SWE 621:
Software Modeling and Architectural Design

Lecture Notes on Software Design

Lecture 13 - Relational Database Design

Hassan Gomaa
Dept of Computer Science
George Mason University
Fairfax, VA


Copyright © 2011 Hassan Gomaa
All rights reserved. No part of this document may be reproduced in any form or by any means, without the prior written permission of the author.
This electronic course material may not be distributed by e-mail or posted on any other World Wide Web site without the prior written permission of the author.
Steps in Using COMET/UML

1. Develop Software Requirements Model
2. Develop Software Analysis Model
3. **Develop Software Design Model**
   - Design Overall Software Architecture (Chapter 12, 13)
   - Design Distributed Component-based Subsystems (Chapter 12-13, 15)
   - Structure Subsystems into Concurrent Tasks (Chapter 18)
   - Design Information Hiding Classes (Chapter 14)
   - **Design relational database (Chapter 15)**

Relational Database Design

- **Objective**: Map static model to relational database
- Each entity class from static model that needs to be stored in relational database
  - Entity class maps to one (or more) relation(s) (table)
  - Attributes mapped to columns of table
  - Each object instance maps to a row of table
- **Relational Database Design**
  - Primary keys
  - Foreign keys for associations
  - Association classes
  - Aggregation/Composition Hierarchy
  - Generalization/Specialization hierarchy
Entity Classes and Relational Tables

Account entity class

<table>
<thead>
<tr>
<th>Account</th>
</tr>
</thead>
<tbody>
<tr>
<td>accountNumber: Integer</td>
</tr>
<tr>
<td>balance: Real</td>
</tr>
</tbody>
</table>

Account relational table

<table>
<thead>
<tr>
<th>Account Number</th>
<th>Balance</th>
</tr>
</thead>
<tbody>
<tr>
<td>1234</td>
<td>398.07</td>
</tr>
<tr>
<td>5678</td>
<td>439.72</td>
</tr>
<tr>
<td>7287</td>
<td>551.65</td>
</tr>
</tbody>
</table>

Primary Keys

- Each relation must have a primary key
- Primary Key
  - Combination of one or more attributes
  - Uniquely locates a row in relation
    - E.g., Account Number is primary key of Account relation
    - Account (Account number, Balance)
      - (underline = primary key)
Relational Database Design

Foreign Keys

- Associations in relational databases
  - Many-to-many association in static model
    - Maps to a relation
  - One-to-one and one-to-many associations
    - Use Foreign keys
- Foreign key
  - Primary key of one table that is embedded in another table
  - Represents mapping of association between relations into a table
  - Allows navigation between tables

One-to-one or Zero-or-one Association

- One-to-one association maps to
  - Foreign key in one of relations
- Zero-or-one association maps to
  - Foreign key in optional relation
- E.g., Customer Owns Debit Card
- Static model (Figure 15.16)
  - Customer (Customer Name, Customer Id, Customer Address)
  - Debit Card (Card Id, PIN, Expiration date, Status, Limit, Total)
One-to-one or Zero-or-one Association

- Relational Database Design
  - Customer Id chosen as primary key of Customer
    - Customer (Customer Name, Customer Id, Customer Address)
  - Card id chosen as primary key of Debit Card relation
- Customer Id chosen as foreign key in Debit Card
- Represents association between Customer and Debit Card relations
  - Debit Card (Card Id, PIN, Expiration date, Status, Customer Id)
  - (underline = primary key, italic = foreign key)
One-to-Many Association

- One-to-many association maps to
  - Foreign key in “many” relation
  - E.g., Customer Owns Account
- Static Model (Figure 15.15)
  - Customer (Customer Name, Customer Id, Customer Address)
  - Account (Account number, Balance)
- Relational Database Design
  - Primary key of “one” relation (Customer) is chosen as foreign key in “many” relation (Account)

Figure 15.15 Example of one-to-many association
One-to-Many Association

- Relational Database Design
  - Customer Id is chosen as primary key of Customer relation
    - Customer (Customer Name, Customer Id, Customer Address)
  - Account Number is chosen as primary key of Account relation
  - Customer Id is foreign key in Account relation
    - Account (Account Number, Balance, Customer Id)

Static Model
Association Class

- Class models association between two or more classes
  - Usually for many-to-many associations
- Association class is mapped to associative relation
- Associative relation
  - Relation to represent association between two or more relations
  - Primary key of associative relation
    - Concatenated key
    - Formed from primary key of each relation that participates in association
Static Model
Association Class

- E.g., Hours association class
  - Represents association between Project and Employee classes
  - Mapped to Associative relation Hours
- Static Model (Figure 15.17)
  - Project (Project id, Project name)
  - Employee (Employee id, Employee name, Employee address)
  - Hours (Hours Worked)
    - Hours Worked is attribute of association

Figure 15.17 Example of association class
Relational Database Design
Associative Relation

• Relational Database Design
  – Project (Project id, Project name)
  – Employee (Employee id, Employee name, Employee address)
• Project id and Employee id
  – Form concatenated primary key of Hours relation
  – Also foreign keys
    • Hours (Project id, Employee id, Hours worked)

Static Model
Aggregation/Composition Hierarchy

• Whole/part relationship
  – Aggregate/Composite (whole) class is mapped to relation
  – Each part class is mapped to relation
  – Primary key of composite/aggregate relation
    • All of primary key of component relation
      – 1-1 aggregation
    • Part of primary key of component relation
      – 1-n aggregation
    • Foreign key
      – If not needed to uniquely identify component relation
Relational Database Design
Aggregation/Composition Hierarchy

- E.g., Static Model (Figure 15.18)
  - Department IS PART OF College
  - Admin Office IS PART OF College
  - College (College name)
  - Admin Office (Location)
  - Department (Department name, Location)

- Relational Database Design
  - Primary key of aggregate relation = College name
  - College (College name)
  - Admin Office (College name, Location)
  - Department (Department name, College name, Location)
Static Model
Generalization / Specialization Hierarchy

- Three alternative mappings from Generalization / Specialization Hierarchy to relational database
  - Superclass & subclasses mapped to relations
  - Subclasses only mapped to relations
  - Superclass only mapped to relation

Relational Database Design
Generalization / Specialization Hierarchy

- Superclass & subclasses mapped to relations
  - Superclass mapped to table
    - Discriminator is attribute of superclass table
  - Each subclass mapped to table
  - Shared id for primary key
    - Same primary key in superclass and subclass tables
  - Clean and extensible
  - However, superclass / subclass navigation may be slow
Figure 15.19 Generalization / specialization

Relational Database Design
Generalization / Specialization Hierarchy

- Superclass & subclasses mapped to relations
- E.g.: Account Generalization / Specialization Hierarchy
- Static Model (Figure 15.19)
  - Superclass: Account (Account number, Balance)
  - Subclass: Checking Account (Last Deposit Amount)
  - Subclass: Savings Account (Interest)
- Relational Database Design
  - Account (Account number, Account Type, Balance)
  - Checking Account (Account Number, Last Deposit Amount)
  - Savings Account (Account Number, Interest)
Relational Database Design
Generalization / Specialization Hierarchy

- Subclasses only mapped to relations
  - Map each subclass to relation
  - No superclass relation
  - Superclass attributes replicated for each subclass table

- Can use if
  - Subclass has many attributes
  - Superclass has few attributes
  - Application knows what subclass to search

Relational Database Design
Generalization / Specialization Hierarchy

- Subclasses only mapped to relations
- Example of Account Generalization / Specialization Hierarchy
- Static Model (Figure 15.19)
  - Superclass: Account (Account number, Balance)
  - Subclass: Checking Account (Last Deposit Amount)
  - Subclass: Savings Account (Interest)

- Relational Database Design
  - Checking Account ([Account Number, Balance, Last Deposit Amount])
  - Savings Account ([Account Number, Balance, Interest])
Relational Database Design
Generalization / Specialization Hierarchy

• Superclass only mapped to relation
• All subclass attributes brought up to superclass table
  – Discriminator is attribute of superclass table
  – Each record in superclass table uses attributes relevant to one subclass
  – Other attribute values are null
• Can use if
  – Superclass has many attributes
  – Subclass has few attributes
  – Only two or three subclasses

Example of Account Generalization / Specialization Hierarchy

Static Model (Figure 15.19)
  – Superclass: Account (Account number, Balance)
  – Subclass: Checking Account (Last Deposit Amount)
  – Subclass: Savings Account (Interest)

Relational Database Design
  – Account (Account Number, Account Type, Balance, Last Deposit Amount, Interest)
Figure 21.4 Conceptual static model for Banking System - entity classes

Example of Relational Database Design

- Banking System static mode (Figure 21.4)
- Bank Information (underline = primary key, italic = foreign key):

Bank (bankName, Bank Address, bankId)
ATM Info (bankId, ATMId, ATM Location, ATM Address)
Customer (customerName, customerId, customerAddress)
Debit Card (cardId, PIN, startDate, expirationDate, status, limit, total, customerId)
Checking Account (accountNumber, accountType, balance, lastDepositAmount)
Savings Account (accountNumber, accountType, balance, interest)
Card Account (cardId, accountNumber)
Customer Account (customerId, accountNumber)

Assumption: Account type is determined from account number
Steps in Using COMET/UML

1. Develop Software Requirements Model
2. Develop Software Analysis Model
3. Develop Software Design Model
   - Design Overall Software Architecture (Chapter 12, 13)
   - Design Distributed Component-based Subsystems (Chapter 12-13,15)
   - Structure Subsystems into Concurrent Tasks (Chapter 18)
   - Design Information Hiding Classes (Chapter 14)
   - Design relational database (Chapter 15)

Figure 6.1 COMET object-oriented software life cycle model