Business Training Services

Structure Query Language (SQL)

Trainer: Khattar Daou, M.S., Ph.D., MCT, MOS

Microsoft Certified Trainer (MCT),

Microsoft Office Specialist (MOS)

KDaou@DatscoTraining.com

Datsco

Software Development, Training, and Consulting

SQL – Day1

Introduction to Relational Database

Class contents

- Data Management
- Three Perspectives of Metadata in a Database
- Physical and Logical Data Independence
- What is a Database System?
- What is a Database?
- Components of a Database System
- Types of Database Systems
- The Relational Database Model
- Steps in Database Design
- Relationship Types

(Continued on next slide.)

Terminology

- **Data** – Raw facts from which the required information is derived. Data have little meaning unless they are grouped in a logical manner
- **Information** – Data that has been organized into a specific context such that it has value to its recipient.
- **Metadata** – A lens through which data takes on specific meaning and yields information.
- **Record** – A logically connected set of one or more fields that describes a person, place, event, or thing.
- **File** – A *collection of related records* that contain information of interest to the end user

Data Management

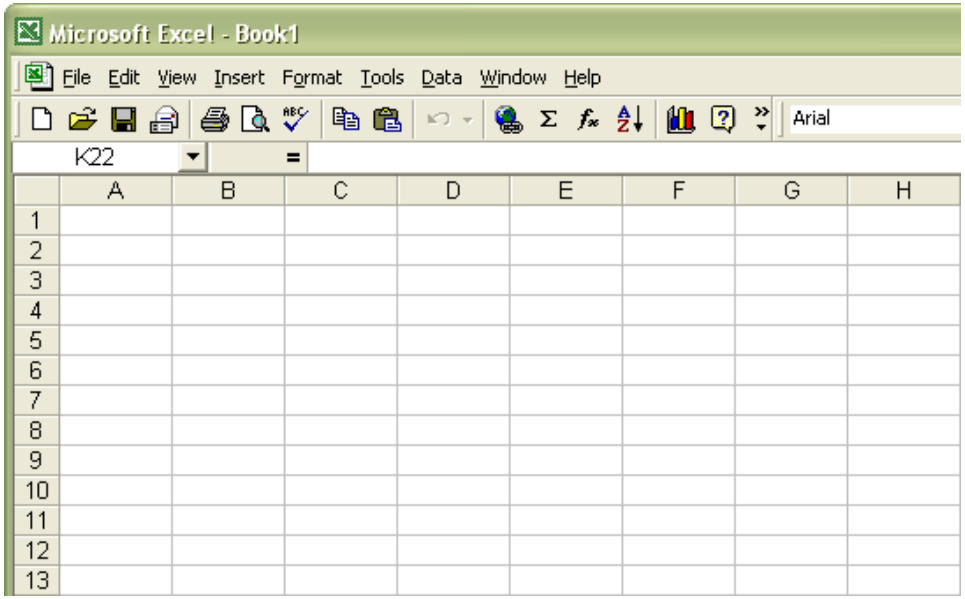
Four actions are involved in data management:

1. **C**reation of data
2. **R**etrieval of data
3. **U**pdate or modification of data
4. **D**eletion of data

For that, data must be accessed and, for the ease of access, data must be organized.

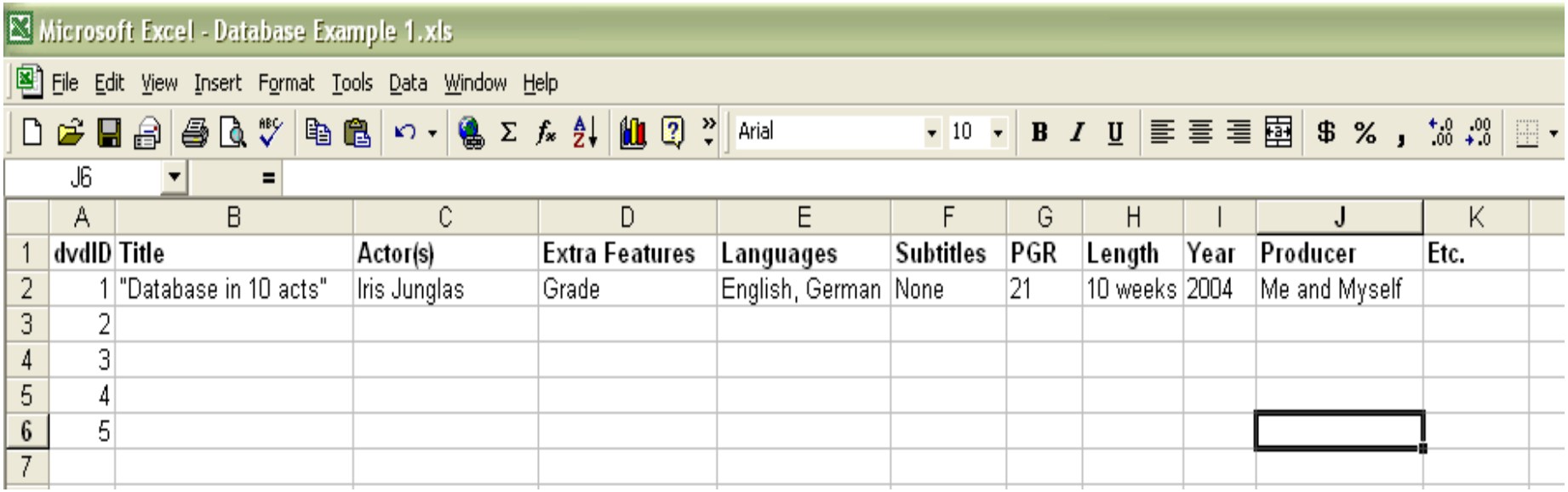
Exercise

Assume you want to organize your DVD collection. The only tool available is an Excel sheet. What would your columns and rows in Excel look like?



Exercise (continued)

Maybe like this?



	A	B	C	D	E	F	G	H	I	J	K
1	dvdID	Title	Actor(s)	Extra Features	Languages	Subtitles	PGR	Length	Year	Producer	Etc.
2	1	"Database in 10 acts"	Iris Junglas	Grade	English, German	None	21	10 weeks	2004	Me and Myself	
3	2										
4	3										
5	4										
6	5										
7											

Data Management

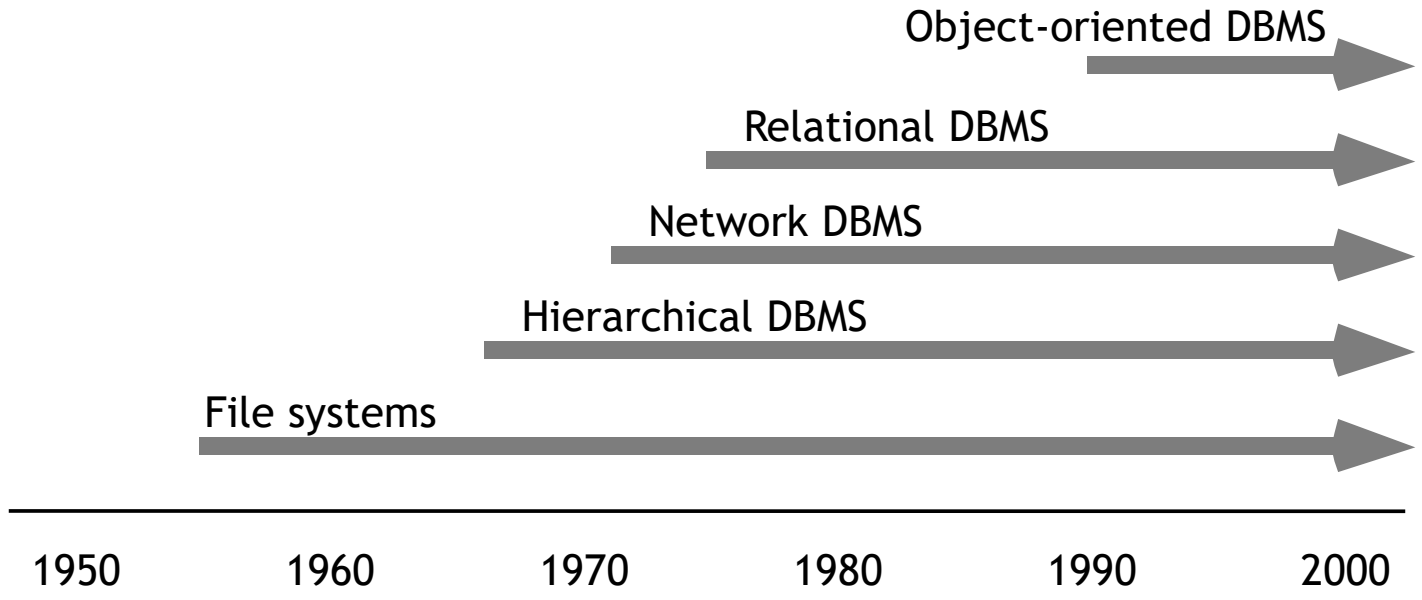
Only two approaches for accessing data exist:

- Sequential access –*requires that one pass through the first n-1 records in a data set to get to the n^{th} record*
- Direct access – *allows for the retrieval of the n^{th} record in a data set without having to look at the previous n-1 records*

Important:

A DBMS facilitates access of data without burdening the user with details of how the data is physically organized.

History of Data Management



Limitations of File-Processing Systems

The predecessor of a database system where records were stored in separate non-integrated files.

- Lack of Data Integrity
  Data integrity (data values are correct, consistent, complete, and current) is often violated in isolated environments.
- Lack of Standards
  Organizations find it hard to enforce standards for naming data items as well as for accessing, updating, and protecting data.
- Lack of Flexibility/Maintainability
  File-processing systems are not amenable to structural changes in data and are therefore dependent upon a programmer who can either write or modify program code.

Limitations of File-Processing Systems (continued)

The limitations to file-processing systems are due to:

- Lack of Data Integration
  Data are separated and isolated in a file-processing environment.
- Lack of Program-Data Independence
  The structure of each file is embedded in the application programs.

Limitations of File-Processing Systems (continued)

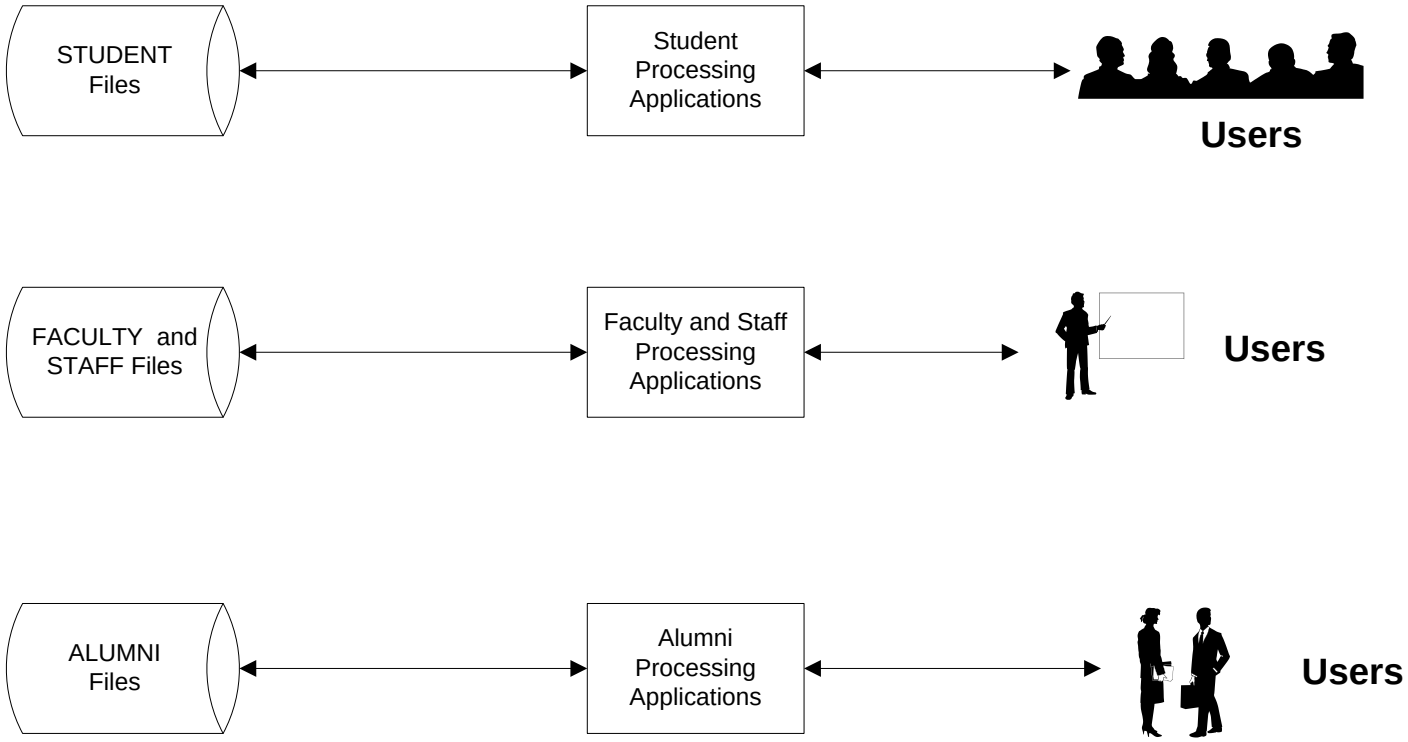


Figure 1.1 An example of a file processing environment

So, What Is Desirable?

- Integrated data – ensures that data is correct, consistent, complete, and current
  - Not data in isolation to be integrated by the application program/programmer.
- Data Independence - the ability to modify a schema definition in one level without affecting a schema definition in a higher level.
  - Application program(s) immune to changes in *storage structure* and *access strategy*
  - Independent user views of data

History of Data Management

In the 1970s, the Standards Planning and Requirements Committee (SPARC) of the American National Standards Institute (ANSI) proposed what came to be known as the ANSI/SPARC three-schema architecture: conceptual, internal and external schema.

Three Perspectives of Metadata in a Database

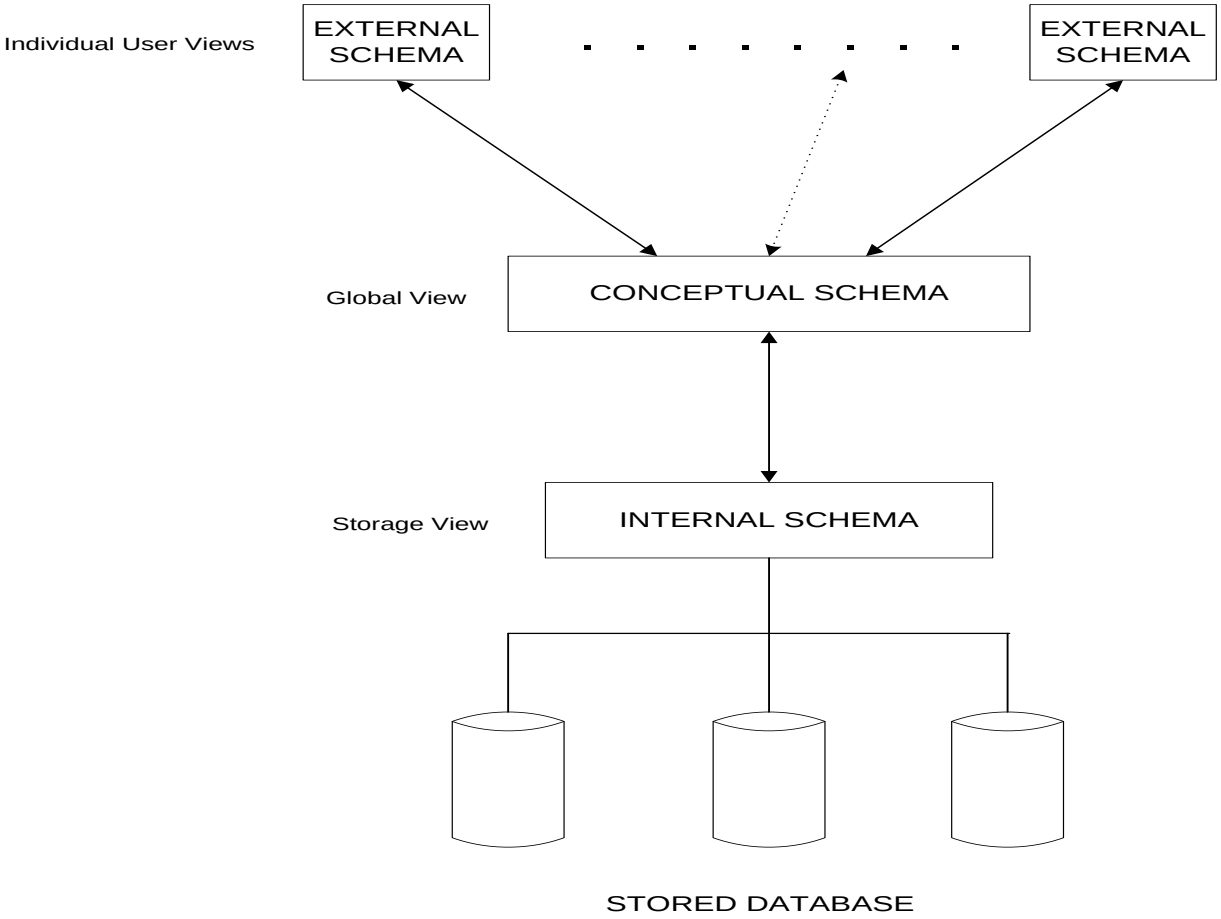


Figure 1.2 The ANSI/SPARC three-schema Architecture

Conceptual Schema

- Core of the architecture
- Represents the global view of the structure of the entire database for a community of users
- Captures data specification (metadata)
- Describes all data items and relationships between data together with integrity constraints
- Separates data from the program (or views from the physical storage structure)
- Technology independent

Internal Schema

- Describes the physical structure of the stored data (e.g., how the data is actually laid out on storage devices)
- Describes the mechanism used to implement access strategies (e.g., indexes, hashed addresses, etc.)
- Technology dependent
- Concerned with the efficiency of data storage and access mechanisms

External Schema

- Represents different user views, each describing portions of the database
- Technology independent
- Views are generated exclusively by logical references

Physical and Logical Data Independence

- Physical Data Independence – The ability to modify the internal schema without causing the application program in the external schema to be rewritten.

  Definition: External views unaffected by changes to the internal structure

  How? Introduction of conceptual schema between the external views and the internal (physical) schema

Physical and Logical Data Independence (continued)

- Logical Data Independence – The immunity of a user view from changes in the other user views.

  Definition: External views unaffected by design changes (growth or restructuring) in conceptual schema

  How? External views generated exclusively through logical reference to elements in the conceptual schema

  Consequence: External views unaffected by changes to other external views

What is a Database System?

- A collection of general-purpose software that facilitates the processes of defining, constructing, and manipulating a database for various applications.
- A *self-describing* collection of *integrated* records

  Self-describing – The structure of the database (metadata) is recorded within the database system – not in the application programs.

  Integrated – The responsibility for 'integrating' data items as needed is assumed by the DBMS instead of the programmer.

What is a Database?

Database – a collection of data that's related to a particular topic or purpose.

- Data consists of recorded facts that have implicit meaning.
- Viewed through the lens of metadata, the meaning of recorded data becomes explicit.
- A database is self-describing in that the metadata is recorded within the database – not in applications programs.

What is a Database Management System

Database Management System (DBMS)

A collection of programs (software) that manage the database structure and that control shared access to the data in the database.

DBMS Functions

- Stores the definitions of data and their relationships (metadata) in a data dictionary; any changes made are automatically recorded in the data dictionary.
- Creates the complex structures required for data storage.
- Transforms entered data to conform to the data structures in item 2.
- Creates a security system and enforces security within that system.
- Creates complex structures that allow multiple-user access to the data.
- Performs backup and data recovery procedures to ensure data safety.
- Promotes and enforces integrity rules to eliminate data integrity problems.
- Provides access to the data via utility programs and from programming languages interfaces.
- Provides end-user access to data within a computer network environment

Components of a DBMS

The major components of a DBMS include one or more query languages; tools for generating reports; facilities for providing security, integrity, backup and recovery; a data manipulation language for accessing the database; and a data definition language used to define the structure of data.

Components of a Database System

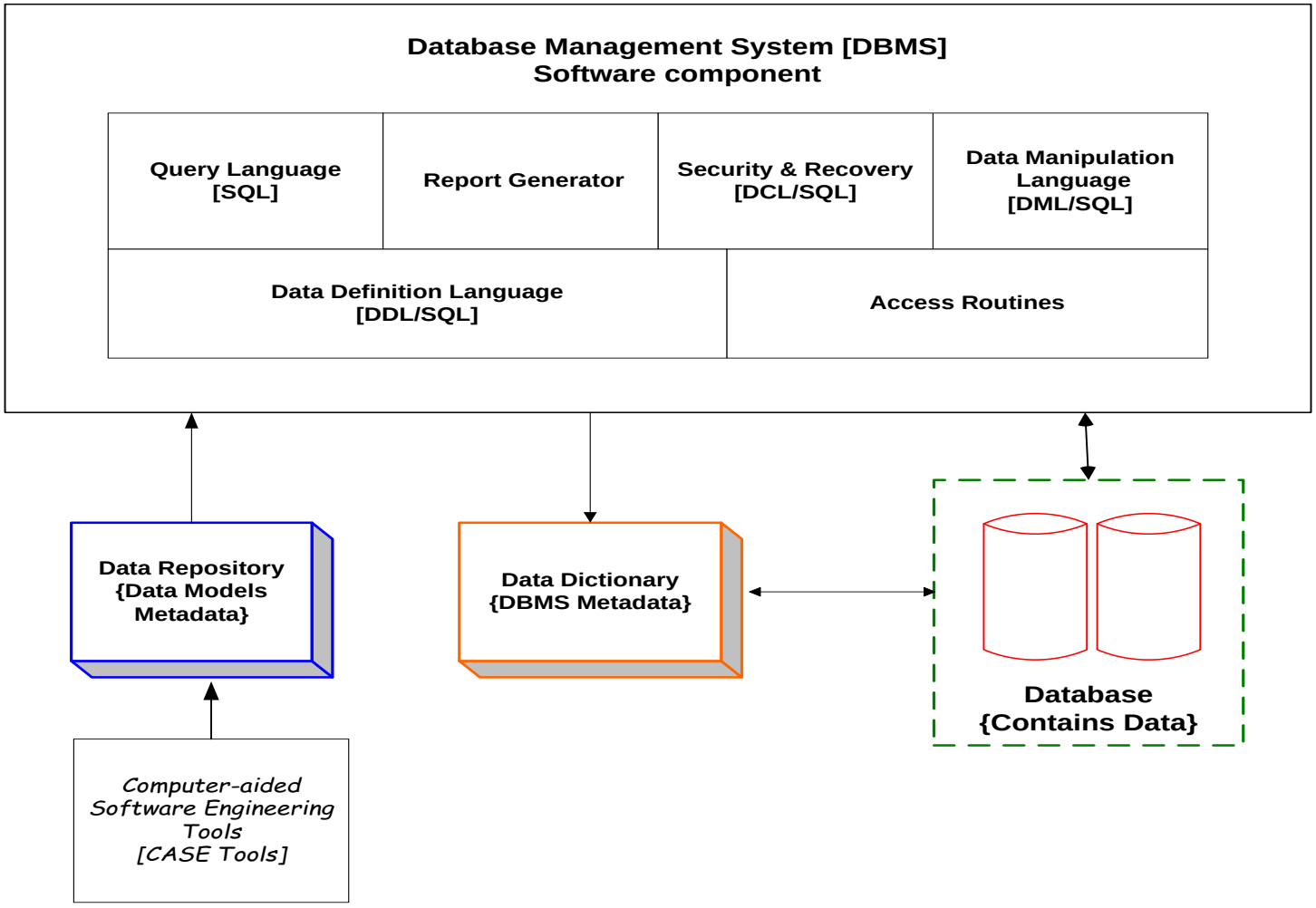


Figure 1.5 Components of a database system

An Example of a Database System

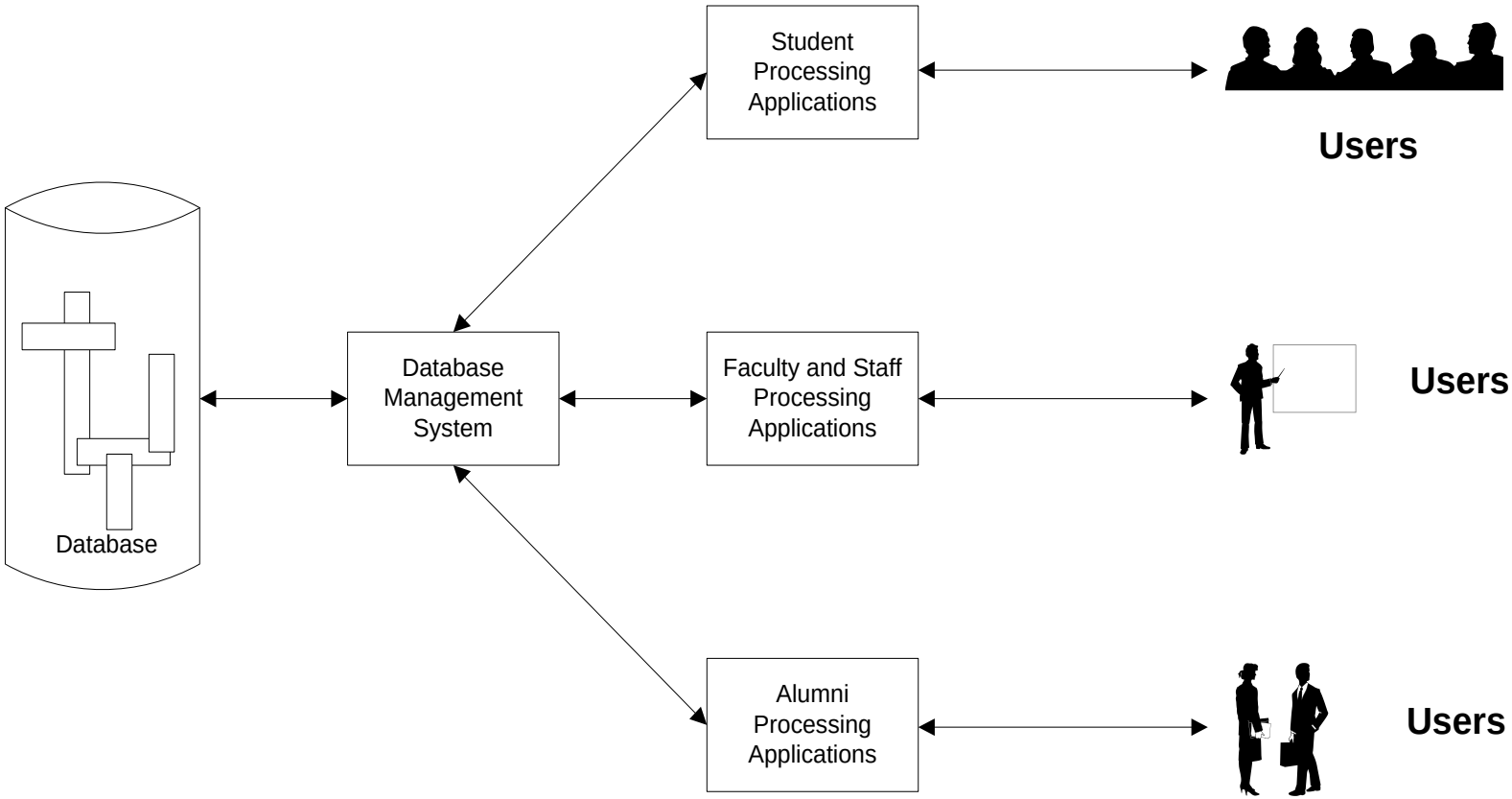


Figure 1.6 An example of a database system

Types of Database Systems

- Number of users
  - Single-user
  - Multiuser
  - Workgroup
  - Enterprise
- Database site location
  - Centralized
  - Distributed
- Database use
  - Transactional (production) database
  - Data warehouse database

Some Commercial DBMS

- IBM & DB2:
  https://www.ibm.com/analytics/us/en/technology/db2/
- Oracle:
  https://www.oracle.com/database/index.html
- Microsoft Access & SQL Server:
  http://www.microsoft.com/en-us/download/details.aspx?id=50040 and
  https://www.microsoft.com/en-us/cloud-platform/sql-server
- MySQL:
  http://www.mysql.com/

Important Terms

- Data Redundancy – exists when unnecessarily duplicated data are found in the database. For example, a customer's telephone number may be found in the customer file, in the sales agent file, and in the invoice file.
- Data Anomaly – an error or inconsistency in the database. A poorly designed database runs the risk of introducing numerous anomalies. Update—an anomaly that occurs during the updating of a record. For example, updating a description column for a single part in an inventory database requires you to make a change to thousands of rows.

Important Terms

Three types of anomalies:

- **Insertion** – an anomaly that occurs during the insertion of a record. For example, the insertion of a new row causes a calculated total field stored in another table to report the wrong total.
- **Deletion** – an anomaly that occurs during the deletion of a record. For example, the deletion of a row in the database deletes more information than you wished to delete.
- **Update** – an anomaly that occurs during the updating of a record. For example, updating a description column for a single part in an inventory database requires you to make a change to thousands of rows.

The Relational Database Model

- Relational database model has a very successful track record and it is the dominant database model in the market.
- Was conceived by E. F. Codd in 1969, then a researcher at IBM.
- The model is based on branches of mathematics called set theory and predicate logic.
- The basic idea behind the relational model is that a database consists of a series of unordered tables (or relations) that can be manipulated using non-procedural operations that return tables
- Can be applied to both databases and database management systems (DBMS) themselves.
- The relational fidelity of database programs can be compared using Codd's 12 rules. The number of rules has been expanded to 300 for determining how DBMS products conform to the relational model

Steps in Database Design

- **Step One**: Determine the purpose of your database
- **Step Two**: Determine the Entities (tables) you need
- **Step Three**: Determine the attributes (fields) in each entity
- **Step Four**: Create the Data Dictionary
- **Step Five**: Determine the Relationships by creating the Entity-Relationship Diagram (ERD)
- **Step Six**: Transfer the ERD into a relational system (tables, fields, data types, etc.)

Steps in Database Design (continued)

Step 1: Determine the purpose of your database

When designing a database, you have to make decisions regarding how best to take some system in the real world and model it in a database.

The benefits of a database that has been designed according to the relational model are numerous. Some of them are:

- Data entry, updates and deletions will be efficient.
- Data retrieval, summarization and reporting will also be efficient.
- Since the database follows a well-formulated model, it behaves predictably.
- Since much of the information is stored in the database rather than in the application, the database is somewhat self-documenting.
- Changes to the database schema are easy to make.

Steps in Database Design (continued)

- **Step 2**: Determine the Entities (tables) you need
- Entities (Tables) in the relational model are used to represent “things” in the real world. Each table should represent only one thing. These things (or entities) can be real-world objects or events
- The relational model dictates that each row in a table be unique. If you allow duplicate rows in a table, then there’s no way to uniquely address a given row via programming.
- You guarantee uniqueness for a table by designating a primary key (PK) – a column that contains unique values for a table

Steps in Database Design (continued)

- Primary keys become essential when you start to create relationships that join together multiple tables in a database. A foreign key (FK) is a column in a table used to reference a primary key in another table.
- Foreign Keys must have the same data type and field size as the primary key.

Steps in Database Design (continued)

Step 3: Determine the attributes (fields) in each entity

- Attribute - A fundamental characteristic of an entity type (i.e., the conceptual representation of a property).
- *An attribute that has a discrete factual value and cannot be meaningfully subdivided is called an atomic or simple attribute*
- *A composite attribute can be meaningfully subdivided into smaller subparts with independent meaning*

Steps in Database Design (continued)

- **Step Four**: Create the Data Dictionary
- A DBMS component that stores metadata – data about data. Thus, the Data Dictionary contains the data definition as well as the characteristics and relationships.

Table Name	Attribute Name	Contents	Type	Length	Format	Range	Req'd	Key	Referenced Table
PRODUCT	PROD_CODE	Product code	VCHAR	10	X(10)		Y	PK	
	PROD_DESCRIPTION	Product description	VCHAR	35	X(35)		Y		
	PROD_STOCK_DATE	Stock date	DATE	8	mm/dd/yyyy		Y		
	PROD_ONHAND	Units available	SMALLINT	4	9999	0-9999	Y		
	P_PRICE	Product price	DECIMAL	7	9999.99	0-9999.99	Y		
	VEND_CODE	Vendor code	INTEGER	5	99999	1-99999	Y	FK	VENDOR
VENDOR	VEND_CODE	Vendor code	INTEGER	5	99999	1-99999	Y	PK	
	VEND_NAME	Vendor name	VCHAR	35	X(35)		Y		
	VEND_CONTACT	Contact person	VCHAR	15	X(15)		Y		
	VEND_AREACODE	Area code	NCHAR	3	X(3)		Y		
	VEND_PHONE	Phone	NCHAR	8	X(8)		Y		

Steps in Database Design (continued)

Step Five: Determine the Relationships by creating the Entity-Relationship Diagram (ERD)

- You define foreign keys in a database to model relationships in the real world. Relationships between real-world entities can be quite complex, involving numerous entities each having multiple relationships with each other.
- We consider only relationships between pairs of entities (tables).
- These tables can be related in one of three different ways: one-to-one, one-to-many or many-to-many

Steps in Database Design (continued)

One-to-One Relationships

- Two tables are related in a one-to-one (1–1) relationship if, for every row in the first table, there is at most one row in the second table. True one-to-one relationships seldom occur in the real world.
- one-to-one relationships may be necessary in a database when you have to split a table into two or more tables because of security or performance concerns

Steps in Database Design (continued)

One-to-Many Relationships

- Two tables are related in a one-to-many (1–M) relationship if for every row in the first table, there can be zero, one, or many rows in the second table, but for every row in the second table there is exactly one row in the first table.

Steps in Database Design (continued)

Many-to-Many Relationships

- Two tables are related in a many-to-many (M–M) relationship when for every row in the first table, there can be many rows in the second table, and for every row in the second table, there can be many rows in the first table.
- Many-to-many relationships can’t be directly modeled in relational database programs.
- These types of relationships must be broken into multiple one-to-many relationships.

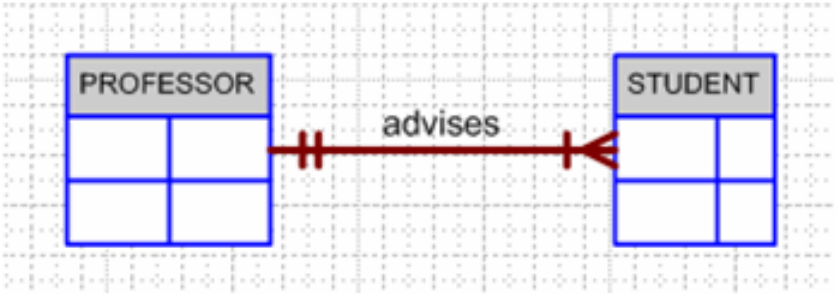
Steps in Database Design (continued)

- **Step Five**: Determine the Relationships by creating the Entity-Relationship Diagram (ERD)

Chen ERD (generated with PowerPoint)

Crow's Foot ERD (generated with PowerPoint)

Chen ERD (generated with Visio Professional)

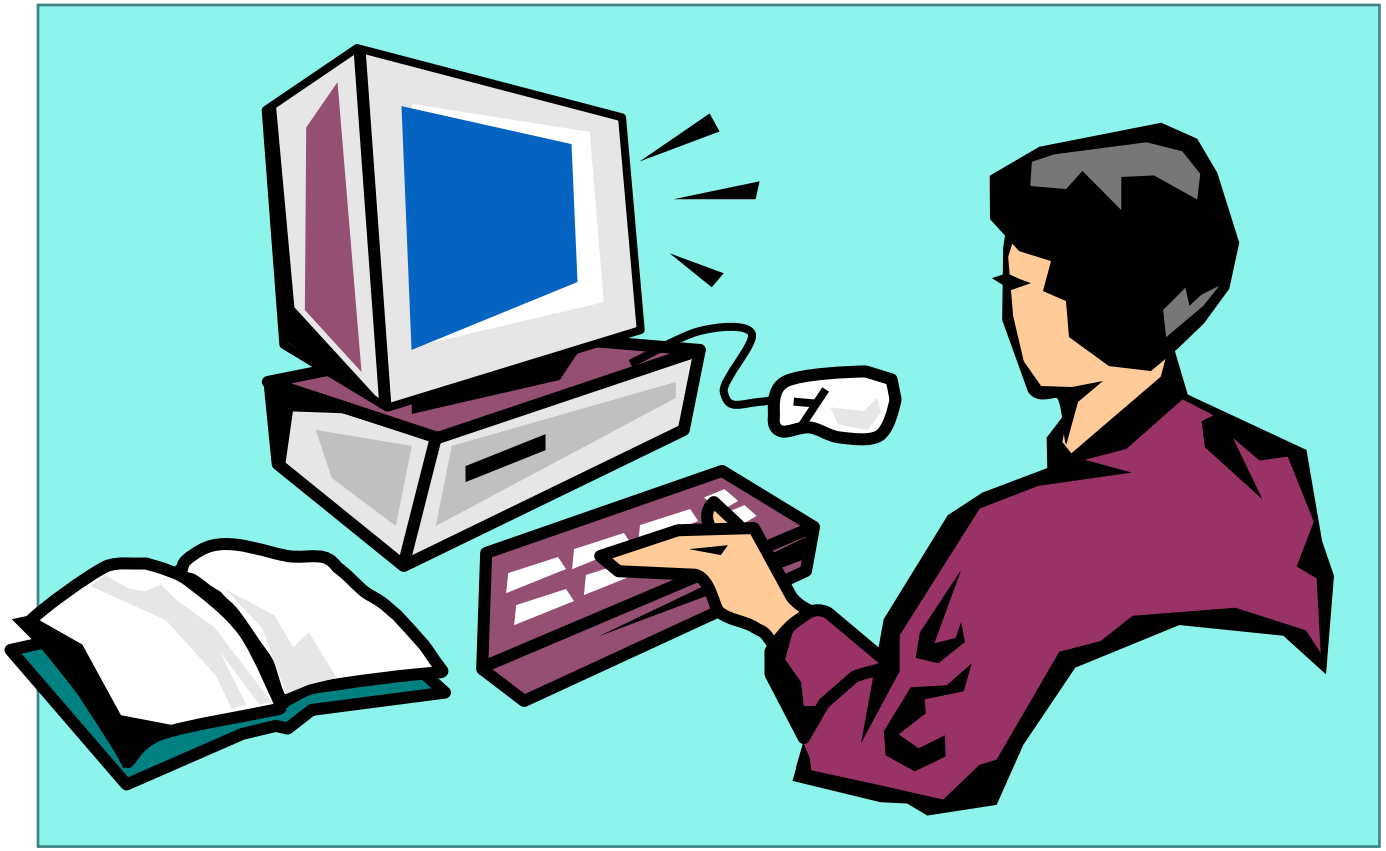


Steps in Database Design (continued)

- **Step Six**: Transfer the ERD into a relational system (tables, fields, data types, etc.)
- Entity → Table, Attribute → Field (column) name
- tblEmployees (EmpID, firstName, lastName,…)

- **Step Seven:** Choose a DBMS – Oracle, SQL Server, Db2, Sybase, …

Lab 1: Introduction to Relational Database



Review

- Determine the Entities (tables) you need
- Determine the attributes (fields) in each entity
- Create the Data Dictionary
- Determine the Relationships by creating the Entity-Relationship Diagram (ERD)
- Transfer the ERD into a relational system (tables, fields, data types, etc.)

References

- Implementing a Microsoft SQL Server 2005 Database Course 2779
- Programming a Microsoft SQL Server 2000 Database. Course Number: 2073A
- Introduction to Database Management. Mark Gillenson. ISBN: 978-0-470-10186-5
- Database Systems: Design, Implementation, & Management. Peter Rob. ISBN: 978-0-619-16033-3
- Relational Database Management System for Windows. Document No. DB64084-0295. Microsoft Press