DBMS

Lecture 3

Normalization

Anomalies

Anomalies

- Insertion Anomaly
- Deletion Anomaly
- Updation Anomaly

Normalization

Normalization

- Normalization is a set of rules/guidelines that is used while designing a database. Or it is imposing some systematic rules on your table so that our table doesn't have these design anomalies.
- These rules help to remove all the anomalies and distribute the data among different related tables and query them efficiently and effectively.
- It removes all the duplication issues and incorrect data issues, helping to have a well designed database.

Normalization

- Different Normalization forms -
  - 1 NF
  - 2 NF
  - 3 NF
  - BCNF
- But before learning them, we need to learn a concept called “Functional Dependency”.

Functional Dependency

Functional Dependency

- It is a relationship that exists between multiple attributes of a relation.
- The attributes of a table is said to be dependent on each other when an attribute of a table uniquely identifies another attribute of the same table.
- For example: Suppose we have a student table with attributes: Stu_Id, Stu_Name, Stu_Age.

Functional Dependency

- If column A of a table uniquely identifies the column B of same table then it can represented as

$$A \rightarrow B$$

(Attribute B is functionally dependent on attribute A)

A : It is a determinant set.

B : It is a dependent attribute.

$\{A \rightarrow B\}$: A functionally determines B.
B is a functionally dependent on A.

Functional Dependency

- Trivial Functional dependency
  - The dependency of an attribute on a set of attributes is known as trivial functional dependency if the set of attributes includes that attribute.
  - $A \rightarrow B$ is trivial functional dependency if B is a subset of A.
- Non-trivial Functional dependency
  - If a functional dependency $X \rightarrow Y$ holds true where Y is not a subset of X then this dependency is called non trivial Functional dependency.

Armstrong’s Axioms

Axioms

- Axioms are nothing but some blind rules. So every FD satisfies these blind rules.
- They are called universal rules -
  - Reflexive property
    - If X is superset of Y, then $X \rightarrow Y$
    - Example : X = {A, B, C} and Y = {B, C}
  - Augmentation property
    - If $W \rightarrow Z$ and $X \rightarrow Y$, then $WX \rightarrow YZ$
  - Transitive property
    - If $X \rightarrow Y$ and $Y \rightarrow Z$, then $X \rightarrow Z$

Axioms

- 2 more rules which are derived from universal rules -
  - Unon Rule -
    - If $X \rightarrow Y$ and $X \rightarrow Z$, then $X \rightarrow YZ$
  - Decomposition Rule -
    - If $X \rightarrow YZ$, then $X \rightarrow Y$ and $X \rightarrow Z$

- We’ll be using these axioms while doing normalization.

Closure

Functional Dependency Closure

- Functional Dependency Set i.e. F -
  - The rules or set of all FDs of a table are called Functional dependency set (i.e. F).
- Closure of Functional Dependency Set i.e. F+,
  - All the rules implied by the given rules in F (using axioms).
- For example -
  - Given rules in F -
    - $A \rightarrow B$, $C \rightarrow D$ and $B \rightarrow C$
  - F+ (Other than above 3)
    - $A \rightarrow C$, $A \rightarrow A$, $B \rightarrow B$, $C \rightarrow C$ and others (Using reflexive)

Attribute Closure

- All the attributes implied by given set of attributes.
- Example 1 -
  - Given the follow dependencies -
    - $A \rightarrow B$, $C \rightarrow D$, $B \rightarrow C$
  - AB+ : all the attributes which are determined by AB together or A or B
    - A, B (Reflexivity)
    - C - transitive
    - D - transitive

Super & Candidate Key

- Super Key
  - It is a set of attributes which uniquely identifies all the attributes of a table.
- Candidate Key
  - It is set of attributes which uniquely identifies all the attributes of a table. Candidate key is also Super key.
  - Difference is that - candidate key is minimal, that means - we cannot remove any attribute from candidate key otherwise it cannot determine all other attributes of table.

Normalization

1 NF

- Rule for 1 NF -
  - In a table, every attribute should be atomic i.e. there cannot be multi-valued or composite attributes.
  - Each record needs to be unique.
- In case of composite attributes, break it into multiple columns.
- In case of multi-values attributes, create a row for each value making the attribute as single value. But that will create lot of duplicacy.

2 NF

2 NF

- Rule for 2 NF -
  - Table must be in 1NF
  - There should be no partial dependency.

Partial Dependency

- **$X \rightarrow a$ is a partial dependency, if**
  - X is a proper subset of some candidate key, and
  - a is non-prime or non-key attribute
- **Proper subset -**
  - all the subsets which are not exactly same as given subset. For example - if given subset is {A, B, C} then every other subset other than {A, B, C} itself is proper subset.
- **Prime Attributes**
  - Those attributes which are part of at least one candidate key.
- **Non-prime attributes**
  - Those attributes which are not part of any candidate key.

Super & Candidate Key

- Super Key
  - It is a set of attributes which uniquely identifies all the attributes of a table.
- Candidate Key
  - It is set of attributes which uniquely identifies all the attributes of a table. Candidate key is also Super key.
  - Difference is that - candidate key is minimal, that means - we cannot remove any attribute from candidate key otherwise it cannot determine all other attributes of table.

2 NF Examples

- Example 1 : R(A, B, C, D)
  Given FDs : $A \rightarrow B$, $C \rightarrow D$
- Example 2 : R(A, B, C, D)
  Given FDs : $AB \rightarrow C$, $BC \rightarrow D$, $A \rightarrow D$
- Example 3 : R(A, B, C, D)
  Given FDs : $A \rightarrow B$, $A \rightarrow C$, $B \rightarrow A$

3 NF

3 NF

- Rules for 3 NF -
    - Table must be in 2NF
    - There should be no transitive dependency.

Transitive Dependency

- When some non-prime attribute depends on some other non-prime attribute.
- That means, there should not be any dependency where

  $$X \rightarrow a$$

  X is a non-prime attribute and a is also non-prime.
- Or in other words, transitive dependency is when some non-prime attribute transitively depends on candidate key.

3NF Examples

- Example 1 : R(A, B, C)
  Given FDs : $A \rightarrow B$, $B \rightarrow C$

- Example 2 : R(A, B, C, D)
  Given FDs : $AB \rightarrow C$, $B \rightarrow C$, $C \rightarrow B$

Problem with Transitive Dependency

BCNF

Previous Normal Forms

- 2NF
  - When there is no partial dependency.
  - Partial dependency - when any non-prime attribute depends on a part of candidate key.
- 3NF
  - No transitive dependency
  - Transitive dependency - When some non-prime attribute depends on other non-prime attribute

Previous Normal Forms

- Till now we didn't consider this kind of dependency where Y is prime

  $X \rightarrow Y$

  We considered the case where Y is always non-prime and X can be prime or non-prime.

- When Y is prime, X can be prime or non-prime. These kind of cases are dealt by BCNF

BCNF

- If there exists any functional dependency such that -

  $$X \rightarrow Y$$

  X should always be super-key.

Example