Azure Data Lake: What, Why, and How

January 8, 2019

Melissa Coates
Solution Architect, BlueGranite


SQLChick.com


@SQLChick


Blue-Granite.com

Agenda

Azure Data Lake: What, Why, and How

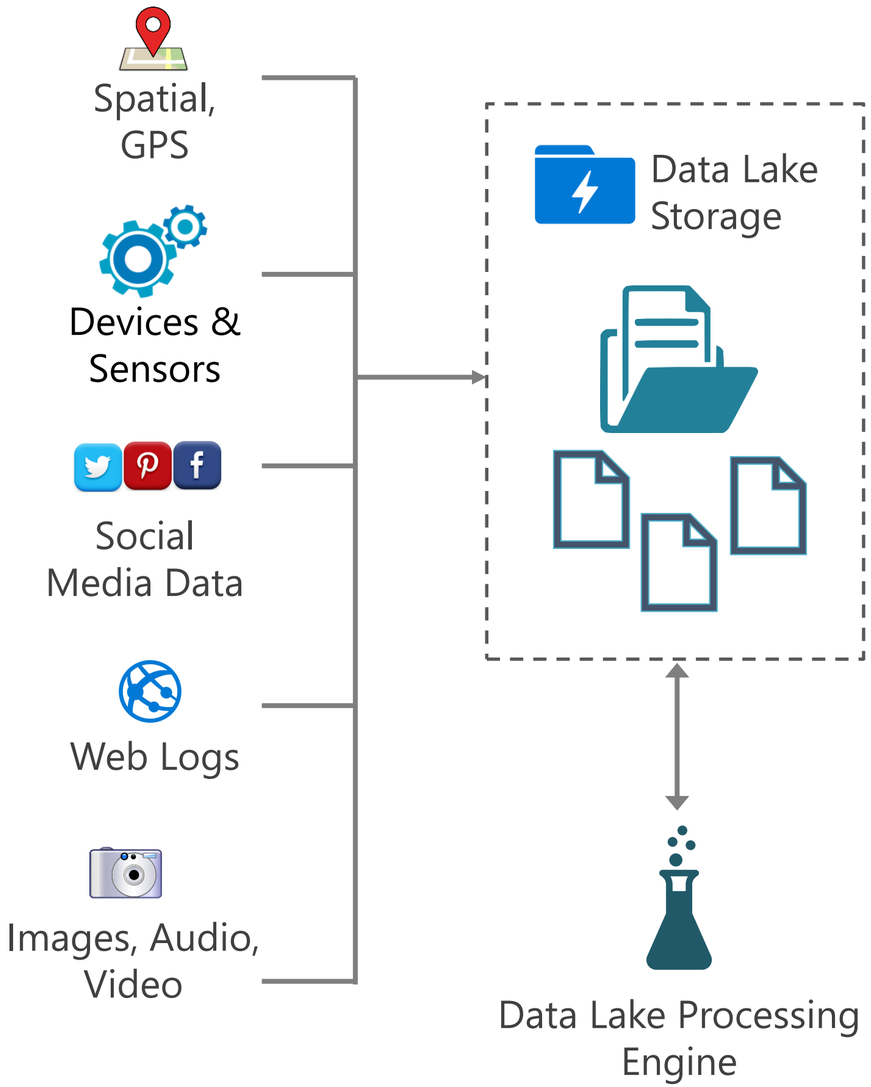
- Data Lake Overview & Use Cases
- Big Data in Azure
- Data Storage in Azure
- Compute in Azure
- Integrating Azure Data Lake in a Multi-Platform Architecture
- Suggestions for Getting Started with a Data Lake Project

As of
Jan 2019:

Things are changing rapidly. Azure Data Lake Storage Gen2 is in public preview.

Data Lake Overview & Use Cases

What is a Data Lake?

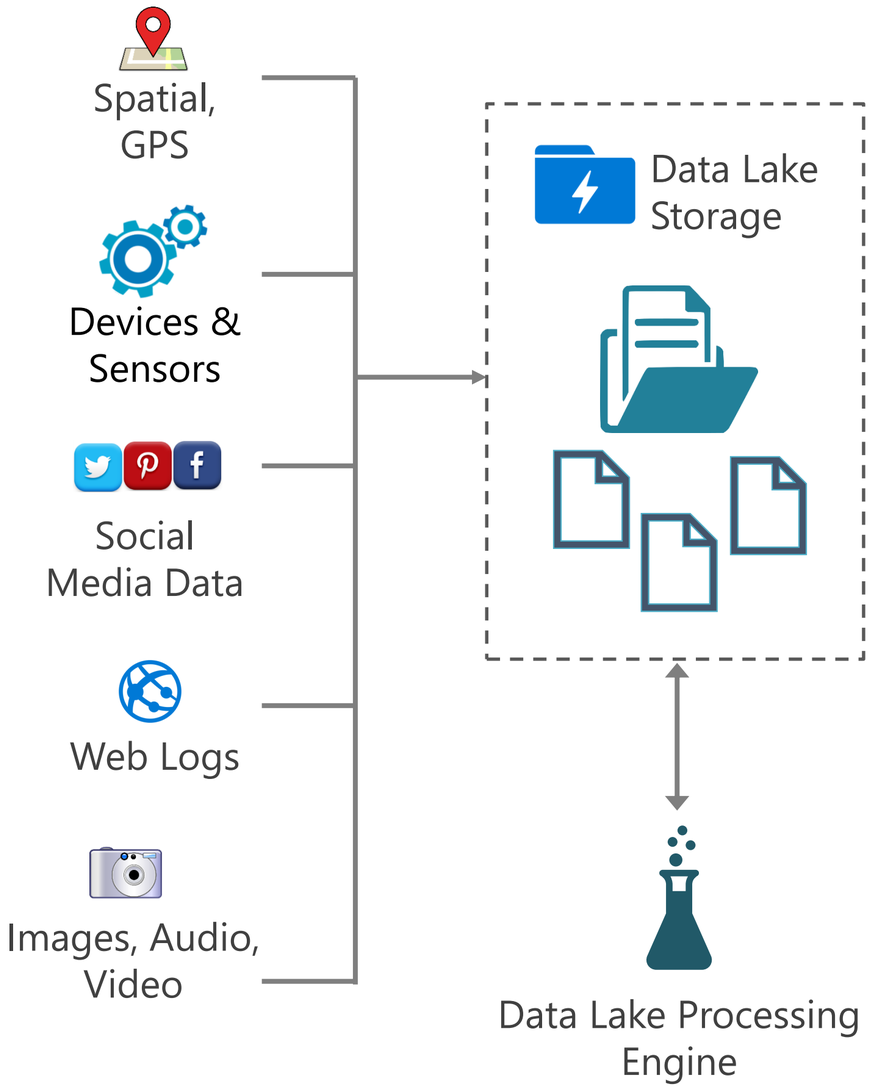


A repository for storing large quantities of disparate sources of data in its native format

One architectural platform to house all types of data:

- ✓ Machine-generated data (ex: IoT, logs)
- ✓ Human-generated data (ex: tweets, e-mail)
- ✓ Traditional operational data (ex: sales, inventory)

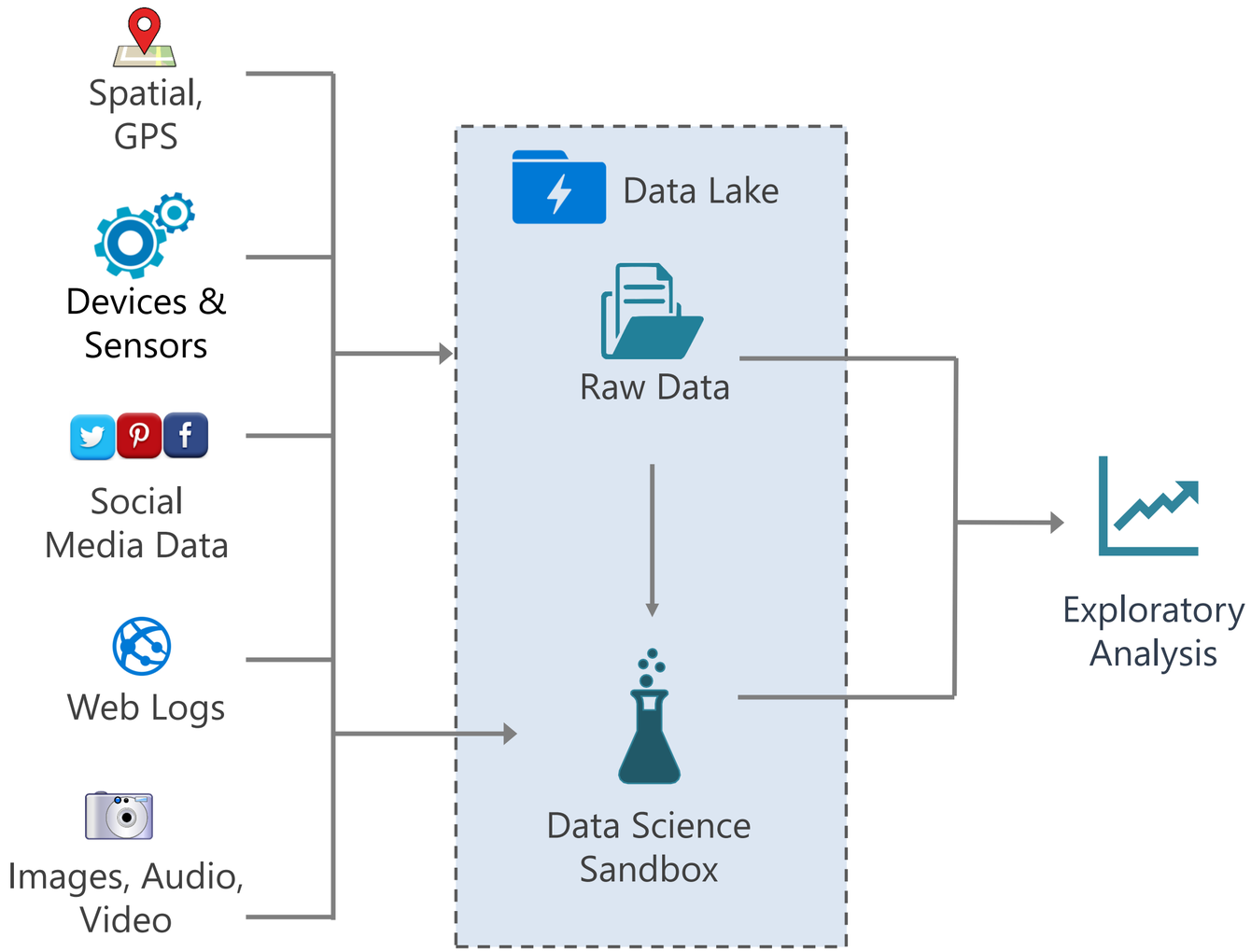
Objectives of a Data Lake



- ✓ Reduce up-front effort to ingest data
- ✓ Defer work to ‘schematize’ until value is known
- ✓ Allow time for defining business value of the data
- ✓ Store low latency data
- ✓ Access to new data types
- ✓ Facilitate advanced analytics scenarios & new use cases
- ✓ Store large volumes of data cost efficiently

Data Lake Use Cases

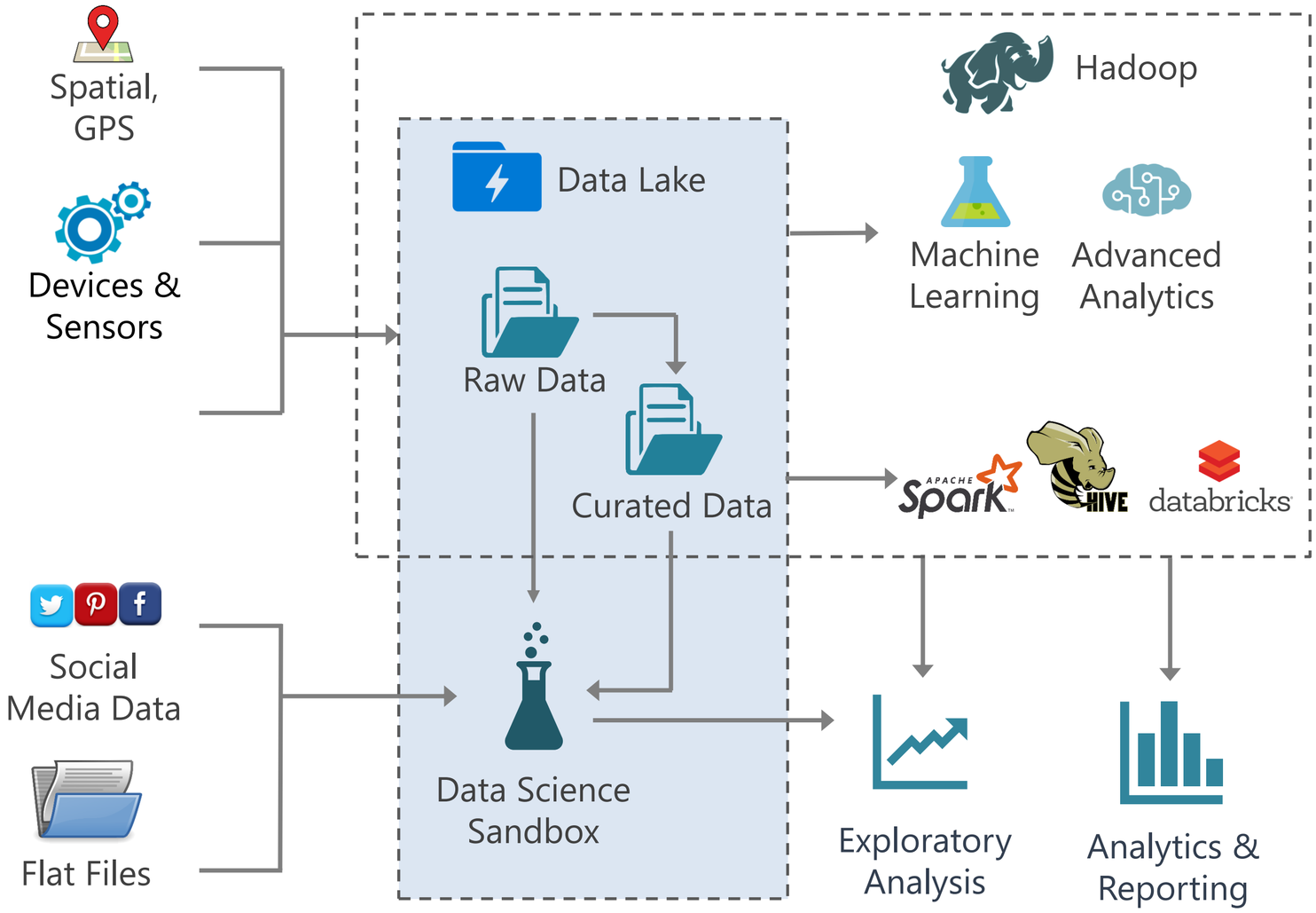
Ingestion of New File Types



- ✓ Preparatory file storage for multi-structured data
- ✓ Exploratory analysis to determine value of new data types & sources
- ✓ Affords additional time for longer-term planning while accumulating data or handling an influx of data

Data Lake Use Cases

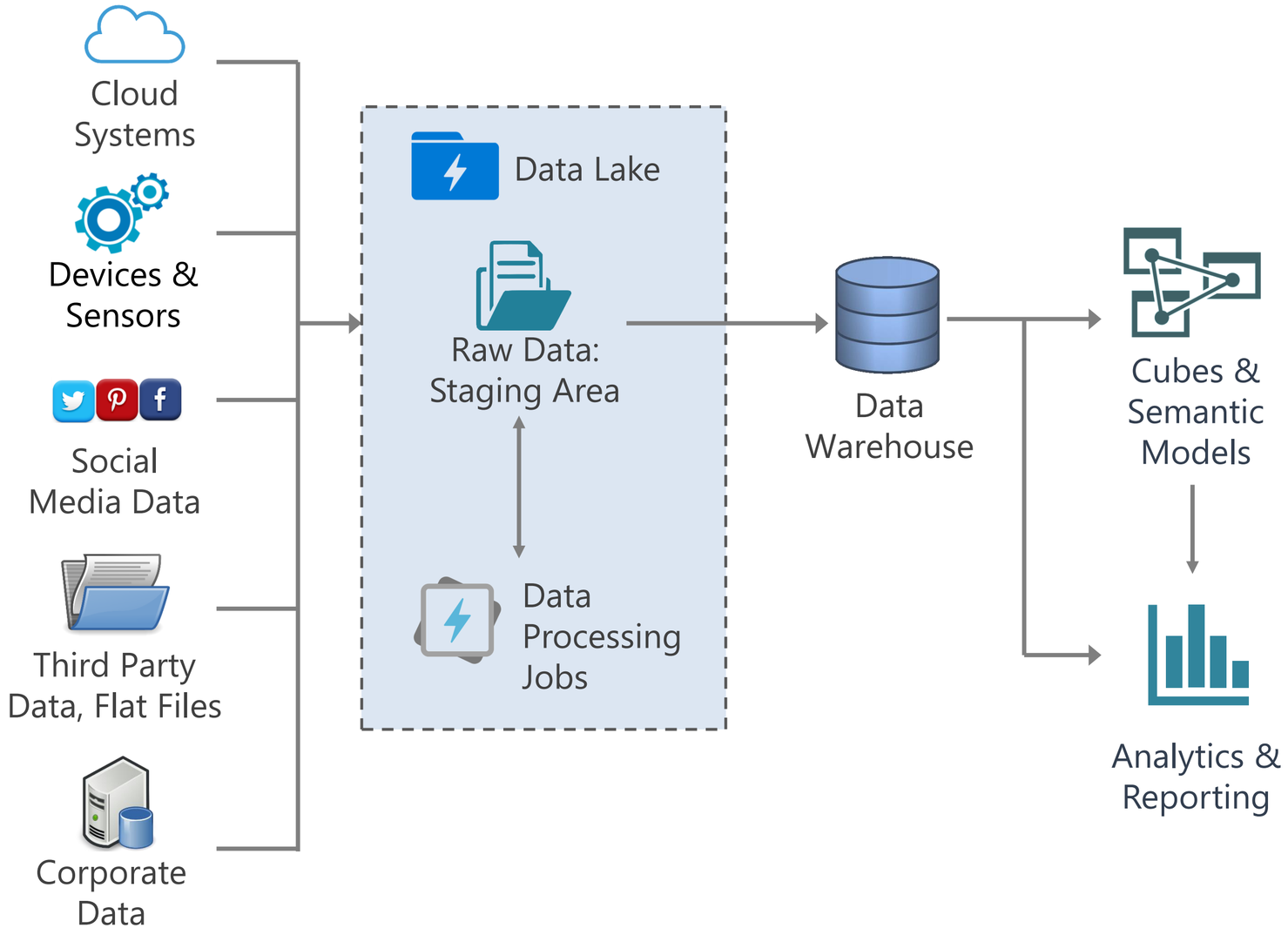
Data Science Experimentation | Hadoop Integration



- ✓ Big data clusters
- ✓ SQL-on-Hadoop solutions
- ✓ Integrate with open source projects such as Hive, Spark, Storm, Kafka, etc.
- ✓ Sandbox solutions for initial data prep, experimentation, and analysis
- ✓ Migrate from proof of concept to operationalized solution

Data Lake Use Cases

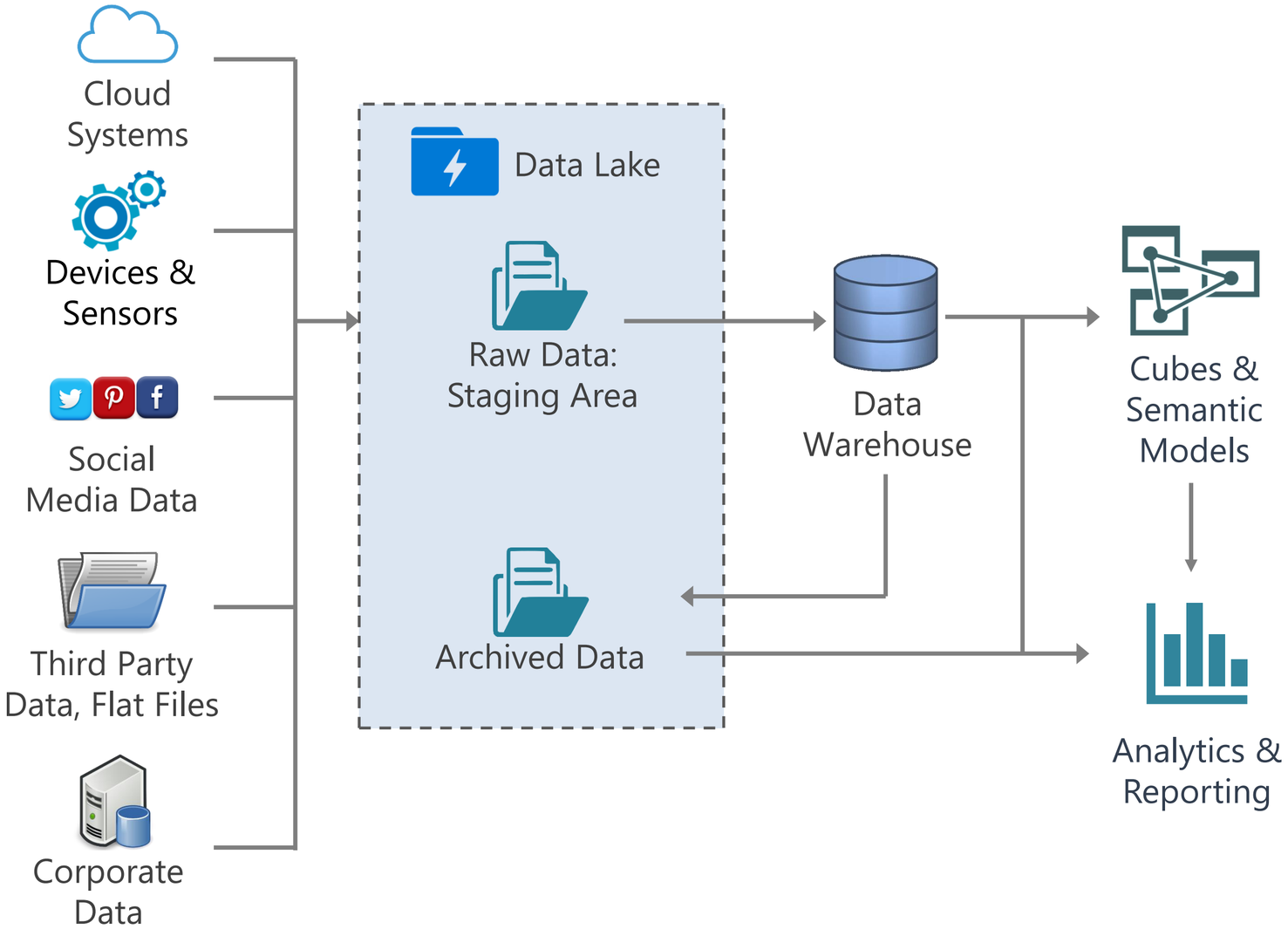
Data Warehouse Staging Area



- ✓ ELT strategy (extract>load>transform)
- ✓ Reduce storage needs in relational platform by using the data lake as landing area
- ✓ Practical use for data stored in the data lake
- ✓ Potentially also handle data transformations in the data lake

Data Lake Use Cases

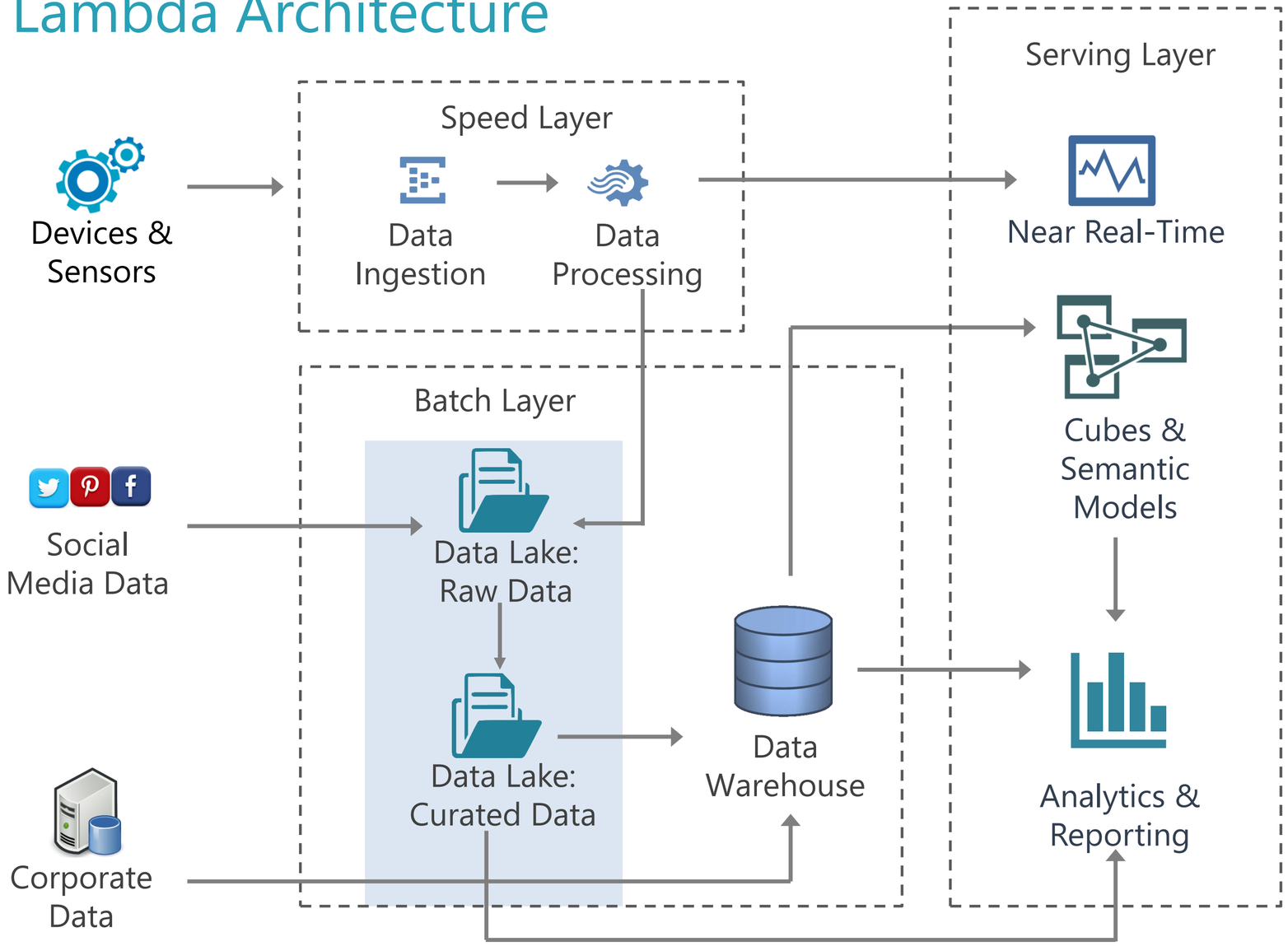
Active Archiving



- ✓ Offload aged data from data warehouse back to the data lake
- ✓ An "active archive" available for querying when needed
- ✓ Federated queries to access:
  current data in the DW + archive data in the data lake

Data Lake Use Cases

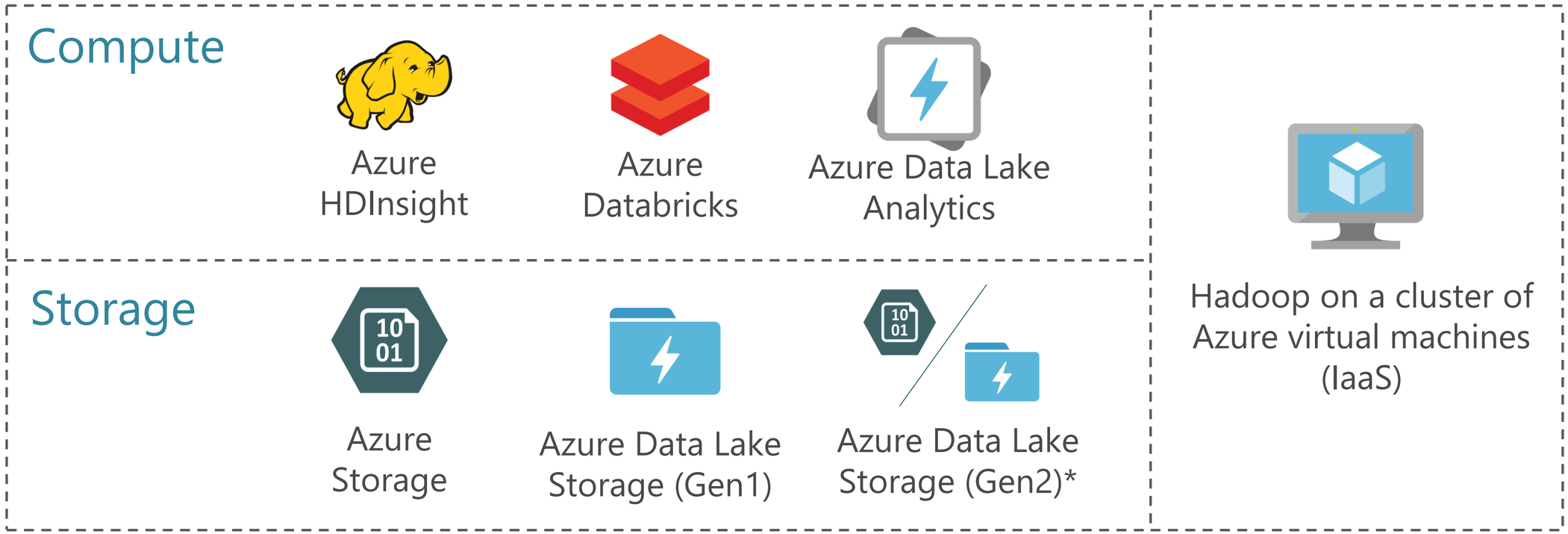
Lambda Architecture



- ✓ Support for low-latency, high-velocity data in near real time
- ✓ Support for batch-oriented operations

Big Data in Azure

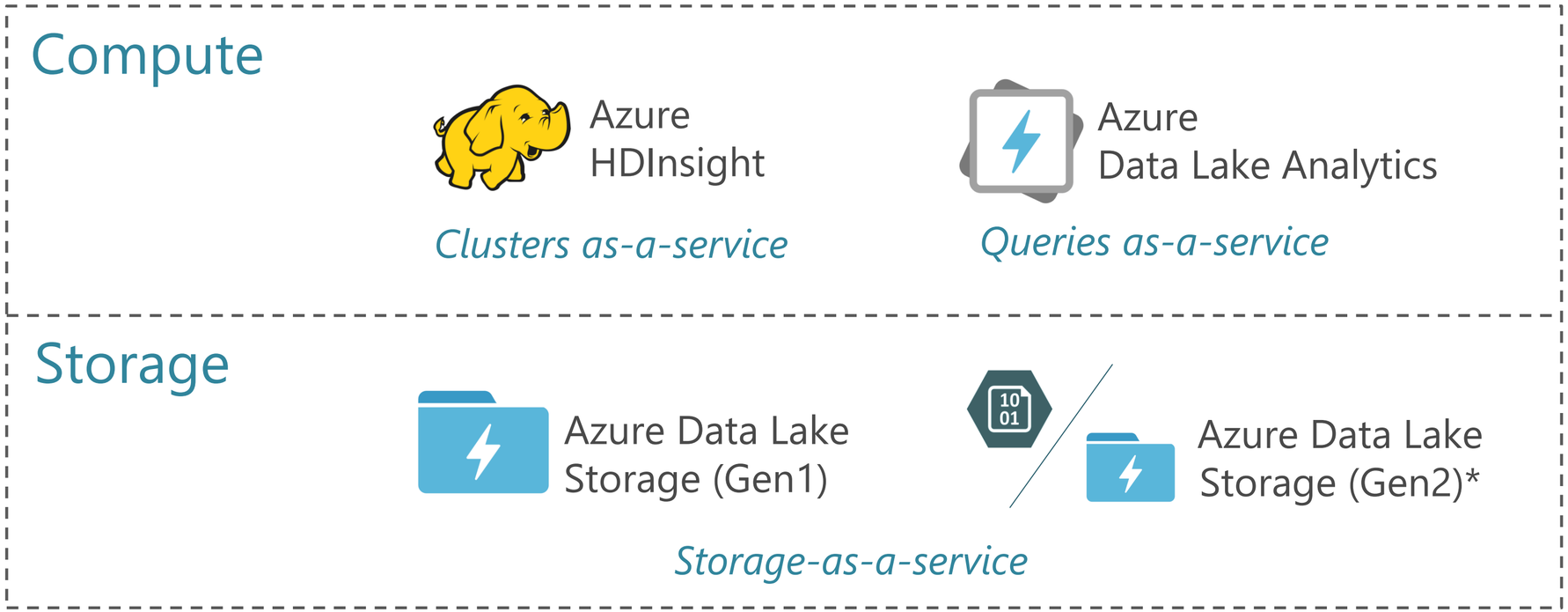
Big Data in Azure



*In Public Preview

Azure Data Lake

Azure Data Lake is a collection of the following services:

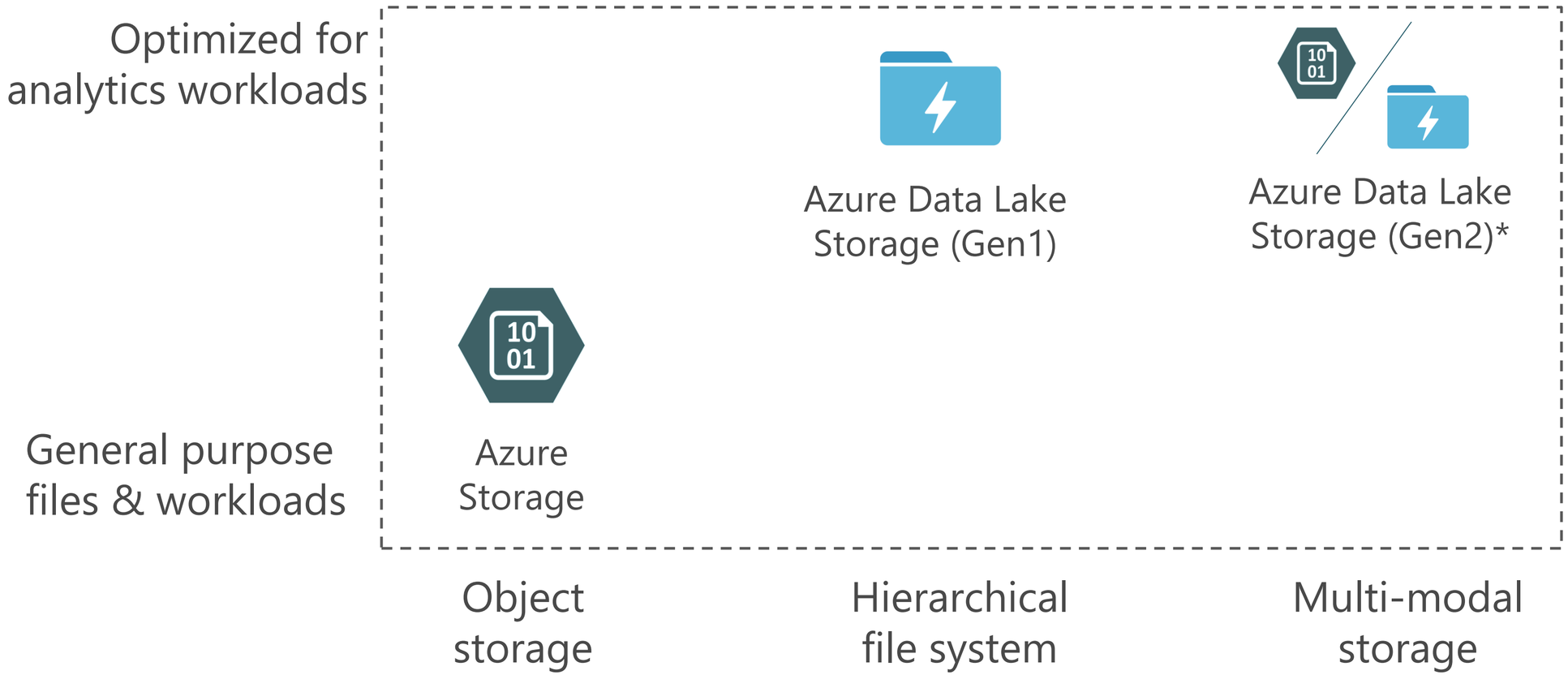


*In Public Preview

Data Storage in Azure

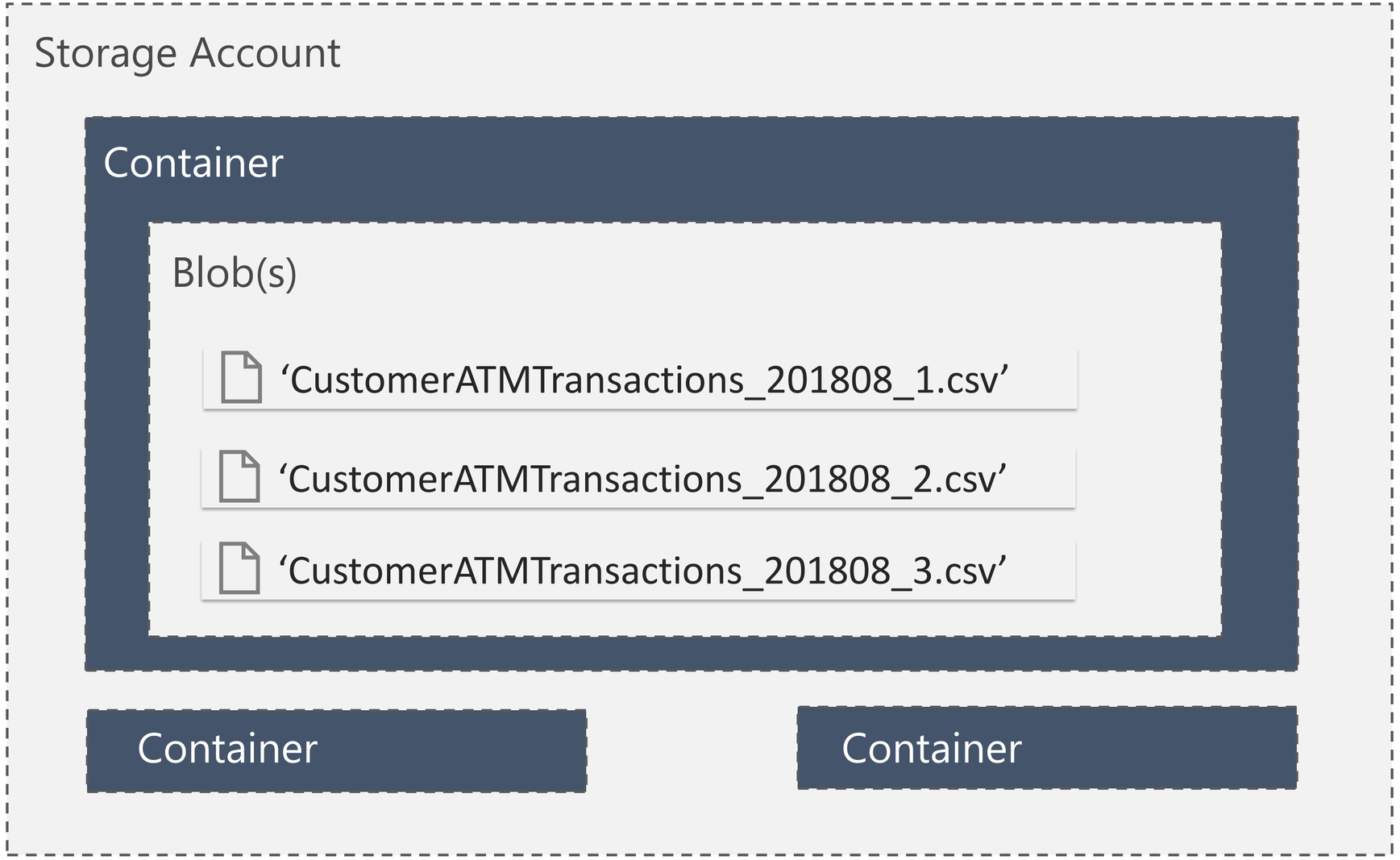
Big Data in Azure: Storage

(Excluding relational and NoSQL data storage options)



*In Public Preview

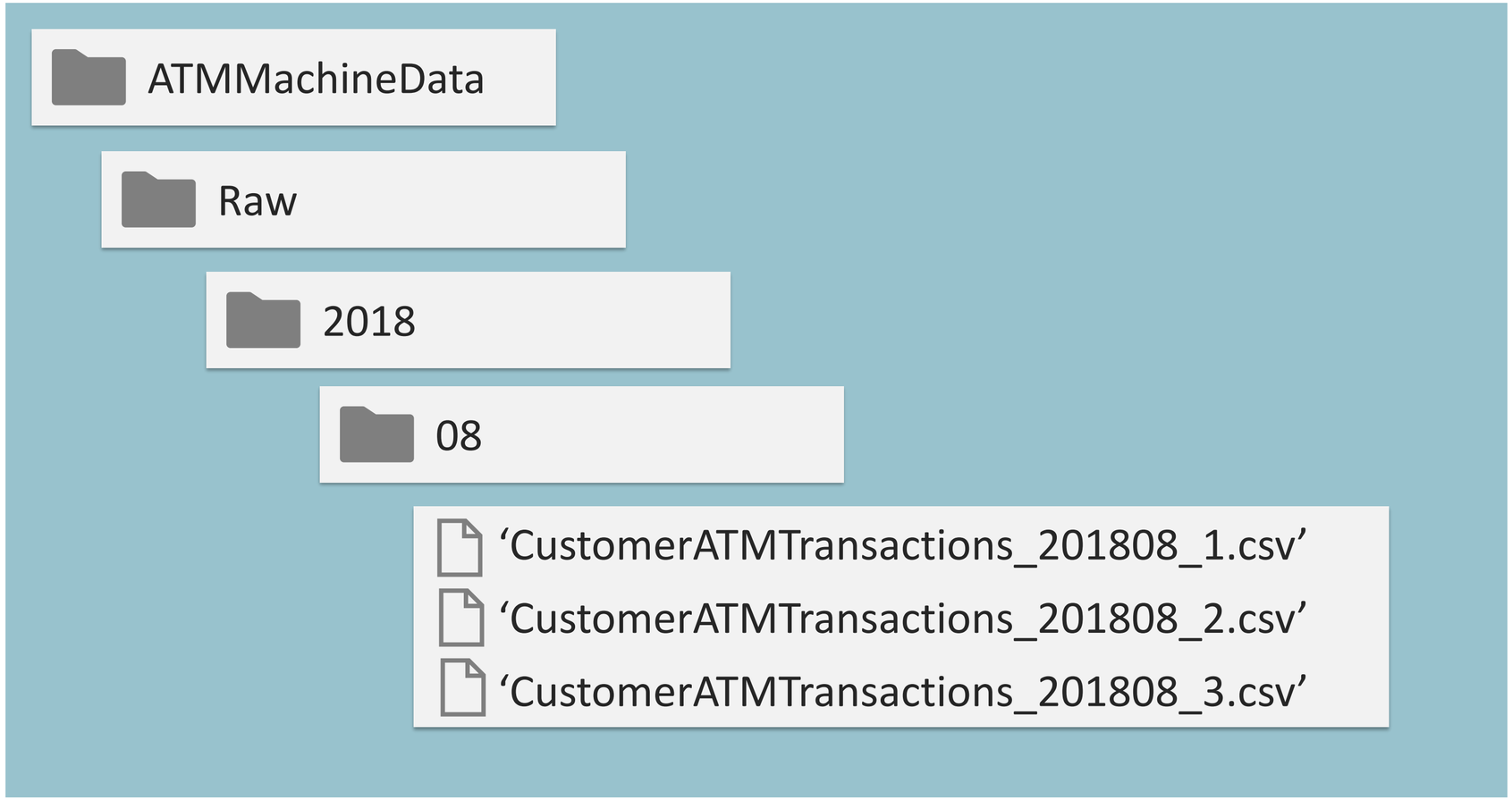
Azure Storage



Object-based storage manages data as discrete units.

Folders are part of the URI, but they’re merely simulated. There is no folder-level security, nor folder-specific performance optimizations.

Azure Data Lake Storage Gen 1



Hierarchical file-based storage supports nesting of files within folders.

Folder-level security can be implemented, as well as certain performance optimizations.

Previously – An Either/Or Decision

Object store

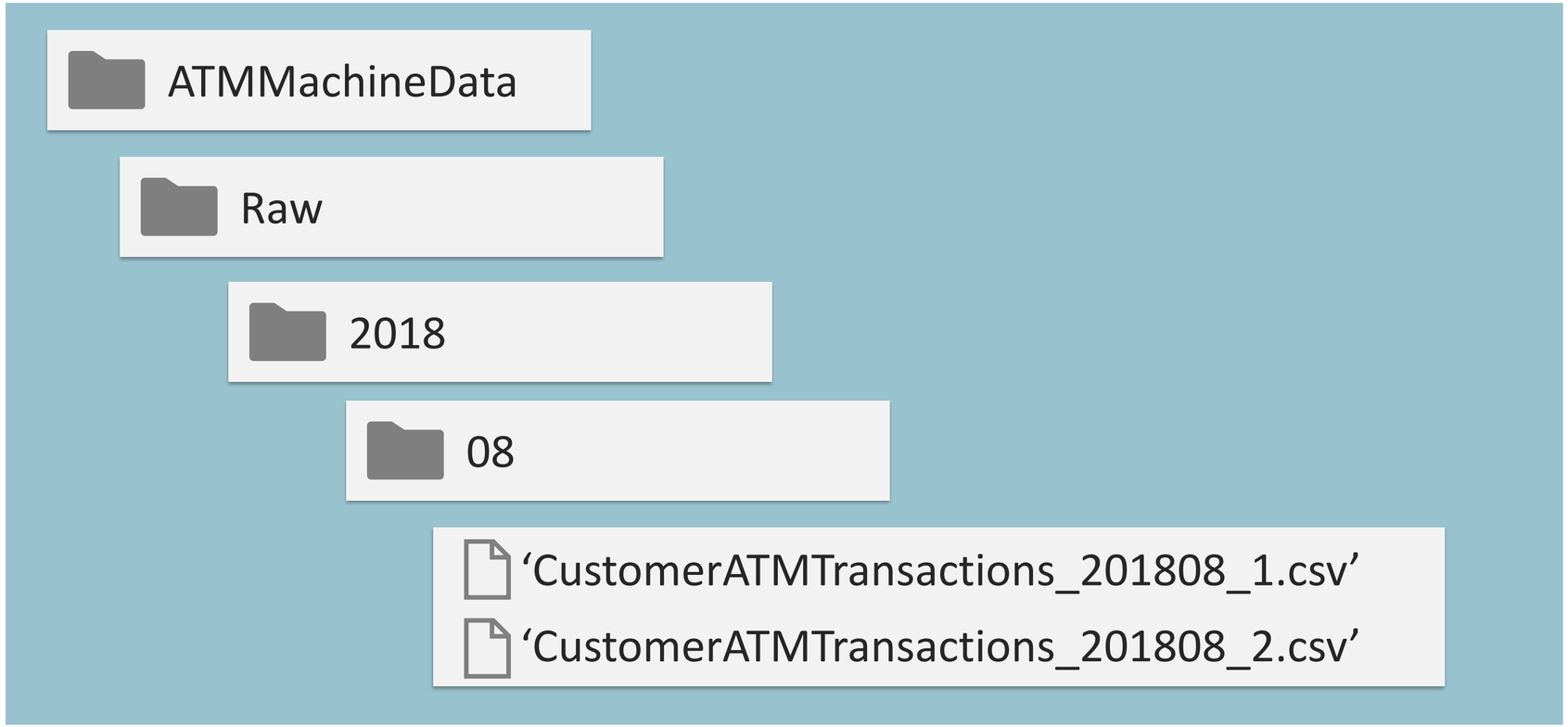
‘/ATMMachineData/RawData/2018/08/CustomerATMTransactions_201808_1.csv’

‘/ATMMachineData/RawData/2018/08/CustomerATMTransactions_201808_2.csv’

Azure Storage

-OR-

Hierarchical storage



Azure Data Lake Storage Gen 1

Deciding Between Storage Services

Azure Storage

General purpose object store (containers > blobs)

Addtitional features not available in ADLS Gen1

- Data replication and redundancy options
- Available in all regions globally
- Hot/cold/archive tiers
- Lifecycle management (in preview at this time)
- Metadata (key/value pairs)

ADLS (Gen 1)

Hierarchical file system (folders > files)

Optimized for analytics workloads

- Hadoop and big data optimizations
- Parallelized reads and writes
- Scaled out over multiple nodes
- Low latency writes with I/O throughput
- Fine-grained security via access control lists

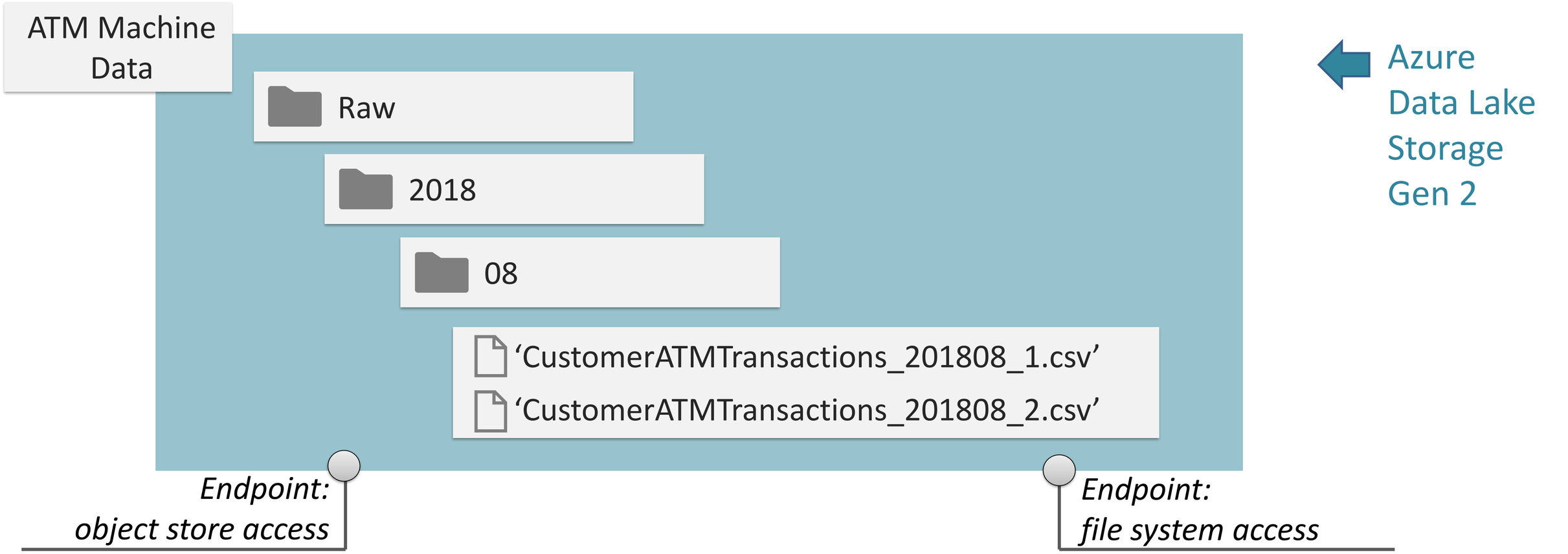
ADLS (Gen 2)

- Multi-modal combining features from both of the above
- Not a separate service: Azure Storage with new features
- Enable the “hierarchical namespace” (HNS) to use

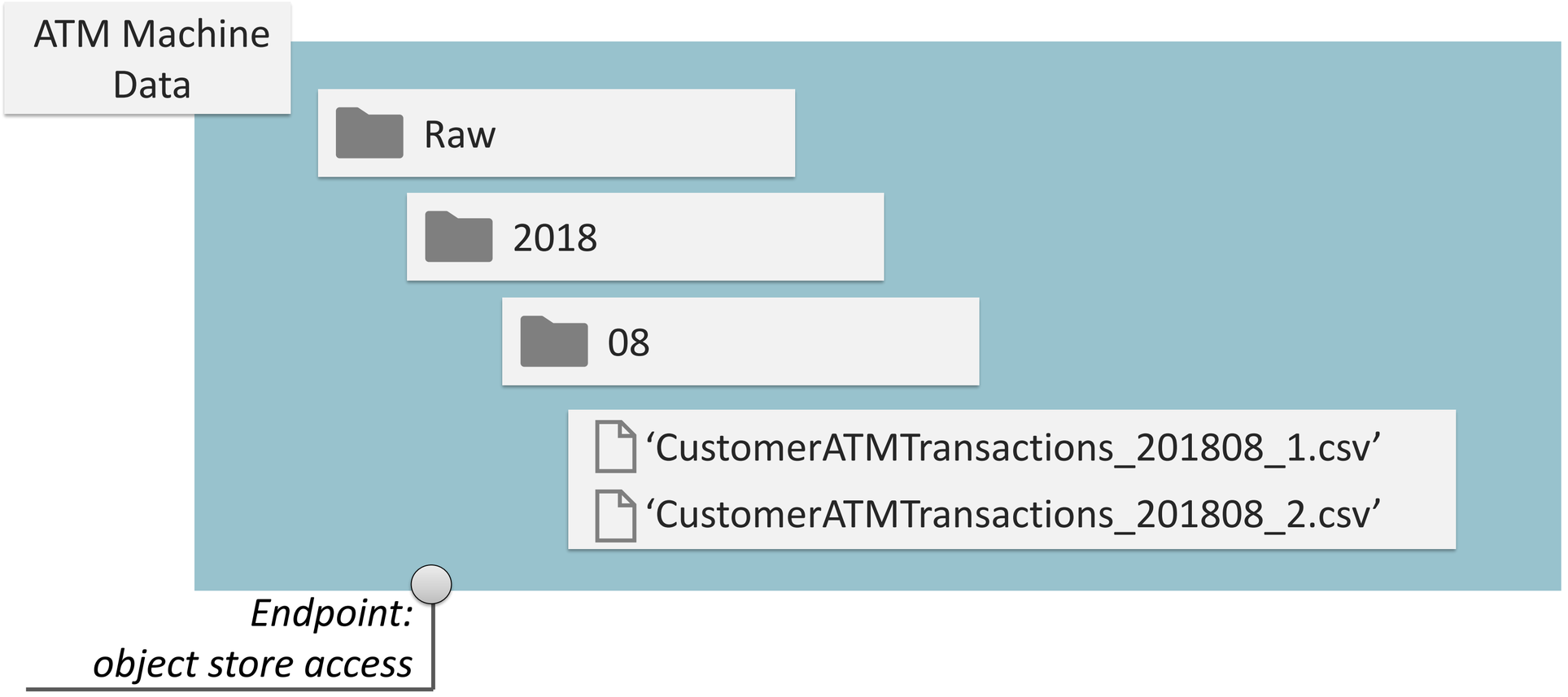
New Multi-Modal Storage Option: ADLS Gen 2

The long-term vision:

The data is stored once, and accessed through either endpoint based on use case / data access pattern. Files & folders are ‘first class citizens.’

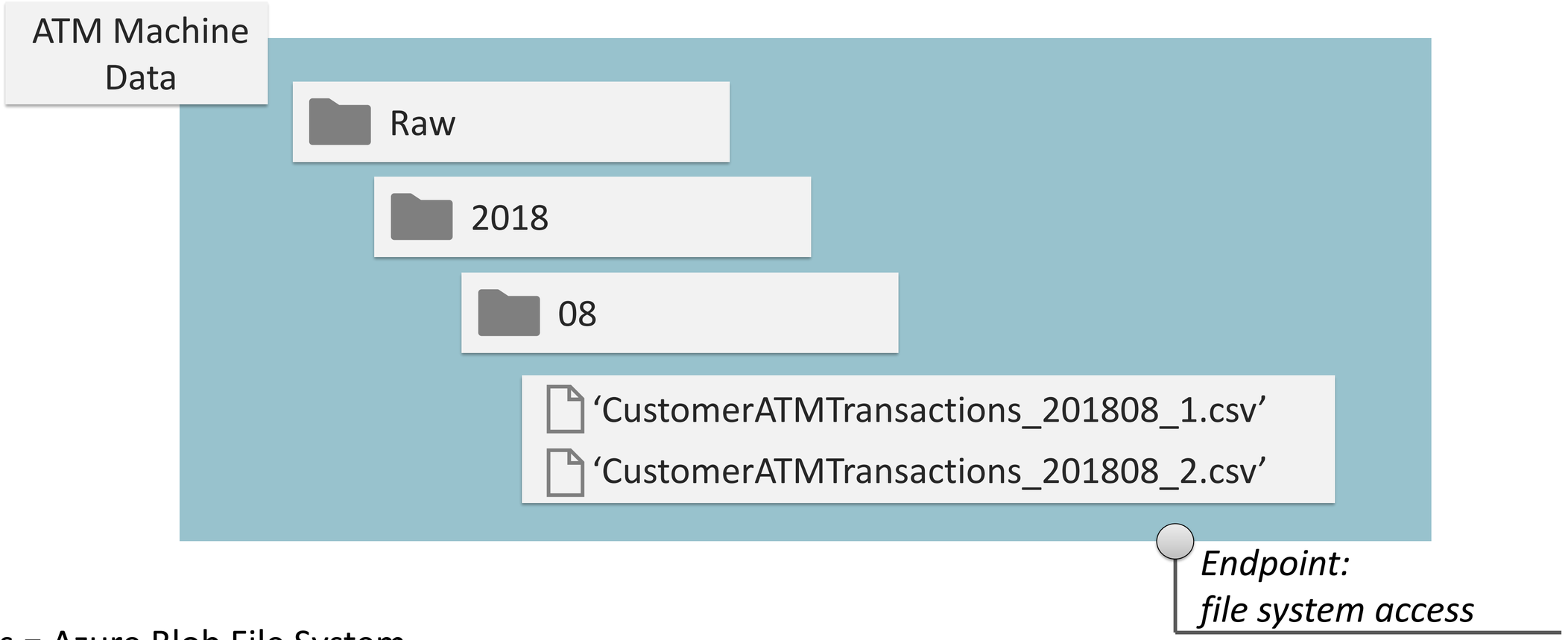


Object Store Endpoint: wasb[s]



wasb[s]://containername@accountname.blob.core.windows.net/raw/2018/08/CustomerATMTransactions_2018_1.csv

File System Endpoint: abfs[s]



afbs = Azure Blob File System

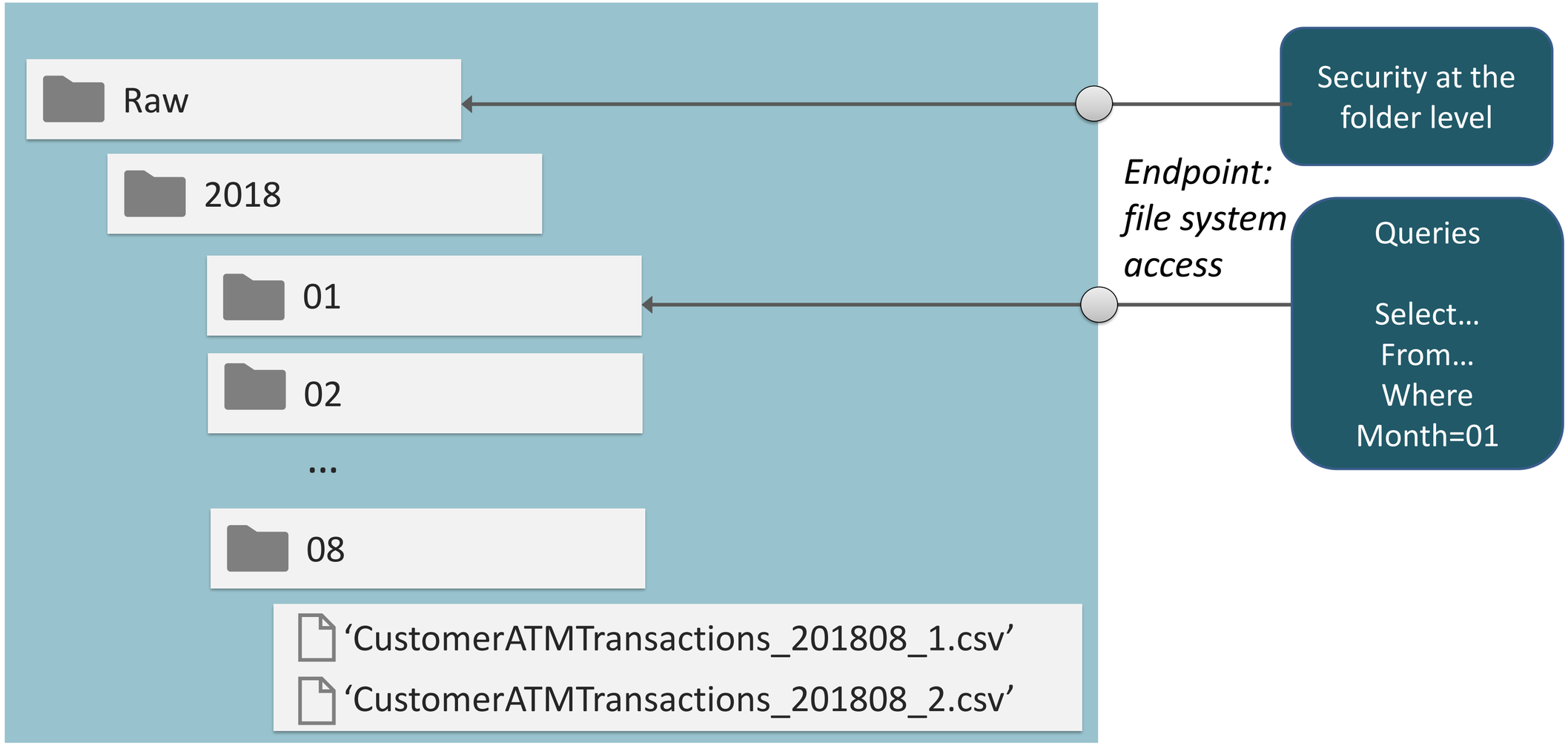
abfs[s]://filesystemname@accountname.dfs.core.windows.net/raw/2018/08/CustomerATMTransactions_2018_1.csv

afbs is the driver
abfss = SSL

dfs is the endpoint

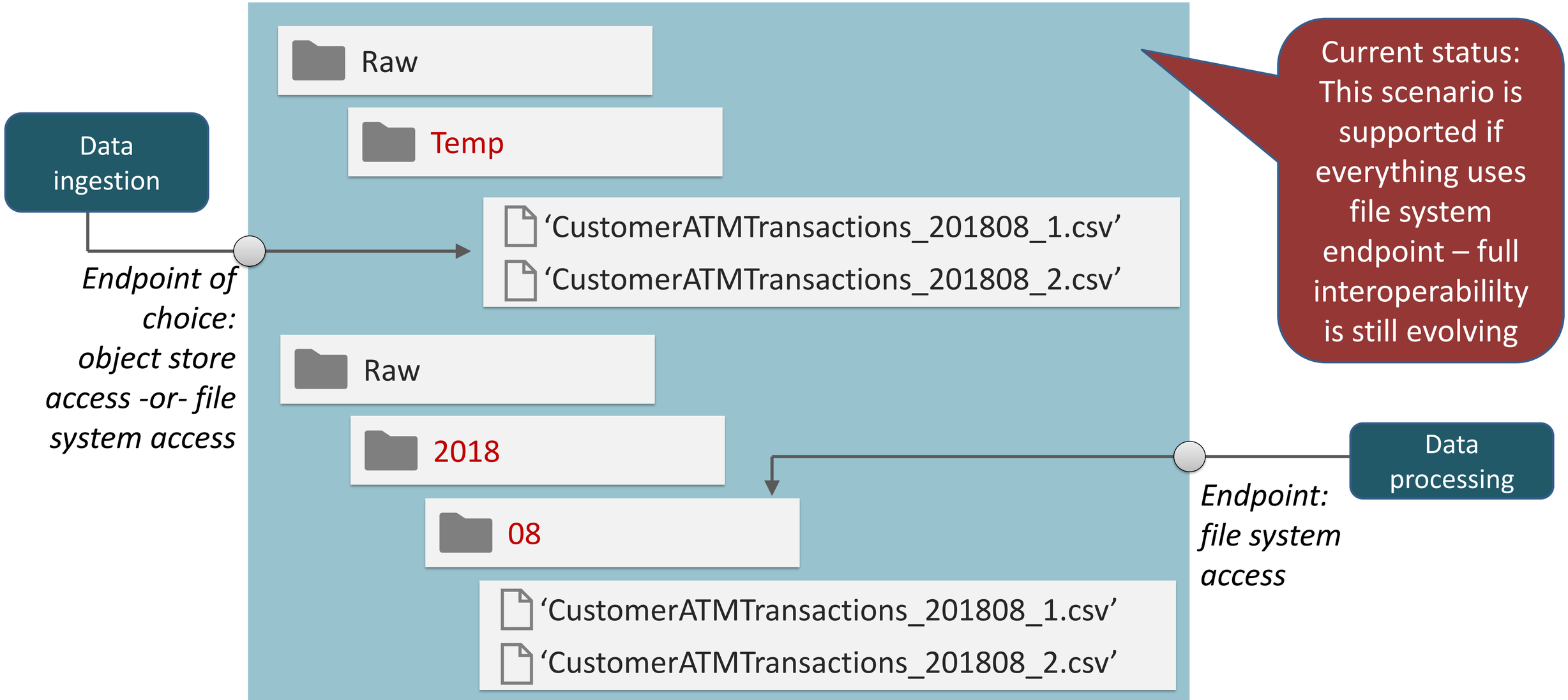
Multi-Modal Advantages with ADLS Gen 2 – Example 1

Leverage partition scans & partition pruning to improve query performance:



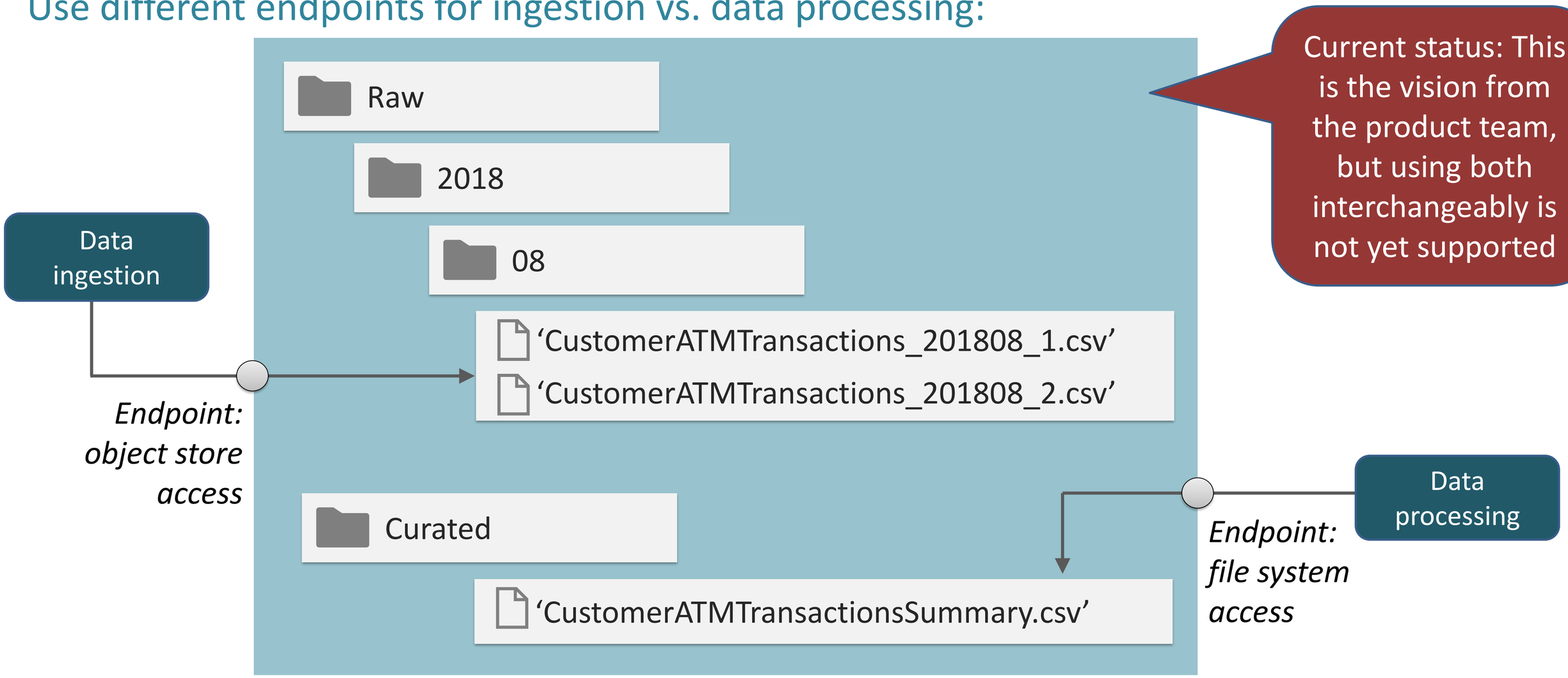
Multi-Modal Advantages with ADLS Gen 2 – Example 2

Metadata-only changes with significantly better performance using the file system endpoint:



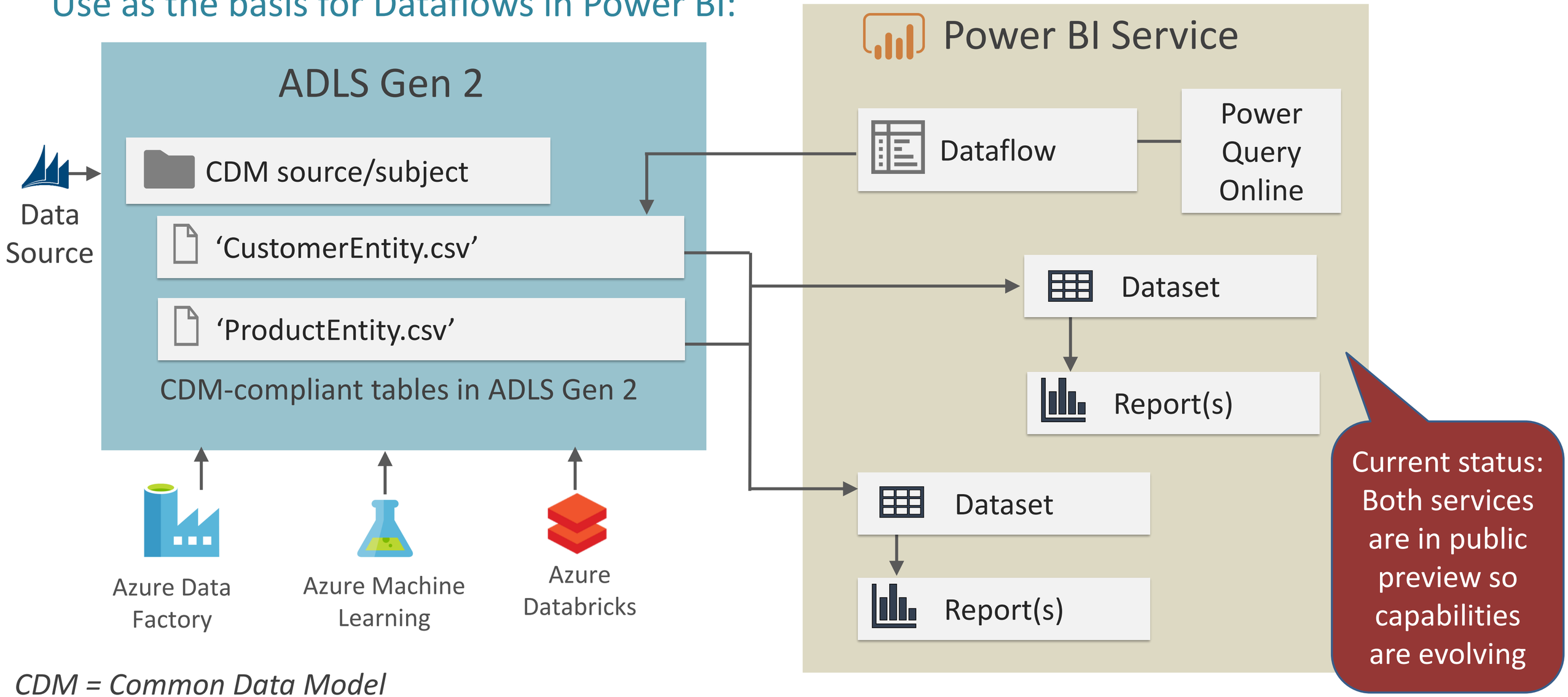
Multi-Modal Advantages with ADLS Gen 2 – Example 3

Use different endpoints for ingestion vs. data processing:



Multi-Modal Advantages with ADLS Gen 2 – Example 4

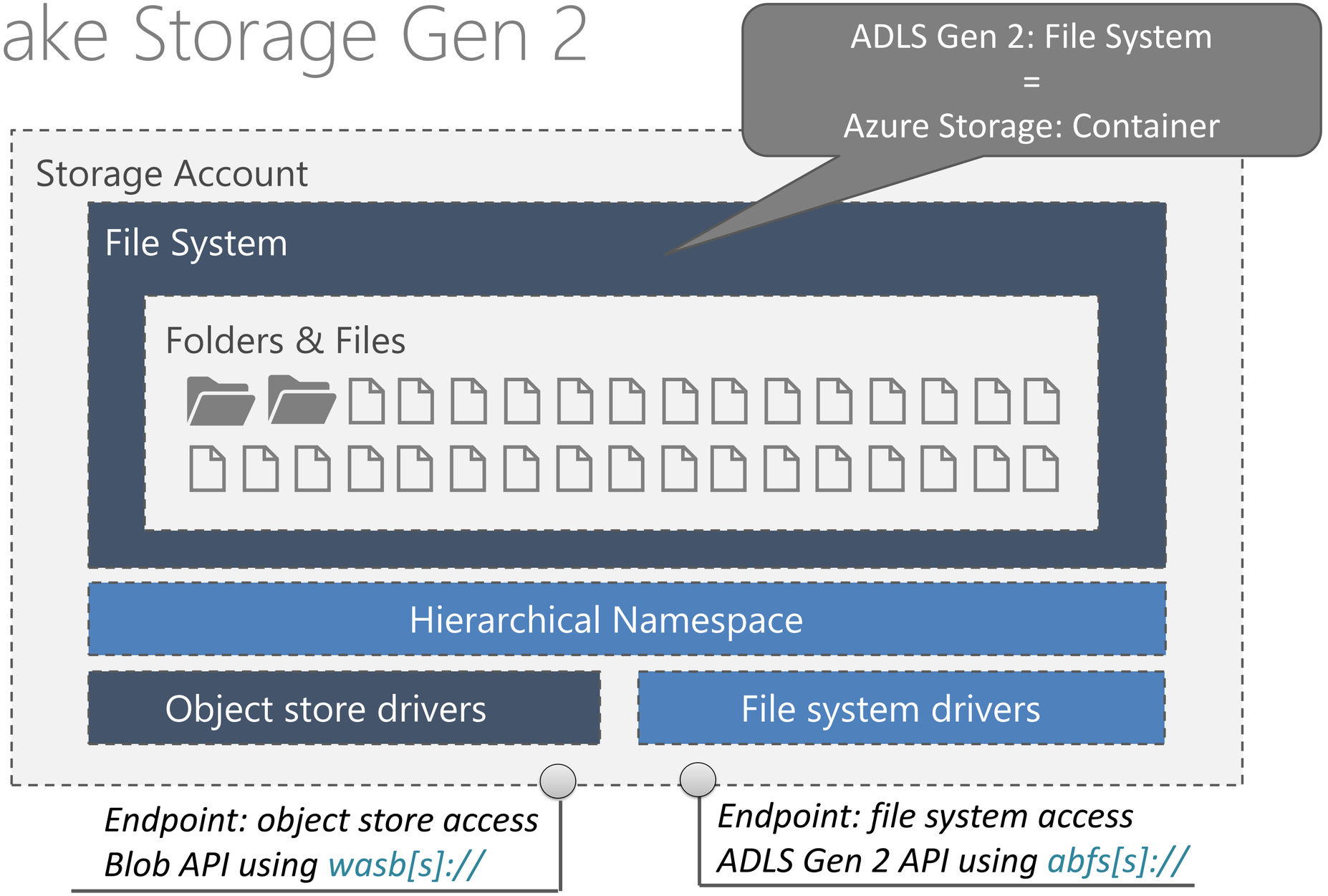
Use as the basis for Dataflows in Power BI:



CDM = Common Data Model

Azure Data Lake Storage Gen 2

ADLS Gen 2 = Azure Storage with the Hierarchical Namespace (HNS) enabled.



Adapted from: https://azure.microsoft.com/en-us/blog/a-closer-look-at-azure-data-lake-storage-gen2/

When to **Disable** the Hierarchical Namespace?

- ✓ General purpose file storage such as backups and VHDs
- ✓ Classic object store use cases which do not benefit from hierarchical storage or a high degree of organization (ex: image storage)
- ✓ Custom apps, APIs, or legacy systems which only use the Blob API and/or are unaware of file system semantics

The HNS is enabled at the storage account level.
Product team says there's no harm or performance difference
(even for raw I/O) if the HNS is enabled but not used.
However, you'll pay extra cost (~30% extra) on every transaction if HNS is enabled.

Summary: Goals of ADLS Gen 2

- *Unify* the data lake story on Azure
- Take advantage of the *best of both feature sets* (object storage & hierarchical storage)
- *Multiple protocol endpoints* to allow flexibility for use cases
- *Avoid duplicating data* for specific use cases or tools ('islands of data')
- *Overcome limitations of object storage* (ex: metadata only operations)
- Improved *performance for big data analytics*
- Implement *full data lifecycle* and *data policies*
- *Low cost* with high-performing *throughput*
- Integrate with the new '*dataflows*' functionality in Power BI

Summary: Current State of the Storage Options

Azure Storage

- ✓ Still a very valid option for object store workloads

ADLS (Gen 1)

- ✓ Fully supported in existing regions
- ✓ No new features

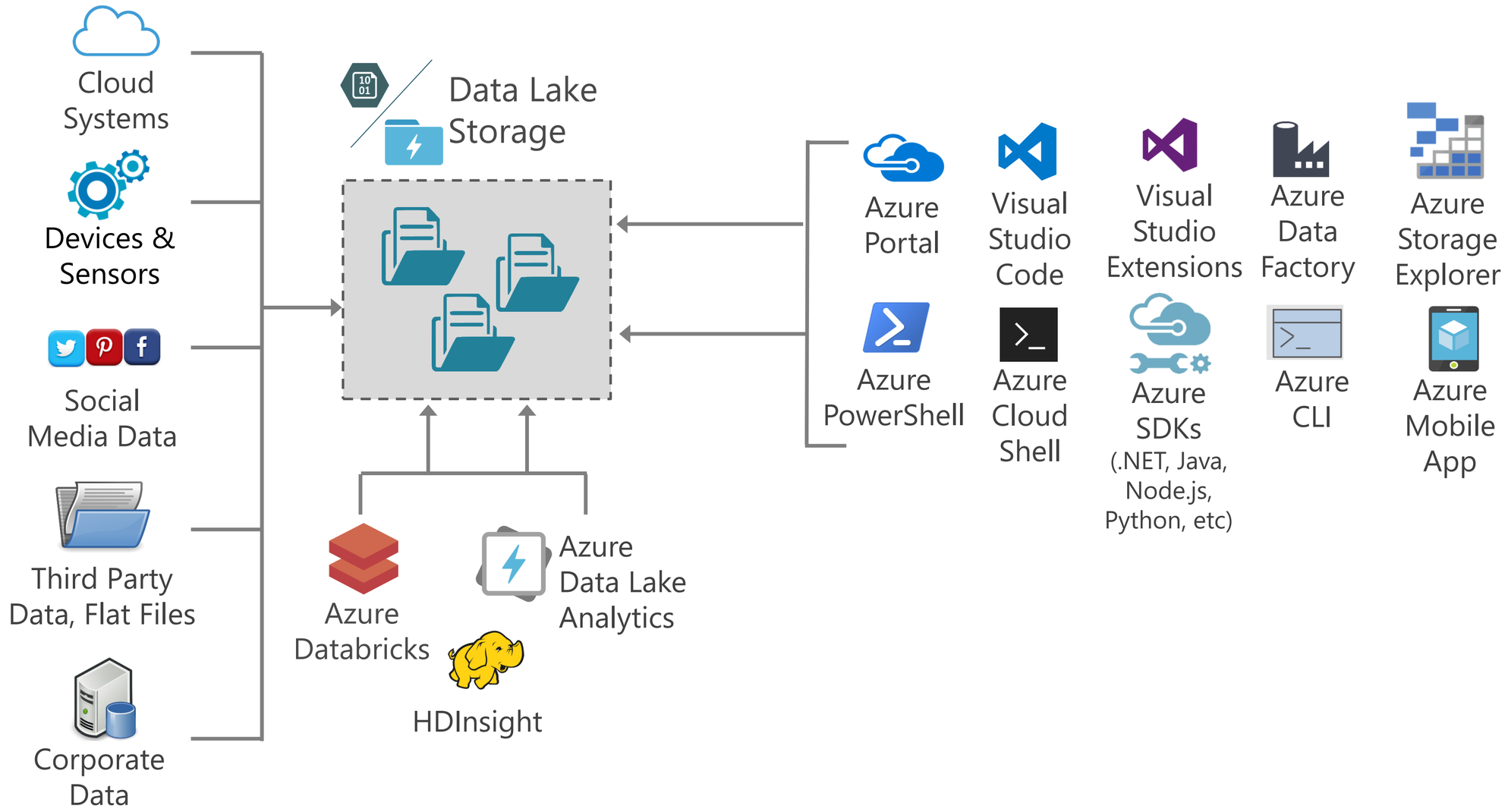
ADLS (Gen 2)

- ✓ In public preview
- ✓ Feature support is evolving over time

Demo:
ADLS Gen 2

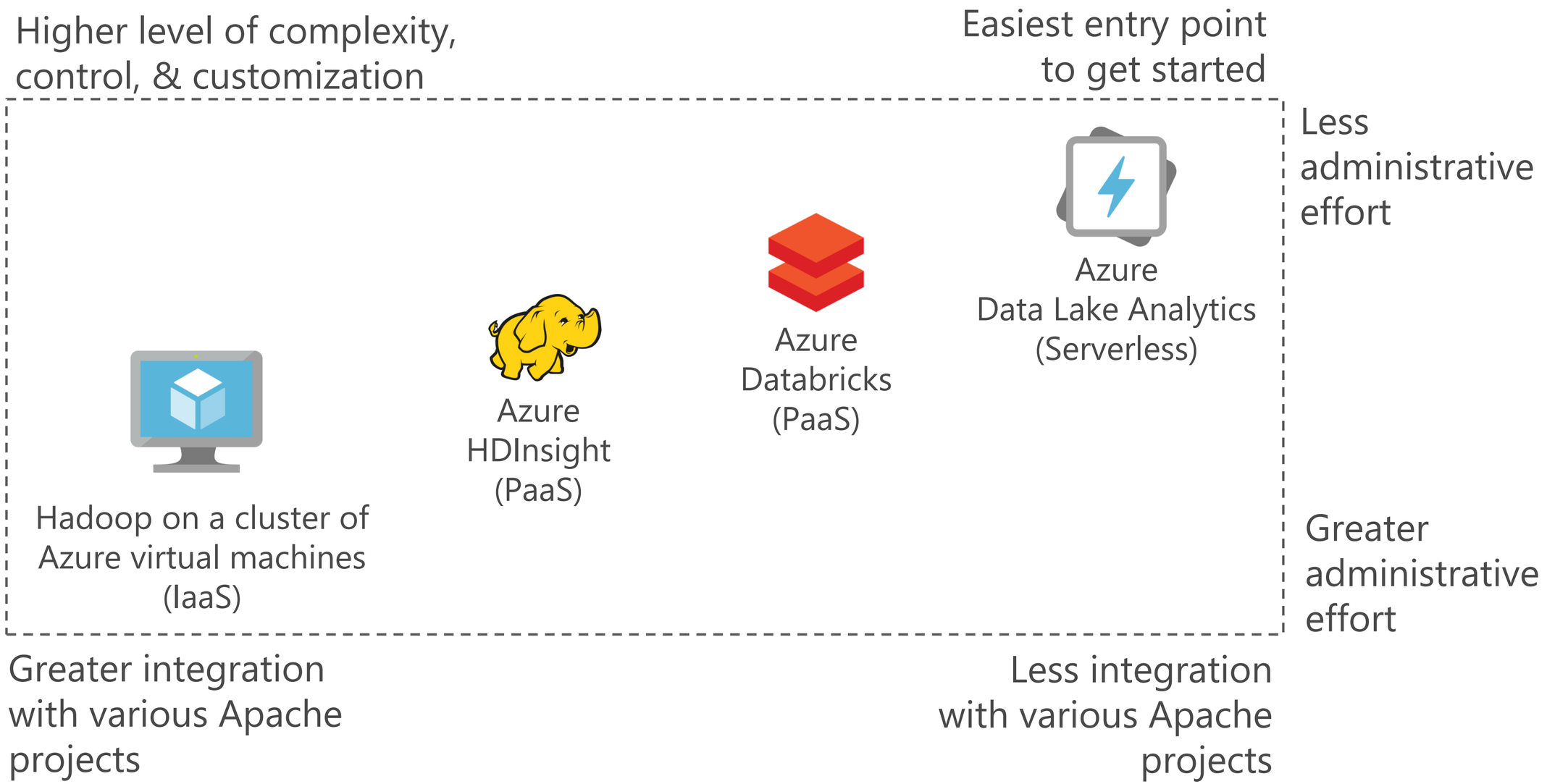
Compute in Azure

Compute Services and Data Management

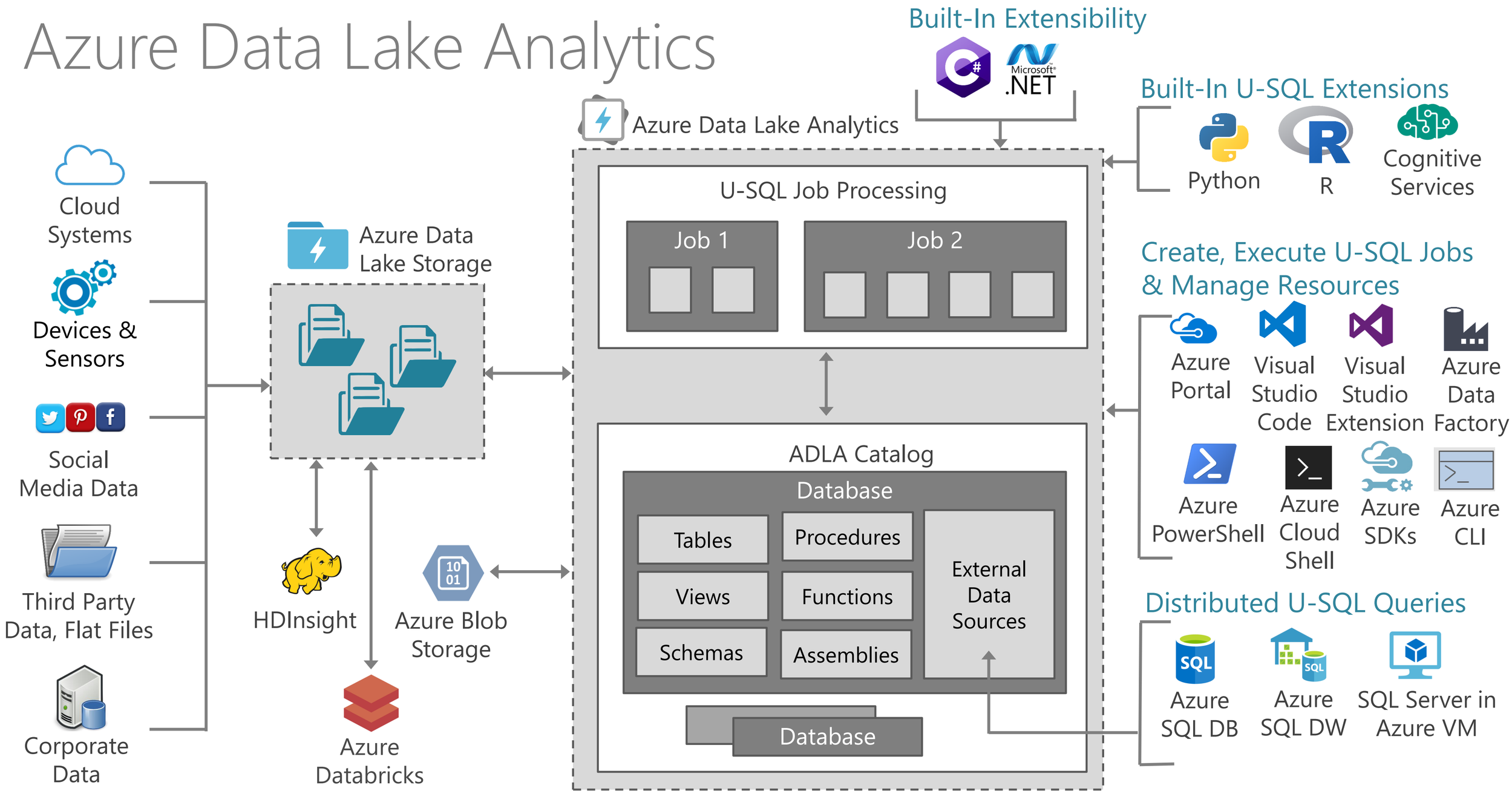


*In Public Preview

Big Data in Azure: Compute



Azure Data Lake Analytics

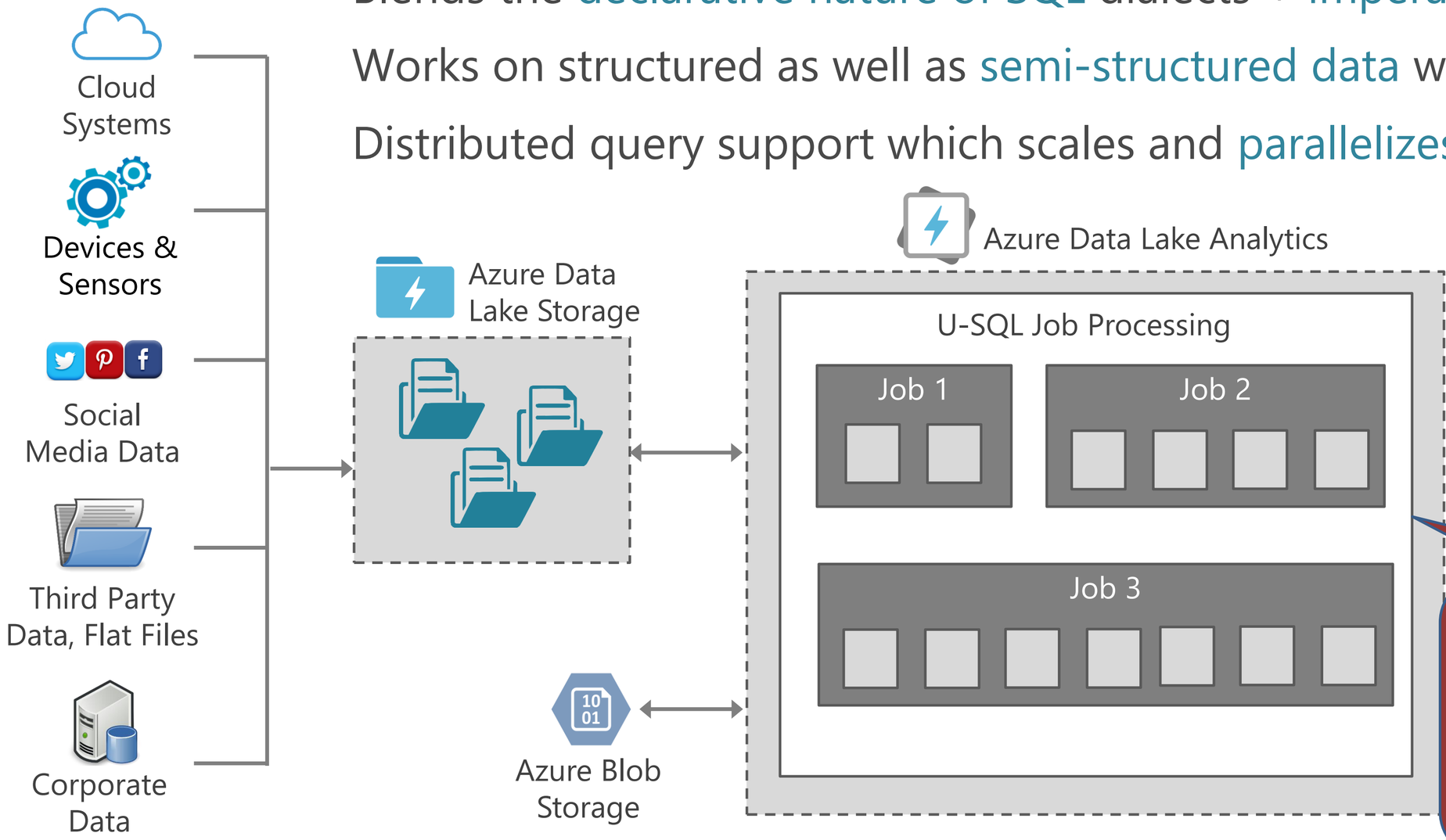


U-SQL: Unified SQL

Blends the declarative nature of SQL dialects + imperative nature of C#.

Works on structured as well as semi-structured data without reformatting.

Distributed query support which scales and parallelizes across nodes.



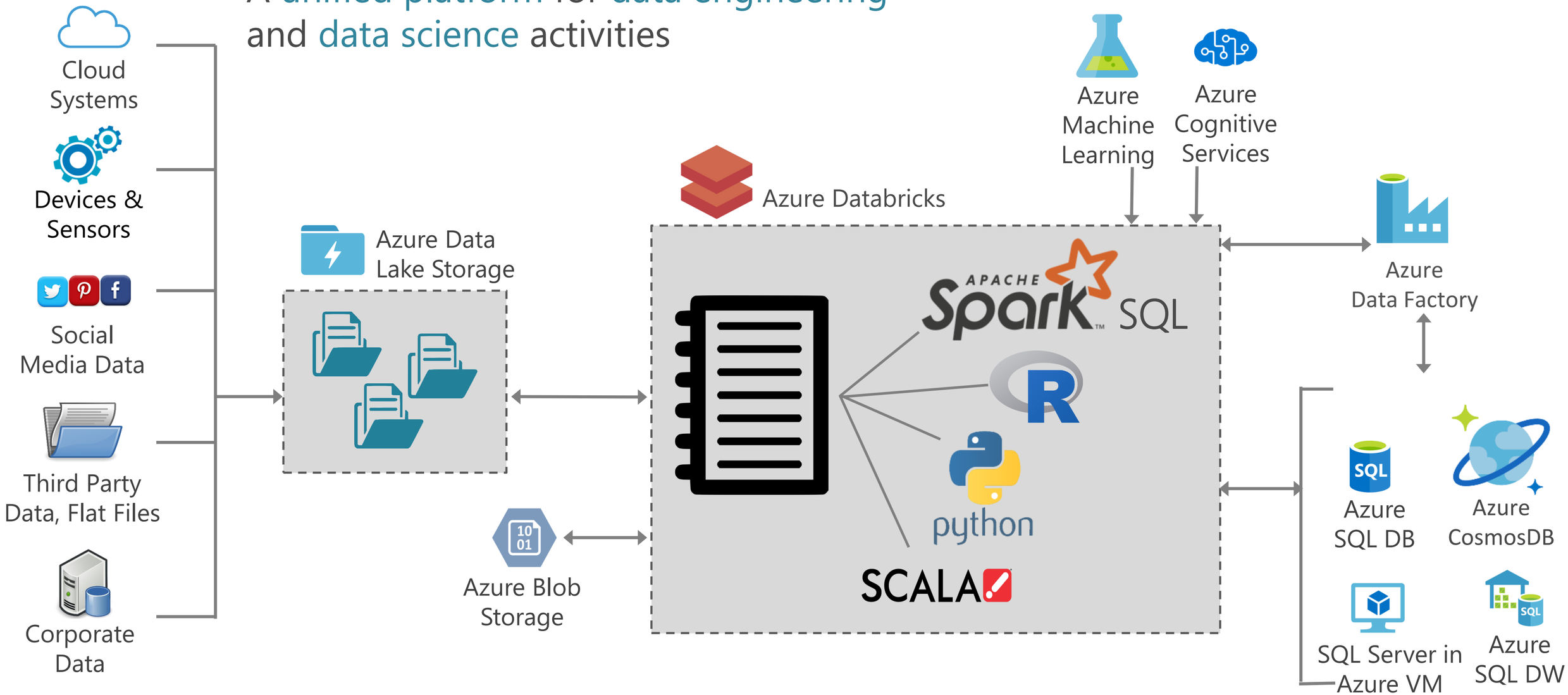
Supported:

- Batch processing queries
- File output to ADLS Gen 1 or Blob Storage

Current status:
No integration with ADLS Gen 2 at this time (it requires the flat object store endpoint)

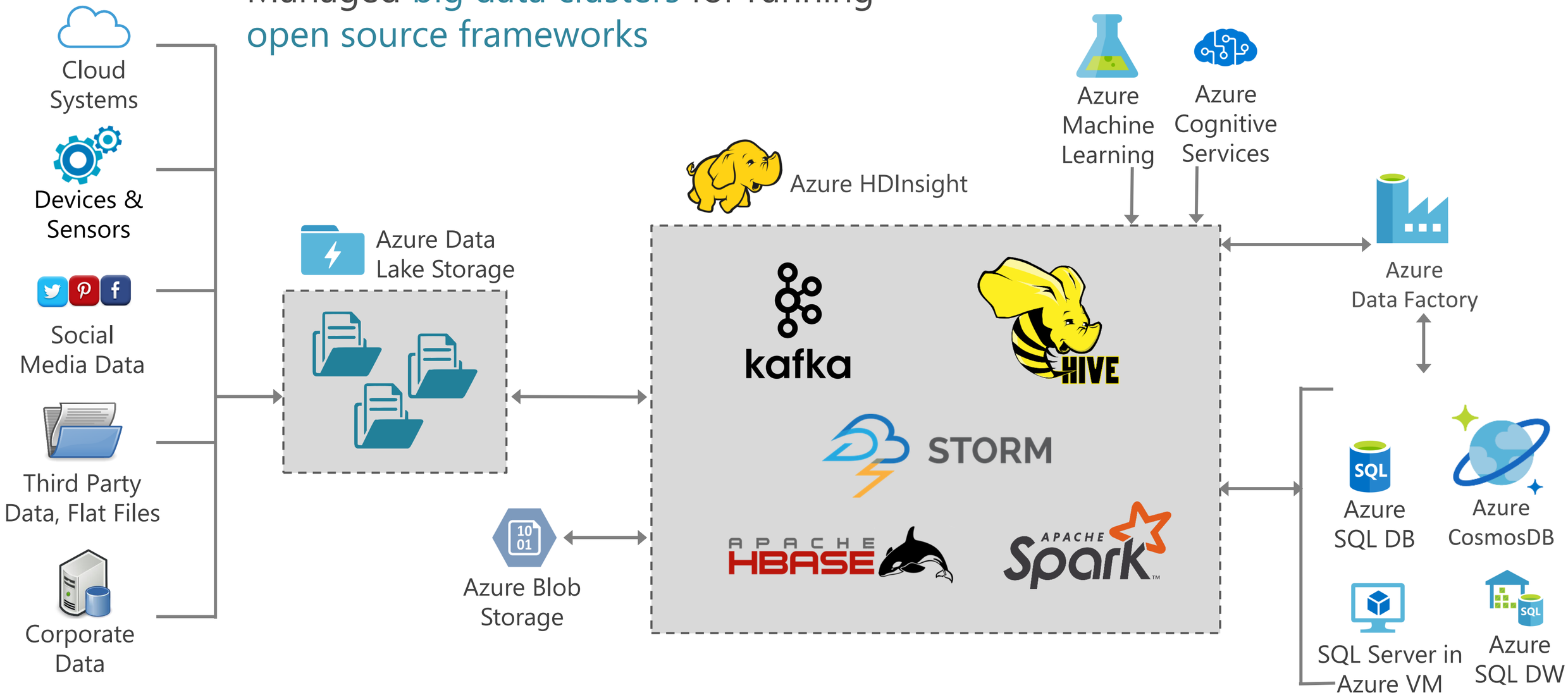
Azure Databricks

A unified platform for data engineering and data science activities



Azure HDInsight

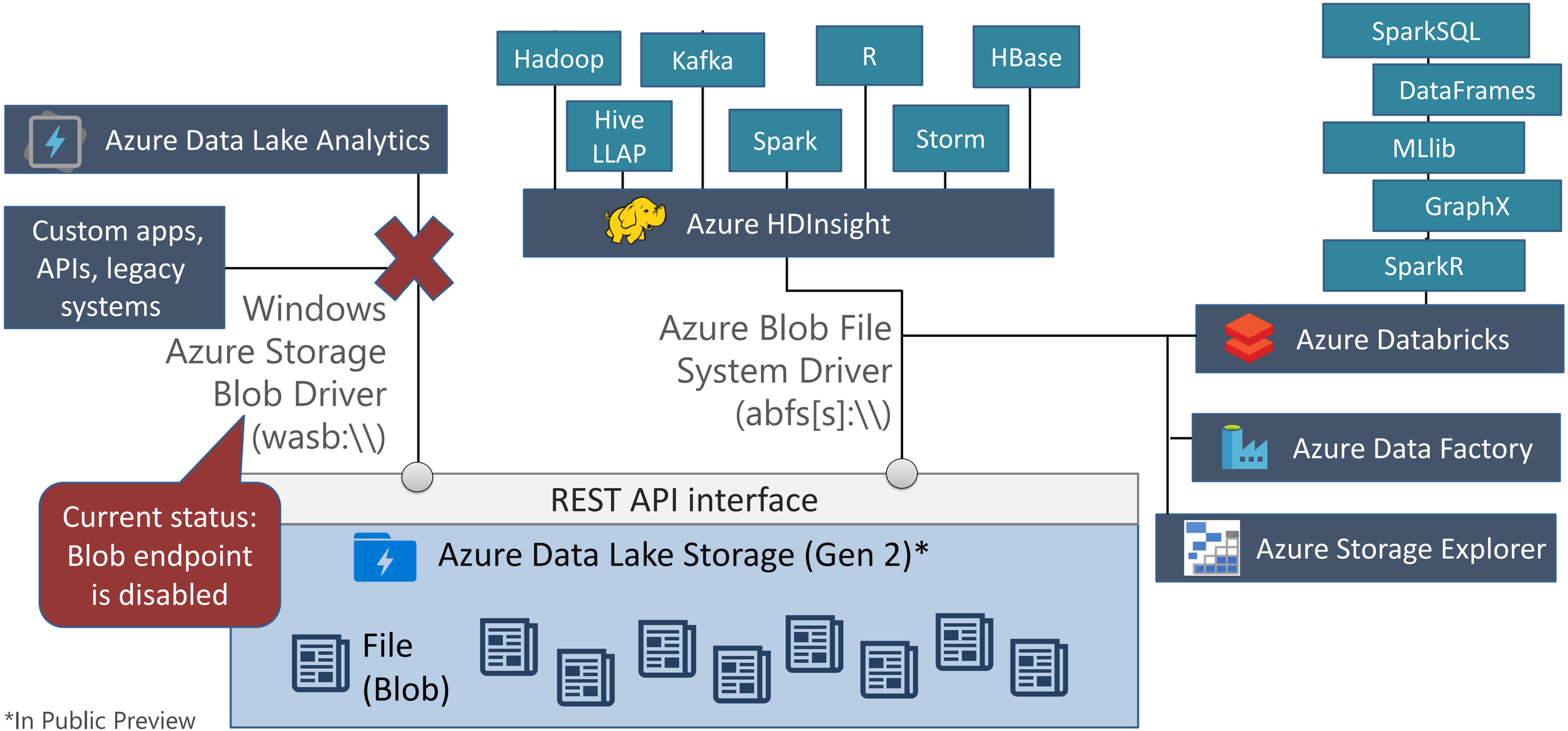
Managed big data clusters for running open source frameworks



Deciding Between Compute Services

	Hadoop VM	HDInsight	Databricks	ADLA
Type:	IaaS	PaaS	PaaS	Serverless
Purpose:	Running your own cluster of Hadoop virtual machines	Running a managed big data cluster	Running a managed, optimized Spark framework	Running U-SQL batch jobs
Suitable for:	Full control over everything; investment in distributions such as Hortonworks, Cloudera, MapR	Integration with open source Apache projects and/or greater control over clusters	Collaborative notebooks; easier deployments; utilizing Spark in a variety of ways	Focus on running individual jobs (scripts) rather than managing a cluster
			1st choice	Caution

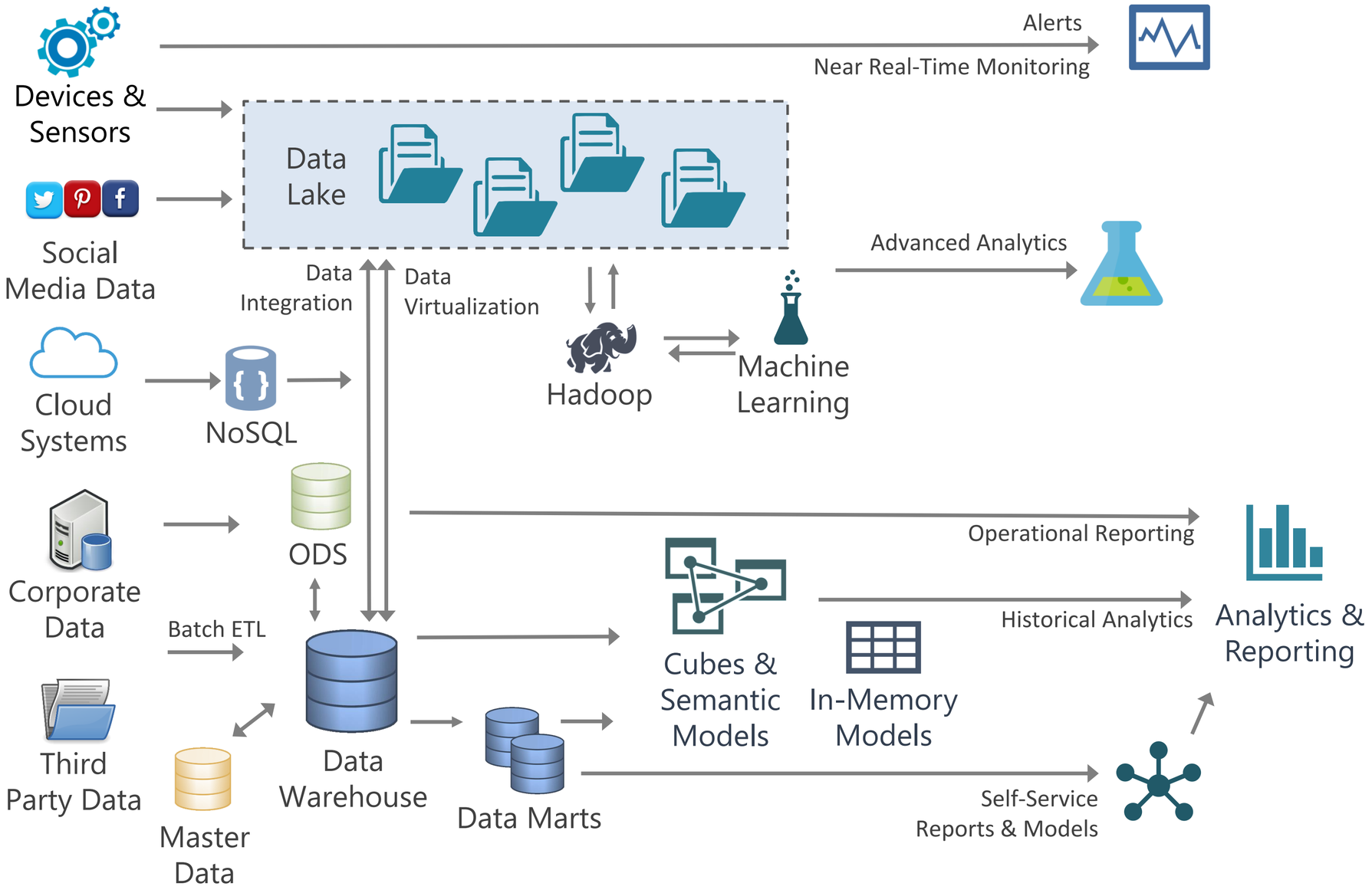
Interacting with ADLS Gen 2



*In Public Preview

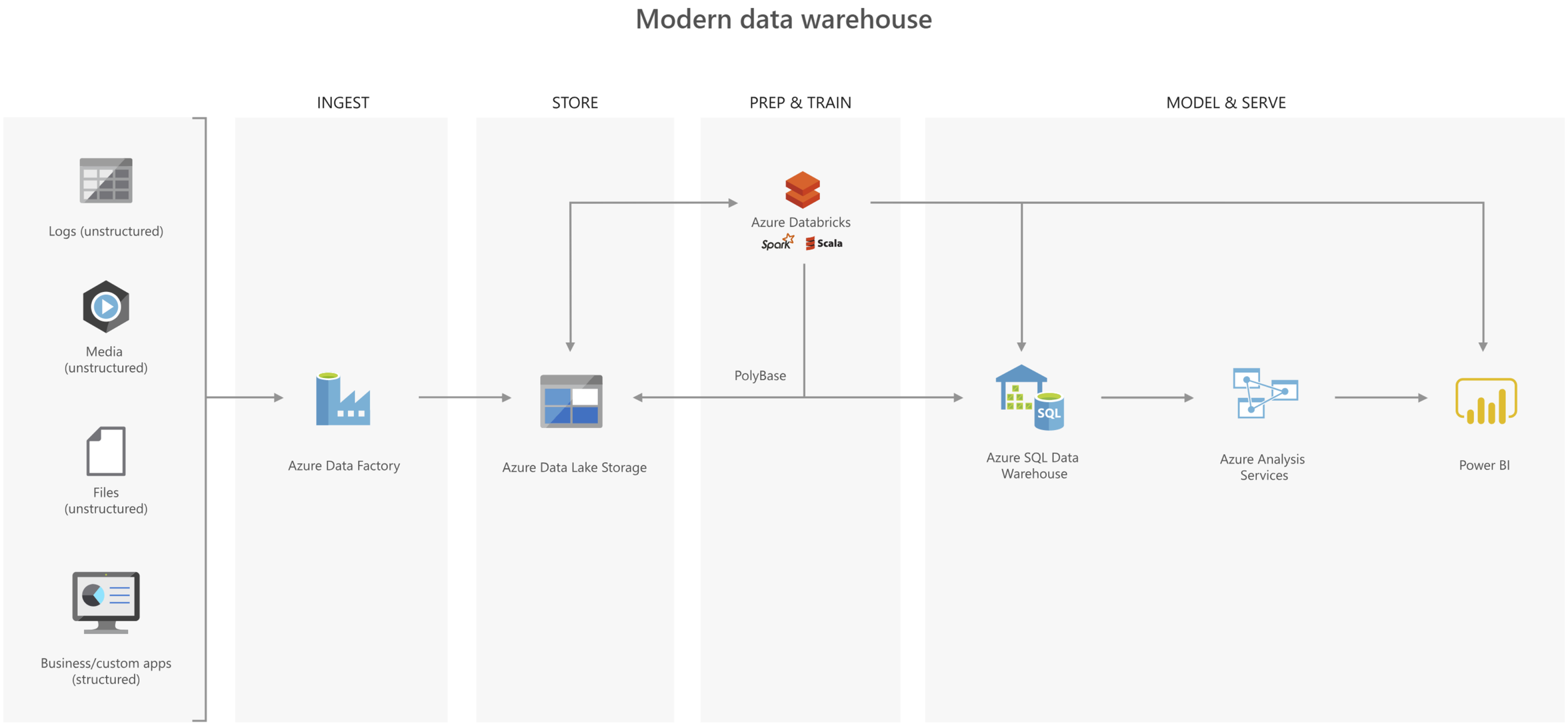
Integrating Azure Data Lake in a Multi-Platform Architecture

Multi-Platform Architecture



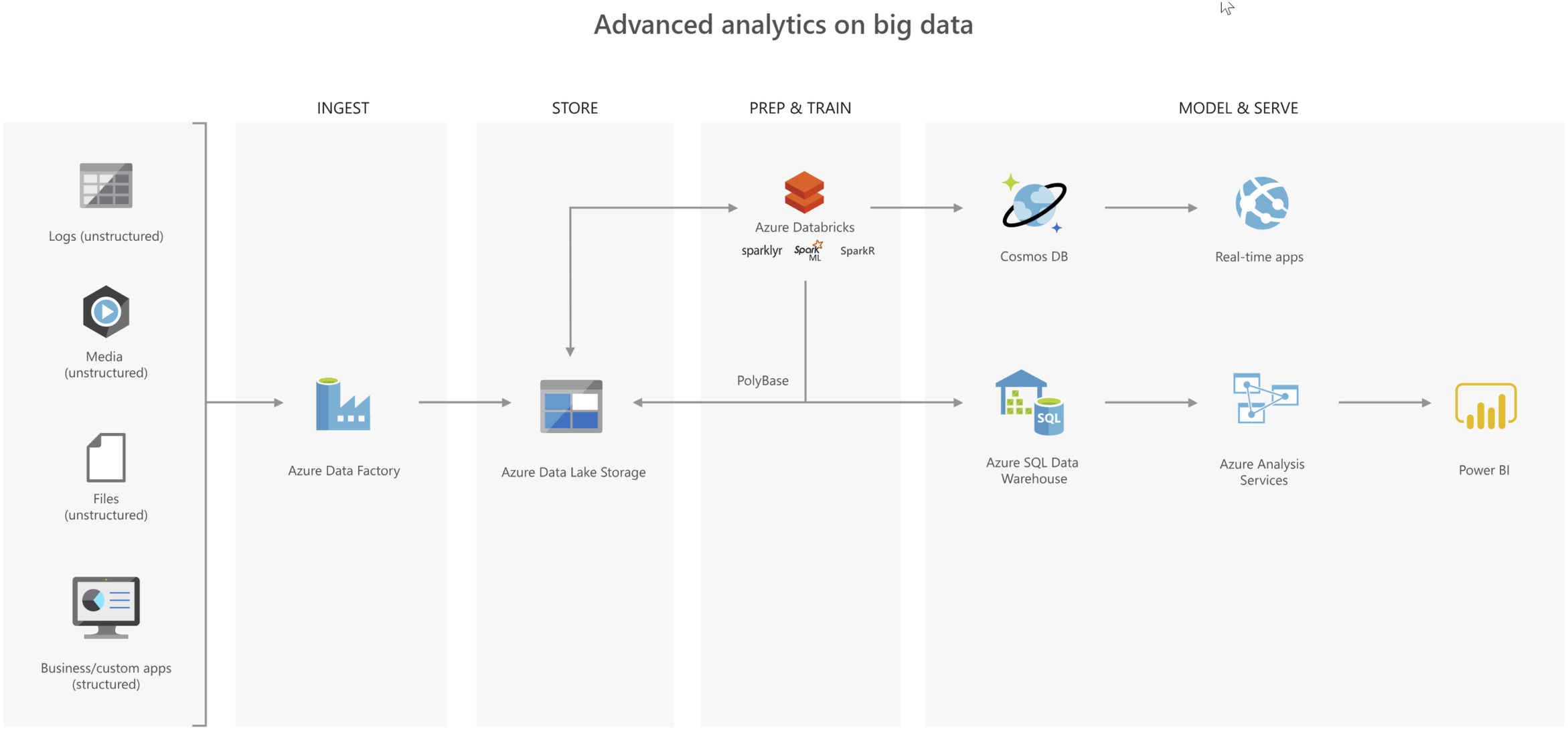
- ✓ Handle a variety of data types & sources
- ✓ Larger data volumes at lower latency
- ✓ Bimodal: self-service + corporate BI to support all types of users
- ✓ Newer cloud services
- ✓ Advanced analytics scenarios
- ✓ Balance data integration & data virtualization

Azure Data Lake Implementation Options



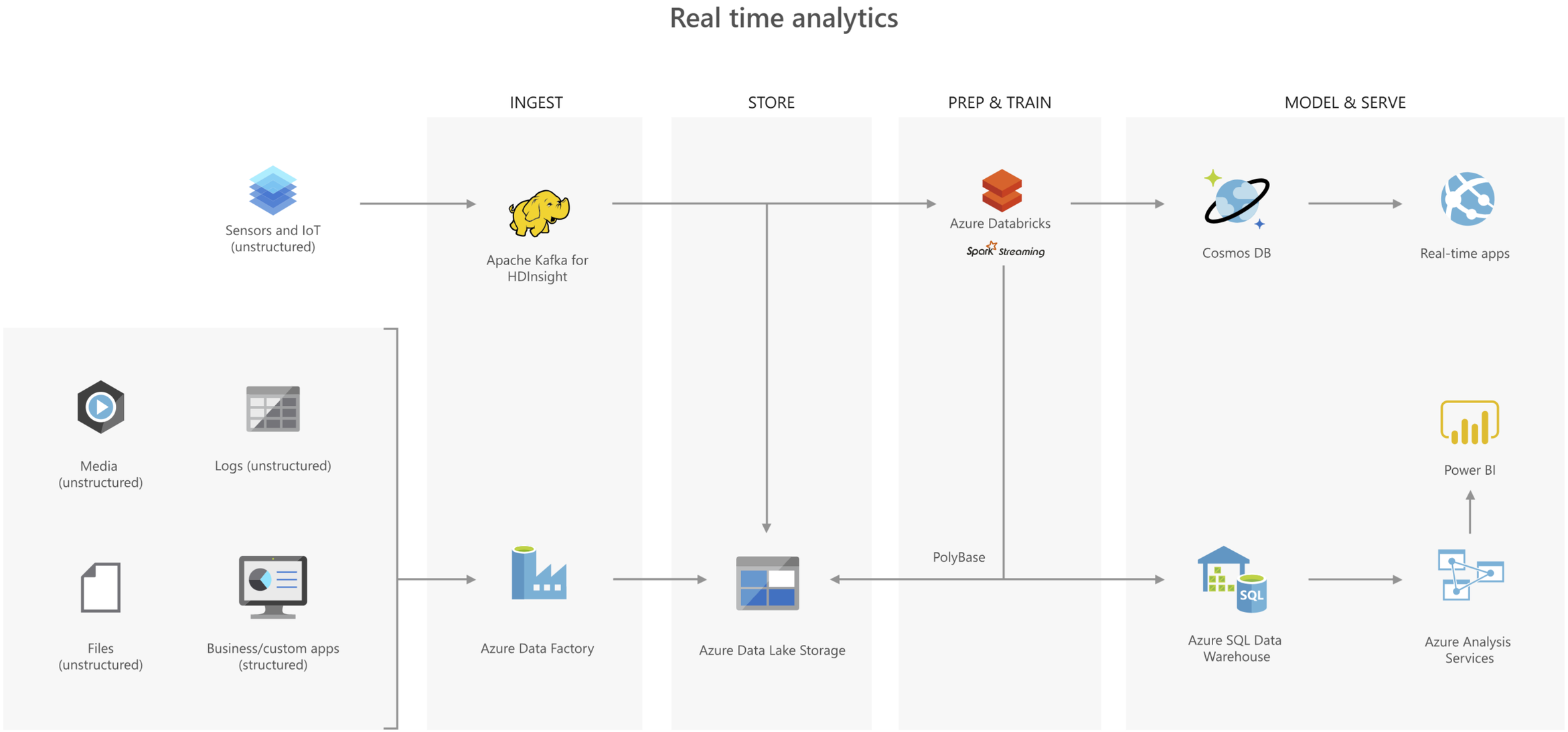
Source: https://azure.microsoft.com/en-us/services/storage/data-lake-storage/

Azure Data Lake Implementation Options



Source: https://azure.microsoft.com/en-us/services/storage/data-lake-storage/

Azure Data Lake Implementation Options



Source: https://azure.microsoft.com/en-us/services/storage/data-lake-storage/

Suggestions for Getting Started with a Data Lake Project

Getting Started With a Data Lake Project

Is a Data Lake the Right Choice?

Do you have *non-relational* data?

Various data types, various sources, whether it's "big data" or not

Do you have *IoT* type of data?

Streaming or micro-batch pipelines

Do you have *advanced analytics scenarios* on unusual datasets?

Do you need to *offload ETL processing (ELT)* and/or *archive data* from a data warehouse or other systems to low-cost storage?

Readiness:

Are you ready willing to learn *different development patterns* and/or *new technologies*?

Are you ready to handle the *trade-offs of 'schema on read'* vs 'schema on write'?

Getting Started With a Data Lake Project

Data

What types of data ingestion pipelines do you have, at what frequency?

- Batch
- Micro-batch
- Streaming

What are the current + anticipated data size volumes, and in what format?

- Structured data
- Semi-structured data
- Unstructured data
- Geospatial data

The data itself influences how simple or complicated an architecture can get

To what extent does semi-structured data need to be integrated with the structured data?

Getting Started With a Data Lake Project

Data Movement & Storage

What level of data integration (ETL or ELT) vs. data virtualization provides optimal data access?

- Data movement can be expensive
- Data might be too large to practically move
- Time window for data processing may be small
- Latency (freshness) of data varies

How much data movement are you willing to do?

Which do you value more?

- Polyglot persistence strategy (“best fit engineering”)
- Architectural simplicity

A multi-platform architecture is more appealing if you subscribe to a polyglot persistence strategy. Success very much depends on staff skills.

Getting Started With a Data Lake Project

Information Delivery

What are the expectations + needs of your user population?

- Casual users
- Data analysts
- Data scientists
- IT, BI specialists, big data engineers

The user base translates into expectations for how the information is to be delivered, which translates into technology choices

What type of data consumption do you support?

- Centralized reporting & analytics
- Decentralized self-service models
- Departmental or subject-specific data marts
- Application integration

The more diverse your user population is, the more likely you will have a multi-platform architecture with both schema-on-read and schema-on-write

Getting Started With a Data Lake Project

Organizing the Data Lake

- ✓ Based on optimal data retrieval & security boundaries
- ✓ Avoid a chaotic, unorganized data swamp
- ✓ Take advantage of data pruning optimizations (esp year/month/day) when running queries

Common ways to organize and/or tag 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

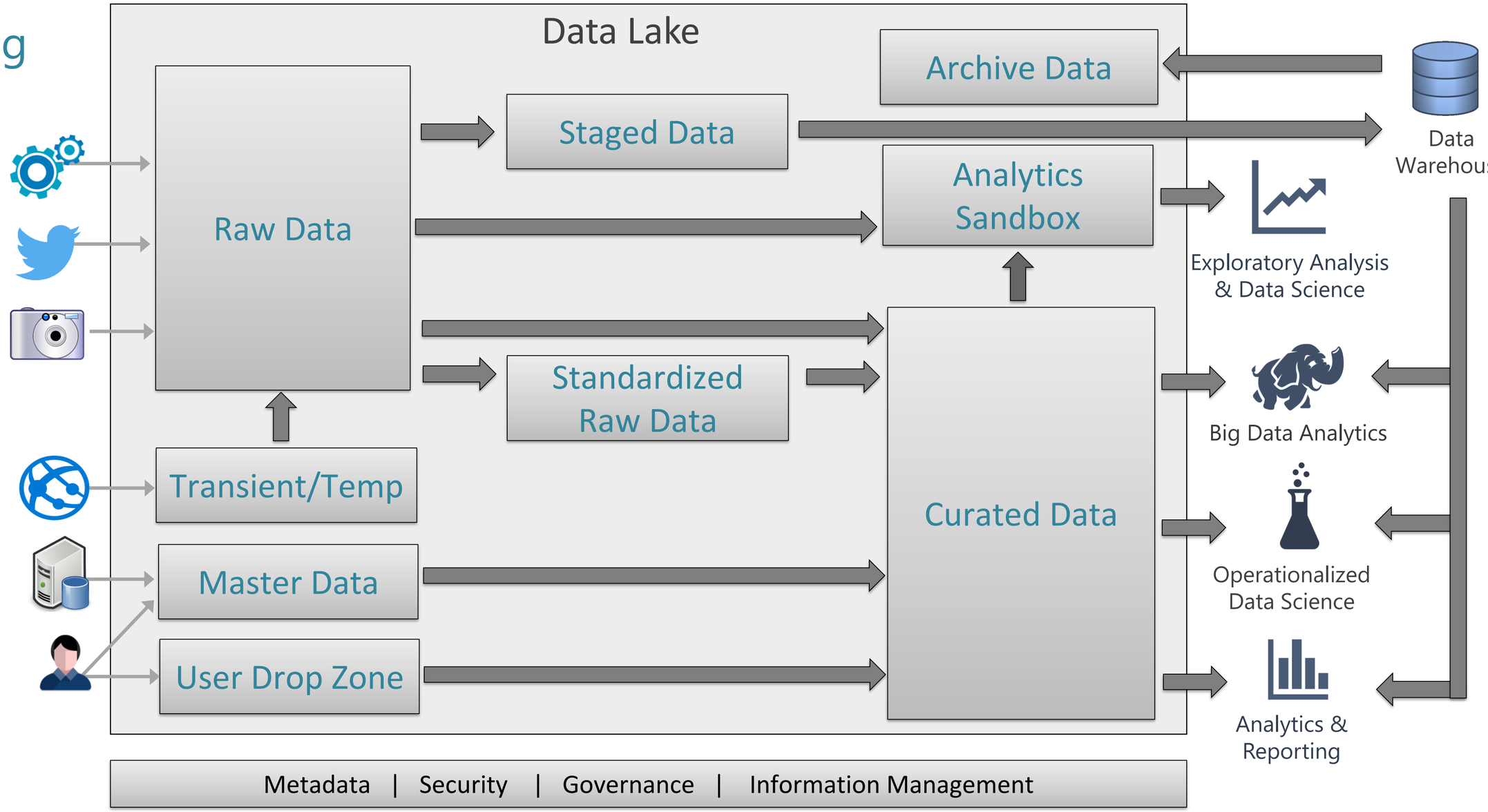
Probability of Data Access
Recent/current data
Historical data
etc…

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

Getting Started With a Data Lake Project

Organizing the Data Lake

Although a data lake emphasizes getting started quickly, there is still up-front planning



Getting Started With a Data Lake Project

Data Lake Challenges

Technology	Process	People	Data
✓ Complex, multi-layered architecture	✓ Right balance of deferred work vs. up-front work to minimize chaos	✓ Expectations & trust	✓ Data volumes
✓ Unknown storage & scalability	✓ Ignoring established best practices for data mgmt	✓ Data stewardship	✓ Read & write performance
✓ Data retrieval	✓ Data quality	✓ Redundant effort	✓ Relating disparate data
✓ Working with un-curated data	✓ Governance	✓ Data engineering skillsets	✓ Schema changes over time
✓ Performance	✓ Security	✓ Ownership changes between teams to operationalize solutions	✓ Diversity of file formats & types
✓ Change management	✓ Disaster recovery for large solutions		
✓ Evolving, maturing tech			

Getting Started With a Data Lake Project

Always do a proof of concept before making a big commitment, including file management, data access & security.

Data tagging & cataloging is critical. Capture metadata whenever possible.

Consider the experience level of your staff, and the ability to support a complex solution.

Assess the impact of open source technologies vs. proprietary technologies.

Getting Started With a Data Lake Project

Cloud offerings/features/functionality are constantly changing. It is very challenging to keep up.

Your goals for going to the cloud will consistently involve trade-offs: cost, complexity, security.

Though traditional data warehousing is evolving, the concept of curated, cleansed, user-friendly data structures are still extremely relevant & needed.

Getting Started With a Data Lake Project

Organize data by time series whenever possible.

Consider data access patterns when designing the folder structure. "Pruning" of data can happen when data is set up in a well-designed hierarchy.

Make careful decisions about file formats you select. Every format has tradeoffs.

File sizes:

- There is a 4.77 TB file limit in Gen 2 (there was no specific file limit in Gen1).
- The traditional Hadoop 'small file size' problem still exists (though technology is evolving to help).
- The ideal, practical file size is still ~250MB - ~2GB.

Azure Data Lake: What, Why, and How

Melissa Coates
Solution Architect, BlueGranite

Download a copy of this presentation:

SQLChick.com > Presentations & Downloads page

Appendix A

Suggestions for Continued Learning

Suggestions for Continued Learning (1/4)

Azure Data Lake Developer Center: http://azure.github.io/AzureDataLake/ ← tons of helpful links
U-SQL Center: http://usql.io/
U-SQL Tutorial: https://saveenr.gitbooks.io/usql-tutorial/content/

Azure Data Lake Samples & Documentation: https://github.com/Azure/AzureDataLake/
U-SQL Samples & Documentation: https://github.com/Azure/USQL

Azure Data Lake Release Notes: https://github.com/Azure/AzureDataLake/tree/master/docs/Release_Notes
Contains new features, fixes, deprecations, and breaking changes

Azure Data Lake Team Blog: https://blogs.msdn.microsoft.com/azuredatalake/

Azure Data Lake Extensions for Visual Studio: https://www.microsoft.com/en-us/download/details.aspx?id=49504

Azure Data Lake Course from Microsoft Virtual Academy: https://mva.microsoft.com/en-US/training-courses/introducing-azure-data-lake-17795?l=SugmcFt9D_3111787171 ← highly recommended

ADL Twitter account: https://twitter.com/azuredatalake
ADL team member Twitter accounts: https://twitter.com/MikeDoesBigData + https://twitter.com/saveenr

Suggestions for Continued Learning (2/4)

ADLS Best Practices: https://docs.microsoft.com/en-us/azure/data-lake-store/data-lake-store-best-practices

ADLS Performance Tuning Guidance: https://docs.microsoft.com/en-us/azure/data-lake-store/data-lake-store-performance-tuning-guidance

ADLA Saving Money & Controlling Costs: https://onedrive.live.com/?authkey=%21AHceDeuGX5PKbVw&id=3BDE3286AB2E59F7%211005&cid=3BDE3286AB2E59F7

Azure Data Architecture Guide: https://docs.microsoft.com/en-us/azure/architecture/data-guide/

Creating External Data Sources for PolyBase and Elastic Queries: https://docs.microsoft.com/en-us/sql/t-sql/statements/create-external-data-source-transact-sql

Azure Big Data Blog: https://azure.microsoft.com/en-us/blog/topics/big-data/

Suggestions for Continued Learning (3/4)

Blog Posts

Querying Data in Azure Data Lake Store with Power BI:
https://www.sqlchick.com/entries/2018/5/6/querying-data-in-azure-data-lake-store-with-power-bi

Zones in a Data Lake: https://www.sqlchick.com/entries/2017/12/30/zones-in-a-data-lake

Granting Permissions in Azure Data Lake: https://www.sqlchick.com/entries/2018/3/16/granting-permissions-azure-data-lake

Running U-SQL on a Schedule with Azure Data Factory to Populate Azure Data Lake:
https://www.sqlchick.com/entries/2017/10/8/running-u-sql-on-a-schedule-with-azure-data-factory

Querying Multi-Structured JSON Files with U-SQL in Azure Data Lake: https://www.sqlchick.com/entries/2017/9/4/querying-multi-structured-json-files-with-u-sql-in-azure-data-lake

Handling Row Headers in U-SQL: https://www.sqlchick.com/entries/2017/7/27/handling-row-headers-in-u-sql

Two Ways to Approach Federated Queries with U-SQL and ADLA:
https://www.sqlchick.com/entries/2017/10/29/two-ways-to-approach-federated-queries-with-u-sql-and-azure-data-lake-analytics

Suggestions for Continued Learning (4/4)

Video

What's New with Azure Data Lake Storage Gen 2
https://www.youtube.com/watch?v=DJkFSpis2B0

E-Book

Data Lakes in a Modern Data Architecture
https://www.blue-granite.com/data-lakes-in-a-modern-data-architecture-ebook

Appendix B

Terms & Definitions

Definitions

Data Warehouse	Repository of data from multiple sources, cleansed & enriched for reporting; generally ‘schema on write’
Data Lake	Repository of data for multi-structured data; generally ‘schema on read’
Hadoop	(1) Data storage via HDFS (Hadoop Distributed File System), and (2) Set of Apache projects for data processing and analytics
Lambda Architecture	Data processing & storage with batch, speed, and serving layers
ETL	Extract > Transform > Load: traditional paradigm associated with data warehousing and ‘schema on write’
ELT	Extract > Load > Transform: newer paradigm associated with data lakes & ‘schema on read’
Semantic Model	User-friendly interface for users on top of a data warehouse and/or data lake

Definitions

Data Integration	Physically moving data to integrate multiple sources together
Data Virtualization	Access to one or more distributed data sources without requiring the data to be physically materialized in another data structure
Federated Query	A type of data virtualization: access & consolidate data from multiple distributed data sources
Polyglot Persistence	A multi-platform strategy which values using the most effective technology based on the data itself (“best fit engineering”)
Schema on Write	Data structure is applied at design time, requiring additional up-front effort to formulate a data model (relational DBs)
Schema on Read	Data structure is applied at query time rather than when the data is initially stored (data lakes, NoSQL)

More in-depth definitions: https://www.sqlchick.com/entries/2017/1/9/defining-the-components-of-a-modern-data-warehouse-a-glossary

Definitions

(3/3)