Introduction to Database Management Systems

Dr. Chawalit Jeenanunta
Agenda

- Introduction
- Syllabus
- Term project
- Introduction to DBMS
Introduction

- Please write down
  - your name
  - student ID
  - e-mail address

- chawalitj@gmail.com
Why Use A Database?

- The purpose of a database is to help people and organizations keep track of things
- Problems of using list to store data
  - Data inconsistencies
  - Data privacy: The departments want to share some, but not all, of their data
- Databases store data in single-theme tables
- Tables are related through primary and foreign keys
Components of A Database System

Figure 1.6 Components of a Database System
Initial Vocabulary

- **Data:**
  - raw facts about things and events

- **Information:**
  - transformed data that has value for decision making

- Essential to organize data for retrieval and maintenance
Database
- A collection of data stored in a standardized format, designed to be shared by multiple users.

Database Management System
- Software that defines a database, stores the data, supports a query language, produces reports, and creates data entry screens.

Database System
- The DBMS software together with the data itself. Sometimes, the applications are also included.
Types of Databases and Database Applications

- Numeric and Textual Databases
- Multimedia Databases
- Geographic Information Systems (GIS)
- Data Warehouses
- Real-time and Active Databases
Types of Database

- **Personal database**
  - 1 user; < 10 MB

- **Workgroup database**
  - < 25 users; < 100 MB

- **Organizational database**
  - Hundreds to thousands users
  - >1 Trillion bytes, possibly several databases
Example: Organizational Database

**Figure 1.15** Organizational Database System
### Database Tables (Access)

#### Sale Table

<table>
<thead>
<tr>
<th>SaleID</th>
<th>SaleDate</th>
<th>EmployeeID</th>
<th>CustomerID</th>
<th>SalesTax</th>
</tr>
</thead>
<tbody>
<tr>
<td>4</td>
<td>8/14/2001</td>
<td>3</td>
<td>18</td>
<td>$14.62</td>
</tr>
<tr>
<td>5</td>
<td>10/31/2001</td>
<td>2</td>
<td>21</td>
<td>$6.86</td>
</tr>
<tr>
<td>6</td>
<td>9/15/2001</td>
<td>5</td>
<td>44</td>
<td>$9.82</td>
</tr>
<tr>
<td>7</td>
<td>2/10/2001</td>
<td>4</td>
<td>42</td>
<td>$12.31</td>
</tr>
<tr>
<td>8</td>
<td>3/10/2001</td>
<td>1</td>
<td>15</td>
<td>$17.58</td>
</tr>
<tr>
<td>9</td>
<td>2/10/2001</td>
<td>3</td>
<td>16</td>
<td>$2.81</td>
</tr>
<tr>
<td>10</td>
<td>11/1/2001</td>
<td>8</td>
<td>53</td>
<td>$7.83</td>
</tr>
<tr>
<td>11</td>
<td>12/4/2001</td>
<td>8</td>
<td>60</td>
<td>$3.67</td>
</tr>
<tr>
<td>12</td>
<td>8/15/2001</td>
<td>2</td>
<td>53</td>
<td>$1.19</td>
</tr>
<tr>
<td>13</td>
<td>1/30/2001</td>
<td>7</td>
<td>49</td>
<td>$14.61</td>
</tr>
<tr>
<td>14</td>
<td>9/18/2001</td>
<td>2</td>
<td>9</td>
<td>$3.56</td>
</tr>
<tr>
<td>15</td>
<td>7/20/2001</td>
<td>9</td>
<td>39</td>
<td>$1.13</td>
</tr>
<tr>
<td>16</td>
<td>9/18/2001</td>
<td>8</td>
<td>62</td>
<td>$12.96</td>
</tr>
<tr>
<td>17</td>
<td>2/12/2001</td>
<td>4</td>
<td>71</td>
<td>$16.31</td>
</tr>
<tr>
<td>18</td>
<td>12/21/2001</td>
<td>35</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

#### SaleAnimal Table

<table>
<thead>
<tr>
<th>SaleID</th>
<th>AnimalID</th>
<th>SalePrice</th>
</tr>
</thead>
<tbody>
<tr>
<td>4</td>
<td>8</td>
<td>$183.38</td>
</tr>
<tr>
<td>5</td>
<td>183</td>
<td>$114.30</td>
</tr>
<tr>
<td>6</td>
<td>58</td>
<td>$132.19</td>
</tr>
<tr>
<td>7</td>
<td>24</td>
<td>$147.58</td>
</tr>
<tr>
<td>8</td>
<td>42</td>
<td>$174.27</td>
</tr>
<tr>
<td>9</td>
<td>53</td>
<td>$46.80</td>
</tr>
<tr>
<td>10</td>
<td>9</td>
<td>$1.80</td>
</tr>
<tr>
<td>12</td>
<td>5</td>
<td>$19.80</td>
</tr>
<tr>
<td>13</td>
<td>162</td>
<td>$119.88</td>
</tr>
<tr>
<td>13</td>
<td>199</td>
<td>$100.00</td>
</tr>
<tr>
<td>15</td>
<td>13</td>
<td>$10.80</td>
</tr>
<tr>
<td>16</td>
<td>193</td>
<td>$216.05</td>
</tr>
<tr>
<td>17</td>
<td>11</td>
<td>$184.47</td>
</tr>
<tr>
<td>18</td>
<td>10</td>
<td>$150.11</td>
</tr>
<tr>
<td>19</td>
<td>47</td>
<td>$195.47</td>
</tr>
</tbody>
</table>

#### Animal Table

<table>
<thead>
<tr>
<th>AnimalID</th>
<th>Name</th>
<th>Category</th>
<th>Breed</th>
<th>DateBorn</th>
<th>Gender</th>
<th>Registered</th>
<th>Color</th>
</tr>
</thead>
<tbody>
<tr>
<td>2</td>
<td>Fish</td>
<td>Angel</td>
<td></td>
<td>5/5/2001</td>
<td>Male</td>
<td>AKC</td>
<td>Black</td>
</tr>
<tr>
<td>4</td>
<td>Gary</td>
<td>Dog</td>
<td>Labrador Retriever</td>
<td>3/2/2001</td>
<td>Male</td>
<td>AKC</td>
<td>Spotted</td>
</tr>
<tr>
<td>5</td>
<td>Fish</td>
<td>Shark</td>
<td></td>
<td>1/1/2001</td>
<td>Female</td>
<td></td>
<td>Gray</td>
</tr>
<tr>
<td>6</td>
<td>Rosie</td>
<td>Cat</td>
<td>Oriental Shorthair</td>
<td>8/2/2001</td>
<td>Female</td>
<td>CFA</td>
<td>Gray</td>
</tr>
<tr>
<td>7</td>
<td>Eugene</td>
<td>Cat</td>
<td>Bombay</td>
<td>1/25/2001</td>
<td>Male</td>
<td>CFA</td>
<td>Black</td>
</tr>
<tr>
<td>8</td>
<td>Miranda</td>
<td>Dog</td>
<td>Norfolk Terrier</td>
<td>5/4/2001</td>
<td>Female</td>
<td>AkC</td>
<td>Red</td>
</tr>
<tr>
<td>9</td>
<td>Fish</td>
<td>Guppy</td>
<td></td>
<td>3/10/2001</td>
<td>Male</td>
<td></td>
<td>Gold</td>
</tr>
<tr>
<td>10</td>
<td>Sherri</td>
<td>Dog</td>
<td>Siberian Huskie</td>
<td>9/13/2001</td>
<td>Female</td>
<td>AkC</td>
<td>Black/White</td>
</tr>
<tr>
<td>11</td>
<td>Susan</td>
<td>Dog</td>
<td>Dalmation</td>
<td>1/22/2001</td>
<td>Female</td>
<td>AkC</td>
<td>Spotted</td>
</tr>
</tbody>
</table>

Record: of 196, 191
Database Tables (Oracle)
DBMS Features/Components

- Database engine
  - Storage
  - Retrieval
  - Update
- Query Processor
- Data dictionary
- Utilities
- Security

- Report writer
- Forms generator (input screens)
- Application generator
- Communications
DBMS Query Processor

<table>
<thead>
<tr>
<th>Category</th>
<th>CountOfAnimalID</th>
</tr>
</thead>
<tbody>
<tr>
<td>Dog</td>
<td>100</td>
</tr>
<tr>
<td>Cat</td>
<td>47</td>
</tr>
<tr>
<td>Bird</td>
<td>15</td>
</tr>
<tr>
<td>Fish</td>
<td>14</td>
</tr>
<tr>
<td>Reptile</td>
<td>6</td>
</tr>
<tr>
<td>Mammal</td>
<td>6</td>
</tr>
<tr>
<td>Spider</td>
<td>3</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Field</th>
<th>Category</th>
<th>AnimalID</th>
</tr>
</thead>
<tbody>
<tr>
<td>Table</td>
<td>Animal</td>
<td>Animal</td>
</tr>
<tr>
<td>Totals</td>
<td>Group By</td>
<td>Count</td>
</tr>
<tr>
<td>Sort</td>
<td></td>
<td>Descending</td>
</tr>
<tr>
<td>Criteria</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Or</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
DBMS Report Writer

- All Data
- Database Engine
- Data Dictionary
- Query Processor
- Report Writer

Sales Report

<table>
<thead>
<tr>
<th>Sale</th>
<th>Customer</th>
<th>Employee</th>
</tr>
</thead>
<tbody>
<tr>
<td>4</td>
<td>(444) 687-5581</td>
<td>(444) 331-6629</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Name</th>
<th>Breed</th>
<th>Born</th>
<th>Gender</th>
<th>Registered</th>
<th>Color</th>
<th>ListPrice</th>
<th>SalePrice</th>
</tr>
</thead>
<tbody>
<tr>
<td>別の文字</td>
<td>種類</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Merchandise</th>
<th>Description</th>
<th>Quantity</th>
<th>ListPrice</th>
<th>SalePrice</th>
<th>Extended</th>
</tr>
</thead>
<tbody>
<tr>
<td>1 Dog</td>
<td>DogKennel</td>
<td>1</td>
<td>$45.00</td>
<td>$16.00</td>
<td>$16.00</td>
</tr>
<tr>
<td>36 Dog</td>
<td>Leads</td>
<td>1</td>
<td>$22.00</td>
<td>$19.00</td>
<td>$19.00</td>
</tr>
</tbody>
</table>

Sale Split

- Merchandise: $35.00
- Animal: $183.36
- Total: $218.36
Report Writer (Oracle)
DBMS Input Forms

- All Data
- Database Engine
- Data Dictionary
- Query Processor
- Form Builder

Input Form Design
DBMS Components

- Database Engine
- Data Dictionary
- Security
- Query Processor
- Form Builder
- Report Writer
- Application Generator
- Communication Network
- All Data
DBMS Engine, Security, Utilities

Data Tables

Database Engine

Security

Utilities

User Identification
Access Rights

Concurrency and Lock Manager

Backup and Recovery
Administration

Product
- ItemID: Integer, Unique
- Description: Text, 100 char

Customer
- CustomerID: Integer, Unique
- Name: Text, 50 char

Order
- OrderID
- ODate

Customer
- CustomerID: Integer, Unique
- Name: Text, 50 char

-- Jones
Customer
- CustomerID: Integer, Unique
- Name: Text, 50 char

-- Rojas

-- Dog food
-- Cat food
Typical DBMS Functionality

- Define a database:
  - in terms of data types, structures and constraints
- Construct or Load the Database on a secondary storage medium
- Manipulating the database:
  - querying, generating reports, insertions, deletions and modifications to its content
- Concurrent Processing and Sharing by a set of users and programs
  - yet, keeping all data valid and consistent
Typical DBMS Functionality

Other features:
- Protection or Security measures to prevent unauthorized access
- “Active” processing to take internal actions on data
- Presentation and Visualization of data
Nonprocedural Access

- Query: request for data to answer a question
- Indicate what parts of database to retrieve not the procedural details
- Improve productivity and improve accessibility
- SQL SELECT statement and graphical tools
Application Development

- Form: formatted document for data entry and display
- Report: formatted document for display
- Use nonprocedural access to specify data requirements of forms and reports
Procedural Language Interface

- Combine procedural language with nonprocedural access

Why
- Batch processing
- Customization and automation
- Performance improvement
Transaction Processing

- Transaction: unit of work that should be reliably processed
- Control simultaneous users
- Recover from failures
University Database

Entities:
- students, faculty, courses, offerings, enrollments

Relationships:
- faculty teach offerings,
- students enroll in offerings,
- offerings made of courses, ...

Registration

Faculty Assignment

Grade Recording

Course Scheduling

University Database
Example of a Database

- **Mini-world for the example:** Part of a UNIVERSITY environment.
- **Some mini-world entities:**
  - STUDENTs
  - COURSEs
  - SECTIONs (of COURSEs)
  - (academic) DEPARTMENTs
  - INSTRUCTORs
Example of a Database

- Some mini-world relationships:
  - SECTIONs are of specific COURSEs
  - STUDENTs take SECTIONs
  - COURSEs have prerequisite COURSEs
  - INSTRUCTORs teach SECTIONs
  - COURSEs are offered by DEPARTMENTs
  - STUDENTs major in DEPARTMENTs
University Database (ERD)

**Student**
- StdSSN
- StdClass
- StdMajor
- StdGPA

**Offering**
- OfferNo
- OffLocation
- OffTime

**Faculty**
- FacSSN
- FacSalary
- FacRank
- FacHireDate

**Course**
- CourseNo
- CrsDesc
- CrsUnits

**Enrollment**
- EnrGrade

**Teaches**

**Has**

**Supervises**

**Registers**

**Accepts**
University Database

- **student**
  - StdSSN
  - StdFirstName
  - StdLastName
  - StdCity
  - StdState
  - StdMajor
  - StdClass
  - StdGPA
  - StdZip

- **enrollment**
  - OfferNo
  - StdSSN
  - EnrGrade

- **offering**
  - OfferNo
  - CourseNo
  - OffTerm
  - OffYear
  - OffLocation
  - OffTime
  - FacSSN
  - OffDays

- **faculty**
  - FacSSN
  - FacFirstName
  - FacLastName
  - FacCity
  - FacState
  - FacDept
  - FacRank
  - FacSalary
  - FacSupervisor
  - FacHireDate
  - FacZipCode

- **faculty_1**
  - FacSSN
  - FacFirstName
  - FacLastName
  - FacCity
  - FacState
  - FacDept
Graphical Tool for Nonprocedural Access
Sample Data Entry Form

**Faculty Assignment Form**

- **SocSecNo**: 098-76-5432
- **First Name**: LEONARD
- **Last Name**: VINCE
- **Department**: MS

### Assignments

<table>
<thead>
<tr>
<th>Offer No.</th>
<th>Course No.</th>
<th>Units</th>
<th>Term</th>
<th>Year</th>
<th>Location</th>
<th>Start Time</th>
</tr>
</thead>
<tbody>
<tr>
<td>1234</td>
<td>IS320</td>
<td>4</td>
<td>FALL</td>
<td>2002</td>
<td>BLM302</td>
<td>10:30 AM</td>
</tr>
<tr>
<td>3333</td>
<td>IS320</td>
<td>4</td>
<td>SPRING</td>
<td>2003</td>
<td>BLM214</td>
<td>8:30 AM</td>
</tr>
<tr>
<td>4321</td>
<td>IS320</td>
<td>4</td>
<td>FALL</td>
<td>2002</td>
<td>BLM214</td>
<td>3:30 PM</td>
</tr>
</tbody>
</table>

Record: 1 of 6
# Sample Report

## Faculty Work Load Report for the 2002-2003 Academic Year

<table>
<thead>
<tr>
<th>Department</th>
<th>Name</th>
<th>Term</th>
<th>Offer Number</th>
<th>Units</th>
<th>Limit</th>
<th>Enrollment</th>
<th>Percent</th>
<th>Full</th>
<th>Low Enrollment</th>
</tr>
</thead>
<tbody>
<tr>
<td>FIN</td>
<td>JULIA MILLS</td>
<td>WINTER</td>
<td>5678</td>
<td>4</td>
<td>20</td>
<td>1</td>
<td>5.00%</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Summary for `term = WINTER` (1 detail record)

- Sum: 4
- Avg: 1

Summary for JULIA MILLS

- Sum: 4
- Avg: 1

Summary for `department = FIN` (1 detail record)
Data Independence

- Software maintenance is a large part (50%) of information system budgets
- Reduce impact of changes by separating database description from applications
- Change database definition with minimal effect on applications that use the database
Three Schema Architecture

- View 1
- View 2
- View n

- External to Conceptual Mappings
- Conceptual to Internal Mappings

- External Level
- Conceptual Level
- Internal Level
External Level
Conceptual Level

**Student**
- StdSSN
- StdClass
- StdMajor
- StdGPA

**Offering**
- OfferNo
- OffLocation
- OffTime

**Course**
- CourseNo
- CrsDesc
- CrsUnits

**Faculty**
- FacSSN
- FacSalary
- FacRank
- FacHireDate

**Enrollment**
- EnrGrade

**Teachers**
- Has

**Supervises**

**Registers**
- Accepts
Database Architecture

a) Client, server, and database on the same computer

Client

Server

Database

b) Multiple clients and 1 server on different computers

Client

Server

Database

Client

Client

Client

c) Multiple servers and databases on different computers

Client

Server

Database

Server

Database

Client

Client

Client
Main Characteristics of the Database Approach

- **Self-describing nature of a database system:**
  - A DBMS *catalog* stores the *description* of the database. The description is called *meta-data*. This allows the DBMS software to work with different databases.

- **Insulation between programs and data:**
  - Called *program-data independence*. Allows changing data storage structures and operations without having to change the DBMS access programs.
Main Characteristics of the Database Approach

- **Data Abstraction:**
  - A *data model* is used to hide storage details and present the users with a *conceptual view* of the database.

- **Support of multiple views of the data:**
  - Each user may see a different view of the database, which describes *only* the data of interest to that user.
Main Characteristics of the Database Approach

- **Sharing of data and multiuser transaction processing:**
  - allowing a set of concurrent users to retrieve and to update the database. Concurrency control within the DBMS guarantees that each transaction is correctly executed or completely aborted. OLTP (Online Transaction Processing) is a major part of database applications.
Database Users

- Users may be divided into
  - those who actually use and control the content (called “Actors on the Scene”)
  - those who enable the database to be developed and the DBMS software to be designed and implemented (called “Workers Behind the Scene”).
Database Users

Actors on the scene

- **Database administrators:**
  - responsible for authorizing access to the database, for co-ordinating and monitoring its use, acquiring software, and hardware resources, controlling its use and monitoring efficiency of operations.

- **Database Designers:**
  - responsible to define the content, the structure, the constraints, and functions or transactions against the database. They must communicate with the end-users and understand their needs.

- **End-users:**
  - they use the data for queries, reports and some of them actually update the database content.
Categories of End-users

- **Casual**:  
  - access database occasionally when needed

- **Naïve or Parametric**:  
  - they make up a large section of the end-user population. They use previously well-defined functions in the form of “canned transactions” against the database. Examples are bank-tellers or reservation clerks who do this activity for an entire shift of operations.
Categories of End-users

- **Sophisticated**: these include business analysts, scientists, engineers, others thoroughly familiar with the system capabilities. Many use tools in the form of software packages that work closely with the stored database.

- **Stand-alone**: mostly maintain personal databases using ready-to-use packaged applications. An example is a tax program user that creates his or her own internal database.
Organizational Roles

Specialization

Functional User
- Indirect
- Parametric
- Power

Information Systems
- DBA
- Analyst/Programmer
- Management

Technical
Non Technical
Advantages of Using the Database Approach

- Controlling redundancy in data storage and in development and maintenance efforts.
- Sharing of data among multiple users.
- Restricting unauthorized access to data.
- Providing persistent storage for program Objects
- Providing Storage Structures for efficient Query Processing
Advantages of Using the Database Approach

- Providing backup and recovery services.
- Providing multiple interfaces to different classes of users.
- Representing complex relationships among data.
- Enforcing integrity constraints on the database.
- Drawing Inferences and Actions using rules.
Goal: Build a Business Application

Tools:
- Database Design
- SQL (queries)
- Programming

Best:
Spend your time on design and SQL.

Worst:
Compensate for poor design and limited SQL with programming.
Application Development

tasks

Feasibility
*Identify scope, costs, and schedule*

Analysis
*Gather information from users*

Design
*Define tables, relationships, forms, reports*

Development
*Create forms, reports, and help; test*

Implementation
*Transfer data, install, train, review*

time
1. Identify business rules.

2. Define tables and relationships.

3. Create input forms and reports.

4. Combine as applications for users.
When not to use a DBMS

- **Main inhibitors (costs) of using a DBMS:**
  - High initial investment and possible need for additional hardware.
  - Overhead for providing generality, security, concurrency control, recovery, and integrity functions.

- **When a DBMS may be unnecessary:**
  - If the database and applications are simple, well defined, and not expected to change.
  - If there are stringent real-time requirements that may not be met because of DBMS overhead.
  - If access to data by multiple users is not required.
When not to use a DBMS

- **When no DBMS may suffice:**
  - If the database system is not able to handle the complexity of data because of modeling limitations
  - If the database users need special operations not supported by the DBMS.