THE UNIVERSITY OF WESTERN AUSTRALIA

# CITS1402 Relational Database Management Systems

## Week 6—System Development Life Cycle

Dr Mehwish Nasim/ CSSE
cits1402-sem1-pmc@uwa.edu.au

Sem1/2024

CITS1402 Week6

# Contents

Main stages of database system development lifecycle.

Main phases of database design: conceptual, logical, and physical design.

Benefits of CASE tools.

Distinction between data administration and database administration.

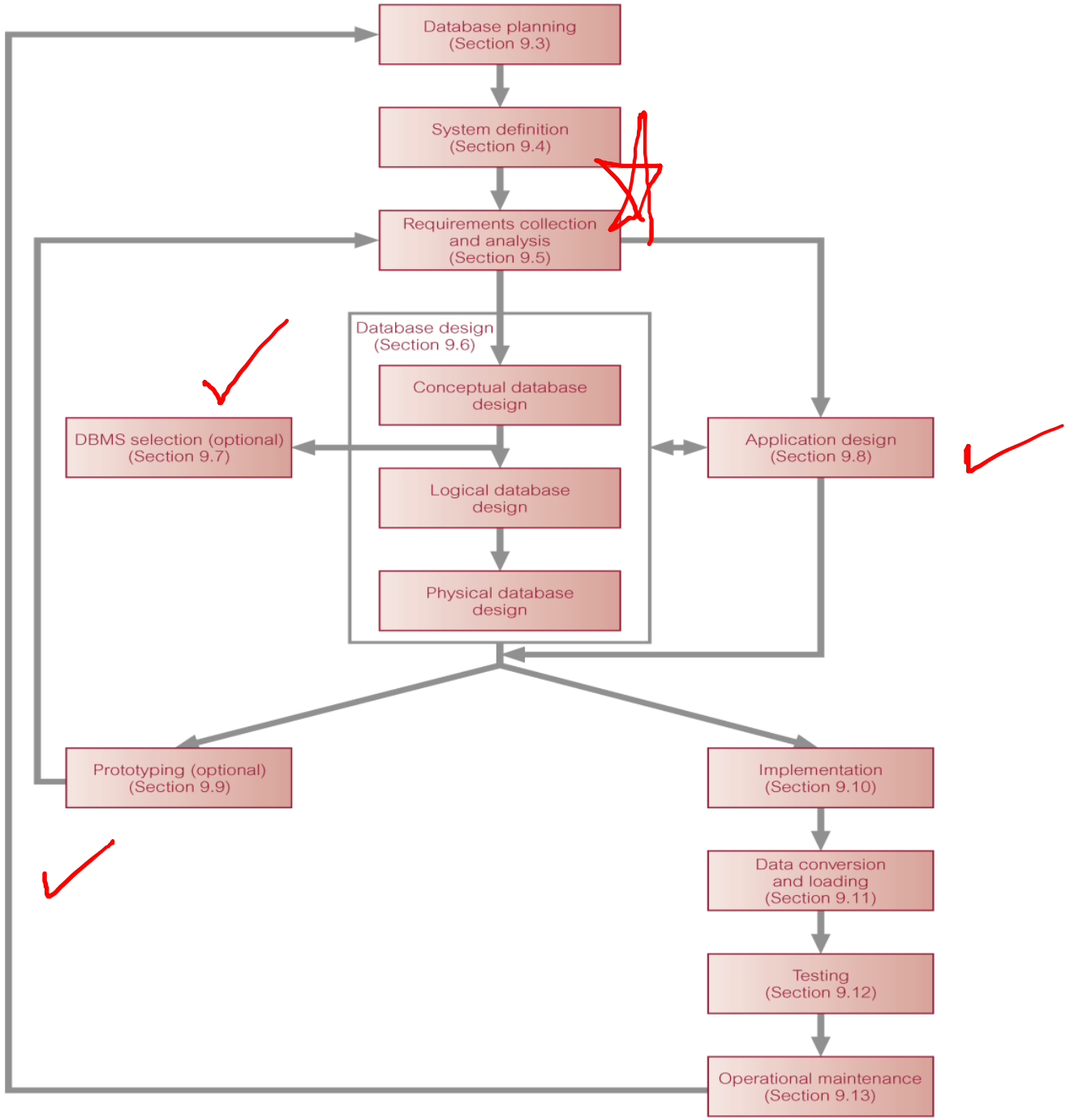
# Chapter 10 - Objectives

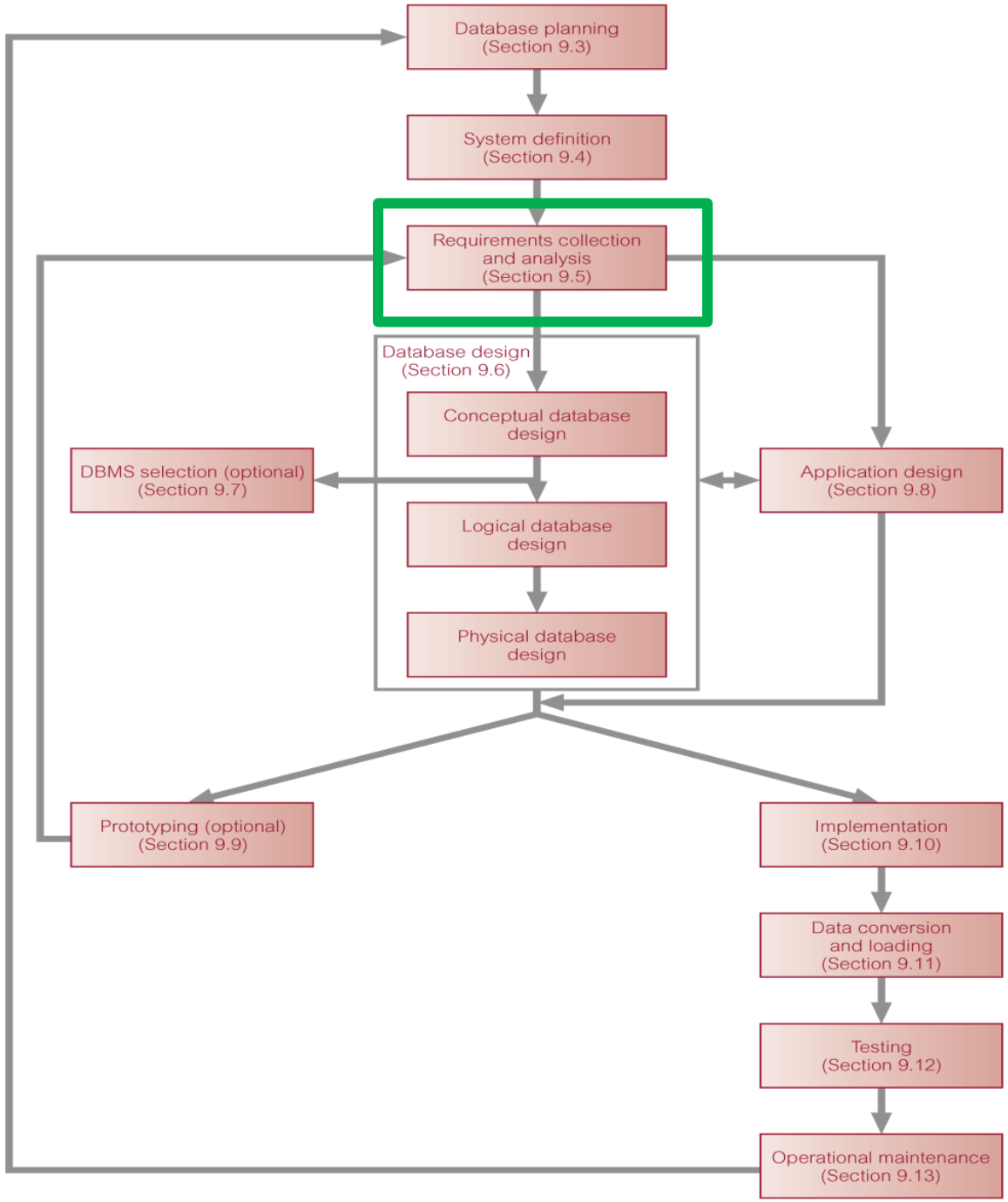
Main stages of database system development lifecycle.

Main phases of database design: conceptual, logical, and physical design.

Benefits of CASE tools.

Distinction between data administration and database administration.


Database planning
(Section 9.3)
System definition
(Section 9.4)
Requirements collection
and analysis
(Section 9.5)
Database design
(Section 9.6)
Conceptual database
design
DBMS selection (optional)
(Section 9.7)
Logical database
design
Physical database
design
Application design
(Section 9.8)
Prototyping (optional)
(Section 9.9)
Implementation
(Section 9.10)
Data conversion
and loading
(Section 9.11)
Testing
(Section 9.12)
Operational maintenance
(Section 9.13)


Database planning
(Section 9.3)
System definition
(Section 9.4)
Requirements collection
and analysis
(Section 9.5)
Database design
(Section 9.6)
Conceptual database
design
DBMS selection (optional)
(Section 9.7)
Logical database
design
Physical database
design
Application design
(Section 9.8)
Prototyping (optional)
(Section 9.9)
Implementation
(Section 9.10)
Data conversion
and loading
(Section 9.11)
Testing
(Section 9.12)
Operational maintenance
(Section 9.13)

# Requirements Collection and Analysis

Information is gathered for each major user view including:

- a description of data used or generated;
- details of how data is to be used/generated;
- any additional requirements for new database system.

Information is analyzed to identify requirements to be included in new database system.

Described in the requirements specification.

SRS

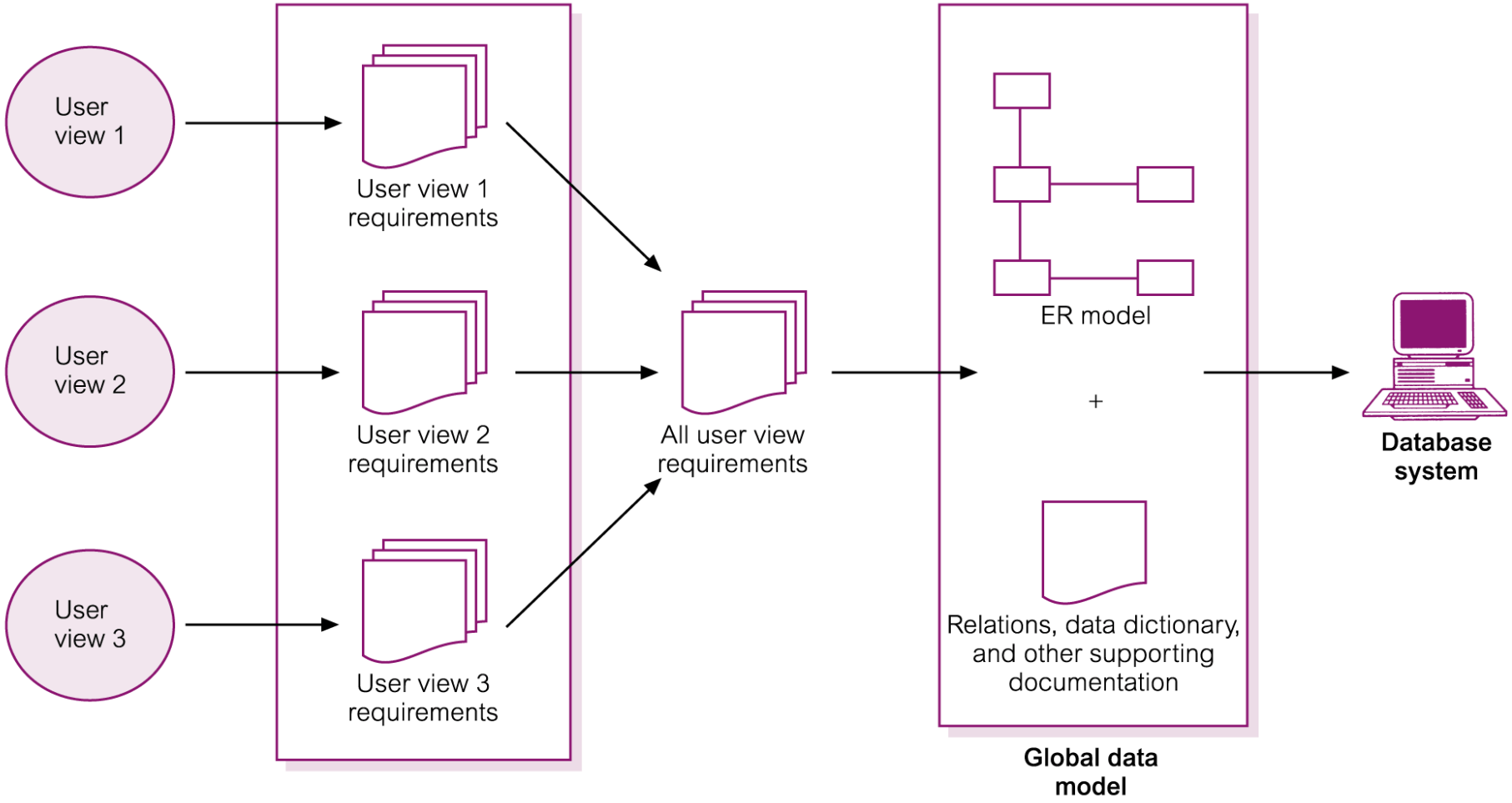
# Requirements Collection and Analysis

Another important activity is deciding how to manage the requirements for a database system with multiple user views.

Three main approaches:

- centralized approach;
- view integration approach;
- combination of both approaches.

# Centralized Approach to Managing Multiple User Views

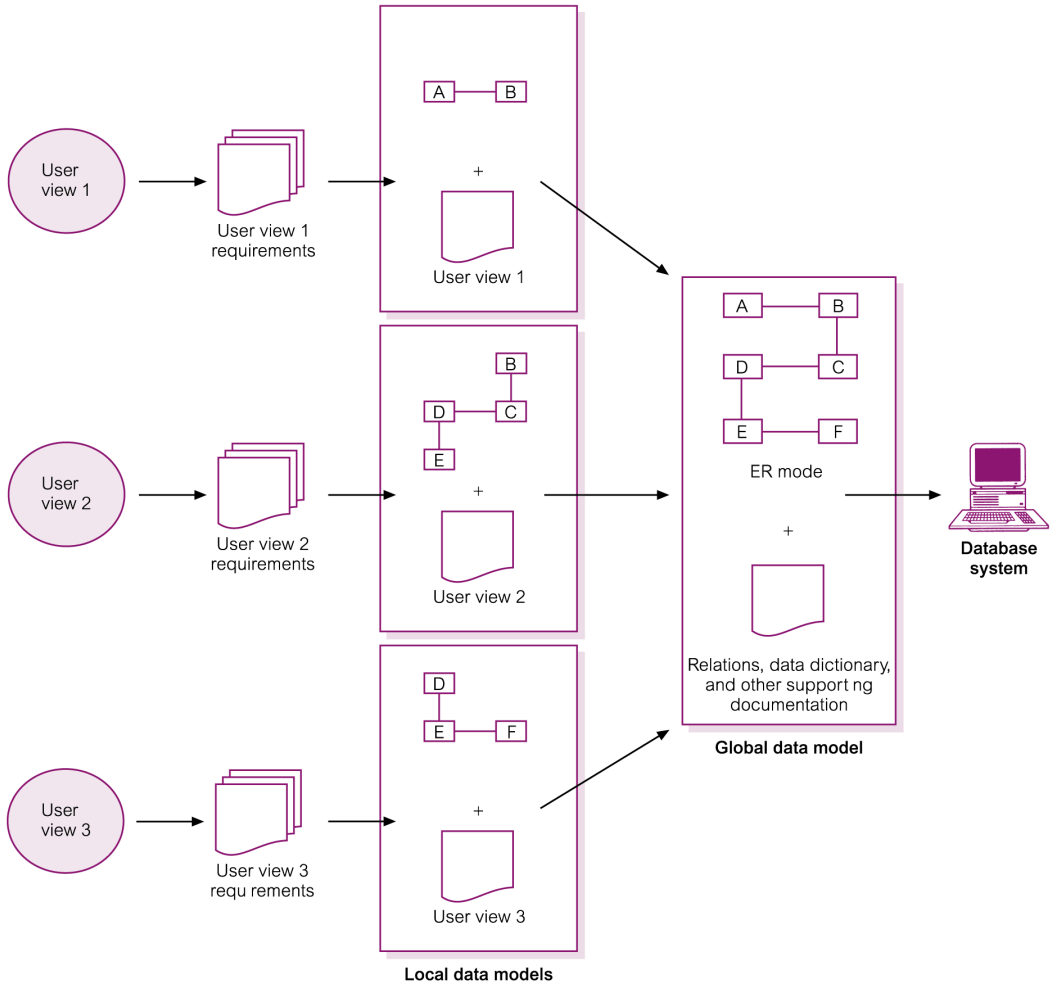


# Requirements Collection and Analysis

Centralized approach

- Requirements for each user view are merged into a single set of requirements.
- A data model is created representing all user views during the database design stage.

# View Integration Approach to Managing Multiple User Views



# Requirements Collection and Analysis

View integration approach

Data model representing single user view (or a subset of all user views) is called a local data model.

Each model includes diagrams and documentation describing requirements for one or more but not all user views of  database.

Local data models are then merged at a later stage during database design to produce a global data model, which represents all user views for the database.

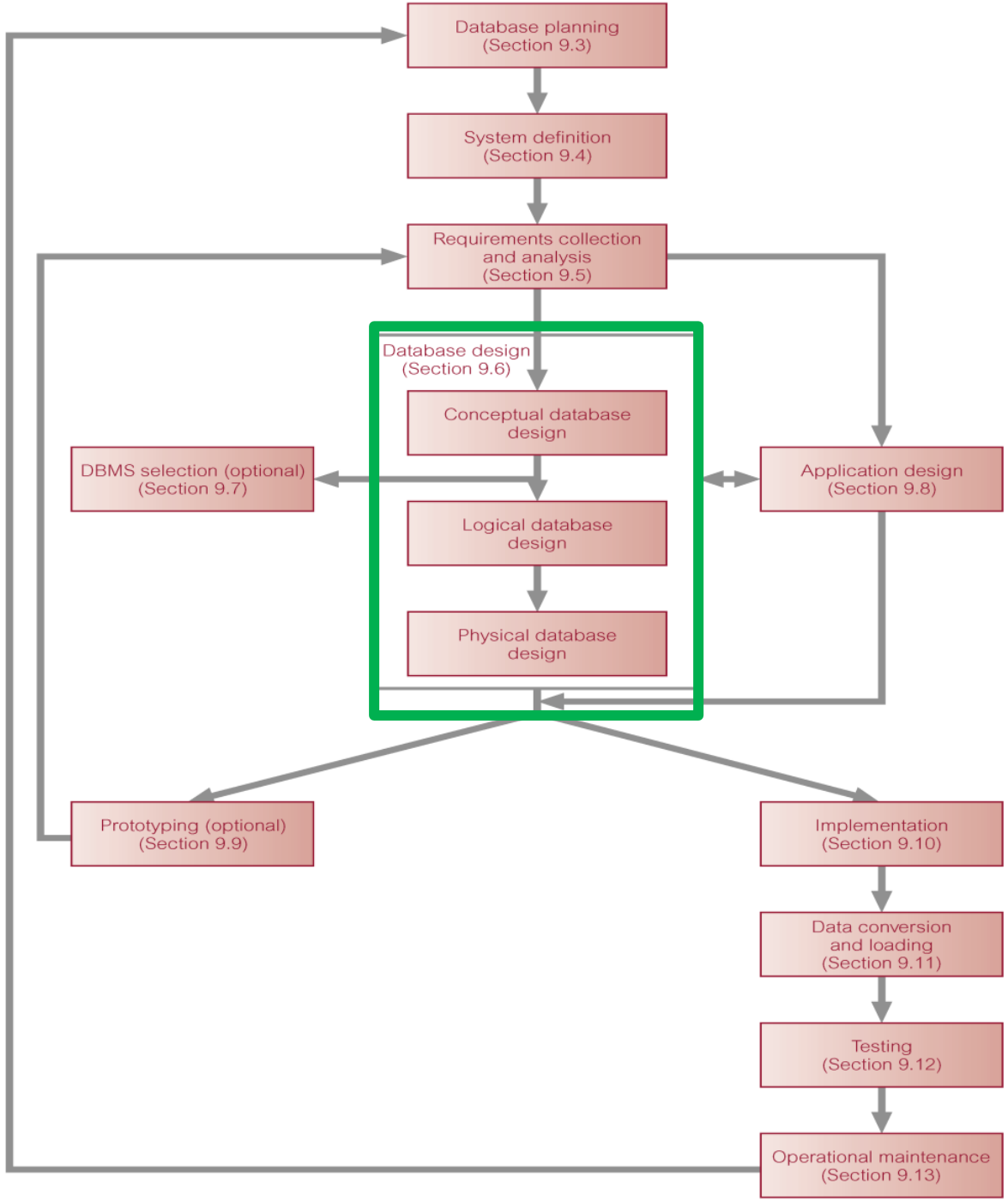
# Chapter 10 - Objectives

Main stages of database system development lifecycle.

Main phases of database design: conceptual, logical, and physical design.

Benefits of CASE tools.

Distinction between data administration and database administration.


Database planning
(Section 9.3)
System definition
(Section 9.4)
Requirements collection
and analysis
(Section 9.5)
Database design
(Section 9.6)
Conceptual database
design
DBMS selection (optional)
(Section 9.7)
Logical database
design
Physical database
design
Application design
(Section 9.8)
Prototyping (optional)
(Section 9.9)
Implementation
(Section 9.10)
Data conversion
and loading
(Section 9.11)
Testing
(Section 9.12)
Operational maintenance
(Section 9.13)

# Database Design

Process of creating a design for a database that will support the enterprise's mission statement and mission objectives for the required database system.

Main approaches include:

- Top-down – entities and relationships
- Bottom-up – attributes first
- Inside-out – major entities first
- Mixed

# Database Design

Main purposes of data modeling include:

- to assist in understanding the meaning (semantics) of the data;
- to facilitate communication about the information requirements.

Building data model requires answering questions about entities, relationships, and attributes.

# Database Design

A data model ensures we understand:

- each user’s perspective on the data;
- nature of the data itself, independent of its physical representations;
- use of data across user views.

# Criteria to Produce an Optimal Data Model

| | |
|---|---|
| *Structural validity* | Consistency with the way the enterprise defines and organizes information. |
| *Simplicity* | Ease of understanding by IS professionals and non-technical users. |
| *Expressibility* | Ability to distinguish between different data, relationships between data, and constraints. |
| *Nonredundancy* | Exclusion of extraneous information; in particular, the representation of any one piece of information exactly once. |
| *Shareability* | Not specific to any particular application or technology and thereby usable by many. |
| *Extensibility* | Ability to evolve to support new requirements with minimal effect on existing users. |
| *Integrity* | Consistency with the way the enterprise uses and manages information. |
| *Diagrammatic representation* | Ability to represent a model using an easily understood diagrammatic notation. |

# Database Design

Three phases of database design:

- Conceptual database design
- Logical database design
- Physical database design

# Database Design

Three phases of database design:

- Conceptual database design
- Logical database design
- Physical database design

# Conceptual Database Design

Process of constructing a model of the data used in an enterprise, independent of all physical considerations.

Data model is built using the information in  users' requirements specification.

Conceptual data model is source of information for logical design phase.

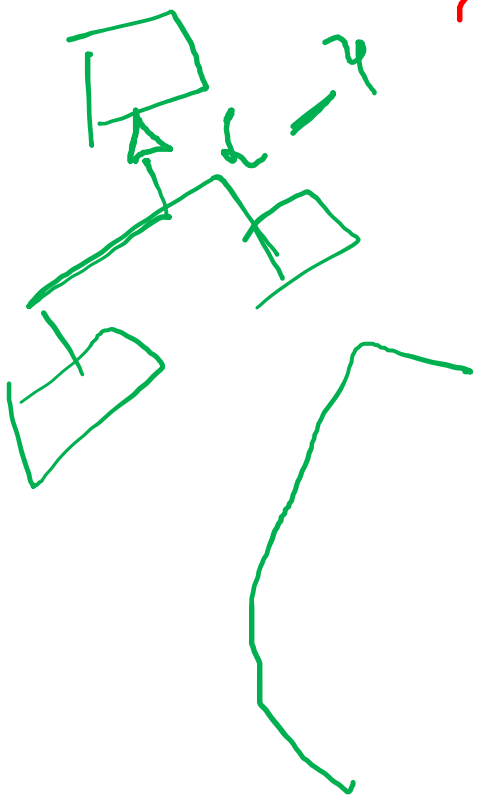
Composite

Email {1..3}

/attr.

foreign key?

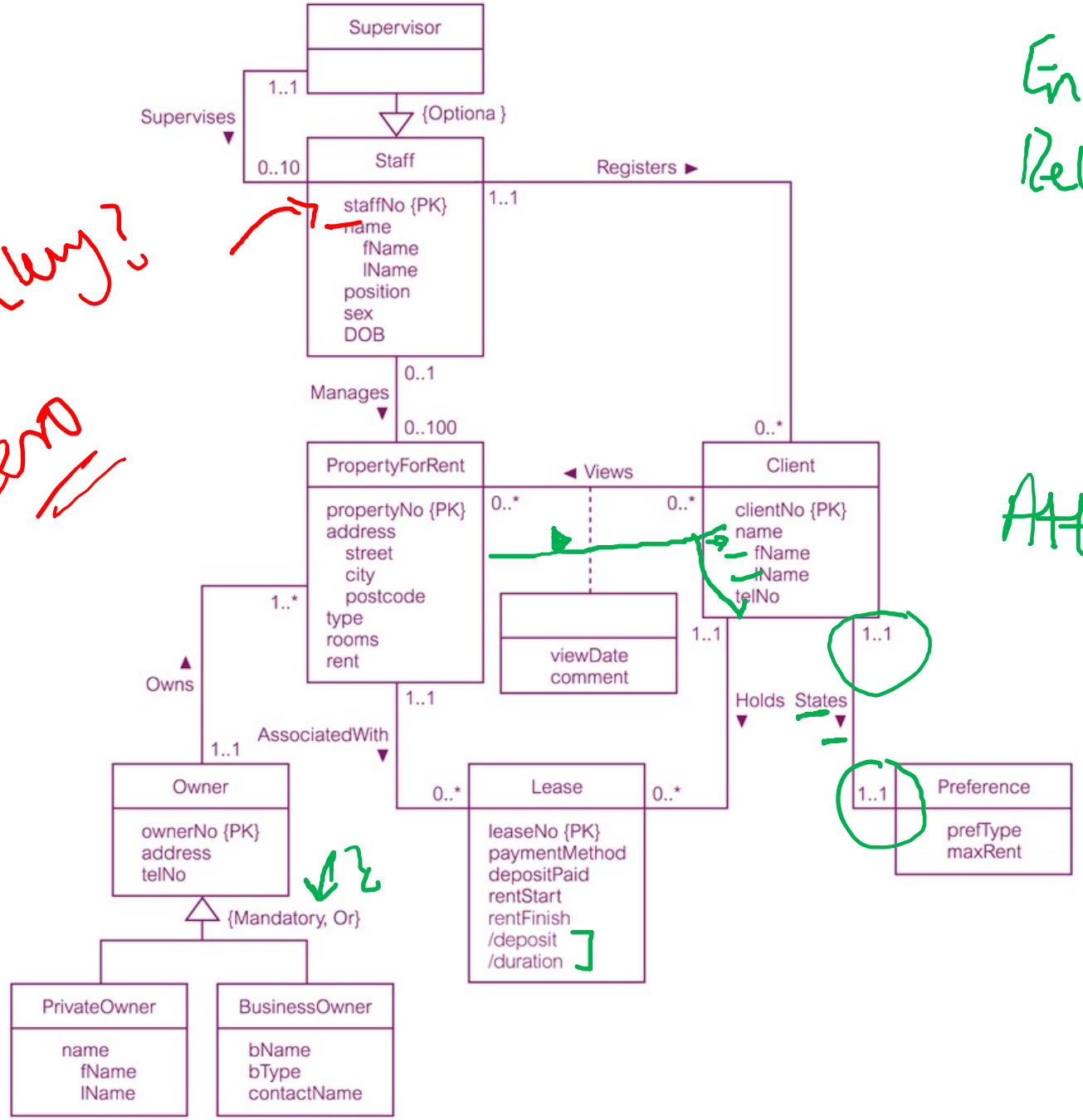
zero



Entities — Name, EER

Relationships — Attr. Rel., Name, Multiplicity, Direction

Attr — Name, Types
- composite
- Multivalued
- Derived



# Logical Database Design

Process of constructing a model of the data used in an enterprise based on a specific data model (e.g. relational)

But independent of a particular DBMS and other physical considerations.

Conceptual data model is refined and mapped on to a logical data model.

# Relations for the Staff user views of *DreamHome*

| | |
|---|---|
| **Staff** (staffNo, fName, lName, position, sex, DOB, supervisorStaffNo)<br>**Primary Key** staffNo<br>**Foreign Key** supervisorStaffNo **references** Staff(staffNo) | **PrivateOwner** (ownerNo, fName, lName, address, telNo)<br>**Primary Key** ownerNo |
| **BusinessOwner** (ownerNo, bName, bType, contactName, address, telNo)<br>**Primary Key** ownerNo<br>**Alternate Key** bName<br>**Alternate Key** telNo | **Client** (clientNo, fName, lName, telNo, prefType, maxRent, staffNo)<br>**Primary Key** clientNo<br>**Foreign Key** staffNo **references** Staff(staffNo) |
| **PropertyForRent** (propertyNo, street, city, postcode, type, rooms, rent, ownerNo, staffNo)<br>**Primary Key** propertyNo<br>**Foreign Key** ownerNo **references** PrivateOwner(ownerNo) and BusinessOwner(ownerNo)<br>**Foreign Key** staffNo **references** Staff(staffNo) | **Viewing** (clientNo, propertyNo, dateView, comment)<br>**Primary Key** clientNo, propertyNo<br>**Foreign Key** clientNo **references** Cl ent(clientNo)<br>**Foreign Key** propertyNo **references** PropertyForRent(propertyNo) |
| **Lease** (leaseNo, paymentMethod, depositPaid, rentStart, rentFinish, clientNo, propertyNo)<br>**Primary Key** leaseNo<br>**Alternate Key** propertyNo, rentStart<br>**Alternate Key** clientNo, rentStart<br>**Foreign Key** clientNo **references** Client(clientNo)<br>**Foreign Key** propertyNo **references** PropertyForRent(propertyNo)<br>**Derived** deposit (PropertyForRent.rent*2)<br>**Derived** duration (rentFinish – rentStart) | |

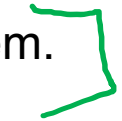
# Physical Database Design

Process of producing a description of the database implementation on secondary storage.

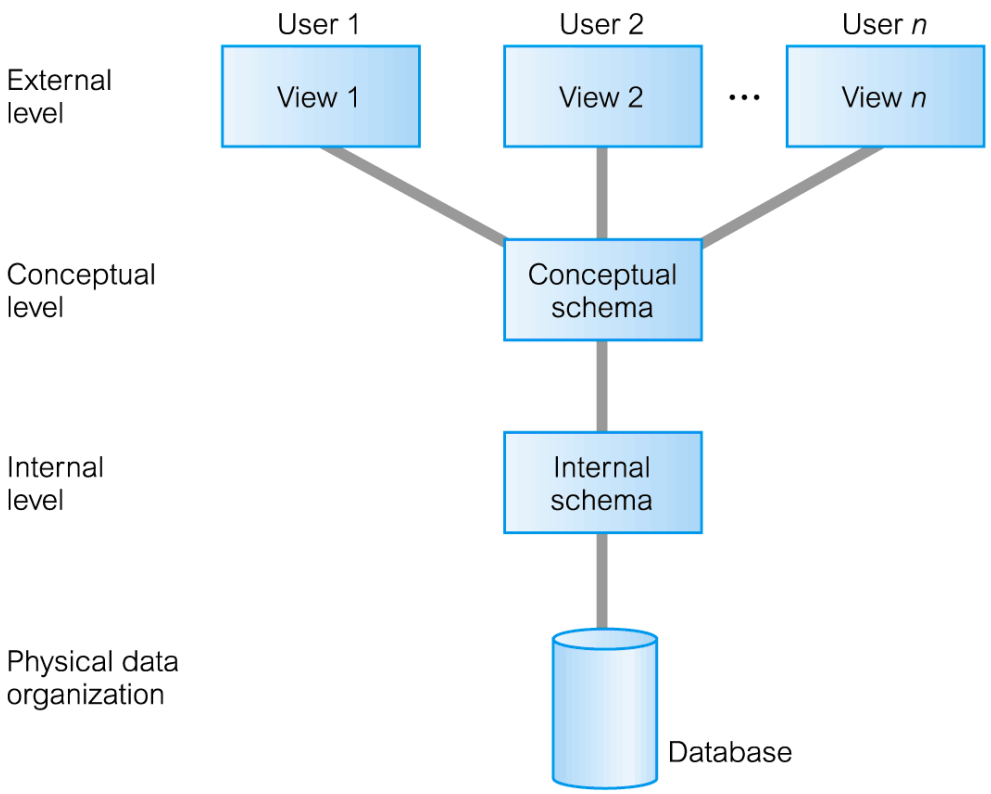
Describes base relations, file organizations, and indexes used to achieve efficient access to data.

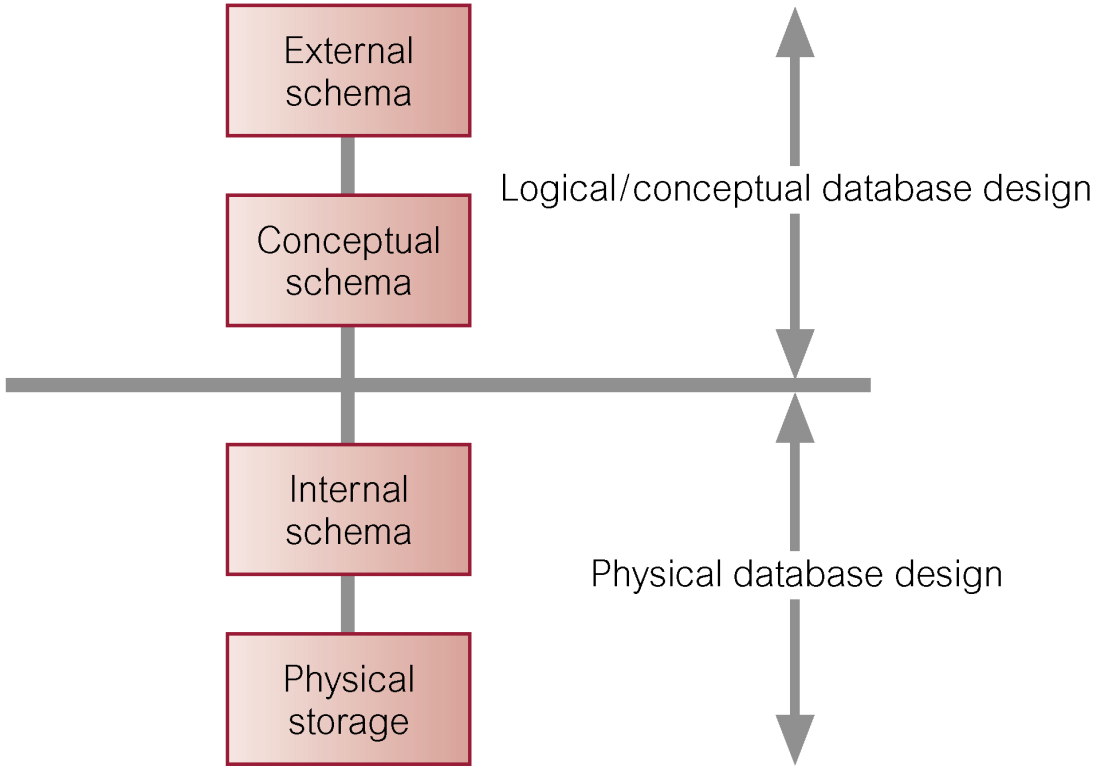
Also describes any associated integrity constraints and security measures.

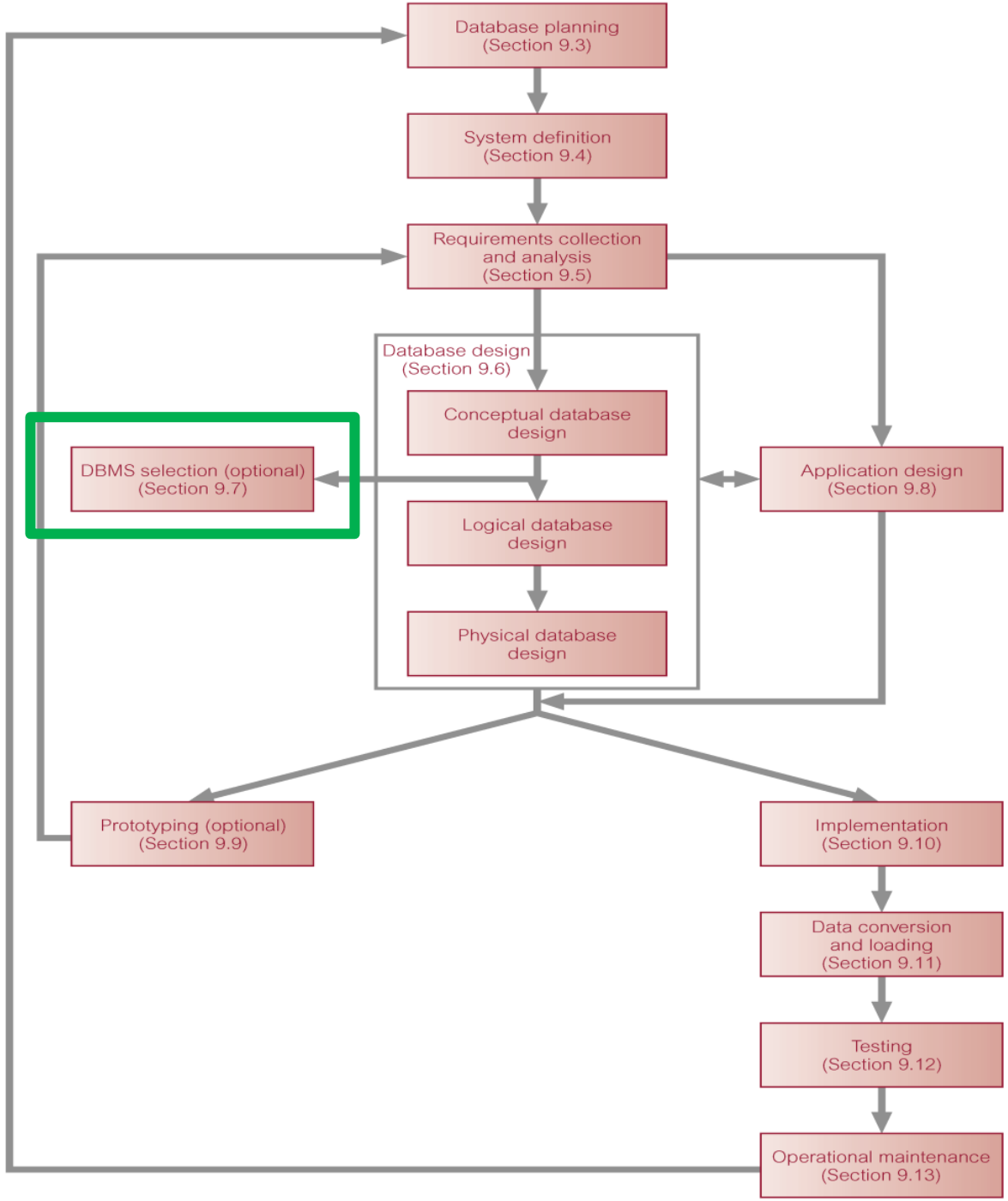
Tailored to a specific DBMS system.



# Three-Level ANSI-SPARC Architecture and Phases of Database Design




Database planning
(Section 9.3)
System definition
(Section 9.4)
Requirements collection
and analysis
(Section 9.5)
Database design
(Section 9.6)
Conceptual database
design
DBMS selection (optional)
(Section 9.7)
Logical database
design
Physical database
design
Application design
(Section 9.8)
Prototyping (optional)
(Section 9.9)
Implementation
(Section 9.10)
Data conversion
and loading
(Section 9.11)
Testing
(Section 9.12)
Operational maintenance
(Section 9.13)

# DBMS Selection

Selection of an appropriate DBMS to support the database system.

Undertaken at any time prior to logical design provided sufficient information is available regarding system requirements.

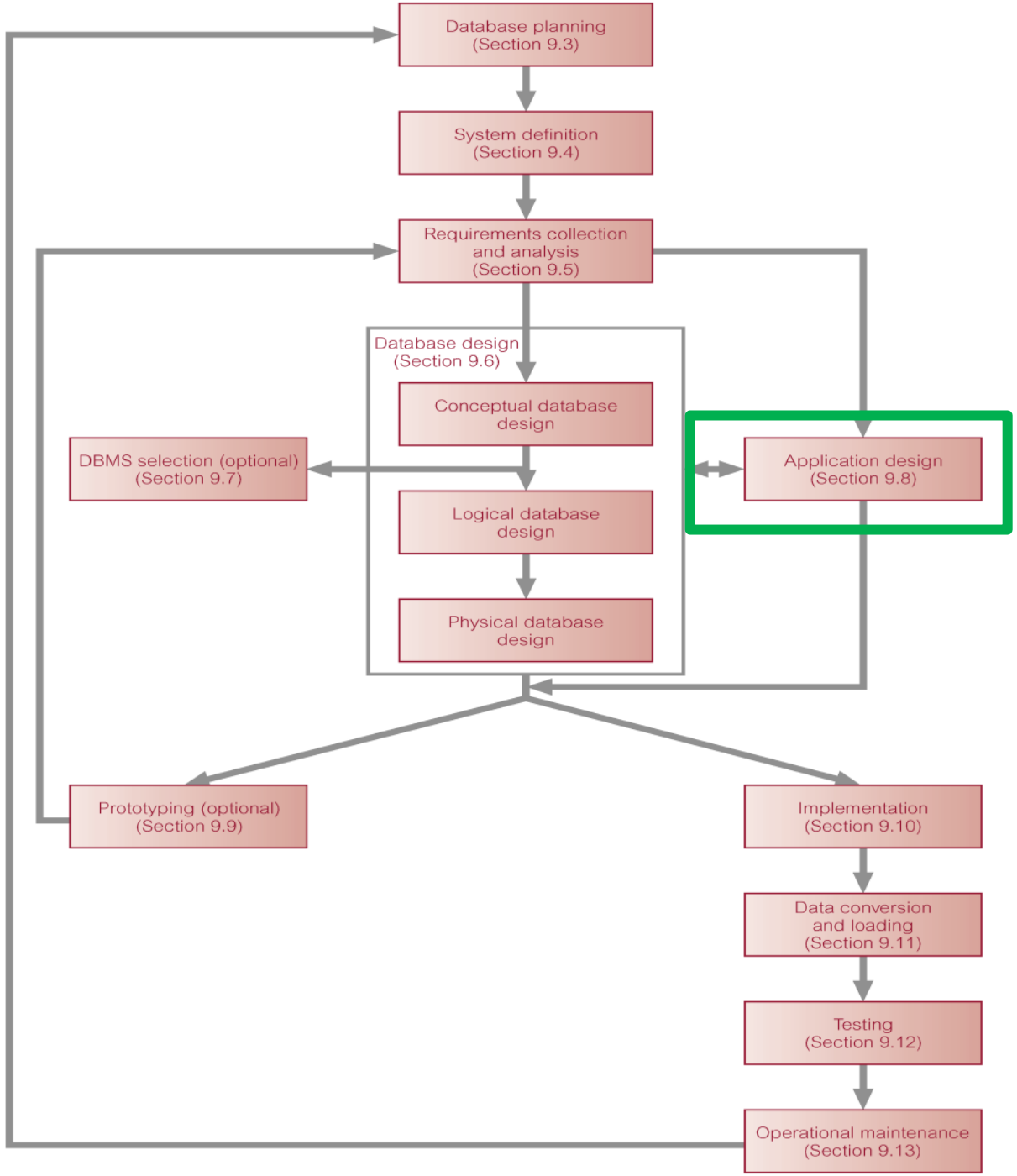
# DBMS Evaluation Features

| Data definition | Physical definition |
| --- | --- |
| Primary key enforcement | File structures available |
| Foreign key specification | File structure maintenance |
| Data types available | Ease of reorganization |
| Data type extensibility | Indexing |
| Domain specification | Variable length fields/records |
| Ease of restructuring | Data compression |
| Integrity controls | Encryption routines |
| View mechanism | Memory requirements |
| Data dictionary | Storage requirements |
| Data independence | |
| Underlying data model | |
| Schema evolution | |

| Accessibility | Transaction handling |
| --- | --- |
| Query language: SQL2/SQL:2003/ODMG compliant | Backup and recovery routines |
| Interfacing to 3GLs | Checkpointing facility |
| Multi-user | Logging facility |
| Security | Granularity of concurrency |
| – Office Access controls | Deadlock resolution strategy |
| – Authorization mechanism | Advanced transaction models |
| | Parallel query processing |

# DBMS Evaluation Features

| Utilities | Development |
|---|---|
| Performance measuring | 4GL/5GL tools |
| Tuning | CASE tools |
| Load/unload facilities | Windows capabilities |
| User usage monitoring | Stored procedures, triggers, and rules |
| Database administration support | Web development tools |
| **Other features** | |
| Upgradability | Interoperability with other DBMSs and other systems |
| Vendor stability | Web integration |
| User base | Replication utilities |
| Training and user support | Distributed capabilities |
| Documentation | Portability |
| Operating system required | Hardware required |
| Cost | Network support |
| Online help | Object-oriented capabilities |
| Standards used | Architecture (2- or 3-tier client/server) |
| Version management | Performance |
| Extensibile query optimization | Transaction throughput |
| Scalability | Maximum number of concurrent users |
| Support for analytical tools | XML support |


Database planning
(Section 9.3)
System definition
(Section 9.4)
Requirements collection
and analysis
(Section 9.5)
Database design
(Section 9.6)
Conceptual database
design
DBMS selection (optional)
(Section 9.7)
Logical database
design
Physical database
design
Application design
(Section 9.8)
Prototyping (optional)
(Section 9.9)
Implementation
(Section 9.10)
Data conversion
and loading
(Section 9.11)
Testing
(Section 9.12)
Operational maintenance
(Section 9.13)

# Application Design

Design of user interface and application programs that use and process the database.

Database design and application design are parallel activities.

Includes two important activities:

- transaction design;
- user interface design.

# Application Design - Transactions

An action, or series of actions, carried out by a single user or application program, which accesses or changes content of the database.

Should define and document the high-level characteristics of the transactions required.

# Application Design - Transactions

Important characteristics of transactions:

- data to be used by the transaction;
- functional characteristics of the transaction;
- output of the transaction;
- importance to the users;
- expected rate of usage.

Three main types of transactions:

- retrieval,
- update, and
- mixed.

# Application Design - Transactions

(a) List the details of branches in a given city.

(b) Identify the total number of branches in each city.

(c) List the name, position, and salary of staff at a given branch, ordered by staff name.

(d) Identify the total number of staff and the sum of their salaries.

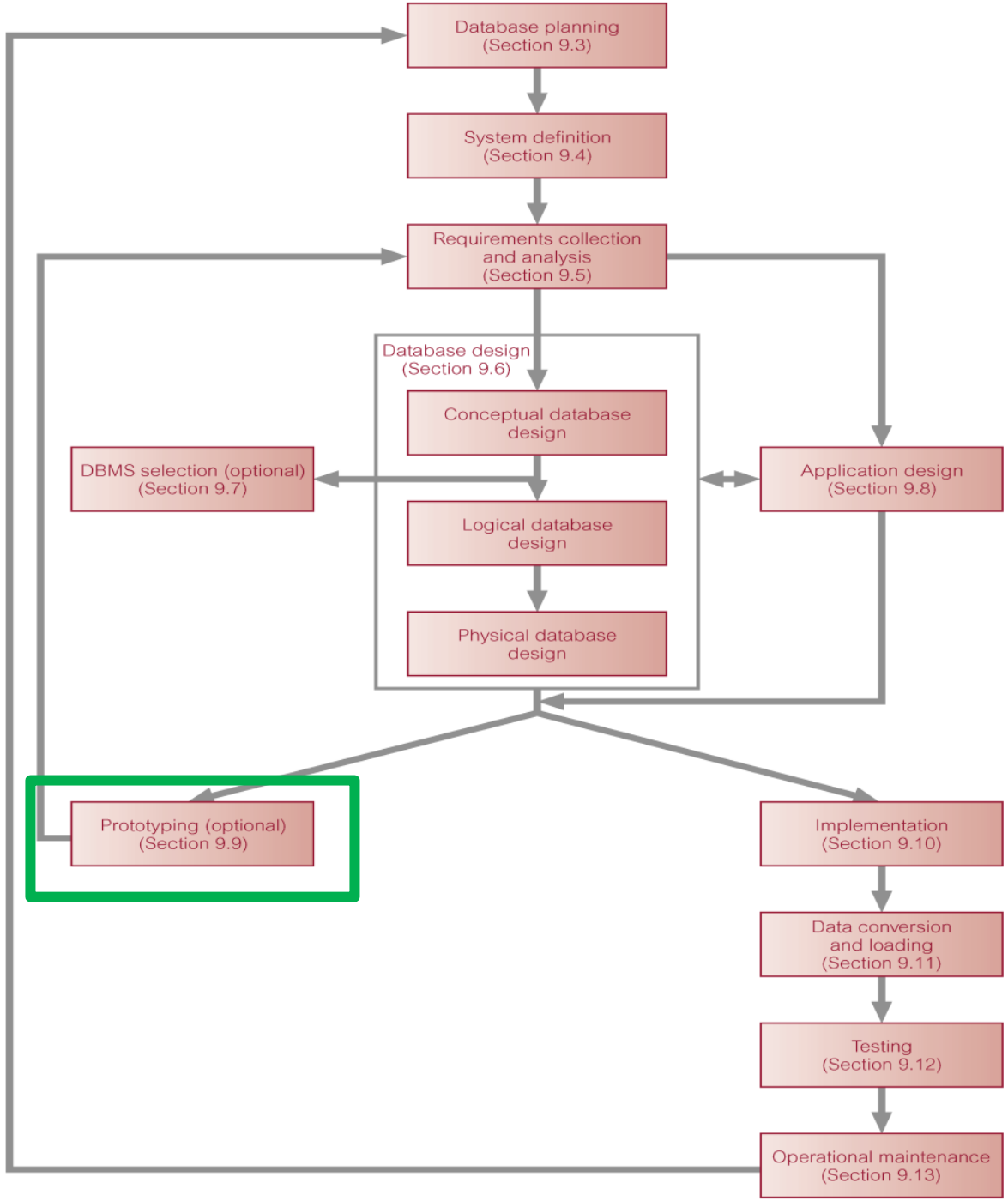
(e) Identify the total number of staff in each position at branches in Glasgow.

(f) List the name of each Manager at each branch, ordered by branch address.

(g) List the names of staff supervised by a named Supervisor.

(h) List the property number, address, type, and rent of all properties in Glasgow, ordered by rental amount.

(i) List the details of properties for rent managed by a named member of staff.

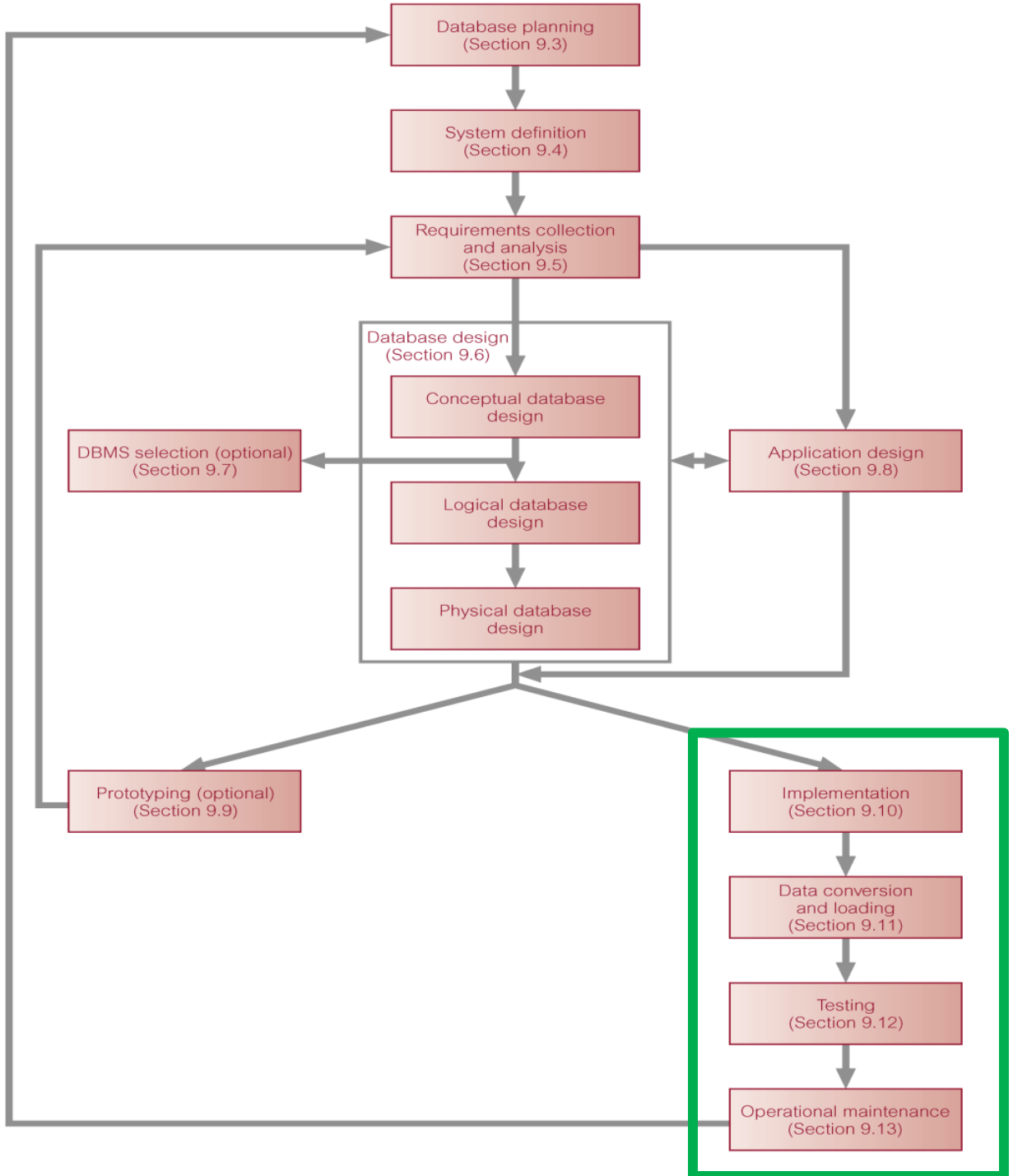

Database planning
(Section 9.3)
System definition
(Section 9.4)
Requirements collection
and analysis
(Section 9.5)
Database design
(Section 9.6)
Conceptual database
design
DBMS selection (optional)
(Section 9.7)
Logical database
design
Physical database
design
Application design
(Section 9.8)
Prototyping (optional)
(Section 9.9)
Implementation
(Section 9.10)
Data conversion
and loading
(Section 9.11)
Testing
(Section 9.12)
Operational maintenance
(Section 9.13)

# Prototyping

Building working model of a database system.

Purpose

- to identify features of a system that work well, or are inadequate;
- to suggest improvements or even new features;
- to clarify the users' requirements;
- to evaluate feasibility of a particular system design.


Database planning
(Section 9.3)
System definition
(Section 9.4)
Requirements collection
and analysis
(Section 9.5)
Database design
(Section 9.6)
Conceptual database
design
DBMS selection (optional)
(Section 9.7)
Logical database
design
Physical database
design
Application design
(Section 9.8)
Prototyping (optional)
(Section 9.9)
Implementation
(Section 9.10)
Data conversion
and loading
(Section 9.11)
Testing
(Section 9.12)
Operational maintenance
(Section 9.13)

# Implementation

Physical realization of the database and application designs.

- Use DDL to create database schemas and empty database files.
- Use DDL to create any specified user views.
- Use 3GL or 4GL to create the application programs. This will include the database transactions implemented using the DML, possibly embedded in a host programming language.

# Data Conversion and Loading

Transferring any existing data into new database and converting any existing applications to run on new database.

Only required when new database system is replacing an old system.

DBMS normally has utility that loads existing files into new database.

May be possible to convert and use application programs from old system for use by new system.

# Testing

Process of running the database system with intent of finding errors.

Use carefully planned test strategies and realistic data.

**Testing cannot show absence of faults; it can show only that software faults are present.**

Demonstrates that database and application programs appear to be working according to requirements.

# Testing

Should also test usability of system.

Evaluation conducted against a usability specification.

Examples of criteria include:

- Learnability;
- Performance;
- Robustness;
- Recoverability;
- Adaptability.

# Operational Maintenance

Process of monitoring and maintaining database system following installation.

Monitoring performance of system.
  if performance falls, may require tuning or reorganization of the database.

Maintaining and upgrading database application
  when required

Incorporating new requirements into database application.

# Chapter 10 - Objectives

Main stages of database system development lifecycle.

Main phases of database design: conceptual, logical, and physical design.

Benefits of CASE tools.

Distinction between data administration and database administration.

# CASE Tools

Computer-Aided Software Engineering

Support provided by CASE tools include:

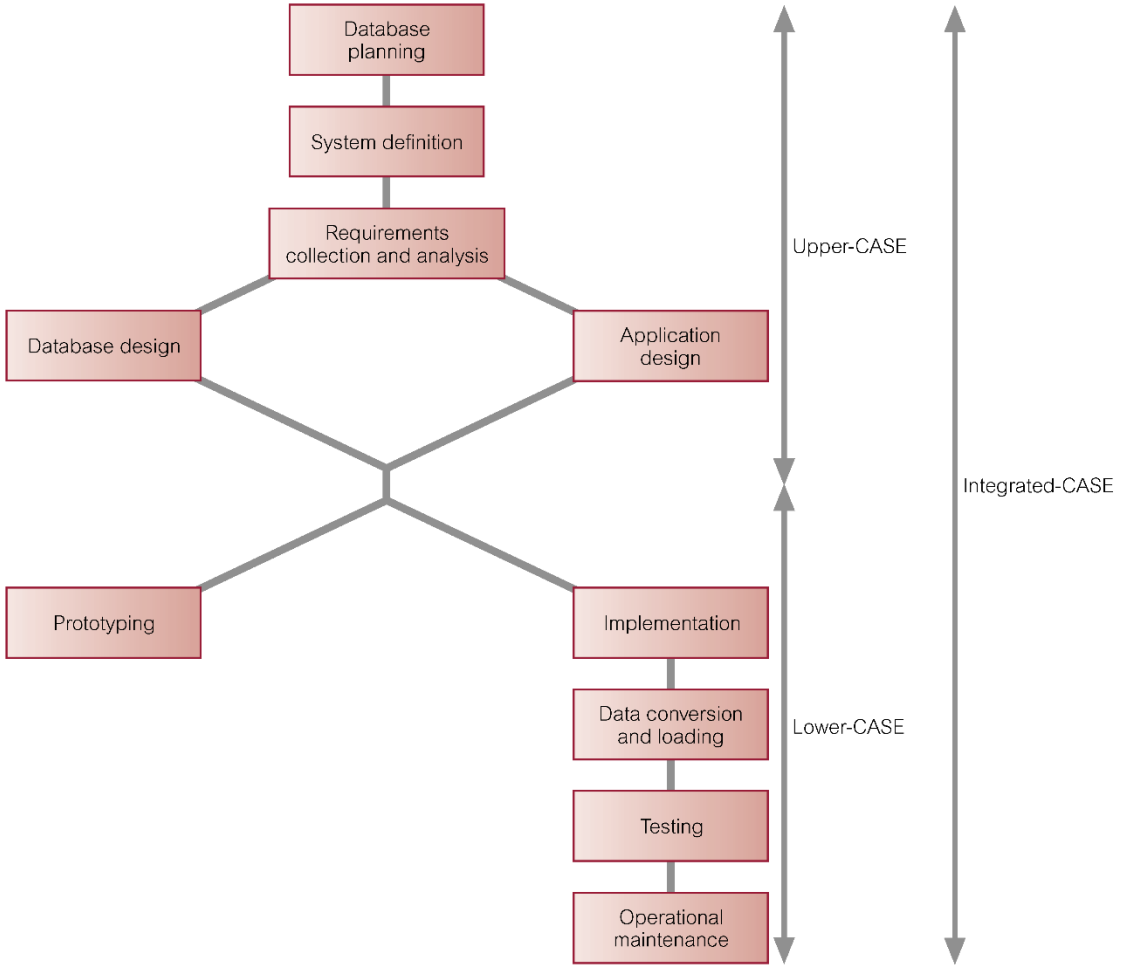
- data dictionary to store information about database system's data;
- design tools to support data analysis;
- tools to permit development of corporate data model, and conceptual and logical data models;
- tools to enable prototyping of applications.

# CASE Tools

Provide following benefits:

- Standards
- Integration
- Support for standard methods
- Consistency
- Automation

# CASE Tools and Database System Development Lifecycle



# Chapter 10 - Objectives

Main stages of database system development lifecycle.

Main phases of database design: conceptual, logical, and physical design.

Benefits of CASE tools.

Distinction between data administration and database administration.

# Data Administration and Database Administration

The Data Administrator (DA) and Database Administrator (DBA) are responsible for managing and controlling the corporate data and corporate database, respectively.

DA is more concerned with early stages of database system development lifecycle

- Planning, Conceptual and Logical Modelling

DBA is more concerned with later stages.

- Physical, Maintenance, Tuning

# Chapter 10 - Objectives

Main stages of database system development lifecycle.

Main phases of database design: conceptual, logical, and physical design.

Benefits of CASE tools.

Distinction between data administration and database administration.