Overview

Chapters 8 and 9 discussed the design of the application software in the business layer, the middle tier of a three-tier system architecture.

Chapter 10 is concerned with the interface between the application layer and the storage layer.

Overview (continued)

A data base in the storage layer implements the persistence needed by objects in the business layer.

These objects communicate with the data base via an interface which accesses the operations of the database management system (DBMS).
The Singleton pattern produces a single, globally visible object to act as this interface.
The design of the data base is derived from the objects, attributes, and associations shown in the design class diagram.

The Three-Tier System Architecture

Persistence

Many instances of concepts in the real world have long lives inherently.
Software objects operate in the volatile memory of a computer. Making them persistent requires storing them in memory which lasts even when power is turned off.
Show Off Time

• Who here is a good data designer?
  – Willing to answer a few questions
• Let’s look at a very simple example
  – What are the persistent data needs?

Data Bases and Database Management Systems

A data base is a systematically partitioned, reusable, integrated collection of data which can be shared by many individual users as well as by multiple applications.

Users and application software access a data base through an interface – a database management system (DBMS).

Data Bases and Database Management Systems (continued)

Data bases provide:
• Logical independence of the data and
• Physical independence of the data

Data bases enforce:
• Integrity constraints and
• Security
Database Structures

Ideally, objects should be stored in object data bases; that technology, however, is still evolving.

In practice, most current data bases are relational – organized as tables, or relations.

Database Operations on Objects

Every database management system must provide these operations on objects:

1. **Create**: Establish a new object.
2. **Remove**: Delete an existing object.
3. **Store**: Modify the value of one or more attributes of an existing object.
4. **Load**: Read the attribute data for one object

Database Operations on Associations

Every database management system must provide these operations on associations:

1. **Create**: Establish a new link (an instance of an association).
2. **Remove**: Delete an existing link.
System Start-up and Shutdown

At system start-up, new objects must be instantiated, and a load operation must read each object’s attribute values from the database.

To prevent loss of data during an unexpected system shutdown, it is desirable to use a create, delete, or store operation whenever an object is created or destroyed or whenever an attribute value is modified.

Global Visibility

Most objects in the business layer need visibility to the database and need to use the operations of the DBMS.

The Singleton pattern provides this global visibility.

Figure 10.3 shows the Singleton object added to the design class diagram. (Note that the dependency arrows show indirect visibility to this object.)

The Singleton Pattern

FIGURE 10.3
Database Requirements – An Example

**FIGURE 10.4**

<table>
<thead>
<tr>
<th>OBJECT</th>
<th>ATTRIBUTES</th>
</tr>
</thead>
<tbody>
<tr>
<td>Course</td>
<td>number, title, units</td>
</tr>
<tr>
<td>Department</td>
<td>name, code</td>
</tr>
<tr>
<td>Department Class Schedule</td>
<td>time, date</td>
</tr>
<tr>
<td>Professor</td>
<td>name</td>
</tr>
<tr>
<td>Section</td>
<td>course, university, title, units, sectionnumber, credit, prerequisites, meetingtime, meetingday, roomnumber, professoroffices</td>
</tr>
<tr>
<td>Student</td>
<td>idnumber, first, last, address, phone</td>
</tr>
</tbody>
</table>

Database Requirements – An Example (continued)

**FIGURE 10.5**

<table>
<thead>
<tr>
<th>OBJECT</th>
<th>ASSOCIATIONS</th>
</tr>
</thead>
<tbody>
<tr>
<td>Course</td>
<td>None</td>
</tr>
<tr>
<td>Department</td>
<td>Department Class Schedule, Course</td>
</tr>
<tr>
<td>Department Class Schedule</td>
<td>Section</td>
</tr>
<tr>
<td>Professor</td>
<td>None</td>
</tr>
<tr>
<td>Section</td>
<td>Professor, Student</td>
</tr>
<tr>
<td>Student</td>
<td>Section</td>
</tr>
</tbody>
</table>

Database Requirements – An Example (continued)

**FIGURE 10.6**

<table>
<thead>
<tr>
<th>OBJECT / ASSOCIATION</th>
<th>LOAD / INSERTION OPERATIONS</th>
</tr>
</thead>
<tbody>
<tr>
<td>Course</td>
<td>loadCourse(DepartmentCode, coursenumber)</td>
</tr>
<tr>
<td>Department</td>
<td>loadDepartment(DepartmentCode)</td>
</tr>
<tr>
<td>Department Class Schedule</td>
<td>loadDepartmentClassSchedule(term, year)</td>
</tr>
<tr>
<td>Professor</td>
<td>loadProfessor(yourownid)</td>
</tr>
<tr>
<td>Section</td>
<td>loadSection(department, course, sectionnumber, sectionlocation, term, year)</td>
</tr>
<tr>
<td>Student</td>
<td>loadStudent(studentsid)</td>
</tr>
<tr>
<td>Student/Section</td>
<td>loadStudentSection(studentsid, course, sectionnumber, sectionlocation, term, year)</td>
</tr>
</tbody>
</table>

*Note that this operation will link the section object to the associated student object and also link the student object to the associated section object.*
These requirements may be mapped into an appropriate structure for the type of database.
Review Questions

• See text Review Questions:
  – 10-1, 10-2, 10-5, 10-7*, 10-8*, 10-11 in particular

• Key Ideas:
  – Data base management – why needed, what data goes there?
  – Ties between the programming language view and the data base view – know difference?

Learning Objectives

• Understand the need for persistent objects.

• Learn why the third tier of the three-tier architecture is kept separate.

• Know the types of data bases which are available.

• Use the Singleton pattern to access the third tier.

Learning Objectives
(continued)

• If you have a background in relational data bases, define relational database tables and write the Structured Query Language (SQL) statements required.

• Determine how to assign the three tiers to a processor.