Warmup

Attributes:

Author,

Address (of a person),

Title (of a book),

Genre (of a book),

Pages (in a book).

Suppose we have the FDs:

Author -> Address

Title -> Genre Pages

Decompose into BCNF.

- There are other types of decomposition besides BCNF. Why should we use this one and not another?
- We'd like a decomposition to:
    1. eliminate anomalies
    2. let us recover the original relation with a join (lossless join property)
    3. let us recover the original FDs when recovering the original relation (dependency preservation property)
- BCNF decomposition gives us 1 & 2, but not 3.

- BCNF decomposition guarantees:
  - There are no redundancy, insertion, update, or deletion anomalies.
  - We can recover the original relation with a *natural join.* *(lossless join property)*
- However, we might lose some of the original FDs in the natural join.

Suppose we want to store information about the closest restaurant to various people that serves a certain type of cuisine.

Name	Type	Closest Restaurant of Type
Alice	BBQ	Cozy Corner
Alice	Thai	Bhan Thai
Bob	Pizza	Broadway Pizza
Charlie	Doughnuts	Gibson’s Donuts
Charlie	Thai	Bangkok Alley
Charlie	BBQ	Cozy Corner

What are FDs/keys? Is this in BCNF?

Name	Closest
Alice	Cozy Corner
Alice	Bhan Thai
Bob	Broadway Pizza
Charlie	Donald's Donuts
Charlie	Bangkok Alley
Charlie	Cozy Corner

Restaurant	Type
Cozy Corner	BBQ
Bhan Thai	Thai
Broadway Pizza	Pizza
Donald's Donuts	Doughnuts
Bangkok Alley	Thai

Book's example

- Traveling shows:
  - Store theater names, the cities they are in, and the title of the show playing.
  - A show never plays at more than one theater per city.
- theater -> city
- title city -> theater

3^{rd} Normal Form (3NF)

- Allows for lossless joins and dependency preservation.
- Does not fix all anomalies.
- 3NF is a weaker condition than BCNF (anything in BCNF is automatically in 3NF).

3^{rd} Normal Form (3NF)

- A relation R is in 3NF iff for every nontrivial FD A1…An -> B for R, one of the following is true:
  - A1…An is a superkey for R (BCNF test)
  - Each B is a ***prime*** attribute (an attribute in *some* key for R)

Example

- Say we have R(Course, Dept, Prof, Semester, Year)
- R(C, D, P, S, Y) has FDs
  - PSY → CD
  - CD → S
- Keys are {P, S, Y} and {C, D, P, Y}
- CD → S violates BCNF
- However, R is in 3NF because S is part of a key

3NF Decomposition

- Given a relation R and set F of functional dependencies:

1. Find a minimal basis, G, for F.
2. For each FD X -> A in G, use XA as the schema of one of the relations in the decomposition.
3. If none of the sets of schemas from Step 2 is a superkey for R, add another relation whose schema is a key for R.

Example

- Example:

  R(A, B, C)

  F: {A$\rightarrow$B, C$\rightarrow$B}

- What is the minimal basis set of FDs?

- What is the decomposition to 3NF?

More redundancy?

Course	Textbook	Prof
ENGL 101	Writing for Dummies	Smith
ENGL 101	Wikipedia Is Not a Primary Source	Smith
ENGL 101	Writing for Dummies	Jones
ENGL 101	Wikipedia Is Not a Primary Source	Jones
COMP 142	How to Program in C++	Smith
COMP 142	How to Program in C++	Jones

Every professor always uses the same set of books.

Is this in BCNF?

- Redundancies can still arise in relations that conform to BCNF.
- Occurs when a single table tries to contain two (or more) many-one (or many-many) relationships.

Course	Textbook	Prof
ENGL 101	Writing for Dummies	Smith
ENGL 101	Wikipedia Is Not a Primary Source	Smith
ENGL 101	Writing for Dummies	Jones
ENGL 101	Wikipedia Is Not a Primary Source	Jones
COMP 142	How to Program in C++	Smith
COMP 142	How to Program in C++	Jones

Multivalued dependencies

- A ***MVD*** is a constraint that two sets of attributes are ***independent*** of each other.
- A MVD A1…An ->-> B1…Bm holds in R if in every instance of R:
  - for every pair of tuples t and u that agree on all the As, we can find a tuple v in R that agrees
    - with both t and u on the As
    - with t on the Bs
    - with u on all those attributes of R that are not As or Bs
- In other words, the information in A1..An determines the values of the set of tuples for B1..Bm ***and*** those tuples are independent of any other attributes in the relation.

- Consider a table with actors/actresses, their street addresses with cities/states/zips, and the movies they've been in (title/year).

- Consider a MVD A1…An ->-> B1…Bm.
- Call attributes not in A’s or B’s the Cs.
- This MVD holds in R if:
  – whenever we have two tuples of R that agree on the A’s but differ on the B’s and C’s we should be able to find (or create) two new tuples with the same A’s but swapped B’s and C’s.
- Equivalently:
  – If knowing A1…An determines a unique set of tuples for B1..Bm that is independent of the C’s.

Course	Textbook	Prof
ENGL 101	Writing for Dummies	Smith
ENGL 101	Wikipedia Is Not a Primary Source	Smith
ENGL 101	Writing for Dummies	Jones
ENGL 101	Wikipedia Is Not a Primary Source	Jones
COMP 142	How to Program in C++	Smith
COMP 142	How to Program in C++	Jones

- Course →→ Textbook is an MVD
- What else?

FDs vs MVDs

- A FD A -> B says "Each A determines a unique B"
  - or, "Each A determines 0 or 1 Bs."
- A MVD A ->-> B says "Each A determines a set of Bs ***where the Bs are independent of anything in the relation that is not an A or a B.***"

Rules for MVDs

- **FD promotion:** Every FD A→B is an MD A→→B
- **Trivial MDs:**

1. If A→→B, then A→→AB
2. If A1, A2..., An and B1, B2, ..., Bm make up *all* the attributes of a relation, then A1, A2, ...An →→ B1, B2, ...Bm holds in the relation

- **Transitive rule:** Given A→→B and B→→C, we can infer A→→C.
- **Complementation rule:** if we know A→→B, then we know A→→C, where all the Cs are attributes not among the As or Bs.

- Note that the **splitting rule does not hold!** If $A \to\to BC$, then it is not true that $A \to\to B$ and $A \to\to C$.

Fourth Normal Form (4NF)

- "Stronger" than BCNF.
- A relation R is in 4NF iff:
  - for all MVDs A1…An ->-> B1…Bm,
    {A1, …, An} is a superkey of R.

4NF Decomposition

- Consider relation R with set of attributes X
- A1 A2 … An →→ B1 B2 … Bm violates 4NF
- Decompose R into two relations whose attributes are:

1. The As and Bs together, i.e., {A1 A2 … An, B1, B2, …, Bm}
2. All the attributes of R which are not Bs, i.e. X – {B1, B2 …, Bm}
3. Recursively check if the new relations are in 4NF and repeat

Example

Course	Textbook	Prof
ENGL 101	Writing for Dummies	Smith
ENGL 101	Wikipedia Is Not a Primary Source	Smith
ENGL 101	Writing for Dummies	Jones
ENGL 101	Wikipedia Is Not a Primary Source	Jones
COMP 142	How to Program in C++	Smith
COMP 142	How to Program in C++	Jones

- Course ->-> Textbook
- Course ->-> Professor

Example

`Drinkers(name, addr, phones, beer)`

- FD: `name` $\rightarrow$ `addr`
- Nontrivial MVD's:

  `name` $\rightarrow\rightarrow$ `phone` and
  `name` $\rightarrow\rightarrow$ `beer`.
- Only key: {`name, phones, beer`}
- All three dependencies above violate 4NF.
- Successive decomposition yields 4NF relations:

  D1(<u>name</u>, addr)
  D2(<u>name</u>, <u>phones</u>)
  D3(<u>name</u>, <u>beer</u>)

Relationships Among Normal Forms

- 4NF implies BCNF, i.e., if a relation is in 4NF, it is also in BCNF
- BCNF implies 3NF, i.e., if a relation is in BCNF, it is also in 3NF

Property	3NF	BCNF	4NF
Eliminate redundancy due to FDs	Maybe	Yes	Yes
Eliminate redundancy due to MVDs	No	No	Yes
Preserves FDs	Yes	Maybe	Maybe
Preserves MVDs	Maybe	Maybe	Maybe

Normal Forms

- First Normal Form: each attribute is atomic
- Second Normal Form: No non-trivial FD has a left side that is a proper subset of a key
- Third Normal Form: just discussed it
- Fourth Normal Form: just discussed it
- Fifth Normal Form: outside the scope of this class
- Sixth Normal Form: different versions exist. One version developed for temporal databases
- Seventh Normal Form
  – just kidding ☺

Database Design Mantra

- “everything should depend on the key, the whole key, and nothing but the key”