CS425 – Fall 2017
Boris Glavic
Course Information

Modified from:
Database System Concepts, 6th Ed.
©Silberschatz, Korth and Sudarshan
See www.db-book.com for conditions on re-use
Hi, I am Boris Glavic, Assistant Professor in CS
Hi, I am Boris Glavic, Assistant Professor in CS

I am a database guy!
Hi, I am **Boris Glavic**, Assistant Professor in CS

I am a **database** guy!

I will teach you: database stuff
Why are Databases Important?

■ What do Databases do?
1. Provide persistent storage
2. Efficient declarative access to data -> Querying
3. Protection from hardware/software failures
4. Safe concurrent access to data
What happens if you do not pay attention?
Who uses Databases?

- Most big software systems involve DBs!
  - Business Intelligence ⇒ e.g., IBM Cognos
  - Web based systems
  - ...

- **You!** (desktop software)
  - Your music player ⇒ e.g., Amarok
  - Your Web Content Management System
  - Your email client
  - Half of the apps on your phone
  - ...

- Every big company
  - Banks
  - Insurance
  - Government
  - Google, …
Who Produces Databases?

- **Traditional relational database systems is big business**
  - IBM ⇒ DB2
  - Oracle ⇒ Oracle 😊
  - Microsoft ⇒ SQLServer
  - Open Source ⇒ MySQL, Postgres, SQLite, …

- **Emerging distributed systems with DB characteristics and Big Data**
  - Cloud storage and Key-value stores ⇒ Amazon S3, Google Big Table, . . .
  - Big Data Analytics ⇒ Hadoop, Google Map & Reduce, . . .
  - SQL on Distributed Platforms ⇒ Hive, Tenzing, …
Why are Database Interesting (for Students)?

- **The pragmatic perspective**
  - Background in databases makes you competitive in the job market ;-)

- **Systems and theoretical research**
  - Database research has a strong systems aspect
    - Hacking complex and large systems
    - Low-level optimization
      - cache-conscious algorithms
      - Exploit modern hardware
  - Databases have a strong theoretical foundation
    - Complexity of query answering
    - Expressiveness of query languages
    - Concurrency theory
    - …
Why are Database Interesting (for Students)?

Connection to many CS fields

- Distributed systems
  - Getting more and more important
- Compilers
- Modeling
- AI and machine learning
  - Data mining
- Operating and file systems
- Hardware
  - Hardware-software co-design
Webpage and Faculty

■ Course Info

- **Course Webpage:** [http://cs.iit.edu/~cs425](http://cs.iit.edu/~cs425)
- **Google Group:** [https://groups.google.com/d/forum/cs425-2017-fall-group](https://groups.google.com/d/forum/cs425-2017-fall-group)
  - Used for announcements
  - Use it to discuss with me, TA, and fellow students
- **Syllabus:** [http://cs.iit.edu/~cs425/files/syllabus.pdf](http://cs.iit.edu/~cs425/files/syllabus.pdf)
- **Git Repos:** [https://github.com/IITDBGroup/cs425](https://github.com/IITDBGroup/cs425)

■ Faculty

- **Boris Glavic** ([http://cs.iit.edu/~glavic](http://cs.iit.edu/~glavic))
- **Email:** bglavic@iit.edu
- **Phone:** 312.567.5205
- **Office:** Stuart Building, room 226C
- **Office Hours:** Mondays, 12pm-1pm (and by appointment)
TAs

- TAs
  - TBA
Workload and Grading

- **Exams**
  - Midterm (25%)
  - Final (35%)

- **Homework Assignments** (preparation for exams!) – 20%
  - HW1 (Relational algebra)
  - HW2 (SQL)
  - HW3 (Database modeling)

- **Course Project** (20%)
  - In groups of 3 students
  - Given an example application (e.g., ticketing system)
    - Develop a database model
    - Derive a database schema from the model
    - Implement the application accessing the database
Course Objectives

■ Understand the underlying ideas of database systems
■ Understand the relational data model
■ Be able to write and understand SQL queries and data definition statements
■ Understand relational algebra and its connection to SQL
■ Understand how to write programs that access a database server
■ Understand the ER model used in database design
■ Understand normalization of database schemata
■ Be able to create a database design from a requirement analysis for a specific domain
■ Know basic index structures and understand their importance
■ Have a basic understanding of relational database concepts such as concurrency control, recovery, query processing, and access control
PostgreSQL

In this course we will use PostgreSQL, a powerful open source database management system

- [https://www.postgresql.org/](https://www.postgresql.org/)
Course Project

- Forming groups
  - Your responsibility!
  - Inform me + TA
  - Deadline: TBA

- Git repositories
  - Create an account on Bitbucket.org (https://bitbucket.org/) using your IIT email
  - We will create a repository for each student
  - Use it to exchange code with your fellow group members
  - The project has to be submitted via the group repository

- Timeline:
  - Brainstorming on application (by Sep 11th)
  - Design database model (by Nov 12th)
  - Derive relational model (by Nov 25th)
  - Implement application (by end of the semester)
Fraud and Late Assignments

- All work has to be original!
  - Cheating = 0 points for assignment/exam
  - Possibly E in course and further administrative sanctions
  - Every dishonesty will be reported to office of academic honesty

- Late policy:
  - -20% per day
  - No exceptions!

- Course projects:
  - Every student has to contribute in every phase of the project!
  - Don’t let others freeload on you hard work!
    - Inform me or TA immediately
Reading and Prerequisites

- **Textbook:** Silberschatz, Korth and Sudarshan
  - *Database System Concepts, 6th edition*
  - McGraw Hill
  - publication date: 2006,

- **Prerequisites:**
  - CS 331 or CS401 or CS403
Self-study

- **I expect you to learn by yourself how to effectively use the following technologies**
  - **Git** – a version control system
    - You have to submit your project through git and should also use git to collaborate with your project group members
    - We provide some useful examples/scripts through git
  - **Docker** – a virtualization platform (think VMs, but more lightweight)
    - The easiest way to get postgres running is by using the docker image we provide
  - **PostgreSQL**
    - I expect you to learn how to start/stop/configure a postgres server and how to connect to a running postgres server

- **Help is on the way!**
  - [https://github.com/IITDBGroup/cs425](https://github.com/IITDBGroup/cs425)
PostgreSQL Overview

**Client/Server Architecture**

- **Postgres Cluster**
  - A directory on the machine running the server that stores data and configuration files

- **Postgres Server**
  - A postgres server handles the data of single cluster
  - Clients connect to the server via network (TCP/IP)
    - Send commands and receive results

- **Clients**
  - GUI clients: e.g., PGAdmin (https://www.pgadmin.org/)
  - CLI clients: e.g., the built-in `psql` tool
  - Programming Language Libraries
    - Java: JDBC (https://jdbc.postgresql.org/)
    - Python: `psycopg` (http://initd.org/psycopg/)
    - ...
Get Your Hands Dirty

■ Get a working version of the PostgreSQL server
  ● Your options
    ▶ Install locally
      – Installer packages for windows exists
      – Most Linux distributions have a postgres package
      – Installation from source is not that hard
    ▶ Get our docker image (docker pull iitdbgroup/cs425)
      – It’s an extension of the official postgres image which loads our running example university database

■ Validate your installation
  ● Create a database cluster (the directory PostgreSQL uses to store data)
  ● Check that you can start/stop the server
  ● Check that you can connect to the running server using psql or any other client

■ https://github.com/IITDBGroup/cs425
Jupyter notebook

- **Jupyter notebooks**
  - Notebooks mix documentation and code
  - Over the course of the class I will put SQL examples we discuss in class into a notebook that is shared through the class repository:
    - classnotebook-2017-Fall/CS425-2017-Notebook.ipynb

- **Find the classnotebook**
  - [https://github.com/IITDBGroup/cs425](https://github.com/IITDBGroup/cs425)
Outline

■ Introduction
■ Relational Data Model
■ Formal Relational Languages (relational algebra)
■ SQL
■ Database Design
■ Transaction Processing, Recovery, and Concurrency Control
■ Storage and File Structures
■ Indexing and Hashing
■ Query Processing and Optimization