Data Lake in a Day
aka.ms/dliad

Dave Lusty
Nick Hurt

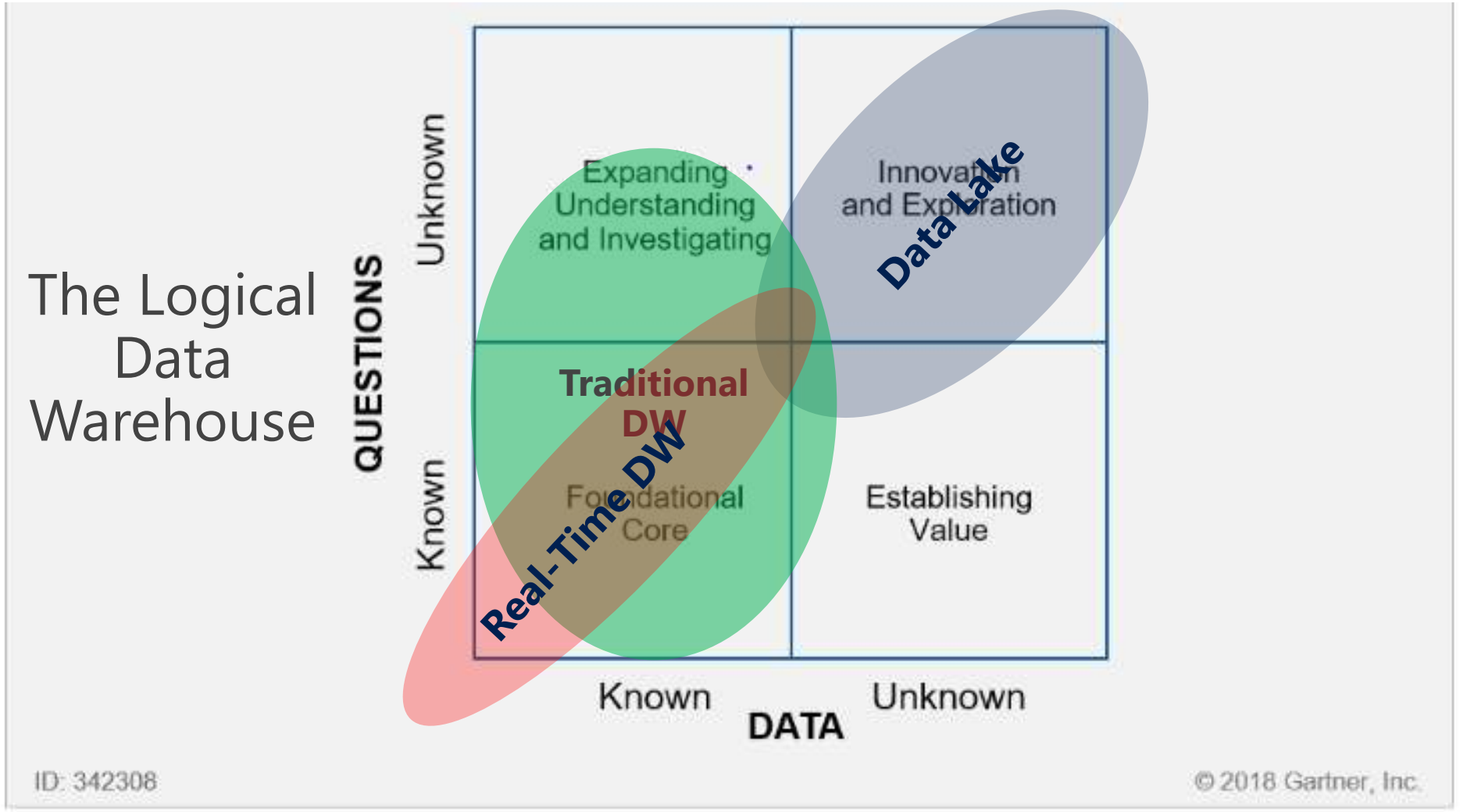
What is a data lake?

- A storage repository that holds a large amount of data in its native, raw format.
- Data lake stores are optimized for scaling to terabytes and petabytes of data.
- The data typically comes from multiple heterogeneous sources, and may be structured, semi-structured, or unstructured.
- The idea with a data lake is to store everything in its original, untransformed state.

Benefits of a data lake

- Business benefits
  - Facilitate early exploration and innovation
  - Single source of truth – all data in a centralized/standardised repository
  - Fine and coarse grain access control using ADLS
  - Store everything at lowest cost, high redundancy & availability
- Technical Benefits
  - High fidelity – preserve data in raw format
  - High availability and throughput - ingest at high velocity
  - Schema on read – reduces up-front effort
  - Decouples storage and compute for scalability and cost-efficiency

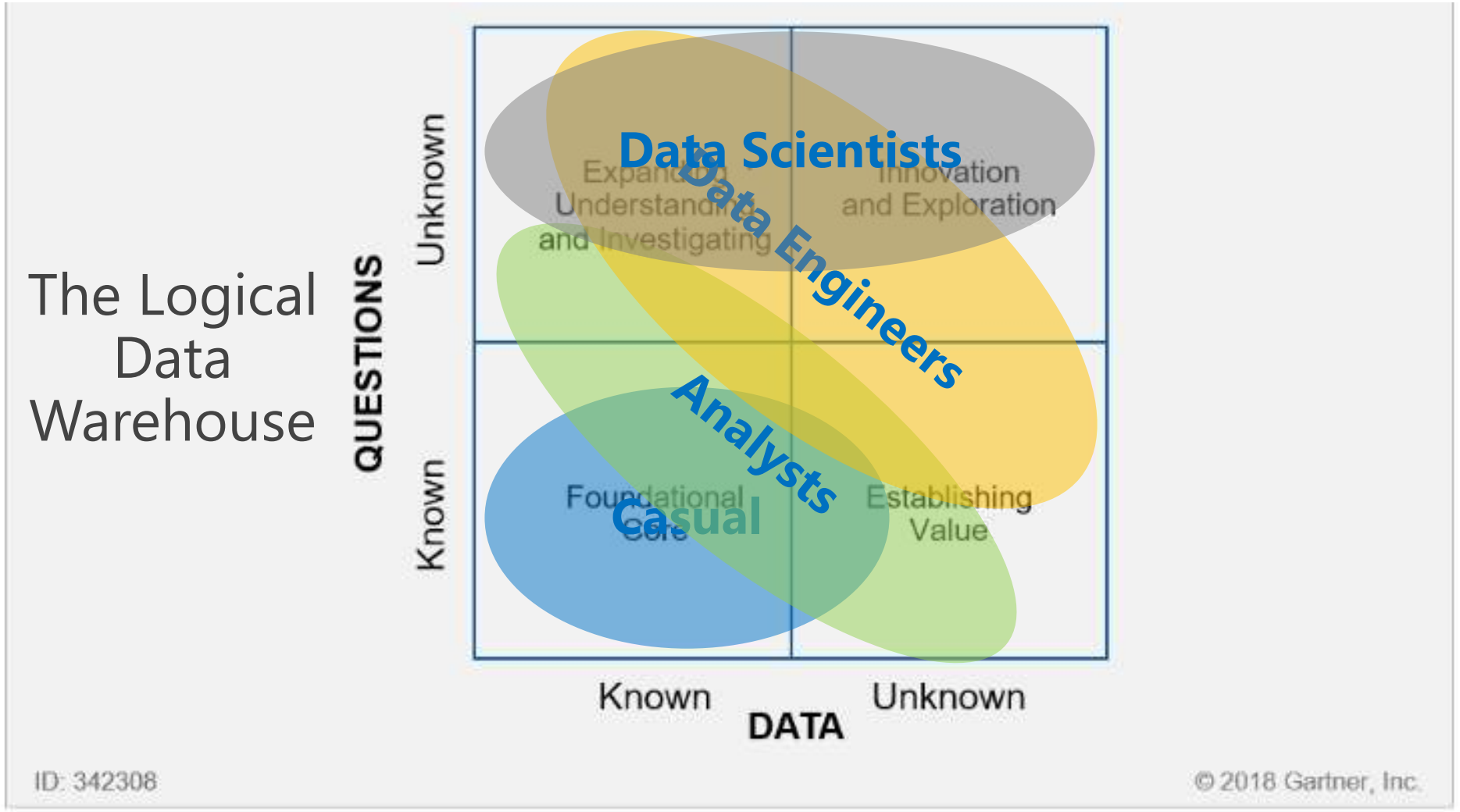
Gartner Data Management Infrastructure Model



Source: Gartner (March 2018)

Source: Gartner "The Practical Logical Data Warehouse: A Strategic Plan for a Modern Data Management Solution for Analytics", March 2018 (G00342308)

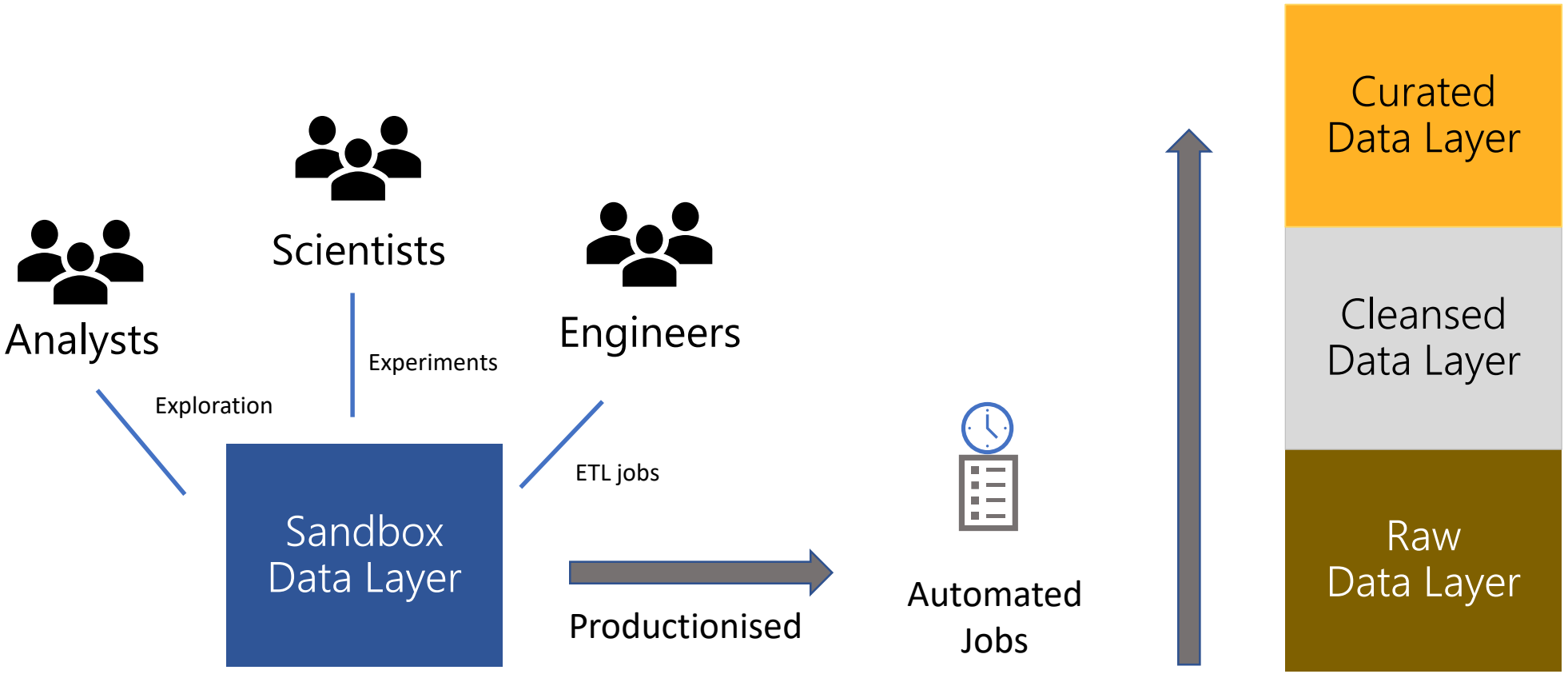
Gartner Data Management Infrastructure Model



Source: Gartner (March 2018)

Source: Gartner "The Practical Logical Data Warehouse: A Strategic Plan for a Modern Data Management Solution for Analytics", March 2018 (G00342308)

Data Lake Layers



Data Lake Zones & Characteristics

- Raw layer – the reservoir zone
  - Original format
  - Immutable
  - Consider lifecycle management and archive sub-layer
  - Permission by source system ingestion
- Cleansed layer – the filtration zone
  - Schema defined
  - Data is cleansed, validated, standardized, harmonized, enriched
  - AKA Staging or Enriched
  - Permissioned by automated production jobs
- Curated layer – the consumption zone
  - Data marts/Star schema
  - Optimized for analytics
  - Includes ML models
  - Permission by department or function
  - Highly documented
- Sandbox Layer – the laboratory zone
  - Wrangling, experimentation
  - Staging / work area
  - Permission both read and write per user, team or project

Folder structure guidance

- Raw zone: organise by source system then entity
  - Optimal for folder security:
  \Raw\DataSource\ Entity\YYYY\MM\DD\FileData_YYYY_MM_DD.csv
  - Tedious for folder security:
  \Raw\YYYY\MM\DD\DataSource \Entity \FileData_YYYY_MM_DD.csv
- Cleansed zone: organise by department first
- Curated zone: organise by consumer group or by data mart
- Confidential or sensitive data can be stored in a separate zone or sub-folders:
  - \Raw\General\DataSource\Entity\YYYY\MM\DD\File.csv
  - \Raw\Sensitive\DataSource\Entity\YYYY\MM\DD\File.csv
- Zones may be top level, 2nd level folders, filesystems, different storage accounts

Organizing a Data Lake – Folder structure

Objectives

- ✓ Plan the structure based on optimal data retrieval
- ✓ Avoid a chaotic, unorganized data swamp

Common ways to organize the data:

Time Partitioning
Year/Month/Day/Hour/Minute

Subject Area

Security Boundaries
Department
Business unit
etc…

Downstream App/Purpose

Data Retention Policy
Temporary data
Permanent data
Applicable period (ex: project lifetime)
etc…

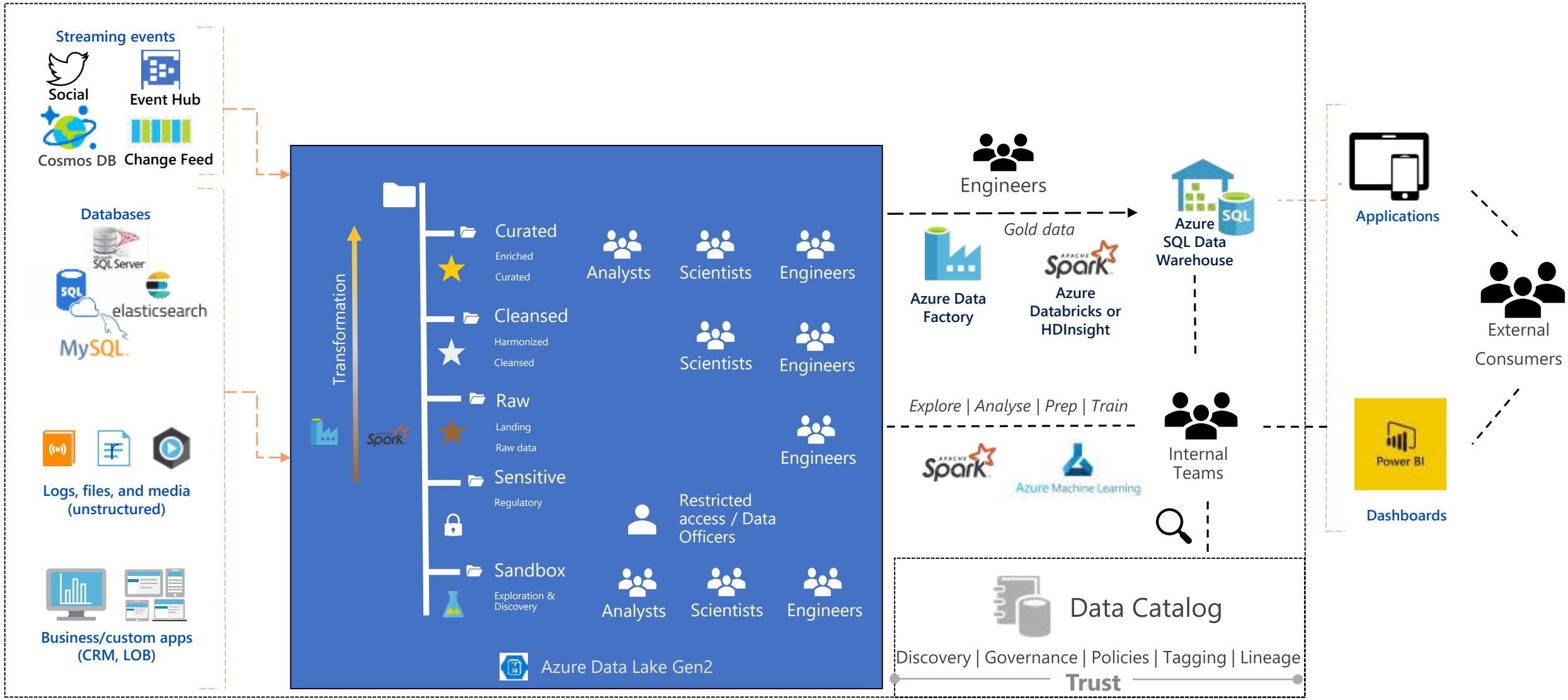
Business Impact / Criticality
High (HBI)
Medium (MBI)
Low (LBI)
etc…

Owner / Steward / SME

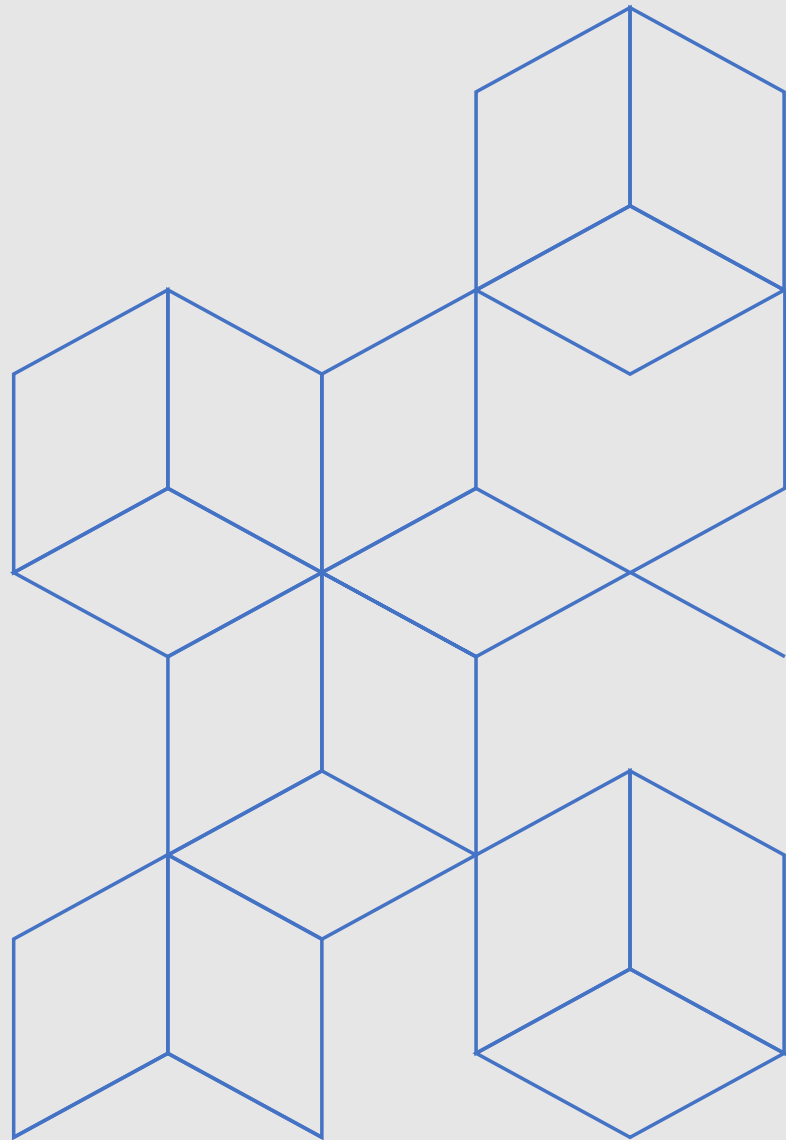
Probability of Data Access
Recent/current data
Historical data
etc…

Confidential Classification
Public information
Internal use only
Supplier/partner confidential
Personally identifiable information (PII)
Sensitive – financial
Sensitive – intellectual property
etc…

Data Lake Architecture – Concepts, Tools & Process



Modern Data Warehouse on Azure



MODERN DATA WAREHOUSING

AZURE DATA FACTORY

Ingest from 75+ sources,
up to 1 Gbps

Hybrid data integration
with drag-n-drop UI

Execute SQL Server
Integration Services packages

AZURE DATABRICKS

Fast, easy, collaborative Apache®
Spark™-based analytics platform

Native integration
with Azure services

Enterprise-grade Azure security

AZURE SQL DATA WAREHOUSE

Unlimited scale and performance

Flexibility to fit your needs

Secure, trusted and compliant

AZURE DATA LAKE STORAGE
G E N 2

- ✓ Combines Hierarchical File and Object Store
- ✓ Optimized for Analytics Workloads
- ✓ Azure Active Directory
- ✓ Exabyte Scale

Azure Data Lake Storage Gen2

A **"no-compromises"** Data Lake: secure, performant, massively-scalable Data Lake storage that brings the cost and scale profile of object storage together with the performance and analytics feature set of data lake storage

SECURE

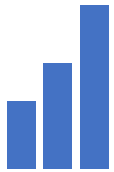
- ✓ Support for fine-grained ACLs, protecting data at the file and folder level
- ✓ Multi-layered protection via at-rest Storage Service encryption and Azure Active Directory integration

MANAGEABLE

- ✓ Automated Lifecycle Policy Management
- ✓ Object Level tiering

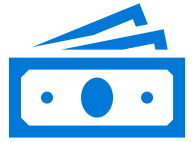
FAST

- ✓ Atomic directory operations means jobs complete faster



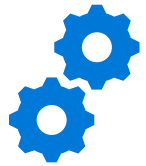
SCALABLE

- ✓ No limits on data store size
- ✓ Global footprint (50 regions)



COST EFFECTIVE

- ✓ Object store pricing levels
- ✓ File system operations minimize transactions required for job completion

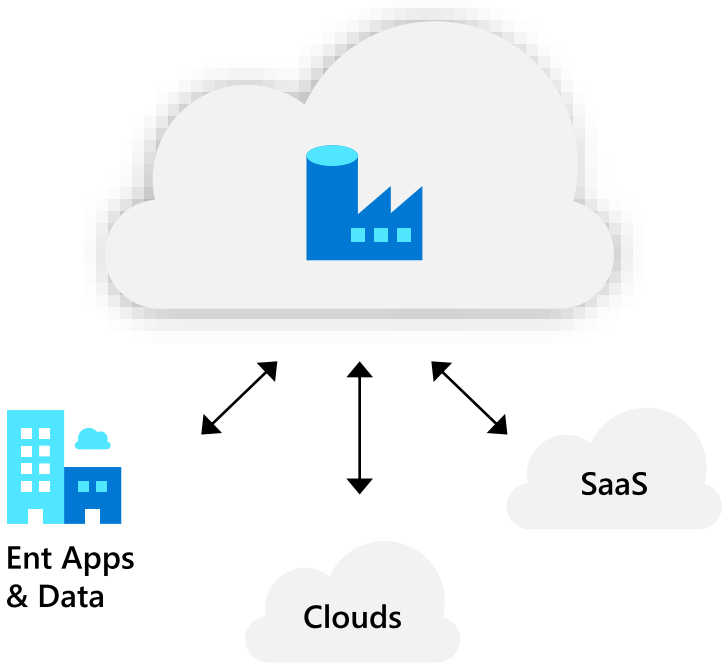


INTEGRATION READY

- ✓ Optimized for Spark and Hadoop Analytic Engines
- ✓ Tightly integrated with Azure end to end analytics solutions

Azure Data Factory

Data Integration Service: Serverless, Scalable, Hybrid



Data Movement and Transformation @Scale

Cloud & Hybrid w/ 80+ connectors provided
Up to 2 GB/s, ETL/ELT in the cloud

Hybrid Pipeline Model

Seamlessly span: on premise, Azure, other clouds & SaaS
Run on-demand, scheduled, data-availability or on event

Author & Monitor

Programmability w/ multi-language SDK
Visual Tools

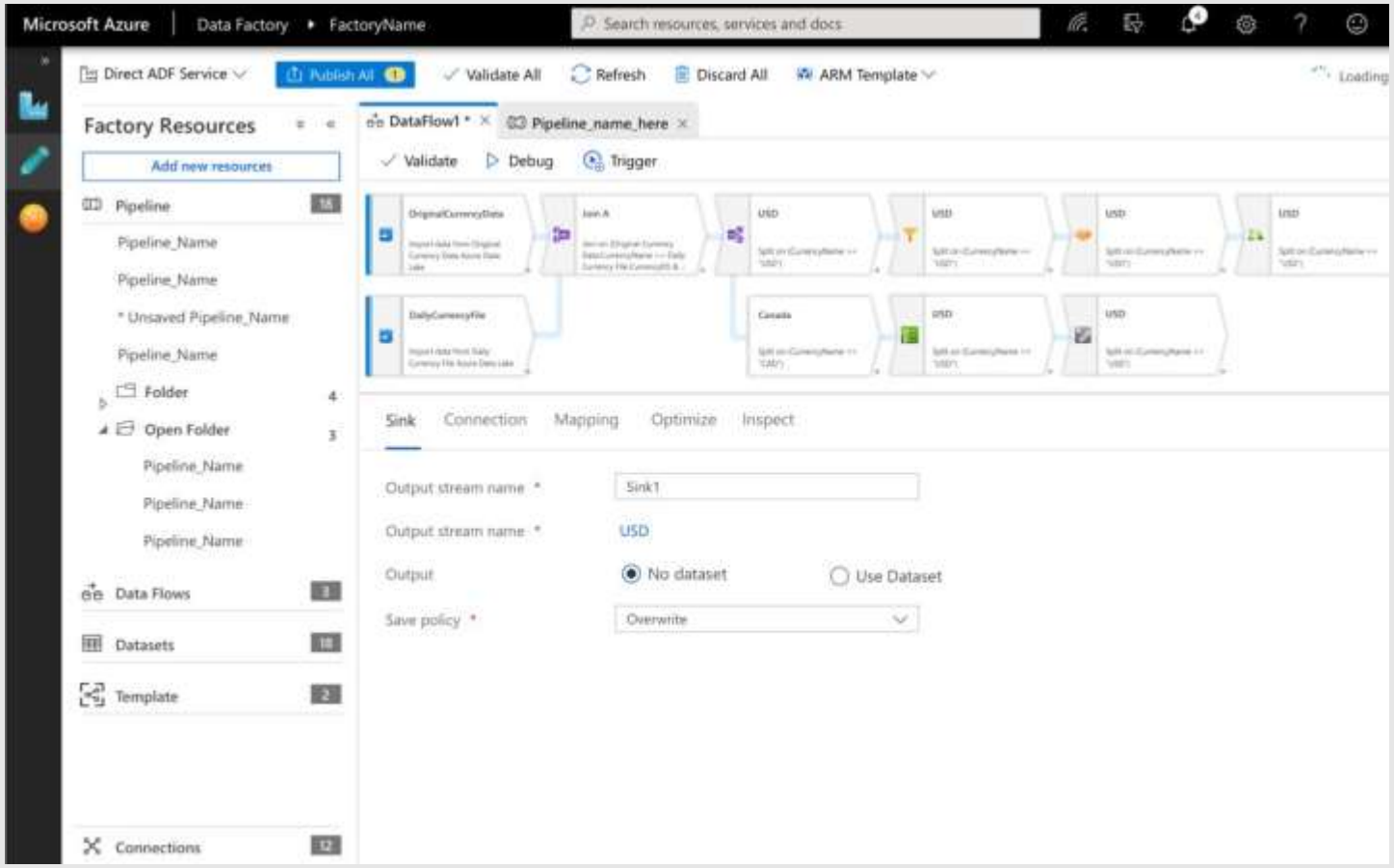
SSIS Package Execution

Lift existing SQL Server ETL to Azure
Use existing tools (SSMS, SSDT)

What is ADF Mapping Data Flow?

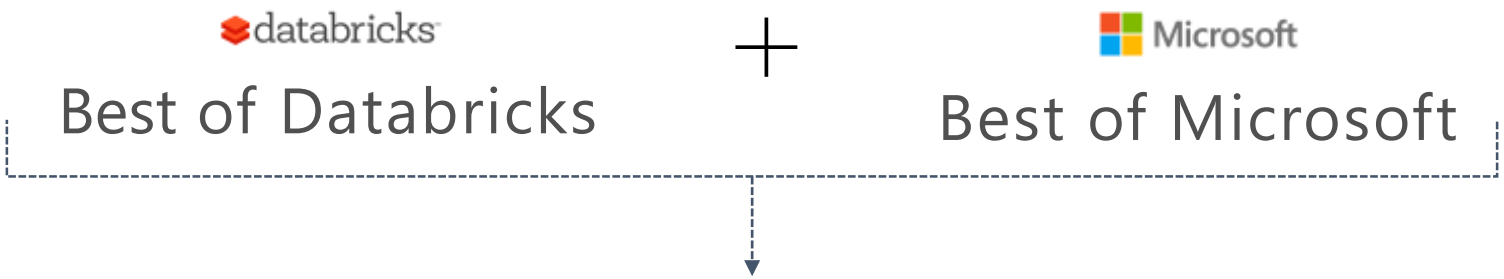
Data Flow is a new feature of Azure Data Factory that allows you to build data transformations in a visual user interface

- Transform Data, At Scale, in the Cloud, Zero-Code
  - Cloud-first, scale-out ELT
  - Code-free dataflow pipelines
- Serverless scale-out transformation execution engine
- Maximum Productivity for Data Engineers
  - Does NOT require understanding of Spark / Scala / Python / Java
- Resilient Data Transformation Flows
  - Built for big data scenarios with unstructured data requirements
  - Operationalize with Data Factory scheduling, control flow and monitoring



What is Azure Databricks?

A fast, easy and collaborative Apache® Spark™ based analytics platform optimized for Azure



- Designed in collaboration with the founders of Apache Spark
- One-click set up; streamlined workflows
- Interactive workspace that enables collaboration between data scientists, data engineers, and business analysts.
- Native integration with Azure services (Power BI, SQL DW, Cosmos DB, Blob Storage)
- Enterprise-grade Azure security (Active Directory integration, compliance, enterprise-grade SLAs)

Why Spark?

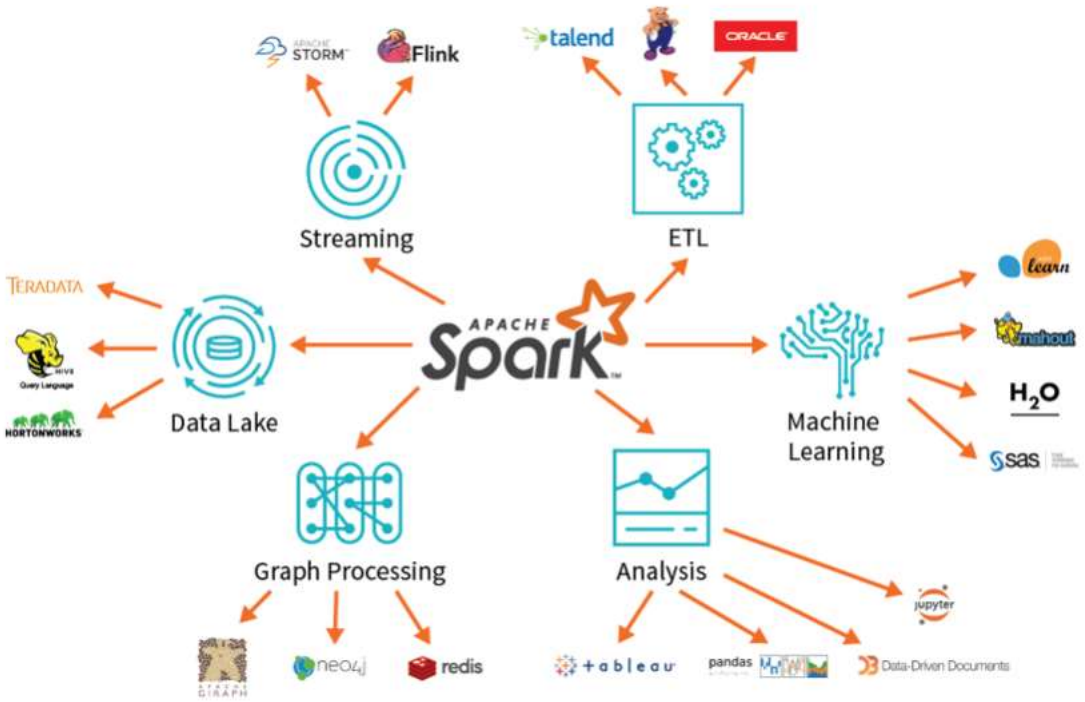
- Open-source data processing engine built around **speed, ease of use, and sophisticated analytics**
- In memory engine that is up to **100 times faster than Hadoop**
- **Largest open-source data project** with 1000+ contributors
- **Highly extensible** with support for Scala, Java and Python alongside Spark SQL, GraphX, Streaming and Machine Learning Library (Mllib)

AZURE DATABRICKS

- Azure Databricks is a **first party** service on Azure.
  - Unlike with other clouds, it is not an Azure Marketplace or a 3rd party hosted service.
- Azure Databricks is integrated seamlessly with Azure services:
  - Azure Portal: Service an be launched directly from Azure Portal
  - Azure Storage Services: Directly access data in Azure Blob Storage and Azure Data Lake Store
  - Azure Active Directory: For user authentication, eliminating the need to maintain two separate sets of uses in Databricks and Azure.
  - Azure SQL DW and Azure Cosmos DB: Enables you to combine structured and unstructured data for analytics
  - Apache Kafka for HDInsight: Enables you to use Kafka as a streaming data source or sink
  - Azure Billing: You get a single bill from Azure
  - Azure Power BI: For rich data visualization
- Eliminates need to create a separate account with Databricks.

WHEN TO USE AZURE DATABRICKS

- Read and process huge files and data sets
- Query, explore, and visualize data sets
- Join disparate data sets found in data lakes
- Train and evaluate machine learning models
- Process live streams of data
- Perform analysis on large graph data sets and social networks



(Demo 02 Why Spark, if interested)

Data Lake Storage Best Practices

- Plan zones, folder structure and security groups upfront
- Raw data
  - stored in original format
  - Immutable & highly secure
  - life-cycle policies to reduce cost – coming soon to ADLS gen2
- In non-raw zone use Parquet/Delta format – performance & compression
- Each folder should contain the same file format
- Use consistent naming conventions
- Optimal file size
  - Large files are better than smaller files
    - more cost-effective & yields better analytic performance
    - Azure Data Lake Storage Gen2 is highly optimised to perform faster on larger files. This means that your analytics jobs will run faster, when operating on larger files, thus further reducing your TCO for running analytics jobs. files > 4 MB in size incurs a lower price for every 4 MB block of data read beyond the first 4 MB. eg to read a single file that is 16 MB is cheaper than reading 4 files that are 4 MB each.
    - Databricks / Spark recommendations – 64MB – 1GB

Best Practices for Spark Optimization

- Prefer columnar formats over text for analytics
- Parquet
  - Column pruning
  - Predicate pushdown – file skipping
- Avoid CSV/JSON except for raw zone
  - Must parse whole row
  - Inferring schema can be slow
  - No predicate push down
- Avoid large GZIP text files
- Avoid over/under partitioning
  - Don’t chose partition column with high cardinality. Consider hash partitioning

Data Lake Security Best Practices

- Permissions:
  - assign relevant permissions as early as possible in your design/development/operationalization phase
  - Use RBAC to manage the resource – account admins
  - Use ACLS to view/edit/manage the data (POSIX style) – users & jobs
  - Restrict write permissions except for
    - Users' sandbox/work/private zone
    - Automated jobs using service principal
  - Use security groups, resist assigning ACLs to individuals
    - Changing permissions at the group level will require recursive operations (slow)
- Access to raw data should be restricted

Common scenarios for permissions

Execute permissions are required to traverse the folder structure from top level down

Operation	/	Oregon/	Portland/	Data.txt
Read Data.txt	`--X`	`--X`	`--X`	`R--`
Append to Data.txt	`--X`	`--X`	`--X`	`RW-`
Delete Data.txt	`--X`	`--X`	`-WX`	`---`
Create Data.txt	`--X`	`--X`	`-WX`	`---`
List /	`R-X`	`---`	`---`	`---`
List /Oregon/	`--X`	`R-X`	`---`	`---`
List /Oregon/Portland/	`--X`	`--X`	`R-X`	`---`

Data Lake Operational Best Practices

- Access to raw data should be restricted
- Raw zone
  - Optimal write performance
  - Security & folder structure based on source systems
- Curated zone
  - Optimal data retrieval
  - Ease of data discovery
- CICD / script everything:
  - Create zones, folder structures
  - Regen from raw zone
  - RBAC, Groups & ACLs – check Powershell support for ADLSv2
  - Different environments – dev, test, prod should be separate accounts
- DataOps – DevOps for data

Multiple or Single Data Lakes

- Reasons for multiple
  - Regulatory constraints / Data sovereignty
  - Organizational barriers
  - Predictability
- Benefits of single/centralized
  - Optimized resource usage
  - Administrative and operational cost
  - Data redundancy mitigation
  - Reuse
  - Enterprise scale projects

Guidance for ADLS

- For gen2 check the supported features wrt multi-protocol access (Blob interop) & SDK
- Data Lake Storage limits
  - Max accounts per sub = 10 (request increase)
  - Max access & default ACLs per file or folder = 32 (hard limit)
  - 2 PB for US and Europe, 500 TB for all other regions, which includes the UK
    - Limits can be raised on request
  - Max request rate 20,000 per second per storage account
  - See other limits here
- ADLS doesn't have a true inheritance model, use default ACLs
  - Changing the default ACL on a parent does not affect the access ACL or default ACL of child items that already exist.
- Scripting permission changes on existing folder requires recursion
- RBAC (filesystem level) evaluated at higher priority over ACLs
- Storage Blob Data Owner built-in role and SAS auth gain super-user access

Can the data warehouse replace the data lake?

- Challenges of cost
  - Storing all historical and raw data becomes costly
    - DW is an expensive for good reason – premium storage, high performance analytics
  - Large data warehouses have high TCO
    - backups, administration, maintenance require expertise
- Barriers to access
  - Access (exploration, discovery vs analytics) to the data requires dedicated access through the compute layer for simple data retrieval/loading activities
  - Data becomes “locked” due to tightly coupled storage and compute
  - Does not facilitate simple, low-cost, low-impact sharing of data sets
- Limited insights
  - Low fidelity: Data is usually partial, incomplete or discarded to keep costs down
  - DW technologies may not easily or efficiently support raw format — semi-structured data analysis
- Challenges of impact and scale
  - Transformation workloads and data sharing cannot scale without significantly increasing cost
  - Mixed workloads (other than analytics) compete for resources
  - DW has limited concurrency for a given cost
  - DW resources should be conserved for consistent low-latency, high performance analytics i.e. avoid ELT

Can data lake technologies replace the data warehouse? Unlikely due to:

- 1. Workload management/isolation
- 2. Price-Performance - predictability of pricing. predictability of performance. Result-set caching.
- 3. Governance/Security - AD integration, row and column level security, dynamic data masking
- 4. Concurrency at a fixed scale. Auto scale is great but usually ends in cache-eviction.

Store in DL vs DW

- Store in DW when:
  - Mission critical data
  - Time sensitive analytics
  - Interactive reporting/dashboarding
  - Cost of analytics requires predictability
- Keep in data lake
  - During exploration
  - Low-value data - value TBD
  - Older, less frequently used – active archive

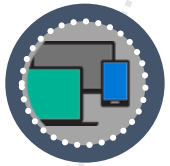
Why your data estate needs a catalog

Data Consumers & Data Producers

What data do I have?
What is in the data?

Where did the data originate?
Can I make a change?

Who else used the data?
Can I trust the data?



Data Officers

What business sensitive or PII data exists in my org?
How can I understand risk level across my data?

Who is using the data and for what purpose?

Is our data compliant with regulations and corporate policies?

Azure Data Catalog Empowers You to Answer These Questions

Additional reading..

- https://docs.microsoft.com/en-us/azure/architecture/data-guide/scenarios/data-lake
- https://www.blue-granite.com/data-lakes-in-a-modern-data-architecture-ebook
- https://www.sqlchick.com/entries/2019/1/20/faqs-about-organizing-a-data-lake
- https://www.sqlchick.com/entries/2017/12/30/zones-in-a-data-lake
- https://www.sqlchick.com/entries/2016/7/31/data-lake-use-cases-and-planning
- https://www.jamesserra.com/archive/2019/09/ways-to-access-data-in-adls-gen2/
- https://www.amazon.co.uk/dp/1491931558/ref=cm_sw_em_r_mt_dp_U_h.GRDb0QB8VXD
- https://medium.com/microsoftazure/analytics-and-the-importance-of-the-data-lake-on-azure-b5c66ca4a561