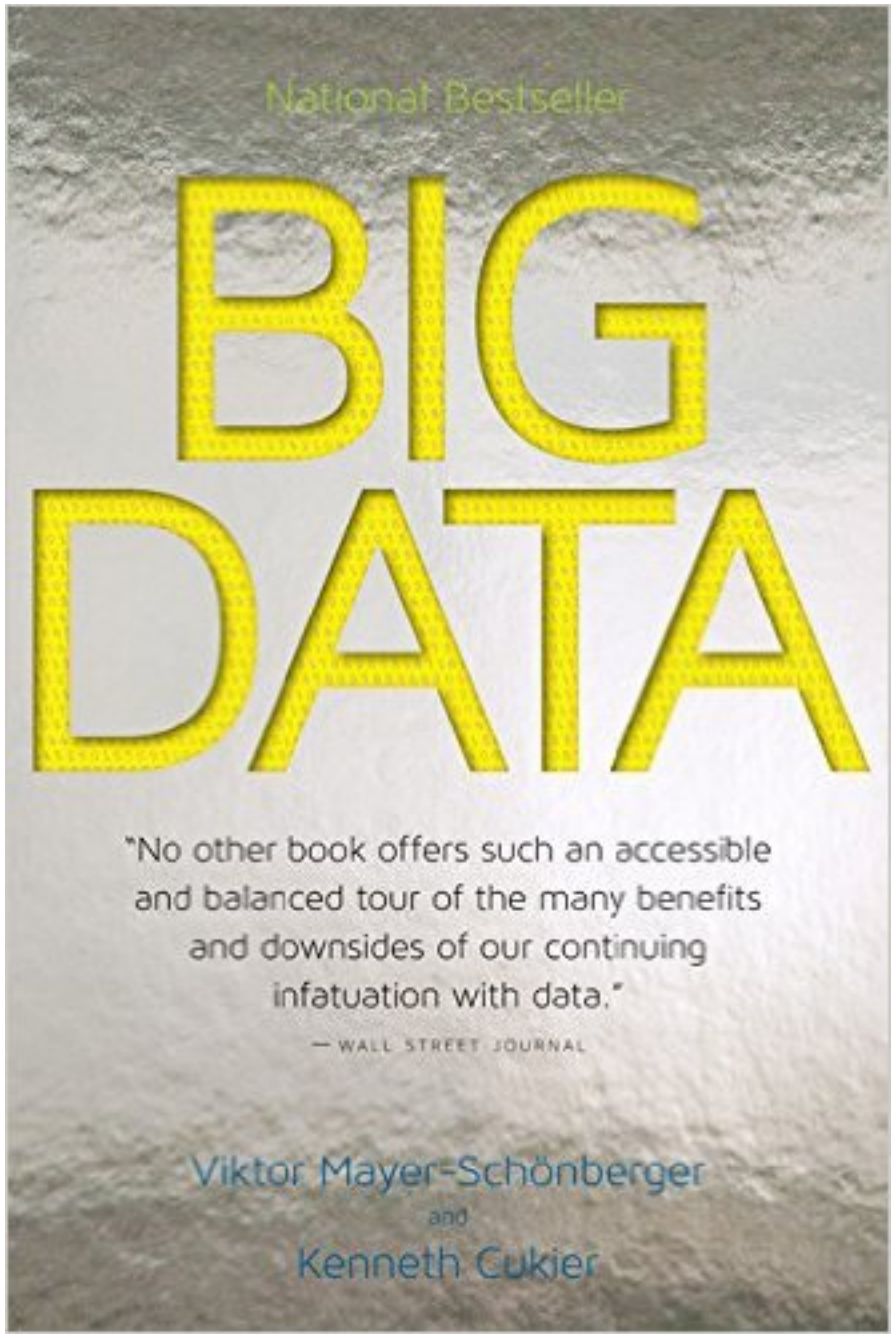
Lecture 3: Big

Core 109S IDWT?, Spring 2017
Michael Hay

Discussion

- The book describes three shifts in mindset in Ch. 2-5.
- What is the shift described in Ch. 2 "More"? (Find good quotes to support your answers!)
- Example from chapter?
- What did you find interesting?
- Connection to Facebook study?



National Bestseller

BIG DATA

"No other book offers such an accessible and balanced tour of the many benefits and downsides of our continuing infatuation with data."

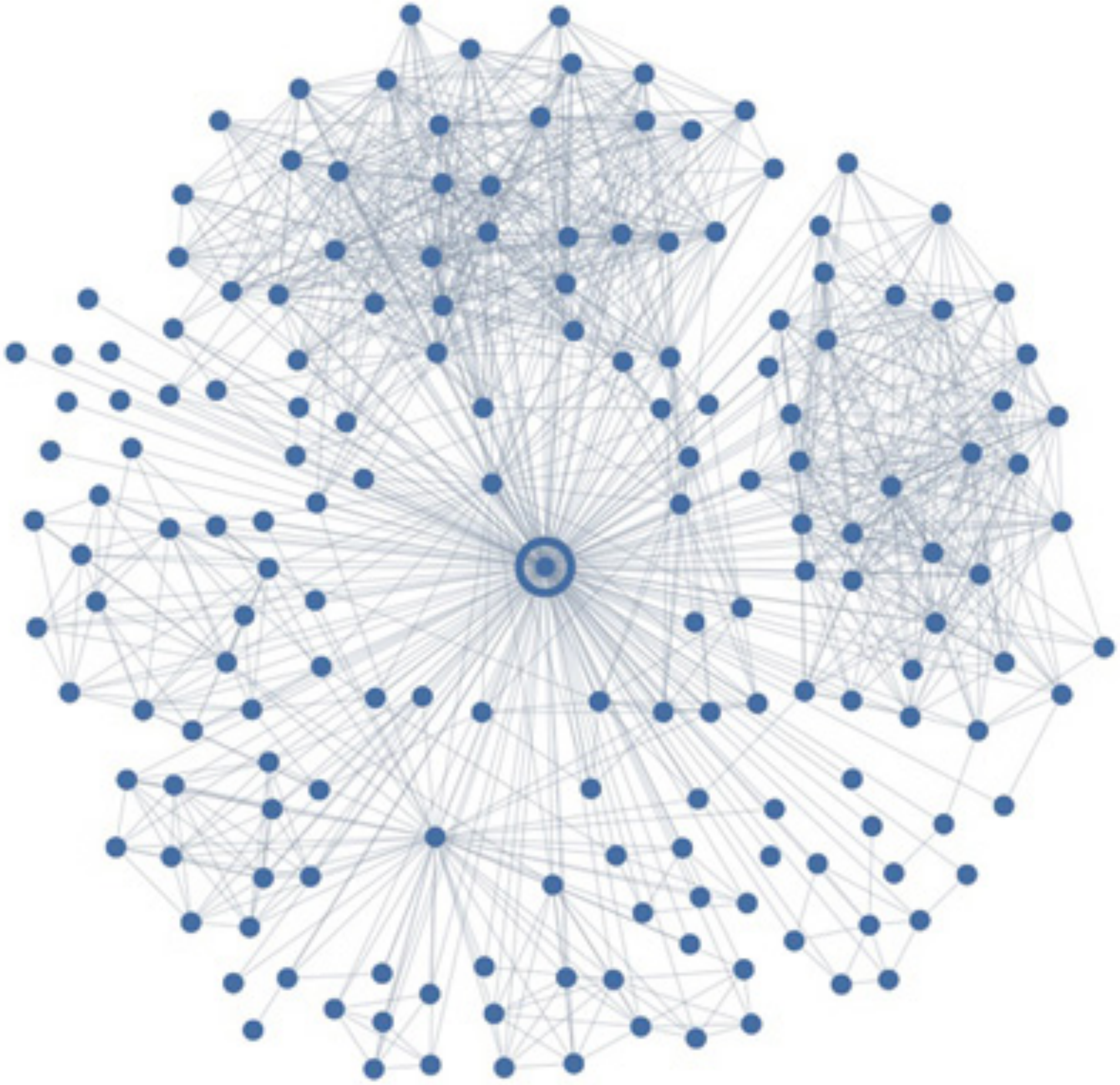
— WALL STREET JOURNAL

Viktor Mayer-Schönberger

and

Kenneth Cukier

Social networks



"Thus, instead of embeddedness, we propose that the link between an individual u and his or her partner v should display a 'dispersed' structure: the mutual neighbors of u and v are not well-connected to one another, and hence u and v act jointly as the only intermediaries between these different parts of the network."

https://arxiv.org/pdf/1310.6753v1.pdf

https://bits.blogs.nytimes.com/2013/10/28/spotting-romantic-relationships-on-facebook

Databases and SQL

- Over the next couple of weeks, we'll focus on relational databases and the *structured query language* (SQL).
- Why?
  - Practical skill: go beyond the spreadsheet!
  - Formal programming languages are an essential ingredient to computer science
  - Want you to experience challenge of translating from "plain English" to *formal specification*

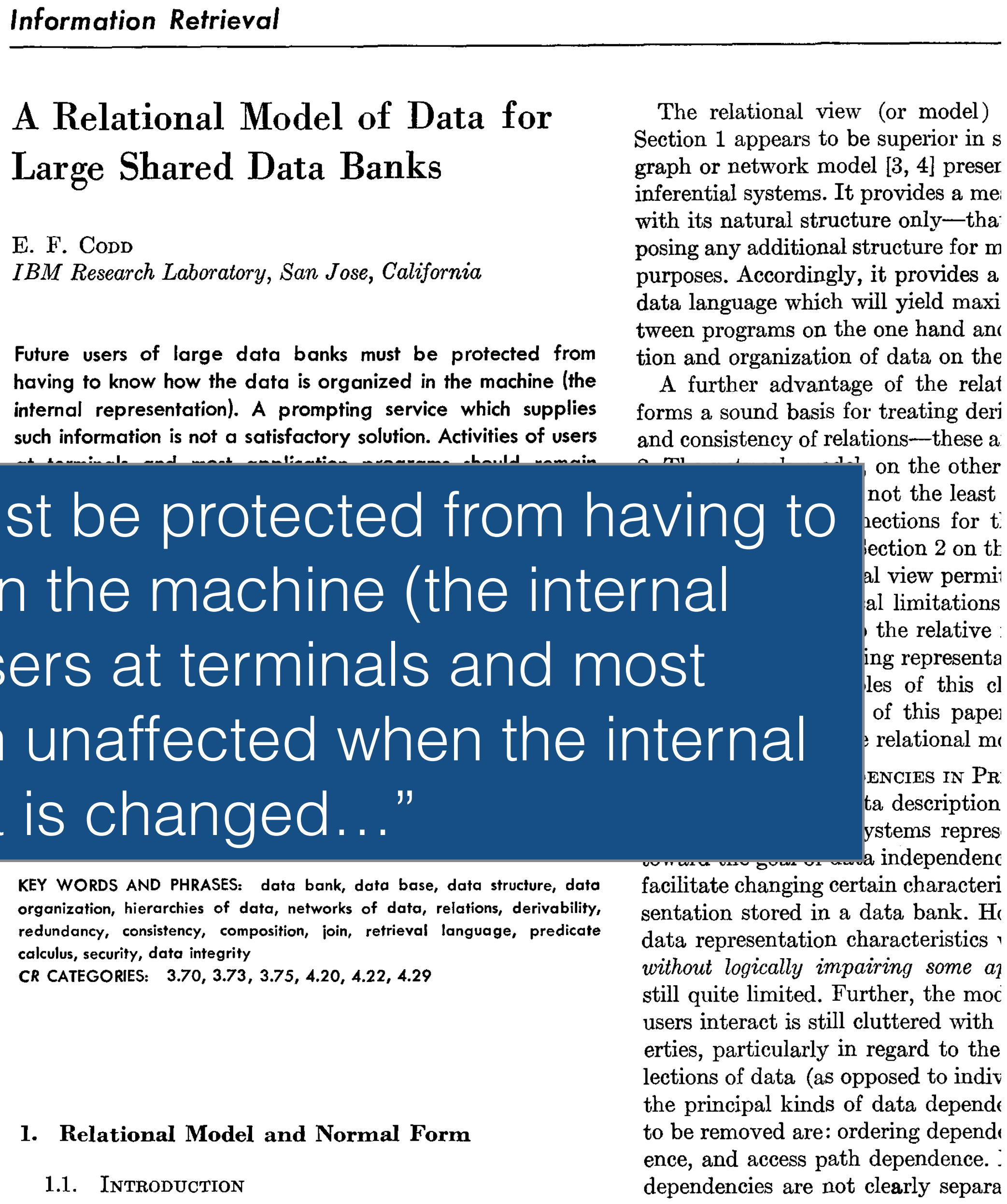
Invention of relational databases

- Invented by E.F. Codd in 1970

- Radical idea: describe data in a logical way that is *independent* of how it is actually stored on disk

> "Future users of large data banks must be protected from having to know how the data is organized in the machine (the internal representation)… Activities of users at terminals and most application programs should remain unaffected when the internal representation of data is changed…"

- Invented in 1970 yet did not see widespread adoption for another 10 years… why?



Information Retrieval

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

KEY WORDS AND PHRASES: data bank, data base, data structure, data organization, hierarchies of data, networks of data, relations, derivability, redundancy, consistency, composition, join, retrieval language, predicate calculus, security, data integrity
CR CATEGORIES: 3.70, 3.73, 3.75, 4.20, 4.22, 4.29

1. Relational Model and Normal Form

1.1. Introduction

The relational view (or model)
Section 1 appears to be superior in s
graph or network model [3, 4] preser
inferential systems. It provides a me
with its natural structure only—tha
posing any additional structure for m
purposes. Accordingly, it provides a
data language which will yield maxi
tween programs on the one hand and
tion and organization of data on the
A further advantage of the relat
forms a sound basis for treating deri
and consistency of relations—these a
, on the other
not the least
nections for t
Section 2 on th
al view permi
al limitations
the relative
ing representa
les of this cl
of this pape
relational m
ENCIES IN PR
ta description
ystems repres
a independenc
facilitate changing certain characteri
sentation stored in a data bank. Ho
data representation characteristics
without logically impairing some a
still quite limited. Further, the mod
users interact is still cluttered with
erties, particularly in regard to the
lections of data (as opposed to indiv
the principal kinds of data depende
to be removed are: ordering depende
ence, and access path dependence.
dependencies are not clearly separa

Relational Database: Definitions

- *Relational database:* a set of *relations*
- *Relation:* made up of 2 parts:
  - *Schema* : name of relation, plus name and type/domain of each column.
  - *Instance* : a *table,* with rows and columns.

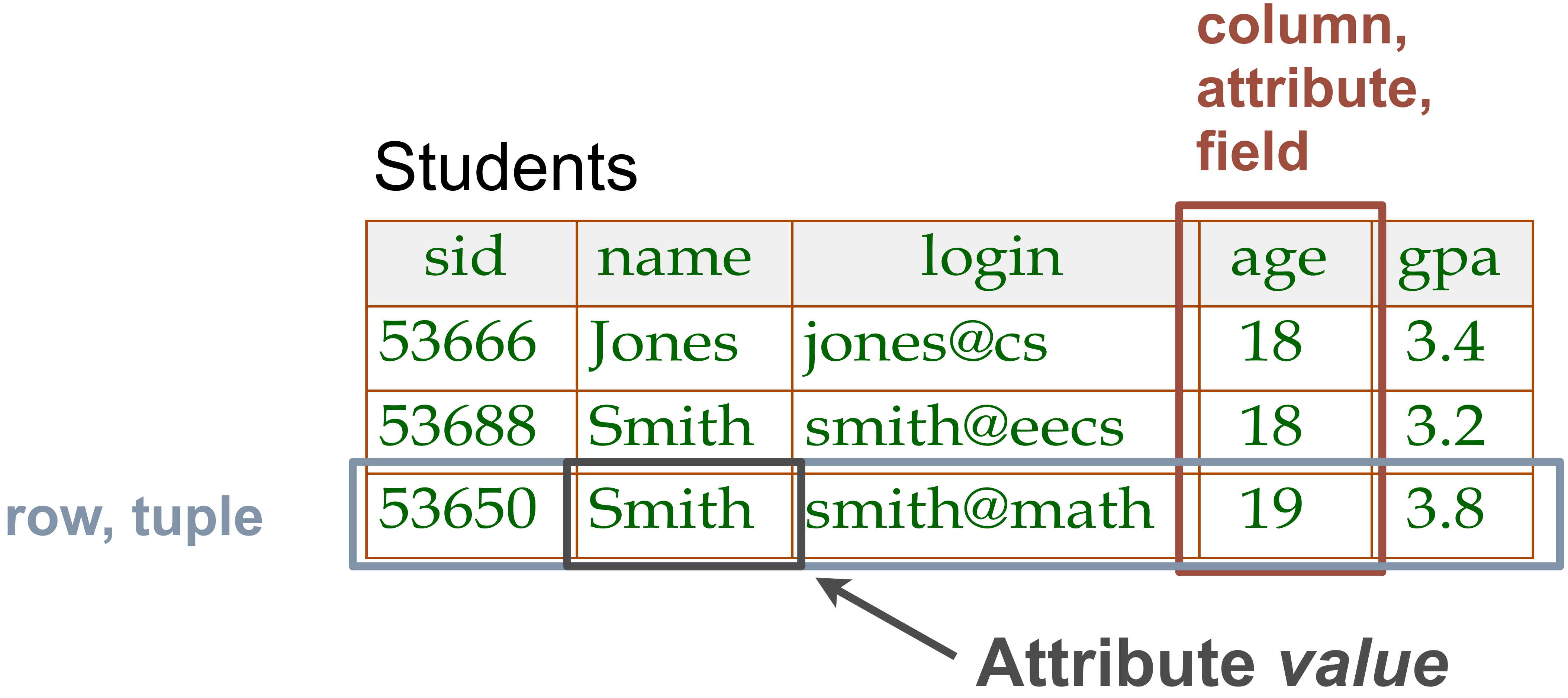
Students(*sid*: string, *name*: string, *login*: string, *age*: integer, *gpa*: real).

Types

Types will be important later: certain operations behave differently, depending on the type

- **Strings**: a string is a sequence of characters. Used to store names, addresses, documents, etc.
  - char (a string of characters)
  - varchar (variable length character string)
- **Int**: stores an integer {…, -2, -1, 0, 1, 2, …}
- **Real**: stores (approximation of) real-valued number (3.14159…)
  - Also known as float, double

Relational instance: a table



Students

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

column,
attribute,
field

row, tuple

Attribute *value*

Example Database

STUDENT

sid	name
1	Jill
2	Bo
3	Maya

Takes

sid	cid
1	445
1	483
3	435

COURSE

cid	title	sem
445	DB	F12
483	AI	S14
435	Arch	F12

PROFESSOR

fid	name
1	Diao
2	Saul
8	Weems

Teaches

fid	cid
1	445
2	483
8	435

Real Schemas Are Complex

Wikipedia Schema

Relation instance: dimensions

- Cardinality
  - number of rows
- Arity/Degree
  - number of attributes
  - unary (1), binary (2), ternary (3), ...

Example Database

STUDENT

sid	name
1	Jill
2	[illegible]
3	[illegible]

binary table w/ cardinality 3

Takes

sid	cid
1	445
1	483
3	435

COURSE

cid	title	sem
445	DP	F12
483	[illegible]	S14
435	[illegible]	F12

ternary table w/ cardinality 3

PROFESSOR

fid	name
[illegible]	Diao
[illegible]	Saul

binary table w/ cardinality 2

Teaches

fid	cid
1	445
2	483
8	435

Keys

- A **key** is a set of one or more attributes whose values are *guaranteed* to identify tuples in the relation uniquely

- Book distinguishes between superkey, candidate key, primary key, foreign key
  - You don't need to know this level of detail.
  - "key" ≈ record identifier
  - "foreign key" ≈ reference to a record of some other (foreign) table.

Example Database

STUDENT

sid	name
1	Jill
2	Bo
3	Maya

Takes

sid	cid
1	445
1	483
3	435

COURSE

cid	title	sem
445	DB	F12
483	AI	S14
435	Arch	F12

What are the keys of these relations?

Are there any foreign keys?

PROFESSOR

fid	name
1	Diao
2	Saul
8	Weems

Teaches

fid	cid
1	445
2	483
8	435

The SQL Query Language

Structured **Q**uery **L**anguage

- Developed by IBM (system R) in the 1970s
- Need for a standard since it is used by many vendors
- Evolving standard
  - SQL-86
  - SQL-89 (minor revision)
  - SQL-92 (major revision)
  - SQL-99 (major extensions)
  - SQL-2003 (minor revisions)
  - SQL-2006 (XML related revisions)
  - SQL-2008 (minor revisions)

Two parts of SQL

- Data Definition Language (DDL)
  - Create/alter/delete tables and their attributes
  - establish and modify schema
- Data Manipulation Language (DML)
  - Query and modify database instance

SQL Overview

- SQL Preliminaries
- Query capabilities
  - SELECT-FROM-WHERE blocks,
  - Basic features, ordering, duplicates
  - Set ops (union, intersect, except)
  - Aggregation & Grouping
  - Nested queries (correlation)
  - Null values
- Modifying the database
- Views

Review in the textbook, Ch 3

Creating Relations in SQL

- Creates the **Student** relation. Observe that the type (domain) of each field is specified, and enforced by the DBMS whenever tuples are added or modified.

```
CREATE TABLE Student
        (sid CHAR(20),
         name CHAR(20),
         login CHAR(10),
         age INTEGER,
         gpa REAL)
```

- As another example, the **Takes** table holds information about courses that students take.

```
CREATE TABLE Takes
        (sid CHAR(20),
         cid CHAR(20),
         grade CHAR(2))
```

Data Types in SQL

- **Characters:**
  - CHAR(20) -- fixed length
  - VARCHAR(40) -- variable length
- **Numbers:**
  - BIGINT, INT, SMALLINT, TINYINT
  - REAL, FLOAT -- differ in precision
  - MONEY
- **Times and dates:**
  - DATE
  - DATETIME
- Others...

Destroying and Altering Relations

```
DROP TABLE Student
```

- Destroys the relation Student. The schema information *and* the tuples are deleted.

```
ALTER TABLE Student
        ADD COLUMN firstYear integer
```

- The schema of Student is altered by adding a new field; every tuple in the current instance is extended with a *null* value in the new field.

SQL Overview

- Query capabilities
  - SELECT-FROM-WHERE blocks,
  - Basic features, ordering, duplicates
  - Set operations (union, intersect, except)
  - Aggregation & Grouping
  - Nested queries (correlation)
  - Null values