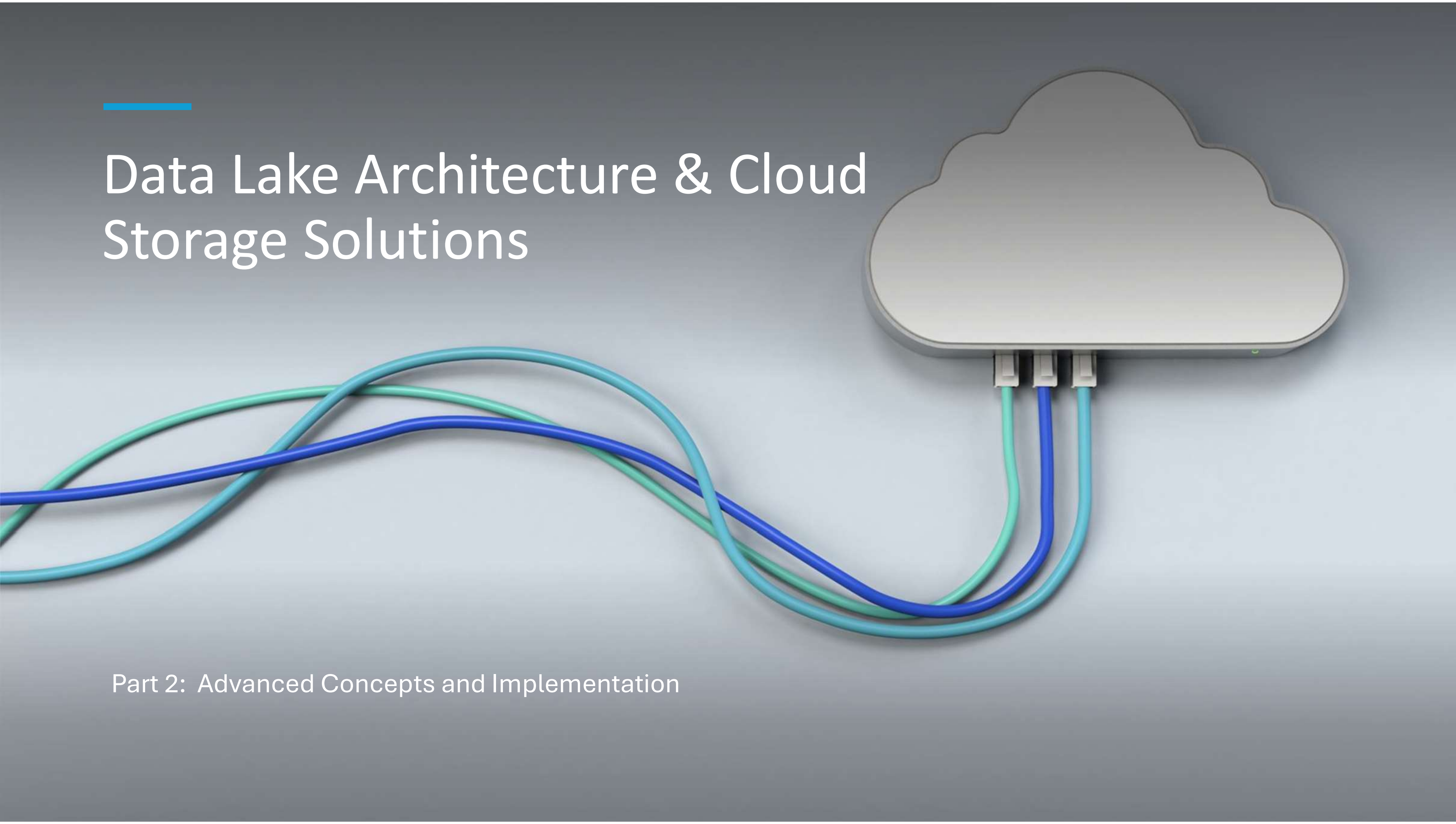
Data Lake Architecture & Cloud Storage Solutions

Part 2: Advanced Concepts and Implementation



The Data Lake Challenge: Beyond Raw Storage

Limitations of Basic Object Storage

- Lack of ACID transactions
- No schema enforcement or evolution
- Limited metadata management
- Performance issues with direct queries
- Consistency challenges

Solution: Table Formats for Data Lakes

- No support for record-level updates or deletes
- Bring database-like features to object storage
- Enable ACID transactions
- Support schema enforcement and evolution
- Improve query performance
- Enable time travel and versioning

Data Lake Architecture Patterns

Medallion Architecture (Databricks)

- **Bronze Layer**: Raw data, exactly as ingested
- **Silver Layer**: Cleansed and conformed data
- **Gold Layer**: Business-level aggregates
- Each layer implemented as Delta tables
- Progressive refinement of data quality

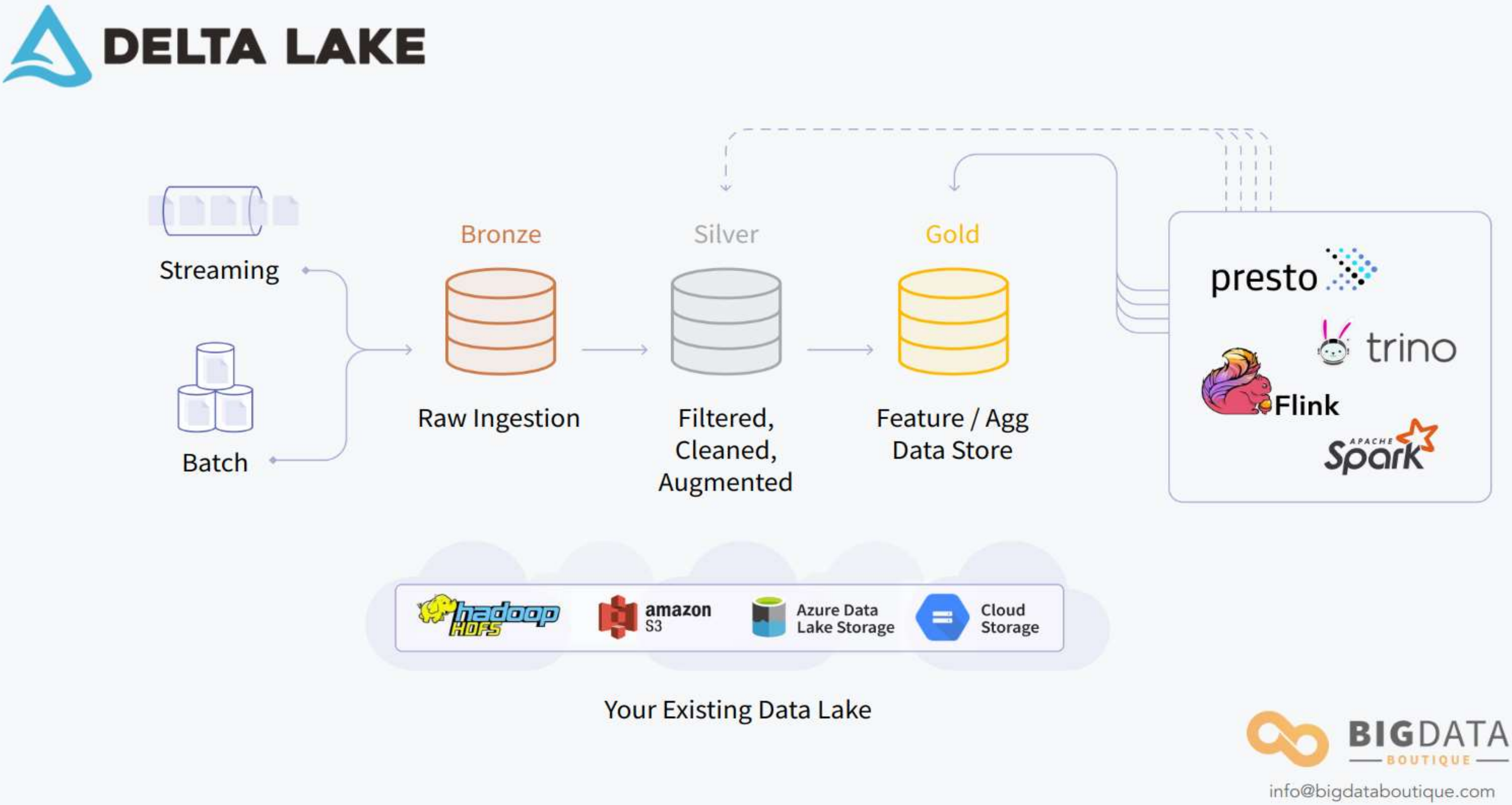
Lakehouse Architecture

- Combines data lake storage with data warehouse features
- Structured data management on top of object storage
- ACID transactions and schema enforcement
- Performance optimizations for analytics
- Support for diverse workloads (BI, ML, data science)

Table Formats for Data Lakes: Delta Lake

- Open-source storage layer by Databricks
- ACID transactions on cloud object stores
- Schema enforcement and evolution
- Time travel (data versioning)
- Audit history
- Uses Parquet files with transaction log
- Optimized for Spark integration

"Delta Lake uses a transaction log that is compacted into Apache Parquet format to provide ACID properties, time travel, and significantly faster metadata operations for large tabular datasets." - Armbrust et al., VLDB 2020



Source: https://bigdataboutique.com/blog/delta-lake-by-databricks-an-introduction-f78147

Table Formats for Data Lakes: Apache Iceberg

Open-source table format by Netflix

Schema evolution and hidden partitioning

Time travel and snapshot isolation

ACID transactions

Engine-agnostic (Spark, Flink, Presto, Trino)

Optimized for large analytical workloads

Table Formats for Data Lakes: Apache Hudi

Open-source table format by Uber

Upsert and delete capabilities

Incremental data processing

Near real-time analytics

Snapshot and incremental queries

Optimized for streaming data pipelines

Comparison of Table Formats

Feature	Delta Lake	Apache Iceberg	Apache Hudi
Created by	Databricks	Netflix	Uber
ACID Transactions	Yes	Yes	Yes
Schema Evolution	Yes	Yes	Yes
Time Travel	Yes	Yes	Yes
Update/Delete	Yes	Yes	Yes
Primary Engine	Spark	Engine-agnostic	Spark, Flink
Metadata Storage	Transaction Log	Metadata Files	Timeline
Partitioning	Standard	Hidden	Standard
Streaming Support	Strong	Limited	Strong
Community Adoption	High	Growing	Moderate

Data Lake Processing Engines: Batch Processing

Apache Spark: Distributed analytics engine

- In-memory processing
- SQL, DataFrame, and ML APIs
- Support for multiple languages (Scala, Java, Python, R)

Apache Hadoop: MapReduce-based processing

- Disk-based processing
- High fault tolerance
- Java-based programming model

Apache Hive: SQL-like queries on Hadoop

- HiveQL query language
- Schema-on-read capabilities
- Optimized for large batch queries

Data Lake Processing Engines: Stream Processing

Apache Kafka Streams: Stream processing library

- Built on Kafka
- Exactly-once semantics
- Stateful and stateless processing

Apache Flink: Unified batch and stream processing

- True streaming (not micro-batching)
- Exactly-once guarantees
- Complex event processing
- State management

Apache Spark Structured Streaming: Extension of Spark

- Micro-batch processing
- Integration with Spark SQL
- Stateful processing

Query Engines for Data Lakes

Interactive Query Engines

- **Presto/Trino**: Distributed SQL query engine
  - Federation across multiple data sources
  - In-memory processing
  - ANSI SQL support
- **Apache Drill**: Schema-free SQL query engine
  - Dynamic schema discovery
  - Nested data support
  - SQL:2003 compliance
- **Dremio**: Self-service data platform
  - Data reflections for acceleration
  - Virtual datasets
  - SQL interface

Participation Question

Go to the Activity Google Slide:

https://docs.google.com/presentation/d/1UMe6eEPrtnIY-E6y1Y2XoKK7fKHML9IVj07quI87ALY/edit?usp=sharing

For your scenario, sketch on paper or a shared whiteboard:

1. **Bronze** (raw ingestion):
   – What raw files arrive? (e.g. JSON sensor dumps, CSV click logs)
   – How would you partition directory layout?
2. **Silver** (clean & conform):
   – List the basic cleaning steps (dedupe, type-cast, filter bad records).
   – Note any joins or enrichments.
3. **Gold** (business/ML ready):
   – Define the final aggregates or feature tables (hourly averages, user funnels, risk scores).

While you map these, **choose one table format** and justify in 1 sentence:

- **Delta Lake** for strong ACID + time travel,
- **Iceberg** for hidden partitioning + engine agnosticism,
- **Hudi** for incremental upserts & real-time pipelines

Data Governance and Metadata Management

Why Data Governance Matters

- Prevents data lakes from becoming "data swamps"
- Ensures data quality and consistency
- Enables self-service analytics
- Supports compliance requirements
- Improves data discovery and understanding

Key Components

1. **Data Catalog**: Inventory of all data assets
2. **Metadata Management**: Technical and business metadata
3. **Data Lineage**: Tracking data origin and transformations
4. **Data Quality**: Monitoring and enforcement
5. **Access Control**: Securing sensitive data
6. **Lifecycle Management**: Retention and archiving policies

Data Catalog Solutions

AWS Glue Data Catalog

- Integrated with AWS services
- Automatic schema discovery
- Versioning support
- Integration with Lake Formation for security

Azure Purview

- Automated data discovery and classification
- Data lineage visualization
- Integration with Azure services
- AI-powered data understanding

Google Data Catalog

- Fully managed and serverless
- Integration with BigQuery and GCS
- Tag templates for custom metadata
- Fine-grained access control

Metadata Management Approaches

Technical Metadata

- Schema definitions
- Data types and formats
- Partitioning information
- Storage locations
- Update frequency

Business Metadata

- Business definitions
- Data owners
- Data stewards
- Data classification
- Usage policies

Operational Metadata

- Processing history
- Data quality metrics
- Usage statistics
- Performance metrics

Future Trends in Data Lake Technologies

- Data Mesh
- Data Lakehouse Convergence
- Serverless Data Processing
- AI/ML Integration
- Real-Time Data Lakes
- Multi-Cloud Data Lakes

Netflix: Data Lake on AWS S3

- Petabyte-scale data lake on S3
- Created Apache Iceberg for table format
- Uses Presto/Trino for interactive queries
- Spark for batch processing
- Custom metadata management

Uber: Data Lake for Real-Time Analytics

- Apache Hudi for data lake table format
- Kafka for real-time data ingestion
- Spark for processing
- Custom data catalog (Databook)
- Presto for interactive queries

Capital One: Cloud-Native Data Lake

- AWS-based data lake architecture
- Lake Formation for governance
- Redshift Spectrum for analytics
- Custom data quality framework
- Self-service data access

Summary: Data Lake Architecture & Cloud Storage

- Data lakes provide flexible, scalable storage for all data types
- Modern data lakes leverage cloud object storage (S3, GCS, Azure Blob)
- Table formats (Delta Lake, Iceberg, Hudi) add database-like features
- Data governance and metadata management are critical for success
- Processing engines should be selected based on workload requirements
- Future trends point toward lakehouse convergence and data mesh