Basic Relational Design

"The key, the whole key, and nothing but the key."

Georgia Tech

Basic Relational Design

- In a future series of lectures we'll learn relational design in detail
- For now we'll learn a semi-formal approach to normalizing a database schema to Third Normal Form (3NF)

Georgia Tech

Basic Design Process

- Start with relation schemas derived from EER model
- Enumerate functional dependencies for each relation schema
- Decompose non-3NF relation schemas into 3NF schemas

Georgia Tech

Functional Dependencies

A generalization of superkeys.
Given a relation schema R, and subsets of attributes X and Y, the functional dependency

$$X \rightarrow Y$$

Means that for any pair of tuples t_1 and t_2 in $r(R)$

$$\text{if } t_1[X] = t_2[X]$$
$$\text{then } t_1[Y] = t_2[Y]$$

In other words, whenever the attributes on the left side of a functional dependency are the same for two tuples in the relation, the attributes on the right side of the functional dependency will also be equal.

Georgia Tech

Relations Satisfy FDs

A	B	C	D
a_1	b_1	c_1	d_1
a_1	b_2	c_1	d_2
a_2	b_2	c_2	d_2
a_2	b_2	c_2	d_3
a_3	b_3	c_2	d_4

$A \rightarrow C$ is satisfied because no two tuples with the same A value have different C values.
$C \rightarrow A$ is not satisfied because we find a $t_1 = (a_2, b_3, c_2, d_3)$ and $t_2 = (a_3, b_3, c_2, d_4)$

Georgia Tech

Satisfying vs. Holding

We say that a functional dependency f holds on a relation if it is not legal to create a tuple that does not satisfy f. Alternately, we say that a relation schema (not just a particular state) satisfies a functional dependency.

name	street	city
Alice	Elm	Charlotte
Bob	Peachtree	Atlanta
Charlie	Elm	Charlotte

Here $street \rightarrow city$ is satisifed by this relation state. However, we would not say that the functional dependency holds, or that the relation schema satisfies the functional dependency because we know there can be different cities with the same street names.

Georgia Tech

Trivial Functional Dependencies

A functional dependency is trivial if it is satisfied by all relations.
Formally, a functional dependency $X \rightarrow Y$ is trivial if $Y \subseteq X$
For example:

- $A \rightarrow A$
- $AB \rightarrow A$
- $AB \rightarrow B$

are trivial.
We don't write trivial functional dependencies when we enumerate a set of functional dependencies that hold on a schema for the purposes of normalization or normal form testing.

Georgia Tech

Normal Forms

A normal form is a set of conditions based on functional dependencies that acts as tests for the "goodness" of the design of a relation schema.
Normalization is the process of decomposing existing relation schemas into new relation schemas that satisfy normal forms for the purpose of:

- minimizing redundancy, and
- minimizing insertion, deletion, and update anomalies (we'll learn later)

We cover first, second, third, and Boyce-Codd normal forms in this class (only 3NF for today). Each higher normal form subsumes the normal forms below it, e.g., a 3NF schema is also in 2NF and 1NF. The normal form of a relation schema is the highest normal form it satisfies.

Georgia Tech

First Normal Form (1NF)

Every attribute value is atomic, which is effectively guaranteed by most RDBMS systems today.
The following relation is not in 1NF:

Dname	Dnumber	Dmgr_{ssn}	Dlocations
Research	5	333445555	{Bellaire, Sugarland, Houston}
Admin	4	987654321	{Stafford}
HQ	1	888665555	{Houston}

Because Dlocations values are not atomic.

Fixing Non 1NF Schemas

Many ways to fix (see book). Best way is to decompose into two schemas:

Dname	Dnumber	Dmgr_{ssn}
Research	5	333445555
Admin	4	987654321
HQ	1	888665555

Dnumber	Dlocation
5	Bellaire
5	Sugarland
5	Houston
4	Stafford
1	Houston

Georgia Tech

Second Normal Form (2NF)

A prime attribute is part of any candidate key. A nonprime attribute is not part of any candidate key.
A relation is in 2NF if it is in 1NF and no nonprime attribute has a partial dependency on the primary key, i.e., every attribute is fully dependent on the primary key.

Georgia Tech

2NF Test

Given
EMP_{PROJ}(Ssn _, Pnumber, Hours, Ename, Pname, Plocation)
and

- FD1: Ssn, Pnumber $\rightarrow$ Hours
- FD2: Ssn $\rightarrow$ Ename,
- FD3: Pnumber $\rightarrow$ Pname, Plocation

EMP_{PROJ} is not in 2NF due to FD2. Nonprime attribute Ename is partially dependent on the primary key Ssn, Pnumber.
EMP_{PROJ} is also not in 2NF due to FD3. Nonprime attributes Pname and Plocation are only partially dependent on the primary key.

Fixing non 2NF Schemas

Move the nonprime attributes that are dependent on part of the primary key to their own schemas with the part of the primary key on which they are fully dependent.
$\text{EMP}_{\text{PROJ}}(\underline{\text{Ssn}\_}, \underline{\text{Pnumber}}, \text{Hours}, \text{Ename}, \text{Pname}, \text{Plocation})$
Becomes
$\text{EMP}(\underline{\text{Ssn}\_}, \text{Ename})$
$\text{EMP}_{\text{PROJ}}(\underline{\text{Ssn}\_}, \underline{\text{Pnumber}}, \text{Hours})$
$\text{PROJ}(\underline{\text{Pnumber}\_}, \text{Pname}, \text{Plocation})$

Georgia Tech

Third Normal Form (3NF)

A schema is in 3NF if it is in 2NF and no nonprime attribute is transitively dependent on the primary key.
Given
EMP_{DEPT}(Ssn _, Ename, Bdate, Address, Dnumber, Dname, $Dmgr_{ssn}$)
and

- FD1: Ssn $\rightarrow$ Ename, Bdate, Address, Dnumber, Dname, $Dmgr_{ssn}$
- FD2: Dnumber $\rightarrow$ Dname, $Dmgr_{ssn}$

EMP_{DEPT} is not in 3NF because Dname and $Dmgr_{ssn}$ are transitively dependent on Ssn via dependency on Dnumber.

Georgia Tech

Fixing Non-3NF Schemas

Move the nonprime attributes that are transitively dependent on the primary key through another attribute to a separate schema along with the attribute through which they are transitively dependent on the PK.
EMP_{DEPT}($Ssn\_$, Ename, Bdate, Address, Dnumber, Dname, $Dmgr_{ssn}$)
becomes
EMP($Ssn\_$, Ename, Bdate, Address, Dnumber)
DEPT($Dnumber\_$, Dname, $Dmgr_{ssn}$)
Note that a natural join on Dnumber will recover the original relation.

Georgia Tech

Basic Relational Design Summary

- Every relation must have a key, and the 1NF assumption of the relational model asures that attributes are atomic. (Dont' "hide" extra information in strings!)
  - "The key,"
- A relation is in 2NF if it is in 1NF and no nonprime attribute has a partial dependency on the primary key, i.e., every attribute is fully dependent on the primary key.
  - "the whole key,"
- A schema is in 3NF if it is in 2NF and no nonprime attribute is transitively dependent on the primary key.
  - "and nothing but the key.

Normalize relations schemas by decomposing according to problematic functional dependencies.

Georgia Tech