# CSE 541: Database Systems I

Course Introduction

Slides Credit: Tianzheng Wang @ SFU

# What is a Database?

# Database

- A collection of *inter-related* data.

- Models real-world activities
  - Entities (e.g. students, courses)
  - Relationships (e.g. Dong is teaching CSE 541)

| Name | Gender | GPA |
|---|---|---|
| Mike | Male | 4.0 |
| Bob | Male | 3.6 |
| Alice | Female | 3.8 |

- Generally, data can be anything: text, images, tables, etc.

# Where/When a Database is used?

# Database in Real Life

- Examples
  - Amazon: Online Shopping
  - LionPath: Course Management System
  - Chase: Banking System
  - Delta: Airline Reservation System
- **Database is everywhere in our modern life!!**

- Core component of most computer applications.

# Database Management System (DBMS)

- What is a DBMS?
  - A piece of software designed to store and manage databases
  - Arguably one of the most complicated software stack.
- **Why not** just directly managed by applications?

- Examples (Relational DBMS)
  - Commercial: Oracle, IBM DB2, Microsoft SQL Server
  - Open source: MySQL (Sun/Oracle), PostgreSQL, SQLite

- More generally: Document-based (MongoDB), KV-Store (RocksDB), Computation Framework (Spark)

# What you have probably learned:
# The use of (relational) database systems

CMPSC 431W equivalent

SQL

ER Model

DB Apps

…

*Coming up next:*

- The making of (relational) database systems
  - CSE 541 (this course)
    - Classic, fundamental problems
    - Recent advances
- Research in database/data-intensive systems
  - Revisit prior ideas, propose new ideas, build new systems
  - Seminar based special topics courses (CSE 597)
  - Or join our research group

# CSE 541

**A hands-on course about database systems internals**

- How are database systems designed?
  - Storing and managing data in different storage devices
  - Ensure data survive failures or even bugs
  - Access data quickly
  - Correctly handle user requests
- What are the principles behind it?
- How to get it right and make it fast and reliable?
  - Write efficient large-scale programs
  - Leverage multicore processors, memory space, networking…
    - Systems programming (read: C/C++)

*“OK cool. But why should I care?”*

# Why Take this Course?

DBMS developers are **in demand** and there are many challenging unsolved problems in data management and processing.

If you are good enough to write code for a DBMS, then you can write code on **almost anything else**.

# Intellectual Curiosity

**Learn about large-scale DBMS/software system design & engineering**

- You will get to know deeply about
  - Classic database systems designs and new trends
  - Practical implementation
  - Fundamental principles
- “Side” benefits
  - Concepts learned here useful in many areas, not just databases
  - Become comfortable with working with a lot of C/C++ code and popular tools in industry
  - ~~Work in a team (if you choose to) and with others’ code~~
  - Will make you comfortable with building almost any software systems in the future
  - Build up confidence in working with new things and picking up existing code quickly

# Career and $$$

- Many Turing Awards in database area (**four so far**)
- High-demand on the job market – academia & industry
- $50B and growing market, many companies investing heavily

Amazon Aurora
ORACLE®
VOLTDB
PostgreSQL
TERADATA.
SAP®
TiDB
IBM
DB2
Z/OS
Google Cloud Spanner
MySQL™
Microsoft® SQL Server®
MEMSQL

*+ Many more…*

# This course is *not* about:

A checklist for dropping this course:

- "Big Data" systems
  - Hadoop, Spark, Hive, MapReduce, etc. – go to 410
- The application side of databases
  - Data Science, NLP, Data Mining, Machine Learning, etc. – go to 445/448/583/586 etc.
  - Learning SQL, e.g., using MySQL to store data for a website – go to 431W
- Theory in Databases
  - Will touch upon the necessary theory related to systems
  - But focus is on practices: system design & implementation

**Bottom line: drop this course if you:**

- Are not interested in relational database systems, and/or
- Don't want to (learn) write C/C++ code

# Course Logistics

- Instructor: Dong Xie (dongx@psu.edu)
- Textbook:
  - Database Management Systems (3rd Edition)
- Required background:
  - CMPSC 431W or equivalent, CMPSC 473/CSE 511 preferred.
  - Ability to program in C/C++ (**very important**)
- Office hours
  - **Instructor:** Tuesday 1:00 – 3:00 PM or by appointment

# Communication

- Lecture: Interrupt me any time when
  - I am speaking too fast.
  - You don't understand what I am talking about.
  - You have a database-related question.
- Canvas
  - Announcements will be here. Make sure you are set up to get notified.
  - Forum-like discussion. *Keep the discussion open.*
  - Private Messages when necessary.
- Email:
  - Include your full name and also course number (CSE 541)

# Course Organization

- Projects (60%)
  - Individual project
  - 4 main projects + project 0 as warm up
  - Covering: storage engine, indexing, query processing, query optimization
- Exams (40%)
  - Midterm Exam: 20%
  - Final Exam: 20%
- Grading
  - Standard 90/80/70/60 grading scale
  - No late turn in lose 10% per day down to E

# Academic Integrity

- Do not cheat, or we will hunt you down. **DO NOT**:
  - Copy code from another student
  - Even look at code from another student
  - Copy code from the web
  - Ask for answers on StackOverflow or a similar website
- Discussion is ok, but must be your own work
  - Must provide proper citations

- For the project, you must use a private repo and remain private forever
  - Otherwise treated as plagiarism (sharing solutions to others)
  - **No public repo allowed even after taking the course**
  - Or We will go back to you

- See policy linked from Syllabus

# What I am expecting

- No one knows everything since born.

- Independent Critical Thinking
- Not only learn **what** and **how**, But also ask **why**.

- Questions are encouraged.
- Learn to ask **right** and **good** questions.

- ***Get your hands dirty!***

# Topics

- Storage Management
- Indexing

- Transactions and Concurrency
- Logging and Crash Recovery

- Query Evaluation
- Query Optimization
- External Sorting

- Distributed/Parallel Database Systems
- Practical Issues
- Interactions with OS/HW

*Might be adjusted along the way