Objectives

• In this chapter, you will learn:
  – That successful database design must reflect the information system of which the database is a part
  – That successful information systems are developed within a framework known as the Systems Development Life Cycle (SDLC)
Objectives (cont’d.)

– That within the information system, the most successful databases are subject to frequent evaluation and revision within a framework known as the Database Life Cycle (DBLC)

– How to conduct evaluation and revision within the SDLC and DBLC frameworks

– About database design strategies: top-down vs. bottom-up design and centralized vs. decentralized design
The Information System

• Provides for data collection, storage, and retrieval

• Composed of:
  – People, hardware, software
  – Database(s), application programs, procedures

• Systems analysis
  – Process that establishes need for and extent of information system

• Systems development
  – Process of creating information system
The Information System (cont’d.)

• Applications
  – Transform data into information that forms basis for decision making
  – Usually produce the following:
    • Formal report
    • Tabulations
    • Graphic displays
  – Composed of the following two parts:
    • Data
    • Code: program instructions
FIGURE 9.1 Generating information for decision making

Data
001010011001
1101000010010
0001000110101
1011100010101
1010110001101
0001010000010
0110001001011

Application code

Information

Decisions

SOURCE: Course Technology/Cengage Learning
The Information System (cont’d.)

• Performance depends on three factors:
  – Database design and implementation
  – Application design and implementation
  – Administrative procedures

• Database development
  – Process of database design and implementation
  – Implementation phase includes:
    • Creating database storage structure
    • Loading data into the database
    • Providing for data management
The Systems Development Life Cycle

- Traces history (life cycle) of information system
- Database design and application development mapped out and evaluated
- Divided into following five phases:
  - Planning
  - Analysis
  - Detailed systems design
  - Implementation
  - Maintenance
- Iterative rather than sequential process
FIGURE 9.2
The Systems Development Life Cycle

<table>
<thead>
<tr>
<th>Phase</th>
<th>Action(s)</th>
<th>Section</th>
</tr>
</thead>
<tbody>
<tr>
<td>Planning</td>
<td>Initial assessment, Feasibility study</td>
<td>9.2.1</td>
</tr>
<tr>
<td></td>
<td>User requirements</td>
<td>9.2.2</td>
</tr>
<tr>
<td></td>
<td>Existing system evaluation</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Logical system design</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Detailed system specification</td>
<td>9.2.3</td>
</tr>
<tr>
<td>Implementation</td>
<td>Coding, testing, and debugging</td>
<td>9.2.4</td>
</tr>
<tr>
<td></td>
<td>Installation, fine-tuning</td>
<td></td>
</tr>
<tr>
<td>Maintenance</td>
<td>Evaluation, Maintenance, Enhancement</td>
<td>9.2.5</td>
</tr>
</tbody>
</table>

SOURCE: Course Technology/Cengage Learning
Planning

• General overview of company and objectives
• Assessment of flow-and-extent requirements
  – Should the existing system be continued?
  – Should the existing system be modified?
  – Should the existing system be replaced?
• Study and evaluate alternate solutions
  – Technical aspects of hardware and software requirements
  – System cost
  – Operational cost
Analysis

• Problems defined during planning phase are examined in greater detail during analysis
• Thorough audit of user requirements
• Existing hardware and software systems are studied
• Goal:
  – Better understanding of:
    • System’s functional areas
    • Actual and potential problems
    • Opportunities
Detailed Systems Design

- Designer completes design of system’s processes
- Includes all necessary technical specifications
- Steps laid out for conversion from old to new system
- Training principles and methodologies are also planned
  - Submitted for management approval
Implementation

• Hardware, DBMS software, and application programs are installed
  – Database design is implemented
• Cycle of coding, testing, and debugging continues until database is ready for delivery
• Database is created and system is customized
  – Creation of tables and views
  – User authorizations
Maintenance

- Three types of maintenance activity:
  - Corrective maintenance
  - Adaptive maintenance
  - Perfective maintenance

- Computer-aided systems engineering (CASE)
  - Produce better systems within reasonable amount of time and at reasonable cost
  - CASE-produced applications are structured, documented, and standardized
The Database Life Cycle

• Six phases:
  – Database initial study
  – Database design
  – Implementation and loading
  – Testing and evaluation
  – Operation
  – Maintenance and evolution
FIGURE 9.3 The Database Life Cycle

<table>
<thead>
<tr>
<th>Phase</th>
<th>Action(s)</th>
<th>Section</th>
</tr>
</thead>
</table>
| Database initial study | • Analyze the company situation  
                      • Define problems and constraints  
                      • Define objectives  
                      • Define scope and boundaries | 9.3.1   |
| Database design        | • Create the conceptual design  
                      • DBMS software selection  
                      • Create the logical design  
                      • Create the physical design | 9.3.2   |
| Implementation and loading | • Install the DBMS  
                       • Create the database(s)  
                       • Load or convert the data | 9.3.3   |
| Testing and evaluation | • Test the database  
                       • Fine-tune the database  
                       • Evaluate the database and its application programs | 9.3.4   |
| Operation              | • Produce the required information flow | 9.3.5   |
| Maintenance and evolution | • Introduce changes  
                       • Make enhancements | 9.3.6   |

SOURCE: Course Technology/Cengage Learning
The Database Initial Study

• Overall purpose:
  – Analyze company situation
  – Define problems and constraints
  – Define objectives
  – Define scope and boundaries

• Interactive and iterative processes required to complete first phase of DBLC successfully
A summary of activities in the database initial study

- Analysis of the company situation
  - Company objectives
  - Company operations
  - Company structure

- Definition of problems and constraints

- Database system specifications
  - Objectives
  - Scope
  - Boundaries

SOURCE: Course Technology/Cengage Learning
The Database Initial Study (cont’d.)

• Analyze the company situation
  – General conditions in which company operates, its organizational structure, and its mission
  – Discover what company’s operational components are, how they function, and how they interact
The Database Initial Study (cont’d.)

• Define problems and constraints
  – Formal and informal information sources
  – Finding precise answers is important
  – Accurate problem definition does not always yield a solution
The Database Initial Study (cont’d.)

• Database system objectives must correspond to those envisioned by end users
  – What is proposed system’s initial objective?
  – Will system interface with other systems in the company?
  – Will system share data with other systems or users?

• Scope: extent of design according to operational requirements

• Boundaries: limits external to system
Database Design

• Necessary to concentrate on data characteristics required to build database model

• Two views of data within system:
  – Business view
    • Data as information source
  – Designer’s view
    • Data structure, access, and activities required to transform data into information
FIGURE 9.5 Two views of data: business manager and database designer

Manager's view
- What are the problems?
- What are the solutions?
- What information is needed to implement the solutions?
- What data are required to generate the desired information?

Designer's view
- How must the data be structured?
- How will the data be accessed?
- How are the data transformed into information?

SOURCE: Course Technology/Cengage Learning
FIGURE 9.6 Database design process

<table>
<thead>
<tr>
<th>Section</th>
<th>Stage</th>
<th>Steps</th>
<th>Activities</th>
</tr>
</thead>
<tbody>
<tr>
<td>9.4</td>
<td>Conceptual Design</td>
<td>• Data analysis and requirements</td>
<td>• Determine end-user views, outputs, and transaction requirements</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• Entity Relationship modeling and normalization</td>
<td>• Define entities, attributes, domains, and relationships</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• Data model verification</td>
<td>• Draw ER diagrams; normalize entity attributes</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• Distributed database design*</td>
<td>• Identify ER modules and validate insert, update, and delete rules</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>• Validate reports, queries, views, integrity, access, and security</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>• Define the fragmentation and allocation strategy</td>
</tr>
<tr>
<td>9.5</td>
<td>DBMS Selection</td>
<td>Select the DBMS</td>
<td>DBMS and Hardware Independent</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>• Determine DBMS and data model to use</td>
</tr>
<tr>
<td>9.6</td>
<td>Logical Design</td>
<td>• Map conceptual model to logical model components</td>
<td>DBMS Dependent</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• Validate logical model using normalization</td>
<td>• Define tables, columns, relationships, and constraints</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• Validate logical model integrity constraints</td>
<td>• Normalized set of tables</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• Validate logical model against user requirements</td>
<td>• Ensure entity and referential integrity; define column constraints</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>• Ensure the model supports user requirements</td>
</tr>
<tr>
<td>9.7</td>
<td>Physical Design</td>
<td>• Define data storage organization</td>
<td>Hardware Dependent</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• Define integrity and security measures</td>
<td>• Define tables, indexes, and views' physical organization</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• Determine performance measures*</td>
<td>• Define users, security groups, roles, and access controls</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>• Define database and query execution parameters</td>
</tr>
</tbody>
</table>

* See Chapter 12, Distributed Database Management Systems

* See Chapter 11, Database Performance Tuning and Query Optimization

SOURCE: Course Technology/Cengage Learning
Implementation and Loading

• Actually implement all design specifications from previous phase:
  – Install the DBMS
    • Virtualization: creates logical representations of computing resources independent of physical resources
  – Create the Database
  – Load or Convert the Data
Testing and Evaluation

• Occurs in parallel with applications programming
• Database tools used to prototype applications
• If implementation fails to meet some of system’s evaluation criteria:
  – Fine-tune specific system and DBMS configuration parameters
  – Modify physical or logical design
  – Upgrade software and/or hardware platform
Testing and Evaluation
(cont’d.)

• Integrity
  – Enforced via proper use of primary, foreign key rules

• Backup and Recovery
  – Full backup
  – Differential backup
  – Transaction log backup
Operation

• Once database has passed evaluation stage, it is considered operational
• Beginning of operational phase starts process of system evolution
• Problems not foreseen during testing surface
• Solutions may include:
  – Load-balancing software to distribute transactions among multiple computers
  – Increasing available cache
Maintenance and Evolution

• Required periodic maintenance:
  – Preventive maintenance (backup)
  – Corrective maintenance (recovery)
  – Adaptive maintenance
  – Assignment of access permissions and their maintenance for new and old users
  – Generation of database access statistics
  – Periodic security audits
  – Periodic system-usage summaries
Conceptual Design

- Data modeling creates an abstract database structure
  - Represents real-world objects
- Embodies clear understanding of business and its functional areas
- Ensure that all data needed are in model, and that all data in model are needed
- Requires four steps
Data Analysis and Requirements

• Discover data element characteristics
  – Obtains characteristics from different sources
• Requires thorough understanding of the company’s data types and their extent and uses
• Take into account business rules
  – Derived from description of operations
Entity Relationship Modeling and Normalization

• Designer enforces standards in design documentation
  – Use of diagrams and symbols, documentation writing style, layout, other conventions
• Business rules must be incorporated into conceptual model
• ER model is a communications tool as well as design blueprint
<table>
<thead>
<tr>
<th>STEP</th>
<th>ACTIVITY</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Identify, analyze, and refine the business rules.</td>
</tr>
<tr>
<td>2</td>
<td>Identify the main entities, using the results of Step 1.</td>
</tr>
<tr>
<td>3</td>
<td>Define the relationships among the entities, using the results of Steps 1 and 2.</td>
</tr>
<tr>
<td>4</td>
<td>Define the attributes, primary keys, and foreign keys for each of the entities.</td>
</tr>
<tr>
<td>5</td>
<td>Normalize the entities. (Remember that entities are implemented as tables in an RDBMS.)</td>
</tr>
<tr>
<td>6</td>
<td>Complete the initial ER diagram.</td>
</tr>
<tr>
<td>7</td>
<td>Validate the ER model against the end users’ information and processing requirements.</td>
</tr>
<tr>
<td>8</td>
<td>Modify the ER model, using the results of Step 7.</td>
</tr>
</tbody>
</table>
Data Model Verification

• Verified against proposed system processes
• Revision of original design
  – Careful reevaluation of entities
  – Detailed examination of attributes describing entities
• Define design’s major components as modules:
  – Module: information system component that handles specific function
# Data Model Verification (cont’d.)

## The ER Model Verification Process

<table>
<thead>
<tr>
<th>STEP</th>
<th>ACTIVITY</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Identify the ER model’s central entity.</td>
</tr>
<tr>
<td>2</td>
<td>Identify each module and its components.</td>
</tr>
</tbody>
</table>
| 3    | Identify each module’s transaction requirements:  
     | Internal: updates/inserts/deletes/queries/reports  
     | External: module interfaces |
| 4    | Verify all processes against system requirements. |
| 5    | Make all necessary changes suggested in Step 4. |
| 6    | Repeat Steps 2–5 for all modules. |
FIGURE 9.12 Iterative ER model verification process

1. Identify central entity, modules, and components
2. Define processes and transaction steps
3. Verify ER model
   - Does ER require changes?
     - Yes: Make changes to ER model
     - No: ER model verified

SOURCE: Course Technology/Cengage Learning
Distributed Database Design

• Portions of database may reside in different physical locations
  – Database fragment: subset of a database stored at a given location
• Processes accessing the database vary from one location to another
• Designer must also develop data distribution and allocation strategies
DBMS Software Selection

• Critical to information system’s smooth operation
• Common factors affecting purchasing decisions:
  – Cost
  – DBMS features and tools
  – Underlying model
  – Portability
  – DBMS hardware requirements
Logical Design

- Map conceptual design to specific data model
- Still independent of physical-level details
- Requires all objects be mapped to specific constructs used by selected database software
  - Definition of attribute domains, design of required tables, and access restriction formats
  - Tables must correspond to entities in conceptual design
- Translates software-independent conceptual model into software-dependent model
Map the Conceptual Model to the Logical Model

- Map the conceptual model to the chosen database constructs
- Five mapping steps involved:
  - Strong entities
  - Supertype/subtype relationships
  - Weak entities
  - Binary relationships
  - Higher degree relationships
Validate the Logical Model Using Normalization

• Translation requires the definition of the attribute domains and appropriate constraints
• All defined constraints must be supported by the logical data model
• Special attention should be placed at this stage to ensure security is enforced
  – May have to consider security restrictions at multiple locations
Validate Logical Model Integrity

Constraints

• All defined constraints must be supported by the logical data model

• Ensure:
  – All views can be resolved
  – Security is enforced to ensure the privacy of the data
Validate the Logical Model against User Requirements

- Final step in the logical design process
- Validate all logical model definitions against all end-user data, transaction, and security requirements
Physical Design

• Process of selecting data storage and data access characteristics of database
• Storage characteristics are a function of:
  – Device types supported by hardware
  – Type of data access methods supported by system
  – DBMS
• More complex when data are distributed
Define Data Storage Organization

• Designer must determine several attributes:
  – Data volume
  – Data usage patterns
• These in turn influence:
  – Location and physical storage organization for each table
  – What indexes and the type of indexes to be used for each table
  – What views and the type of views to be used on each table
Define Integrity and Security Measures

• Define user and security groups and roles
  – Database role: set of database privileges that could be assigned as a unit to a user or group

• Assign security controls
  – Specific access rights on database objects to a user or group of users
  – Can also revoke during operation to assist with backups or maintenance events
Determine Performance Measures

• Performance can be affected by characteristics:
  – Storage media
  – Seek time
  – Sector and block (page) size
  – And more…

• Fine-tuning the DBMS and queries to ensure that they will meet end-user performance requirements
Database Design Strategies

• Top-down design
  – Identifies data sets
  – Defines data elements for each of those sets
    • Definition of different entity types
    • Definition of each entity’s attributes

• Bottom-up design
  – Identifies data elements (items)
  – Groups them together in data sets
Figure 9.14 Top-down vs. bottom-up design sequencing

Conceptual model

Entity

Attribute

Entity

Attribute

Attribute

Attribute

Source: Course Technology/Cengage Learning
Centralized vs. Decentralized Design

• Centralized design
  – When data component is composed of small number of objects and procedures
  – Typical of small systems

• Decentralized design
  – Data component has large number of entities
  – Complex relations on which complex operations are performed
  – Problem is spread across several operational sites
Decentralized design

Data component

Submodule criteria

Conceptual models

Engineering

Purchasing

Manufacturing

Views

Processes

Constraints

Views

Processes

Constraints

Views

Processes

Constraints

Verification

Aggregation

Conceptual model

Data dictionary

SOURCE: Course Technology/Cengage Learning
Centralized vs. Decentralized Design (cont’d.)

• All modules are integrated into one model
• Aggregation problems to be addressed:
  – Synonyms and homonyms
  – Entity and entity subtypes
  – Conflicting object definitions
**Summary of aggregation problems**

*Synonyms:* Two departments use different names for the same entity.

- Label used:
  - Department A: X
  - Department B: Y

*Homonyms:* Two different entities are addressed by the same label. (Department B uses the label X to describe both entity X and entity Y.)

- Label used:
  - X
  - X

*Entity and entity subclass:* The entities X1 and X2 are subsets of entity X.

**Example:**

```
Entity X
   / \                       / \                   / \
  /   \                     /   \                 /   \
Entity X1    Entity X2    EMPLOYEE    SECRETARY    PILOT
  \   /                       \   /                 \   /
Department A  Department B  Name Address Phone  Typing speed Classification  Hours flown License
```

*Conflicting object definitions:* Attributes for the entity PROFESSOR

- **Conflicting definitions**
  - Primary key: PROF_SSN, PROF_NUM
  - Phone attribute: 898-2853, 2853

*Common attributes*:

- Name
- Address
- Phone

*Distinguishing attributes*:

- Payroll Dept.
- Systems Dept.
Summary

• Information system facilitates transformation of data into information
  – Manages both data and information
• SDLC traces history (life cycle) of an application within the information system
• DBLC describes history of database within the information system
Summary (cont’d.)

• Database design and implementation process moves through series of well-defined stages
• Conceptual design subject to several variations:
  – Top-down vs. bottom-up
  – Centralized vs. decentralized