# Relational Database Design Theory

Introduction to Databases CompSci 316 Spring 2020



1

3

# Announcements (Thu. Feb. 13)

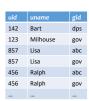
- HW3: Q4-Q5 due Saturday 02/15 \*\*12 NOON\*\*
- Midterm next Tuesday 02/18 in class
  - · Open book, open notes
  - · No electronic devices, no collaboration
  - Everything covered until and including TODAY Thursday 02/13 included!
  - · Sample midterm on sakai -> resources -> midterm
  - HW1, HW2 sample solutions on sakai
- We will move some office hours to next Monday for the midterm
  - Follow piazza announcements

2

#### Today's plan

- Start database design theory
  - Functional dependency, BCNF
- Review some concepts in between and at the end
  - Weak entity set, ISA, multiplicity, etc. in ER diagram
  - Outer joins, different join types
  - Triggers
  - EXISTS
  - Foreign keys

Motivation

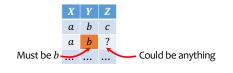


- Why is UserGroup (uid, uname, gid) a bad design?
- Wouldn't it be nice to have a systematic approach to detecting and removing redundancy in designs?
  - Dependencies, decompositions, and normal forms

4

# Functional dependencies

- A functional dependency (FD) has the form X → Y, where X and Y are sets of attributes in a relation R
- *X* → *Y* means that whenever two tuples in *R* agree on all the attributes in *X*, they must also agree on all attributes in *Y*



FD examples

Address (street address, city, state, zip)

5

# Redefining "keys" using FD's

A set of attributes *K* is a key for a relation *R* if

- $K \rightarrow \text{all (other)}$  attributes of R
  - That is, *K* is a "super key"
- No proper subset of *K* satisfies the above condition
  - That is, *K* is minimal

#### Reasoning with FD's

Given a relation R and a set of FD's  $\mathcal{F}$ 

- Does another FD follow from  $\mathcal{F}$ ?
  - Are some of the FD's in F redundant (i.e., they follow from the others)?
- Is *K* a key of *R*?
  - What are all the keys of *R*?

7

8

#### Attribute closure

 Given R, a set of FD's F that hold in R, and a set of attributes Z in R:

The closure of Z (denoted  $Z^+$ ) with respect to  $\mathcal F$  is the set of all attributes  $\{A_1,A_2,...\}$  functionally determined by Z (that is,  $Z\to A_1A_2$ ...)

- Algorithm for computing the closure
- Example
  On board
  Using next slide

 $\mathcal{F}$  includes:

uid → uname, twitterid twitterid → uid

uid, gid  $\rightarrow$  fromDate

- Start with closure = Z
- If  $X \to Y$  is in  $\mathcal F$  and X is already in the closure, then also add Y to the closure
- Repeat until no new attributes can be added

#### A more complex example

UserJoinsGroup (uid, uname, twitterid, gid, fromDate)

Assume that there is a 1-1 correspondence between our users and Twitter accounts

- uid  $\rightarrow$  uname, twitterid
- twitterid  $\rightarrow$  uid
- uid, gid  $\rightarrow$  fromDate

Not a good design, and we will see why shortly

9

10

# Example of computing closure

- {gid, twitterid}+ = ?
- twitterid → uid
  - Add uid
  - Closure grows to { gid, twitterid, uid }
- uid  $\rightarrow$  uname, twitterid
  - Add uname, twitterid
  - Closure grows to { gid, twitterid, uid, uname }
- uid, gid → fromDate
  - Add fromDate
  - Closure is now all attributes in UserJoinsGroup

# Using attribute closure

Given a relation R and set of FD's  $\mathcal{F}$ 

- Does another FD  $X \to Y$  follow from  $\mathcal{F}$ ?
  - Compute  $X^+$  with respect to  $\mathcal F$
  - If  $Y \subseteq X^+$ , then  $X \to Y$  follows from  $\mathcal F$
- Is *K* a key of *R*?
  - Compute  $K^+$  with respect to  $\mathcal{F}$
  - If  $K^+$  contains all the attributes of R, K is a super key
  - Still need to verify that K is minimal (how?)

11

# Rules of FD's

All intuitive but check yourself!

- Armstrong's axioms
  - Reflexivity: If  $Y \subseteq X$ , then  $X \to Y$
  - Augmentation: If  $X \to Y$ , then  $XZ \to YZ$  for any Z
  - Transitivity: If  $X \to Y$  and  $Y \to Z$ , then  $X \to Z$
- Rules derived from axioms
  - Splitting: If  $X \to YZ$ , then  $X \to Y$  and  $X \to Z$
  - Combining: If  $X \to Y$  and  $X \to Z$ , then  $X \to YZ$
- "Using these rules, you can prove or disprove an FD given a set of FDs

(Problems with) Non-key FD's

- Consider a non-trivial FD  $X \to Y$  where X is not a super key
  - Since X is not a super key, there are some attributes (say Z) that are not functionally determined by X



That *b* is associated with *a* is recorded multiple times: redundancy, update/insertion/deletion anomaly

13

# Example of redundancy

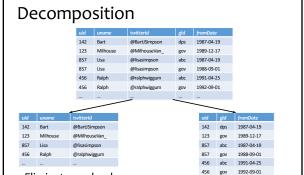
UserJoinsGroup (uid, uname, twitterid, gid, fromDate)

• uid  $\rightarrow$  uname, twitterid

(... plus other FD's)

uid	uname	twitterid	gid	fromDate
142	Bart	@BartJSimpson	dps	1987-04-19
123	Milhouse	@MilhouseVan_	gov	1989-12-17
857	Lisa	@lisasimpson	abc	1987-04-19
857	Lisa	@lisasimpson	gov	1988-09-01
456	Ralph	@ralphwiggum	abc	1991-04-25
456	Ralph	@ralphwiggum	gov	1992-09-01

14

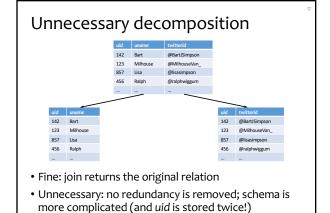


• Eliminates redundancy

• To get back to the original relation: ⋈

15

16



Bad decomposition 142 dps 123 gov 857 abc 1989-12-17 1987-04-19 456 abc 1991-04-25 142 dps 142 1987-04-19 857 abc 857 1987-04-19 456 1991-04-25 456 456 1992-09-01 • Association between gid and from Date is lost

• Join returns more rows than the original relation

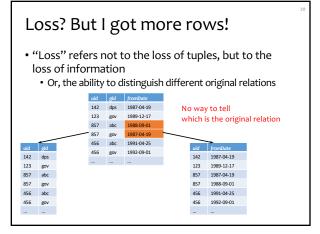
17

#### Lossless join decomposition

Example on board Check definition yourself

- Decompose relation R into relations S and T
  - $attrs(R) = attrs(S) \cup attrs(T)$
  - $S = \pi_{attrs(S)}(R)$
  - $T = \pi_{attrs(T)}(R)$
- The decomposition is a lossless join decomposition if, given known constraints such as FD's, we can guarantee that  $R=S\bowtie T$
- Any decomposition gives  $R \subseteq S \bowtie T$  (why?)
  - A lossy decomposition is one with  $R \subset S \bowtie T$

19



20

# Questions about decomposition

- When to decompose
- How to come up with a correct decomposition (i.e., lossless join decomposition)

#### An answer: BCNF

- A relation R is in Boyce-Codd Normal Form if
  - For every non-trivial FD  $X \rightarrow Y$  in R, X is a super key
  - That is, all FDs follow from "key → other attributes"
- When to decompose
  - · As long as some relation is not in BCNF
- How to come up with a correct decomposition
  - Always decompose on a BCNF violation (details next)
  - Then it is guaranteed to be a lossless join decomposition!

21 22

# BCNF decomposition algorithm

- Find a BCNF violation
  - That is, a non-trivial FD X → Y in R where X is not a super key of R
- Decompose R into  $R_1$  and  $R_2$ , where
  - $R_1$  has attributes  $X \cup Y$
  - R<sub>2</sub> has attributes X ∪ Z, where Z contains all attributes of R that are in neither X nor Y
- Repeat until all relations are in BCNF

BCNF decomposition example

uid → uname, twitterid twitterid → uid uid, gid → fromDate

User JoinsGroup (uid, uname, twitterid, gid, fromDate)

BCNF violation: uid → uname, twitterid

User (uid, uname, twitterid)

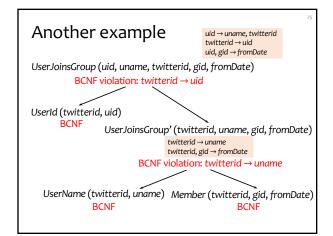
uid → uname, twitterid

uid → uname, twitterid

twitterid → uid

BCNF

23 24



Why is BCNF decomposition lossless

Given non-trivial  $X \to Y$  in R where X is not a super key of R, need to prove:

- Anything we project always comes back in the join:
  - $R \subseteq \pi_{XY}(R) \bowtie \pi_{XZ}(R)$  Sure; and it doesn't depend on the FD
- Check and prove yourself!
- Anything that comes back in the join must be in the original relation:
  - $R \supseteq \pi_{XY}(R) \bowtie \pi_{XZ}(R)$
  - Proof will make use of the fact that  $X \to Y$

25 26

#### Recap

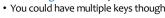
27

 Functional dependencies: a generalization of the key concept

- Non-key functional dependencies: a source of redundancy
- BCNF decomposition: a method for removing redundancies
  - BNCF decomposition is a lossless join decomposition
- BCNF: schema in this normal form has no redundancy due to FD's

#### Summary

Philosophy behind BCNF:
 Data should depend on the key, the whole key, and nothing but the key!



• Other normal forms

- 4NF and Multi-valued-dependencies: later in the course
- Not covered

28

- 3NF: More relaxed than BCNF; will not remove redundancy if doing so makes FDs harder to enforce
- 2NF: Slightly more relaxed than 3NF
- 1NF: All column values must be atomic

