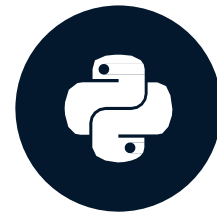


# Introducing data pipelines

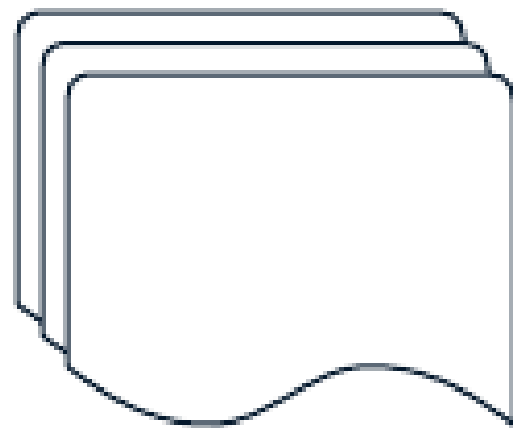
INTRODUCTION TO DATA PIPELINES



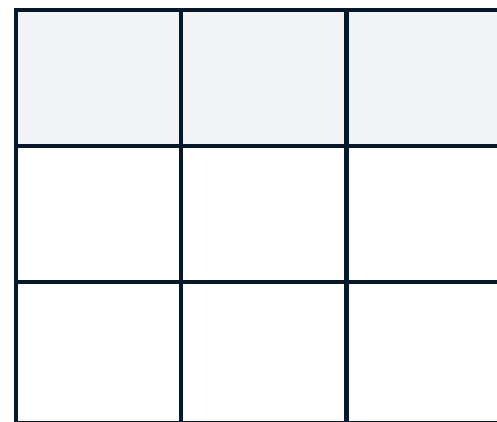
# What is a data pipeline?

## Definition:

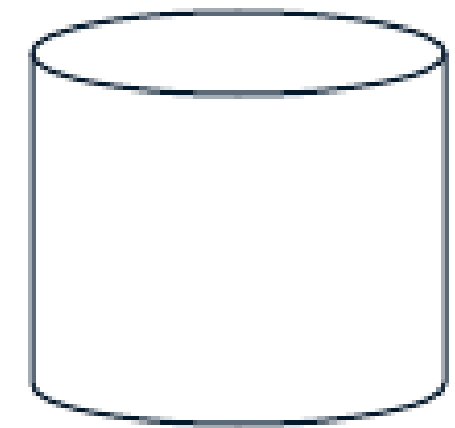
- An **automated** process that extracts data from a source system, transforms it into a desired model, and loads the data into a file, database, or other data storage tool.



**Extract**



**Transform**



**Load**

# Data pipeline users

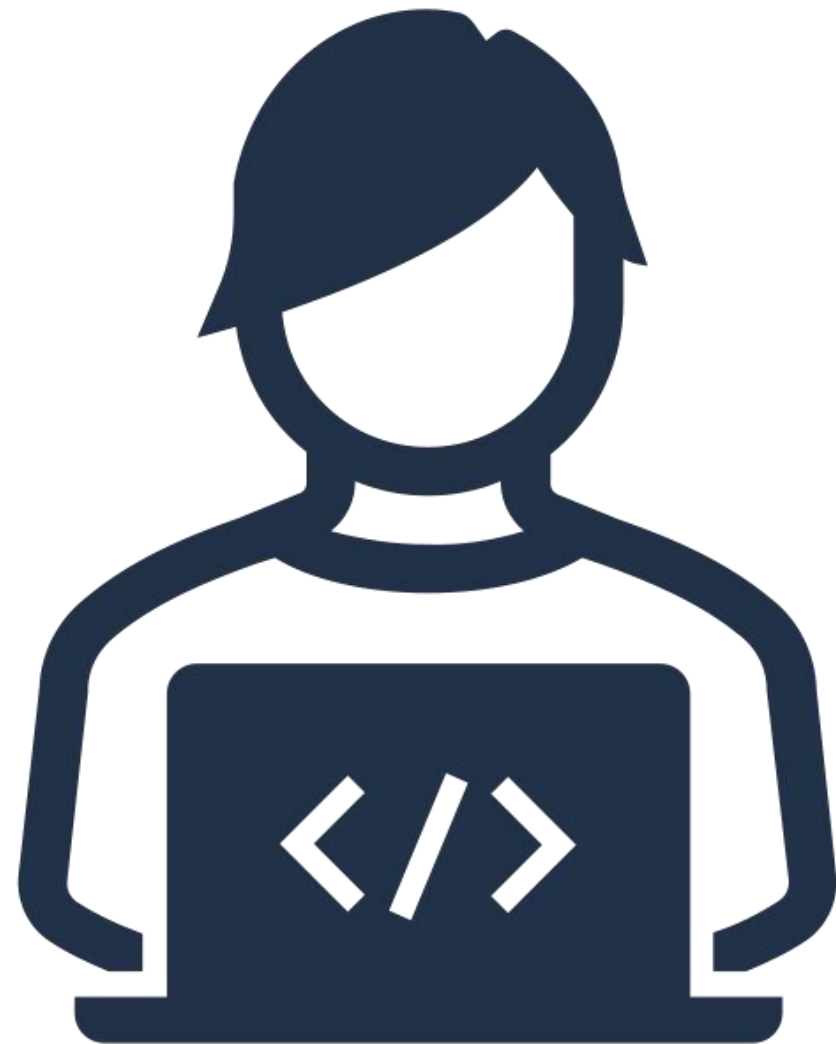
Data pipelines are used by:

- Data Analysts
- Data Scientists
- Machine Learning Engineers
- Business Intelligence Analysts
- ... other data pipelines

Data pipelines are used to:

- Populate dashboards.
- Build machine learning models
- Support ad hoc analyses.
- ... pretty much anything data-related!

# Building data pipelines



Data Engineers build data pipelines!

- Help to power data-driven insights and decisions
- Use tools such as Python and SQL
- Leverage orchestration tools, such as Airflow

# Components of an ETL pipeline

## Extract

- Pull data from a source system
- File, database, API

## Transform

- Convert raw data to desired model
- Simple and advanced transformations

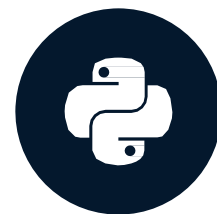
## Load

- Persist data for downstream use
- Files, data warehouse, API



# Designing data pipelines

INTRODUCTION TO DATA PIPELINES



# Architecture diagrams

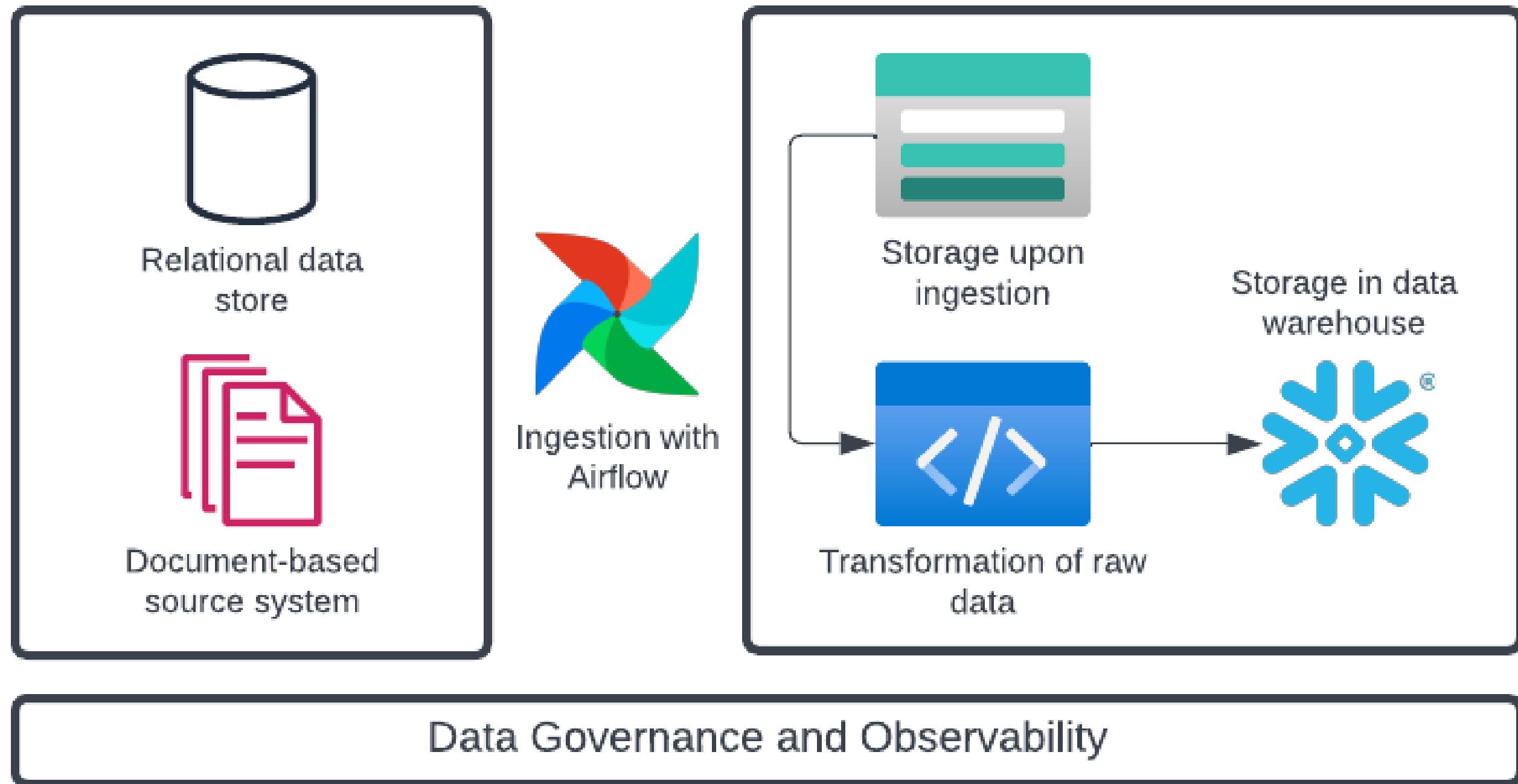
## Architecture diagrams:

- Visually represent components in a data pipeline
- Illustrates dependencies between steps in a data solution
- Identifies technical or security vulnerabilities

## Useful for:

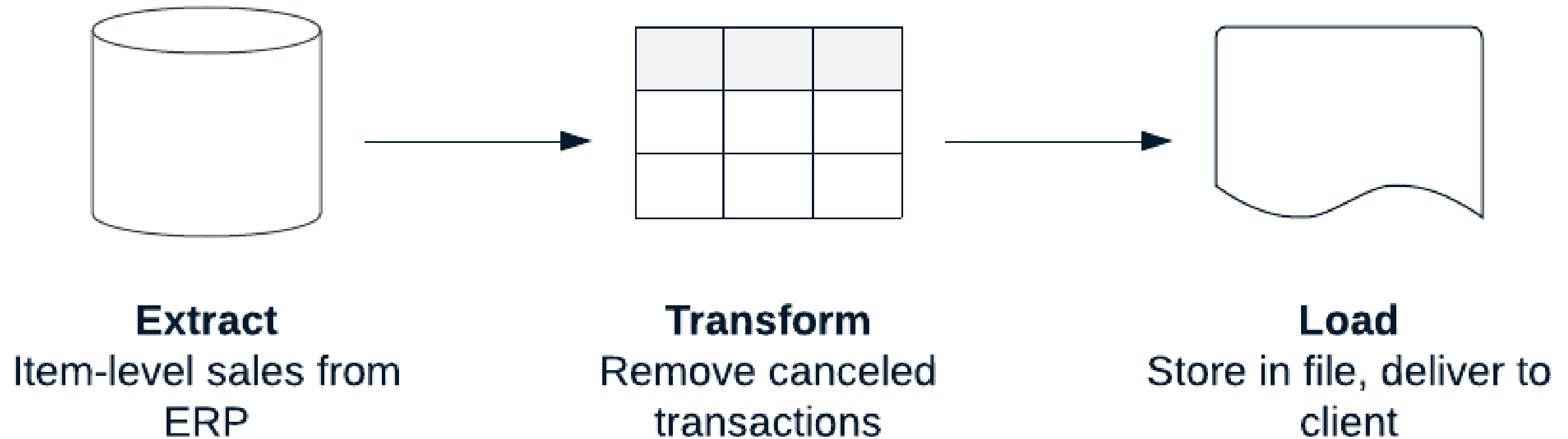
- Determining where previous tooling/logic can be applied
- Ensuring requirements are met before development begins
- Sharing pipeline design with non-technical stakeholders

# Architecture diagrams

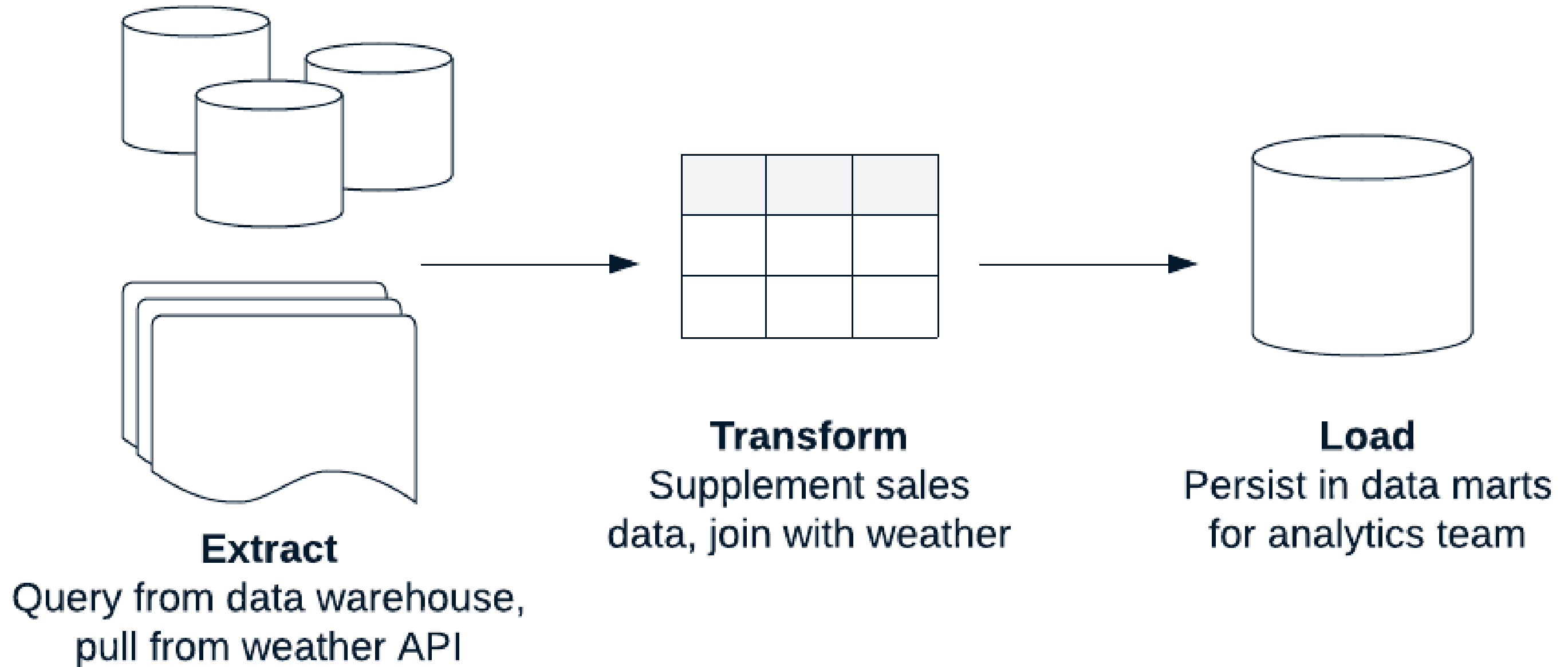




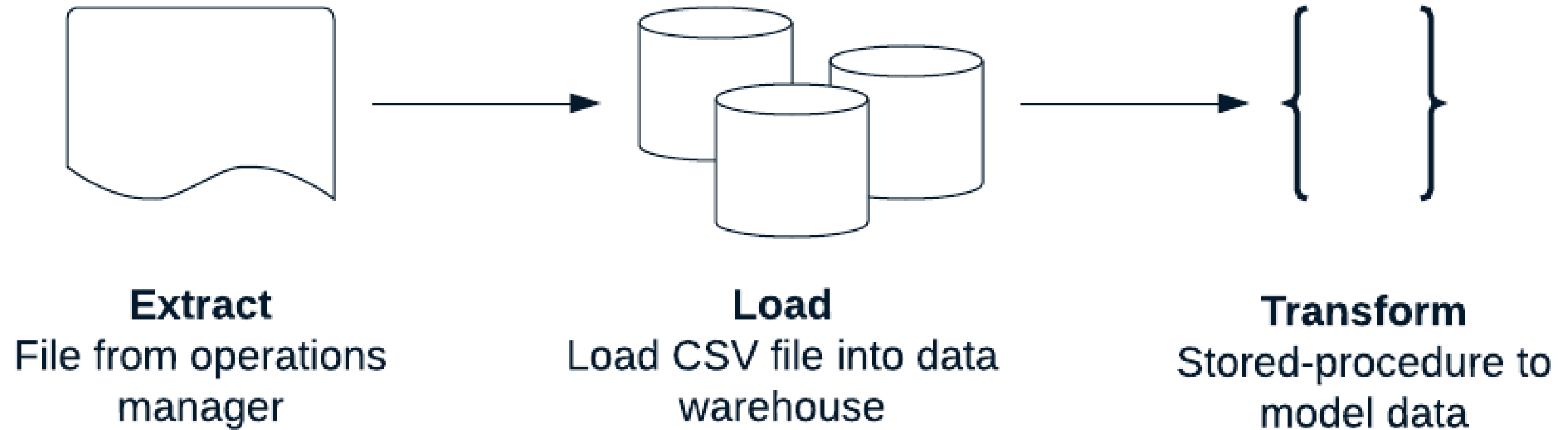
# Data delivery pipeline



# Data ingestion pipeline



# Designing ELT pipelines



ELT is best applied when:

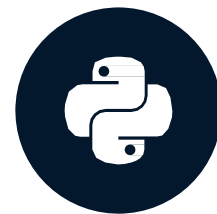
# Designing data pipelines

A thorough **design process** helps to ensure that the right solution is implemented, and can mitigate wasted time later in the development process.

- Understand source system details
- Determine nature of transformations
- Ensure persisted data is accessible
- Align stakeholders on implementation details

# Qualities of great data pipelines

INTRODUCTION TO DATA PIPELINES



# Building quality data pipelines

## Resiliency

- Handle routine failures
- Automatically retry on failures

## Idempotency

- A data pipeline can be run multiple times, and the output remains the same

## Scalability

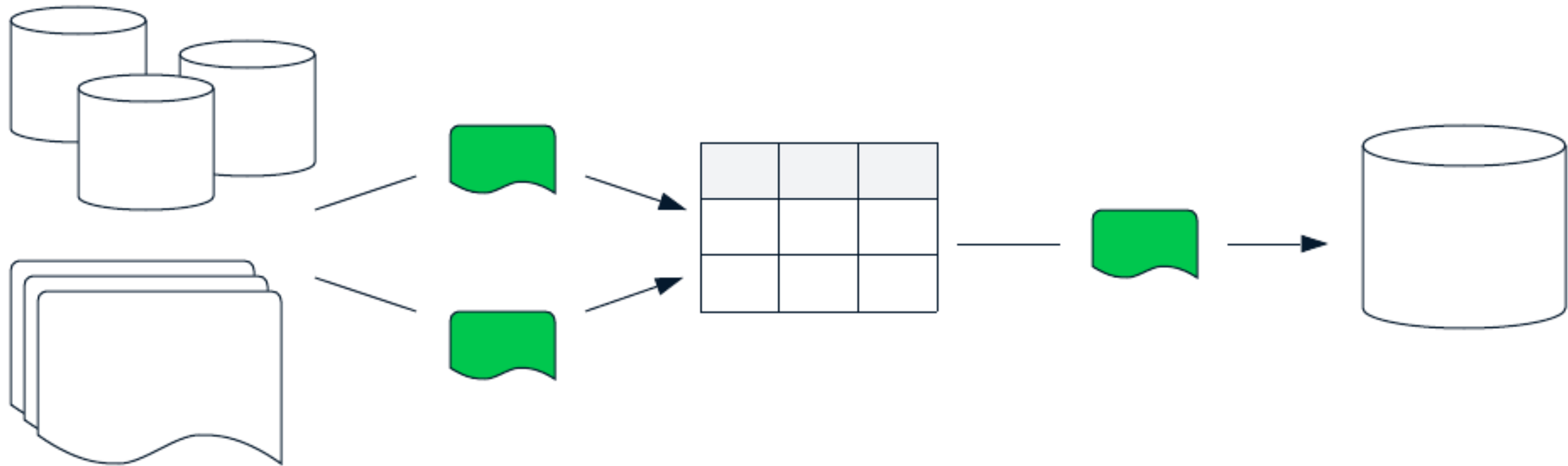
- Large amounts of data
- High frequency of invocations

## Transparency

- Avoid black-box transformations
- Well-tested and documented

# Data persistence

- Allows for pipeline to be rerun from last point of failure
- Easy to document the journey that data takes throughout the pipeline

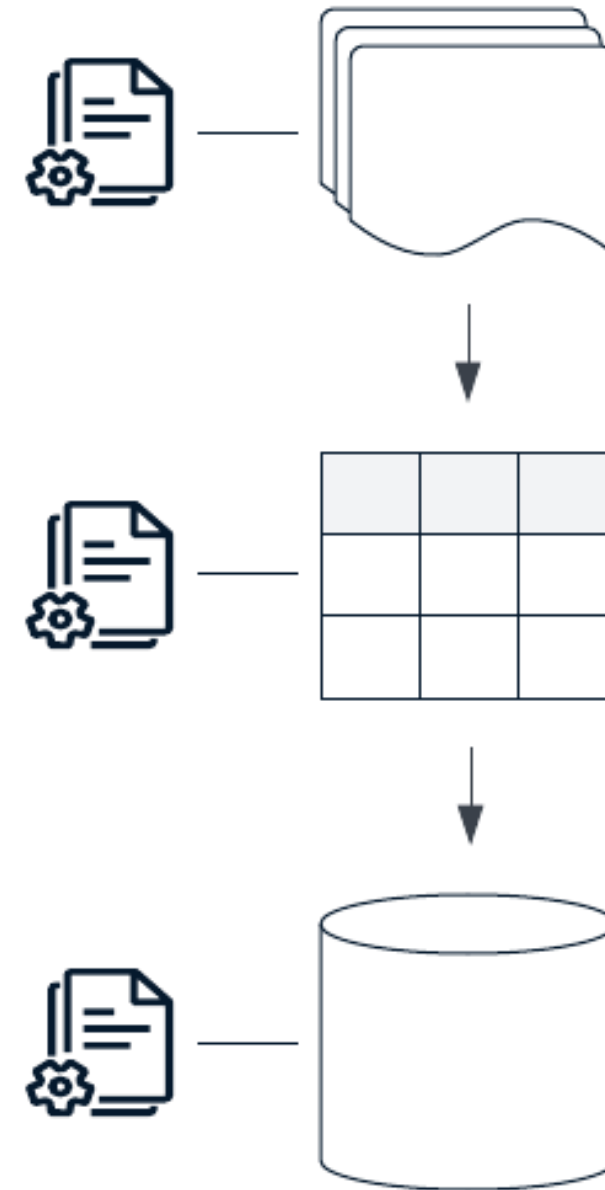


# Data lineage

## Definition:

Documenting the journey that data takes through a data pipeline

- Increases transparency of data solutions
- Instills trust in data consumers
- Provides a starting point for troubleshooting





# Ensuring a data pipeline works as expected

Before deployment:

- Unit tests
- End-to-end testing
  - Small and large data
  - Empty files
  - Bad data
- Code review

In production:

- Logging
- Alerting upon retries and failures.
- Communicate with data consumers
- Check final storage medium.

