Data Management and Versioning

Robert Clements

MSDS Program

University of San Francisco

What to Expect

- Goal: to learn about the importance of data versioning in the model development process.

- How: in the lab we will use the very popular DVC (data version control) tool.

- Note: we are not going to build data pipelines (data engineering) but instead use version control to keep track of our data used for our models.

NAS, Network Drives, File systems

All types of files, just like on your laptop or cloud drive

NAS, Network Drives, File systems

Object Storage (S3, Azure Blob, GCS)

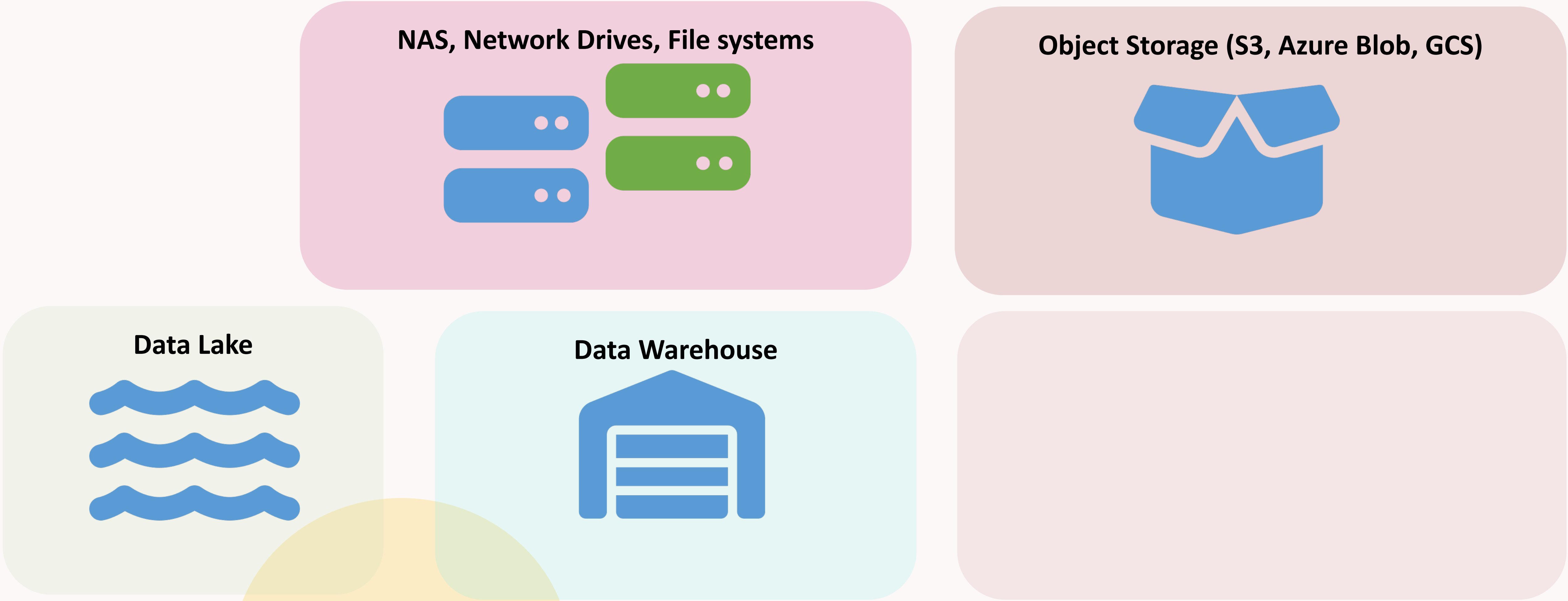
Similar to file system, store binaries, with redundancy and security.

NAS, Network Drives, File systems

Object Storage (S3, Azure Blob, GCS)

Data Lake

Dumping ground for raw data.



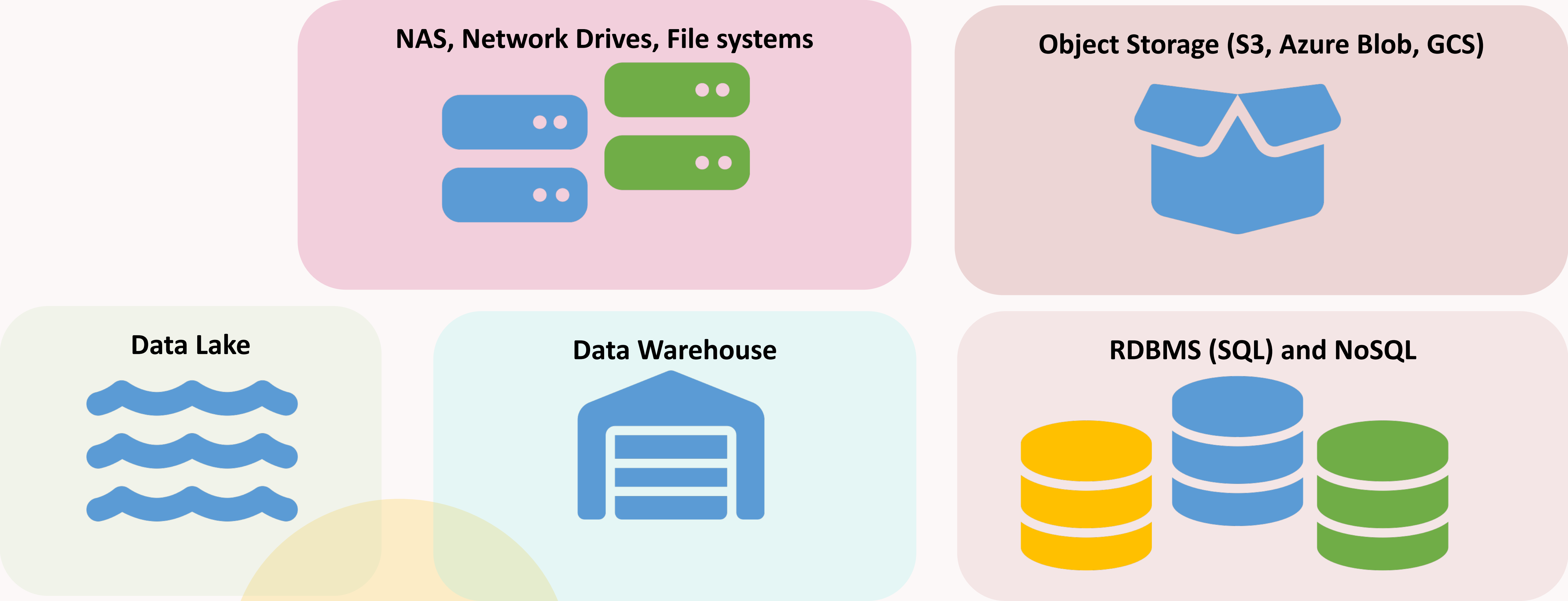
NAS, Network Drives, File systems

Object Storage (S3, Azure Blob, GCS)

Data Lake

Data Warehouse

Nice, clean data using the extract-transform-load process.



NAS, Network Drives, File systems

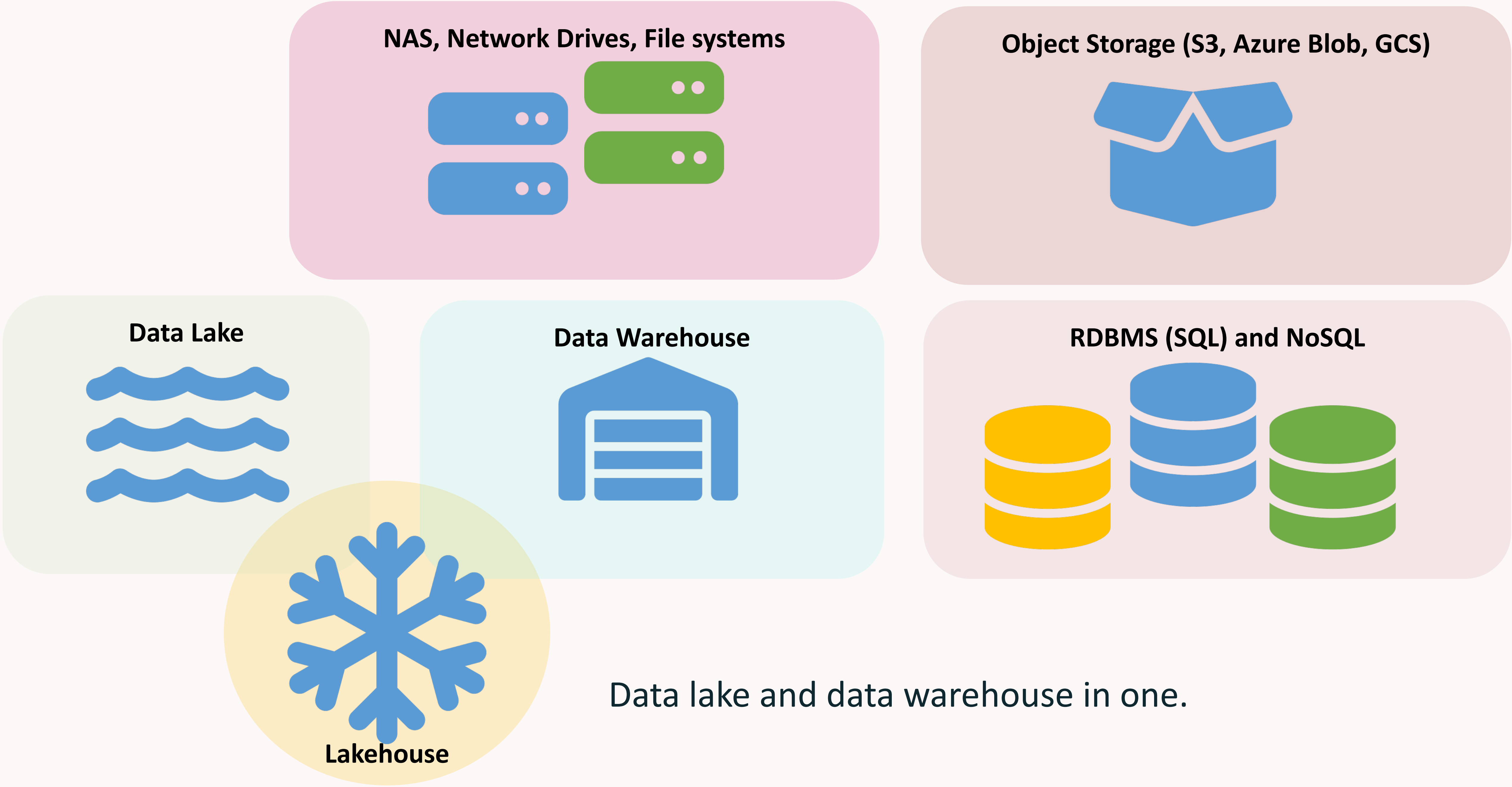
Object Storage (S3, Azure Blob, GCS)

Data Lake

Data Warehouse

RDBMS (SQL) and NoSQL

Structured, semi-structured, unstructured and persistent data for analytics.



NAS, Network Drives, File systems

Object Storage (S3, Azure Blob, GCS)

Data Lake

Data Warehouse

RDBMS (SQL) and NoSQL

Lakehouse

Data lake and data warehouse in one.

Data Pipelines

Though we won’t be building pipelines, it’s useful to know the main tools involved here tend to be Airflow, Prefect, dbt, Dagster, Metaflow

Data Version Control

- Likely to iterate through many versions of data during development process
- Ideally can tie data to model/experiment
- data_v1.csv, data_v2.csv or dev_data.temp1, dev_data.temp2, etc. is bad practice and error-prone
- Recreating intermediate and final datasets from scratch is an option
  - True reproducibility
  - Sometimes not possible if org has bad data practices
- A good tool should make it easy to log and find a dataset used for a particular experiment

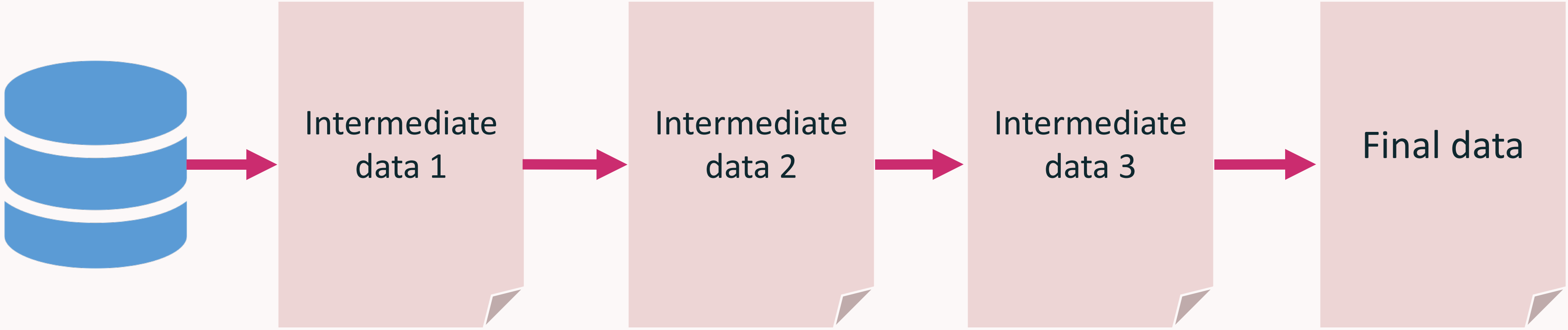
Data_v1
Data_v1.1
Data_v2
Data_v3
Data_v3.1

DVC

- Two main options: Git Large File Storage (LFS) and Data Version Control (DVC)
- DVC is integrated with DagsHub, which we will look at later
- DVC is similar to git
- CLI and VS Code extension
- Works on more than just data (e.g. models and experiments), but we’ll only use it for versioning data

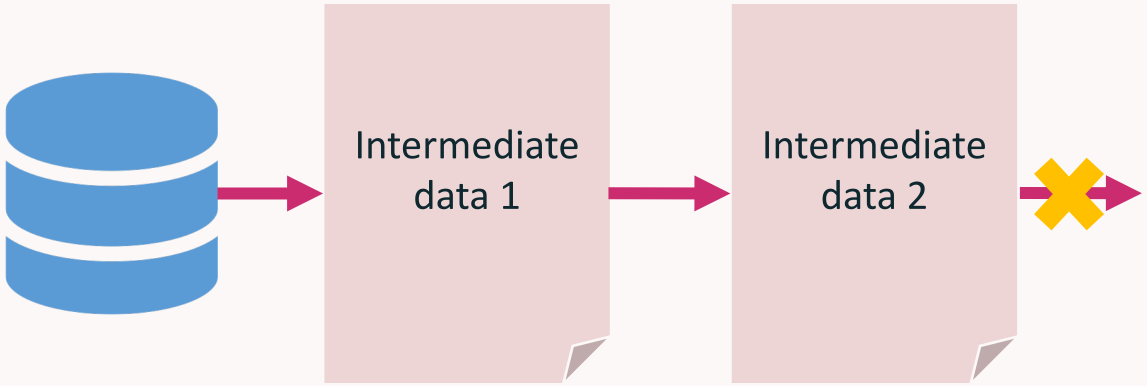
Reproducible Pipelines

- All data should be reproducible, nothing adhoc



Reproducible Pipelines

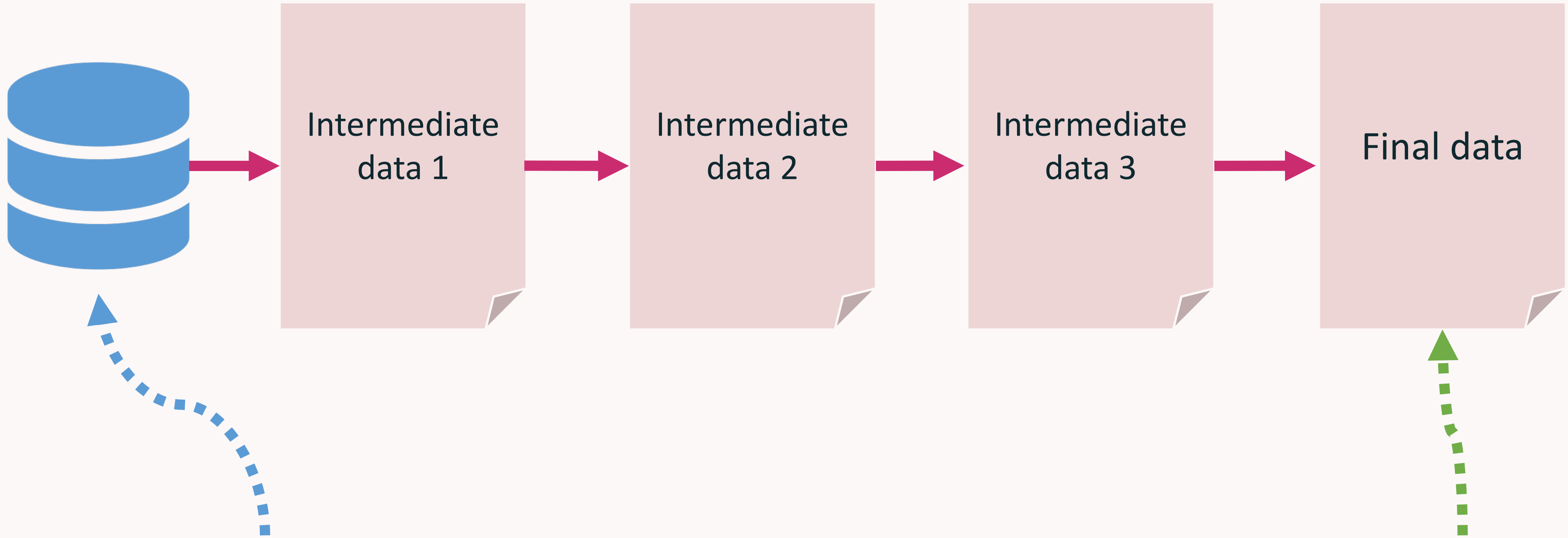
- All data should be reproducible, nothing adhoc



Oops, our data got deleted, corrupted, or accidentally modified. We should be able to recreate it.

Reproducible Pipelines

- All data should be reproducible, nothing adhoc



So long as **this** doesn’t change, we should be able to get back to **this** with code, without needing DVC, and without needing the intermediate data sets.

DVC Demo

Data Versioning Lab