

1 Definitions

In the common understanding the term database [1] refers to data set organized by database management system (DBMS) or simply a database system.

Database [3] - accessible to many users set of logically connected data along with a definition of their structure, designed to meet the needs of data processing by institutions.

Expectations towards the DBMS system [1]:

- "Allow users to create new databases and specify their schema (logical structure of the data), using a specialized *data definition language*."
- "Give users the ability to query the data (a 'query' is database lingo for a question about the data, using an appropriate language, often called *query language* or *data-manipulation language*."
- "Support the storage of very large amounts of data - many terabytes or more - over a long period of time, allowing efficient access to the data for queries and database modifications."
- "Enable *durability*, the recovery of the database in the face of failures, errors of many kinds or intensional misuse."
- "Control access to the data from many users at once, without allowing unexpected interactions among users (called *isolation*) and without actions on the data to be performed partially but not completely (called *atomicity*)."

Advantages of DBMS [3]:

- data redundancy control,
- data consistency,
- more information based on the same data,
- unified access to data,
- data integrity improvement,
- security improvement,
- standardization,
- cost savings,
- reconcile contradicting requirements,
- scope and time of access to data improvement,
- performance increment,
- easier maintenance resulting from data independence,
- capabilities of multi user access increment,
- extended backup and disaster recovery support.

Disadvantages of DBMS [3]:

- complexity,
- size,
- cost,
- extra hardware costs,
- migration costs,
- greater failure range.

2 Data model

Data model [3] - It is a coherent set of concepts used to describe data and relationships between them and to manipulate the data and the relationships, as well as to do expressing constraints imposed on data in an institution.

In data model [3] the following three parts can be distinguished:

- Structure of the data - it contains set of rules describing how databases should be constructed,
- Operations on the data - it describes types of operations allowed to be executed on the data,
- Constraints on the data - optional part containing set of integrity rules, which guarantee data consistency.

Categories of data models [3]:

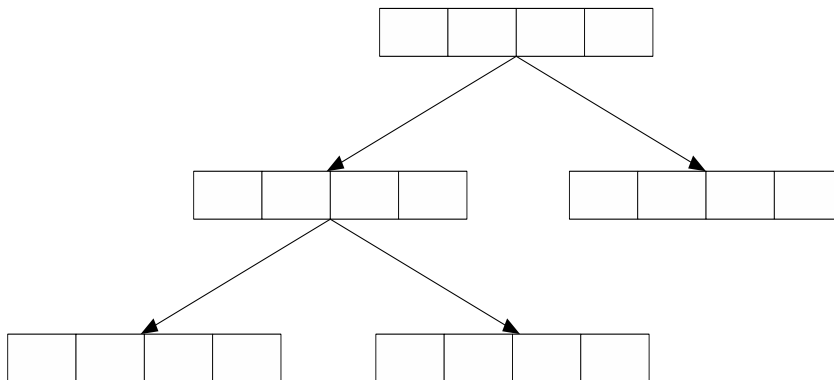
- based on objects,
- record,
- physical.

In record data model [3] database consists of a certain number of given format records, which may belong to different types. Every record type defines given number of fields of usually fixed lengths.

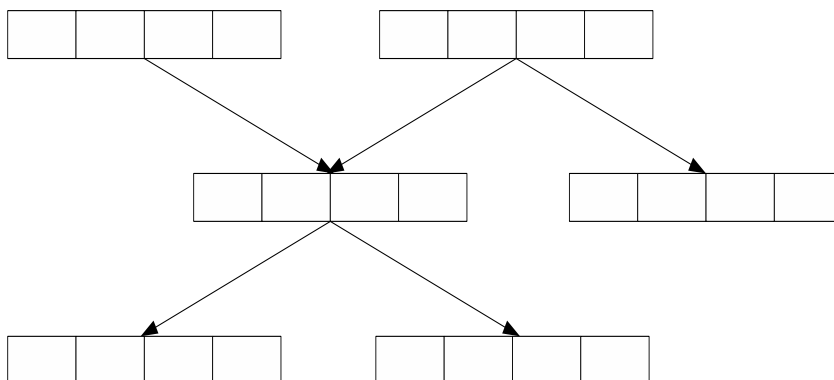
Record data models [3]:

- network data model,
- hierarchical data model,
- relational data model.

In a hierarchical data model [3] data are represented as collections of records. In this model node can have only one father. Graph representing the model is a tree, records are nodes and sets are edges.



In network data model [3] data are organized as collections of records, a relationships between them are presented as sets. Records are organized as generalized graph structures, where records are nodes and sets as graph edges.



Relational model [3] is based on the mathematical concept of relation. Data and relationships are represented as tables, where each of them contains a number of columns with unique identifiers.

The relational model is based on the concept of relation, proposed by E.F. Codd from IBM in 1970. The SQL language is a standard that allows you to define data and operations on it. In 1985 Codd proposed set of rules for relational databses [3].

Rule 0 - Foundational rule

"For any system that is advertised as or claimed to be a relational database management system, that system must be able to manage databases entirely through its relational capabilities."

Rule 12 - Nonsubversion rule

"If a relational system has a low-level (single-record-at-a-time) language, that low level cannot be used to subvert or bypass the integrity rules and constraints expressed in the higher-level relational language (multiple-records-at-a-time)."

Rule 1 - Information representation

"All information in a relational database is represented explicitly at the logical level and in exactly one way: by values in tables."

Rule 6 - View updating

"All views that are theoretically updatable are also updatable by the system."

Rule 3 - Systematic treatment of null values

"Nulls (...) are supported for representing missing information and inapplicable information in a systematic way, independent of data type."

Rule 10 - Integrity independence

"Integrity constraints specific to a particular relational database must be definable in the relational data sublanguage and storable in the catalog, not in the application programs."

Rule 2 - Guaranteed access

"Each and every datum (atomic value) in a relational database is guaranteed to be logically accessible by resorting to a combination of table name, primary key value, and column name."

Rule 4 - Dynamic online catalog based on the relational mode

"The database description is represented at the logical level in the same way as ordinary data, so that authorized users can apply the same relational language to its interrogation as they apply to the regular data."

Rule 5 - Comprehensive data sublanguage

"A relational system may support several languages and various modes of terminal use (for example, the fill-in-the-blanks mode). However, there must be at least one language whose statements can express all of the following items: (1) data definition; (2) view definition; (3) data manipulation (interactive and by program); (4) integrity constraints; (5) authorization; (6) transaction boundaries (begin, commit, and rollback)."

Rule 7 - High-level insert, update, delete

"The capability of handling a base relation or a derived relation (that is, a view) as a single operand applies not only to the retrieval of data but also to the insertion, update, and deletion of data."

Rule 8 - Physical data independence

"Application programs and terminal activities remain logically unimpaired whenever any changes are made in either storage representations or access methods."

Rule 9 - Logical data independence

"Application programs and terminal activities remain logically unimpaired when information-preserving changes of any kind that theoretically permit unimpairment are made to the base tables."

Rule 11 - Distribution independence

"The data manipulation sublanguage of a relational DBMS must enable application programs and inquiries to remain logically the same whether and whenever data is physically centralized or distributed."

Data models based on objects [3]:

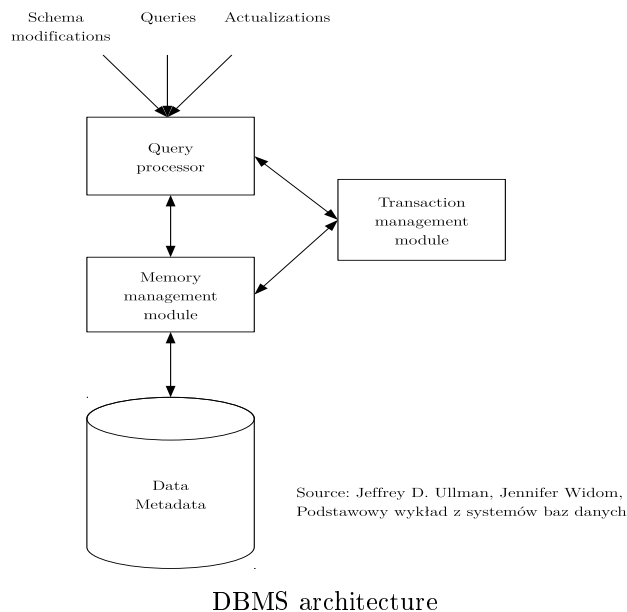
- entity-relationship model,
- semantic model,
- functional model,
- object model.

Object model, object-oriented model [3] - data model which basis are concepts of object, class, inheritance, polymorphism and other object-oriented features. The object model is often understood as a variant entity - relationship model, where in addition methods are assigned to entities (object classes).

Physical data models [3] represent how the data are stored in a computer. They provide following information: records structure, record arrangement and access paths.

3 DBMS architectures

3.1 Main elements of DBSM



DBMS architecture

Transaction management module [1] is responsible for system integrity. It must guarantee that simultaneously processed queries will not interfere with each other and no data will be lost even if a system failure occurs.

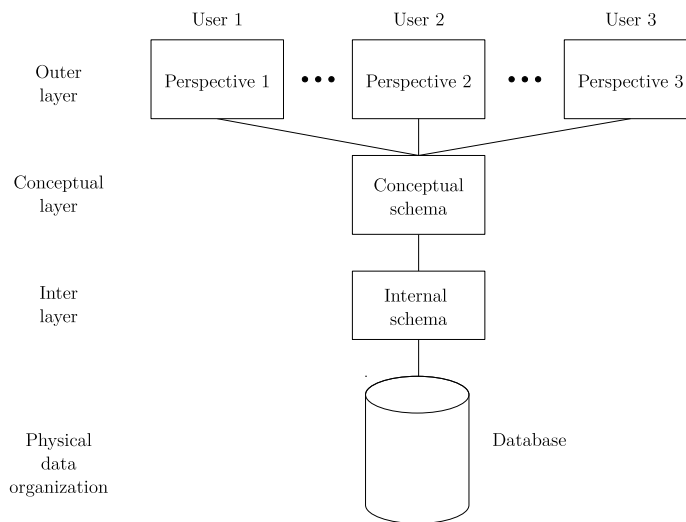
Memory management module [1] consists of two parts:

1. Files management module stores information about location where files are stored on a disk and on request of buffers management module sends contents of the block (blocks) where a file is stored.

2. Buffers management module handles operational memory. Files management module transfers data blocks from disk, and buffers management module choose pages in operational memory, which will be allocated to the blocks.

Query management module [1] transforms query or operation on a database, expressed usually in high level language (for example SQL), into a series of commands requesting delivery of data.

3.2 Three-tier ANSI-SPARC architecture



Source: T. Connolly, C. Begg, Systemy baz danych, RM, 2004

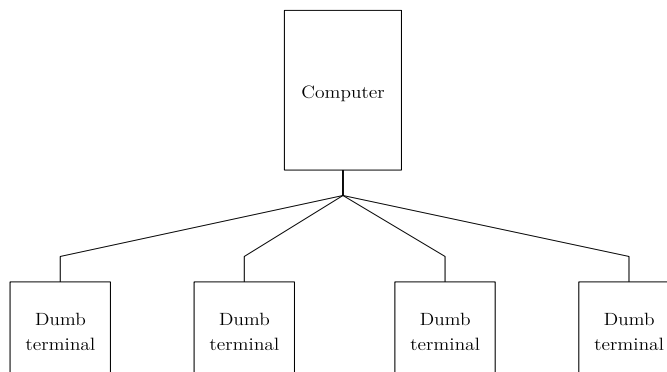
Three-tier ANSI-SPARC architecture

Three-tier ANSI-SPARC architecture consists of following layers [3]: outer layer, inter layer and conceptual layer. The first one represents how data is seen by users and the second one represents how the data is seen by operational system and DBMS. The conceptual layer is responsible for mapping between these two layers and ensures its independence.

3.3 Multi-user DBMS Architectures:

Multi-user DBMS Architectures:

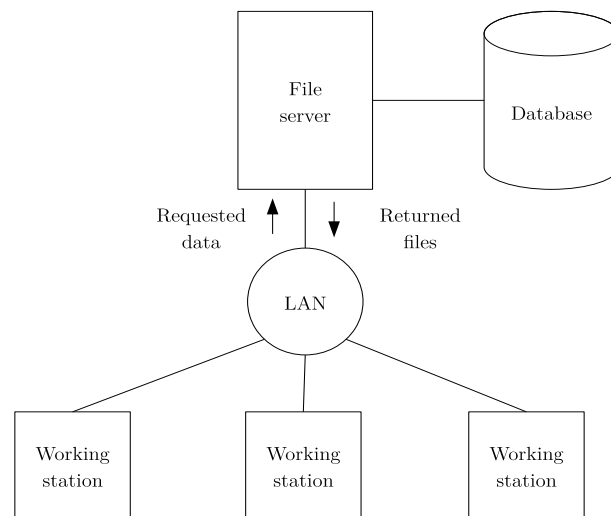
- remote processing,
- file server,
- client - server,
- other, for example three-tier client-server architecture.



Source: Jeffrey D. Ullman, Jennifer Widom,
Podstawowy wykład z systemów baz danych

Remote processing

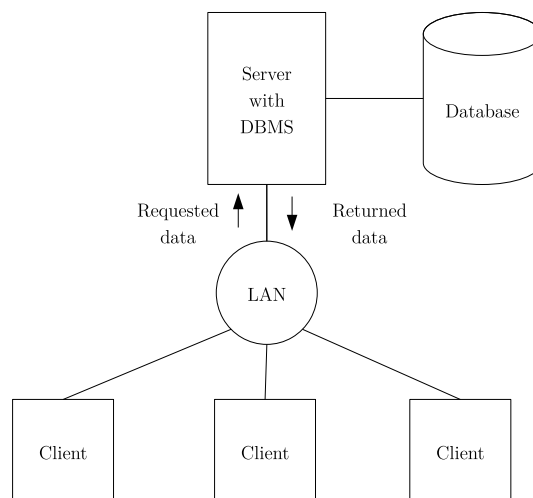
User terminal [3] were unable to operate independently. There were used to send messages to application and back through subsystem of operating system responsible for controlling communications. The application were using DBMS services.



Source: Thomas Connolly, Carolyn Begg,
Systemy baz danych, Wydawnictwo RM, 2004

File server

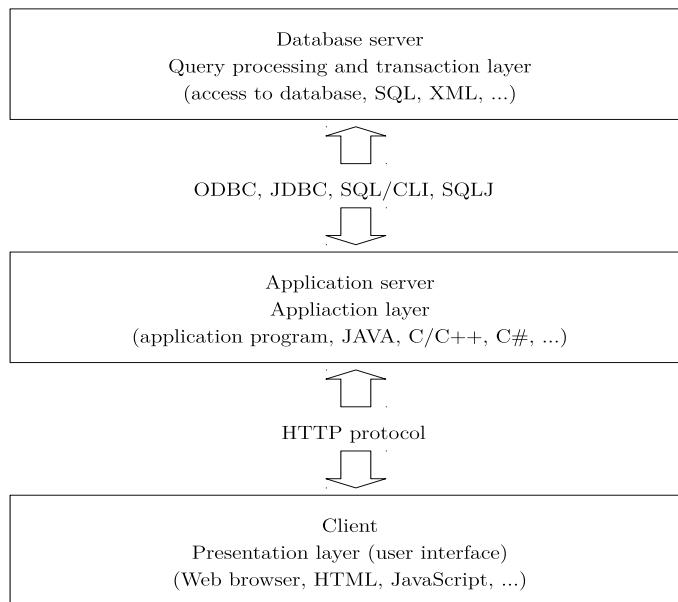
File server responsibility [3] is storing files used by applications and DBMS. The applications and DBMS are executed on every working station and refer to files on the file server.



Source: Thomas Connolly, Carolyn Begg,
Systemy baz danych, Wydawnictwo RM, 2004

Client - server

A client [3] handle user interface and control in an application. Client accepts user requests, validates them and creates a database request. Then it sends them as a message to the server, waits for a response and formats it according to requirements of a user. A server [3] receives and processes the request to the database and then sends the results back to a client.

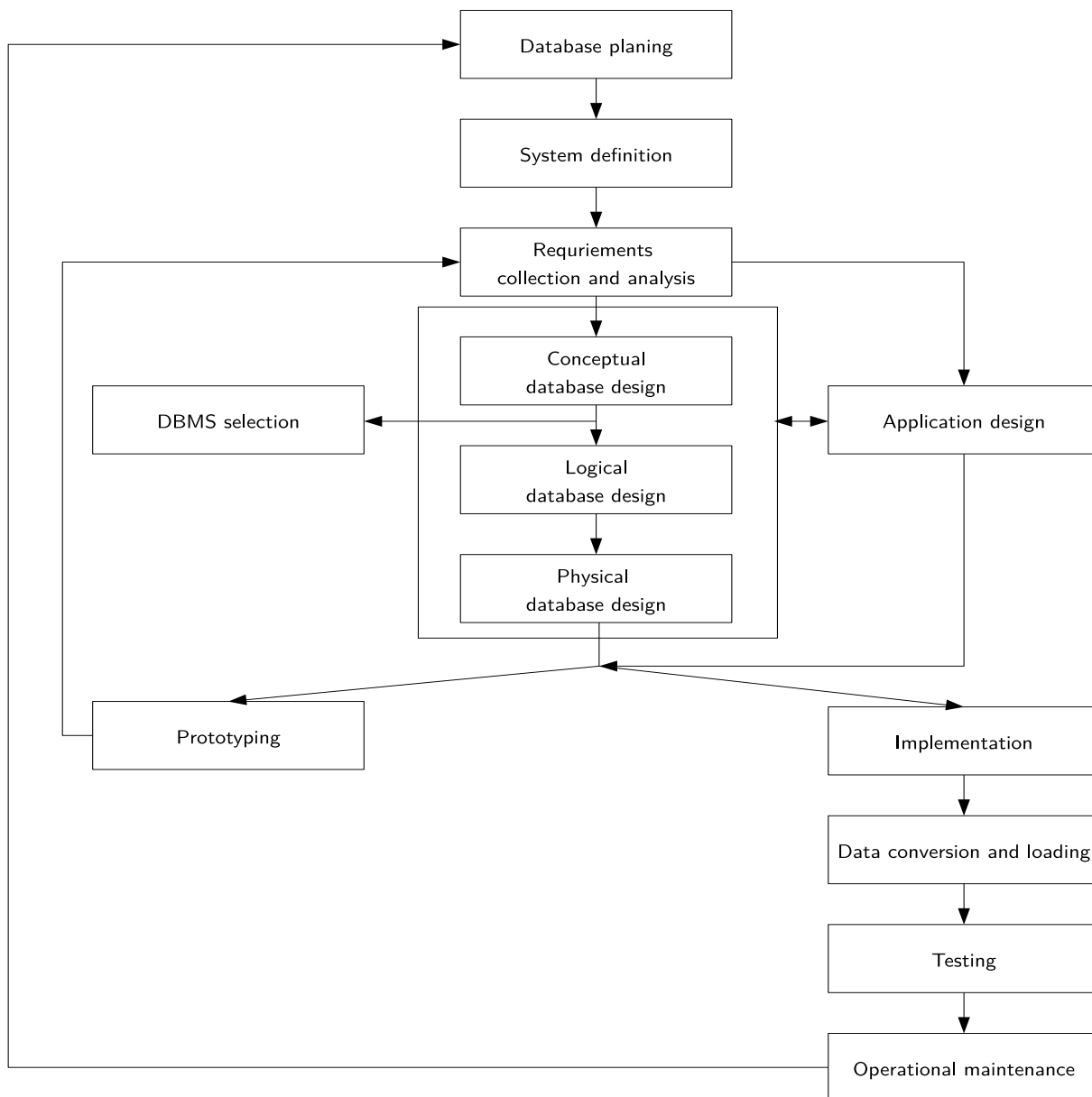


Source: Ramez Elmasri, Shamkant B. Navathe,
Wprowadzeni do systemów baz danych, Helion, 2005

Three-tier client-server architecture

Presentation layer [7] is responsible for interface construction and interaction with a user. Application layer is responsible for application logic. Database server handles queries and update requests of data layer from application layer, processes queries and sends their results.

4 Database system development life cycle



Source:

Thomas Connolly, Carolyn Begg, Systemy baz danych, Wydawnictwo RM, 2004

Summary of the main activities involved in each phase of the database system development life cycle [3]:

- Database planing - planning the most effective methods of implementing life cycle phases.
- System definition - determining the scope and boundaries of a database system.
- Requiements collection and analysis - collection and analysis of requirements for the database system.
- Database design - conceptual, logical and physical database design.
- DBMS selection - selection of a DBMS appropriate for the database application.
- Application design - designing user interfaces and application programs that will use the database.

- Prototyping (optional) - building a working model of a database system that allows designers and users to visualize and evaluate how the final system will work and look.
- Implementation - creating database definitions and application programs.
- Data conversion and loading - transferring data from the old system to the new one.
- Testing - testing database application and checking compliance with user requirements.
- Operational maintenance - the database application is fully implemented. The system is monitored and maintained. As needed, new requirements are added to database system by going through the above phases again.

Design methodology - A structured approach using procedures, techniques, tools and documentation aids for the supporting and facilitating of the design process.

Conceptual database design [3] - the process of creating a model of the data used in the enterprise, independent of any details of the physical representation.

Logical database design [3] - the process of creating a model of the data used in the enterprise, adapted to a specific data model, but independent of the specific DBMS and other details of the physical implementation.

Physical database design [3] - the process of creating a description of the implementation of the database on secondary storage; this description includes the base relation, file organization, indexes used for effective data access, and integrity constraints and security considerations.

References

- [1] Jeffrey D. Ullman, Jennifer Widom, "Podstawowy wykład z systemów baz danych", WNT, 2001
- [2] Jeffrey D. Ullman, Jennifer Widom, "Systemy baz danych, Pełny wykład", WNT, 2006
- [3] Thomas Connolly, Carolyn Begg, "Systemy baz danych", Wydawnictwo RM, 2004
- [4] Kevin Loney, Bob Bryla, Oracle Database 10g, Podręcznik administratora baz danych. Helion 2008
- [5] Scott Urman, Ron Hardman, Michael McLaughlin, "Oracle Database 10g, Programowanie w języku PL/SQL", Helion, 2008
- [6] Price J, "Oracle Database 11g i SQL Programowanie", Helion, 2009
- [7] Ramez Elmasri, Shamkant B. Navathe, "Wprowadzeni do systemów baz danych", Helion, 2005
- [8] Sharon Allen, "Modelowanie danych", Helion, 2006
- [9] Jose A. Ramalho, Oracle 8i, Mikom, 2001
- [10] Dokumentacja do bazy danych Oracle dostępna na stronie www.oracle.com
- [11] Sadalage P. J., Fowler M., "NoSQL Distilled: A Brief Guide to the Emerging World of Polyglot Persistence", Addison-Wesley Professional, 2012
- [12] Michael Stonebraker, "New Sql: An Alternative to Nosql and Old Sql For New Oltp Apps", June 16, 2011

Materials for the subject are available at: <http://weaii-moodle.tu.kielce.pl/>