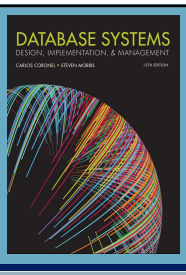


Chapter 9
Database Design

Learning Objectives

- After completing this chapter, you will be able to:
  - Describe the role of database design as the foundation of a successful information system
  - Describe the five phases in the Systems Development Life Cycle (SDLC)
  - Design databases using the six phases in the Database Life Cycle (DBLC) framework
  - Conduct evaluation and revision within the SDLC and DBLC frameworks
  - Distinguish between top-down and bottom-up approaches in database design
  - Distinguish between centralized and decentralized conceptual database design

The Information System (1 of 2)

- The database is part of a larger whole known as an information system (IS)
  - Provides for data collection, storage, and retrieval
    - People, hardware, and software
    - Database(s), application programs, and procedures
- Systems analysis: establishes need for and extent of information system
  - Systems development: process of creating information system

The Information System (2 of 2)

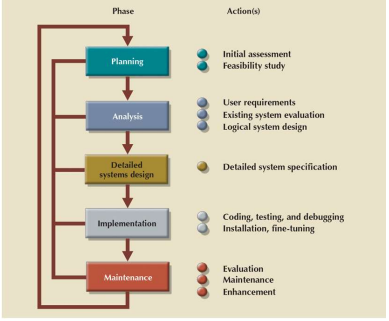
- Performance factors of an information system
  - Database design and implementation
  - Application design and implementation
  - Administrative procedures
- Database development
  - Process of database design and its implementation

Systems Development Life Cycle (SDLC) (1 of 2)

- Traces history of an information system
  - Provides a picture within which database design and application development are mapped out and evaluated
- Traditional SDLC is divided into five phases
  - Planning: yields a general overview of the company and its objectives
    - Should the existing system be continued/modified/replaced?
    - Feasibility Study - Technical aspects of hardware/software requirements. System costs (can we afford it?). Operational Cost (do we have human, technical and financial resources to keep the system operational?
  - Analysis: problems defined during planning phase are examined in greater detail
    - Requirements of the current system's end users? Do they fit in the overall IS requirements?
  - Systems design: designer completes the design of the system's processes
    - Design includes tech specs for the screens, menus, reports. Conversion steps from old to new. Training principles need to be approved.
  - Implementation: hardware, DBMS software, and application programs are installed, and the database design is implemented
  - Maintenance: corrective (in response to system errors), adaptive (due to changes in business environment), and perfective (enhance the system).

Systems Development Life Cycle (SDLC) (2 of 2)

FIGURE 9.2 THE SYSTEMS DEVELOPMENT LIFE CYCLE (SDLC)

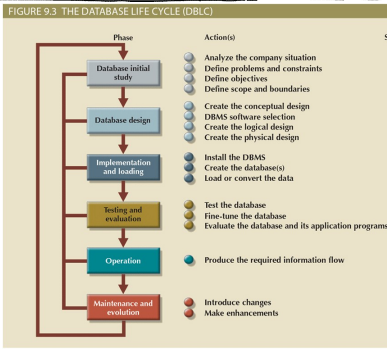


The Database Life Cycle (1 of 5)

- The Database Life Cycle (DBLC) contains six phases
  - Database initial study: define problems, constraints, objectives, scope, and boundaries
  - Database design: making sure that the final product meets user and system requirements
  - Implementation and loading: DBMS is installed, database is created, and data is loaded or converted
  - Testing and evaluation: database is tested, fine-tuned, and evaluated
    - Full backup/dump: all database objects are backed up in their entirety
    - Differential backup: only modified/updated objects since last full backup are backed up
    - Transaction log backup: only the transaction log operations that are not reflected in a previous backup are backed up
  - Operation: problems are identified and solutions implemented
  - Maintenance and evolution: preventative, corrective, adaptive, etc.

The Database Life Cycle (2 of 5)

FIGURE 9.3 THE DATABASE LIFE CYCLE (DBLC)



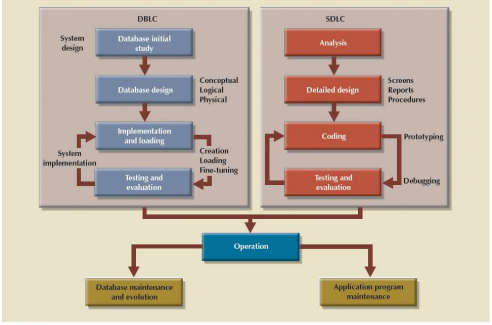
The Database Life Cycle (4 of 5)

Table 9.1: Common Sources of Database Failure

Source	Description	Example
Software	Software-induced failures may be traceable to the operating system, the DBMS software, application programs, or viruses and other malware.	In April 2017, a new vulnerability was found in the Oracle E -Business Suite, that allows an unauthenticated attacker to create, modify, or delete critical data.
Hardware	Hardware-induced failures may include memory chip errors, disk crashes, bad disk sectors, and disk-full errors.	A bad memory module or a multiple hard disk failure in a database system can bring it to an abrupt stop.
Programming exemptions	Application programs or end users may roll back transactions when certain conditions are defined. Programming exemptions can also be caused by malicious or improperly tested code that can be exploited by hackers.	In February 2016 a group of unidentified hackers fraudulently instructed the New York Federal Reserve Bank to transfer $81 million from the central bank of Bangladesh to accounts in the Philippines. The hackers used fraudulent messages injected by malware disguised as a PDF reader.
Transactions	The system detects deadlocks and aborts one of the transactions. (See Chapter 10.)	Deadlock occurs when executing multiple simultaneous transactions.
External factors	Backups are especially important when a system suffers complete destruction from fire, earthquake, flood, or other natural disaster.	In August 2015, lightning struck a local utility provider's grid near Google's data centers in Belgium. Although power backup kicked in automatically, the interruption was long enough to cause permanent data loss in affected systems.

The Database Life Cycle (5 of 5)

FIGURE 9.8 PARALLEL ACTIVITIES IN THE DBLC AND THE SDLC



CENGAGE

10

Test your knowledge

- **You have been assigned to design the database for a car repair garage. A set of activities is listed below. Indicate the most appropriate sequence of activities by labeling each of the following steps in the correct order.**

____ **Normalize the conceptual model.**
____ **Obtain a general description of company operations.**
____ **Load the database.**
____ **Create a description of each system process.**
____ **Test the system.**
____ **Draw a data flow diagram and system flowcharts.**
____ **Create a conceptual model, using ER diagrams.**
____ **Create the application programs.**
____ **Interview the mechanics.**
____ **Create the file (table) structures.**
____ **Interview the shop manager.**

CENGAGE

11

Solution

- __7__ Normalize the conceptual model.
- __3__ Obtain a general description of company operations.
- __9__ Load the database.
- __4__ Create a description of each system process.
- _11__ Test the system.
- __6__ Draw a data flow diagram and system flow charts.
- __5__ Create a conceptual model, using E-R diagrams.
- _10__ Create the application programs.
- __2__ Interview the mechanics.
- __8__ Create the file (table) structures.
- __1__ Interview the shop manager.

CENGAGE

12

Homework

- How do the activities fit in the SDLC phases of Analysis, Design, Implementation?

Conceptual Design

- Goal: design a database independent of database software and physical details
  - Conceptual data model: describes main data entities, attributes, relationships, and constrains
    - Designed as software and hardware independent
- Minimum data rule
  - All that is needed is there, and all that is there is needed

Conceptual Design

	Table 9.2: Conceptual Design Steps
Step	**Activity**
1	Data analysis and requirements
2	Entity relationship modeling and normalization
3	Data model verification
4	Distributed database design

Conceptual Design

- Data analysis and requirements
  - Designers efforts are focused
    - Information needs, users, sources and constitution
  - Answers obtained from a variety of sources
    - Developing and gathering end-user data views
    - Directly observing current system: existing and desired output
    - Interfacing with the systems design group
- Entity relationship modeling and normalization
  - All objects (entities, attributes, relations, views, and so on) are defined in a data dictionary, which is used in tandem with the normalization process

Conceptual Design

	Table 9.3: Developing the Conceptual Model Using ER Diagrams
Step	**Activity**
1	Identify, analyze, and refine the business rules
2	Identify the main entities, using the results of Step 1
3	Define the relationships among the entities, using the results of Steps 1 and 2
4	Define the attributes, primary keys, and foreign keys for each of the entities
5	Normalize the entities (remember that entities are implemented as tables in an RDBMS)
6	Complete the initial ER diagram
7	Validate the ER model against the end users' information and processing requirements
8	Modify the ER model, using the results of Step 7

Conceptual Design

- Data model verification
  - Verified against proposed system processes
  - Run through a series of tests
- Important concepts
  - Module: information system component that handles specific business function
  - Cohesivity: strength of the relationships among the module's entities
  - Module coupling: extent to which modules are independent to one another
    - Low coupling decreases unnecessary intermodule dependencies

Conceptual Design

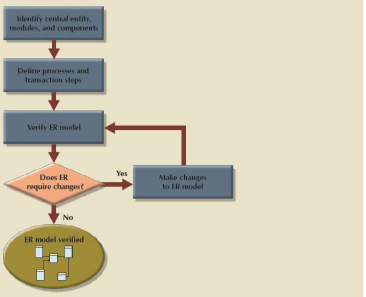
Table 9.5: The ER Model Verification Process	
Step	**Activity**
1	Identify the ER model's central entity
2	Identify each module and its components
3	Identify each module's transaction requirements: • Internal: updates/inserts/deletes/queries/reports • External: module interfaces
4	Verify all processes against the module's processing and reporting requirements
5	Make all necessary changes suggested in Step 4
6	Repeat Steps 2–5 for all modules

CENGAGE

19

Conceptual Design

FIGURE 9.12 ITERATIVE ER MODEL VERIFICATION PROCESS



CENGAGE

20

Conceptual Design

- Distributed database design
  - Portions of database may reside in different physical locations
  - Database fragment: subset of a database stored at a given location
  - Ensures database integrity, security, and performance

CENGAGE

21

DBMS Software Selection

- Factors that affect the purchasing
  - Cost
  - DBMS features and tools
  - Underlying model
  - Portability
  - DBMS hardware requirements

Logical Design

- Goal: design an enterprise-wide database that is based on a specific data model but independent of physical-level details
  - Requires that all objects in the conceptual model be mapped to the specific constructs used by the selected database model
- Validates logical model
  - Using normalization
  - Integrity constraints
  - Against user requirements

Physical Design

- Process of data storage organization and data access characteristics of the database; ensures integrity, security, and performance
  - Define data storage organization
  - Define integrity and security measures
  - Determine performance measures

Database Design Strategies

- Top-down design starts by identifying the data sets and then defines the data elements for each of those sets
  - Involves the identification of different entity types and the definition of each entity's attributes
- Bottom-up design first identifies the data elements (items) and then groups them together in data sets
  - First defines attributes, and then groups them to form entities

Centralized versus Decentralized Design

- Centralized design: process by which all database design decisions are carried out centrally by a small group of people
  - Suitable in a top-down design approach when the problem domain is relatively small, as in a single unit or department in an organization
- Decentralized design: process in which conceptual design models subsets of an organization's database requirements, which are then aggregated into a complete design
  - Such modular designs are typical of complex systems with a relatively large number of objects and procedures

Summary

- An information system is designed to help transform data into information and to manage both data and information
- The Systems Development Life Cycle (SDLC) traces the history of an application within the information system
- The Database Life Cycle (DBLC) describes the history of the database within the information system
- The conceptual portion of the design may be subject to several variations based on two basic design philosophies: bottom-up versus top-down and centralized versus decentralized