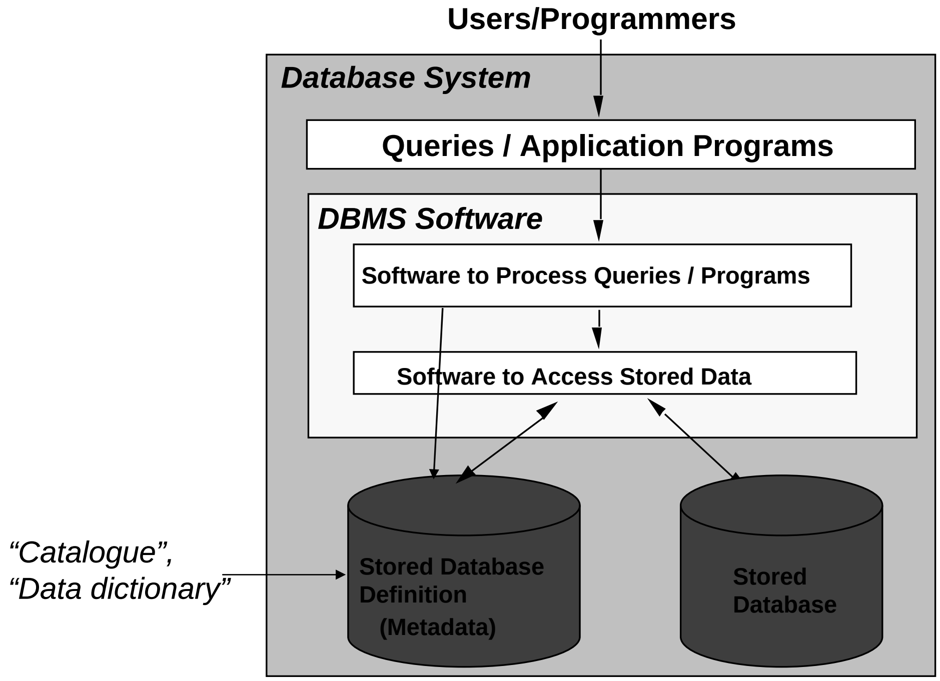
Introduction to Database Systems

Fundamental Concepts

Werner Nutt

1

A DBMS Presents Programmers and Users with a Simplified Environment



2

Data Model, Schema and Instance

Data Model

- A set of concepts that can be used to describe the *structure* of a database: the data types, relationships, constraints, semantics and operational behaviour
- Hides details of data storage

Schema

- A formal definition that fixes all the *relevant features* of those parts of the real world that are of interest to the users of the database
- The schema of a db is held in the *data dictionary*

Schema (in relational data model) →

```
Student(studno,name,address)
Course(courseno,lecturer)
```

Instance →

```
Student(123,Egger,Bozen)
Course(CS321,Nutt)
```

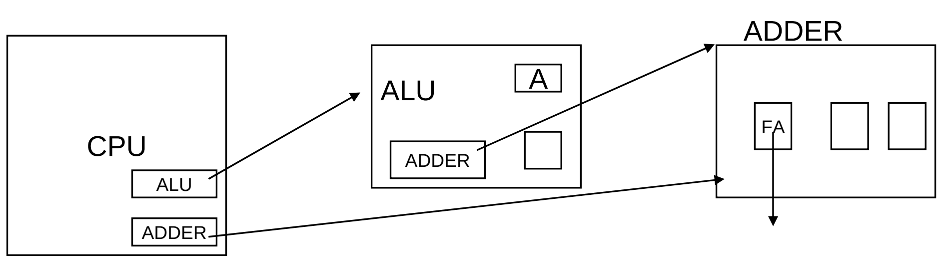

Other Data Models

Relational model is good for:

- Large amounts of data and simple operations
- Limited navigation, touching only small numbers of relations/tables

Difficult applications for relational model:

- VLSI design (CAD in general)



- CASE
- Graphical data
- Bill of materials, transitive closure

Object Data Models

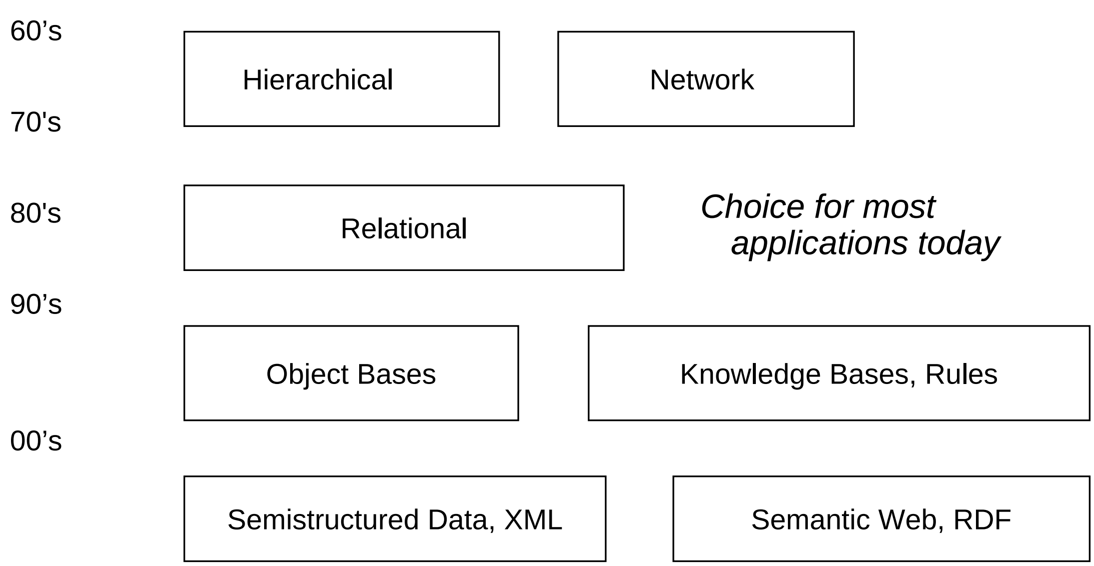
Where number of “relations” is large, relationships are complex
- Object Data Model
- “Knowledge Data Model” (= Objects + Deductive Rules)

Object Data Model (Principles)

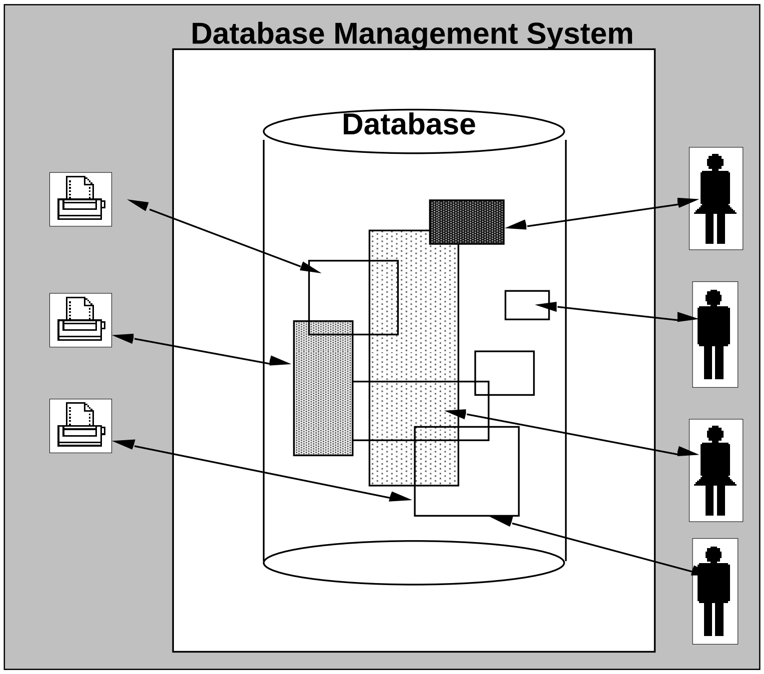
1. Complex Objects –
   Nested Structure (pointers or references)
2. Encapsulation, set of methods/access functions
3. Object Identity
4. Inheritance – Defining new classes like old classes

Object model: usually, objects are found via explicit navigation.
Also query language in some systems.

Data Models



Sharing—Multiple *views* of data



Characteristics of the DB Approach

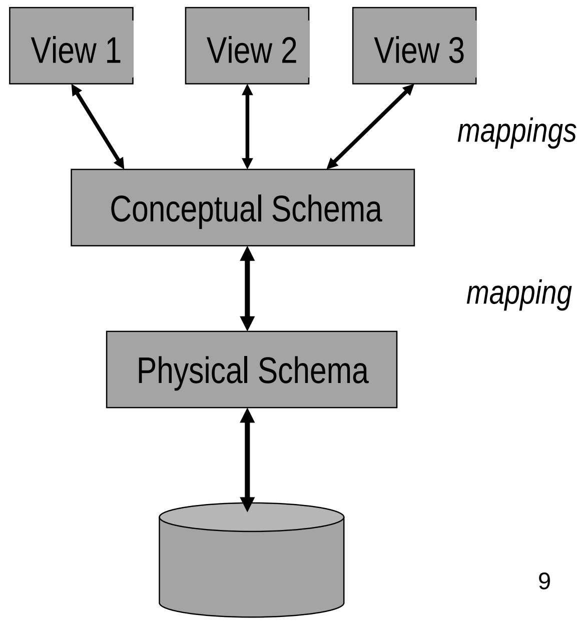
- *Insulation* of programs and data from each other
- Support of *multiple user views*
- Use of a *catalogue* to store the schema

➔ *How can one realise these principles?*

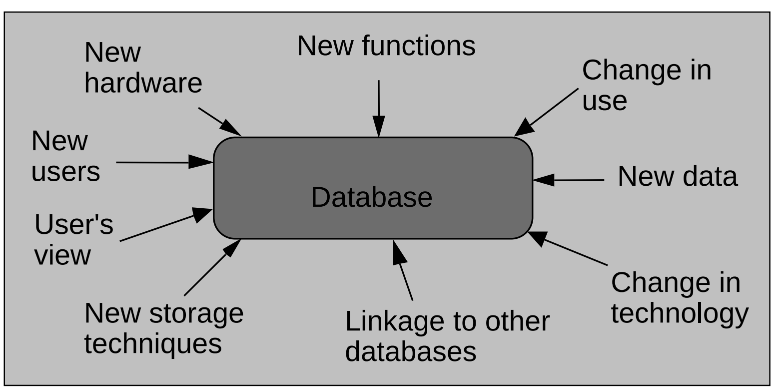
Three Levels of Abstraction

ANSI/SPARC architecture for DBMSs (1978):

- Many *external views*
- One *conceptual* (= logical) *schema*
- One *physical* (= internal) *schema*
  - Views describe how users see the data
  - Conceptual schema defines logical structure
  - Physical schema describes the files and indexes used



Data Independence



- **Logical** data independence
  - change the logical schema without having to change the external schemas
- **Physical** data independence
  - change the internal schema without having to change the logical schema

Change the mapping, not the schema!

Database Languages

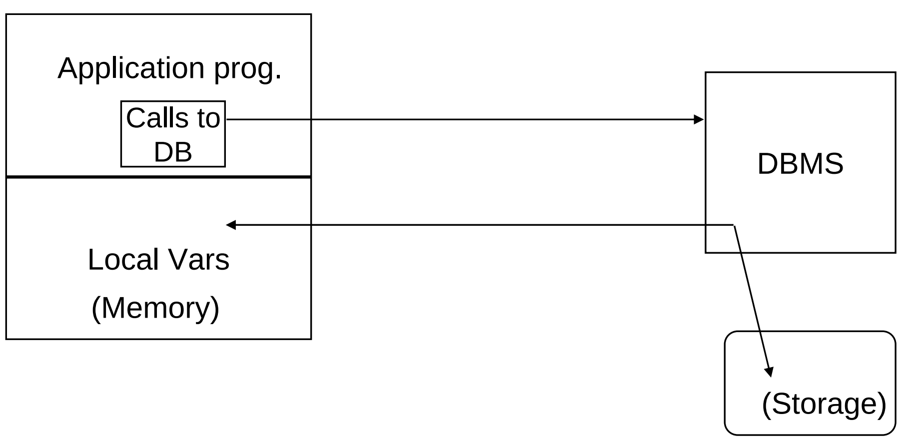
- Data Definition Language (DDL)
  - Commands for setting up the schema of a database
  - The process of designing a schema can be complex, may use a design methodology and/or tool
- Data Manipulation Language (DML)
  - Commands to manipulate data in database:

    RETRIEVE, INSERT, DELETE, MODIFY

  - Also called "query language"

Host Languages

C, C++, Fortran, Lisp, Java, Perl, …



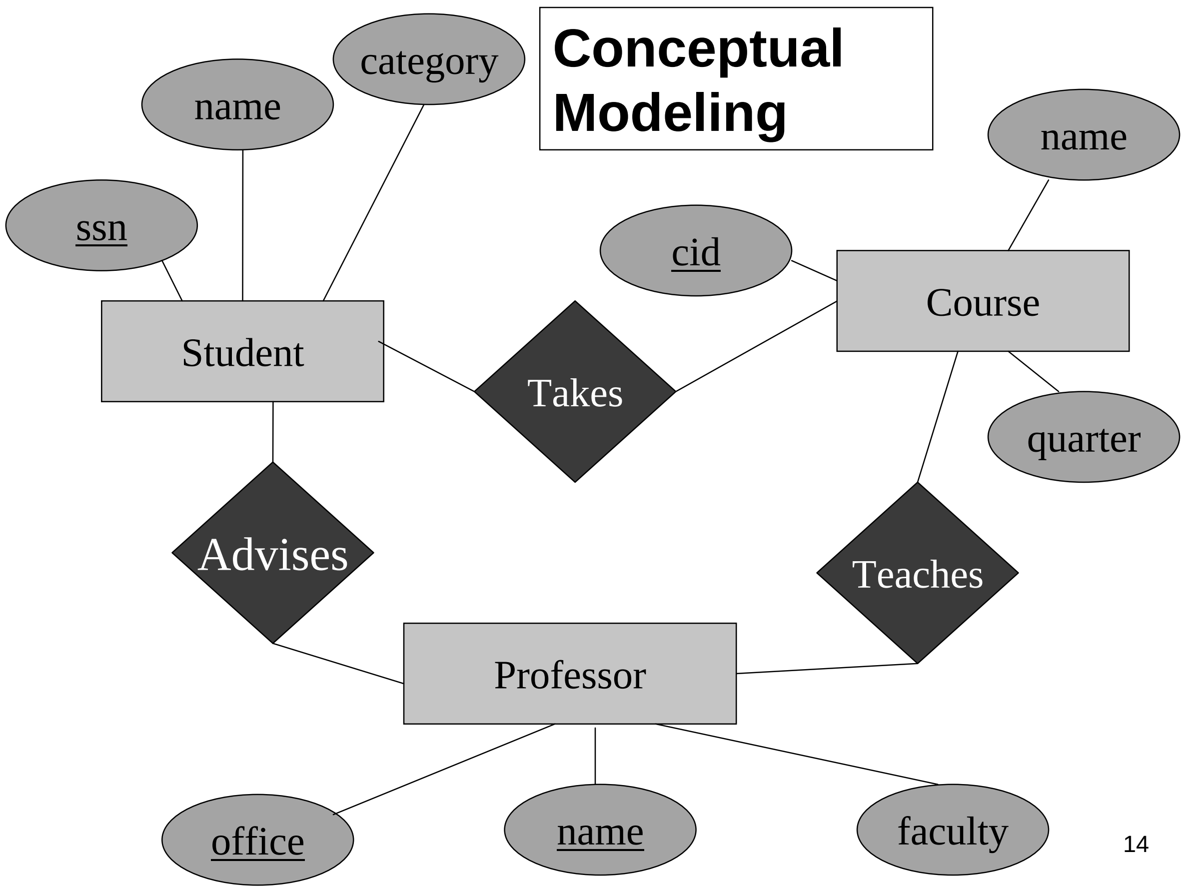
Host language is completely general (Turing complete) but gives no support for data manipulation

Query language—less general, "non procedural" and optimizable

Building an Application with a DBMS

- Requirements gathering (natural language, pictures)
- Requirements modeling (conceptual data model, ER)
  - Decide what *entities* should be part of the application and how they should be *related*
- Schema design and implementation
  - Decide on a set of *tables*, *attributes*
  - Create the tables in the database system
  - Populate database (insert records/tuples)
- Write application programs using the DBMS
  - … a lot easier now that the data management is taken care of

Conceptual Modeling



Schema Design and Implementation

- Tables:

Student:

SSN	Name	Category
123-45-6789	Charles	undergrad
234-56-7890	Dan	grad
	...	...

Takes:

SSN	CID
123-45-6789	CSE444
123-45-6789	CSE444
234-56-7890	CSE142
	...

Course:

CID	Name	Quarter
CSE444	Databases	fall
CSE541	Operating systems	winter

- The logical schema separates the logical view from the physical view of the data.

Querying a Database

- *"Find all courses that Mary takes"*
- **S**(tructured) **Q**(uery) **L**(anguage)

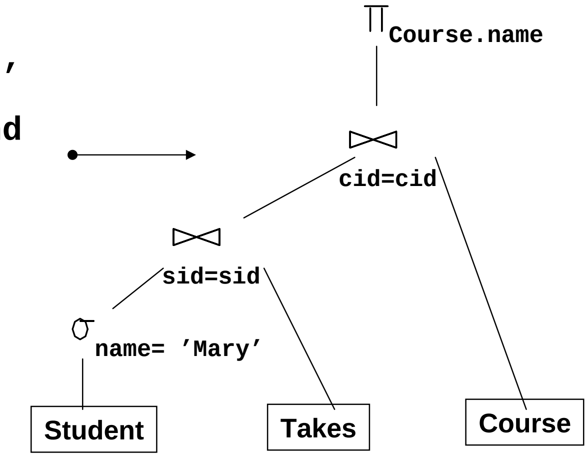
```
select c.name
from   Student s, Takes t,
       Course c
where  s.name = 'Mary' and
       s.ssn  = t.ssn and
       t.cid  = c.cid
```

- The query processor figures out how to answer the query efficiently

Query Optimization

Goal: *Declarative SQL query* → *Query execution plan*

```
select c.name
from   Student s, Takes t,
       Course c
where  s.name = 'Mary' and
       s.ssn  = t.ssn and
       t.cid  = c.cid
```



Plan: Tree of relational algebra operators,
choice of algorithm for each operator

Ideally: Find best plan   Practically: Avoid worst plans!

Traditional and Novel Data Management Issues

- Traditional Data Management:
  - Relational data for enterprise applications
  - Storage
  - Query processing/optimization
  - Transaction processing
- Novel Data Management:
  - Integration of data from multiple databases, warehousing
  - Data management for decision support, data mining
  - Managing documents, audio, and visual data
  - Exchange of data on the web: XML
  - Data Streams
  - Incomplete and probabilistic data