Today ...

- Overview
- Basic Concepts
**Course Overview**

**Communication:**
- Website: [www.cs.gonzaga.edu/bowers/courses/cpsc321](http://www.cs.gonzaga.edu/bowers/courses/cpsc321)
- Piazza (you should have received an invite)
- GitHub classroom (requires GitHub account)
- Blackboard: grades

**Getting help:**
- **Office hours:** 11–12 T/W/R, 1–2 W, or by appointment
- **Piazza:** preferred for basic questions, clarifications, etc.

**Grading:**
- 10% participation (including attendance)
- 50% homework assignments
- 15% individual project
- 5% quizzes
- 5% midterm exam
- 15% final exam

**No Textbook:** but some readings as part of homework
Homework and Quiz basics

(Some) collaboration on homework is encouraged ...

- discuss and talk about assignments
- work together (as long as you are doing your own work)
- plagiarism not allowed (e.g., do not split up work, share code)
- homework due anytime on the day it is due

Quizzes

- roughly weekly
- individual work
Goals and Topics

Goals for the Course

1. How and when to use (relational) database systems
2. Database design, including basic normal forms and indexes
3. Read and write SQL
4. Use database systems within applications (basics)
5. Hands-on experience with a database system (MySQL) and dynamic SQL

Topics we’ll cover

- Relational data model
- Basic and intermediate SQL
- Logical database design (ER, normalization)
- Basic database internals (storage)
- Physical database design (indexes)
- Transactions and recovery (as time allows)

Our Focus

- system user as an application developer (vs database system implementation)
Some Basic Concepts

1. **Persistence**
   - storage of data beyond (the running of) the application that created it
     - example: saving a document to a file in a word processor

2. **Data Store**
   - a repository for storing collections of data
     - example: a directory of photos you’ve taken

3. **Database**
   - a data store where information is organized / structured
     - example: when, where photos taken, whose in each photo, etc.
   - the organization makes it easier to obtain information and answer questions
     - example: how many pictures was Bob in from 2020–2021?
     ⇒ we’ll give a more specific definition of “database” later ...

4. **Database Management System (DBMS)**
   - a software system for managing databases
   - sometimes just called a “database system”
5. High-Level **DBMS supported operations** performed by a user

- **design** an organization for data (called a “schema”)
- **update** data by adding, removing, modifying
- **retrieve** data (via a “query language”)
- **manage** who has permission to access data

6. **Relational DBMSs** and **SQL**

- store data using the “*relational model*” ... tables of rows and columns
- SQL (structured query language) typically used for DBMS operations

7. **SQL Command (Sub) Languages**

- **Data Definition Language (DDL)** ... CREATE, ALTER, DROP
- **Data Manipulation Language (DML)** ... INSERT, UPDATE, DELETE
- **Data Query Language (DQL)** ... SELECT, WITH
- **Data Control Language (DCL)** ... GRANT, REVOKE (accounts, privileges)

(*) Note that other data models exist as well

- e.g., document, tree, graph, key-value, time-series models
- with their own data languages (or even with SQL)
8. **On-Line Transaction Processing (OLTP)**
   • applications that perform many insert, update, delete, and simple queries
   • usually need fast response times and many concurrent transactions
   • also known as operational databases (support up to minute application needs)

9. **On-Line Analytic Processing (OLAP)**
   • applications that perform more complex queries over large portion of database
   • usually read heavy (with fewer inserts, updates, deletes)
   • OLAP queries sometimes called decision support queries
   • often connected with data warehouses ...

10. **Data Warehouse**
    • data from multiple (e.g., operational) databases loaded into a single database
    • data is typically more static, used for finding long-term (historical) trends
    • sometimes known as analytic databases
    • imply "integration" into a common schema / organization for analytics
    • usually requires extract-transform-load (ETL) activities ("data engineering")
    • alternatives: data lakes, virtualization (federation), hubs

(*) online implies “interactive” (as requests come in), offline implies “batched”
11. DBMSs support **transactional concurrency**
   - a single transaction can involve multiple data reads and writes
   - most DBMSs (seamlessly) handle multiple transactions at once
   - leads to system design and implementation challenges ...

12. **ACID transaction concurrency properties**
   - Atomicity: all or nothing execution of transaction
   - Isolation: transactions execute as if no others execute at same time
   - Durability: transaction effect on DB must not be lost after it completes
   - Consistency: transactions expected to preserve constraints of DB

(*) Relational DBMSs are typically “ACID Compliant” (vs “NoSQL” systems)

Next ...
   - The Relational Model