# CSC 365

## Introduction to Database Systems

# Cal Poly Course Numbers Demystified

CAL POLY

- When there is a need to store one hundred pieces of data, allocate an array of size **101**
  - null-terminators are sneaky; off-by-one errors are a pain

- A good *systems program* executes without fault **357** days in any given year

- An *algorithm* is acceptable if it stays within its complexity bounds **349** days out of the year

A *Database System* must run flawlessly,
**365** days a year, 24 hours per day,
*no exceptions*

**Please note: the first three statements are *entirely* false.**

# What is a Database?



What's the first thing that comes to mind
when you hear the word "database"?

# What is a Database?

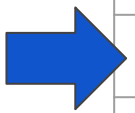


- A **database** is a collection of interrelated information.

- In common usage, *database* refers to a collection of data managed by a **Database Management System DBMS**.

What core functionality should be provided by a database management system?

| | |
|---|---|
| **High Performance** | Thousands of tasks per second |
| **Available** | No downtime / outages |
| **Easy to Use** | Powerful operations on large amounts of data |
| **Safe & Reliable** | Maintains consistency of data, no data loss |
| **Multi-User** | Many users operating simultaneously on the same data |
| **Persistent** | Data is long-lived, retained between program executions |
| **Large Amounts of Data** | Much larger than main memory |
| **General Purpose** | Common tools and techniques for many problem domains, ability to handle wide variety of end-user queries |



| | |
|---|---|
| **High Performance** | Thousands of tasks per second |
| **Available** | No downtime / outages |
| **Easy to Use** | Powerful operations on large amounts of data |
| **Safe & Reliable** | Maintains consistency of data, no data loss |
| **Multi-User** | Many users operating simultaneously on the same data |
| **Persistent** | Data is long-lived, retained between program executions |
| **Large Amounts of Data** | Much larger than main memory |
| **General Purpose** | Common tools and techniques for many problem domains, ability to handle wide variety of end-user queries |

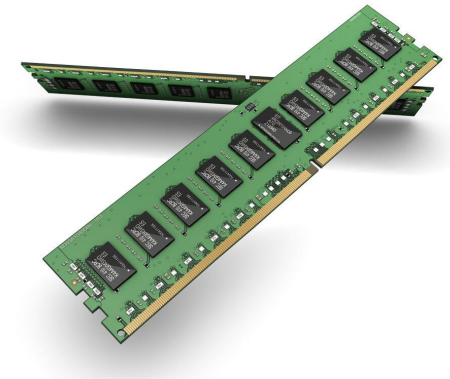
# Handling "Large" Amounts of Data



**Goal**: Efficiently take roll, based on a roster that is sorted alphabetically by first name (last name used to break ties)

**Artificial Constraint**: A maximum of 7 names can be accessed (ie. stored in main memory) at any given point in time.

Formulate an algorithm that can be used to complete this task.



External Sorting (Wikipedia)

| | |
|---|---|
| **High Performance** | Thousands of tasks per second |
| **Available** | No downtime / outages |
| **Easy to Use** | Powerful operations on large amounts of data |
| **Safe & Reliable** | Maintains consistency of data, no data loss |
| **Multi-User** | Many users operating simultaneously on the same data |
| **Persistent** | Data is long-lived, retained between program executions |
| **Large Amounts of Data** | Much larger than main memory |
| **General Purpose** | Common tools and techniques for many problem domains, ability to handle wide variety of end-user queries |

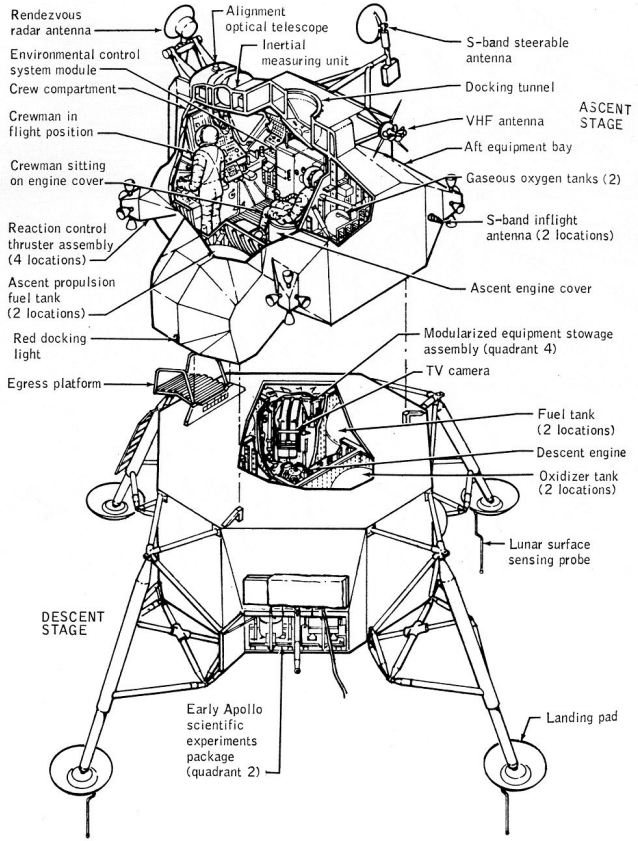
# Space-Age Data Models

CAL POLY

Suppose we'd like to store information about parts and projects, including which parts are used for certain projects.

- For each **part**, we need to track:
  - PartNo
  - Name
  - Quantity on Hand

- For each **project**, we track:
  - ProjId
  - Name
  - Part(s) committed (and quantity of each)




LUNAR MODULE CONFIGURATION FOR INITIAL LUNAR LANDING

# Option 1: Projects Subordinate to Parts

CAL POLY

- Part 1 / Bolt (threaded) / 1000 on hand
  - Project A / Apollo 8 / Qty committed: 100
- Part 2 / Electrical Wiring / 500 ft on hand
  - Project B / Apollo 10 / Qty committed: 100
  - Project C / Apollo 11 / Qty committed: 50
- Part 3 / Insulation / 100 sq ft on hand
  - Project A / Apollo 8 / Qty committed: 10
  - Project B / Apollo 10 / Qty committed: 15
  - Project C / Apollo 11 / Qty committed: 20

- ❑ List all projects to which Part #2 is committed
- ❑ List all parts needed for Project C

# Option 2: Parts Subordinate to Projects

CAL POLY

- Project A / Apollo 8
  - Part 1 / Bolt (threaded) / 1000 on hand / Qty committed: 100
  - Part 3 / Insulation / 100 sq ft on hand / Qty committed: 10 sq ft
- Project B / Apollo 10
  - Part 2 / Electrical Wiring / 500 ft on hand / Qty committed: 100
  - Part 3 / Insulation / 100 sq ft on hand / Qty committed: 15
- Project C / Apollo 11
  - Part 2 / Electrical Wiring / 500 ft on hand / Qty committed: 50
  - Part 3 / Insulation / 100 sq ft on hand / Qty committed: 20

- ❏ List all projects to which Part #2 is committed
- ❏ List all parts needed for Project C

# Option 3: Parts and Projects as Peers, Commitment under Projects

CAL POLY

- Part 1 / Bolt (threaded) / 1000 on hand
- Part 2 / Electrical Wiring / 500 ft on hand
- Part 3 / Insulation / 100 sq ft on hand

- Project A / Apollo 8
  - Part 1 / Qty committed: 100
  - Part 3 / Qty committed: 10 sq ft
- Project B / Apollo 10
  - Part 2 / Qty committed: 100
  - Part 3 / Qty committed: 15
- Project C / Apollo 11
  - Part 2 / Qty committed: 50
  - Part 3 / Qty committed: 20

# Option 4: Parts and Projects as Peers, Commitment under Parts

CAL POLY

- Part 1 / Bolt (threaded) / 1000 on hand
  - Project A / Qty committed: 100
- Part 2 / Electrical Wiring / 500 ft on hand
  - Project B / Qty committed: 100
  - Project C /Qty committed: 50
- Part 3 / Insulation / 100 sq ft on hand
  - Project A / Qty committed: 10
  - Project B / Qty committed: 15
  - Project C / Qty committed: 20

- Project A / Apollo 8
- Project B / Apollo 10
- Project C / Apollo 11

# Option 5: Parts, Projects, Commitment as Peers

CAL POLY

**Parts**

- Part 1 / Bolt (threaded) / 1000 on hand
- Part 2 / Electrical Wiring / 500 ft on hand
- Part 3 / Insulation / 100 sq ft on hand

**Projects**

- Project A / Apollo 8
- Project B / Apollo 10
- Project C / Apollo 11

**Commitment**

- Part 1 / Project A / 100
- Part 2 / Project B / 100
- Part 2 / Project C / 50
- Part 3 / Project A / 10
- Part 3 / Project B / 15
- Part 3 / Project C / 20

# Data Models



- Flat file  (1960s - 80s)
- Hierarchical (1970s - 80s)
- Network/Graph (1970s - )

- **Relational (1970s -  )**

- Object-oriented (1990s - )
- Semi-structured / XML (1990s- )
- Key-value / non-relational / NoSQL / NewSQL (2000s - )

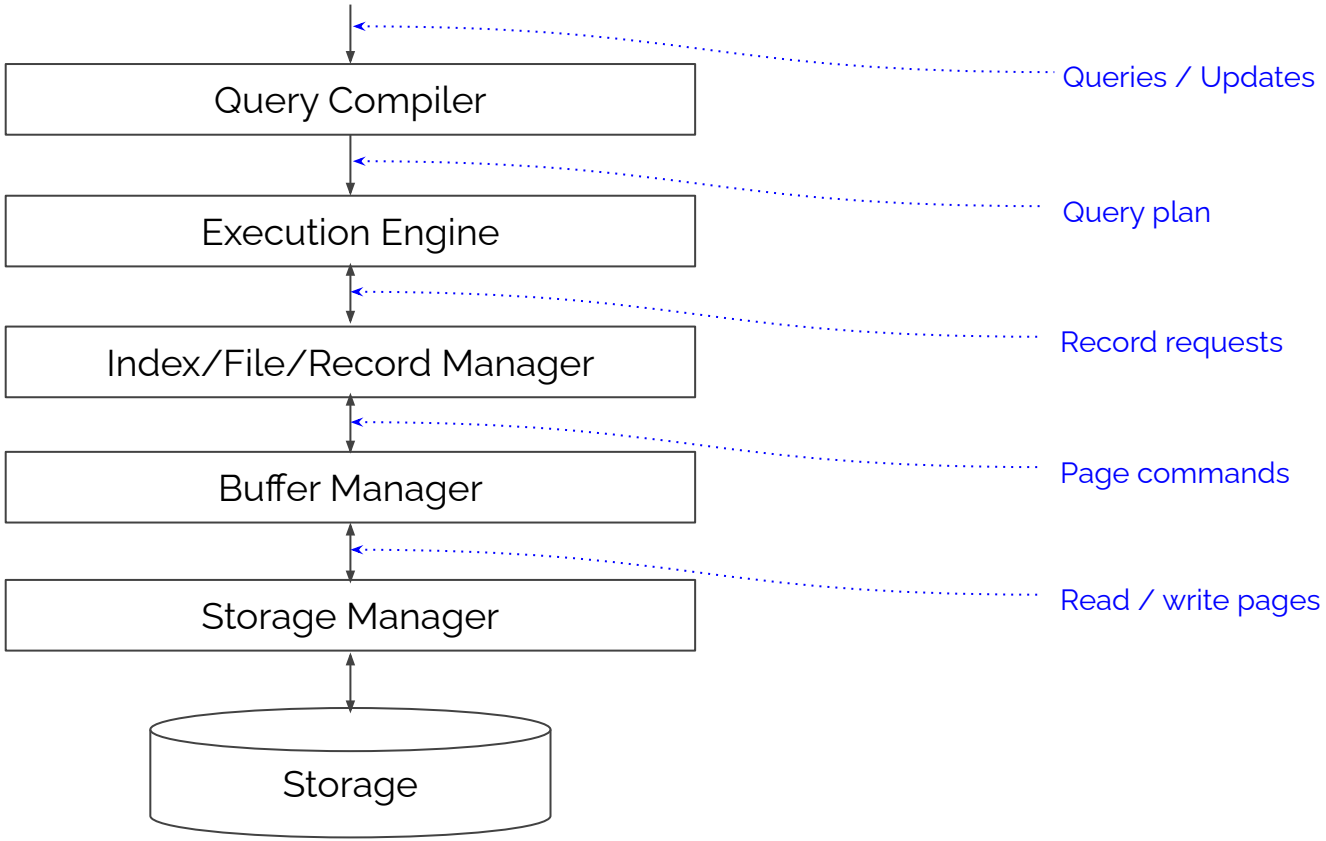
# Database Management System (DBMS)



- Allows users to describe data format (database **schema**)
- Capable of storing *very large* amounts of data
- Provides answers to user information needs (**queries**)
- Offers control over data access by **multiple users**

# DBMS System Components

CAL POLY



# Relational Database Management Systems (RDBMS)



- Many implementations:
  - Oracle
  - Microsoft SQL Server
  - Microsoft Access
  - IBM DB2
  - MySQL
  - PostgreSQL
  - SQLite

- All based on the same fundamental concepts (relational model, SQL)
- Vendor-specific extensions

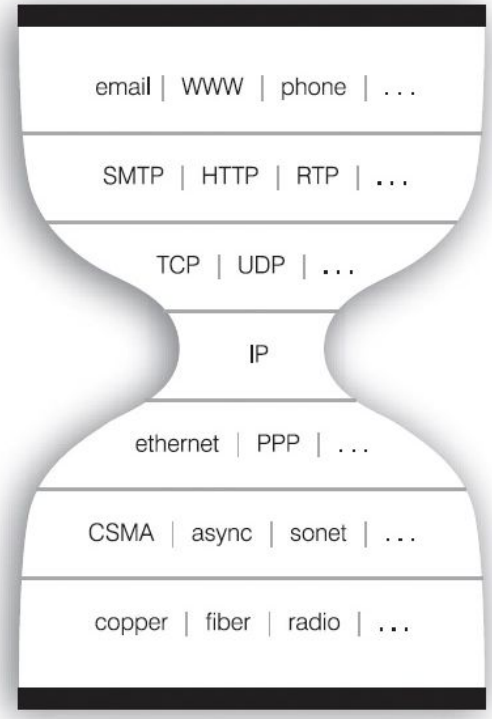
# High-Level Course Outline

CAL POLY

1. Data and Data Management, Relational Data Model
2. Relational Algebra
3. Structured Query Language (SQL)
4. Database Connectivity: Java (JDBC), Python (DB-API)
5. DBMS Implementation Topics:
    a. Transaction Control
    b. Query processing
    c. RDBMS architecture

# Networking "Hourglass" *(for context)*

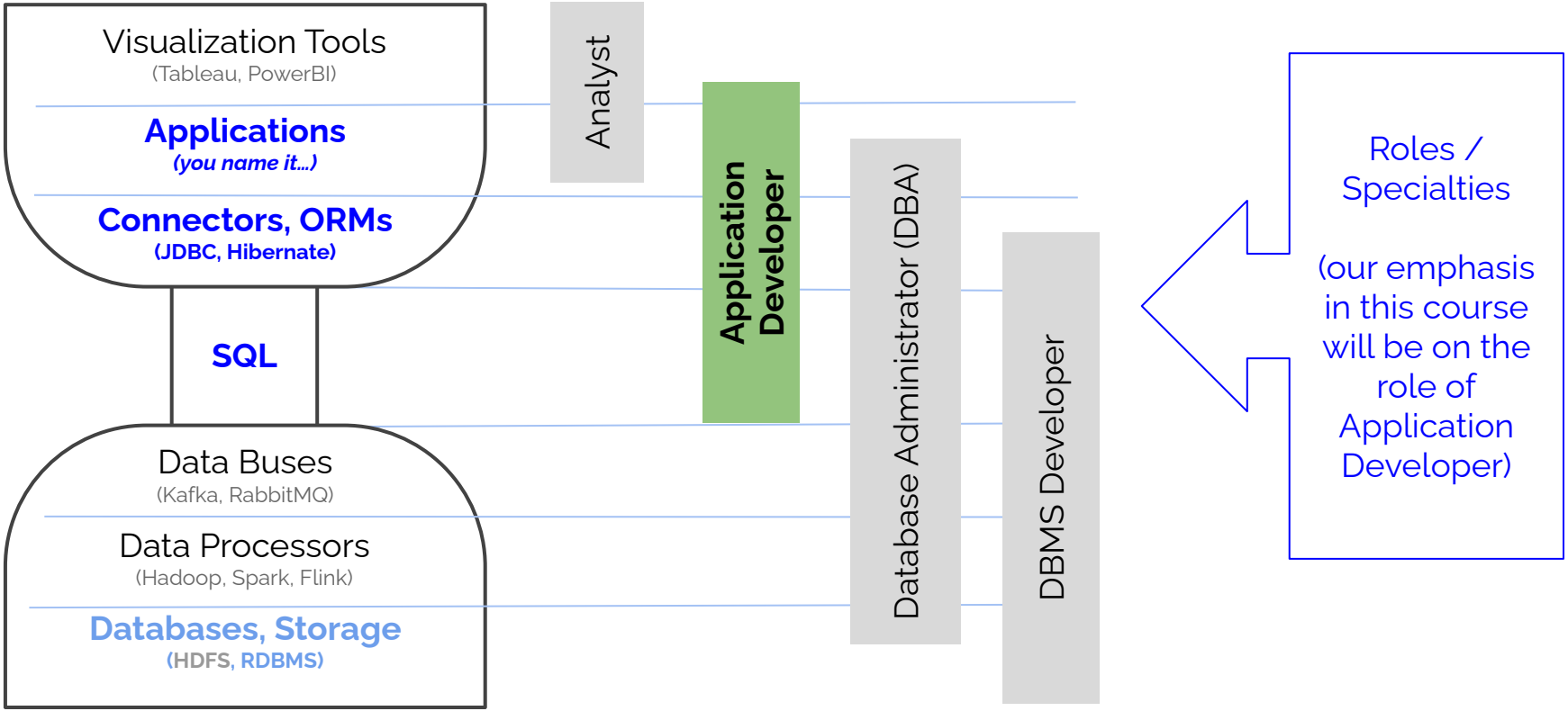
CAL POLY




Source: https://www.w3.org/DesignIssues/diagrams/layers/IP-hourglass-zittrain.png

# SQL / Data Management Hourglass

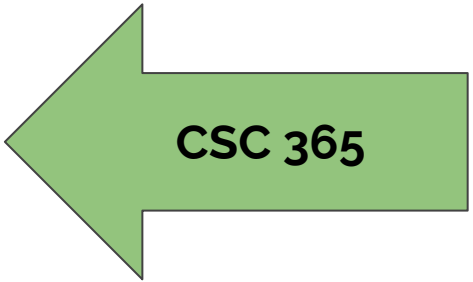
CAL POLY



# Views of Data



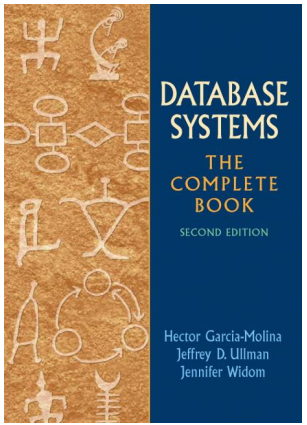
- **Conceptual**: captures relationships in data (CSC 366)
- **Logical**: captures the format of the data as understood by the Database Management System (DBMS)
- **Physical**: represents the exact way in which data is stored and accessed by the DBMS (CSC 468)

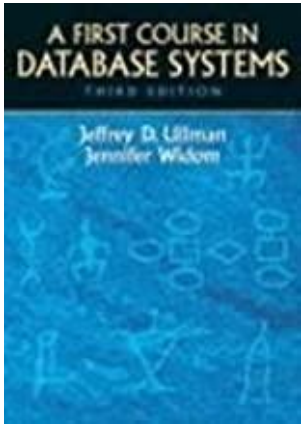


# Syllabus / Schedule Overview

CAL POLY

- Communication
  - I encourage you to use the Canvas forums for general-interest questions
- Textbook (optional)
- Homework & Labs
- Quizzes & Exams
  - Quizzes every other week
  - Comprehensive Final Exam
    - Lecture (theory)
    - Lab (practical exercises)



# Prerequisite Topics



- Data structures, algorithms, discrete math
  - Algebraic expressions and laws (associative, commutative, distributive)
  - Logic, sets
  - Data structures

- Java / Python