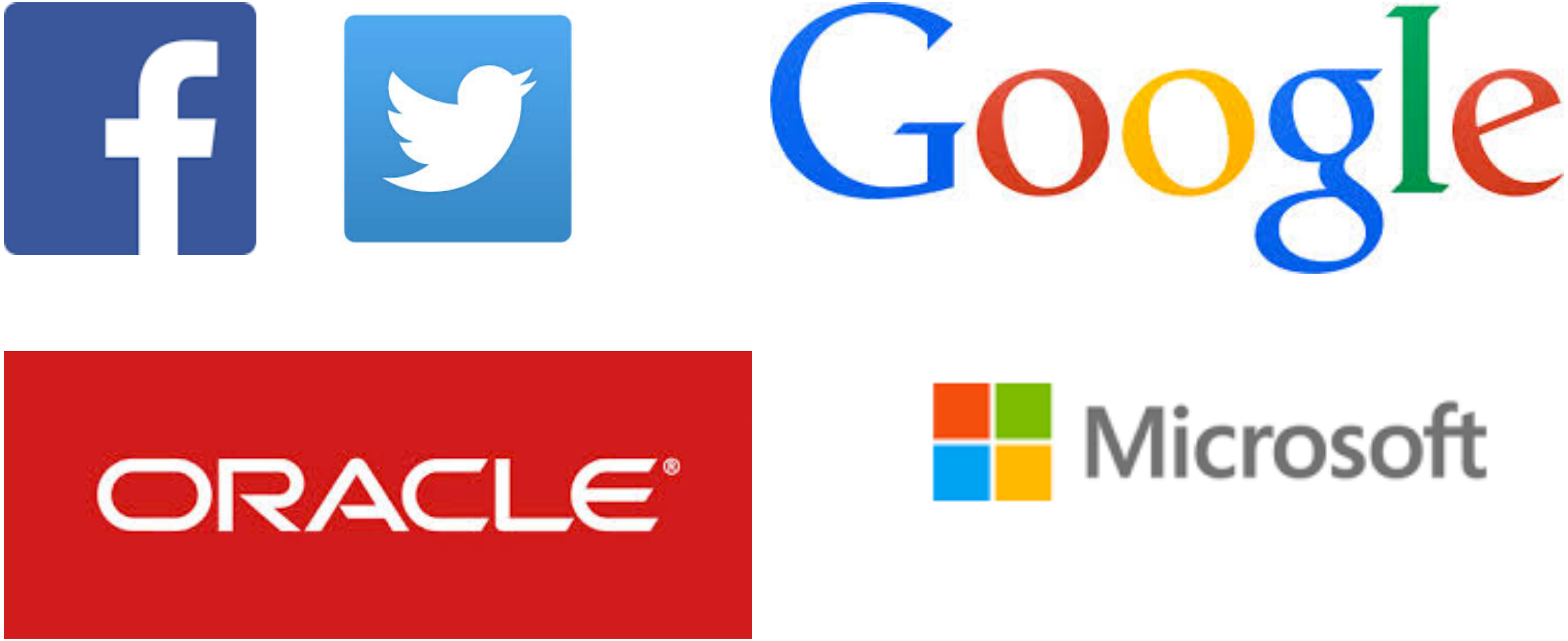
L01: Course Overview

CS 3200 sp18 s2: Database design

1/8/2018

The world is increasingly **driven by data**…

This class teaches **the basics** of how to use & manage data.



Increasingly many companies see themselves as **data driven**.

Key Questions We Will Answer

- How can we **collect and store** large amounts of data?
  - By building tools and data structures to efficiently index and serve data
- How can we **efficiently query** data?
  - By compiling high-level declarative queries into efficient low-level plans
- How can we **safely update** data?
  - By managing concurrent access to state as it is read and written
- How do different database systems manage **design trade-offs**?
  - e.g., at scale, in a distributed environment?

When you’ll use this material

- Building almost any software application
  - e.g., mobile, cloud, consumer, enterprise, analytics, machine learning
  - Corollary: every application you use uses a database
  - Bonus: every program consumes data (even if only the program text!)
- Performing data analytics
  - Business intelligence, data science, predictive modeling
  - (Even if you’re using Pandas https://pandas.pydata.org/, you’re using relational algebra!)

- Building data-intensive tools and applications
  - Many core concepts power deep learning frameworks to self-driving cars

Today’s Lecture

1. Introduction, admin & setup

2. Overview of the relational data model

3. Overview of DBMS topics: Key concepts & challenges

What you will learn about in this section

1. Motivation for studying DBs

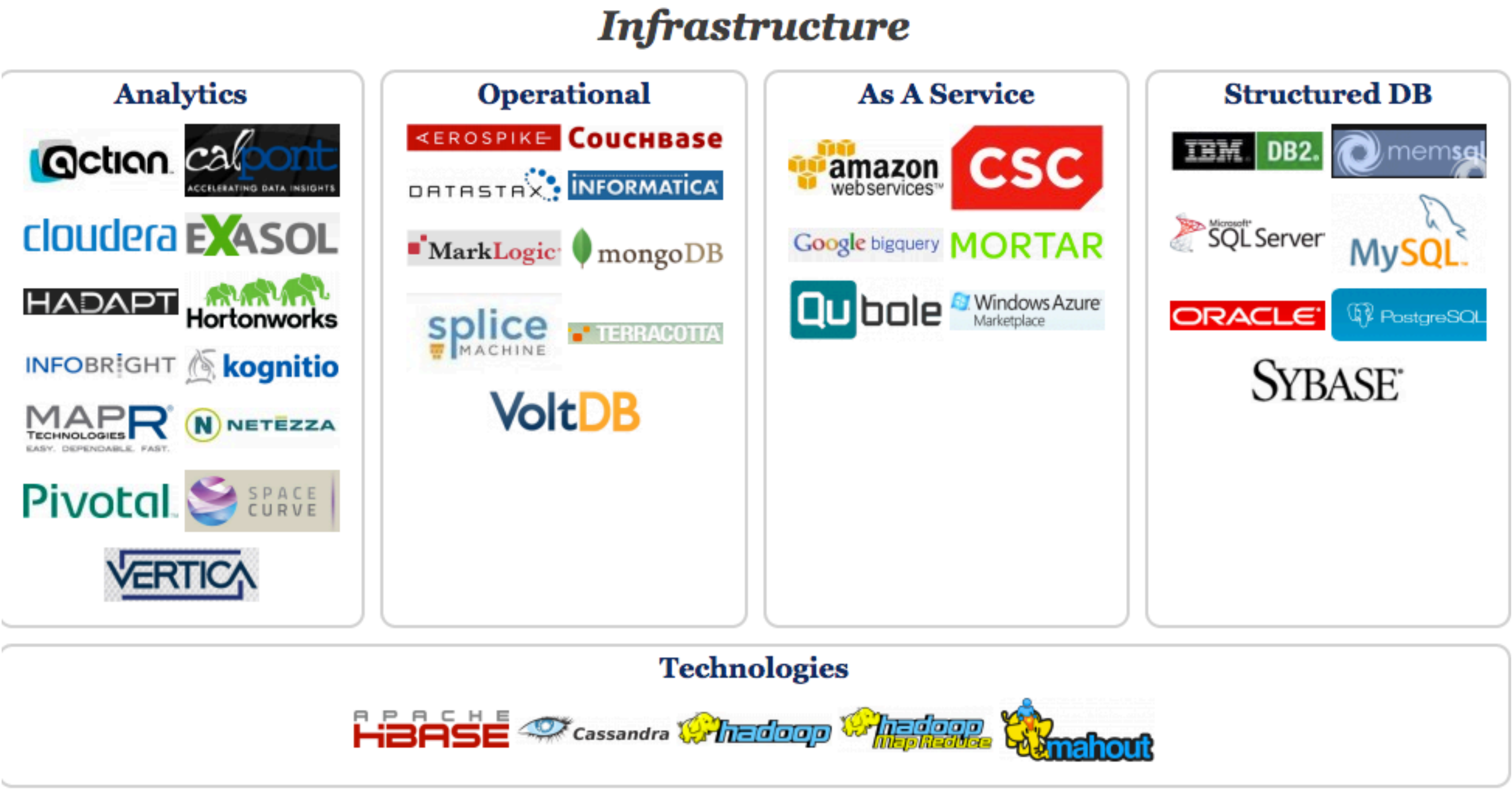
2. Administrative structure

3. Course logistics

4. Overview of lecture coverage

5. Some thoughts on Pedagogy

Big Data Landscape… Infrastructure is Changing



New tech. *Same* Principles.

http://www.bigdatalandscape.com/

Some "birth-years". When was SQL born?

- 2004: Facebook

- 1998: Google
- 1995: Java, Ruby
- 1993: World Wide Web
- 1991: Python

- 1985: Windows

Some "birth-years"

- 2004: Facebook

- 1998: Google
- 1995: Java, Ruby
- 1993: World Wide Web
- 1991: Python

- 1985: Windows

- 1974: SQL

Why should you study databases?

- Mercenary- make more $$$:
  - Startups need DB talent right away = low employee #
  - Massive industry…

- Intellectual:
  - Science: data poor to data rich
    - No idea how to handle the data!
  - Fundamental ideas to/from all of CS:
    - Systems, theory, AI, logic, stats, analysis….

Microsoft ORACLE IBM Google

Many great computer systems ideas started in DB.

What this course is (and is not)

- Discuss **fundamentals of data management**
  - How to design databases, query databases, build applications with them.
  - How to debug them when they go wrong!
  - Not how to be a DBA or how to tune Oracle 12g.

- We'll cover **how database management systems work**

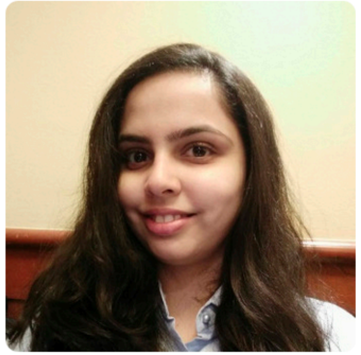
- And some (but not all of) **the principles of how to build** them

Who we are…

- Instructor (me) Wolfgang Gatterbauer
  - Faculty in the DATA lab (https://db.ccis.northeastern.edu/)
  - First year at Northeastern!
  - Taught before at University of Washington and CMU's business school
  - Research: theoretic foundations for scalable data management
  - Office hours: W 2:00-4:00, WVH 450

Teaching Assistants

Disha Sule (Teaching Assistant)



E-mail	sule.d@husky.neu.edu
Office Hours	TBD

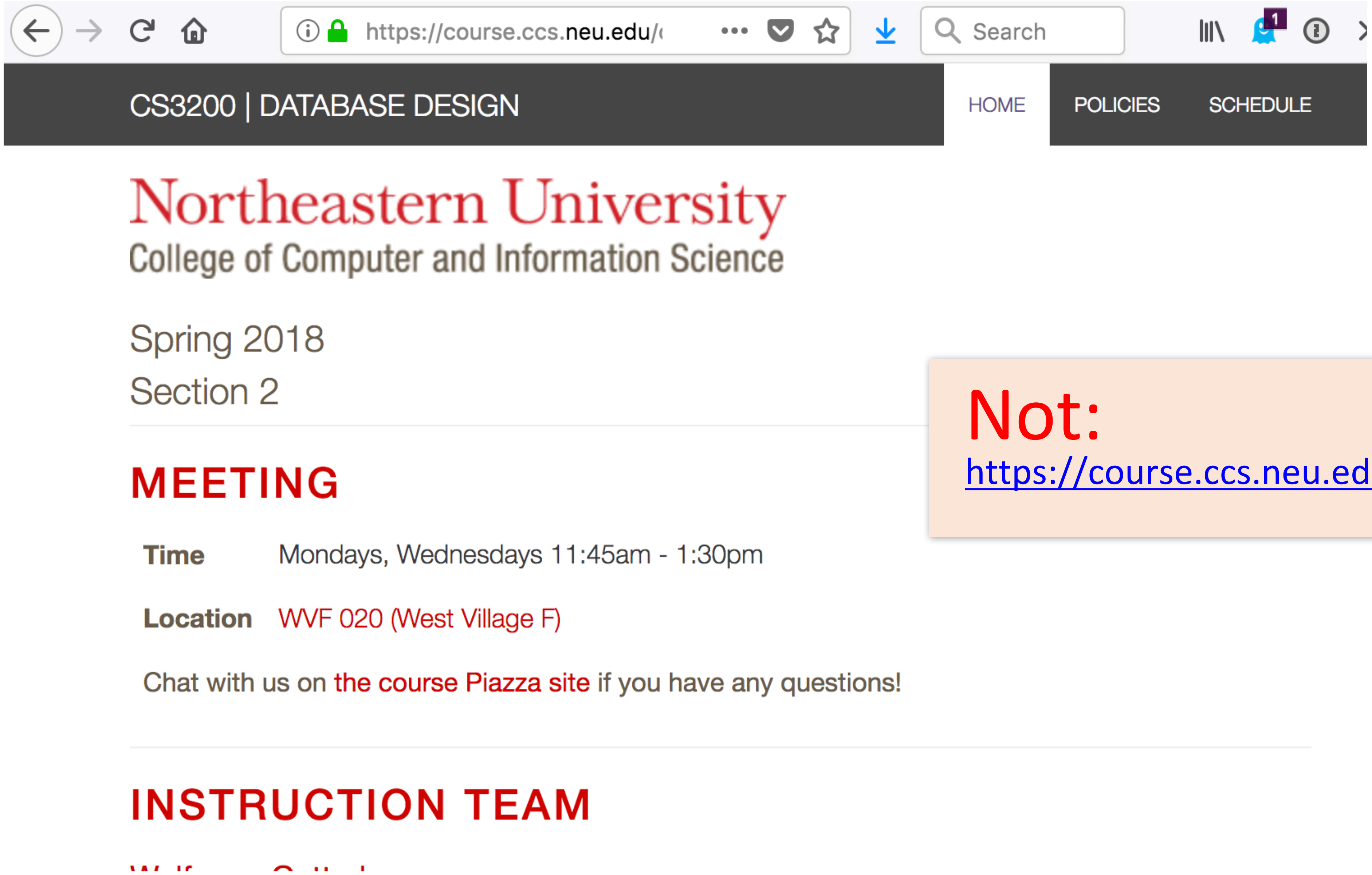
Priyal Mittal (Teaching Assistant)

E-mail	mittal.pr@husky.neu.edu
Office Hours	TBD

Sumit Bhanwala (Teaching Assistant)

E-mail	bhanwala.s@husky.neu.edu
Office Hours	TBD

https://course.ccs.neu.edu/cs3200sp18s2/



CS3200 | DATABASE DESIGN

HOME POLICIES SCHEDULE

Northeastern University
College of Computer and Information Science

Spring 2018
Section 2

MEETING

Time	Mondays, Wednesdays 11:45am - 1:30pm
Location	WVF 020 (West Village F)

Chat with us on the course Piazza site if you have any questions!

INSTRUCTION TEAM

Not:
https://course.ccs.neu.edu/cs3200sp18s3/

Communication w/ Course Staff

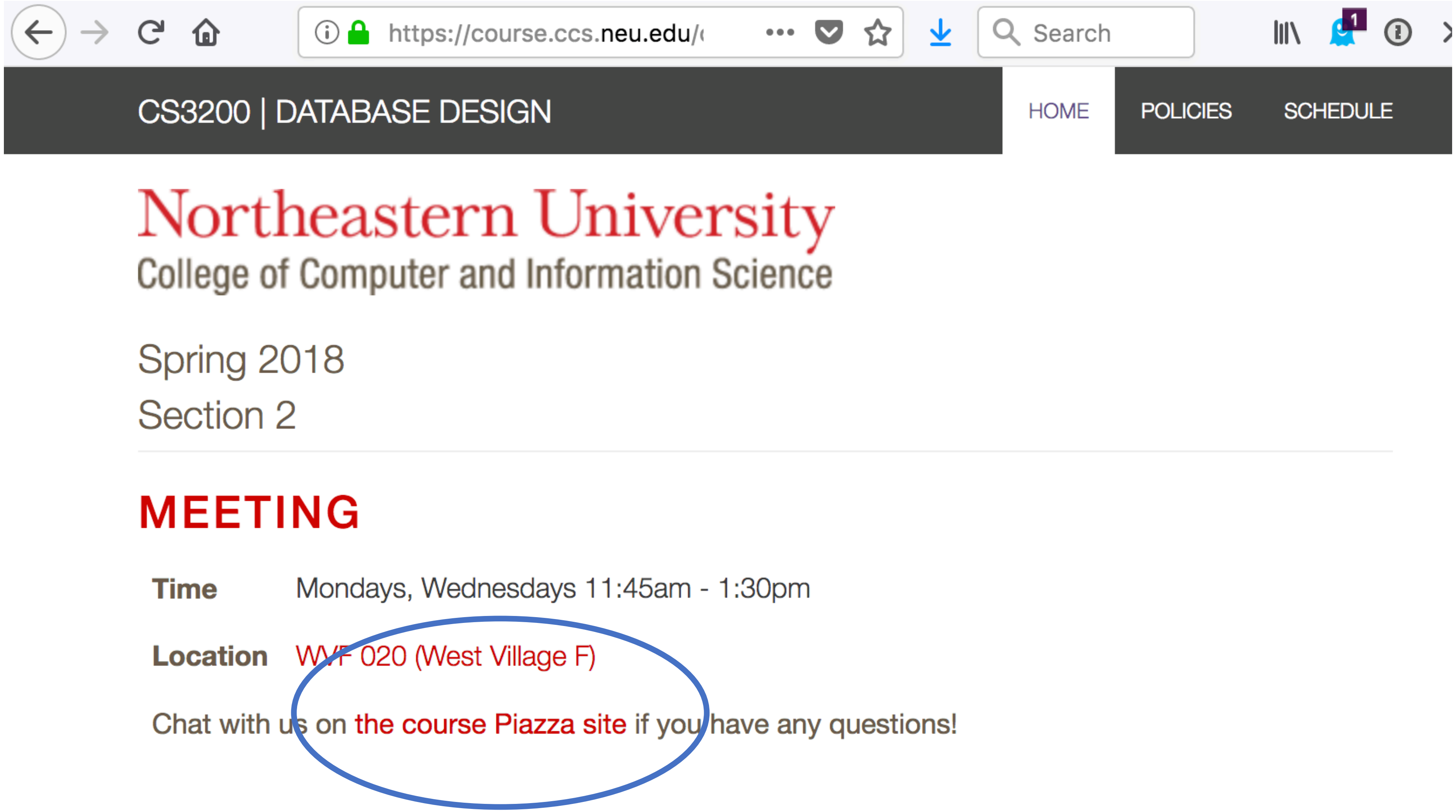
- Piazza
- Office hours
- By appointment!

TAs OHs to be listed on the course website!

Meeting location: TBD:

(either 4th floor or 1st floor WVH)

Piazza



Spring 2018
Section 2

MEETING

Time	Mondays, Wednesdays 11:45am - 1:30pm
Location	WVF 020 (West Village F)

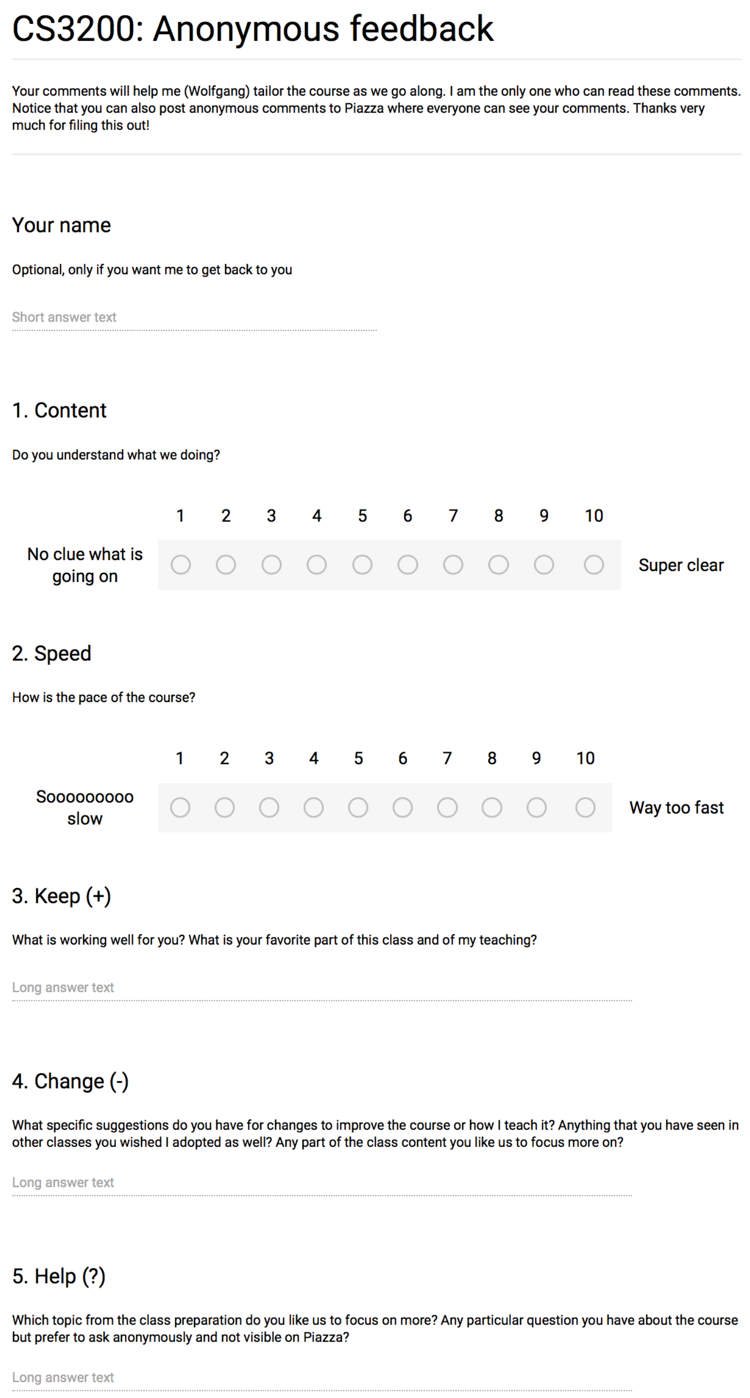
Chat with us on the course Piazza site if you have any questions!

The goal is to get you to answer each other's questions so you can benefit and learn from each other.

Please use this simple way to let me know what works or not!

https://goo.gl/sLJJeH

Piazza is visible to everyone in this class. This form only to me



CS3200: Anonymous feedback

Your comments will help me (Wolfgang) tailor the course as we go along. I am the only one who can read these comments. Notice that you can also post anonymous comments to Piazza where everyone can see your comments. Thanks very much for filing this out!

Your name

Optional, only if you want me to get back to you

Short answer text

1. Content

Do you understand what we doing?

	1	2	3	4	5	6	7	8	9	10	
No clue what is going on	○	○	○	○	○	○	○	○	○	○	Super clear

2. Speed

How is the pace of the course?

	1	2	3	4	5	6	7	8	9	10	
Sooooooooo slow	○	○	○	○	○	○	○	○	○	○	Way too fast

3. Keep (+)

What is working well for you? What is your favorite part of this class and of my teaching?

Long answer text

4. Change (-)

What specific suggestions do you have for changes to improve the course or how I teach it? Anything that you have seen in other classes you wished I adopted as well? Any part of the class content you like us to focus more on?

Long answer text

5. Help (?)

Which topic from the class preparation do you like us to focus on more? Any particular question you have about the course but prefer to ask anonymously and not visible on Piazza?

Long answer text

Important!

- Students with documented disabilities should send in their accommodation letter from the Disability Resource Center at 20 Dodge Hall by the **end of this week** to me.

Lectures

- Lecture slides cover **essential material**
  - This is your best reference.
  - We are trying to get away from book, but do have pointers

- Try to cover same thing in many ways: Lecture, lecture notes, homework, exams (no shock)
  - Attendance makes your life easier…

Attendance

- I dislike mandatory attendance… but in the past we noticed…
  - People who did not attend did worse ☹
  - People who did not attend used more course resources ☹
  - People who did not attend were less happy with the course ☹

- In previous school: mandatory attendance
- This year: voluntary (to start!) -- reserve right to change

Graded Elements

- Gradiance quizzes + participation (10%)
- Homeworks (25%)
- Group project (25%)
- Three exams (40% = 10% + 10% + 20%)

Homeworks are typically due Wednesday end of day, and are posted at least 1 week before due date

Un-Graded Elements

- Readings provided to help you!
  - Only items in lecture, homework, or project are fair game.

- In-class activities are mainly to help / be fun!
  - Will occur during class- not graded, but count as part of lecture material (fair game as well)

What is expected from you

- Attend lectures
  - If you don’t, it’s at your own peril

- Be active and think critically
  - Ask questions, post comments on forums

- Do programming and homework projects
  - Start early and be honest

- Study for exams

Interested in Research?

Paper at SIGMOD 2018

- R. Li, M. Riedewald, **Xinyan Deng**
  Submodularity of Distributed Join Computation

Poster presentation at Northeast Database day 2018

- R. Li, **Aditya Ghosh**, M. Riedewald, W. Gatterbauer
  Optimizing Data Partitioning for Distributed Band Joins
- P. Ojha, **Paul Langton**, W. Gatterbauer
  Scalable Compatibility Estimation in Large Network Data

In progress http://queryviz.com

Lectures: 1st half - from a user's perspective

1. **SQL**: Relational data models & Queries
   - ~ 5 lectures
   - How to manipulate data with SQL, a declarative language
     - reduced expressive power but the system can do more for you

2. **Database Design**: Design theory and constraints
   - ~ 6 lectures
   - Designing relational schema to keep your data from getting corrupted

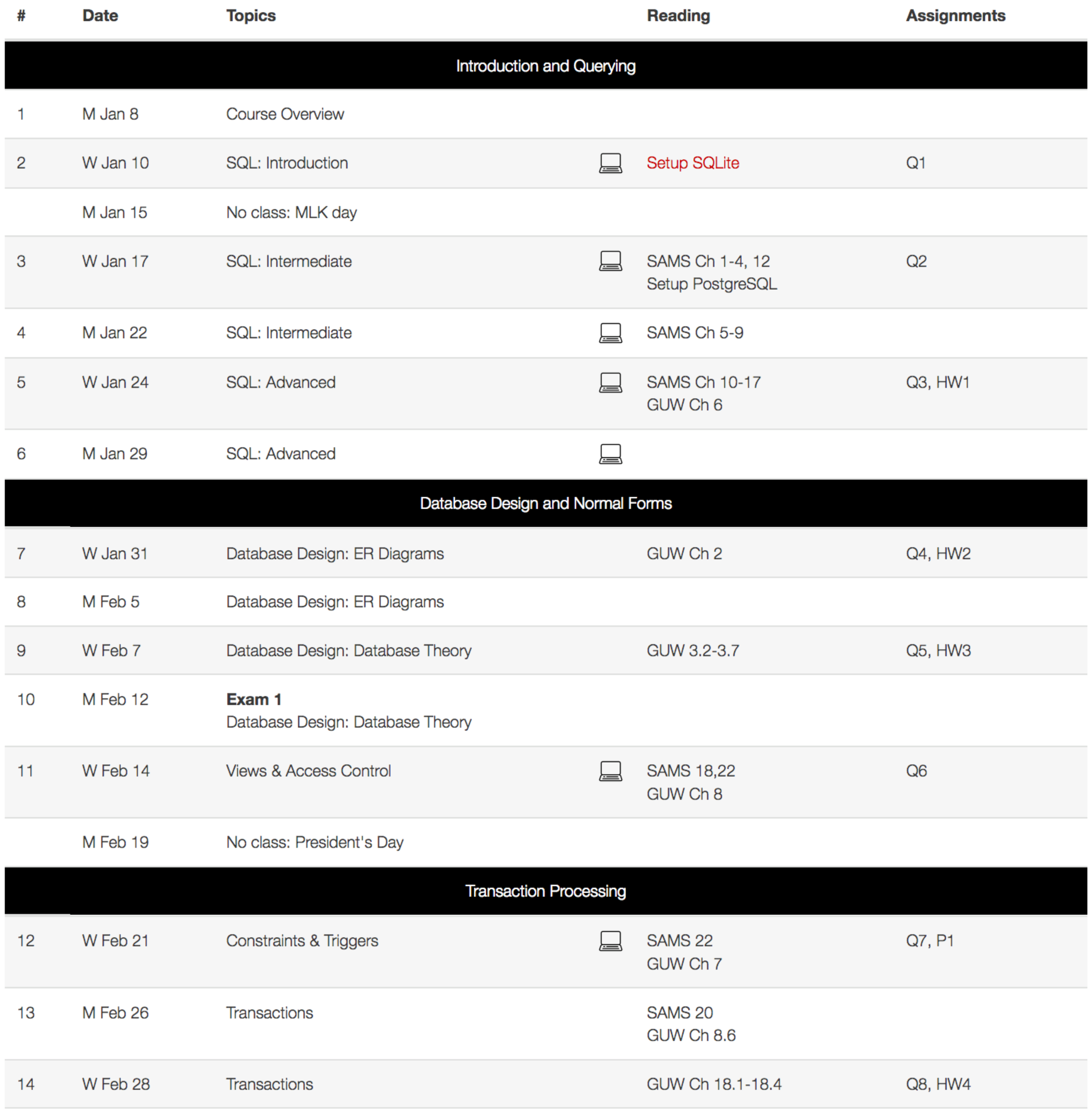
3. **Transactions**: Syntax & supporting systems
   - ~ 3 lectures
   - A programmer's abstraction for data consistency

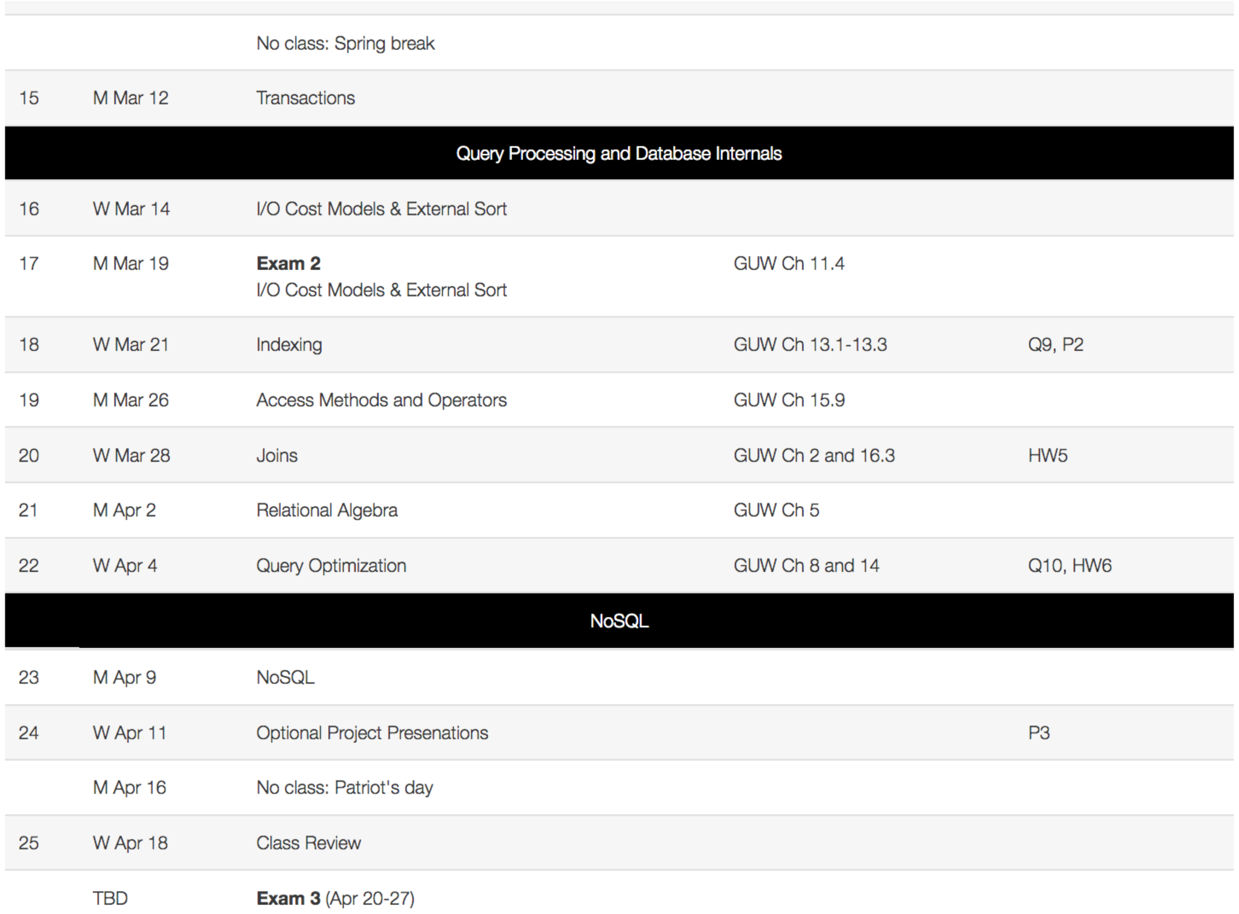
Lectures: 2nd half - understanding how it works

4. **Database internals**: Query Processing
   - ~ 7 lectures
   - Indexing
   - External Memory Algorithms (IO model) for sorting, joins, etc.
   - Basics of query optimization (Cost Estimates)
   - Relational algebra

5. **NoSQL**
   - ~0-2 lectures
   - Key-Value Stores
   - (More in CS6240: Large-Scale Parallel Data Processing)

https://course.ccs.neu.edu/cs3200sp18s2/sched.html



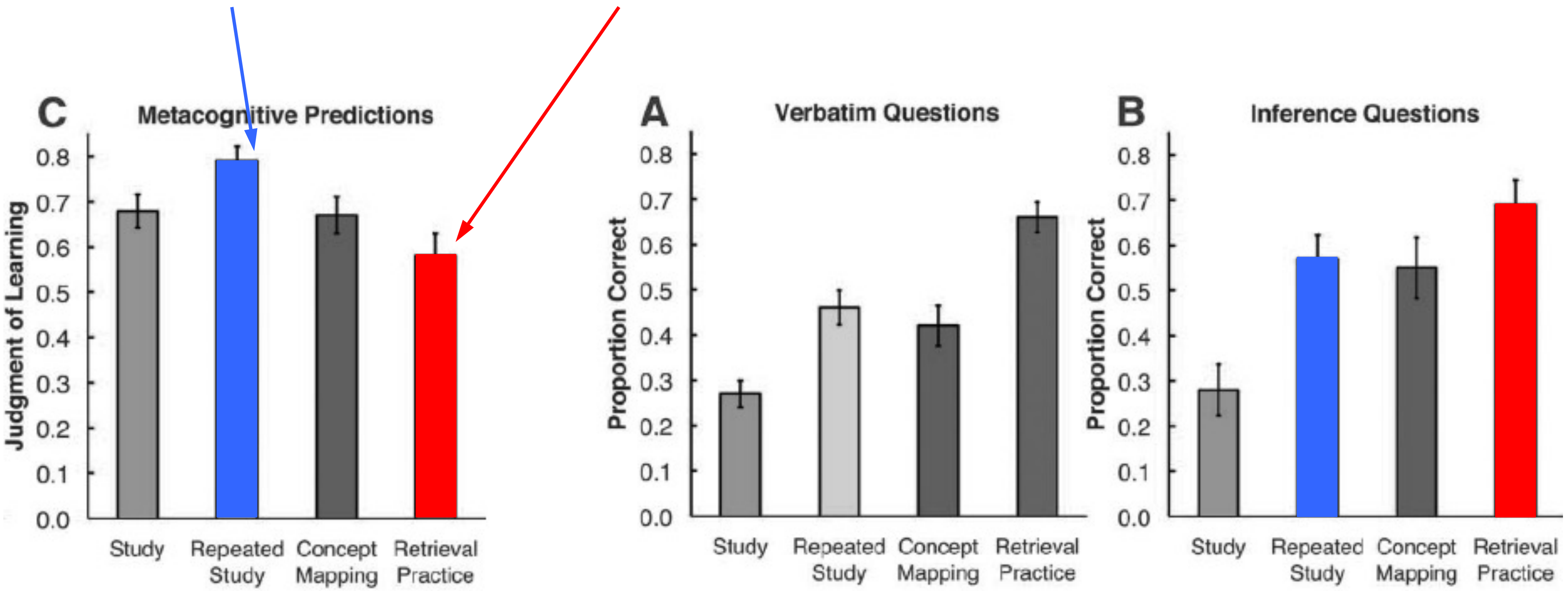


#	Date	Topics		Reading	Assignments
		Introduction and Querying			
1	M Jan 8	Course Overview			
2	W Jan 10	SQL: Introduction	💻	Setup SQLite	Q1
	M Jan 15	No class: MLK day			
3	W Jan 17	SQL: Intermediate	💻	SAMS Ch 1-4, 12 Setup PostgreSQL	Q2
4	M Jan 22	SQL: Intermediate	💻	SAMS Ch 5-9	
5	W Jan 24	SQL: Advanced	💻	SAMS Ch 10-17 GUW Ch 6	Q3, HW1
6	M Jan 29	SQL: Advanced	💻		
		Database Design and Normal Forms			
7	W Jan 31	Database Design: ER Diagrams		GUW Ch 2	Q4, HW2
8	M Feb 5	Database Design: ER Diagrams			
9	W Feb 7	Database Design: Database Theory		GUW 3.2-3.7	Q5, HW3
10	M Feb 12	**Exam 1** Database Design: Database Theory			
11	W Feb 14	Views & Access Control	💻	SAMS 18,22 GUW Ch 8	Q6
	M Feb 19	No class: President's Day			
		Transaction Processing			
12	W Feb 21	Constraints & Triggers	💻	SAMS 22 GUW Ch 7	Q7, P1
13	M Feb 26	Transactions		SAMS 20 GUW Ch 8.6	
14	W Feb 28	Transactions		GUW Ch 18.1-18.4	Q8, HW4
		No class: Spring break			
15	M Mar 12	Transactions			
		Query Processing and Database Internals			
16	W Mar 14	I/O Cost Models & External Sort			
17	M Mar 19	**Exam 2** I/O Cost Models & External Sort		GUW Ch 11.4	
18	W Mar 21	Indexing		GUW Ch 13.1-13.3	Q9, P2
19	M Mar 26	Access Methods and Operators		GUW Ch 15.9	
20	W Mar 28	Joins		GUW Ch 2 and 16.3	HW5
21	M Apr 2	Relational Algebra		GUW Ch 5	
22	W Apr 4	Query Optimization		GUW Ch 8 and 14	Q10, HW6
		NoSQL			
23	M Apr 9	NoSQL			
24	W Apr 11	Optional Project Presenations			P3
	M Apr 16	No class: Patriot's day			
25	W Apr 18	Class Review			
	TBD	**Exam 3** (Apr 20-27)			

Studying material: "Under which study condition do you think you learn better?"

passive reading

active Q&A

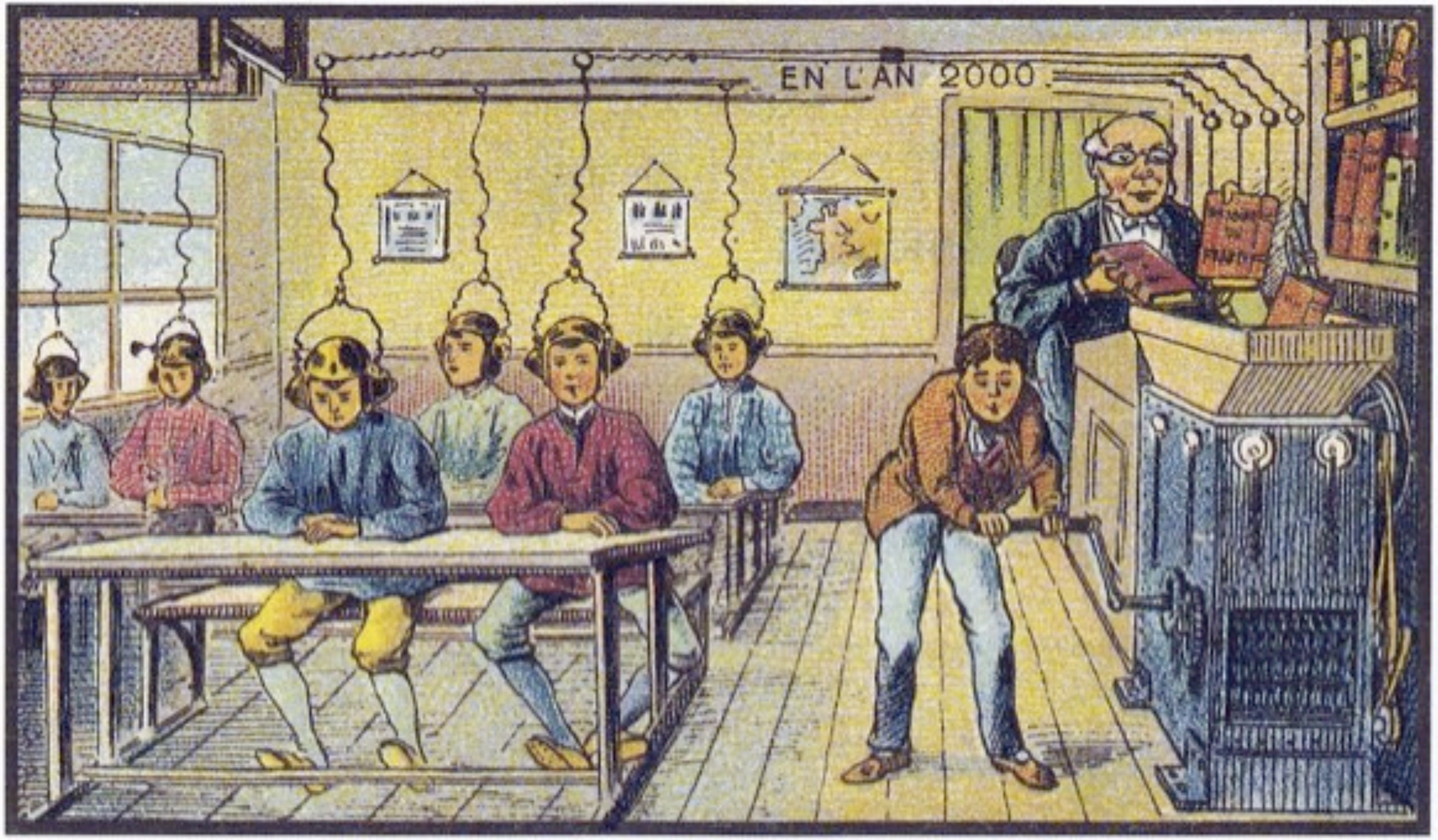


Judged performance
(=what people think)

Actual performance
(=what is actually working)

Source: Karpicke & Blunt, "Retrieval Practice Produces More Learning than Elaborative Studying with Concept Mapping," Science, 2011.

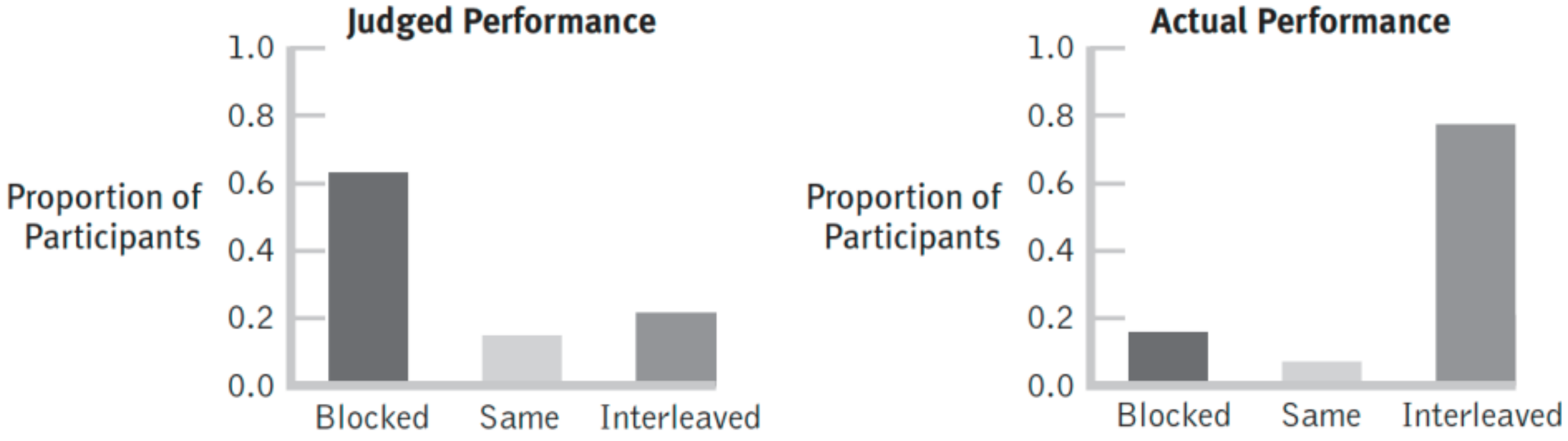
The year 2000 imagined in 1900

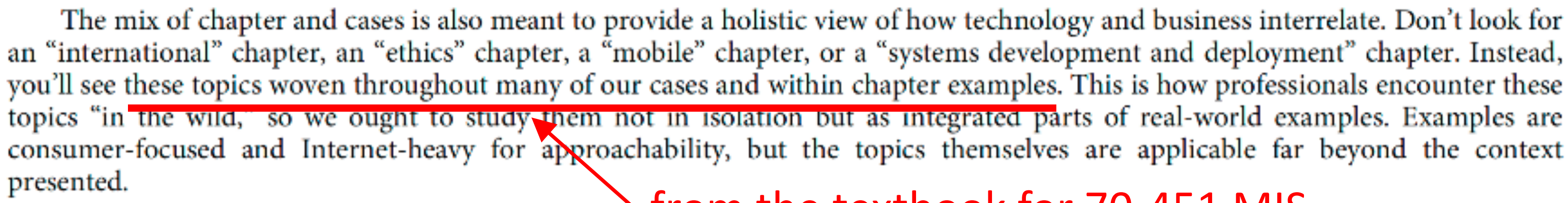


At School

Source: http://5.mshcdn.com/wp-content/gallery/the-year-2000-as-imagined-in-1900/future.jpg

Sequencing Material: "Under which teaching condition do you think you learn better?"



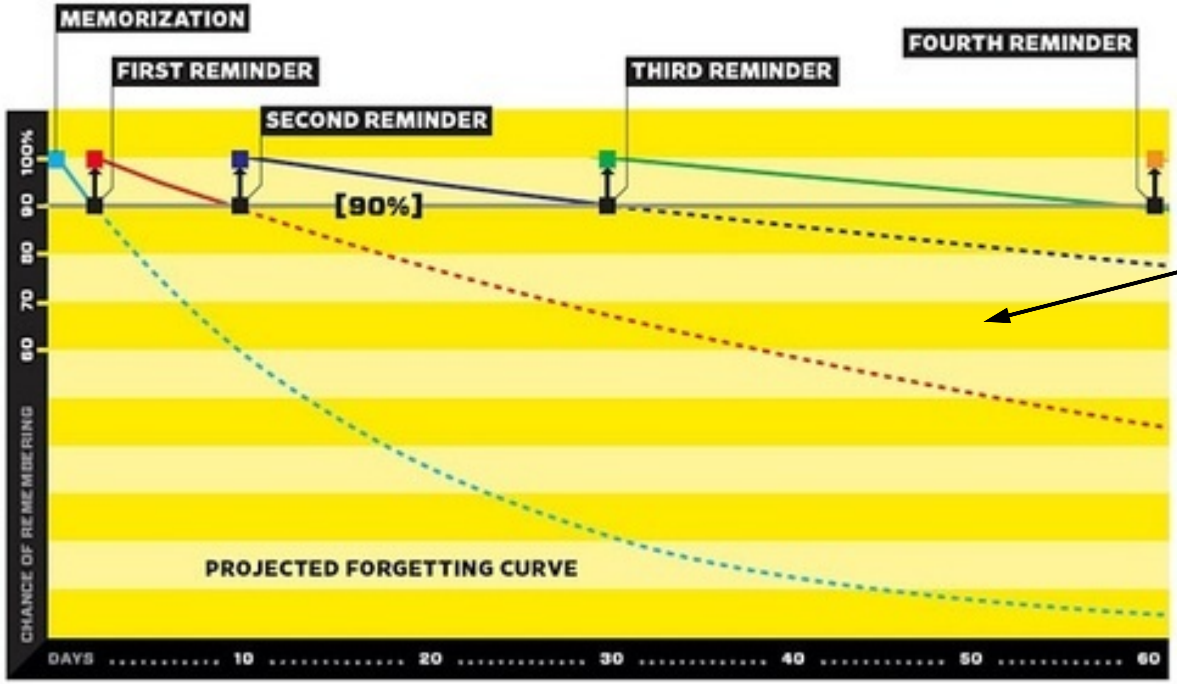


The mix of chapter and cases is also meant to provide a holistic view of how technology and business interrelate. Don't look for an "international" chapter, an "ethics" chapter, a "mobile" chapter, or a "systems development and deployment" chapter. Instead, you'll see these topics woven throughout many of our cases and within chapter examples. This is how professionals encounter these topics "in the wild," so we ought to study them not in isolation but as integrated parts of real-world examples. Examples are consumer-focused and Internet-heavy for approachability, but the topics themselves are applicable far beyond the context presented.

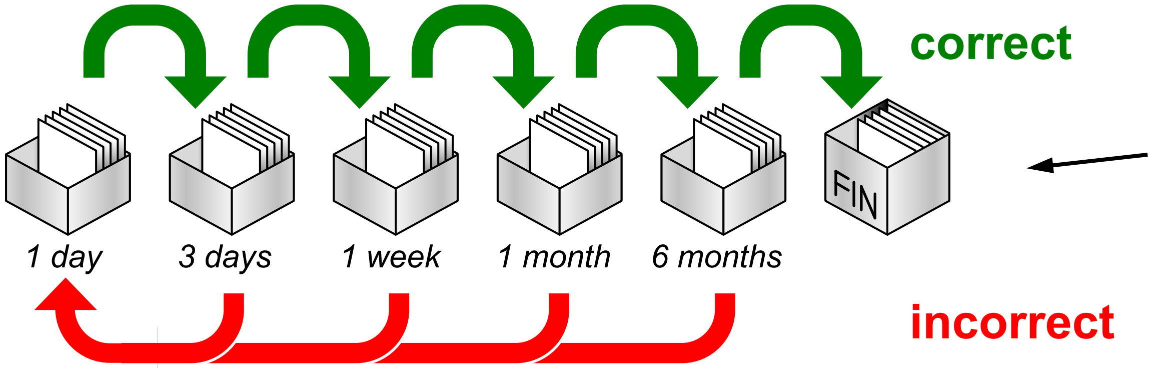
from the textbook for 70-451 MIS

Source: Bjork & Bjork, "Making things hard on yourself, but in a good way: Creating desirable difficulties to enhance learning," Psychology and the real world (...), 2011.

Spaced Repetition



Ebbinghaus Forgetting Curve



Leitner System (Pimsleur's graduated interval recall)

Sources: http://www.wired.com/2008/04/ff-wozniak/, Gatterbauer & Suciu, "Managing Structured Collections of Community Data," CIDR 2011.

The "Surfer Analogy" for time management

Source: http://stwww.surfermag.com/files/2013/10/Yak_Charlie-970x646.jpg

Today’s Lecture

1. Introduction, admin & setup

2. Overview of the relational data model

3. Overview of DBMS topics: Key concepts & challenges

What you will learn about in this section

1. Definition of DBMS

2. Data models & the relational data model

3. Schemas & data independence

What is a DBMS?

- A large, integrated collection of data

- Models a real-world enterprise
  - Entities (e.g., Students, Courses)
  - Relationships (e.g., Alice is enrolled in 145)

> A Database Management System (DBMS) is a piece of software designed to store and manage databases

A Motivating, Running Example

- Consider building a course management system (CMS):
  - Students
  - Courses
  - Professors

Entities

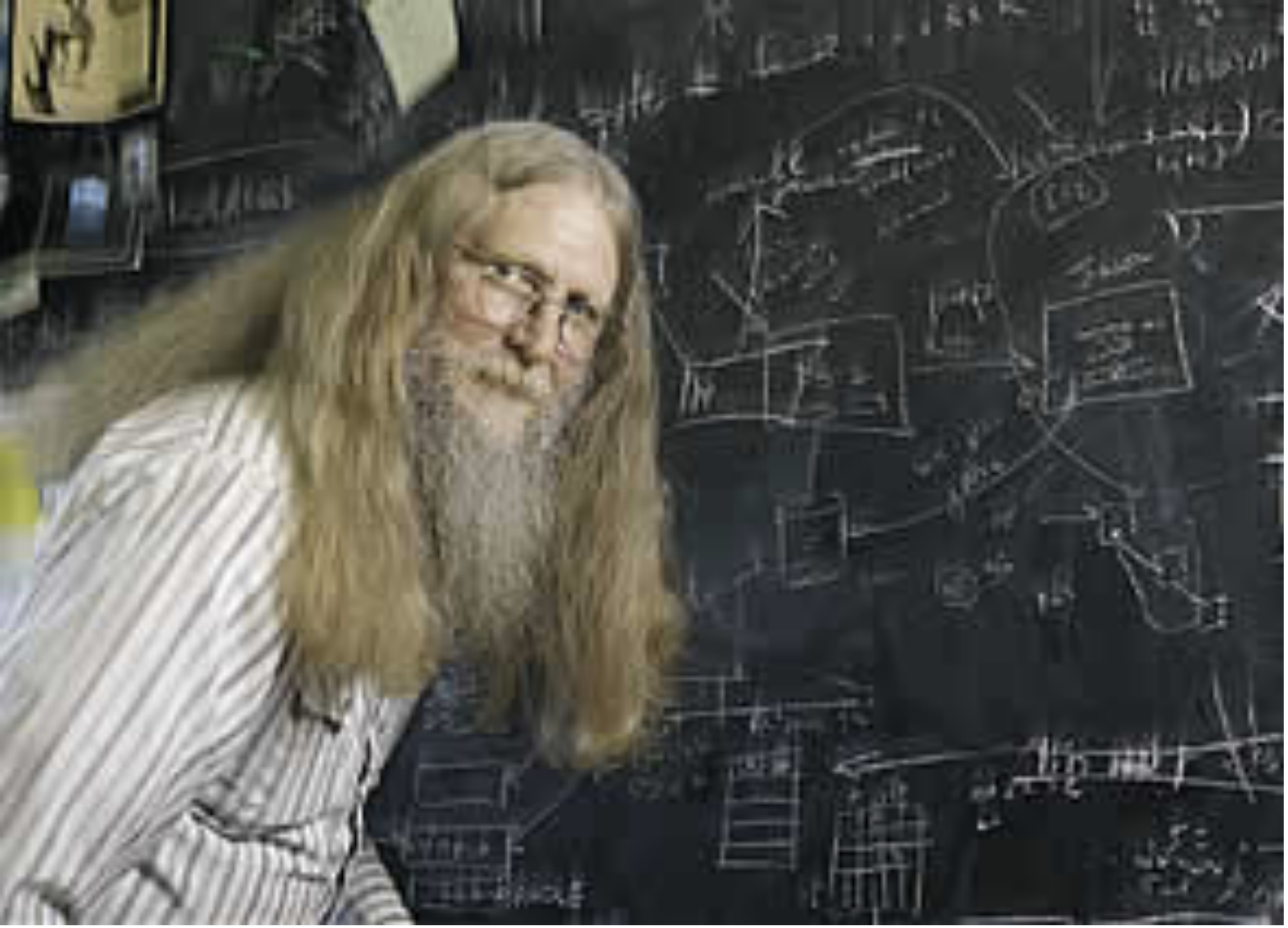
  - Who takes what
  - Who teaches what

Relationships

Data models

- A **data model** is a collection of concepts for describing data
  - The relational model of data is the most widely used model today
    - Main Concept: the relation- essentially, a table

- A **schema** is a description of a particular collection of data, **using the given data model**
  - E.g. every relation in a relational data model has a schema describing types, etc.



“Relational databases are the foundation of western civilization”

Bruce Lindsay, IBM Research

As quoted in: https://dl.acm.org/citation.cfm?id=1083803

Modeling the CMS

- Logical Schema
  - Students(sid: string, name: string, gpa: float)
  - Courses(cid: string, cname: string, credits: int)
  - Enrolled(sid: string, cid: string, grade: string)

Relations

sid	Name	Gpa
101	Bob	3.2
123	Mary	3.8

Students

cid	cname	credits
564	564-2	4
308	417	2

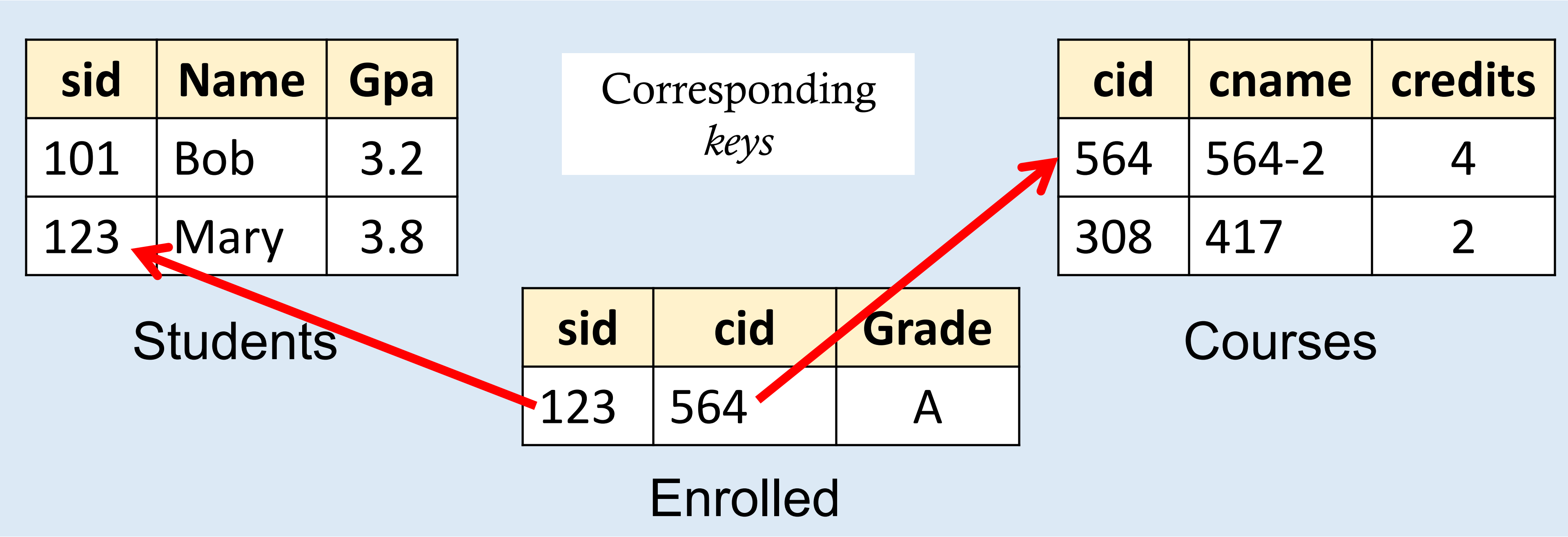
Courses

sid	cid	Grade
123	564	A

Enrolled

Modeling the CMS

- Logical Schema
  - Students(sid: string, name: string, gpa: float)
  - Courses(cid: string, cname: string, credits: int)
  - Enrolled(sid: string, cid: string, grade: string)



sid	Name	Gpa
101	Bob	3.2
123	Mary	3.8

Students

Corresponding *keys*

cid	cname	credits
564	564-2	4
308	417	2

Courses

sid	cid	Grade
123	564	A

Enrolled

Other Schemata…

- **External Schema**: (Views)
  - Course_info(cid: string, enrollment: integer)
  - Derived from other tables

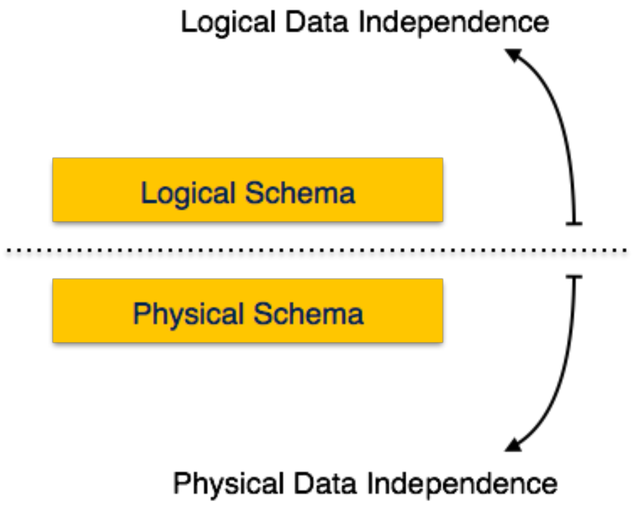
⬆ Applications

- **Logical Schema**: Previous slide

- **Physical Schema**: describes data layout
  - Relations as unordered files
  - Some data in sorted order (index)

⬇ Administrators

Data independence



- Concept: Applications do not need to worry about how the data is structured and stored

<u>Logical data independence:</u> protection from changes in the *logical structure of the data*

I.e. should not need to ask: can we add a new entity or attribute without rewriting the application?

<u>Physical data independence:</u> protection from *physical layout changes*

I.e. should not need to ask: which disks are the data stored on? Is the data indexed?

One of the most important reasons to use a DBMS

Today’s Lecture

1. Introduction, admin & setup

2. Overview of the relational data model

3. Overview of DBMS topics: Key concepts & challenges

What you will learn about in this section

1. Transactions

2. Concurrency & locking

3. Atomicity & logging

4. Summary

Challenges with Many Users

- Suppose that our CMS application serves 1000’s of users or more- what are some challenges?

  - Security: Different users, different roles

  We won’t look at too much in this course, but is extremely important

  - Performance: Need to provide concurrent access

  Disk/SSD access is slow, DBMS hide the latency by doing more CPU work concurrently

  - Consistency: Concurrency can lead to update problems

  DBMS allows user to write programs as if they were the **only** user

Transactions

- A key concept is the **transaction (TXN)**: an **atomic** sequence of db actions (reads/writes)

<u>Atomicity</u>: An action either completes *entirely* or *not at all*

Acct	Balance
a10	20,000
a20	15,000

Transfer $3k from a10 to a20:

1. Debit $3k from a10
2. Credit $3k to a20

Acct	Balance
a10	17,000
a20	18,000

Written naively, in which states is **atomicity** preserved?

- Crash before 1,
- After 1 but before 2,
- After 2.

DB Always preserves atomicity!

Transactions

- A key concept is the **transaction (TXN)**: an **atomic** sequence of db actions (reads/writes)
  - If a user cancels a TXN, it should be as if nothing happened!

Atomicity: An action either completes *entirely* or *not at all*

- Transactions leave the DB in a **consistent** state
  - Users may write integrity constraints, e.g., ‘each course is assigned to exactly one room’

Consistency: An action results in a state which conforms to all integrity constraints

However**,** note that the DBMS does not understand the *real* meaning of the constraints– consistency burden is still on the user!

Challenge: Scheduling Concurrent Transactions

- The DBMS ensures that the execution of $\{T_1,\ldots,T_n\}$ is equivalent to some **serial** execution

- One way to accomplish this: **Locking**
  - Before reading or writing, transaction requires a lock from DBMS, holds until the end

- Key Idea: If T_i wants to write to an item x and T_j wants to read x, then T_i, T_j **conflict**. Solution via locking:
  - only one winner gets the lock
  - loser is blocked (waits) until winner finishes

A set of TXNs is **isolated** if their effect is as if all were executed serially

What if T_i and T_j need X and Y, and T_i asks for X before T_j, and T_j asks for Y before T_i? -> *Deadlock!* One is aborted…

All concurrency issues handled by the DBMS…

Ensuring Atomicity & Durability

- DBMS ensures **atomicity** even if a TXN crashes!

- One way to accomplish this: **Write-ahead logging (WAL)**

- Key Idea: Keep a log of all the writes done.
  - After a crash, the partially executed TXNs are undone using the log

Write-ahead Logging (WAL): Before any action is finalized, a corresponding log entry is forced to disk

We assume that the log is on "stable" storage

All atomicity issues also handled by the DBMS…

A Well-Designed DBMS makes many people happy!

- End users and DBMS vendors
  - Reduces cost and makes money

- DB application programmers
  - Can handle more users, faster, for cheaper, and with better reliability / security guarantees!

- Database administrators (DBA)
  - Easier time of designing logical/physical schema, handling security/authorization, tuning, crash recovery, and more…

Must still understand DB internals

Summary of DBMS

- DBMS are used to maintain, query, and manage large datasets.
  - Provide concurrency, recovery from crashes, quick application development, integrity, and security

- Key abstractions give **data independence**

- DBMS R&D is one of the broadest, most exciting fields in CS. Fact!