CS186 - Introduction to Database Systems

Fall Semester 2013
Prof. Michael Franklin

"Knowledge is of two kinds: we know a subject ourselves, or we know where we can find information upon it."

-- Samuel Johnson (1709-1784)

What I know about 186 Enrollment

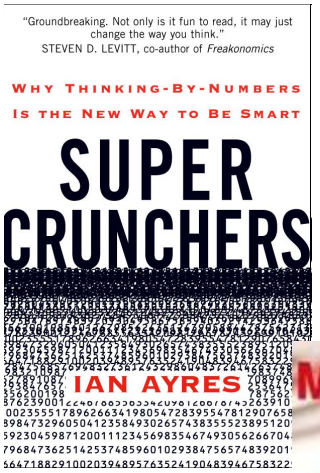
- **We are at capacity (room, section & GSI/TA)**
  - actually oversubscribed – some attrition built in
  - about 40 people from the waitlist were let in yesterday
- **Wait list will be processed in order, but only as (if) people drop the course.**
  - The further down the list you are the worse your odds
  - This happens for the next week and a half or so
- **We won't be able to let in Concurrent Enrollment Students**
- **A (smaller) offering of CS 186 is scheduled for next semester**
  - Taught by visiting DB professor from Oxford
- **Michael-David Sasson (CS office) handles it from here**

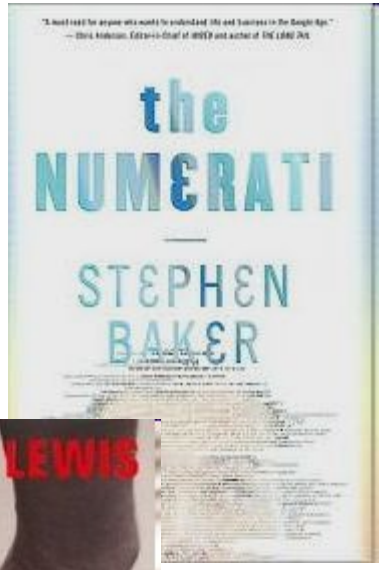
Plan for Today

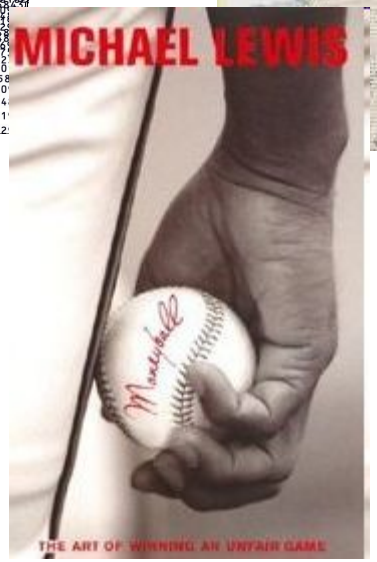
- Why Study Databases?
- What are Databases and DBMSs?
- Overview of the Course
- Introduction to SQL

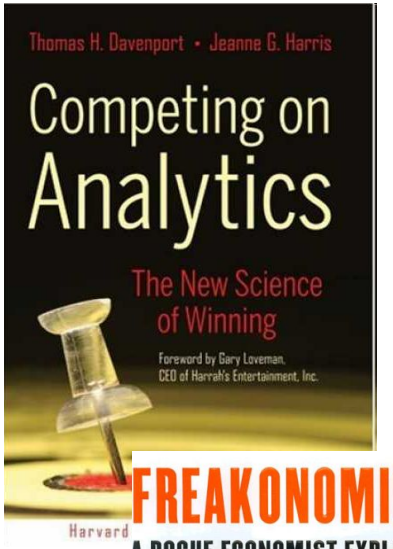
Databases – Why Study Them?

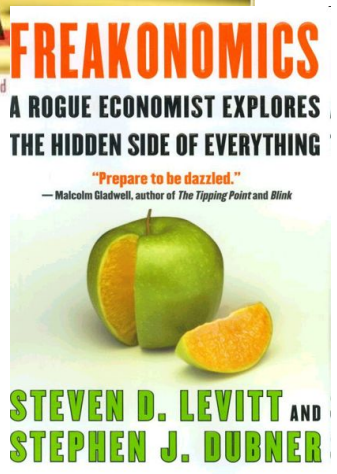
Databases – Why Study Them?







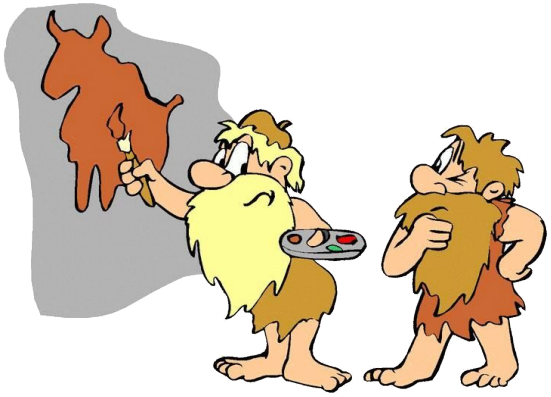


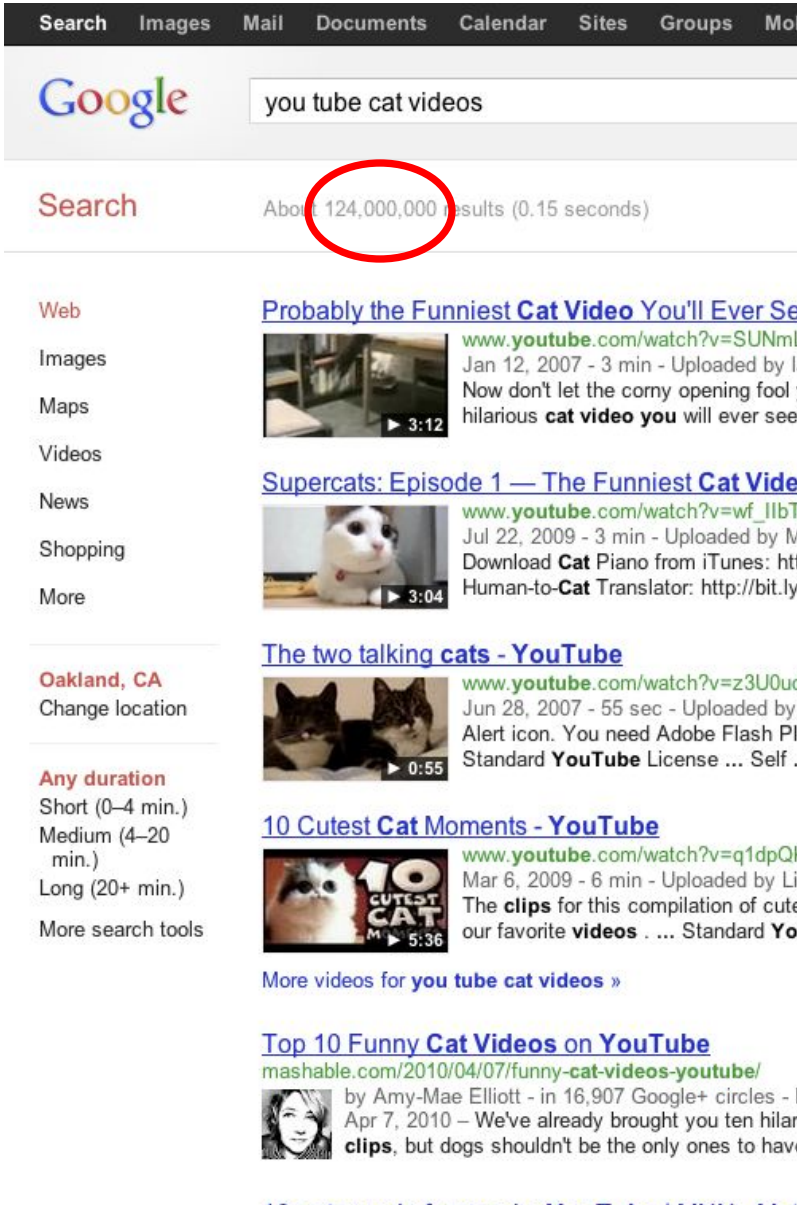


The "Big Data" Buzz – Why?

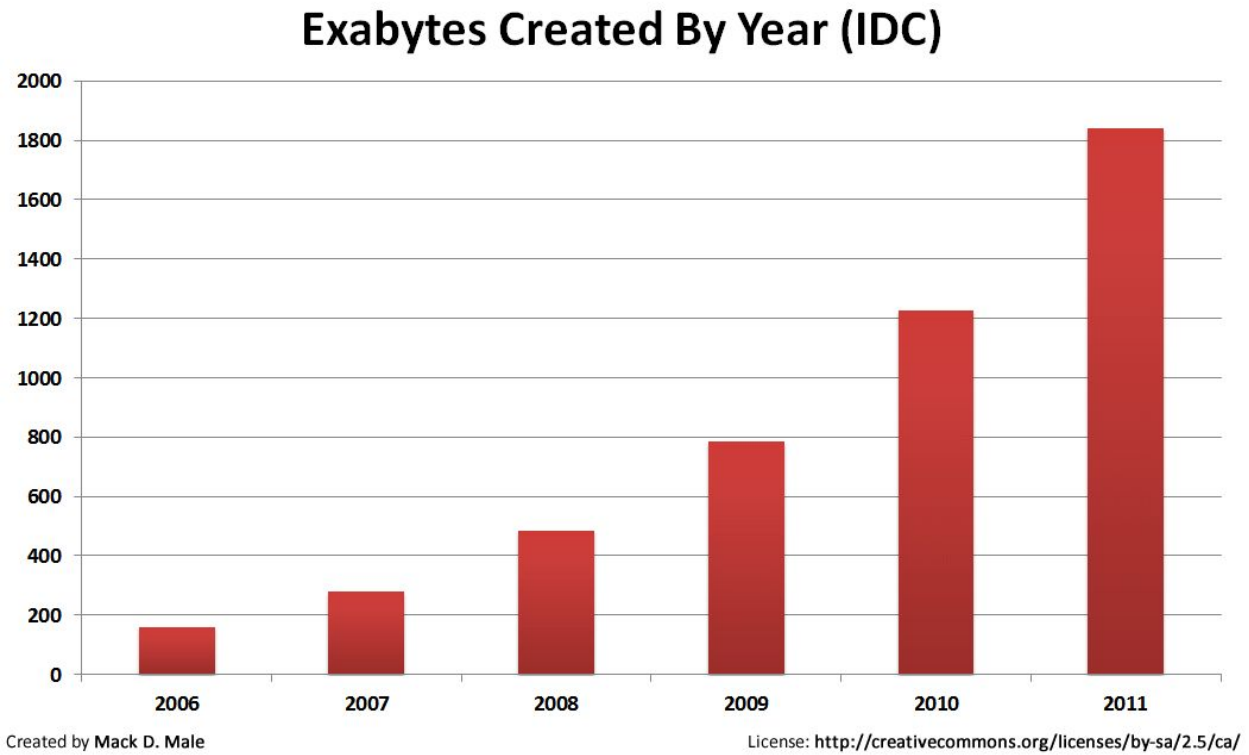
"Between the dawn of civilization and 2003, we only created five exabytes of information; now we're creating that amount every two days." Eric Schmidt, Google

(and others)





The "Big Data" Buzz – Why?



SI decimal prefixes		Binary usage
Name (Symbol)	Value	
kilobyte (kB)	10^3	2^{10}
megabyte (MB)	10^6	2^{20}
gigabyte (GB)	10^9	2^{30}
terabyte (TB)	10^{12}	2^{40}
petabyte (PB)	10^{15}	2^{50}
exabyte (EB)	10^{18}	2^{60}
zettabyte (ZB)	10^{21}	2^{70}
yottabyte (YB)	10^{24}	2^{80}

"The sexy job in the next 10 years will be statisticians." "Data Scientists"?

Hal Varian
Prof. Emeritus UC Berkeley
Chief Economist, Google

Sources Driving "Big Data"

It's All Happening On-line

Every:
Click
Ad impression
Billing event
Fast Forward, pause,…
Friend Request
Transaction
Network message
Fault
…

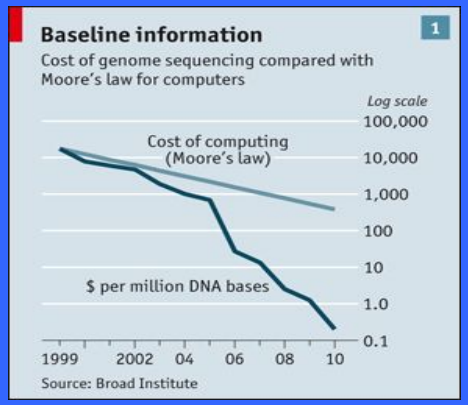
User Generated (Web & Mobile)



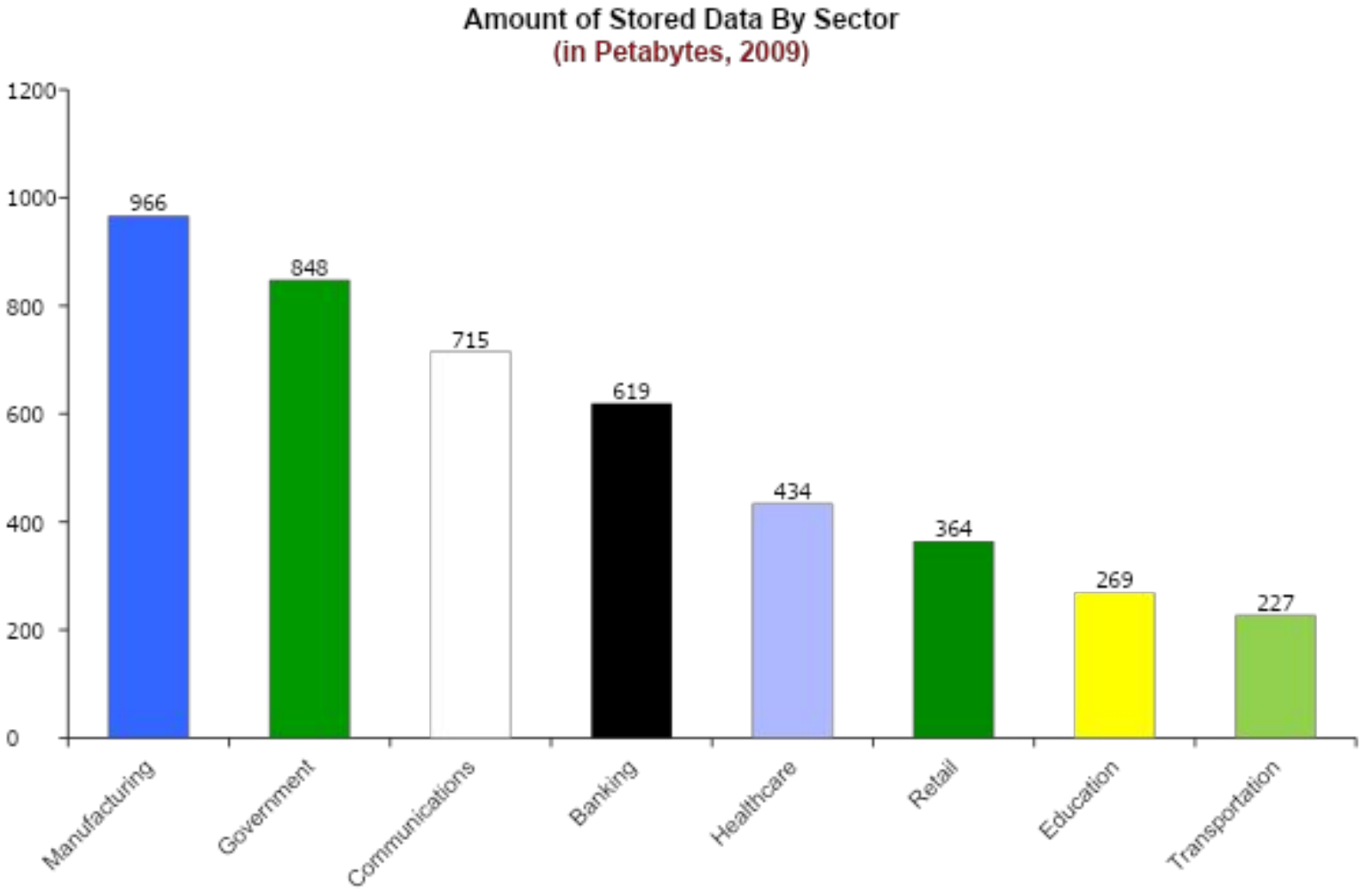

. . .
..


Internet of Things / M2M

Scientific Computing



Some Numbers by Industry



Sources:
"Big Data: The Next Frontier for Innovation, Competition and Productivity."
US Bureau of Labor Statistics | McKinsley Global Institute Analysis

AMPLab@UC Berkeley

Office of Science and Technology Policy
Executive Office of the President
New Executive Office Building
Washington, DC 20502

FOR IMMEDIATE RELEASE
March 29, 2012

Contact: Rick Weiss 202 456-6037 rweiss@ostp.eop.gov
Lisa-Joy Zgorski 703 292-8311 lisajoy@nsf.gov

OBAMA ADMINISTRATION UNVEILS "BIG DATA" INITIATIVE: ANNOUNCES $200 MILLION IN NEW R&D INVESTMENTS

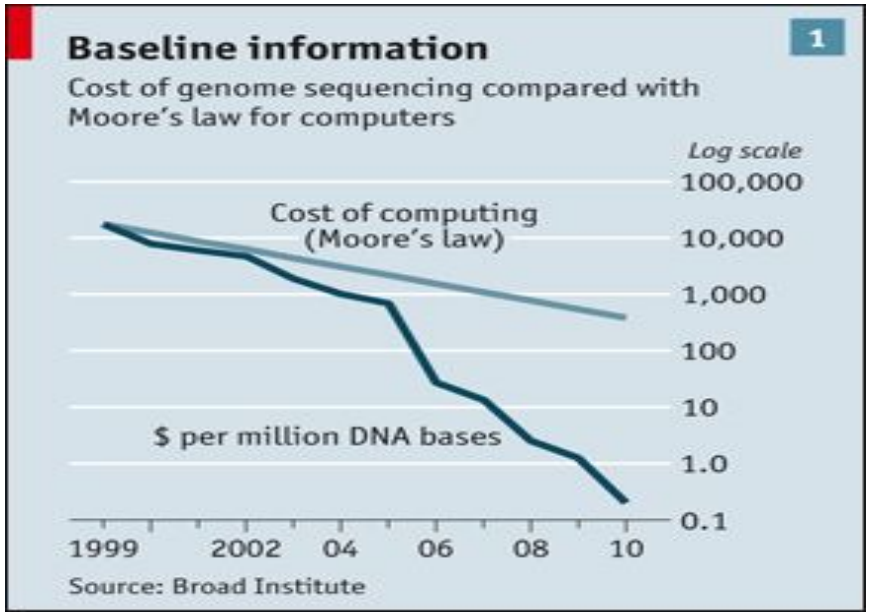
National Science Foundation: In addition to funding the Big Data solicitation, and keeping with its focus on basic research, NSF is implementing a comprehensive, long-term strategy that includes new methods to derive knowledge from data; infrastructure to manage, curate, and serve data to communities; and new approaches to education and workforce development. Specifically, NSF is:

- Encouraging research universities to develop interdisciplinary graduate programs to prepare the next generation of data scientists and engineers;
- Funding a $10 million Expeditions in Computing project based at the University of California, Berkeley, that will integrate three powerful approaches for turning data into information - machine learning, cloud computing, and crowd sourcing;

Big Data, Societal-Scale App?

- **Cancer Tumor Genomics**
- **Vision: Personalized Therapy**
  - "…10 years from now, each cancer patient is going to want to get a genomic analysis of their cancer and will expect customized therapy based on that information."

Director, The Cancer Genome Atlas (TCGA), *Time Magazine,* 6/13/11



UCSF cancer researchers + UCSC cancer genetic database + UCB AMPLab

What: Current Market

- **Relational DBMSs anchor the software industry**
  - Elephants: Oracle, IBM, Microsoft, Teradata, HP, EMC, …
  - Open source: MySQL, PostgreSQL
  - New "Big Data" Entrants: Hive & Pig (Hadoop), Shark (Spark),
- **Obviously, Search**
  - Google & Bing
- **Open Source "NoSQL"**
  - Hadoop MapReduce, Spark
  - Key-value stores: Cassandra, Riak, Voldemort, Mongo, …
- **Cloud services**
  - Amazon, Google AppEngine, MS Azure, Heroku, …

What is a Database?

A *database* is an integrated and organized collection of data

Key Concept: Structured Data

- A *data model* is a collection of concepts for describing data.
- A *schema* is a description of a particular collection of data, using a given data model.

- The *relational model of data* is the most widely used model today.
  - Main concept: *relation*, basically a table with rows and columns.
  - Every relation has a *schema*, which describes the columns, or fields.

What is a Relational Database?

[The Relational Model] provides a basis for a high level data language which will yield <u>maximal independence</u> between programs on the one hand and machine representation on the other. (E.F. Codd, *1981 Turing Award winner*)

In Other Words…

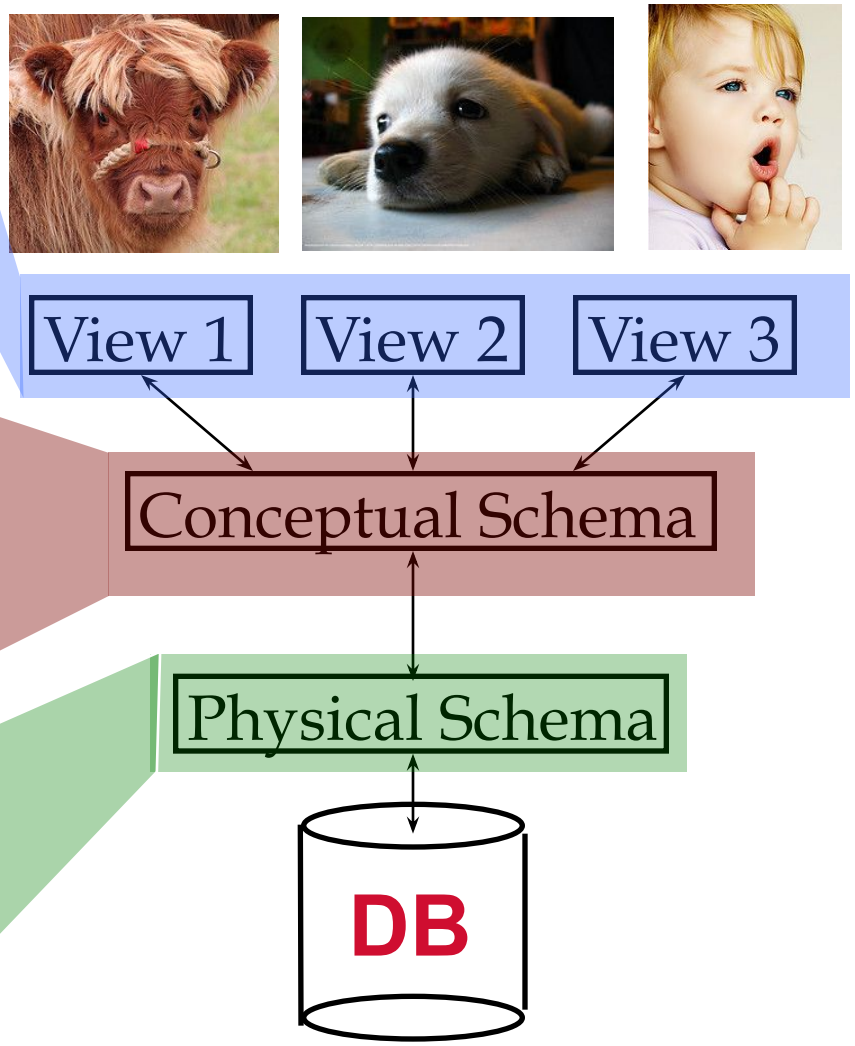
Relational DataBase Management Systems were invented to let you use one set of data in multiple ways, including ways that are unforeseen at the time the database is built and the 1st applications are written.

(Curt Monash, analyst/blogger)

That is, think about the data independently of any particular program.

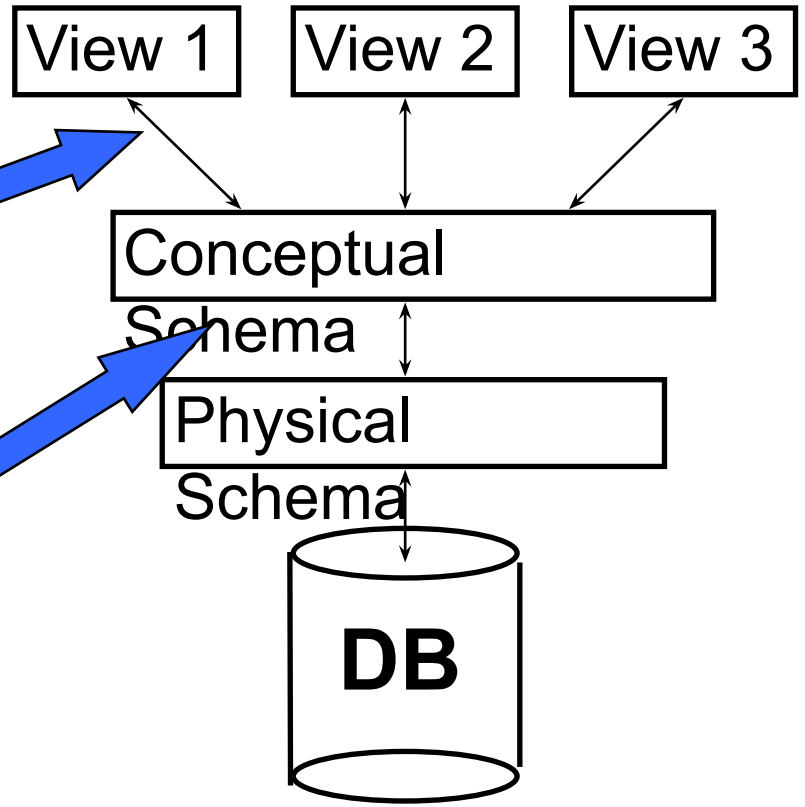
ANSI/SPARC Model

- **Views describe how users see the data.**
- **Conceptual schema defines logical structure**
- **Physical schema describes the files and indexes used.**



Data Independence: Two Flavors

- A Simple Idea: Applications should be insulated from how data is structured and stored.
- Logical data independence: Protection from changes in *logical* structure of data.
- Physical data independence: Protection from changes in *physical* structure of data.



- Q: Why is this particularly important for DBMS? (compared to your favorite programming language)

Example: University Database

- **Conceptual schema:**

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

  Courses(cid*: string*, cname*:string*, credits*:integer*)

  Enrolled(sid*:string*, cid*:string*, grade*:string*)

  Note: fields high-lighted in green are unique keys or "primary keys"

e.g.: An Instance of Students Relation

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

Example: University Database

- **Conceptual schema:**

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

  Courses(cid: string, cname:string, credits:integer)

  Enrolled(sid:string, cid:string, grade:string)

- **Physical schema:**
  - Relations stored as unordered files
  - Index on first column of Students, first 2 cols of Enrolled
- **External Schema (View):**

  Course_info(cid: string, enrollment: integer)

```
CREATE VIEW Course_info AS
SELECT cid, Count (*) as enrollment
FROM Enrolled
GROUP BY cid
```

What is a DBMS?

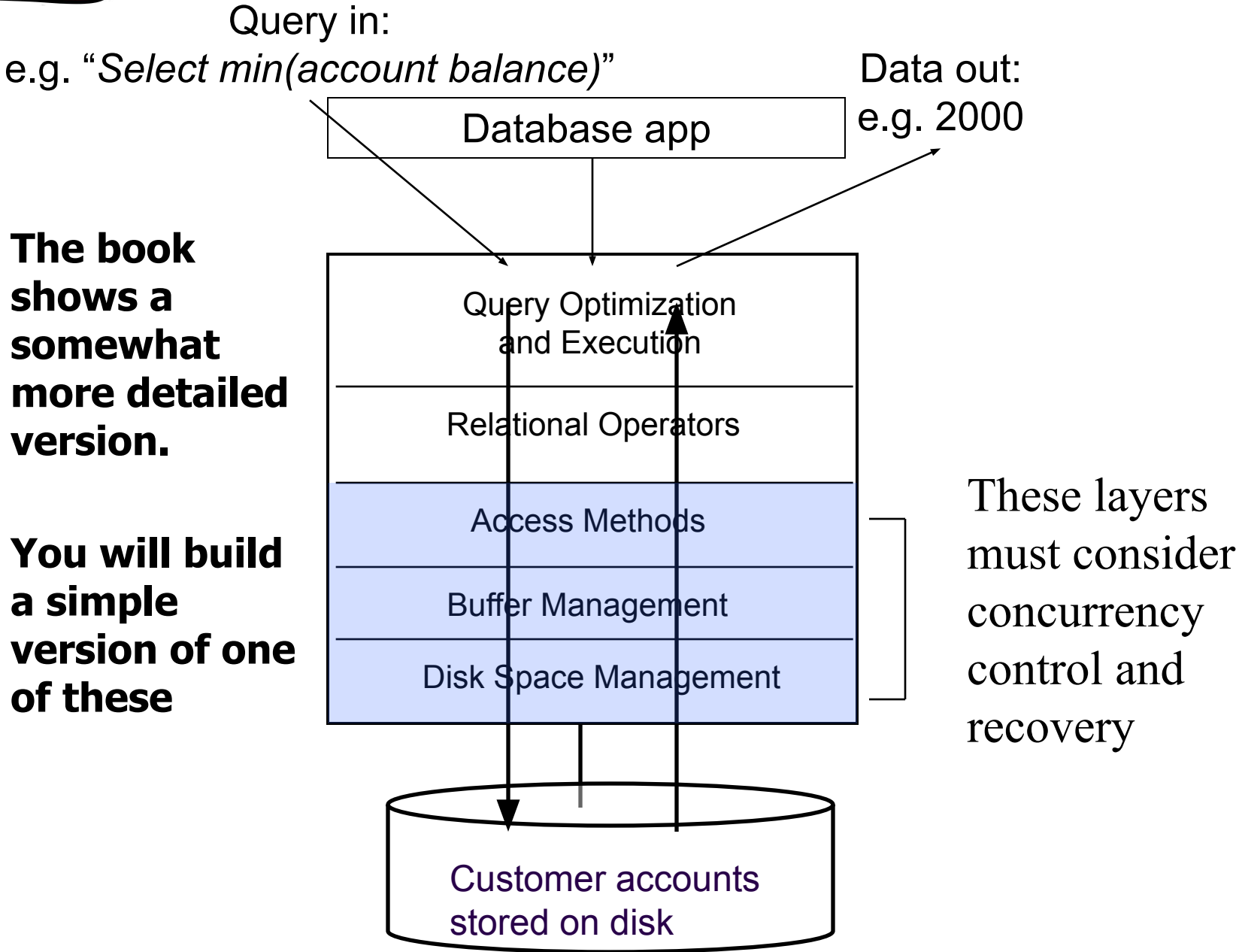
A *database* is an integrated and organized collection of data

A *Database Management System (DBMS)* is software that stores, manages and/or facilitates access to databases.

A DBMS Provides Users with the following:

- **"Declarative" Queries & Data Independence**
  - Say what you want, not how to get it
- **Help avoiding data corruption**
- **Protection from other users/jobs/queries**
  - As if you are the only person accessing the DB
- **Fault Tolerance and Durability**
  - Database is protected even if failures occur in the middle of processing
- Usually a bunch of **tools and interfaces for building applications**

A DBMS "Lasagna" Diagram

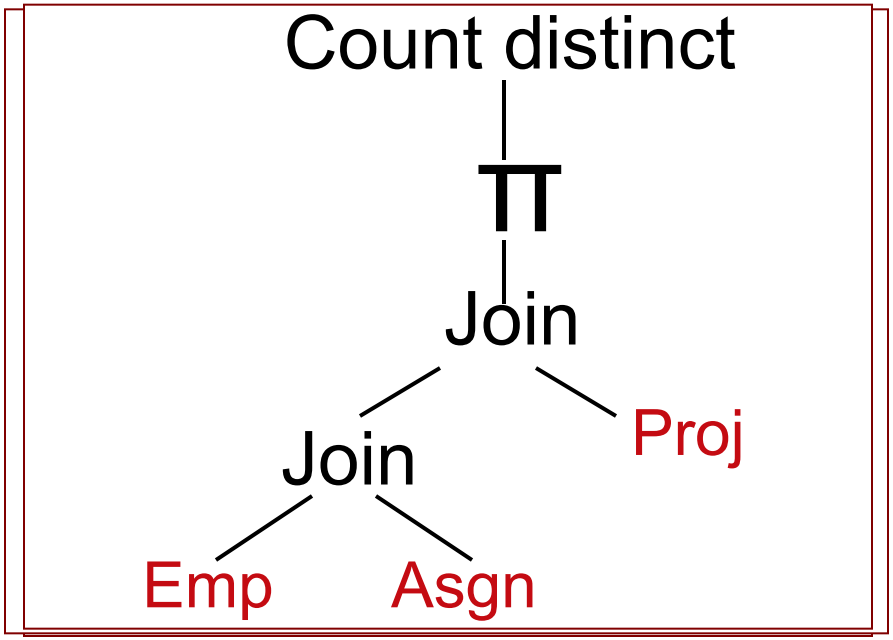


- **The book shows a somewhat more detailed version.**
- **You will build a simple version of one of these**

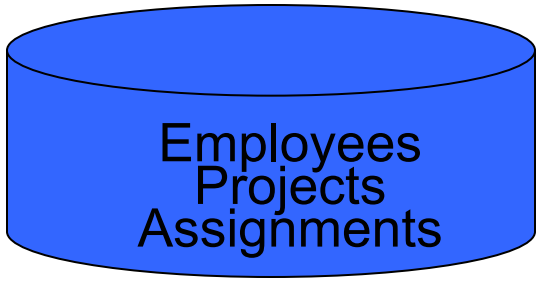
These layers must consider concurrency control and recovery

Key Concepts: Queries, Query Plans, and Operators

```
SELECT
    COUNT DISTINCT (E.eid)
FROM Emp E, Proj P, Asgn A
WHERE E.eid = A.eid
    AND P.pid = A.pid
    AND E.loc <> P.loc
```



System handles query plan generation & optimization; ensures correct execution.



Issues: view reconciliation, operator ordering, physical operator choice, memory management, access path (index) use, …

Plan for Today

- Why Study Databases?
- What are Databases and DBMSs?
- **Overview of the Course**
- Introduction to SQL

What will we learn?

- Design patterns for dealing with (Big) Data
- When, why and how to structure your data
- How Oracle and (a bit of) Google work
- SQL ... and (a bit of) noSQL
- Managing concurrency
- Fault tolerance and Recovery

- Useful concepts for Computer Science in general
  - and other sciences and endeavors as well!

Who?

- **Instructor**
  - Prof. Michael Franklin (franklin@cs)
- **TAs**
  - Lu Cheng[+]
  - Daniel Haas[#]
  - Evan Sparks[#]
  - Liwen Sun*[#]
  - Victor Zhu[+]

[*] Veteran CS 186 GSI

[+] Former Star CS 186 Student (Undergrad)

[#] Database Grad Student, AMPLab Member

How? Workload

- **A <u>New Set of Projects</u>:**
  - "SimpleDB" projects from MIT/UW
  - Done individually or in pairs
  - Java-based implementations of key DBMS functions
  - 5 project phases:
    - Files, Operators, Optimizer, Transactions, Recovery
- **Short weekly quizzes called "bunnies" (so as to be less scary – unless your aMonty Python fan)**
- **Some Homeworks on SQL, Database Design, etc.**
  - Likely using SQLite
- **Exams – 1 Midterm & 1 Final**
- **1 Additional Project if you are in 286a (TBD)**

How? Administrivia

- http://inst.eecs.berkeley.edu/~cs186
- *or* tinyurl.com/cs186fall2013
- (site under construction)
- **Lecture notes will be posted (usually before lecture)**

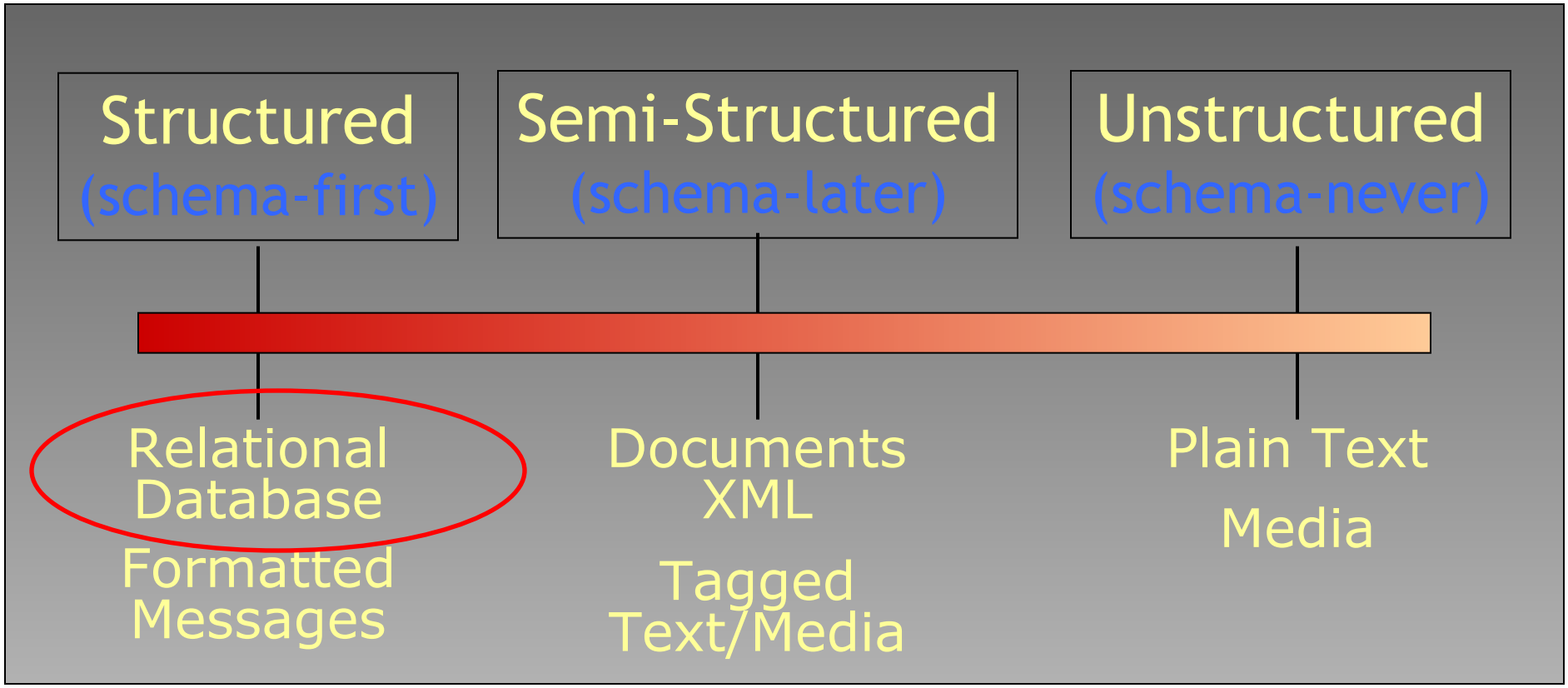
- We will be using Piazza for most communication,

- Office Hours: Prof. Franklin M 11-12, Th 3-4 pm in 449 Soda Hall
- TAs hours TBD
- Sections start *next* week

Plan for Today

- Why Study Databases?
- What are Databases and DBMSs?
- Overview of the Course
- **Introduction to SQL**

The Structure Spectrum



The Relational Model

- **The Relational Model is Ubiquitous**
  - MySQL, PostgreSQL, Oracle, DB2, SQLServer, …l
  - Foundational work done at
    - IBM Santa Teresa Labs (now IBM Almaden in SJ) – "System R"
    - UC Berkeley CS – the "Ingres" System
  - Note: some Legacy systems use older models
    - e.g., IBM's IMS
- **Object-oriented concepts have been merged in**
    - Early work: POSTGRES research project at Berkeley
    - Informix, IBM DB2, Oracle 8i
- **As has support for XML (semi-structured data)**

An Aside:

Q: In which Year did each of the following happen?

a) First man to walk on the moon.

b) Woodstock.

c) Relational Model of data management first proposed.

d) Cal last went to the Rose Bowl.

Relational Database: Definitions

- *Relational database:* a set of *relations*
- *Relation:* made up of 2 parts:

  Schema : specifies name of relation, plus name and type of each column

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

  Instance : the actual data at a given time
  - #rows = *cardinality*
  - #fields = *degree / arity*

Some Synonyms

Formal	Not-so-formal 1	Not-so-formal 2
Relation	Table	
Tuple	Row	Record
Attribute	Column	Field
Domain	Type	

Ex: Instance of Students Relation

sid	nam	login	ag	gp
53 66 6	Jone	jone @ s	1 8	3. 4
53 88 6	Smith	smith@ ec	1 8	3. 2
53 50 6	Smith	smith @ ath m	1 9	3. 8

- Cardinality = 3, arity = 5 , all rows distinct
- Do all values in each column of a relation instance have to be unique?

SQL - A language for Relational DBs

- **Say: "ess-cue-ell" or "sequel"**
  - But spelled "SQL"
- **Data Definition Language (DDL)**
  - create, modify, delete relations
  - specify constraints
  - administer users, security, etc.
- **Data Manipulation Language (DML)**
  - Specify queries to find tuples that satisfy criteria
  - add, modify, remove tuples
- **The DBMS is responsible for efficient evaluation.**

The SQL Query Language

- The most widely used relational query language.
- Originally IBM, then ANSI in 1986
- **Current standard is SQL-2011**
  - 2008 added x-query stuff, new triggers,…
  - 2003 was last major update: XML, window functions, sequences, auto-generated IDs. Not fully supported yet
- SQL-1999 Introduced "Object-Relational" concepts.
  - Also not fully supported yet.
- **SQL92 is a basic subset**
  - Most systems support at least this
- PostgreSQL has some "unique" aspects (as do most systems).
- SQL is not synonymous with Microsoft's "SQL Server"

Creating Relations in SQL

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

```
CREATE TABLE
Students
  (sid CHAR(20),
   name CHAR(20),
   login CHAR(10),
   age INTEGER,
   gpa FLOAT)
```

Table Creation (continued)

- **Another example: the Enrolled table holds information about courses students take.**

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

Adding and Deleting Tuples

- **Can insert a single tuple using:**

```
INSERT INTO Students (sid, name, login, age, gpa)
 VALUES ('53688', 'Smith', 'smith@ee', 18, 3.2)
```

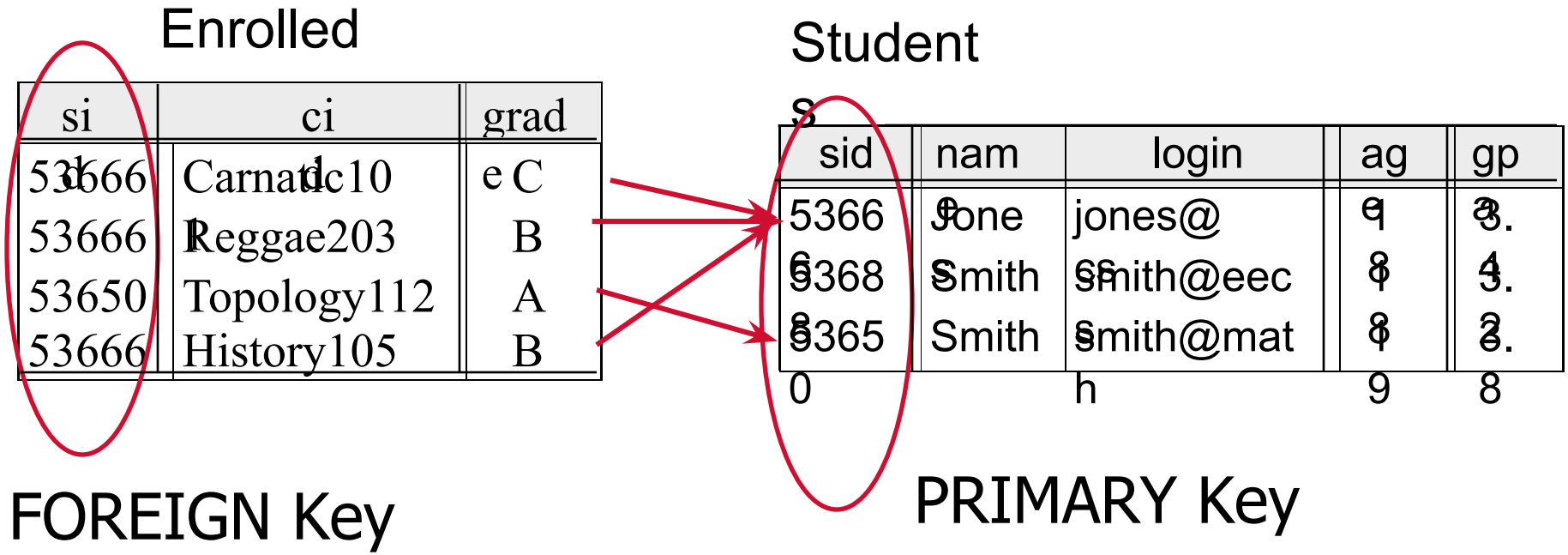
- **Can delete all tuples satisfying some condition (e.g., name = Smith):**

```
DELETE
  FROM Students S
 WHERE S.name = 'Smith'
```

Powerful variants of these commands are available; more later!

Keys

- **Keys are a way to associate tuples in different relations**
- **Keys are one form of integrity constraint (IC)**



Primary Keys

- **A set of fields is a superkey if:**
  - No two distinct tuples can have same values in all key fields
- **A set of fields is a key for a relation if :**
  - It is a superkey
  - No subset of the fields is a superkey
- **what if >1 key for a relation?**
  - One of the keys is chosen (by DBA) to be the primary key. Other keys are called candidate keys.
- **E.g.**
  - sid is a key for Students.
  - What about name?
  - The set {sid, gpa} is a superkey.

Primary and Candidate Keys in SQL

- **Possibly many *candidate keys* (specified using UNIQUE), one of which is chosen as the *primary key*.**

- Keys must be used carefully!
- "For a given student and course, there is a single grade."

```
CREATE TABLE
Enrolled
  (sid CHAR(20)
   cid  CHAR(20),
   grade CHAR(2),
   PRIMARY KEY
(sid,cid))
```

VS.

```
CREATE TABLE
Enrolled
   (sid CHAR(20)
    cid  CHAR(20),
    grade CHAR(2),
    PRIMARY KEY
(sid),
    UNIQUE (cid,
    grade))
```

"Students can take only one course, and no two students in a course receive the same grade."

Foreign Keys, Referential Integrity

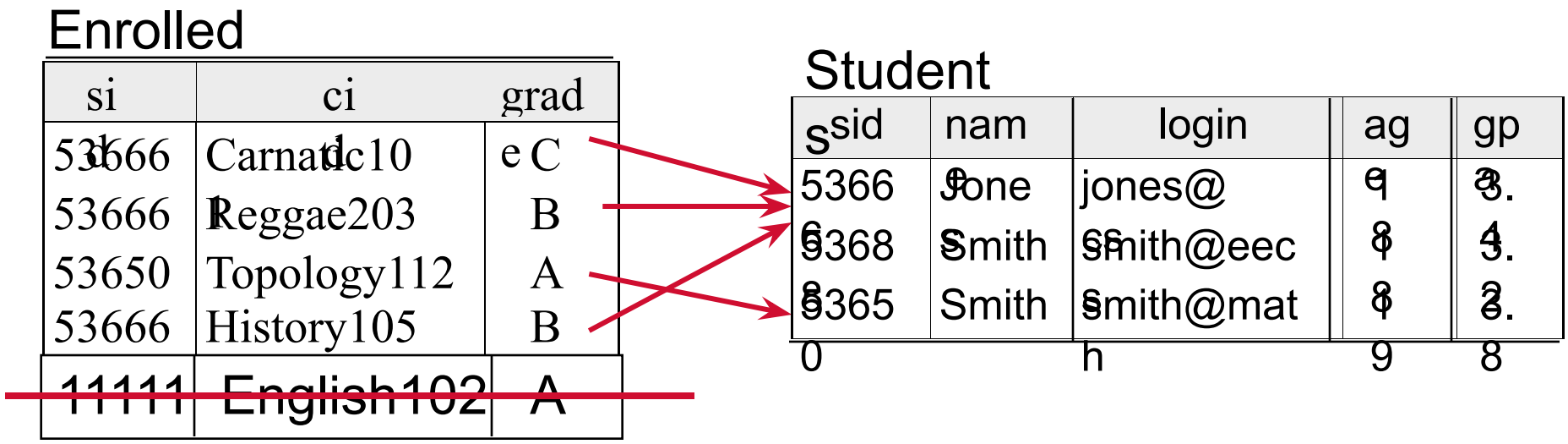
- ***Foreign key*: a "logical pointer"**
  - Set of fields in a tuple in one relation that `refer' to a tuple in another relation.
  - Reference to *primary key* of the other relation.

- **All foreign key constraints enforced?**
  - *referential integrity*!
  - i.e., no dangling references.

Foreign Keys in SQL

- **E.g. Only students listed in the Students relation should be allowed to enroll for courses.**
  - *sid* is a foreign key referring to Students:

```
CREATE TABLE Enrolled
(sid CHAR(20),cid CHAR(20),grade CHAR(2),
 PRIMARY KEY (sid,cid),
 FOREIGN KEY (sid) REFERENCES Students);
```



Next Up

- **We'll talk a bit about the SQL DML**
- **Then we'll start describing the DBMS from storage on up**