Databases

Database Schemas and Design

Dr Konrad Dabrowski
konrad.dabrowski@durham.ac.uk

Online Office Hour:
Mondays 13:30–14:30
See Duo for the Zoom link

Outline of this Lecture

- Transactions and concurrency control
- Abstract data models
  - Structured data vs. Semi-Structured data
  - Relational Data Model
  - Entity-Relationship (ER) diagrams
- Database schemas and data independence
  - external / conceptual / internal schema
- 3-level Database architecture
- Database design methodology

Transactions and Concurrency Control

- DB Management System (DBMS):
  A software system that enables users to define / create / maintain / control the access to the DB
- We need to trust a DBMS

  $\Longrightarrow$ mechanisms to ensure that the database:
  - is reliable
  - remains always in a consistent state
- Especially when:
  - software / hardware failures
  - multiple users access the database simultaneously, e.g.
    - bank accounts
    - flight reservations

Transactions and Concurrency Control

- Database recovery: the process of restoring a database to a correct state after a failure

- Concurrency control protocols: prevent database accesses from interfering with each other

Central notion in a DBMS:

- Transaction: an action (or series of actions) carried out by a single user / program, which reads / updates the database

  – one *logical unit of work*: "one action" in the real world,

  e.g. move £100 from an account to another

Transactions and Concurrency Control

- At the end of a transaction:
  - database again in consistent state
  - valid integrity / referential constraints
- During the execution of a transaction:
  - maybe in an inconsistent state,
    i.e. constraints may be violated!
- A transaction can have two outcomes:
  - committed
    - when it completes successfully
  - rolled back
    - when it does *not* complete successfully

$\Longrightarrow$ a transaction is either performed entirely or not at all!

Transactions and Concurrency Control

- Concurrency Control: the process of managing simultaneous operations on the DB, without having them interfere with each other
- Two transactions may be:
  - both correct by themselves, but
  - when they are executed simultaneously, they may cause inconsistency of the database
- Main purpose: when many users access the DB
- Very different from multi-user Operating Systems:
  - an OS allows two people to *edit* a document at the same time
  - if both write, then one's changes get lost – not in a DBMS!

Abstract Data Models

- Data Definition Language (DDL): *(creation of a DB)*
  - Specifies entities / attributes / relationships / constraints for the stored data (used by the DBA)

However:

- DDL is too low-level to describe organization of the data in a simple way, understandable by most of users (i.e. not only by the DBA!)

$\Longrightarrow$ We need a Data Model:

- a collection of intuitive concepts describing *data*, their *relationships* and *constraints*

Types of data organization

- Three characterizations of data:
  - *Structured* data
  - *Semi-structured* data (XML)
  - *Unstructured* data
- Structured data:
  - data represented in a strict format (i.e. schema)
    - relational data model (tables, tuples, attributes)
  - the DBMS checks to ensure that the data follows:
    - the structures (table, attributes, domains)
    - the integrity & referential constraints

    that are specified in the schema

Types of data organization

- Semi-structured data (e.g. XML):
  - self describing data
  - the “schema” information is mixed with the data values
- How do we end up with such data?
  - sometimes data is collected *ad hoc*
    - i.e. no predefined structure
    - for instance: details of all research projects
  - not known in advance how it will be stored / managed
- This data may have some structure, but:
  - not all the parts of the data have the same structure
  - each data object may have *different attributes* that are *not* known in advance

Types of data organization

- Unstructured data:
  - very limited indication of the type / structure of data
- Typical examples:
  - a text document with *some* information within it
  - a web page in HTML that contains *some* data
- Example: a cooking recipe in an HTML Document

Flour: 80 cl

Yeast: 10 grams

Water: 80 cl (warm)

Salt: 1 teaspoon

Attention: Cook for 3 hours!

Abstract data models

In this course we study structured data:

- Relational Data Model
  - relations are tables (columns + rows),
  - attributes are columns,
  - tuples are rows
- Sometimes too low level for big companies:
  - designers, programmers, end users understand data and its use in different ways!
- We need a model of communication that is non-technical and free of ambiguities

  $\Longrightarrow$ Entity-Relationship (ER) model

Entity-Relationship (ER) model

- Top-down approach to database design
  - graphical description of the DB
- Basic concepts:
  - the important data objects (entities)
  - the important properties of the entities (attributes)
  - the associations between the entities (relationships)
- Furthermore:
  - constraints on the entities, relationships, and attributes
- Several notations for representing the ER model
  - Crow's foot notation
  - UML notation (Unified Modeling Language)

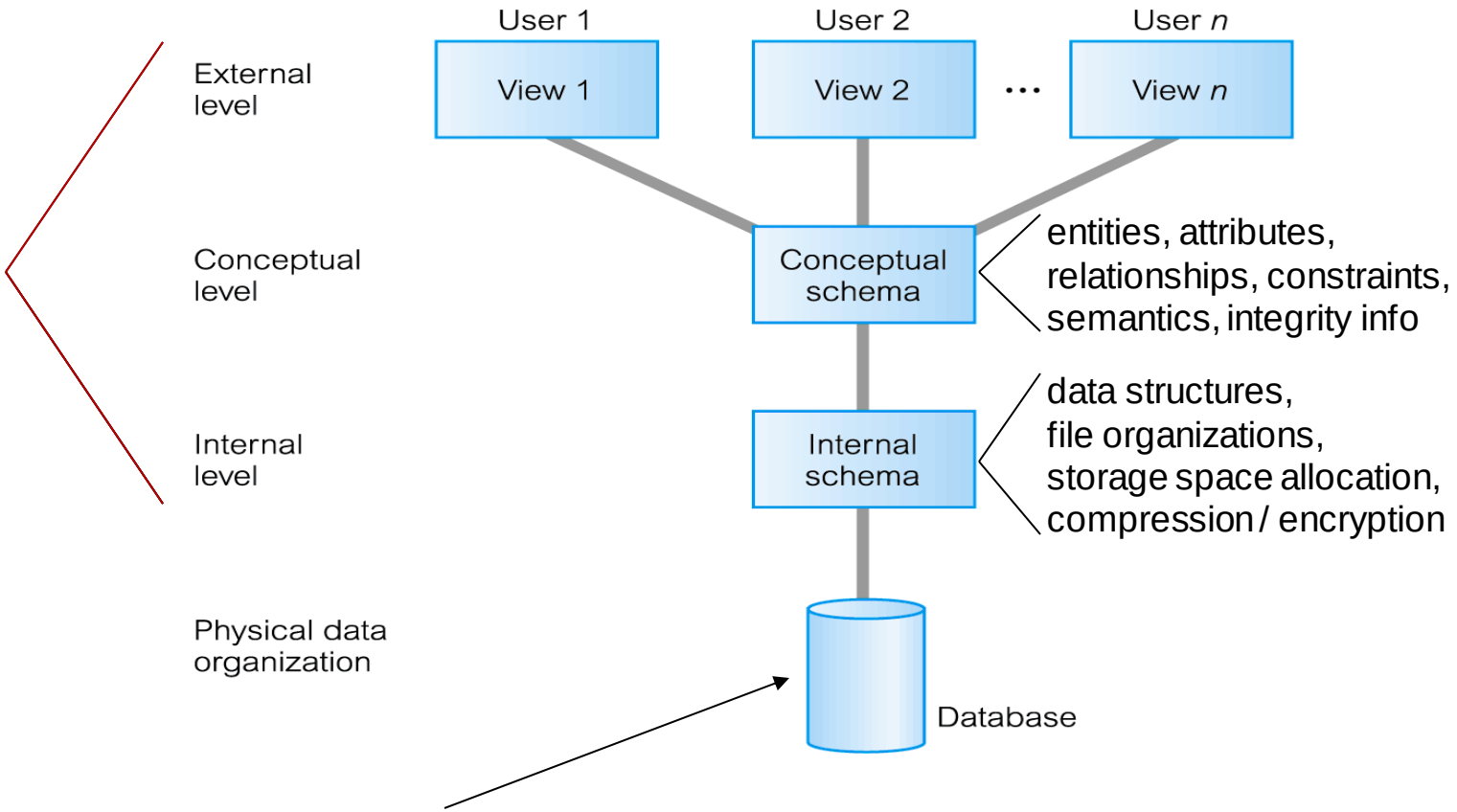
3-Level ANSI-SPARC* Architecture

- External level:
  the part of the data that is relevant to each user
  (user's view of data)
- Conceptual level:
  the logical structure of data,
  as it is seen by the DB Administrator (DBA)
- Internal level:
  physical representation of data in the DB,
  i.e. underlying data structures, algorithms, …

* ANSI-SPARC: *American National Standards Institute – Standards Planning And Requirements Committee*

3-Level ANSI-SPARC* Architecture

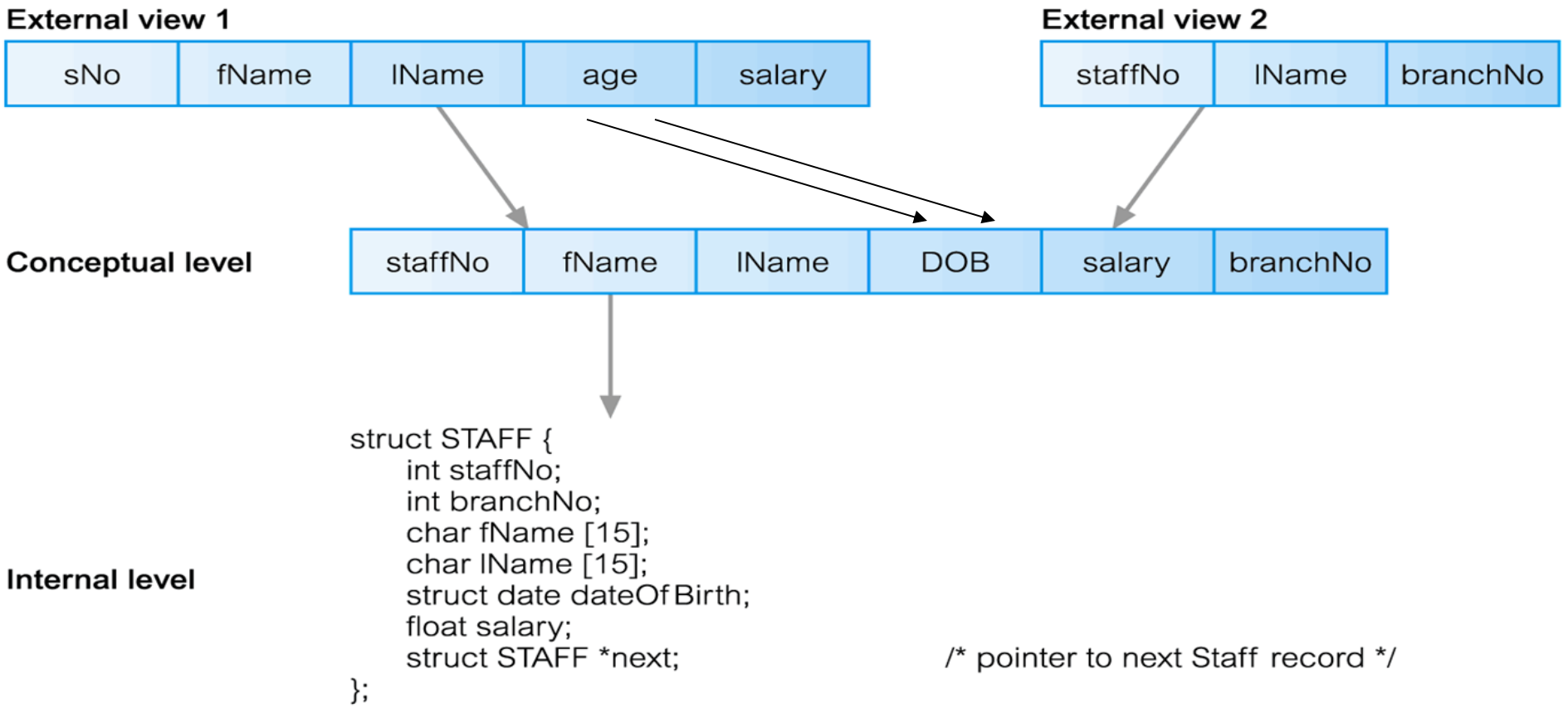
DB schema:



- DBMS: handles size / physical organization of DB
- We focus on the conceptual level

* ANSI-SPARC: *American National Standards Institute – Standards Planning And Requirements Committee*

3-Level ANSI-SPARC* Architecture



* ANSI-SPARC: *American National Standards Institute – Standards Planning And Requirements Committee*

DB schema

- DB schema: total description of the DB
- DB instance: its data at a particular moment

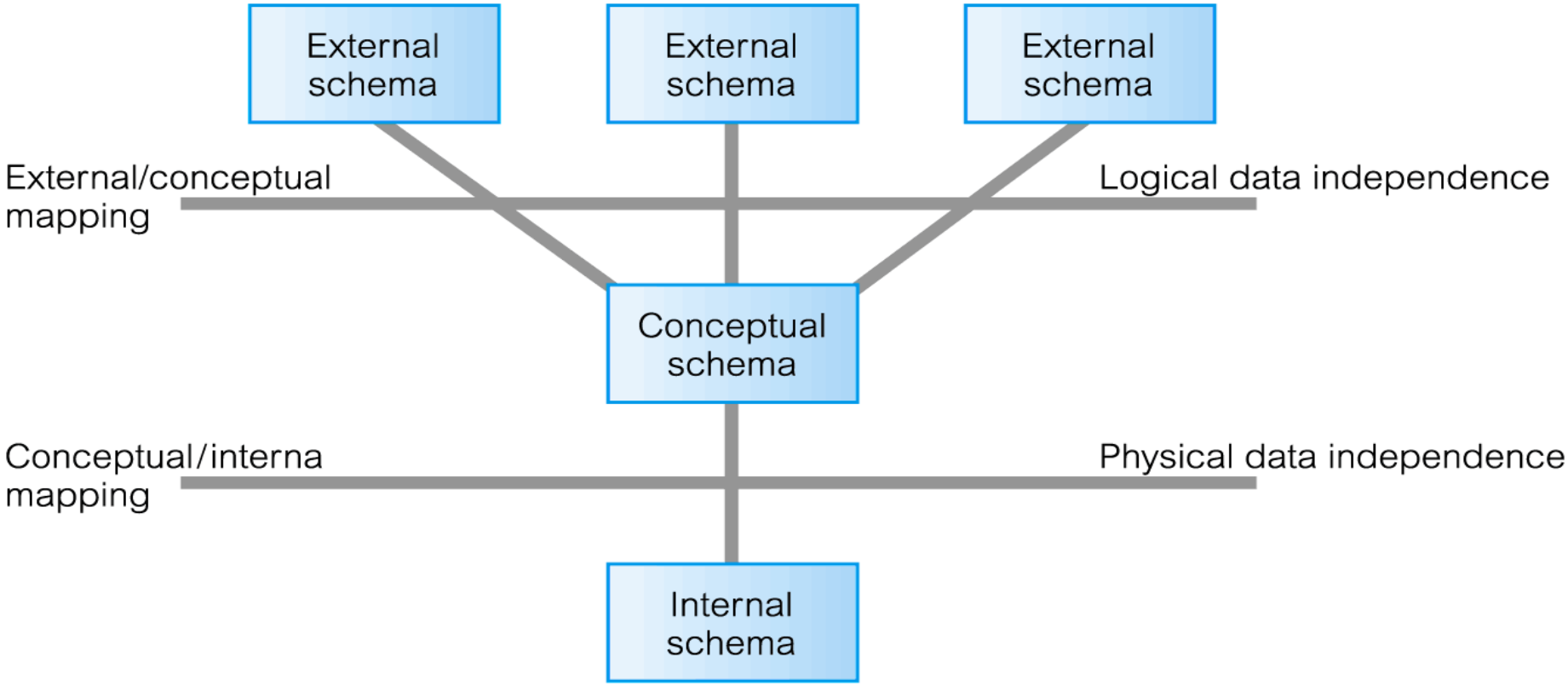
Objectives of a DB schema:

- All users have access at every point:
  - to the same DB instance
  - with customized views of parts of the data
- Data independence:
  - upper levels in the DB schema are *not* affected by changes to lower levels

Data independence

- Logical data independence:
  - External schemas (views) remain the same
    if we change the logical structure of the data
    (i.e. conceptual schema)
- Physical data independence:
  - Conceptual schema remains the same if we change the internal schema (data structures, algorithms, …)
  - Users will notice only change in performance

Data independence

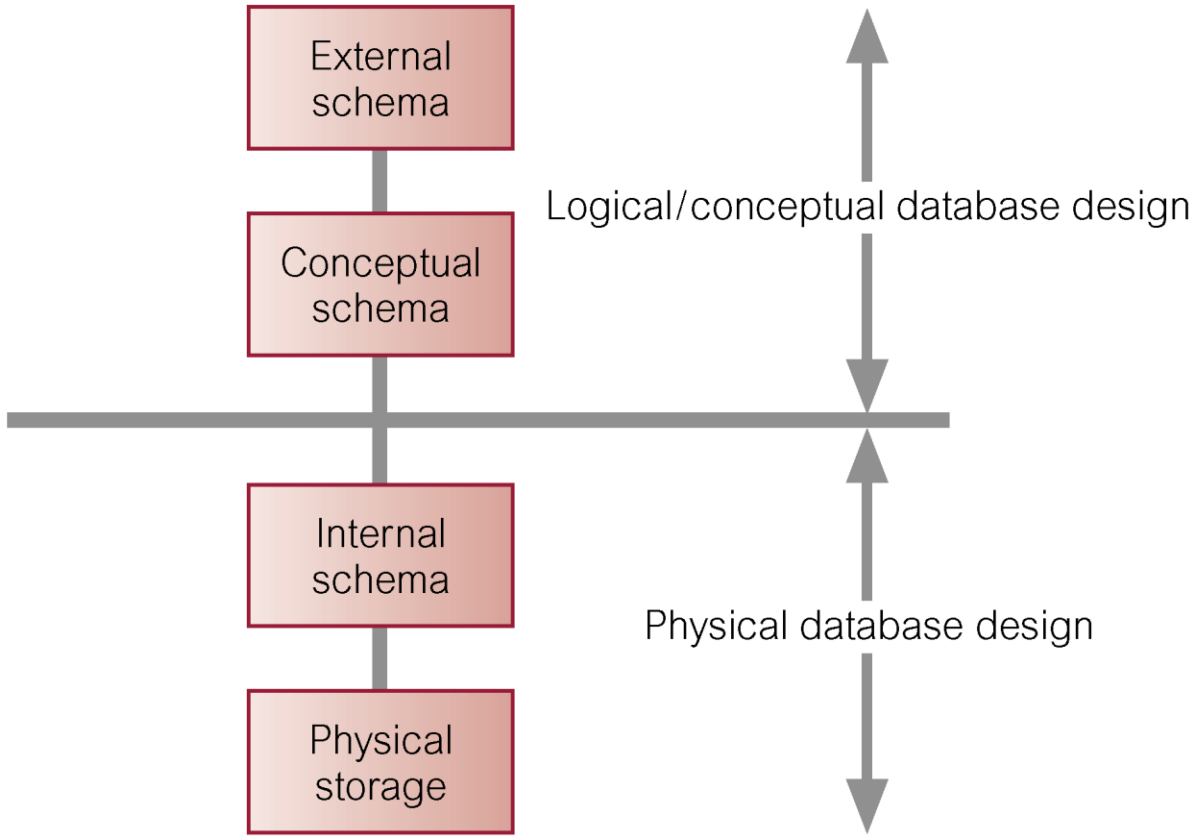


Three main phases of Database Design

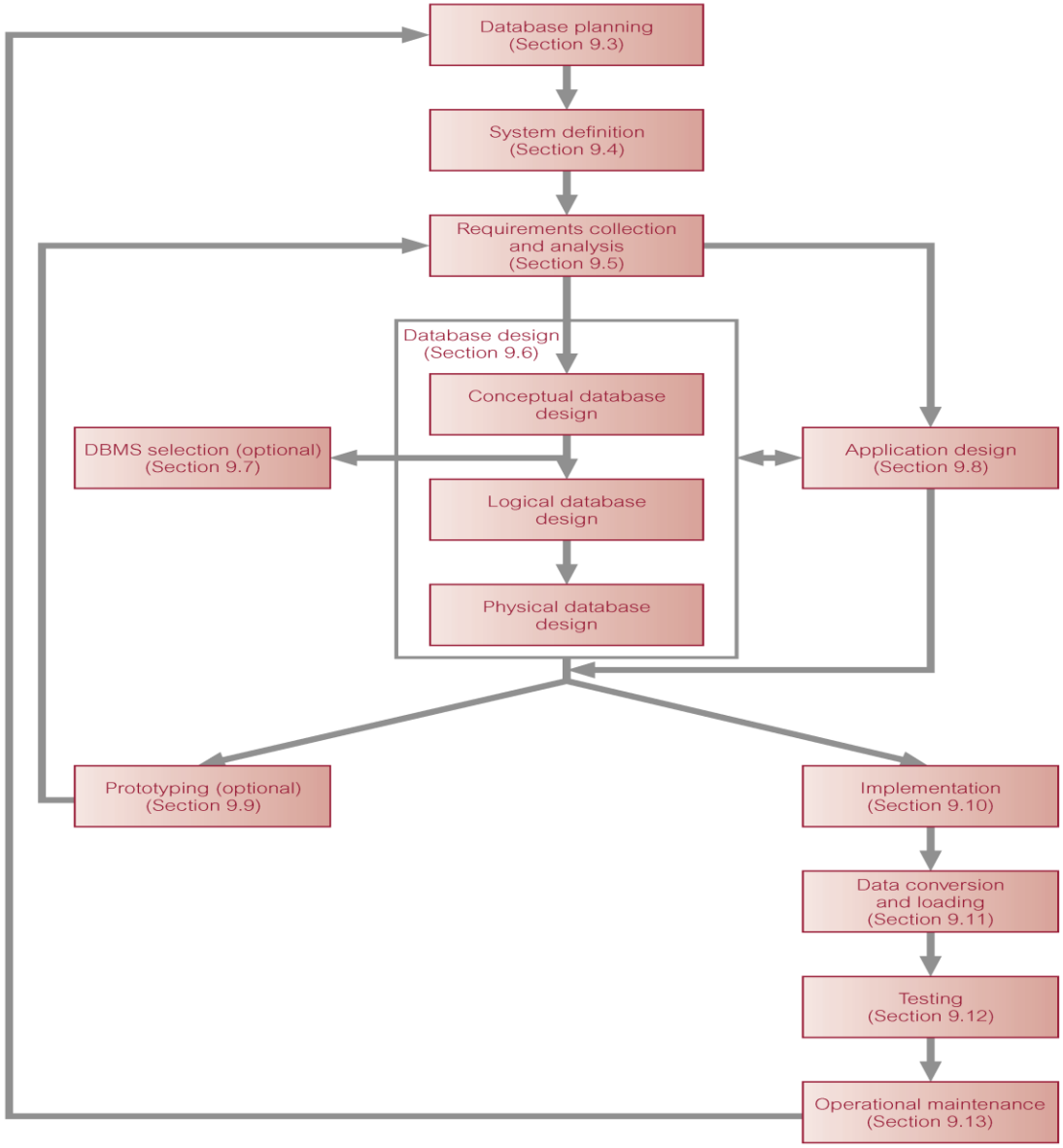
- Conceptual design
  - construct a first, high-level model of the data: ER model
    - identify the appropriate entities, their relationships and their constraints
  - using the users' requirements specification
  - independently of any physical considerations
  - it serves as the fundamental understanding of the system
- Logical design
  - construct the relational data model of the data
  - using the conceptual design
    - map entities / relationships $\longrightarrow$ to tables
  - use normalization techniques to eliminate data redundancy / anomalies
- Physical design
  - describe the database implementation of the logical design
  - specific storage structures / access methods / security protection
  - aim is optimum performance

Three main phases of Database Design

- Classification of the three design phases into the 3-level ANSI-SPARC* Architecture:



All stages of DB system development lifecycle



Make sure you have MySQL access

1. Go to https://community.dur.ac.uk/php.myadmin/
2. Log in with your CIS ID and CIS password to reach the phpmyadmin login page
3. On the phpmyadmin page, log in with your CIS ID and the password you were sent by CIS on 11th January 2021, in an email titled "Durham CIS - MySQL accounts"