Building a Data Lake in Azure with Spark and Databricks

Presented at CHUG on 2019-08-08

Alex Wiss-Wolferding
Principal Consultant
Cloud Engineering

Presenter Profile

- 5+ years with Clarity Insights
- 4+ years with big data technologies (Hadoop, Spark)
- 3+ years building data and analytics solutions in the cloud (AWS/Azure)
- Certified GCP cloud architect
- Certified in designing and implementing big data analytics solutions on Azure
- Certified Hortonworks developer and administrator
- Once built a Hortonworks cluster from old Clarity laptops

CLARITY INSIGHTS

Unleash your Insights

Who We Are

Founded in
2008

300+
consultants

National
presence

200%
growth since 2011

Problems We Solve

Marketing

Optimize omni-channel marketing measurement

Improve trade promotion spending

Improve demand forecasting

Data infrastructure Modernization

Create a single view of the customer

Lower costs of ownership

Improve capabilities

Operational Efficiency

Supply chain optimization

Improve operations using IoT data

Customer/Member Experience

Reduce churn

Increase cross-sell and upsell

Who We've Worked With

The world's biggest social media company

Many of the nation's most respected healthcare brands

The 3 biggest media and communication companies in the world

Top 20 CPG and retail companies

Agenda

- Decisions, Decisions
  - What Is a Data Lake and Why Would You Want One?
  - Why Azure and Databricks?
- Building the Lake
  - Getting Data into the Lake
  - Organizing the Lake
  - Using Delta Lake for Integrated/Curated Layers
  - Securing the Lake
- Gaining Value from the Lake
  - Enabling Ad-Hoc Analytics Directly Against the Lake

Decisions, Decisions

What is a Data Lake and Why Would I Want One?

What Is It?

A data lake is a single repository that can potentially store all data within an organization/enterprise, including structured, semi-structured, and unstructured data entities. Lakes typically adhere to certain principles:

- **All data** ingested into the lake should be stored in its **raw format**, so that it can be accessed by multiple consumers for multiple use cases.
- Storing data in the lake should be **cost-effective**, so that there is little or no concern as to how much data is stored.
- Data in the lake should be **cataloged,** so that users know what is in the lake and what its **value** is.
- The data lake and any compute contexts around it should be **decoupled**, so that there is no processing bottleneck when accessing data in the lake.

"If you think of a datamart as a store of bottled water – cleansed and packaged and structured for easy consumption – the data lake is a large body of water in a more natural state. The contents of the data lake stream in from a source to fill the lake, and various users of the lake can come to examine, dive in, or take samples."

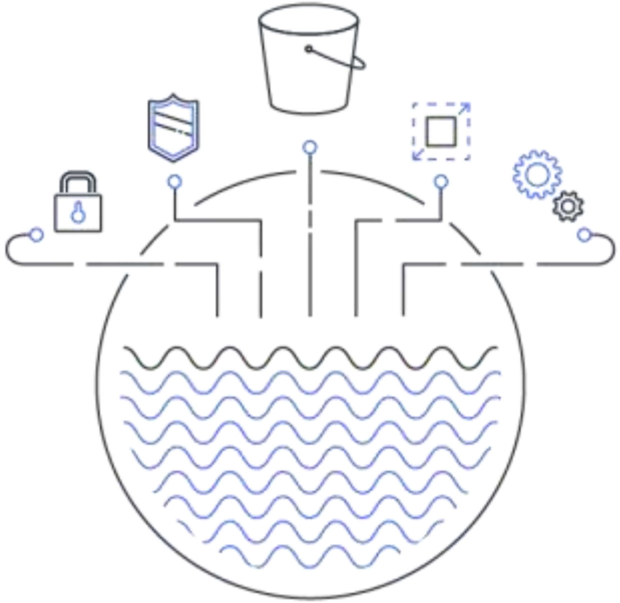
James Dixon, CTO of Pentaho

What Can It Do for Me?

A data lake allows rapid ingestion and co-mingling of many different data sources, which can enable a multitude of different use cases including machine learning and artificial intelligence.

Lakes can also act as a data hub and exchange, taking the load off source systems (that may not scale well) and serving as a single source for data sharing.

While a lake should be built with specific use cases in mind, the ingestion and storage of raw data ensures that future use cases are supported while working towards current ones.



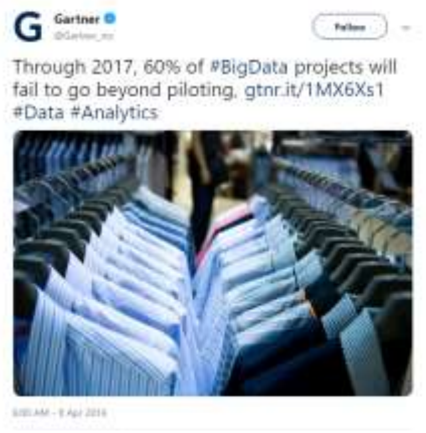
Challenges

A data lake is not a “silver bullet” for big data analytics.

In 2016 Gartner famously estimated that 60% of big data projects will fail, never going beyond the pilot phase.

Failures can often be attributed to a number of missteps when building a data lake:

- **Over-modeling** of data in the lake, which takes too much time to ingest.
- Lack of **clear direction** for what data is valuable and to do with the data once it’s in the lake.
- Lack of rigor when cataloging data in the lake and assigning **value** to the data in the lake.
- Lack of **availability** of data in the lake for both operational and ad-hoc projects.



Gartner

Through 2017, 60% of #BigData projects will fail to go beyond piloting. gtnr.it/1MX6Xs1 #Data #Analytics

“We see customers creating big data graveyards, dumping everything into Hadoop distributed file system (HDFS) and hoping to do something with it down the road. But then they just lose track of what’s there.
The main challenge is not creating a data lake, but taking advantage of the opportunities it presents.”

Sean Martin, CTO of Cambridge Semantics

Why Azure and Databricks?

Why Azure?

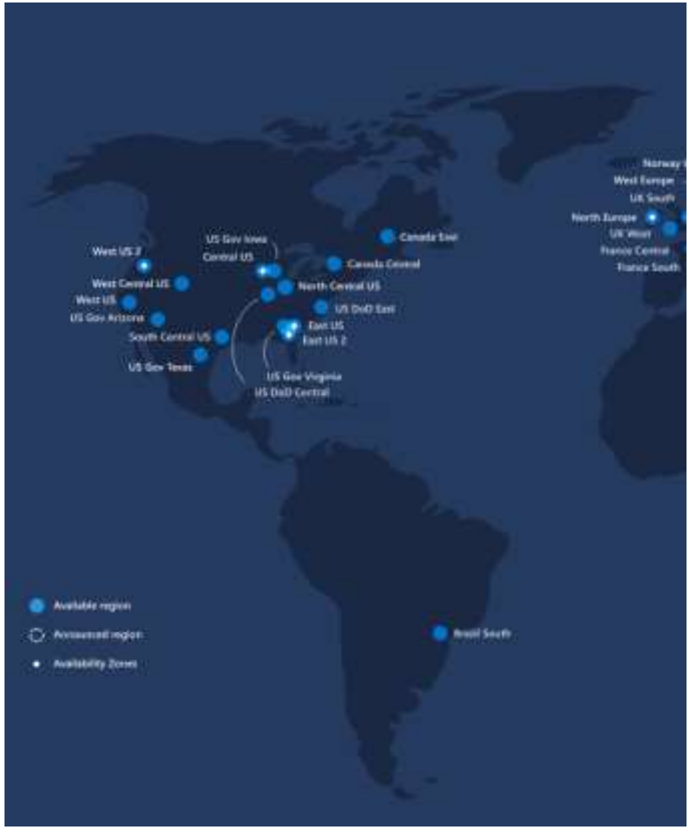
Microsoft Azure is a public cloud provider with SaaS, PaaS, and IaaS offerings across 54 global regions (as of August 2019).

Microsoft reported that their “commercial cloud” business - which includes Azure, as well as other offerings like Office 365 - grew to $9.6 billion Q1 of 2019. AWS reported sales of $7.7 billion for the same time period.

Azure integrates seamlessly with many other Microsoft and Windows offerings, including Active Directory.

Offerings within Azure that are relevant to our conversation today include:

- Azure Data Lake Store (Gen 2)
- Azure Databricks
- Azure Event Hubs
- Azure Storage (Blob, Queue, and Table)



Source: https://www.parkmycloud.com/blog/aws-vs-azure-vs-google-cloud-market-share/

Why Spark/Azure Databricks?

Spark is a fast and scalable processing engine for both batch and streaming workloads.

Databricks is a cloud-native PaaS for Apache Spark. It allows teams to build massively scalable applications using Spark without needing to deal with infrastructure. It handles cluster creation and management and provides added features like auto-scaling and scheduling.

It also provides a notebook interface, allowing ad-hoc exploration and analytics of the lake and rapid prototyping of new pipelines at full scale.

Azure Databricks is a first-class service offering within the Azure platform, integrating tightly with many of their other core offerings (Azure Data Lake Store, specifically).

databricks
INCREASE PRODUCTIVITY AND COLLABORATION
BUILD ON A SECURE, TRUSTED CLOUD
SCALE WITHOUT LIMITS

Building the Lake

Getting Data into the Lake

Common Types of Sources

Batch

- Flat files (CSV, Excel, JSON, etc.) from storage services (NFS/SMB, SFTP, Box, Google Drive, etc.).
- Existing data warehouses and marts.
- Queryable APIs.
- Legacy systems (mainframe, etc.).

Streaming/Event-Driven

- Internal/external services capable of sending webhooks.
- Pub/sub brokers (Apache Kafka, Azure Event Hubs, etc.).
- Web services with streaming capabilities.
- IoT devices.

An Argument for Custom Ingestion

Azure provides services specifically targeted towards data ingestion (Azure Data Factory, Azure Logic Apps, and Azure Event Hubs Capture). These services allow rapid prototyping and development of simple pipelines, but lack higher-level functionality and customization that will likely be required to build your lake. Shortcomings include:

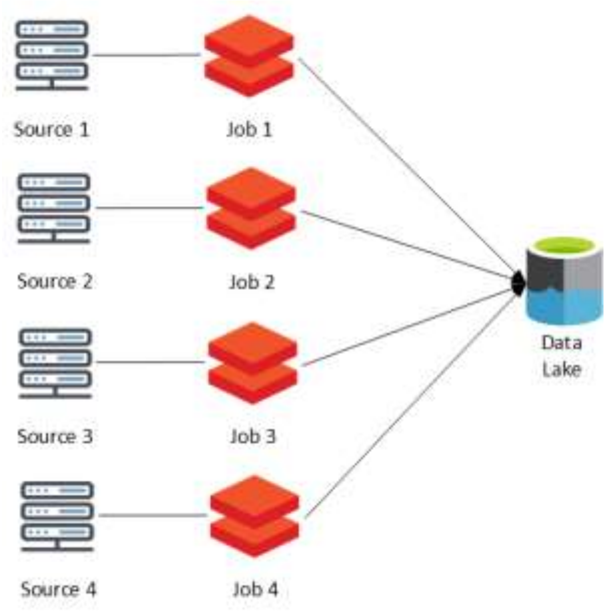
- Limited list of supported sources (and lack of customization).
- Less transparent scalability/performance.
- Limited customization for target path structure/compression/format.
- Cost is more difficult to control.
- Config-based tools make unit testing difficult.

Using Azure Databricks to write custom ingestion code (potentially supplementing with Azure Functions for certain sources) allows full customization while keeping the operational management and development overhead low.

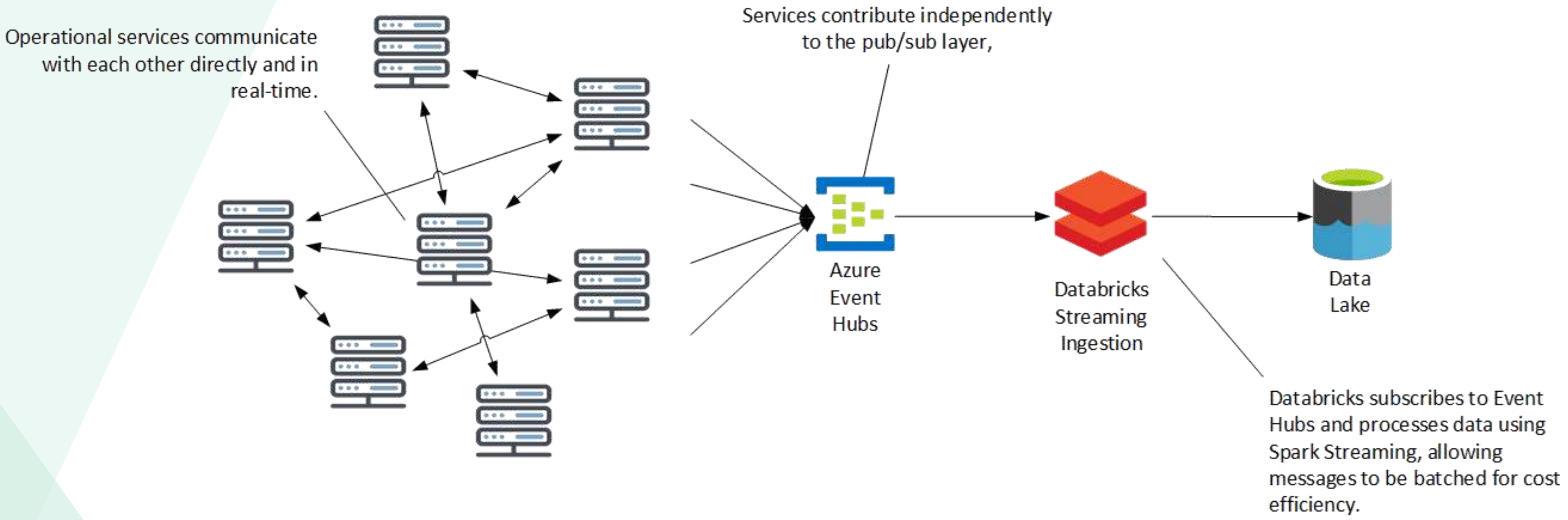
Advantages of Using Databricks for Ingestion

Using Databricks to run Apache Spark for ingestion provides several advantages over other Azure services, as well as over an on-prem or IaaS Spark implementation:

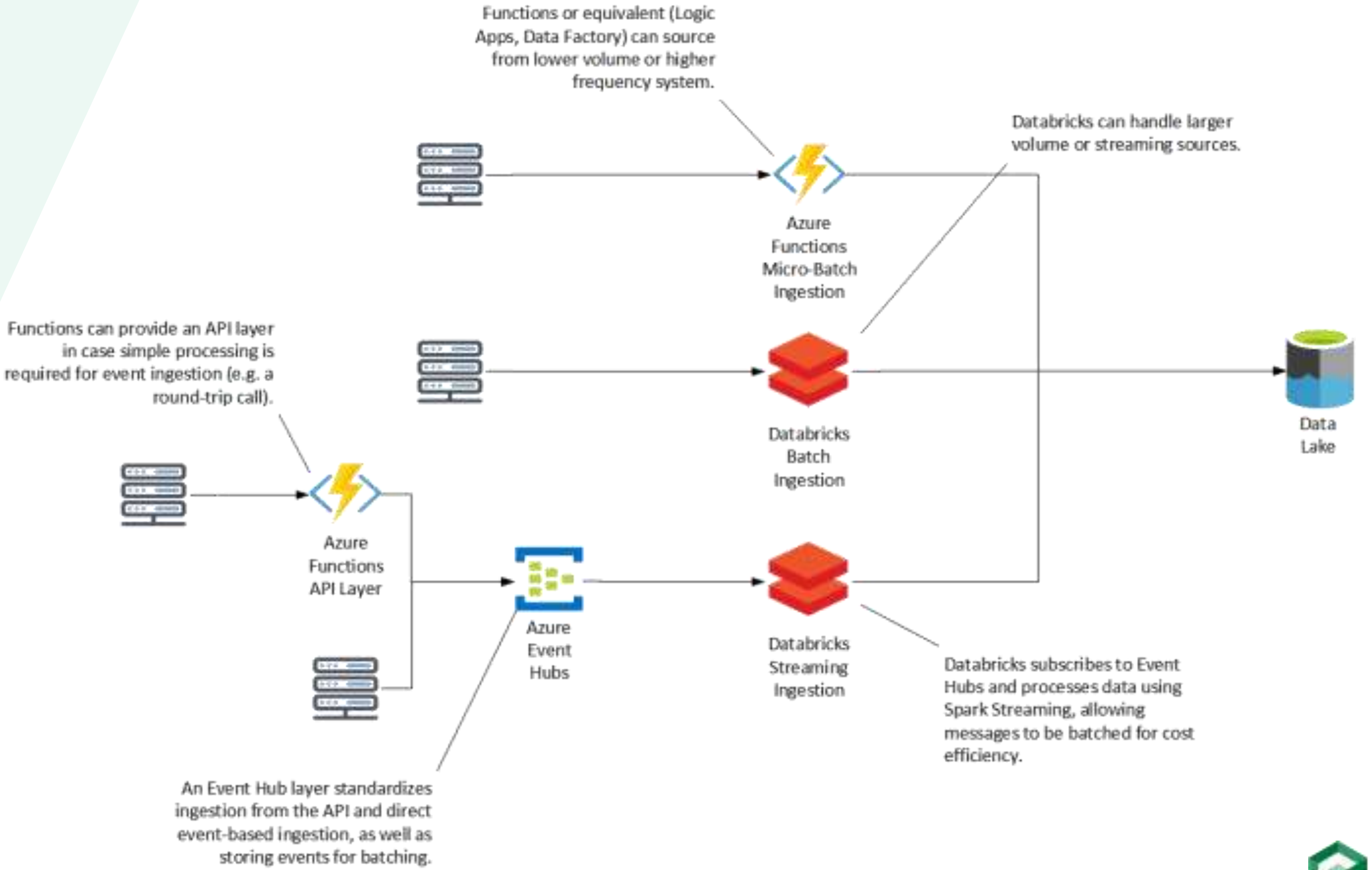
- Full access to Apache Spark, including the ability to add additional libraries.
- Databricks "job" construct allows clusters to spin up per execution, each with their own configuration.
- Ability to standardize path structure across all different source types.
- More granular control over ingestion frequency and scheduling.



Data Lakes in a Microservice Architecture


Operational services communicate with each other directly and in real-time.
Services contribute independently to the pub/sub layer,
Azure Event Hubs
Databricks Streaming Ingestion
Data Lake
Databricks subscribes to Event Hubs and processes data using Spark Streaming, allowing messages to be batched for cost efficiency.

Data Lakes in a Complex Enterprise Architecture


Functions or equivalent (Logic Apps, Data Factory) can source from lower volume or higher frequency system.
Databricks can handle larger volume or streaming sources.
Azure Functions Micro-Batch Ingestion
Functions can provide an API layer in case simple processing is required for event ingestion (e.g. a round-trip call).
Databricks Batch Ingestion
Data Lake
Azure Functions API Layer
Azure Event Hubs
Databricks Streaming Ingestion
Databricks subscribes to Event Hubs and processes data using Spark Streaming, allowing messages to be batched for cost efficiency.
An Event Hub layer standardizes ingestion from the API and direct event-based ingestion, as well as storing events for batching.

Organizing the Lake

Picking a Path Structure

While there are common patterns for path structure of the lake, it should ultimately be structured based on what's important to the organization. Paths within a lake are often made up of some combination of the following:

- Lake layer
- Region
- Subject area
- Source system
- Security level

Examples:

```
/<layer>/<region>/<security level>/<subject area>/<data set>/

/<layer>/<security level>/<subject area>/<data set>/region=<region>/

/<layer>/<subject area>/<data set>/security_level=<security level>/
```

Whatever you choose, **be consistent**.

Creating Layers in the Lake

Raw ("Bronze")	Integrated ("Silver")	Curated ("Gold")
• Stores all source data in its raw format. • Is always append-only. • Allows users to rapidly access data and prototype new pipelines/queries/models. • Often uses lifecycle management and cold storage for cost savings.	• Data is lightly processed to a standardized format and convention. • Data types can be applied. • File formats/compression applied (ORC, Parquet, etc.). • Can be append-only or potentially have other load strategies.	• Data must be manually modeled and curated for use. • May resemble a star or snowflake schema with heavily normalized data. • Can feed data into an EDW or potentially replace an EDW all together. • May contain the output of AI/ML models.

Cataloging Lake Data

Without cataloging and stewardship, data lakes can rapidly become data swamps.

There are many tools on the market for building out a data catalog and business glossary (a subset is listed on the right). Common capabilities include:

- Automated indexing of data stores.
- Crowd-sourcing of business metadata.
- Automated data profiling.
- Data quality monitoring.

Alation

Collibra

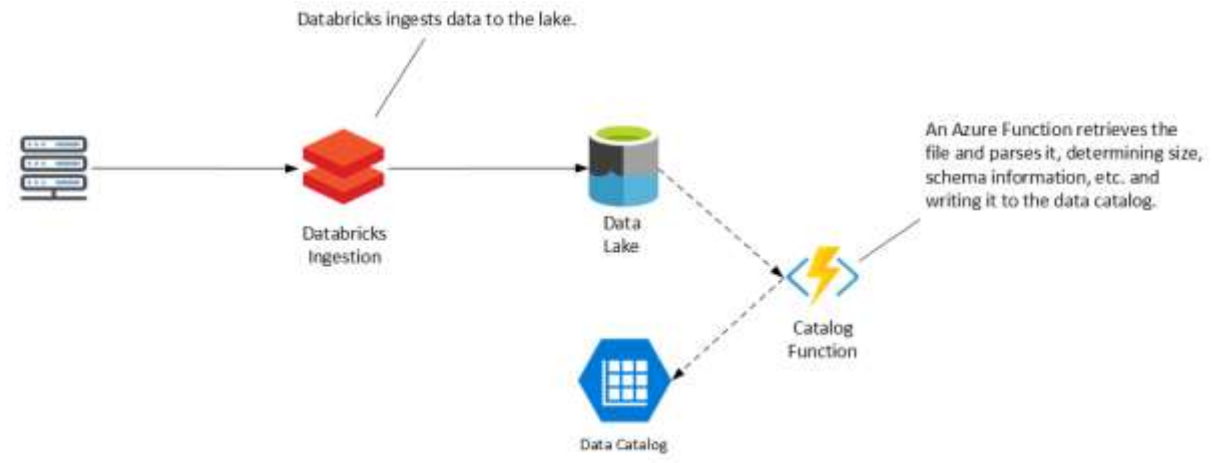
Informatica Data Catalog

Azure Data Catalog

…or Build It Yourself

While tools on the market provide advanced functionality, they may be overkill for your or not provide the necessary granular functionality. A custom built solution can either supplement or replace a separate solution.

Custom built solutions can handle more operational use cases, such as sharing ingested data with other consumers. This can be managed with a data catalog table or topic that keeps track of all files ingested, allowing other systems to subscribe and copy data as it’s ingested

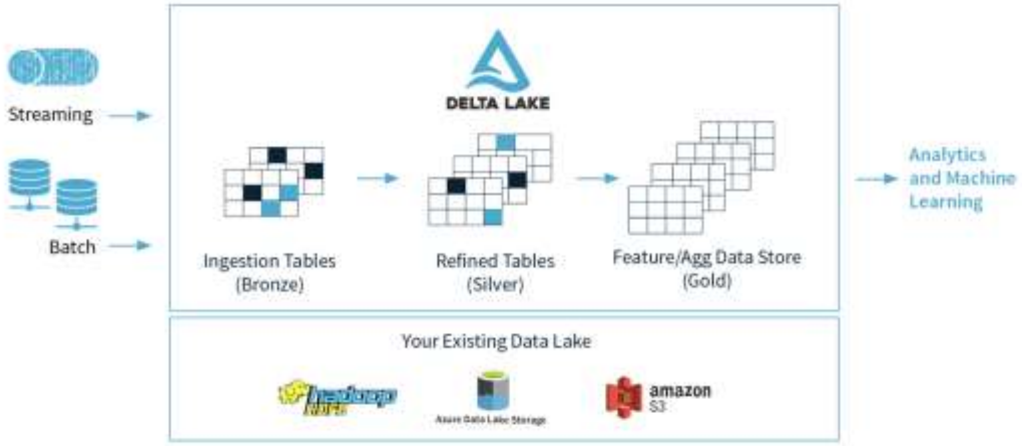

Databricks ingests data to the lake.
Databricks Ingestion
Data Lake
An Azure Function retrieves the file and parses it, determining size, schema information, etc. and writing it to the data catalog.
Catalog Function
Data Catalog

Using Databricks and Delta Lake for Silver/Gold Layers

What is Delta Lake?

Delta Lake is an open-source storage layer that brings transactions and additional storage functionality to Apache Spark. Some highlights include:

- ACID transactions
- Metadata handling
- Time travel/data versioning
- Schema enforcement/evolution
- Updates/merges/deletes



Utilizing Delta Lake

With delta, we can create integrated and curated layers that more closely resemble tables instead than raw files, allowing for faster access and better ease-of-use.

Users/systems can use time-travel to look at the data historically, choosing a point-in-time view, taking complexity out of the code needed to load tables.

ACID transactions ensure consistent data for readers.

Delta in the Integrated Layer

Using a destructive update load strategy instead of append-only allows downstream consumers to see current versions of data instead of having to stitch together raw files.

Delta tables allow automated schema evolution, so as sources and and rename columns they can make it to the integrated layer instantly.

Delta in the Curated Layer

Fact and dimension tables can be built to handle slowly-changing dimensions (SCDs).

This curated layer can replace an expensive MPP or slow data warehouse, serving data to users as well as reporting tools (through Databricks).

Securing the Lake

Decide on an RBAC Strategy

Similar to the path structure, the security of the lake should be based on what attributes of the data are important for restricting access. Role groups and role-based access control (RBAC) should be created and granted to portions of the chosen path structure.

Examples:

Group LakeAccessUKFinance can access /uk/finance/ and all subfolders

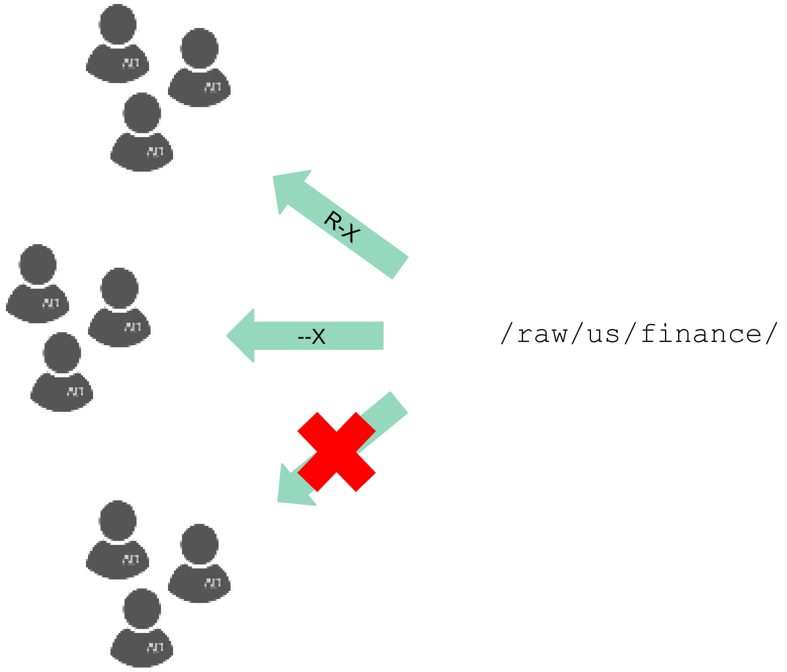
Group LakeAccessRawDB2 can access /uk/raw/db2/ and all subfolders

Group LakeAccessUSRestricted can access /us/restricted/ and all subfolders

Group LakeAccessGermanyCuratedHR can access /curated/germany/hr/ and all subfolders

Security on Azure Data Lake Store (Gen 2) and Databricks

- Azure Data Lake Store Gen 2 (ADLS) supports full POSIX-style ACLs.
- ACLs are fully integrated with Azure Active Directory users, service principals, and groups.
- RBAC role assignments allow permission to be granted to role groups in Active Directory.
- Permissions apply no matter what tool a user is using to access the data.
- Databricks supports pass-through authentication and authorization to ADLS.

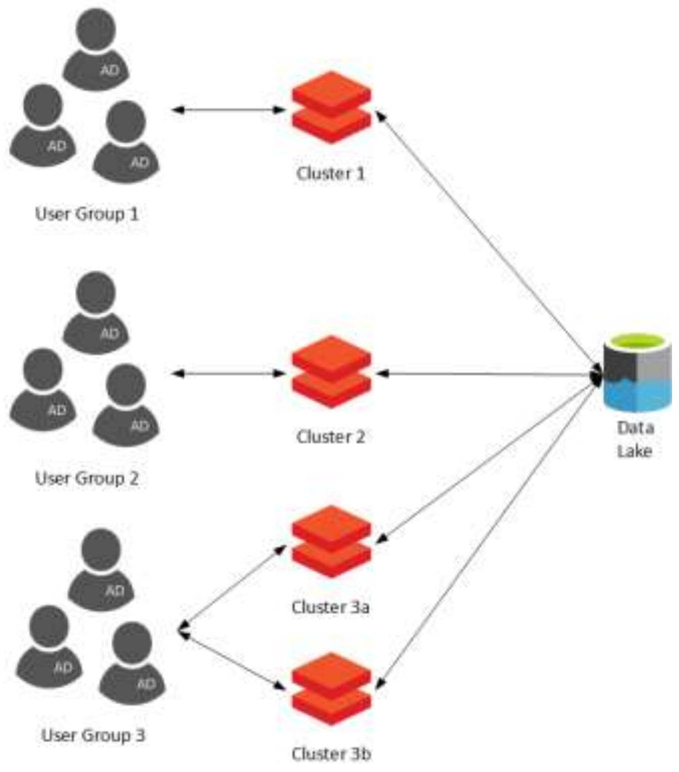


Getting Value from the Lake

Enabling Ad-Hoc Analytics Directly Against the Lake

Databricks as a Query Engine

- Different user groups can have one or more clusters assigned to them, eliminating any resource contention between teams.
- Pass-through authentication ensures that RBAC is maintained when queried through clusters.
- Users can interact with the data directly using SQL, Scala, R, and Python.
- Allows rapid productionalization of ad-hoc analysis, since Databrick/Spark is used to build the integrated and curated layers of the lake.
- Clusters can auto-scale, making analysis large quantities of raw data feasible.



CLARITY INSIGHTS

Thank You!