Interests

Petabyte scale data systems (from cs145 -> Infolab -> now)

Building new data systems, products (and teams)

- Scaled to billions of consumers, billions of ad $s, millions of web publishers, trillions of data rows, million QPS systems
- E.g., AdSense, Search, Dremel/BigQuery, Gmail/Google Apps, Sitemaps, Warp, Google Maps, Healthcare data, etc.
Hello TAs

Class Logistics
http://cs145.stanford.edu
Course Summary

We’ll learn How To…

- **Query** over small-med-large data sets with **SQL**? [Weeks 1 and 2]
  - On relational engines, and “big data” engines (e.g, MySQL, BigQuery, SPARK-like)

- **Scale** for big data sets? On Cloud Clusters? [Weeks 3, 4, 5]
  - Analytics (“Online Analytic Processing – OLAP”, 1st principles of scale)

- **Update** data sets? [Weeks 6, 7]
  - Writes, Transactions (“Online Transaction Processing - OLTP”), Logging, ACID properties

- **Design** “good” databases? [Weeks 8, 9]
  - Schema design, functional dependencies, query optimizers

**Project**: Query-Visualize-Learn on GB/TB scale data sets on a Cloud [sql + python]
Grading breakdown for CS 145?

- Projects: 50% (10 + 15 + 25)
- Test: 15% [Oct 28], Finals: 25% [Dec 9, 7-10pm]
- Problem set (4): 10% (ungraded, turn in on time)

[Bonus credit to students with insightful piazza/in-class participation]

Difference between problem set and projects?

**Projects** (3)
- apply class material in a real world manner on large data
- 2 late days
- Students -- “Very practical”, “creative outlet”, ...

**Problem Set** (4)
- accompany the material taught in class; self-grading
- Students -- “Best practice material for Tests”

Discussion sections?

TAs will do 4 biweekly sections to accompany the release of each problem set. They are optional. See online schedule for dates/times.
Join our Piazza: [here](#)

Add CS 145 in Gradescope with code: **P5BVKB**

Check that you are added to Canvas.

Get your GCP credits for projects -- [instructions](#)

If you require special accommodations (OAE) for exams, please email vatong AT stanford.edu.

Review [Stanford Honor Code](#) and [Stanford CS Dept Honor Code Rules](#).
We follow the Stanford Honor Code and Stanford CS Dept Honor Code Rules. Any work submitted for grading should not be derived from or influenced by the work of others. All submissions are subject to plagiarism detection tools. Per university policy, suspected violations are referred to the Office of Community Standards. For more information regarding the honor code policy, please refer to the course website.

Examples of honor code violations include (but are not limited to):

- reusing your own or another student’s assignment work from previous quarter
- sharing codes for assignments and projects
- sharing your responses/answers/code/design with other students nor publicly
- joint development/debugging
- use of web or public resources for public solutions
- copying code or answers
- posting up/dispersing your solutions or code on public repos

If you have any questions about the honor code and expectations, please reach out to the teaching staff via piazza and we will be happy to clarify for you.”
Applications of DBs and Data systems

Properties of general DBs, special-purpose DBs, data lakes

Unpack a DB: Example of a mobile game using a DB
- For Whom and Why?
- Sample data architectures
Details + Big picture

Focus on ‘atomic’ examples

Take in big picture, flavor of issues, how pieces fit
Example: Youtube DB
Every minute on the Internet
Example Self Driving Cars
The coming flood of data in autonomous vehicles

- Radar: ~10-100 KB per second
- Sonar: ~10-100 KB per second
- GPS: ~50 KB per second
- Cameras: ~20-40 MB per second
- Lidar: ~10-70 MB per second

Autonomous vehicles: 4,000 GB per day... each day

Intel
Example

Unpack Cars DB

Front panel metrics
Speed, distance
ETA

Read

Road models
Drive models

Modify

Collisions
Traffic signals

Learn

Cars DB
Example

Unpack ATM DB: Transaction

- Read Balance
- Give money
- Update Balance

vs

- Read Balance
- Update Balance
- Give money
Scenarios
1. Crash before 1?
2. After 1 but before 2? [Bad!! a10: 17,000, a20: 15,000]
3. After 2?

Transfer $3k from a10 to a20:
1. Debit $3k from a10
2. Credit $3k to a20

<table>
<thead>
<tr>
<th>Acct</th>
<th>Balance</th>
</tr>
</thead>
<tbody>
<tr>
<td>a10</td>
<td>20,000</td>
</tr>
<tr>
<td>a20</td>
<td>15,000</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Acct</th>
<th>Balance</th>
</tr>
</thead>
<tbody>
<tr>
<td>a10</td>
<td>17,000</td>
</tr>
<tr>
<td>a20</td>
<td>18,000</td>
</tr>
</tbody>
</table>
Goals of Standard Databases

Platform to store, manage data

Supporting:
- Scale
- Speed
- Stability
- Evolution
- Reliability
- Cost efficiency

Operations (any DB)
- Read
- Learn
- Modify

Platform to store, manage data
Goals of Special Databases

DBs are often optimized for key use cases

- Store current data (e.g., lot of reads)
- Optimize historical data (e.g., logs)
- Run batch Workloads (e.g. training)

> 100 viable data engines on market
(MySql, Postgres, Oracle, IBM/SAP to data clouds on AWS/Azure, GCP, to Spark, Cockroach/Spanner, Mongo, ….)
For Whom?
For What?
How?
Applications of DBs and Data systems

Properties of general DBs, special-purpose DBs, data lakes

Unpack a DB: Example of a mobile game using a DB
  - For Whom and Why?
  - Sample data architectures
How?

Example

Game App

DB v0

Q1: 1000 users/sec?
Q2: Offline?
Q3: Support v1, v1' versions?

Q4: Which user cohorts?
Q5: Next features to build?
Q6: Predict ads demand?
Q7: How to model/evolve game data?
Q8: How to scale to millions of users?
Q9: When machines die, restore game state gracefully?

App designer

Real-Time User Events

DBMS

DB

Report & Share Business/Product Analysis

Systems designer

Product/Biz designer
How?

Example Game App

Data system “v1” on Cloud

1 Log user actions
2 Store in DB, after Extract-Transform-Load
3 Run queries in a peta scale analytics system
4 Visualize query results
How?

Example Game App

Data system

“How?”

0 Log user actions
In local DB

1 Data sync to cloud

2 Store in DB, after ETL

3 Run queries in a petabyte scale analytics system

4 Visualize query results

“v2” Cloud + Local

Data system

Mobile Game

Real-Time User Events

Local DB

Data Sync

Data Processing
MySQL, Dataflow

Analytics Engine

BigQuery

Cloud Datalab

Report & Share Business/Product Analysis
Summary

Data bases

Data systems

Data lakes

DBs - General + Optimized

Data System - Connect DBs to solve a problem

Data Lake - Set of Data Systems for different data (e.g., Netflix has HD movies (1GB?) and user logs)
Why build a data lake on Amazon S3?

Amazon S3 is designed for 99.999999999% (11 nines) of data durability. With that level of durability, you can expect that if you store 10,000,000 objects in Amazon S3, you should only expect to lose a single object every 10,000 years. The service automatically creates and stores copies of all uploaded S3 objects across multiple systems. This means your data is available when needed and protected against failures, errors, and threats.

- **Security by design**
  Protect data with an infrastructure designed for the most data-sensitive organizations

- **Scalability on demand**
  Instantly scale up storage capacity, without lengthy resource procurement cycles

Durable against the failure of an entire AWS Availability Zone
Automatically store copies of data across a minimum of three Availability Zones (AZs). To provide fault tolerance, Availability Zones are separated by several miles—but no more than a hundred to ensure low latencies.

AWS services for analytics, HPC, AI, ML, and media data processing
Use AWS native services to run applications on your data lake

Integrations with third-party service providers
Bring preferred analytics platforms to your S3 data lake from the APN.

Wide range of data management features
Comprehensive flexibility to operate at an object level while managing at scale, configure access, enable cost efficiencies, and audit data across an S3 data lake.
Amazon S3 data lake lifecycle

A data lake built on Amazon S3 lets you store everything in one place, dive into your data with flexible access, future-proof your storage, and connect to powerful insights.

Ingest and store data
- Migrate data from a variety of data sources
- Real-time data movement
- Remove silos with one data lake for structured and unstructured data
- Unmatched scale, durability, security, and performance

Catalog and transform data
- Know your data with better management and higher quality data
- AWS Glue crawls, catalogs, and indexes data for searchability
- AWS Glue automates the effort in building, maintaining and running ETL jobs

Analyze
- Run AWS analytics and machine learning services to gain insights
  - Amazon Athena
  - Amazon Redshift
  - Amazon SageMaker
  - Amazon Rekognition
  - Amazon FSx for Lustre
  - Amazon EMR

Extract value from data
- Improve customer interactions
- Guide R&D innovation choices
- Maximize operational efficiencies

Data Lake - Set of Data Systems for different data (e.g., Netflix has HD movies (1GB?) and user logs)
Example applications of DBs and Data systems

Properties of general DBs, special-purpose DBs, data lakes

Unpack a DB: Example of a mobile game using a DB
  - For Whom and Why?
  - Sample data architectures
Course Summary

We’ll learn How To...

- **Query** over small-med-large data sets with **SQL**? [Weeks 1 and 2]
  - On relational engines, and “big data” engines (e.g, SPARK-like)

- **Scale** for big data sets? On Clusters? [Weeks 3, 4, 5]
  - OLAP/Analytics, 1st principles of scale

- **Update** data sets? [Weeks 6, 7]
  - Writes, Transactions, Logging, ACID properties

- **Design** “good” databases? [Weeks 8, 9]
  - Schema design, functional dependencies, query optimizers

**Project:** Query-Visualize-Learn on GB/TB scale data sets on a Cloud [sql + python]