Relational Database Design
Part I

CPS 116
Introduction to Database Systems

Announcements (Tue. Sep. 1)

- cps116.cod accounts created; change your password!
  - Let me know if you have NOT received the email
- cps116@cs.duke.edu address disabled due to spam
- Homework #1 is out
  - Due in two weeks, but start early—as soon as any portion has been covered
  - Remember to purchase Gradiance access
- Readings: see Tentative Syllabus on course website
  - Also posted chapters in the first edition

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

<table>
<thead>
<tr>
<th>Student</th>
<th>Name</th>
<th>Age</th>
<th>GPA</th>
</tr>
</thead>
<tbody>
<tr>
<td>456</td>
<td>Ralph</td>
<td>8</td>
<td>2.3</td>
</tr>
</tbody>
</table>

- Is $name$ a key of $Student$?
  - Yes? Seems reasonable for this instance
  - No! Student names are not unique in general
- Key declarations are part of the schema

More examples of keys

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

E/R basics

- Entity: a "thing," like an object
- Entity set: a collection of things of the same type, like a relation of tuples or a class of objects
  - Represented as a rectangle
- Relationship: an association among entities
- Relationship set: a set of relationships of the same type (among same entity sets)
  - Represented as a diamond
- Attributes: properties of entities or relationships, like attributes of tuples or objects
  - Represented as ovals

An example E/R diagram

- Students enroll in courses
  - `SID name` → `Students` → `Enroll` → `Course` → `CID title`
- A key of an entity set is represented by underlining all attributes in the key
  - A key is a set of attributes whose values can belong to at most one entity in an entity set—like a key of a relation

Attributes of relationships

- Example: students take courses and receive grades
  - `SID name` → `Students` → `Enroll` → `Course` → `CID title`
- Where do the grades go?
  - With `Students`?
    - But a student can have different grades for multiple courses
  - With `Course`?
    - But a course can assign different grades for multiple students
  - With `Enroll`?
More on relationships

- There could be multiple relationship sets between the same entity sets
  - Example: Students Enroll Courses; Students TA Courses
- In a relationship set, each relationship is uniquely identified by the entities it connects
  - Example: Between Bart and CPS116, there can be at most one Enroll relationship and at most one TA relationship

What if Bart took CPS116 twice and got two different grades?

Multiplicity of relationships

- $E$ and $F$: entity sets
- Many-many: Each entity in $E$ is related to 0 or more entities in $F$ and vice versa
  - Example:
- Many-one: Each entity in $E$ is related to 0 or 1 entity in $F$, but each entity in $F$ is related to 0 or more in $E$
  - Example:
- One-one: Each entity in $E$ is related to 0 or 1 entity in $F$ and vice versa
  - Example:
  - “One” (0 or 1) is represented by an arrow
  - “Exactly one” is represented by a rounded arrow

N-ary relationships

- Example: Each course has multiple TA’s; each student is assigned to one TA

Meaning of an arrow into $E$: Pick one entity from each of the other entity sets; together they must be related to either 0 or 1 entity in $E$

N-ary versus binary relationships

- Can we model $n$-ary relationships using just binary relationships?

No; for example:
  - Bart takes CPS116 and CPS114
  - Lisa TA’s CPS116 and CPS114
  - Bart is assigned to Lisa in CPS116, but not in CPS114

Roles in relationships

- An entity set may participate more than once in a relationship set
  - May need to label edges to distinguish roles
- Examples
  - People are married as husband and wife; label needed
  - People are roommates of each other; label not needed

Weak entity sets

- Sometimes, the key of an entity set $E$ comes not completely from its own attributes, but from the keys of other (one or more) entity sets; to which $E$ is linked by many-one (or one-one) relationship sets
  - Example: Rooms inside Buildings are partly identified by Buildings’ name
  - $E$ is called a weak entity set
    - Denoted by double rectangle
    - The relationship sets through which $E$ obtains its key are drawn as double diamonds
Weak entity set examples

- Seats in rooms in buildings

- Why must double diamonds be many-one/one-one?
  - With many-many, we would not know which entity provides the key value!

Modeling \( n \)-ary relationships

- An \( n \)-ary relationship set can be replaced by a weak entity set (called a connecting entity set) and \( n \) binary relationship sets

ISA relationships

- Similar to the idea of subclasses in object-oriented programming: subclass = special case, fewer entities, and possibly more properties
  - Represented as a triangle (direction is important)
- Example: Graduate students are students, but they also have offices

Summary of E/R concepts

- Entity sets
  - Keys
  - Weak entity sets
- Relationship sets
  - Attributes of relationships
  - Multiplicity
  - Roles
- Binary versus \( n \)-ary relationships
  - Modeling \( n \)-ary relationships with weak entity sets and binary relationships
  - ISA relationships

Case study 1

- Design a database representing cities, counties, and states
  - For states, record name and capital (city)
  - For counties, record name, area, and location (state)
  - For cities, record name, population, and location (county and state)
- Assume the following:
  - Names of states are unique
  - Names of counties are only unique within a state
  - Names of cities are only unique within a county
  - A city is always located in a single county
  - A county is always located in a single state

Case study 1: first design

- County area information is repeated for every city in the county
  - Redundancy is bad (why?)
- State capital should really be a city
  - Should “reference” entities through explicit relationships
Case study 1: second design

- Technically, nothing in this design could prevent a city in state X from being the capital of another state Y, but oh well…

Case study 2

- Design a database consistent with the following:
  - A station has a unique name and an address, and is either an express station or a local station
  - A train has a unique number and an engineer, and is either an express train or a local train
  - A local train can stop at any station
  - An express train only stops at express stations
  - A train can stop at a station for any number of times during a day
  - Train schedules are the same everyday

Case study 2: first design

- Nothing in this design prevents express trains from stopping at local stations
  - We should capture as many constraints as possible
- A train can stop at a station only once during a day
  - We should not introduce constraints

Case study 2: second design

Is the extra complexity worth it?