Bases de Dados

Introduction

BD 2018/2019

Departamento de Engenharia Informática
Instituto Superior Técnico

Slides baseados nos slides oficiais dos livros Database Management Systems, de Ramakrishnan e Gehrke e Database System Concepts, de Silberschatz, Korth e Sudarshan

Outline

Outline

Database Management System (DBMS)

DBMS contains information about a particular enterprise

- Collection of interrelated data (i.e. a database)
- Set of programs to access the data
- An environment that is both convenient and efficient to use

Database Applications

...some examples...

Banking (transactions); Airlines (reservations, schedules); Universities (registration, grades); Sales (customers, products, purchases); Online retailers (order tracking, recommendations); Manufacturing (production, inventory, orders, supply chain); Human resources (employee records, salaries, tax deductions); Social Networks (users, connections); ...

Databases touch all aspects of our lives

Why Database Systems?

- In the early days, database applications were built directly on top of file systems

Why Database Systems?

- In the early days, database applications were built directly on top of file systems
- Drawbacks of using file systems to store data:

Why 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

Why 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

Why 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
  - 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

Why Database Systems? (cont.)

- Drawbacks of using file systems (cont.)

Why Database Systems? (cont.)

- 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

Why Database Systems? (cont.)

- 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

Why Database Systems? (cont.)

- 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

Why Database Systems? (cont.)

- 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

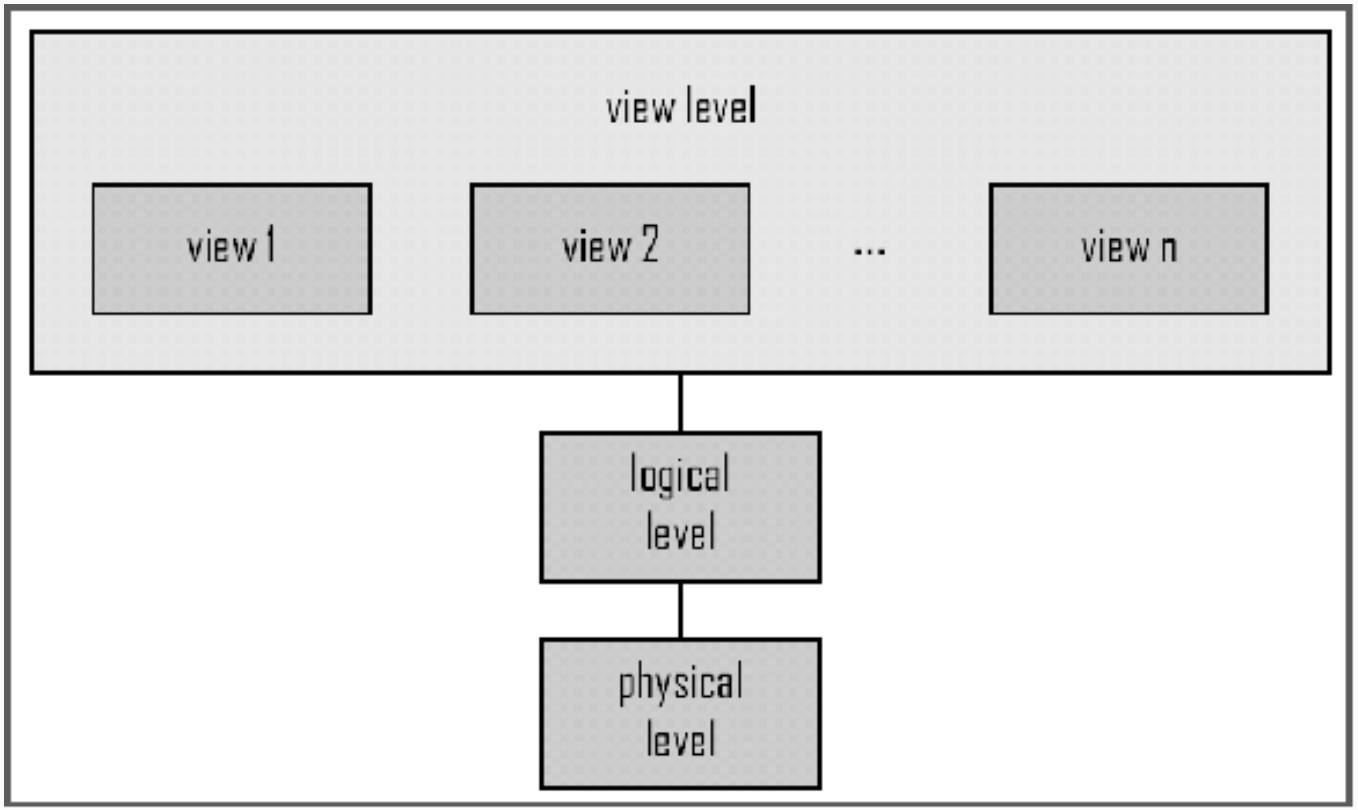
Why Not Database Systems?

- Complex software application
- Not suitable for real-time applications
- Not suitable for non-structured data
  - e.g. text, graphs, ...
- Possibly high startup overhead
  - Investment in HW and SW
  - Learning curve

Outline

Levels of Abstraction

We can separate the way we view a database in several levels



Levels of Abstraction (cont.)

- Physical level: describes how a record (e.g., customer) is stored.
- Logical level: describes data stored in database, and the relationships among the data.
- View level: application programs hide details of data types. Views can also hide information (such as an employee's salary) for security purposes.

Data Independence

- Logical Data Independence - the ability to modify the logical schema without changing the application interface
- Physical Data Independence - the ability to modify the physical schema without changing 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.

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

Data Models

- A collection of tools for describing
  - Data
  - Data relationships
  - Data semantics
  - Data constraints
- Examples:
  - Relational model
  - Entity-Relationship data model (mainly for database design)
  - Semistructured data model (XML)
  - Object-based data models (Object-oriented and Object-relational)

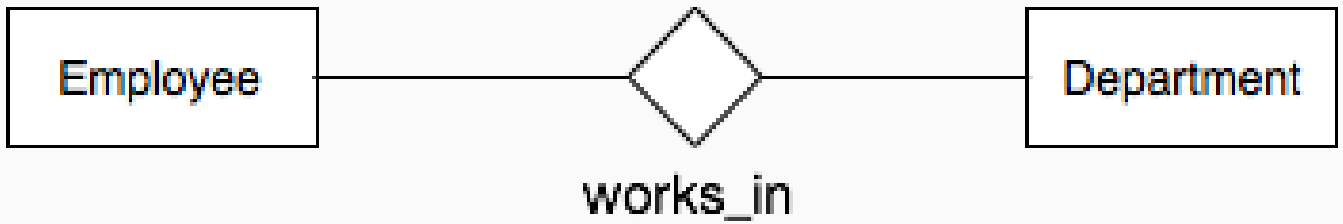
Relational Model

- Example of tabular data in the relational model

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

The Entity-Relationship Model

- Models an enterprise as a collection of entities and relationships
  - Entity: a *thing* or *object* in the enterprise that is distinguishable from other objects, described by a set of attributes
  - Relationship: an association among several entities
- Represented diagrammatically by an entity-relationship diagram:



Data Manipulation and Definition Languages

- Data Manipulation Language (DML)
  - Language for accessing and manipulating the data organized by the appropriate data model
    - Also known as query language
  - 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
  - SQL is the most widely used query language
- Data Definition Language (DDL)
  - Specification notation for defining the database schema
  - DDL compiler generates a set of tables stored in the database

SQL

Widely used declarative language

- Example: find the name of the customer with customer-id 192-83-7465

```
select customer_name
from customer
where customer_id = '192-83-7465'
```

- Example: creating the 'account' table

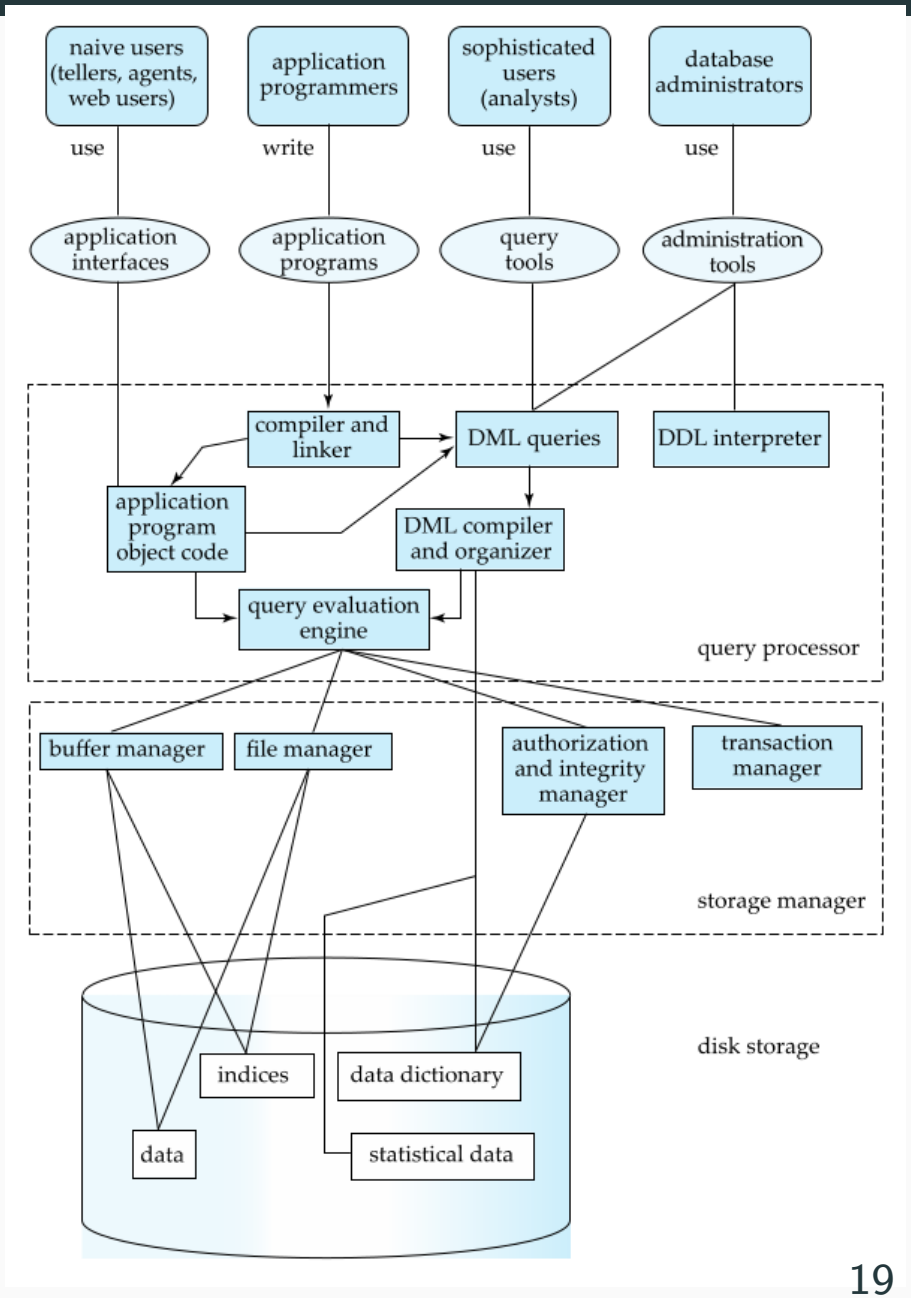
```
create table account (
        account_number char(10),
        balance integer
)
```

Outline

Overall System Structure

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

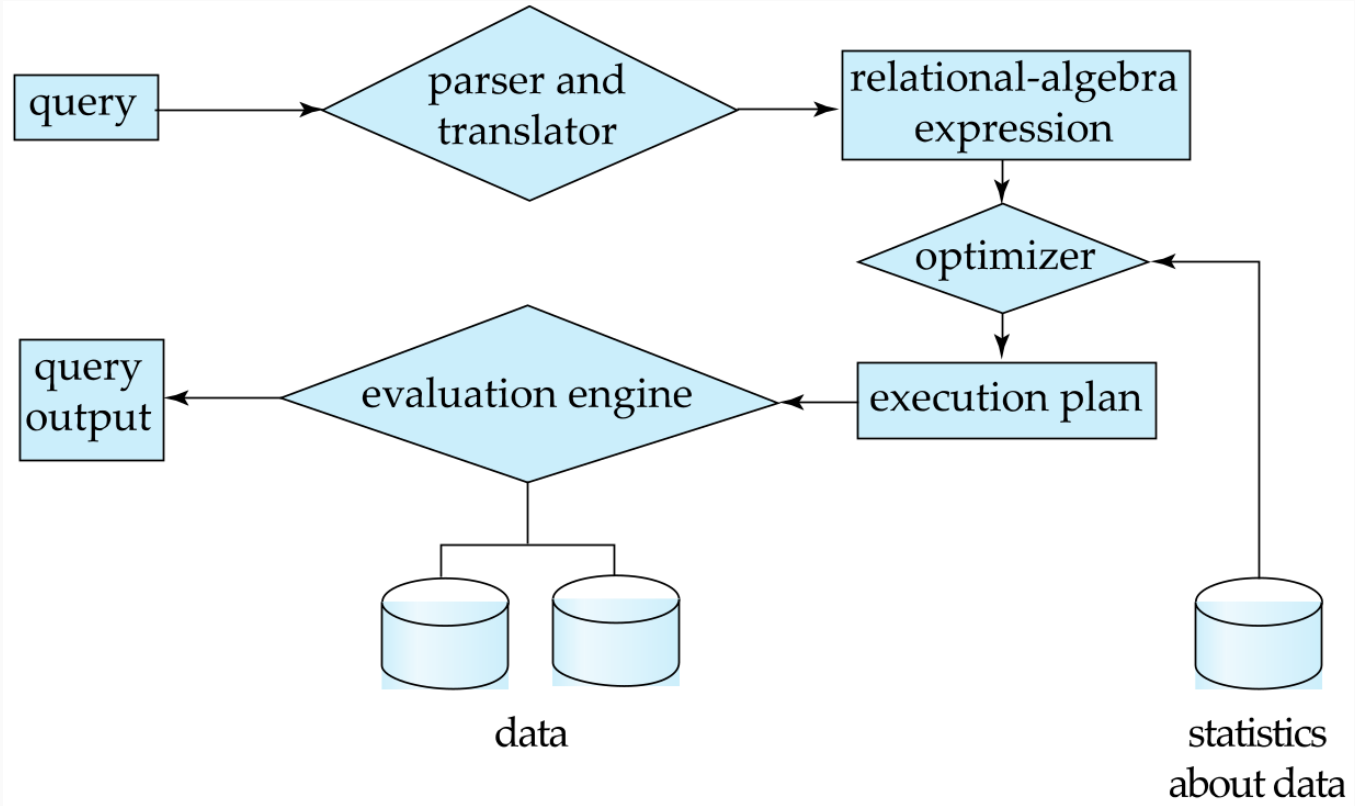


Storage Management

- 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

Query Processing

1. Parsing and translation
2. Optimization
3. Evaluation



Query Processing (cont.)

- 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
- Alternative ways of evaluating a given query
  - Equivalent expressions
  - Different algorithms for each operation

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.

Outline

Database Design

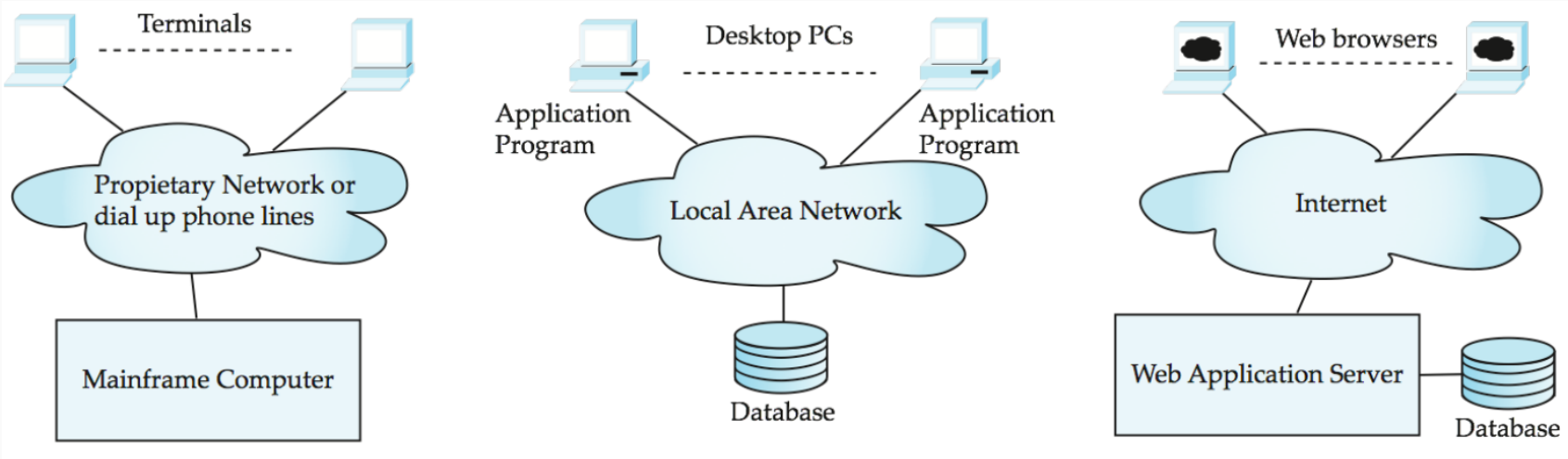
The process of designing the general structure of the database:

- Logical Design - Deciding on the database schema. Database design requires that we find a "good" collection of relation schemas.
  - What attributes should we record in the database?
  - 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

Application Architectures

Three distinct era's of application architecture:

- mainframe era (1960's and 70's)
- personal computer era (1980's)
- Web era (1990's onwards)



Database Users

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

Database Administrator

- 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

Questions?