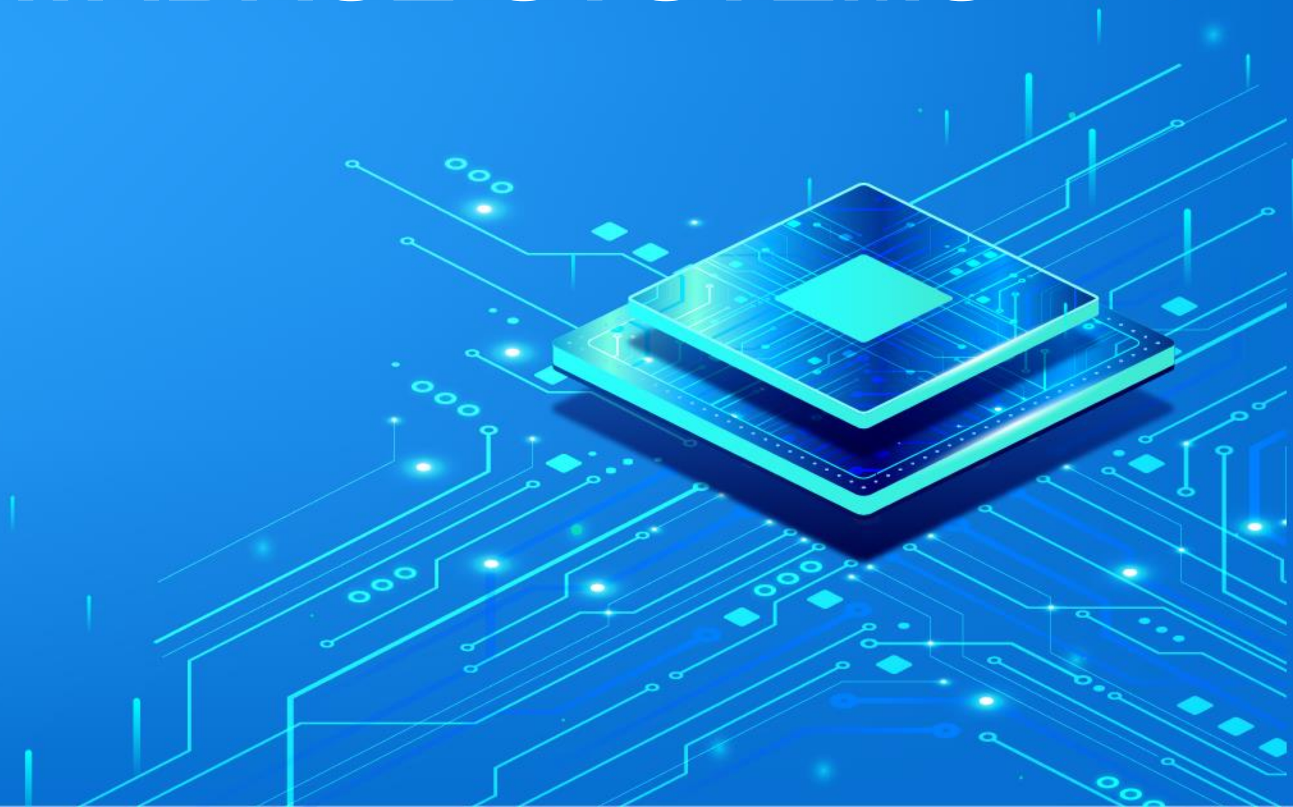
CMC UNIVERSITY

CHAPTER 1. INTRODUCTION TO DATABASE SYSTEMS



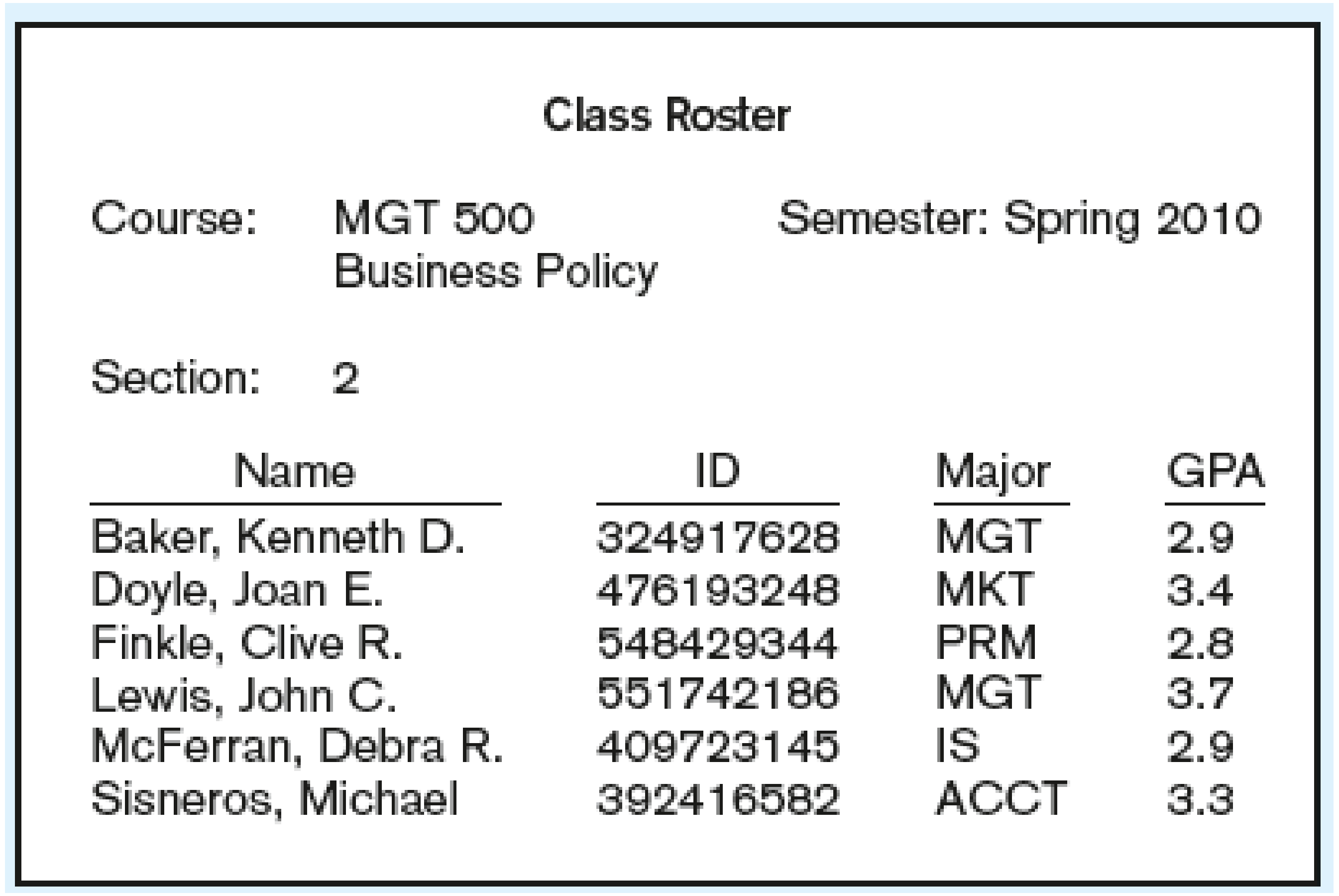
Outline

1.1.1 What is a database?

- ✓ Data: facts can be recorded and have implicit meaning
  - ✓ Structured: numbers, text, dates
  - ✓ Unstructured: images, videos, documents
- ✓ Information: data is processed to increase understanding of the data user
- ✓ Metadata: data that describes the attributes and context of user data

CMC UNIVERSITY

1.1.1 What is a database?



Class Roster

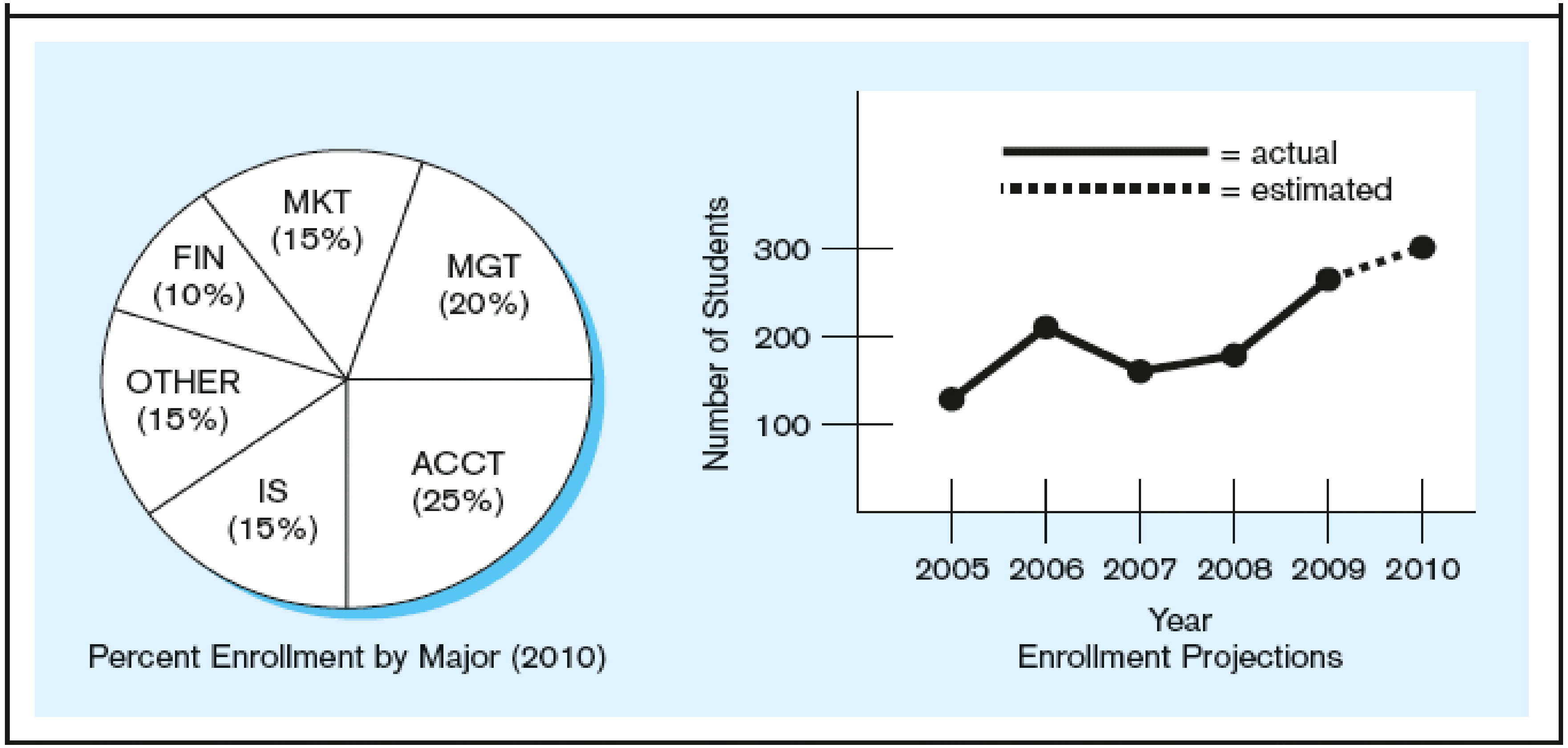
Course:	MGT 500 Business Policy	Semester: Spring 2010
Section:	2	

Name	ID	Major	GPA
Baker, Kenneth D.	324917628	MGT	2.9
Doyle, Joan E.	476193248	MKT	3.4
Finkle, Clive R.	548429344	PRM	2.8
Lewis, John C.	551742186	MGT	3.7
McFerran, Debra R.	409723145	IS	2.9
Sisneros, Michael	392416582	ACCT	3.3

Data in context

Users understand data through context

1.1.1 What is a database?



Summary data

Data is transformed into useful information that helps managers make decisions and interpretations

1.1.1 What is a database?

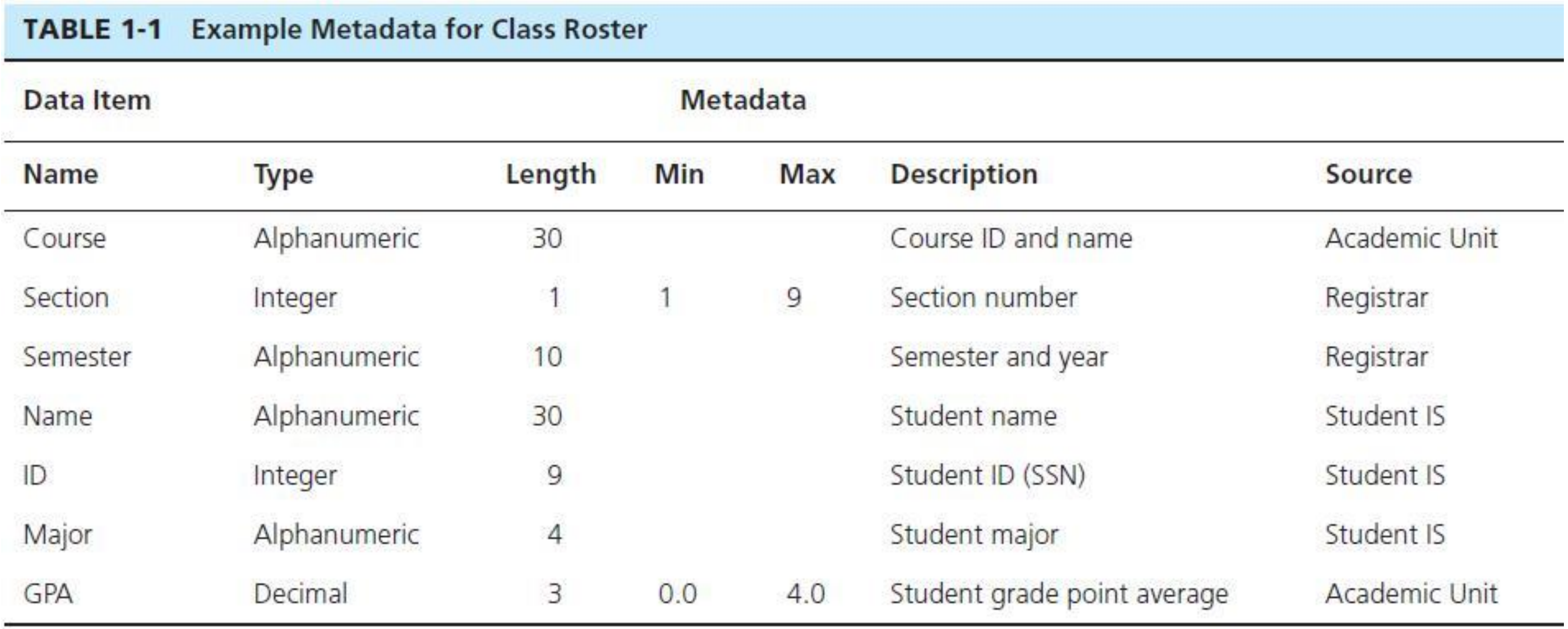


TABLE 1-1 Example Metadata for Class Roster

Data Item	**Metadata**					
Name	**Type**	**Length**	**Min**	**Max**	**Description**	**Source**
Course	Alphanumeric	30			Course ID and name	Academic Unit
Section	Integer	1	1	9	Section number	Registrar
Semester	Alphanumeric	10			Semester and year	Registrar
Name	Alphanumeric	30			Student name	Student IS
ID	Integer	9			Student ID (SSN)	Student IS
Major	Alphanumeric	4			Student major	Student IS
GPA	Decimal	3	0.0	4.0	Student grade point average	Academic Unit

Describes the attributes or characteristics of the data, including data type, field size, allowed values, and data context

1.1.1 What is a database?

A **database** is

- a ***collection*** of related data that reflects the activity of an organization
- ***stored*** on memory devices
- there are ***many users*** for different purposes

Example:

- A company database, storing data of hundreds of employees
- A university database, storing data of thousands of students
- The computerized catalog of a large library may contain half a million entries organized under different categories—by primary author’s last name, by subject, by book title—with each category organized alphabetically

1.1.1 What is a database?

- The database of a social media company such as ***Facebook***, which has billions of users

- A large commercial database is ***Amazon.com***, contains data for over 60 million active users, and millions of books, CDs, videos, DVDs, games, electronics, apparel, and other items

The database occupies over 42 terabytes and is stored on hundreds of computers (called servers). Millions of visitors access Amazon.com each day and use the database to make purchases.

1.1.1 What is a database?

A database has the following implicit properties:

- A database represents some aspect of the real world, sometimes called the ***miniworld*** or the ***universe of discourse*** (UoD). Changes to the miniworld are reflected in the database.
- A database is a logically coherent collection of data with some inherent meaning. A random assortment of data cannot correctly be referred to as a database.
- A database is designed, built, and populated with data for a specific purpose. It has an intended group of users and some preconceived applications in which these users are interested.

1.1.2 What is a database management system?

A **database management system** (DBMS) is a computerized system that enables users to *create* and *maintain* a database.

The DBMS is a general-purpose software system that facilitates the processes of *defining*, *constructing*, *manipulating*, and *sharing* databases among various users and applications:

- *Defining* a database involves specifying the data types, structures, and constraints of the data to be stored in the database. The database definition or descriptive information is also stored by the DBMS in the form of a database catalog or dictionary; it is called *meta-data*
- *Constructing* the database is the process of storing the data on some storage medium that is controlled by the DBMS

1.1.2 What is a database management system?

- *Manipulating* a database includes functions such as querying the database to retrieve specific data, updating the database to reflect changes in the miniworld, and generating reports from the data.
- *Sharing* a database allows multiple users and programs to access the database simultaneously.

Important functions provided by the DBMS

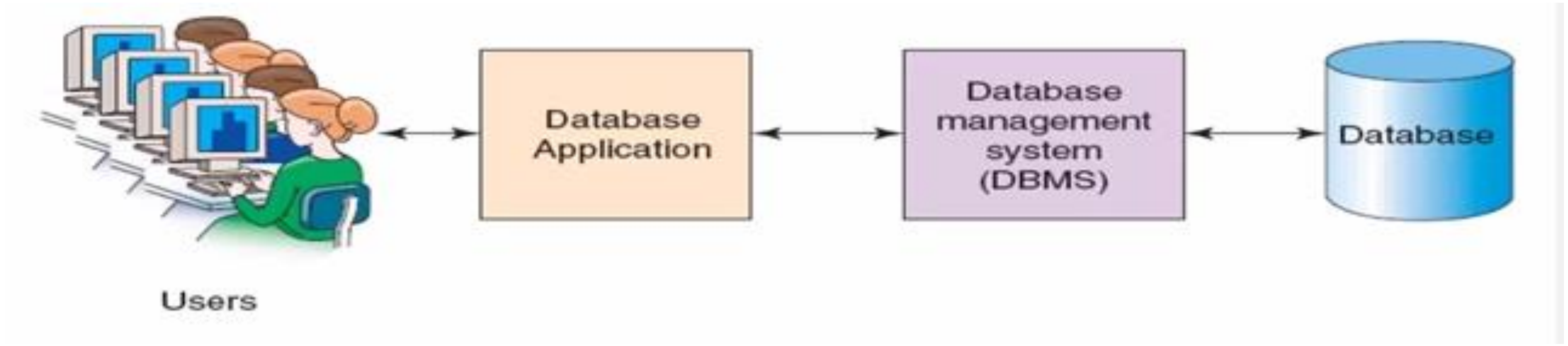
- An application program accesses the database by sending queries or requests for data to the DBMS. A *query* typically causes some data to be retrieved; a *transaction* may cause some data to be read and some data to be written into the database.
- *Protection* includes system protection against hardware or software malfunction (or crashes) and security protection against unauthorized or malicious access
- The DBMS must be able to *maintain* the database system by allowing the system to evolve as requirements change over time

1.1.2 What is a database management system?

- DBMS used on personal computers: MS. Access
- DBMS used for organizations: MS. SQL server, Oracle, MySQL, DB2
- NoSQL: MongoDB, Casscandra, Couchbase, Neo4j
- DBMS used for mobile devices: SQLite

CMC UNIVERSITY

1.1.3 Components of a database system



A database system can also be more concisely said to include a DBMS and a DB

Example of a database

UNIVERSITY database

STUDENT

Name	Student_number	Class	Major
Smith	17	1	CS
Brown	8	2	CS

COURSE

Course_name	Course_number	Credit_hours	Department
Intro to Computer Science	CS1310	4	CS
Data Structures	CS3320	4	CS
Discrete Mathematics	MATH2410	3	MATH
Database	CS3380	3	CS

Example of a database

SECTION

Section_identifier	Course_number	Semester	Year	Instructor
85	MATH2410	Fall	07	King
92	CS1310	Fall	07	Anderson
102	CS3320	Spring	08	Knuth
112	MATH2410	Fall	08	Chang
119	CS1310	Fall	08	Anderson
135	CS3380	Fall	08	Stone

GRADE_REPORT

Student_number	Section_identifier	Grade
17	112	B
17	119	C
8	85	A
8	92	A
8	102	B
8	135	A

PREREQUISITE

Course_number	Prerequisite_number
CS3380	CS3320
CS3380	MATH2410
CS3320	CS1310

1.2 Purpose of Database Systems

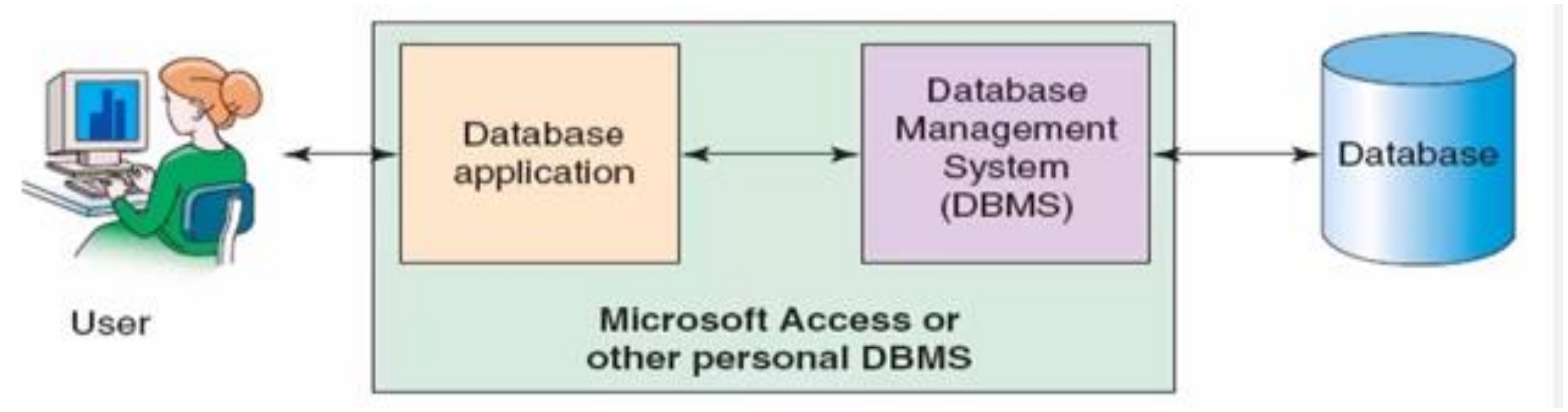
- In the early days, database applications were built directly on top of file systems
- Drawbacks of using file systems to store data:
  - Data redundancy and inconsistency
    - Multiple file formats, duplication of information in different files
  - Difficulty in accessing data
    - Need to write a new program to carry out each new task
  - Data isolation — multiple files and formats
  - Integrity problems
    - Integrity constraints (e.g. account balance > 0) become "buried" in program code rather than being stated explicitly
    - Hard to add new constraints or change existing ones

1.2 Purpose of Database Systems

- Drawbacks of using file systems (cont.)
  - Atomicity of updates
    - Failures may leave database in an inconsistent state with partial updates carried out
    - Example: Transfer of funds from one account to another should either complete or not happen at all
  - Concurrent access by multiple users
    - Concurrent accessed needed for performance
    - Uncontrolled concurrent accesses can lead to inconsistencies
      - Example: Two people reading a balance and updating it at the same time
  - Security problems
    - Hard to provide user access to some, but not all, data
- Database systems offer solutions to all the above problems

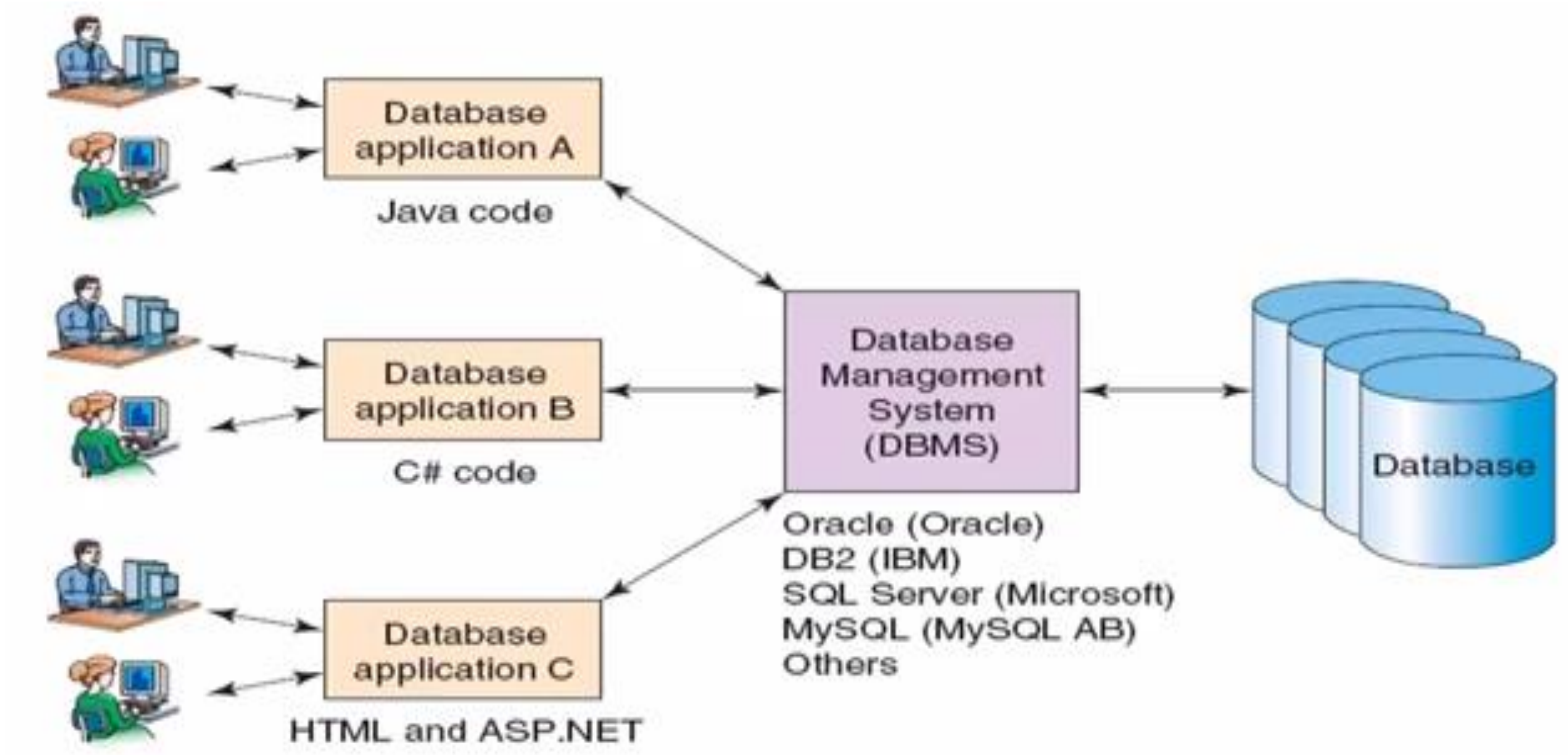
1.2 Purpose of Database Systems

Personal Database Systems


User
Database application
Database Management System (DBMS)
Database
Microsoft Access or other personal DBMS

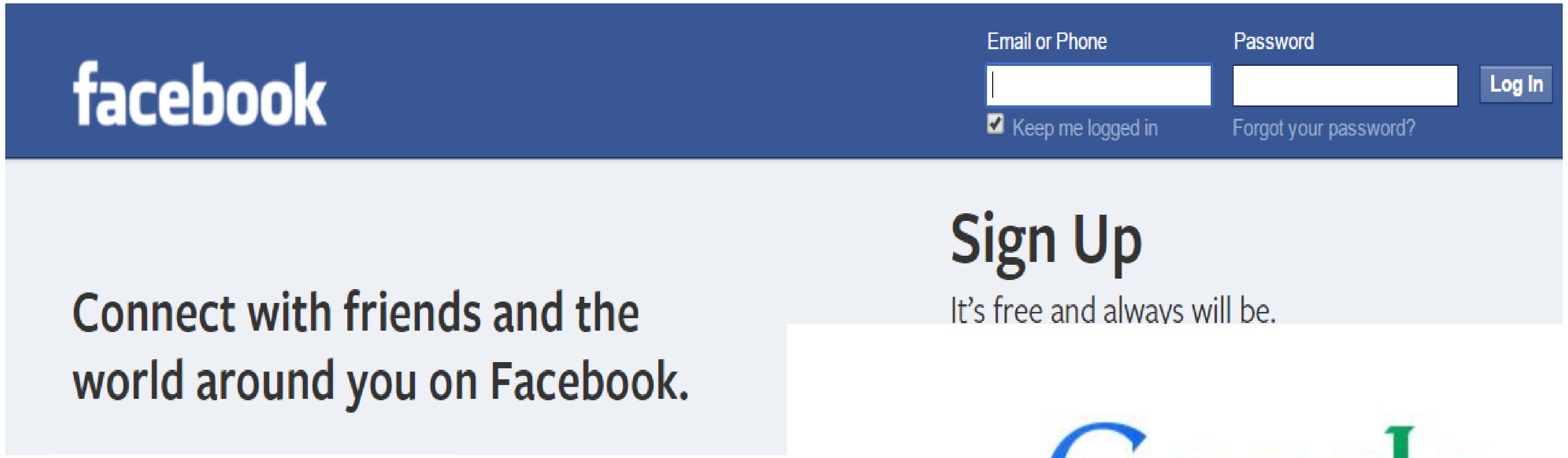
1.2 Purpose of Database Systems

Organizational Database Systems



1.3 Database-System Applications

- *The tools & services (where they put their data?)*



Sign Up

It's free and always will be.

Connect with friends and the world around you on Facebook.

All the data of these applications has its database

1.3 Database-System Applications

- ***Enterprise Information***

- Sales: For customer, product, and purchase information.

- Accounting: For payments, receipts, account balances, assets and other accounting information.

- Human resources: For information about employees, salaries, payroll taxes, and benefits, and for generation of paychecks.

- Manufacturing: For management of the supply chain and for tracking production of items in factories, inventories of items in warehouses and stores, and orders for items.

- Online retailers: For sales data noted above plus online order tracking, generation of recommendation lists, and maintenance of online product evaluations.

1.3 Database-System Applications

- ***Banking and Finance***

- Banking: For customer information, accounts, loans, and banking transactions.
- Credit card transactions: For purchases on credit cards and generation of monthly statements.
- Finance: For storing information about holdings, sales, and purchases of financial instruments such as stocks and bonds; also for storing real-time market data to enable online trading by customers and automated trading by the firm.

1.3 Database-System Applications

- ***Universities***

For student information, course registrations, and grades (in addition to standard enterprise information such as human resources and accounting).

- ***Airlines***

For reservations and schedule information. Airlines were among the first to use databases in a geographically distributed manner.

- ***Telecommunication***

For keeping records of calls made, generating monthly bills, maintaining balances on prepaid-Sales: For customer, product, and purchase information.

- ***Databases touch all aspects of our lives***

1.4 View of Data

A major purpose of a database system is to provide users with an abstract view of the data. That is, the system hides certain details of how the data are stored and maintained.

Levels of Abstraction

Physical level: describes how a record (e.g., customer) is stored.

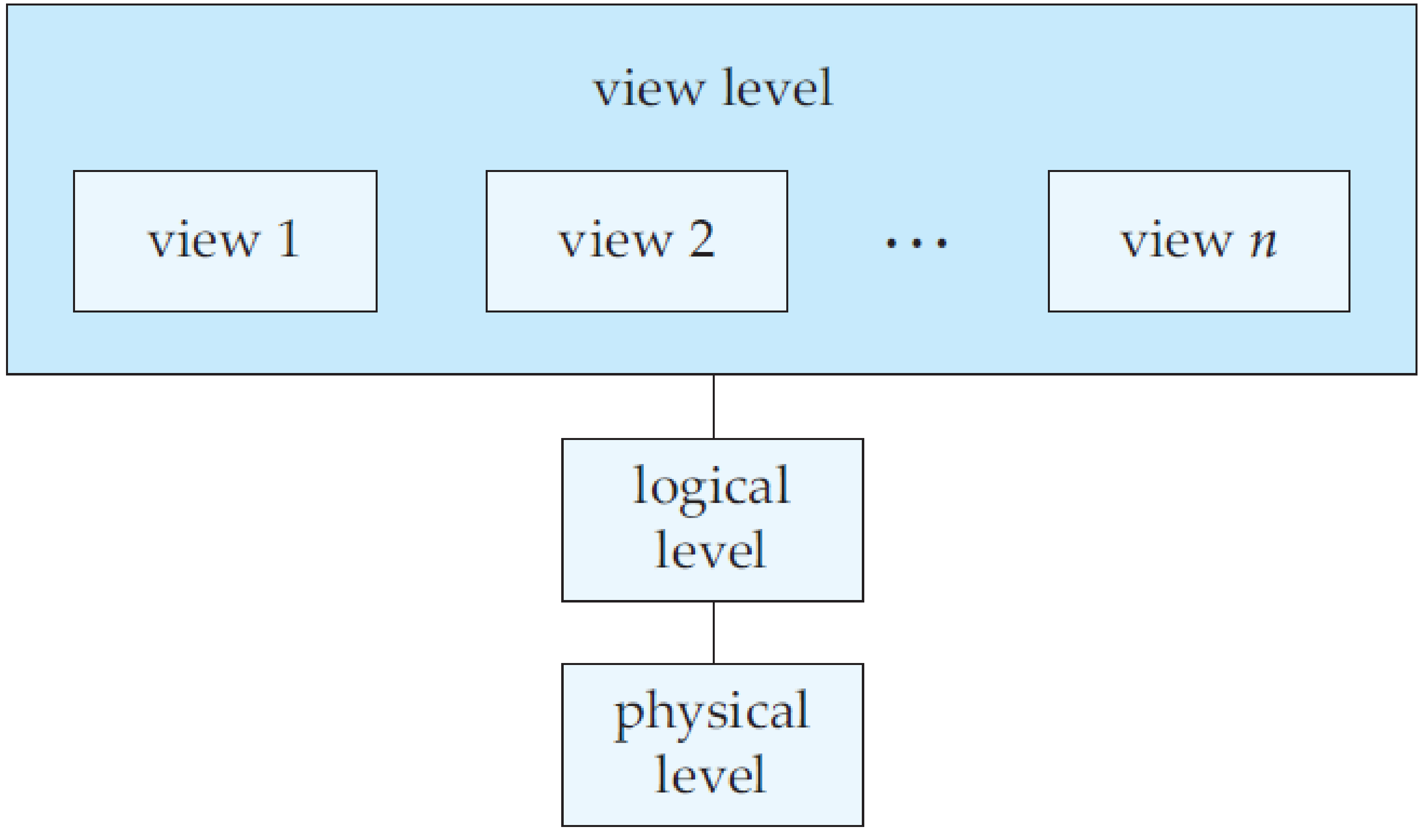
Logical level: describes data stored in database, and the relationships among the data.

```
type customer = record
        customer_id : string;
        customer_name : string;
        customer_street : string;
        customer_city : integer;
end;
```

View level: application programs hide details of data types. Views can also hide information (such as an employee's salary) for security purposes.

1.4.1 Data Abstraction

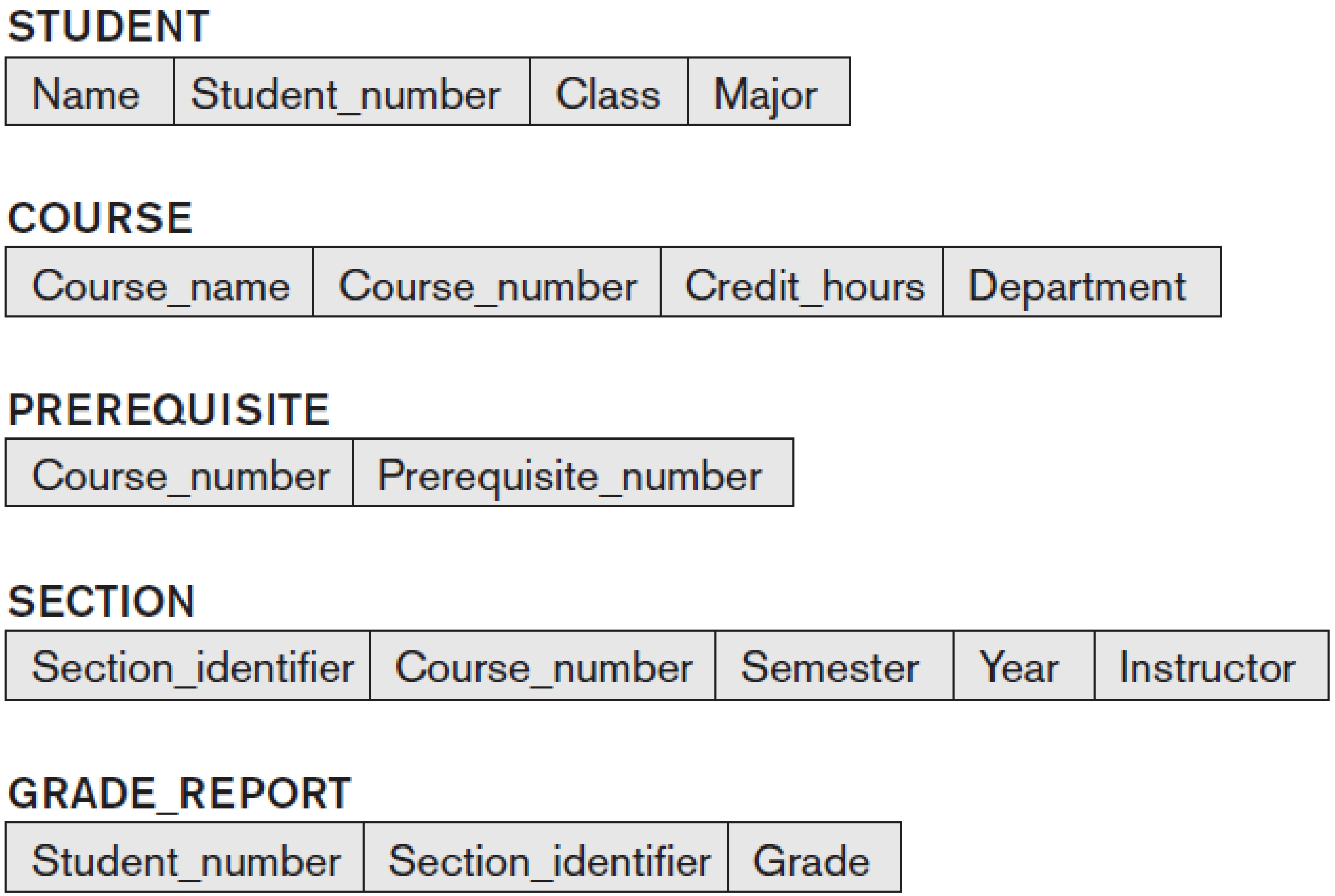
The three levels of data abstraction



1.4.2 Instances and Schemas

- Similar to types and variables in programming languages
- **Schema** – the logical structure of the database
  - Example: The database consists of information about a set of customers and accounts and the relationship between them)
  - Analogous to type information of a variable in a program
  - **Physical schema**: database design at the physical level
  - **Logical schema**: database design at the logical level
- **Instance** – the actual content of the database at a particular point in time
  - Analogous to the value of a variable
- **Physical Data Independence** – the ability to modify the physical schema without changing the logical schema
  - Applications depend on the logical schema
  - In general, the interfaces between the various levels and components should be well defined so that changes in some parts do not seriously influence others.

1.4.2 Instances and Schemas



STUDENT

Name	Student_number	Class	Major

COURSE

Course_name	Course_number	Credit_hours	Department

PREREQUISITE

Course_number	Prerequisite_number

SECTION

Section_identifier	Course_number	Semester	Year	Instructor

GRADE_REPORT

Student_number	Section_identifier	Grade

Schema diagram for the university database

1.4.3 Data Models

Data model: a collection of tools for describing *data*, *data relationships*, *data semantics*, *data constraints*.

A database model consists of 3 components:

- The section describes the structure of the database.
- The section describes operations and defines allowed operations on data.
- The section describes integrity constraints to ensure data accuracy

1.4.3 Data Models

- Object-based data models
  - Entity-Relationship data model (mainly for database design)
  - Object-oriented and Object-relational model
  - Semantic data model
  - Functional data model
- Record-based data models
  - Relational model
  - Network model
  - Hierarchical model
- Self-describing data models
  - XML
  - The key-value stores
  - NoSQL syaatems

Entity-Relationship data model

- ✓ An **entity** is a "thing" or "object" in the real world that is distinguishable from all other objects.

Example: each person in a university is an entity.

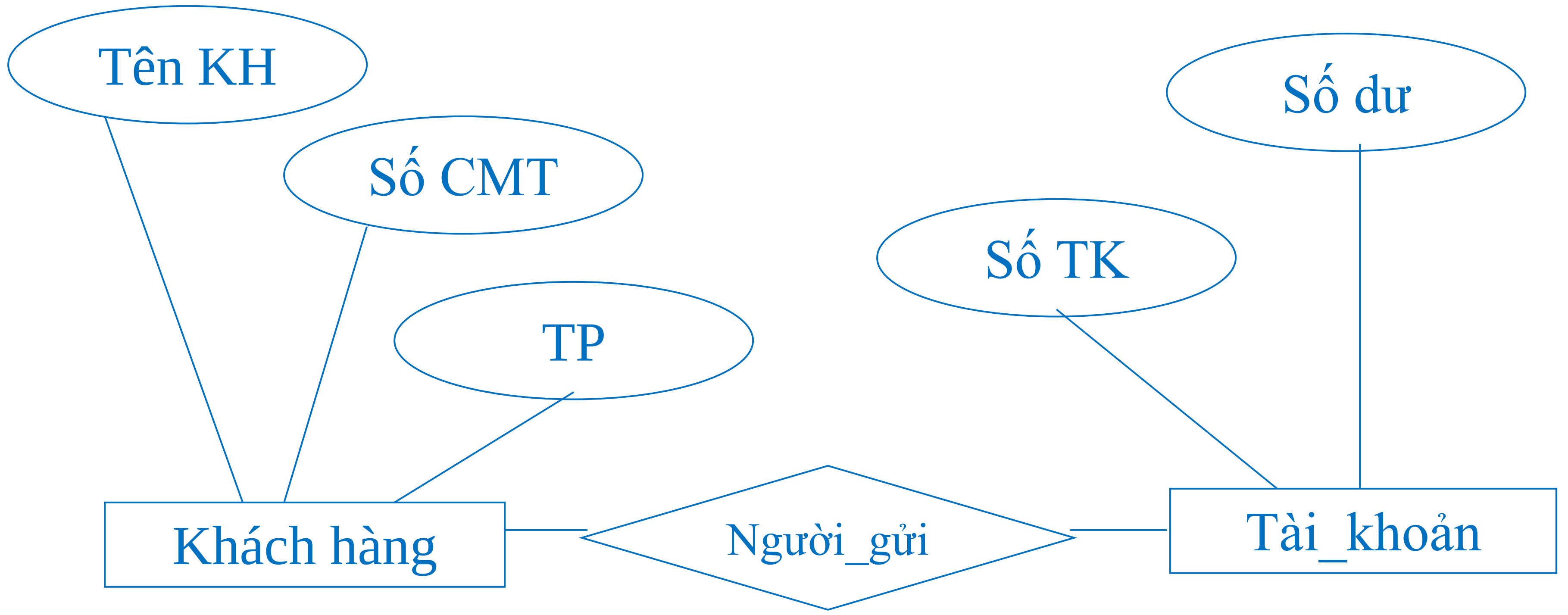
- ✓ A **relationship** is an association among several entities.

Example, we can define a relationship *advisor* that associates instructor Katz with student Shankar

- ✓ **Constraints** are mapping cardinalities, or cardinality ratios, express the number of entities to which another entity can be associated via a relationship set.

CMC UNIVERSITY

Entity-Relationship data model



Object-oriented model

- ✓ **Object**: contains properties and methods.
- ✓ **Class**: a set of objects with the same set of properties and methods.
- ✓ **Message**: an object can access another object's data by calling that object's methods.

Relational model

- ✓ Data are presented in ***tables***. Each table consists of ***rows*** (records) and ***columns*** (fields | attributes).
- ✓A **relationship** between tables: due to the duplicate occurrence of some attributes in more than one table

Tên KH	Số CMT	TP	Số TK
Ban	CM1024	Hà Nội	A101
Giao	CM5566	Yên Bái	A215

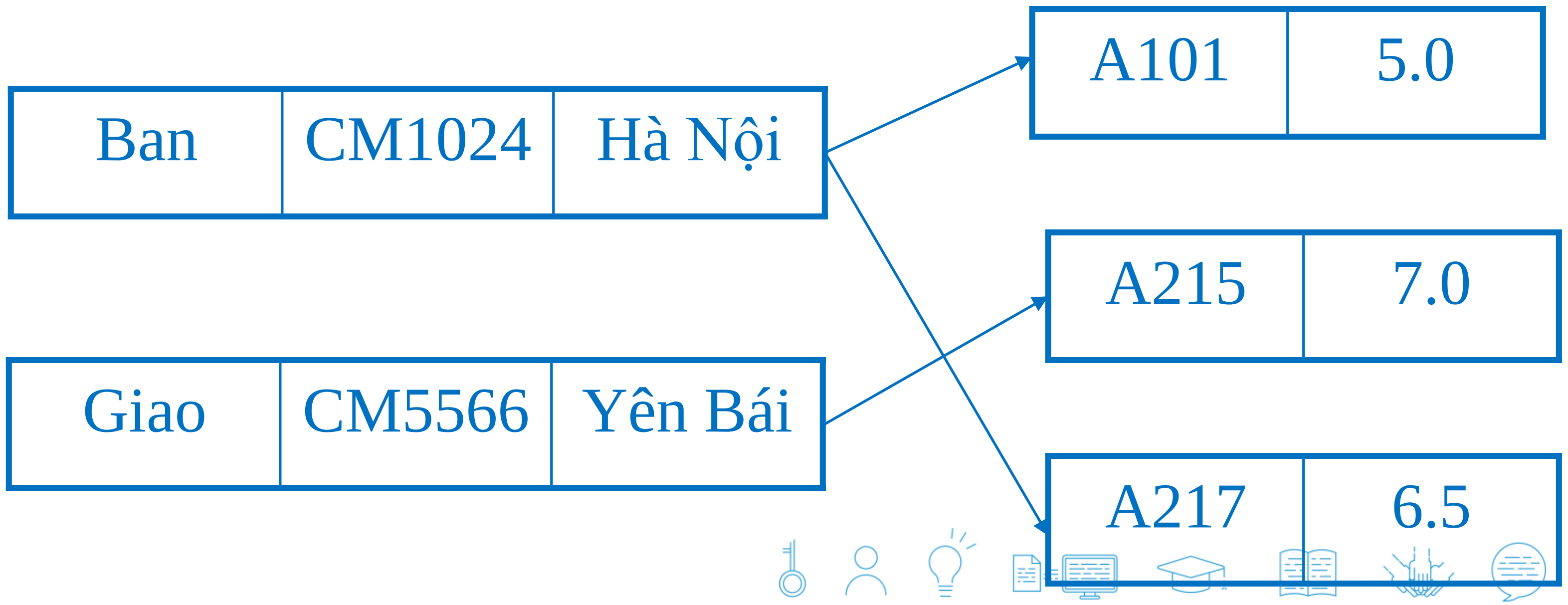
Số TK	Số dư
A101	5.0
A215	7.0
A217	6.5

Network Model

✓ Data is represented by a set of records (like Pascal's records).

✓ Links: are represented by links that can be viewed as pointers, distinguishing the object that is the owner of the link and the component objects.

The diagram of records and the links between them is a set of graphs.

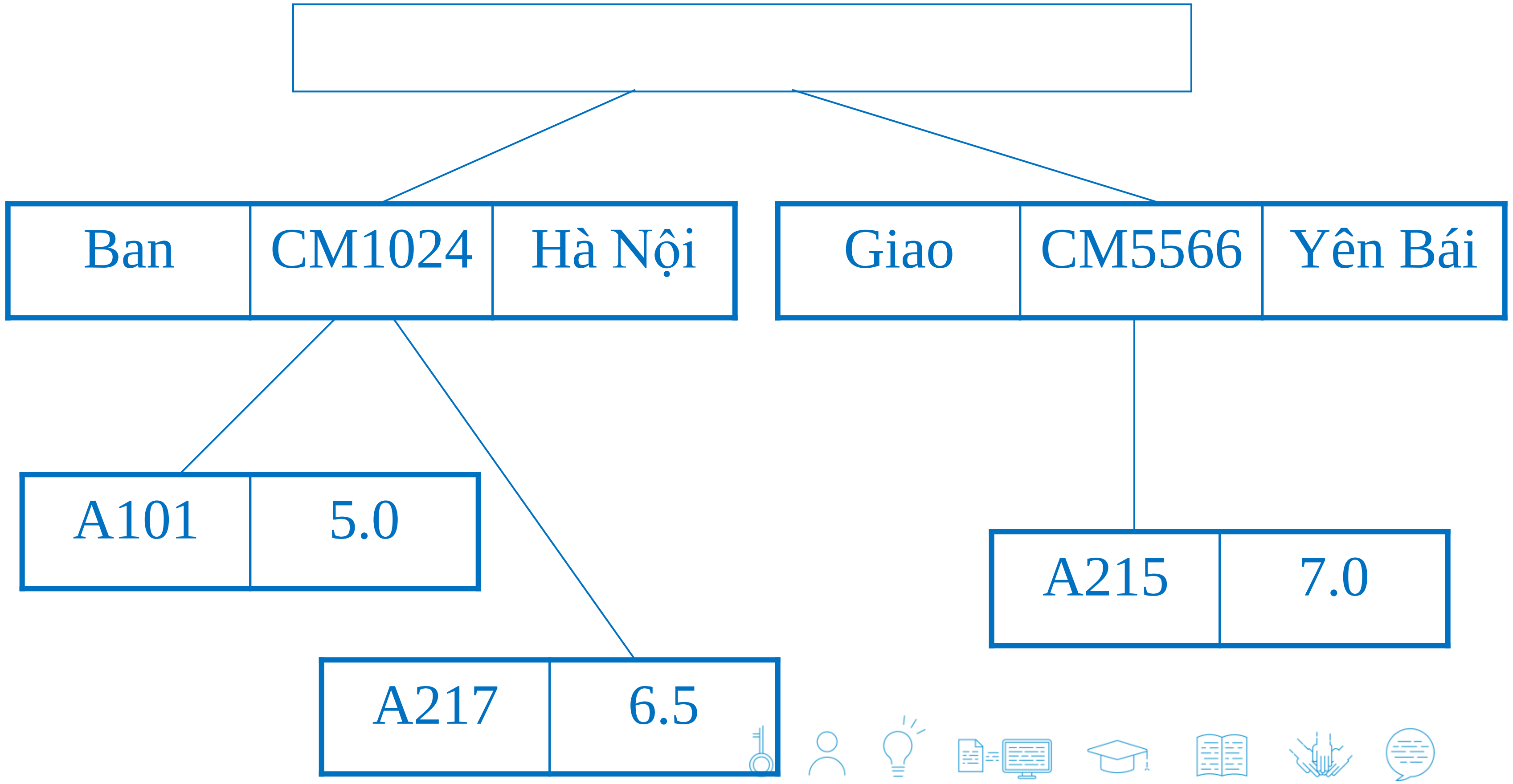


Hierarchical model

✓ Data is represented by a set of records (like Pascal's records).

✓ Links: are represented by links that can be viewed as pointers. The relationship between two objects is expressed in the form of parent-child.

The diagram of records and the links between them is structured like ***trees***.



CMC UNIVERSITY

Self-describing Data Models

- ✓ Combine the data description with the data values themselves
- ✓ Data models: XML, key-value store, NOSQL

Key

KH1 →

```
{
    _id:                    "KH1",
    Tên KH:                 "Ban",
    Số CMT:             "CM-1024",
    Tài_khoản: [
                    {Số TK: "A101",
                     Số dư: "5.0"
                    },
                    {Số TK: "A217",
                     Số dư: "6.5"
                    },
                 ]
}
```

An example of document in NOSQL MongoDB

1.5 Database Languages

Data-manipulation language (DML)

- A *data-manipulation language* (DML) is a language that enables users to access or manipulate data as organized by the appropriate data model. The types of access are:

  - Retrieval of information stored in the database

  - Insertion of new information into the database

  - Deletion of information from the database

  - Modification of information stored in the database

1.5 Database Languages

Data-manipulation language (DML)

- Two classes of languages
  - **Procedural** – user specifies what data is required and how to get those data
  - **Declarative (nonprocedural)** – user specifies what data is required without specifying how to get those data
- DML also known as query language

  - A query is a statement requesting the retrieval of information.

  - The portion of a DML that involves information retrieval is called a query language
- SQL is the most widely used query language

1.5 Database Languages

Data-definition language (DDL)

- Specify a database schema by a set of definitions expressed by a special language called a *data-definition language* (DDL). The DDL is also used to specify additional properties of the data.

Example: **create table** *account* (
account-number **char**(10),
balance **integer**)

CMC UNIVERSITY

1.5 Database Languages

Data-definition language (DDL)

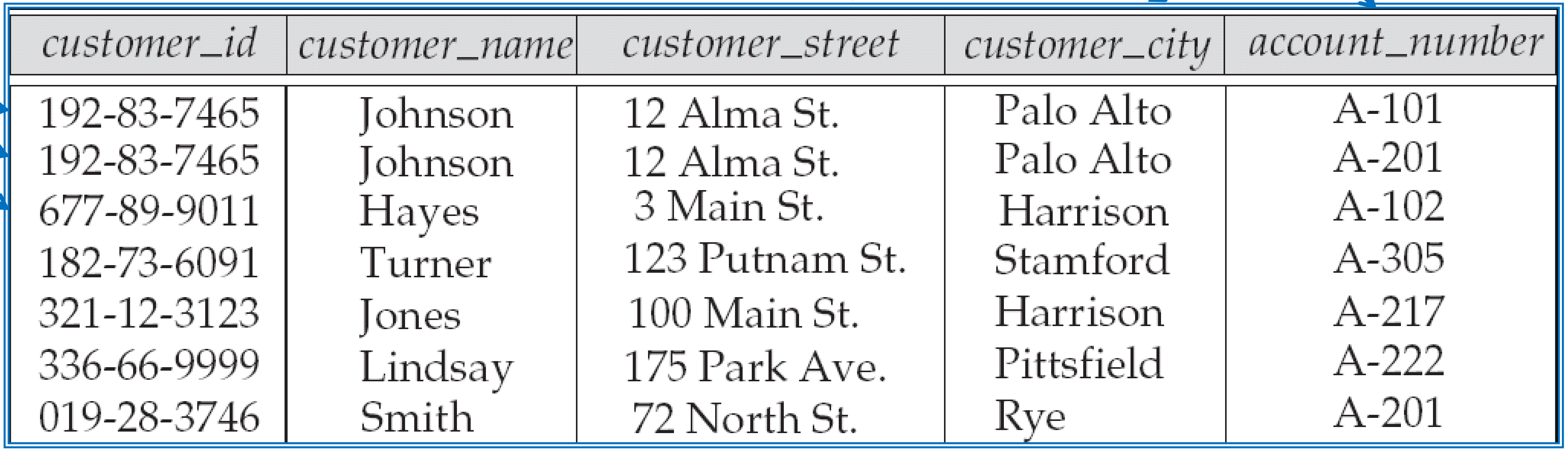
- DDL compiler generates a set of tables stored in a *data dictionary*
- Data dictionary contains metadata (i.e., data about data)
  - Database schema
  - Data *storage and definition* language
    - Specifies the storage structure and access methods used
  - Integrity constraints
    - Domain constraints
    - Referential integrity (**references** constraint in SQL)
    - Assertions
  - Authorization

1.6 Relational Database

- Example of tabular data in the relational model

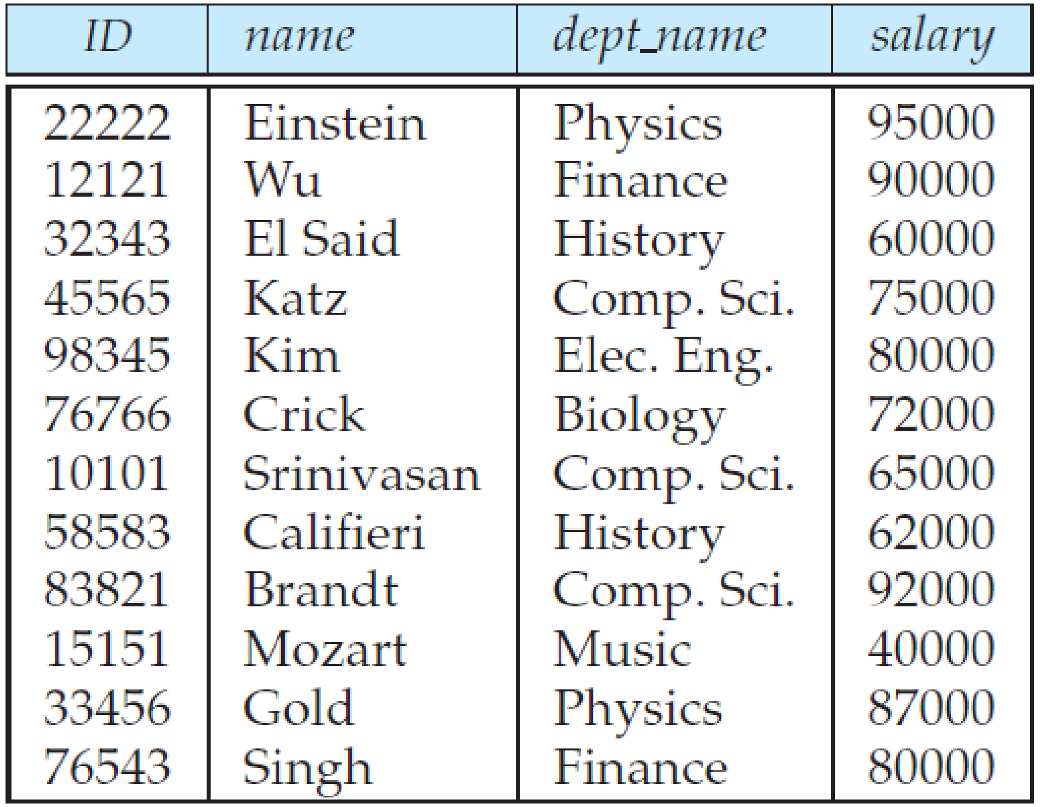
Attributes

Records



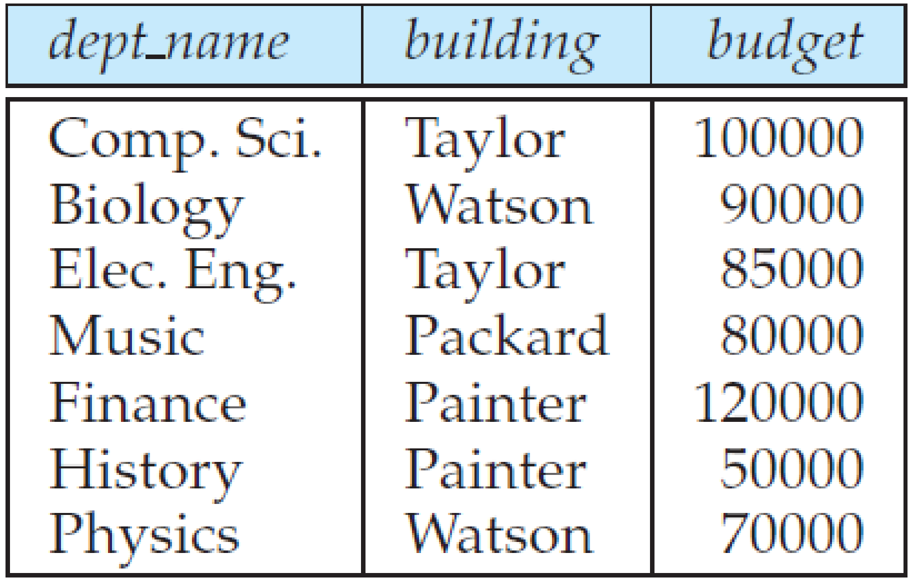
customer_id	*customer_name*	*customer_street*	*customer_city*	*account_number*
192-83-7465	Johnson	12 Alma St.	Palo Alto	A-101
192-83-7465	Johnson	12 Alma St.	Palo Alto	A-201
677-89-9011	Hayes	3 Main St.	Harrison	A-102
182-73-6091	Turner	123 Putnam St.	Stamford	A-305
321-12-3123	Jones	100 Main St.	Harrison	A-217
336-66-9999	Lindsay	175 Park Ave.	Pittsfield	A-222
019-28-3746	Smith	72 North St.	Rye	A-201

1.6 Relational Database



ID	*name*	*dept_name*	*salary*
22222	Einstein	Physics	95000
12121	Wu	Finance	90000
32343	El Said	History	60000
45565	Katz	Comp. Sci.	75000
98345	Kim	Elec. Eng.	80000
76766	Crick	Biology	72000
10101	Srinivasan	Comp. Sci.	65000
58583	Califieri	History	62000
83821	Brandt	Comp. Sci.	92000
15151	Mozart	Music	40000
33456	Gold	Physics	87000
76543	Singh	Finance	80000

(a) The *instructor* table



dept_name	*building*	*budget*
Comp. Sci.	Taylor	100000
Biology	Watson	90000
Elec. Eng.	Taylor	85000
Music	Packard	80000
Finance	Painter	120000
History	Painter	50000
Physics	Watson	70000

(b) The *department* table

A sample relational database

CMC UNIVERSITY

1.6 Relational Database

- The SQL query language is nonprocedural. A query takes as input several tables (possibly only one) and always returns a single table.

Example 1: finds the names of all instructors in the History department

```
select instructor.name
from instructor
where instructor.dept_name = 'History';
```

the result of executing this query is a table with a single column labeled *name*, and a set of rows which contains the name of an instructor whose *dept_ name*, is History

1.6 Relational Database

Example 2: finds the instructor ID and department name of all instructors associated with a department with budget of greater than $95,000

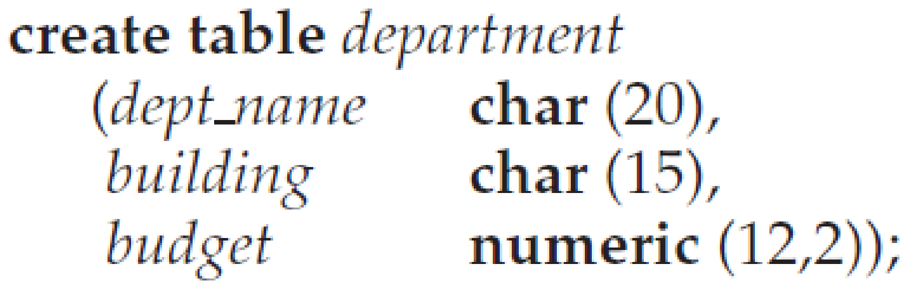
```
select instructor.ID, department.dept_name
from instructor, department
where instructor.dept_name= department.dept_name and
        department.budget > 95000;
```

- Application programs generally access databases through one of
  - Language extensions to allow embedded SQL
  - Application program interface (e.g., ODBC/JDBC) which allow SQL queries to be sent to a database

1.6 Relational Database

- SQL provides a rich DDL that allows one to define tables, integrity constraints, assertions, etc.

Example: the following SQL DDL statement defines the department table:



```
create table department
    (dept_name    char (20),
     building     char (15),
     budget       numeric (12,2));
```

- Execution of the above DDL statement creates the *department* table with three columns: *dept_name*, *building*, and *budget*, each of which has a specific data type associated with it.

1.6 Relational Database

Database Access from Application Programs

- SQL does not support some computations that are possible using a general-purpose programming language
- SQL also does not support actions such as input from users, output to displays, or communication over the network.
- The above computations and actions must be written in a host language, such as C, C++, or Java, with embedded SQL queries that access the data in the database. Application programs are programs that are used to interact with the database in this fashion.

1.6 Relational Database

Database Access from Application Programs

Examples in a university system: programs allow students to register for courses, generate class rosters, calculate student GPA, generate payroll checks, etc. To access the database, DML statements need to be executed from the host language. There are two ways to do this:

- By providing an application program interface (set of procedures) that can be used to send DML and DDL statements to the database and retrieve the results.

The Open Database Connectivity (ODBC) standard for use with the C language is a commonly used application program interface standard. The Java Database Connectivity (JDBC) standard provides corresponding features to the Java language.

- By extending the host language syntax to embed DML calls within the host language program. Usually, a special character prefaces DML calls, and a preprocessor, called the DML precompiler, converts the DML statements to normal procedure calls in the host language.

1.7 Database Design

The process of designing the general structure of the database:

- Gathering requirements and analysis
- Concept design
- Logical design – Deciding on the database schema. Database design requires that we find a "good" collection of relation schemas.
  - Business decision – What attributes should we record in the database?
  - Computer Science decision – What relation schemas should we have and how should the attributes be distributed among the various relation schemas?
- Physical design – Deciding on the physical layout of the database

1.7 Database Design

- *Concept design*

A high-level data model provides the database designer with a conceptual framework in which to specify the data requirements of the database users, and how the database will be structured to fulfill these requirements:

- specification of user requirements: characterize fully the data needs of the prospective database users
- choose a data model, and by apply the concepts of the chosen data model, translates these requirements into a conceptual schema of the database
- The designer reviews the schema to confirm that all data requirements are indeed satisfied and are not in conflict with one another; examine the design to remove any redundant features

The focus at this point is on describing the data and their relationships, rather than on specifying physical storage details

1.7 Database Design

In terms of the relational model, the conceptual-design process involves decisions on what attributes we want to capture in the database and how to group these attributes to form the various tables. There are principally two ways to tackle the problem:

- Use the entity-relationship model;
- Employ a set of algorithms (collectively known as normalization) that takes as input the set of all attributes and generates a set of tables

1.7 Database Design

- *Logic design*

The designer maps the high-level conceptual schema onto the implementation data model of the database system that will be used.

- *Physical* design

The designer uses the resulting system-specific database schema in the subsequent physical-design phase, in which the physical features of the database are specified. These features include the form of file organization and the internal storage structures

1.7 Database Design

Example: how a database for a university could be designed.

The initial specification of user requirements may be based on interviews with the database users, and on the designer's own analysis of the organization.

Here are the major characteristics of the university:

• The university is organized into departments. Each department is identified by a unique name (*dept_name*), is located in a particular *building*, and has a budget.

• Each department has a list of courses it offers. Each course has associated with it a *course_id*, *title*, *dept_name*, and *credits*, and may also have have associated *prerequisites*.

• Instructors are identified by their unique *ID*. Each instructor has *name*, associated department (*dept_name*), and *salary*.

1.7 Database Design

- Students are identified by their unique ID. Each student has a *name*, an associated major department (*dept_name*), and *tot_cred* (total credit hours the student earned thus far).
- The university maintains a list of classrooms, specifying the name of the *building*, *room_number*, and room *capacity*.
- The university maintains a list of all classes (sections) taught. Each section is identified by a *course_id*, *sec_id*, *year*, and *semester*, and has associated with it a *semester*, *year*, *building*, *room_number*, and *time_slot_id* (the time slot when the class meets).
- The department has a list of teaching assignments specifying, for each instructor, the sections the instructor is teaching.
- The university has a list of all student course registrations, specifying, for each student, the courses and the associated sections that the student has taken (registered for).

CMC UNIVERSITY

1.8 Data Storage and Querying

A database system is partitioned into modules that deal with each of the responsibilities of the overall system. The functional components of a database system can be broadly divided into the storage manager and the query processor components.

The storage manager is important because databases typically require a large amount of storage space

The query processor is important because it helps the database system to simplify and facilitate access to data.

1.8 Data Storage and Querying

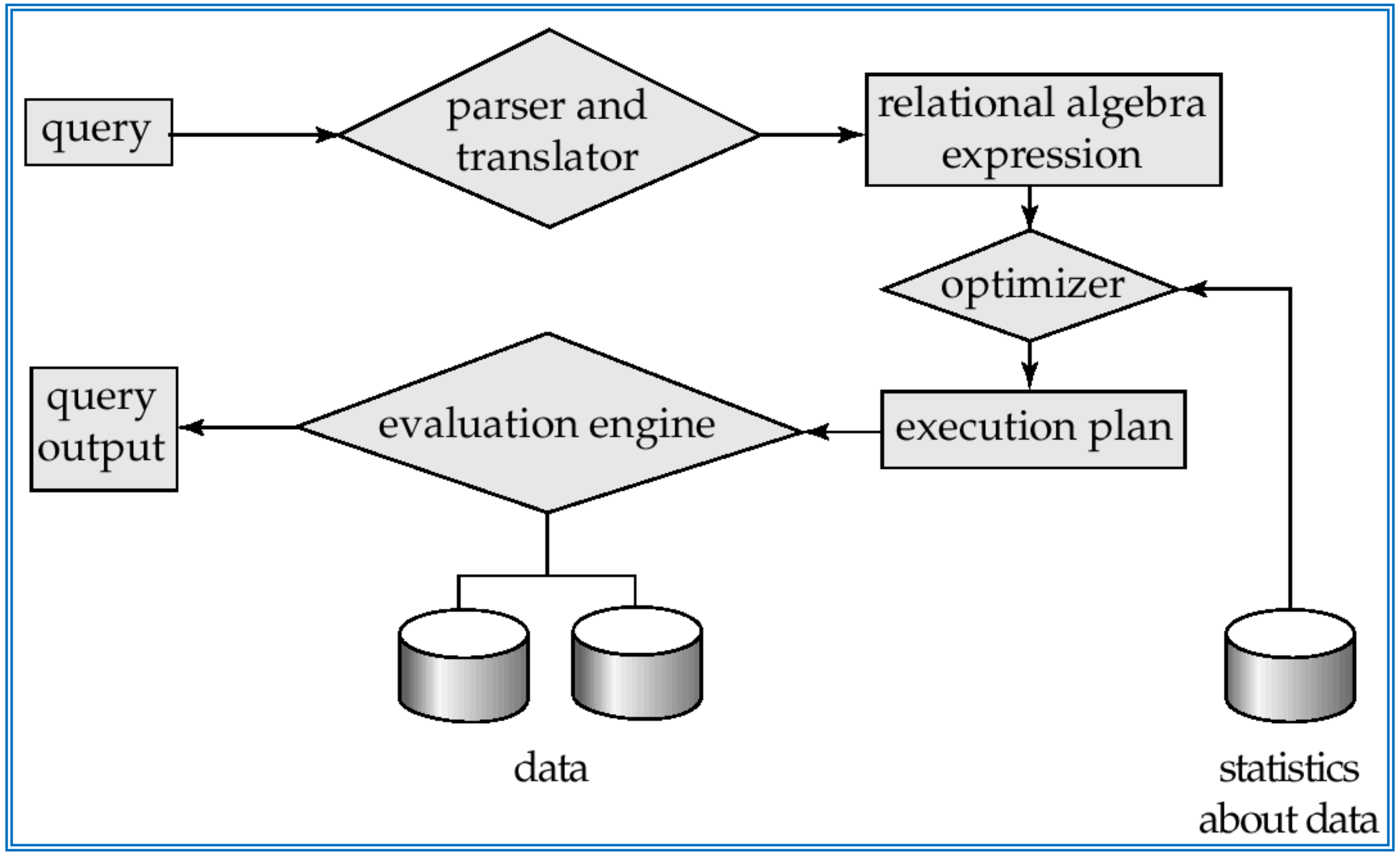
Storage Management

- **Storage manager** is a program module that provides the interface between the low-level data stored in the database and the application programs and queries submitted to the system.
- The storage manager is responsible to the following tasks:
  - Interaction with the file manager
  - Efficient storing, retrieving and updating of data
- Issues:
  - Storage access
  - File organization
  - Indexing and hashing

1.8 Data Storage and Querying

Query Processing

1. Parsing and translation
2. Optimization
3. Evaluation



1.8 Data Storage and Querying

Query Processing

- Alternative ways of evaluating a given query
  - Equivalent expressions
  - Different algorithms for each operation
- Cost difference between a good and a bad way of evaluating a query can be enormous
- Need to estimate the cost of operations
  - Depends critically on statistical information about relations which the database must maintain
  - Need to estimate statistics for intermediate results to compute cost of complex expressions

1.9 Transaction Management

- A **transaction** is a collection of operations that performs a single logical function in a database application
- **Transaction-management component** ensures that the database remains in a consistent (correct) state despite system failures (e.g., power failures and operating system crashes) and transaction failures.
- **Concurrency-control manager** controls the interaction among the concurrent transactions, to ensure the consistency of the database.

1.10 Database Architecture

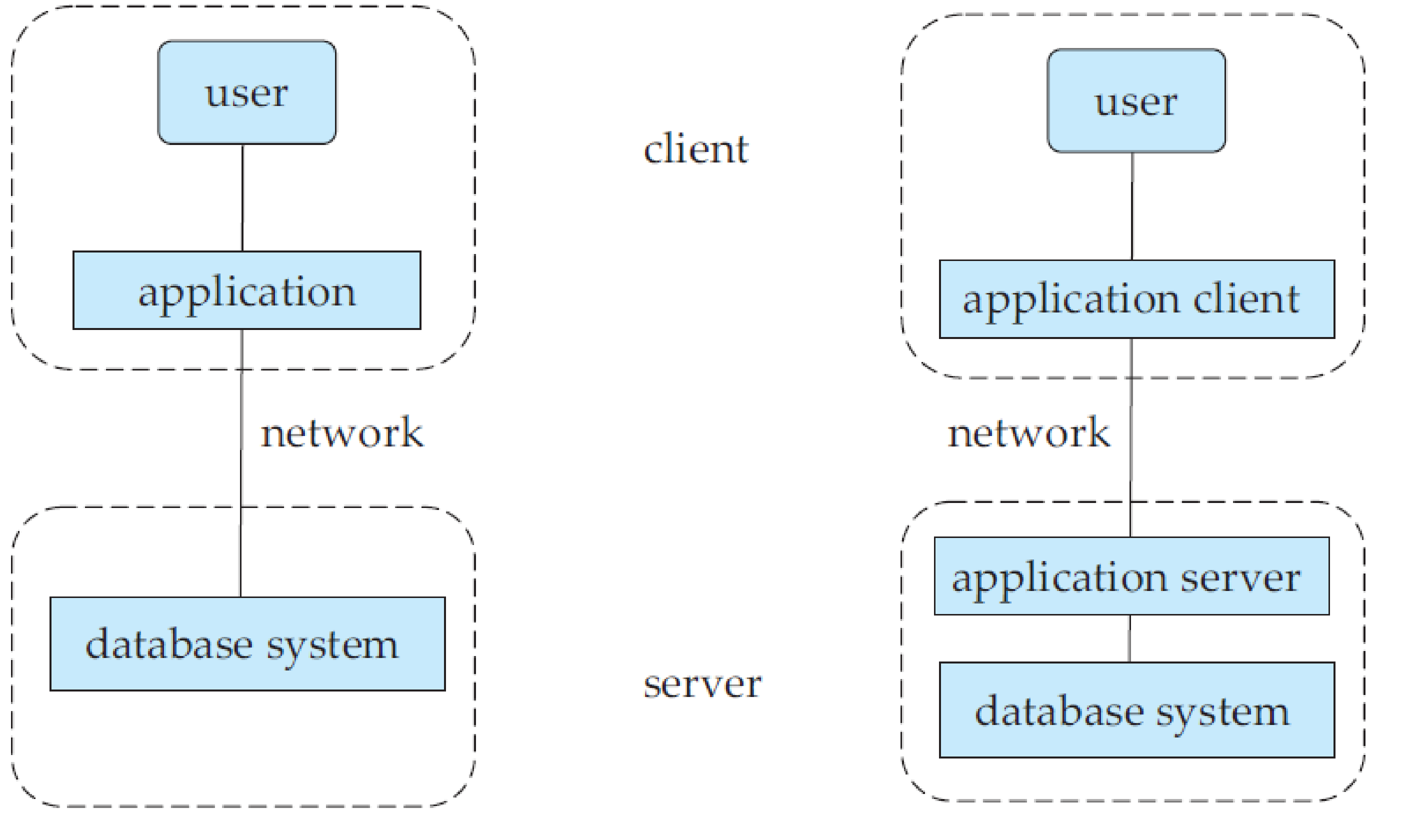
The architecture of a database systems is greatly influenced by the underlying computer system on which the database is running:

- Centralized
- Client-server
- Parallel (multi-processor)
- Distributed

1.10 Database Architecture

Database applications are usually partitioned into two or three parts.

- In a two-tier architecture, the application resides at the client machine, where it invokes database system functionality at the server machine through query language statements.
- In a three-tier architecture, the client machine acts as merely a front end and does not contain any direct database calls. Instead, the client end communicates with an application server, usually through a forms interface.



(a) Two-tier architecture (b) Three-tier architecture

1.11 Data mining and information retrieval

Data mining

- The term data mining refers loosely to the process of semiautomatically analyzing large databases to find useful patterns. That is, data mining deals with "knowledge discovery in databases."

- Some types of knowledge discovered from a database can be represented by a set of rules. Example of a rule, stated informally: "Young women with annual incomes greater than $50,000 are the most likely people to buy small sports cars."
- Other types of knowledge are represented by equations relating different variables to each other, or by other mechanisms for predicting outcomes when the values of some variables are known

1.11 Data mining and information retrieval

Data mining

- There are a variety of possible types of patterns that may be useful, and different techniques are used to find different types of patterns
- Usually there is a manual component to data mining, consisting of *preprocessing data* to a form acceptable to the algorithms, and *postprocessing* of discovered patterns to find novel ones that could be useful
- To execute queries efficiently on diverse data, companies have built data *warehouses*. Data warehouses gather data from multiple sources under a unified schema, at a single site

1.11 Data mining and information retrieval

Information retrieval

- Textual data, too, has grown explosively. Textual data is unstructured, unlike the rigidly structured data in relational databases
- Querying of unstructured textual data is referred to as *information retrieval*.
- The emphasis in the field of information systems is different from that in database systems, concentrating on issues such as *querying based on keywords*; *the relevance of documents to the query*; and the *analysis*, *classification*, *and indexing* of documents.

1.12 Database Users and Administrators

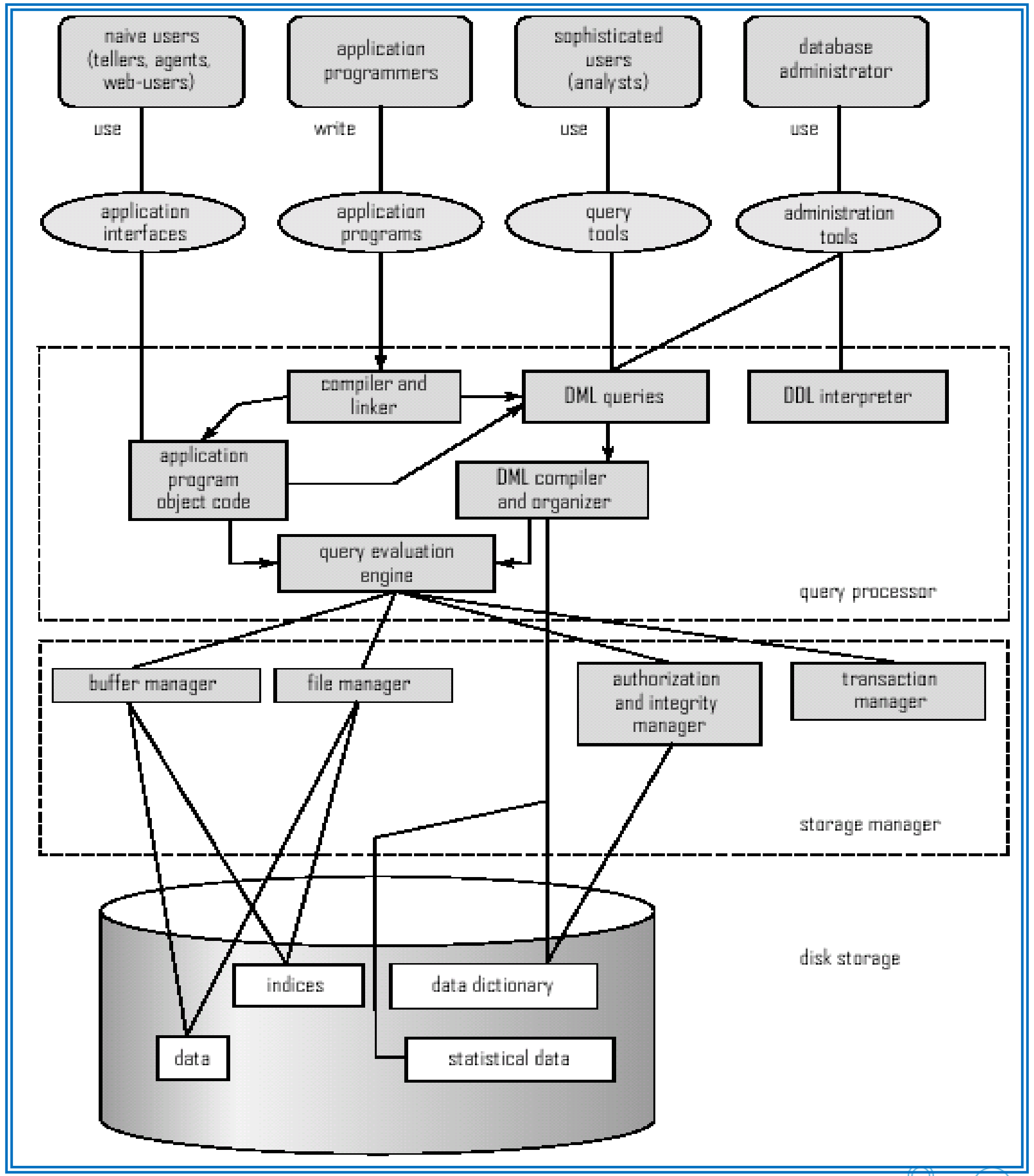
Users are differentiated by the way they expect to interact with the system

- **Application programmers** – interact with system through DML calls
- **Sophisticated users** – form requests in a database query language
- **Specialized users** – write specialized database applications that do not fit into the traditional data processing framework
- **Naïve users** – invoke one of the permanent application programs that have been written previously
  - Examples, people accessing database over the web, bank tellers, clerical staff

1.12 Database Users and Administrators

- Coordinates all the activities of the database system; the database administrator has a good understanding of the enterprise’s information resources and needs.
- Database administrator's duties include:
  - Schema definition
  - Storage structure and access method definition
  - Schema and physical organization modification
  - Granting user authority to access the database
  - Specifying integrity constraints
  - Acting as liaison with users
  - Monitoring performance and responding to changes in requirements

Overall System Structure



1.13 History of Database Systems

- 1950s and early 1960s:
  - Data processing using magnetic tapes for storage
    - Tapes provide only sequential access
  - Punched cards for input
- Late 1960s and 1970s:
  - Hard disks allow direct access to data
  - Network and hierarchical data models in widespread use
  - Ted Codd defines the relational data model
    - Would win the ACM Turing Award for this work
    - IBM Research begins System R prototype
    - UC Berkeley begins Ingres prototype
  - High-performance (for the era) transaction processing

1.13 History of Database Systems

- 1980s:
  - Research relational prototypes evolve into commercial systems
    - SQL becomes industrial standard
  - Parallel and distributed database systems
  - Object-oriented database systems
- 1990s:
  - Large decision support and data-mining applications
  - Large multi-terabyte data warehouses
  - Emergence of Web commerce
- 2000s:
  - XML and XQuery standards
  - Automated database administration
  - NoSQL

CMC UNIVERSITY

THANK YOU