Database Design

- **Goal**: specification of database schema
- **Methodology**:
  - Use **E-R model** to get a high-level graphical view of essential components of enterprise and how they are related
  - Convert E-R diagram to DDL
- **E-R Model**: enterprise is viewed as a set of
  - **Entities**
  - **Relationships** among entities

**Entity Type**

- **Entity type** described by set of attributes
  - **Person**: Id, Name, Address, Hobbies
- **Domain**: possible values of an attribute
  - Value can be a set (in contrast to relational model)
    - (111111, John, 123 Main St, {stamps, coins})
- **Key**: minimum set of attributes that uniquely identifies an entity (candidate key)
- **Entity Schema**: entity type name, attributes (and associated domain), key constraints

**Relationships**

- **Relationship**: relates two or more entities
  - **Person majors in Computer Science**
- **Relationship Type**: set of similar relationships
  - **Student (entity type) related to Department (entity type) by MajorIn (relationship type)**
- **Distinction**:
  - **relation** (relational model) - set of tuples
  - **relationship** (E-R Model) – describes relationship between entities of an enterprise
  - Both entity types and relationship types (E-R model) may be represented as relations (in the relational model)

**Entities**

- **Entity**: an object that is involved in the enterprise
  - Ex: John, CSE305
- **Entity Type**: set of similar objects
  - Ex: students, courses
- **Attribute**: describes one aspect of an entity type
  - Ex: name, maximum enrollment

**Entity Type (con’t)**

- Graphical Representation in E-R diagram:
Attributes and Roles

- **Attribute** of a relationship type describes the relationship
  - e.g., John majors in CS since 2000
    - John and CS are related
    - 2000 describes relationship value of SINCE attribute of MajorsIn relationship type
- **Role** of a relationship type names one of the related entities
  - e.g., John is value of Student role, CS value of Department role of MajorsIn relationship type
  - (John, CS; 2000) describes a relationship

Relationship Type

- Described by set of attributes and roles
  - e.g., MajorsIn: Student, Department, Since
  - Here we have used as the role name (Student) the name of the entity type (Student) of the participant in the relationship, but ...
Single-role Key Constraint
• If, for a particular participant entity type, each entity participates in *at most* one relationship, corresponding role is a key of relationship type
  – E.g., Professor role is unique in WorksIn
• Representation in E-R diagram: arrow

Entity Type Hierarchies
• One entity type might be subtype of another
  – Freshman is a subtype of Student
• A relationship exists between a Freshman entity and the corresponding Student entity
  – e.g., Freshman John is related to Student John
• This relationship is called *IsA*
  – Freshman IsA Student
  – The two entities related by IsA are always descriptions of the same real-world object

IsA

Properties of IsA
• *Inheritance* - Attributes of supertype apply to subtype.
  – E.g., GPA attribute of Student applies to Freshman
  – Subtype *inherits* all attributes of supertype.
  – Key of supertype is key of subtype
• *Transitivity* - Hierarchy of IsA
  – Student is subtype of Person, Freshman is subtype of Student, so Freshman is also a subtype of Student

Advantages of IsA
• Can create a more concise and readable E-R diagram
  – Attributes common to different entity sets need not be repeated
  – They can be grouped in one place as attributes of supertype
  – Attributes of (sibling) subtypes can be different
Constraints on Type Hierarchies

- Might have associated constraints:
  - Covering constraint: Union of subtype entities is equal to set of supertype entities
    - Employee is either a secretary or a technician (or both)
  - Disjointness constraint: Sets of subtype entities are disjoint from one another
    - Freshman, Sophomore, Junior, Senior are disjoint set

Participation Constraint

- If every entity participates in at least one relationship, a participation constraint holds:
  - A participation constraint of entity type \( E \) having role \( \rho \) in relationship type \( R \) states that for \( e \) in \( E \) there is an \( r \) in \( R \) such that \( \rho(r) = e \).
  - e.g., every professor works in at least one department

Participation and Key Constraint

- If every entity participates in exactly one relationship, both a participation and a key constraint hold:
  - e.g., every professor works in exactly one department

Representation of Entity Types in the Relational Model

- An entity type corresponds to a relation
- Relation’s attributes = entity type’s attributes
  - Problem: entity type can have set valued attributes, e.g.,
    - Person: Id, Name, Address, Hobbies
  - Solution: Use several rows to represent a single entity
    - (111111, John, 123 Main St, stamps)
    - (111111, John, 123 Main St, coins)
  - Problems with this solution:
    - Redundancy
    - Key of entity type (Id) not key of relation
    - Hence, the resulting relation must be further transformed (Chapter 6)

Representation of Relationship Types in the Relational Model

- Typically, a relationship becomes a relation in the relational model
- Attributes of the corresponding relation are:
  - Attributes of relationship type
  - For each role, the primary key of the entity type associated with that role
- Example:
  - $2000\text{Courses}$ (CrsCode, SectNo, Enroll)
  - Teaching (CrsCode, SectNo, Id, RoomNo, TAs)

Representation of Relationship Types in the Relational Model

- Candidate key of corresponding table = candidate key of relation
  - Except when there are set valued attributes
  - Example: Teaching (CrsCode, SectNo, Id, RoomNo, TAs)
    - Key of relationship type = (CrsCode, SectNo)
    - Key of relation = (CrsCode, SectNo, TAs)

<table>
<thead>
<tr>
<th>CrsCode</th>
<th>SectNo</th>
<th>Id</th>
<th>RoomNo</th>
<th>TAs</th>
</tr>
</thead>
<tbody>
<tr>
<td>CSE305</td>
<td>1</td>
<td>1234</td>
<td>Hum 22</td>
<td>Joe</td>
</tr>
<tr>
<td>CSE305</td>
<td>1</td>
<td>1234</td>
<td>Hum 22</td>
<td>Mary</td>
</tr>
</tbody>
</table>
Representation in SQL

- Each role of relationship type produces a foreign key in corresponding relation
  - Foreign key references table corresponding to entity type from which role values are drawn

Example 1

CREATE TABLE WorksIn
(Since DATE, -- attribute
Status CHAR (10), -- attribute
ProfId INTEGER, -- role (key of Professor)
DeptId CHAR (4), -- role (key of Department)
PRIMARY KEY (ProfId), -- since a professor works in at most one department
FOREIGN KEY (ProfId) REFERENCES Professor(Id),
FOREIGN KEY (DeptId) REFERENCES Department )

Example 2

CREATE TABLE Sold
(Price INTEGER, -- attribute
Date DATE, -- attribute
ProjId INTEGER, -- role
SupplierId INTEGER, -- role
PartNumber INTEGER, -- role
PRIMARY KEY (ProjId, SupplierId, PartNumber, Date),
FOREIGN KEY (ProjId) REFERENCES Project,
FOREIGN KEY (SupplierId) REFERENCES Supplier (Id),
FOREIGN KEY (PartNumber) REFERENCES Part (Number) )

Representing Type Hierarchies in the Relational Model

- Supertypes and subtypes can be realized as separate relations
  - Need a way of identifying subtype entity with its (unique) related supertype entity
  - Choose a candidate key and make it an attribute of all entity types in hierarchy

Type Hierarchies and the Relational Model

- Translated by adding the primary key of supertype to all subtypes. Plus foreign key from subtypes to the supertype.
Type Hierarchies and the Relational Model

- Redundancy eliminated if IsA is not disjoint
  - For individuals who are both employees and students, Name and DOB are stored only once

<table>
<thead>
<tr>
<th>Person</th>
<th>Employee</th>
<th>Student</th>
</tr>
</thead>
<tbody>
<tr>
<td>SSN</td>
<td>Name</td>
<td>DOB</td>
</tr>
<tr>
<td>1234</td>
<td>Mary</td>
<td>1950</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>SSN</th>
<th>Department</th>
<th>Salary</th>
</tr>
</thead>
<tbody>
<tr>
<td>1234</td>
<td>Accounting</td>
<td>35000</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>SSN</th>
<th>GPA</th>
<th>StartDate</th>
</tr>
</thead>
<tbody>
<tr>
<td>1234</td>
<td>3.5</td>
<td>1997</td>
</tr>
</tbody>
</table>

Other representations are possible in special cases, such as when all subtypes are disjoint

- See in the book

Representing Participation Constraints in the Relational Model

- Inclusion dependency: Every professor works in at least one dep’t.
  - in the relational model: (easy)
    - Professor (Id) references WorksIn (ProfId)
  - in SQL:
    - Simple case: If ProfId is a key in WorksIn (i.e., every professor works in exactly one department) then it is easy:
      - FOREIGN KEY (Id) REFERENCES WorksIn (ProfId)
    - General case – ProfId is not a key in WorksIn, so can’t use foreign key constraint (not so easy):
      - CREATE ASSERTION ProfIdsInDepts
        CHECK ( NOT EXISTS ( SELECT * FROM Professor P WHERE NOT EXISTS ( SELECT * FROM WorksIn W WHERE P.Id = W.ProfId ) ) )

Representing Participation and Key Constraint in SQL

- If both participation and key constraints apply, use foreign key constraint in entity table (but beware: if candidate key in entity table is not primary, presence of nulls violates participation constraint).

CREATE TABLE Professor ( Id INTEGER, PRIMARY KEY (Id), FOREIGN KEY (Id) REFERENCES WorksIn (ProfId) );

Representing Participation and Key Constraint in the Relational Model

- Example: (can’t use foreign key in Professor if ProfId is not a candidate key in WorksIn)

<table>
<thead>
<tr>
<th>Id</th>
<th>ProfId</th>
</tr>
</thead>
<tbody>
<tr>
<td>1123</td>
<td>1123</td>
</tr>
<tr>
<td>4100</td>
<td>4100</td>
</tr>
<tr>
<td>3216</td>
<td>3216</td>
</tr>
</tbody>
</table>

Professor

WorksIn

Department

Example:

<table>
<thead>
<tr>
<th>Id</th>
<th>ProfId</th>
</tr>
</thead>
<tbody>
<tr>
<td>xxxxx</td>
<td>1123</td>
</tr>
<tr>
<td>yyyy</td>
<td>4100</td>
</tr>
<tr>
<td>zzzzzz</td>
<td>3216</td>
</tr>
</tbody>
</table>

Professor

WorksIn

Department
Participation and Key Constraint in Relational Model (again)

- Alternative solution if both key and participation constraints apply: merge the tables representing the entity and relationship sets
  - Since there is a 1-1 and onto relationship between the rows of the entity set and the relationship sets, might as well put all the attributes in one table

Entity or Attribute?

- Sometimes information can be represented as either an entity or an attribute.

Entity or Relationship?

- Sometimes information can be represented as either an entity or an attribute.

(Non-) Equivalence of Diagrams

- Transformations between binary and ternary relationships.

Example

<table>
<thead>
<tr>
<th>Name</th>
<th>Id</th>
<th>DeptId</th>
</tr>
</thead>
<tbody>
<tr>
<td>xxxxxx</td>
<td>1123</td>
<td>CSE</td>
</tr>
<tr>
<td>yyyyyy</td>
<td>4100</td>
<td>ECO</td>
</tr>
<tr>
<td>zzzzzz</td>
<td>3216</td>
<td>AMS</td>
</tr>
</tbody>
</table>

Prof.WorksIn