Some Reminders for a Seamless Online Class…

- Please turn on your video
- Mute yourself (press and hold spacebar when you’d like to talk)
- Don’t do anything you wouldn’t do in an in-person class
- I will occasionally check the chat for messages if you’d like to share there instead
- Please say your name before you speak

Recap

- Data-savviness is the future!
- "Classical" relational databases
    - Notion of a DBMS
    - 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
    - Semi-structured data and document stores
    - Unstructured data and search engines
    - Cell-structured data and spreadsheets
    - Next: Dataframes and dataframe systems

So far…

- After relational/structured data, we've studied
  - Semi-structured data, where the schema is nested, flexible, and non-atomic…
  - Cell-structured data, where there is no schema, and cells can be filled in with data or with computation
- We're now going to look at yet another way to relax requirements from a database

Classical Database Assumption III

- *Data systems should manage data in relations that are unordered, with a predefined and rarely-modified schema, with queries that operate on relations as a whole, and primarily leverage relational algebra.*

- We'll consider dataframe systems, where these assumptions are relaxed before returning to these assumptions.

Any users of dataframes?

- Who here has used Pandas (or another dataframe library), and what for?

Relational databases are painful for exploratory data science

- From a data model standpoint:
  - Data needs to be defined "schema-first": each column needs to have a pre-declared type

- From a querying standpoint:
  - The declarative nature of SQL makes it hard to incrementally construct and debug queries
    - Direct manipulation is one possible solution but for those conversant in programming may be overkill, especially on large data
  - Hard to mix-and-match programming (e.g. python) and dataframes
  - Hard to employ linear algebra (e.g., transpose, matrix multiplication)

What are dataframes?

- Dataframes are an abstraction (a data model and query language) used to
  - represent, structure, clean, and analyze data
  - during exploratory data analysis and data science
- Perform relational, cell-structured, and linear algebraic operations
- The convenience, flexibility, and power of dataframes make it a one-stop-shop for all stages of data science
- Dataframes are embedded in conventional programming languages like Python or R.
  - Pandas, Python's dataframe, short for "Python Data Analysis Library"
  - R comes bundled with an native dataframe library

Dataframes: Popularity

- Python's popularity (now exceeding Java and C++) has been attributed to that of Pandas
- Pandas has been
  - downloaded 200M times
  - a dependency for 150K+ packages on GitHub

- R's dataframe libraries are similarly popular
- All the more reason for us to study it!

Dataframes: History

More in our paper [Petersohn, …, P., 2020]

- S: the predecessor of R was developed in Bell Labs in 1976: a prog. language for statistics
- Dataframes were added by Chambers and Hastie to S in 1990
  - *"We have introduced into S a class of objects called data.frames, which can be used if convenient to organize all of the variables relevant to a particular analysis …"*
- They then wrote a book about it in 1992, where they state:
  - *"Data frames are more general than matrices in the sense that matrices in S assume all elements to be of the same mode—all numeric, all logical, all character string, etc"*
  - *"... data frames support matrix-like computation, with variables as columns and observations as rows, and, in addition, they allow computations in which the variables act as separate objects, referred to by name."*
- To them, dataframes were removing a key limitation of matrices, which is that all data is of the same type.
  - Of course, relations don't have this limitation, but they also do not support matrix-like computation.

Dataframes: History

More in our paper [Petersohn, …, P., 2020]

- Dataframes were added by Chambers and Hastie to S in 1990
  - To them, dataframes were removing a key limitation of matrices, which is that all data is of the same type.
    - Of course, relations don't have this limitation, but they also do not support matrix-like computation.
- R was released in 1995 (with a stable version in 2000) as an open-source implementation of S, gaining popularity in the statistics community
- Wes McKinney brought dataframe capabilities to Python in the form of Pandas in 2008.

Dataframe Basic Demo

- The demo will show the fact that dataframes mix features from databases, spreadsheets, and matrices, all in a convenient syntax in a conventional programming language

Next Week

- Dataframe data model & query language
- Comparison with relational algebra
- Comparison with spreadsheets
- Comparison with linear algebra
- Functionality explosion
  - Challenges in learning, optimization
- Limitations of dataframes & when to use them