Analysis and Normal Forms 2

Lecture 9

Monday - Oct 2, 2023

Housekeeping

- Review week 7 materials
- Review Homework 5
- Quiz 3 next Wednesday, October 11 covering normalization

Module	Week	Date	Day	Lectures/Quizzes	Deliverables/Notes
Normal forms	7	10/2	Mon	MTG11: L9 (Analysis and Normal Forms 2)	
Normal forms	7	10/4	Wed	MTG12: L10 (Analysis and Normal Forms 3)	
Normal forms	8	10/9	Mon	MTG13: L11 (Quiz review session)	
Normal forms	8	10/11	Wed	MTG14: Quiz 3 today (Analysis and Normal Forms)	
Normal forms	8	10/15	Sun		PrjDel 4 due (Phase 1 submission); HW5 due (Analysis and Normal Forms)
Intro to SQL	9	10/16	Mon	MTG15: L12 (Intro to SQL)	
Intro to SQL	9	10/18	Wed	MTG16: L13 (Intro to API / Python Flask and Click)	
Intro to SQL	9	10/20	Fri		No classes: Reading Day; Midterm Grades Due

Terms and definitions

Terms and definitions - Schema - Example 1

A *schema* is a collection of relations (tables). A *schema* is said to be in *Boyce-Codd Normal Form* when ALL the relations in the *schema* are in BCNF. BCNF minimizes redundancy and duplicate information.

RID	Course Code	First	Last	Language
1	CMSC508	John	Leonard	SQL
1	CMSC508	John	Leonard	Python
1	CMSC508	John	Leonard	Perl
2	CMSC508	Alberto	Cano	SQL
2	CMSC508	Alberto	Cano	Python
2	CMSC508	Alberto	Cano	C++
3	CMSC475	John	Leonard	Javascript
3	CMSC475	John	Leonard	Python
4	CMSC441	Bob	Dahlberg	COBOL
4	CMSC441	Bob	Dahlberg	FORTRAN
5	CMSC320	Sarah	Adams	C++
5	CMSC320	Sarah	Adams	Java
5	CMSC320	Sarah	Adams	Python
6	CMSC210	Michael	Turner	Java
6	CMSC210	Michael	Turner	C#
7	CMSC515	Emily	Parker	Python

Course Code	Course Name
CMSC210	Software Design
CMSC320	Data Structures
CMSC430	Web Development
CMSC441	Capstone
CMSC475	UI/UX design
CMSC508	Databases
CMSC515	Computer Vision
CMSC610	Machine Learning

First	Last
Alberto	Cano
Bob	Dahlberg
Sarah	Adams
John	Leonard
Michael	Turner
Emily	Parker

Language
SQL
Python
Perl
C++
Javascript
COBOL
FORTRAN
Java
C#
MATLAB
HTML
CSS
R

RID
1
2
3
4
5
6
7
8
9

Terms and definitions - Schema - Example 2

RID	Course	Instructor	Languages
1	CMSC508 Databases	John Leonard	SQL, Python, Perl
2	CMSC508 Databases	Alberto Cano	SQL, Python, C++
3	CMSC475 UI/UX design	John Leonar	Javascript, Python
4	CMSC441 Capstone	Bob Dahlberg	COBOL, FORTRAN
5	CMSC320 Data Structures	Sarah Adams	C++, Java, Python
6	CMSC210 Software Design	Michael Turner	Java, C#
7	CMSC515 Computer Vision	Emily Parker	Python, MATLAB
8	CMSC430 Web Development	Jessica Clark	HTML, CSS, JavaScript
9	CMSC610 Machine Learning	Albert Cano	Python, R

The *schema* on the left contains multi-valued dependencies, making it difficult to retrieve information.

However, this schema does make it easy to see groups of related data stored within each row. We use *functional dependencies* to document these relationships.

Functional dependencies are an INPUT - that is, the database designer is responsible for describing the relationships between columns in their schema.

We define *normal forms* to describe the strictness of the *functional dependencies*.

Given any schema S consisting of relations R and set of functional dependencies F for each R, a schema is said to be in BCNF when all the relations in S are in BCNF.

The schema on the left, which consists of only one relation, has multivalued dependencies and transitive dependencies and is not in BCNF.

Our goal is to design and implement schemas that are BCNF.

Functional Dependencies - Defined

Given a relation R, a set of attributes X is said to *functionally determine* another set of attributes Y, (written $X \rightarrow Y$):

- If and only if, each value of X is associated with only one value of Y, where X is the antecedent and Y is the consequent of the functional dependency.

A functional dependency $X \rightarrow Y$ holds over relation R if, for every allowable instance r of R:

- For $t_1 \in r$ and $t_2 \in r$, $\Pi_X(t_1) = \Pi_X(t_2)$ implies $\Pi_Y(t_1) = \Pi_Y(t_2)$

If the X values agree then the Y values must also agree.

Summary

- A FD is a statement about relationships between attributes (columns).
- A FD can only and must be identified based on semantics of the application not on current values.
- Given some allowable instances of R, we can check if they violate some FDs, but we cannot tell if always holds over R!

Functional Dependencies - Notations

RID	Course	Instructor	Languages
1	CMSC508 Databases	John Leonard	SQL, Python, Perl
2	CMSC508 Databases	Alberto Cano	SQL, Python, C++
3	CMSC475 UI/UX design	John Leonar	Javascript, Python
4	CMSC441 Capstone	Bob Dahlberg	COBOL, FORTRAN
5	CMSC320 Data Structures	Sarah Adams	C++, Java, Python
6	CMSC210 Software Design	Michael Turner	Java, C#
7	CMSC515 Computer Vision	Emily Parker	Python, MATLAB
8	CMSC430 Web Development	Jessica Clark	HTML, CSS, JavaScript
9	CMSC610 Machine Learning	Albert Cano	Python, R

The *schema* S on the left consists of one *relation* R.

Notation 1:

- Response(RID, Course, Instructor, Languages)
- Functional Dependencies:
  - RID -> Course, Instructor, Languages
  - Instructor -> Languages

Notation 2:

- R(A,B,C,D)
- F(A->B,C,D; C->D)

Big questions:

- Is S in BCNF?
- If not, how do we get there?

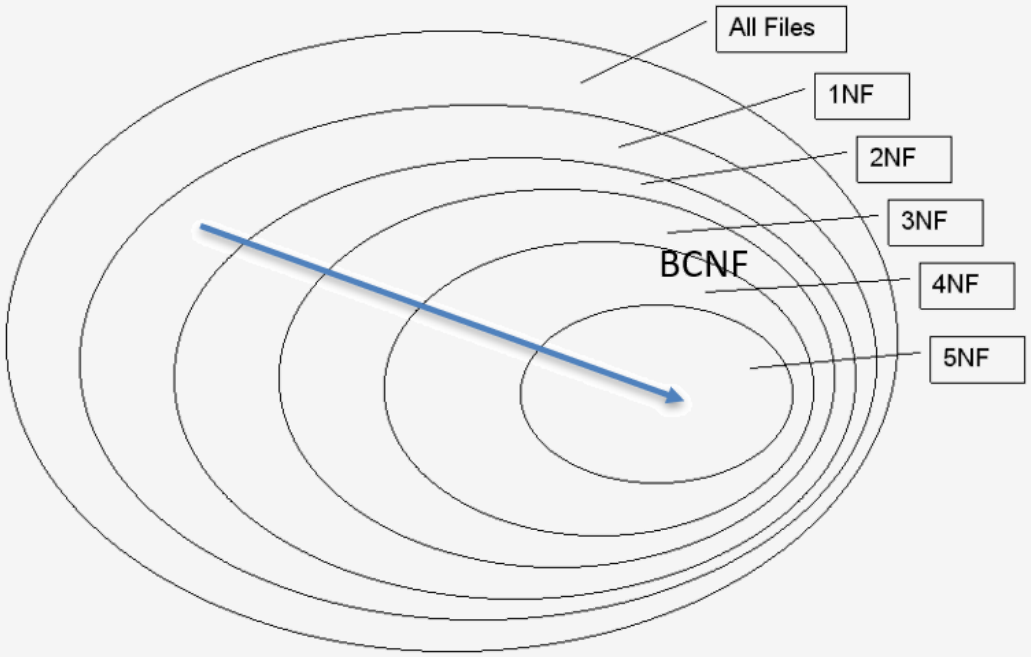
Normal Forms - Overview

Database normalization is the process of reorganizing the relations R in a schema S to minimize data redundancy.

Normalization involves subdividing a relation R into less redundant and smaller relations (fewer columns) without losing information by leveraging the functional dependencies (relationships) of R.

The objective is to isolate related data to minimize duplicates and so modifications of an attribute can be made in just one table and then propagated through the rest of the database using the defined foreign keys and joins.

Escalating through the different normal forms removes more and more redundancy.



Normal Forms - Defined

The usual suspects

First Normal Form - 1NF

A relation is in 1NF if and only if the domain of each attribute contains only atomic (indivisible) values and the value of each attribute contains only a single value from that domain.

Second Normal Form - 2NF

A relation is in 2NF if and only if it is in 1NF and all non-prime attributes (attributes not part of any candidate key) are fully functionally dependent on the entire candidate key.

Third Normal Form - 3NF

A relation is in 3NF if and only if it is in 2NF, and it has no transitive dependencies.

Boyce-Codd Normal Form - BCNF

A relation is in BCNF if and only if it is in 1NF, and for every non-trivial functional dependency A -> B, A is a superkey.

Crazy talk

Fourth Normal Form - 4NF

A relation is in 4NF if and only if it is in BCNF, and it has no multi-valued dependencies.

Fifth Normal Form - 5NF

A relation is in 5NF if and only if it is in 4NF, and it avoids join dependencies.

Sixth Normal Form - 6NF

A relation is in 6NF if and only if it is in 5NF, and it further eliminates all join dependencies and assures that every join dependency can be enforced by the superkeys of the relation.

Seventh Normal Form - 7NF

A relation is in 7NF if and only if it is in 6NF, and it eliminates all combinatorial join dependencies, ensuring that every possible join dependency is addressed.

Recognizing BCNF

Boyce-Codd Normal Form (BCNF) - Example 1

A relation R with functional dependencies F is in BCNF if for all $X \rightarrow Y$ in F^{+}_{min}:

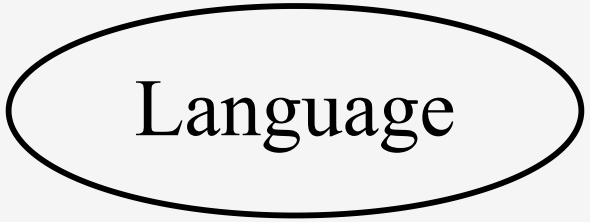
- $Y \subseteq X$ (the trival FD), **OR**
- X is a superkey for R

A relation is in BCNF if and only if it is in 1NF, and for every non-trivial functional dependency $A \rightarrow B$, A is a superkey.

Trival: $X \rightarrow X$

Trivial FD

Language
SQL
Python
Perl
C++
Javascript



Boyce-Codd Normal Form (BCNF) - Example 2

A relation R with functional dependencies F is in BCNF if for all $X \rightarrow Y$ in F^{+}_{min}:

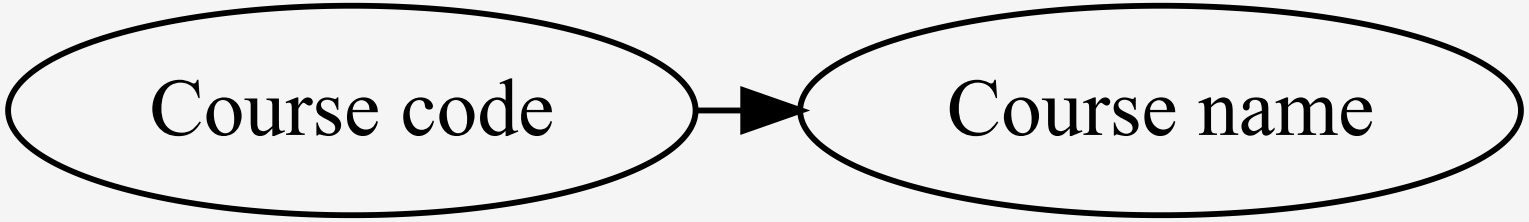
- $Y \subseteq X$ (the trival FD), **OR**
- X is a superkey for R

A relation is in BCNF if and only if it is in 1NF, and for every non-trivial functional dependency $A \rightarrow B$, A is a superkey.

Non-composite key

Course Code	Course Name
CMSC210	Software Design
CMSC320	Data Structures
CMSC430	Web Development
CMSC441	Capstone
CMSC475	UI/UX design
CMSC508	Databases
CMSC515	Computer Vision
CMSC610	Machine Learning

Non-composite key: $X \rightarrow Y$



Boyce-Codd Normal Form (BCNF) - Example 3

A relation R with functional dependencies F is in BCNF if for all $X \rightarrow Y$ in F^{+}_{min}:

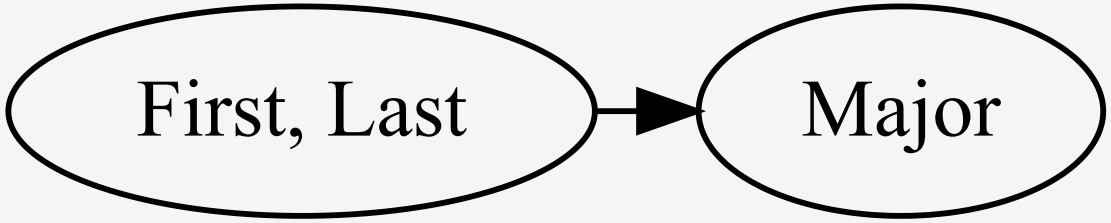
- $Y \subseteq X$ (the trival FD), **OR**
- X is a superkey for R

A relation is in BCNF if and only if it is in 1NF, and for every non-trivial functional dependency $A \rightarrow B$, A is a superkey.

Composite key

First	Last	Major
Alberto	Cano	Comp Sci
Bob	Dahlberg	History
Sarah	Adams	Biology
John	Leonard	Civil Engr
Michael	Turner	Comp Sci
Emily	Parker	English
Jessica	Clark	Mathematics

Composite key: $A, B \rightarrow C$



Boyce-Codd Normal Form (BCNF) - Terms

A relation R with functional dependencies F is in BCNF if for all $X \rightarrow Y$ in F^+_{min}:

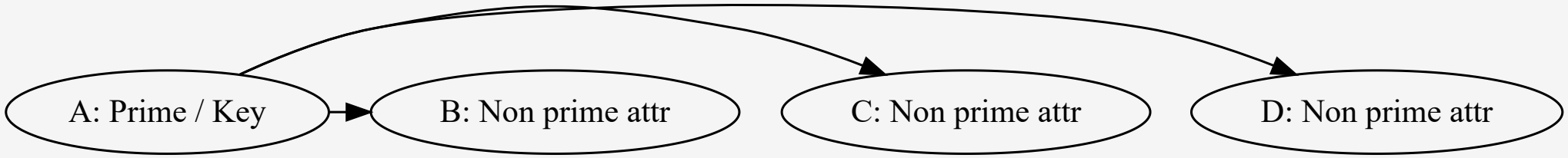
- $Y \subseteq X$ (the trival FD), **OR**
- X is a superkey for R

A relation is in BCNF if and only if it is in 1NF, and for every non-trivial functional dependency $A \rightarrow B$, A is a superkey.

Terms

- Schemas and relations
- Functional dependencies
- Keys: Superkey, key, composite key
- Prime and non-prime attributes
- Armstrong's Axioms
- Attribute closures
- Minimum candidate keys
- F^+_{min} - minimal cover set

Non-composite key: $A \rightarrow B, C, D$



The superkey, the whole key, and nothing but the key, so help me Codd!

Boyce-Codd Normal Form (BCNF) - An algorithm

Algorithm

Repeat for each relation R in schema S:

- Calculate attribute closures $\{X\}^+$
- Determine minimum candidate keys CK_{min}
- Determine prime and non-prime attributes
- Determine minimal cover set F^+_{min}
- Determine highest normal form of relation R
- Decompose input relation R into BCNF

Until all relations R in schema S are BCNF.

Skills for CMSC 508

- By inspection determine if a relation R is BCNF,
- Translate schema S into relational algebraic form,
- Document functional dependencies for each R in S,
- Run calculator by Raymond Cho,
- Describe reports and algorithms used by calculator,
- Interpret reports to find necessary outputs,
- Identify decomposed BCNF relations from report.

Boyce-Codd Normal Form (BCNF) - A Worked Example

RID	Course	Instructor	Languages
1	CMSC508 Databases	John Leonard	SQL, Python, Perl
2	CMSC508 Databases	Alberto Cano	SQL, Python, C++
3	CMSC475 UI/UX design	John Leonard	Javascript, Python
4	CMSC441 Capstone	Bob Dahlberg	COBOL, FORTRAN
5	CMSC320 Data Structures	Sarah Adams	C++, Java, Python
6	CMSC210 Software Design	Michael Turner	Java, C#
7	CMSC515 Computer Vision	Emily Parker	Python, MATLAB
8	CMSC430 Web Development	Jessica Clark	HTML, CSS, JavaScript
9	CMSC610 Machine Learning	Alberto Cano	Python, R

The *schema* S on the left consists of one *relation* R.

In relational model notation:

- $R(A, B, C, D)$
- $F(A \rightarrow B, C, D; B, C \rightarrow D)$

Now for some quick math:

- Inside the calculator by Raymond Cho,

And the resulting schema in BCNF:

- $R_0(A, B, C)$, and
- $R_1(B, C, D)$

Housekeeping

- Review week 7 materials
- Review Homework 5
- Quiz 3 next Wednesday, October 11 covering normalization
- On Wednesday do more problems and explore Homework 5.

Module	Week	Date	Day	Lectures/Quizzes	Deliverables/Notes
Normal forms	7	10/2	Mon	MTG11: L9 (Analysis and Normal Forms 2)	
Normal forms	7	10/4	Wed	MTG12: L10 (Analysis and Normal Forms 3)	
Normal forms	8	10/9	Mon	MTG13: L11 (Quiz review session)	
Normal forms	8	10/11	Wed	MTG14: Quiz 3 today (Analysis and Normal Forms)	
Normal forms	8	10/15	Sun		PrjDel 4 due (Phase 1 submission); HW5 due (Analysis and Normal Forms)
Intro to SQL	9	10/16	Mon	MTG15: L12 (Intro to SQL)	
Intro to SQL	9	10/18	Wed	MTG16: L13 (Intro to API / Python Flask and Click)	