Lecture 04: Functional Dependencies

15-445/645 Database Systems (Fall 2017)
Carnegie Mellon University
Prof. Andy Pavlo

Database Design

- Metrics for a good database design
  1. Data Integrity (no loss of data)
  2. Good performance
- Integrity vs performance is a common tradeoff in databases

Redundancy

- Issues with example database
  - Duplicate columns
  - Duplicate entries (Obama appears twice)
  - Update anomalies: Changes to room numbers means all records need to be updated
  - Insert anomalies: May be impossible to add student to DB if theyre not enrolled in a course
  - Delete anomalies: If all students are deleted, we may lose the room number for a course
- The changes done to better this database is called **decomposition**

Functional Dependencies

- A functional dependency is a form of constraint
- Basic idea: A value of a variable depends on the value of another variable
- Example: sid→name because name depends on sid (”student ID implies name”)
- You can check if an FD is violated by an instance, but you can’t prove a FD using just an instance
- Two FDs X→Y and X→Z can be written X→YZ
  - But XY→Z is not the same as X→Z and X→Y
- Defining FDS in SQL

```
CREATE ASSERTION student-name
   CHECK (NOT EXISTS
   (SELECT * FROM students AS s1,
                  students AS s2
   WHERE s1.sid = s2.sid
   AND s1.name <> s2.name))
```

- Issues with FDs in SQL
  - Performance: Need to validate FD across entire table when inserting or updating a tuple
  - **No major DBMS supports SQL-92 assertions**
- **FDs are important because they allow us to decide if our database design is correct**

Closures and Canonical Covers

- Given a set of FDs $f_1, ..., f_n$ we define the Closure F+ as the set of all implied FDs
- Given a closure, the attribute closeure is: Given an attribute X , the closure X+ is the set of all attributes such that X→A can be inferred using Armstrong's axioms
- Why are closures important
  - Checking closure at runtime is expensive
  - We want minimal set of FDs that is enough to ensure correctness
- The minimal set of all FDs is called **the canonical cover**
- A canonical cover F_c must have the following properties
  1. The RHS of every FD is a single attribute
  2. The closure of F_c is identical to the closure of F
  3. The F_c is minimal (deleting any attribute from LHS or RHS of an FD violates property #2)
- Why do canonical covers matter
  - the canonical cover is the minimum number of assertions needed to assure database integrity and correctness
  - They allow us to find the **super key**

Super and Candidate Keys

- **Super key:** set of attributes where no distinct tuples have the same values for these attributes
  - Allow us to determine whether we can decompose a table into multiple sub-tables
  - Allow us to ensure that we are able to recreate the original relation through joins
- **Candidate key:** set of attributes that uniquely identify a tuple according to a key constraint
- A candidate key is a super key, but not all super keys are candidates

Schema Decompositions

- **Objective:** Split a single relation R into a set of relations $R_1, ..., R_n$
- **Goals (in order of importance)**
  1. (MANDATORY) Lossless joins: Want to be able to construct original relation by joining smaller ones using a natural join
  2. Dependency preservation: Minimize cost of global integrity constraints based on FD's. (very expensive)
  3. Redundancy avoidance: avoid unnecessary data duplication
- A schema preserves dependencies if its original FD's do not span multiple tables

Lossless Joins

→ Motivation: Avoid information loss.
→ Goal: No noise introduced when reconstituting universal relation via joins.
→ Test: At each decomposition $R=(R_1 \cup R_2)$, check whether $(R_1 \cap R_2) \rightarrow R_1$ or $(R_1 \cap R_2) \rightarrow R2$.

Dependency Preservation

→ Motivation: Efficient FD assertions.
→ Goal: No global integrity constraints that require joins of more than one table with itself.
→ Test: $R=(R_1 \cup ... \cup R_n)$ is dependency preserving if closure of FD's covered by each R_1 = closure of FD's covered by $R=F$.

Redundancy Avoidance

→ Motivation: Avoid update, delete anomalies.
→ Goal: Avoid update anomalies, wasted space.
→ Test: For an $X \rightarrow Y$ covered by R_n, X should be a super key of R_n.