

Data Management Layer Design

Slide 1

Objectives

- Become familiar with several object-persistence formats.
- Be able to map problem domain objects to different object-persistence formats.
- Be able to design the data access and manipulation classes.

Slide 2

Data Management Layer

- Choose object-persistence format to support the system
 - Problem domain objects drive object storage design
- Design of Data Storage
 - Must optimize processing efficiency
- Data access and manipulation
 - Separate problem domain classes from storage format
 - Handle all communication with the database

Slide 3

Object Persistence Formats

- Files (Sequential and Random)
- Relational databases
- Object-relational databases
- Object-oriented databases

Slide 4

Customer Order File

| Order Number | Date | Card ID | Last Name | First Name | Amount | Tax | Total | Price Customer | Payment Type |
|--------------|----------|---------|-----------|------------|----------|--------|----------|----------------|--------------|
| 234 | 11/23/00 | 2242 | Dalbary | Ann | \$ 90.00 | \$5.85 | \$ 95.85 | Y | MC |
| 235 | 11/23/00 | 9500 | Chen | April | \$ 12.00 | \$0.60 | \$ 12.60 | Y | VISA |
| 236 | 11/23/00 | 1556 | Frackler | Chris | \$ 30.00 | \$2.50 | \$ 32.50 | N | VISA |
| 237 | 11/23/00 | 2242 | Dalbary | Ann | \$ 75.00 | \$4.68 | \$ 79.68 | Y | AMEX |
| 238 | 11/23/00 | 2242 | Dalbary | Ann | \$ 40.00 | \$3.90 | \$ 43.90 | Y | MC |
| 239 | 11/23/00 | 1035 | Black | John | \$ 90.00 | \$4.50 | \$ 94.50 | Y | AMEX |
| 240 | 11/23/00 | 9501 | Kaplan | Bruce | \$ 30.00 | \$2.50 | \$ 32.50 | N | VISA |
| 241 | 11/23/00 | 1133 | Williams | Mary | \$100.00 | \$9.00 | \$109.00 | N | MC |
| 242 | 11/24/00 | 9500 | Chen | April | \$ 60.00 | \$3.00 | \$ 63.00 | Y | VISA |
| 243 | 11/24/00 | 4234 | Bailey | Ryan | \$ 90.00 | \$4.50 | \$ 94.50 | Y | VISA |
| 244 | 11/24/00 | 9500 | Chen | April | \$ 24.00 | \$1.20 | \$ 25.20 | Y | VISA |
| 245 | 11/24/00 | 2242 | Dalbary | Ann | \$ 12.00 | \$0.78 | \$ 12.78 | Y | AMEX |
| 246 | 11/24/00 | 4234 | Bailey | Ryan | \$ 20.00 | \$1.00 | \$ 21.00 | Y | MC |
| 247 | 11/24/00 | 2243 | Jones | Chris | \$ 30.00 | \$2.50 | \$ 32.50 | N | VISA |
| 248 | 11/24/00 | 4234 | Bailey | Ryan | \$ 12.00 | \$0.60 | \$ 12.60 | Y | AMEX |
| 249 | 11/24/00 | 5927 | Lee | Clara | \$ 50.00 | \$2.50 | \$ 52.50 | N | AMEX |

Slide 5

Application File Types

- Master Files
- Look-up files
- Transaction files
- Audit file
- History file

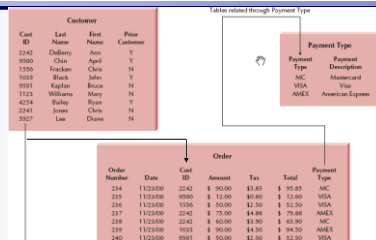
Slide 6

Relational Databases

- Collection of tables
 - Comprised of fields that define entities
 - Primary key has unique values in each row of a table
- Tables related to each other
 - Primary key field of a table is a field of another table and called a foreign key
 - Relationship established by a foreign key of one table connecting to the primary key of another table

Slide 7

Customer Order Database



Slide 8

Database Management System (DBMS)

- Software that creates and manipulates a database
- RDBMS is a DBMS for a relational database
- RDBMS usually support Referential Integrity

Slide 9

Referential Integrity

- the idea of ensuring that values linking the tables together through the primary and foreign keys are valid and correctly synchronized.

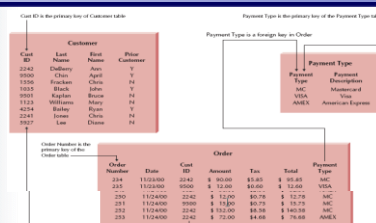
Slide 10

Referential Integrity Example

- Cust. ID is a primary key for the customer table
- Cust. ID is a foreign key for the order table
- A violation of referential integrity would happen if an order was entered in the order table for a Cust. ID that had not been entered into the customer table first
- An RDBMS prevents such a record from being entered

Slide 11

Example of Referential Integrity



Slide 12

Referential Integrity:

- All payment type values in Order must exist in the Payment Type table.
- All Cust ID values in Order must exist in the Customer table.

Structured Query Language (SQL)

- Standard language for accessing data in tables
- SQL Commands
 - Create, edit, and delete tables
 - Add, edit, and delete data
 - Display data from one or more related tables
 - Display data computed from data in one or more related tables

Slide 13

Object-Relational Databases

- Relational database management systems with extensions that handle object storage in the relational table structure
- This is done by user defined types
 - Example: Create a map data type

Slide 14

Vendors Support ORDBMS

- SQL designed for simple data types
- Vendors extend SQL to handle user data types in Object Relational Databases
- Usually they don't support most object oriented features e.g. inheritance

Slide 15

Object-Oriented Databases (OODBMS)

- Add persistence extensions to an object-oriented programming language
- Create an entirely separate database management system

Slide 16

OODBMS Terminology

- Extent is a collection of objects
 - Set of instances associated with a particular class (RDBMS table)
 - Each instance of a class has a unique identifier called an object ID
 - Referential integrity still important
 - Supports a form of inheritance

Slide 17

OODBMS Support

- Allow repeating groups or multivalued attributes
- Supports multimedia or other complex data applications
 - CAD/CAM
 - Financial services
 - Geographic information systems
 - Health care

Slide 18

Commercial OODBMSs

- **GemStone** from **Gemstone Systems Inc.**,
- **Objectivity/DB** from **Objectivity Inc.**,
- **ObjectStore** from **Progress Software Corp.**,
- **Ontos** from **Ontos Inc.**,
- **FastObjects** from **Poet Software Corp.**,
- **Jasmine** from **Computer Associates/Fujitsu**,
- **Versant** from **Versant Corp.**

Slide 19

© Pearson Education Limited 1995, 2005

Major Strengths & Weaknesses

- Files
 - Very efficient for given task
 - Manipulation done by OOPL
 - Redundant data usually results
- RDBMS
 - Proven commercial technology
 - Handle diverse data
 - No support for object orientation

Slide 20

More Strengths and Weaknesses

- ORDBMS
 - Inherit RDBMS strengths
 - Support complex data types
 - Limited support for object-orientation (vendor dependent)
- OODBMS
 - Support complex data types
 - Support object-orientation directly
 - Still maturing

Slide 21

Mapping Objects to Object-Persistence Formats

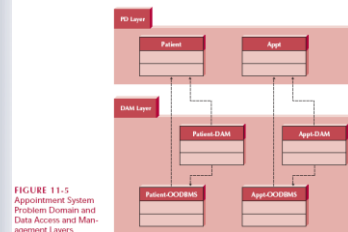


FIGURE 11-5 Appointment System Problem Domain and Data Access and Management Layers

Slide 22

Multiple Inheritance Effect Rules

- Results when you have more than one super class
- Rule 1a. Add an attribute(s) to the OODBMS class to represent the additional super class
- Rule 1b. Flatten the inheritance hierarchy and remove additional super classes from the design

Slide 23

Mapping to Single I-B OODBMS

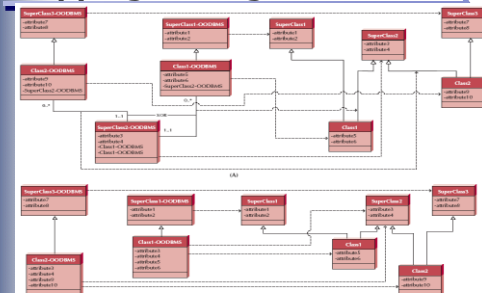


FIGURE 11-6 Mapping Problem Domain Objects to Single Inheritance-Based OODBMS

Using Rule 1a

- Added an attribute to Class1-OODBMS that represents an association with Super-Class2-OODBMS,
- Added attributes to Class2-OODBMS that represents an association with Super-Class2-OODBMS,
- Added a pair of attributes to SuperClass2-OODBMS that represents an association with Class1-OODBMS and Class2-OODBMS, for completeness sake, and
- Added associations between Class2-OODBMS and SuperClass2-OODBMS and Class1-OODBMS and SuperClass2-OODBMS that have the correct multiplicities and the XOR constraint explicitly shown.

Slide 25

Mapping PDO to ORDBMS

- Rule 1: Map all concrete problem domain classes to the ORDBMS tables. Also, if an abstract problem domain class has multiple direct subclasses, map the abstract class to an ORDBMS table.
- Rule 2: Map single valued attributes to columns of the ORDBMS tables.
- Rule 3: Map methods and derived attributes to stored procedures or to program modules.
- Rule 4: Map single-valued aggregation and association relationships to a column that can store an Object ID. Do this for both sides of the relationship.
- Rule 5: Map multi-valued attributes to a column that can contain a set of values.
- Rule 6: Map repeating groups of attributes to a new table and create a one-to-many association from the original table to the new one.
- Rule 7: Map multi-valued aggregation and association relationships to a column that can store a set of Object IDs. Do this for both sides of the relationship.
- Rule 8: For aggregation and association relationships of mixed type (one-to-many or many-to-one), on the single-valued side (1, 1 or 0, 1) of the relationship, add a column that can store a set of Object IDs. The values contained in this new column will be the Object IDs from the instances of the class on the multi-valued side. On the multi-valued side (1..* or 0..*), add a column that can store a single Object ID that will contain the value of the instance of the class on the single-valued side.
- For generalization/inheritance relationships:
- Rule 9a: Add a column(s) to the table(s) that represents the subclass(es) that will contain an Object ID of the instance stored in the table that represents the superclass. This is similar in concept to a foreign key in an RDBMS. The multiplicity of this new association from the subclass to the "superclass" should be 1..1. Add a column(s) to the table that represents the superclass that will contain an Object ID of the instance stored in the table that represents the subclass(es). If the superclass are concrete, that is, they can be instantiated themselves, then the multiplicity from the superclass to the subclass is 0..1 otherwise, it is 1..1. Furthermore, an exclusive-or (XOR) constraint must be added between the associations. Do this for each superclass.
- OR
- Rule 9b: Flatten the inheritance hierarchy by copying the superclass attributes down to all of the subclasses and remove the superclass from the design.¹⁰

FIGURE 11-7 Mapping Problem Domain Objects to ORDBMS Schema

Slide 26

Mapping Table to PD Classes

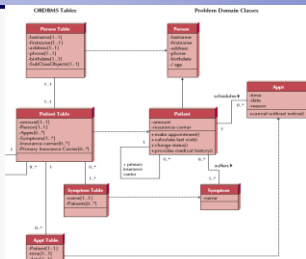


FIGURE 11-8 Mapping Problem Domain Objects to ORDBMS Schema Example

Slide 27

Mapping PD Objects to RDBMS Schema

- Rule 1:** Map all concrete problem domain classes to the RDBMS tables.
 - Rule 2:** Map single valued attributes to columns of the tables.
 - Rule 3:** Map methods to stored procedures or to program modules.
 - Rule 4:** Map single-valued aggregation and association relationships to a column that can store the key of the related table
 - Rule 5:** Map multi-valued attributes and repeating groups to new tables and create a one-to-many association from the original table to the new ones.
 - Rule 6:** Map multi-valued aggregation and association relationships to a new associative table that relates the two original tables together. Copy the primary key from both original tables to the new associative table
 - Rule 7:** For aggregation and association relationships of mixed type, copy the primary key from the single-valued side (1, 1 or 0, 1) of the relationship to a new column in the table on the multi-valued side (1..* or 0..*) of the relationship that can store the key of the related table
 - Rule 8a:** Ensure that the primary key of the subclass instance is the same as the primary key of the superclass..
- OR
- Rule 8b:** Flatten the inheritance

Slide 28

Mapping RDBMS Tables to Problem Domain Classes

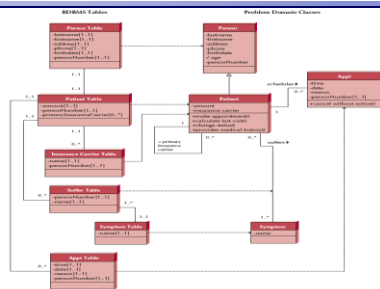


FIGURE 11-10 Mapping Problem Domain Objects to RDBMS Schema Example

Slide 29

Optimize RDBMS Object Storage

- No redundant data
 - Wastes space
 - Allow more room for error
- Few null values in tables
 - Difficult to interpret

Slide 30

Example of Non-normalized Data

Sample Records:

| Order Number | Date | City | Last Name | First Name | State | Zip | Prod. 1 Number | Prod. 1 Desc. | Prod. 1 Price | Prod. 1 Qty. | Prod. 2 Number | Prod. 2 Desc. | Prod. 2 Price | Prod. 2 Qty. |
|--------------|---------|------|-----------|------------|-------|------|----------------|----------------|---------------|--------------|----------------|----------------|---------------|--------------|
| 129 | 1/12/90 | Rock | John | MOE | 0107 | 222 | Shrimp Pie | \$45.00 | 2 | | | | | |
| 208 | 1/12/90 | Rock | John | MOE | 0105 | 444 | Wine Cab/Pk | \$20.00 | 1 | | | | | |
| 271 | 1/12/90 | Rock | John | MOE | 0105 | 222 | Butter-Spense | \$12.00 | 2 | | | | | |
| 241 | 1/12/90 | 112 | Williams | Harry | CA | 0106 | 444 | Wine Cab/Pk | \$20.00 | 2 | | | | |
| 252 | 1/12/90 | 112 | Williams | Harry | CA | 0108 | 222 | Butter-Spense | \$12.00 | 2 | | | | |
| 207 | 1/12/90 | 112 | Williams | Harry | CA | 0106 | 222 | Butter-Spense | \$12.00 | 2 | | | | |
| 206 | 1/12/90 | 112 | Williams | Harry | CA | 0108 | 015 | Cheddar Pie | \$45.00 | 2 | | | | |
| 134 | 1/12/90 | 2142 | Judithy | Ann | DC | 0105 | 015 | Cheddar Pie | \$45.00 | 1 | | | | |
| 217 | 1/12/90 | 2142 | Judithy | Ann | DC | 0105 | 111 | Wine Cab/Pk | \$20.00 | 1 | 444 | Wine Cab/Pk | \$20.00 | 1 |
| 205 | 1/12/90 | 2142 | Judithy | Ann | DC | 0105 | 222 | Butter-Spense | \$12.00 | 1 | | | | |
| 210 | 1/12/90 | 2142 | Judithy | Ann | DC | 0105 | 222 | Butter-Spense | \$12.00 | 1 | 444 | Wine Cab/Pk | \$20.00 | 2 |
| 212 | 1/12/90 | 2142 | Judithy | Ann | DC | 0105 | 222 | Butter-Spense | \$12.00 | 1 | 444 | Wine Cab/Pk | \$20.00 | 1 |
| 223 | 1/12/90 | 2142 | Judithy | Ann | DC | 0105 | 222 | Butter-Spense | \$12.00 | 1 | | | | |
| 201 | 1/12/90 | 2142 | Judithy | Ann | DC | 0105 | 333 | Jams & Jellies | \$20.00 | 2 | | | | |
| 213 | 1/12/90 | 2142 | Judithy | Ann | DC | 0105 | 015 | Cheddar Pie | \$45.00 | 2 | | | | |
| 204 | 1/12/90 | 2142 | Judithy | Ann | DC | 0105 | 015 | Cheddar Pie | \$45.00 | 2 | | | | |
| 249 | 1/12/90 | 2142 | Judithy | Ann | DC | 0105 | 222 | Butter-Spense | \$12.00 | 1 | 333 | Jams & Jellies | \$20.00 | 2 |
| 211 | 1/12/90 | 2142 | Judithy | Ann | DC | 0105 | 222 | Butter-Spense | \$12.00 | 1 | | | | |
| 242 | 1/12/90 | 3100 | Chm | April | KS | 0105 | 333 | Jams & Jellies | \$20.00 | 1 | | | | |
| 244 | 1/12/90 | 3100 | Chm | April | KS | 0105 | 222 | Butter-Spense | \$12.00 | 2 | | | | |
| 231 | 1/12/90 | 3100 | Chm | April | KS | 0105 | 111 | Wine Cab/Pk | \$20.00 | 2 | | | | |

FIGURE 11-11 Optimizing Storage

Normalization

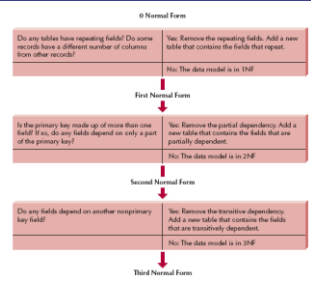
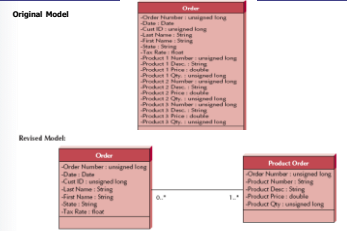


FIGURE 11-12 The Steps of Normalization

Normalization Example



3NF Normalized Model

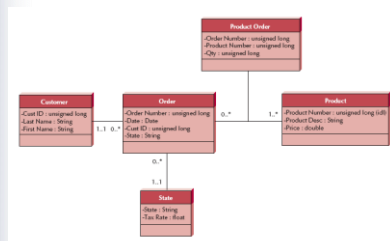


FIGURE 11-15 3NF Normalized Model

Problems with RDBMS

- To access data in multiple tables, the tables must be joined
- This can result in many database operations and lead to huge tables and slow processing

Speeding up access

- Denormalization – Adds data from one table to another in order to speed processing and eliminate a join operation
- Example: Add customer last name to order table to avoid joining order to customer to get just last name

Data Sizing Example

| Field | Average Size |
|------------------------|--------------|
| Order Number | 8 |
| Date | 7 |
| Card ID | 4 |
| Last Name | 13 |
| First Name | 9 |
| State | 2 |
| Amount | 4 |
| Tax Rate | 2 |
| Record Size | 49 |
| Overhead | 30% |
| Total Record Size | 63.7 |
| Initial Table Size | 50,000 |
| Initial Table Volume | 3,185,000 |
| Growth Rate/Month | 1,000 |
| Table Volume @ 3 years | 5,478,200 |

FIGURE 11-20
Calculating Volumetrics

Slide 43

DESIGNING DATA ACCESS AND MANIPULATION CLASSES

- Design data access and manipulation classes
- Prevent data management functionality from creeping into the problem domain classes

Slide 44

Mapping PD Objects

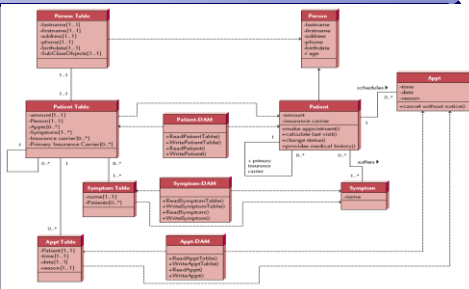


FIGURE 11-21 Mapping Problem Domain Objects to ORDBMS Using Data Access and Manipulation Classes

Slide 45

Summary

- Choose an object-persistent format
 - Files (sequential or Random Access)
 - Databases (RDBMS, ORDBMS, OODBMS)
- Map problem domain objects to Data
- Optimizing object storage
 - Normalization
 - Denormalization, clustering, Indexes
- Design Data Access and Manipulation Classes

Slide 46