Навчальна дисципліна

БАЗИ ДАНИХ

Лектор - к.т.н., доцент

Баклан Ігор Всеволодович

Site: baklaniv.at.ua

E-mail: iaa@ukr.net

2016-2017

Лекція №1.

Введення до баз даних

Topics covered include the following:

- Definition and Rationale
- Objectives of a Database System
- Advantages of a Database System
- Approaches to Database Design
- Desirable Features of a Database System
- Database Development Life Cycle
- Summary and Concluding Remarks

1.1 Definitions and Rationale

A *database system* (DBS) is a computerized record keeping system with the overall purpose of maintaining information and making it available whenever required. The database typically stores related data in a computer system.

A *database management system* (DBMS) is a set of programs that allow for the management of a database. Starting in chapter 2 and extending to subsequent chapters, we will cover several of the critical functions of a DBMS. Some of the more obvious ones are the following:

- Data definition (relation, dependencies, integrity constraints, views, etc.)
- Data manipulation (adding, updating, deleting, retrieving, reorganizing, and aggregating data)
- Data security and integrity checks
- Programming language support

Components of a DBS include:

- Hardware and operating system
- DBMS
- Database
- Related software systems and/or applications
- End users

End users communicate with the software systems/applications, which in turn, communicate (through the programming interface) with the DBMS. The DBMS communicates with the operating system (which in turn communicates with the hardware) to store data in and/or extract data from the database. Figure 1-1 illustrates.

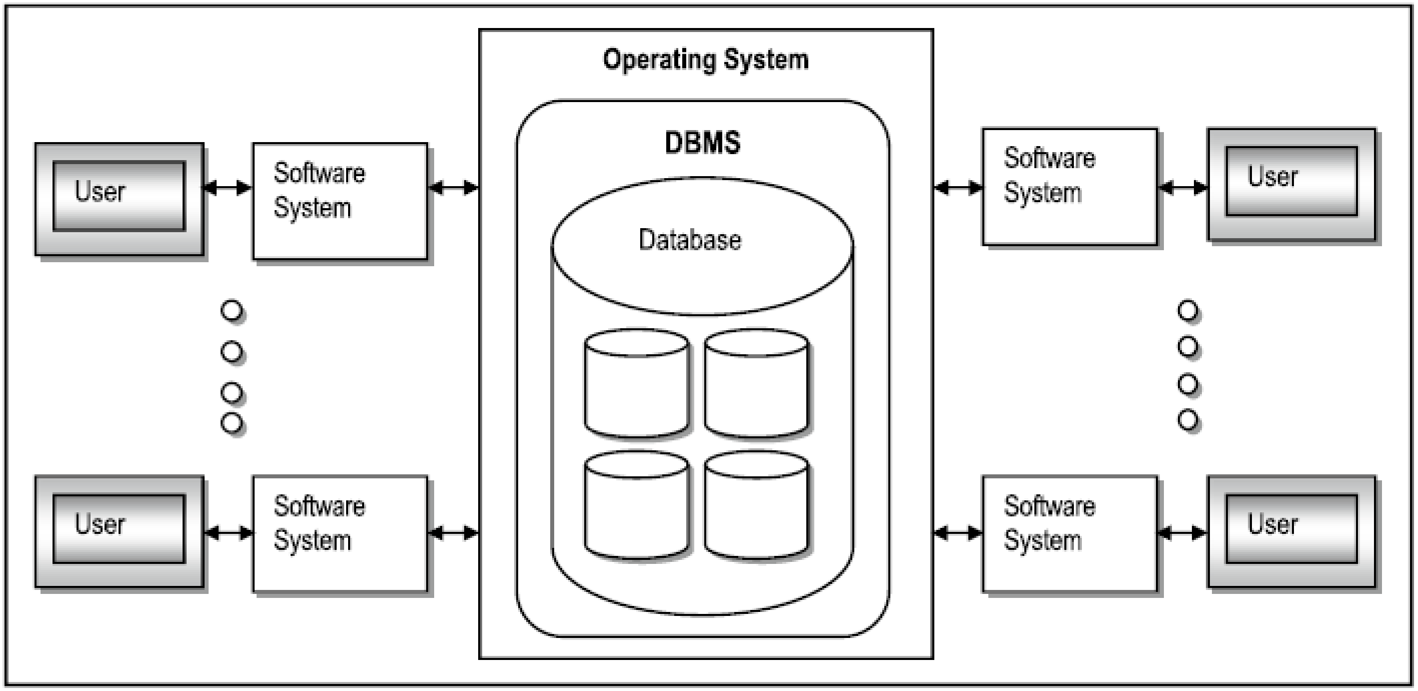
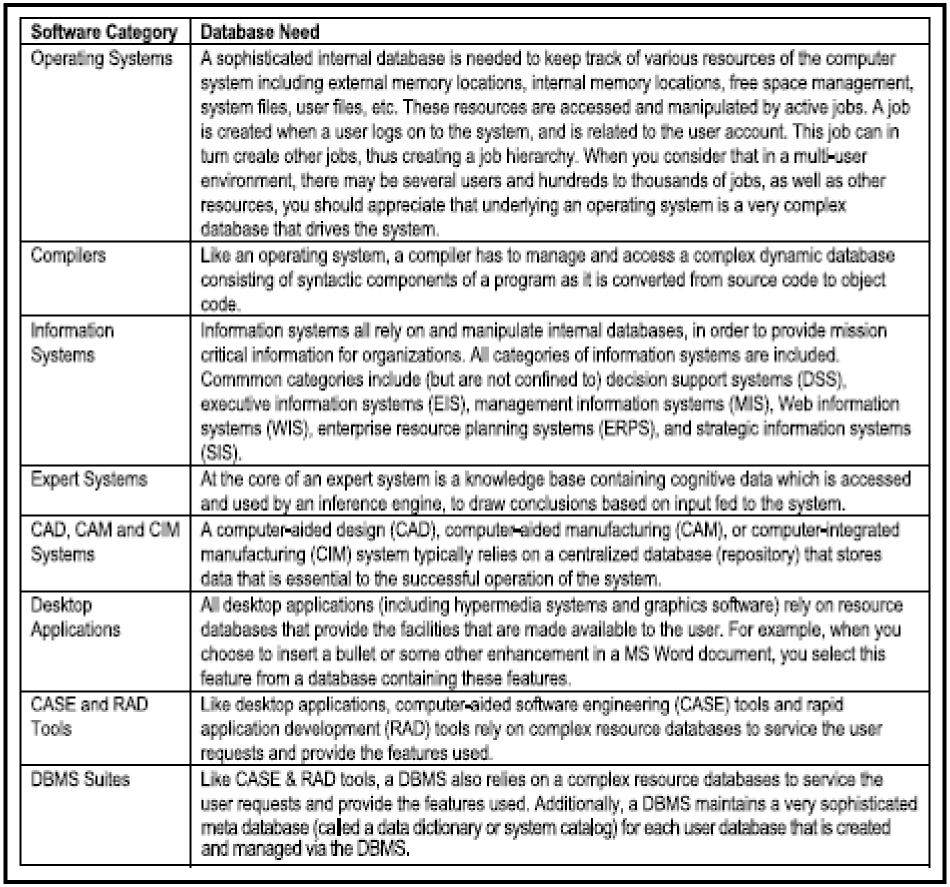


Figure 1-1. *Simplified Representation of a DBS*



Software Category	Database Need
Operating Systems	A sophisticated internal database is needed to keep track of various resources of the computer system including external memory locations, internal memory locations, free space management, system files, user files, etc. These resources are accessed and manipulated by active jobs. A job is created when a user logs on to the system, and is related to the user account. This job can in turn create other jobs, thus creating a job hierarchy. When you consider that in a multi-user environment, there may be several users and hundreds to thousands of jobs, as well as other resources, you should appreciate that underlying an operating system is a very complex database that drives the system.
Compilers	Like an operating system, a compiler has to manage and access a complex dynamic database consisting of syntactic components of a program as it is converted from source code to object code.
Information Systems	Information systems all rely on and manipulate internal databases, in order to provide mission critical information for organizations. All categories of information systems are included. Commmon categories include (but are not confined to) decision support systems (DSS), executive information systems (EIS), management information systems (MIS), Web information systems (WIS), enterprise resource planning systems (ERPS), and strategic information systems (SIS).
Expert Systems	At the core of an expert system is a knowledge base containing cognitive data which is accessed and used by an inference engine, to draw conclusions based on input fed to the system.
CAD, CAM and CIM Systems	A computer-aided design (CAD), computer-aided manufacturing (CAM), or computer-integrated manufacturing (CIM) system typically relies on a centralized database (repository) that stores data that is essential to the successful operation of the system.
Desktop Applications	All desktop applications (including hypermedia systems and graphics software) rely on resource databases that provide the facilities that are made available to the user. For example, when you choose to insert a bullet or some other enhancement in a MS Word document, you select this feature from a database containing these features.
CASE and RAD Tools	Like desktop applications, computer-aided software engineering (CASE) tools and rapid application development (RAD) tools rely on complex resource databases to service the user requests and provide the features used.
DBMS Suites	Like CASE & RAD tools, a DBMS also relies on a complex resource databases to service the user requests and provide the features used. Additionally, a DBMS maintains a very sophisticated meta database (called a data dictionary or system catalog) for each user database that is created and managed via the DBMS.

Figure 1-2. Illustrations of the Importance of Database

1.2 Objectives of a Database System

The primary objectives of a database system include the following:

- Security and protection — prevention of unauthorized users; protection from inter-process interference
- Reliability — assurance of stable, predictable performance
- Facilitation of multiple users
- Flexibility — the ability to obtain data and effect action via various methods
- Ease of data access and data change
- Accuracy and consistency
- Clarity — standardization of data to avoid ambiguity
- Ability to service unanticipated requests
- Protection of intellectual Investment
- Minimization of data proliferation — new application needs may be met with existing data rather than creating new files and programs
- Availability — data is available to users whenever it is required

Among the significant secondary objectives of a database system are the followin

- Physical data independence — storage hardware and storage techniques are insulated from application programs
- Logical data independence — data items can be added or subtracted or the overall logical structure modified without existing programs being affected
- Control of redundancy
- Integrity controls — range checks and other controls must prevent invalid data from entering the system
- Clear data definition — a data dictionary is usually kept
- Suitably friendly user interface
- Tunability — easy reorganizing the database to improve performance without changing the application programs
- Automatic reorganization of migration to improve performance

Clarification on Data Independence

Data independence is an important concept that needs further clarification: Data independence is the immunity of application programs to changes in structure and access strategy of data. It is necessary for the following reasons:

- Different applications and users will need to have different logical views (interpretation) of data.
- The tuning of the system should not affect the application programs.

Physical data independence implies that the user's view is independent of physical file organization, machine or storage medium. Logical data independence implies that each user (or application program) can have his/her (its) own logical view and does not need a global view of the database.

1.3 Advantages of a Database System

A database system brings a number of advantages to its end users as well as the company that owns it. Some of the advantages are mentioned below:

- Redundancy can be reduced.
- Inconsistencies can be avoided.
- Data can be shared.
- Standards can be enforced.
- Security restrictions can be applied.
- Integrity can be maintained.
- Conflicting requirements can be balanced.
- Improved performance due to speed of processing, reduction in paperwork, etc.
- Maintenance and retrieval of data are very easy — no complicated application program needed.
- Is not solely dependent on the high level language (HLL) programming for use.
- Logical views of data stored can be easily created.
- Record structures can change without any adverse effect on data retrieval (due to data independence).

1.4 Approaches to Database Design

Analysis of the management of data via computerized systems reveals five approaches that have been pursued over the past forty years:

- Instant small system — uses one file
- File processing systems — involve many files
- Other non-relational systems e.g. hierarchical, inverted, and network approaches
- Relational databases (the focus of this course) — pioneered by prominent individuals such as Edgar Codd, Ronald Fagin, Christopher Date, among others
- Object databases — a contemporary approach (also discussed later in the course)

Since the 1970s, relational databases have dominated the field of database systems. With the advancement of object databases (notably since the 1990s), relational databases continue to maintain dominance. Later in the course, you will understand why this dominance is likely to continue. For now, we assert that relational databases will be around for a long time in the foreseeable future, and they will be complemented (rather than challenged) by object databases.

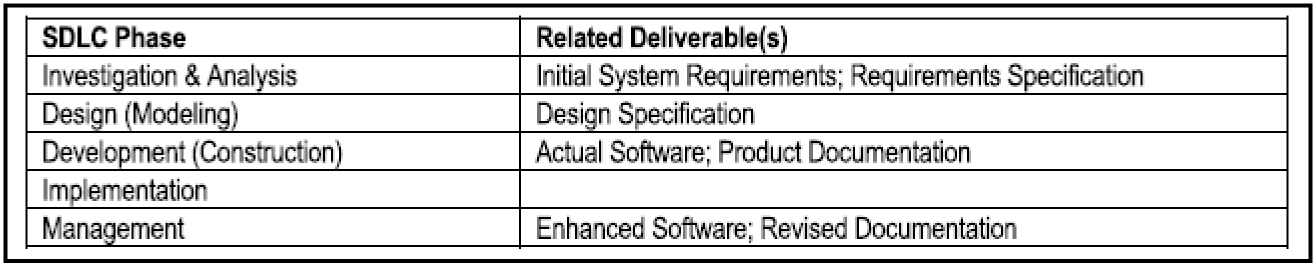
1.5 Desirable Features of a DBS

Contemporary database systems must live up to de facto standards set by the software engineering industry. Roughly speaking, a well-designed database system must exhibit the following features (more specific standards will be discussed later in the course):

- Provide most (at least 70%) of the advantages mentioned earlier
- Meet most (at least 70%) of the objectives mentioned earlier
- Provide for easy communication with other systems
- Be platform independent
- Have a friendly user interface
- Be thoroughly documented

1.6 Database Development Life Cycle

You are no doubt familiar with the software development life cycle (SDLC). For the purpose of review, it is presented in Figure 1-3.

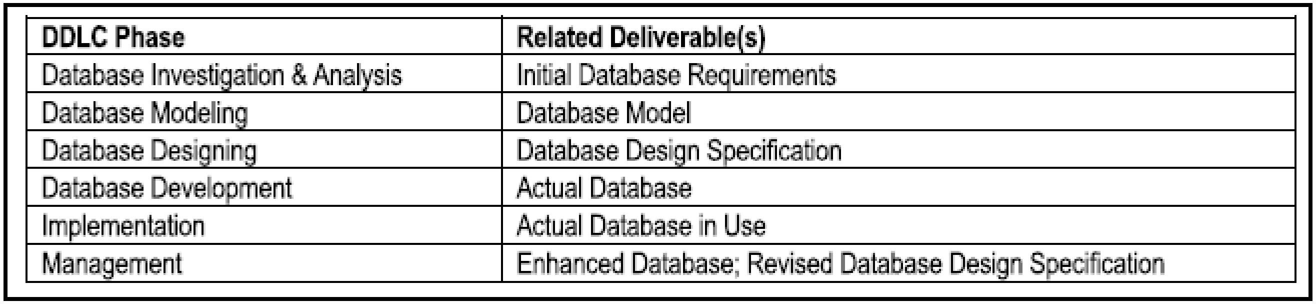


SDLC Phase	Related Deliverable(s)
Investigation & Analysis	Initial System Requirements; Requirements Specification
Design (Modeling)	Design Specification
Development (Construction)	Actual Software; Product Documentation
Implementation	
Management	Enhanced Software; Revised Documentation

Figure 1-3. *Software Development Life Cycle*

As mentioned earlier (section 1.1), databases do not exist in a vacuum, but are usually part of a software system. A *database development life cycle* (DDLC) may therefore be perceived from two perspectives:

- It may be viewed as being identical and concurrent with the SDLC. At each phase in the SDLC, consideration is given to the database as an integral part of the software product.
- If we consider that in many cases, the database has to be constructed and implemented, and managed as a separate resource that various software systems can tap into, then we may construct a similar but different life cycle for the database as illustrated in Figure 1-4.



DDLC Phase	Related Deliverable(s)
Database Investigation & Analysis	Initial Database Requirements
Database Modeling	Database Model
Database Designing	Database Design Specification
Database Development	Actual Database
Implementation	Actual Database in Use
Management	Enhanced Database; Revised Database Design Specification

Figure 1-4. *Database Development Life Cycle*

Note:

1. Applying basic investigation skills that you would have acquired in your software engineering course covers the database investigation and analysis phase. This course assumes that you have acquired those skills. The course therefore concentrates on the other phases.

2. With experience, the database modeling and database designing phases can be merged into one phase. This will be further clarified in chapters 3 through 5.

3. Once the database is in implementation phase, management of it becomes an ongoing experience, until the database becomes irrelevant to the organization.

1.7 Summary and Concluding Remarks

- A database system is a computerized record keeping system with the overall purpose of maintaining information and making it available on demand.
- The DBMS is the software that facilitates creation and administration of the database.
- The DBS is made up of the hardware, the operating system, the DBMS, the actual database, the application programs and the end users.
- There are several primary and secondary objectives of a DBS, which are of importance to the CS professional.
- A DBS brings a number of significant advantages to the business environment.
- There are five approaches to constructing a DBS. Three of them are traditional and are no longer used. The two contemporary approaches are the relational approach and the object-oriented approach. For various reasons, the relational approach is fundamental to a course in database systems.

- In striving to acquire a DBS, it is advisable to aspire for a minimum or 70% of the objectives and advantages. Additionally, one should strive for platform independence, user-friendliness, and thorough documentation.
- The database development life cycle outlines the main activities in the useful life of a DBS.