CompSci 516
Database Systems

Lecture 1
Introduction
and
SQL

Instructor: Sudeepa Roy

Course Website

- http://www.cs.duke.edu/courses/spring22/compsci516/

- Please check frequently for updates!

Instructor

- Sudeepa Roy
  - sudeepa@cs.duke.edu
  - https://users.cs.duke.edu/~sudeepa/

- About myself
  - Assistant Professor in CS
  - PhD: UPenn, Postdoc: Univ. of Washington
  - Joined Duke CS in Fall 2015
  - Research interests:
    - Data Analysis, causality, query optimization, data science, database theory, applications of data, uncertain data,…

Three TAs

- Yuxi Liu (grad TA)

- Shweta Patwa (grad TA)

- Joon Young Lee (UTA)

What are the goals of this course?

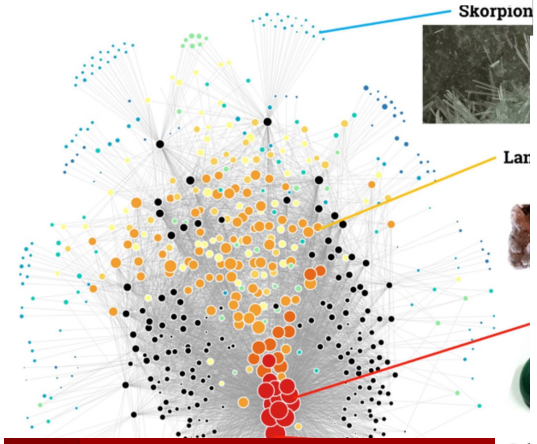
- Learn about “Database Systems” or Data Management in general

Why do we care about data? (easy)

How big data can help find new mineral deposits

Valentina Ruiz Leotaud | Aug. 2, 2018, 4:11 PM |

PEOPLEMINE | FACEBOOK | LINKEDIN | TWITTER | EMAIL | PRINT



POLITICO

Cambridge Analytica whistleblower Chris Wylie speaks during a press conference at the Frontline Club on March 26, 2018 in London | Dan Kitwood/Getty Images

Cambridge Analytica helped 'cheat' Brexit vote and US election, claims whistleblower

Giving evidence to MPs, Chris Wylie claimed the company's actions during the Brexit campaign were 'a breach of the law.'

By MARK SCOTT | 3/27/18, 5:46 PM CET | Updated 3/29/18, 9:18 PM CET

The New York Times

When Sports Betting Is Legal, the Value of Game Data Soars

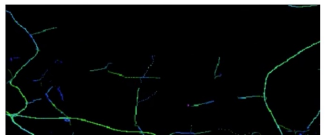
A trader working at William Hill, an international sports betting book, in Las Vegas.

Bridget Bennett for The New York Times

Sections | The Washington Post | Try 1 month for $1 | Sign In

Transportation

Researchers are trying to teach computers to forecast traffic like the weather



… The three years of gathering and analyzing data culminated in what U.S. Sailing calls their "Rio Weather Playbook," a body of critical information about each of the seven courses only available to the U.S. team…

— FiveThirtyEight, "**Will Data Help U.S. Sailing Get Back On The Olympic Podium?**" Aug 15, 2016

Data =

Money

Information

Power

Fun

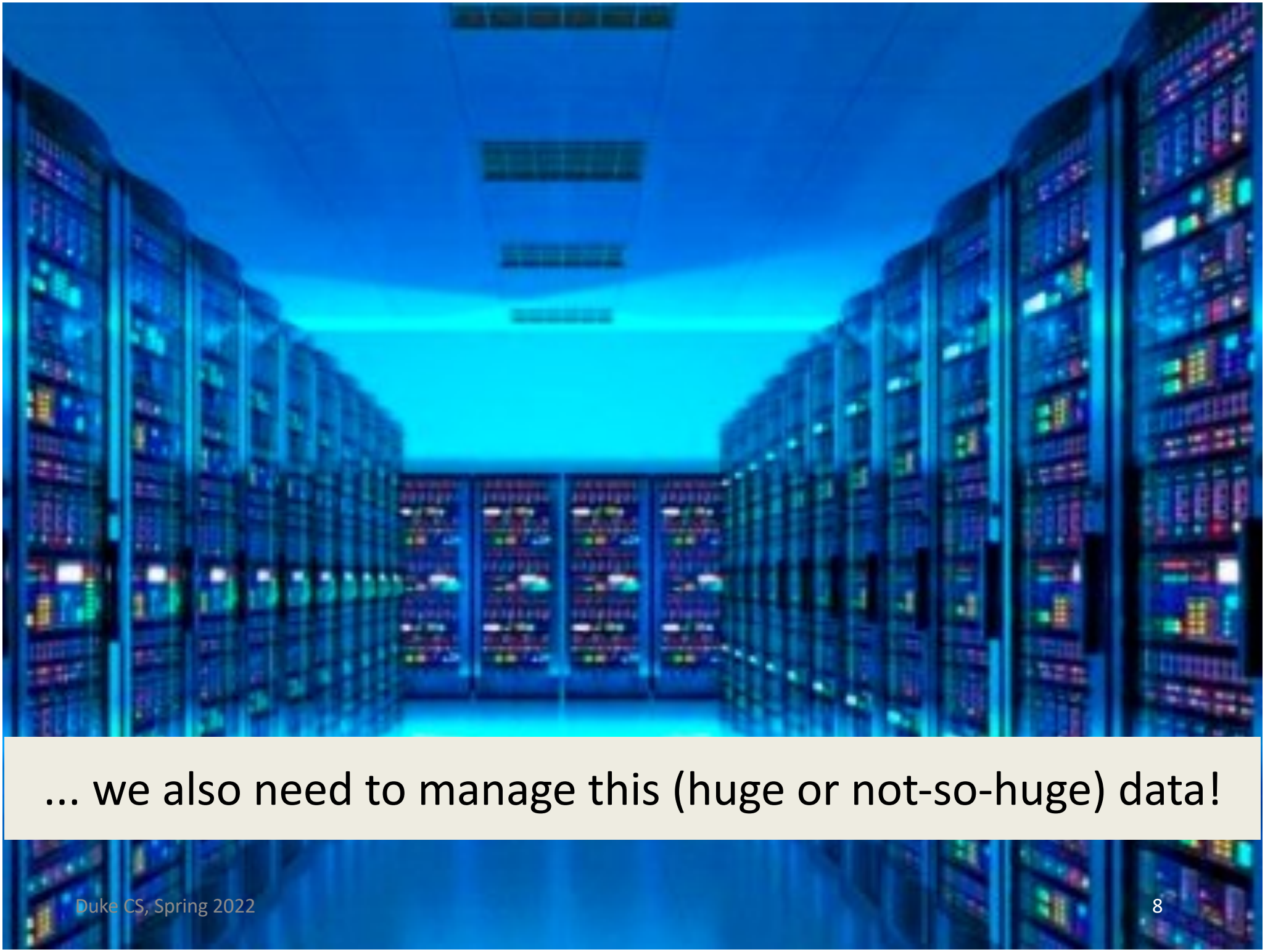
in

Science, Business, Politics, Security Sports, Education, ….

Wait.. don't we need to take a Machine Learning or Statistics course for those things?

Yes, but..

Pic: https://www.technobuffalo.com/sites/technobuffalo.com/files/styles/xlarge/public/wp/2012/05/confused-student.jpg



... we also need to manage this (huge or not-so-huge) data!

Also think about building any application based on data from scratch

- E.g., your own version of mini-Amazon or a Book Selling Platform
- Large data! (think about all books in the world or even in English)

- How do we start?
- A short background survey first…

Who are the key people? (book-selling website)

Who are the key people? (book-selling website)

- At least two types:
  - Database admin (assuming they own all copies of all the books)
  - Users who purchase books
  - Let's proceed with these two only

- Other people:
  - Sellers
  - HR
  - Finance
  - Who deal with the warehouse of the books
  - ….

What should the user be able to do?

- i.e. what the interface look like? (think about Amazon)

What should the user be able to do?

- i.e. what the interface look like? (think about Amazon)

1. Search for books
   - With author, title, topic, price range, ….
2. Purchase books
3. Bookmark/add to wishlist

What should the platform do?

What should the platform do?

1. Returns books as searched by the authors
2. Check that the payment method is valid
3. Update no. of copies as books are sold
4. Manage total money it has
5. Add new books as they are published
6. ….

What are the desired and necessary properties of the platform?

What are the desired and necessary properties of the platform?

- Should be able to handle a large amount of data
- Should be efficient and easy to use (e.g., search with authors as well as title)
- If there is a crash or loss of power, information should not be lost or inconsistent
  - Imagine a user was in the middle of a transaction when a crash happened, paid the money, but the book has not been purchased
- No surprises with multiple users logged in at the same time
  - Imagine one last copy of a book that two users are trying to purchase at the same time
- Easy to update and program
  - For the admin

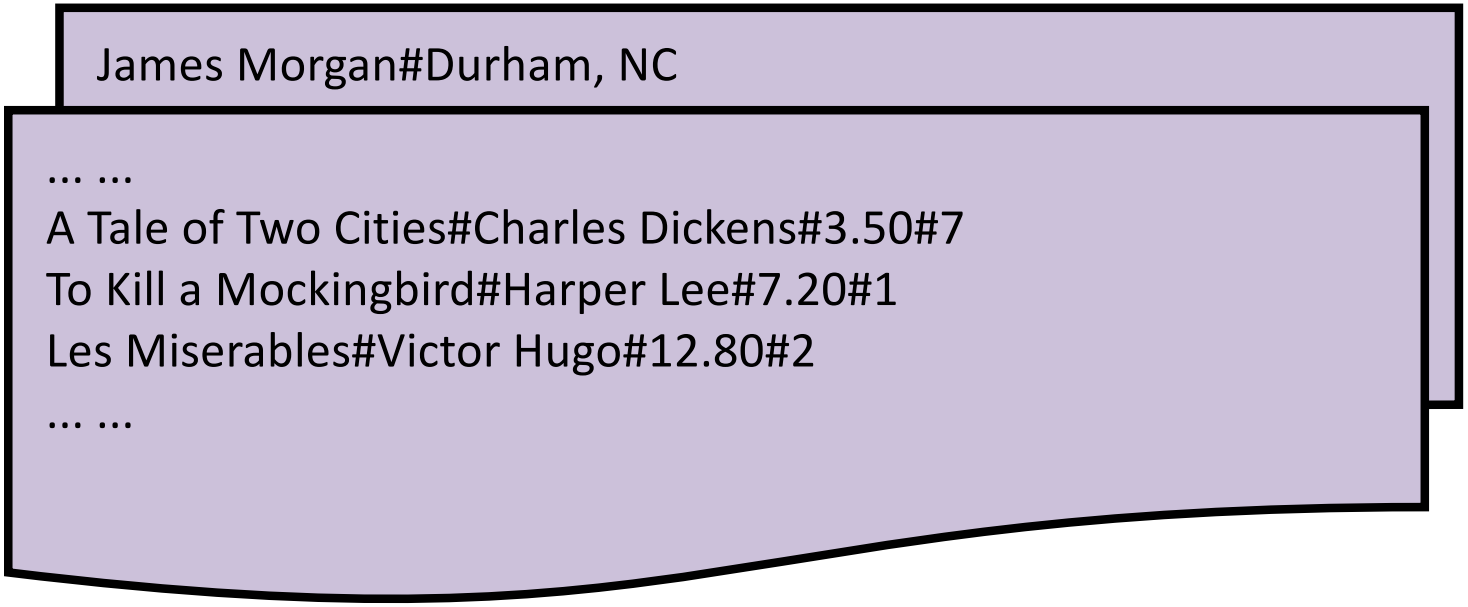
That was the design phase (a basic one though)

https://i1.wp.com/dynamiclandscapes.vita-learn.org/wp-content/uploads/2019/05/Lets-code.jpg?resize=768%2C432&ssl=1

How about C++, Java, or Python?

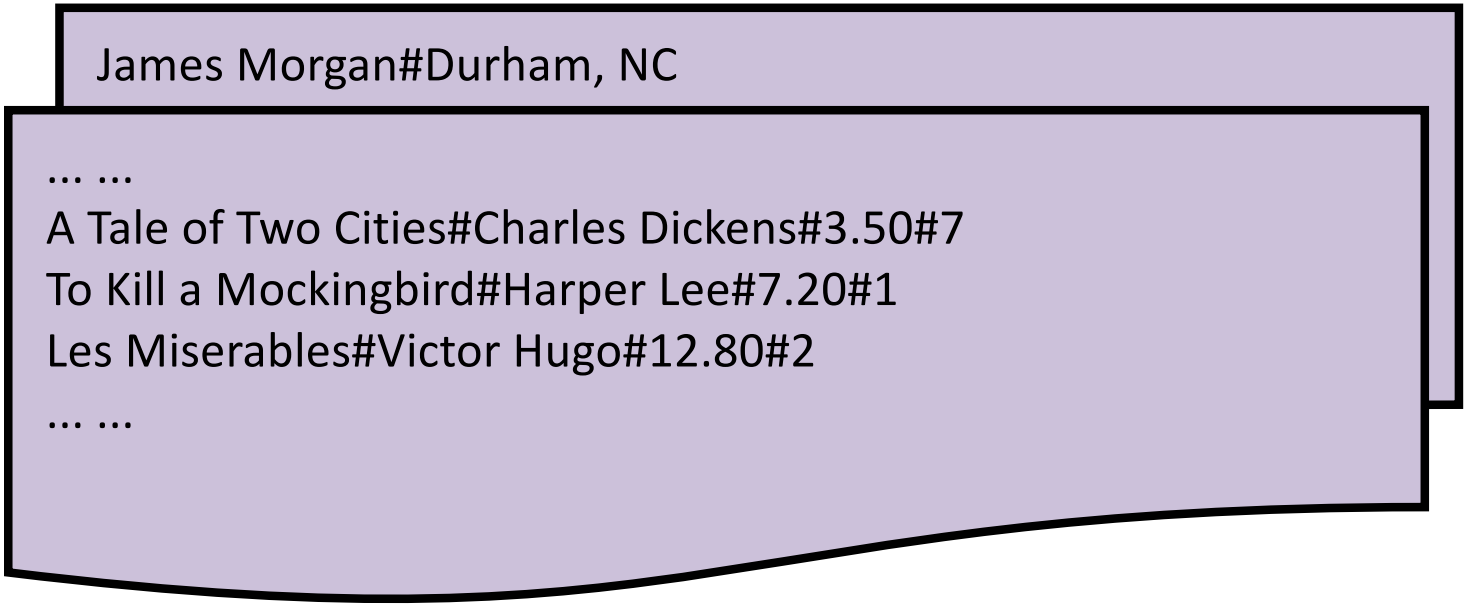
On data stored in large files

Sounds simple!



- Text files – for books, customer, …
- Books listed with title, author, price, and no. of copies
- Fields separated by #’s

Query by programming



- James Morgan wants to buy "To Kill a Mockingbird"
- A simple script
  - Scan through the books file
  - Look for the line containing "To Kill a Mockingbird"
  - Check if the no. of copies is >= 1
  - Bill James $7.20 and reduce the no. of copies by 1

Better idea than scanning?

Binary search! Keep file sorted on titles

What if he changes the "query" and wants to buy a book by Victor Hugo?

Revisit: What are the desired and necessary properties of the platform?

- Should be able to handle a large amount of data
- Should be efficient and easy to use (e.g., search with authors as well as title)
- If there is a crash or loss of power, information should not be lost or inconsistent
  - Imagine a user was in the middle of a transaction when a crash happened, paid the money, but the book has not been purchased
- No surprises with multiple users logged in at the same time
  - Imagine one last copy of a book that two users are trying to purchase at the same time
- Easy to update and program
  - For the admin

Try to open a 10-100 GB file

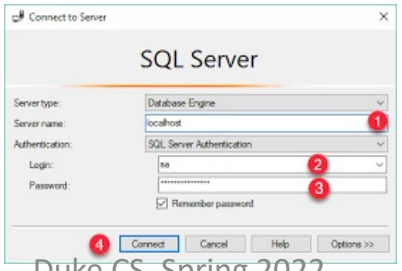
Try to search both on a large flat file

Imagine programmer's task

Imagine adding a new book or updating Copies (+ allow search) on a 10-100 GB text file

Solution?

- DBMS = Database Management System



PostgreSQL

MySQL

ORACLE DATABASE

IBM DB2

mongoDB.

A DBMS takes care of all of the following (and more):

In an easy-to-code, efficient, and robust way

- Sh… to handle a large amount of data
- …uld be efficient and easy to use (e.g., search with a…rs as well as title)
- If there is a crash o… of power, information should not be lost or i…nsistent
  - …a user was in the middle of a transaction when a c… happened, paid the money, but the book has not b…
- No surprises with multiple users logged in …
  - Imagine one last copy of a book that two users a… to purchase at the same time
- Easy to update and program …
  - For the admin

Optimization

Index

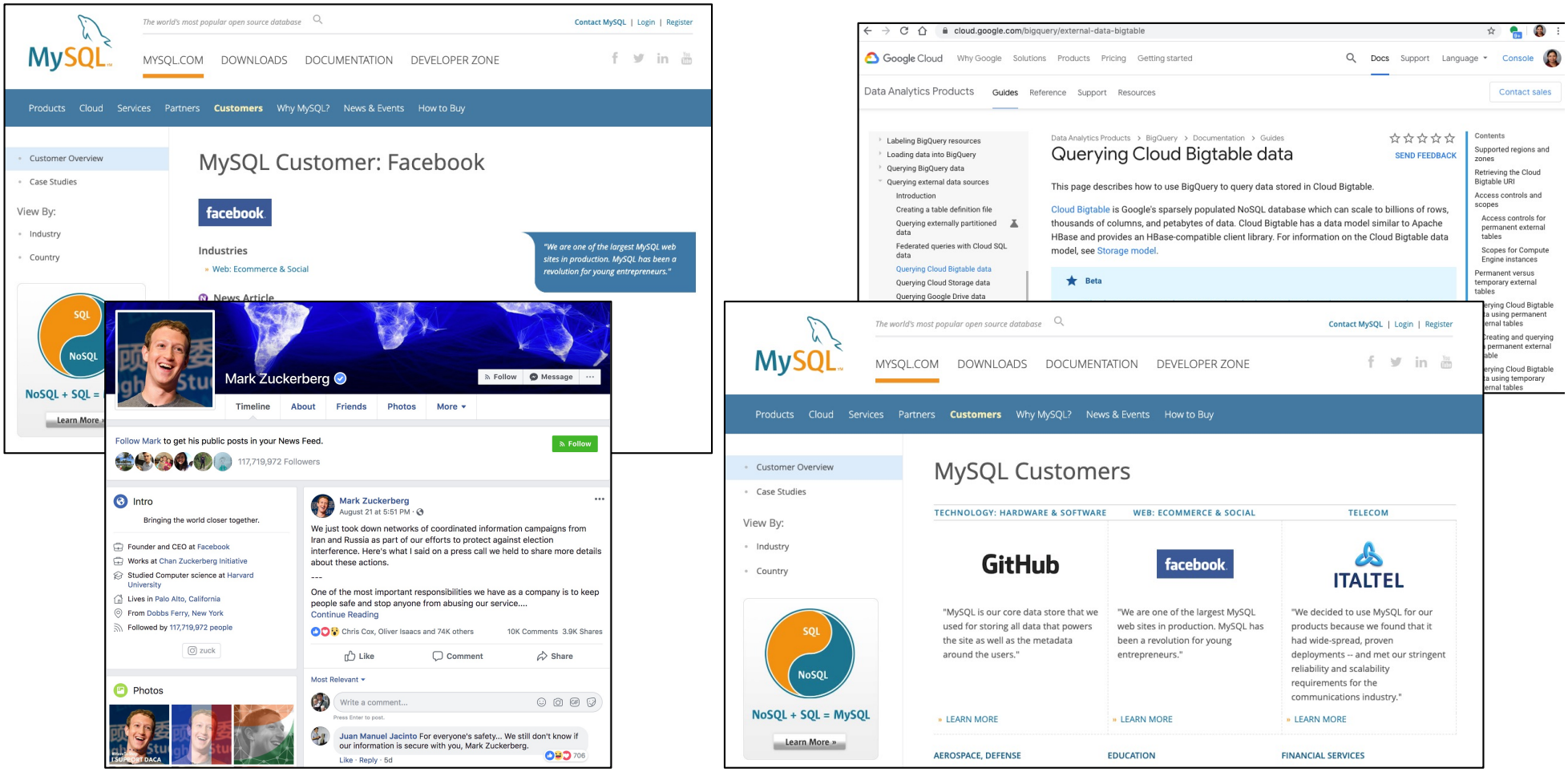
Recovery

Concurrency Control

Declarative

* We will learn these in the course!

DBMS helps the big ones!



Note: Not always the "standard" DBMS (called Relational DBMS), but we need to know pros and cons of all alternatives

CompSci 516: how database systems work and can be used by users

- This is a graduate-level database course in CS
  - We will cover principles, internals, and applications of database systems
- How can a user use a DBMS (programmer's/designer's perspective)
  - Run queries, update data, views (SQL, Relational Algebra, Relational calculus)
  - Design a good database (normalization, constraints)
  - Use different types of data (Mostly relational, also XML/JSON)
- How does a DBMS work (system's or admin's perspective, also for programmers for writing better queries)
  - Storage, index, query processing, join algorithms, query optimizations
  - Transactions: recovery and concurrency control
- Glimpse of advanced topics and other DBMS
  - NOSQL, Spark (big data), data mining, Datalog/recursive queries, Parallel and distributed DBMS

What this course is NOT about

- Spark, AWS, cluster computing…
  - Partially covered in a HW and a lecture
- Machine learning based analytics
- Statistical methods for data analytics
- Python, R
- Data mining..

Warnings!

- Course-load: Three solid programming-heavy and problem-solving HW on three systems + project + quizzes/labs + midterm + final

- HWs will be mostly "disjoint" from lectures and largely self-taught
  - use online tutorials and help of TAs, save enough time
  - HW1: Traditional DBMS -- SQL and Postgres (and some XML too!)
  - HW2: Distributed data processing -- Spark and AWS
  - HW3: NOSQL, MongoDB

- Lecture materials are tested in exams
  - Make sure that you follow the lectures before the next class
  - Some practice problems in quizzes and in class

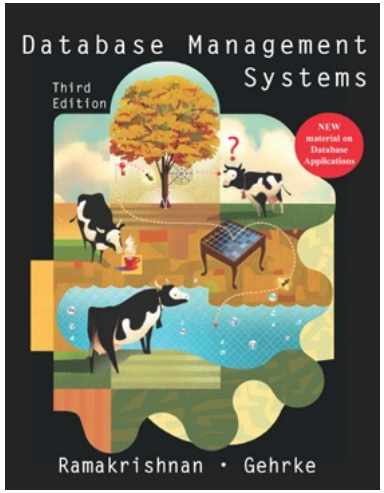
- We will assume backgrounds in some programming language, and in algorithms, data structures, sets, basic systems
  - If you are from a non-CS background, would need to learn these as needed quickly

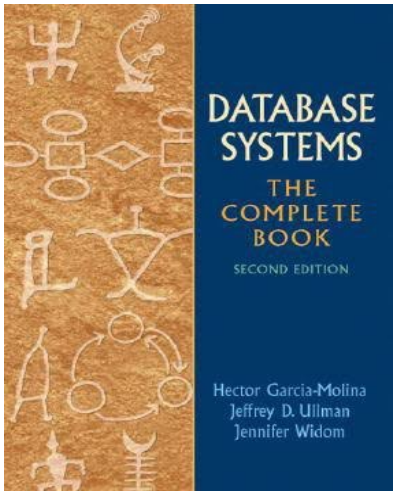
- There will be some "theory", algorithms, maths, and proofs, e.g.,
  - Relational calculus ≡ First-order logic ($\forall$, $\exists$, $\wedge$, $\vee$, $\Rightarrow$)
  - Database normal forms, fixpoints (recursive queries), directed acyclic graphs (transactions)

Logistics

- Discussion forum: Ed
  - All enrolled students are already there
- To reach course staff:
  - Private message on Ed to reach all staff, or send instructor email only for logistics
- Lecture slides will be uploaded before the class as incomplete notes
  - will be updated after the class
- Other tools
  - Gradescope
  - Sakai
  - Gradiance (code will be sent)

Reading Material





- Will mostly follow the ”cowbook” by Ramakrishnan-Gehrke
  - The chapter numbers will be posted
- You do not have to buy the books, but it will be good to consult them from time to time
- You should be prepared to do quite a bit of reading from various books and papers

Grading

- Three Homework: 30%
- Group Project: 13%
- Midterm: 20%
- Final: 25%
  - Exams are comprehensive, closed book/notes, no collaboration
- Class participation: 12%
  - Quizzes/labs in class or short deadline (lowest score dropped): 10%
  - Communication: 2%

Please bring laptops every day!

Grading Strategy

- Absolute and adjustable grading
  - 90% or above: Guaranteed A-range grades (A-, A, or A+)
  - 80-90%: Guaranteed B-range grades (B-, B, or B+)
  - Etc. (see website)
- Thresholds may slide down (your grade may go up) based on class performance
- +/- will depend on the class performance
  - Topper of the class gets A+, and all and only "above expectation" performances get A+.
- Everyone can get good grade by working hard!

Homework

- Due in about 2 weeks after they are posted/previous hw is due
  - ALWAYS start early!
  - Part of the homework may be due in 1 week
- Late policy:
  - 5% penalty per hour unless there are valid reasons or permissions
  - Check out website
  - Start early and do not count on late days!
- contact the instructor if you have a *valid* reason to be late
  - Another exam, project, hw is NOT a valid reason – we will always be fair to all
- To be done individually, but discussions allowed
  - More details in the next class

Projects

- 13% weight
- In groups of 4
  - Groups of smaller and larger sizes need instructor's permission
- Flexible in terms of topic related to data management
  - Fixed option: Database-backed website (template provided)
  - Open options (needs to be approved by the instructor): analyzing data, any db-related research problem
  - Running a standard ML classification/regression on a dataset is not enough
- Work done should be at least equivalent to
  - 1.3 hw * no. of group members
- Weekly updates + proposal + midterm report + final report + demo/presentation
- More information and ideas for projects will be posted later

Some Important Dates

- January 20:
  - Team members' names + tentative project topic due
- February 1:
  - Project proposal due
- February 15:
  - Midterm
- HW1 will be released next week

Please ask questions in class!

- In general, actively participate in the class!
  - Ask questions in class and on Ed
  - Stop me as many times as you need to understand the lectures
  - Answer each other's questions on Ed – this is a big class and getting answers from TAs may take time, although we will monitor all threads

- Anonymous feedback form link on Ed
  - To be checked at least once weekly
  - All feedback, suggestions, concerns welcome!

Let's get started!

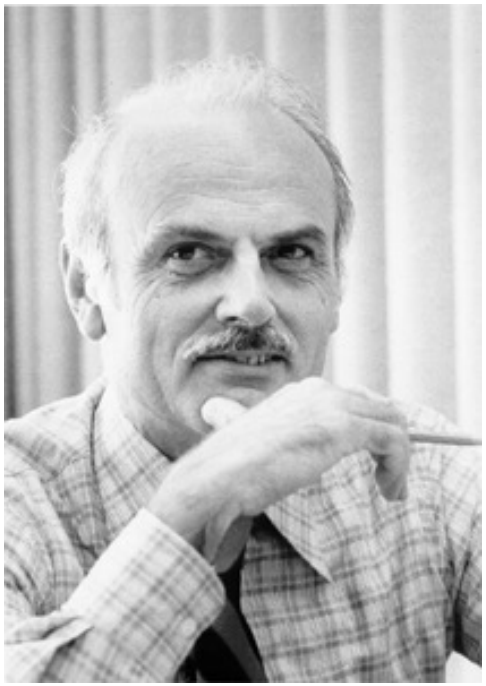
Relational Data Model

What is a good model to store data?
Tree? Nested data? Graph?

(just) Tables!

https://studentreasures.com/blog/student-publishing/young-authors-in-the-spotlight-celebrating-the-winners-of-the-2018-national-book-challenge/

Edgar F. Codd (1923-2003)



- Pilot in the Royal Air Force in WW2
- Inventor of the relational model and algebra while at IBM (1970)
- Turing Award, 1981

RDBMS = Relational DBMS

Motivation of relational model

- Simplicity
- Easy query optimizations
- Separation of abstraction and operations

http://en.wikipedia.org/wiki/File:Edgar_F_Codd.jpg

Relational Data Model

Students

sid	name	login	age	gpa
53666	Jones	jones@cs	18	3.4
53688	Smith	smith@ee	18	3.2
53650	Smith	smith1@math	19	3.8
53831	Madayan	madayan@music	11	1.8
53832	Guldu	guldu@music	12	2.0

- The data description construct is a Relation
  - Represented as a "table"
  - Basically a "set" of records (set semantic)
  - order does not matter
  - and all records are distinct
- however, it is true for the relational model, not for standard DBM
  - allow duplicate rows (bag semantic)
  - unless restricted by key constraints. Why?

Bag: {1, 1, 2, 2, 3, 2, 1, 5, 6, 1}
Set: {1, 2, 3, 5, 6}

Bag vs. Set

Students

sid	name	login	age	gpa
53666	Jones	jones@cs	18	3.4
53688	Smith	smith@ee	18	3.2
53650	Smith	smith1@math	19	3.8
53831	Madayan	madayan@music	11	1.8
53832	Guldu	guldu@music	12	2.0

- Why "bag semantic" and not "set semantic" in standard DBMSs?
  - Primarily performance reasons
  - Duplicate elimination is expensive (requires sorting)
  - Some operations like "projection"s are much more efficient on bags than sets

Relational Data Model

Students

sid	name	login	age	gpa
53666	Jones	jones@cs	18	3.4
53688	Smith	smith@ee	18	3.2
53650	Smith	smith1@math	19	3.8
53831	Madayan	madayan@music	11	1.8
53832	Guldu	guldu@music	12	2.0

Attribute/ Column/ Field

Tuple/ Row/ Record

Value

What is a poorly chosen attribute in this relation?

- Relational database = a set of relations
- A Relation : made up of two parts
  1. Schema
  2. Instance

Schema and Instance

- One schema can have multiple instances
- Schema:
  - A template for describing an entity/relationship (e.g. students)
  - specifies name of relation + name and type of each column

  e.g. Students(sid: string, name: string, login: string, age: integer, gpa: real).
- Instance:
  - When we fill in actual data values in a schema
  - a table, has rows and columns
  - each row/tuple follows the schema and domain constraints
  - #Rows = cardinality, #fields = degree or arity
  - example below

sid	name	login	age	gpa
53666	Jones	jones@cs	18	3.4
53688	Smith	smith@ee	18	3.2
53650	Smith	smith1@math	19	3.8

Cardinality = 3, degree = 5

SQL
(Structured Query Language)

Relational Query Languages

- A major strength of the relational model: supports simple, powerful querying of data.

- Queries can be written intuitively, and the DBMS is responsible for an efficient evaluation
  - The key: precise semantics for relational queries
  - Based on a sound theory!
  - Allows the optimizer to extensively re-order operations, and still ensure that the answer does not change.

The SQL Query Language

- Developed by IBM (systemR) in the 1970s based on Ted Codd's relational model
  - First called "SEQUEL" (Structured English Query Language)
- First commercialized by Oracle (then Relational Software)in 1979
- Standards by ANSI and ISO since it is used by many vendors
  - SQL-86, -89 (minor revision), -92 (major revision), -96, -99 (major extensions), -03, -06, -08, -11, -16

Purposes of SQL

- Data Manipulation Language (DML)
  - Querying: SELECT-FROM-WHERE
  - Modifying: INSERT/DELETE/UPDATE (next week)
- Data Definition Language (DDL)
  - CREATE/ALTER/DROP (next week)

The SQL Query Language

- To find all 18 year old students, we can write:

all attributes

```
SELECT *
FROM Students S
WHERE S.age=18
```

sid	name	login	age	gpa
53666	Jones	jones@cs	18	3.4
53688	Smith	smith@ee	18	3.2

- To find just names and logins, replace the first line:

```
SELECT S.name, S.login
```

End of Lecture-1 (01/06)

- TODOs:
  1. Install Postgres and load MovieLens
     - If you have questions, ask on Ed
  2. You can start looking for project team members