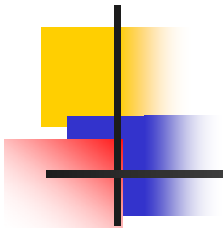
- Formal technique for analyzing a relation based on its primary key and the functional dependencies between the attributes of that relation.
- Often executed as a series of steps. Each step corresponds to a specific normal form, which has known properties.
- As normalization proceeds, the relations become progressively more restricted (stronger) in format and also less vulnerable to update anomalies.
- As normalization proceeds, the relations become progressively more restricted (stronger) in format and also less vulnerable to update anomalies.

- Unnormalized Form (UNF)

- A table would be in unnormalized form if it contains some multi-valued attributes or repeating groups.
- To create an unnormalized table Transform the data from the information source (e.g. form) into table format with columns and rows.
- Example:

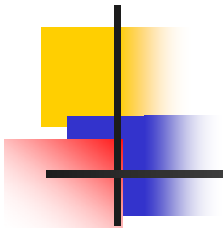
Dname	Dnumber	DMGRSSN	Dlocations
Research	5	222	Dammam, Khobar, Dhahran
Administration	4	333	Jubail
Headquarters	1	444	Khobar

DEPARTMENT



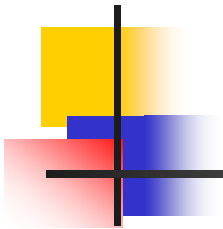
- Converting From UNF to Normalized Form

- 1NF +
- UNF to 1NF +
- Example: UNF to 1NF +
- 2NF +
- 1NF to 2NF +
- Example: 1NF to 2NF +
- 3NF +
- 2NF to 3NF +
- Example: 2NF to 3NF +
- BCNF +
- 3NF to BCNF +
- Example: 3NF to BCNF +



-- First Normal Form (1NF)

- A relation in which the intersection of each row and column contains one and only one value.



-- UNF to 1NF

- Nominate an attribute or group of attributes to act as the key for the unnormalized table.
- Identify the repeating group(s) in the unnormalized table which repeats for the key attribute(s).
- Remove the repeating group by
  - Entering appropriate data into the empty columns of rows containing the repeating data (‘flattening’ the table).
  - Or by
  - Placing the repeating data along with a copy of the original key attribute(s) into a separate relation.

-- Example: UNF to 1NF

UNF

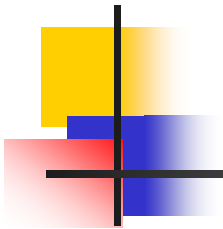
Dname	Dnumber	DMGRSSN	Dlocations
Research	5	222	Dammam, Khobar, Dhahran
Administration	4	333	Jubail

1NF

Dname	Dnumber	DMGRSSN	Dlocations
Research	5	222	Dammam
Research	5	222	Khobar
Research	5	222	Dhahran
Administration	4	333	Jubail

-- Second Normal Form (2NF)

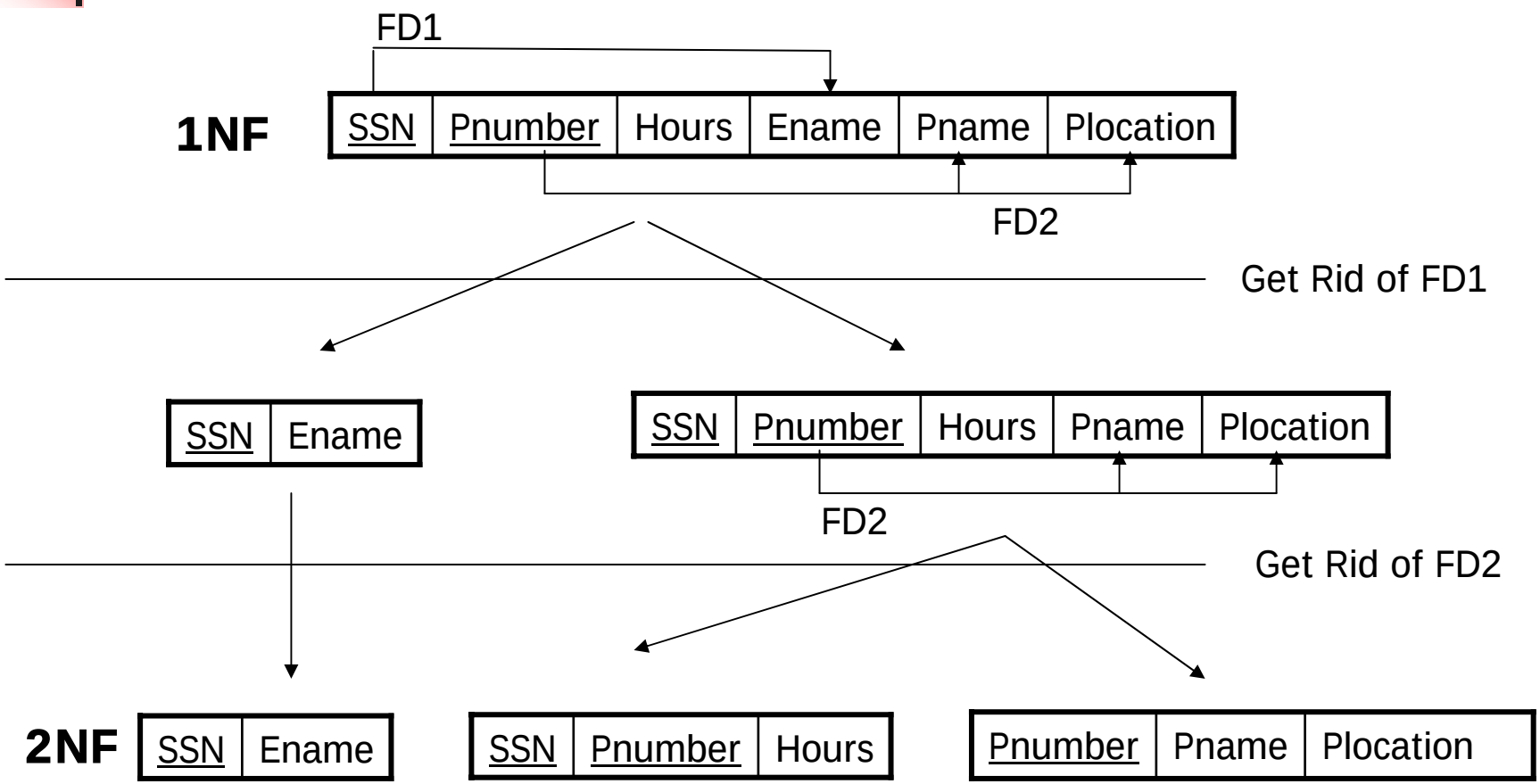
- Based on the concept of full functional dependency.
- Full functional dependency indicates that if
  - *A* and *B* are attributes of a relation,
  - *B* is fully dependent on *A* if *B* is functionally dependent on *A* but not on any proper subset of *A*.
- Example: In the last table *{SSN, Pnumber} → Hours* is a full dependency. (neither *SSN → Hours* nor *Pnumber → hours* hold.
- A relation that is in 1NF and every non-primary-key attribute is fully functionally dependent on the primary key.



-- 1NF to 2NF

- If a primary key is a single attribute then the relation is already in 2NF.
- In the case of composite key, the following steps are applied to convert to 2NF:
  - Identify the primary key for the 1NF
  - Identify the functional dependencies in the relation
  - If partial dependency exists on the primary key remove them by placing them in a new relation along with a copy of their determinant.

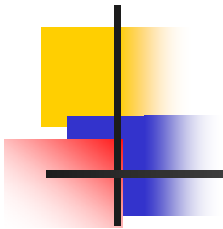
--- Example: 1NF to 2NF



-- Third Normal Form (3NF)

- 3NF is based on the concept of transitive dependency.

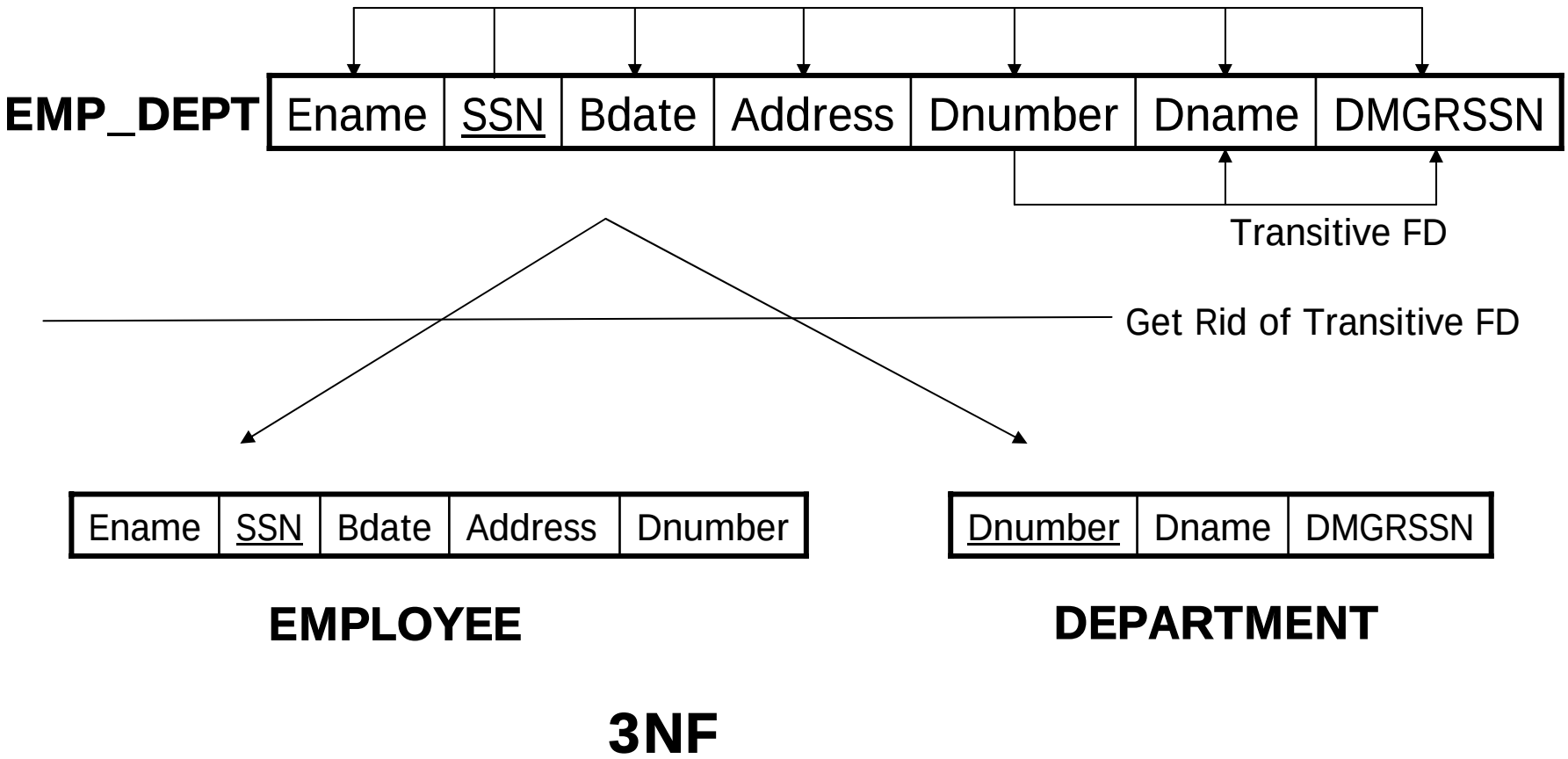
- Transitive Dependency is a condition where
  - *A*, *B* and *C* are attributes of a relation such that if *A* → *B* and *B* → *C*,
  - then *C* is transitively dependent on *A* through *B*. (Provided that *A* is not functionally dependent on *B* or *C*).

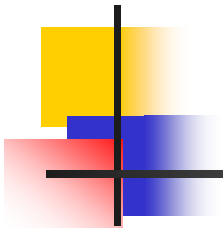


-- 2NF to 3NF

- Identify the primary key in the 2NF relation.
- Identify functional dependencies in the relation.
- If transitive dependencies exist on the primary key remove them by placing them in a new relation along with a copy of their dominant.

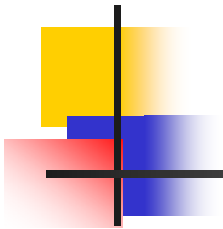
-- Example: 2NF to 3NF





-- Boyce-Codd Normal Form (BCNF)

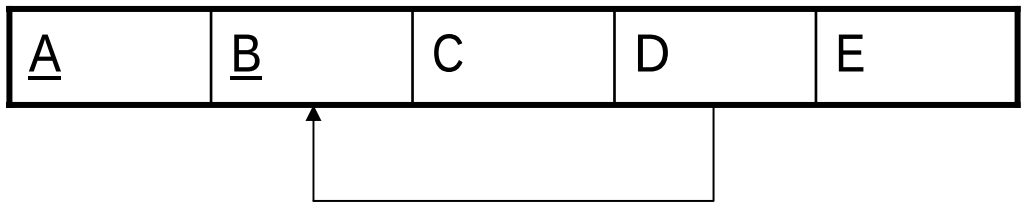
- Based on functional dependencies that takes into account all candidate keys in a relation.
- For a relation with only one candidate key, 3NF and BCNF are equivalent.
- A relation is in BCNF, if and only if every determinant is a candidate key.
- Violation of BCNF may occur in a relation that
  - contains two (or more) composite keys
  - which overlap and share at least one attribute in common.



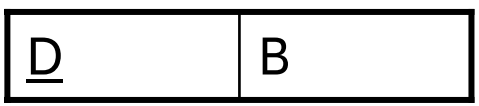
-- 3NF to BCNF

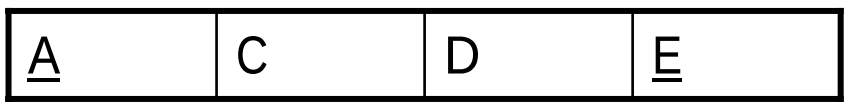
- Identify all candidate keys in the relation.
- Identify all functional dependencies in the relation.
- If functional dependencies exists in the relation where their determinants are not candidate keys for the relation, remove the functional dependencies by placing them in a new relation along with a copy of their determinant.

-- Example: 3NF to BCNF

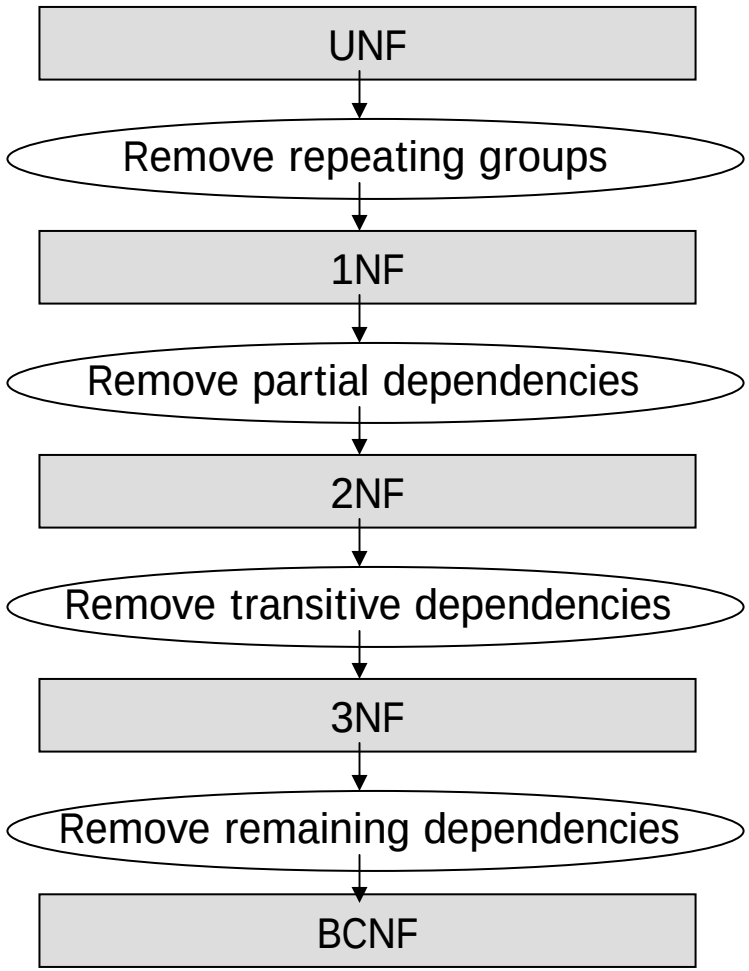


BCNF Normalization



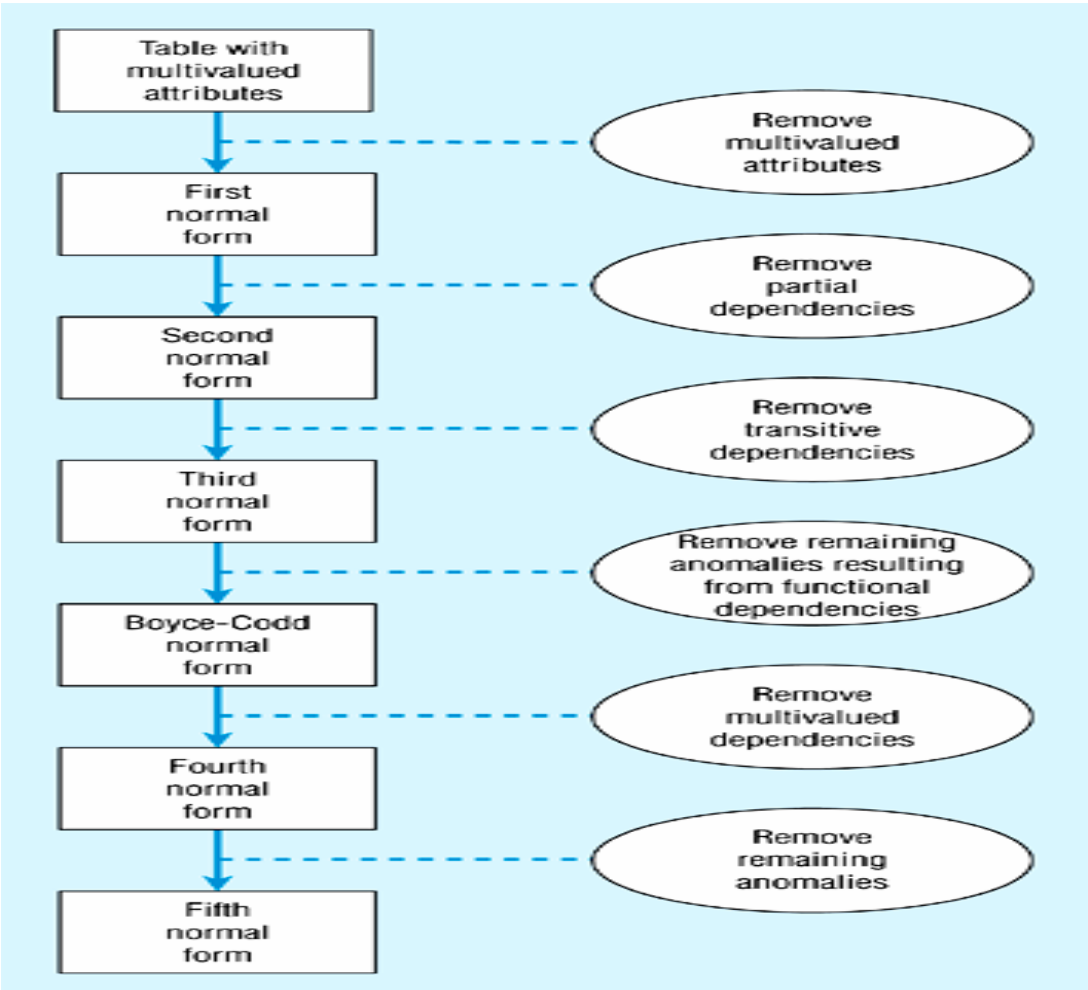


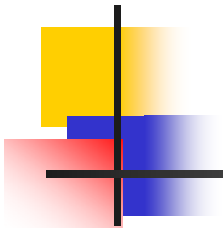
- Summary


UNF
Remove repeating groups
1NF
Remove partial dependencies
2NF
Remove transitive dependencies
3NF
Remove remaining dependencies
BCNF

- Summary ...

Steps in Normalization





... - Summary

- Definition
  - A functional dependency X → A is called **trivial** if, $A \subseteq X$.
  - A relation is in **1NF** if, and only if, every attribute is single-valued for each tuple.
  - A relation is in **2NF** if, and only if, it is in 1NF and all the non-key attributes are fully functionally dependent on the key.
  - A relation is in **3NF** if, when ever a non-trivial functional dependency X→ A exists, then either X is a superkey or A is a member of some candidate key.
  - A relation is in **BCNF** if, whenever a non-trivial functional dependency X → A exists, then X is a superkey.