



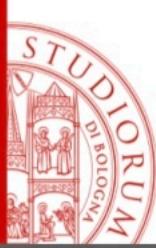
Databases

Introduction



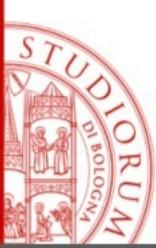
Studying Method

- Self study, focus on the fundamental concepts and refer to personal experiences
- Doing exercises
- Developing a project or attending the exercises during the lectures, using a proper tool (such as *DB2*, *SQLServer*, *Oracle*, *PostgreSQL*, *MySQL*, *MS Access*, ...)



Data base

- An organized set of data that supports on carrying out specific activities (within an institution, an enterprise, an office, a human)



Point of view

- Methodological
- Technological



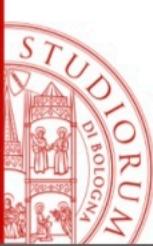
Contents

- Models for organizing data
- Languages for using data
- Systems for handling data
- Database design methodologies



Information system

- A component of an institution that handles the interesting pieces of information (e.g. in order to pursue the corporate goals)
- Each institution has its own information system, that could be not explicitly identifiable within its organization
- The information system supports other “sub-systems”. For this very reason it should be studied inside its operative context.



Handling the information

- Gathering, acquisition
- Storing, preservation
- Elaboration, transformation, production
- Distribution, communication, exchange



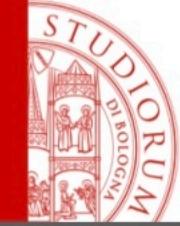
Information system and automation

- The idea of “information system” is independent from any computer automation:
 - Some organizations have the only mission of handling data (e.g. demographic recordings and banking services) and they existed for centuries



IT System

- One of the automated parts of the Information System:
It's the information system component that handle information using computer technologies



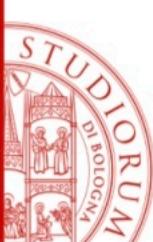
IT System

Enterprise

Organization

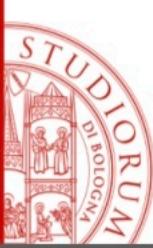
Information System

IT System



Handling the information

- Different human activities could represent the pieces of information differently:
 - informal ideas
 - natural language (written or word of mouth, formal or conversational, in different languages...)
 - drawings, plottings, diagrams
 - numbers and codes
- Such information has different storage media:
 - Human brain, paper, electronical devices



Information and data (1)

- Within (e.g.) IT Systems, the **information** is roughly expressed as **data**



Information and data (2)

(Oxford Dictionary)

information: Facts provided or learned about something or someone.

data: Pieces of information. A fixed starting point of a operation.

Information and data



Mon-Fri

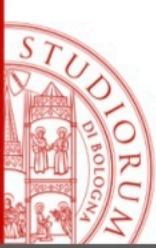


Saturday



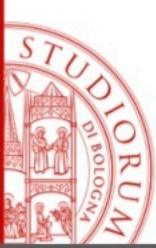
Holiday

- What does those numbers mean?
- Road signs in Finland; they are hours
- But what is the difference between them?
- The datum is useless without its “interpretation”



Handling information

- Data is the result of the process of organizing, coding and storing of information
- E.g., within demographic registries when referring to citizens
 - Wordy descriptions
 - Name and Surname
 - Personal pieces of information
 - Personal Tax Code



Why data?

- It is difficult to precisely represent very rich pieces of information and knowledge
- Data are a strategic resource, because they are more stable than other representation (business processes, technologies, human roles):
 - e.g., data in banks and demographic registries



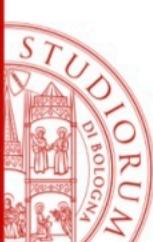
Databases

(generical meaning)

- An organized set of data that supports on carrying out specific activities of an entity (an institution, an enterprise, an office, a human)

(specific meaning)

- Set of data handled by a DBMS



DataBase Management System (DBMS)

- Any system handling data collections that are:
 - big
 - persistent
 - shared
- While ensuring:
 - privacy
 - reliability
 - efficiency
 - effectiveness



DBMS

- Software as a product and as a service (complessi) available on the market:
 - DB2
 - Oracle
 - SQLServer
 - MySQL
 - PostgreSQL
 - Access
 - BigQuery
 - SQLite



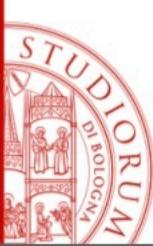
Databases are ... big

- their size are (by far) greater than the computing systems' main memory
- Only the devices' physical limit is considered
- Some example of big sizes
 - 1-5 Terabyte (transactional data)
 - 30-50 Terabyte (decision data)
 - 500-800 Terabyte (scientific data)
 - 100 billions of records



Data bases are ... persistent

- their lifetime is independent from the lifetime of the computer processes using such data



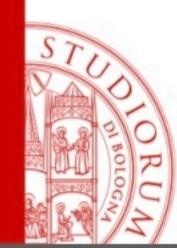
Databases are ... shared

- Every (big) corporation is sliced in different areas, e.g. it could carry out different activities
- Each area/activity has a information (sub)system (that is not necessarily detached from the central one)



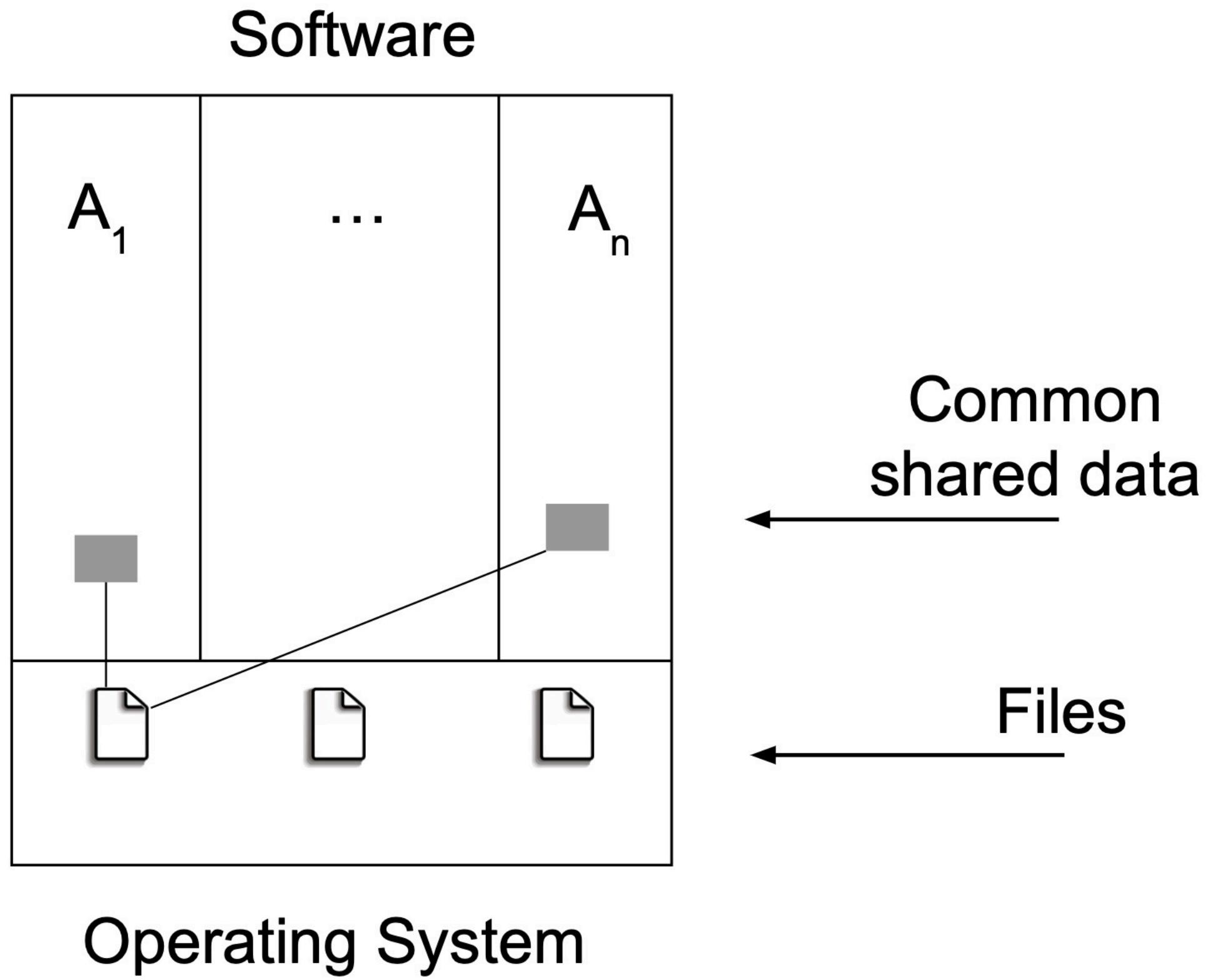
Problems

- Redundancy...
 - same data appears many times
- ...could cause inconsistency:
 - different descriptions could not match

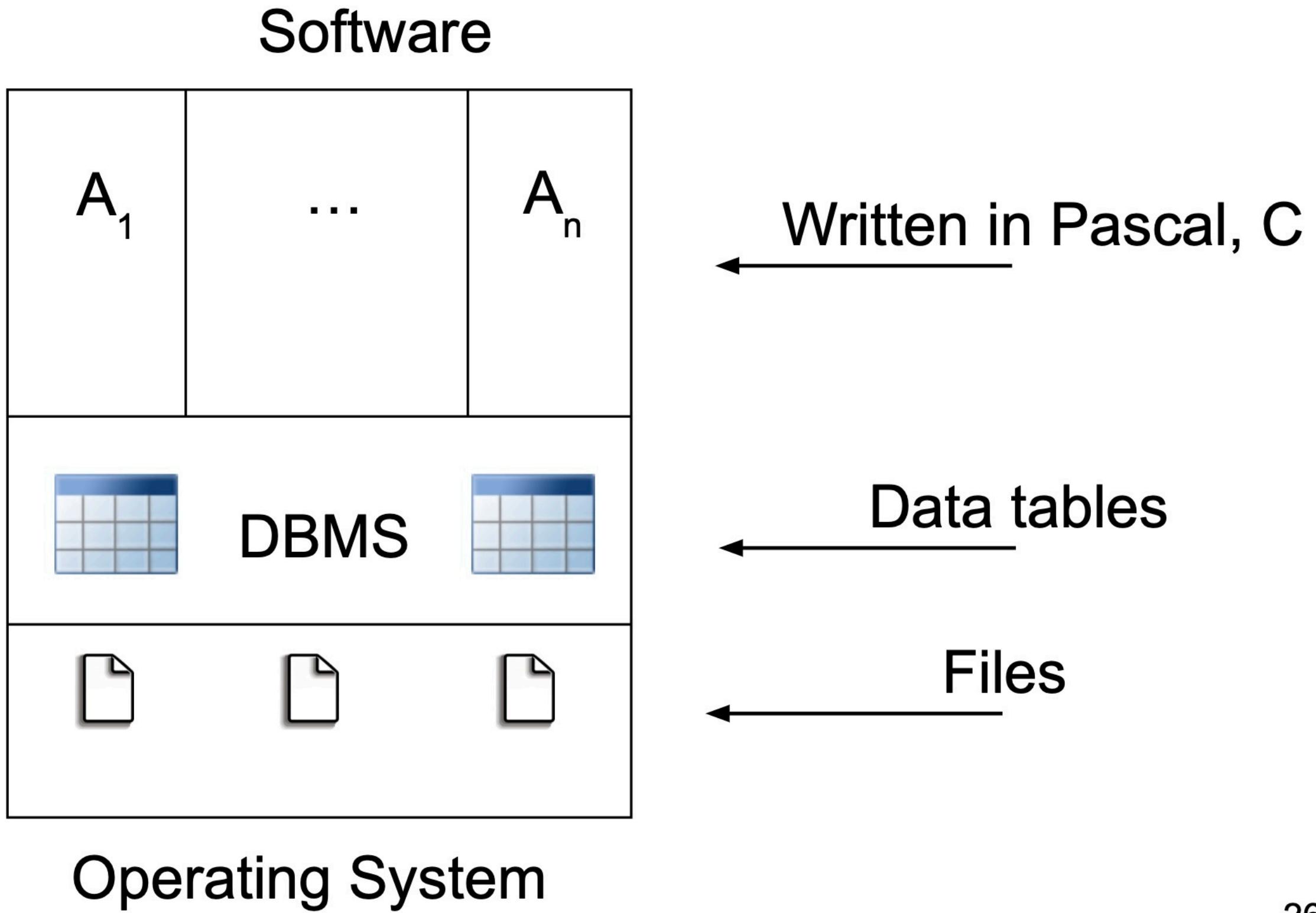


**Let's see how Data Management
evolved through time**

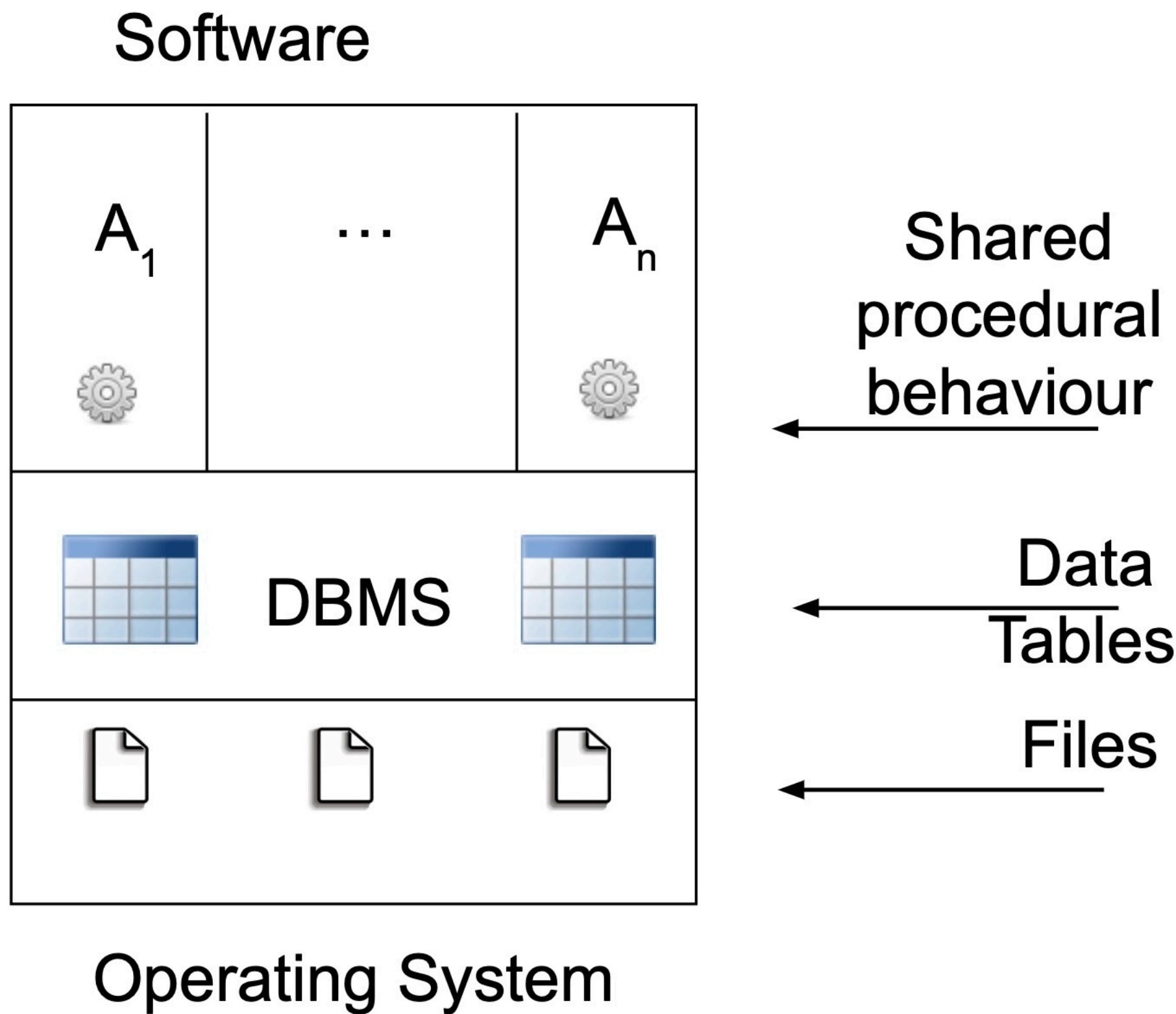
The 70s: no DBMS

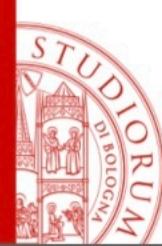


The 80s: first DBMSs



The 90s: the procedural behaviour is shared among the software applications



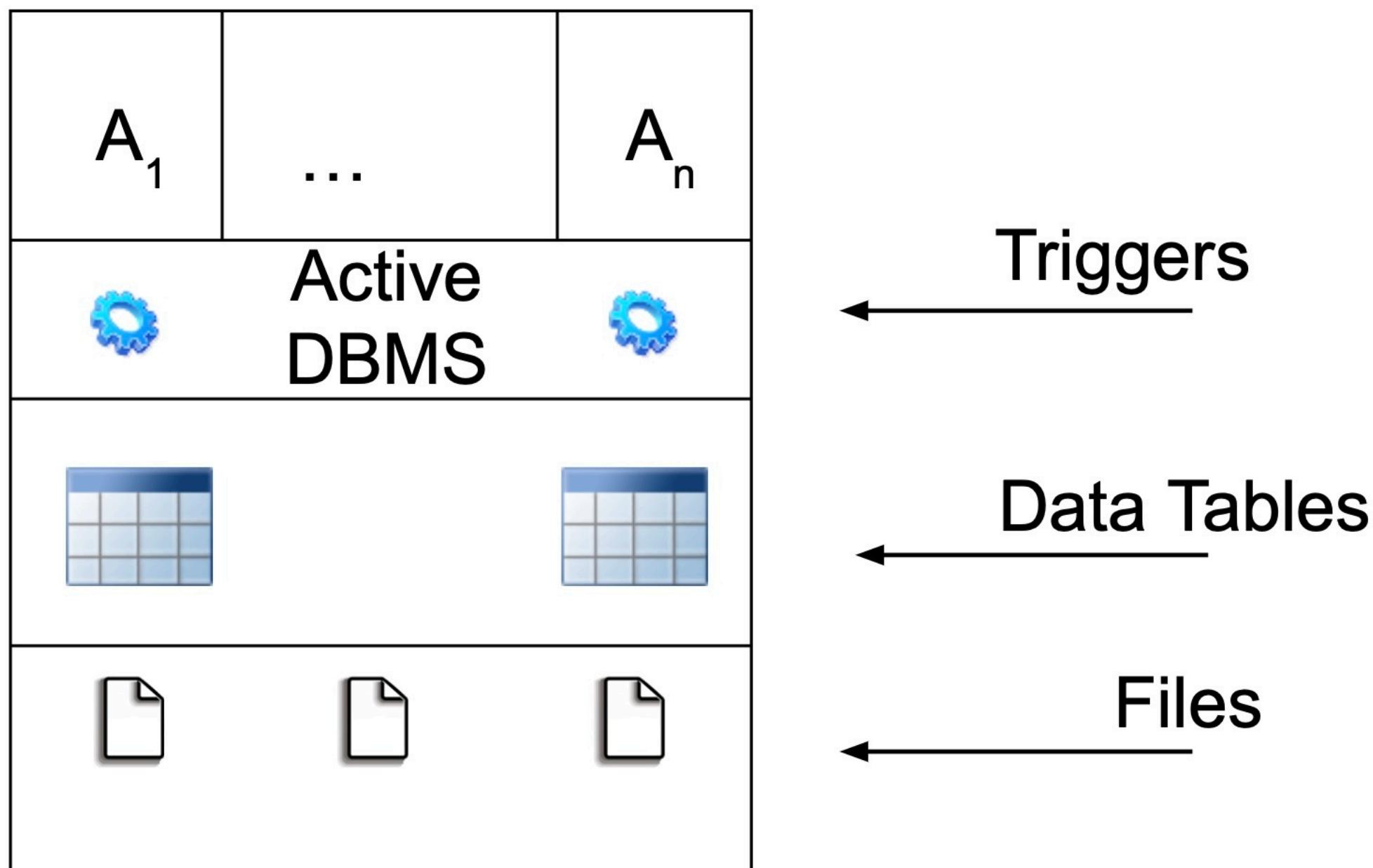


Advances in DBMSs: Stored procedures

- Stored procedures have been introduced to share the common procedural behaviours between different software.
- Stored procedures are not standardized and are affected by the problem of impedance mismatch (we'll see later) with the language used to express such procedures
- As a result specific rules (*triggers*) have been introduced to model the procedural behaviour shared among different software and are handled by the DBMS itself.

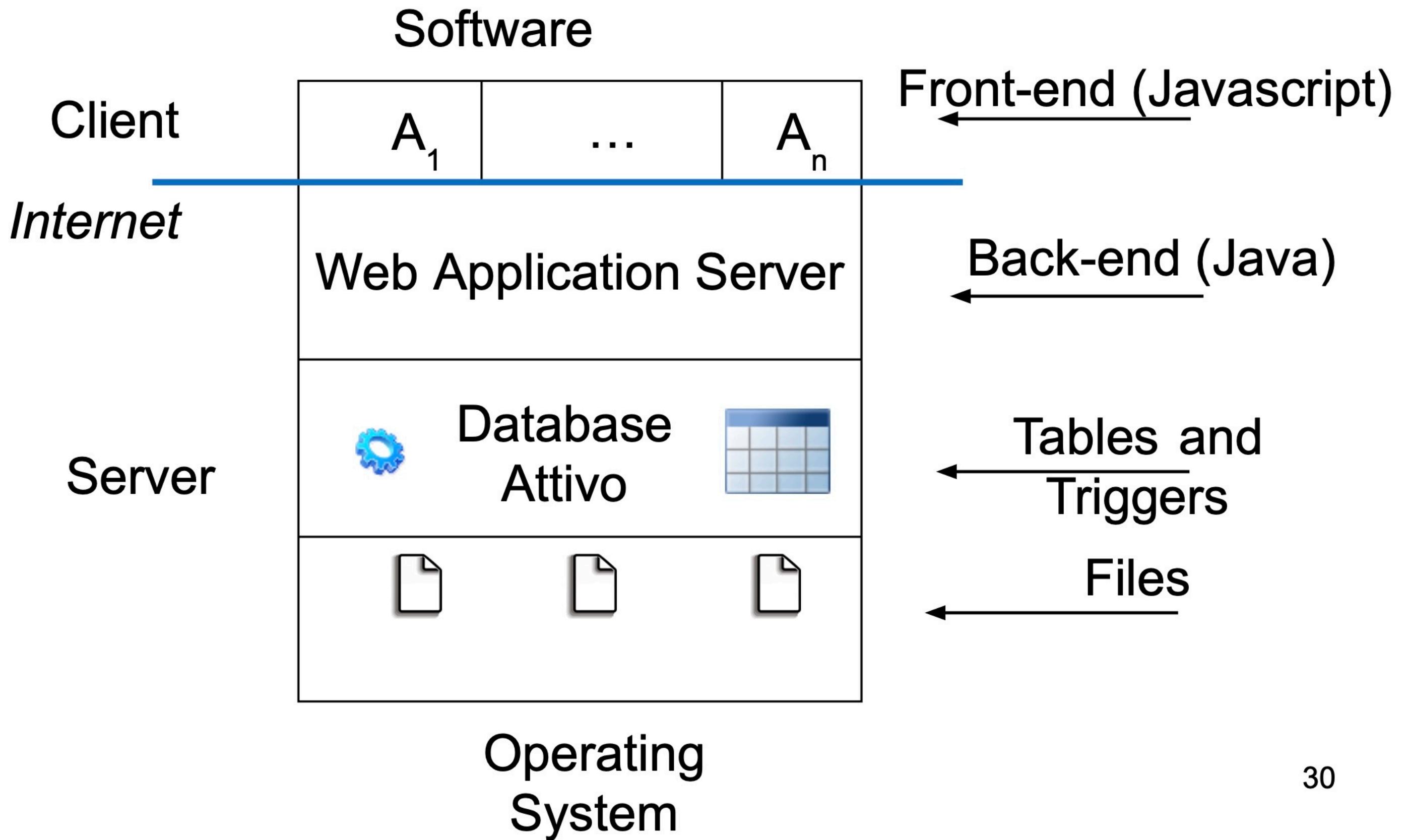
The 90s: Active DBMS

Software

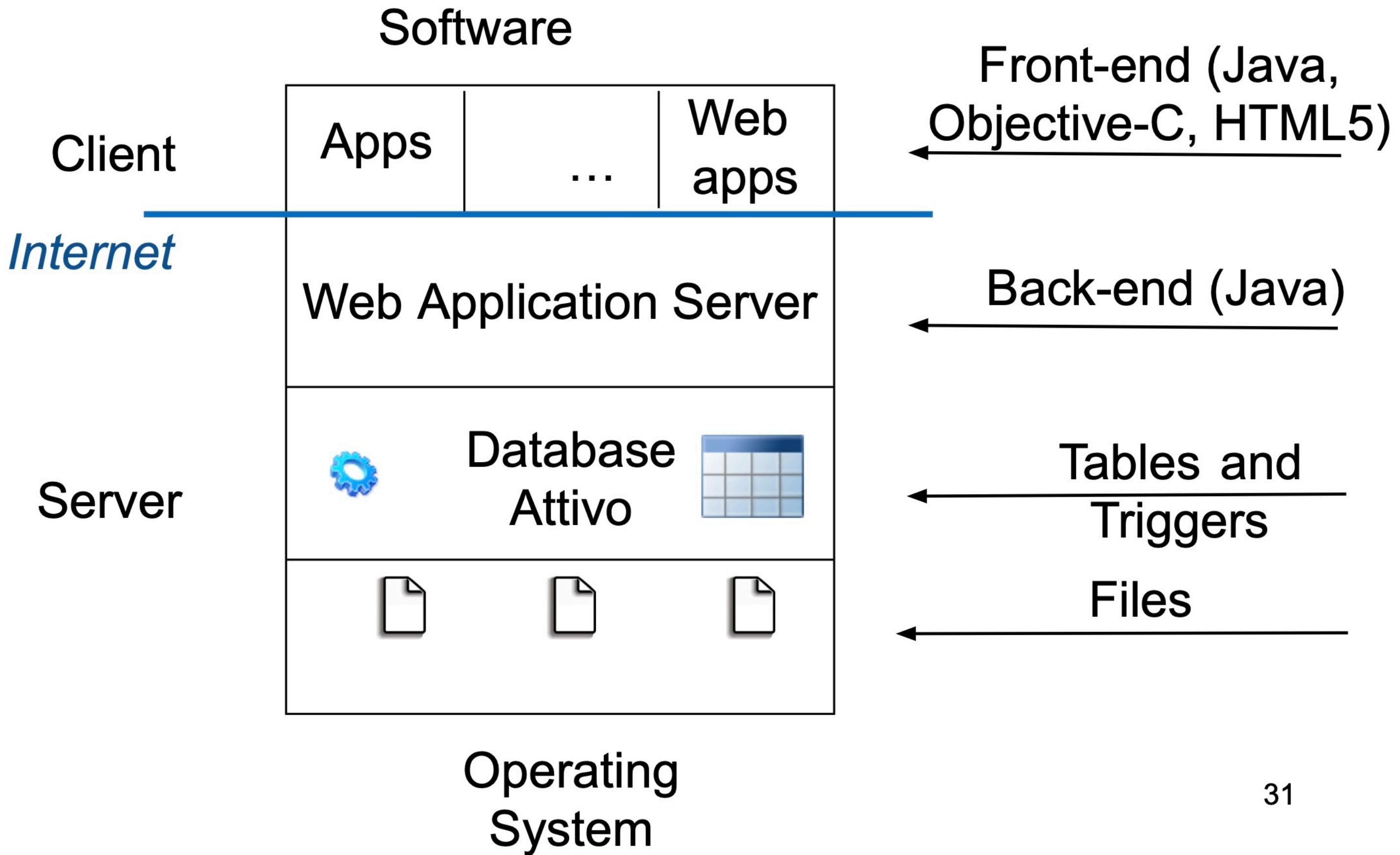


Operating
System

Advance in the 2000s



The 2010s: mobile apps





Databases are ... shared

- A data base is an **integrated** and **shared** resource among the software applications
- As a consequence
 - Different activities build up on shared data:
 - requires **permission mechanisms**
 - More users access shared data:
 - requires **concurrency checks**



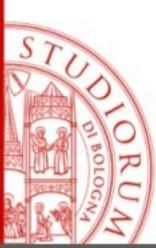
Databases preserve ... privacy

- We could implement permission mechanisms such as:
 - “*User A is allowed to read all the data and to edit data X*”
 - “*User B is allowed to read data X and to edit data Y*”



Databases preserve ... reliability

- Reliability (w.r.t. data bases):
 - are tolerant w.r.t both hardware and software failures
- A database is a precious resource and, consequently, should be preserved for a long time to come
- Crucial technique:
 - Transactions handling



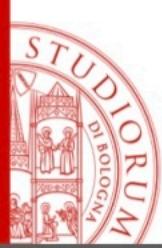
Transaction

- A set of non-detachable operations (*atomic*)...
- that are correct even on a concurrent system...
- ...with final effects



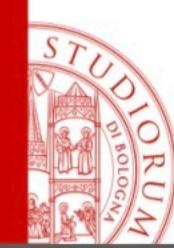
Transactions are ... atomic

- A sequence of related operations:
 - “*Move the money from the bank account A to B: either the withdraw is performed on A and transferred to B or no operation is carried out*”
- ... either is performed as a whole, or is not performed at all:
 - “*either the withdraw is performed on A and transferred to B or no operation is carried out*”



Transactions are ... concurrent

- Concurrent transactions must be coherent.
 - *If two different checks are issued on a same bank account and are cashed contemporarily
... you must not miss one of them*
 - *If two different agencies want to book the same (available) sit on a train
... you must not book it twice*



The effect of transactions are permanent

- The **commit** of a transaction requires that the final result should be logged and tracked, even in a concurrent context or when failures occur.



Databases must be ... efficient

- They try to use profitably the memory (*primary* and *secondary*) and time (of *running* and *response*) resources.
- Since DBMS provide many operations, they could be implemented inefficiently. For this reason there are huge investments and competition.
- Efficiency is also a result of software quality



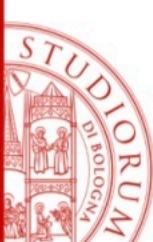
Databases must be ... effective

- Databases try to improve the activities of their users, by providing powerful and flexible functionalities:
 - a frequent topic of the lectures is how DBMSs pursue effectiveness



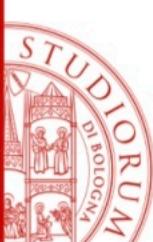
DBMS vs file system

- Big collections of data could be also handled by simpler systems, such as standard Operating Systems' **file system**
- File systems provide a coarse grained access to data: "all or nothing"
- DBMSs extend the file system's features, by providing more services and in a more integrated fashion



Describing data

- Each software (that does not rely on a DBMS) contains an internal description of the file structure that is going to be processed. As a drawback, multiple software could have multiple data description: this situation generates inconsistency within data representations.
- In DBMSs, a portion of the database (the catalogue or dictionary) contains a centralized description of the data: the same description is shared among all the software applications.



Describing data in DBMS

- Data are described at different levels of abstraction
 - Data are not dependent from their physical representation:
 - Programs use a high level representation, independent from the data representations at a lower level. As a consequence, a change in the lower level does not require changes in the programs.
 - This is more precisely obtained through the **data model** concept



Data Model

- A set of constructs that are used to organize and describe the behaviour of the data
- Crucial component: providing structure to data (via type constructor)
- The data model provides some default data type constructors, as it usually happens on current programming languages.
- E.g.: the relational model provides the constructor relation, allowing to define a set of homogeneous records



Organizing data in a database

Scheduling

| Lectures | Lecturer | Room | Time |
|------------|-------------|------|-------|
| Calculus I | Luigi Neri | N1 | 8:00 |
| Databases | Pier Rossi | N2 | 9:45 |
| Chemistry | Nicola Mori | N1 | 9:45 |
| Physics I | Mario Bruni | N1 | 11:45 |
| Physics II | Mario Bruni | N3 | 9:45 |
| IT Systems | Pier Rossi | N3 | 8:00 |



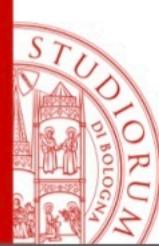
Organizing data in DBMS

A schema

Scheduling

| Lectures | Lecturer | Room | Time |
|------------|-------------|------|-------|
| Calculus I | Luigi Neri | N1 | 8:00 |
| Databases | Pier Rossi | N2 | 9:45 |
| Chemistry | Nicola Mori | N1 | 9:45 |
| Physics I | Mario Bruni | N1 | 11:45 |
| Physics II | Mario Bruni | N3 | 9:45 |
| IT Systems | Pier Rossi | N3 | 8:00 |

Some instances



Schema and instances

- In every database there exist:
 - the **schema**: it is time invariant and describes the data's structure (intensional aspect)
 - e.g.: tables' headers
 - the **instance**: the actual values that could rapidly change (extensional aspect)
 - e.g.: the tables' body



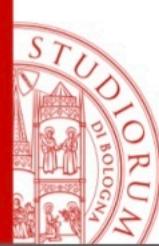
Two different aspects

- Logical Models
- Conceptual Modelling



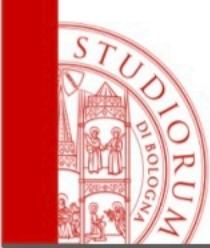
Logical Models

- DBMS use such models to store and organize the data
 - used by software at a higher level
 - independent from the physical representation
- e.g.: relational, lattice, hierarchical, object, XML

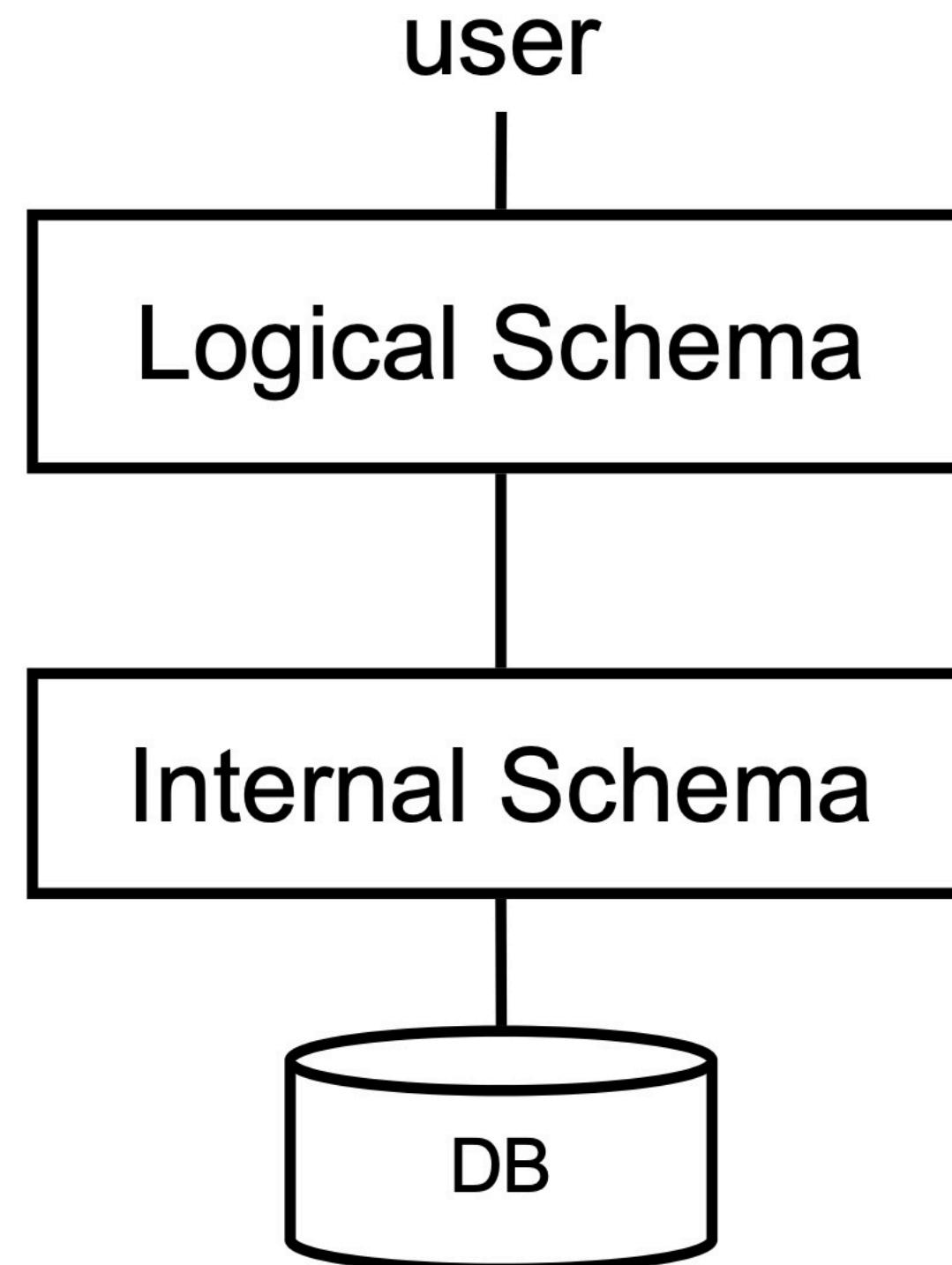


Conceptual Modelling

- They represent data independently from each specific system
 - they describe concepts from the real world
 - they are used in the preparatory phase of a project
- The most diffused data model language is **Entity-Relationship**



Simplified Architecture of a DBMS





Simplified Architecture of a DBMS: schema

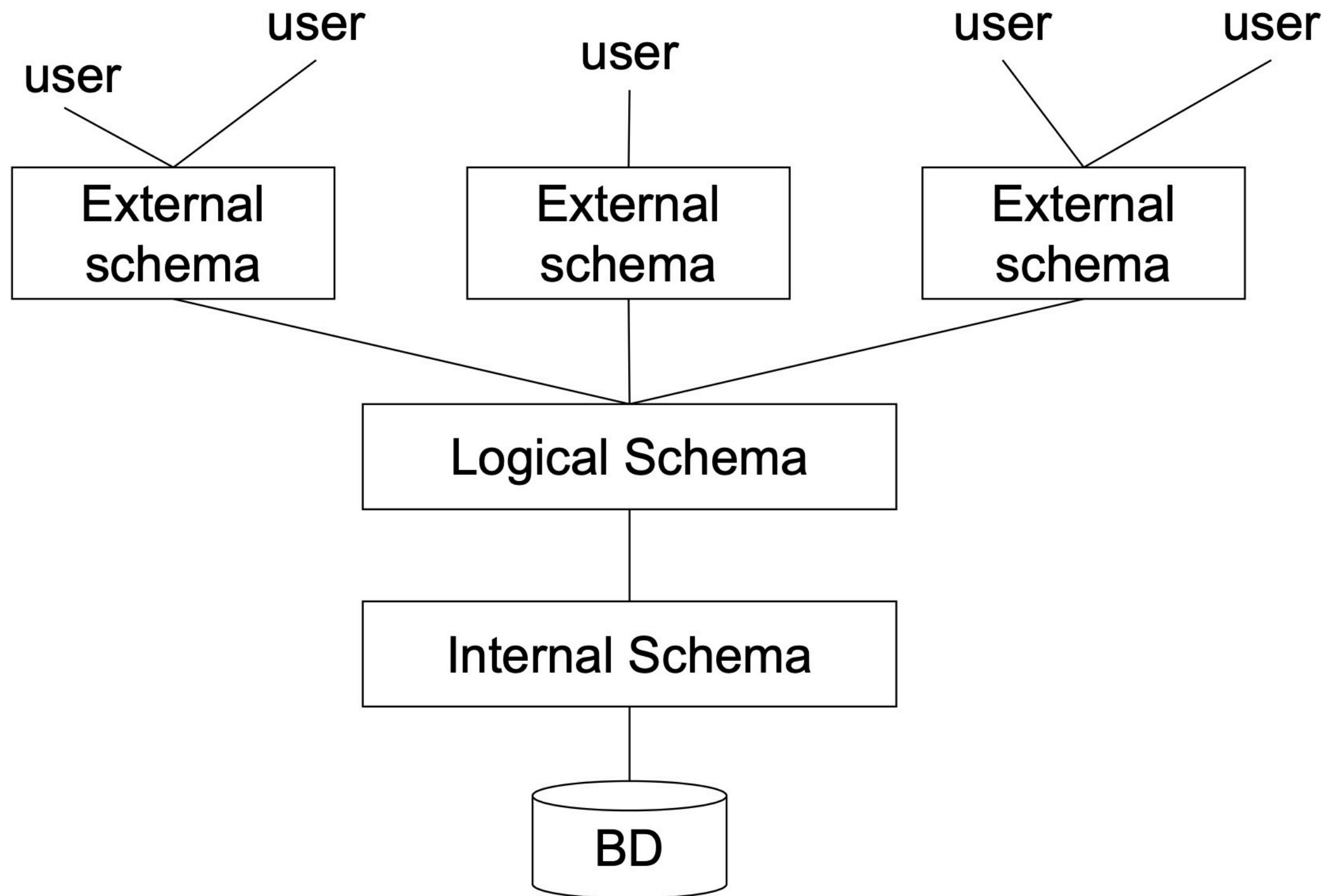
- **Logical schema:** describes the structure of the database through the logical data model (e.g., the table's structure)
- **Internal (or physical) schema:** it represents the logical schema at the storage level using specific *raw* data structures (such as files, records and data pointers)



Data independence

- The data representation at the logical level is independent from the one at the physical level:
 - a table is handled “as it is” independently from its underlying representation (that could even vary over time)
- For this very reason in this course we will see only the logical level and not the internal one.

Standard three-tiered ANSI/SPARC architecture





ANSI/SPARC architecture: schema

- a **logical schema** describes the structure of the database through the “main” logical data model
- an **internal schema** represents the conceptual level at the storage level using specific *raw* data structures (such as files, records and data pointers)
- an **external schema** describes part of the database in a logical model (partial “views”, derived tables, even from different models)



A View

Lectures

| Lecture | Lecturer | Room |
|-----------|----------|------|
| Databases | Rossi | DS3 |
| Systems | Neri | N3 |
| Networks | Bruni | N3 |
| Controls | Bruni | G |

Rooms

| Name | Building | Floor |
|------|-----------|--------|
| DS1 | OMI | Ground |
| N3 | OMI | Ground |
| G | Pincherle | First |

LectureRooms

| Lecture | Room | Building | Floor |
|----------|------|-----------|--------|
| Systems | N3 | OMI | Ground |
| Networks | N3 | OMI | Ground |
| Controls | G | Pincherle | First |



Data independence

- Is the consequence of having different layers
- The data access is provided by the external level (that could coincide with the logical one)
- two types of independence:
 - Physical data independence
 - Logical data independence



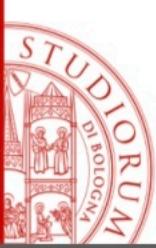
Physical data independence

- the logical and external levels have a representation which is independent from the physical level
 - a relation is “handled” in the same way independently from its physical implementation
 - The physical implementation could undergo some changes without the need of changing the software



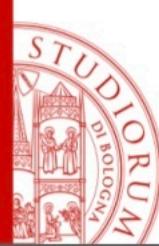
Logical data independence

- The external level is independent from the logical level
- Edits and Updates on “views” do not require any change at the logical level
- “Transparent” edits of the logical schema keep the external schema unaltered



Database languages

- Another aspect of the database effectiveness is given by its availability through different languages and interfaces
 - “interactive” textual languages (**SQL**)
 - statements (SQL) injected in a **host language** (Pascal, Java, C ...)
 - statements (SQL) injected in an *ad hoc* language with other features (e.g. plottings, printing tables)
 - user friendly interfaces (with no textual interface)



SQL, an interactive language

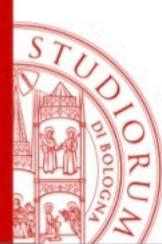
Lectures

| Lecture | Lecturer | Room |
|-----------|----------|------|
| Databases | Rossi | DS3 |
| Systems | Neri | N3 |
| Networks | Bruni | N3 |
| Controls | Bruni | G |

Rooms

| Name | Building | Floor |
|------|-----------|--------|
| DS1 | OMI | Ground |
| N3 | OMI | Ground |
| G | Pincherle | First |

- “Retrieve all the lectures that are held on the ground floor”



SQL, an interactive language

```
SELECT Lecture, Room, Floor  
FROM Rooms, Lectures  
WHERE Name = Room  
AND Floor = 'Ground'
```

| Lecture | Room | Floor |
|----------|------|--------|
| Systems | N3 | Ground |
| networks | N3 | Ground |



SQL injected in a host language

```
write('name of the city?'); readIn(citta);
EXEC SQL DECLARE P CURSOR FOR
  SELECT NAME, INCOME
  FROM PEOPLE
  WHERE CITY = :city ;
EXEC SQL OPEN P ;
EXEC SQL FETCH P INTO :name, :income ;
while SQLCODE = 0 do begin
  write('name of the person:', name, 'rise?');
  readIn(rise);
EXEC SQL UPDATE PEOPLE
  SET INCOME = INCOME + :rise
    WHERE CURRENT OF P
EXEC SQL FETCH P INTO :name, :income
end;
EXEC SQL CLOSE CURSOR P
```



Ad Hoc SQL (Oracle PL/SQL)

```
declare Stip number;
begin
    SELECT SALARY INTO STIP FROM EMPLOYEE
    WHERE NUMBER = '575488' FOR UPDATE OF SALARY;
    if Stip > 30 then
        UPDATE EMPLOYEE SET SALARY = SALARY * 1.1
        WHERE NUMBER = '575488';
    else
        UPDATE EMPLOYEE SET SALARY = SALARY * 1.15
        WHERE NUMBER = '575488';
    end if;
    commit;
exception
    when no_data_found then
        INSERT INTO ERRORS
        VALUES('NUMBER DOES NOT EXIST',SYSDATE);
end;
```



User friendly interface

Receive Shipments

| | | Dates | | | | | |
|-----------|--------------|--------------------------|-------------------------------------|--------|------------|-----------|--|
| Manifest | Shipper | Recv'd | Req'd | Recv'd | Tent. Ship | Firm Ship | |
| 52101HATH | Deaton, Inc. | <input type="checkbox"/> | <input checked="" type="checkbox"/> | 1/1/01 | 1/1/01 | | |

Defaults

| | |
|-----------|------|
| Shipping: | \$1 |
| Markup%: | 20 |
| Dispo: | n/a |
| Rating: | Good |

Manifest Items

| Customer | PO No | Plant | Cost | Qty | UOM |
|---------------------|----------|-------------------------|------|-----|-----------|
| ▶ 2840 Chimney Rock | PO-12345 | Bamboo - Bambusa edulis | \$50 | 10 | 10 Gallon |

Copyright Blue Claw Database Design, LLC

Record: **1** of 1

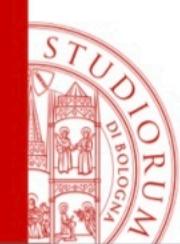
New Inventory Items

| Tag | Avail | Plant | UOM | Rating | In | Out | Cost \$ | Ship \$ | Sell \$ | Dispo |
|-----|-------------------------------------|-------------------------|-----------|--------|----------|-----|---------|---------|---------|-------|
| ▶ | <input checked="" type="checkbox"/> | Bamboo - Bambusa edulis | 10 Gallon | Good | 09/02/01 | | \$50.00 | \$0.00 | \$60.00 | |
| | <input checked="" type="checkbox"/> | Bamboo - Bambusa edulis | 10 Gallon | Good | 09/02/01 | | \$50.00 | \$0.00 | \$60.00 | |
| | <input checked="" type="checkbox"/> | Bamboo - Bambusa edulis | 10 Gallon | Good | 09/02/01 | | \$50.00 | \$0.00 | \$60.00 | |
| | <input checked="" type="checkbox"/> | Bamboo - Bambusa edulis | 10 Gallon | Good | 09/02/01 | | \$50.00 | \$0.00 | \$60.00 | |
| | <input checked="" type="checkbox"/> | Bamboo - Bambusa edulis | 10 Gallon | Good | 09/02/01 | | \$50.00 | \$0.00 | \$60.00 | |
| | <input checked="" type="checkbox"/> | Bamboo - Bambusa edulis | 10 Gallon | Good | 09/02/01 | | \$50.00 | \$0.00 | \$60.00 | |
| | <input checked="" type="checkbox"/> | Bamboo - Bambusa edulis | 10 Gallon | Good | 09/02/01 | | \$50.00 | \$0.00 | \$60.00 | |
| | <input checked="" type="checkbox"/> | Bamboo - Bambusa edulis | 10 Gallon | Good | 09/02/01 | | \$50.00 | \$0.00 | \$60.00 | |

Record: **1** of 10

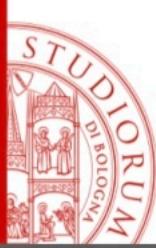
Create New Inventory Items From Manifest **Load Previously Received Items** **Move New Items to Inventory** **Clear New Inventory Items**

Record: **1** of 1



A remark (separating data from software)

- data manipulation language (DML)
query and update (instances of)
databases
- data definition language (DDL)
is used for defining the schemas (logical,
external, physical) and other operations



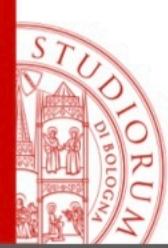
A DDL operation (over the schema)

```
CREATE TABLE hours (
    course CHAR(20) ,
    teacher      CHAR(20) ,
    room        CHAR(4) ,
    hour        CHAR(5) )
```



Actors and Users

- DBMS designers and developers
- database designers and database admins (DBA)
- software designer and developers of end-user applications
- users
 - final users (terminal operators): they execute pre-defined sets of operations (transactions)
 - end/casual user: they execute operations that are not defined a-priori, by using interactive languages



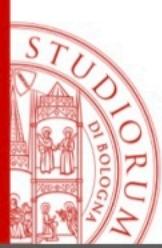
Database administrator (DBA)

- A specific person or a group of people that are in charge of the controlling and handling the database in a centralized way (e.g. efficiency, reliability, permissions)
- A *DBA* often also designs the database, except for some complex projects



Transactions (user's point of view)

- Any software implementing frequent and well known operations, defined a-priori, with very few expected exceptions.
- Examples:
 - paying money into one's account
 - issuing a birth certificate
 - a registration at the registry office
 - flight booking
- The transactions are implemented in a specific host language (either well known or “ad hoc”)



Transactions, two point of views

- The user's:
 - any available software implementing an useful functionality/operation
- The system's:
 - unbreakable sequence of operations
(see **reliability**)



DBMS: pros

Pros

- data as a shared resource, data bases modelling the environment
- standardizable and scaling up centralised data management
- provides integrated services
- reduces data redundancies and inconsistencies
- Independence of data (supporting the development and management of software applications)



DBMS: cons

Cons

- products can be costly as well as adopting such solutions
- functionalities are not detachable (the efficiency is reduced)