Welcome to

Comp 115: Databases

http://www.cs.tufts.edu/comp/115/

Instructor: Manos Athanassoulis

gmail: manos@cs.tufts.edu
Today

big data

data-driven world

databases & database systems

when you see this, I want you to speak up!
[and you can always interrupt me]

no smartphones

no laptop
Big Data

marketing term ...

but ...

science / government / business / personal data

exponentially growing data collections

So, it is all good!
How big is “Big”?  

Every day, we create 2.5 exabytes* of data — 90% of the data in the world today has been created in the last two years alone.  

[Understanding Big Data, IBM]  

*exabyte = $10^9$ GB
Using Big Data

- experimental physics (IceCube, CERN)
- biology
- neuroscience

- data mining business datasets
- machine learning for corporate and consumer

- data analysis for fighting crime

... are only some examples
Data-Driven World

Big Data V’s

Volume

Velocity

Variety

Veracity

Information is transforming traditional business.

[“Data, data everywhere”, Economist]
Data-Driven World

Reporting
Logging
Transactions
Business Analysis

Discovery
Exploration
Data-to-Insight
Automated Decisions

Behind all these: use & manage data
Comp 115

we live in a data-driven world

Comp115 is about the basics for storing, using, and managing data
your lecturer (that’s me!)

Manos Athanassoulis
name in greek: Μάνος Αθανασούλης

grew up in Greece
enjoys playing basketball and the sea

BSc and MSc @ University of Athens, Greece
PhD @ EPFL, Switzerland
Research Intern @ IBM Research Watson, NY
Postdoc @ Harvard University

some awards:
SNSF Postdoc Mobility Fellowship
IBM PhD Fellowship

http://manos.athanassoulis.net
Office: Halligan Hall 228B
Office Hours: M/W after class

Myrtos, Kefalonia, Greece
your awesome TAs

Elif
Sam
Deanna
Taus
your awesome head TA

Sam Lasser
grad Student in PL

ta115@cs.tufts.edu
Data

to make data usable and manageable we organize them in collections
Databases

a large, integrated, *structured* collection of data

intended to model some *real-world* enterprise

**Examples**: a university, a company, social media

University: students, professors, courses
  what is missing?
  -- how to connect these?
  -- enrollment, teaching

What about a company? What about social media?
Database Systems

a.k.a. database management systems (DBMS)
a.k.a. data systems

Sophisticated pieces of software... ... which store, manage, organize, and facilitate access to my databases...

... so I can do things (and ask questions) that are otherwise hard or impossible
“relational databases are the foundation of western civilization”

Bruce Lindsay, IBM Research
ACM SIGMOD Edgar F. Codd Innovations award 2012
Ok but what really IS a database system?

Is the WWW a DBMS?

Is a File System a DBMS?

Is Facebook a DBMS?
Is the WWW a DBMS?  

Fairly sophisticated search available  
web crawler *indexes* pages for fast search

.. but 

data is *unstructured* and *untyped* 
no will-defined "correct answer"  
cannot update the data 

freshness? consistency? fault tolerance?  

web sites **use** a *DBMS* to provide these functions  
e.g., amazon.com *(Oracle)*, facebook.com *(MySQL and others)*

*Not really!*
“Search” vs. Query

What if you wanted to find out which actors donated to Barrack Obama’s presidential campaign 8 years ago?

Try “actors donated to obama” in your favorite search engine.
“Search” vs. Query

“Search” can return only what’s been “stored”

E.g., best match at Google:
A “Database Query” Approach

where can we find data for ”all actors”?

where can we find data for ”all donations”?
A “Database Query” Approach

16,684-16,733 of 1,865,455 names.

Sort by: STARmeter | A-Z | Height | Birth Date | Death Date

16684. Adam Sandler
Producer, Grown Ups
Adam Sandler was born on September 1, 1966, in Stanley Sandler. At 17, he took himself to the University of Massachusetts, where he studied theater and film. His early years were marked by several natural comic. He nurtured his talent while working in a New York City improv troupe.

16685. Adam Sandler
Producer, Episode #38.2

16686. Adam Sandoval
Actor, Unspeakable

16687. Adam Sandroni
Actor, Joey's Girl

Your search has generated too many results. Only the top 1000 records are being displayed. If you would like to refine your search, return to the form page.

<table>
<thead>
<tr>
<th>Candidate</th>
<th>Contributor</th>
<th>Employer</th>
<th>Date</th>
<th>Amount</th>
</tr>
</thead>
<tbody>
<tr>
<td>Obama, Barack</td>
<td>Budinger, William</td>
<td>Not employed</td>
<td>7/31/08</td>
<td>$30,800</td>
</tr>
<tr>
<td>Obama, Barack</td>
<td>BOSLER, JAMES FORT WORTH, TX 76126</td>
<td>NOT EMPLOYED/RETIERED</td>
<td>8/28/08</td>
<td>$28,500</td>
</tr>
<tr>
<td>Obama, Barack</td>
<td>HIGDON, JOE WASHINGTON,DC 20008</td>
<td>NOT EMPLOYED/RETIERED</td>
<td>8/28/08</td>
<td>$28,500</td>
</tr>
<tr>
<td>Obama, Barack</td>
<td>MYERS, DEBRA RANCHO PALOS VERDE,CA 90725</td>
<td>SELF EMPLOYED/PHYSICIAN</td>
<td>8/31/08</td>
<td>$28,500</td>
</tr>
<tr>
<td>Obama, Barack</td>
<td>WOODROW DR JR INDIANAPOLIS,IN 46204</td>
<td>MYERS VENTURES LLC/MGR DIRECTOR</td>
<td>8/31/08</td>
<td>$28,500</td>
</tr>
</tbody>
</table>

www.imdb.com/search/name?gender=ma...
<table>
<thead>
<tr>
<th>Contributor</th>
<th>Employer</th>
<th>Date</th>
<th>Amount</th>
</tr>
</thead>
<tbody>
<tr>
<td>ROCK, CHRIS MR NEW YORK, NY 10019</td>
<td>ACTOR</td>
<td>4/20/07</td>
<td>$9,200</td>
</tr>
<tr>
<td>DOUGLAS, MICHAEL UNIVERSAL CITY, CA 91608</td>
<td>ACTOR/ PRODUCER</td>
<td>3/30/07</td>
<td>$4,600</td>
</tr>
<tr>
<td>DOUGLAS, MICHAEL UNIVERSAL CITY, CA 91608</td>
<td>ACTOR/ PRODUCER</td>
<td>3/30/07</td>
<td>$2,300</td>
</tr>
<tr>
<td>ROCK, CHRIS MR NEW YORK, NY 10019</td>
<td>ACTOR</td>
<td>4/20/07</td>
<td>$2,300</td>
</tr>
<tr>
<td>CARIDES, GEORGIA NEW YORK, NY 10017</td>
<td>ACTOR</td>
<td>5/18/07</td>
<td>$1,000</td>
</tr>
<tr>
<td>CARTER COVINGTON, CLAUDIA ACTORS THEATRE PART TIME/ACTOR/NEW CHARLOTTE, NC 28207</td>
<td>ACTOR/PRODUCER/NEW CHARLOTTE, NC 28207</td>
<td>5/20/08</td>
<td>$1,000</td>
</tr>
<tr>
<td>FOX, RICK ENCINO, CA 91316</td>
<td>ACTOR/PRODUCER</td>
<td>6/16/08</td>
<td>$1,000</td>
</tr>
<tr>
<td>HILDRETH, THOMAS W LOS ANGELES, CA 90068</td>
<td>ACTOR</td>
<td>9/29/08</td>
<td>$1,000</td>
</tr>
<tr>
<td>RENNER, CARL BEVERLY HILLS, CA 90210</td>
<td>ACTOR/BESSONE@ROADRUNNER.COM</td>
<td>8/28/08</td>
<td>$1,000</td>
</tr>
<tr>
<td>SIMMONS, HENRY WEST HOLLYWOOD, CA 90046</td>
<td>ACTOR</td>
<td>6/4/07</td>
<td>$1,000</td>
</tr>
</tbody>
</table>
Is a File System a DBMS?

Not really!

Thought Experiment 1:

– You and your project partner are editing the same file.
– You both save it at the same time.
– Whose changes survive?

A) Yours  B) Partner’s  C) Both  D) Neither  E) ???

Thought Experiment 2:

– You’re updating a file.
– The power goes out.
– Which of your changes survive?

A) All  B) None  C) All Since last save  D) ???
Is Facebook a DBMS?

Is the data structured & typed?

Does it offer well-defined queries?

Does it offer properties like “durability” and “consistency”?

Facebook is a data-driven company that uses several database systems (>10) for different use-cases (internal or external).
Why take this class?

computation to information

corporate, personal (web), science (big data)

database systems everywhere

data-driven world, data companies

DBMS: much of CS as a practical discipline

languages, theory, OS, logic, architecture, HW
Comp 115 in a nutshell

**model**

data representation model

**query**

query languages – ad hoc queries

**access** (concurrently multiple reads/writes)

ensure *transactional* semantics

**store** (reliably)

maintain *consistency/semantics* in *failures*
A “free taste” of the class

data modeling
query languages
concurrent, fault-tolerant data management
DBMS architecture

Coming in next class
Discussion on database systems designs
Components of a “classic” DBMS

DBMS: a set of cooperating software modules
Describing Data: Data Models

**data model**: a collection of concepts describing data

**relational model** is the most widely used model today

**key concepts**

**relation**: basically a table with rows and columns

**schema**: describes the columns (or fields) of each table
Schema of “University” Database

**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
Levels of Abstraction

what the users see

what is the *data model*

how the data is *physically* stored

<table>
<thead>
<tr>
<th>Physical Schema</th>
<th>Conceptual Schema</th>
<th>External Schema 1</th>
<th>External Schema 2</th>
</tr>
</thead>
</table>

e.g., files, indexes
Schemata of “University” Database

Conceptual Schema

Students

\[ \text{sid: string, name: string, login: string, age: integer, gpa: real} \]

Courses

\[ \text{cid: string, cname: string, credits: integer} \]

Enrolled

\[ \text{sid: string, cid: string, grade: string} \]

Physical Schema

relations stored in heap files
indexes for sid/cid
Schemata of “University” Database

External Schema
   a “view” of data that can be derived from the existing data

example: Course Info
   Course_Info (cid: string, enrollment: integer)
Data Independence

Abstraction offers “application independence”

**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?

Applications can treat DBMS as black boxes!
Queries

”Bring me all students with gpa more than 3.0”

“SELECT * FROM Students WHERE gpa>3.0”

SQL – a powerful declarative query language

treats DBMS as a black box

What if we have multiples accesses?
Concurrency Control

multiple users/apps

Challenges

how frequent access to slow medium

how to keep CPU busy

how to avoid short jobs waiting behind long ones

e.g., ATM withdrawal while summing all balances

interleaving actions of different programs
Concurrency Control

Problems with *interleaving* actions of different programs

Bad interleaving:
- Savings -= 100
- Print balances
- Checking += 100

Printout is missing 100$!
Concurrency Control

Problems with *interleaving* actions of different programs

What is a correct interleaving?

- Savings $\rightarrow -$ 100
- Checking $\rightarrow +$ 100
- Print balances

How to achieve this interleaving?
Scheduling Transactions

Transactions: atomic sequences of Reads & Writes

\[ T_{\text{Bill}} = \{ R_{1\text{ savings}}, R_{1\text{ checking}}, W_{1\text{ savings}}, W_{1\text{ checking}} \} \]
\[ T_{\text{Alice}} = \{ R_{2\text{ savings}}, R_{2\text{ checking}} \} \]

How to avoid previous problems?
Scheduling Transactions

All interleaved executions equivalent to a **serial**

All actions of a transaction executed **as a whole**

How to achieve one of these?
Locking

before an object is accessed a lock is requested
Locking

before an object is accessed a lock is requested
before an object is accessed a lock is requested
Locking

locks are held until the end of the transaction

[this is only one way to do this, called “strict two-phase locking”]
Locking

\[ T_1 = \{ R_{1\text{Savings}}, R_{1\text{Checking}}, W_{1\text{Savings}}, W_{1\text{Checking}} \} \]

\[ T_2 = \{ R_{2\text{Savings}}, R_{2\text{Checking}} \} \]

Both should lock Savings and Checking

What happens:

if \( T_1 \) locks Savings & Checking ?
\( T_2 \) has to wait

if \( T_1 \) locks Savings & \( T_2 \) locks Checking ?
we have a deadlock
How to solve deadlocks?

we need a mechanism to *undo*

also when a transaction is *incomplete*

e.g., *due to a crash*

*what can be an undo* mechanism?

*log every action before it is applied!*
Transactional Semantics

Transaction: one execution of a user program

multiple executions $\rightarrow$ multiple transactions

Every transaction:

- **Atomic**: executed entirely or not at all
- **Consistent**: leaves DB in a consistent state
- **Isolated**: as if it is executed alone
- **Durable**: once completed is never lost
Transactional Semantics

Transaction: one execution of a user program
multiple executions $\rightarrow$ multiple transactions

Every transaction:

- **Atomic**  “executed entirely or not at all”
- **Consistent**  “leaves DB in a consistent state”
- **Isolated**  “as if it is executed alone”
- **Durable**  “once completed is never lost”
Who else needs transactions?

- lots of data
- lots of users
- frequent updates
- background game analytics

Scaling games to epic proportions,
by W. White, A. Demers, C. Koch, J. Gehrke and R. Rajagopalan
ACM SIGMOD International Conference on Management of Data, 2007
Only “classic” DBMS?

No, there is much more!

NoSQL & Key-Value Stores: No transactions, focus on queries
Graph Stores
Querying raw data without loading/integrating costs
Database queries in large datacenters
New hardware and storage devices

... many exciting open problems!
Comp 115: Databases

Database Systems Architectures
Class administrativia
Class project administrativia
Additional Accommodations

If you require additional accommodations please contact the Student Accessibility Services office at Accessibility@tufts.edu or 617-627-4539 to make an appointment with an SAS representative to determine which are the appropriate accommodations for your case.

Please be aware that accommodations cannot be enacted retroactively, making timeliness a critical aspect for their provision.

More details about accessibility services in the syllabus:
http://www.cs.tufts.edu/comp/115/syllabus.html