Relational Database Design
Part I

CPS 116

Introduction to Database Systems

Announcements (September 4)

- rack040 accounts created; change your password!
  - Let me know if you have NOT received the email
- Homework #1 is out (earlier than scheduled)
  - Due in two weeks
- Reading: see Tentative Syllabus on course website

Relational model: review

- A database is a collection of relations (or tables)
- Each relation has a list of attributes (or columns)
- Each attribute has a domain (or type)
- Each relation contains a set of tuples (or rows)

Keys

- A set of attributes K is a key for a relation R if
  - In no instance of R will two different tuples agree on all attributes of K
    - That is, K is a "tuple identifier"
  - No proper subset of K satisfies the above condition
    - That is, K is minimal
- Example: *Student* (*SID*, *name*, *age*, *GPA*)
  - *SID* is a key of *Student*
  - *age* is not a key (not an identifier)
  - {*SID*, *name*} is not a key (not minimal)

Schema vs. data

Student

SID	name	age	GPA
142	Bart	10	2.3
123	Milhouse	10	3.1
857	Lisa	8	3.2
456	Ralph	8	2.3

- Is *name* a key of *Student*?
  - Yes?
  - No!
- Key declarations are part of the schema

More examples of keys

- *Enroll* (*SID*, *CID*)
  - ☞A key can contain multiple attributes!
- *Address* (*street_address*, *city*, *state*, *zip*)
- A relation can have multiple keys!
  - We typically pick one as the "primary" key, and underline all its attributes, e.g.,

Usage of keys

- More constraints on data, fewer mistakes
- Look up a row by its key value
  - Many selection conditions are "key = value"
- "Pointers"
  - Example: *Enroll* (*SID*, *CID*)
    - *SID* is a key of *Student*
    - *CID* is a key of *Course*
    - An *Enroll* tuple "links" a *Student* tuple with a *Course* tuple
  - Many join conditions are "key = key value stored in another table"

Database design

- Understand the real-world domain being modeled
- Specify it using a database design model
  - More intuitive and convenient for schema design
  - But not necessarily implemented by DBMS
  - A few popular ones:
    - Entity/Relationship (E/R) model
    - Object Definition Language (ODL)
    - UML (Unified Modeling Language)
- Translate specification to the data model of DBMS
  - Relational, XML, object-oriented, etc.
- Create DBMS schema

Entity-relationship (E/R) model

- Historically and still very popular
- Can think of as a "watered-down" object-oriented design model
- Primarily a design model—not directly implemented by DBMS
- Designs represented by E/R diagrams
  - We use the style of E/R diagram covered by GMUW; there are other styles/extensions
  - Very similar to UML diagrams