DATABASE SYSTEM DEVELOPMENT LIFECYCLE

Objectives

- Describe database development lifecycle
- Explain information system lifecycle
- Explain the steps in conceptual database design
- Explain the steps in logical database design
- Explain the steps in physical database design

Introduction

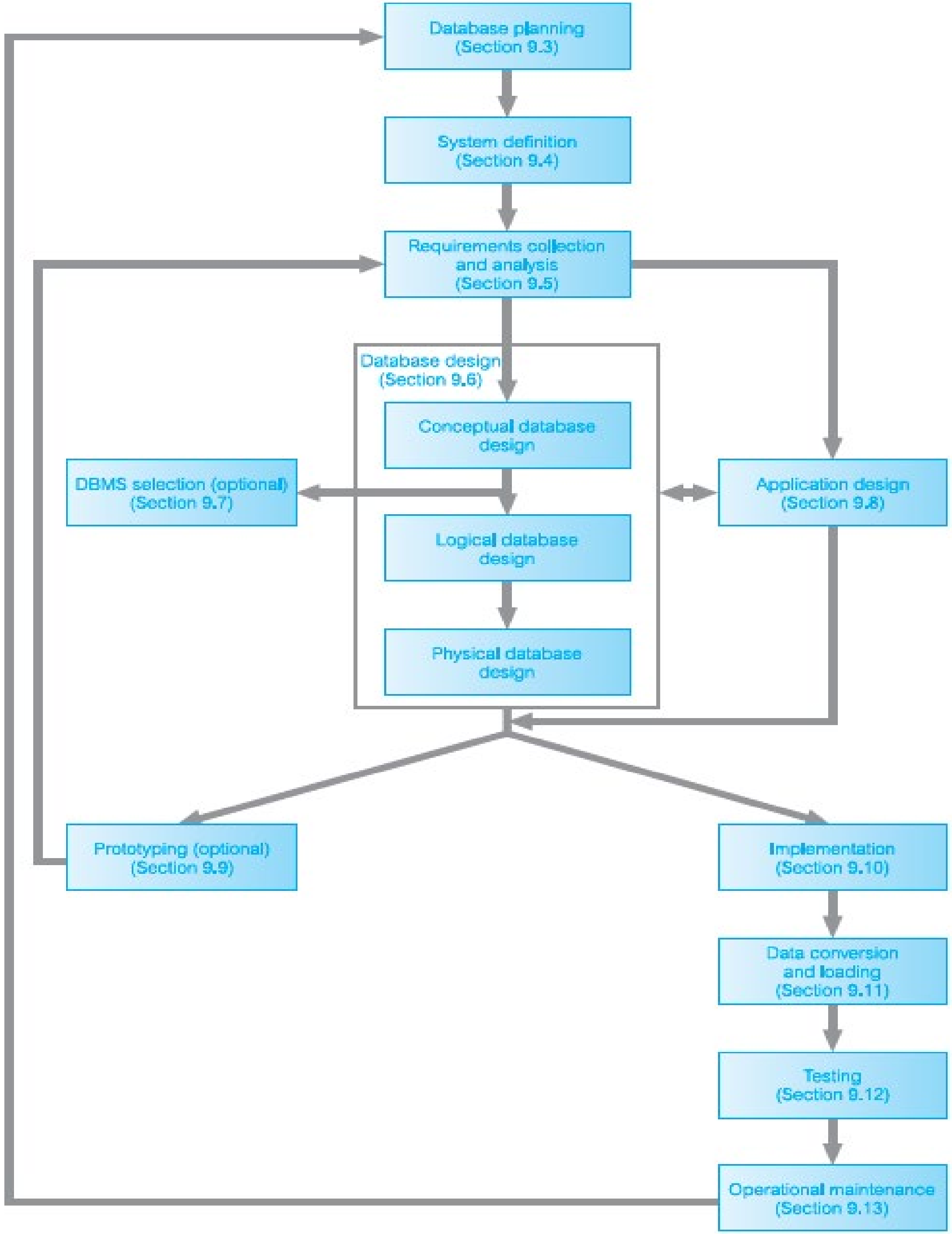
Proliferation of software applications – requiring constant maintenance

Maintenance absorb resource – projects are over budget, late, unreliable , difficult to maintain, performed poorly.

Lead to the issue : Software crisis, later referred to as software depression

Information Systems Lifecycle

- Information system (IS) : The resources that enable the collection, management, control, and dissemination of information throughout an organization.
- Database is a fundamental component of an IS.
- Lifecycle of an organization's IS is inherently linked to the lifecycle of the database system that supports it.



Database System Development Lifecycle

Database Planning

- Database Planning: The management activities that allow stages of database system development lifecycle to be realized as efficiently and effectively as possible.

Issues

- Enterprise plan and goals
- Evaluation of current IS
- IT opportunities

Steps

- Define mission statement
- Identify mission objectives
- Development of standards

System Definition

- System Definition: Describe the scope and boundaries of the database system and the major user views
- User View: Define what is required of a database system from perspective of:
  - a particular job role (such as Manager or Supervisor) or
  - enterprise application area (such as marketing, personnel, or stock control).

Requirements Collection and Analysis

- Requirements Collection and Analysis: Process of collecting and analyzing information about the part of organization to be supported by the database system and using this information to identify users' requirements of new system.

- 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

Database Design

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.

Design Methodology

Design Methodology: A structured approach that uses procedures, techniques, tools, and documentation aids to support and facilitate the process of design.

Critical Success Factors in Database Design

- Work interactively with the users as much as possible.
- Follow a structured methodology throughout the data modeling process.
- Employ a data-driven approach.
- Incorporate structural and integrity considerations into the data models.
- Combine conceptualization, normalization, and transaction validation techniques into the data modeling methodology.
- Use diagrams to represent as much of the data models as possible.
- Use a Database Design Language (DBDL) to represent additional data semantics.
- Build a data dictionary to supplement the data model diagrams.
- Be willing to repeat steps.

Database Design

Approaches

- Bottom-Up
- Top-down
- Inside-out
- Mixed Strategy

Data Modeling

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

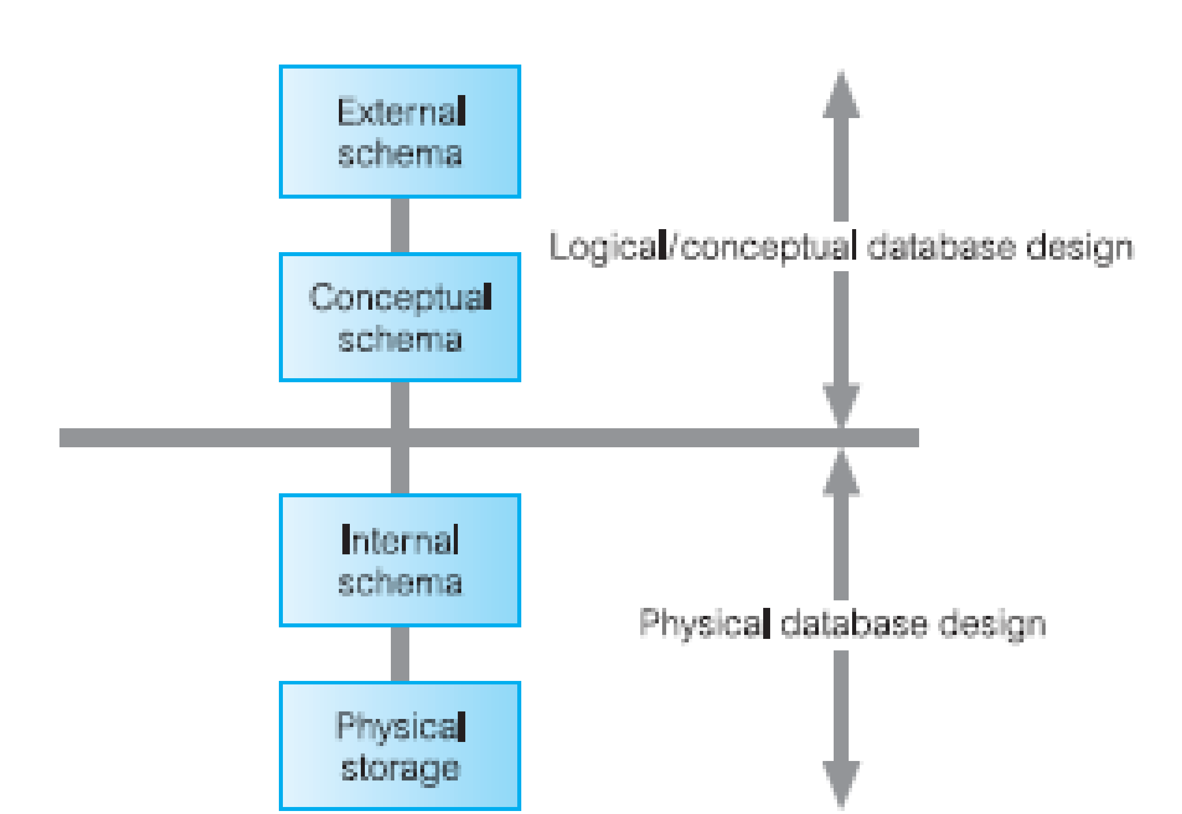
Phase of Database Design

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

Criteria to produce 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
Non-redundancy	• 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.

3 Level of ANSI-SPARC Architecture and Phases of Database Design



DBMS Selection

- 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.
- Main steps to selecting a DBMS:
  - define Terms of Reference of study;
  - shortlist two or three products;
  - evaluate products;
  - recommend selection and produce report.

Application Design

- 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 (retrieval, update, mixed transaction);
  - user interface design.

Prototyping

- Building working model of a database system

Implementation

- Physical realization of the database and application designs.

Data Conversion and Loading

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

Testing

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

Operational Maintenance

- Process of monitoring and maintaining database system following installation.

"

STEPS IN DATABASE DESIGN

STEP 1: Conceptual Database Design

- The process of constructing a model of the data used in an enterprise, independent of ***all*** physical considerations
- Obj: To build a conceptual data model of the data requirements of the enterprise.
  - Model comprises
  - entity types
  - relationship types
  - attributes and attribute domains
  - primary and alternate keys
  - integrity constraints.

- Step 1.1 Identify entity types
- Step 1.2 Identify relationship types
- Step 1.3 Identify and associate attributes with entity or relationship types
- Step 1.4 Determine attribute domains
- Step 1.5 Determine candidate, primary, and alternate key attributes
- Step 1.6 Consider use of enhanced modeling concepts (optional step)
- Step 1.7 Check model for redundancy
- Step 1.8 Validate conceptual model against user transactions
- Step 1.9 Review conceptual data model with user

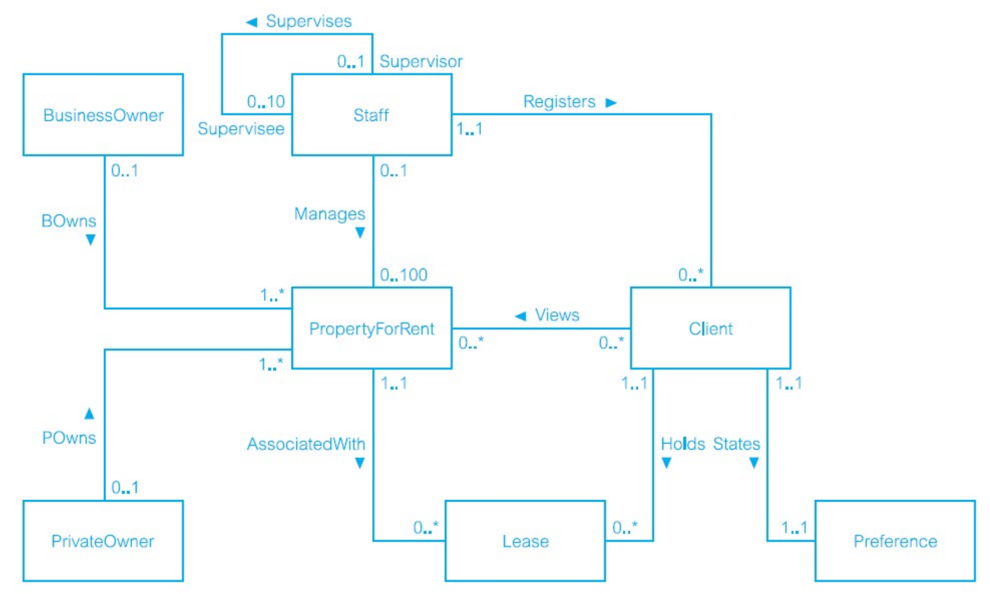
Step 1.1: Identify entity types

- To identify the required entity types by examining the users' requirements specification
- Document entity types

Entity name	*Description*	*Aliases*	*Occurrence*
Staff	General term describing all staff employed by *DreamHome*.	Employee	Each member of staff works at one particular branch.
PropertyForRent	General term describing all property for rent.	Property	Each property has a single owner and is available at one specific branch, where the property is managed by one member of staff. A property is viewed by many clients and rented by a single client, at any one time.

Step 1.2: Identify relationship types

- To identify the important relationships that exist between the entity types.
- Use ERD
- Determine the multiplicity constraints of relationship types
- Check for fan and chasm traps
- Document relationship types



Entity name	*Multiplicity*	*Relationship*	*Multiplicity*	*Entity name*
Staff	0..1 0..1	*Manages* *Supervises*	0..100 0..10	**PropertyForRent** **Staff**
PropertyForRent	1..1	*AssociatedWith*	0..*	**Lease**

Step 1.3 Identify and associate attributes with entity or relationship types

To associate attributes with the appropriate entity or relationship types and document the details of each attribute.

- Simple/composite attributes
- Single/multi-valued attributes
- Derived attributes
- Identify potential problems
- Document attributes

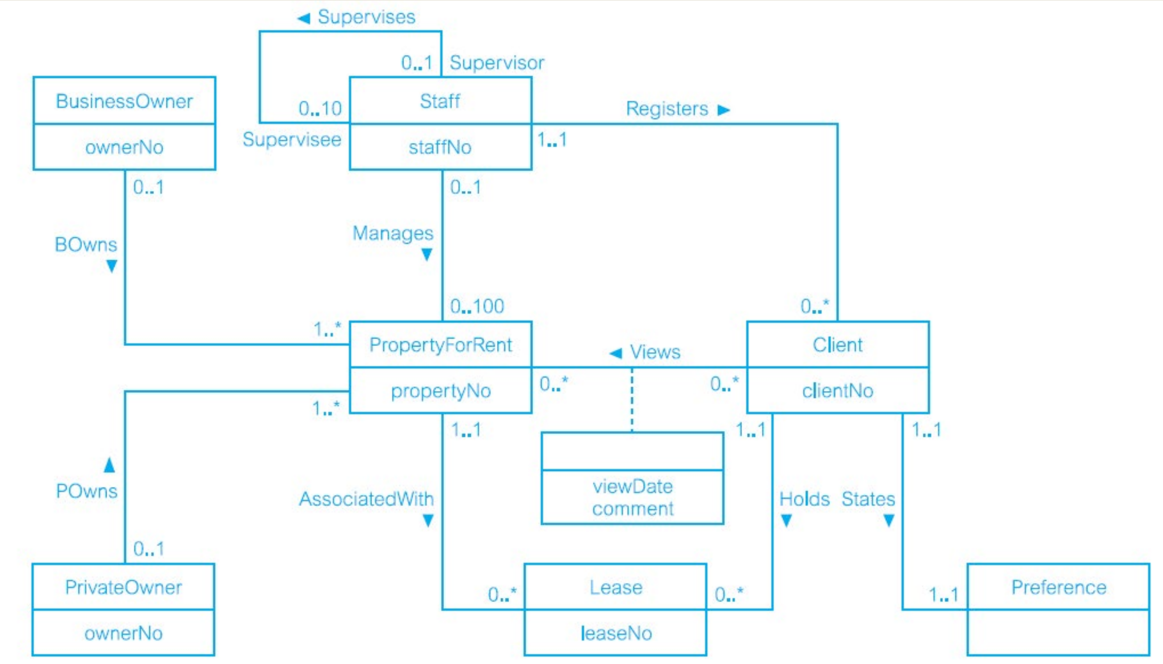
Entity name	*Attributes*	*Description*	*Data Type & Length*	*Nulls*	*Multi-valued*	...
Staff	**staffNo**	Uniquely identifies a member of staff	5 variable characters	No	No	
	name					
	fName	First name of staff	15 variable characters	No	No	
	lName	Last name of staff	15 variable characters	No	No	
	position	Job title of member of staff	10 variable characters	No	No	
	sex	Gender of member of staff	1 character (M or F)	Yes	No	
	DOB	Date of birth of member of staff	Date	Yes	No	
PropertyForRent	**propertyNo**	Uniquely identifies a property for rent	5 variable characters	No	No	

Step 1.4 Determine attribute domains

- To determine domains for the attributes in the data model and document the details of each domain
- Document attribute domains

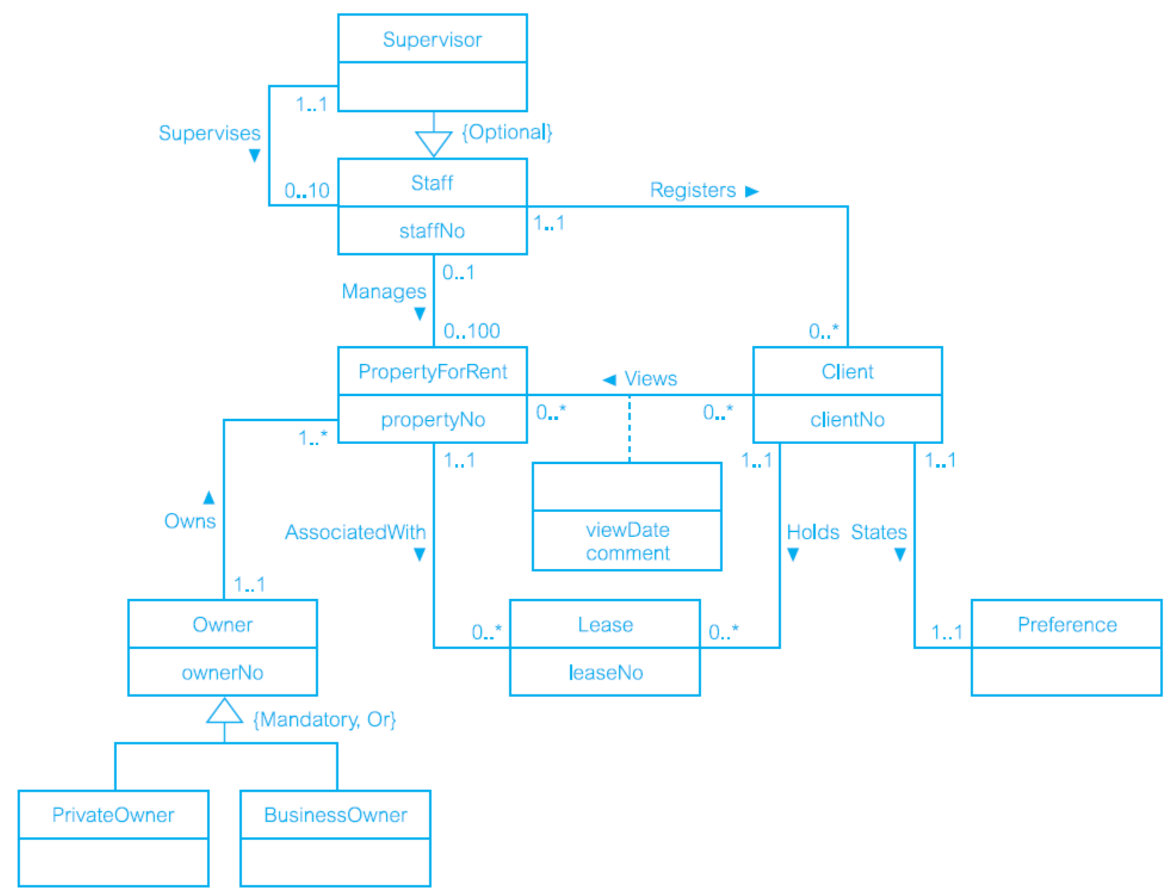
Step 1.5 Determine candidate, primary, and alternate key attributes

- To identify the candidate key(s) for each entity and if there is more than one candidate key, to choose one to be the primary key and the others as alternate keys.
- Document primary and alternate keys



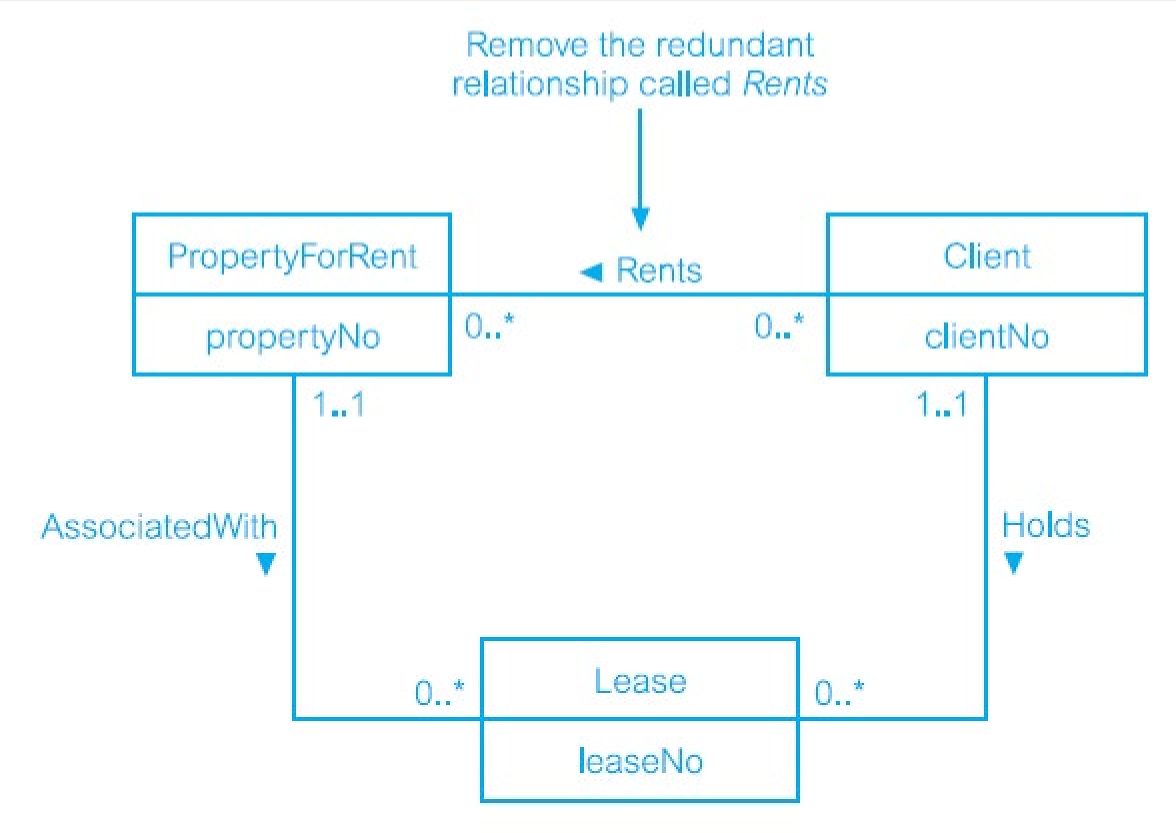
Step 1.6: Consider use of enhanced modeling concepts (optional step)

- To consider the use of enhanced modeling concepts, such as specialization / generalization, aggregation, and composition.



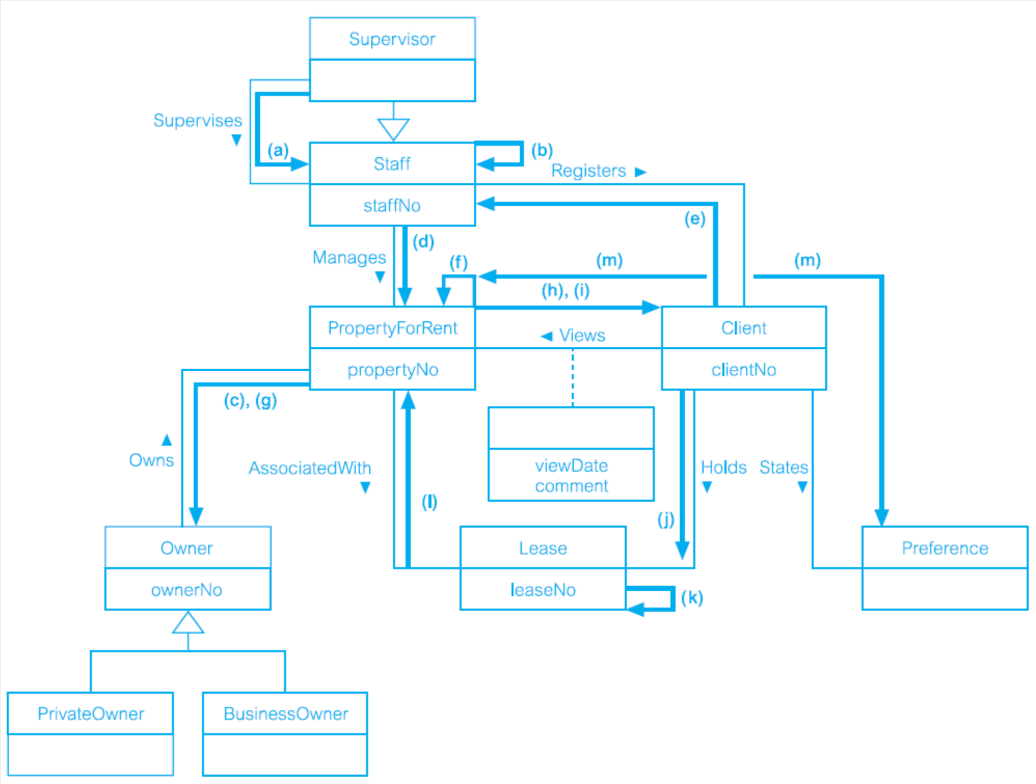
Step 1.7 Check model for redundancy

- To check for the presence of any redundancy in the model and to remove any that does exist.
- Re examine one-to-one relationship
- Remove redundant relationships
- Consider time dimension



Step 1.8 Validate conceptual model against user transactions

- To ensure that the conceptual model supports the required transactions.
- Describing the transaction
- Using transaction pathways



Step 1.9 Review conceptual data model with user

- To review the conceptual data model with the user to ensure that the model is a 'true' representation of the data requirements of the enterprise.

STEP 2: Logical Database Design

- The 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.
- Obj: To translate the conceptual data model into a logical data model and then to validate this model to check that it is structurally correct using normalization and supports the required transactions.

Step 2.1 Derive relations for logical data model

Step 2.2 Validate relations using normalization

Step 2.3 Validate relations against user transactions

Step 2.4 Check integrity constraints

Step 2.5 Review logical data model with user

Step 2.6 Merge logical data models into global model (optional step)

Step 2.7 Check for future growth

Step 2.1 Derive relations for logical data model

- Obj: To create relations for the logical data model to represent the entities, relationships, and attributes that have been identified.

Strong entity types	Weak entity types	One-to-many (1:*) binary relationship types
One-to-one (1:1) binary relationship types	One-to-one (1:1) recursive relationship types	Superclass/subclass relationship types
Many-to-many (*:*) binary relationship types	Complex relationship types	Multi-Valued Attributes

Step 2.2 Validate relations using normalization

- To validate the relations in the logical data model using normalization

Step 2.3 Validate relations against user transactions

- To ensure that the relations in the logical data model support the required transactions

Step 2.4 Check integrity constraints

- To check integrity constraints are represented in the logical data model

- Required data
- Attribute domain constraints
- Multiplicity
- Entity integrity
- Referential integrity
- General constraints

Staff (staffNo, fName, lName, position, sex, DOB, supervisorStaffNo)
Primary Key staffNo
Foreign Key supervisorStaffNo **references** Staff(staffNo) ON UPDATE CASCADE ON DELETE SET NULL

Client (clientNo, fName, lName, telNo, prefType, maxRent, staffNo)
Primary Key clientNo
Foreign Key staffNo **references** Staff(staffNo) ON UPDATE CASCADE ON DELETE NO ACTION

PropertyForRent (propertyNo, street, city, postcode, type, rooms, rent, ownerNo, staffNo)
Primary Key propertyNo
Foreign Key ownerNo **references** PrivateOwner(ownerNo) and BusinessOwner(ownerNo) ON UPDATE CASCADE ON DELETE NO ACTION
Foreign Key staffNo **references** Staff(staffNo) ON UPDATE CASCADE ON DELETE SET NULL

Viewing (clientNo, propertyNo, dateView, comment)
Primary Key clientNo, propertyNo
Foreign Key clientNo **references** Client(clientNo) ON UPDATE CASCADE ON DELETE NO ACTION
Foreign Key propertyNo **references** PropertyForRent(propertyNo) ON UPDATE CASCADE ON DELETE CASCADE

Lease (leaseNo, paymentMethod, depositPaid, rentStart, rentFinish, clientNo, propertyNo)
Primary Key leaseNo
Alternate Key propertyNo, rentStart
Alternate Key clientNo, rentStart
Foreign Key clientNo **references** Client(clientNo) ON UPDATE CASCADE ON DELETE NO ACTION
Foreign Key propertyNo **references** PropertyForRent(propertyNo) ON UPDATE CASCADE ON DELETE NO ACTION

Step 2.5 Review logical data model with user

- To review the logical data model with the users to ensure that they consider the model to be a true representation of the data requirements of the enterprise.

Step 2.6 Merge logical data models into global Model (optional step)

- To merge logical data models into a single global logical data model that represents all user views of a database.

Step 2.6.1 Merge local logical data models into global model	Step 2.6.2 Validate global logical data model	Step 2.6.3 Review global logical data model with users.

Step 2.7 Check for future growth

- To determine whether there are any significant changes likely in the foreseeable future and to assess whether the logical data model can accommodate these changes.

STEP 3: Physical Database Design

- The process of producing a description of the implementation of the database on secondary storage; it describes the base relations, file organizations, and indexes design used to achieve efficient access to the data, and any associated integrity constraints and security measures.
- Sources of information for physical design process includes logical data model and documentation that describes model.
- Logical database design is concerned with the what, physical database design is concerned with the how.

Step 3 Translate logical data model for target DBMS

- Step 3.1 Design base relations
- Step 3.2 Design representation of derived data
- Step 3.3 Design general constraints

Step 4 Design file organizations and indexes

- Step 4.1 Analyze transactions
- Step 4.2 Choose file organizations
- Step 4.3 Choose indexes
- Step 4.4 Estimate disk space requirements

Step 5 Design user views

Step 6 Design security mechanisms

Physical Database Design

- To produce a relational database schema from the logical data model that can be implemented in the target DBMS.
- Need to know functionality of target DBMS such as how to create base relations and whether the system supports the definition of:
  - PKs, FKs, and AKs;
  - required data – i.e. whether system supports NOT NULL;
  - domains;
  - relational integrity constraints;
  - general constraints.

Step 3.1 Design base relations

- For each relation, need to define:
  - the name of the relation;
  - a list of simple attributes in brackets;
  - the PK and, where appropriate, AKs and FKs.
  - referential integrity constraints for any FKs identified

```
    Domain PropertyNumber:      variable length character string, length 5
    Domain Street:              variable length character string, length 25
    Domain City:                variable length character string, length 15
    Domain Postcode:            variable length character string, length 8
    Domain PropertyType:        single character, must be one of 'B', 'C', 'D', 'E', 'F', 'H', 'M', 'S'
    Domain PropertyRooms:       integer, in the range 1–15
    Domain PropertyRent:        monetary value, in the range 0.00–9999.99
    Domain OwnerNumber:         variable length character string, length 5
    Domain StaffNumber:         variable length character string, length 5
    Domain BranchNumber:        fixed length character string, length 4
PropertyForRent(
    propertyNo    PropertyNumber    NOT NULL,
    street        Street            NOT NULL,
    city          City              NOT NULL,
    postcode      Postcode,
    type          PropertyType      NOT NULL DEFAULT 'F',
    rooms         PropertyRooms     NOT NULL DEFAULT 4,
    rent          PropertyRent      NOT NULL DEFAULT 600,
    ownerNo       OwnerNumber       NOT NULL,
    staffNo       StaffNumber,
    branchNo      BranchNumber      NOT NULL,
    PRIMARY KEY (propertyNo),
    FOREIGN KEY (staffNo) REFERENCES Staff(staffNo) ON UPDATE CASCADE ON DELETE SET NULL,
    FOREIGN KEY (ownerNo) REFERENCES PrivateOwner(ownerNo) and BusinessOwner(ownerNo)
                          ON UPDATE CASCADE ON DELETE NO ACTION,
    FOREIGN KEY (branchNo) REFERENCES Branch(branchNo)
                          ON UPDATE CASCADE ON DELETE NO ACTION);
```

Step 3.1 Design base relations

- For each relation, need to define:
  - the name of the relation;
  - a list of simple attributes in brackets;
  - the PK and, where appropriate, AKs and FKs.
  - referential integrity constraints for any FKs identified

- From data dictionary, we have for each attribute:
- its domain, consisting of a data type, length, and any constraints on the domain;
- an optional default value for the attribute;
- whether it can hold nulls;
- whether it is derived, and if so, how it should be computed.

Step 3.1 Design base relations

```
Domain PropertyNumber:      variable length character string, length 5
Domain Street:              variable length character string, length 25
Domain City:                variable length character string, length 15
Domain Postcode:            variable length character string, length 8
Domain PropertyType:        single character, must be one of 'B', 'C', 'D', 'E', 'F', 'H', 'M', 'S'
Domain PropertyRooms:       integer, in the range 1–15
Domain PropertyRent:        monetary value, in the range 0.00–9999.99
Domain OwnerNumber:         variable length character string, length 5
Domain StaffNumber:         variable length character string, length 5
Domain BranchNumber:        fixed length character string, length 4
PropertyForRent(
    propertyNo   PropertyNumber   NOT NULL,
    street       Street           NOT NULL,
    city         City             NOT NULL,
    postcode     Postcode,
    type         PropertyType     NOT NULL DEFAULT 'F',
    rooms        PropertyRooms    NOT NULL DEFAULT 4,
    rent         PropertyRent     NOT NULL DEFAULT 600,
    ownerNo      OwnerNumber      NOT NULL,
    staffNo      StaffNumber,
    branchNo     BranchNumber     NOT NULL,
    PRIMARY KEY (propertyNo),
    FOREIGN KEY (staffNo) REFERENCES Staff(staffNo) ON UPDATE CASCADE ON DELETE SET NULL,
    FOREIGN KEY (ownerNo) REFERENCES PrivateOwner(ownerNo) and BusinessOwner(ownerNo)
                                  ON UPDATE CASCADE ON DELETE NO ACTION,
    FOREIGN KEY (branchNo) REFERENCES Branch(branchNo)
                                  ON UPDATE CASCADE ON DELETE NO ACTION);
```

Step 3.2 Design representation of derived data

- To decide how to represent any derived data present in logical data model in target DBMS.
- Examine logical data model and data dictionary and produce list of all derived attributes.
- Derived attribute can be stored in database or calculated every time it is needed.

- Option selected is based on:
- additional cost to store the derived data and keep it consistent with operational data from which it is derived;
- cost to calculate it each time it is required.
- Less expensive option is chosen subject to performance constraints.

Step 3.2 Design representation of derived data

PropertyForRent

propertyNo	street	city	postcode	type	rooms	rent	ownerNo	staffNo	branchNo
PA14	16 Holhead	Aberdeen	AB7 5SU	House	6	650	CO46	SA9	B007
PL94	6 Argyll St	London	NW2	Flat	4	400	CO87	SL41	B005
PG4	6 Lawrence St	Glasgow	G11 9QX	Flat	3	350	CO40		B003
PG36	2 Manor Rd	Glasgow	G32 4QX	Flat	3	375	CO93	SG37	B003
PG21	18 Dale Rd	Glasgow	G12	House	5	600	CO87	SG37	B003
PG16	5 Novar Dr	Glasgow	G12 9AX	Flat	4	450	CO93	SG14	B003

Staff

staffNo	fName	lName	branchNo	noOfProperties
SL21	John	White	B005	0
SG37	Ann	Beech	B003	2
SG14	David	Ford	B003	1
SA9	Mary	Howe	B007	1
SG5	Susan	Brand	B003	0
SL41	Julie	Lee	B005	1

Step 3.3 Design general constraints

- To design the general constraints for target DBMS.

- Some DBMS provide more facilities than others for defining enterprise constraints. Example:

```
CONSTRAINT StaffNotHandlingTooMuch
    CHECK (NOT EXISTS (SELECT staffNo
                       FROM PropertyForRent
                       GROUP BY staffNo
                       HAVING COUNT(*) > 100))
```

Step 4: Design File Organizations and Indexes

- To determine the optimal file organizations to store the base relations and the indexes that are required to achieve acceptable performance, that is, the way in which relations and tuples will be held on secondary storage.

Step 4.1 Analyze transactions

- To understand the functionality of the transactions that will run on the database and to analyze the important transactions.

- Attempt to identify performance criteria, such as:
  - transactions that run frequently and will have a significant impact on performance;
  - transactions that are critical to the business;
  - times during the day/week when there will be a high demand made on the database (called the peak load).

Step 4.1 Analyze transactions

- To focus on areas that may be problematic:
  - Map all transaction paths to relations.
  - Determine which relations are most frequently accessed by transactions.
  - Analyze the data usage of selected transactions that involve these relations.

Step 4.1 Analyze transactions

Table 17.1 Cross-referencing transactions and relations.

Transaction/ Relation	(A)				(B)				(C)				(D)				(E)				(F)			
	I	R	U	D	I	R	U	D	I	R	U	D	I	R	U	D	I	R	U	D	I	R	U	D
Branch										X				X								X		
Telephone																								
Staff		X				X				X								X				X		
Manager																								
PrivateOwner	X																							
BusinessOwner	X																							
PropertyForRent	X					X	X	X						X				X				X		
Viewing																								
Client																								
Registration																								
Lease																								
Newspaper																								
Advert																								

I = Insert; R = Read; U = Update; D = Delete

Step 4.2 Choose file organizations

- To determine an efficient file organization for each base relation.
- File organizations include Heap, Hash, Indexed Sequential Access Method (ISAM), B+-Tree, and Clusters.
- Some DBMSs may not allow selection of file organizations.

Step 4.3 Choose indexes

- To determine whether adding indexes will improve the performance of the system.
- One approach is to keep tuples unordered and create as many <u>secondary indexes</u> as necessary
- Another approach is to order tuples in the relation by specifying a primary or clustering index.

Step 4.4 Estimate disk space requirements

- To estimate the amount of disk space that will be required by the database.

Step 5: Design User Views

- To design the user views that were identified during the Requirements Collection and Analysis stage of the database system development lifecycle.

Step 6: Design Security Measures

- To design the security measures for the database as specified by the users.
- Database security generally provided by DBMS:
  - System security (cover access and use of database at the system level)
  - Data security (cover access and use of database objects and the action that user can have on the objects)

Summary

Database planning
System definition
Requirements collection and analysis
Database design
DBMS selection (optional)
Application design
Prototyping (optional)
Implementation
Data conversion and loading
Testing
Operational maintenance

Information System Lifecycle

Database System Development Lifecycle

Conceptual Database Design

1.1 Identify Entity types
1.2 Identify relationship types
1.3 Identify associate attribute
1.4 Determine attribute domains
1.5 Determine keys
1.6 Enhanced modeling concepts
1.7 Check redundancy
1.8 Validate conceptual models
1.9 Review conceptual models

Logical Database Design

2.1 Derive relations
2.2 & 2.3 Validate relations
2.4 Check constraint
2.5 Review logical data model
2.6 merge logical data model

Physical Database Design

Step 3: Translate logical data model
Step 4: Design file organizations and index
Step 5: Design user views
Step 6 Design Security mechanism

References

- Thomas Connolly and Carolyn Begg, Database Systems: A Practical Approach to Design, Implementation, and Management, 6th Edition, Pearson, 2015, ISBN: 978- 01329432