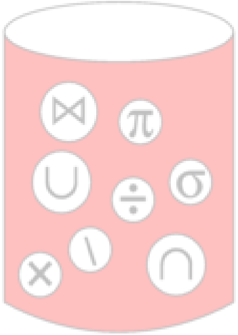


EEECS-3421A:

“Introduction to Database Systems”

Fall 2020

Parke Godfrey

Welcome!

initial: *2020 September 10*

Printable version of talk

To generate a **pdf** of the talk:

- Follow this link: Introduction | 2020-09-10 [to **pdf**].
- Then ask the browser to *print* to get a **PDF**. (*Sadly, this only works correctly in **Chrome** or **Chromium**!*)

Introduction

What is a database?

What is a database?

- A *systematic* collection of data.

What is a database?

- A *systematic* collection of data.
- Which begs the question, how is the data organized?

What is a database?

- A *systematic* collection of data.
- Which begs the question, how is the data organized?
  - We need some general notion of *data model*.

What is a database?

- A *systematic* collection of data.
- Which begs the question, how is the data organized?
  - We need some general notion of *data model*.
  - For a *given* collection of data, we need to specify *its* organization.

    We will call this a *schema*.

What is a database?

- A *systematic* collection of data.
- Which begs the question, how is the data organized?
  - We need some general notion of *data model*.

    data model → how data is organized

  - For a *given* collection of data, we need to specify *its* organization.

    We will call this a *schema*. → organization of data
- How do we retrieve specific data out of a database?

What is a database?

- A *systematic* collection of data.
- Which begs the question, how is the data organized?
  - We need some general notion of *data model*.
  - For a *given* collection of data, we need to specify *its* organization.

    We will call this a *schema*.
- How do we retrieve specific data out of a database?
- How do we *add* and *delete* information?

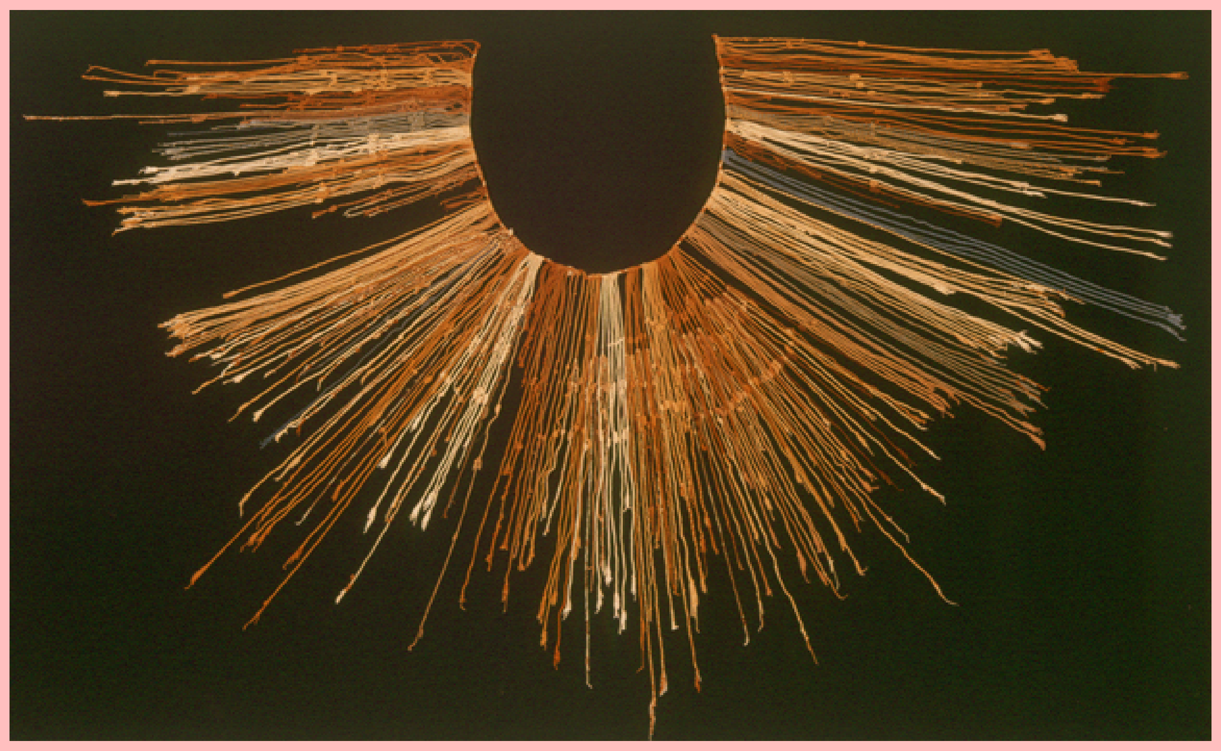
(We will learn this during this course)

Quipu

"…talking knots, were recording devices historically used in a number of cultures and particularly in the region of Andean South America." Quipu @ Wikipedia *2018-01-04*

Quipu

"…talking knots, were recording devices historically used in a number of cultures and particularly in the region of Andean South America." Quipu @ Wikipedia *2018-01-04*



How much data?

How much data?

smartphone calendar	30,000 bytes	KB

How much data?

smartphone calendar	30,000 bytes	KB
university course database	30,000,000 bytes	MB

How much data?

smartphone calendar	30,000 bytes	KB
university course database	30,000,000 bytes	MB
audio collection of Beethoven	30,000,000,000 bytes	GB

How much data?

smartphone calendar	30,000 bytes	KB
university course database	30,000,000 bytes	MB
audio collection of Beethoven	30,000,000,000 bytes	GB
printed collectoion of US Library of Congress	30,000,000,000,000 bytes	TB

How much data?

smartphone calendar	30,000 bytes	KB
university course database	30,000,000 bytes	MB
audio collection of Beethoven	30,000,000,000 bytes	GB
printed collectoion of US Library of Congress	30,000,000,000,000 bytes	TB
data processed by Google *daily*	30,000,000,000,000,000 bytes	PB

How much data?

smartphone calendar	30,000 bytes	KB
university course database	30,000,000 bytes	MB
audio collection of Beethoven	30,000,000,000 bytes	GB
printed collectoion of US Library of Congress	30,000,000,000,000 bytes	TB
data processed by Google *daily*	30,000,000,000,000,000 bytes	PB
total global Internet traffic per month	30,000,000,000,000,000,000 bytes	EB

How much data?

smartphone calendar	30,000 bytes	KB	(Kilo)
university course database	30,000,000 bytes	MB	(Mega)
audio collection of Beethoven	30,000,000,000 bytes	GB	(Giga)
printed collectoion of US Library of Congress	30,000,000,000,000 bytes	TB	(Tera)
data processed by Google *daily*	30,000,000,000,000,000 bytes	PB	(Peta)
total global Internet traffic per month	30,000,000,000,000,000,000 bytes	EB	(Exa)

http://en.wikipedia.org/wiki/Petabyte
http://en.wikipedia.org/wiki/Exabyte
http://www.jameshuggins.com/h/tek1/how_big.htm

Universality of Database Management

Almost all of the above — all except the last — are stored in databases.

Universality of Database Management

Almost all of the above — all except the last — are stored in databases.

And these are “queried” using a common *query language*.

Universality of Database Management

Almost all of the above — all except the last — are stored in databases.

And these are “queried” using a common *query language*.

How universal, we shall have to see…

Break Out

Let's build a *Registrar Database*!

Break Out

Let's build a *Registrar Database*!

- What *information* do we keep?
  What *format*?

Break Out

Let's build a *Registrar Database*!

- What *information* do we keep?
  What *format*?
- How do we *manage* our database?

Break Out

Let's build a *Registrar Database*!

- What *information* do we keep?
  What *format*?
- How do we *manage* our database?
  - We'll write a program to do it!
    What *programming language*?

Break Out

Let's build a *Registrar Database*!

- What *information* do we keep?
  What *format*?
- How do we *manage* our database?
  - We'll write a program to do it!
    What *programming language*?
  - What are the *issues* we face?

Break Out

Let's build a *Registrar Database*!

- What *information* do we keep?
  What *format*?
- How do we *manage* our database?
  - We'll write a program to do it!
    What *programming language*?
  - What are the *issues* we face?

Break Out…

Break Out

Let's build a *Registrar Database*!

- What *information* do we keep?
  What *format*? (XML, speadsheet, database?)
- How do we *manage* our database?
  - We'll write a program to do it!
    What *programming language*?
  - What are the *issues* we face?

Break Out…

…and discuss

So what is a database *system*?

So what is a database *system*?

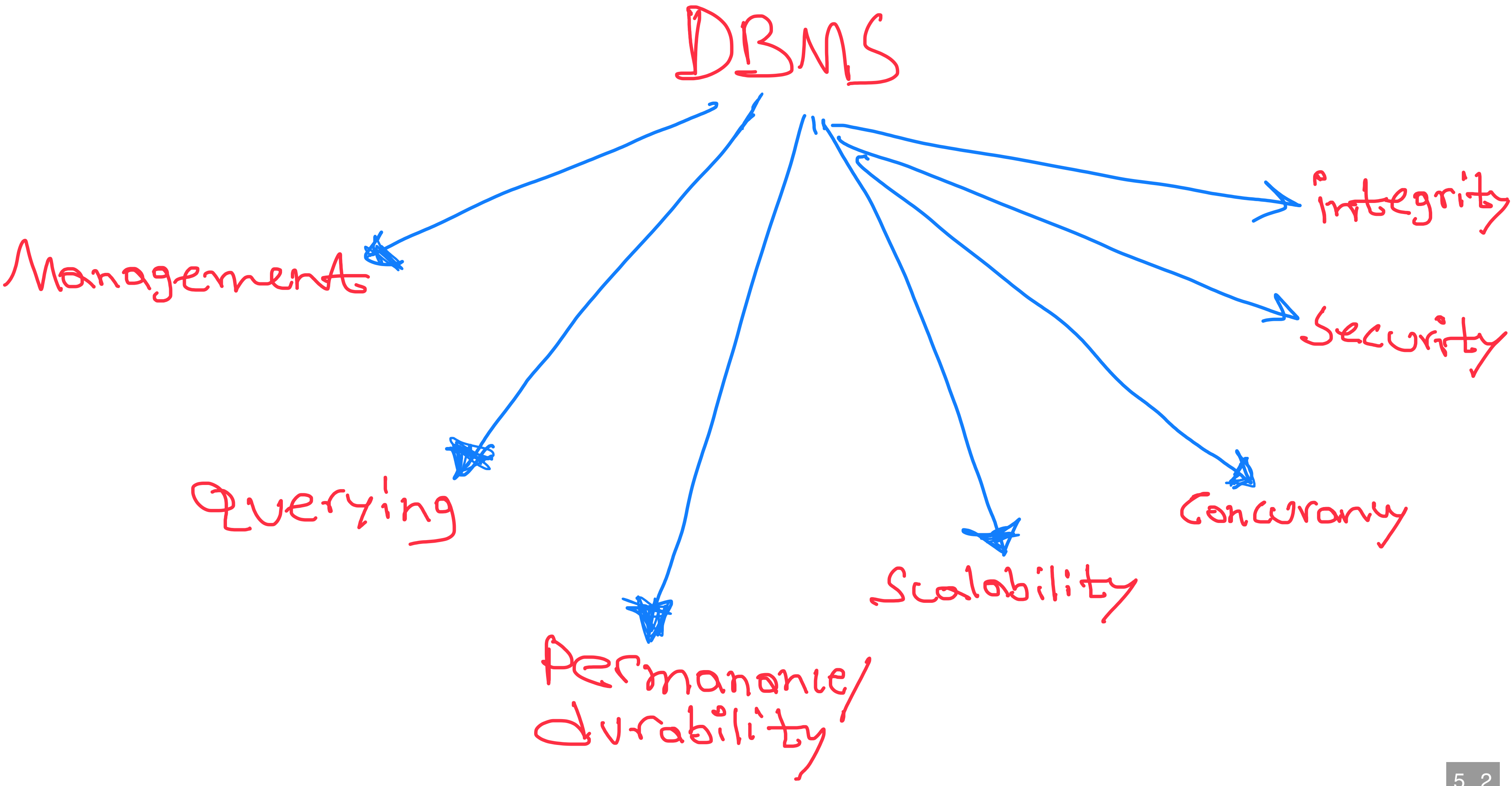
A system to manage — *create*, *update*, and *query* — databases for us.

So what is a database *system*?

A system to manage — *create*, *update*, and *query* — databases for us.

What more *functionality* should such a system provide?

A database system should support…



A database system should support…

- **management**: *manage* databases (and the *data* in them)
  - *create* databases
  - *update* the data
  - *query* the data

A database system should support…

- **management**: *manage* databases (and the *data* in them)
    - *create* databases
    - *update* the data
    - *query* the data
- **querying**: provide a powerful, easy way for users & applications to *retrieve* given data from a database
    - What should this “API” look like?
    - For users?
      For application programs?

A database system should support…

- **management**: *manage* databases (and the *data* in them)
    - *create* databases
    - *update* the data
    - *query* the data
- **querying**: provide a powerful, easy way for users & applications to *retrieve* given data from a database
    - What should this “API” look like?
    - For users?
      For application programs?
- **permanence** / **durability**: *ensure* data is never lost

A database system should support…

- **management**: *manage* databases (and the *data* in them)
    - *create* databases
    - *update* the data
    - *query* the data
- **querying**: provide a powerful, easy way for users & applications to *retrieve* given data from a database
    - What should this “API” look like?
    - For users?

      For application programs?
- **permanence** / **durability**: *ensure* data is never lost
- **scalability**: *scale up* painlessly to *very large* databases

A database system should support…

- **management**: *manage* databases (and the *data* in them)
  - *create* databases
  - *update* the data
  - *query* the data
- **querying**: provide a powerful, easy way for users & applications to *retrieve* given data from a database
  - What should this “API” look like?
  - For users?
    For application programs?
- **permanence** / **durability**: *ensure* data is never lost
- **scalability**: *scale up* painlessly to *very large* databases
- **integrity**: *ensure* correctness (*quality*) of the data

A database system should support…

- **management**: *manage* databases (and the *data* in them)
  - *create* databases
  - *update* the data
  - *query* the data
- **querying**: provide a powerful, easy way for users & applications to *retrieve* given data from a database
  - What should this “API” look like?
  - For users?
    For application programs?
- **permanence** / **durability**: *ensure* data is never lost
- **scalability**: *scale up* painlessly to *very large* databases
- **integrity**: *ensure* correctness (*quality*) of the data
- **concurrency**: handle many users & applications *at the same time*

A database system should support…

- **management**: *manage* databases (and the *data* in them)
    - *create* databases
    - *update* the data
    - *query* the data
- **querying**: provide a powerful, easy way for users & applications to *retrieve* given data from a database
    - What should this “API” look like?
    - For users?
      For application programs?
- **permanence** / **durability**: *ensure* data is never lost
- **scalability**: *scale up* painlessly to *very large* databases
- **integrity**: *ensure* correctness (*quality*) of the data
- **concurrency**: handle many users & applications *at the same time*
- **security**: *manage* and *enforce* privileges for which users & applications can *see* and *manipulate* which data

A database system should support…

- **management**: *manage* databases (and the *data* in them)
  - *create* databases
  - *update* the data
  - *query* the data
- **querying**: provide a powerful, easy way for users & applications to *retrieve* given data from a database
  - What should this "API" look like?
  - For users?
    For application programs?
- **permanence** / **durability**: *ensure* data is never lost
- **scalability**: *scale up* painlessly to *very large* databases
- **integrity**: *ensure* correctness (*quality*) of the data
- **concurrency**: handle many users & applications *at the same time*
- **security**: *manage* and *enforce* privileges for which users & applications can *see* and *manipulate* which data

And do this all *very* efficiently, of course!

A different perspective

A different perspective

This is a different perspective than most of us are used to.

A different perspective

This is a different perspective than most of us are used to.

- *Databases* is a *data-centric* way of looking at things.

A different perspective

This is a different perspective than most of us are used to.

- *Databases* is a *data-centric* way of looking at things.
- The main paradigm is the *query*.

Paradigm → a pattern/model

A history of database systems

Tabulating Machine Company

- It is hard to trace the origins of an idea.

A history of database systems

Tabulating Machine Company

- It is hard to trace the origins of an idea.
- But one could make the case that the field of *databases* all started in 1896...

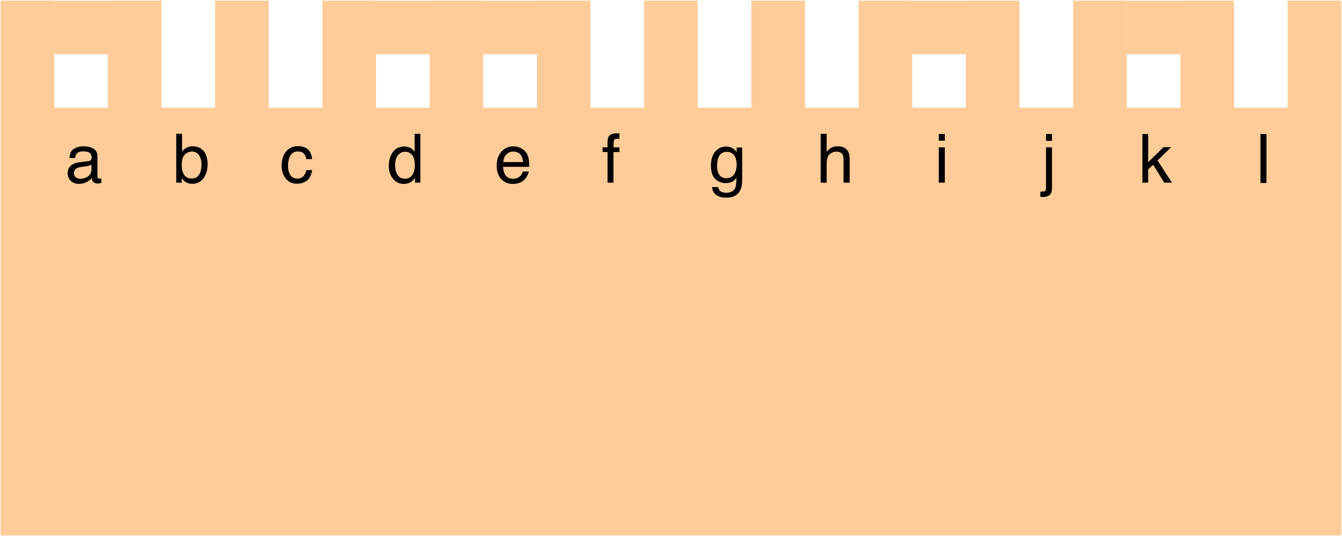
A history of database systems

Tabulating Machine Company

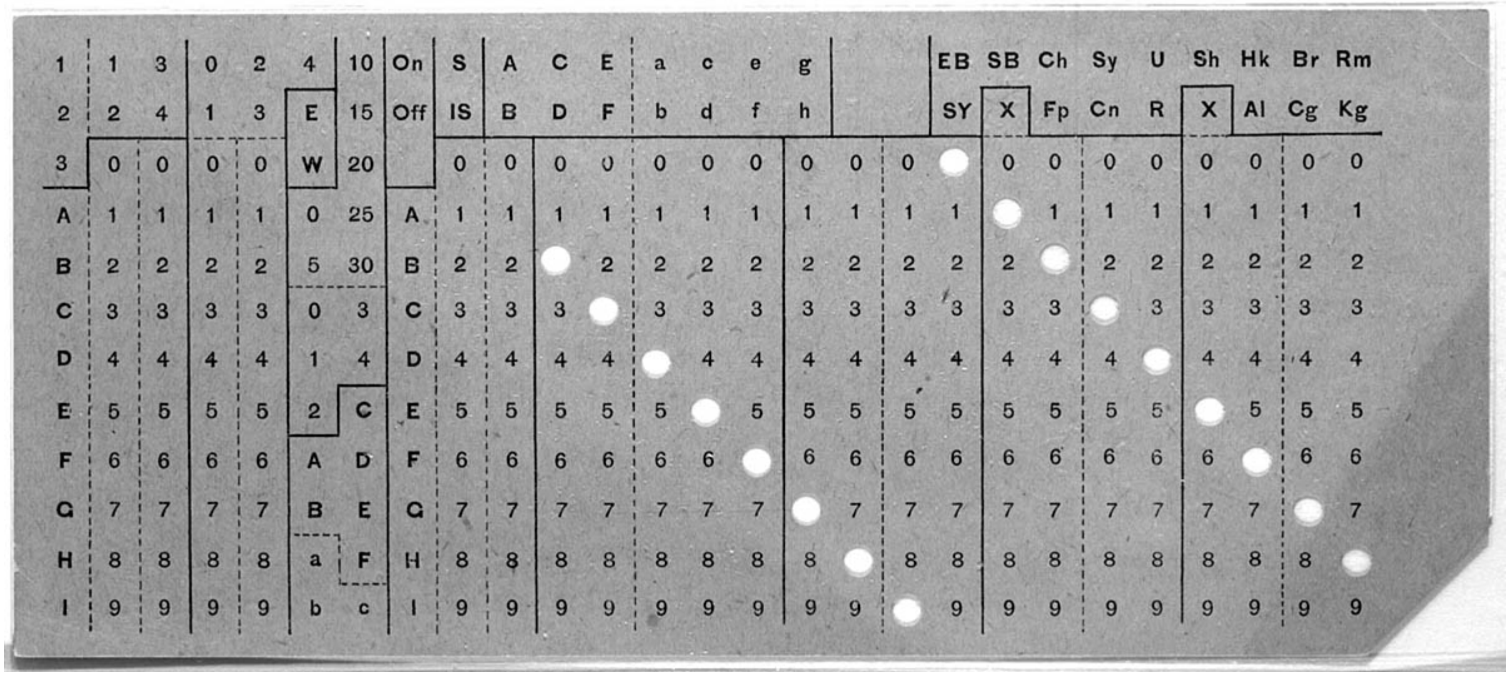
- It is hard to trace the origins of an idea.
- But one could make the case that the field of *databases* all started in 1896...

“IBM” was founded in 1896 as TMC — Tabulating Machine Company — by Herman Hollerith.

Query by selection



Hollerith Punched Cards


1 1 3 0 2 4 10 On S A C E a c e g EB SB Ch Sy U Sh Hk Br Rm
2 2 4 1 3 E 15 Off IS B D F b d f h SY X Fp Cn R X Al Cg Kg
3 0 0 0 0 W 20 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
A 1 1 1 1 0 25 A 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1
B 2 2 2 2 5 30 B 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2
C 3 3 3 3 0 3 C 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3 3
D 4 4 4 4 1 4 D 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4 4
E 5 5 5 5 2 C E 5 5 5 5 5 5 5 5 5 5 5 5 5 5 5 5 5
F 6 6 6 6 A D F 6 6 6 6 6 6 6 6 6 6 6 6 6 6 6 6 6
G 7 7 7 7 B E G 7 7 7 7 7 7 7 7 7 7 7 7 7 7 7 7 7
H 8 8 8 8 a F H 8 8 8 8 8 8 8 8 8 8 8 8 8 8 8 8 8
I 9 9 9 9 b c I 9 9 9 9 9 9 9 9 9 9 9 9 9 9 9 9 9

The relational data model

- **No**…When did modern *computer* database systems come about?!

The relational data model

- **No**…When did modern *computer* database systems come about?!
- In the 1960's.

The relational data model

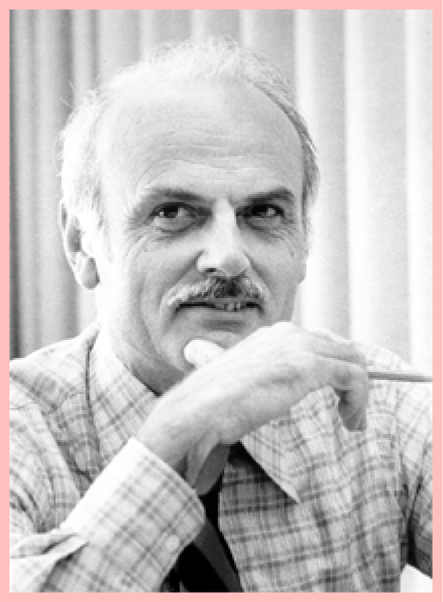
- **No**…When did modern *computer* database systems come about?!
- In the 1960's.
- But our story starts in 1970.

The relational data model

- **No**…When did modern *computer* database systems come about?!
- In the 1960's.
- But our story starts in 1970.

Edgar F. Codd invents the *relational data model*, and its *first-order theory*. [1970]

An IBM team implements *System R* (the first *cost-based relational query optimizer*).



Oracle

Larry Ellison implements *Oracle* from the System R paper, and markets Oracle. [1978]

Oracle

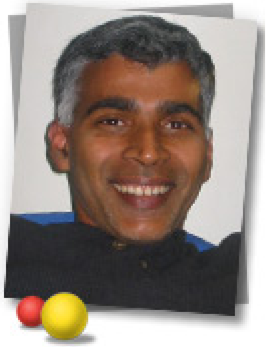
Larry Ellison implements *Oracle* from the System R paper, and markets Oracle. [1978]

IBM wakes up, markets DB2. [~1983]

BigTable

Google implements its own *Big Table* to store the entire WWW.

BigTable was designed and implemented by Jeffery Dean and Sanjay Ghemawat. [2005]

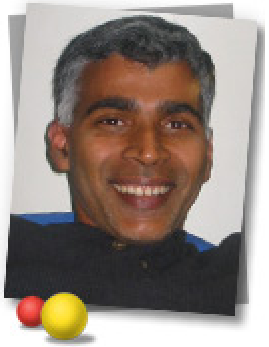


BigTable

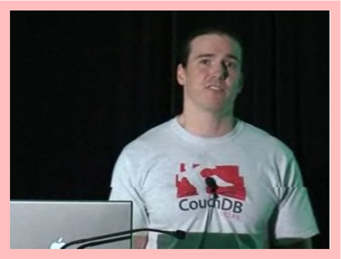
Google implements its own *Big Table* to store the entire WWW.

BigTable was designed and implemented by Jeffery Dean and Sanjay Ghemawat. [2005]

This marks the true start of “big data” revolution.



NoSQL



Facebook and eBay deploys a radically different family of data storage engines, known as NoSQL (”Not only” SQL).

CouchDB is implemented by Damien Katz, a former IBM engineer. [2005]

NoSQL

Facebook and eBay deploys a radically different family of data storage engines, known as NoSQL (”Not only” SQL).

CouchDB is implemented by Damien Katz, a former IBM engineer. [2005]

Scale out versus *scale up*.

Embedded: data in the small

Relational database engines are scaled down to be embedded in mobile devices: Android and iOS.

SQLite is used by both smart phone OSes.

SQLite was implemented by Richard Hipp, for on-board data management of Navy missile system. [2008]

Embedded: data in the small

Relational database engines are scaled down to be embedded in mobile devices: Android and iOS.

SQLite is used by both smart phone OSes.

SQLite was implemented by Richard Hipp, for on-board data management of Navy missile system. [2008]

Sybase Anywhere started in 1992! A Waterloo-Kitchner company.

Now part of SAP.

This course

Outline

Outline

- data models

Outline

- data models
- design

Outline

- **data models**
- **design**
- **queries** (SQL)

Outline

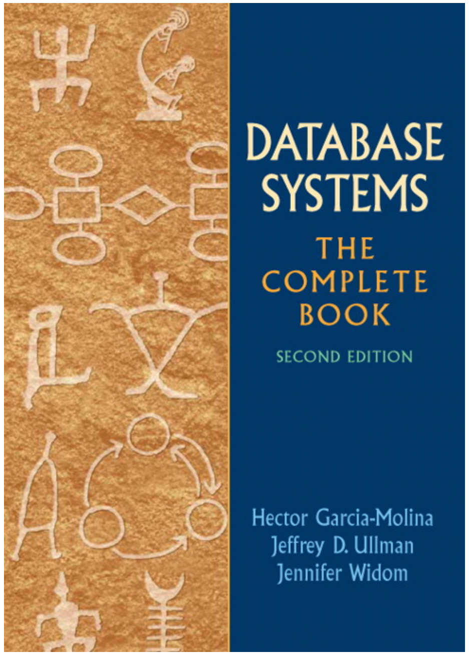
- **data models**
- **design**
- **queries** (SQL)
- **applications**

Outline

- **data models**
- **design**
- **queries** (SQL)
- **applications**

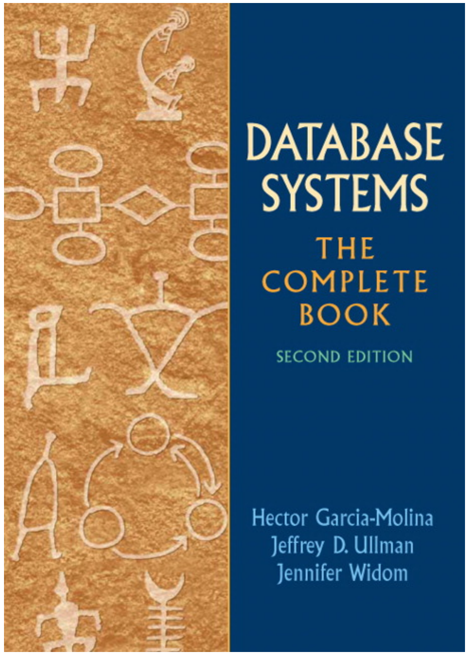
- class site
- syllabus: schedule

Textbook



H. Garcia-Molina, J. Ullman, & J. Widom
Database Systems: The Complete Book
Pearson / Prentice Hall, 2nd Edition

Textbook



H. Garcia-Molina, J. Ullman, & J. Widom
Database Systems: The Complete Book
Pearson / Prentice Hall, 2nd Edition

@amazon.ca

Structure

Structure

- four *projects* (7½% × 2 + 10% × 2 = 30%)

Structure

- four *projects* (7½% × 2 + 10% × 2 = 30%)
- four *quizzes* (4% × 4 = 16%)

Structure

- four *projects* (7½% × 2 + 10% × 2 = 30%)
- four *quizzes* (4% × 4 = 16%)
- *midterm* (15%)

Structure

- four *projects* (7½% × 2 + 10% × 2 = 30%)
- four *quizzes* (4% × 4 = 16%)
- *midterm* (15%)
- *final exam* (34%)

Structure

- four *projects* (7½% × 2 + 10% × 2 = 30%)
- four *quizzes* (4% × 4 = 16%)
- *midterm* (15%)
- *final exam* (34%)

- class site
- syllabus
  - criteria
  - schedule

What this course is *not* about

What this course is *not* about

- This course is *not* about *implementing* database systems.
    - It is about “implementing” (*designing*) databases (*schema*), querying them, and building applications that use databases.
    - **EECS-4411**, *Database Management Systems*, is about *building* database systems.

What this course is *not* about

- This course is *not* about *implementing* database systems.
  - It is about "implementing" (*designing*) databases (*schema*), querying them, and building applications that use databases.
  - **EECS-4411**, *Database Management Systems*, is about *building* database systems.
- This course is *not* about *big data*, per se, which is about *scale out*.
  - **EECS-4415**, *Big Data Systems*, is about that.
  - **EECS-4414**, *Information Networks*, is about *analytics* / *data mining* over very large *social-network* data ("graph databases").