Recap

- "Classical" relational databases
  - The relational data model and algebra: bags and sets
  - SQL Queries, Modifications, DDL
  - Database Design
  - Views, constraints, triggers, and indexes
  - Query processing & optimization
  - Transactions
- Non-classical data systems
  - Data preparation:
    - Semi-structured data and document stores
    - Unstructured data and search engines
  - Data Exploration:
    - Cell-structured data and spreadsheets
    - Dataframes and dataframe systems
    - OLAP, summarization, and visual analytics
  - Batch Analytics:
    - Compression and column stores
    - Parallel data processing and map-reduce
    - Streaming, sketching, approximation
  - Special Topics:
    - Graph processing systems
    - Security and Privacy

Recap: The Classics

- "Classical" relational databases
  - The relational data model and algebra: bags and sets
  - SQL Queries, Modifications, DDL
  - Database Design
  - Views, constraints, triggers, and indexes
  - Query processing & optimization
  - Transactions

- The key takeaways for relational databases:
  - Tables and relational algebra as a basis for data-based computation
  - The notion of "declarative" querying
  - Query processing and optimization automatically handled for you
  - Transactions for supporting updates
  - Ways to control performance and enforce constraints: views, indexes, constraints, triggers

Non-Classical Data Systems

- Various axes of relaxation/specialization for structure or workload
- Structure
  - Tabular Structure
    - Relational to semi-structured (JSON, doc stores) to unstructured (text search) to somewhere-in-between (graph-structured)
  - Schema presence/absence
    - Schema present to schema flexible (dataframes) to no schema (spreadsheets)
- Workload
  - Workloads focused on reading/writing
    - Read-write to read-only with flexible workload (col stores) to read-only with fixed workload (OLAP) to read-only exploration-oriented workloads (visual analytics)
  - Workloads at scale
    - Normal single-node workloads to large workloads (parallel dbs and map-reduce) to extremely large workloads (streaming, approximation)

- Non-classical data systems
  - Data preparation:
    - Semi-structured data and document stores
    - Unstructured data and search engines
  - Data Exploration:
    - Cell-structured data and spreadsheets
    - Dataframes and dataframe systems
    - OLAP, summarization, and visual analytics
  - Batch Analytics:
    - Compression and column stores
    - Parallel data processing and map-reduce
    - Streaming, sketching, approximation
  - Special Topics:
    - Graph processing systems
    - Security and Privacy

What did I not cover

- From the classical side:
  - Database internals stuff
  - Details about query optimization, indexes, …
  - Details about functional dependencies, normalization
  - Various forms of logging and recovery; concurrency control
- From the non-classical side:
  - Transaction processing at scale and weaker consistency models
  - Virtual machines, containerization, …
  - Resource management, pricing, cloud computation
  - Deep learning systems

What I hope you took away from the class!

- An appreciation for the robustness and generality of RDBMSs
  - While understanding that new scenarios may require tweaks to many fundamental relational database assumptions
- An understanding for design decisions underlying data systems, capturing issues ranging from ease-of-use to scalability to flexibility
- Hands-on experience with:
  - At least three types of data systems, plus conceptual understanding of many more
  - Building an end-to-end data application powered by data systems

- This is an amazing time to be studying data systems and data engineering!
  - Data rules everything around us, and infrastructure to support analysis on data is going be increasingly essential even in the era of AI and ML

Where can I go from here?

- CS186/286a: focuses on database internals
- CS286b: research-oriented database internals class

- No better way to learn more about data systems than by building them from scratch!
  - Or contributing to existing open-source efforts
  - Happy to talk about research on developing data systems if you're interested!