

DataBase (Management) Systems

DBMS

Aim:

- Overall (simplified, abstract) view
- Basic notions
- Words

Notion of a Database

What is data?

What is information?

What is a Database?

ABKR=AdatBázisKezelőRendszer

DBMS=DataBaseManagementSystem

Notion of a Database

Huge amount of structured information which can be accessed and manipulated efficiently through a user-friendly software. This structured information and the software together can be called as DBMS. The part in which the structured information is stored is the Database.

The DBMS is defined rather through its properties.

Properties:

1.

- New database can be constructed
- Logical structure of data can be described

Adatdefiníciós Nyelv

Data Definition Language = DDL

2.

- Data can be get back through queries efficiently
- Lekérdező nyelv-Data Query Language

- Data can be modified easily

Adatmanipulációs nyelv-Data Manipulation Language (DML)

3.

- Safe storage:

- it must be defended against unauthorized users

Grants

- it must be defended against failors:

DBMS Recovery Systems

4. Concurrent usage

Transaction Management

Concurrency Control

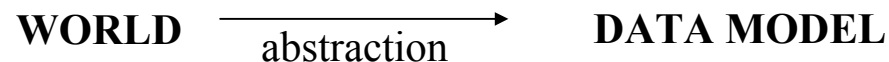
Data(base) models

Data models serve for describing the structure of the database. The process of it is an abstract phase, in which the first step is gathering the facts of the real world we want to store, and also the interrelationships among these data. In the second step the facts and properties must be represented in a standardized way in order to implement them in the computer. This standard representation, the output of that phase, is the data model.

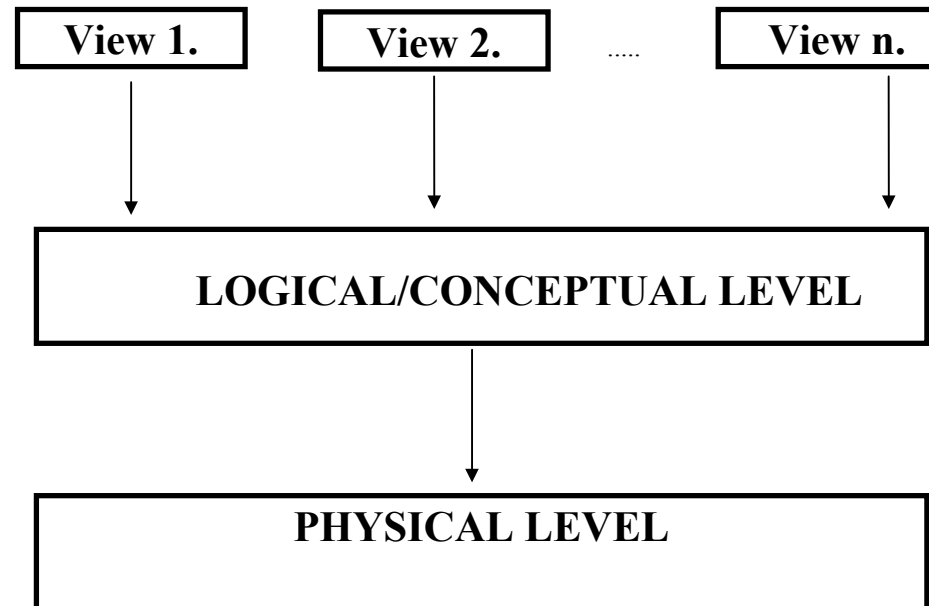
In a data model we do not care with the concret values of a data, rather only the relationships among these data.

Many database can satisfy a given database model. The structured data, with the given values, satisfying the database model is called the **database instance**.

The notions mentioned above will be detailed later.



Data abstraction



I. View level: What a user can see from the database –not the whole model can be seen (subschemas - SDDL)

II. Logical/Conceptual level szint: The complete database model (schema - DDL),

III. Physical level: Storage and access methods, files, indeces, other storing structures

Data independence:

logical: logical level can be modified without modifying physical level

physical: physical level can be modified without modifying logical level

Example: Consider an $n \times m$ array.

Physical level: $m \times n$ (consecutive) memory location

Logical level:

View level: $2^{(m \times n)}$ possible view

Model: logical level schema

```
type tgrid:=array[1..2, 1..3] of integer;
var vgrid1, vgrid2:=tgrid;
```

An instance:

1	2	3
4	5	6

Logical/Conceptual level

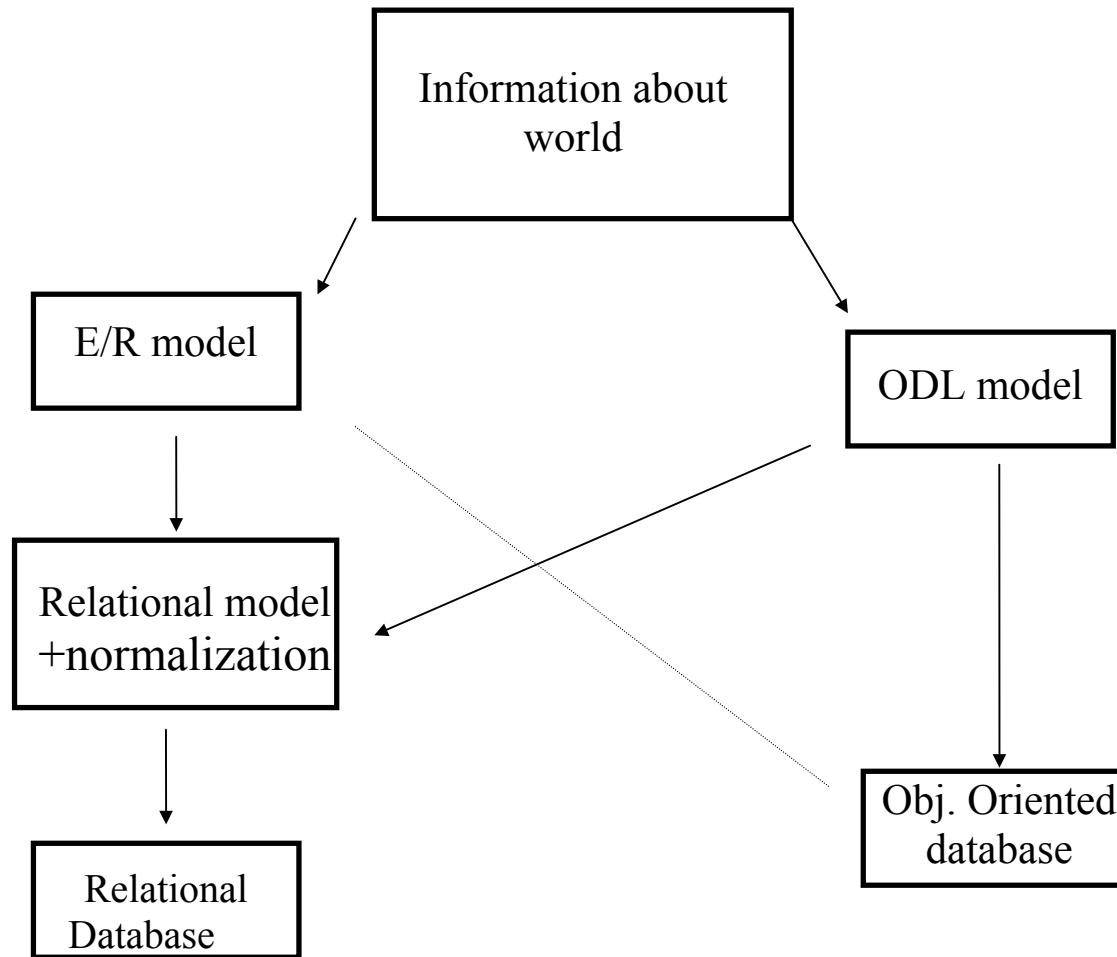
Object based models:

- **Entity/Relationship diagram (graphical standard)**
- Objectumoriented model, ODL (C++ like standard for describing data structures)

Record based models:

- **Relational model**
- Network model
- Hierarchical model

Database modelling and implementation



History by topic

Examples:

Banking systems

User Programs:

- accounts-deposits, withdraws
- new accounts
- balance
- monthly reports

Data items:

customers(name, address, account-no)

accounts(account-no, balance, type)

Type could be: savings or checking

(Airlines) Seat reservation systems

Data items:

customers(name, address, phone, flight-number)

flights(flight-number, departure, arrival, airport)

seats(flight-number, seat-id., name)

Company administration

- sells, bills (in-out)
- what could be the data items?

File management systems

- Files could have different structures (sequential, ..)
- New questions – new programs
- Data could be stored in different storage places*which make writing new programs difficult
- Constraints can be hardly checked
- Atomicity level is the whole file
- Concurrency is almost impossible

- Safety can not be guaranteed
- Abstraction level is low-the programmer should know the physical level for writing new programs

History by models

First DBMS

- network model (graph)
 - hierarchical model (tree)
-

- relational model (Codd, 1970)

Who use DBMS?

Administrator:

Tasks:

- logical schema modification
- logical schema definition
- physical schema modification
- constraints definitions
- constraints modifications
- authorization
- routine maintenance

Other users:

- programmers
- naive users
- sophisticated users

DBMS STRUCTURE

Schema definition

Storage manager (program):

- authorization and integrity manager
- file-manager
- buffer manager

Transaction manager (program):

Controls the correct concurrent running

- Atomicity: all-or-none
- Consistency: correctness
- Isolation: if I were alone..

Tools: locking, validation, time-stamping

- Durability: ..forever...?

Tools: journal-files, mirroring..

Query processing (programs):

- DDL / DML interpreter / compiler
- DML precompiler
- Query evaluation-optimization

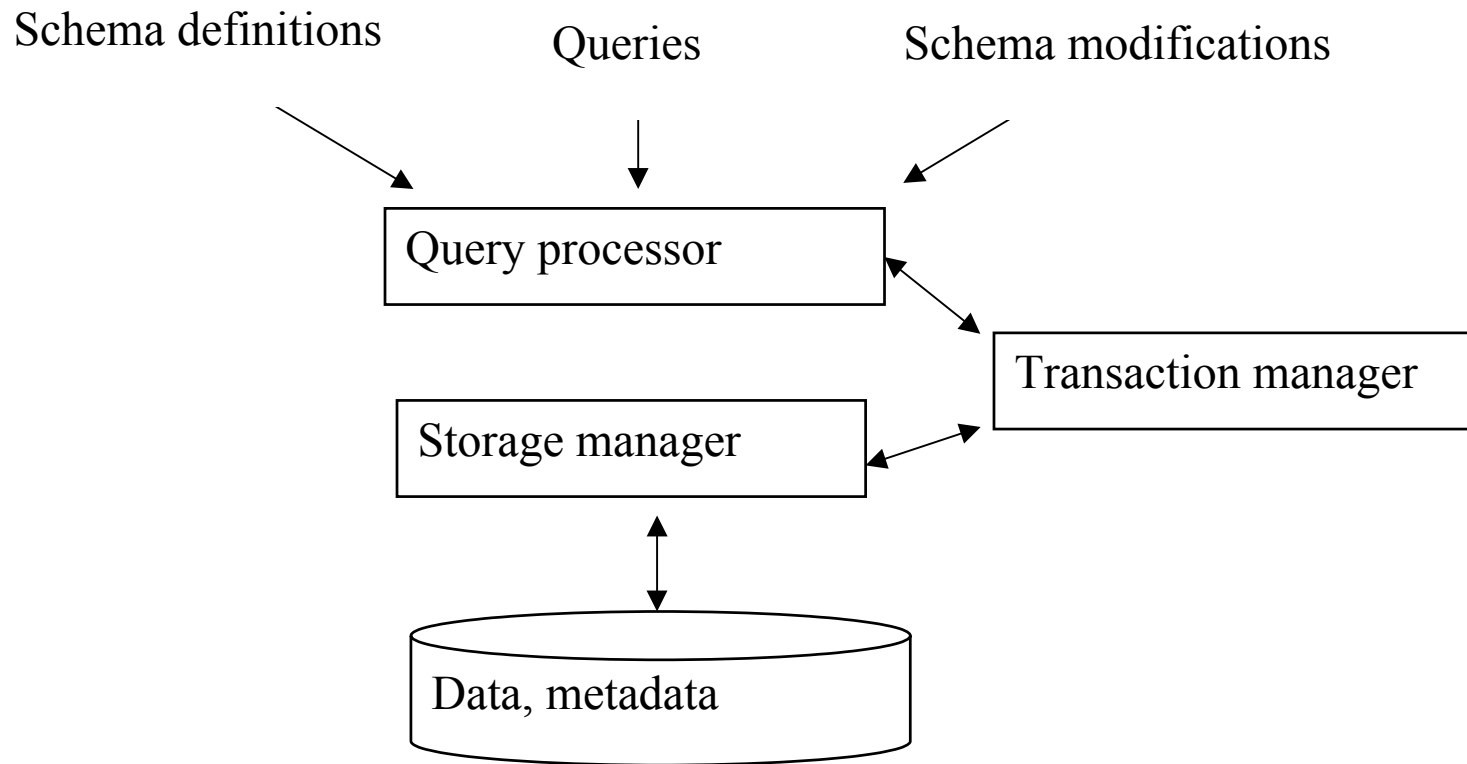
Physical storage (device):

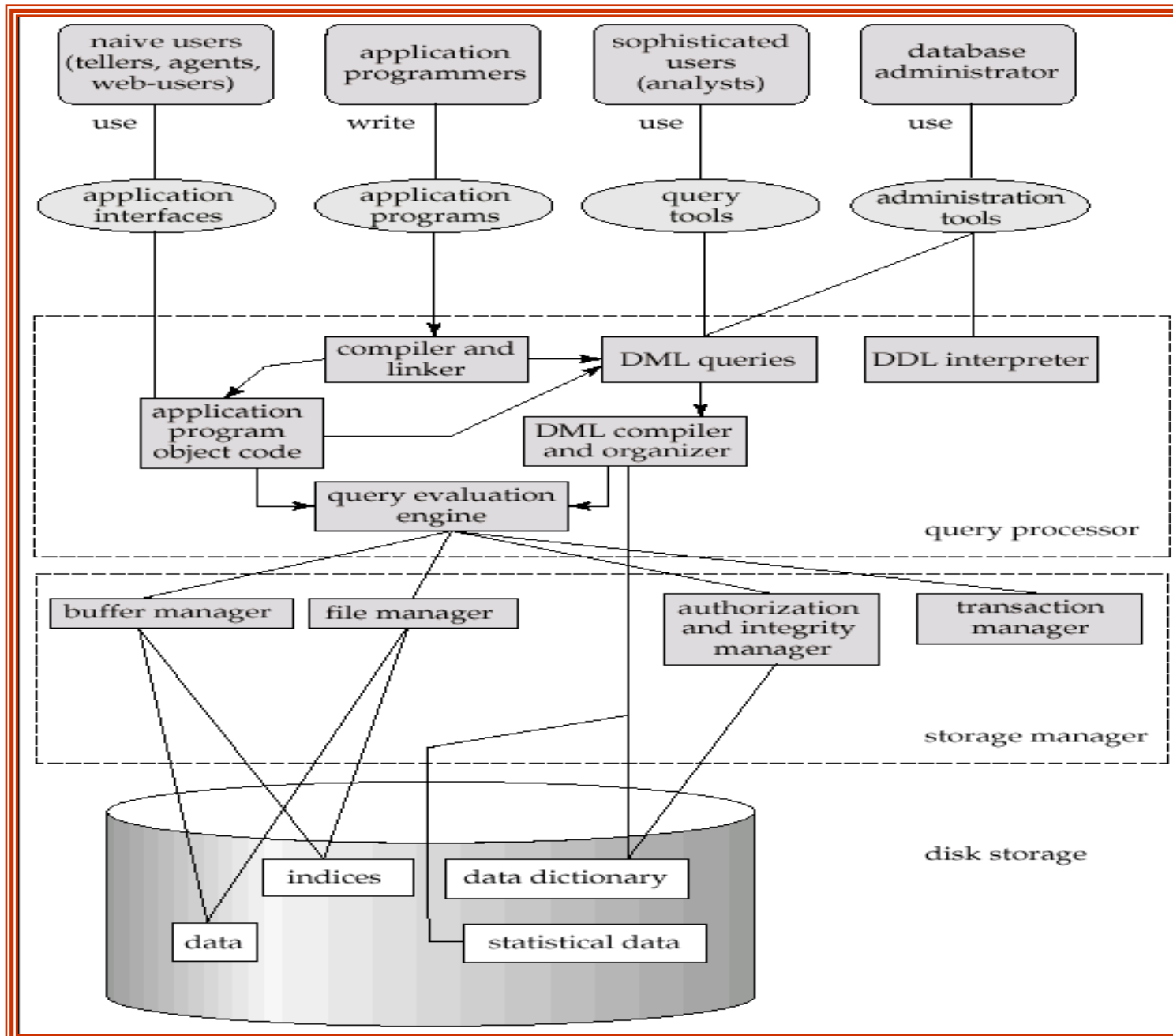
- real data (the „pure” database)
- metadata:
 - indexes
 - statistical data
 - data dictionary-records data structures

DBMS STRUCTURE (cont.)

Klient - server architectures

DBMS simplified structure





Other directions (will be partly covered):

- Object oriented database systems
- -Activev elements:
 - constraints
 - triggers
- Multimedia databases
- Data Warehouse

Summary

- Database: huge amount of information
- DBMS: 1. DDL
 - 2. DML
 - 3. Safety
 - 4. Concurrency

Each part must be efficient and „easy” to use

DBMS history:

- by topic
- by model

■ DBMS structure:

- User interface
- Query Processor
- Storage manager
- Transaction manager
- Physical storage

■ New directions:

multimedia objects, data warehousing, information standards