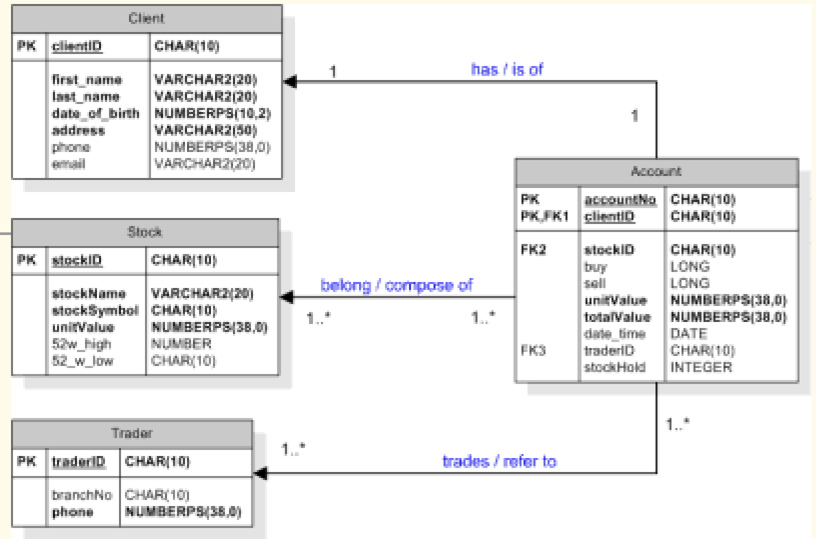
# Conceptual Database Design 1



| Client | | |
|---|---|---|
| PK | clientID | CHAR(10) |
| | first_name | VARCHAR2(20) |
| | last_name | VARCHAR2(20) |
| | date_of_birth | NUMBERPS(10,2) |
| | address | VARCHAR2(50) |
| | phone | NUMBERPS(38,0) |
| | email | VARCHAR2(20) |

| Stock | | |
|---|---|---|
| PK | stockID | CHAR(10) |
| | stockName | VARCHAR2(20) |
| | stockSymbol | CHAR(10) |
| | unitValue | NUMBERPS(38,0) |
| | 52w_high | NUMBER |
| | 52_w_low | CHAR(10) |

| Trader | | |
|---|---|---|
| PK | traderID | CHAR(10) |
| | branchNo | CHAR(10) |
| | phone | NUMBERPS(38,0) |

| Account | | |
|---|---|---|
| PK | accountNo | CHAR(10) |
| PK,FK1 | clientID | CHAR(10) |
| FK2 | stockID | CHAR(10) |
| | buy | LONG |
| | sell | LONG |
| | unitValue | NUMBERPS(38,0) |
| | totalValue | NUMBERPS(38,0) |
| | date_time | DATE |
| FK3 | traderID | CHAR(10) |
| | stockHold | INTEGER |

# Topics List

- Conceptual Database Design
- Build conceptual data model

# Conceptual Database Design

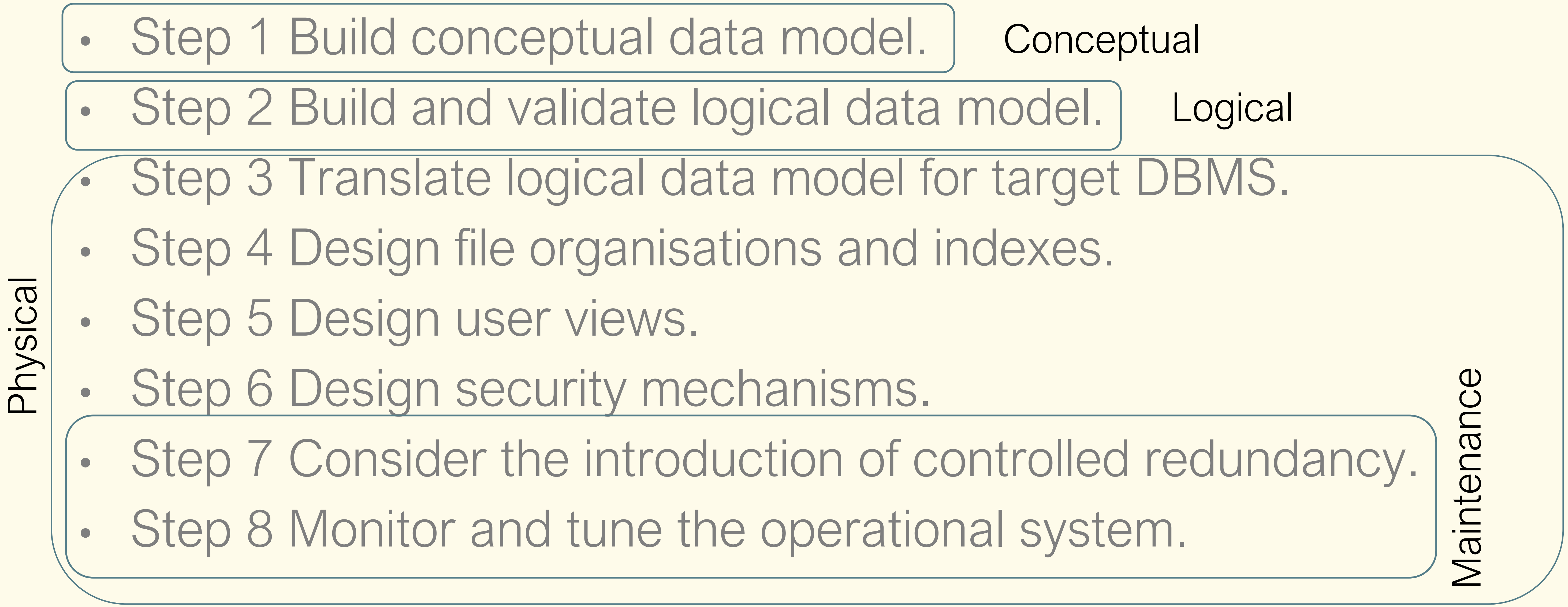
- The process of constructing a model of the data used in an enterprise, independent of *all* physical considerations.

- The goal of this phase is to produce a conceptual schema (which includes identification of the important entity types, relationship types, and attributes) for the database that is independent of a specific DBMS.

- We will use Entity Relationship (ER) modelling during this phase.

# Conceptual Database Design

- Overview Database Design Methodology:



  - Step 1 Build conceptual data model.
  - Step 2 Build and validate logical data model.
  - Step 3 Translate logical data model for target DBMS.
  - Step 4 Design file organisations and indexes.
  - Step 5 Design user views.
  - Step 6 Design security mechanisms.
  - Step 7 Consider the introduction of controlled redundancy.
  - Step 8 Monitor and tune the operational system.

Conceptual

Logical

Physical

Maintenance

# Conceptual Database Design

## Build conceptual data model

- Step 1.1 Identify entity types.
- Step 1.2 Identify relationship types.
    - 1.2.1 Cardinality.
    - 1.2.2 Participation.
- 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 modelling 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.

# Topics List

- Conceptual Database Design
- Build conceptual data model

# Build conceptual data model

## Identify entity types

- Identify entity types that we need to represent in the database.
- Typical examples:
    - People: staff, clients/customers, patients, members, owners, contacts, other individuals
    - Objects: stock items, real estate, offices
    - Organisations: firms (suppliers), departments, charities, clubs, committees
    - Object classes: recordings, films, books, types of stock, biological species, work roles
    - Events: concerts, examinations, lecture courses, consultations, sales
- Document the entity types.

# Build conceptual data model

## Identify relationship types

- Identify the relationships between entity types that need to be recorded.

- Typical examples:
  - Ownership: person *owns* object
  - Lines of command: person *supervises* person
  - Participation: person *participates in* event
  - Part of relationship: item *is part of* order; person *belongs to* organisation
  - Location: house *is located in* region
  - Personal: person *is married to* person; person *is parent of* person

# Build conceptual data model

## Identify relationship types

- Use entity–relationship (ER) modelling to understand the relationship types.
- Determine the *Structural Constraints* of each relationship type.
- Document the relationship types.

# Build conceptual data model

## Identify and associate attributes with entity or relationship types.

- Attributes can be identified where a *noun* or a *noun phrase* is a property, quality, identifier, or characteristic of one of the entity or relationship types previously found.
- Identify whether attributes are:
  - Simple/composite.
  - Single/multi-valued.
  - Derived.
- Document the attributes.

# Build conceptual data model

## Documenting attributes

- Record the following information for each attribute:
  - attribute name and description;
  - data type and length;
  - any aliases that the attribute is known by;
  - whether the attribute must always be specified (in other words, whether the attribute allows or disallows nulls);
  - whether the attribute is multi-valued;
  - whether the attribute is composite, and if so, which simple attributes make up the composite attribute;
  - whether the attribute is derived and, if so, how it should be computed;
  - default values for the attribute (if specified).

# Build conceptual data model

## Determine attribute domains

- A domain is a pool of values from which one or more attributes draw their values.
- A domain specifies:
  - allowable set of values for the attribute;
  - size and format of the attribute.
- Document the attribute domains.

# Build conceptual data model

## Guidelines for choosing a primary key

- Select the candidate key
  - with the minimal set of attributes;
  - that is less likely to have its values changed;
  - that is less likely to lose uniqueness in the future;
  - with fewest characters (for those with textual attribute(s));
  - with the smallest maximum value (for numerical attributes);
  - that is easiest to use from the users’ point of view.