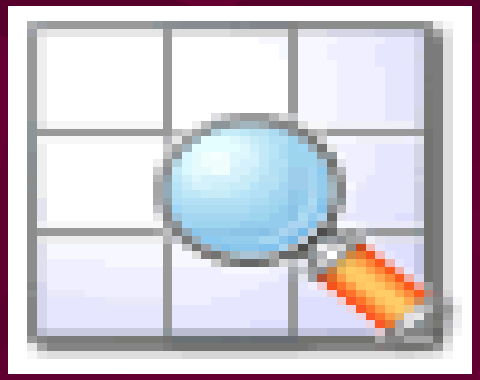
Database Design



The Information System

- 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

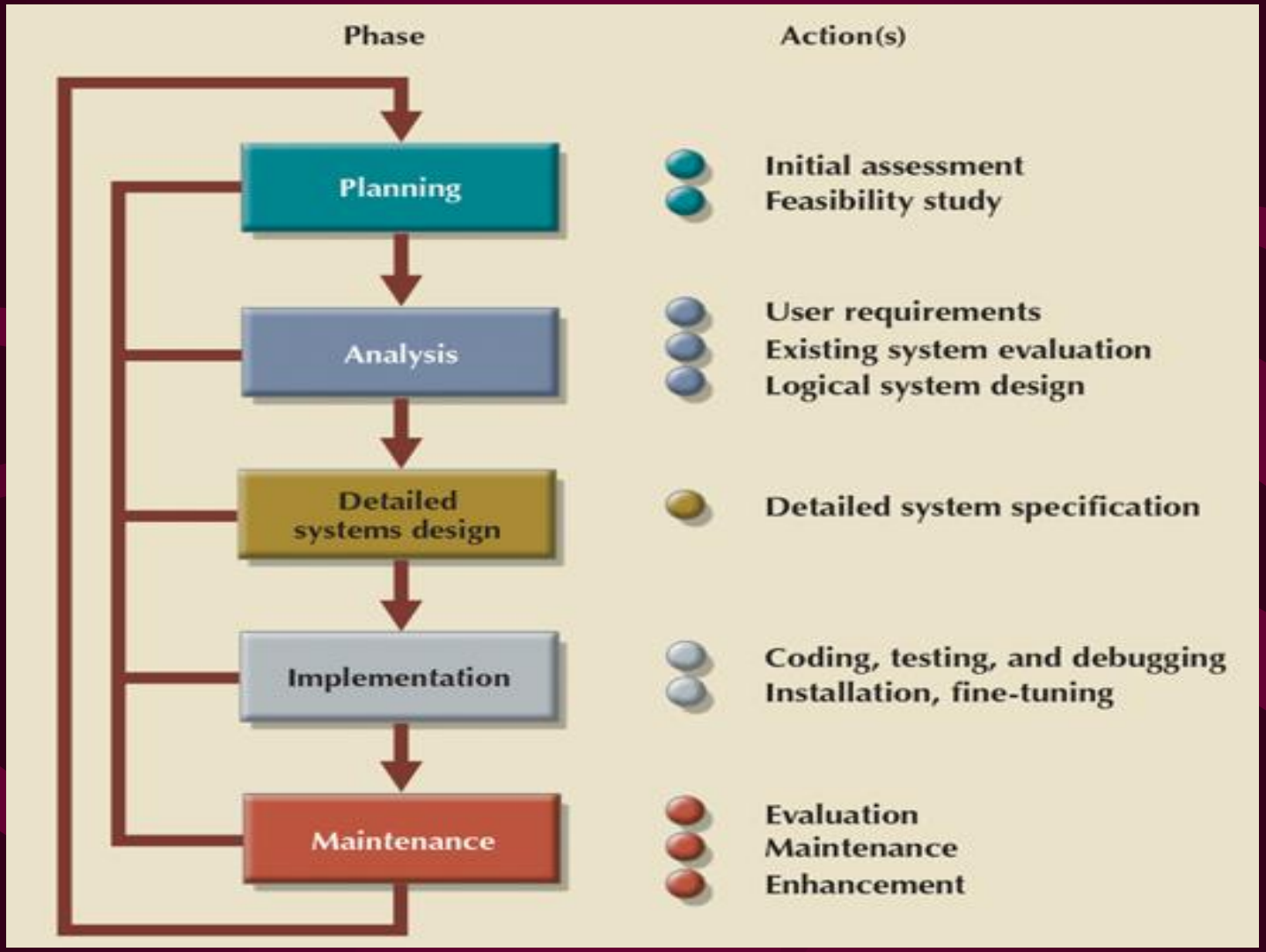
The Information System (con't)

- Performance factors of an information system
  - Database design and implementation
  - Application design and implementation
    - User interface
    - Business logic
  - Administrative procedures
    - Control and security
- Database development
  - Process of database design and its implementation

Systems Development Life Cycle (SDLC)

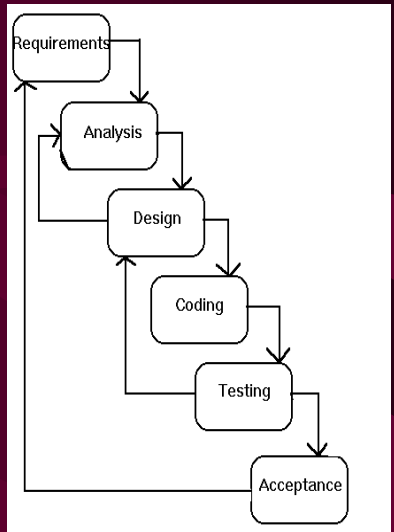
- Traces history of an information system
  - Provides a picture within which database design and application development are mapped out and evaluated
- Traditional SDLC is divided into several phases
  - Planning: yields a general overview of the company and its objectives
  - Analysis: problems defined during planning phase are examined in greater detail
  - Detailed systems design: designer completes the design of the system's processes
  - Implementation: hardware, DBMS software, and application programs are installed, and the database design is implemented
  - Maintenance: corrective, adaptive, and perfective

SDLC (con't)


Phase
Action(s)
Planning
Initial assessment
Feasibility study
Analysis
User requirements
Existing system evaluation
Logical system design
Detailed systems design
Detailed system specification
Implementation
Coding, testing, and debugging
Installation, fine-tuning
Maintenance
Evaluation
Maintenance
Enhancement

SDLC Pros & Cons

- Major advantages
  - Control
  - Accountability
  - Error detection
- Major drawbacks
  - Relatively inflexible
  - Time-consuming and expensive
  - Discourages changes once user requirements are done
  - Becomes unstable if the initial requirements are significantly in error or if they change much



The waterfall model (Systems Development Life Cycle)

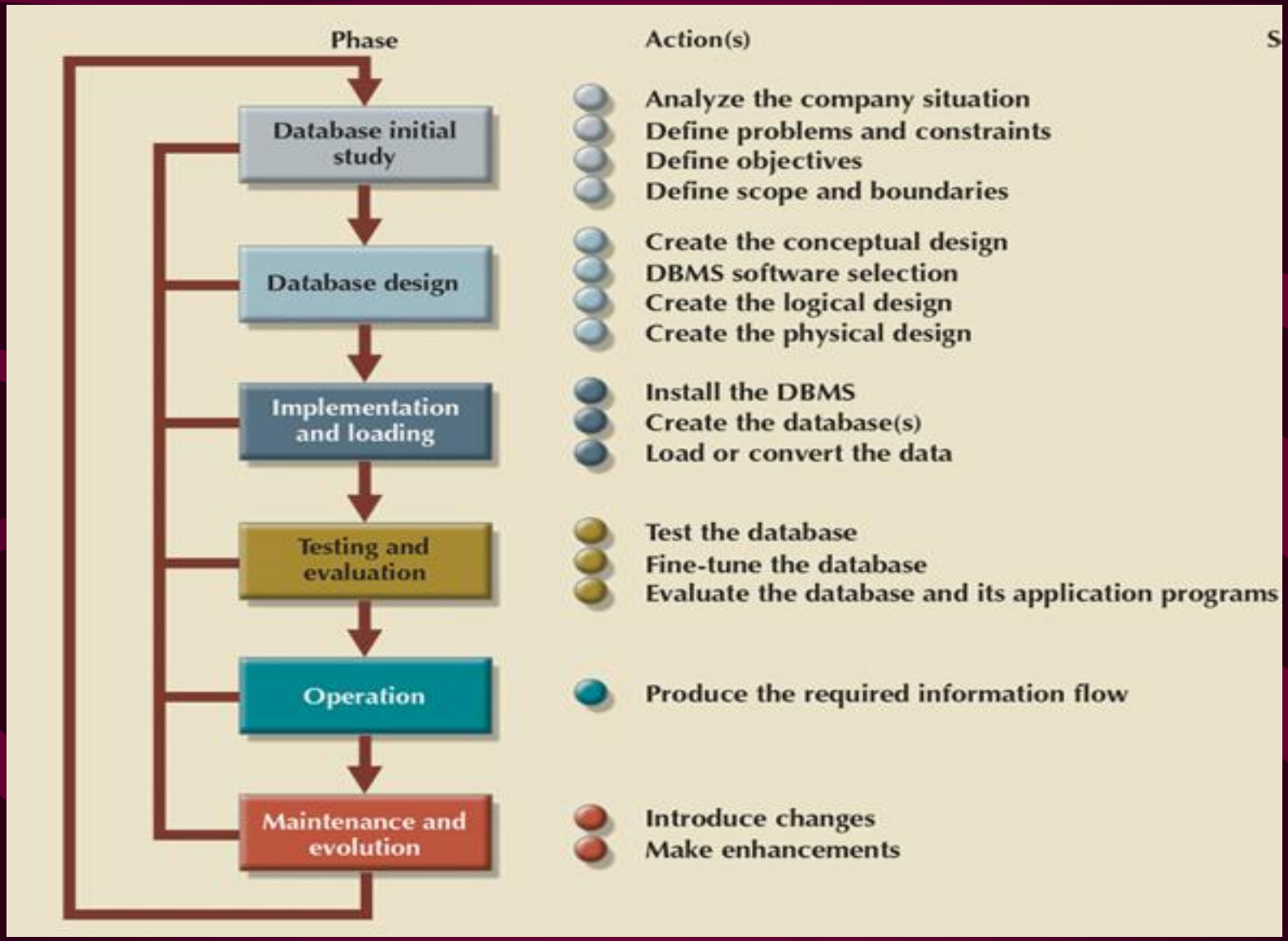
SDLC Pros & Cons (con't)

- Big IT projects are getting harder to complete successfully as evidenced by project success rates:
  - Projects over $10 million have success rates of only 2%
  - Projects between $3 and $10 million have success rates from 23% to 11%
  - Projects under $3 million have success rates from 33% to 46%
- Thus alternatives to the classical SDLC have been developed

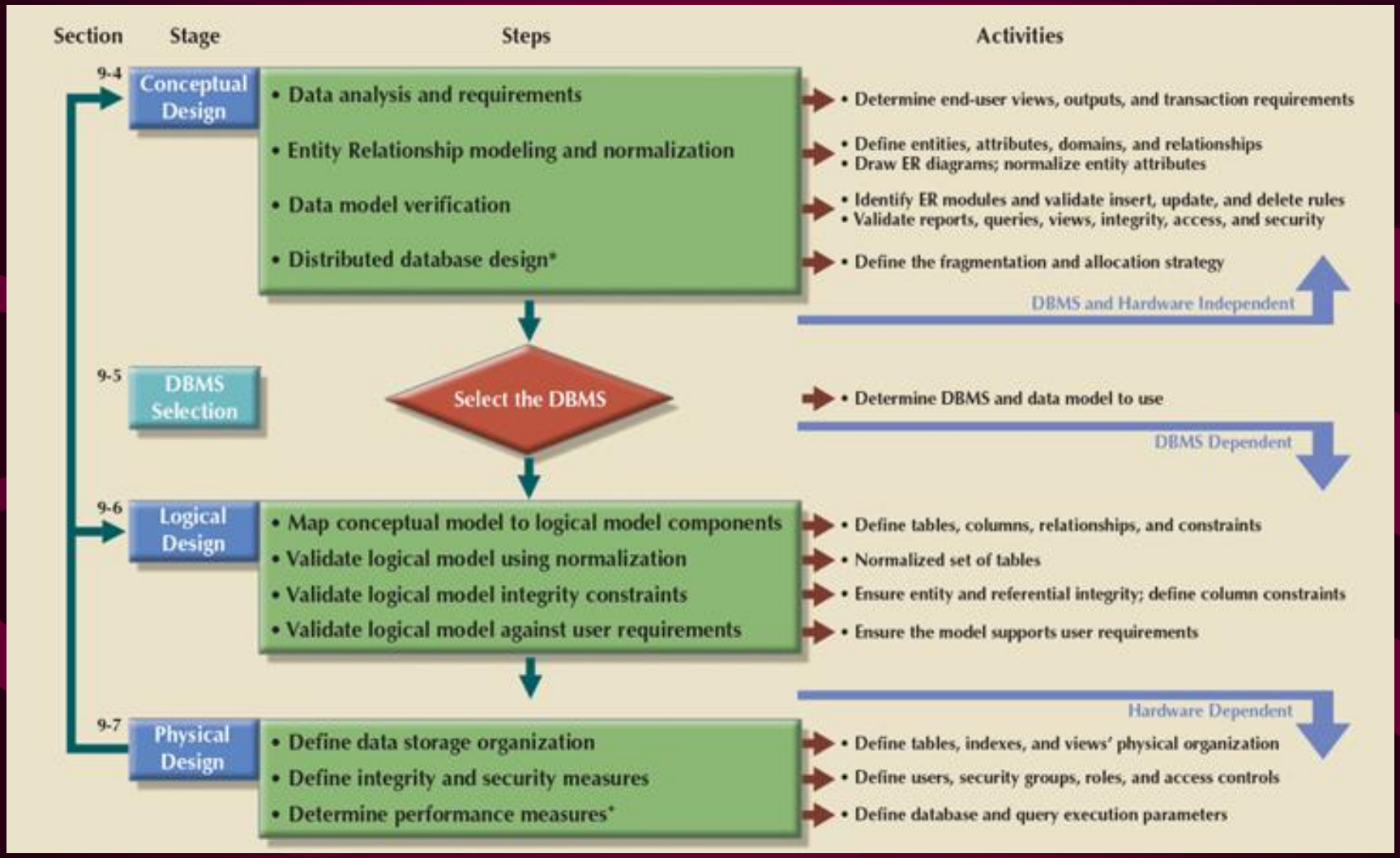
The Database Life Cycle

- 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 (con't)



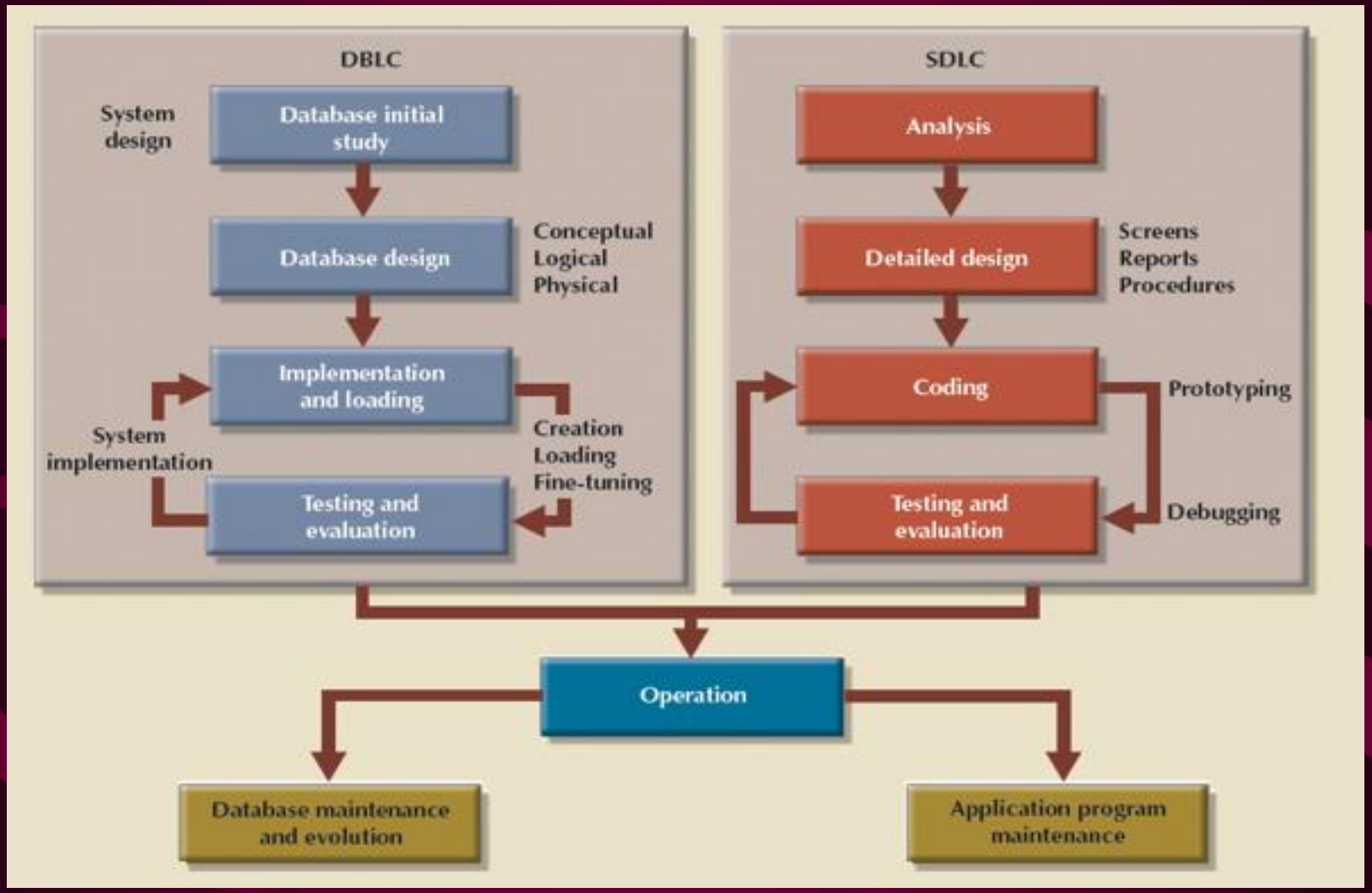
The Database Life Cycle (con't)



Sources of Database Failuers

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.

DBLC & SDLC


DBLC
System design
Database initial study
Database design
Conceptual
Logical
Physical
Implementation and loading
System implementation
Creation
Loading
Fine-tuning
Testing and evaluation
SDLC
Analysis
Detailed design
Screens
Reports
Procedures
Coding
Prototyping
Testing and evaluation
Debugging
Operation
Database maintenance and evolution
Application program maintenance

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 Steps

Step	Activity
1	Data analysis and requirements
2	Entity relationship modeling and normalization
3	Data model verification
4	Distributed database design

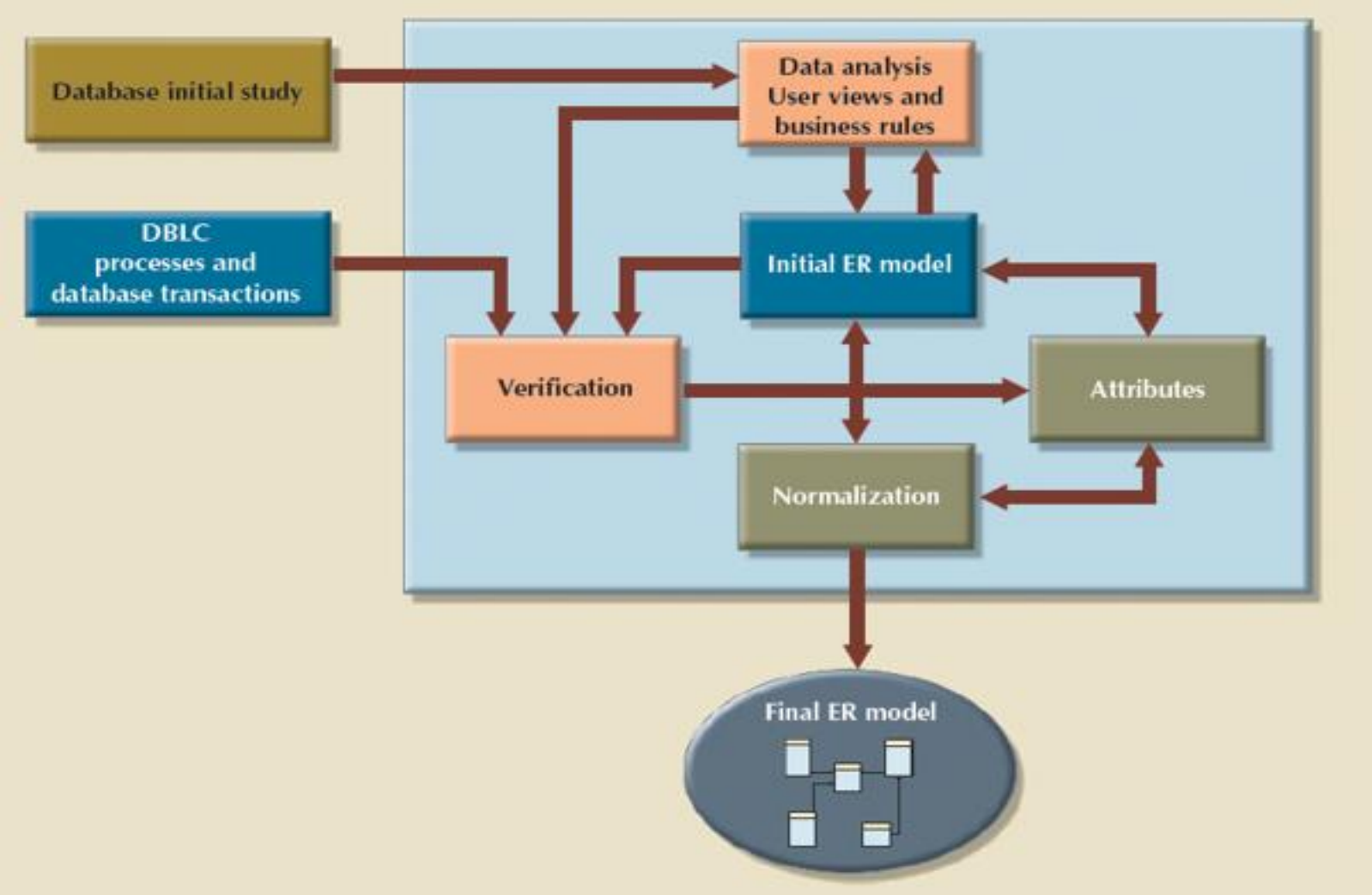
Conceptual Design Steps (con't)

- 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

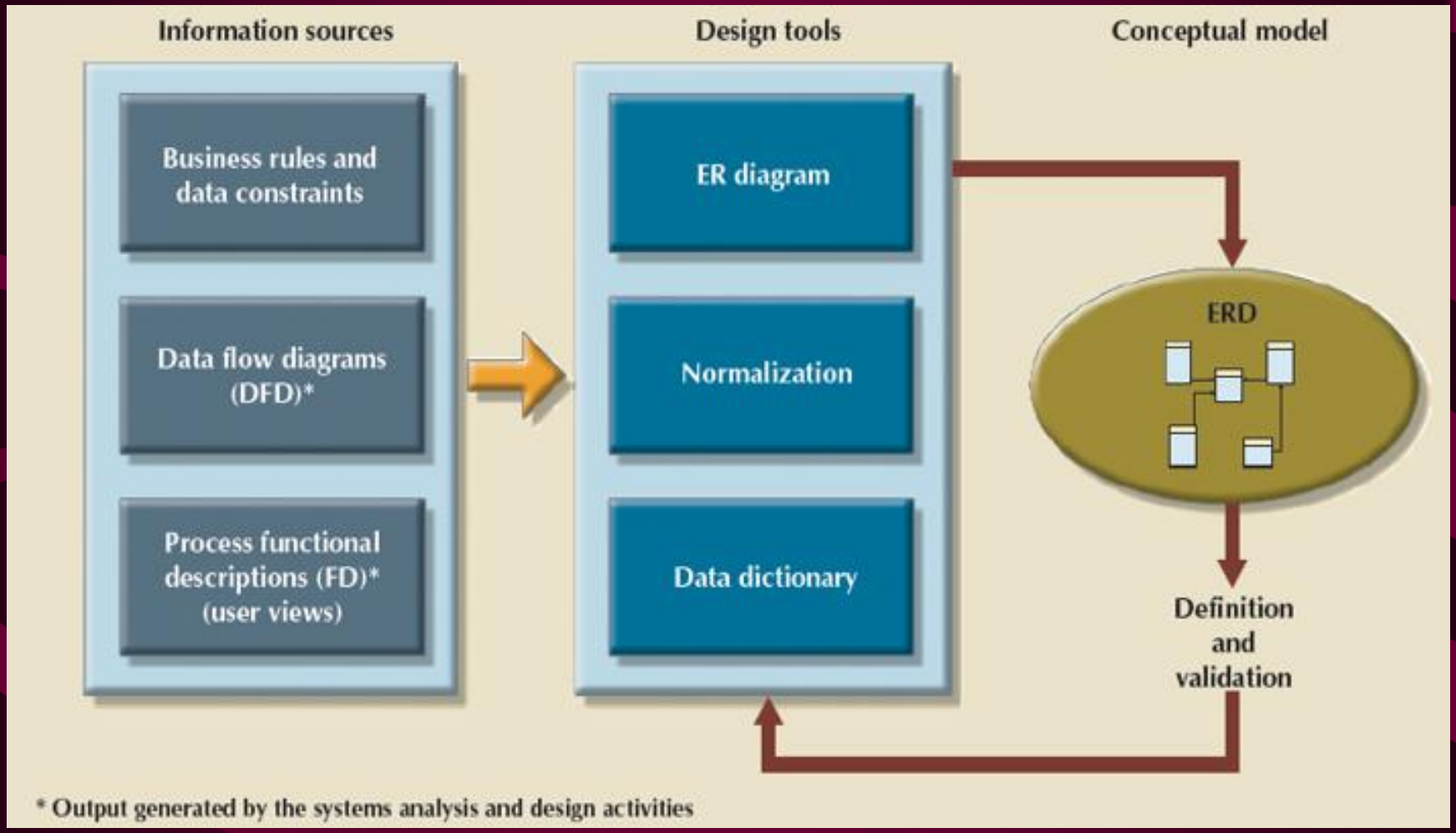
Developing the Conceptual Model via ER

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

Iterative ER Model Process



Conceptual Design Tools and Info

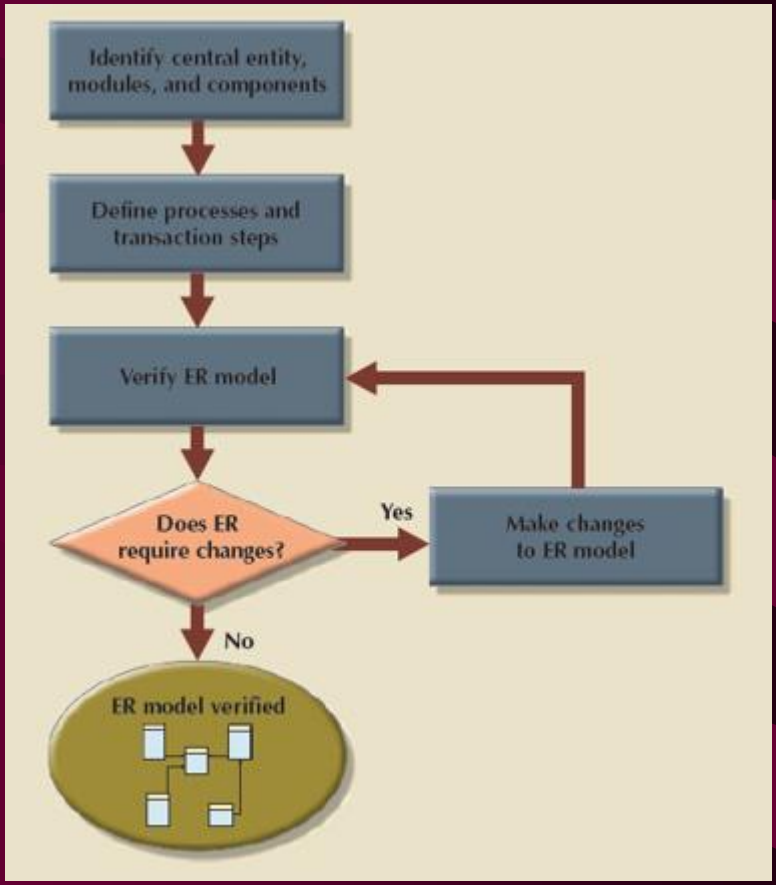


Conceptual Design Steps (con't)

- 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

ER Verification

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



DBMS Software Selection

- Factors that affect the purchasing
  - Cost
  - Size of the database (# of entities and # of each entity instances)
  - DBMS features and tools
  - IS staff skill level
  - 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

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

Mapping Conceptual Model to Relational Model

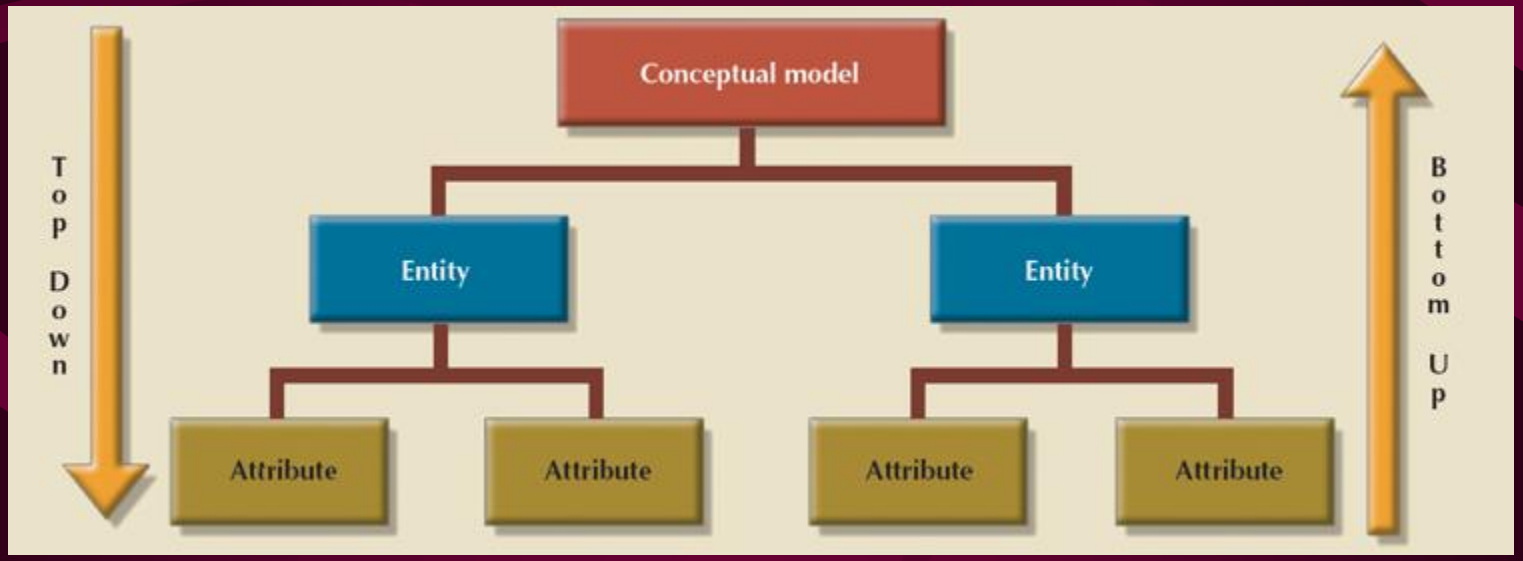
Step	Activity
1	Map strong entities
2	Map supertype/subtype relationships
3	Map weak entities
4	Map binary relationships
5	Map higher-degree relationships

Physical Design

- Process of data storage organization and data access characteristics of the database; ensures integrity, security, and performance
  - Define data storage organization
    - Allocate physical disk units to tables and indices
  - Define integrity and security measures
  - Determine performance measures

Database Design Strategies

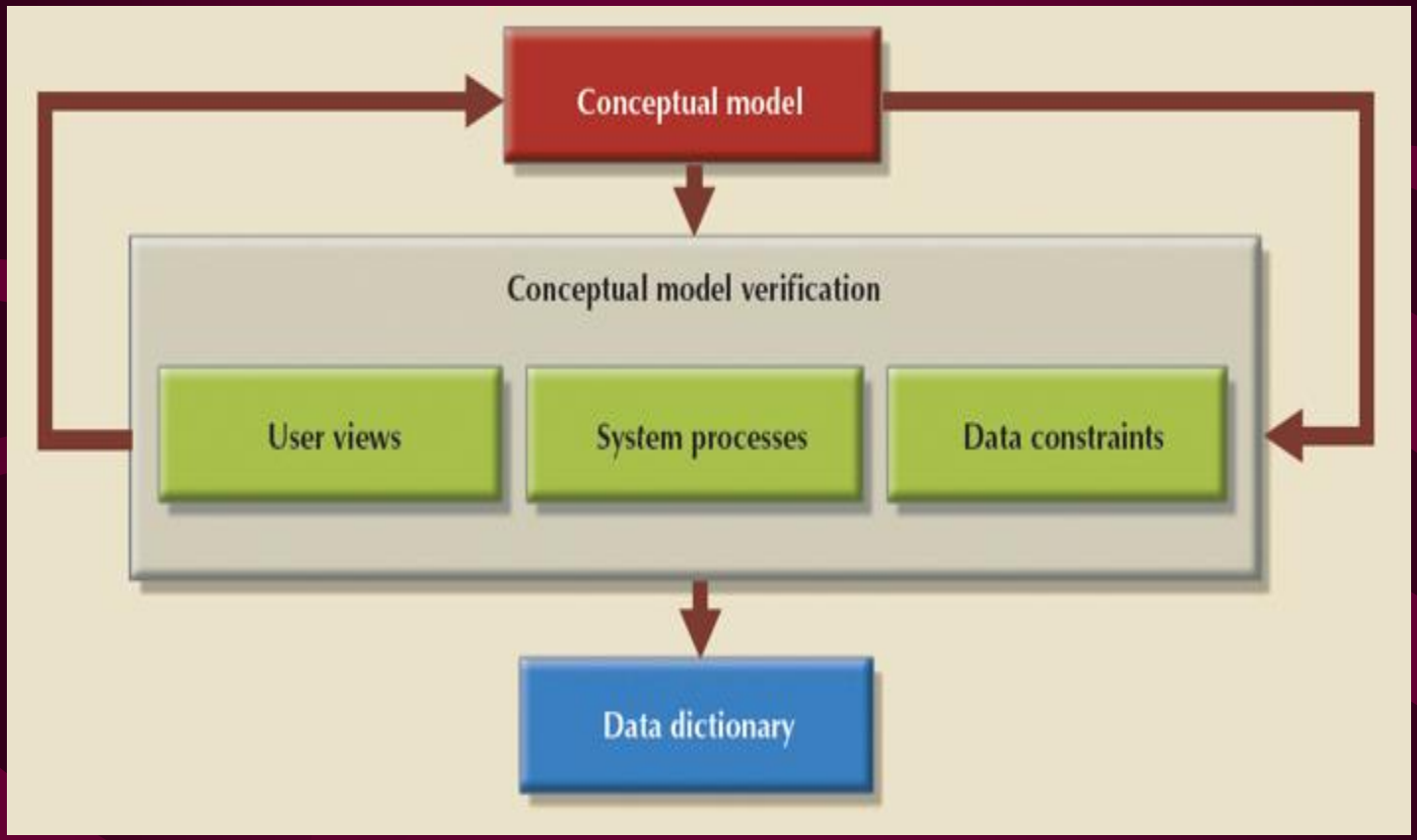
- Top-down design starts by identifying the data sets and then defines the data elements for each of those sets
  - Involves the identification of different entity types and the definition of each entity's attributes
- Bottom-up design first identifies the data elements (items) and then groups them together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



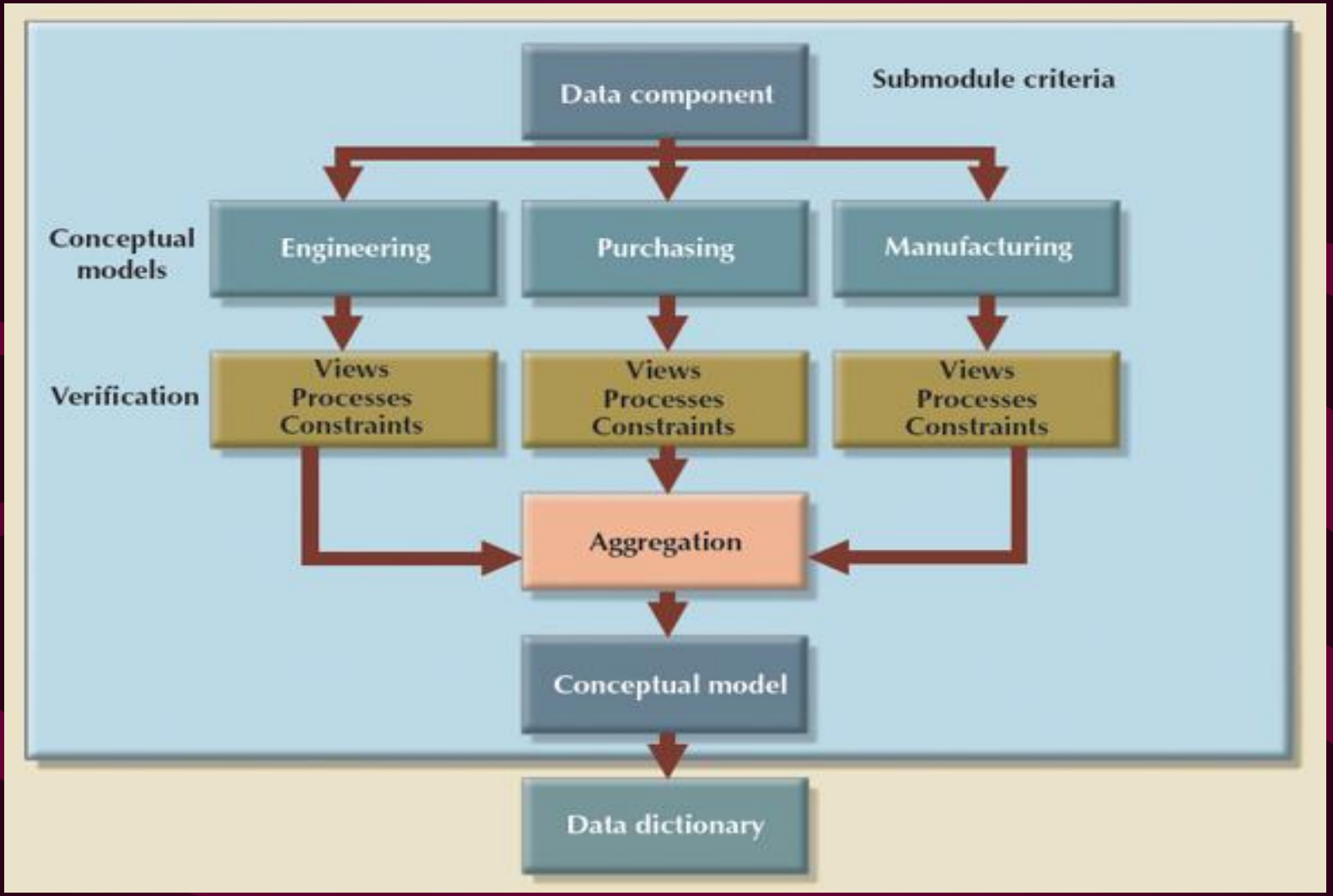
Centralized versus Decentralized Design

- 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

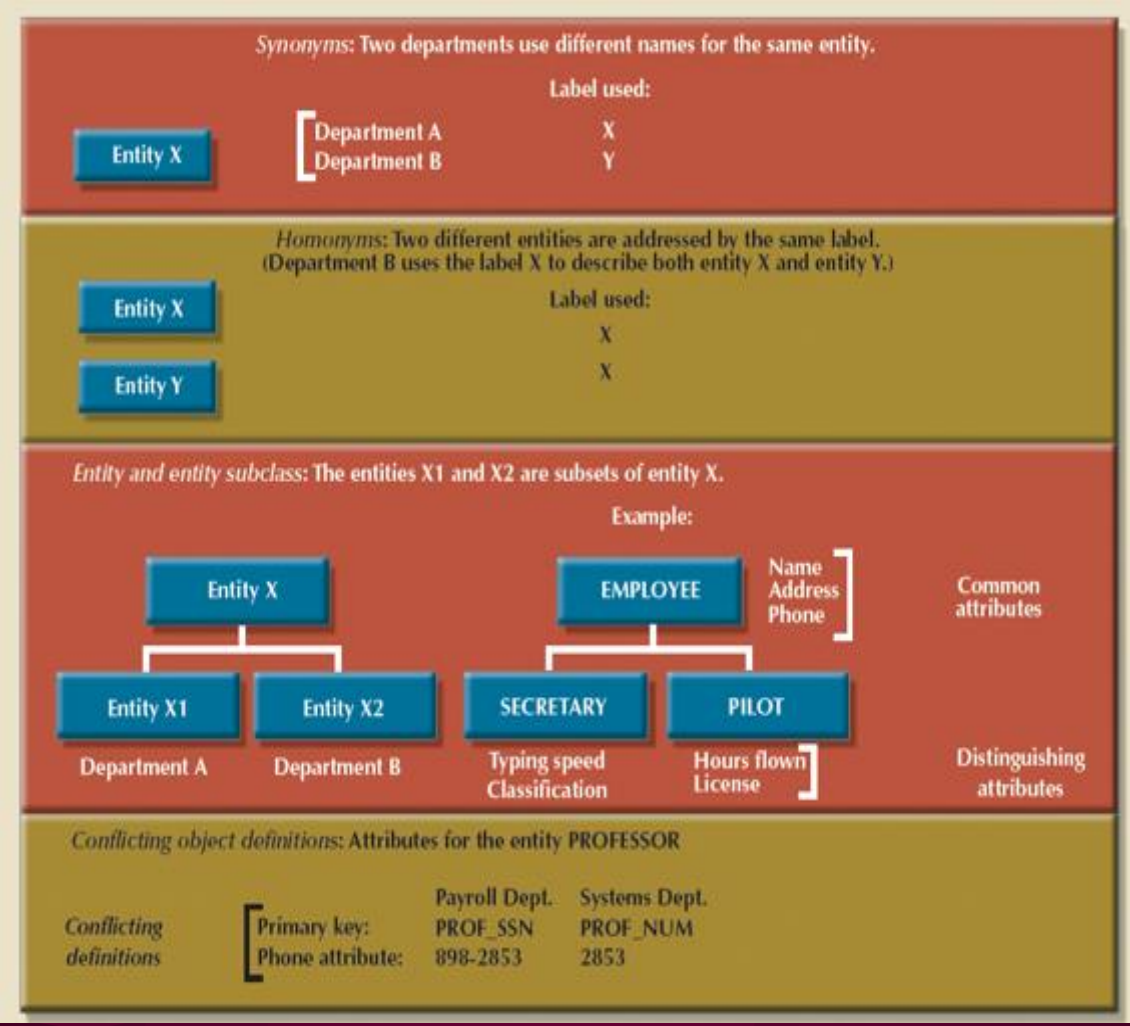
Centralized Design



Decentralized Design



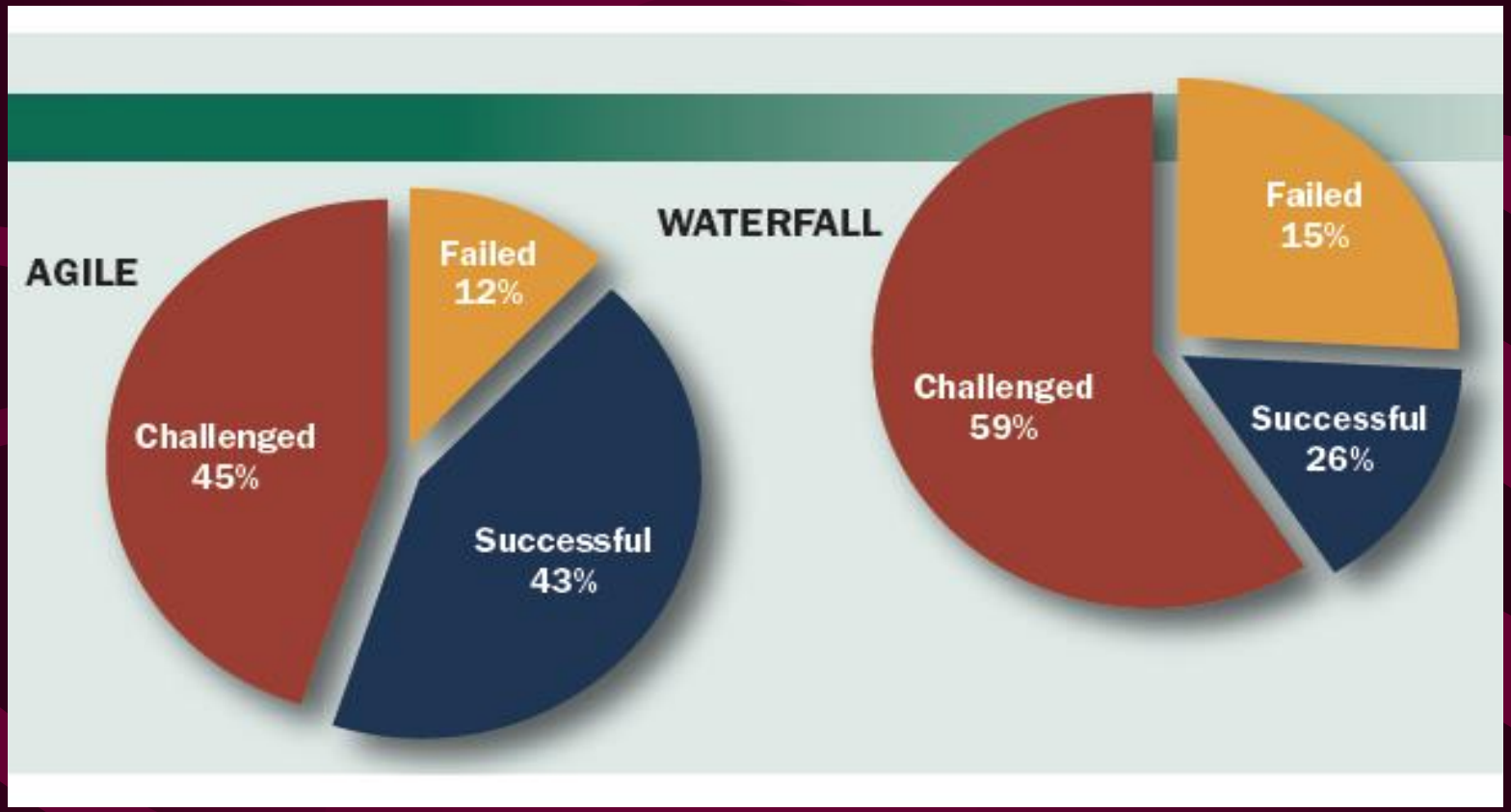
Aggregation Problem



Agile Methods

- Agile Methods address the need to build systems faster
  - Are the newest development
  - Emphasize continuous feedback
  - Iterative development
  - Based upon the "Agile Manifesto"
  - Iterative (Spiral) model
  - Examples are Scrum and Extreme Programming (XP)

Agile vs Waterfall (SDLC)

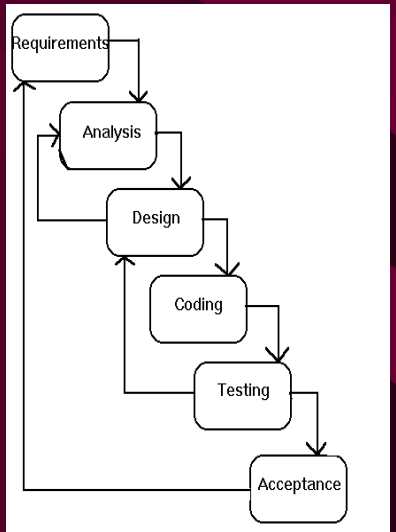


References

- Designing Data-Intensive Applications: The Big Ideas Behind Reliable, Scalable, and Maintainable Systems by Martin Kleppmann
- Systems Analysis and Design (Shelly Cashman Series) by Scott Tilley and Harry J. Rosenblat
- Systems Analysis and Design (MindTap Course List) by Scott Tilley

Homework

- Textbook Chapter 9
- Review questions 1 thru 9



The waterfall model (Systems Development Life Cycle)