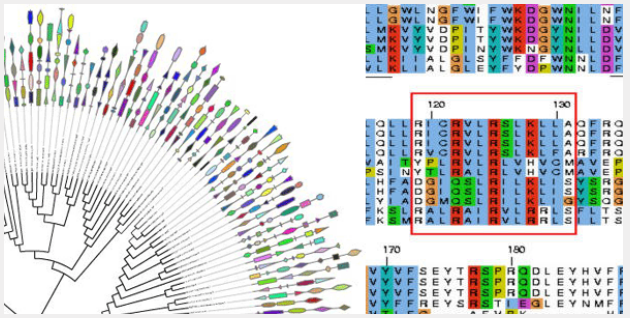
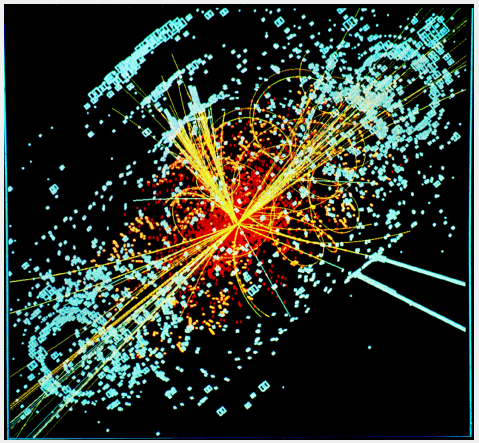
CS 564: DATABASE MANAGEMENT SYSTEMS

Fall 2018

DATA IS EVERYWHERE!

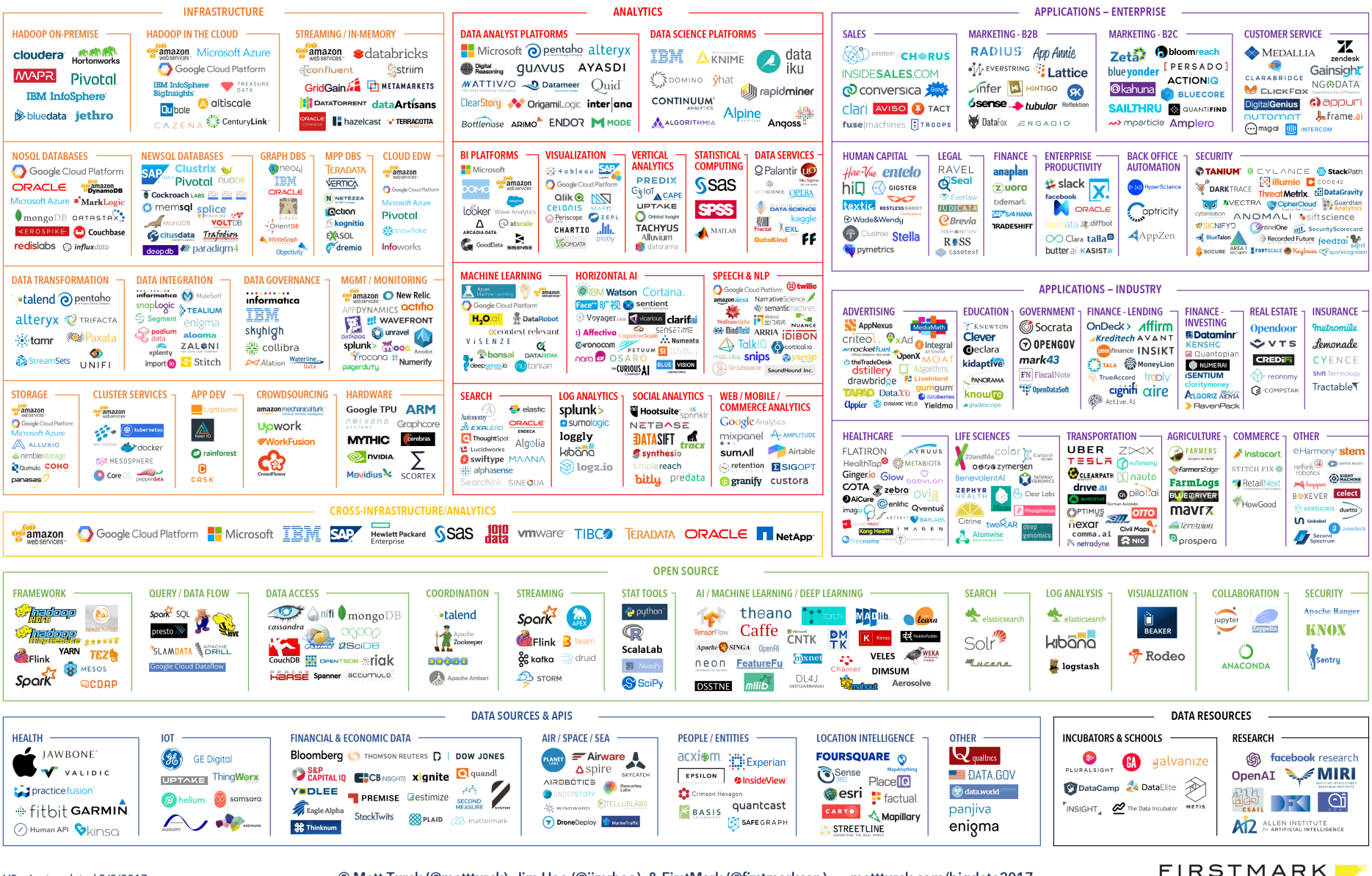


- Our world is increasingly data driven
  - scientific discoveries
  - online services (social networks, online retailers)
  - decision making



- **Databases** are the core technology

BIG DATA LANDSCAPE 2017



V2 – Last updated 5/3/2017

 mattturck.com/bigdata2017

FIRSTMARK
EARLY STAGE VENTURE CAPITAL

WHAT IS THIS CLASS ABOUT?

The fundamentals of data management

- how we design and query a database?
- how do database management systems work?
- how do we build a DBMS?

Course Logistics

INSTRUCTOR

Paris Koutris

- paris@cs.wisc.edu
- Office hours @ CS4363
  - *Monday* 4:00-5:00 pm (after class)
  - *Thursday* 11:00-12:00 am

ABOUT ME

- undergrad in Athens, Greece
- Ph.D. in University of Washington (the other UW)
- at UW-Madison since Fall 2015!

Research Interests

- massively parallel processing
- data pricing
- managing uncertain data
- data structures for query processing

TAs

Ankur Goswami

Minh Le

Zhiyi Chen

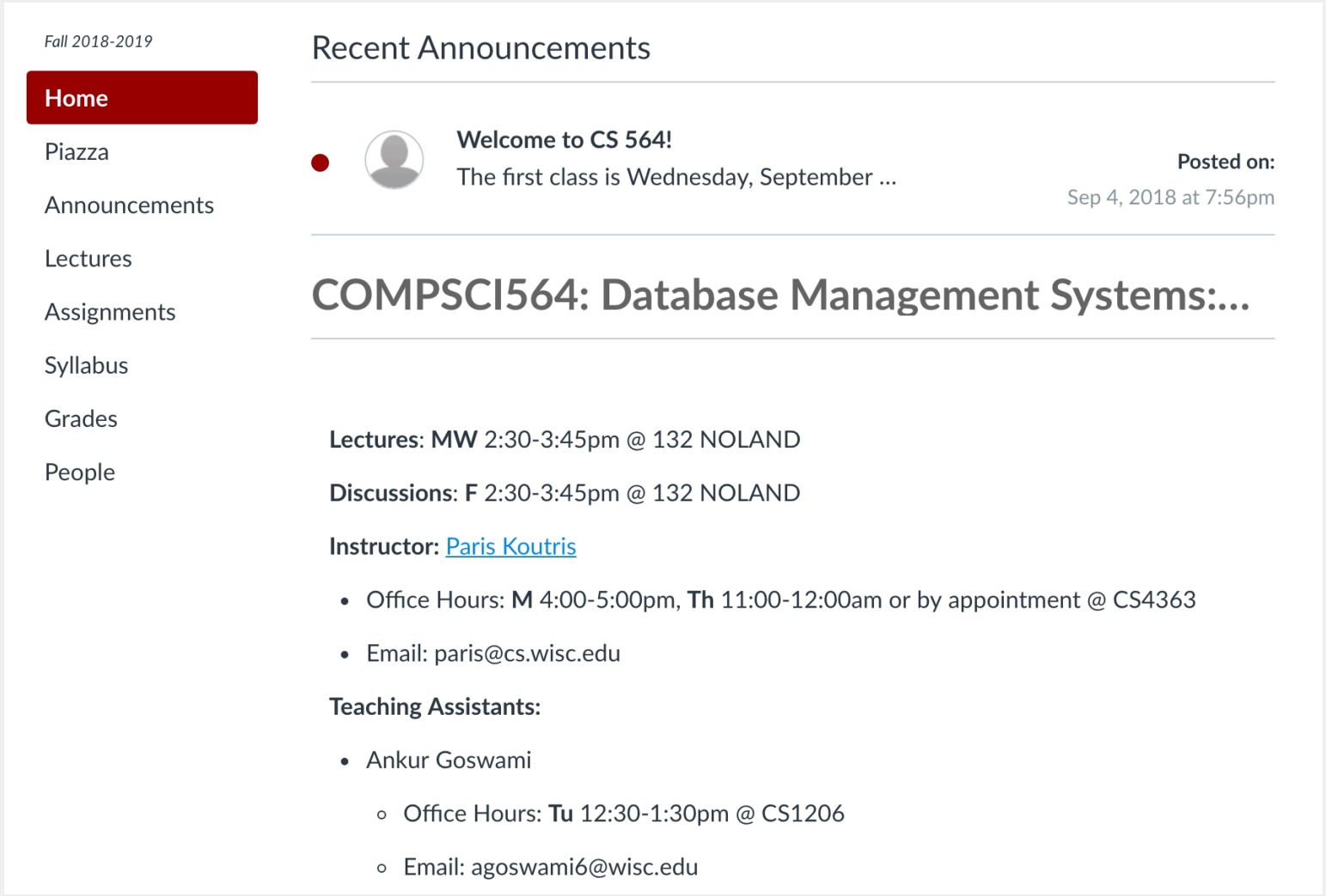
COURSE FORMAT

- Lectures **M+W** 2:30-3:45 pm
- Discussions **F** 2:30-3:45 pm

- 3 programming projects (in groups of 3)
- 3 problem sets (individual)

- Midterm Exam
- Final Exam

CANVAS HAS EVERYTHING!



COMMUNICATION

Mailing List:

compsci564-1-f18@lists.wisc.edu

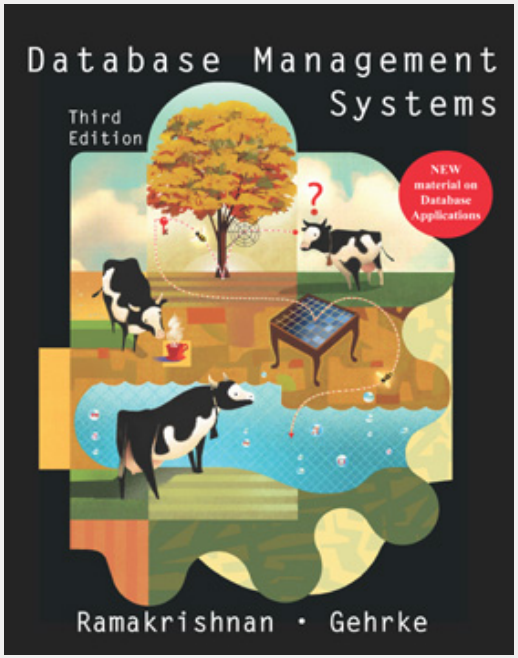
Piazza: through Canvas

- Questions (answer each other's questions!!)
- Discussions

TEXTBOOK

- Database Management Systems (3d edition)

- **Come to the lectures!**
  - take notes
  - ask questions
  - participate



PREREQUISITES

- Data structures and algorithm background necessary!
  - **CS 367** is a must

- For the programming projects
  - programming-heavy
  - C++ will be used for the database internals
  - Python is also required

GRADE DISTRIBUTION

- Programming Projects (3): 7.5% each
- Problem Sets (3): 7.5% each
- Midterm: 20%
- Final: 35%

PROBLEM SETS

Individual assignments: Python + Jupyter notebooks

- **Problem Set #1**
  - *SQL*
- **Problem Set #2**
  - *Normalization & Relational Algebra*
- **Problem Set #3**
  - *I/O cost & Query Optimization*

PROGRAMMING PROJECTS

In groups of 3: Python and C++

- **Project #1**
  - *ER Modeling & Schema Design*
- **Project #2**
  - *Buffer Manager*
- **Project #3**
  - *Web Interface*

EXAMS

- **Midterm Exam**
  - *when*: October 24 (2:30-3:45 pm)
  - *where*: in class

- **Final Exam**
  - *when*: December 19 (7:25-9:25 pm)
  - *where*: TBD

WHAT IS EXPECTED FROM YOU

- Attend the lectures
- Participate and ask questions
- Do the assignments (start early!)
- Study for the exams

COURSE OVERVIEW

Part A: Databases from the user's perspective

- Module **A1**: SQL
- Module **A2**: ER Model + DB Design
- Module **A3**: Relational Algebra

COURSE OVERVIEW

Part B: Database internals

- Module **B1**: Basics of DB Internals
- Module **B2**: Indexes
- Module **B3**: Query Processing
- Module **B4**: Transactions

Before We Start

JUPYTER NOTEBOOK

- Jupyter notebooks are interactive shells which save output in a nice notebook format

- You'll use these for
  - in-class activities
  - interactive lecture supplements/recaps
  - problem sets, projects, …

JUPYTER NOTEBOOK SETUP

- Install on your laptop via the instructions on the website
- Other options running via one of the alternative methods:
  - Ubuntu VM
  - CS Machines
- Come to office hours if you need help with installation!

DATABASES: A SHORT INTRO

DATABASES

What is a database?

- A collection of files storing related data

What are some examples of databases?

- payroll database
- Amazon's product information
- bank account database

DBMS

*What is a Database Management System (**DBMS**)?*

- A program written by someone else that allows us to manage efficiently a large database and allows data to persist over long periods of time

What are examples of DBMSs?

- SQL Server, Microsoft Access (Microsoft)
- DB2 (IBM)
- Oracle
- MySQL, PostgreSQL, SQLite

EXAMPLE: ONLINE BOOKSTORE

- What data do we need to store?
- How will we use the data stored?

EXAMPLE: ONLINE BOOKSTORE

- What functionality do we want to support?
  - efficient querying
  - multiple users
  - recovery after crashes
  - security, user authorization

DATA STORAGE

- Data stored for a long period of time (persistent data): *the data outlives the application*
- Large amounts of data (100s of GB)
- User authorization on which data to access
- Protection from system crashes

QUERIES & UPDATES

- Store and retrieve data in an efficient way
  - Organize data on disk
  - Index data for faster access
- Make efficient use of memory hierarchy
- Safely allow concurrent access to the data
- Allow the data to be updated safely

CONCURRENCY CONTROL

- Alice and Bob have the same number for a gift certificate of $100 at the online bookstore
  - Alice @ her office orders ”Book A” for $30
  - Bob @ his office orders ”Book B” for $60

- Questions:
  - What is the ending credit?
  - What if second book costs $80?
  - What if system crashes?

SCHEMA CHANGE

- Say that we need to add a new field to books
  - entails changing file formats
  - need to rewrite virtually all applications

WHAT CAN A DBMS DO?

- All the above!!
- Automate a lot of boring operations on data
  - don't have to program over and over
  - can write complex data manipulations in just a few lines
- Make execution very fast
  - scales up to very large data sets
- Make concurrent access/modification possible
  - many users can use the data at the same time

KEY CONCEPTS

- **Data model:** abstraction that describes the data
- **Schema:** describes a specific database using the "language" of the data model
- **Query Language:** high-level language to allow a user to pose queries easily
  - Declarative languages (SQL)
- **Query optimizer/compiler:** code that evaluates the query efficiently

DATA INDEPENDENCE

The application does not change when the underlying data structure or storage changes

- Physical independence: can change how data is stored on disk without maintenance to applications
- Logical independence: can change schema without affecting applications

RELATIONAL MODEL

- The data is stored in tables (relations in the mathematical sense)
- A database is a set of tables

schema

name	price	author	hardcover
007456	The Da Vinci Code	Dan Brown	yes
909405	Ender's Game	Orson Scott Card	no
…	…	…	…

record/tuple

QUERYING THE DATA

- SQL or other declarative languages
- Example: *find all books written by Dan Brown*

```
SELECT *
FROM books B
WHERE B.author = “Dan Brown”
```

QUERY PROCESSOR

- **Optimizer**: what is the best imperative execution plan for the given query?
- **Evaluation**: execute the plan as efficiently as possible

INTERESTED IN MORE?

CS 764

- gory details on how a DBMS works
- transactions/concurrency/internals

CS 784

- the theory behind databases