Database Design

Overview

• What is database?
• Why do we bother?
• Relational database
• Entity-Relationship Modeling
• Mapping class diagrams to tables
What Is Database?

- A tool that stores data, and lets you create, read, update, and delete the data
- Information container
- Various types of database
  - Flat files
  - spreadsheets
  - XML
  - relational databases
    - MySQL, Oracle, DB2, Access

Why Do We Use Database?

- Every non-trivial application uses databases to keep program states, to store, manipulate, and retrieve data
- Database plays a critical role in applications
  - Corrupted data => execution failure
  - Poor data organization => poor performance
- A poorly designed database application allows developers to put in arbitrary data
  - Enter a string “none” as a phone number
Relational Database

- A digital database with a collection of tables
  - Each table contains rows and columns, with a unique key for each row
  - Each entity type described in a database has its own table
    - E.g., “Employee”, “Item”, “Order”
  - Each row represents an instance of the entity
    - E.g., “John Jenny”, “Soap”
  - Each column represents an attribute
    - E.g., “phone number”, “price”

Relational Databases (cont.)

<table>
<thead>
<tr>
<th>User_id</th>
<th>Name</th>
<th>Phone</th>
</tr>
</thead>
<tbody>
<tr>
<td>001</td>
<td>John</td>
<td>123458</td>
</tr>
<tr>
<td>002</td>
<td>Jane</td>
<td>125434</td>
</tr>
</tbody>
</table>

Primary Key/Unique Key: to uniquely specify a tuple in a table

Foreign Key: an attribute in a relational table that matches the primary key column of another table. It can be used to cross-reference tables.
Entity-Relationship Models

- Entity-relationship (ER) diagrams are similar to semantic object modelings (class diagrams)
- It uses different notations
- Focuses more on relations and less on class structure

Entities and Attributes

- An entity is similar to a semantic object
- It includes attributes that describe the object

Employee

- EmployeeId
- FirstName
- LastName
Relationships

• An ER diagram indicates a relationship between entities with a diamond
• Sometimes arrows are added to indicate direction of relationship

Cardinality

• Numbers used to describe relationship quantitatively
Inheritance

- A triangle named “IsA” represents the inheritance relationship

Mapping Class Diagrams to Tables
Mapping Classes to Tables

<table>
<thead>
<tr>
<th>Course</th>
<th>Student</th>
</tr>
</thead>
<tbody>
<tr>
<td>courseId</td>
<td>studentId</td>
</tr>
<tr>
<td>name</td>
<td>firstName</td>
</tr>
<tr>
<td>description</td>
<td>lastName</td>
</tr>
<tr>
<td>...</td>
<td>...</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Courses</th>
<th>Students</th>
</tr>
</thead>
<tbody>
<tr>
<td>CourseId</td>
<td>StudentId</td>
</tr>
<tr>
<td>Name</td>
<td>FirstName</td>
</tr>
<tr>
<td>Description</td>
<td>LastName</td>
</tr>
</tbody>
</table>

Key Points about Tables

- Sometimes you need to explicitly add a primary key to distinguish data in tables
- Database usually provides functionality to automatically increment primary key

<table>
<thead>
<tr>
<th>Sale</th>
<th>Sales</th>
</tr>
</thead>
<tbody>
<tr>
<td>date: Date</td>
<td>SaleId</td>
</tr>
<tr>
<td>isComplete:bool</td>
<td>Date</td>
</tr>
<tr>
<td>...</td>
<td>IsComplete</td>
</tr>
</tbody>
</table>
Mapping Associations

Mapping One-to-One Associations

<table>
<thead>
<tr>
<th>Sale</th>
<th>Receives</th>
<th>Payment</th>
</tr>
</thead>
<tbody>
<tr>
<td>date: Date</td>
<td>1</td>
<td>method: String</td>
</tr>
<tr>
<td>isComplete: bool</td>
<td>1</td>
<td>due: float</td>
</tr>
<tr>
<td></td>
<td></td>
<td>amountPaid: float</td>
</tr>
<tr>
<td></td>
<td></td>
<td>change: float</td>
</tr>
</tbody>
</table>

Sale 1 Receives 1 Payment
What Are the Tables?

<table>
<thead>
<tr>
<th>Sale</th>
<th>Payment</th>
</tr>
</thead>
<tbody>
<tr>
<td>SaleId</td>
<td>PaymentId</td>
</tr>
<tr>
<td>Date</td>
<td>Method</td>
</tr>
<tr>
<td>IsComplete</td>
<td>Due</td>
</tr>
<tr>
<td>PaymentId</td>
<td>AmountPaid</td>
</tr>
<tr>
<td></td>
<td>Change</td>
</tr>
</tbody>
</table>

Mapping One-to-Many Associations

```
SalesLineItem
<table>
<thead>
<tr>
<th>quantity:Integer</th>
<th>Described-by</th>
</tr>
</thead>
<tbody>
<tr>
<td>getSubtotal()</td>
<td>1</td>
</tr>
</tbody>
</table>

ProductSpecification
| desc:String           | Id:DescID    |
| price:Money           | ...          |

SalesLineItem
| N                     |
| Described-by          |

ProductSpecification
| 1                     |
| Described-by          |
```
What Are the Tables?

<table>
<thead>
<tr>
<th>SalesLineItem</th>
<th>ProductSpecification</th>
</tr>
</thead>
<tbody>
<tr>
<td>Id</td>
<td>DescId</td>
</tr>
<tr>
<td>Quantity</td>
<td>Desc</td>
</tr>
<tr>
<td>DescId</td>
<td>Price</td>
</tr>
</tbody>
</table>

Mapping Many-to-One Associations

<table>
<thead>
<tr>
<th>Sale</th>
<th>Contains</th>
<th>SalesLineItem</th>
</tr>
</thead>
<tbody>
<tr>
<td>...</td>
<td>1</td>
<td>...</td>
</tr>
<tr>
<td>...</td>
<td>1..*</td>
<td>quantity:Integer</td>
</tr>
</tbody>
</table>

Sale 1 contains 1..N SalesLineItem
What Are the Tables?

<table>
<thead>
<tr>
<th>Sale</th>
<th>SalesLineItem</th>
</tr>
</thead>
<tbody>
<tr>
<td>SaleId</td>
<td>Id</td>
</tr>
<tr>
<td>Date</td>
<td>SaleId</td>
</tr>
<tr>
<td>IsComplete</td>
<td>Quantity</td>
</tr>
</tbody>
</table>

Mapping Many-to-Many Associations to Table

- **Course**
  - courseId
  - name
  - description

- **Student**
  - studentId
  - firstName
  - lastName

- **Takes**
  - 1..N

- **Contains**
  - 1..*

- **Student**
  - 1..N

- **Course**
  - 1..N
What Are the Tables?

Courses
- CourseId
- Name
- Description

StudentCourses
- StudentId
- CourseId

Students
- StudentId
- FirstName
- LastName

Multiple Many-to-Many

- What if we want to know students’ enrollment over time for each year and semester
  - E.g., to distinguish students enrolled different time?)

Course
courseId
name
description
...

Session
year
semester
instructor

Student
studentId
firstName
lastName
...

Contains 1 1:*
Multiple-Object Associations

• Definition
  – Many different kinds of objects are collectively associated with each other

• Case study
  – Making a movie requires a whole horde of people including a director, a bunch of actors, and a huge number of crew members
Class Diagram for Movie-making

- **Movie**
  - movieId
  - title
  - description

- **Director**
  - directorId
  - name

- **Crewmember**
  - crewmemberId
  - name

- **Actor**
  - actorId
  - name

ER Diagram for Movie Making

- **Director**
  - 1

- **Actor**
  - 1..N

- **Crewmember**
  - 1..N

- **Create**
  - 1

- **Movie**
Consider the Relationship as a Combination of Simpler Relationships

```
Actor
  1:1 Helps create
      1:1 Helps create
      1:1 Helps create
      1 Helps create
        1
             Movie

Director
  1 Helps create

Crewmember
  1:1 Helps create
```

---

**Actors**
- ActorId
- Name

**MovieActors**
- MovieId
- ActorId

**Movie**
- MovieId
- Title
- Description
- DirectorId

**Crewmembers**
- CrewmemberId
- Name

**MovieCrewmembers**
- MovieId
- CrewmemberId

**Directors**
- DirectorId
- Name
Repeated Attribute Associations

- Some entities have multiple attributes that represent either the same kind of data or very similar kind of data
  - Some people may have multiple phone numbers for different purposes
  - Some people may have only one number to serve all purposes

How to design the tables to avoid repetition or sparse data?

```
Person
- personId
- firstName
- lastName
- workPhone
- cellPhone
- homePhone
- dayPhone
- nightPhone
- ...
```
Reflexive Associations

• An object refers to an object of the same class
  – One-to-One reflexive association
  – One-to-Many reflexive association
  – Many-to-Many reflexive association
One-to-One Reflexive Association

Person

<table>
<thead>
<tr>
<th>personId</th>
<th>firstName</th>
<th>lastName</th>
<th>spouse</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>...</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Person

<table>
<thead>
<tr>
<th>Is married to</th>
</tr>
</thead>
</table>

Persons

<table>
<thead>
<tr>
<th>PersonId</th>
<th>FirstName</th>
<th>LastName</th>
<th>SpouseId</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

One-to-Many Reflexive Association

Employee

<table>
<thead>
<tr>
<th>personId</th>
<th>firstName</th>
<th>lastName</th>
<th>manager</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>...</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Employee

<table>
<thead>
<tr>
<th>Is managed by</th>
</tr>
</thead>
</table>

Employees

<table>
<thead>
<tr>
<th>PersonId</th>
<th>FirstName</th>
<th>LastName</th>
<th>ManagerId</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>0..1</td>
</tr>
<tr>
<td>0..1</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
Many-to-Many Reflexive Association

Node

| nodeId | X | Y | … |

Node

1..N

Connects to

Distance

Nodes

| NodeId |

| X |

| Y |

Links

| FromNodeId |

| ToNodeId |

| Distance |

Inheritance

Person

| personId |

| firstName |

| lastName |

| email |

| address |

| … |

Customer

| customerId |

| address |

| … |

Employee

| employeeId |

| ssn |

| specialities |

| … |

Vendor

| vendorId |

| companyName |

| notes |

| … |
ER-Diagram

Person

IsA

Customer
Employee
Employee

Tables

Persons
- PersonId
- FirstName
- LastName
- Email
- Address

Customers
- customerId
- personId

Employees
- employeeId
- personId
- ssn
- specialities

Vendors
- vendorId
- personId
- CompanyName
- Notes