CSCI E-103

Data Engineering for Analytics to Solve Business Challenges

Data Lake

Lecture 5

Anindita Mahapatra & Eric Gieseke

Harvard Extension, Fall 2025

Agenda

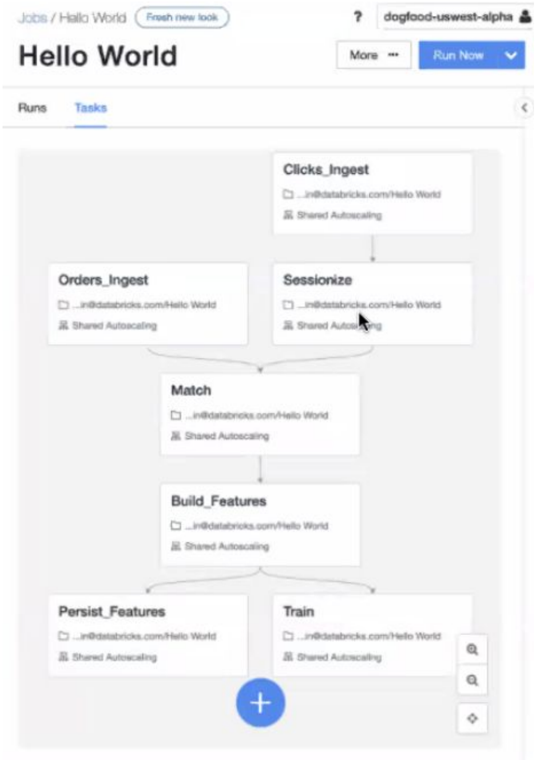
- Review previous class
- Hydrate your lake
- Moving from **data silos** to data consolidation
- How to prevent a Data Lake from turning into a **data swamp**
- Quality, Performance and Governance
- Data Lake Best Practices

- Lab
    - Creating a Data Lake
    - Use Case: Customer 360
    - Industry: Travel
        - Orchestrate Multiple data sources into a single Data Lake using a multi hop pattern
        - It is essential to create a consolidated view of the data & flatten it some to deliver fast queries

Review Past Lecture

DAG stands for	Directed Acyclic Graph
A Spark job consists of many ___ & ___	Stages & stages consist of Tasks
Monitor infrastructure usage using	Ganglia Metrics
__ exhibit 'Schema On Read'	Data Lakes
Compartmentalized data leads to	Data Silos
Ungoverned Data Lake leads to	Data Swamp
Data Warehouse + Data Lake =	Lakehouse
Partition: use a column whose cardinality is very high	False
Metadata brings better discoverability &	Governance
Unstructured data can reside in a	Data Lake

Job Orchestration



Notebooks can be put on schedule => Job

Notebook Workflows (%run, can pass parameters)

Job

- Schedule, Retry, Timeout, Alert
- Definition
  - code, cluster resources, permissions

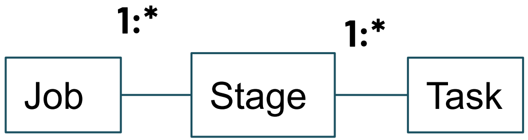
Jobs have Dependencies

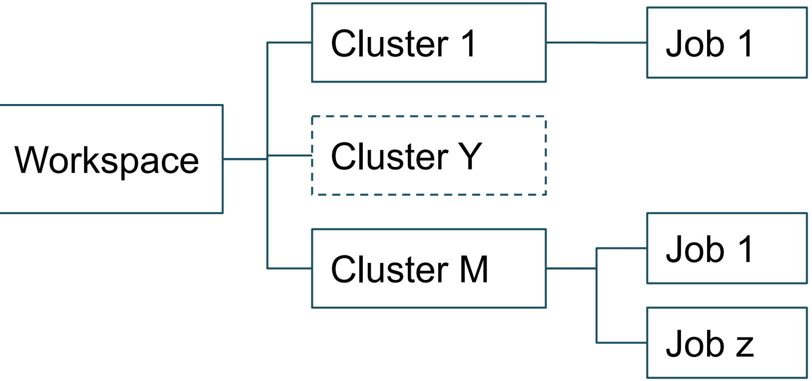
- DAG : Directed Acyclic Graphs
- Multi Task jobs
- Repair/Run

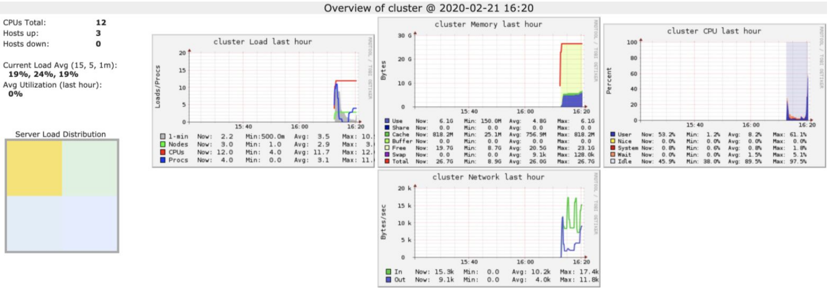
Create Visually or using APIs

Monitor Jobs

Spark







Different jobs have different needs - Choose appropriate node instance type

Optimized for memory/storage/compute/all purpose

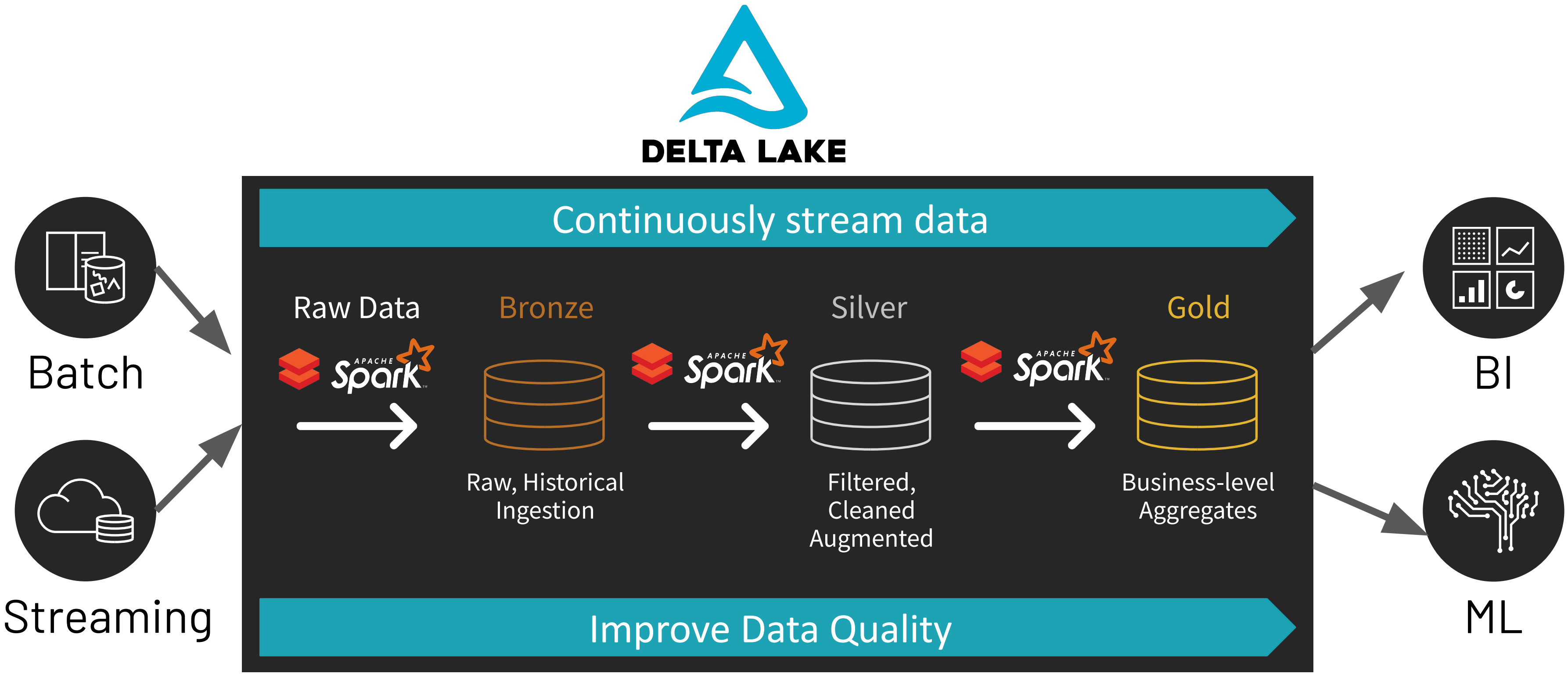
Ganglia Metrics

- CPU
- Network
- Memory

Job slowness caused by

- Spill
- Shuffle
- Skew/Stragglers
- Small Files

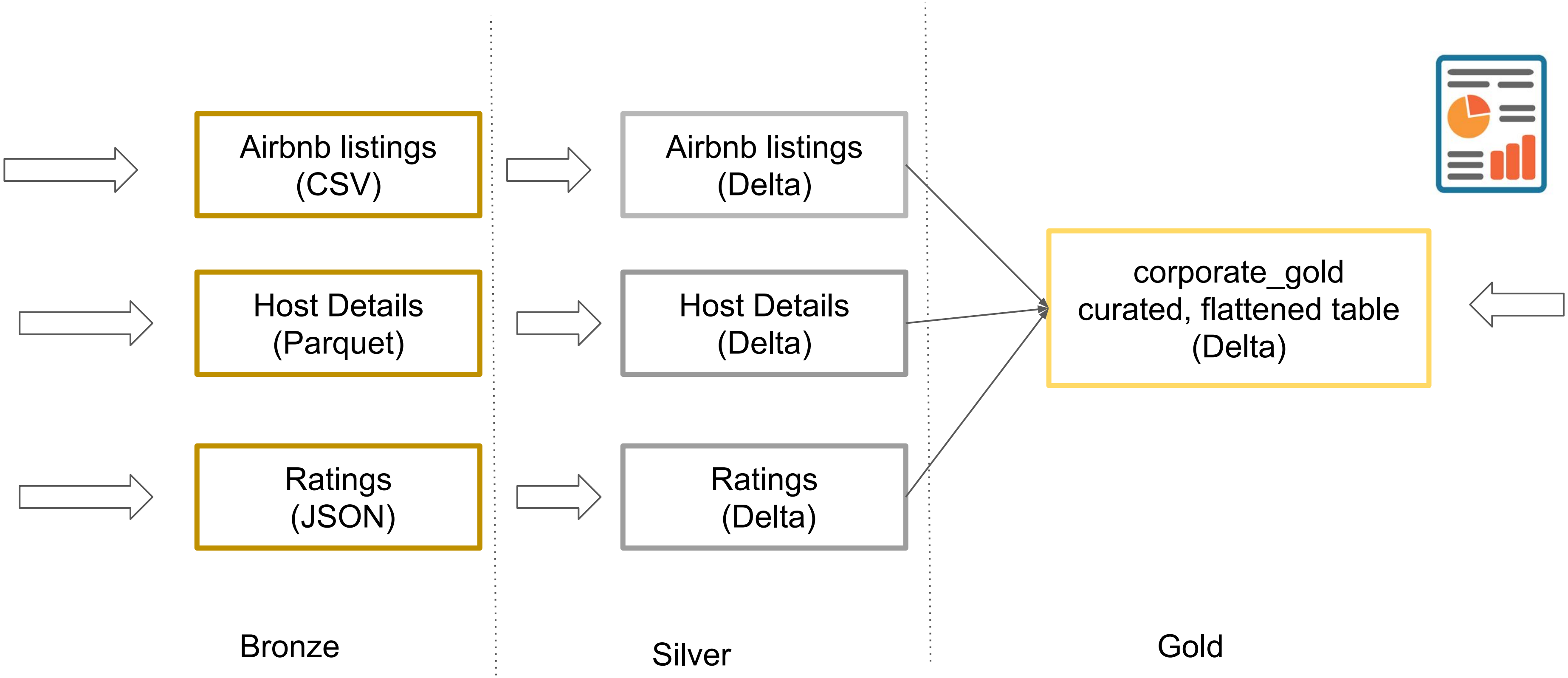
Continuously bring complete fresh data to teams



Data Consolidation Tools On Databricks

AutoLoader
Copy Into
Local File Upload (UI)
DLT - Delta Live Tables

Lab - Multi hop architecture - Bronze, Silver, Gold



What AutoLoader Does For You

Auto-ingest new files

Incrementally **process new files** as they land in cloud object storage without costly file listing procedures or state information handling

Scalable to billions of files

Auto Loader configures cloud notification and message queues to ensure that all files are sent to be ingested as they land and can scale to **billions of files** per directory

Infer & evolve Schemas

Identify schema on initialization and **detect changes** over time from structured sources and semi-structured data formats

Add hints for enforcement where some schema is known

Easily get started

Easy to deploy and **easy to use** with automated deployment of services making Auto Loader nearly set and forget

Auto Loader can ingest `JSON`, `CSV`, `PARQUET`, `AVRO`, `ORC`, `TEXT`, and `BINARYFILE` file formats.
2 modes - Dir/File Listing & Event Notification (cloud notification & message queues)
Rescue data column - never lose data again
Streaming or in Batch - easy to switch from batch to streaming and vice versa
Detect Schema changes - get notification when schema is changed from the source.

Doc Link Data Patterns

Getting started with Auto Loader

```
spark.read
  .format("json")
  .options(format_options)
  .schema(schema)
  .load("/path/to/table")
  ...
  .write
  .mode("append")
  .save("/path/to/table")
```

```
spark.readStream
  .format("cloudFiles")
  .option("cloudFiles.format", "json")
  .options(format_options)
  .schema(schema)
  .load("/path/to/table")
  ...
  .writeStream
  .option("checkpointLocation", ...)
    .start("/path/to/table")
```

Copy Into

- Load data from a file location into a Delta table.
- This is a re-triable and idempotent operation; files in the source location that have already been loaded are skipped.

```sql
CREATE TABLE IF NOT EXISTS my_pipe_data;

COPY INTO my_pipe_data
  FROM 's3a://my-bucket/pipeData'
  FILEFORMAT = CSV
  FORMAT_OPTIONS ('mergeSchema' = 'true',
                  'delimiter' = '|',
                  'header' = 'true')
  COPY_OPTIONS ('mergeSchema' = 'true');
```

	Spark Structured Streaming pipelines	DLT pipelines
Run on the Databricks Data Intelligence Platform	✓	✓
Powered by Spark Structured Streaming engine	✓	✓
Unity Catalog integration	✓	✓
Orchestrate with Databricks Workflows	✓	✓
Ingest from dozens of sources — from cloud storage to message buses	✓	✓
Dataflow orchestration	Manual	Automated
Data quality checks and assurance	Manual	Automated
Error handling and failure recovery	Manual	Automated
CI/CD and version control	Manual	Automated
Compute autoscaling	Basic	Enhanced

Need for a Lakehouse

Deficiencies of Data Lakes & Warehouses

From Data Silos to Data Lakes

- Data silos limit the value of your data
- A Data Lake provides a single source of truth
  - in a single repository
  - saving time, effort, and cost
  - Usually 5 or more distinct sources contribute to single use case
    - to reach a data driven decision
  - ~80% of world's data is unstructured and hence unsuitable for warehouses
  - Usually configured on a cluster of inexpensive and scalable commodity hardware

Why are Data Silos a problem
All storage, no action!

- A data silo is an isolated source of data that is
  - Only accessible to a single Line of Business (LOB) or department.
  - It leads to inefficiencies, wasted resources and obstacles in the form of incomplete data profiles.

Structural

purpose built app and data-sharing isn't a requirement.

Political

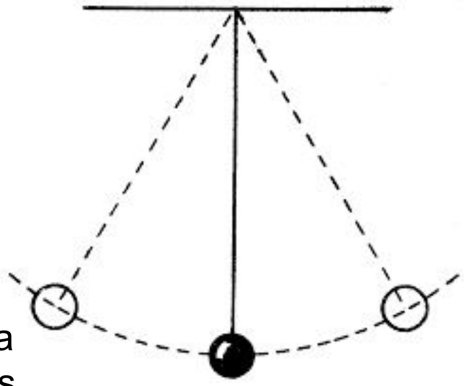
sense of proprietorship over a system's data so it isn't readily shared with others

Growth

technology that's ultimately incompatible with existing systems and data sets.

Vendor Lock-In

technology vendors don't give sufficient data access to its customers

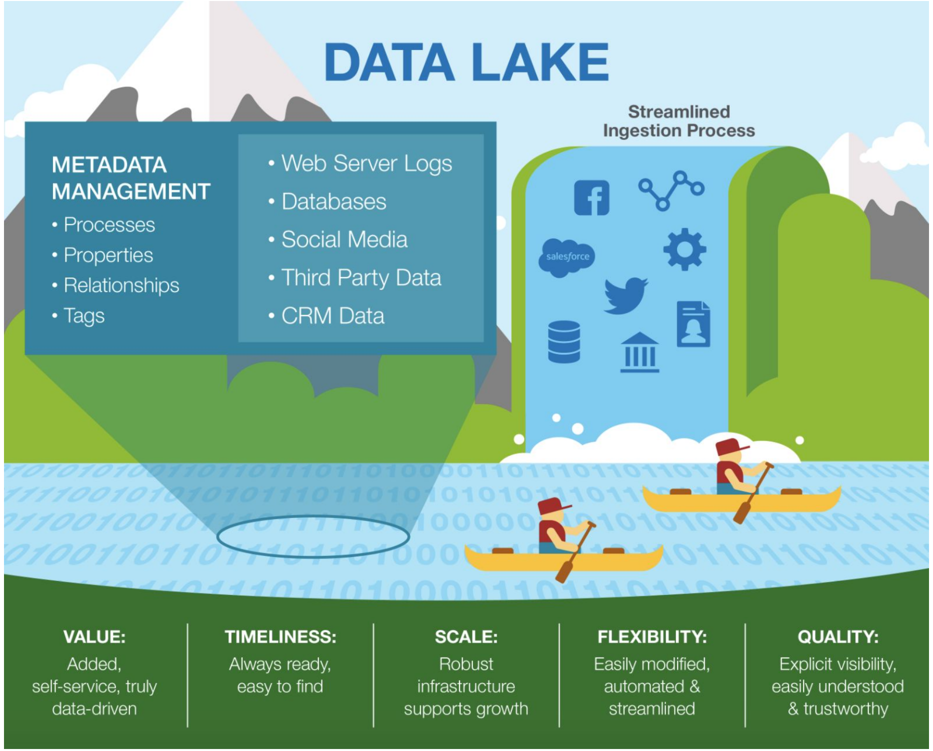


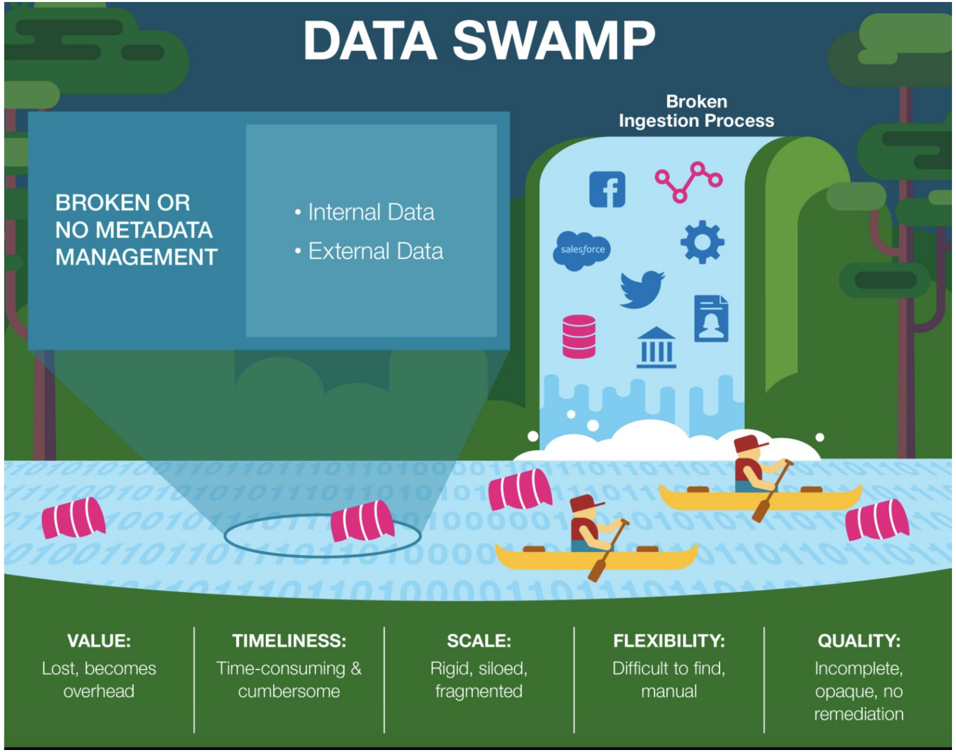
Data Silos to Data Swamps

Data Lakes help avoid Data Silos
However inadequate governance can turn them into Data Swamps

It is no longer a question of whether a data lake is needed, but it is about which solution to use and how to implement it.

Data Lake Vs Data Swamp





Delta Lake helps address inherent challenges of traditional Data Lakes

Challenges of Traditional Data Lakes

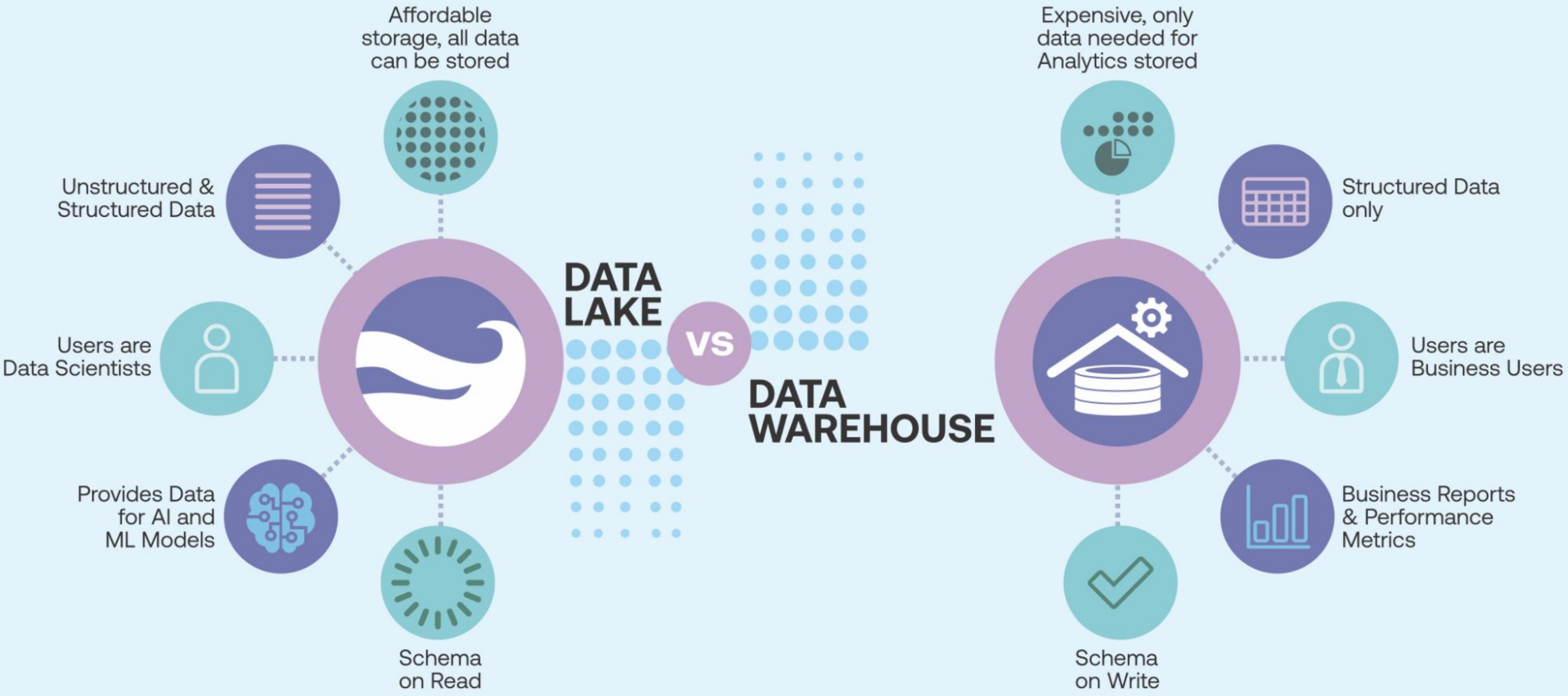
1. Hard to append data
2. Modification of existing data difficult
3. Jobs failing mid way
4. Real-time operations hard
5. Costly to keep historical data versions
6. Difficult to handle large metadata
7. "Too many files" problems
8. Poor performance
9. Data quality issues

Delta Lake

- ACID tx support
- All transactions are recorded and you can go back in time to review previous versions of the data (i.e. *time travel*)
- Spark is built for handling large amounts of data
- All Delta Lake metadata stored in open Parquet format
- Portions of it cached and optimized for fast access
- Data and it's metadata always co-exist. No need to keep catalog<>data in sync
- Schema validation and evolution
- Data skipping: prune files based on statistics on numericals
- Z-ordering: layout to optimize multiple columns

Data Warehouse Vs Data Lake

Schema	Schema on write, use with BI/analytics, ETL: define use cases ahead of time Simpler User Access	Schema on read, ELT Flexible configuration
Volume	Large	Magnitudes larger, Better/Cheaper scalability
Velocity	Batch	Batch + Streaming
Variety	Structured, processed data	Structured, Semi, unstructured, raw
Data Format	Proprietary, ready for BI Reporting, Silo	Open + Proprietary - Single Repository
Cost	Cost Scales with volume	Low Cost, High Volumes

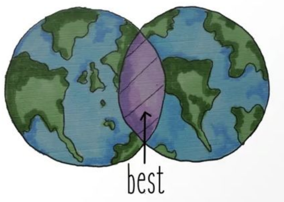

Affordable storage, all data can be stored
Unstructured & Structured Data
Users are Data Scientists
Provides Data for AI and ML Models
Schema on Read
DATA LAKE
VS
DATA WAREHOUSE
Expensive, only data needed for Analytics stored
Structured Data only
Users are Business Users
Business Reports & Performance Metrics
Schema on Write

When to use what
Hybrid is also an option ...

Data Lake	Data Warehouse
● Data that needs to be aggregated is <u>not known in advance</u> ● There are <u>huge datasets</u> that may be growing, and storage costs may be an issue ● Data is collected from many sources and has <u>different formats</u> that do not adhere to a tabular or relational model ● Complete, raw datasets are needed for objectives like data exploration, predictive analytics, and machine learning ● How data elements relate with each other is not yet known	● Organizations know which data needs to be stored and are so familiar with it, that they can delete redundant data or make copies easily ● Data formats do not change and are not anticipated to change in the future. ● The purpose of the data is generation of typical <u>business reports</u> and fast querying is needed ● Data is precise and carefully selected ● Data needs to be compliant with regulatory or business requirements and needs special handling for auditing or security purposes

If used in hybrid mode, Data Lake should come before the Data Warehouse

What if you did not have to choose ...



	Data lake	Data lakehouse	Data warehouse
Types of data	All types: Structured data, semi-structured data, unstructured (raw) data	All types: Structured data, semi-structured data, unstructured (raw) data	Structured data only
Cost	$	$	$$$
Format	Open format	Open format	Closed, proprietary format
Scalability	Scales to hold any amount of data at low cost, regardless of type	Scales to hold any amount of data at low cost, regardless of type	Scaling up becomes exponentially more expensive due to vendor costs
Intended users	Limited: Data scientists	Unified: Data analysts, data scientists, machine learning engineers	Limited: Data analysts
Reliability	Low quality, data swamp	High quality, reliable data	High quality, reliable data
Ease of use	Difficult: Exploring large amounts of raw data can be difficult without tools to organize and catalog the data	Simple: Provides simplicity and structure of a data warehouse with the broader use cases of a data lake	Simple: Structure of a data warehouse enables users to quickly and easily access data for reporting and analytics
Performance	Poor	High	High

Innovations lay the foundation for Lakehouse architecture

Delta Lake Innovations		Positive Business Impact
Reliability	**ACID Transactions**	Quality data accelerates innovation
		Lower TCO with a simple architecture
Performance	**Indexing**	Automation increases productivity
Governance	**Table ACLs**	Reduces security risk
Quality	**Expectations**	

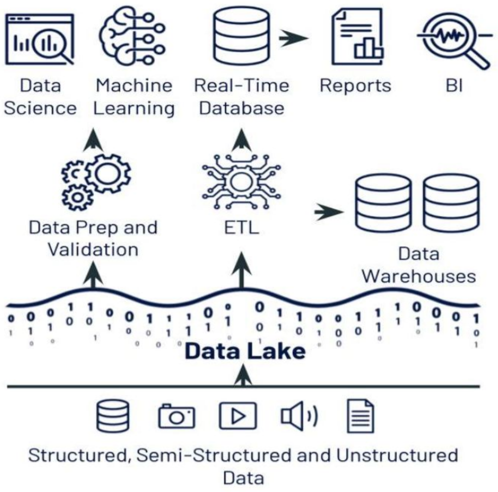
Lakehouse

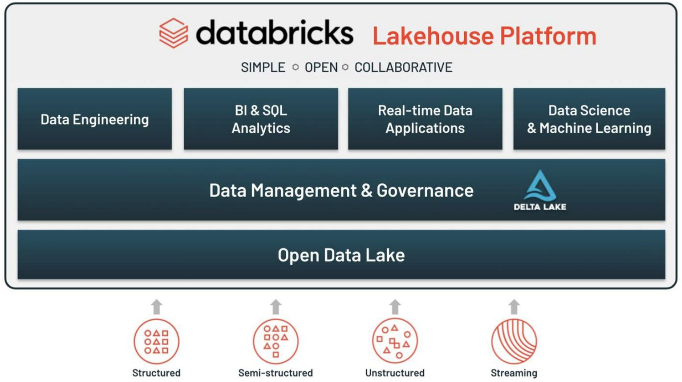
[What is a Lakehouse](#)

Data storage paradigm and architecture that combines characteristics of Data Warehouse (structure and governance inherent to data warehouses) & Data Lake(flexibility, cost effective)

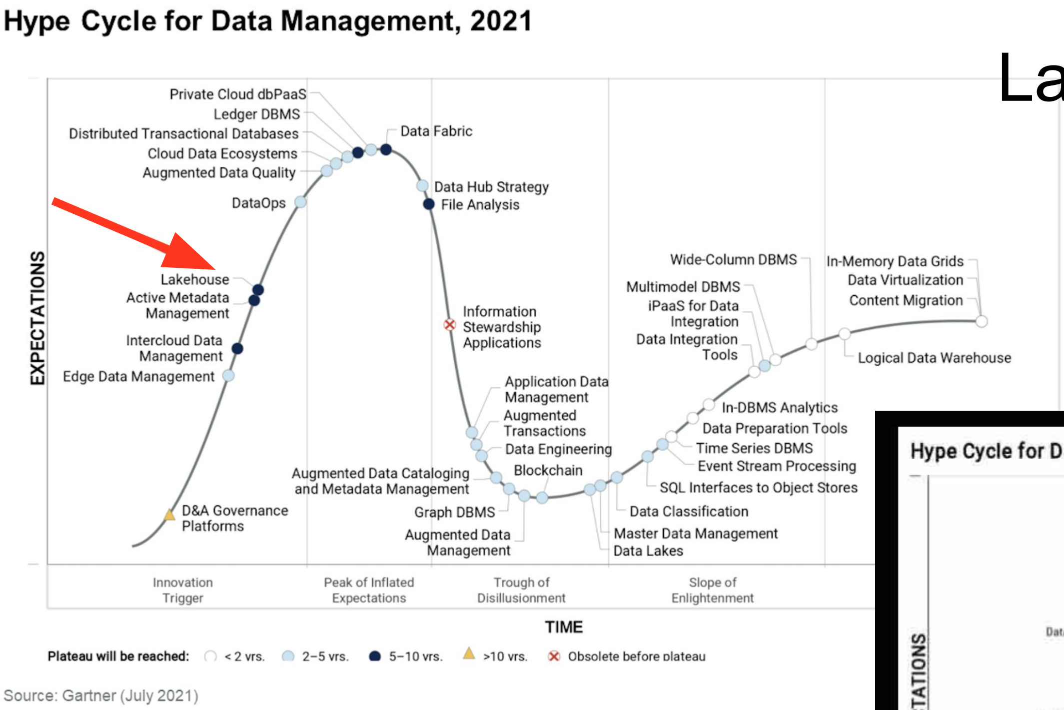
Raw Data ----> Curated Data Lake

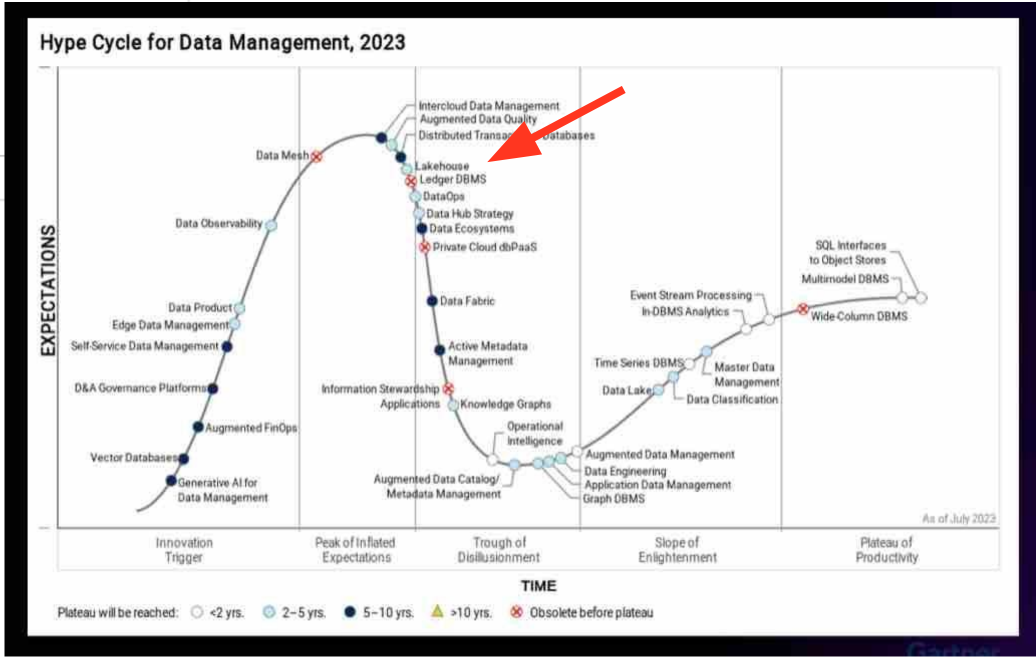
- Pros
  - Directly connect to BI tools
    - connecting the query engine directly to your Data Lake.
  - Reduced Data & Governance Redundancy
    - eliminate the operational overhead of managing Data governance on multiple tools.
  - Reduce costs
- Caution
  - What to do with existing Data warehouses?
    - Data Migration
  - Are the tools mature and unified?
  - Are we moving towards a monolith again?





Lakehouse Maturity

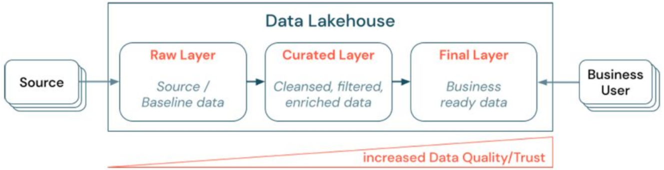




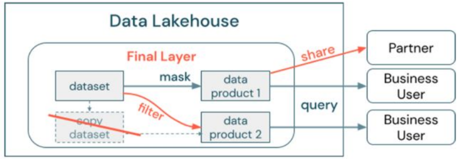
Lakehouse Guiding Principles

Link

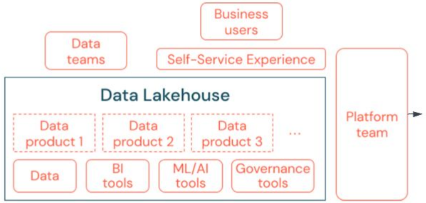
1: Curate Data and Offer Trusted Data-as-Products



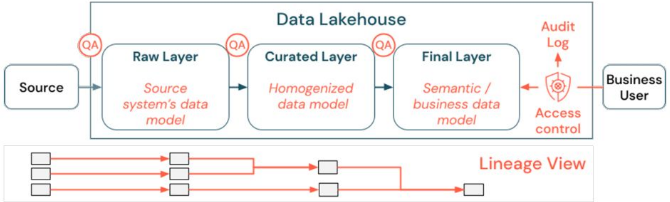
2: Remove Data Silos and Minimize Data Movement



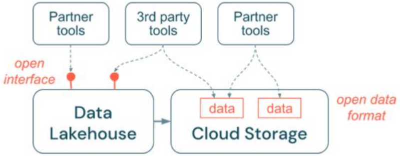
3: Democratize Value Creation through Self-Service Experience



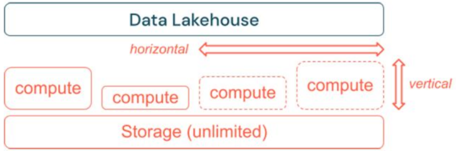
4: Adopt an Organization-wide Data Governance Strategy (quality, catalog, access)



5: Encourage the Use of Open Interfaces and Open Formats



6: Build to Scale and Optimize for Performance & Cost



Best Practices

- Use the data lake as a landing zone for all of your data
- Mask data containing private information before it enters your data lake
- Secure your data lake with role- and view-based access controls
- Build reliability and performance into your data lake by using Delta Lake
- Catalog the data in your data lake (Eg. Unity Catalog)

Why Data Mesh?

Decentralization and distribution of responsibility

Domain Ownership	Data as a Product	Self-Serve Platform	Federated Governance
Decentralized & Autonomous Teams Responsibility owned by those closest to data Map to business org	Data Product Owner Serve Consumers as Customers Measure Success of Products	Distributed and Scalable Easily create & terminate resources on-demand Compute & data locality	Decentralized Domain Self-Sovereignty Interoperability Global Standardization

Databricks capabilities for a Data Mesh

Addressing the 4 key pillars* with Databricks Lakehouse

#1 Domain ownership

Distributed architecture where domain teams, the **data producers**, can take responsibility for their data and its outcomes

- Open and flexible architecture enables workspace/catalog per domain
- Distributed ownership of data assets and pipelines

#2 Data as a product

Applying **product thinking** to analytical data, including providing quality data to **data consumers** beyond the source domain

- Open standards and formats for FAIR data
- ACID guarantees, versions, and audits with **Delta Lake**
- Fresh, high-quality data with **Delta Live Tables**

#3 Self-service infrastructure platform

Domain-agnostic approach to building, executing, and maintaining interoperable data products through common tools

- Unified platform serving all analytics workloads
- Managed orchestration with **Databricks Workflows**
- Auto-scaling, serverless
- Infrastructure as Code (**Terraform**)

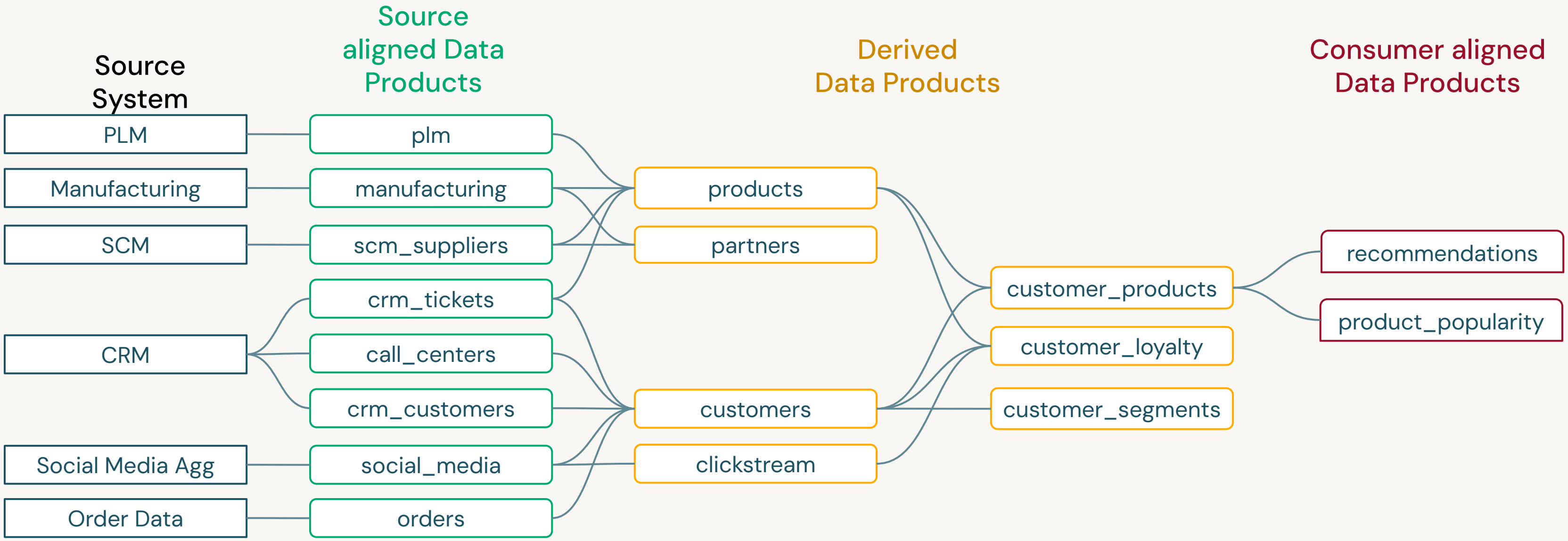
#4 Federated computational governance

Creating a data ecosystem that adheres to organisational rules and industry regulations through standardisation

- Discovery, access and lineage with **Unity Catalog**
- Global policy templates for access to data and compute resources

* Source: http://datamesh-architecture.com

Data product hierarchy

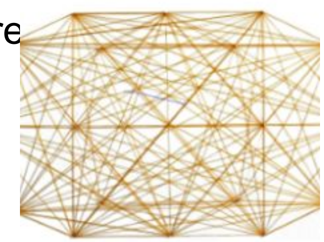


- Represent the relevant data as it is in the operational system with minimal transformation
- Cleansed and transformed to ensure quality
- First step to creating more valuable data products.

- Created by processing and transforming source-aligned data products or other derived data products.
- Satisfy user needs e.g., for decision-making, and automated decision-making.
- Can be reused in other derived data products.

- Consumer-aligned data products are specifically built for end users, e.g. dashboards, reports

Data Fabric (spanning hybrid multi cloud environments, on-prem, edge, everywhere



- Gartner defines data fabric as a design concept that serves as **an integrated layer (fabric) of data and connecting processes** leveraging both human and machine capabilities to access data &place or support its consolidation where appropriate.
- Monitor and manage your data and applications, regardless of where they live.
  - unlocks the best of cloud, core, and edge.
  - a rich set of data management capabilities that ensure consistency across your integrated environments.
- The data mesh is more about people and process than architecture, while a data fabric is an architectural approach that tackles the complexity of data and metadata
- The data fabric is about collecting data and making it available via purpose built APIs
- A single environment consisting of a unified architecture with services and technologies running on it
- A data fabric includes building blocks such as data pipeline, data access, data lake, data store, data policy, ingestion framework, and data visualization.
- A data fabric is technology-centric, while a data mesh focuses on organizational change.

Real State of the Industry

- Organizations struggle with achieving successful big data and artificial intelligence (AI) projects
- Common challenges of Data Lakes related to enterprise-scale data analytics
    - Very complex setup and management
    - Simplified ingestion but nightmarish consumption due to data swamps that are difficult to navigate/manage
    - Lack of BI support (poor performance/response times around e2e latency and concurrency)
- To help organizations draw value from their data lake investments,
    - Databricks created Delta Lake
    - designed to bring the governance, reliability, performance and structure of data warehouses into data lakes
- Reading link

Data Lake Architecture Main Requirements

Every vendor claims to have a Data Lake - what should you check for in a Data Lake?

- Data Catalog
  - Information about the data that exists within your data lake
    - Ex. Metadata, Schema, connectors, description, tags to help discovery
  - For folks within and outside the organization to understand the context of the data
  - Deploy tools that will automatically add entries to the data catalog by scanning each new data asset as it is added to the lake.
- Data Governance
  - refers to the processes, standards, and metrics to manage & ensure that data can fulfill its intended purpose
- Data Quality
  - What data is acceptable for insight generation and meeting SLAs
  - How bad data should be fixed
- Data Security
  - Access Controls
  - Encryption to help prevent unauthorized access to data

ToDos

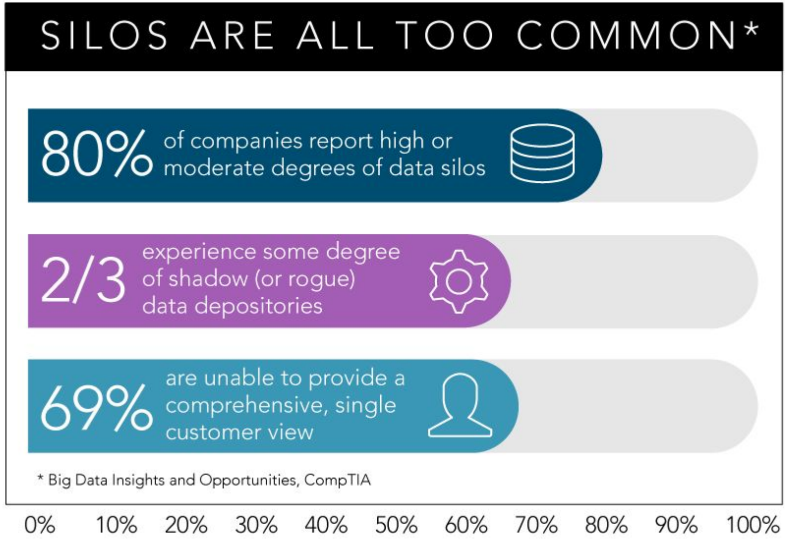
Read Ch-4 - 6 (Data Operations)

Assignment - 2 submission coming along

Assignment - 3 will be on streaming

APPENDIX

5 Steps to Tear silos Down
To Connect Data, and Activate Results



1. Employ a Data Evangelist
2. Institute Common Data Standards
3. Make It Universal
4. Embrace Advanced Analytics
5. Work With Trusted, Third-Party Partners

It requires a combination of <u>technology</u>, <u>people</u>, and <u>process</u> to defines your competitiveness and the ability to transform your business.

Planning a Data Lake

- Ingestion needs
  - Push Vs Pull
  - Streaming Vs Batch
- Security around data access
- Data retention and archival policies
- Encryption requirements
- Governance
- Data quality
- Master data management
- Validity checks necessary
- Metadata management
- Organization of data for optimal data retrieval
- Scheduling and job management
- Logging and auditing
- Enrichment, standardization, cleansing, and curation needs
- Technology choices comprising the overall data lake architecture
  - HDFS, Hadoop components, NoSQL DBs, relational DBs, etc.
- Modular approach to the overall design

Guiding Principle of Mesh architectures

…to **achieve the promise of scale**, while delivering **quality and integrity guarantees** needed to **make data useable**

1. Domain-oriented decentralised data ownership and architecture
2. Data as a product
3. Self-serve data infrastructure as a platform
4. Federated computational governance.

- Let those who know the data best be responsible for its value
- Reduce reliance on central teams whilst avoiding silos – balance local autonomy with global interoperability

Data Mesh is an organizational process – not a purchasable product that you buy
However, the right platform capabilities will ease the implementation of a data mesh

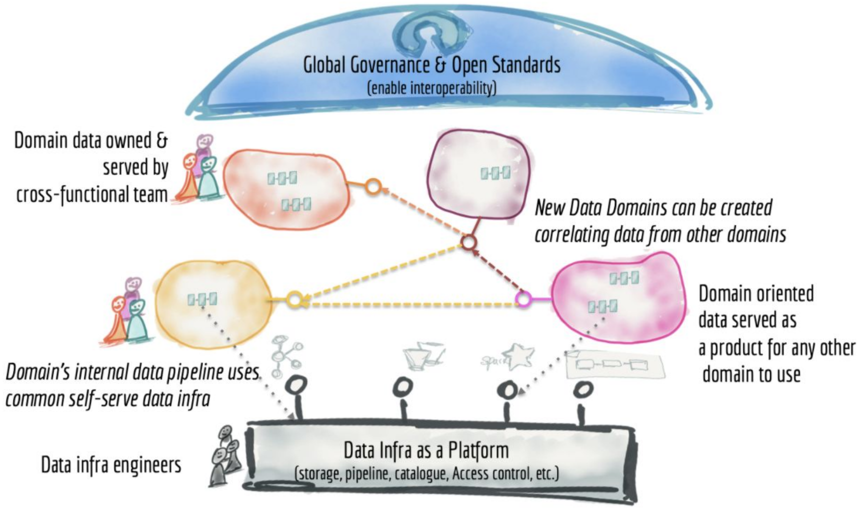
Need for a platform with the **technical enablers** for teams to produce and consume data in a <u>decentralized but governed way</u>

- Lakehouse is a polyglot technology that also works outside a Data Mesh concept
- Lakehouse applies at all scales (startups to large orgs)

Data Mesh

Reference

Video



Convergence of

- *Distributed Domain Driven Architecture*,
- *Self-serve Platform Design*, and
- *Product Thinking* with *Data*.

Distributed data products oriented around domains and owned by *independent cross-functional teams* who have embedded data engineers and data product owners, using common *data infrastructure* as a platform to host, prep and serve their data assets.

A Data Mesh is more conceptually similar to the hydrology (the movement, distribution and management) of resources in a widely distributed ecosystem.

- Centralized to decentralized ownership
- Pipelines as first-class concern to domain data
- Data as a by-product to data as a product
- A siloed data engineering team to cross-functional domain-data teams
- A centralized data lake/warehouse to an ecosystem of data products

Data Lakes/Warehouses are nodes of the mesh

Treat *domain data product* as a first class concern, and data lake/warehouse tooling and pipeline as a second class concern