CS143: Relational Model

Book Chapters

(4th) Chapters 1.3-5, 3.1, 4.11
(5th) Chapters 1.3-7, 2.1, 3.1-2, 4.1
(6th) Chapters 1.3-6, 2.105, 3.1-2, 4.5

Things to Learn

- Data model
- Relational model
- Database construction steps

Data Model

- The way we model/conceptualize/visualize/represent data
- Need some representation to manage data in a computer
- Many different ways to model data
  - **Example (Airline flight):** Graph model
    - Node: city
    - Edge: flight between cities
    - Label on edge: flight time, etc.
  - **Example (Company hierarchy):** Tree model
    - CEO → Presidents → Vice presidents → Department heads ...
  - Many others, like object model, XML model, network model, etc.
  - Models to learn in the class: Relational and E/R model

Example to Use in the Class

- School information
  - Student(sid, name, age, GPA, address, ...)
  - Class(dept, cnum, sec, title, instructor, ...)
  - Enroll(sid, dept, cnum, sec)
  - ...

Relational Model

- **Example:** Student(sid, name, address, age, GPA)
  Student

sid	name	addr	age	GPA
301	John	183 Westwood	19	2.1
303	Elaine	301 Wilshire	17	3.9
401	James	183 Westwood	17	3.5
208	Esther	421 Wilshire	20	3.1

- All data is represented as *relations* (= *tables*)
- Each relation has a set of *attributes* (= *columns*)
- Each relation contains of a set of *tuples* (= *rows*)
- Each attribute has a *domain* (= *type*)
  - Only atomic types
- Similar to Excel spreadsheet

History of Relational Model

- By far, the most significant invention in the history of DBMS
  - E.F. Codd, 1970
  - Completely revolutionized the field ($9B market in 2001 according to Gartner)
  - Before it, network and hierarchical model: difficult to use and pose queries
  - Turing Award, 1981
- *Extremely* simple
- Used by practically all commercial DBMS these days
- An argument for simplicity
- Initial research prototypes
  - IBM System R → IBM DB2
  - Berkeley Postgres → PostgresSQL
- IBM and Oracle: first commercial RDB vendor
  - Still dominate the market

Concepts and Terminology

Schema

The structure of relations in database: relation name, attribute name, domain (optional).

- **Example:**
  - Student(sid, name, address, GPA, age)
  - Course(dept: char(2), cnum: int, sec: int, unit: int, title: char(100))
    char(2): string of length 2

Instance (= Data)

Actual contents (tuples) of relation (explain using the table example)

- Schema ≈ Type, Instance ≈ Value
- Schema ≈ Class, Instance ≈ Instance

Keys

- A set of attributes that are known to be unique in the relation
  - Student(sid, name, address, GPA, age)
  - Course(dept, cnum, sec, unit, instructor, title)
- Multiple keys possible
  - Course(dept, cnum, sec, unit, instructor, title)
  - Course(dept, cnum, sec, unit, instructor, title)
  - Course(dept, cnum, sec, unit, instructor, title)
- **Q:** When do we need keys? How can they be used?

Name Scope

- Names of relation: Unique across relations
- Names of attributes: Unique in a table, same name in different tables OK

Set semantics

- No duplicate tuples (different in SQL. More discussion later)
  - Q: Can a relation with no duplicates have no keys?
- Tuple order does not matter
  - Authors of a paper: Need explicit ordering
- Orders of attributes do not matter

Null value

- Common interpretation
  - Do not know
  - Do not want to say
  - Not applicable
  - Example: Student(id, dept, name, GPA) – before first quarter?
- Complications from Null
  - Example: Student(id, dept, name, age, GPA)
  - Q_1: Find students whose age ≥ 20. Susan's age is Null. Susan in the result?
  - Q_2: Find students whose age < 20. Susan in the result?
  - Q_3: Find students whose age ≥ 20 or age < 20 and. Susan in the result?
    - But $Q_3 = Q_1 \cup Q_2$? Something is wrong.
- Relational algebra, SQL: 3-valued logic
  - Every condition is evaluated as True, False or Unknown
  - Various (arbitrary) rules to deal with anomalous situation
  - More discussion later
- Unfortunately, Nulls are very common in DB.
- Significant complexity in DBMS implementation

Steps in Database Construction

Flow chart diagram

1. *Domain Analysis*: Understand application-domain semantics being captured
   - E/R diagram
   - discussed in the latter half of the course
2. *Database design*: Design tables to capture the information
   - Relational design theory (funtional dependency, normal form, etc)
   - discussed in the latter half of the course
3. *Table creation*: using Database Definition Language
   - DDL: A language to define relations and their characteristics:
     - Schema, integrity constraints, indexes, ...
4. *Load*: typically bulk-load. insert tuple possible
5. *Query and update*: using Data Manipulation Language
   - DML: A language to query and update relations

SQL and DDL, Load, DML

What is SQL?

- Structured Query Language
- The standard language for interacting with all commericial RDBMS
- The history of SQL standard
  - SQL89 (Ansi SQL): first standard
  - SQL92 (SQL2): the main standard. several hundred pages
  - SQL3 (SQL99): no vendor supports it all! 1600 pages
  - SQL4 (SQL03): bug-fix release
  - SQL5 (SQL06): extensions for XML
  - SQL6 (SQL08): merge, truncate, diagnostics
  - We will mainly use the standard SQL92 in class. Individual product uses slight variations of the standard. Some class query may not run on them.
- SQL has many components
  - DDL: Schema definition, constraints, indexes, ...
  - DML: data retrieval, modification, ...
  - Transactions, Authorization, ...
- We learn schema definition part today.

Basic SQL Types

- Basic SQL types (commonly used subset)
  - String
    - Char(n) – padded fixed length
    - Varchar(n) – variable length
  - Number
    - Integer – 32bit
    - Decimal(5,2) – 999.99
    - Real, Double – 32bit, 64bit
  - Datetime
    - Date – '2002-01-15'
    - Time – '13:50:00'
    - Timestamp – '2002-01-15 13:50:00'
- Schema definition (table creation)
  - Course(<u>dept</u>, <u>cnum</u>, <u>sec</u>, unit, instructor, title)

    * `CREATE TABLE Course (`
      `dept CHAR(2) NOT NULL,`
      `cnum INTEGER NOT NULL,`
      `sec INTEGER NOT NULL,`
      `unit INTEGER,`
      `instructor VARCHAR(30),`
      `title VARCHAR(30),`
      `PRIMARY KEY(dept, cnum, sec) )`
    * No Null in primary key

  – Course(dept, cnum, sec, unit, instructor, title)
    Course(dept, cnum, sec, unit, instructor, title)
    Course(dept, cnum, sec, unit, instructor, title)
    * `CREATE TABLE Course (`
      `dept CHAR(2) NOT NULL DEFAULT 'CS',`
      `cnum INTEGER NOT NULL,`
      `sec INTEGER NOT NULL,`
      `unit INTEGER,`
      `instructor VARCHAR(30),`
      `title VARCHAR(30) DEFAULT,`
      `PRIMARY KEY(dept, cnum, sec),`
      `UNIQUE(dept, cnum, instructor),`
      `UNIQUE(dept, sec, title) )`
    * One primary key per table
    * Unique for other keys
    * Primary key, unique are enforced through index (more discussion later)
    * SQL92: No Null in primary key. Null OK for unique (DB2: No Null for unique).
    * DEFAULT for default values

- SQL for dropping a table
  – `DROP TABLE Course`

Loading data

- Vendor specific
- Oracle, MySQL
  – `LOAD DATA INFILE <datafile> INTO TABLE Course`
- DB2
  – `IMPORT FROM <datafile> OF DEL INSERT INTO Course`
    * Bulk-load from a comma separated file

Querying and update: Data Manipulation Language (DML)

- DML: A language to query and update relations
- Relational query languages
  - Relational algebra (formal), SQL (practical)
  - What is a relational query?
    - Input: relation
    - Output: relation
    - *Input relation* $\longrightarrow$ $\boxed{\text{query}}$ $\longrightarrow$ *Output relation*
  - More details later

Things to remember

- Data model
- Schema
- Instance
- Relational model
  - relation, attribute, tuple, domain
  - key
  - null value
  - set semantics
- Database construction steps
  1. Domain analysis
  2. Database design: E/R model, database design theory
  3. Table creation: DDL
  4. Load
  5. Query and update: DML