CS 405G: Introduction to Database Systems

Functional Dependency

Today's Topic

- Functional Dependency.
- Normalization
- Decomposition
- BCNF

Motivation

- How do we tell if a design is bad, e.g., *WorkOn*(*EID*, *Ename*, *PID*, *Pname*, *Hours*)?
  - This design has *redundancy*, because the name of an employee is recorded multiple times, once for each project the employee is taking

EID	PID	Ename	Pname	Hours
1234	10	John Smith	B2B platform	10
1123	9	Ben Liu	CRM	40
1234	9	John Smith	CRM	30
1023	10	Susan Sidhuk	B2B platform	40

Update anomality
Insert anonmaity: Bar not taking classes
Delet anomaly: Bart drops all classes

Why redundancy is bad?

- Waste disk space.
- What if we want to perform update operations to the relation
  - INSERT an new project that no employee has been assigned to it yet.
  - UPDATE the name of "John Smith" to "John L. Smith"
  - DELETE the last employee who works for a certain project

EID	PID	Ename	Pname	Hours
1234	10	John Smith	B2B platform	10
1123	9	Ben Liu	CRM	40
1234	9	John Smith	CRM	30
1023	10	Susan Sidhuk	B2B platform	40

Functional Dependency

- FDs are **constraints** that are derived from the *meaning* and *interrelationships* of the data attributes
- Functional dependencies (FDs) are used to specify *formal measures* of the "goodness" of relational designs
- FDs and keys are used to define **normal forms** for relations

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*
  - $t_1[X] = t_2[X] \Rightarrow t_1[Y] = t_2[Y]$

X	Y	Z
a	b	c
a	b	d

Must be "b"

Could be anything, e.g. d

FD examples

Address (*street_address*, *city*, *state*, *zip*)

- *street_address*, *city*, *state* -> *zip*
- *zip* -> *city*, *state*
- *zip*, *state* -> *zip*?
  - This is a trivial FD
  - Trivial FD: LHS ⊇ RHS
- *zip* -> *state*, *zip*?
  - This is non-trivial, but not completely non-trivial
  - Completely non-trivial FD: LHS ∩ RHS = ?

Functional Dependencies

- An FD is a property of the attributes in the schema R
- The constraint must hold on *every relation instance* r(R)
- If K is a key of R, then K functionally determines all attributes in R (since we never have two distinct tuples with t1[K]=t2[K])

Keys redefined using FD's

Let *attr*(*R*) be the set of all attributes of *R*, a set of attributes *K* is a (candidate) key for a relation *R* if

- *K* -> attr(*R*) - *K*, and
  - That is, *K* is a "super key"
- No proper subset of *K* satisfies the above condition
  - That is, *K* is minimal (full functional dependent)
- *Address* (*street_address*, *city*, *state*, *zip*)
  - {street_address, city, state, zip} — Super key
  - {street_address, city, zip} — Super key
  - {street_address, zip} — Key
  - {zip} — Non-key

Reasoning with FD's

Given a relation R and a set of FD's F

- Does another FD follow from 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?

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 F is the set of all attributes $\{A_1, A_2, \ldots\}$ functionally determined by Z (that is, Z -> A_1 A_2 …)
- Algorithm for computing the closure
  - Start with closure = Z
  - If X -> Y is in F and X is already in the closure, then also add Y to the closure
  - Repeat until no more attributes can be added

A more complex example

WorkOn(*EID*, *Ename*, *email*, *PID*, *Pname*, *Hours*)

- *EID* -> *Ename*, *email*
- *email* -> *EID*
- *PID* -> *Pname*
- *EID*, *PID* -> *Hours*

(Not a good design, and we will see why later)

Example of computing closure

- F includes:
  - *EID* -> *Ename*, *email*
  - *email* -> *EID*
  - *PID* -> *Pname*
  - *EID*, *PID* -> *Hours*
- { *PID*, *email* }$^{+}$ = ?
- closure = { *PID*, *email* }
- *email* -> *EID*
  - Add *EID*; closure is now { *PID, email, EID* }
- *EID* -> *Ename, email*
  - Add *Ename, email*; closure is now { *PID*, *email*, *EID*, *Ename* }
- *PID* -> *Pname*
  - Add *Pname;* close is now { *PID*, *Pname*, *email*, *EID*, *Ename* }
- *EID*, *PID* -> *hours*
  - Add *hours*; closure is now all the attributes in *WorksOn*

Using attribute closure

Given a relation R and set of FD's F

- Does another FD X -> Y follow from F?
  - Compute X^+ with respect to F
  - If $Y \subset X^+$, then X -> Y follow from F
- Is K a super key of R?
  - Compute K^+ with respect to F
  - If K^+ contains all the attributes of R, K is a super key
- Is a super key K a key of R?
  - Test where $K' = K - \{ a \mid a \in K\}$ is a superkey of R for all possible a

Rules of FD's

- Armstrong's axioms
  - Reflexivity: If $Y \subset X$, then $X \rightarrow Y$
  - Augmentation: If $X \rightarrow Y$, then $XZ \rightarrow YZ$ for any Z
  - Transitivity: If $X \rightarrow Y$ and $Y \rightarrow Z$, then $X \rightarrow Z$
- Rules derived from axioms
  - Splitting: If $X \rightarrow YZ$, then $X \rightarrow Y$ and $X \rightarrow Z$
  - Combining: If $X \rightarrow Y$ and $X \rightarrow Z$, then $X \rightarrow YZ$

Using rules of FD's

Given a relation *R* and set of FD's F

- Does another FD *X* -> *Y* follow from F?
  - Use the rules to come up with a proof
- Example:
  - F includes:
    EID -> *Ename*, *email*; *email* -> *EID*; *EID*, *PID* -> *Hours,*
    Pid -> *Pname*
  - *PID*, *email* ->*hours*?
    email -> *EID* (given in F)
    PID, email -> *PID, EID* (augmentation)
    PID, *EID* -> *hours* (given in F)
    PID, *email* -> *hours* (transitivity)

Example of redundancy

- *WorkOn* (*EID*, *Ename*, *email*, *PID*, *hour*)
- We say *X* -> *Y* is a *partial dependency* if there exist a *X'* $\subset$ *X* such that *X'* -> *Y*
  - *e.g. EID, email-> Ename, email*
- Otherwise, *X* -> *Y* is a *full dependency*
  - *e.g. EID, PID -> hours*

EID	PID	Ename	email	Pname	Hours
1234	10	John Smith	jsmith@ac.com	B2B platform	10
1123	9	Ben Liu	bliu@ac.com	CRM	40
1234	9	John Smith	jsmith@ac.com	CRM	30
1023	10	Susan Sidhuk	ssidhuk@ac.com	B2B platform	40

? Why is that