Chapter 9

Database Design

Learning Objectives (1 of 2)

- 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)

Learning Objectives (2 of 2)

- In this chapter, you will learn:
  - 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 versus 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

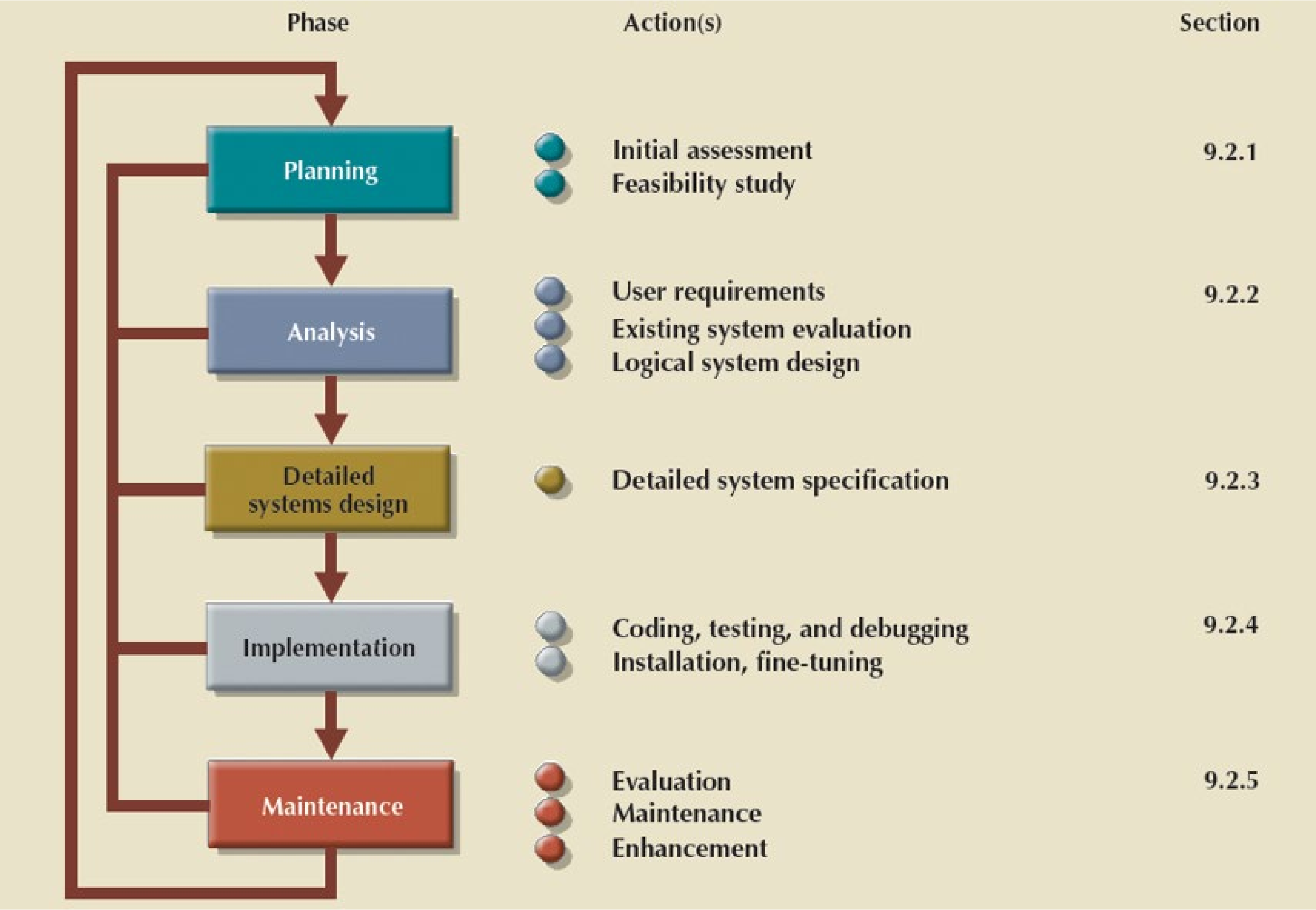
Performance Factors of an Information System

- Database design and implementation
- Application design and implementation
- Administrative procedures
- **Database development**: Process of database design and its implementation

Systems Development Life Cycle (SDLC)

- Traces history of an information system
- Provides a picture within which database design and application development are mapped out and evaluated
- Iterative rather than sequential process

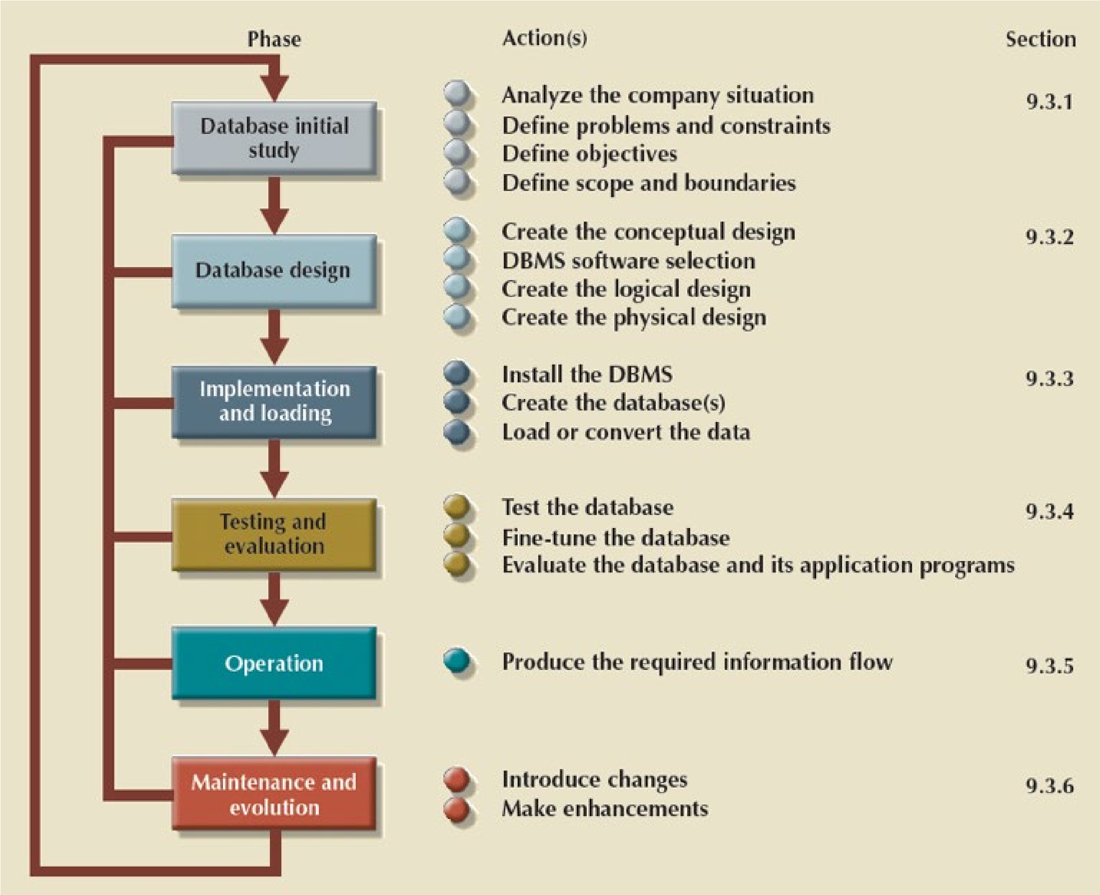
Figure 9.2 - The Systems Development Life Cycle (SDLC)



Computer-Aided Systems Engineering (CASE)

- Includes System Architect and Visio Professional
- Helps produce better systems in a reasonable amounts of time and reasonable cost
- Applications are more structured, better documented and standardized
  - Prolongs operational life of systems
  - Easier and cheaper to update and maintain

Figure 9.3 - The Database Life Cycle (DBLC)

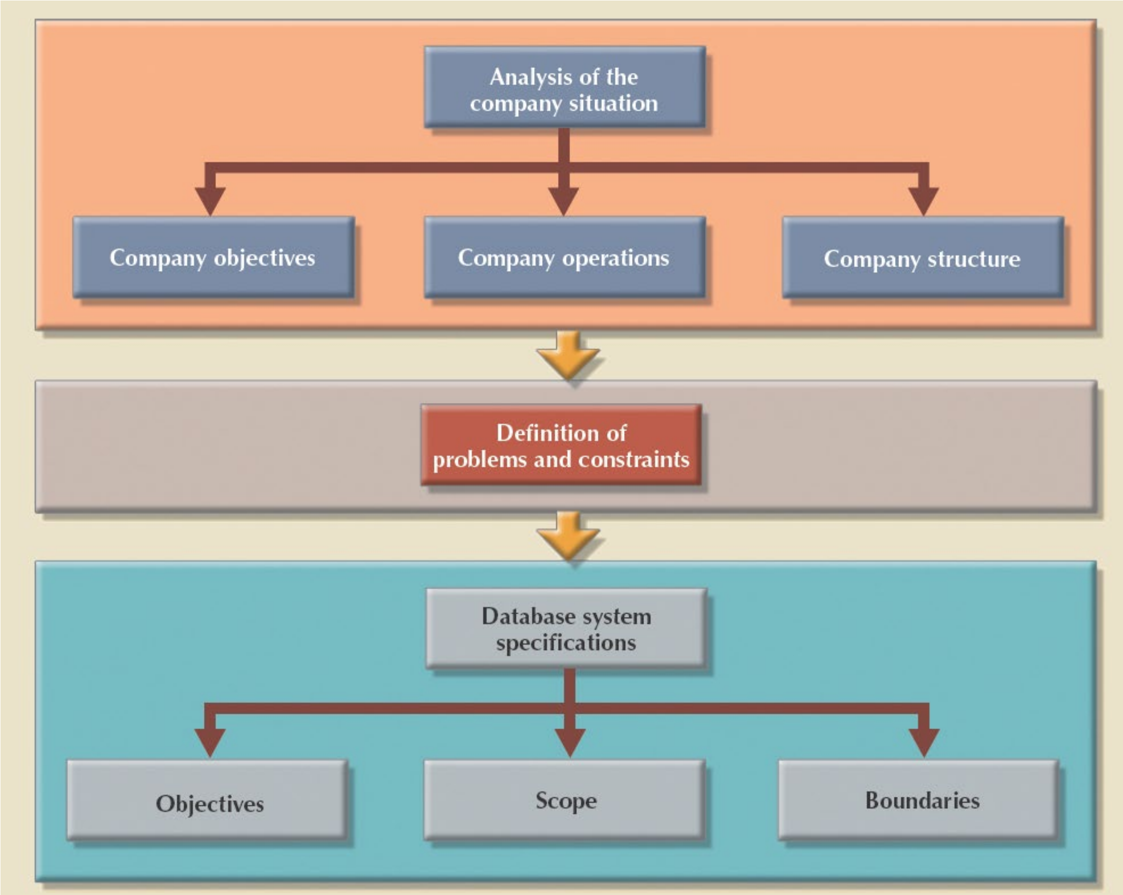


CENGAGE

Purpose of Database Initial Study

- Analyze company situation
- Define problems and constraints
- Define objectives
- Define scope and boundaries

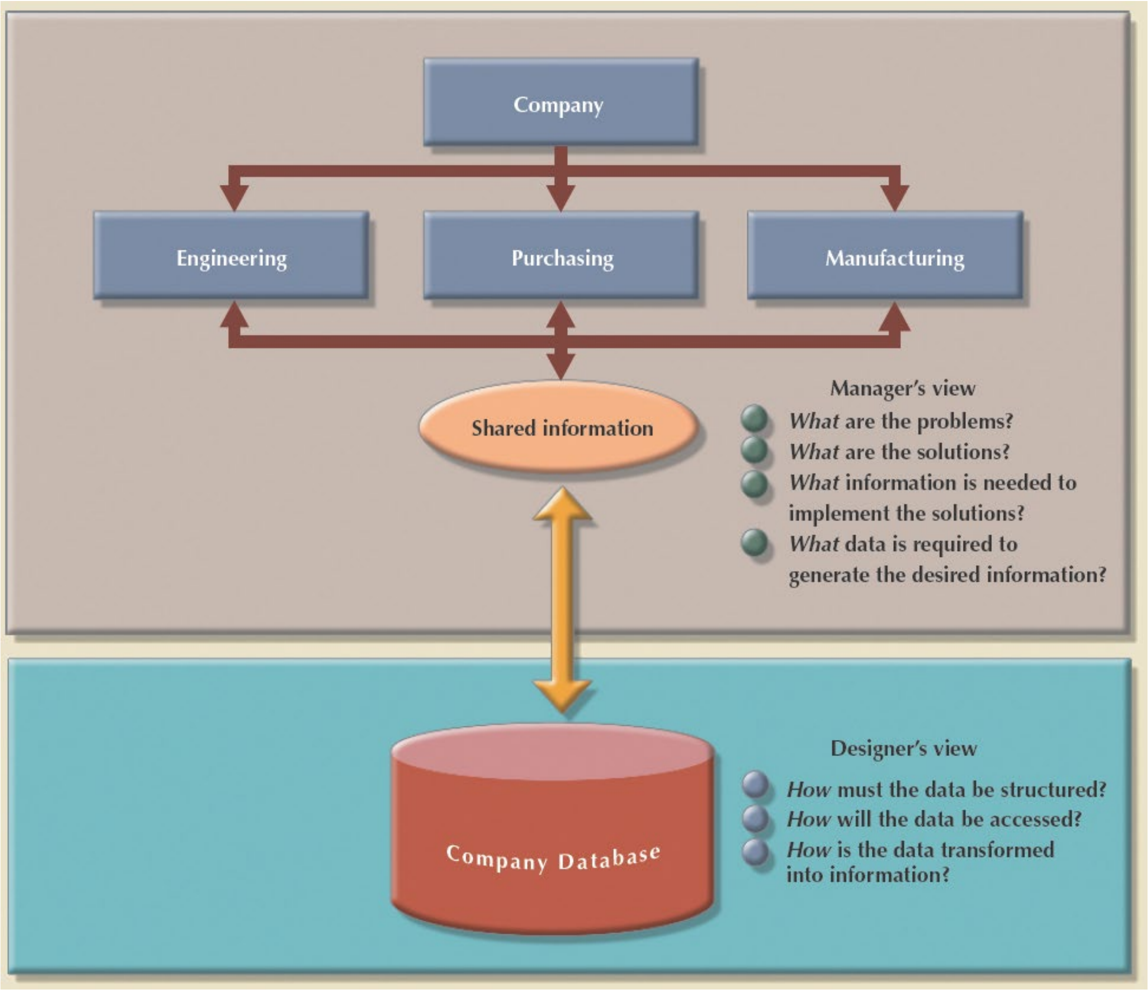
Figure 9.4 - A Summary of Activities in the Database Initial Study



Database Design

- Supports company's operations and objectives
- Most critical phase
  - Ensures final product meets user and system requirements
- Points for examining completion procedures
  - Data component is an element of whole system
  - System analysts/programmers design procedures to convert data into information
  - Database design is an iterative process

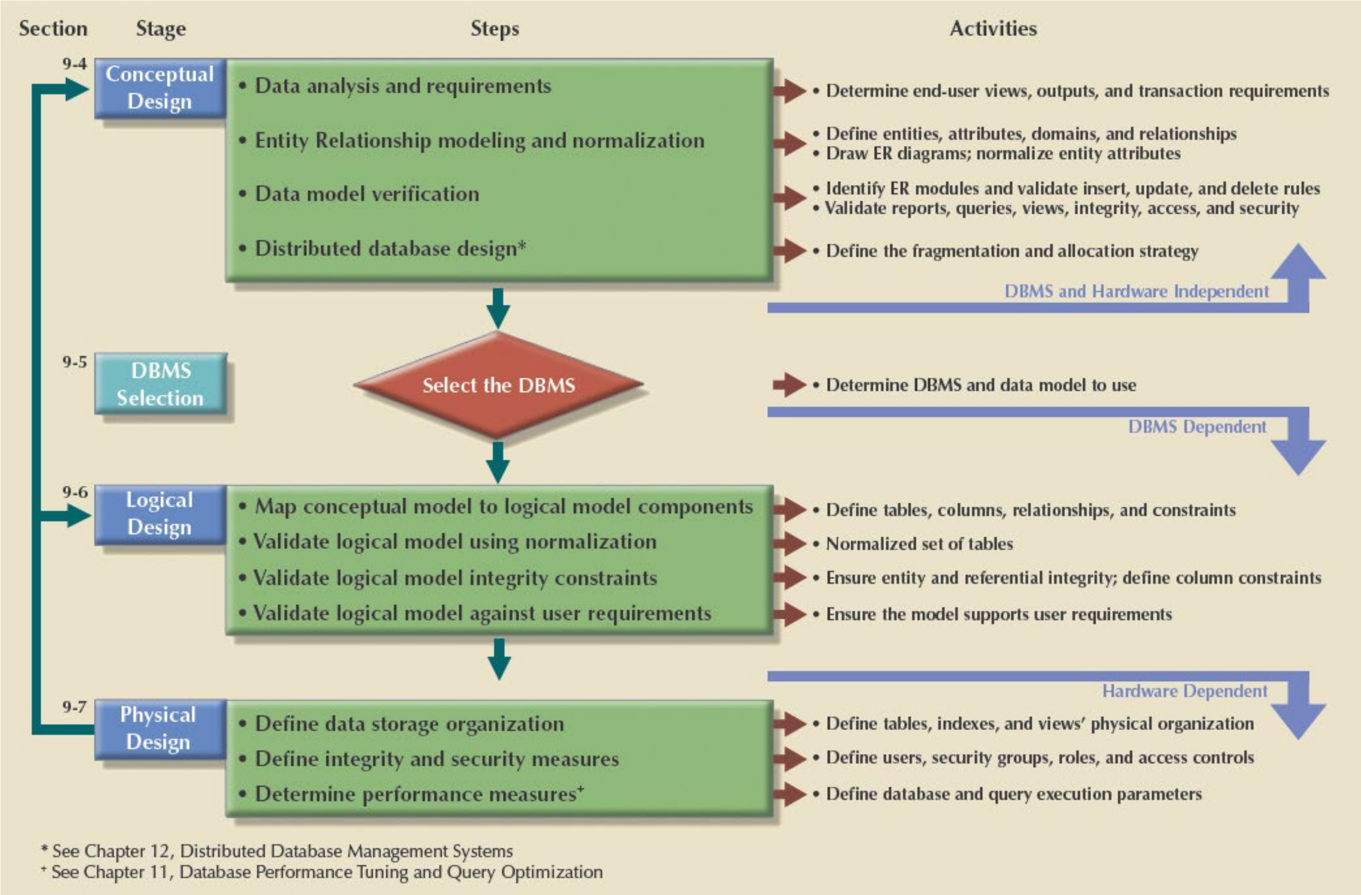
Figure 9.5 - Two Views of Data: Business Manager and Database Designer



Implementation and Loading

- Install the DBMS
  - **Virtualization**: Creates logical representations of computing resources independent of underlying physical computing resources
- Create the databases
  - Requires the creation of special storage-related constructs to house the end-user tables
- Load or convert the data
  - Requires aggregating data from multiple sources

Figure 9.6 - Database Design Process



Testing and Evaluation

- Physical security
- Password security
- Access rights
- Audit trails
- Data encryption
- Diskless workstations
- Optimization

Levels of Database Backups

- **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
- Backups are provided with high security

Create a Database from a dump

- Create a dump
- Create new database from the dump

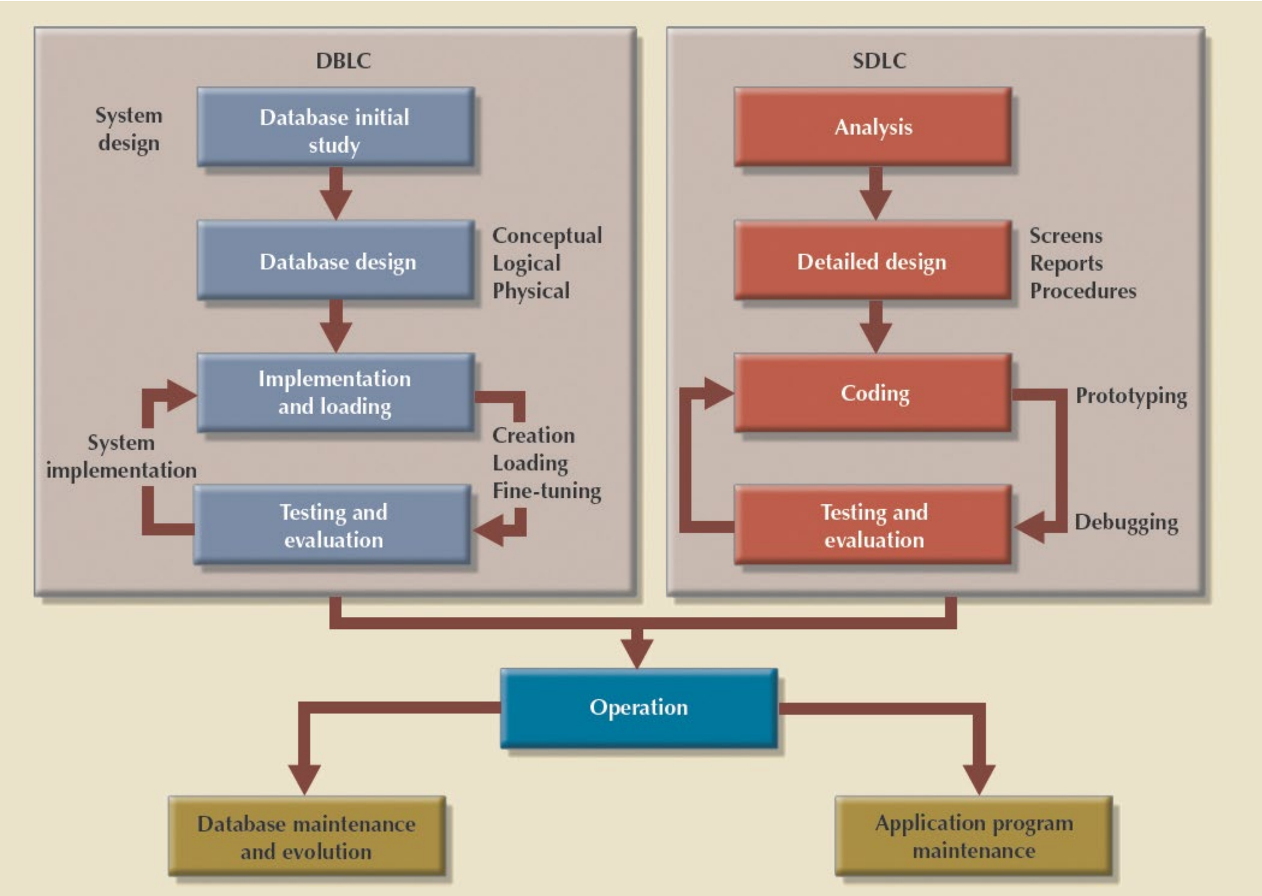
Table 9.1 – Common Sources of Database Failure

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 January 2015, a security vulnerability was found for Oracle E-Business Suite that could cause serious data compromise.[4]
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.	Hackers constantly search for ways to exploit unprotected web database systems. For example, in February 2015, Anthem, the second largest health insurer, announced that it was hacked and data for 80 million customers might have been exposed.[5]
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 2012, Hurricane Sandy hit the northeastern United States, causing data and service losses worth billions of dollars across multiple states.

Maintenance and Evolution

- 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

Figure 9.8 - Parallel Activities in the DBLC and the SDLC



Conceptual Design

- Designs 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

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

Data Analysis and Requirements

- Designers efforts are focused on
  - 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

Description of Operations

Provides precise, up-to-date, and reviewed description of activities defining an organization's operating environment

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.

Figure 9.10 – ER Modeling is an Iterative Process Based on Many Activities

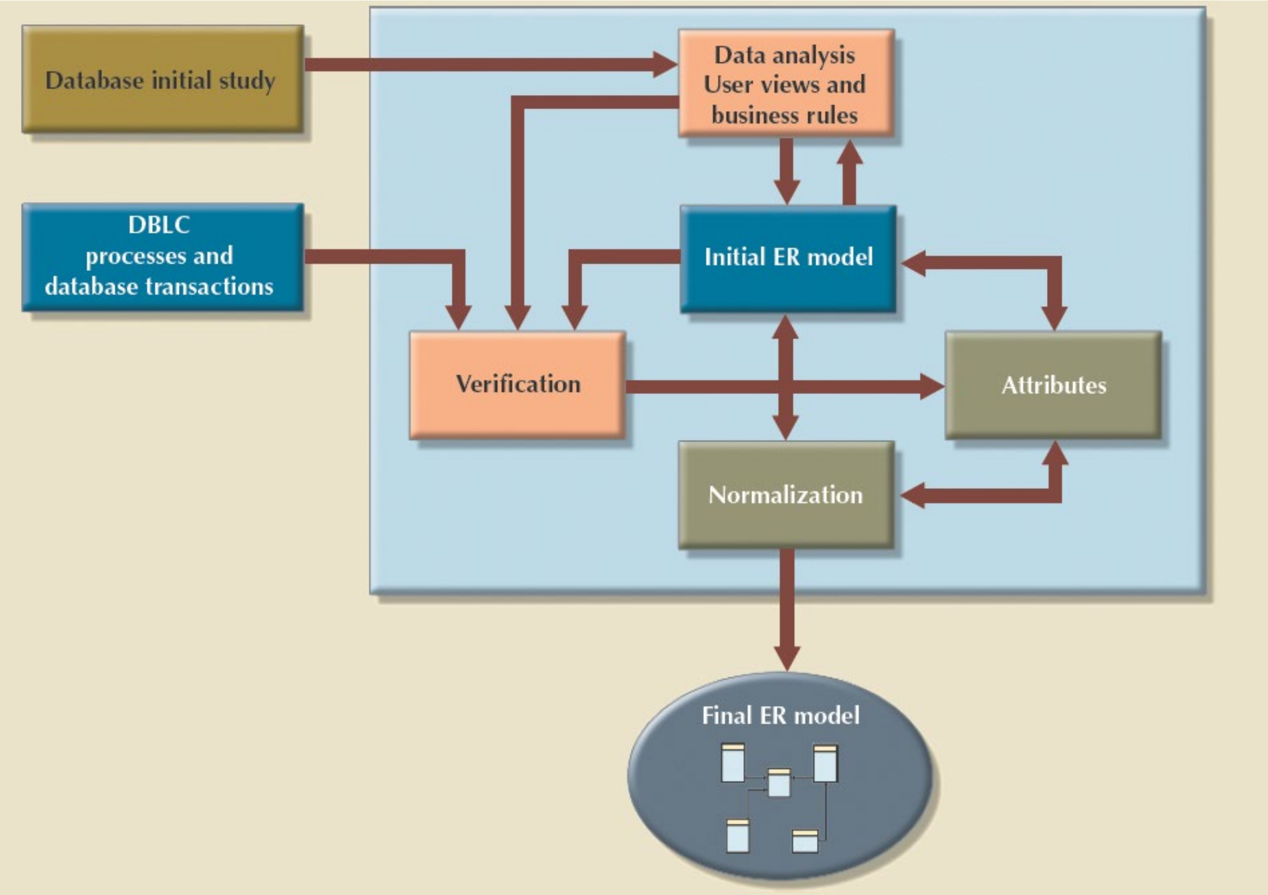
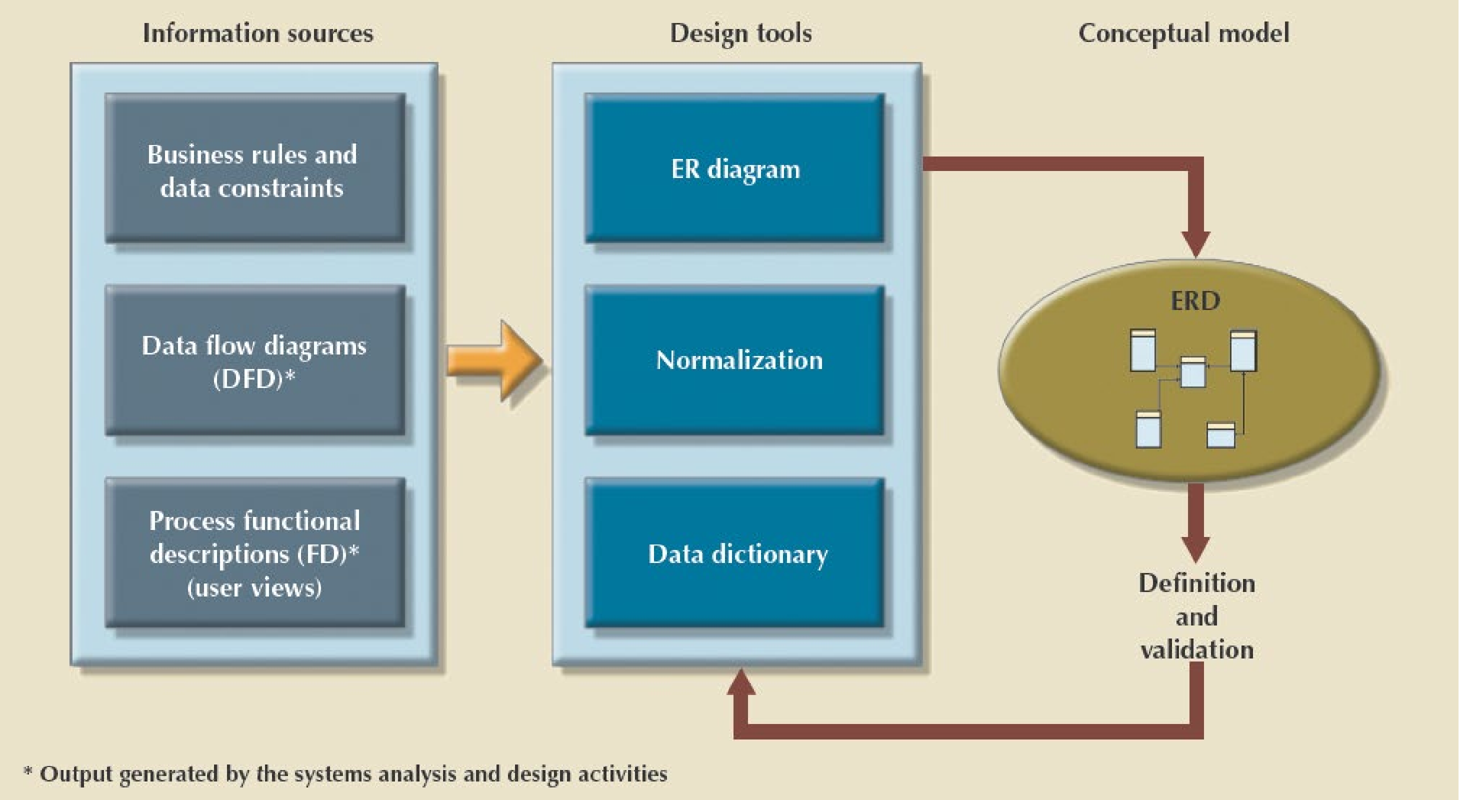


Figure 9.11 - Conceptual Design Tools and Information Sources



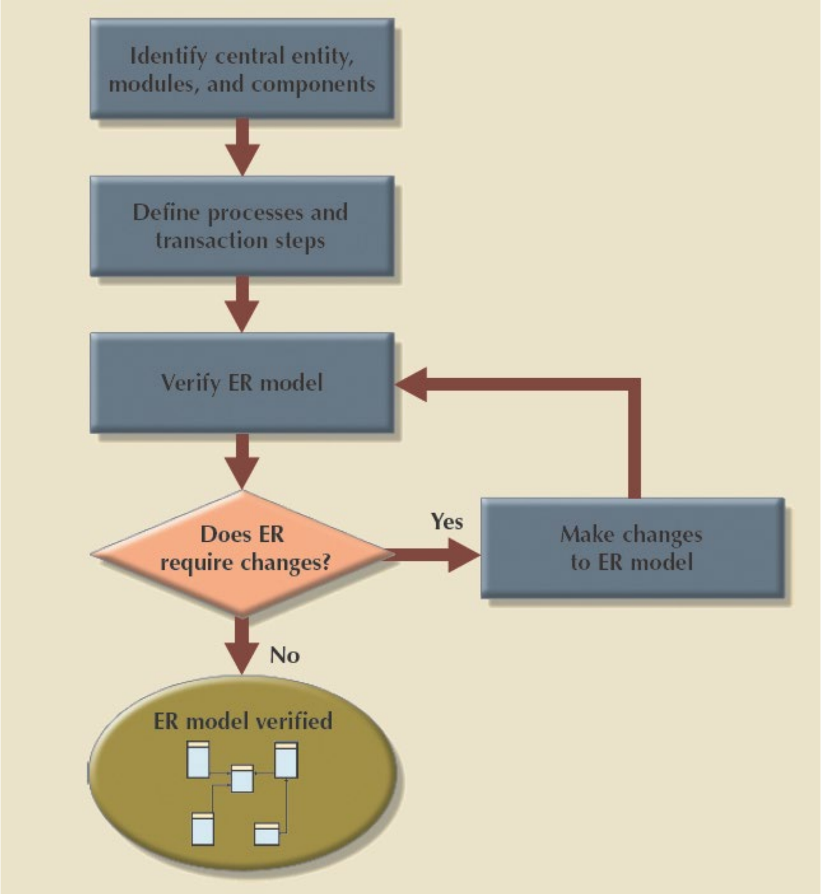
Data Model Verification

- Verified against proposed system processes
- **Module**: Information system component that handles specific business function (different teams can work on different modules)
- Each module is supported by an ER segment that is a subset or fragment of an enterprise ER model.
- Better if modules' ER fragments are merged into a single enterprise ER model which triggers (after individual development)
  - Careful reevaluation of entities
  - Detailed examination of attributes describing entities
- Resulting model verified against each of the module's processes

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 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



Cohesivity and Module Coupling

- **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

Distributed Database Design

- Portions of database may reside in different physical locations
- **Database fragment**: Subset of a database stored at a given location
- Defines the optimum allocation strategy for database fragments to ensure database integrity, security, and performance

DBMS Software Selection

- Cost
- DBMS features and tools
- Underlying model
- Portability
- DBMS hardware requirements

Logical and Physical Design

- **Logical design**: Designs an enterprise-wide database that is based on a specific data model but independent of physical-level details
- Validates logical model:
  - Using normalization
  - Integrity constraints
  - Against user requirements
- **Physical design**: Process of data storage organization and data access characteristics of the database

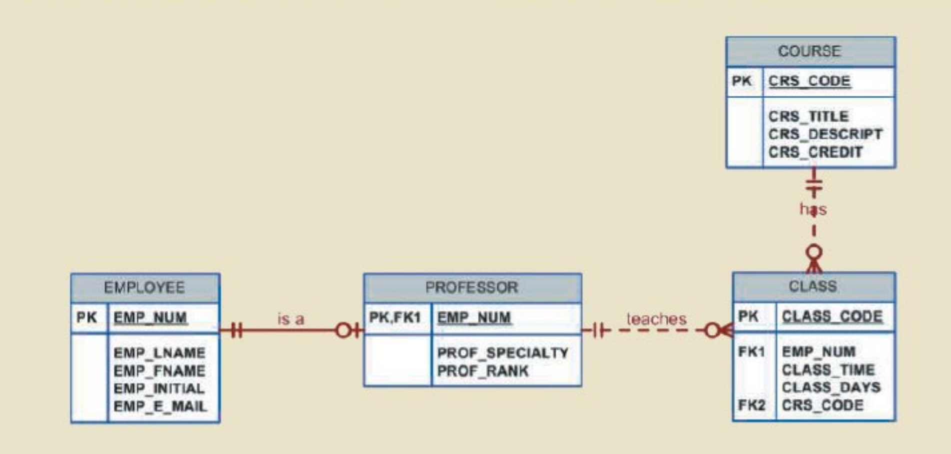
Table 9.6 - Logical Design Steps

STEP	ACTIVITY
1	Map the conceptual model to logical model components.
2	Validate the logical model using normalization.
3	Validate the logical model integrity constraints.
4	Validate the logical model against user requirements.

Table 9.7 - Mapping the Conceptual Model to the Relational Model

STEP	ACTIVITY
1	Map strong entities
2	Map supertype/subtype relationships.
3	Map weak entities.
4	Map binary relationships.
5	Map higher-degree relationships.

FIGURE 9.13 THE SIMPLE COLLEGE CONCEPTUAL MODEL



COURSE (<u>**CRS_CODE**</u>, CRS_TITLE, CRS_DESCRIPT, CRS_CREDIT)

PRIMARY KEY: CRS_CODE

CLASS (<u>**CLASS_CODE**</u>, *EMP_NUM*, CLASS_TIME, CLASS_DAYS, *CRS_CODE*)

PRIMARY KEY: CLASS_CODE

FOREIGN KEYS: EMP_NUM REFERENCES PROFESSOR

CRS_CODE REFERENCES COURSE

Table 9.8 - Physical Design Steps

STEP	ACTIVITY
1	Define data storage organization.
2	Define integrity and security measures.
3	Determine performance measurements.

Physical design: A stage of database design that maps the data storage and access characteristics of a database.

Clustered Tables and Database Role

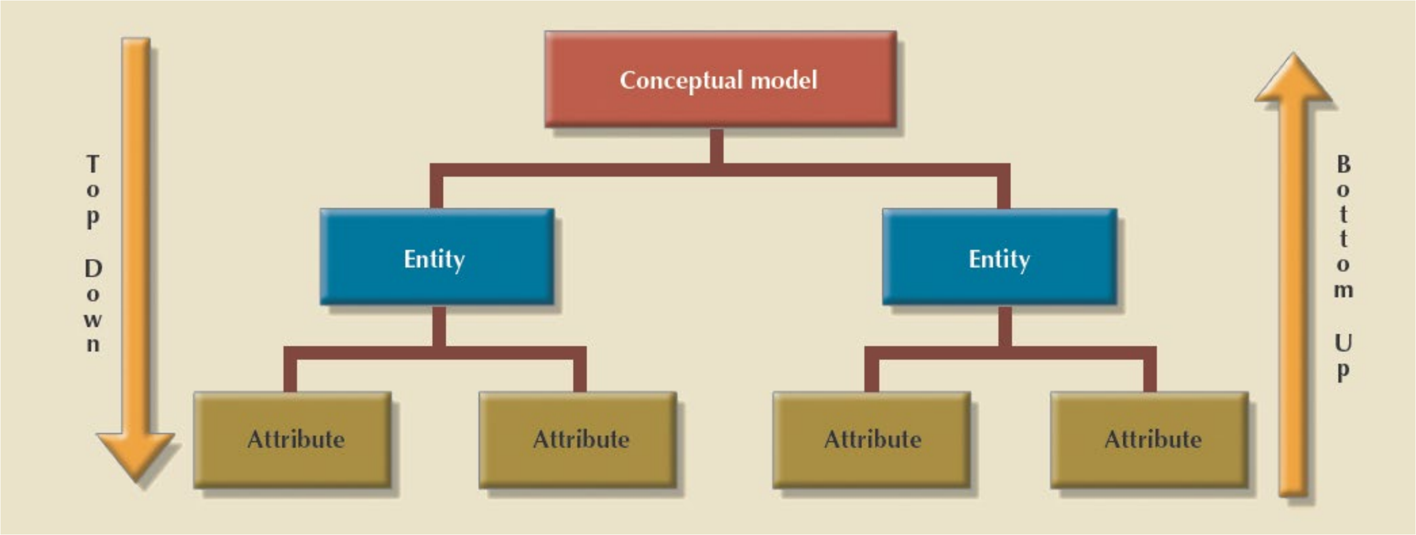
Clustered Tables: Technique that stores related rows from two related tables in adjacent data blocks on disk

Database Role: Set of database privileges that could be assigned as a unit to a user or group

Database Design Strategies

- There are two classical approaches to database design:
- Top-down design
  - starts by identifying the data sets and then defines the data elements for each of those sets. This process 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. In other words, it first defines attributes, and then groups them to form entities.

Figure 9.14 - Top-down versus Bottom-up Design Sequencing



Centralized Versus Decentralized Design

- The two general approaches to database design (bottom-up and top-down) can be influenced by factors
  - scope and size of the system,
  - the company's management style, and
  - the company's structure (centralized or decentralized).
- Depending on these factors, the database design may be based on two very different design philosophies:
  - centralized and decentralized.

Figure 9.15 - Centralized Design

Centralized design is productive when the data component has a relatively small number of objects and procedures. The design can be carried out and represented in a fairly simple database

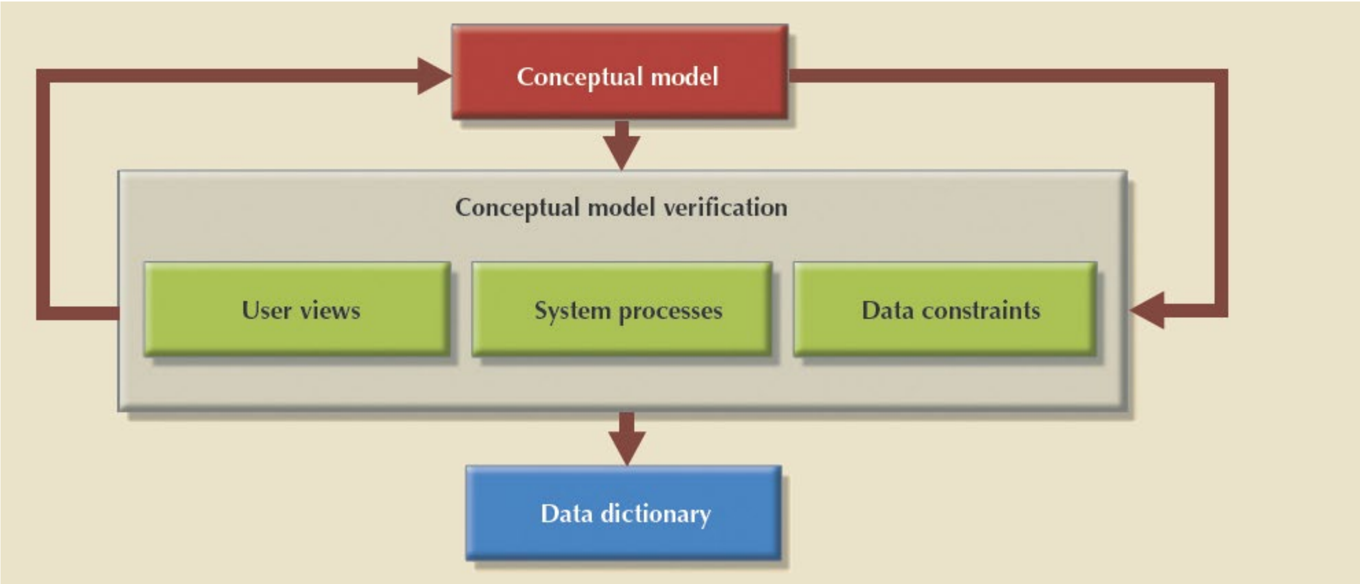
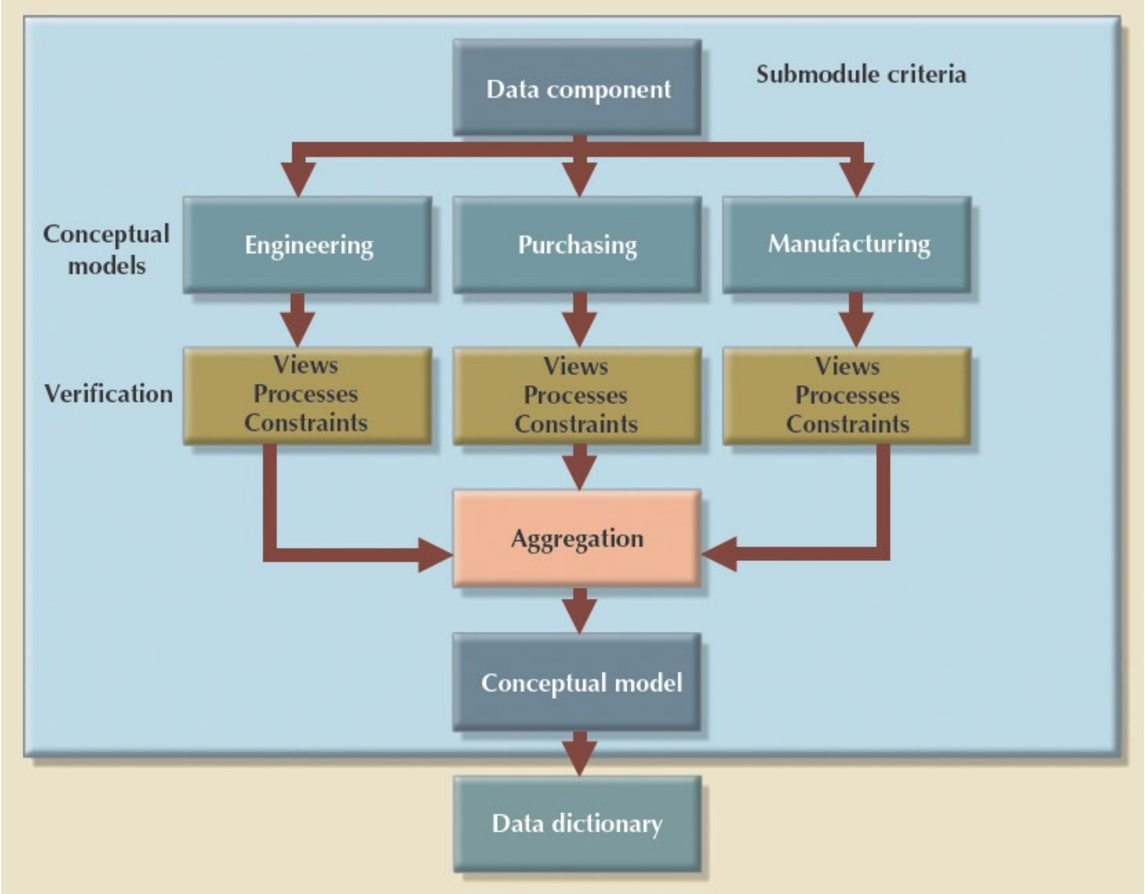
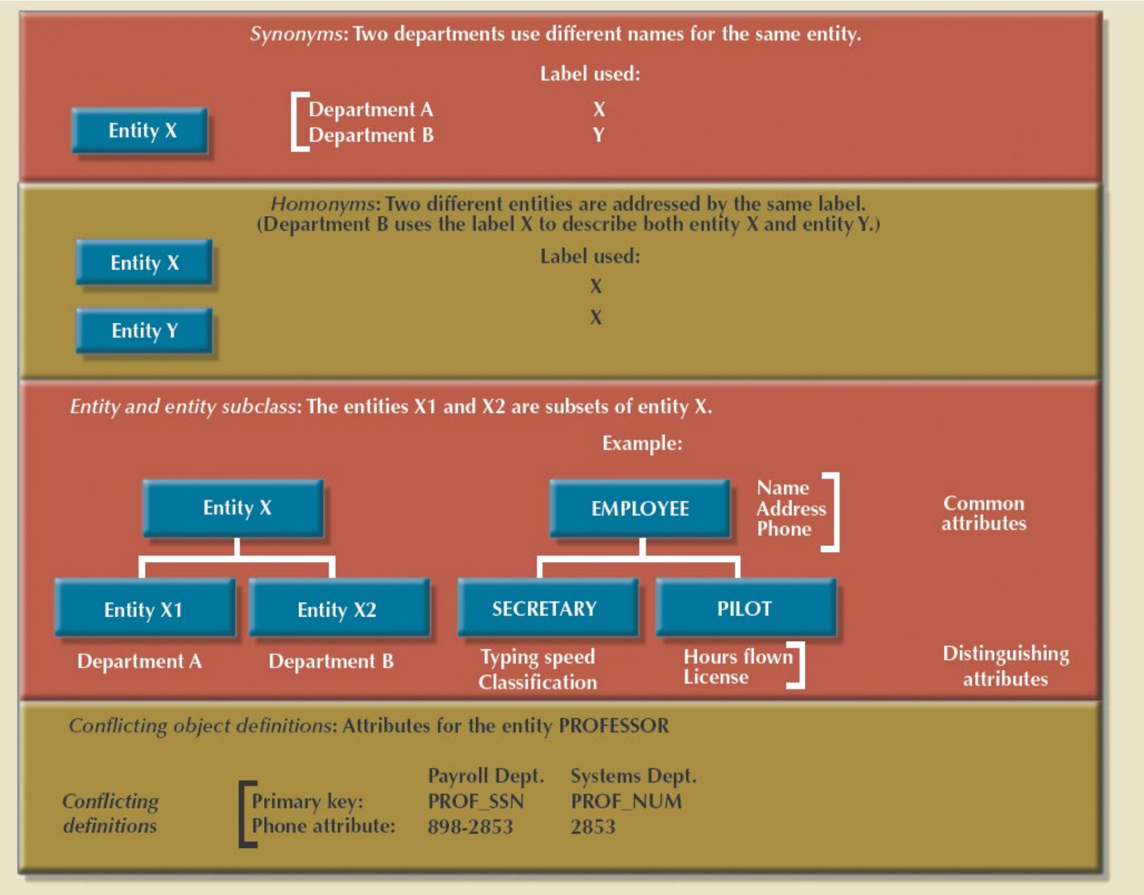


Figure 9.16 - Decentralized Design



Decentralized design might be used when the system's data component has a considerable number of entities and complex relations on which very complex operations are performed.

Figure 9.17 - Summary of Aggregation Problems



CENGAGE