Programming Languages (CS302 2007S)

Class 35: SQL (1): The Relational Model

Held: Wednesday, April 25, 2007

Summary: Today we begin our exploration of declarative database languages by considering one of the first papers on relational databases.

Related Pages:

- EBoard.
- Reading: Codd - A Relational Model for Large Shared Data Banks.

Overview:

- Context.
- Codd’s Contributions.
- Why Read This Paper?
- Thinking About Relations.
- Operations on Relations.

Context

What is a database?

- I’ll let you think about answers to that question.

How do we implement them (without a DBMS)?

- Again, I’ll leave that up to you.

What do we know about the state of computing (and databases) at the time this was written?

- Most of the data was on disk (or tape!), so access was slow.
- DBMS’s were rare.
- Client programmers knew and relied on the structure of data in their code.
- Two primary data models: Hierarchical and Network
  - Hierarchical: Each portion of the database (file) is a homogeneous collection of records. Records may contain other records.
  - Network: Records can be shared by other records.
- ...

Codd’s Contributions

What are the primary contributions of this paper? Here are some.

- A clear attempt to separate what you want to do with data from how the data are represented.
- A model of data that persists to today.
- The use of formal systems to explain the model.

Why do we care about separating access from representation?

- Easier to modify the representation.
- Easier to write programs.
- Easier to modify the representation *on the fly*, as you observe (or are warned of) different patterns of access.

Why Read This Paper?

Up to you to answer this question.

Thinking About Relations

Codd treats relations in a variety of ways in the paper.

- At one level, a relation is simply a *set of tuples*.
- We can also think of each relation as a *predicate* (a function that returns true or false). The relation, when applied to a tuple, returns true only when the tuple belongs to the set.
- We can also think of each relation as a *function* from *n*-*m* of the values in the tuple to tuples of the remaining *m* values.
    - Of course, this is a kind of multiple-valued function. Consider

      ```
      A  B
      0  1
      0  2
      1  1
      3  1
      ```

    - If we supply 0 as the A parameter, we can get back 1 or 2.

Operations on Relations

- Two basic operations
    - *join* - combine two tables in a systematic way to create a new table.
    - *project* - select only certain columns from a table.
- Variations
    - *permute* - projection in which each column appears exactly once.
    - *compose* - combined join and project, typically applicable when we think of relations as

functions.

- ...