Relational Database Design
Part II

CPS 196.3
Introduction to Database Systems

Announcement

- Reminder of the new schedule: 12:50pm-2:05pm Mondays and Wednesdays
- Homework #1 will be assigned on Wednesday
  - Two relational algebra problems have been posted on the newsgroup
- Details of the course project and presentation will be available next Monday

E/R model: review

- Entity sets
  - Keys
  - Weak entity sets
- Relationship sets
  - Attributes on relationships
  - Multiplicity
  - Roles
  - Binary versus N-ary relationships
    - Modeling N-ary relationships with weak entity sets and binary relationships
    - ISA relationships
Database design steps: review

- Understand the real-world domain being modeled
- Specify it using a database design model (e.g., E/R)
- Translate specification to the data model of DBMS (e.g., relational)
- Create DBMS schema

Next: translating an E/R design to a relational schema

Translating entity sets

- An entity set translates directly to a table
  - Attributes → columns
  - Key attributes → key columns

Translating weak entity sets

- Remember the “borrowed” key attributes
- Watch out for attribute name conflicts
Translating relationship sets

- A relationship set translates to a table
  - Keys of connected entity sets → columns
  - Attributes of the relationship set (if any) → columns
  - Multiplicity of the relationship set determines the key of the table

\[ \text{Enroll}(\text{SID}, \text{CID}, \text{grade}) \]

More examples

\[ \text{Students \rightarrow Courses} \]

\[ \text{TA's} \]

\[ \text{Persons \rightarrow Marry} \]

\[ \text{husband, wife} \]

Translating double diamonds

- Recall that a double-diamond relationship set connects a weak entity set to another entity set
- No need to translate because the relationship is implicit in the weak entity set’s translation

\[ \text{RoomInBuilding} \]

\[ \text{is subsumed by} \]

\[ \text{Rooms} \]
Translating subclasses & ISA (approach 1)

- Entity-in-all-superclasses approach
  - An entity is represented in the table for each subclass to which it belongs.
  - A table includes only the attributes attached to the corresponding entity set, plus the inherited key.

- Entity-in-most-specific-class approach
  - An entity is only represented in one table (corresponding to the most specific entity set to which the entity belongs).
  - A table includes the attributes attached to the corresponding entity set, plus all inherited attributes.

- All-entities-in-one-table approach
  - One relation for the root entity set, with all attributes found anywhere in the network of subclasses.
  - Use a special NULL value in columns that are not relevant for a particular entity.
Comparison of three approaches

- Entity-in-all-superclasses
  - Student (SID, name), GradStudent (SID, office)
  - Pro:
  - Con:

- Entity-in-most-specific-class
  - Student (SID, name), GradStudent (SID, name, office)
  - Pro:
  - Con:

- All-entities-in-one-table
  - Student (SID, name, office)
  - Pro:
  - Con:

A complete example

Simplifications and refinements
An alternative design

Train (number, engineer, type)
Station (name, address, type)
TrainStop (train number, station name, time)

- Encode the type of train/station as a column rather than creating subclasses
- Difficult to enforce some constraints
  - Type must be either “local” or “express”
  - Express trains only stop at express stations
  - Fortunately, they can be expressed/declared explicitly as database constraints in SQL
  - Arguably a better design because it is simpler!

Design principles

- KISS
  - Keep It Simple, Stupid!
- Avoid redundancy
  - Redundancy wastes space, complicates updates, and promotes inconsistency
- Use your common sense
  - Warning: Mechanical translation procedures given in this lecture are no substitute for your own judgment