ICCS240 Database Management

Course Overview

Yodsawalai Chodpathumwan

The Sirindhorn International TGGS Thai-German Graduate School of Engineering

Industry-Oriented Graduate Education and Research in Thailand based on the RWTH Aachen Model

Welcome!

Instructor:

Yodsawalai Chodpathumwan / Som

- Email: yodsawalai.c [at] tggs.kmutnb.ac.th
- Office Hour: (usually) before or after class at 1409, or by appointment

Class hours: 4pm-6pm TR

Canvas: **GH6XJ7**

Goals

- Help you become an expert user of database systems
- Help you learn internals of relational database systems

General Expectation/Prerequisite

- Know basic data structures & algorithms
- Know Shell and basic programming

Grading

- Assignments (2-3) 20%
- Project (1) 20%
- Midterm Exams (2) 30%
- Final Exam 30%

$A: \geq 90, \qquad B: \geq 80, \qquad C: \geq 70, \qquad D: \geq 60, \qquad F: < 60$

Assignment Policy

- Assignment due electronically at 11.59pm Bangkok Time
  - Encouraged to hand them in well ahead of deadline

- You are allowed 2 “LATE DAYS” for the term at no grading penalty
- If you have used up the 2 “LATE DAYS”, your submission will not be graded.

Collaboration Policy

- Collaboration is encouraged in most cases.
- You may work with other students. However, each student must write it up separately. Be sure to indicate who you have worked with.
- Sharing code/write-ups is NOT allowed.
- No collaboration or whatsoever on any exams.

Miscellaneous Policy

- All dates/times are Bangkok time (Indochina Time / ICT)

- **No plagiarism!** (or you receive one letter grade lower each single task caught)
  - For example, your final score is “81”, which is supposed to be in the **B**-ranged, if you are caught plagiarism *on two assignments*, then your final letter grade will be automatically changed from B to **D**.

- Email: subject as “[ICCS240] *…your topic…*”

Any question regarding administrative stuffs?

ICCS240 Database Management

Introduction to DB

Yodsawalai Chodpathumwan

The Sirindhorn International TGGS Thai-German Graduate School of Engineering

Industry-Oriented Graduate Education and Research in Thailand based on the RWTH Aachen Model

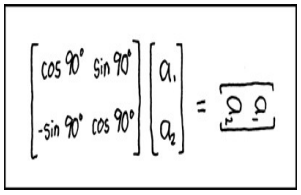
“DATA IS A NEW OIL”

Clive Humby

Data Science Life Cycle

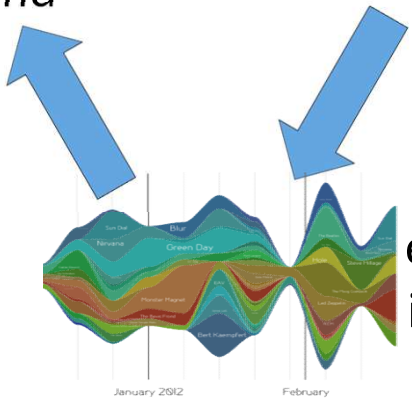
digging around in data

clean, prep



hypothesize model

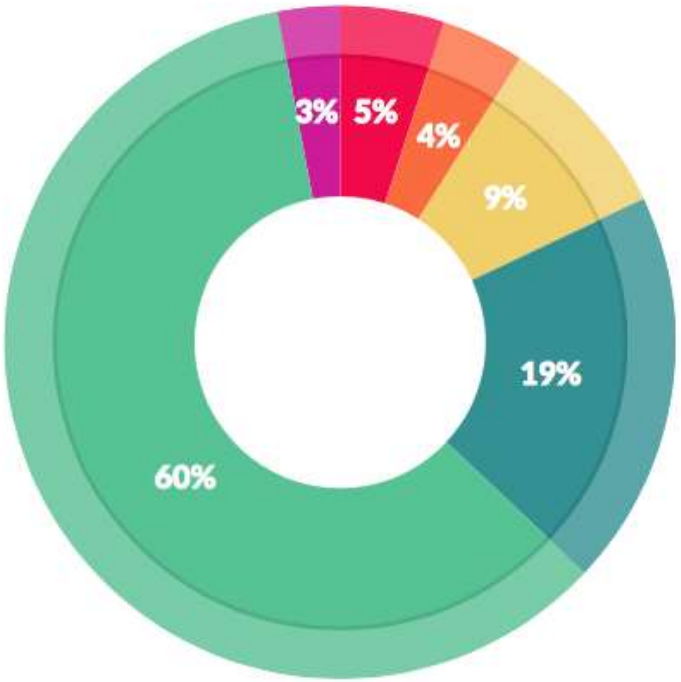
large scale exploitation



evaluate, interpret

Source : T. Rekatsinas, UW-Madison

What data scientists spend time doing?



What data scientists spend the most time doing

- Building training sets: 3%
- Cleaning and organizing data: 60%
- Collecting data sets; 19%
- Mining data for patterns: 9%
- Refining algorithms: 4%
- Other: 5%

This is why we have to learn “data management”

Source : https://visit.figure-eight.com/rs/416-ZBE-142/images/CrowdFlower_DataScienceReport_2016.pdf

Database Systems

What is data?

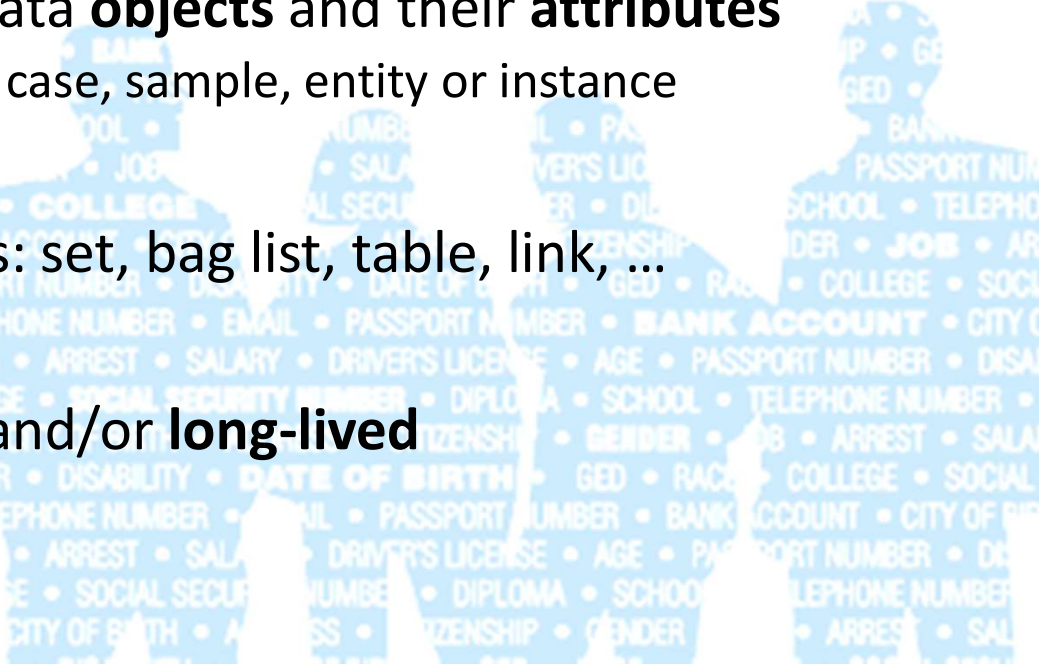
Data represents traces of real-world processes

Data usually refers to collection of data **objects** and their **attributes**

Object is also known as record, point, case, sample, entity or instance

Data may have many different forms: set, bag list, table, link, …

Data is (possibly) **mutable**, **shared**, and/or **long-lived**



Database is essentially a collection of information

The term “database” refers to a collection of **related** data

By **related**, we usually mean the different entities in a database have known relationships.

Entities: University (students, professors, books, courses)

Relationships: Student X is enrolled in course Y taught by professor Z

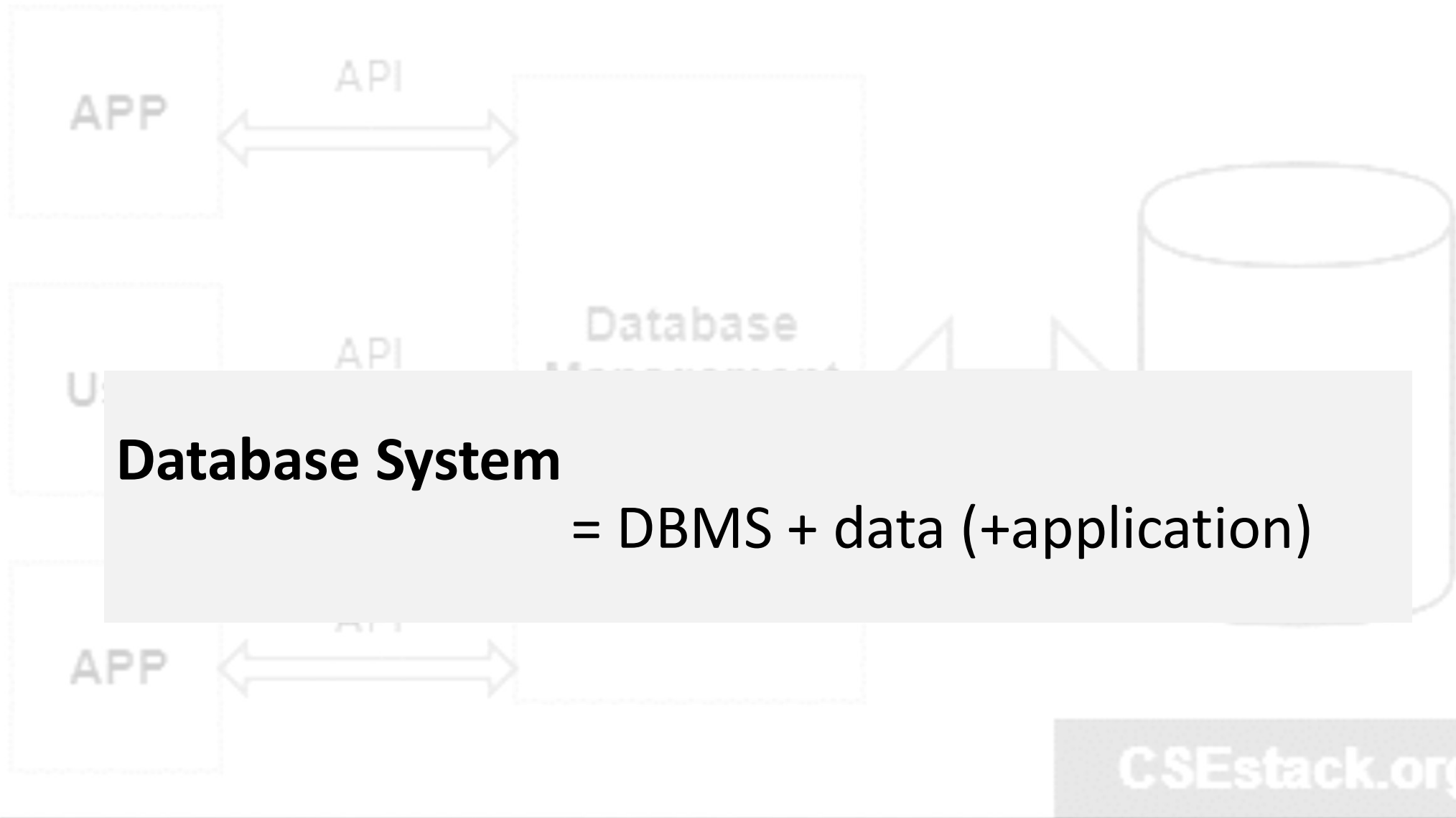
Database Management System

Operations with databases include …

- **Design**
  - *Define* structure and types of data
- **Construction**
  - *Create* structure, *populate* with data
- **Manipulation**
  - *Insert*, *delete*, *update*
  - *Query*: “How much is the tuition fee of graduate student in TGGS?”
  - Create *report*s: “List monthly salaries of employees, organized by department, with average salary for each department.”

A **Database Management System** (DBMS) is a piece of software designed to store and manage databases

Primary Goal of DBMS: **Convenient** & **Efficient**



Database System

= DBMS + data (+application)

Why study databases?

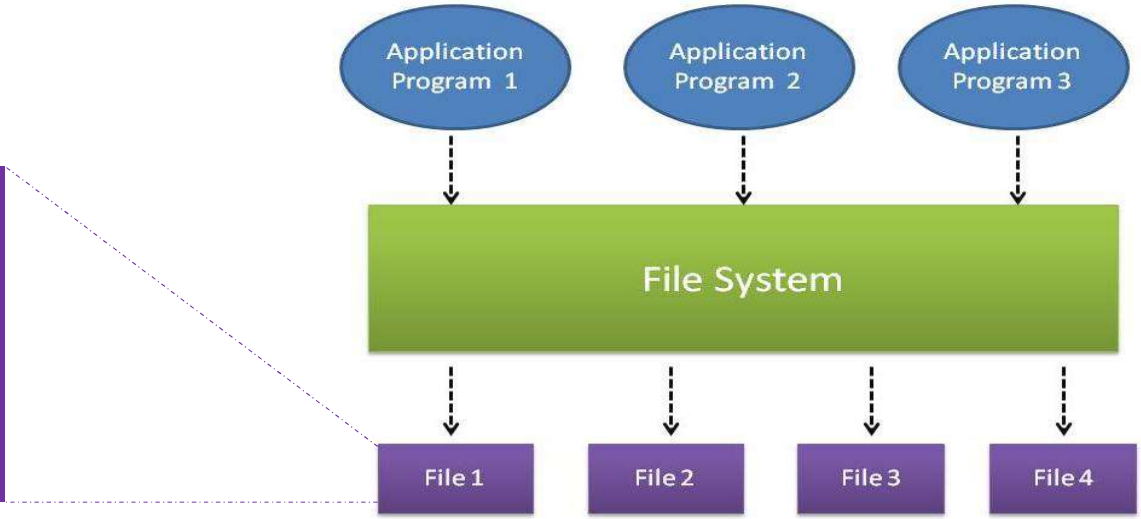
- Focus on **Information** rather than Computation
- DBMS is practically (almost) a center of many applications
- Money

The Motivation of DB

File-based System – *the predecessor*

- **Data is stored in files.**
- Each organization has its own files and each file has a specific set of program that were used to manipulate data in that file

```
[Name,        Course, Grade]
John Smith,   CS22,   B
Mike Stone,   CS123,  A
Jim Gray,     CS127,  A
John Smith,   CS144,  B+
....
```



Limitation of File-based System : (1)

Data Redundancy & Inconsistency

If each application owns its own set of files, this could lead to multiple file formats and duplication of information in different places.

```
CS240-App (Name, Email, Grade)
John Smith, js@muic.io, A+
Jim Gray, jg@microsoft.com, A
```

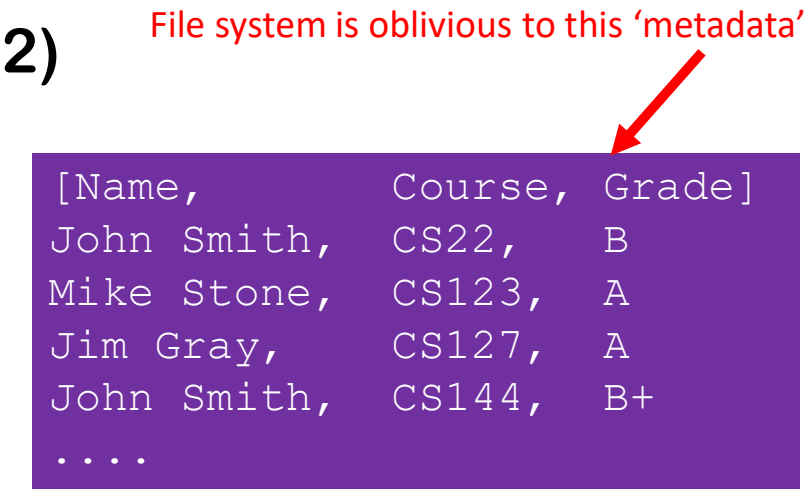
```
CS101-App (Name, Email, Grade)
Mike Gray, mgray@gmail.com, B
J. Smith, js@gmail.com, B+
```

Issue: wasted space; potential inconsistencies

Limitation of File-based System : (2)

Retrieval Problems

File system is oblivious to this 'metadata'



How do we answer question (query)
or update the dataset?

Answer: We'll have to scan and parse the file on every access.

Ideally, data retrieval (a.k.a. query) should be **easy to write** and **efficiently executed**.

More problems : (3)

Data Integrity & Access Control

Without DB, it is generally difficult to

- Share data across applications or even different instances of the same application — because of potential conflicts from simultaneous updates.
- Cope with system crashes
- Prevent simple data-entry errors – checks must be hard-coded into every program
- Security – who check who can do what?

More problems : (4)

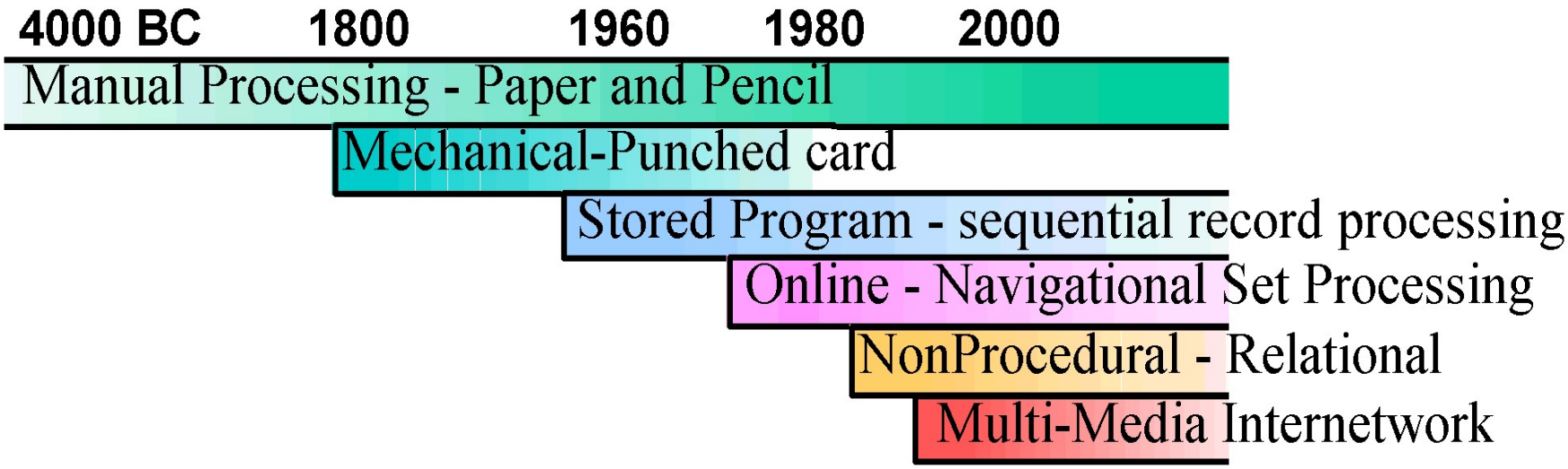
Evolution of Data

We expect data to live long → May change how the data is stored.

e.g., (1) changes in access, (2) tuning

Goal: Avoid rewriting all application. DB should hide (1) and (2) from user.

Evolution of Data Management System



Jim Gray: *Evolution of Data Management*. IEEE Computer 29(10): 38-46 (1996)

View of Data

Data Model

A framework for describing data objects, data relationships, data semantics, and consistency constraints.

Example of proposed data models:

- **Relational Model** uses a collection of tables to represent data and relationships amongst the data.
- **Entity-Relationship (ER) Model** uses a collection of basic objects (entities) and relationships amongst the object. *(It is widely used in database design.)*
- **Object-Based Data Model** (basically) extend the ER model with notions of encapsulation, methods, and object identity.

Data Model & Schema

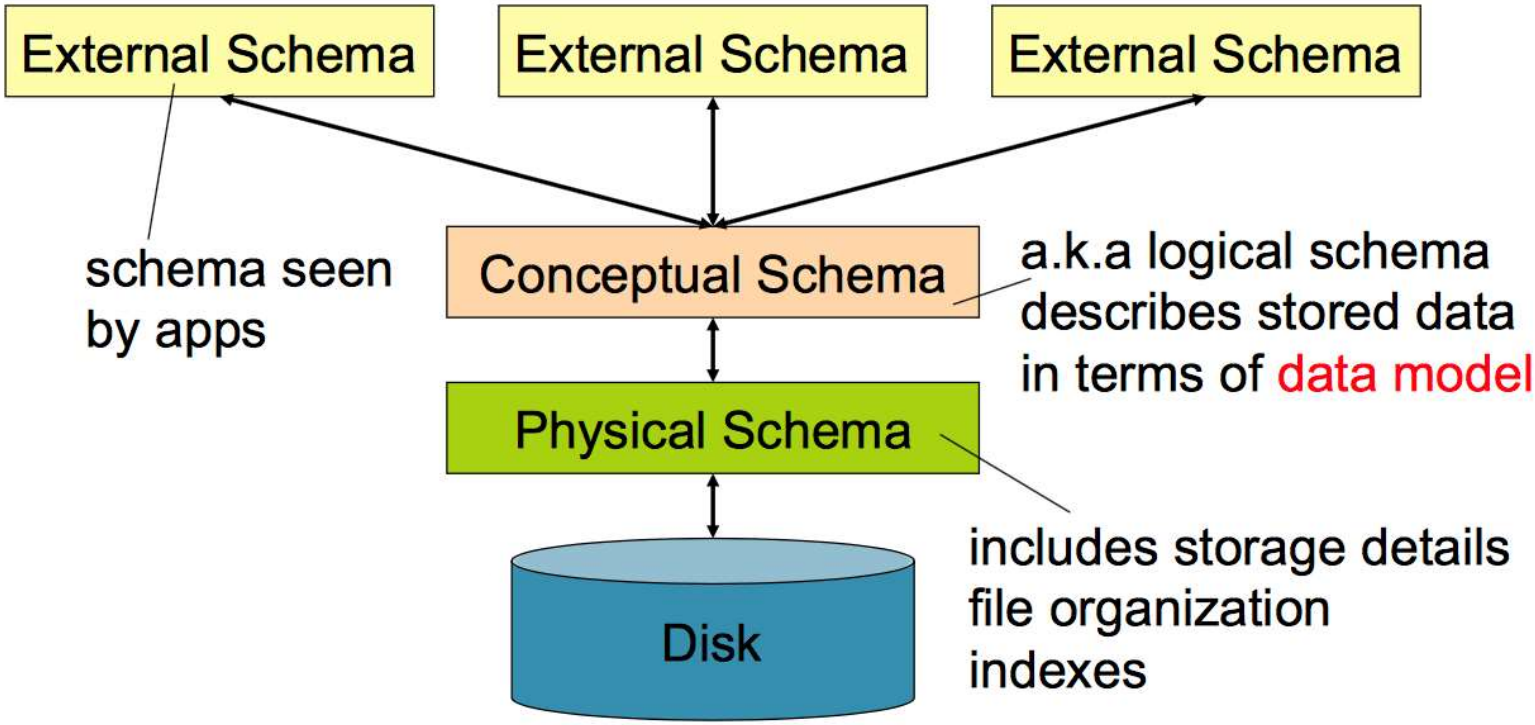
- A **data model** is a collection of concepts for describing data
- A **schema** is a description of a particular collection of data, **using the given data model**
- A **data model** enables users to define the data using high-level constructs without worrying about many low-level details of how data will be stored on disk.

Model : Data in tuples, grouped in relations

Schema : Student(id, name, address)

Instance : Student(`1234`, `Alice`, `USA`)

Levels of abstraction



ANSI/SPARC architecture for DBMSs (1978)

Data Retrieval

Declarative vs. Imperative

Queries are expressed in terms of **declarative** data retrieval.

= user describe ***what*** data to bring up, not ***how*** to bring it up.

We want to write **"List all students with GPA 3.0 or more"**

Instead of writing "Allocate an empty list; Go through student records; For each, compute student GPA using this formula; if the computed GPA is at least 3.0, add the student to the list;"

Benefit of Declarative Approach for DBMS

- Tends to be easier to write
- Generally more efficient to execute – *why?*
- More amenable to change (we don't touch the physical representation)

One of the most widely use data retrieval language is **SQL**

```
Select customer.customer-name
from   customer
where  customer.customer-id = '192-83-7465'
```

Declarative query can be more efficient to execute

Query → [Query Optimizer] → [Query Evaluator]

- When a query is received, it is passed to the query optimizer (compiler for queries).
- Optimizer generates the **best** execution plan for the query.

DB is not a solution to everything

When not to use a DB?

There are situations where not using a DBMS is more prudent choice.

- Simple, Fixed Application: Simple well-defined application that are not expected to change.
- Stringent, Real-time needs: Especially when DB might not meet the requirement.
- Embedded Devices: Standard DB is too large to fit.

To conclude this lecture …

What will we learn in this class?

- Foundation of Databases
- Database Administration
- Data Schema & Models (mostly Relational)
- Behind Database Engines
- Database Design Theory of Relational Databases
- Other Solutions than Relational Databases
- …