# DBMS Intro

Chapter contents:



- Context around DBMSs
- Datawarehouses
- Historical overview
- DBMS fundamentals

- ✎ Understand what are a DB and a DBMS.
- ✎ Recall fundamentals of DBMS development

# Table of contents

2022-2023

# Why Databases (DB)? What are DB/DBMS?

Computer science applications (past&present):

- compute (physics, esp. balistics), simulations...
- manage devices
- communicate
- **store and process data**

## Databases (DB)

*Organized* collection of data. This collection must have a regular structure, the data is meant to capture a subset of the "real" world; the collections are connected to each other and deal with a same subject. The data are organized in a way that facilitates their manipulation (access, updates).

## Database Management System (DBMS)

Dedicated *software* to manage databases. Organizes data storage, handles access to data: queries, updates...

# When are we using a DB?

Enterprise management:

- Accounting
- Supply chain management (inventory...)
- staff management
- orders, reservations

Scientific data (medicine, biology...).

Sensor data

Administrative data (Healthcare, demographics, economy)

Back-end for everyday applications:

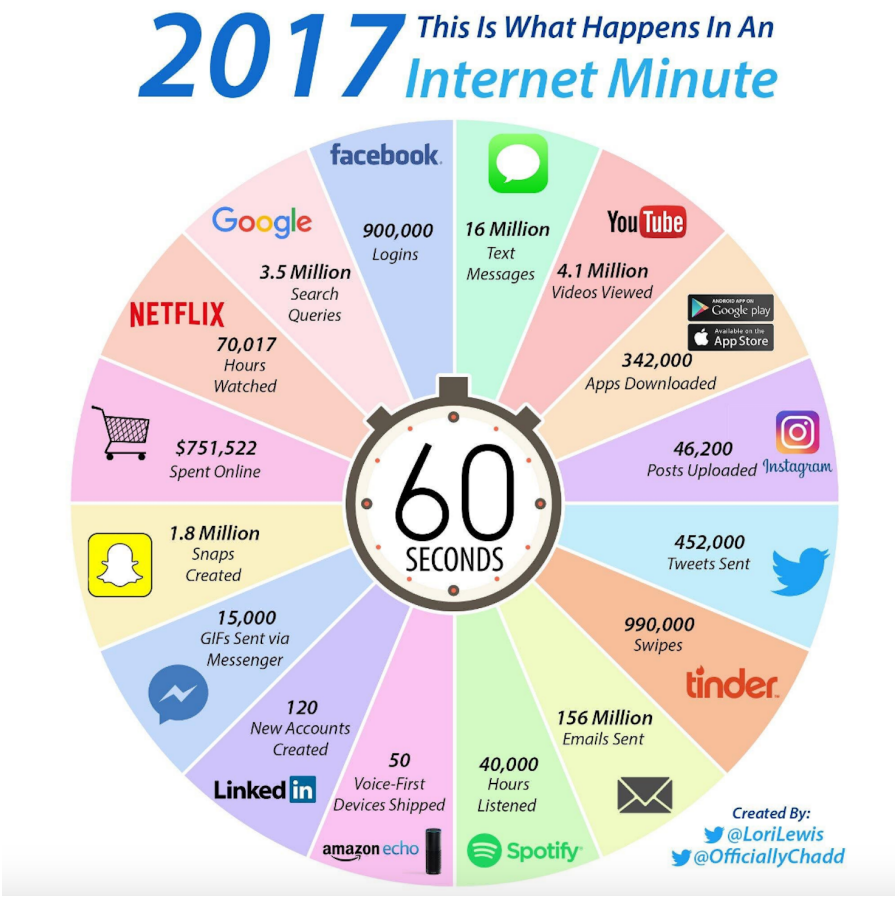
websites: CMS (e.g., *Joomla, Wordpress*)

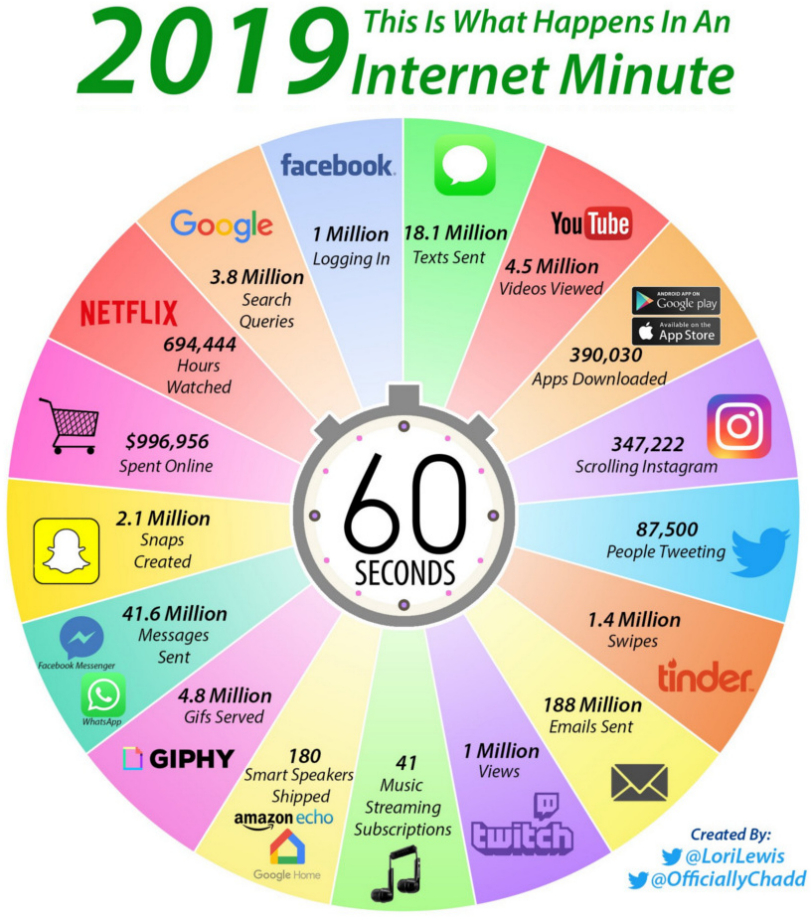
emails/contacts/certificates (Thunderbird, Firefox)

Everyday life use cases:

- Booking hostel, train tickets
- Ordering on e-commerce
- Banking account

# Today's web applications



# DBMS vendors

## Main players fo traditional relational DBMS




open source:

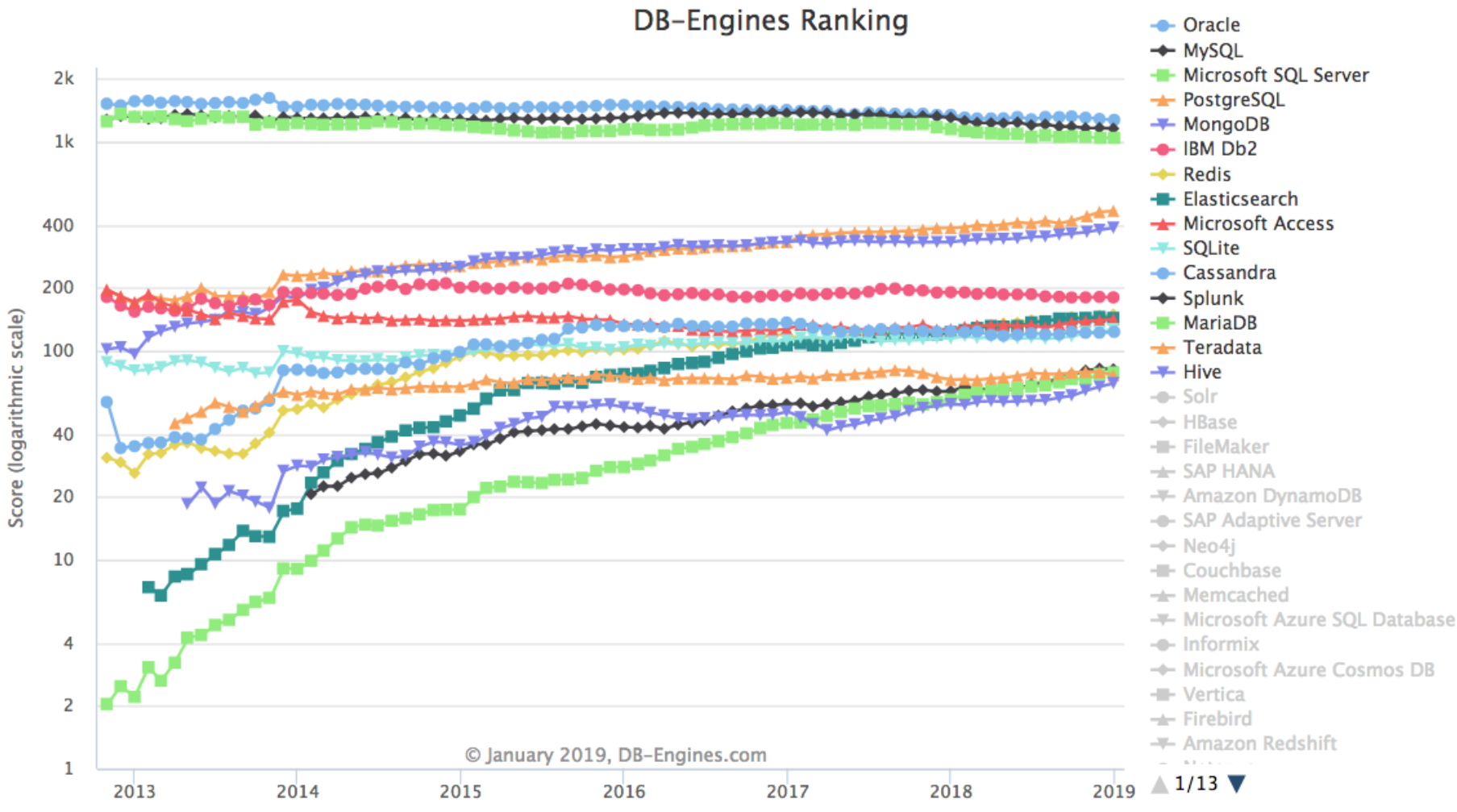





. . . hundreds of systems

# DBMS ranking (“popularity”)

based on mentions in forums, job offers...



# What a DBMS offers (featured in all relational DBMS):

1. *Persistent* storage of data in files: (many GB of data, up to TBs) ***disks, buffer, dictionnaries...***
2. Performance for queries ***index, query optimization...***
3. Multi-user (concurrent access) ***transactions, locks, MVCC...***
4. Security (access control) *access rights...*
5. Easy to use (**queries**, maintenance, evolution)
6. Reliability as a system (recovery after failure...) *logs, backups...*
7. Data integrity (consistency) ***normalization, integrity constraints (FK)***

# When should you *not* use a relational DBMS

**NoSQL** solutions (or plain files) may be a better choice when:

- you have too much data: $> 1TB$
- you need very low latency
- tiny IOT devices with limited memory
- the relational model does not fit your data (graph, time series) and queries.
- your software stack integrates better with other data storage technology

# Table of contents

2022-2023

# What is a DataWarehouse?

widespread feeling that "society is data rich but information poor"

*Business intelligence (BI)*: set of techniques and tools that enable a company to transform business data into into meaningful and useful information for decision making.

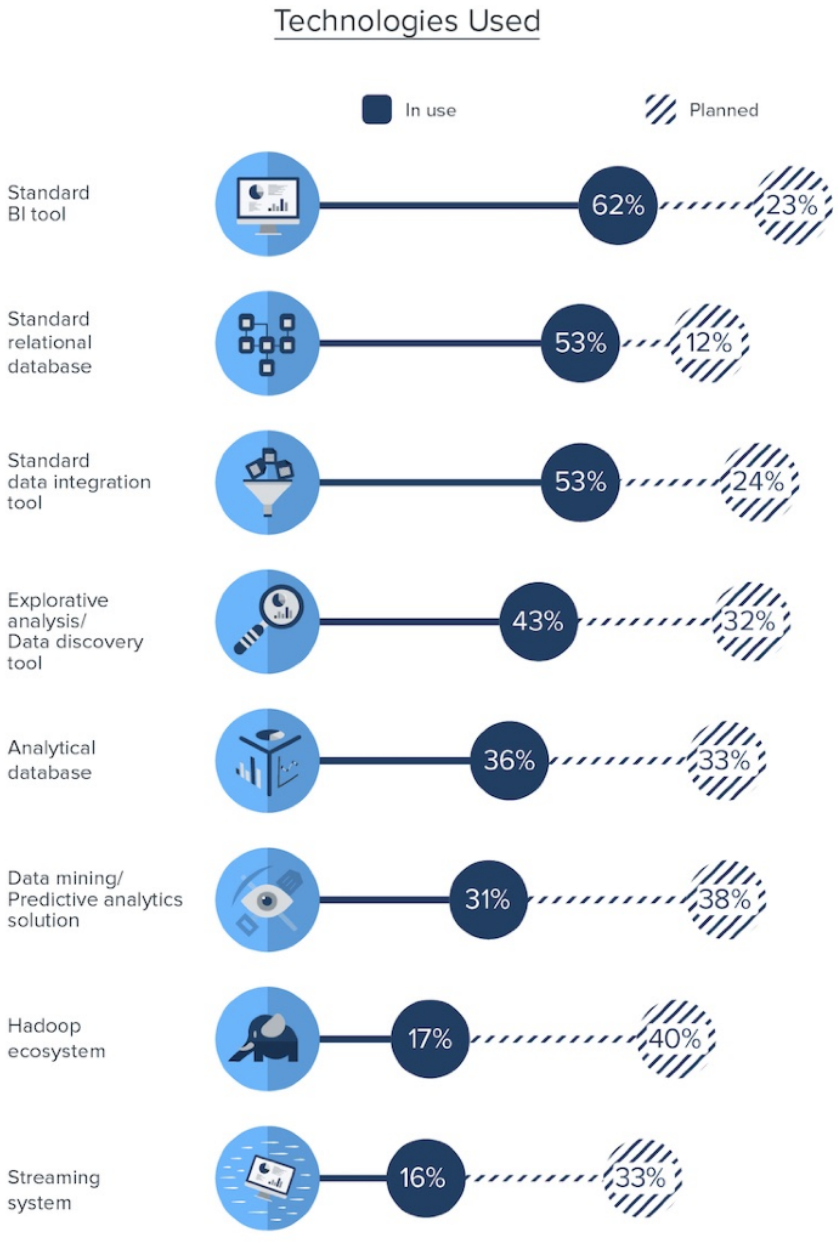
*DataWarehouse (DW):* repository that stores the data and infrastructure to support analysis.

according to R.Kimball:

> *"a copy of transaction data specifically structured for query and analysis"*

cf also Bill Inmon's more precise definition (later).

# Practices are evolving quickly




[http://barc-research.com/big-data-use-cases-2015-infographic/]

# Table of contents

2022-2023

# Collecting and aggregating data: census

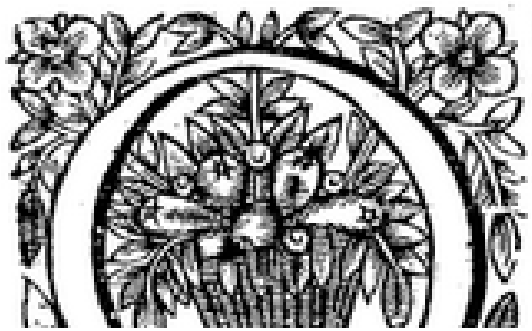
## METHODE
## GENERALLE ET FACILE
### pour faire le dénombrement des Peuples.

N le peut faire par détail d'une maniere fort aiſée qui donnera une connoiſſance parfaite du nombre des Familles, de leur qualité, des lieux, de leurs demeures, & de leur Païs. Pour ce faire il n'y a qu'à ſuivre l'ordre d'une eſpece de formulaire fait en Tables reglées, comme celle qui ſuit, ou le nom, ſurnom, & qualité d'un chacun puiſſent être diſtinguez d'une maniere ſuccinte qui explique tout en peu de mots & ſans confuſion; Il faut pour cét effet prendre du papier un peu grand, le diviſer en autant de colomnes & petits quarrez qu'il ſera beſoin, & en faire un livre au commencement duquel on écrira le nom (du Gouvernement, Province, Prevoſté, & Chaſtellenie) dont on voudra faire le dénombrement, & enſuite

Vauban, 1686

# Collecting and aggregating data: census

## METHODE
## GENERALLE ET FACILE
### pour faire le dénombrement des Peuples.



N le peut faire par détail d'une maniere fort aiſée qui donnera une connoiſſance parfaite du nombre des Familles, de leur qualité, des lieux, de leurs demeures, & de leur Païs. Pour ce faire il n'y a qu'à ſuivre l'ordre d'une eſpece de formulaire fait en

Chaſtellenie de .....

Parroiſſe de S. N. *Ruë Fr.*

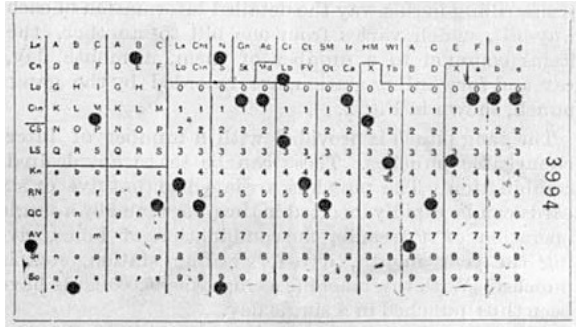
| | Maiſons | Noms & qualitez. | Hommes | Femmes | Grands Garçons | Grandes Filles | Petits Garçons | Petites Filles | Vallets | Servantes | Nomb. des Familles |
|---|---|---|---|---|---|---|---|---|---|---|---|
| | 1 | Mr le Conte de ..... Seigneur du lieu, y reſidant actuellement. | 1 | 1 | 2 | 0 | 0 | 0 | 6 | 2 | 12 |

Vauban, 1686

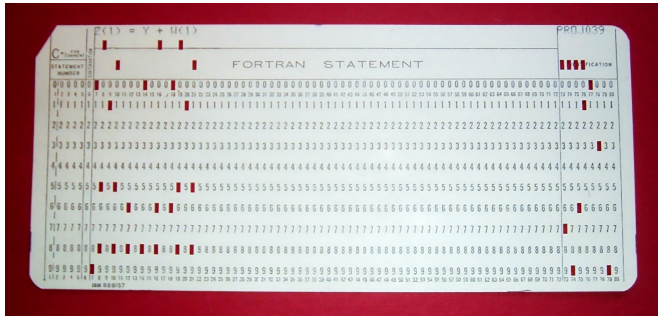
# Automatize census: (Punch cards)

H.Hollerith: mecanography (punch card machines) for 1890 US census.

starts what will become (part of) IBM.



Punch cards: main storage device (data&program) from 1900 to 1950.




80 column card (12lines)

1959 Archive center: 2000 cards per box: total in warehouse $\approxeq$ 4GB.

# Hardware: Storage

## The magnetic drive (HDD)

Introduced by IBM in 1956.

Several disks (platters) spinning fast (7200 rpm), coated with thin (10nm) ferromagnetic layer. Concentric tracks on each disk. 1 read/write head per disk (floats on air cushion).

$\approx$ 100€

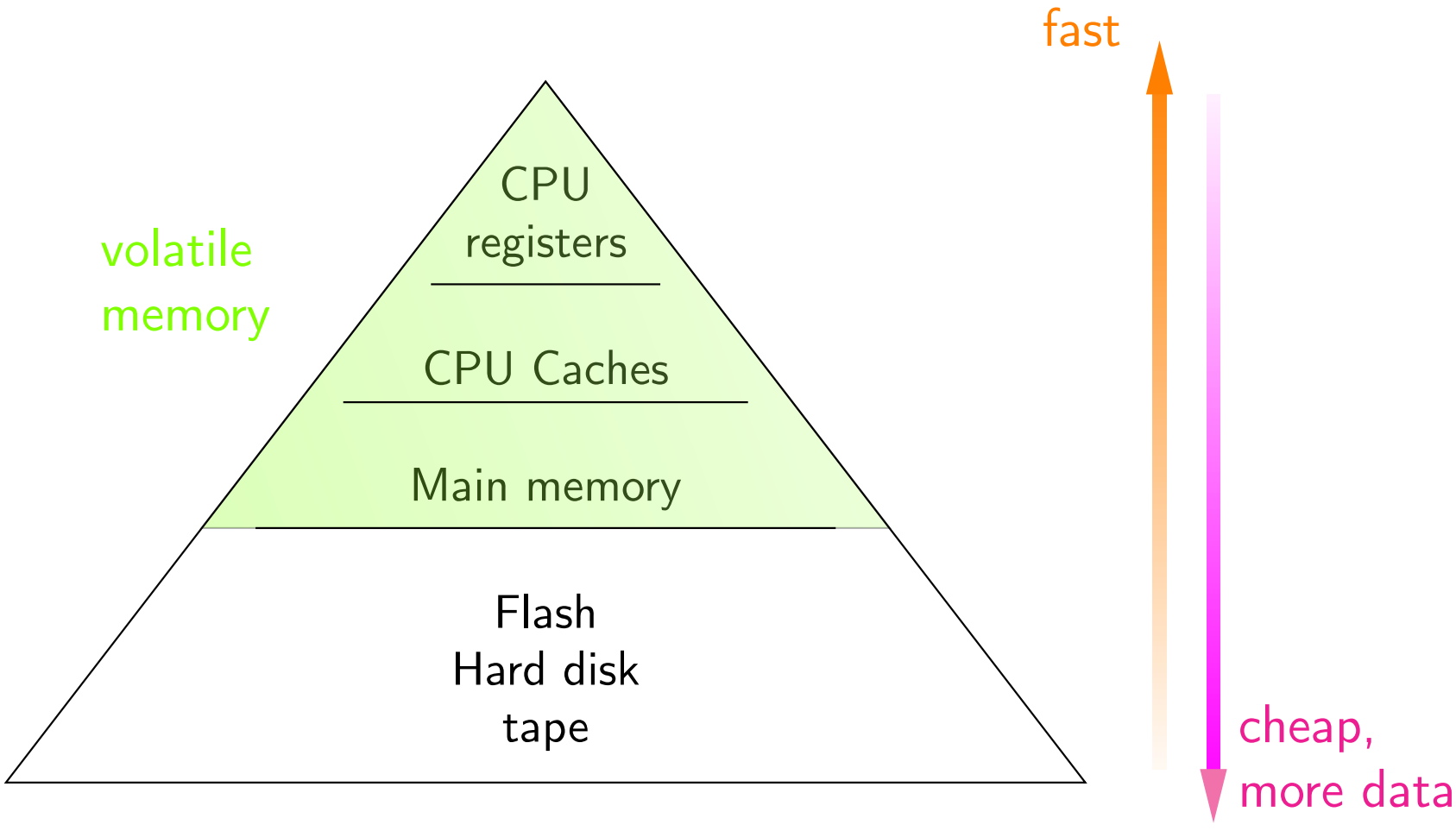
a few TB

100gr.

**Main (secondary) storage device since 1960.**

**The one for which DBMS have been optimized.**

# Hardware: memory hierarchy




On first computers, CPU $\approxeq$ memory access. CPU has become much faster.

**Main memory on PC: 4-16GB.**

# DBMS history

- 1960: First DBMS (accounting, logistics):
  - ⋆ *hierarchical* (IMS, by IBM)
  - ⋆ *network* (IDS for General Electric)

  *Main idea: high performance.*

- 1970: relational model by Codd at IBM San Jose (Almaden). Simple model. Goal: increase productivity. Avoid redundancies/inconsistencies. Theory behind (logics) attracts academics and industry:

  SystemR (IBM,74), Ingres (UC Berkeley,75), Oracle (79)

# DBMS history (2)

- 1980-90: relational model dominates the market. *SQL get richer, new data types (images, text), huge volume (1TB) analytical queries (DataWarehouses). Simplified installation (PC). Prototypes: Object-oriented.*

- 2000: GAFA deploy datacenters, digital economy.
  *Prototypes: XML DB, multimedia. Keyword search, recommending systems. Federated DB: integrate information from multiple (heterogeneous) sources.*

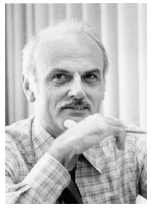
- 2010: Web as a major data source. Clouds storage.
  New hardware (FPGA, GPGPU, SSD).
  Storage/data processing revisited:
  - in-memory DB (columns...)
  - **NoSQL (Not only SQL):** massively parallel architectures (Map/Reduce, key-value systems...).
  - widespread adoption of **AI** and big data technology

# DBMS: Turing awards

Bachman
1973

For his outstanding contributions to database technology

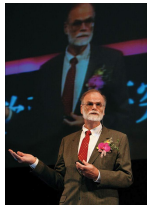
Designed Integrated Data Store(IDS) by 1963. Ideas: store data in single place. Data Manipulation Language. Highly efficient.



Codd
1973

For his fundamental and continuing contributions to the theory and practice of database management systems.

Early parallel programming. relational model: data modeling, normalization, relational algebra, connection to logics. OLAP.



Gray
1998

For seminal contributions to database and transaction processing research and technical leadership in system implementation.

Foundations of transaction processing. GIS. Fault-tolerant DBMS.

Stonebraker
2014

For fundamental contributions to the concepts and practices underlying modern database systems.

INGRES. Postgres. Vertica. VoltDB. SciDB.

# Table of contents

2022-2023

# Data model: the relational model

## The relational model

data represented by tables

- prevailing DB model
- formalized by E.F.Codd (@IBM, Turing'81)
- *1 table= 1 relation*

Ex:

table schema

| NSS | NOM | PRENOM | ADRESSE |
|---|---|---|---|
| 1111 | GROZ | Benoit | 75016 |
| 2222 | COHEN | Sarah | 75008 |
| 3333 | BIDOIT | Nicole | 75014 |

**A Relational Model of Data for Large Shared Data Banks**

E. F. Codd
*IBM Research Laboratory, San Jose, California*

Future users of large data banks must be protected from having to know how the data is organized in the machine (the internal representation). A prompting service which supplies such information is not a satisfactory solution. Activities of users at terminals and most application programs should remain unaffected when the internal representation of data is changed and even when some aspects of the external representation are changed. Changes in data representation will often be needed as a result of changes in query, update, and report traffic and natural growth in the types of stored information.

Existing noninferential, formatted data systems provide users with tree-structured files or slightly more general network models of the data. In Section 1, inadequacies of these models are discussed. A model based on *n*-ary relations, a normal form for data base relations, and the concept of a universal data sublanguage are introduced. In Section 2, certain operations on relations (other than logical inference) are discussed and applied to the problems of redundancy and consistency in the user's model.

Communications of the ACM, 13(6), 1970

# Data model: relational model

## SQL

language used to query and update data in relational model

- introduced in 74, standardized afterward
- successive versions enriched the language (last version: SQL 2016)
- prevailing query language for DBs.
- *declarative* : describes expected output, not computation process

```
SELECT nom,prenom
FROM client
WHERE nom = 'GROZ'
  OR nom = 'BIDOIT';
```

Result is a relation:

```
NOM        PRENOM
---------- ----------
GROZ       Benoit
BIDOIT     Nicole
```

# Schema vs Instance

**Schema**

describes data organization. In relational model: table schema provides the format of each column.

Ex: Clients(id:int, nom:string, prenom:string, adresse:string)

*N.B. this description of data is itself (meta)data, stored in the DB and accessible to user queries.*

**Instance**

the actual data in DB (that must be organized according to the schema).

One can view the instance as the current state of relation (virtual or real), .

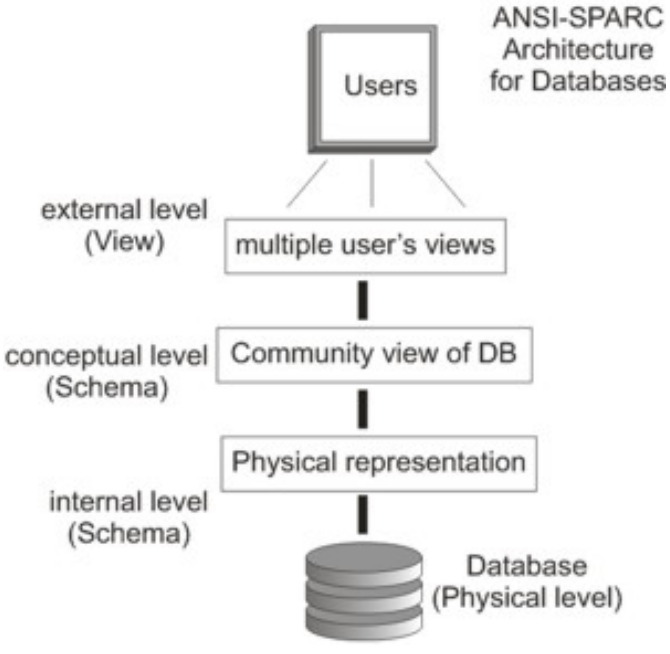
Ex: see on slide 23 an instance of the above DB (Clients).

# 3 abstraction levels in a DBMS

**external:** manages how apps access data; defines what each user sees of the DB.

**logical:** defines the structure of data: the schema

**physical:** defines data storage and organization on disk: files and index




ANSI-SPARC design standard, introduced in 1975, widespread

goal: containing the impact of modifications within a level

# DB Lifecycle.

- Modelization and specifications.
- Schema design, initialization.
- Manipulations (queries, updates)
- Maintenance (optimizations, patches, evolutions)
  tuning can be hard