Department of Computer Science
University of California, Los Angeles

Computer Science 143: Introduction to Database Systems

Ryan R. Rosario
rrosario@cs.ucla.edu

Lecture 1: April 2, 2018

Outline

1. Syllabus / Logistics
2. About Me
3. Course Topics
4. Careers in Data(bases)
5. Lecture 1

Syllabus

Your very own copy of the syllabus is now circulating through the room.

There should be three pages of text.

It is also posted on CCLE. (Sorry, trees...)
Logistics: PTE

The cap for this class was raised about a month back. It is currently waitlisted. The Department of Computer Science has policies regarding who gets priority to enroll in the course. I must admit undergraduate computer science majors first (particularly those graduating).

No decisions on PTEs will be made until the end of the second week.

If you want a PTE, you will need to attend (and sign in to) every lecture and every recitation section for the first two weeks and submit the first assignment (TBA).

I do not mind having more students, but I am also responsible for:

1. Not exceeding the capacity of this room;
2. Not exceeding the capacity of the rooms used for recitations;
3. The sanity of our TAs, who have a lot of other responsibilities such as research and their own coursework.

So we will try to accommodate as many people as we can, but there are no guarantees. I will be judicious because of the PTEs I have given out so far, 90% dropped before the quarter started, and that is unfair to others that tried to enroll.

My Contact Information

My email is rrosario@cs.ucla.edu. Please use this email.

My office is Boelter Hall 3531C.

My office hours will be Monday and Wednesday from 1pm-2pm but are subject to change as I share the office with others.
Our TAs

We have four TAs this quarter.
- Debleena Sengupta (1A)
- Jiaqi Gu (1B)
- Jin Wang (1C)
- Sonali Garg (1D)

Get to know them, even the TAs that are not for your section. They will be assisting you with projects, homework etc. and are good people to know particularly if you are interested in databases and/or data mining.

Lecture Format

Similar to other CS classes, we will take a 15 minute break in the middle (on average).

<table>
<thead>
<tr>
<th>Approx Time</th>
<th>What We Do</th>
</tr>
</thead>
<tbody>
<tr>
<td>8:00-8:50ish</td>
<td>Part 1</td>
</tr>
<tr>
<td>8:50-9:05ish</td>
<td>BREAK</td>
</tr>
<tr>
<td>9:05-9:50</td>
<td>Part 2</td>
</tr>
</tbody>
</table>

Whenever I use slides, I will post them.
Grades

Your grade will be based on how well you perform compared to the rest of the class, and the final grade cutoffs for the class will be developed based on how I feel the class performed compared to my expectations.

<table>
<thead>
<tr>
<th>Activity</th>
<th>Pct</th>
</tr>
</thead>
<tbody>
<tr>
<td>Homework</td>
<td>10%</td>
</tr>
<tr>
<td>Project 1</td>
<td>15%</td>
</tr>
<tr>
<td>Project 2</td>
<td>15%</td>
</tr>
<tr>
<td>Midterm</td>
<td>25%</td>
</tr>
<tr>
<td>Final Exam</td>
<td>35%</td>
</tr>
</tbody>
</table>

Homework

There will be homework assignments throughout the quarter. **Read the directions carefully about how to submit your homework as it may vary depending on the format of the assignment.**

- A mix of conceptual questions from the book or myself;
- I will focus on SQL where necessary;
- These should be treated as interview prep and exam prep;
- Certain exercises will be graded, but “participation” will be taken into account.
- For SQL exercises, you will work with **REAL** data, and I am trying to pick interesting datasets.

First assignment will come on Wednesday or Friday this week and be due the following week. More info to follow.

Midterm

**May 9, 2018**

Plan on:

- No book.
- No notes.
- No cheat sheet.
- No computer or Internet.
- Yes to engaged minds.
Final Exam

June 13, 2018, 3pm-6pm, Room TBD

This gives Seniors plenty of daylight to get into the Inverted Fountain...

The final exam will be **cumulative** and **comprehensive** with some emphasis on the material since the midterm.

Projects

Finally, there will be two projects this quarter. The purpose of the projects is to integrate the course material into something that you can implement, and to show how these concepts are used in real life.

Project 1

In the first project, you will create a website using PHP/HTML/CSS and a database system called MySQL about movies, actors and their reviews.

This will be due around week 4. More specific information to follow.
Project 1

As a heads up, the TAs and I will provide you with a VirtualBox image containing a typical LAMP Stack (Linux/Apache/MySQL/PHP). The TAs can help with setting it up and troubleshooting any issues.

VirtualBox allows you to run a separate operating system on your own system without rebooting.

For project 2, we may either provide instructions for installing additional software, or provide you with a new image.

Project 2

Be very careful selecting your project partners.
- Both students get the same grade;
- Each partnership makes one submission with both names.
Project Partners

There will be no exceptions made for these:

- EVERY GROUP PROJECT
- IN SCHOOL YOU HAVE EVER DONE

Project Partners?

Of course you can work alone...

The Zaniolo Rule

You can break up into two individuals:

\[ r = t_0 \]
\[ r \in [t_0, t]\n
Or two solos can combine into a partnership:

\[ t = t_0 \]
\[ t \in [t_0, T]\n
But once you drop a partner, you cannot get another one:
Project Partners

This rule avoids situations like this:

Late Projects

You (and your partner) will have four grace period days to use for the projects throughout the quarter.

- You can use these days for any submission.
- A maximum of 2 per project.
- 1 hour late = 1 day late.
- Any additional days late will result in half credit or no credit at all.

Textbook

There is a textbook for this course.

*Database System Concepts, Sixth Edition* by Silberschatz et. al.

Previous editions may also work, but we cannot guarantee it. On reserve at SEL.

The authors really like sailboats...

The authors really like sailboats... This is the fifth edition, not the sixth.
Textbook

The authors really like sailboats...

This is the fourth edition, not the sixth.

Other Good Books

You don’t need to buy these, but they may be helpful whether for 143, or for your job.

1. *Database Systems: The Complete Book*
   Hector Garcia-Molina
   Jeffrey D. Ullman
   Jennifer Widom

2. *The Art of SQL*
   Stephanie Faroult
   Peter Robson

3. *Head First SQL*
   Lynn Beighley
Other Good Books

You don’t need to buy these, but they may be helpful whether for 143, or for your job.

Database Design for Mere Mortals
Michael J. Hernandez

SQL Queries for Mere Mortals
John L. Viescas
Michael J. Hernandez

Prerequisites

Computer Science 111 (Operating Systems) is an enforced pre-requisite for this course. As such, I must prioritize students that have completed CS 111 when dealing with wait lists and PTEs. You should

1. Be comfortable at the Unix/Linux command line;
2. Be a “competent” programmer with the ability to learn small parts of other languages when necessary. For example, some past CS 143 classes have used Java for their projects, some have used Scala, some have used C++. (BTW, Project 1 will use PHP)
3. Be able to review (or quickly learn) the relevant material from CS 111 (interprocess communication, resource sharing).

We do not assume any prior knowledge of database systems.

Academic Integrity

The UCLA Department of Computer Science and the instructor both treat academic integrity very seriously. The instructor absolutely does not and will not tolerate any form of academic dishonesty. Students that are suspected of violating UCLA’s academic integrity policies will be referred to the Office of the Dean of Students. If a student is ever in the position where they are unsure if they may be about to partake in behavior that could be deemed dishonest, they should ask the instructor.
Academic Integrity: Particularly for CS

There are additional things to be aware of specifically for CS classes.

1. Clarifying with other students the requirements of the assignment is OK.
2. Discussing algorithms for solving a problem, without code, is OK.
3. Helping someone find a minor bug in their code is OK.
4. Using code provided in the required textbook, from the instructor, from the TAs, or from reference materials provided in the class is OK.

Academic Integrity Agreement

To receive credit for your work, you will need to sign and return the CS 143 Academic Integrity Agreement being circulated. You can return it to me, or your TA. You can also scan it and upload it to CCLE (under Week 1). Read both sides/pages.
whoami

UCLA
Ph.D., Statistics
M.S., Computer Science
B.S., Math of Computation, Statistics

Southern California native.
Originally from Thousand Oaks, CA.

Research Interests

Machine Learning
Natural Language Processing
Web Mining/ Web Research
Psychometrics

I am From Industry

Some places I’ve worked:

Facebook
Amazon
Rubicon
GumGum

as well as consulting and a few ad startups in LA:
What I Did at Facebook: Machine Learning Engineer

- Developed the company-wide sentiment analysis system.
- Developed a system for topic identification and topic summarization.
- Worked on automated tagging of IT tickets.
- Worked with HR’s People Analytics team using classical statistics to analyze inequities in performance review text and leveling by demographics.

You can learn more about what I did in this PyData 2014 Silicon Valley talk:

https://www.youtube.com/watch?v=y3ZTKFZ-1QQ

Purpose of this Course

My goal is to make CS 143 practical and applicable for your careers as Software Developers and Computer Scientists. This means I intend to cover additional topics (as time allows) that are not usually included. The workload should remain the same however.
Purpose of this Course

The purpose of CS 143 is to introduce students to database systems, particularly relational database systems. The class will be roughly divided into these parts (as time allows):

- Part 1: Introduction to the Relational Model
- Part 2: SQL and Database Design
- Part 3: Concurrency and Transactions
- Part 4: Other Relational DBs and Data Warehousing
- Part 5: Big Data Technologies (Hadoop/Spark)
- Part 6: NoSQL and Non-Relational Databases

Databases Where we Least Expect Them: Blockchain

*Blockchain* is similar to a distributed database, with one major difference.

With centralized databases, administrators have all of the power. With blockchain, each participant maintains their own data and updates to the database. All nodes in the system cooperate to make sure the database comes to the same conclusion – a form of security.

Databases Where we Least Expect Them: *git*

“Blockchains allow different parties that do not trust each other to share information without requiring a central administrator. Transactions are processed by a network of users acting as a consensus mechanism so that everyone is creating the same shared system of record simultaneously.”

Source: Coindesk
My Belief

It is my belief that every CS student graduates with an understanding of SQL and database technologies. **Data is not going anywhere.**

I’ve interviewed far too many graduates that cannot write SQL. It is not important for every interview, but you will not want to get caught off-guard.

Data Careers

Data is everywhere! We need people that are familiar with the storage and transmission of data, in addition to the processing and analysis of it.

Some of the common careers centering on data:

- Data Engineer
- Data Scientist
- Software Engineer (certain types)
- User Experience/Interaction Design/Visualization Engineer
- Database Administrator
- Coming Soon: Data Privacy and Security

People move around these all the time; nobody is stuck.

Data Engineer

A **Data Engineer** is typically somebody that collects raw data and gets it into some kind of database, store or warehouse. They also work on transforming and processing data while it is in transit from one database to another, one server to another, or one table to another. Some specialize in the interaction between a user and the data.

If it involves moving in some form, a Data Engineer probably does it. This career is broad, but better defined than some others.
Data Engineer I

Some key responsibilities:

- **Extracting** data from logs, JSON, XML and other structured and unstructured data into something that the DB can deal with (a table for relational DBs).
- **Transforming** data and **loading** it in proper places for the appropriate stakeholders.
- Add new infrastructure for existing users. For example, writing an R package or Python library for accessing.
- Plan the flow of data between production systems and read-only systems like warehouses.

Data Engineer II

- Developing APIs so that different roles within an org can access the same data in whichever workflow is most convenient (i.e. REST for on-demand, CSV for statisticians).
- Developing databases, data stores and warehouses (i.e. Presto, developed by data engineers at Facebook) from scratch.
- Monitoring uptime, debugging data integrity errors and backfilling lost data.

Data Scientist

One of the “hot” career choices currently, but also very poorly defined and often requires wearing many hats, or asking the right questions in interviews. They can be anywhere between data analysts and full-fledged software engineers.

Data Scientists are usually customers of databases, though they often create their own data and can do limited Data Engineer position work.

Some companies require a M.S. or Ph.D. for these roles because these degrees require research based on scientific method.
Data Scientist

The responsibilities of a Data Scientist include:

1. Designing and executing experiments and observational studies (i.e. Which ad performs better? How to test which search algorithm is better?)
2. Communicating with non-technical stakeholders.
3. Analyzing and modeling data to answer questions.
4. Extracting data using SQL or writing map-reduce jobs in Hadoop or Spark.
5. Sometimes developing algorithms (i.e. recommendation systems, classifiers).
7. 95% of the job is cleaning the data.

Any Computer Science major considering working as a Data Scientist should:

1. Consider grad school.
2. Take a wide variety of mathematics, statistics and probability classes.
3. Take a Machine Learning class (there are so many now... in 2006 there was 1).
4. Learn SQL (this class), R, Python.
5. Play with (or scrape, crawl, collect) many different datasets for fun.

User Experience/Visualization Engineer

Niche, and still quite rare is the Visualization Engineer. They typically

1. Extract data and create dashboards
2. Create novel user interfaces for slicing and dicing data used by technical and non-technical stakeholders.
3. Create easy-to-understand and natural visualizations and animations of data.
4. Have front-end or web experience (JavaScript/CSS/HTML5/PHP).
5. Are usually designers or artsy.
6. Are usually some combination of Stats, DESMA, Computer Science.
Database Administrator (DBA)/Consultant

Perhaps the most widely known classical databases position. Most DBAs have specialized training in one or a few database/data store/data warehouse systems (i.e. MySQL, Vertica) and know their ins and outs very well. They are so specialized that they typically find success as consultants:

1. Are fluent in specific variants of SQL and can optimize the heck out of slow queries.
2. Can be involved in the deployment of new data solutions.
3. Generally have very deep knowledge, but not as broad as Data Engineers.
4. Newer variants of this position include Hadoop/Spark Administrators.
5. Typically requires certifications either with or in lieu of CS degree.

Data Security Professionals

One specialization within CS that is going to pick up a lot of steam is the intersection between databases and security/cryptography.

Bad guys/gals lurk everywhere.

A data security issue occurs when an unauthorized party accesses, modifies or shares data.

Data Security Professionals

Our friends in CS 118 have developed several protocols to try to prevent data security issues resulting from violated assumptions about protocols, or nefarious actors:

1. **Secure HTTP (HTTPS)** creating a secure channel over an insecure network using encryption.
2. Wireless protocols such as **WPA2** prevent unauthorized access to networks.
3. **VPN (IPSEC etc.), SSH**, etc. encrypts data as it flows in both directions across an insecure network.
4. **DNSSEC** prevents bad actors from redirecting a user to a nefarious site.

And of course our friends in CS 136 that develop the encryption techniques. But more still needs to be done...

Source:
Data Security Professionals

Visualization of Data Breaches Over Time

Data Privacy Professionals

One more troubling and nuanced issue that is growing in importance is **privacy** and we need good guys/gals to help the cause.

While data security issues arise from *unauthorized* access to data, data privacy issues arise from mostly *authorized* access to data with some caveats:

- The victim may not know that they authorized data use or collection, or
- The victim does not agree with how the data is used.
- As a result, the victim feels powerless and exposed.

Both of these yield situations where victims believe their privacy has been violated.

Data Privacy Professionals

Data privacy is a field that is growing in important year-after-year.

The solution may be legislative and not-scientific. Or it may be scientific. We don’t know yet.

When designing applications that read or write private/personal data, it is critical that the data is secure, and that there are safeguards in place if others wish to retrieve that data.
Data Privacy Professionals

We all recently learned that a firm called Cambridge Analytica obtained data on 50 million Facebook users using a psychometric personality survey from another source.

Facebook for having stipulations against data retention that perhaps could not be enforced.
Facebook for not alerting users that this happened, and making changes to protect user privacy.
Facebook used to have a feature that allowed apps to access profile data about the friends of the person that installed the app.
Cambridge Analytica for explicitly violating the Terms of Service against storing data. (Many other companies have similar clauses in their TOS).
Aleksandr Kogan, a researcher from Cambridge University, for storing the data, and then giving it to Cambridge Analytica.

So who is more at fault? This will be interesting....

But... this isn’t a data breach because users authorized data use by installing the survey app even though they probably did not intend to do so, and probably did not approve of that data use.

An informative article can be found here, and is the source for the image.

Data Privacy Professionals

So how did it work?

1. Kogan paid people on Amazon Mechanical Turk to install a Facebook app called thisisyourdigitallife, a personality survey in 2014.
2. About 270,000 users installed the app.
3. Upon installing the app, he collected the user’s data as well as the data of all of that user’s friends that did not have a particular privacy setting enabled.
4. This yielded data on 50 million users.
5. The data was handed to Cambridge Analytica, who developed psychographic profiles of the users.
6. Kogan was ordered by Facebook to destroy the data.
7. These profiles were used to push ads and other campaign related stuff.
Data Privacy Professionals

One more example. Not to pick on Facebook, but another timely privacy issue arose regarding Facebook apparently collecting phone numbers and text messages from the Messenger App.

Ryan R. Rosario
rrosario@cs.ucla.edu
Computer Science 143: Introduction to Database Systems
UCLA Department of Computer Science

Data Privacy Professionals

Possible uses for this data that are more likely:

1. **Engagement**: Allowing Messenger to serve as the main communication center on the phone (similar to Hangouts).
2. **Expanding what FB knows about you** by using phone contacts that may also be on Facebook but are not directly connected. (Ad Targeting)
3. **People You May Know**
4. **As a baseline for measuring how much time is spent on the phone**, which can be used to compute the percentage of time spent on Facebook.

Ryan R. Rosario
rrosario@cs.ucla.edu
Computer Science 143: Introduction to Database Systems
UCLA Department of Computer Science

Data Privacy Professionals

But Facebook, and companies like it, are not the only entities that can build such profiles.

1. **Credit card companies**;
2. **Cell phone providers** (numbers can be used to detect trafficking and other illegal activities, GPS location);
3. **Cable companies** (what you watch, record);
4. **Internet service providers** (IP addresses visited)

The US Postal Service seems to be the only exempt entity... for now at least...
Data Privacy Professionals

Currently, Data Privacy Professionals seem to come from the social sciences, or from a law background.

Coming up, we will need Computer Scientists that understand systems, and the possibilities of data, software and hardware as well how to protect users from unapproved actions and data use aside from legislation.

Reading

If you want to follow along in the book, these slides cover most of Chapter 1.

For next lecture, you will want to read Chapter 2.

If you are brave, you may want to skim Chapter 6 as well, but we will cover it in lecture in an abbreviated manner, so don’t get bogged down in reading Chapter 6!!!

Way Back When...

Data was originally stored in flat files within an operating system’s filesystem.

A matter of fact, UCLA student records were stored in flat files up until the past ten years or so.¹

System administrators had to manually write individual applications to add/update/delete records from these flat files. For example:

- Adding new students and instructors (all 16,500 in Fall... and many did not even accept!) 
- Generating class rosters.
- Computing GPAs.
- A ton of other stuff.
Major Drawbacks I

As you can imagine, there are a lot of drawbacks to this method.

- **Data Integrity and Redundancy.**
  - Multiple people touch these files and applications. They may choose to use different structures or slightly different formats that mess everything up.
  - Duplicated data. For example, each grade report, roster and registration record contains the same student and course information over and over.
  - higher storage costs and it is possible for the duplicated data to not agree.

Major Drawbacks II

- **Sysadmin is the Bottleneck**
  - Each time someone retrieve a new type of data or report, the sysadmin must write a new program.
  - Change prereqs for a course? New app = wait.
  - Decide to collect new information about students or courses? New apps = wait.
  - Life must have moved more slowly back then. Lots of waiting.

- **Integrity.** There are many constraints at UCLA: enrollment restrictions, major requirements etc. and the system must enforce these constraints. This requires adding new logic to applications, and is difficult when multiple files are involved.

Major Drawbacks III

- **Lack of Atomicity.** (CS 111) What happens if the system fails during a high stakes operation?
  - Bank: If we transfer money from A to B and the system dies after withdrawing the money from A, A and B will be very unhappy.
  - Enrollment: Suppose the Exchange one class with another function drops a class and then adds a class. If this action is not atomic, what happens if the system fails after the drop?
  - We need a way to recover and fallback: either do it all successfully, or do nothing.
  - More sophisticated journaled filesystems like Apple HFS+ or ReiserFS did not exist back then.

Major Drawbacks IV

- **Concurrent Access.** (CS 111)
  Concurrent updates \(\Rightarrow\) Inconsistent data

  Enrollment example: If two students add a class at the same time, and an ancient MyUCLA reads and updates the number of students enrolled in each operation, what happens? We get the wrong count.
Major Drawbacks V

- **Security.** Not everyone should see all of the data. Imposing security constraints across many files in an application is difficult.

UCLA IT found out the hard way.

In the 1990s, DB2 was introduced as the database for certain student records. (There are other database systems such as Microsoft SQL Server and PeopleSoft as well, nowadays).

- When analyzing the data from the original flat file system, it was found that a certain percentage of student records were corrupt.
- Fortunately, because there is so much redundancy in flat files, the data could be recovered.
- It’s likely that most of the data were from alumni that are long gone.

In 2006-08, DB2 became the standard by which production student records were stored. Only the database was changed, the same IBM 3270 front end is still used today.

New technologies such as data warehousing allowed many custom applications to be built against this data such as reporting (Crystal Reports). This allowed UCLA to go almost entirely online, even with rapidly changing data, instead of relying on counselors and staff to do the work.

There Must be a Better Way!

- RDBMS