Relational databases: usage principles

Introduction to Database Systems

Hiba ALQASIR

2021-2022

télécom
saint-étienne
école d'ingénieurs / nouvelles technologies

Course description

Goals of the course

- Understand, use, create relational databases.

Course organization

- CM: 3 × 2h
- TD: 2 × 3h
- TP: 3 × 3h (per group)
- Individual work: 17h
- Evaluation: Exam 40%, TP 30%, project 30%.
- Material: all slides, TD and TP subjects will be posted on Mootse
  `https://mootse.telecom-st-etienne.fr/course/view.php?id=1077`

Topics

1. Introduction to Database Systems.
2. Mathematical Preliminaries, Relational Data Model.
3. Database Structuring and Querying with SQL.

Tools

We will use the following software

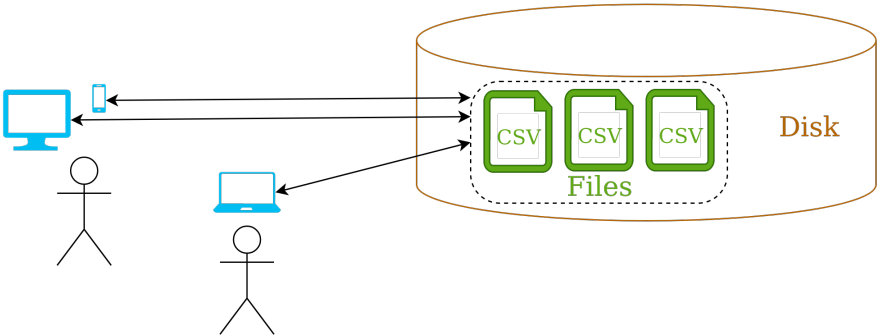
- **MySQL Community Server** as the RBDMS (download).
- **MySQL Workbench** as a graphical client (download).

Database

- A set of organized related data.
- Potentially large, but not necessarily.
- Stored in a persistent manner.
- A database file always has a structure that makes it possible to:
  - distinguish the data from one another,
  - represent their links.

Typical system

Can we build applications based directly on structured files, for example CSV files?



Typical systems challenges

- Heavy access to data – time complexity
- Data privacy and security.
- Concurrency control -access by multiple users.
- Performance: throughput, latency, etc.
- Reliability in the face of hardware crashes, bugs, bad user input, etc.
- Integrity problems – accuracy and consistency.
- Data redundancy and inconsistency – repeated copies.
- Data isolation – changes reflected for all.
- Atomicity problems – everything or nothing.

Database Management System (DBMS)

- DBMS handles the management of data stored in a database.
- Two major functions:
  1. Access to database files
     - guaranteeing their integrity
     - controlling concurrent operations
     - optimizing searches and updates
  2. Interactions with applications and users
     - thanks to query and manipulation languages

Example of a Classic Database Application

Suppose we are building a system to store the information about:

- students
- courses
- professors
- who takes what? who teaches what?

Can we do it without a DBMS ?

Sure, start by storing the data in files
Then write C/Java/JavaScript programs to implement specific tasks.

Students.txt

Courses.txt

Professors.txt

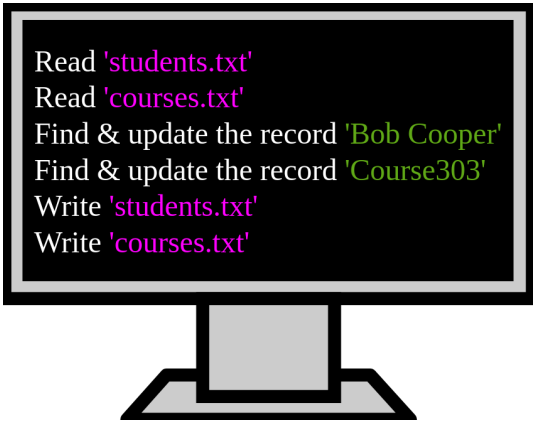
task1.js

task2.java

task3.c

Doing it without a DBMS

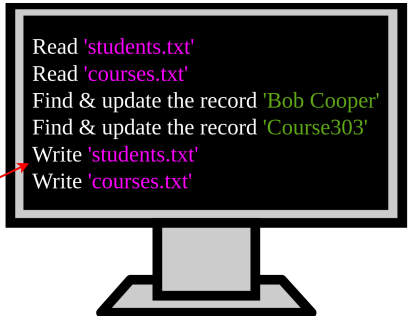
Enroll “Bob Cooper” in “Course303”:
Write a C/Java/JavaScript program to do the following:



Problems without an DBMS

Large data sets

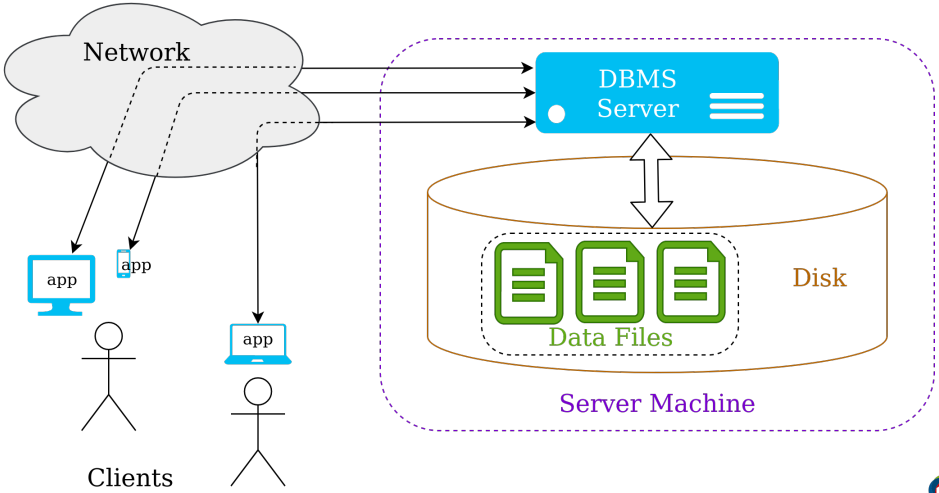
System crashes !!



Students.txt

Simultaneous access by many users

Enters a DMBS



Client server architecture

- Server program (DBMS): manages one or more databases, each consisting of files stored on disk.
  The server program is solely responsible for:
  - all accesses to a database,
  - the use of the resources (memory, disks).
- Client programs (applications): connect to the server program via the network, send it requests and receive data in return.
  They have no direct information about the database.

Panorama of DBMSs

DBMSs have a data model.

- The functions of a DBMS are independent of the data model, but the realization of these functions depends.

Several brands:

- Small: Paradox, MS Access, SQLite
- Medium: dBASE, FoxPro
- Large: Oracle, DB2, SQL Server, Ingres, Informix

Data Models

1. One or more structures to represent information independently of its storage format.
2. One or more languages for querying and interacting with data.

⋆ Examples:

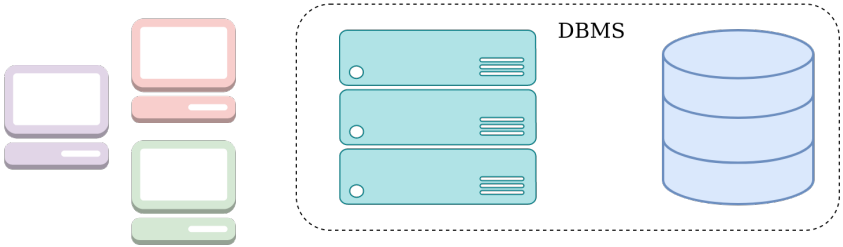
- Relational (Choice for most applications today)
- Hierarchical
- Network
- Object Models
- and many others ...

Data Abstraction

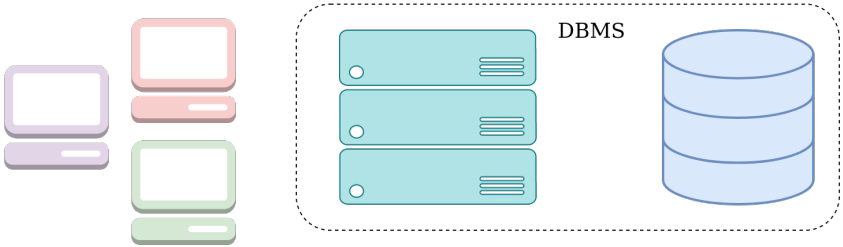
The overall design of the database is called the database schema.

- Physical schema: the data encoding in files stored on disk.
- Logical schema: the representation of data in abstract structures, obtained by conversion of the physical level.
- External views: how end users see the data.

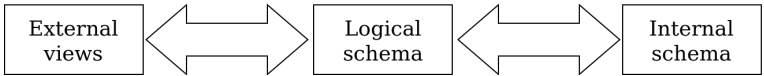
Data Abstraction



Languages



External Level ⟷ Conceptual Level ⟷ Physical Level



Relational Database Management System

- One structure to represent information: Relation (table).
- Languages:
  - Declarative language, based on mathematical logic.
  - Procedural language (algebraic), based on set theory.

SQL

The SQL language has been used since the 1970s in all relational systems, combines the two approaches:

- Data-definition language (DDL):
  for setting up the schema of a database.
  CREATE, DROP, ALTER
- Data-manipulation language (DML):
  to manipulate data in database, also called “query language”
  INSERT, UPDATE, DELETE, SELECT

How the Programmer Sees RDBMS?

- Starts with DDL to create tables:
  CREATE TABLE Students

studentId	name	promotion

- Continues with DML to populate tables:
  INSERT INTO Students

studentId	name	promotion
grumpy2021	Grumpy	2021
dopey2020	Dopey	2020
sneezy2019	Sneezy	2019

Building an Application with RDBMS

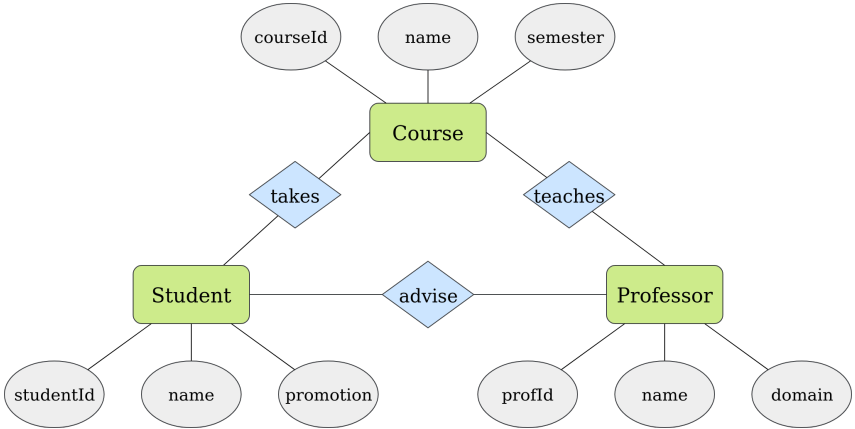
1. Requirements gathering (natural language, pictures)
2. Requirements modeling (conceptual data model)
3. Schema design and implementation
   - Decide on a set of tables, attributes
   - Create the tables in the database system
   - Populate database (insert records/tuples)
4. Write application programs using the RDBMS

Example of a Classic Database Application

Suppose we are building a system to store the information about:

- students
- courses
- professors
- who takes what? who teaches what?

Requirements modeling



Schema design and implementation

studentId	name	promotion
grumpy2021	Grumpy	2021
dopey2020	Dopey	2020
sneezy2019	Sneezy	2019

Students

courseId	name	semester
c11	Fighting sleep	autumn
c23	Combat bad mood	winter
c34	Seasonal sneezing	spring

Courses

studentId	courseId
grumpy2021	c11
grumpy2021	c23
grumpy2021	c34
dopey2020	c34
dopey2020	c11
sneezy2019	c34

Takes

Queries

"Find all courses that Sneezy takes"

```
SELECT   C.name
FROM     Students S, Takes T, Courses C
WHERE    S.name="Sneezy" and
         S.studentId = T.studentId and
         T.courceId = C.courceId
```

We will learn later how to write our query!

Queries

The query processor figures out how to answer the query efficiently.

studentId	name	promotion
grumpy2021	Grumpy	2021
dopey2020	Dopey	2020
sneezy2019	Sneezy	2019

Students

courseId	name	semester
c11	Fighting sleep	autumn
c23	Combat bad mood	winter
c34	Seasonal sneezing	spring

Courses

studentId	courseId
grumpy2021	c11
grumpy2021	c23
grumpy2021	c34
dopey2020	c34
dopey2020	c11
sneezy2019	c34

Takes

Transactions

A transaction = sequence of statements that either all succeed, or all fail

```
BEGIN TRANSACTION;
INSERT INTO Takes
SELECT Students.studentId, Courses.courseId
FROM Students, Courses
WHERE Students.name = 'Sleepy' and
            Courses.name = 'Fighting sleep'
– More updates here....
IF everything-went-OK
THEN COMMIT;
ELSE ROLLBACK
```

Transactions

Transactions have the ACID properties:

- A Atomicity
- C Consistency
- I Isolation
- D Durability

Exercice

Create a database schema allows you to manage an online show ticket sales platform.

- Show
- Room
- Ticket
- Artiste
- Spectator

Installation of a RDBMS

We will use the following software

- **MySQL Community Server** as the RBDMS (download)
  Follow the instructions to install the server and start it.
- **MySQL Workbench** as a graphical client (download)
  Follow the instructions to install the client.
  What are the parameters to connect this client to the server?