# Copyright Notice

DeepLearning.AI

# The Data Engineering Lifecycle & Undercurrents

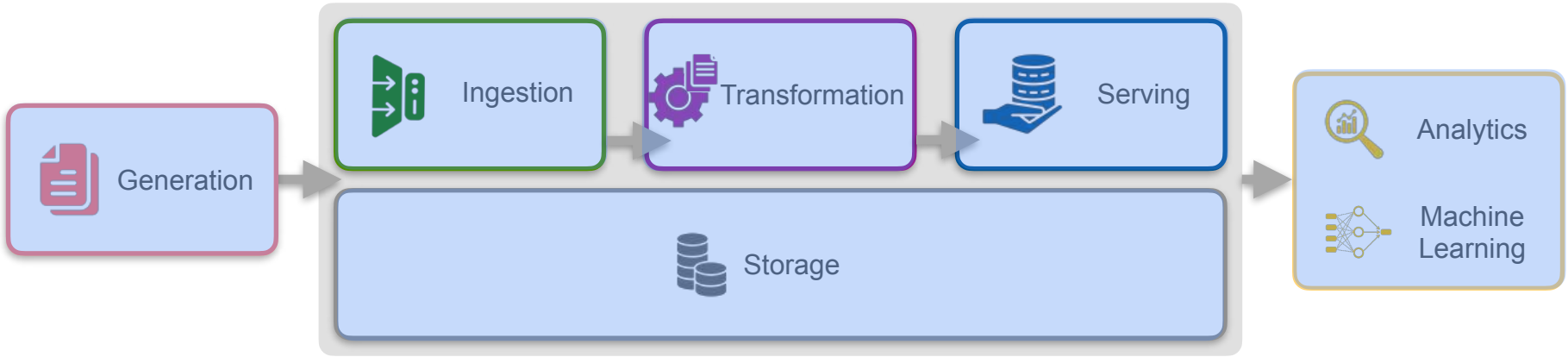
## Week 2

DeepLearning.AI

# The Data Engineering Lifecycle & Undercurrents

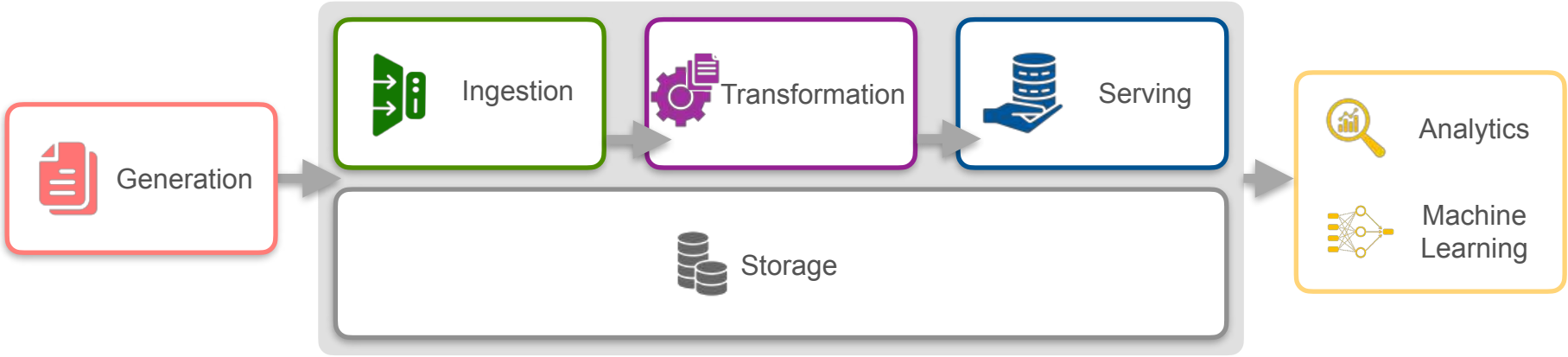
## Week 2 Overview

# The Data Engineering Lifecycle




Get raw data → Turn it into something useful → Make it available for downstream use cases

# The Undercurrents

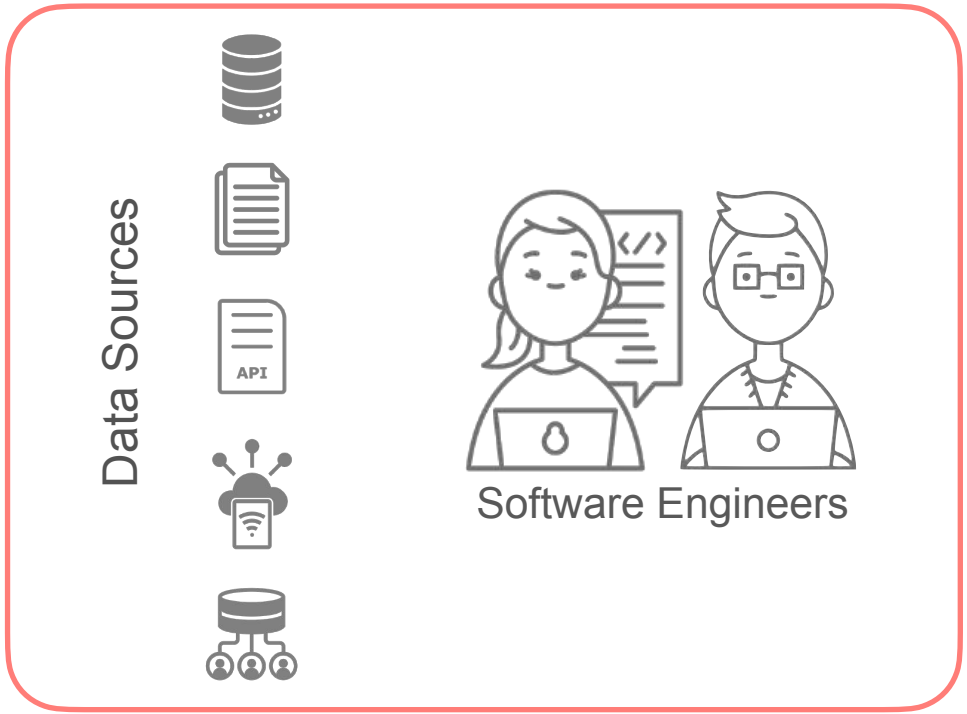


DeepLearning.AI

# The Data Engineering Lifecycle

## Data Generation in Source Systems

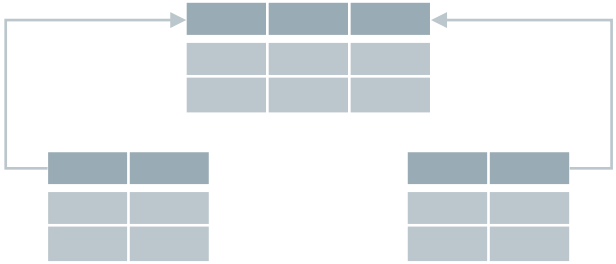
# Source Systems



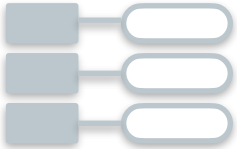
# Source Systems - Databases

**Databases**

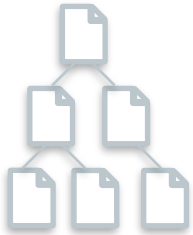
Relational Databases



NoSQL Databases



Key-Value



Document Stores

# Source Systems - Files

Files

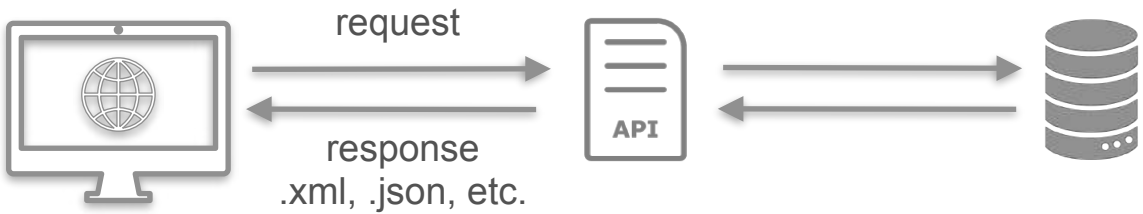
Text


Audio

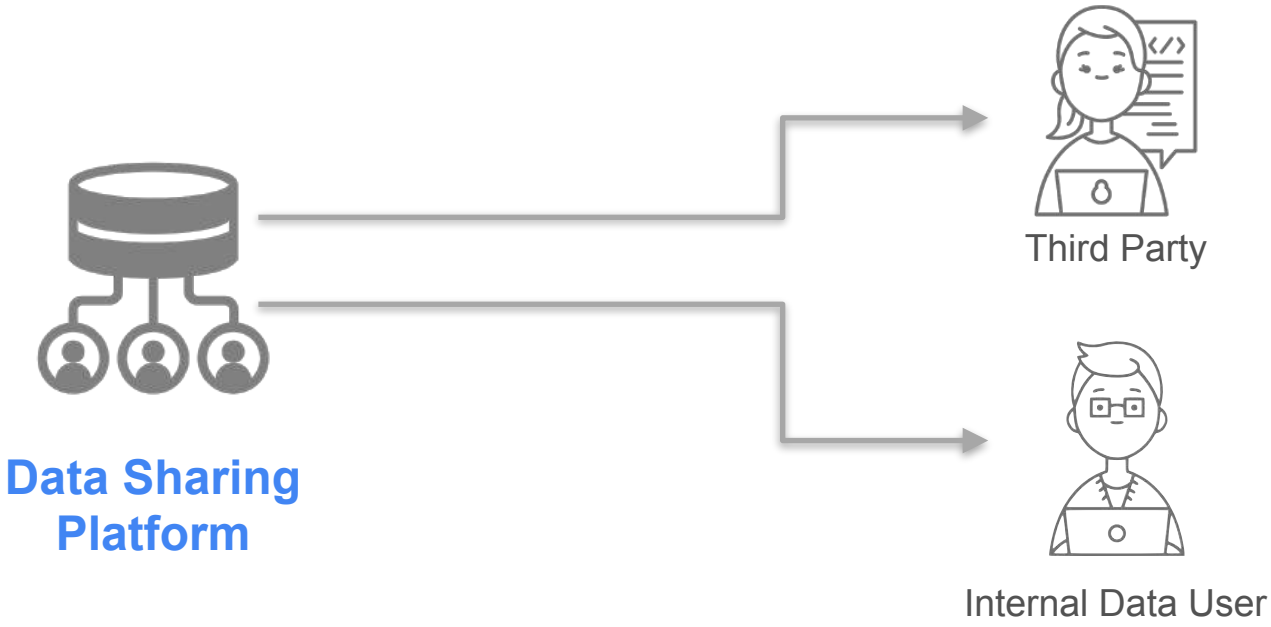

Video

# Source Systems - API


**Application Programming Interface (API)**



# Source Systems - Data Sharing



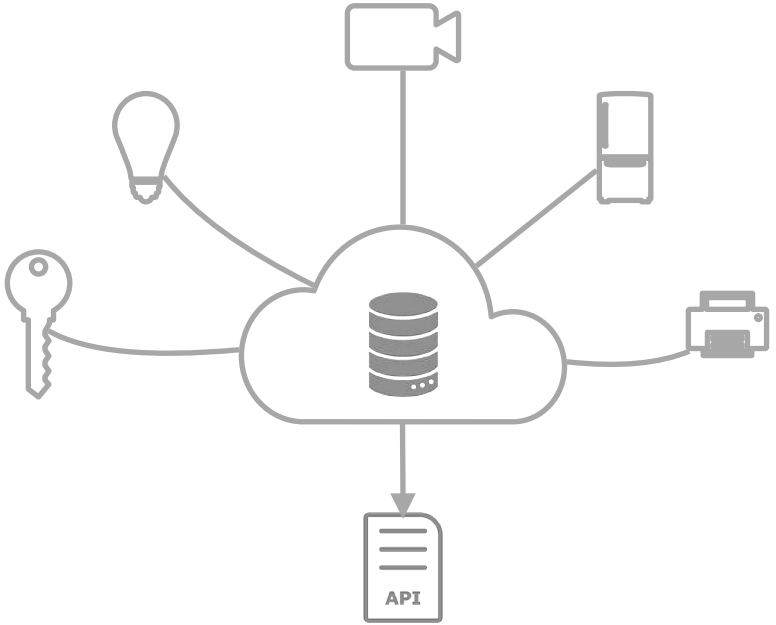
# Source Systems - IoT

**IoT devices**
**Internet of Things**

“Swarm” of IoT devices

# Source Systems - IoT

**IoT devices**
**Internet of Things**



# Source Systems


Deliver data

Downstream Systems

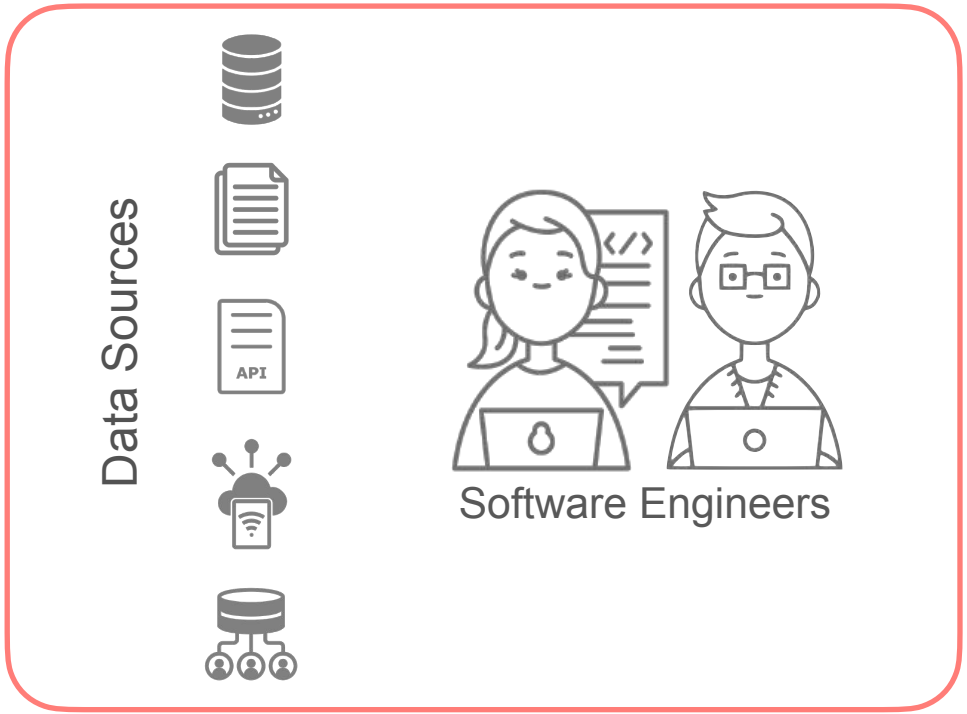
**Unpredictable systems**

- Systems go down
- Change in format/schema of data
- Change in data

# Source Systems


- How are the systems set up?
- What kind of changes are to expect?

# Source Systems




Data Engineer

**Understand how source systems work**

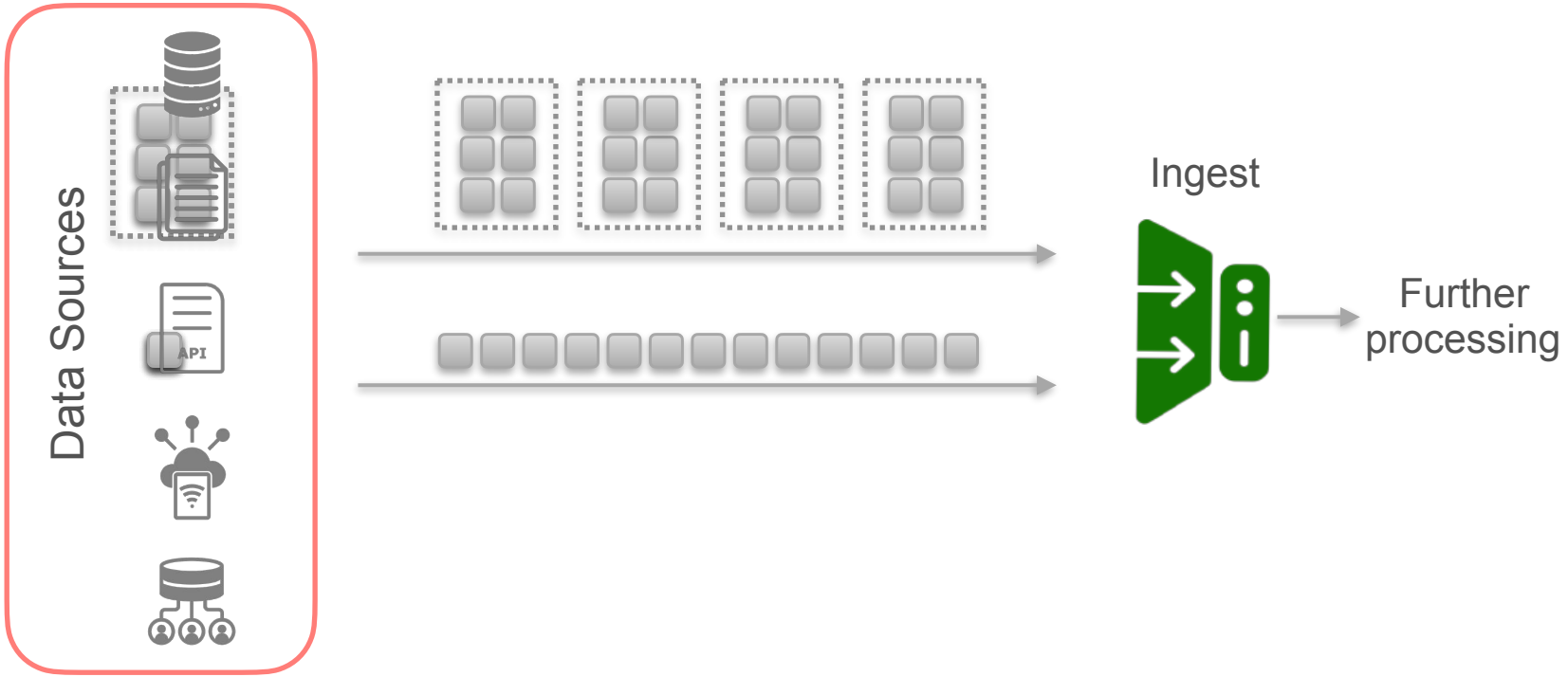
- How they generate data
- How the data may change over time
- How the changes will impact downstream systems

DeepLearning.AI

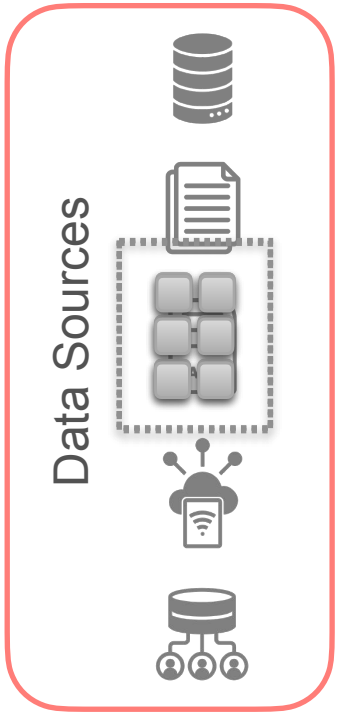
# The Data Engineering Lifecycle

## Ingestion

# Frequency of Ingestion

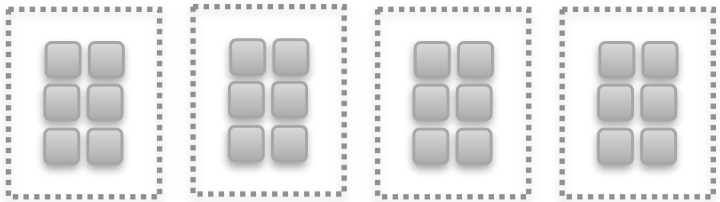


# Batch Ingestion




**Single batch**

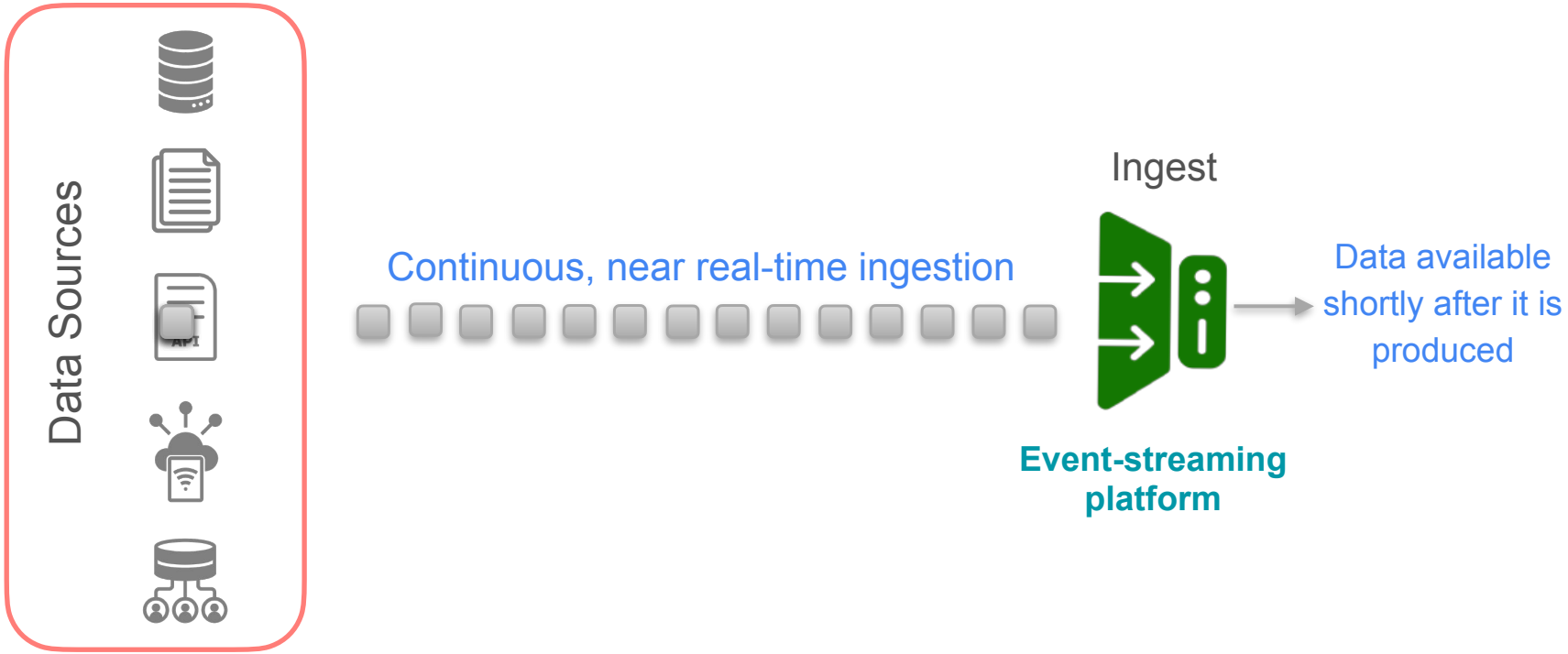
Entire day's worth of data



Ingest


- Based on predetermined time interval
- Based on preset size threshold

# Streaming Ingestion

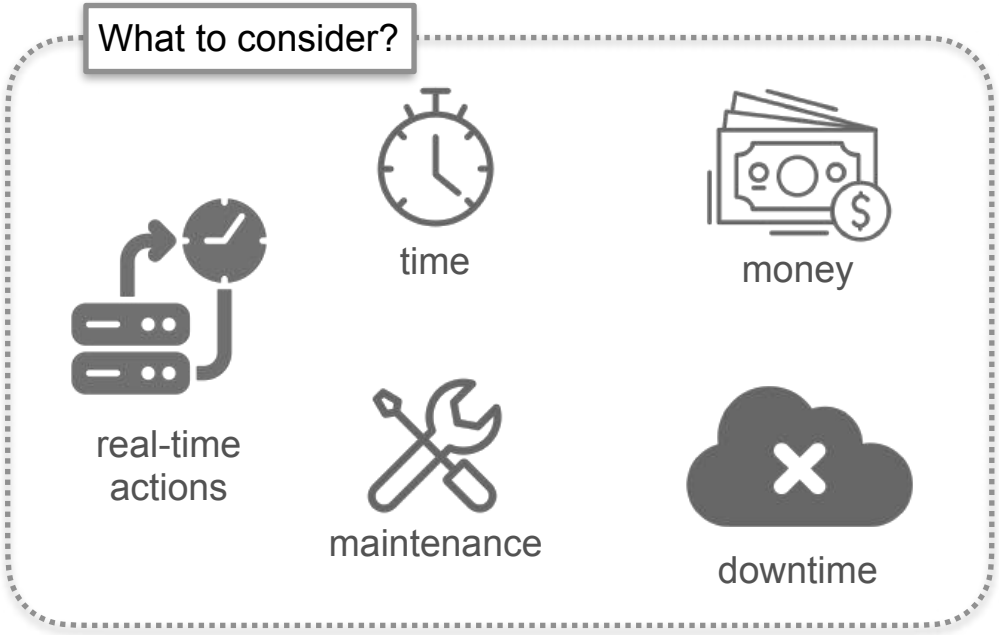


# Ingestion

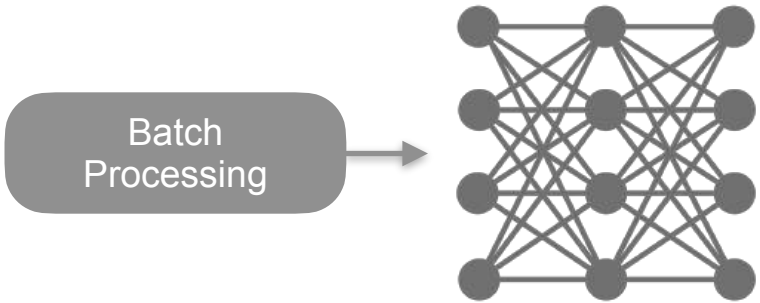
Batch Ingestion

vs.

Stream Ingestion

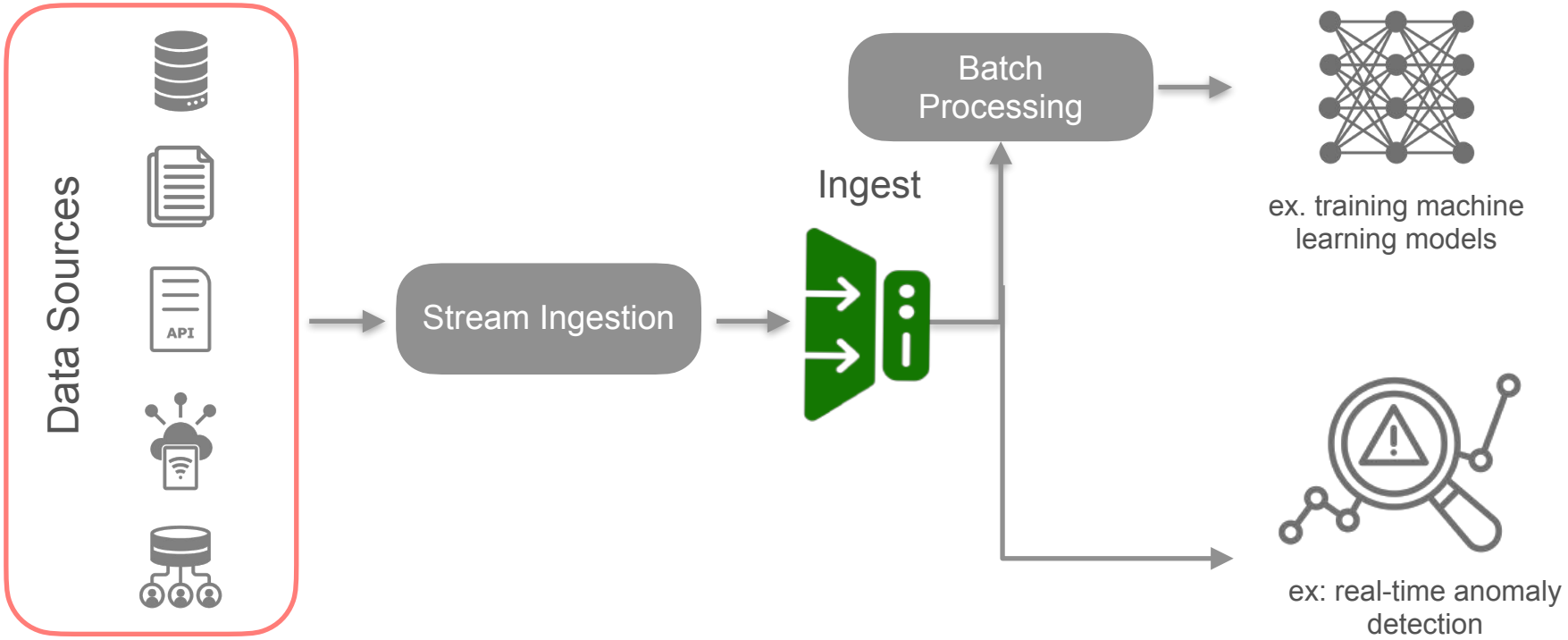


# Streaming and Batch Components




ex. training machine learning models

# Streaming and Batch Components



DeepLearning.AI

# The Data Engineering Lifecycle

## Storage

# Raw Hardware Ingredients

**Solid-state storage**

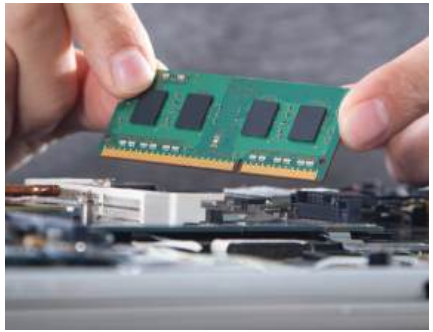

**Magnetic disk**


**Magnetic disk**

- Backbone of modern data storage system
- 2-3 times cheaper than solid-state storage

# Raw Hardware Ingredients

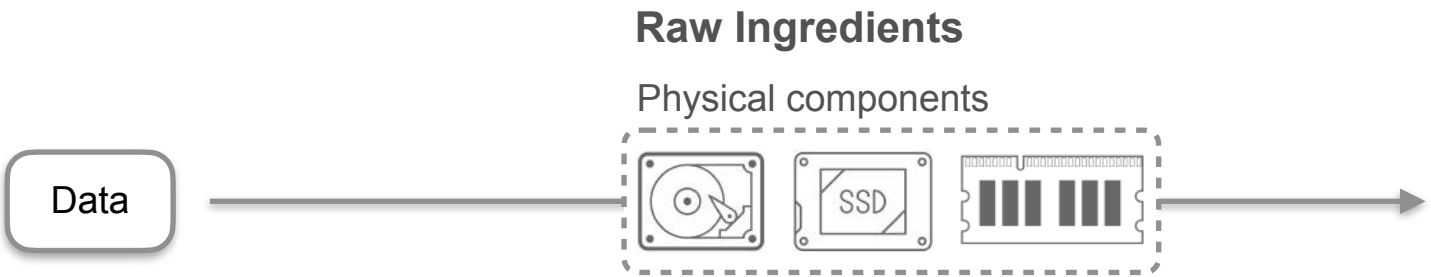
## RAM (Random Access Memory)



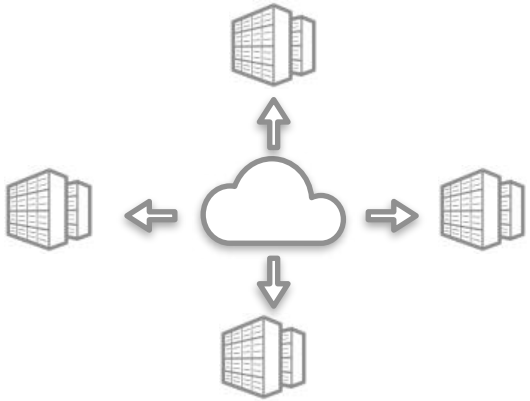
**RAM**

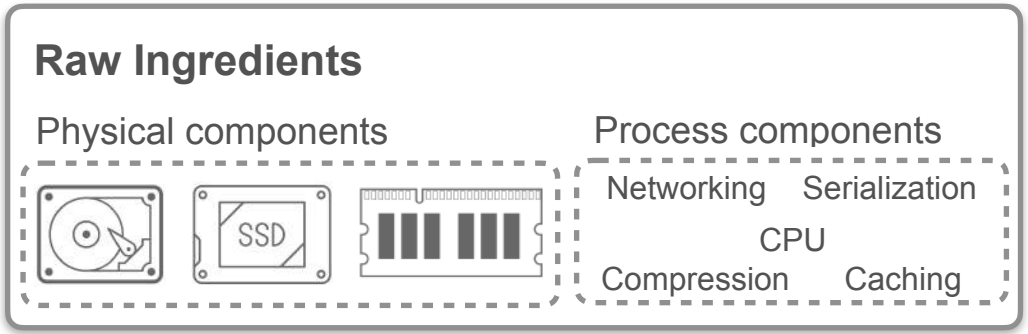
- Faster read and write speeds
- 30 - 50 times more expensive than solid-state storage
- Volatile

# Storage

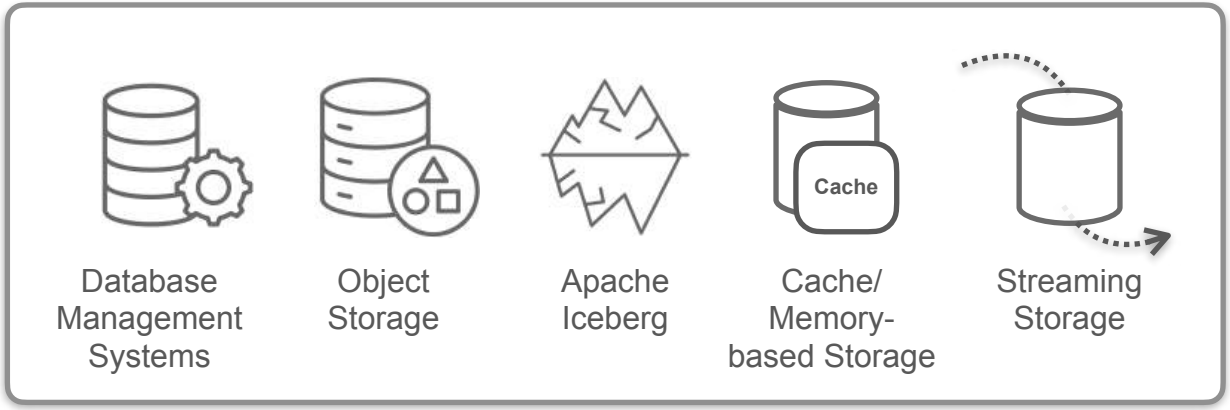


# Storage

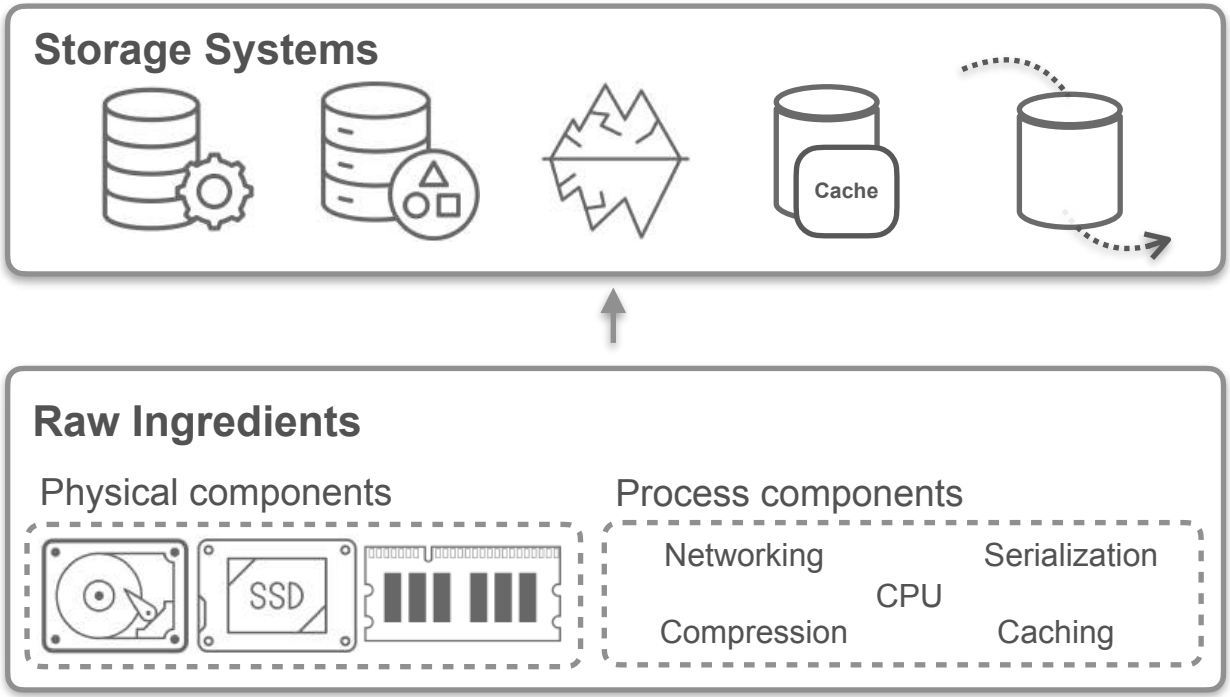



Raw Ingredients
Physical components
Process components
SSD
Networking
Serialization
CPU
Compression
Caching

# Storage Systems



# Storage Systems



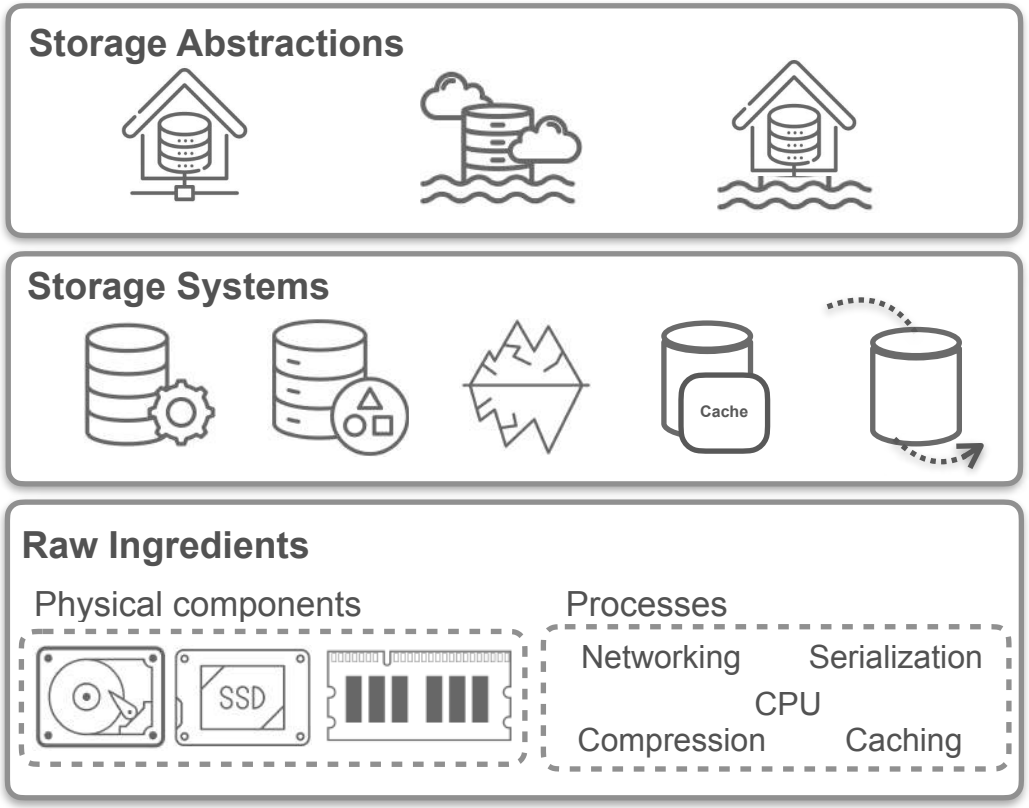
# Storage Abstractions

Storage abstractions: combinations of storage systems

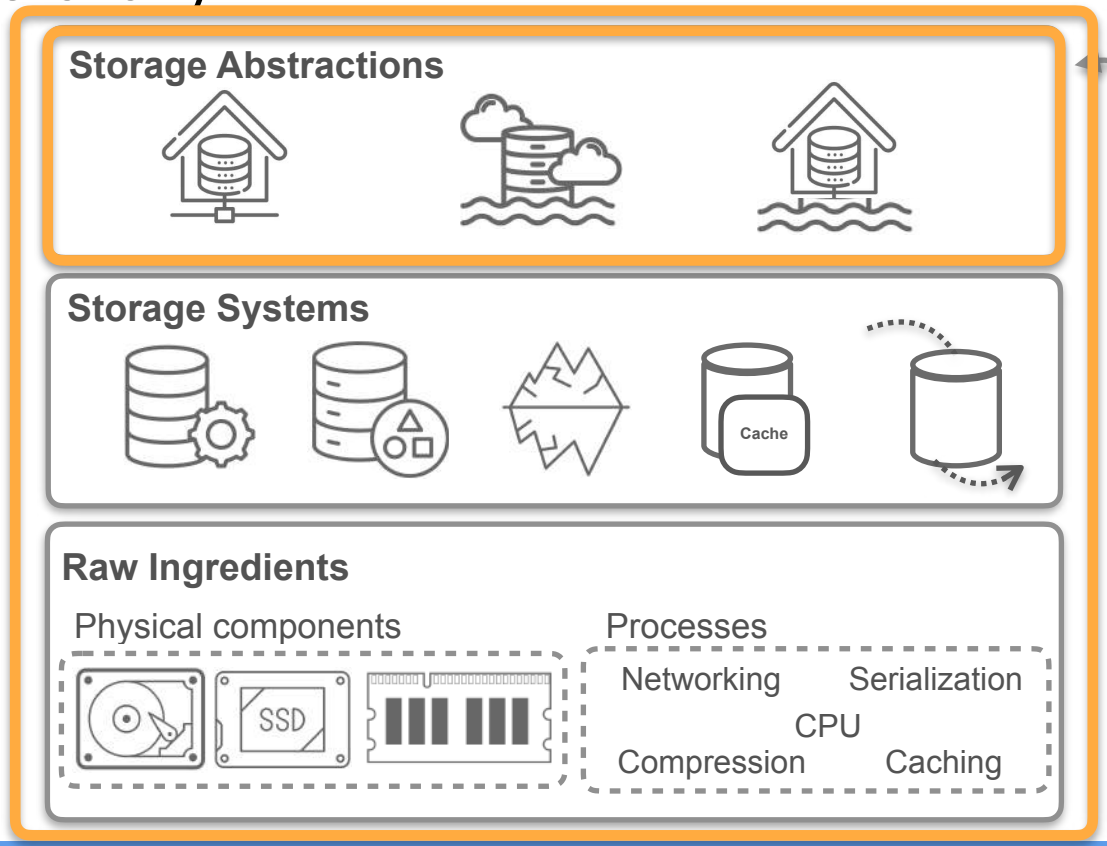

Choose configuration parameters:

- Latency
- Scalability
- Cost

# Storage Hierarchy



# Storage Hierarchy




Work near or at the top

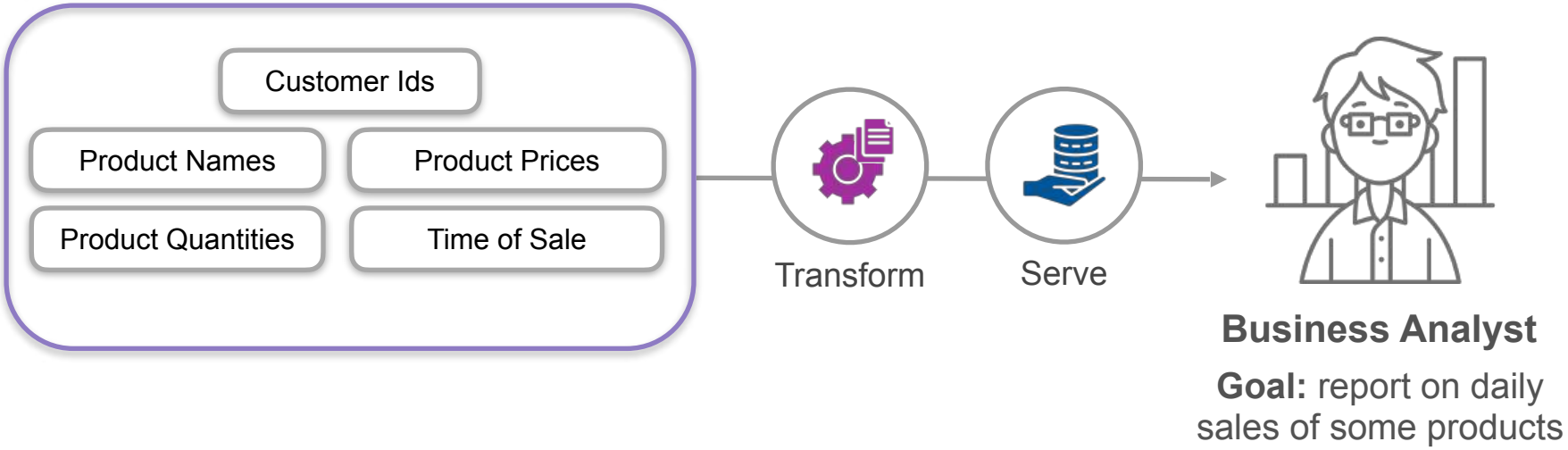
Understand the details of your entire storage solution

DeepLearning.AI

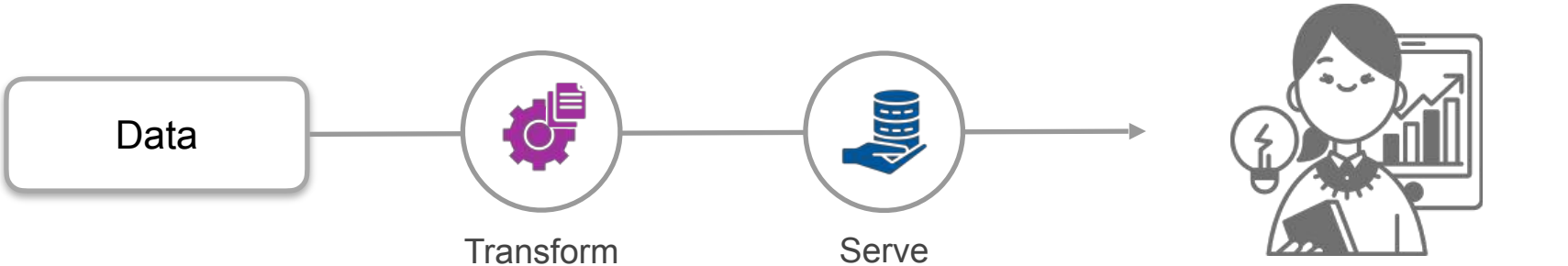
# The Data Engineering Lifecycle

## Queries, Modeling and Transformation

# Transformation



# Transformation



**Query**

Issuing a request to read records from a database or other storage system.

Data Warehouse

Query

- Tabular data
- Semi-structured data

**Query**

Issuing a request to read records from a database or other storage system.

## Query Language


## SQL Commands

### Data Cleaning

DROP
TRUNCATE
TRIM
REPLACE
SELECT DISTINCT

### Data Joining

INNER JOIN
LEFT JOIN
RIGHT JOIN
FULL JOIN
UNION

### Data Aggregating

SUM
AVG
COUNT
MAX
MIN
GROUP BY

### Data Filtering

WHERE
AND
OR
IS NULL
IS NOT NULL
IN
LIKE

**Query**

Issuing a request to read records from a database or other storage system.

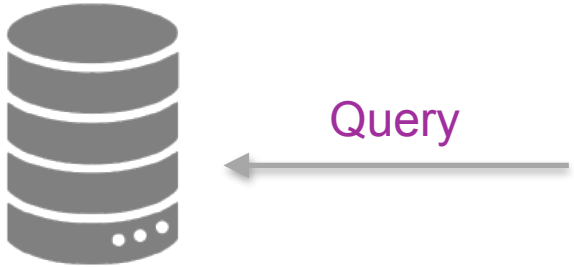
## Poor queries: negative impact on the source database


Source Database

**Query**

Issuing a request to read records from a database or other storage system.

## Poor queries: cause row explosion in your database

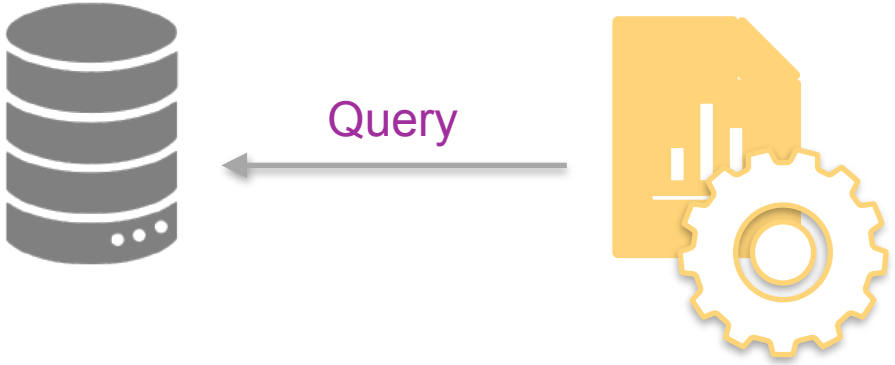



You database

**Query**

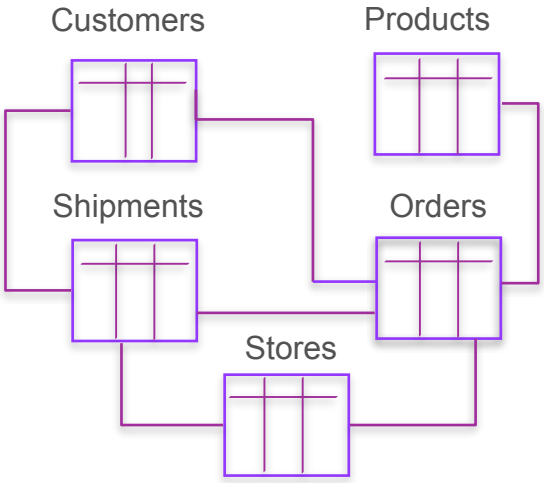
Issuing a request to read records from a database or other storage system.

**Poor queries: cause downstream delays**

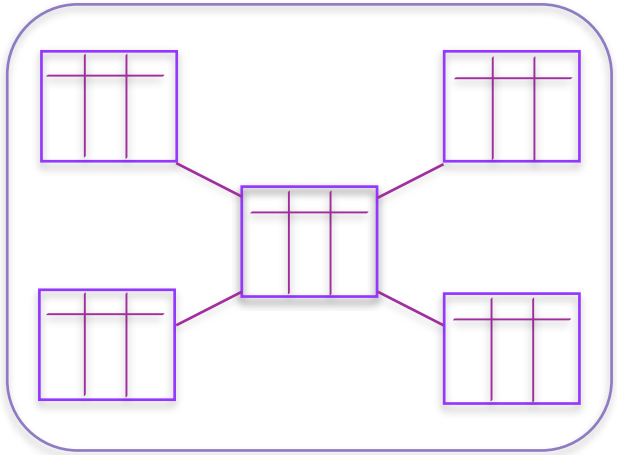


**Data modeling**

Choosing a coherent structure for your data to make it useful for the business.




Modeling



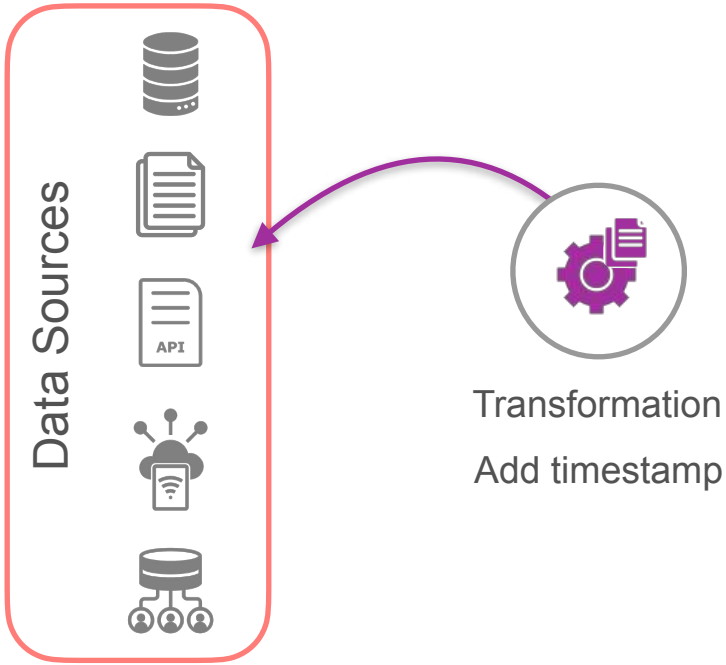
Query

**Business Analyst**

**Goal:** Product sales reports

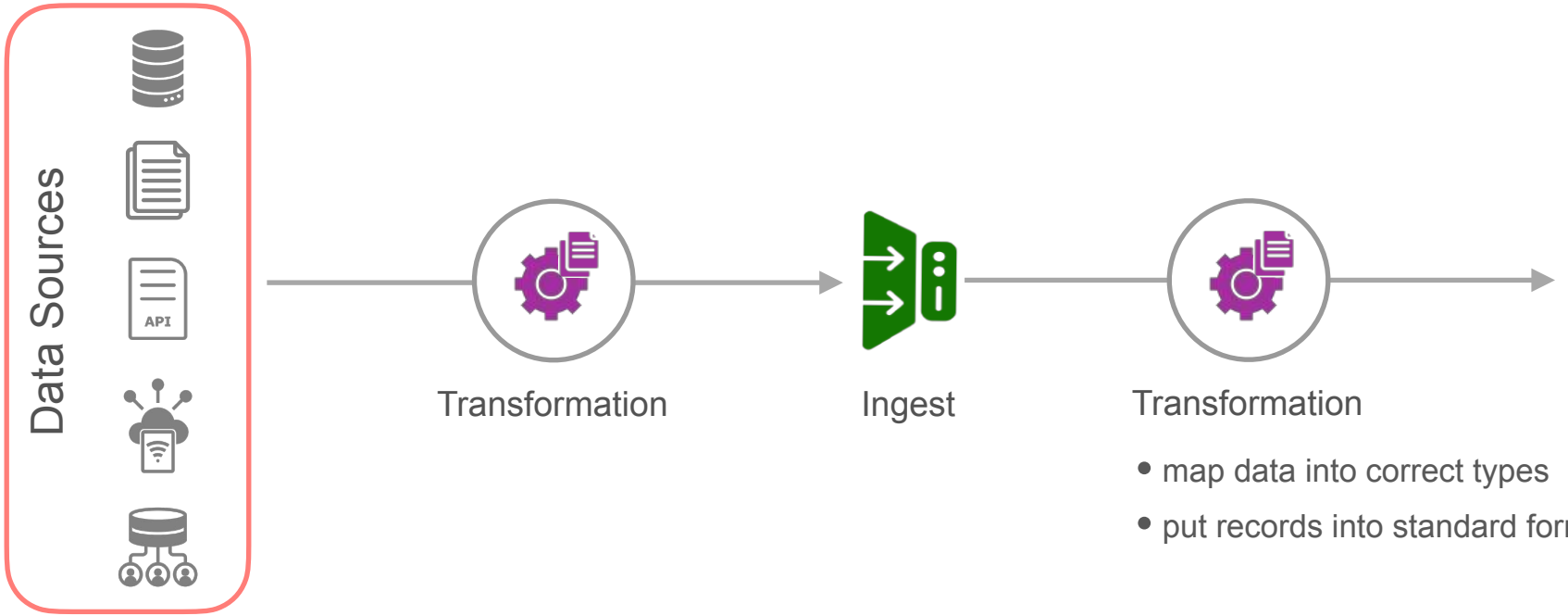
**Data transformation**

Data manipulated, enhanced and saved for downstream use.



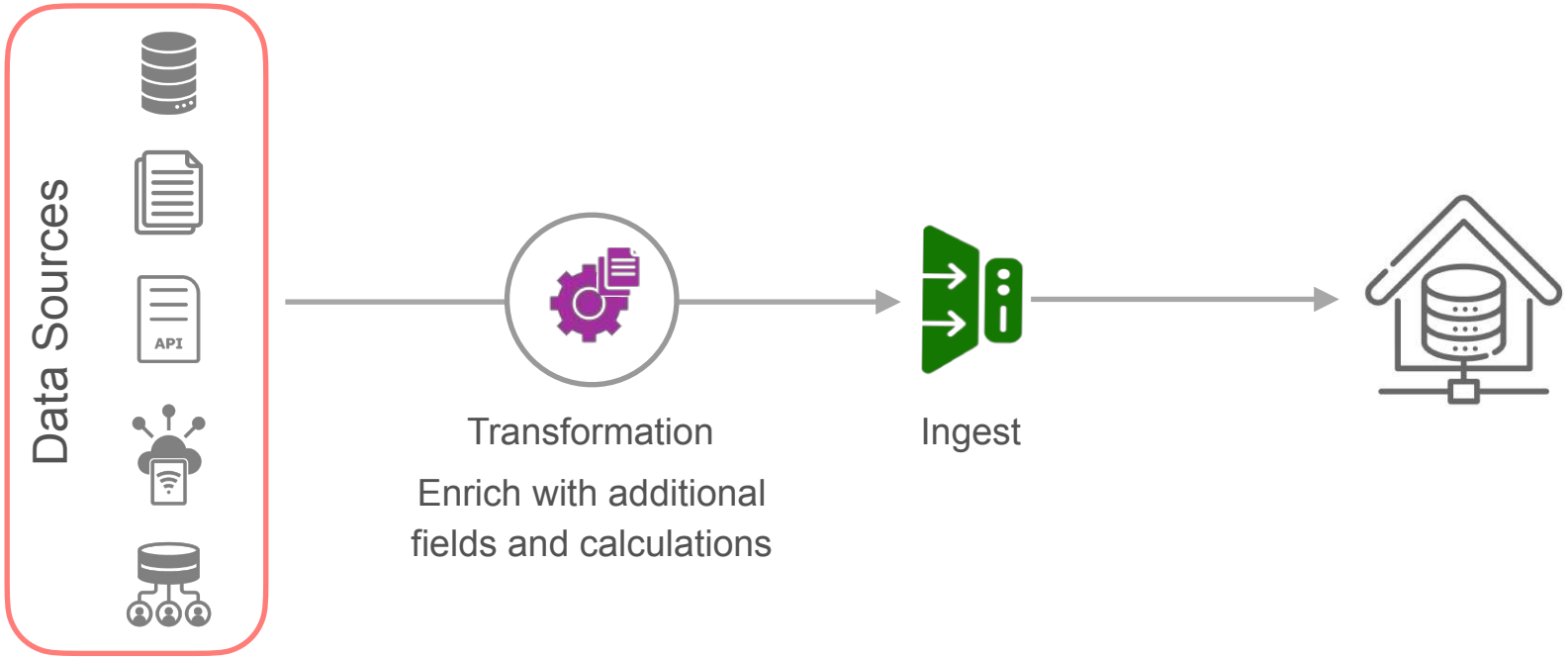
**Data transformation**

Data manipulated, enhanced and saved for downstream use.



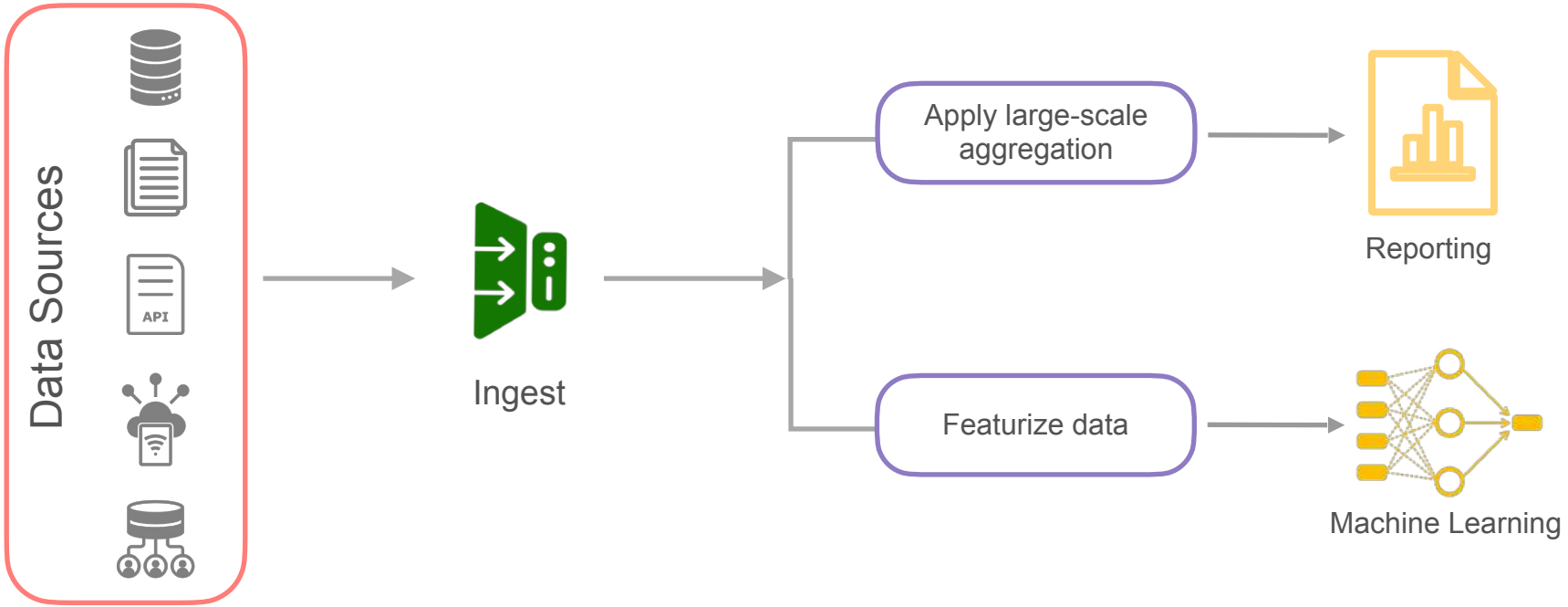
**Data transformation**

Data manipulated, enhanced and saved for downstream use.



**Data transformation**

Data manipulated, enhanced and saved for downstream use.

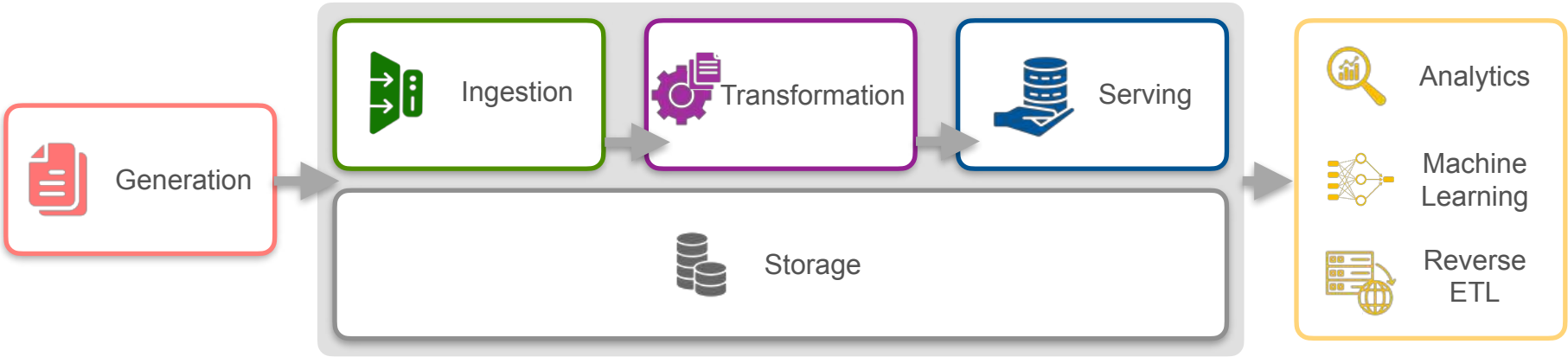


DeepLearning.AI

# The Data Engineering Lifecycle

## Serving Data

# The Data Engineering Lifecycle



# Analytics

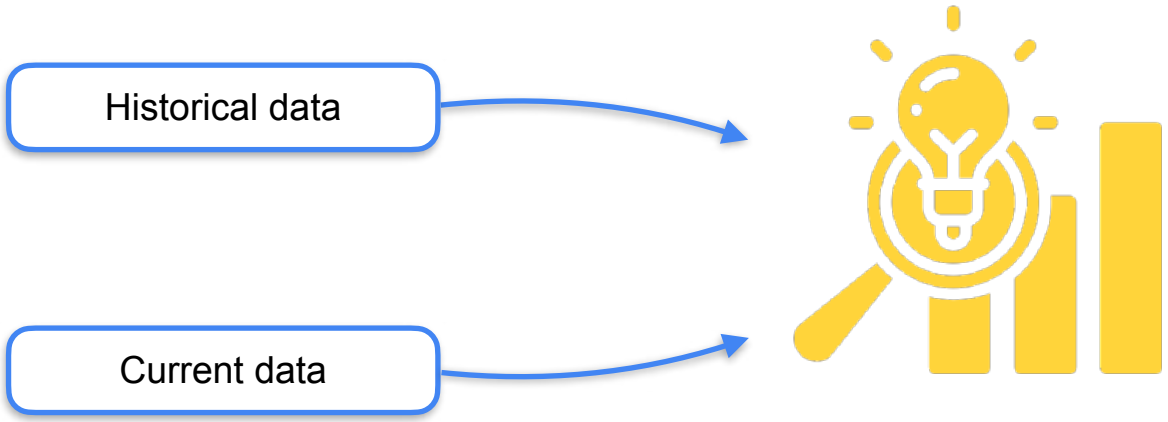
Analytics is the process of identifying key insights and patterns within data.

- Business Intelligence
- Operational Analytics
- Embedded Analytics

# Analytics

**Business Intelligence**

Explore historical and current business data to discover insights



# Analytics

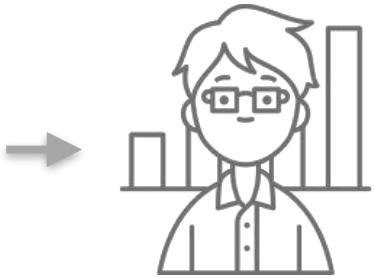
Business Intelligence

Explore historical and current business data to discover insights

# Analytics

Business Intelligence

Explore historical and current business data to discover insights

Spot patterns and trends

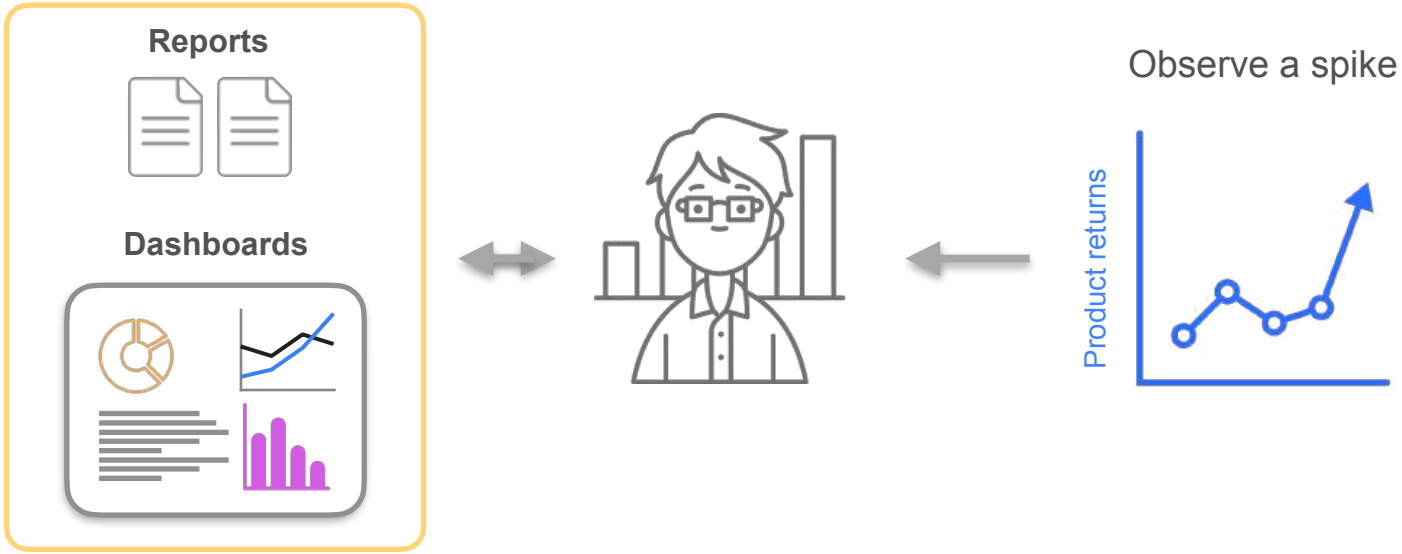
Monitor:

- Campaign engagement
- Regional sales
- Customer experience metrics

# Analytics

Business Intelligence

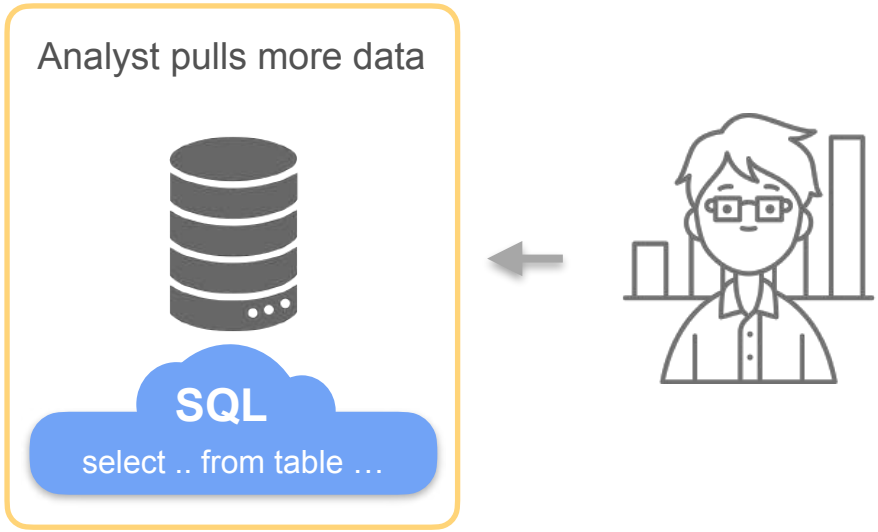
Explore historical and current business data to discover insights



# Analytics

Business Intelligence

Explore historical and current business data to discover insights



# Analytics

Operational Analytics

Monitoring real-time data for immediate action

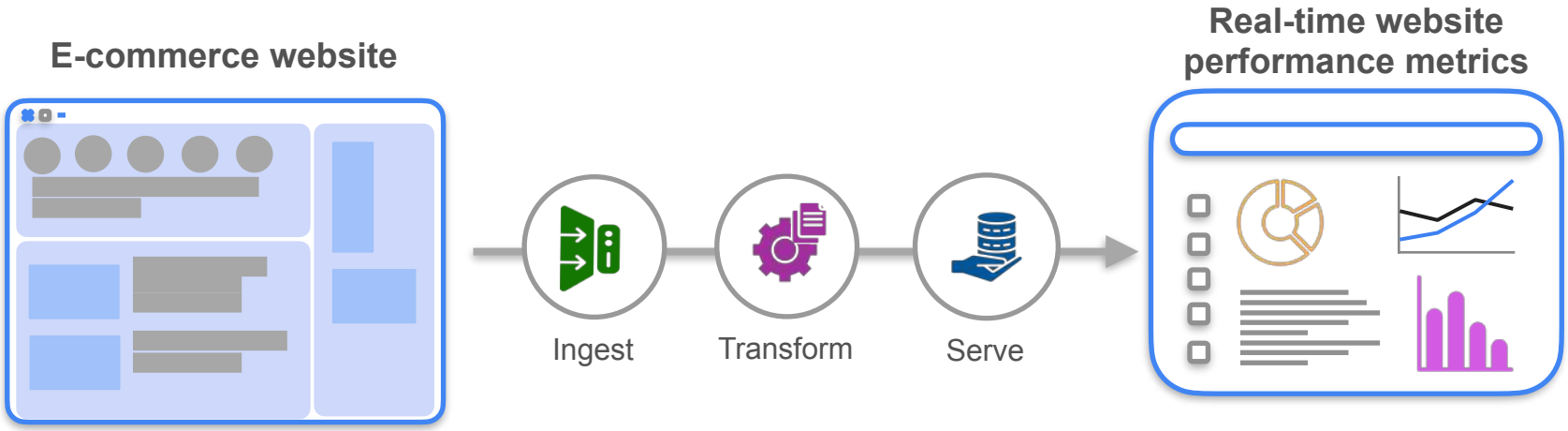
**E-commerce website**

**Real-time website performance metrics**

# Analytics

Operational Analytics

Monitoring real-time data for immediate action

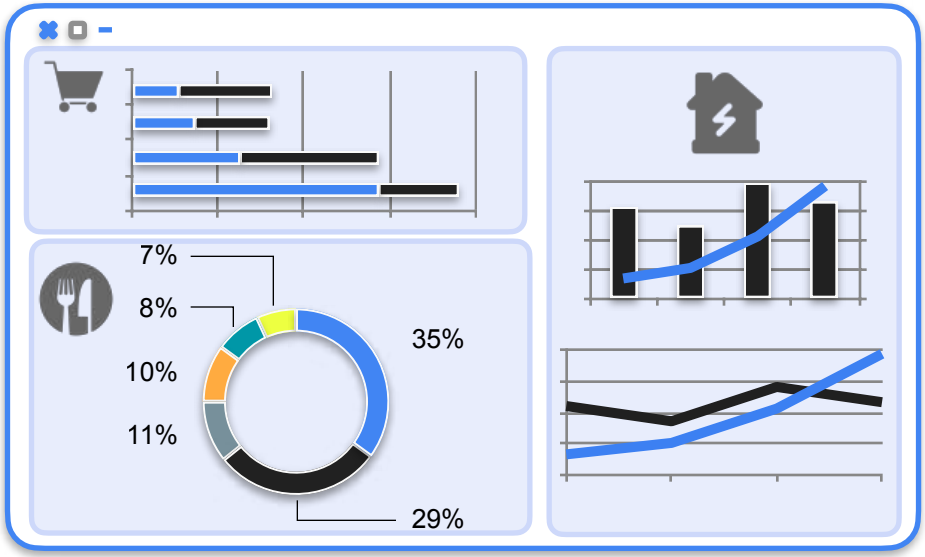


# Analytics

Embedded Analytics

External or customer-facing analytics

Customer-facing dashboards



# Analytics

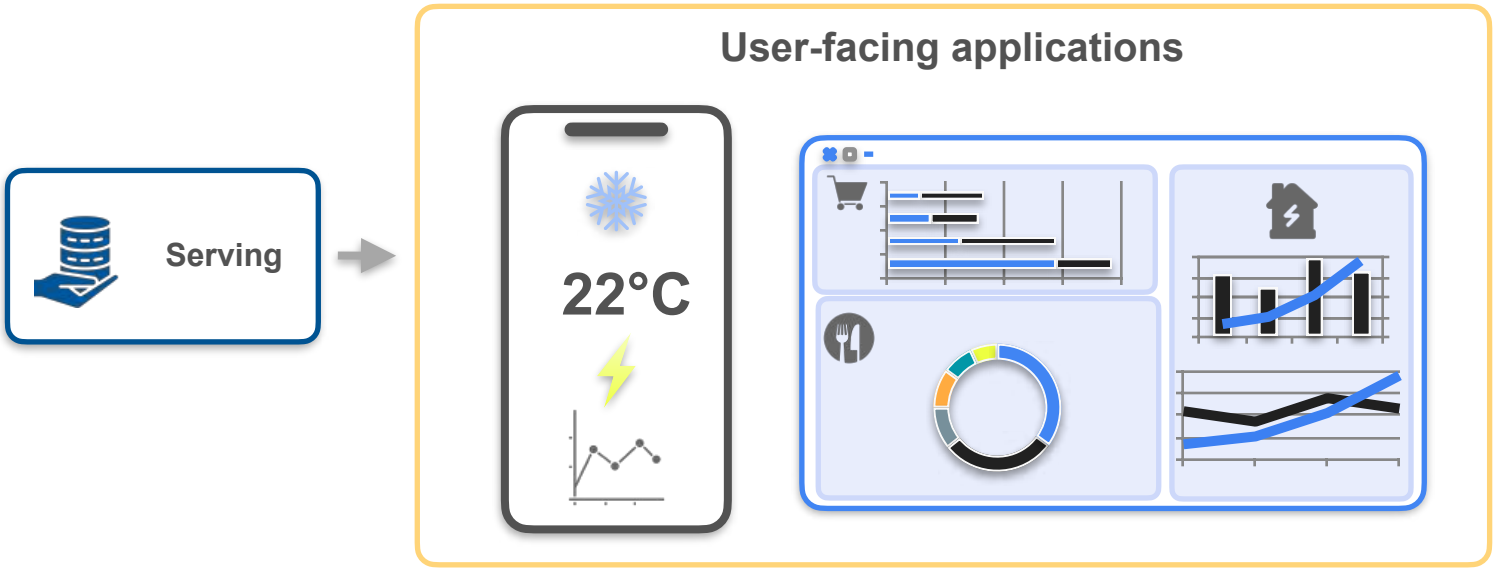
Embedded Analytics

External or customer-facing analytics

# Analytics

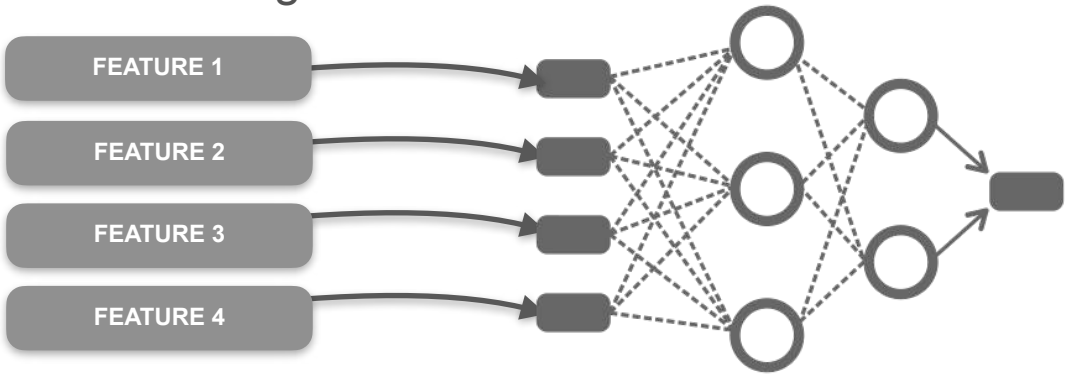
Embedded Analytics

External or customer-facing analytics



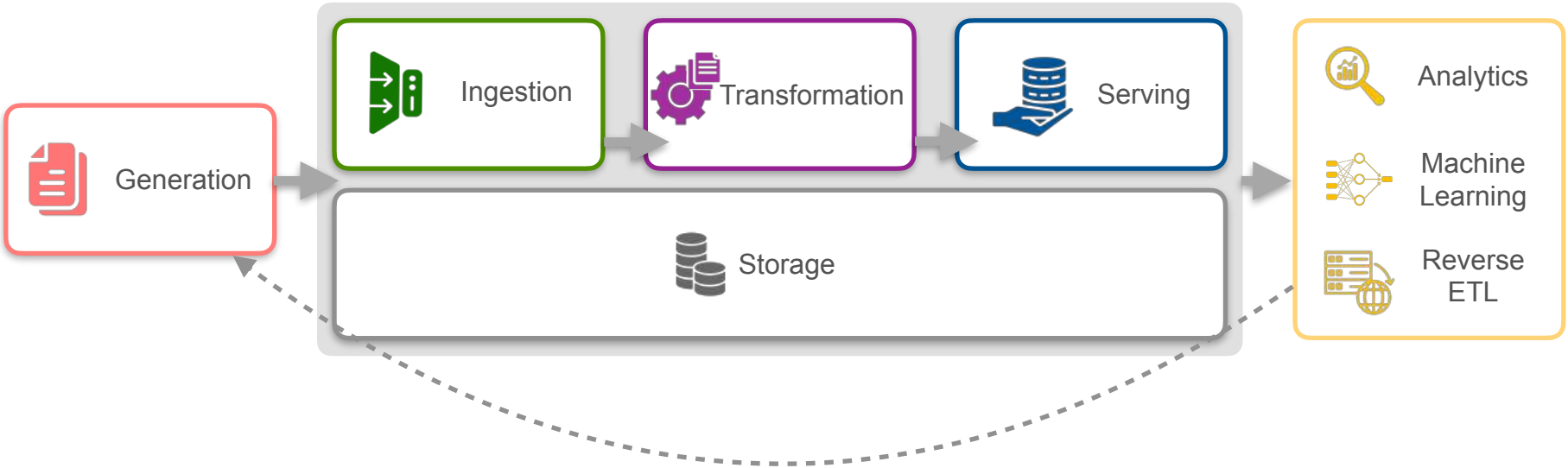
# Machine Learning


- Model training

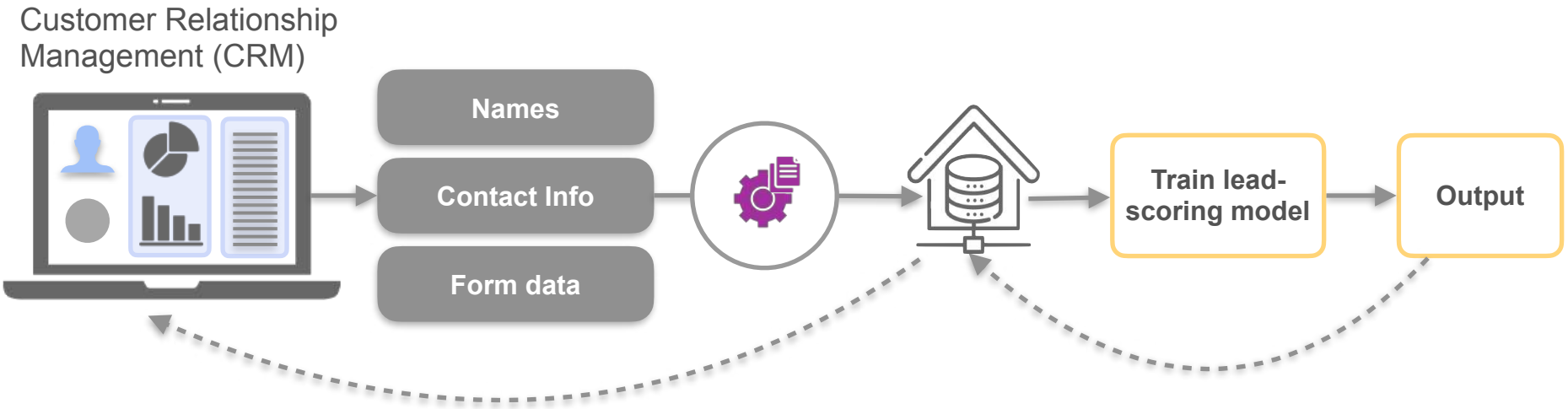



- Real-time inference
- Track data history and lineage

# The Data Engineering Lifecycle



# Reverse ETL

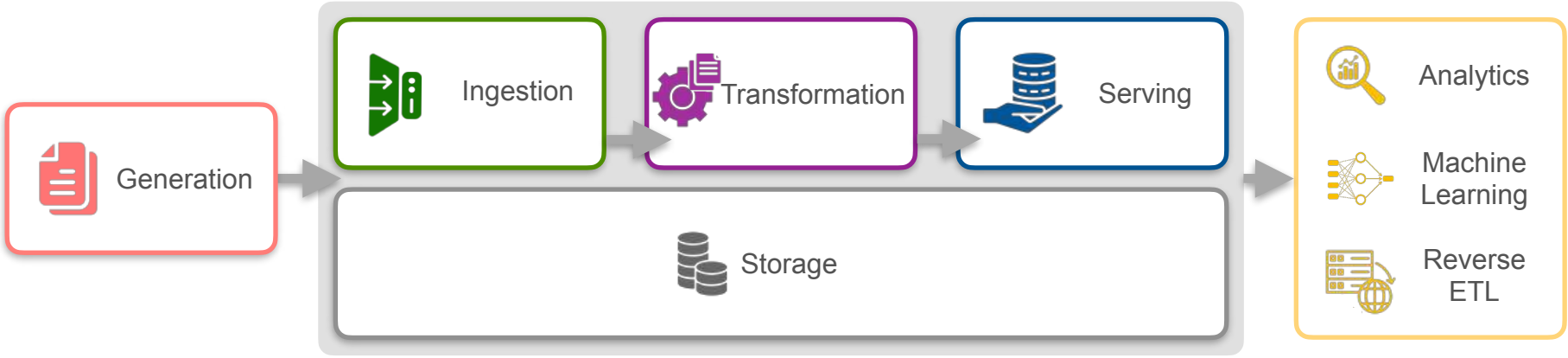

Customer Relationship Management (CRM)
Names
Contact Info
Form data
Train lead-scoring model
Output

DeepLearning.AI

# The Undercurrents of the Data Engineering Lifecycle

## Intro to the Undercurrents

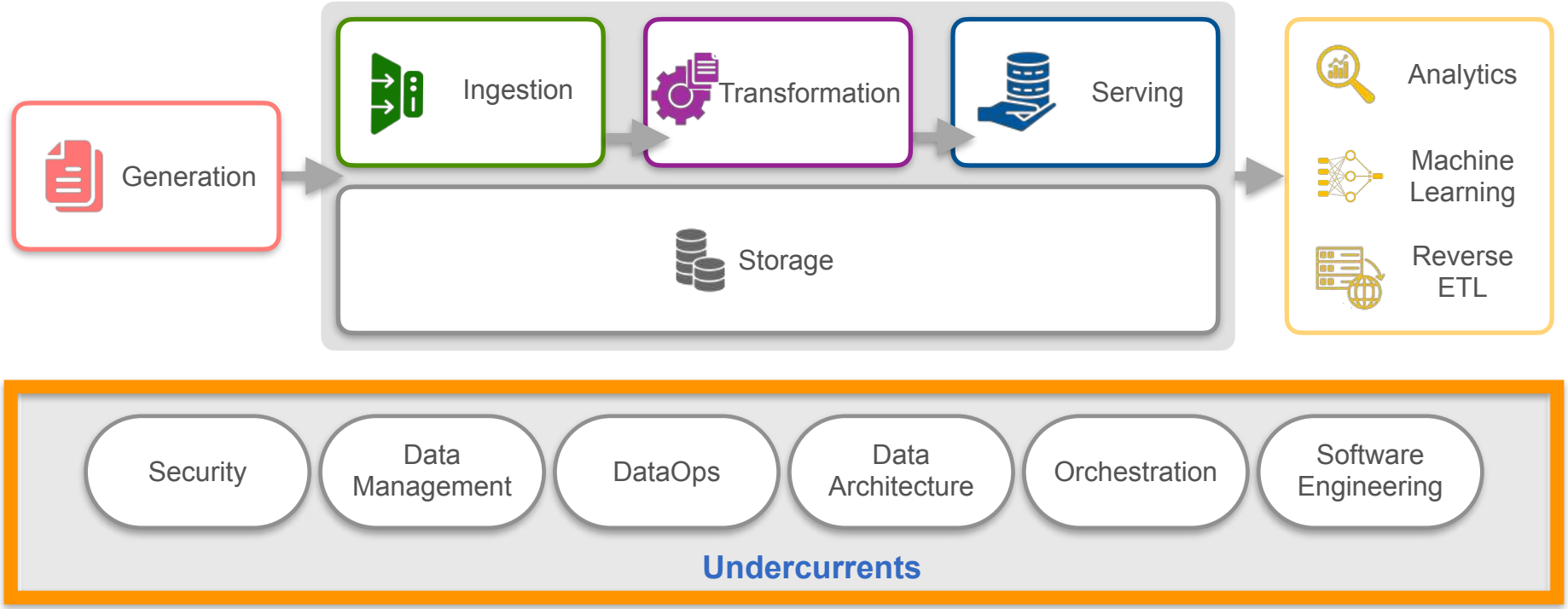
# The Data Engineering Lifecycle & Undercurrents



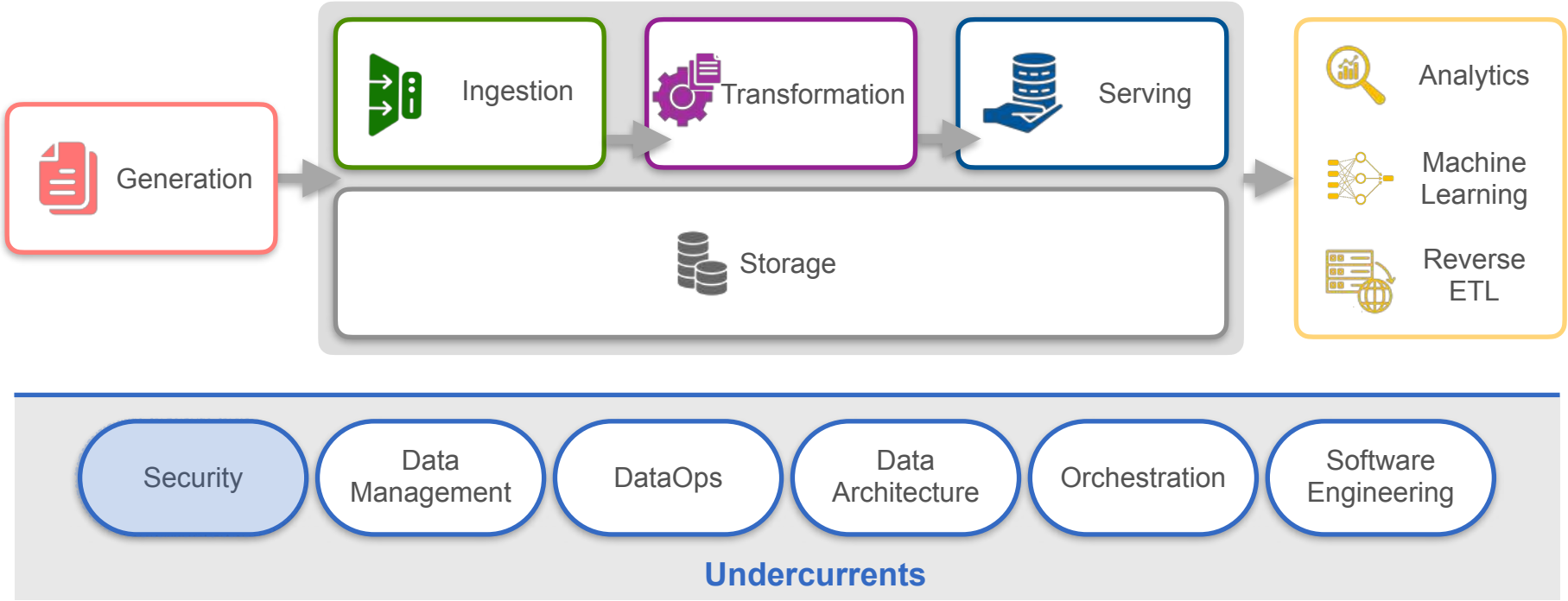

Security
Data Management
DataOps
Data Architecture
Orchestration
Software Engineering

**Undercurrents**

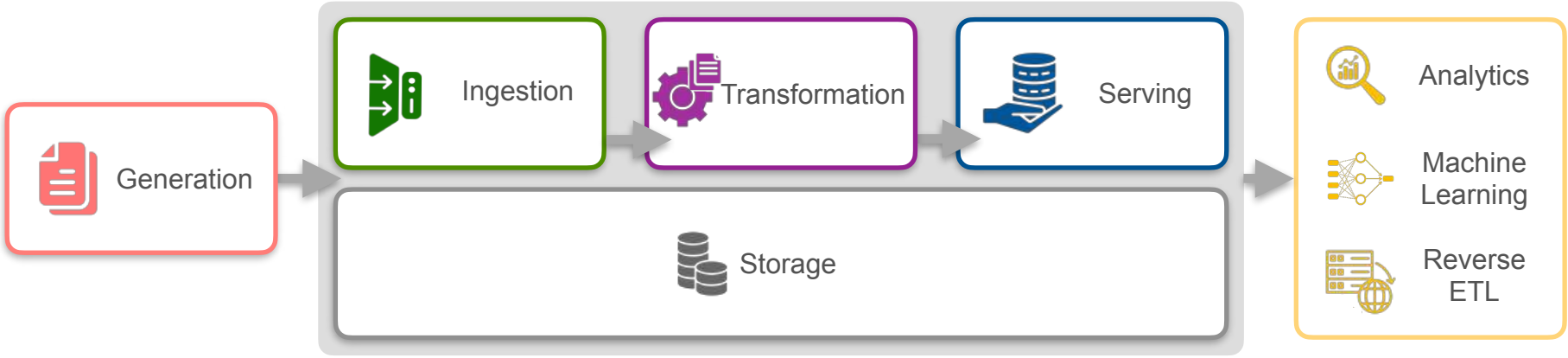
# The Data Engineering Lifecycle & Undercurrents



# The Data Engineering Lifecycle & Undercurrents



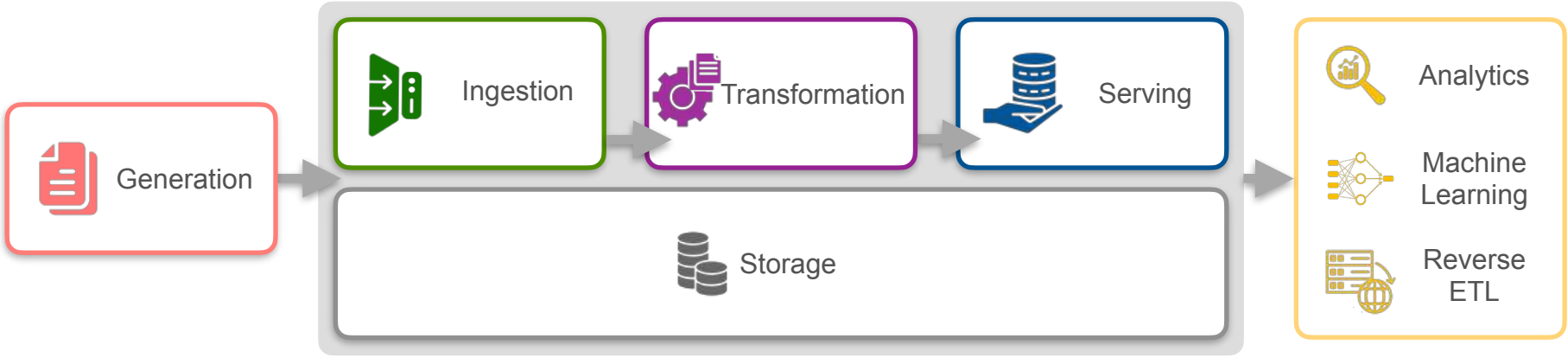
# The Data Engineering Lifecycle & Undercurrents




Security
Data Management
DataOps
Data Architecture
Orchestration
Software Engineering

**Undercurrents**

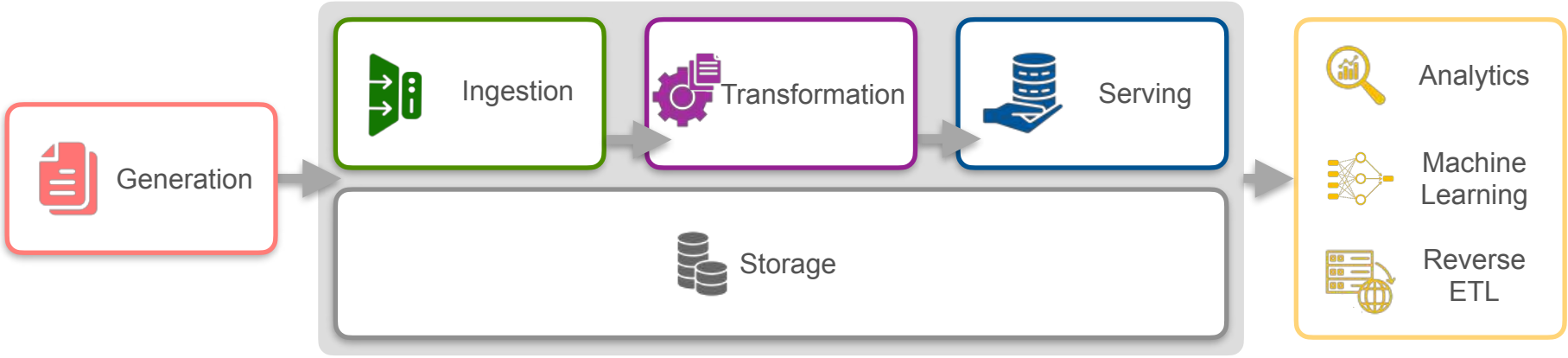
# The Data Engineering Lifecycle & Undercurrents




Security
Data Management
DataOps
Data Architecture
Orchestration
Software Engineering

**Undercurrents**

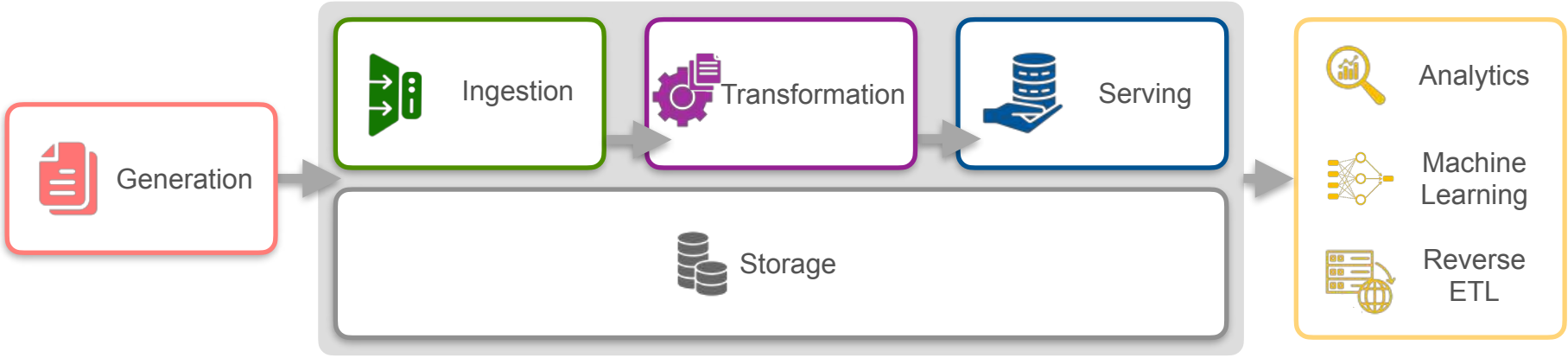
# The Data Engineering Lifecycle & Undercurrents




Security
Data Management
DataOps
Data Architecture
Orchestration
Software Engineering

**Undercurrents**

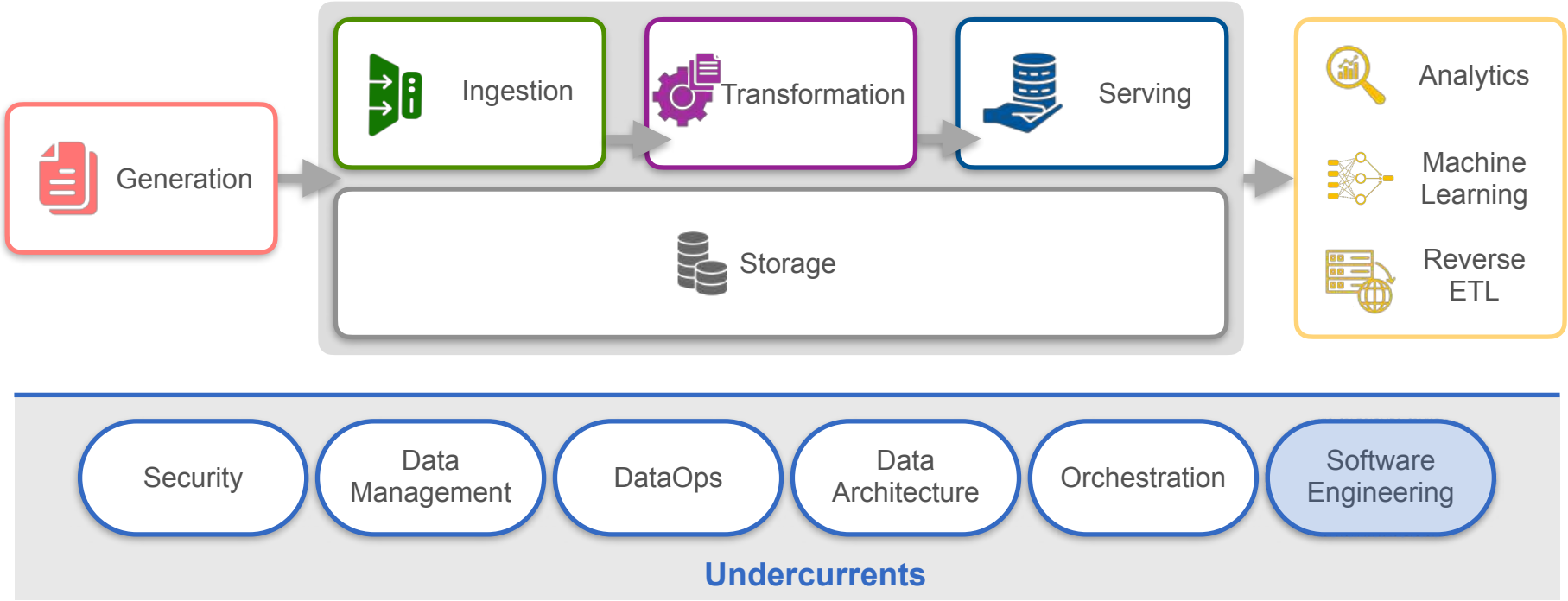
# The Data Engineering Lifecycle & Undercurrents




Security
Data Management
DataOps
Data Architecture
Orchestration
Software Engineering

**Undercurrents**

# The Data Engineering Lifecycle & Undercurrents

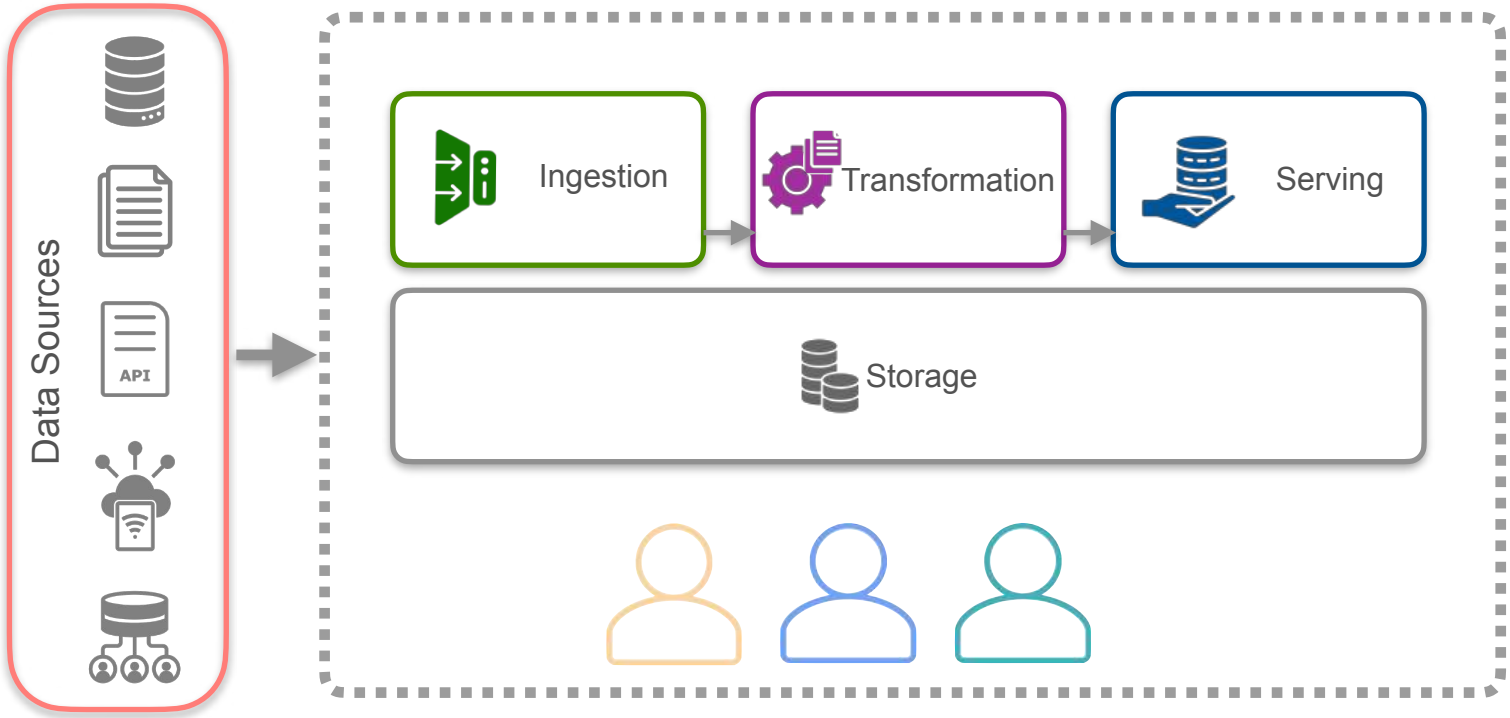


DeepLearning.AI

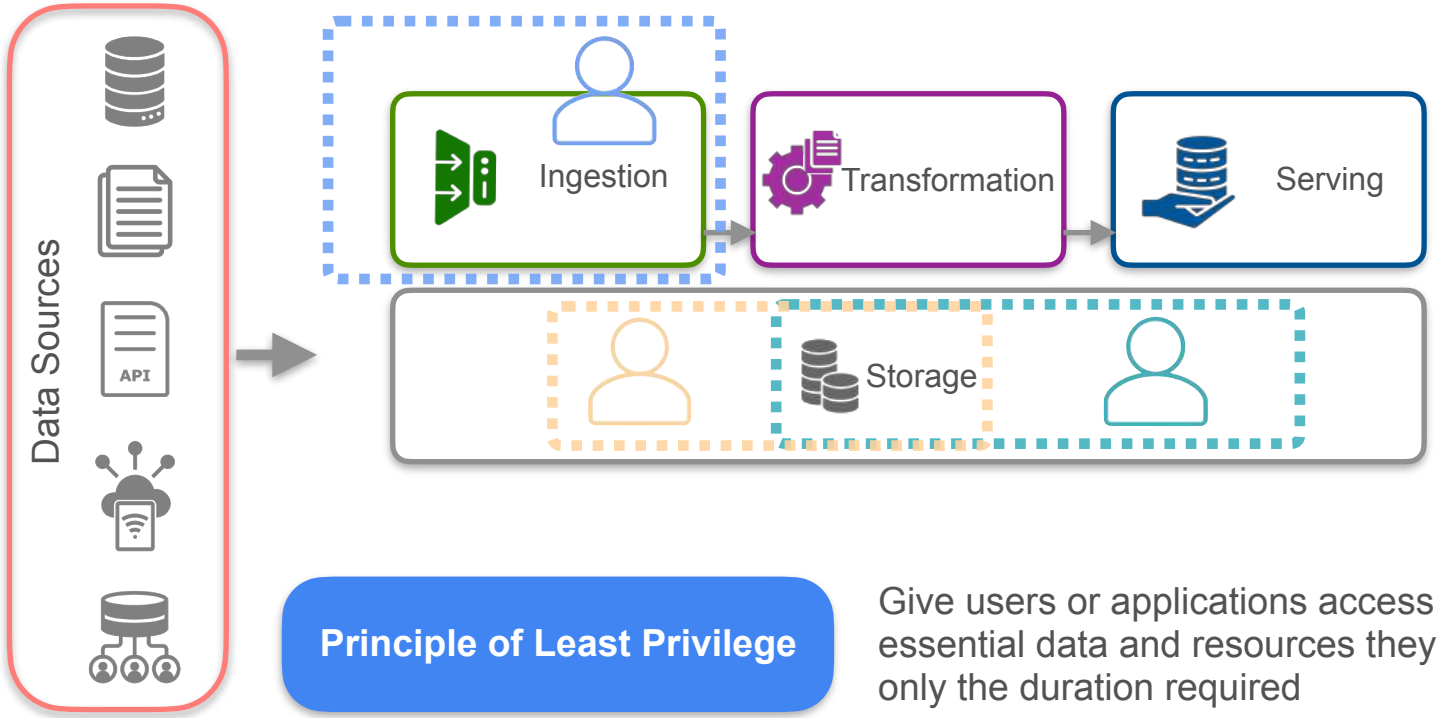
# The Undercurrents of the Data Engineering Lifecycle

## Security

# Security


Data Sources
API
Ingestion
Transformation
Serving
Storage

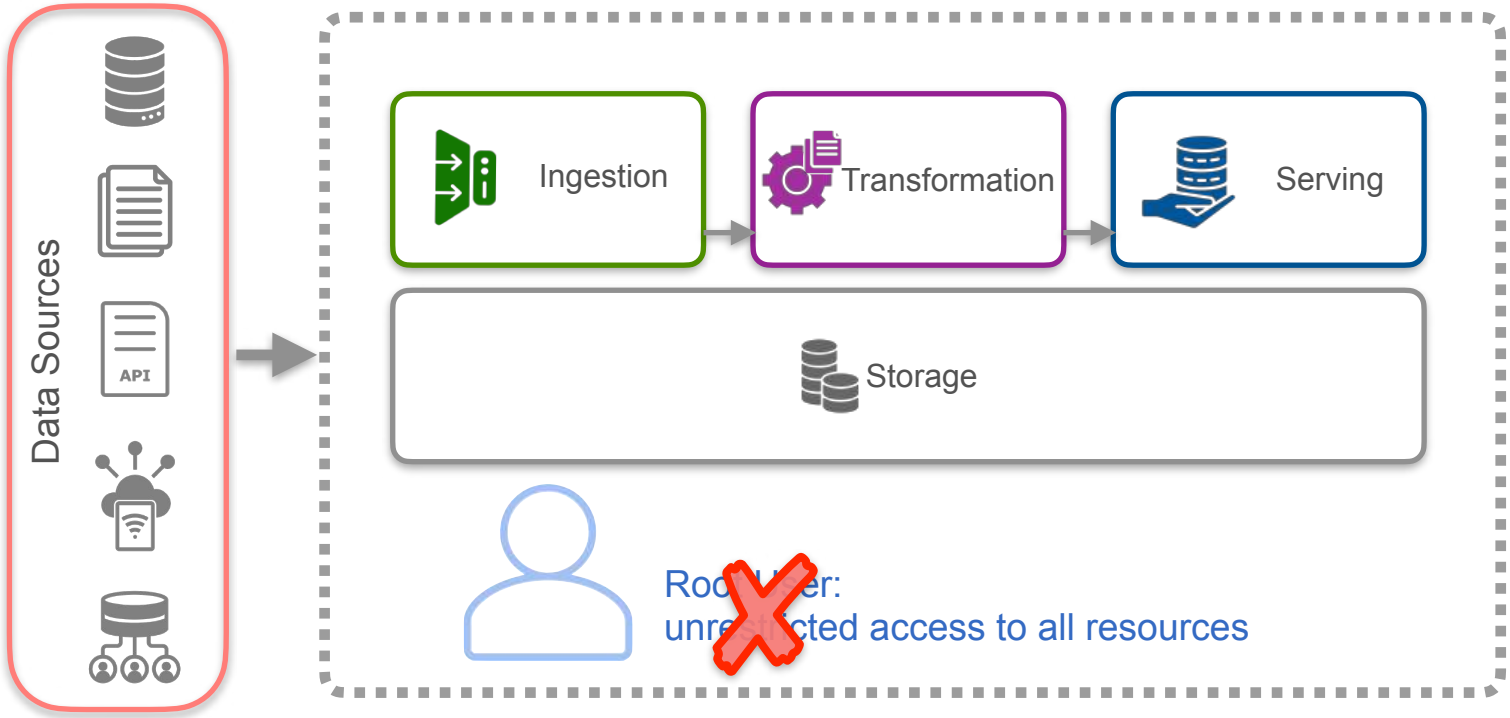
# Security



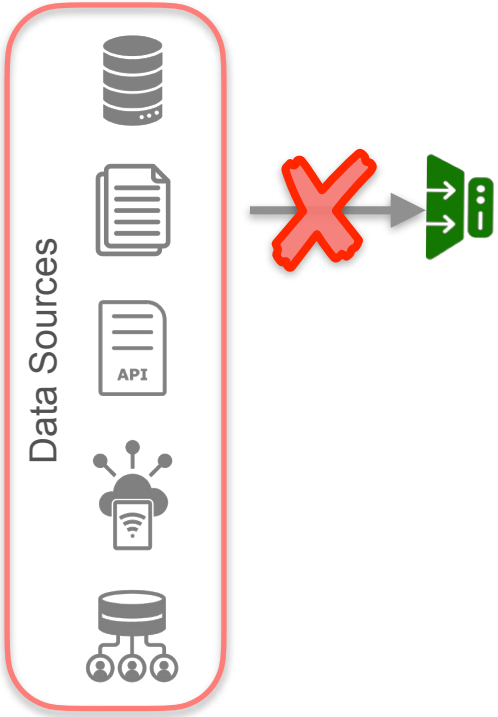

**Principle of Least Privilege**

Give users or applications access to only the essential data and resources they need for only the duration required

# Security


Data Sources
API
Ingestion
Transformation
Serving
Storage
unrestricted access to all resources

# Data Sensitivity




| Id | First Name | Last Name | Credit Card Number |
|---|---|---|---|
| 25 | John | Smith | 457893 |
| 45 | Lara | Jones | 347891 |

Data sensitivity

| Id | First Name | Last Name | Credit Card Number |
|---|---|---|---|
| 25 | J**** | S**** | ****93 |
| 45 | L**** | J**** | ****91 |

# Security in the Cloud

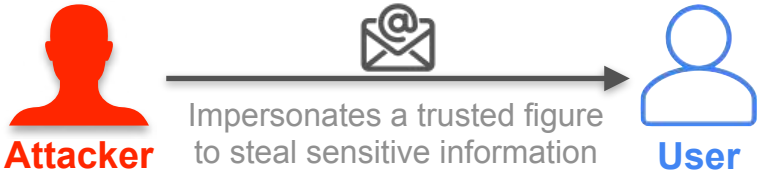
- Identity and Access Management (IAM)
- Encryption Methods
- Networking Protocols

# Security

**Defensive Mindset**

**Be cautious with sensitive data**


**Design for potential attacks**



# Security



Security Theater

DeepLearning.AI

# The Undercurrents of the Data Engineering Lifecycle

## Data Management

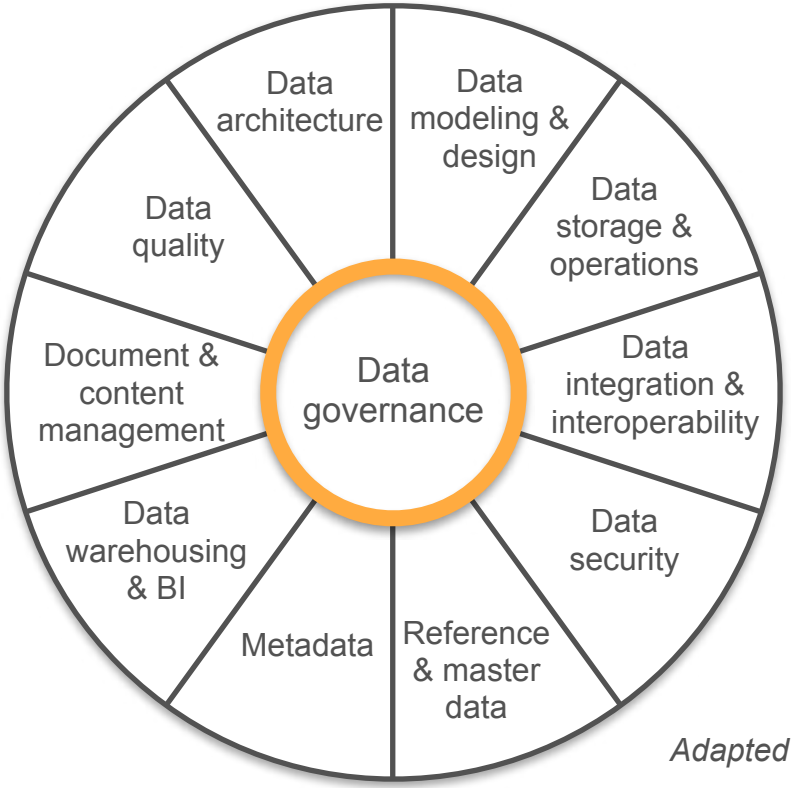
# Data Management

“Data management is the development, execution, and supervision of plans, programs, and practices that deliver, control, protect, and enhance the value of data and information assets throughout their life cycles.”

*DMBOK's Definition*

# Data Management

11
Data Knowledge Areas




*Adapted from: DAMA International*

# Data Governance

“Data governance is, first and foremost, a data management function to ensure the quality, integrity, security, and usability of the data collected by an organization.”

*Data Governance: The definitive Guide*

# Data Governance

"Data governance is, first and foremost, a data management function to ensure the quality, integrity, security, and usability of the data collected by an organization."

*Data Governance: The definitive Guide*

# Data Quality

**High Quality Data**

- Accurate
- Complete
- Discoverable
- Available in a timely manner

**Exactly what stakeholders expect**

**Low Quality Data**

- Inaccurate
- Incomplete
- Hard to find
- Late

**Unusable**

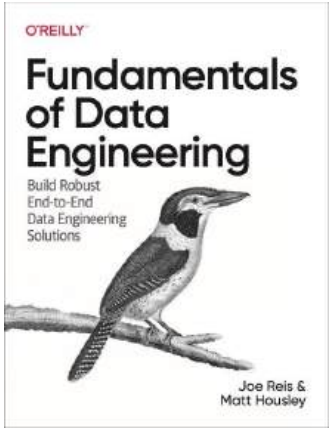
DeepLearning.AI

# The Undercurrents of the Data Engineering Lifecycle

## Data Architecture

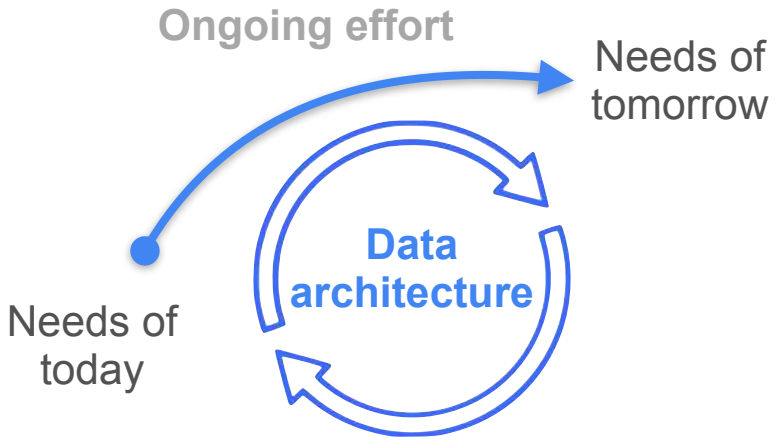
# Data Architecture

“Data architecture is the design of systems to support the evolving data needs of an enterprise, achieved by flexible and reversible decisions reached through a careful evaluation of trade-offs”



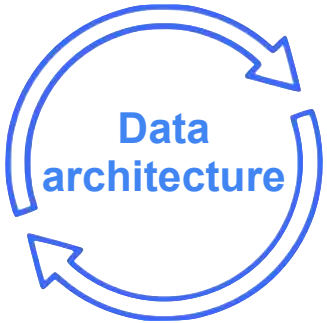
# Data Architecture

“Data architecture is the design of systems to support the evolving data needs of an enterprise, achieved by flexible and reversible decisions reached through a careful evaluation of trade-offs”



# Data Architecture

“Data architecture is the design of systems to support the evolving data needs of an enterprise, achieved by flexible and reversible decisions reached through a careful evaluation of trade-offs”

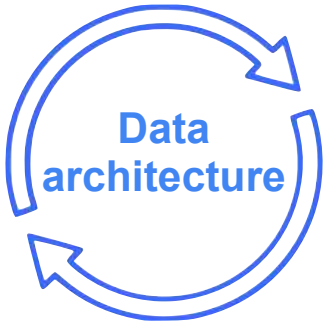


# Data Architecture

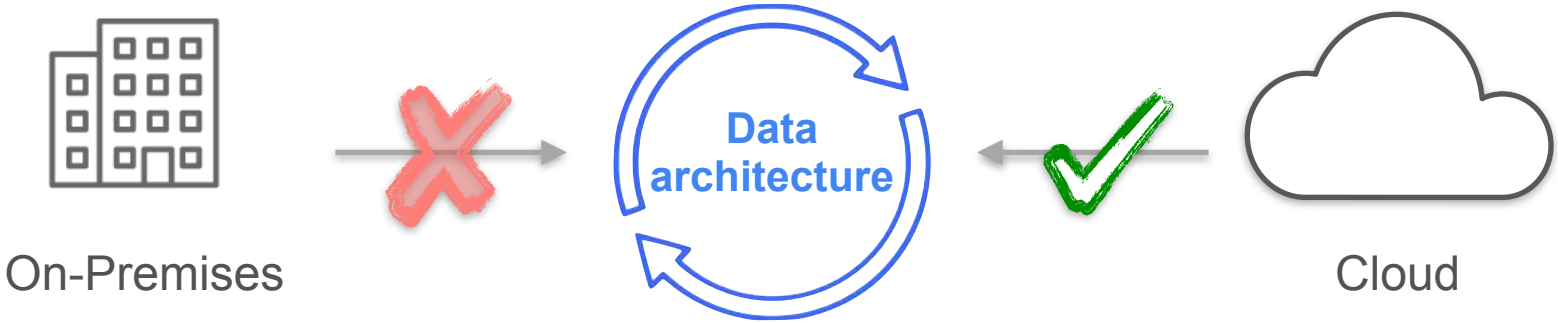
“Data architecture is the design of systems to support the evolving data needs of an enterprise, achieved by flexible and reversible decisions reached through a careful evaluation of trade-offs”

**Trade-offs**

- Performance
- Cost
- Scalability
- …

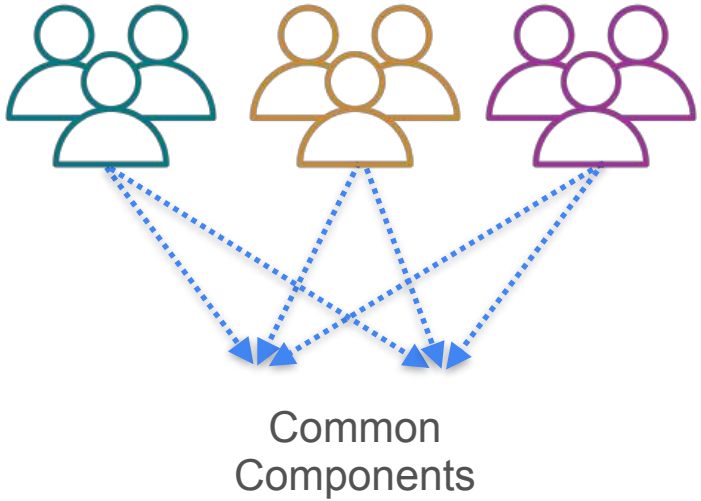


# Data Architecture



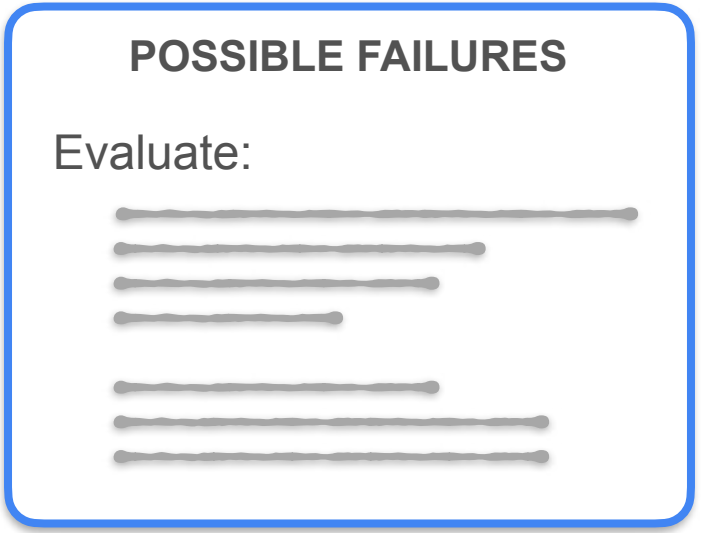
# Principle of Good Data Architecture

1. Choose common components wisely



# Principle of Good Data Architecture

1. Choose common components wisely
2. Plan for failure!

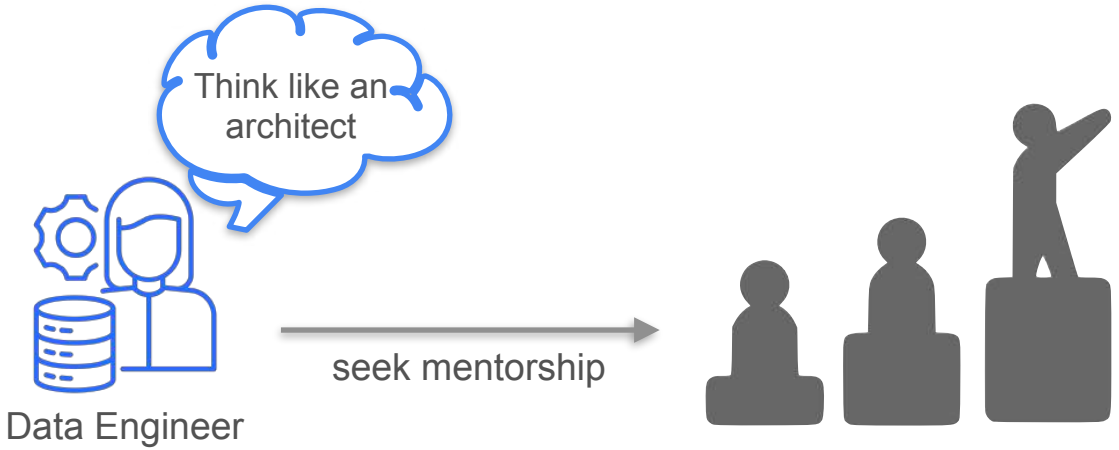


# Principle of Good Data Architecture

1. Choose common components wisely
2. Plan for failure!
3. Architect for scalability

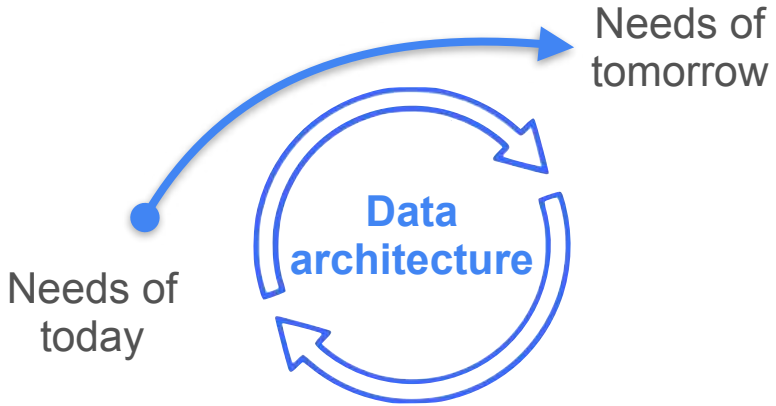
# Principle of Good Data Architecture

1. Choose common components wisely
2. Plan for failure!
3. Architect for scalability
4. Architecture is leadership



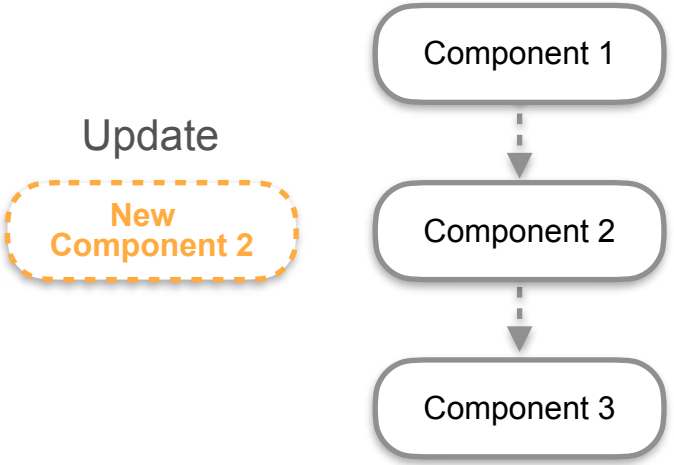
# Principle of Good Data Architecture

1. Choose common components wisely
2. Plan for failure!
3. Architect for scalability
4. Architecture is leadership
5. Always be architecting



# Principle of Good Data Architecture

1. Choose common components wisely
2. Plan for failure!
3. Architect for scalability
4. Architecture is leadership
5. Always be architecting
6. Build loosely coupled systems
7. Make reversible decisions



# Principle of Good Data Architecture

1. Choose common components wisely
2. Plan for failure!
3. Architect for scalability
4. Architecture is leadership
5. Always be architecting
6. Build loosely coupled systems
7. Make reversible decisions
8. Prioritize security

**Principle of least privilege**

**Zero-trust principle**

# Principle of Good Data Architecture

1. Choose common components wisely
2. Plan for failure!
3. Architect for scalability
4. Architecture is leadership
5. Always be architecting
6. Build loosely coupled systems
7. Make reversible decisions
8. Prioritize security
9. Embrace FinOps

DeepLearning.AI

# The Undercurrents of the Data Engineering Lifecycle

## DataOps

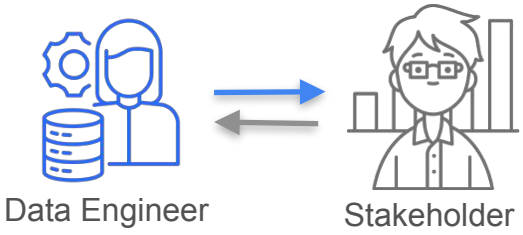
# DataOps

**DataOps**

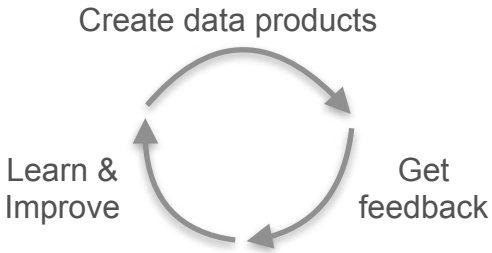
Improves the development process and quality of data products.

It's a set of cultural habits and practices.

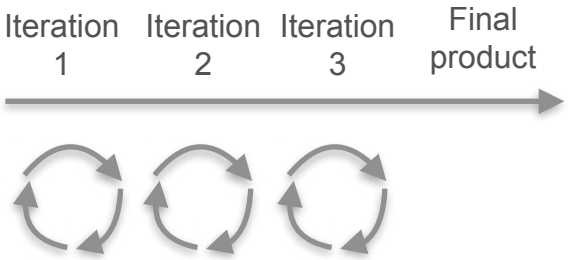
### Communication & Collaboration




### Continuous Improvement

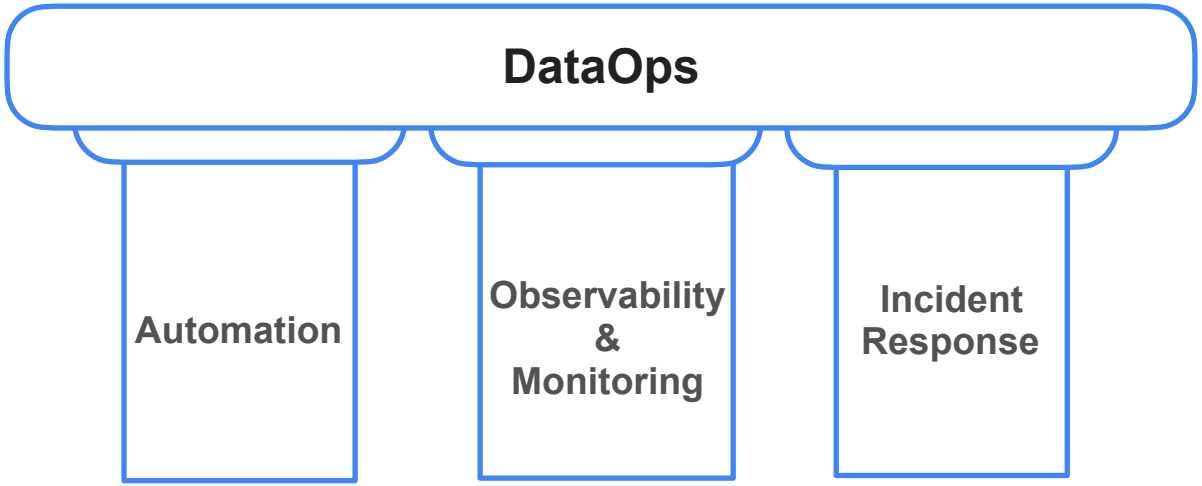



### Rapid Iteration




**DevOps practices** ⟵ **Agile methodology**

# Pillars of DataOps



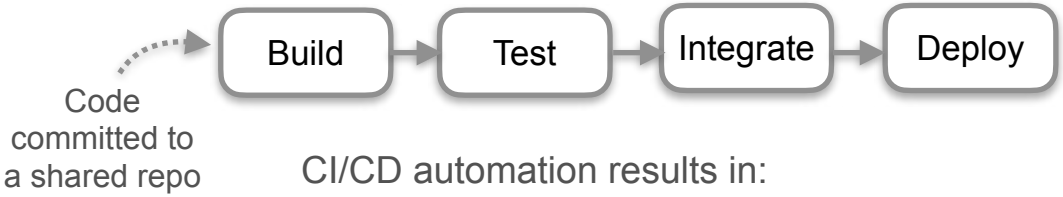

Goal: Provide high-quality data products

# Pillar 1: Automation

**DevOps**
**(Applies to software build)**

**DataOps**

**Continuous Integration and Continuous Delivery (CI/CD)**



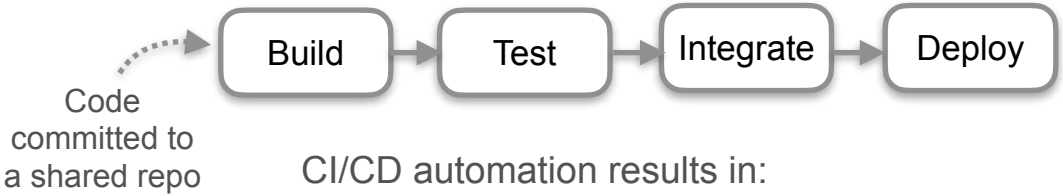

CI/CD automation results in:

- faster deployment
- fewer errors

# Pillar 1: Automation

## DevOps
## (Applies to software build)

**Continuous Integration and Continuous Delivery (CI/CD)**




CI/CD automation results in:

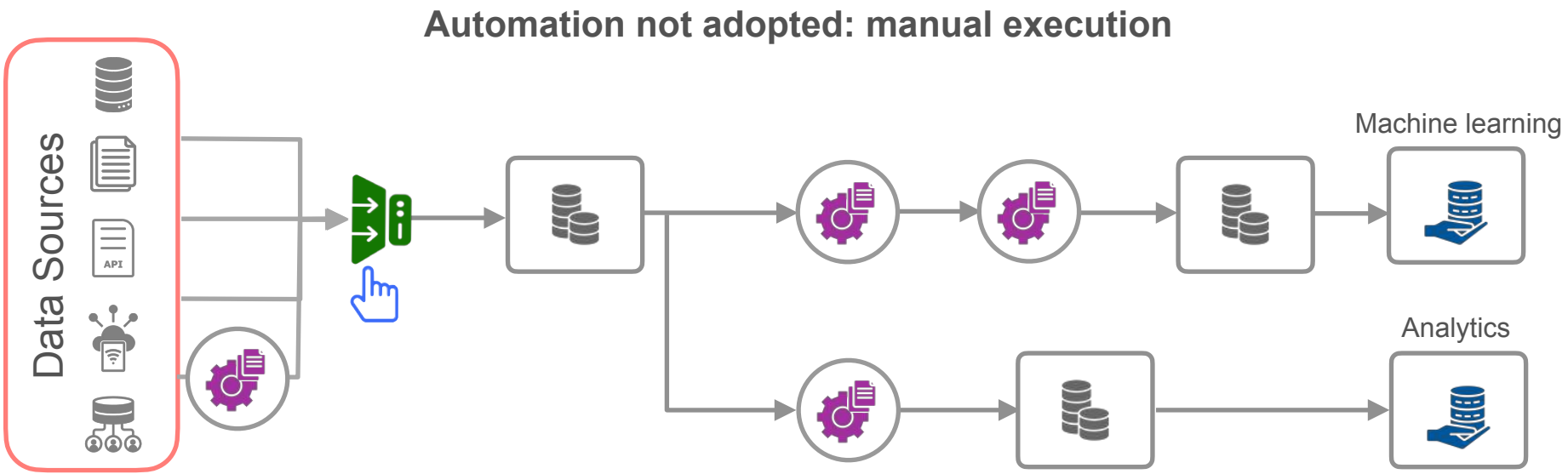
- faster deployment
- fewer errors

## DataOps
## (Applies to data processing)

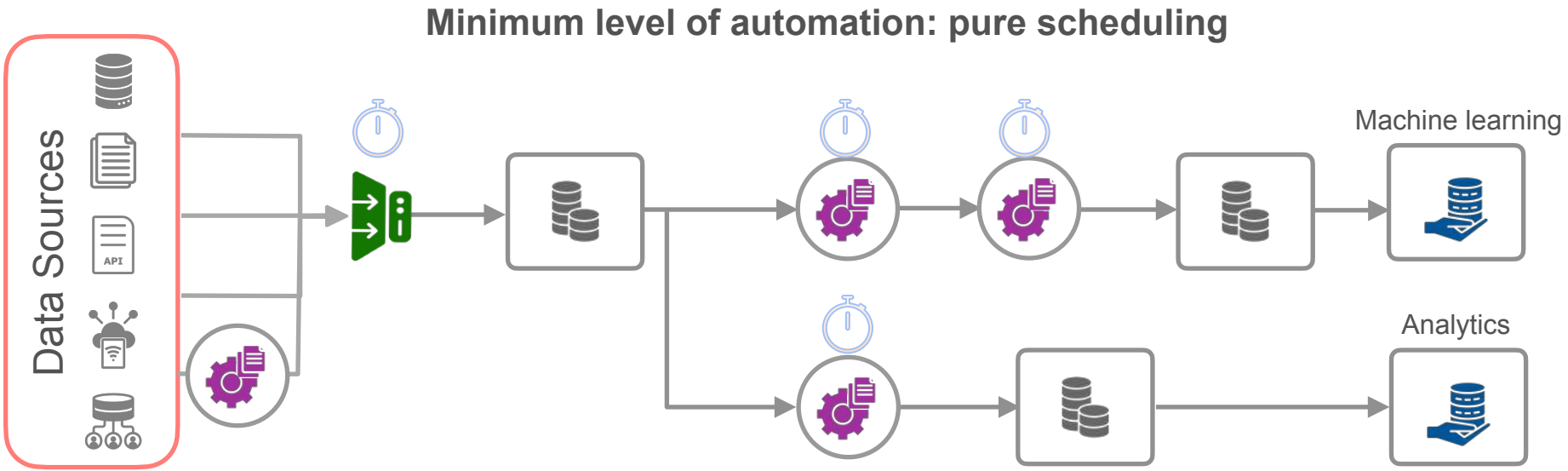
**Automated change management:**

- Code
- Configuration
- Environment
- Data processing pipelines
- Data

# Pillar 1: Automation

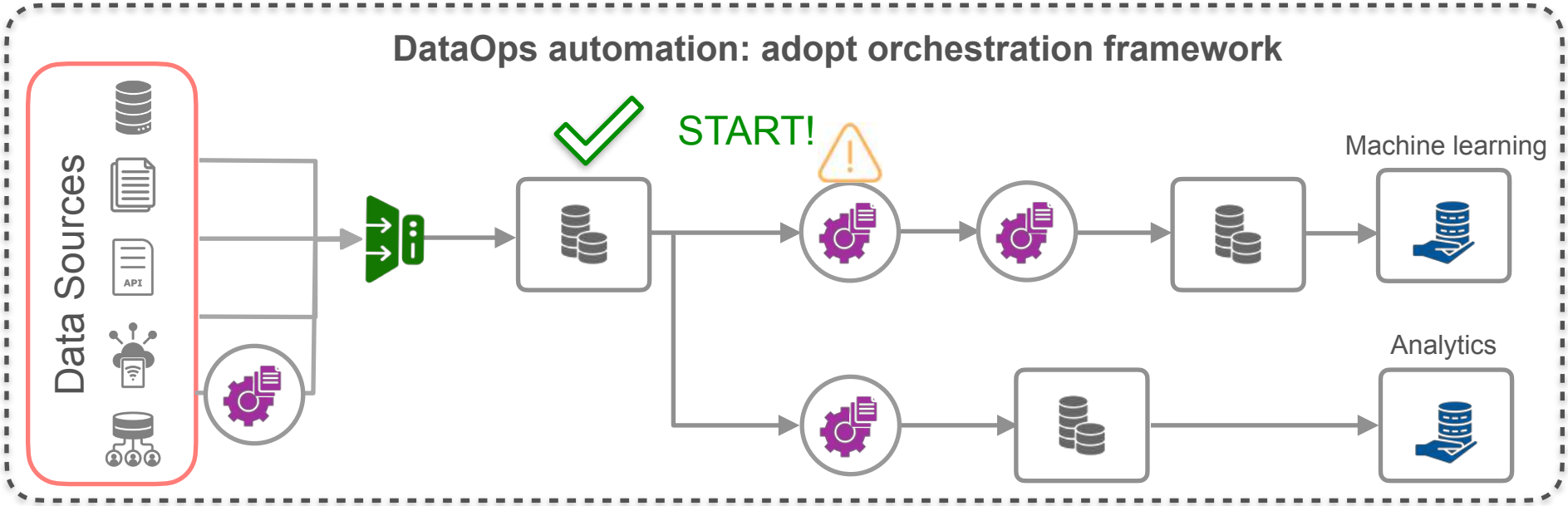


# Pillar 1: Automation



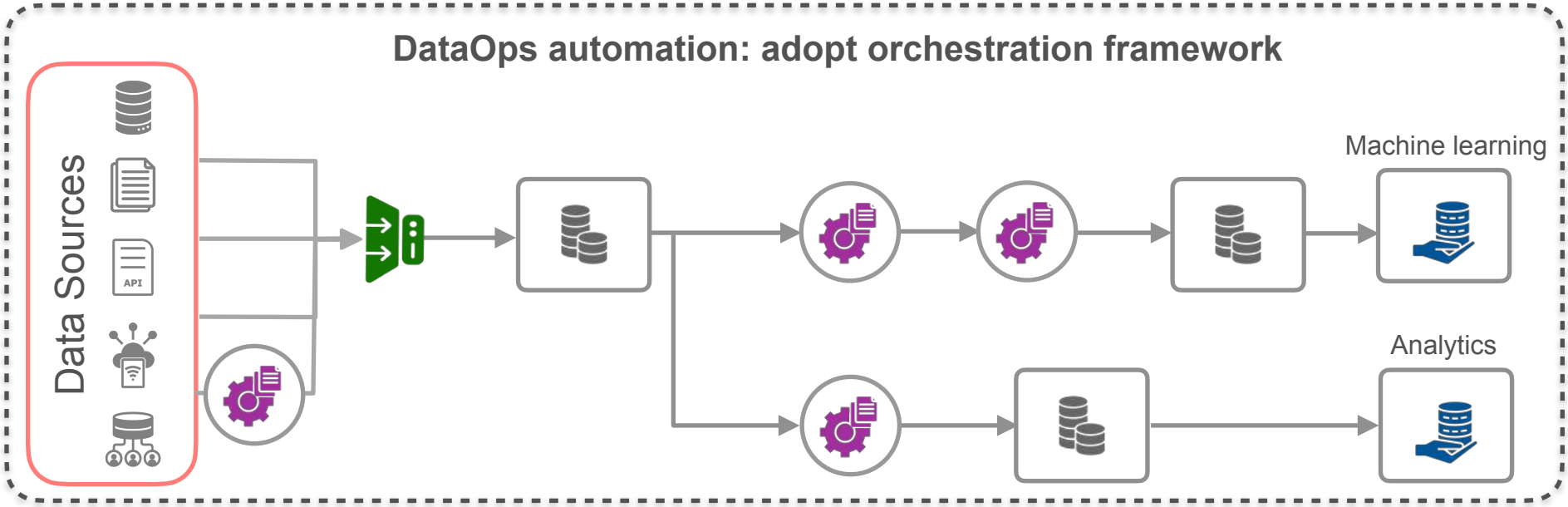

Prone to failure as the number of jobs grows

# Pillar 1: Automation



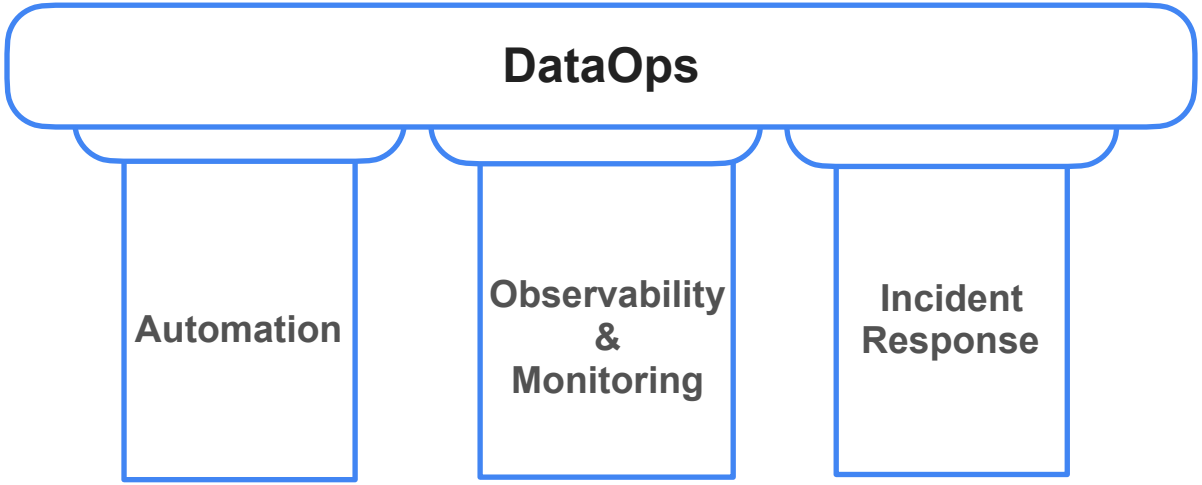

**Checks the dependencies between tasks before each task is run**

# Pillar 1: Automation




**Automatic verification and deployment of new aspects**

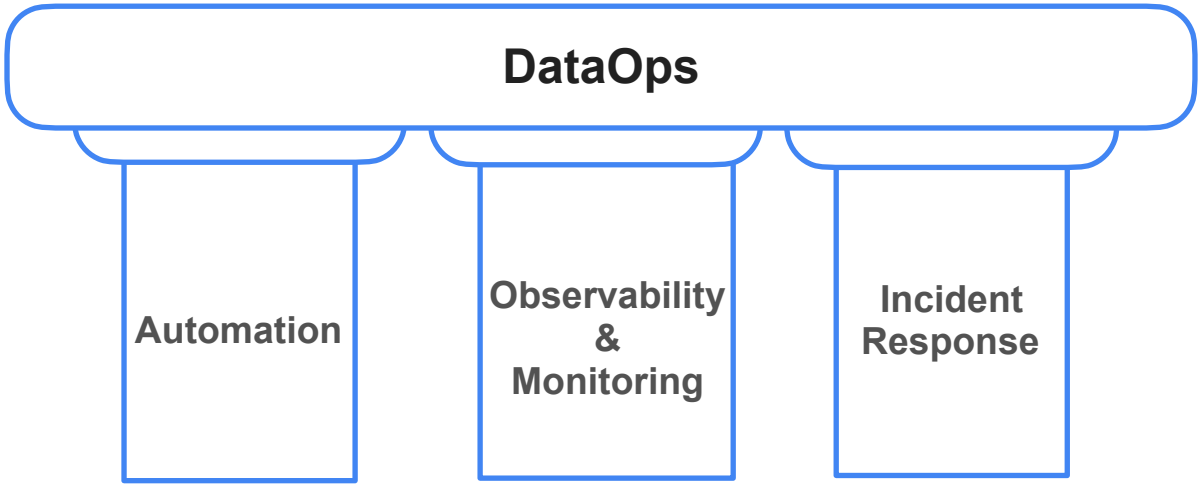
# Pillar 2: Observability & Monitoring




"Everything fails all the time"

- Werner Vogels

# Pillar 3: Incident Response



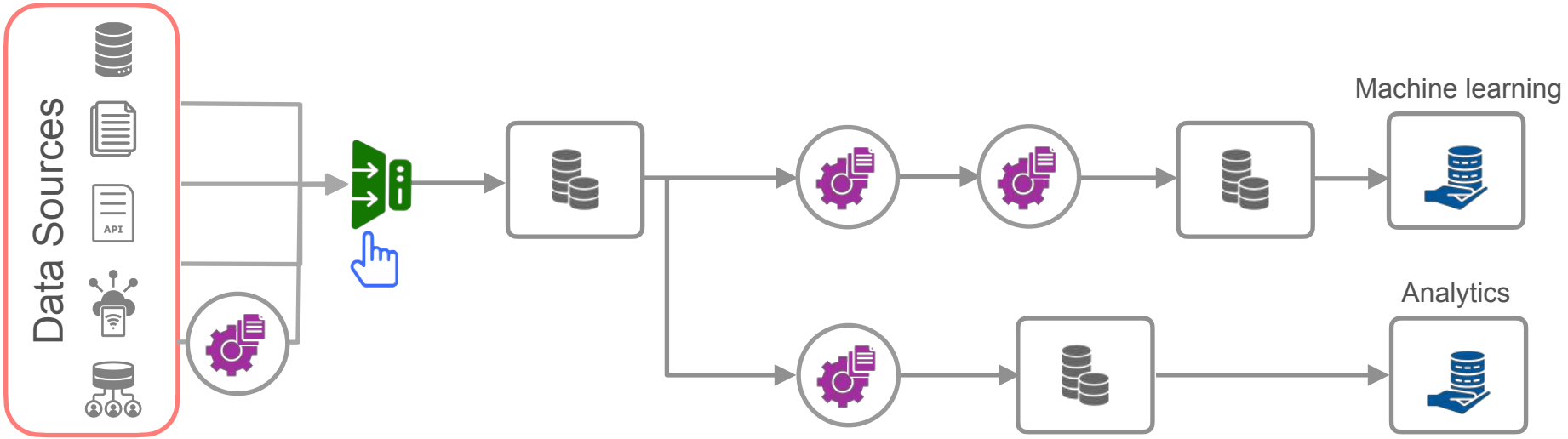

- ✓ Rapidly identify the incident's root causes
- ✓ Quickly resolve an incident
- ✓ Identify technology and tools
- ✓ Coordinate the efforts of the data team

DeepLearning.AI

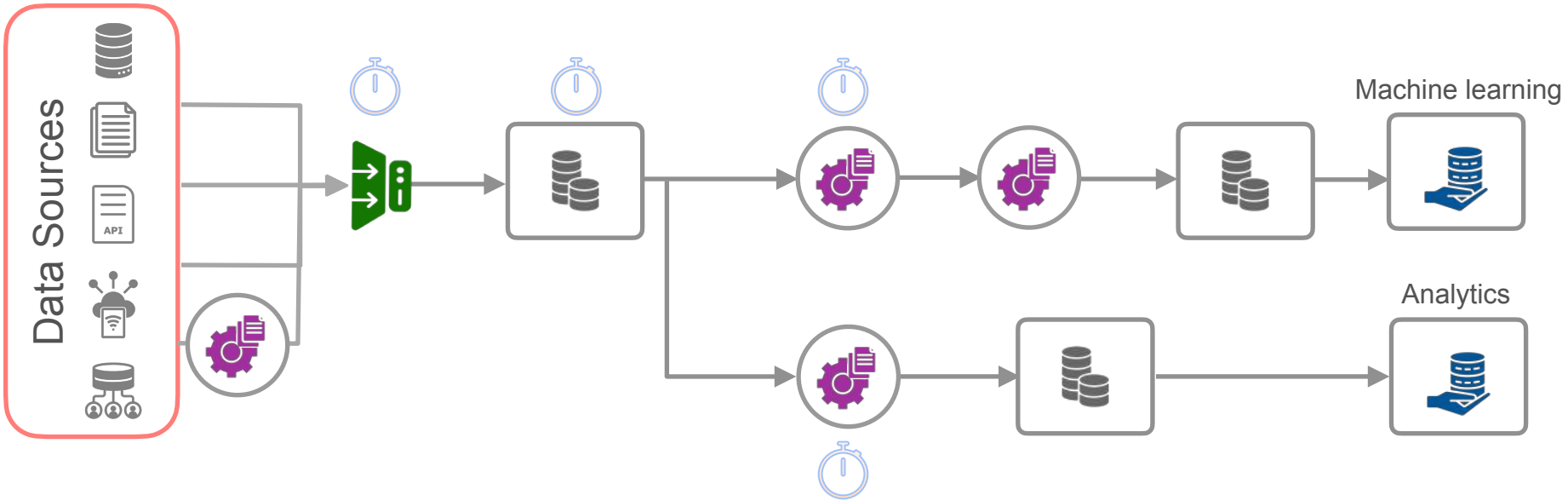
# The Undercurrents of the Data Engineering Lifecycle

## Orchestration

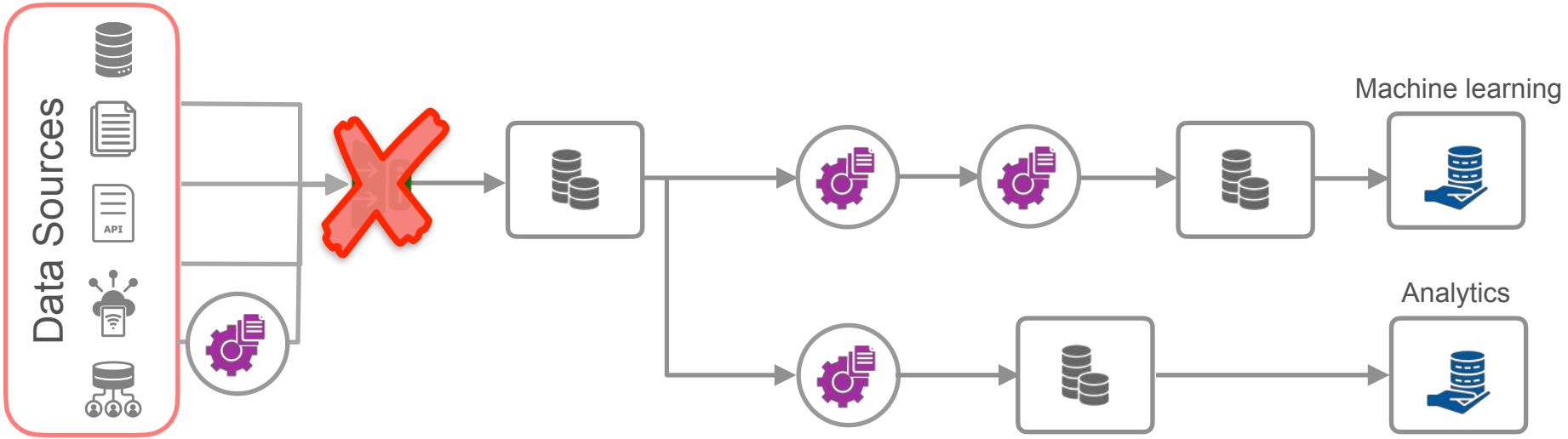
# Manual Execution



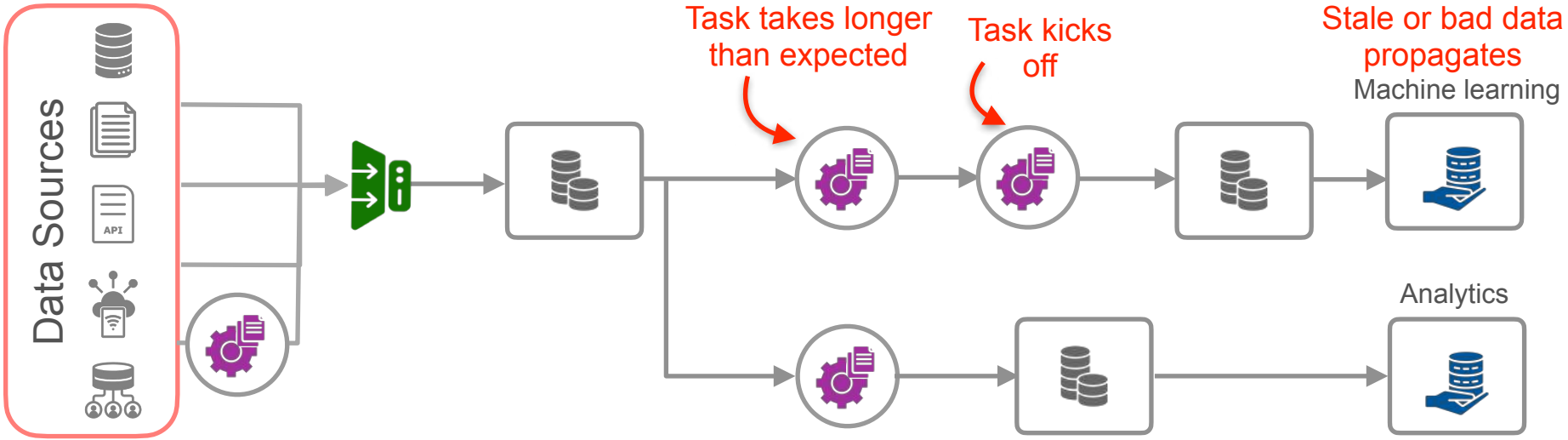
# Pure Scheduling



# Pure Scheduling



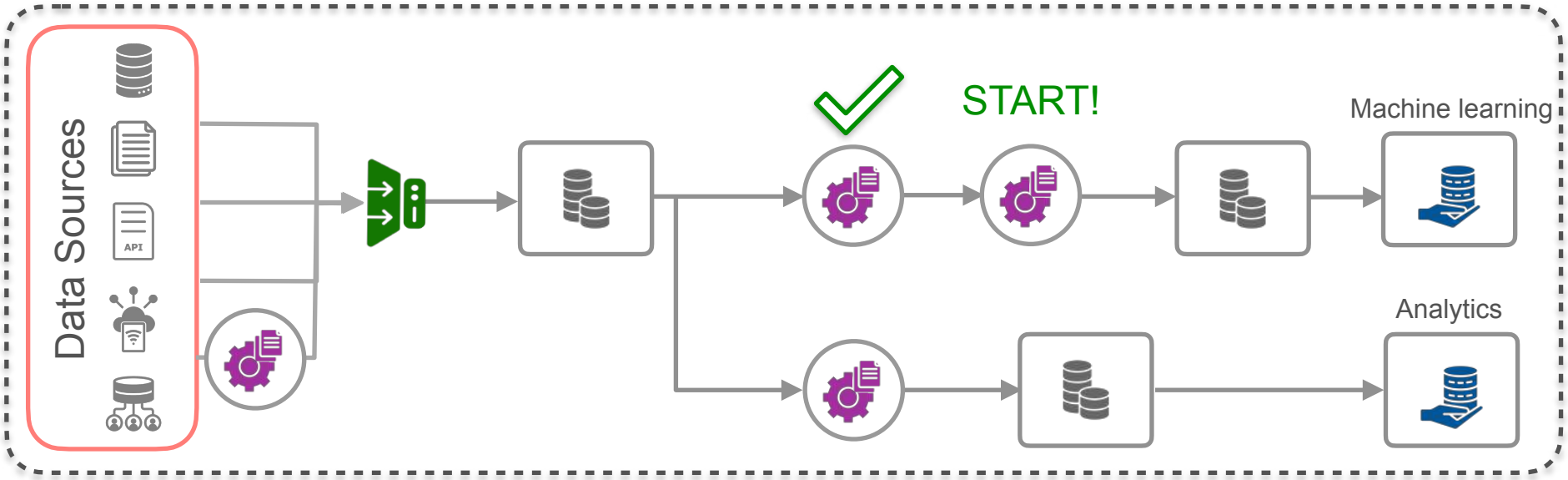
# Pure Scheduling



# Orchestration Frameworks



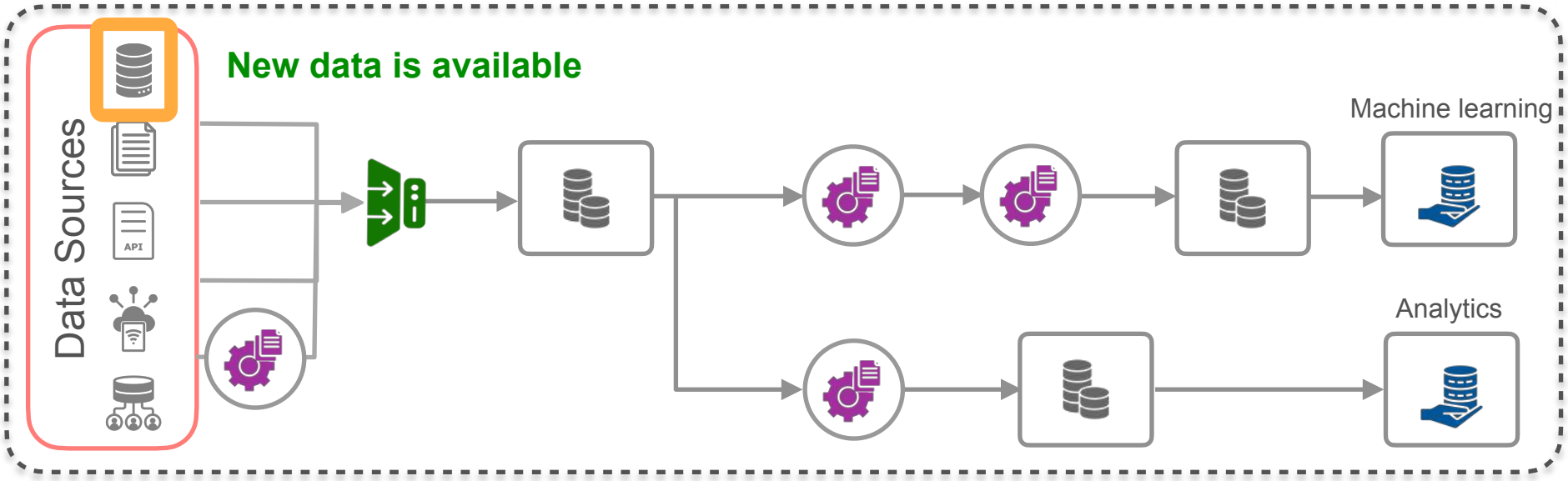

# Time-based scheduling




Orchestration frameworks:

- Automate pipeline with complex dependencies
- Monitor pipeline

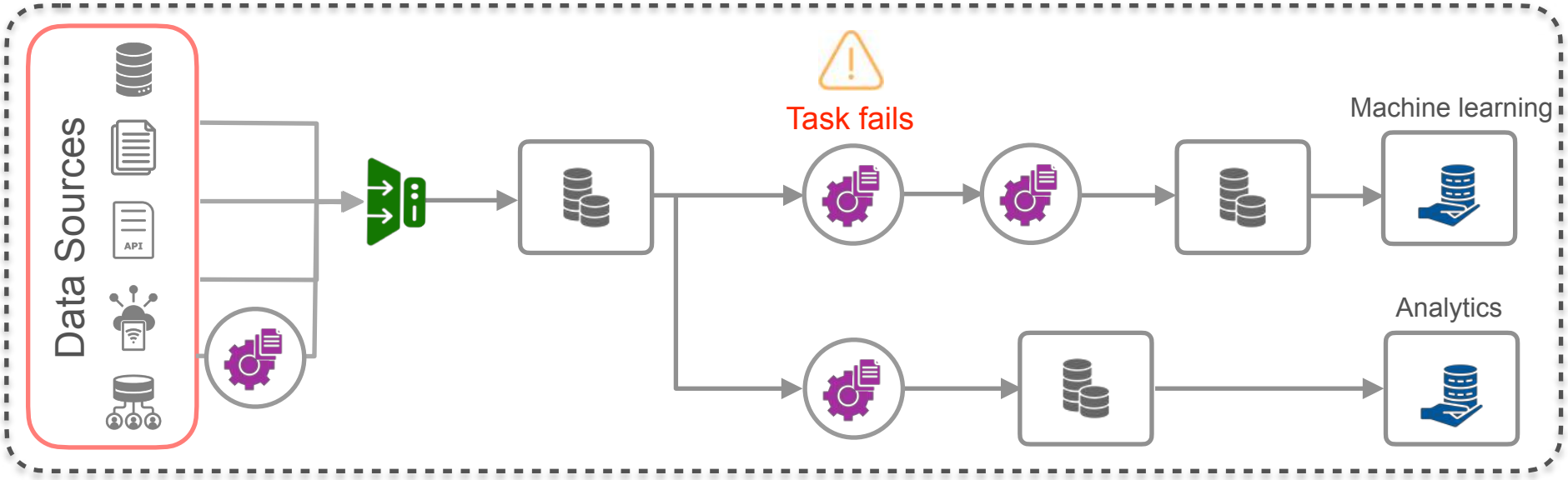
# Event-based triggers




Orchestration frameworks:

- Automate pipeline with complex dependencies
- Monitor pipeline

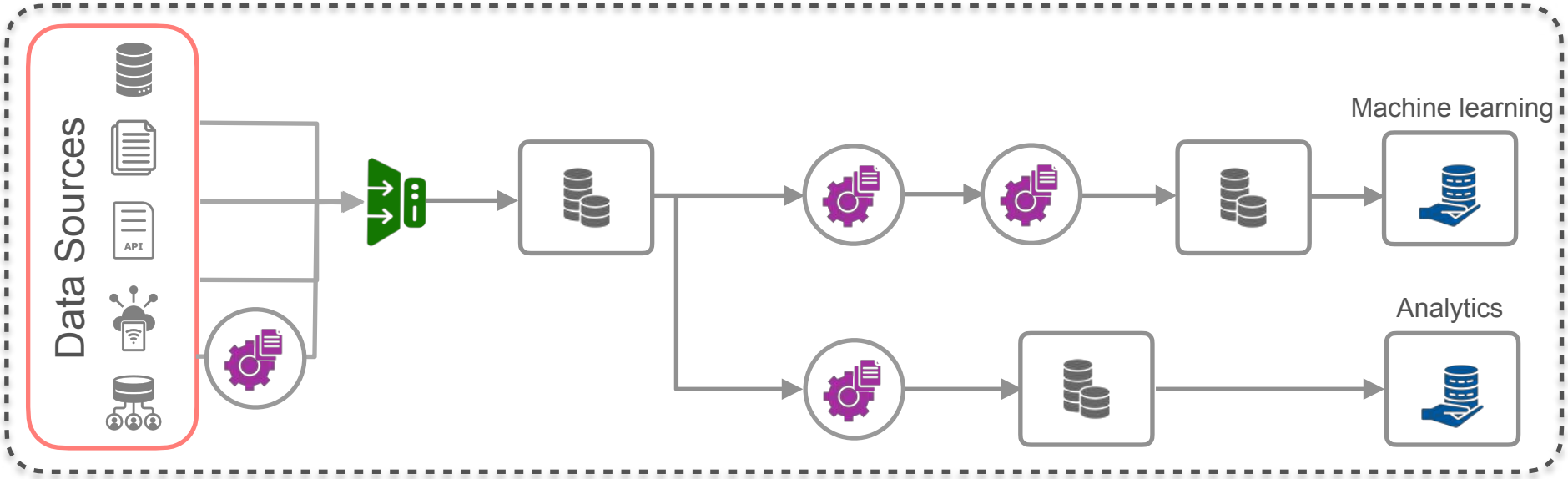
# Set up monitoring & alerts



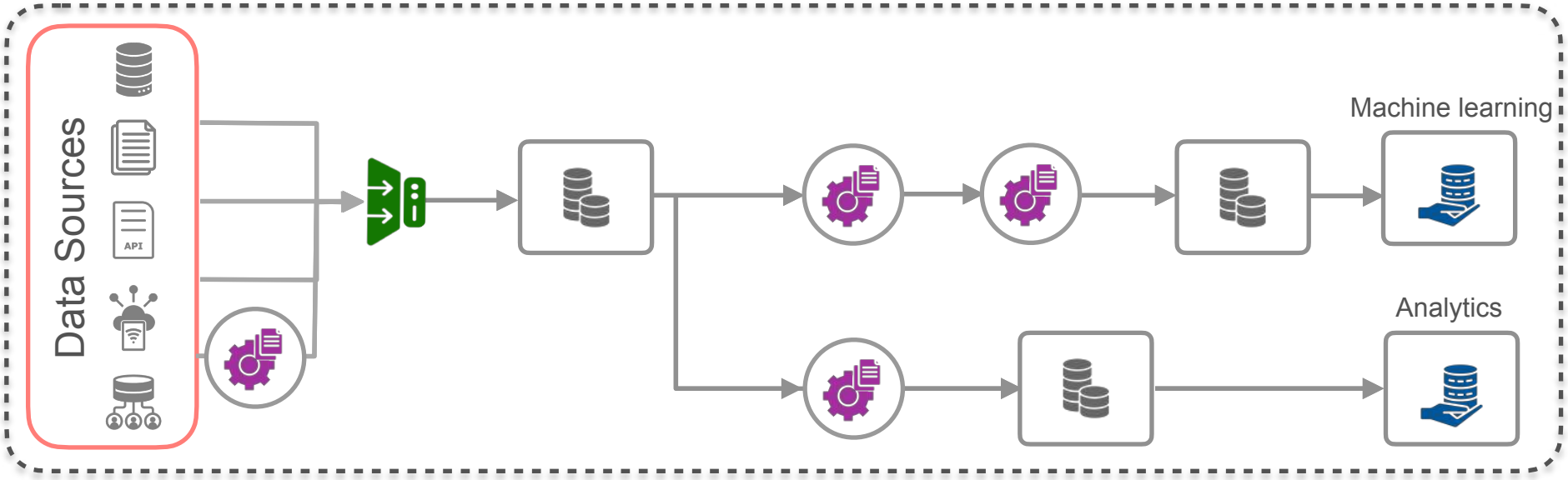

Orchestration frameworks:

- Automate pipeline with complex dependencies
- Monitor pipeline

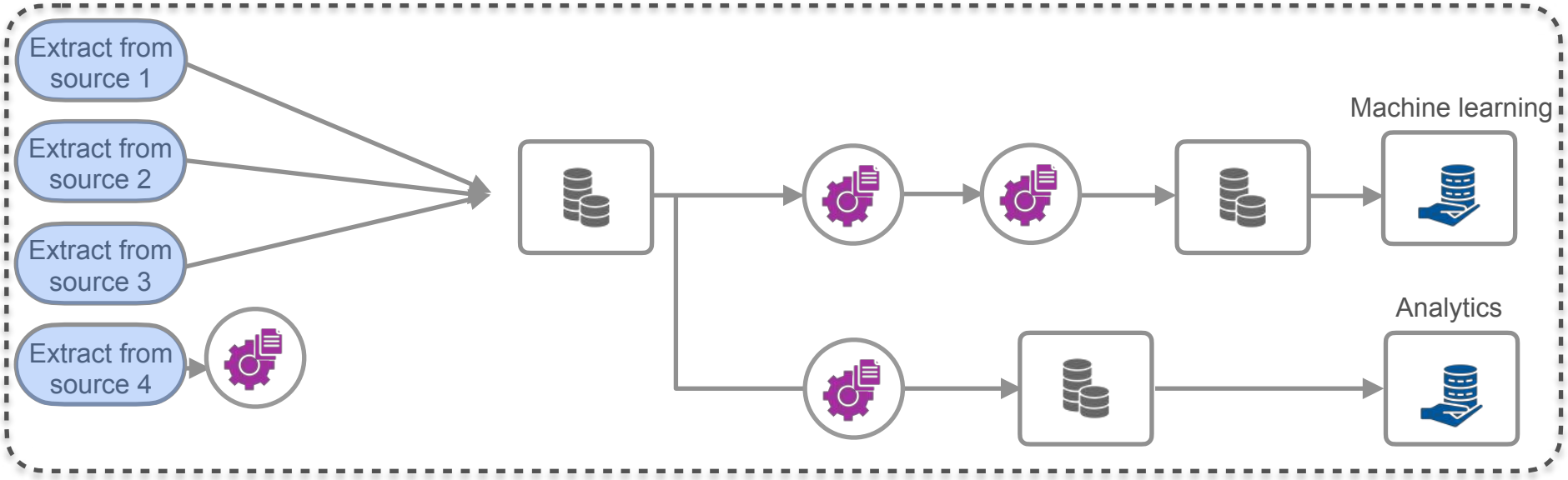
# Directed Acyclic Graph



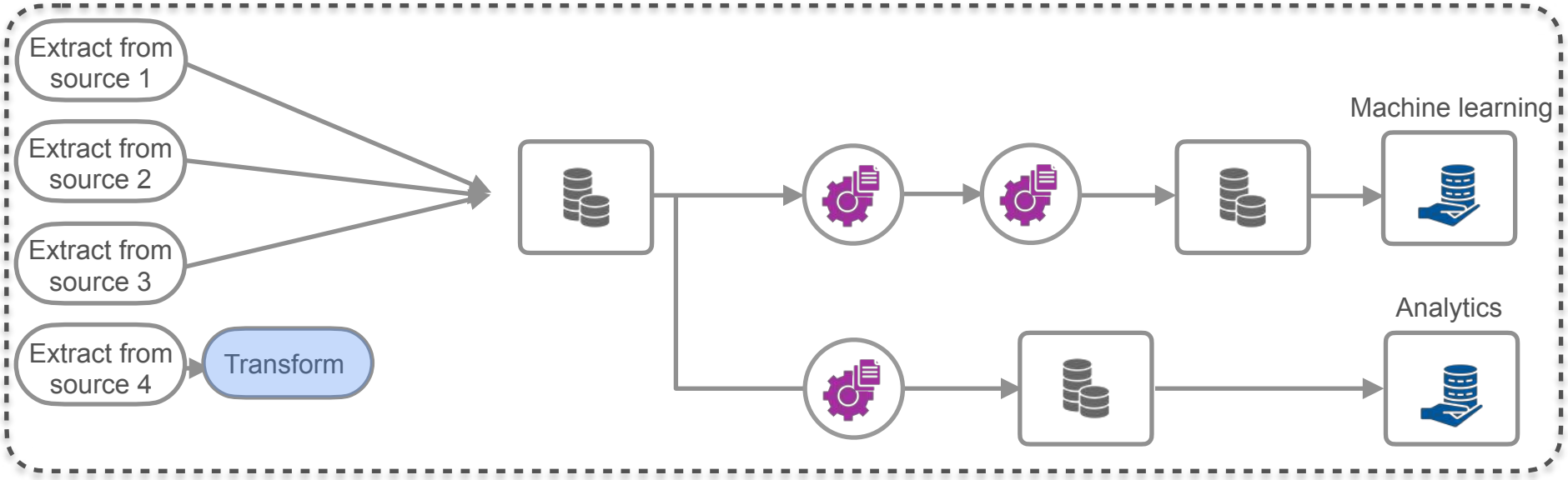
# Directed Acyclic Graph (DAG)



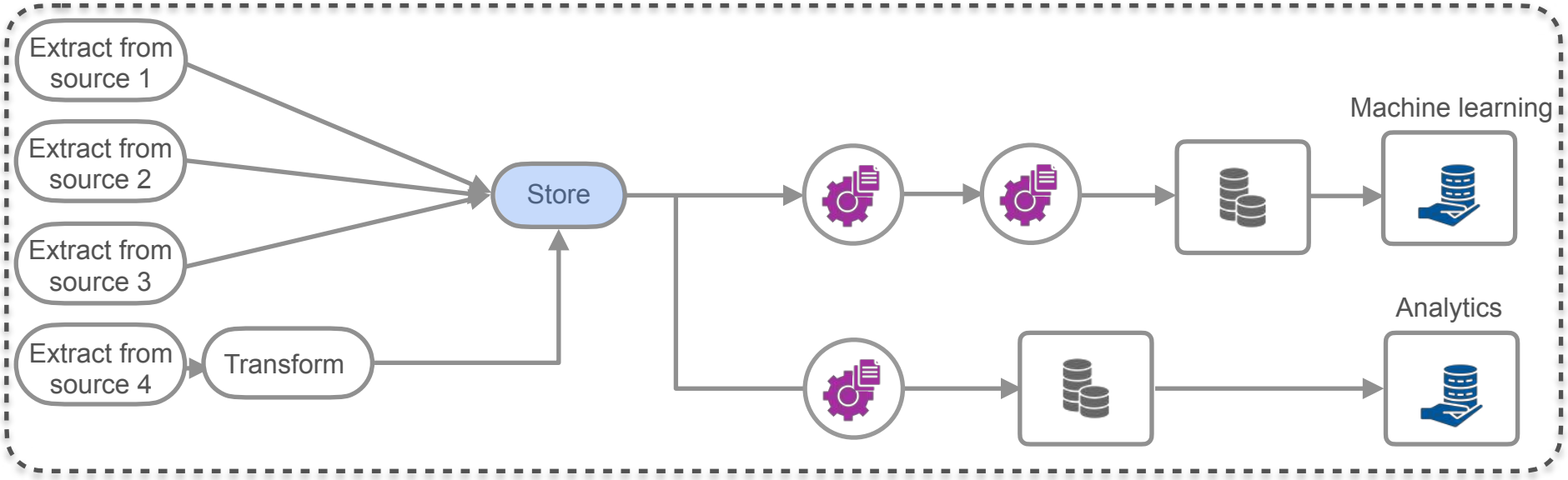
# Directed Acyclic Graph (DAG)



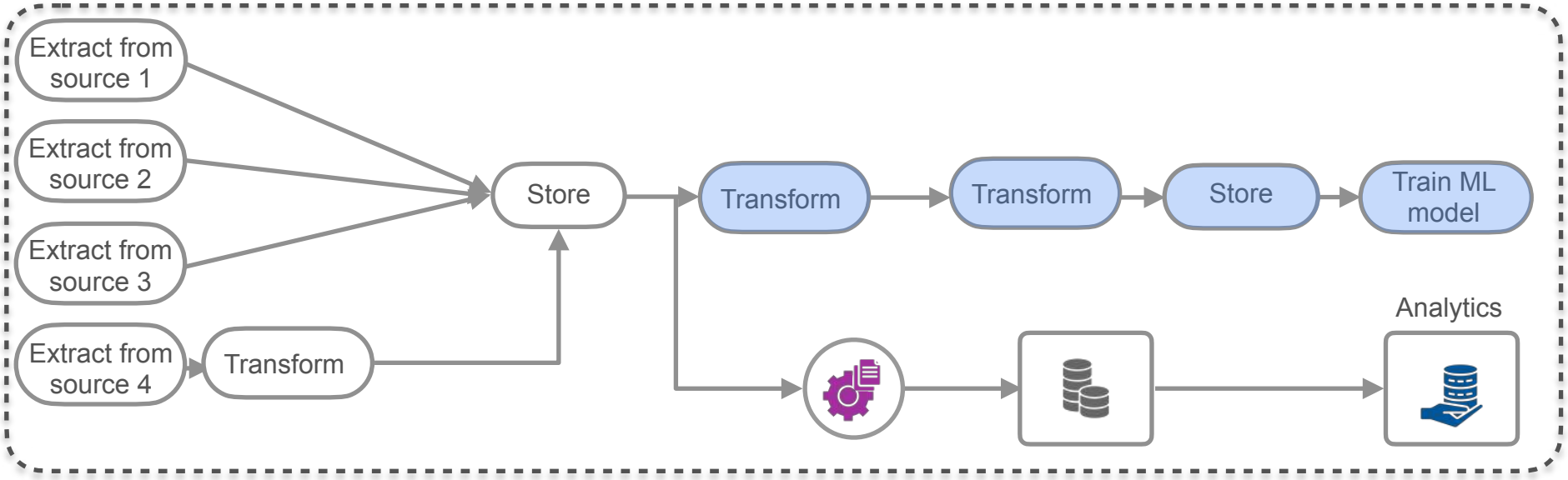
# Directed Acyclic Graph (DAG)



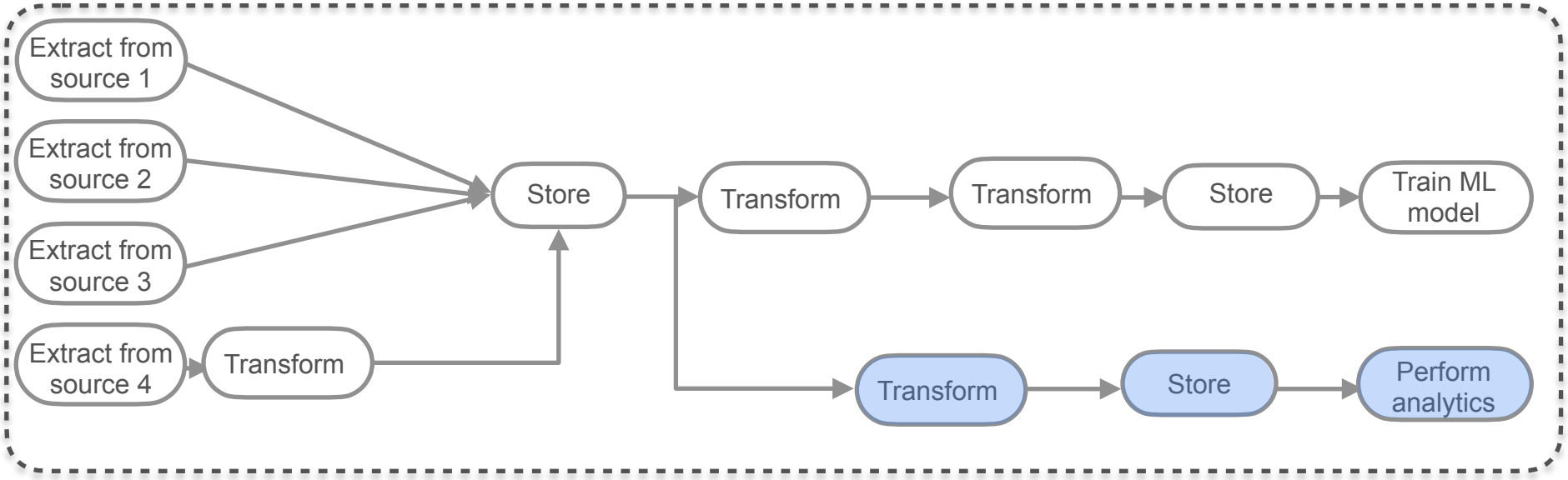
# Directed Acyclic Graph (DAG)



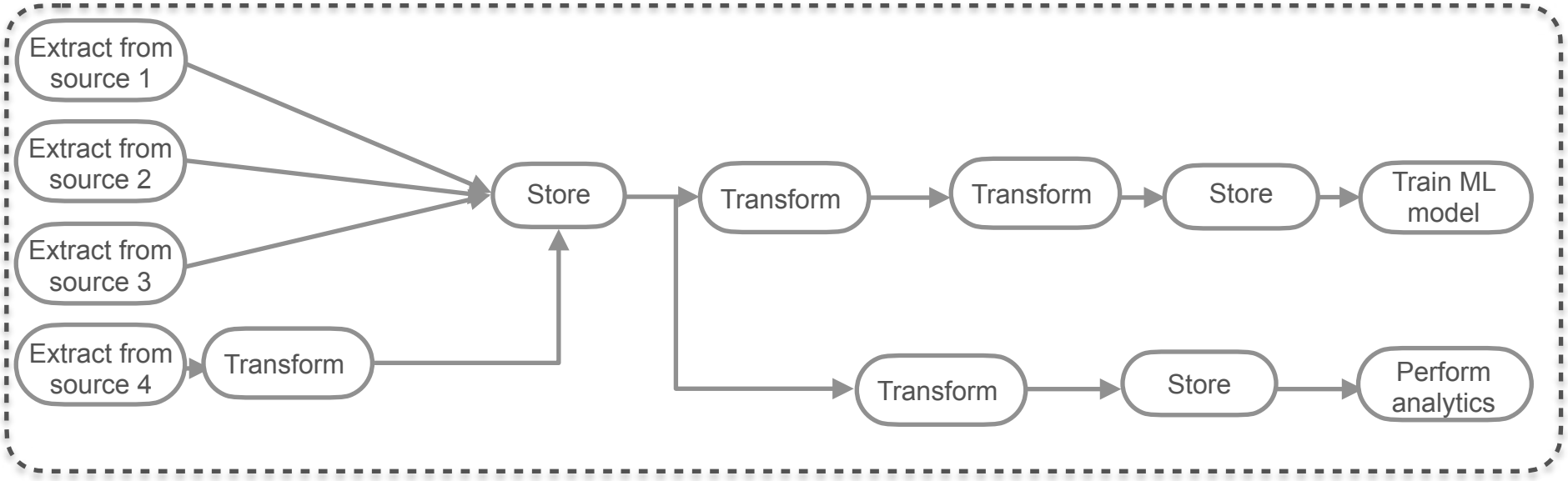
# Directed Acyclic Graph (DAG)



# Directed Acyclic Graph (DAG)

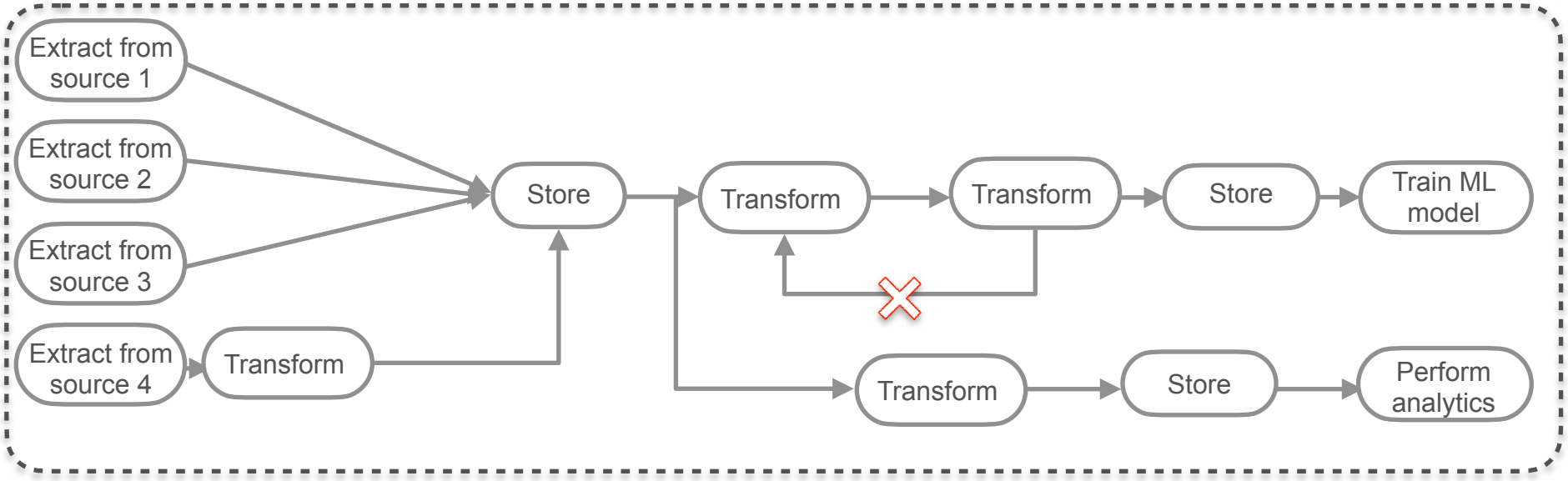


# Directed Acyclic Graph (DAG)




**Directed** Data flows in one direction

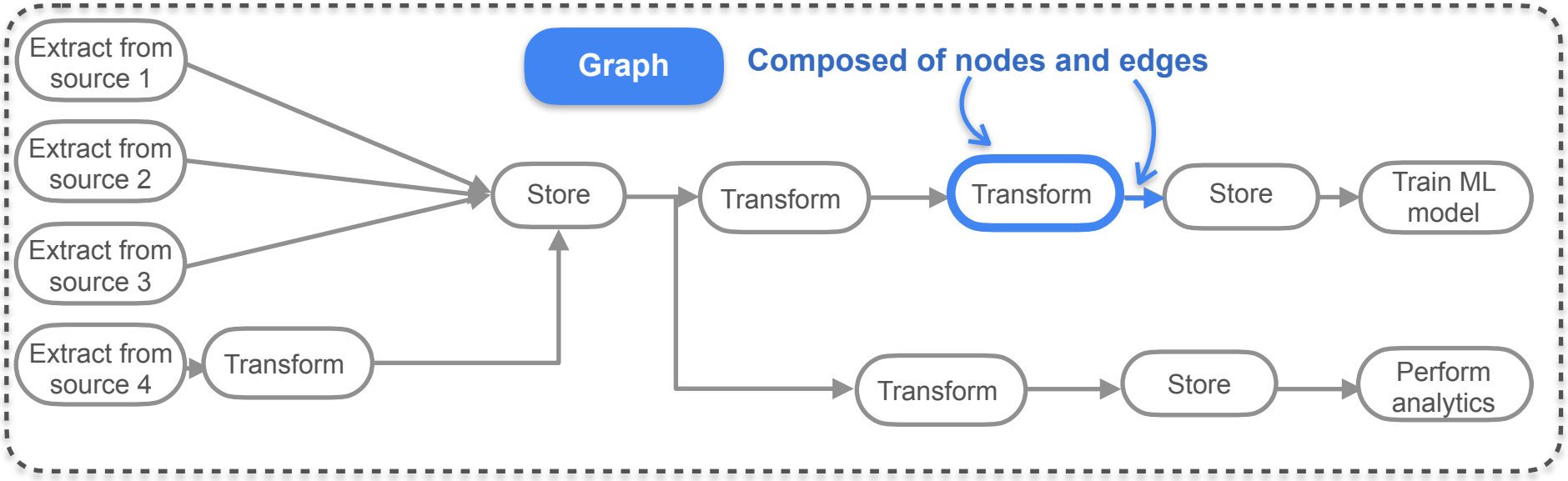
# Directed Acyclic Graph (DAG)




**Directed** Data flows in one direction

**Acyclic** Data doesn't flow backward

# Directed Acyclic Graph (DAG)




**Directed** Data flows in one direction

**Acyclic** Data doesn't flow backward

DeepLearning.AI

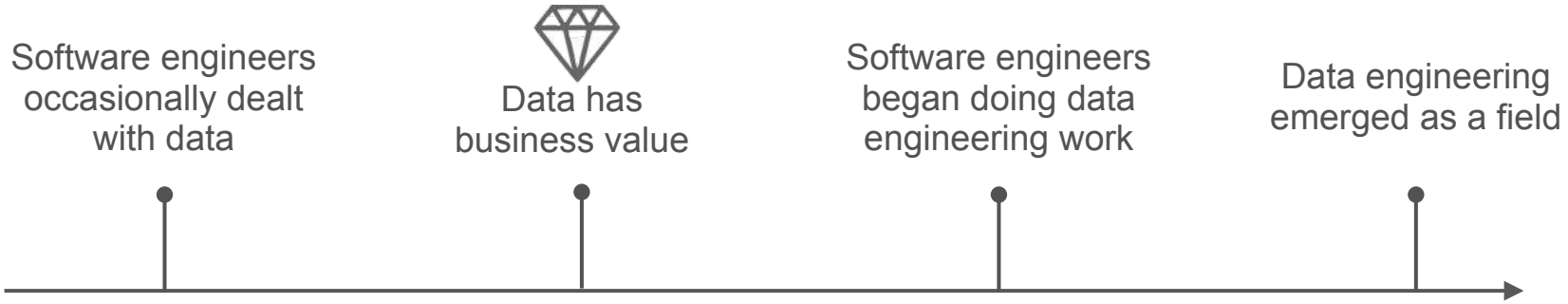
# The Undercurrents of the Data Engineering Lifecycle

## Software Engineering

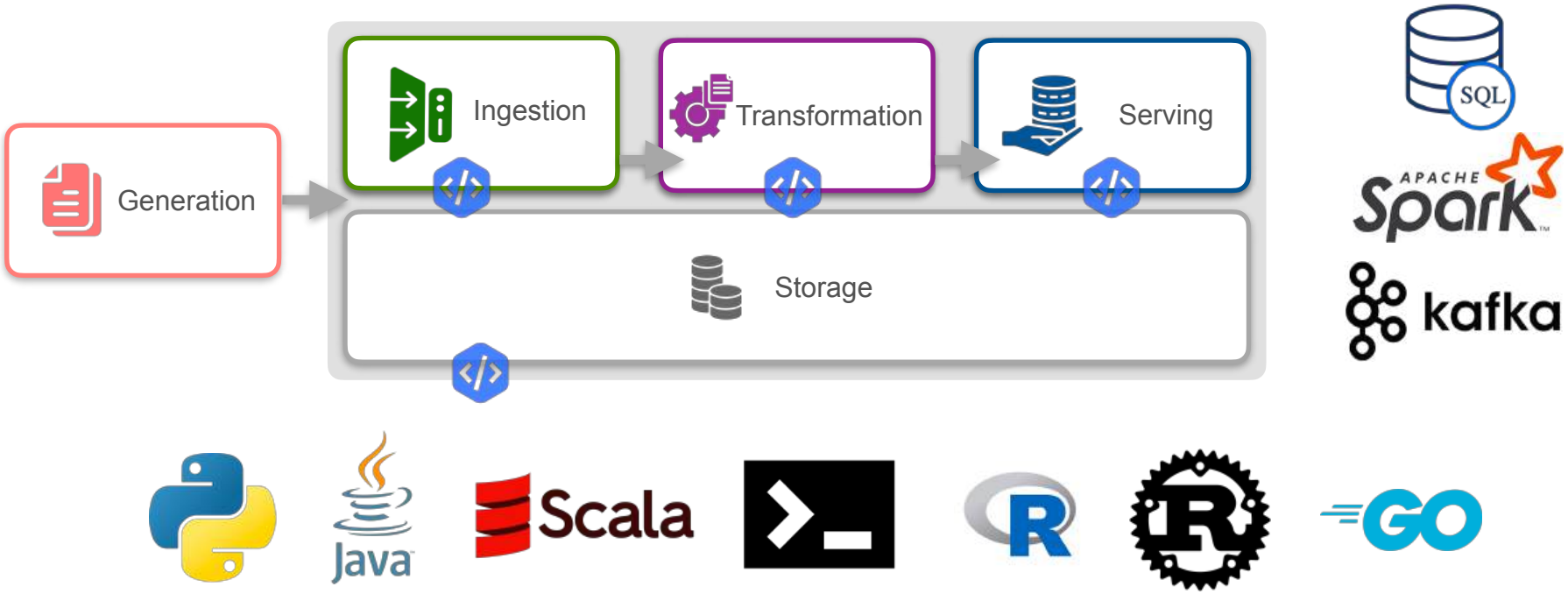
# Software Engineering

**Software engineering**

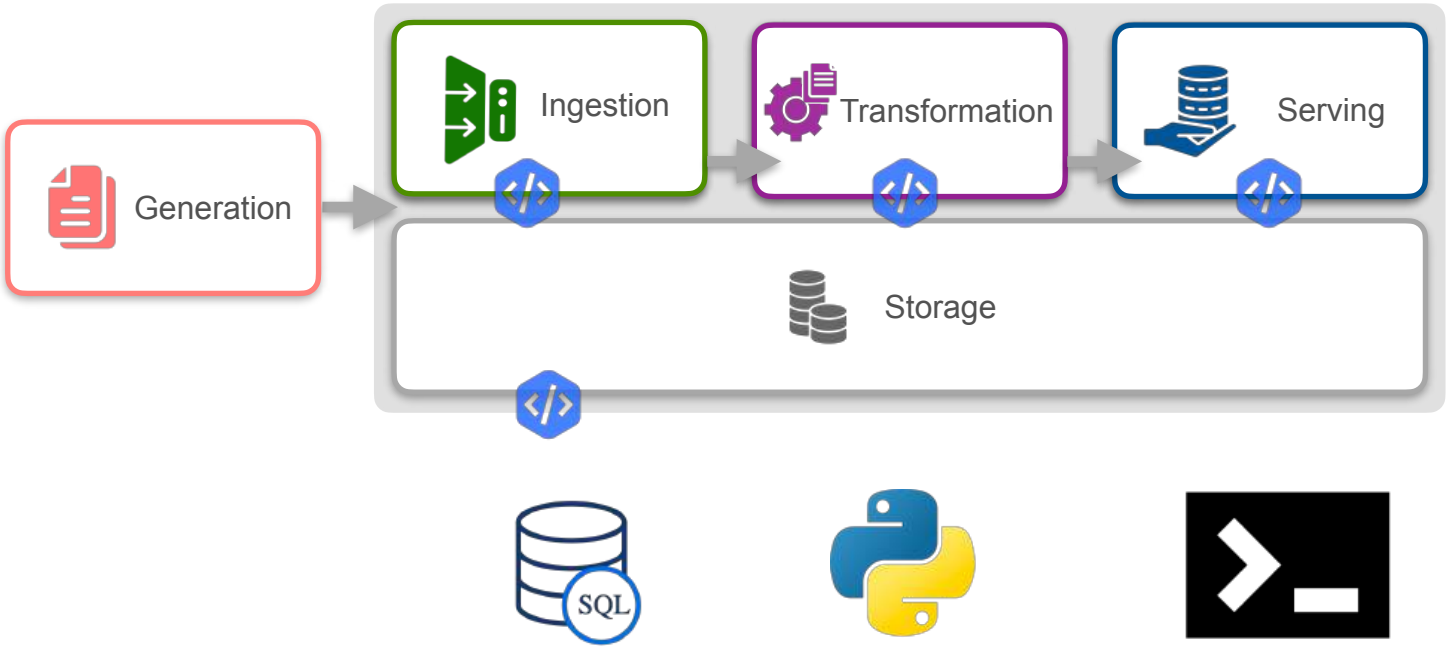
The design, development, deployment, and maintenance of software applications.



# Writing Code as a Data Engineer



# Writing Code as a Data Engineer



# Writing Code as a Data Engineer

**Other coding use cases:**

- Open source frameworks
- Infrastructure as code
- Pipeline as code
- Everyday general-purpose problem solving

DeepLearning.AI

Practical Examples on AWS

# The Data Engineering Lifecycle on AWS

# Databases

Amazon Relational Database Service (RDS)

- Provisions database instances with the relational database engine of your choice
- Simplifies the operational overhead involved with provisioning and hosting a relational database

Amazon DynamoDB

- A serverless NoSQL database option
- Create stand-alone tables that are virtually unlimited in their total size
- Has a flexible schema
- Best suited for applications that require low-latency access to large volumes of data

# Streaming Sources

Source Systems

Amazon Kinesis Data Streams

- Set up as a source system streaming real-time user activities from a sales platform log

Amazon Simple Queue Service (SQS)

- Handle messages when building your own data pipelines outside of these courses.

Amazon Managed Streaming for Apache Kafka (MSK)

- Makes it easier to run Kafka workloads on AWS because the underlying infrastructure is managed for you

## From a Database

AWS Database migration Service (DMS)

- Can migrate and replicate data from a source to a target in an automated way

AWS Glue

- Offers features that support data integration processes

Ingestion

## From a Streaming Source

Amazon Kinesis Data Streams

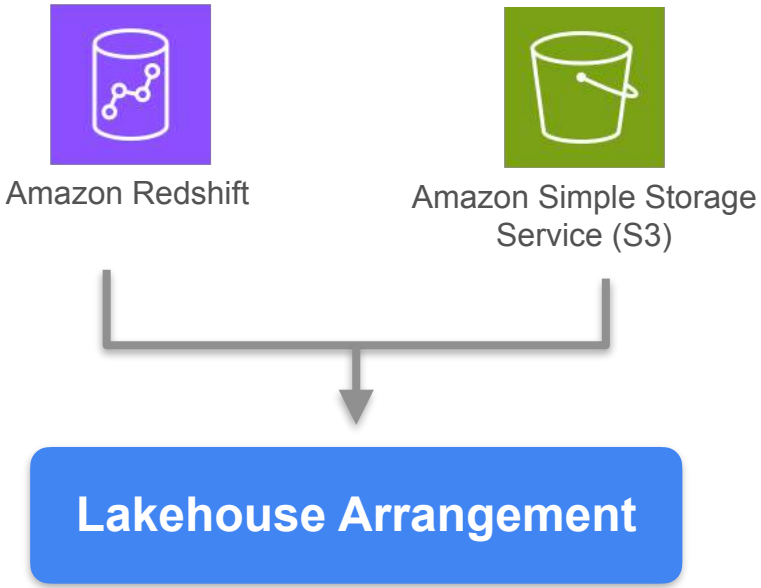
Amazon Data Firehose

Amazon SQS

Amazon MSK

# Traditional Data Warehouse

Storage




Access structured data in your data warehouse and unstructured data in an object storage data lake.

# Data Processing Tools

Transformation

AWS Glue



## Business Intelligence or Analytics

Amazon Athena

Amazon Redshift

- For querying structured and unstructured data

Amazon QuickSight



- Dashboarding tools

## AI or Machine Learning

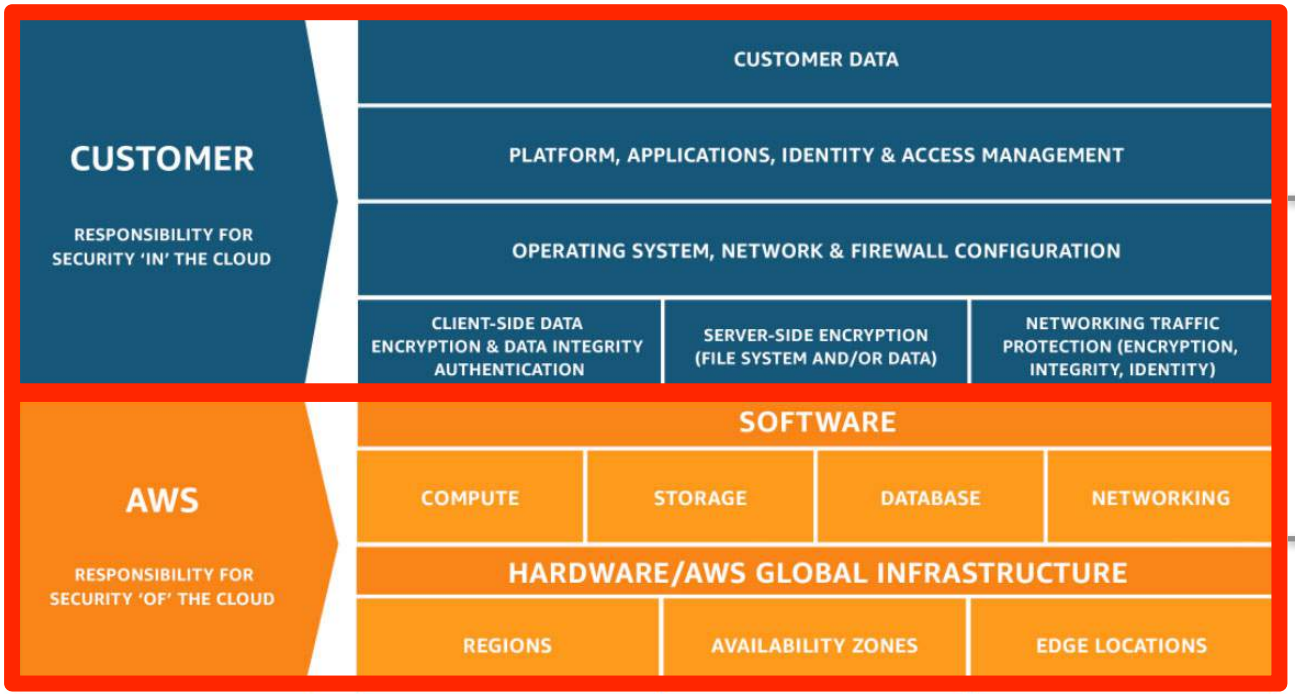
- Serve batch data for model training, and work with some vector database

DeepLearning.AI

# Practical Examples on AWS

## The Undercurrents on AWS

# Security



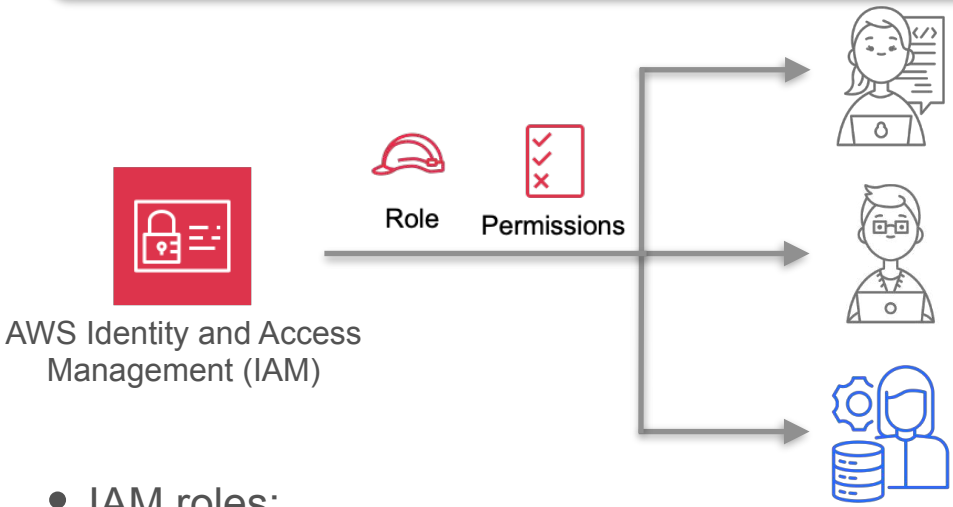

Set access permissions

Amazon Simple Storage Service (Amazon S3)

**Shared Responsibility Model**

# Security

## Identity and Access Management (IAM)




Amazon Virtual Private Cloud (VPC)

Security Groups

Instance level firewalls

- IAM roles:
  - Give users/applications access to temporary credentials
  - Provide appropriate AWS API permissions to various tools or data storage areas

# Data Management

AWS Glue

AWS Glue Crawler

AWS Glue Data Catalog

- Discover, create, and manage metadata for data stored in Amazon S3 or other storage and database systems

AWS Lake Formation

- Centrally manage and scale fine-grained data access permissions

# DataOps

Amazon CloudWatch

- Collects metrics and provides monitoring features for cloud resources, applications, and on-premises resources

Amazon CloudWatch Logs

- Store and analyze operational logs

Amazon Simple Notification Service (SNS)

- Sets up notifications between applications or via text/email that are triggered by events within your system

MC MONTE CARLO

Bigeye

# Orchestration



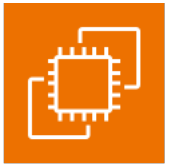

# Architecture

# Software Engineering

AWS Cloud9

IDE for development

hosted on



Amazon Elastic
Compute Cloud (EC2)

AWS CodeDeploy

Automate Code Deployment


DeepLearning.AI

# Lab Walkthrough

## Introduction to the Lab

# Lab Walkthrough Videos

**Video 1**

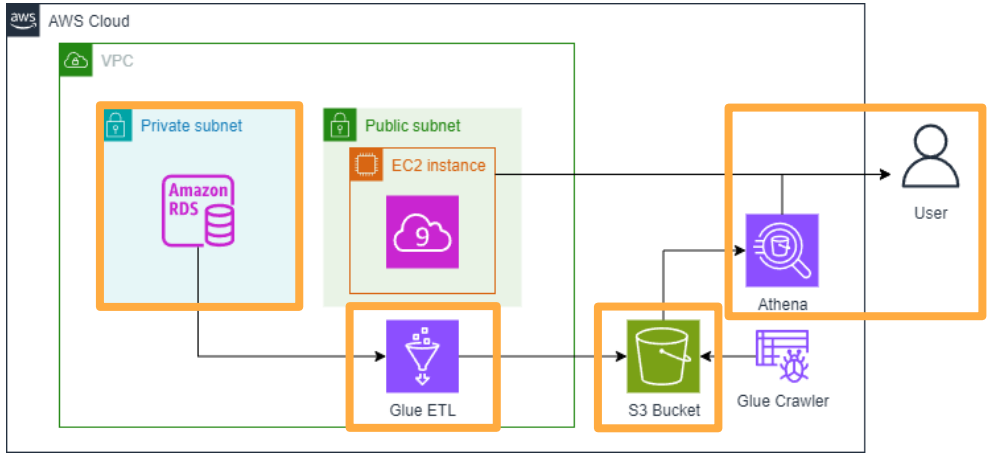
**Introduction to the lab**

**Video 2**

**Setting up the lab**

**Video 3**

**Preview of the lab content**

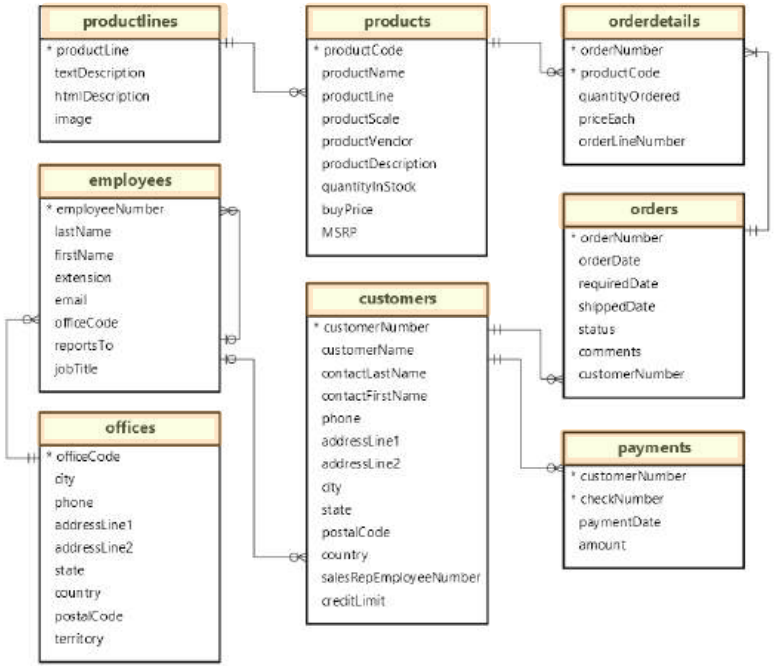




You will learn more about all the tools in the upcoming courses.

# Pipeline Scenario

**Historical purchases & Customers' Info**




Data Engineer

You work at a retailer for scale models of classic cars and other vehicles.

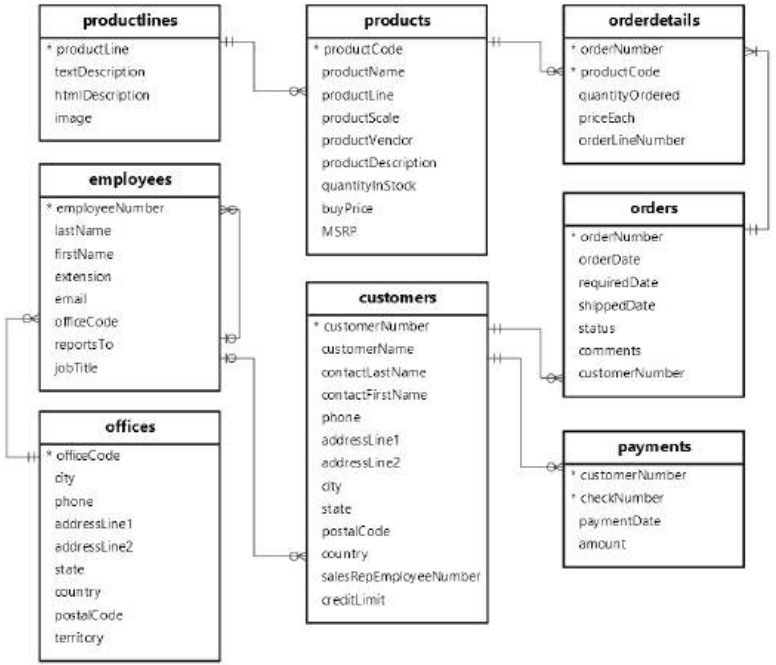
Transform and serve the data

Data Analyst

- Which product lines are more successful?
- How are the sales distributed across different countries?

# Pipeline Scenario

**Historical purchases & Customers' Info**




Data Engineer

Data Analyst

- Extract the data the analyst needs

**Data Modeling** (course 4)

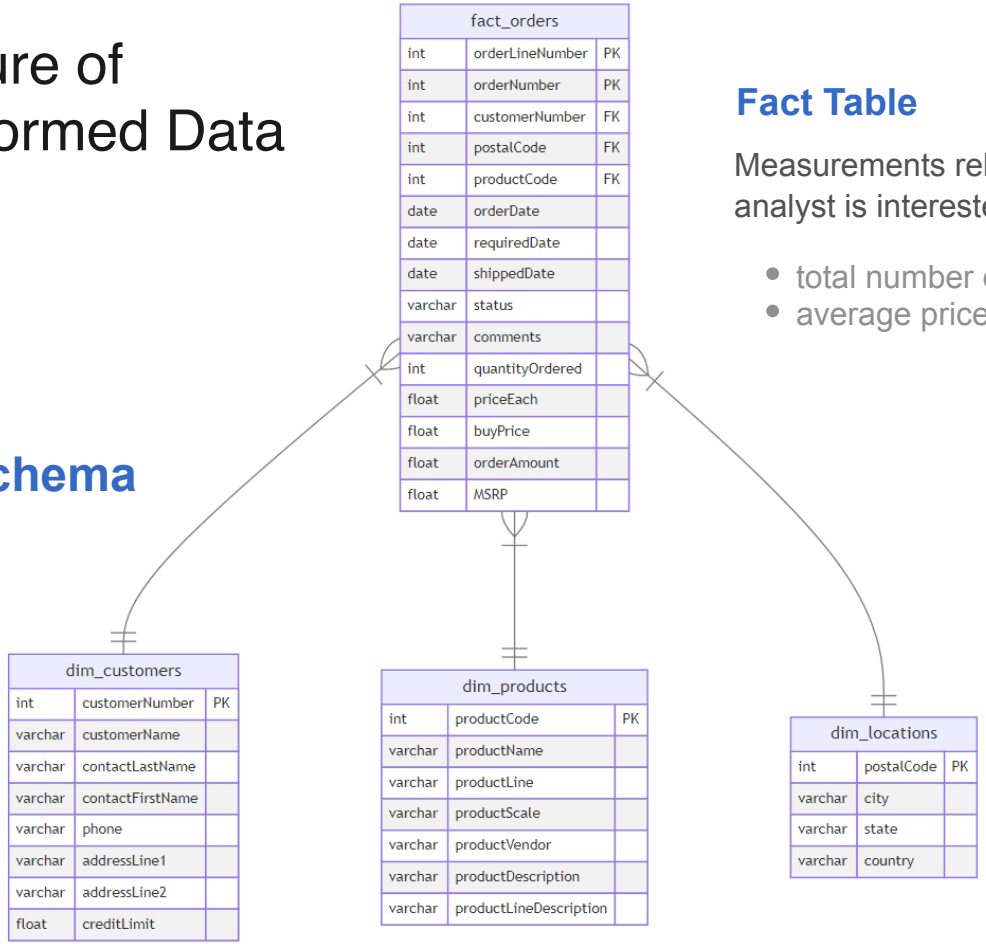
- Transform the data into a structure that is easier to understand and faster to query

Transformation script +
Structure of the data are given to you

- Store the data in a separate storage system

# Structure of Transformed Data

## Star schema




## Fact Table

Measurements related to a sales order that the data analyst is interested in aggregating

- total number of sales
- average price

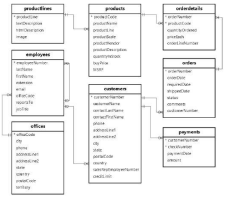
## Dimension Tables

More context (customer locations, order details)

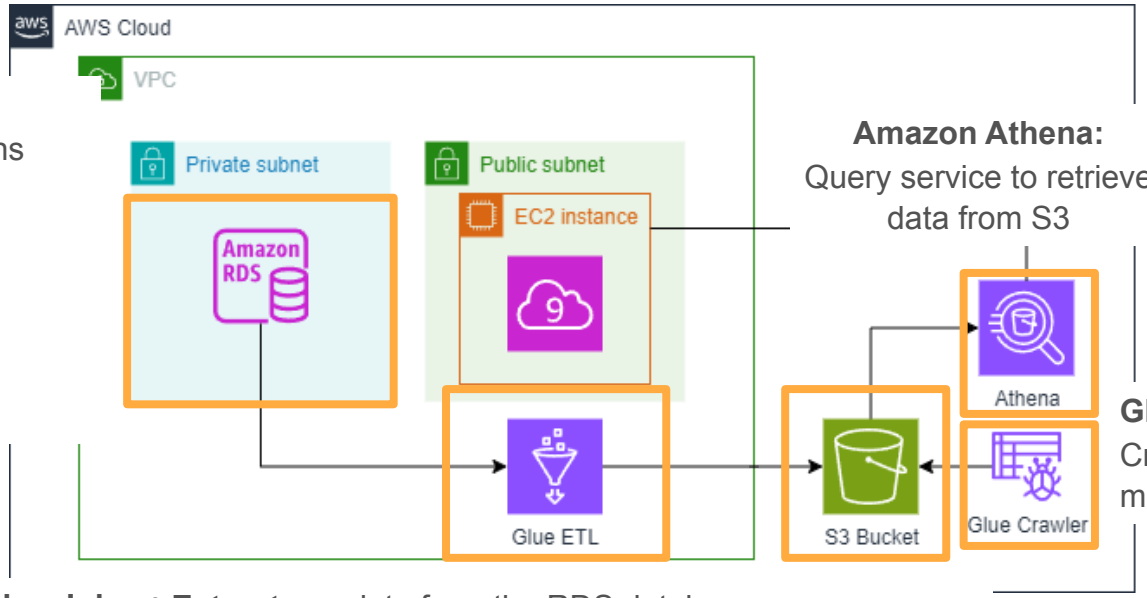
- total number of sales by country
- maximum quantities ordered for each product line

# Architectural Diagram

**RDS MySQL database:**
source system that contains



*Provided to you*



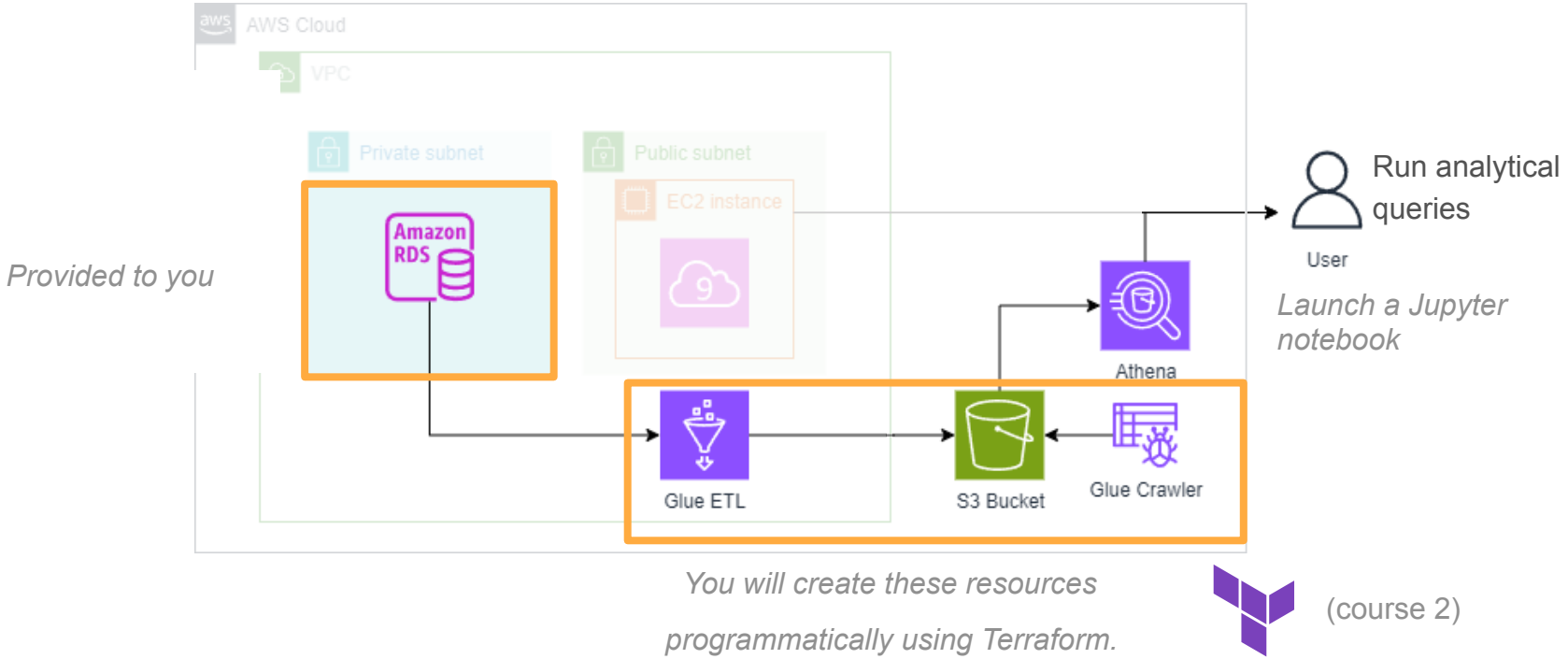

**Amazon Athena:**
Query service to retrieve data from S3

**Glue crawler:**
Crawl over S3 and write metadata to a data catalog

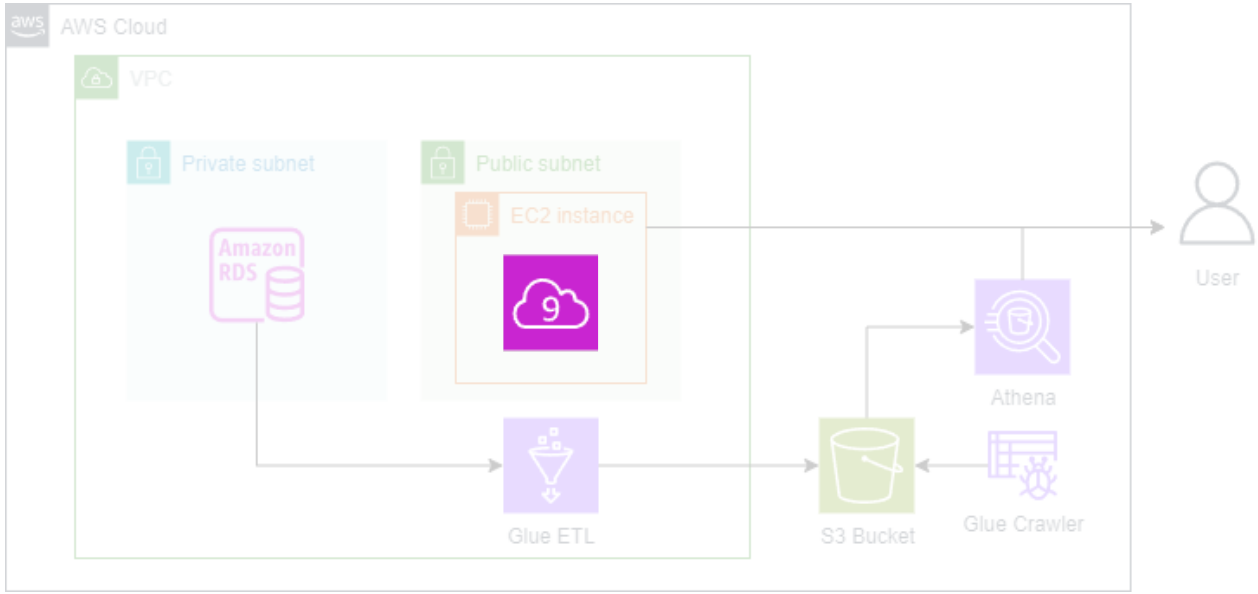
**Glue Job ETL**

- **Extract** raw data from the RDS database
- **Transform** the data by modeling it using the star schema
- **Load** the data into S3 bucket

# Architectural Diagram



# Architectural Diagram




**AWS Cloud9**

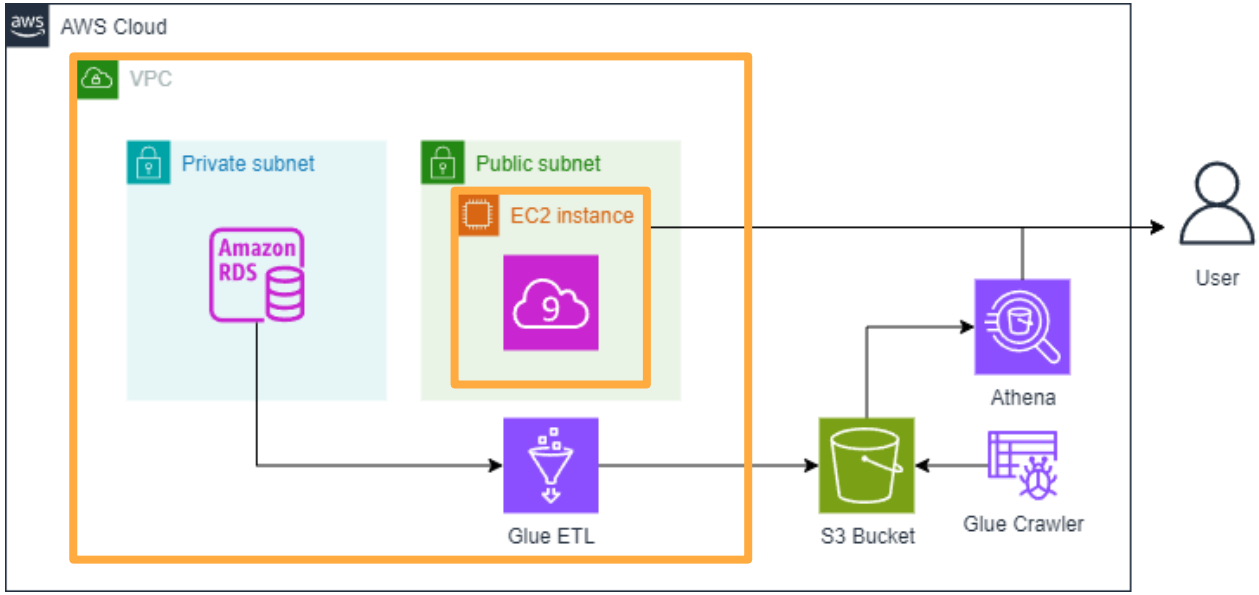
Integrated Development Environment (IDE)

# Architectural Diagram

**Video 2**

**Setting up the lab**

- AWS Cloud9
- Jupyter Notebook




**AWS Cloud9**

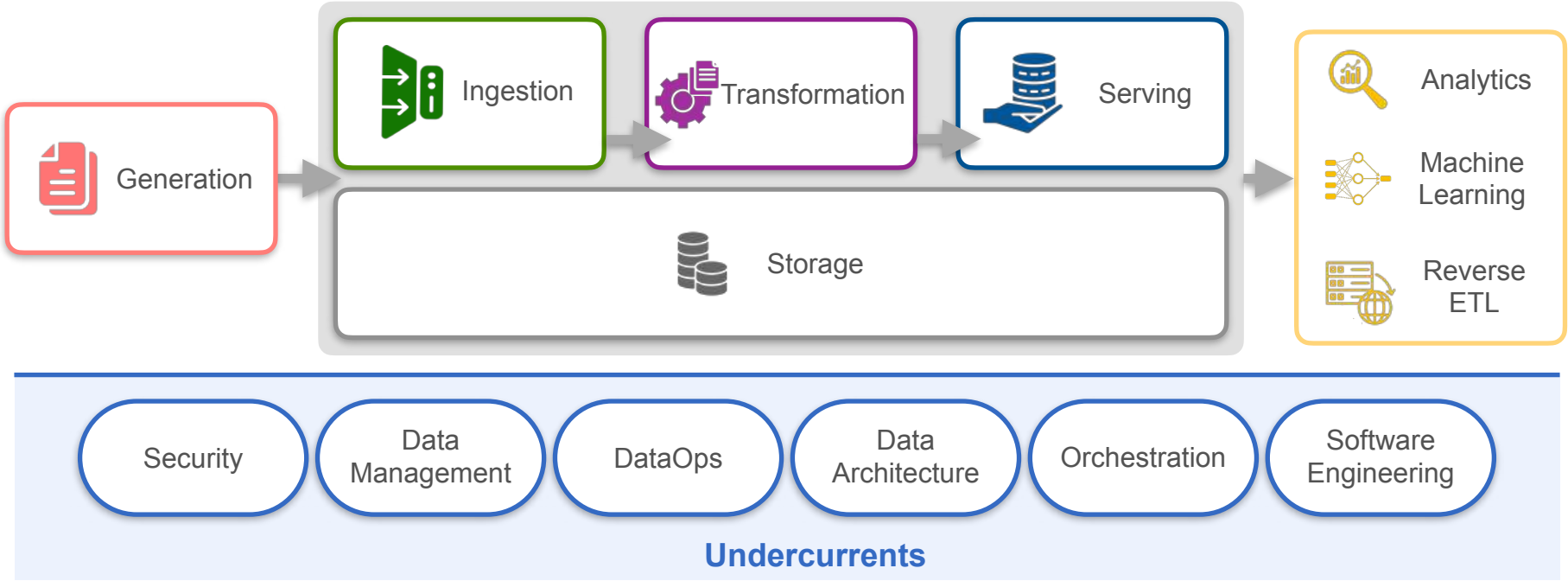
Integrated Development Environment (IDE)

DeepLearning.AI

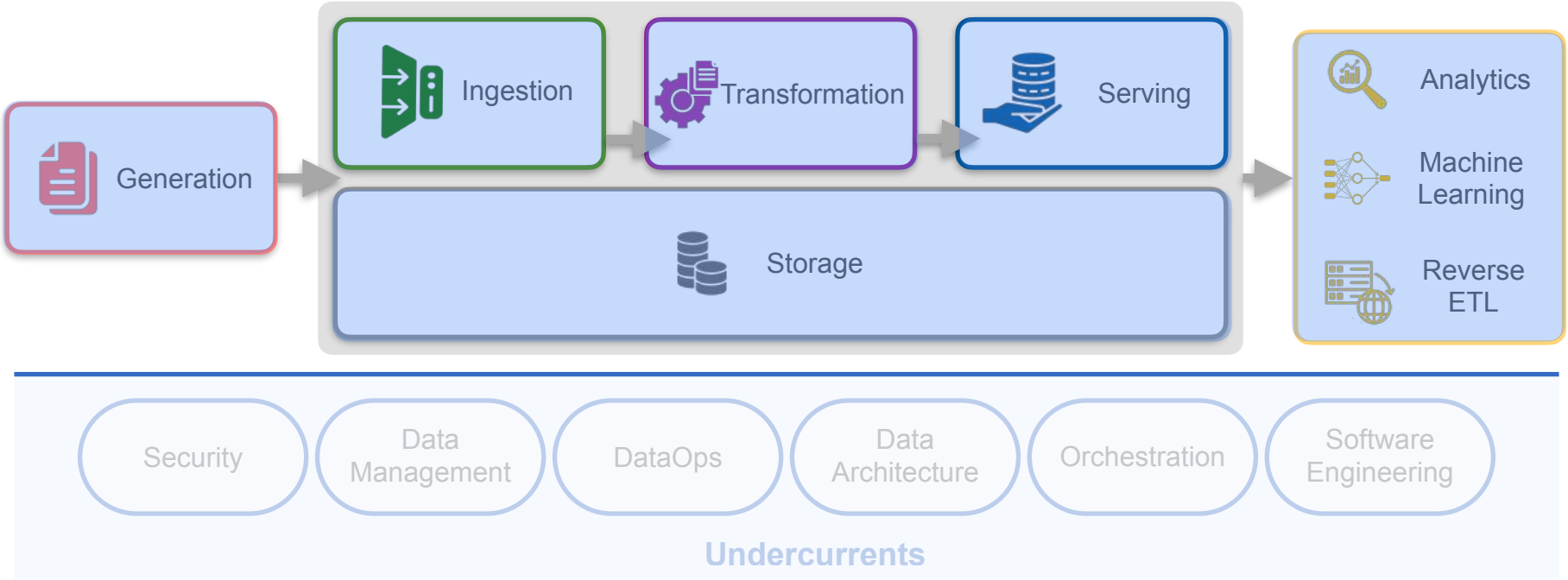
# The Data Engineering Lifecycle & Undercurrents

## Week 2 Summary

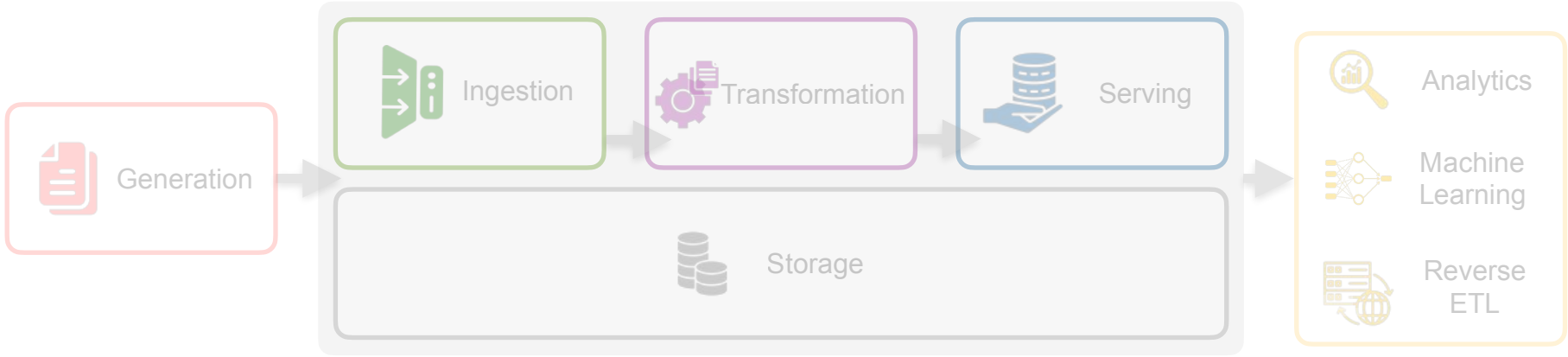
# The Data Engineering Lifecycle



# The Data Engineering Lifecycle



# The Data Engineering Lifecycle




Security | Data Management | DataOps | Data Architecture | Orchestration | Software Engineering

**Undercurrents**

# The Data Engineering Lifecycle