DATABASE SYSTEMS

CS – 355/ CE – 373

Instructor: Maria N. Samad

September 11th, 2024

NORMALIZATION

- A process of analyzing the given relation schemas based on their functional dependencies and primary keys to achieve the desirable properties of:
  - Minimizing redundancy
  - Minimizing insertion, deletion and modification anomalies
  - Non-additive joining of decomposed tables guarantees no spurious tuple generation
  - Dependency Preservation Property – after the decomposition, the functional dependencies must be retained in one form or the other, otherwise the original attributes/keys cannot be restored when joining the decomposed relations

NORMAL FORM

- ***Normal Form*** of a relation refers to the highest normal form condition that it meets and hence indicates the degree to which data has been normalized
- The different types of Normal Forms are:
  - First Normal Form (1NF)
  - Second Normal Form (2NF) (includes 1NF)
  - Third Normal Form (3NF) (includes 2NF + 1NF)
  - Boyce Codd Normal Form (BCNF)
  - Fourth Normal Form
  - Fifth Normal Form
  - Sixth Normal Form

PRIME vs NON-PRIME ATTRIBUTES

- ***Prime Attribute:***
  - An attribute of a relation schema, R that is a member of some candidate key of R, is called a ***prime attribute***
  - For example, given relation is:

    R (A, B, C, D, E, F)
  - And the candidate keys are: **{A, C}**, **{B, D}**
  - Therefore, prime attributes will be: **A, B, C, D**

PRIME vs NON-PRIME ATTRIBUTES

- ***Non-prime Attribute:***
  - An attribute that is not prime
  - Or an attribute of a relation schema, R that is not a member of some candidate key of R, is called a ***non-prime attribute***
  - For example, given relation is:

    R (A, B, C, D, E, F)
  - And the candidate keys are: **{A, C}**, **{B, D}**
  - Therefore, prime attributes will be: **A, B, C, D**
  - And non-prime attributes will be: **E, F**

FIRST NORMAL FORM (1NF)

- First normal form (1NF) states that the domain of an attribute must:
  - Include only atomic (simple, indivisible) values, and
  - The value of any attribute in a tuple must be a single value from the domain of that attribute
- Hence, 1NF disallows having a set of values, a tuple of values, or a combination of both as an attribute value for a single tuple
- In other words, 1NF disallows relations within relations or relations as attribute values within tuples
- The only attribute values permitted by 1NF are single atomic (or indivisible) values

FIRST NORMAL FORM (1NF)

- The basic rules are as follows:
  - For ***composite attributes***, each component becomes a separate attribute in the resultant 1NF
  - For ***multivalued attributes***, each item becomes a separate tuple, however, when defining the 1NF schema, the primary keys must be updated

EXAMPLE

- Consider the following relation (not in 1NF):

DEPARTMENT

Dname	Dnumber	Dmgr_ssn	Dlocations
Research	5	333445555	{Bellaire, Sugarland, Houston}
Administration	4	987654321	{Stafford}
Headquarters	1	888665555	{Houston}

- The following relation is normalized to 1NF:

DEPARTMENT

Dname	Dnumber	Dmgr_ssn	Dlocation
Research	5	333445555	Bellaire
Research	5	333445555	Sugarland
Research	5	333445555	Houston
Administration	4	987654321	Stafford
Headquarters	1	888665555	Houston

FIRST-NORMAL FORM

- **Exercises – Composite attributes are mentioned in round brackets**:
  - *Cars (CarID, (License), Year, Cost)*
    - **1NF – *Cars (CarID, Letter, Digits, Year, Cost)***
  - *Cars (CarID, (License), (CarType), Year, Cost)*
    - **1NF – *Cars (CarID, Letter, Digits, Brand, Model, Year, Cost)***
  - *Book (BookNum, (BookDetails), YearPublished, Cost)*
    - **1NF – *Book (BookNum, Title, Author, ISBN, YearPublished, Cost)***
  - *Person(SSN, (Address), {PhoneNum})*
    - **1NF – *Person (SSN, Street, Zipcode, City, PhoneNum)***

FUNCTIONAL DEPENDENCY

- An important concept in normalization
- This requires that the value for a certain set of attributes determines uniquely the value for another set of attributes
- A set of attributes ***X*** functionally determines a set of attributes *Y* if the value of *X* determines a unique value for *Y*
- $X \rightarrow Y$ holds if whenever two tuples have the same value for *X*, they must have the same value for *Y*

** Content from Dr. Faisal Alvi's slides **

FUNCTIONAL DEPENDENCY

- ***Theorem:***
  - A functional dependency denoted by $X \rightarrow Y$ between two sets of attributes, X and Y that are subsets of relation schema, R holds if and only if:
    - For any two tuples, t1 and t2, we have t1[X] = t2[X]
    - Then we must have t1[Y] = t2[Y]

FUNCTIONAL DEPENDENCY

- The values of Y component of the tuple in relation **depends on** or **is determined by** the corresponding values of X
- Or the values of X component of a tuple in a relation **uniquely or functionally determine** the values of Y component
- Abbreviation – ***FD*** or ***f.d.***
- X is called the **left hand side of FD** or the **determinant**
- Y is called the **right hand side of FD** or the **dependent**

FUNCTIONAL DEPENDENCY (EXAMPLE)

- Identify functional dependencies in the following relation:

A	B	C	D
a1	b1	c1	d1
a1	b2	c2	d2
a2	b2	c2	d3
a3	b3	c4	d3

** Content from Dr. Faisal Alvi's slides **

FUNCTIONAL DEPENDENCY (EXAMPLE)

- Some resultant functional dependencies are:
  - $B \rightarrow C$
  - $C \rightarrow B$
  - $\{A, B\} \rightarrow C$
  - $\{A, B\} \rightarrow D$
  - $\{C, D\} \rightarrow B$

A	B	C	D
a1	b1	c1	d1
a1	b2	c2	d2
a2	b2	c2	d3
a3	b3	c4	d3

EXAMPLE

- Consider the following relation →
- The attributes *building* and *budget* are functionally dependent on *dept_name*.
- Are there any other functional dependencies?

ID	*name*	*salary*	*dept_name*	*building*	*budget*
22222	Einstein	95000	Physics	Watson	70000
12121	Wu	90000	Finance	Painter	120000
32343	El Said	60000	History	Painter	50000
45565	Katz	75000	Comp. Sci.	Taylor	100000
98345	Kim	80000	Elec. Eng.	Taylor	85000
76766	Crick	72000	Biology	Watson	90000
10101	Srinivasan	65000	Comp. Sci.	Taylor	100000
58583	Califieri	62000	History	Painter	50000
83821	Brandt	92000	Comp. Sci.	Taylor	100000
15151	Mozart	40000	Music	Packard	80000
33456	Gold	87000	Physics	Watson	70000
76543	Singh	80000	Finance	Painter	120000

Figure 7.2 The *in_dep* relation.

** Content from Dr. Faisal Alvi's slides **

FULL vs PARTIAL FUNCTIONAL DEPENDENCY

- ***Full Functional Dependency*:**
  - An FD: X $\rightarrow$ Y is a ***full functional dependency*** if removal of any attribute A from X means that the dependency does not hold anymore
  - That is, all attributes of attribute set, X must be used together to define functional dependency
- ***Partial Functional Dependency*:**
  - An FD: X $\rightarrow$ Y is a ***partial functional dependency*** if after removing some attributes from the attribute set X, the functional dependency still holds

FULL vs PARTIAL FUNCTIONAL DEPENDENCY

- For example, consider the following schema:
  Emp_Proj (<u>SSN</u>, <u>ProjectID</u>, EmpName, WorkedHours, ProjectName, ProjectLocation)
- Check if the following is fully or partially dependent?
  - {SSN, ProjectID} $\rightarrow$ WorkedHours?
    - For combined attributes, check if any subset will fulfill the dependency requirement
    - SSN $\rightarrow$ WorkedHours? Not possible, as one employee can work on multiple projects so cannot tell hours he worked on a specific project just by SSN
    - ProjectID $\rightarrow$ WorkedHours? Not possible, because many people work on a project, so if these are the number of hours an employee has worked on a project, then just the Project ID would be insufficient to deduce hours
    - Hence **FULL FD**

FULL vs PARTIAL FUNCTIONAL DEPENDENCY

- Check if the following is fully or partially dependent?
  - {SSN, ProjectID} $\rightarrow$ EmpName?
    - For combined attributes, check if any subset will fulfill the dependency requirement
    - SSN $\rightarrow$ EmpName? Possible, as you can find an employee's name through their SSN
    - Hence **PARTIAL FD**
  - {SSN, ProjectID} $\rightarrow$ ProjectName?
    - For combined attributes, check if any subset will fulfill the dependency requirement
    - SSN $\rightarrow$ ProjectName? Not possible, as an employee can work on several projects so just by their identity we cannot uniquely find the name of the project they worked on
    - ProjectID $\rightarrow$ ProjectName? Possible, because you can find the name of the project through its ID
    - Hence **PARTIAL FD**

FULL vs PARTIAL FUNCTIONAL DEPENDENCY

- Check if the following is fully or partially dependent?
  - {SSN, ProjectID} $\rightarrow$ ProjectLocation?
    - For combined attributes, check if any subset will fulfill the dependency requirement
    - SSN $\rightarrow$ ProjectLocation? Not possible, as an employee can work on several projects so just by their identity we cannot uniquely find the location of the project they are working on
    - ProjectID $\rightarrow$ ProjectLocation? Possible, because you can find the location of the project through its ID
    - Hence **PARTIAL FD**

FUNCTIONAL DEPENDENCY

- Activity Sheet

FUNCTIONAL DEPENDENCY

- Activity Sheet Solution:
  - Functional Dependencies Solution