aws SUMMIT

Building Data Lakes That Cost Less and Deliver Results Faster

John Mallory

Storage Business Development Manager

What to expect from the session

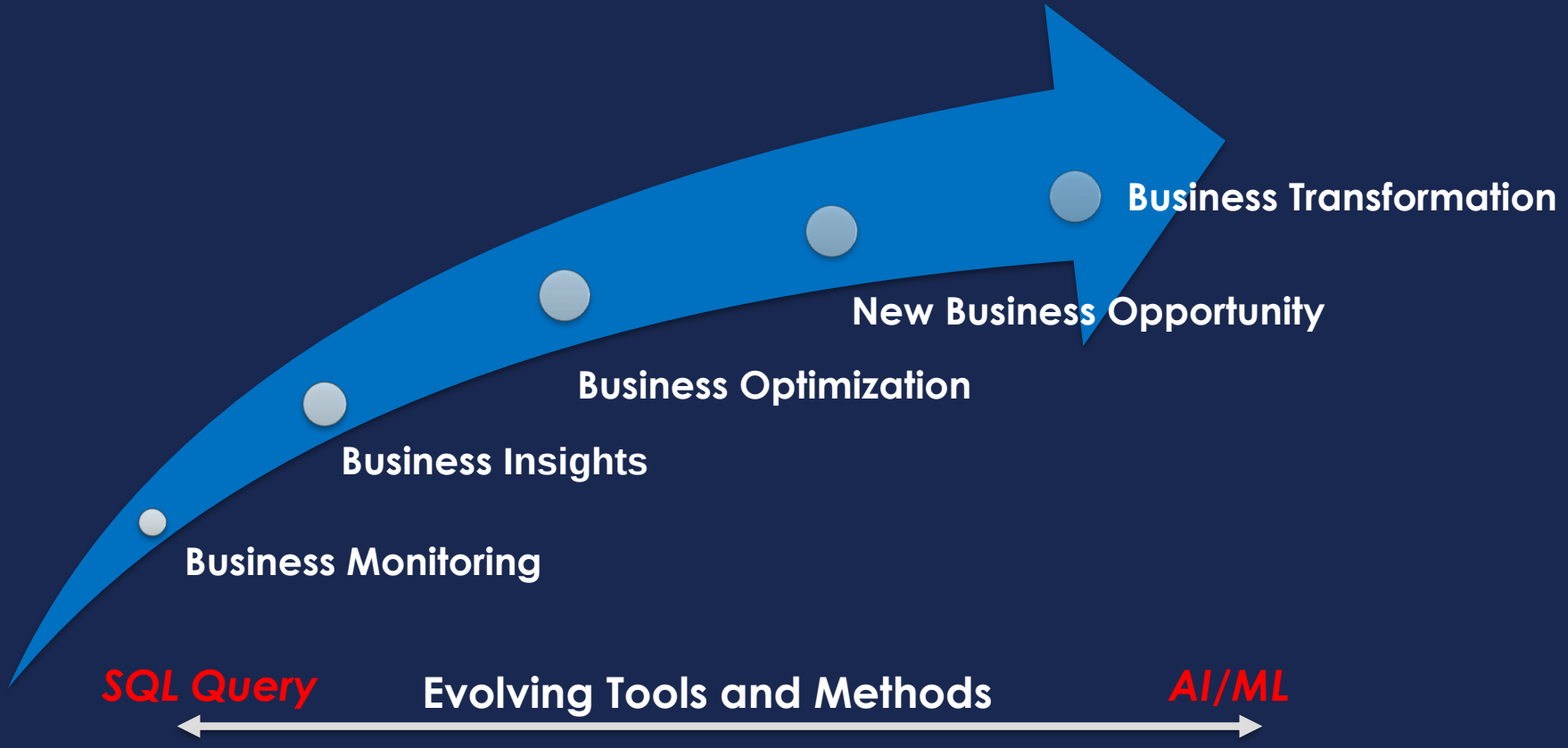
1. Defining the Data Lake

2. Reducing Costs

3. Increasing Performance

4. Planning for the Future

Finding Value in Data is a Journey



Defining the AWS data lake

Data lake is an architecture with a virtually limitless centralized storage platform capable of categorization, processing, analysis, and consumption of heterogeneous data sets

Key data lake attributes

- Decoupled storage and compute
- Rapid ingest and transformation
- Secure multi-tenancy
- Query in place
- Schema on read

Example of AWS Services for Data Lake

Ingest Methods

Amazon Kinesis Firehose

AWS Database Migration Service

AWS Storage Gateway

AWS Snowball*

AWS Direct Connect

Catalog

Amazon DynamoDB

Amazon Elasticsearch

AWS Glue

Central Storage

Processing and Analytics

Amazon Athena

Amazon Kinesis

Amazon Redshift*

Amazon QuickSight

Amazon EMR

Amazon RDS

Amazon Elasticsearch

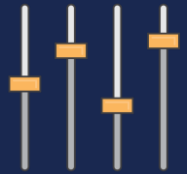
Access and Secure

aws SUMMIT

Reasons to choose Amazon S3 for data lake

Unmatched durability, availability, and scalability

Best security, compliance, and audit capability



Object-level control at any scale

Insights, tools & policies to manage your data

Most ways to bring data in

Twice as many partner integrations

Optimize costs with data tiering

Hot

HDFS

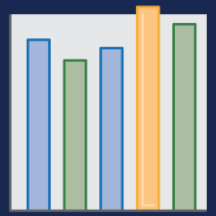
Amazon
S3 standard

Amazon S3—
infrequent access

Amazon
Glacier

Cold

- ✓ Use EMR/Hadoop with local HDFS for hottest data sets
- ✓ Store cooler data in S3 and cold in Glacier to reduce costs
- ✓ Use S3 Analytics to optimize tiering strategy



S3 Analytics

Choosing the Right Data Ingest Tools

Ingest Methods

Amazon Kinesis Firehose

AWS Database Migration Service

AWS Storage Gateway

AWS Snowball*

AWS Direct Connect

IoT, Sensor Data, Clickstream Data, Social Media Feeds, Streaming Logs

Oracle, MySQL, MongoDB, DB2, SQL Server, Amazon RDS

On-premise ERP, Mainframes, Lab Equipment, NAS Storage

Offline Sensor Data, NAS, On-premise Hadoop

On-premise Data Lakes, EDW, Large Scale Data Collection

A Data Lake Needs to Accommodate a Wide Variety of Concurrent Data Sources

Amazon Kinesis—Real Time

Easily collect, process, and analyze video and data streams in real time

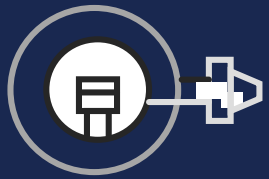
New

Kinesis Video Streams

Capture, process, and store video streams for analytics

Kinesis Data Streams

Build custom applications that analyze data streams



Kinesis Data Firehose

Load data streams into AWS data stores

Kinesis Data Analytics

Analyze data streams with SQL

Ingest Considerations

Separate ingest buckets from S3 Data Lake buckets

- Prepare data before loading into data lake
- Preserve raw assets (potentially lifecycle to Glacier)

Aggregate smaller files/objects ahead of S3 where possible

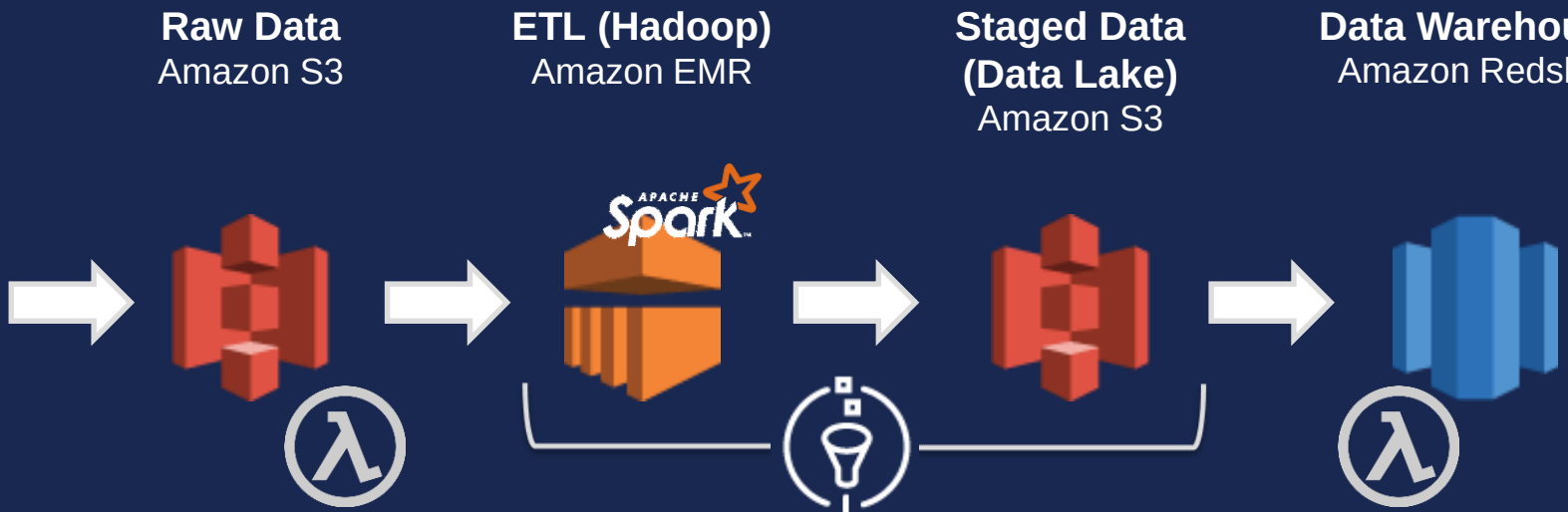
- Reduces transaction costs
- Avoids TPS limits (can also pre-partition S3 bucket)

Consider Storage Gateway/Snowball Edge for file data

- Converts files to S3 objects, while preserving metadata

Common AWS data pipeline configuration

Highly decoupled configurations scale better, are more fault tolerant, and cost optimized



Choosing the Right Data Formats

There is no such thing as the “best” data format

- All involve tradeoffs, depending on workload & tools
- CSV, TSV, JSON are easy, but not efficient
  - Compress & store/archive as raw input
- Columnar compressed are generally preferred
  - Parquet or ORC
  - Smaller storage footprint = lower cost
  - More efficient scan & query
- Row oriented (AVRO) good for full data scans

Key considerations are cost, performance & support

Choosing the Right Data Formats (cont.)

Pay by the amount of data scanned per query

Use Compressed Columnar Formats

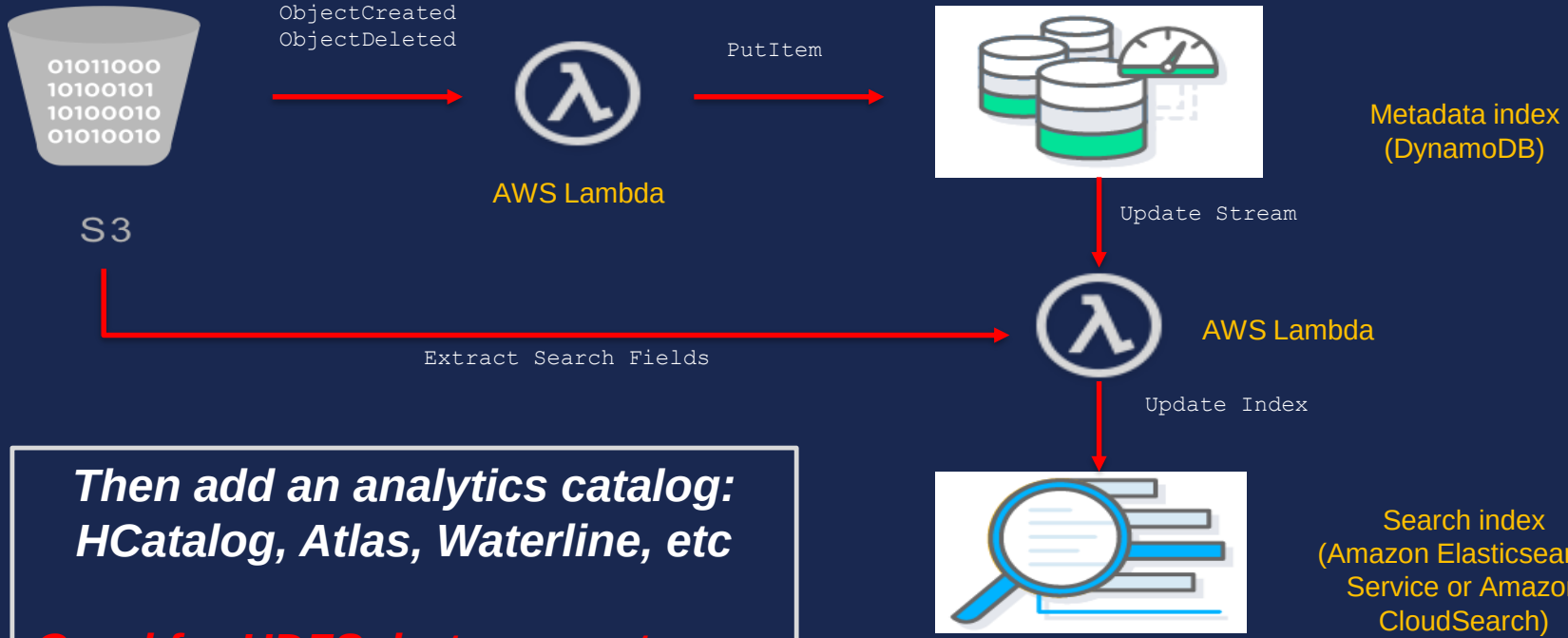
- Parquet
- ORC

Easy to integrate with wide variety of tools

```
SELECT elb_name,
       uptime,
       downtime,
       cast(downtime as DOUBLE)/cast(uptime as DOUBLE) uptime_downtime_ratio
FROM
    (SELECT elb_name,
        sum(case elb_response_code
        WHEN '200' THEN
        1
        ELSE 0 end) AS uptime, sum(case elb_response_code
        WHEN '404' THEN
        1
        ELSE 0 end) AS downtime
    FROM elb_logs_raw_native
    GROUP BY  elb_name)
```

Dataset	Size on Amazon S3	Query Run time	Data Scanned	Cost
Logs stored as Text files	1 TB	237 seconds	1.15TB	$5.75
Logs stored in Apache Parquet format*	130 GB	5.13 seconds	2.69 GB	$0.013
Savings	**87% less with Parquet**	**34x faster**	**99% less data scanned**	**99.7% cheaper**

Catalog Your S3 Data—Old Way



Then add an analytics catalog: HCatalog, Atlas, Waterline, etc

Good for HDFS, but we want more

Amazon's Data Catalog Strategy: AWS Glue

- Glue Catalog Becomes the Hub of the Data Lake
  - Data Discovery & Profiling
  - Data Lineage
  - Data Governance
- Glue Becomes the Serverless ETL Platform
  - Event Driven Workflows
  - Intelligent Auto-Scaling
  - Developer Eco-system
- Glue Drives Interactive Data Preparation
  - Managed Notebook Interface
  - Data Wrangling
  - Support for ML Packages

AWS Glue

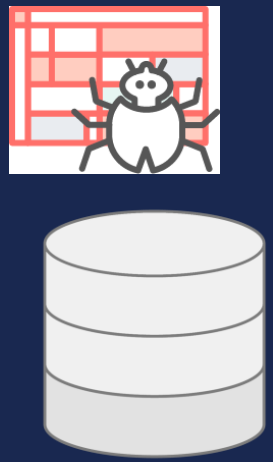
AWS Glue Data Catalog
Central Metadata Catalog for the data lake

Allows you to share catalog & metadata between Amazon Athena, Amazon Redshift Spectrum, EMR & JDBC sources

We added a few extensions:

- **Search** over metadata for data discovery
- **Connection info** – JDBC URLs, credentials
- **Classification** for identifying and parsing files
- **Versioning** of table metadata as schemas evolve and other metadata are updated

AWS Glue Data Catalog Crawlers



AWS Glue Data Catalog - Crawlers
Helping Catalog your data

Crawlers automatically build your Data Catalog and keep it in sync

Automatically discover new data, extracts schema definitions

- Detect schema changes and version tables
- Detect Hive style partitions on Amazon S3

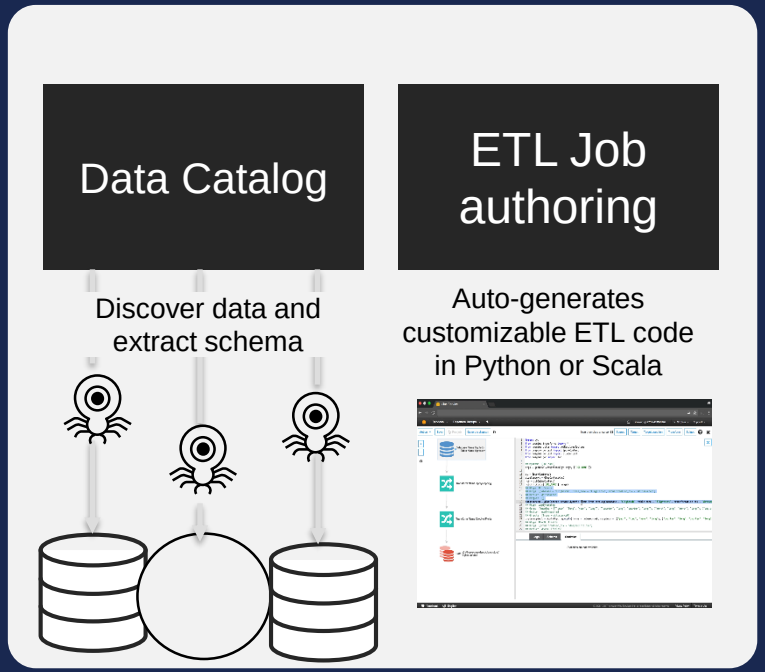
Built-in classifiers for popular types; custom classifiers using Grok expression

Run ad hoc or on a schedule; serverless – only pay when crawler runs

Data Preparation is ~80% of Data Lake Work

- Building training sets
- Cleaning and organizing data
- Collecting data sets
- Mining data for patterns
- Refining algorithms
- Other

AWS Glue ETL Service



Automatically discovers data and stores schema

Data is immediately searchable, and available for ETL

Automatically generates customizable code

Schedules and runs your ETL jobs

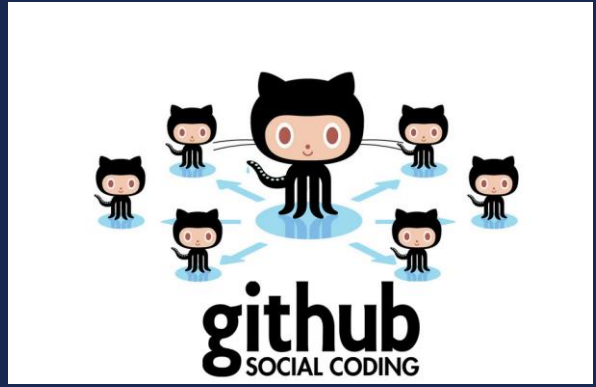
Serverless

AWS Glue Job Authoring: Leveraging the Community

No need to start from scratch.

Use **Glue samples** stored in Github to share, reuse, contribute: https://github.com/awslabs/aws-glue-samples

- Migration scripts to import existing Hive Metastore data into AWS Glue Data Catalog
- Examples of how to use Dynamic Frames and Relationalize() transform
- Examples of how to use arbitrary PySpark code with Glue's Python ETL library



Download **Glue's Python ETL library** to start developing code in your IDE: https://github.com/awslabs/aws-glue-libs

Process data in place…

Amazon Athena

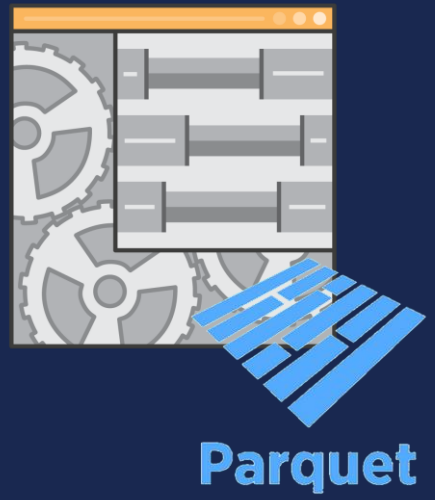
Amazon Redshift Spectrum

Amazon EMR

AWS Glue

Amazon EMR: Decouple compute & storage

Highly distributed processing frameworks such as **Hadoop/Spark**



Compress datasets

Columnar file formats

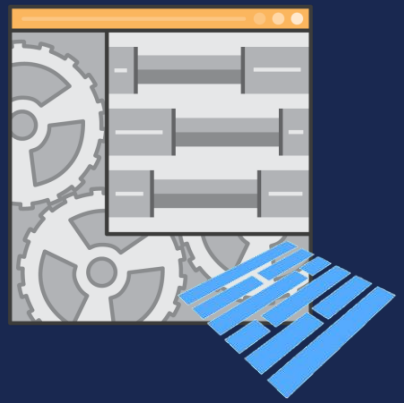
Aggregate small files

S3distcp “group-by” clause

Amazon Redshift Spectrum: Exabyte Scale query-in-place

Structured data w/ joins

Multiple on-demand clusters-scale concurrency



Columnar file formats

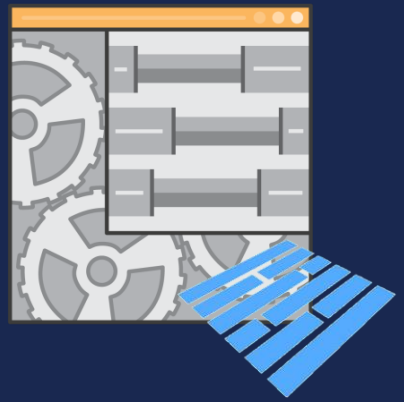
Data partitioning

Better query performance with predicate pushdown

Amazon Athena: Query without ETL

Serverless service

Schema on read



Compress datasets

Columnar file formats

Optimize file sizes

Optimize querying (Presto backend)

Query Data in Glacier (Coming)

Amazon SageMaker

The quickest and easiest way to get ML models from idea to production

End-to-End Machine Learning Platform

Zero setup

Flexible Model Training

Pay by the second

Today: All of these tools…

retrieve a lot of data they don't need and
do the heavy lifting

Introducing…

Amazon S3 Select and Amazon Glacier Select

Select subset of data from an object based on a SQL expression

Motivation Behind S3 Select

GET all the data from S3 objects, and my application will filter the data that I need

Redshift Spectrum Example:

- Beta customer: Run 50,000 queries
- Amount of data fetched from S3: 6 PBs
- Amount of data used in Redshift: 650 TB

Data needed from S3: 10%

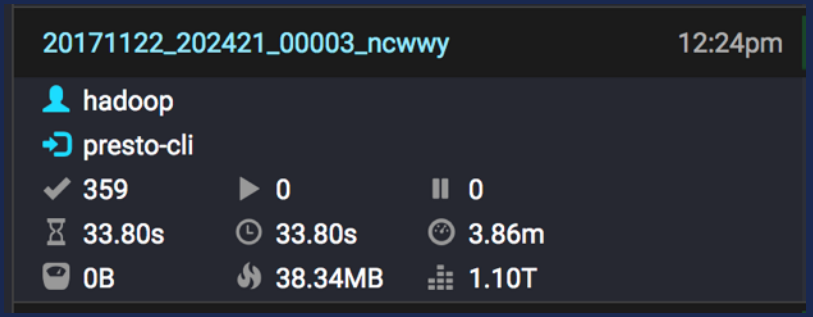
Amazon S3 Select

SELECT a filtered set of data from within an object using standard SQL Statements

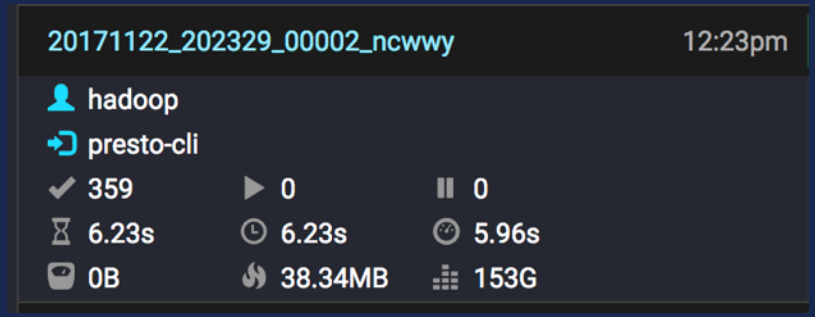
- First content aware API within Amazon S3
- Unlike Amazon Athena and Spectrum, operates within the Amazon S3 system
- SQL Statement operates on a per-object basis—not across a group of objects
- Works and scales like GET requests
- Accessible via SDK (Java, Python), AWS CLI and Presto Connector—others to follow
- Who will use it?
  - Amazon Redshift Spectrum, Amazon Athena, Presto and other custom Query engines
  - Everyone doing log mining

Amazon S3 Select: Accelerating big data

Before

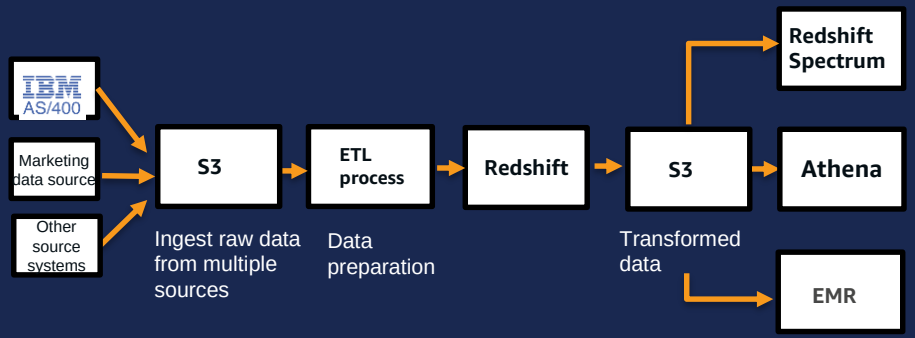


After



5X Faster with 1/40 of the CPU

Sysco—Analytics on the Data Lake



Sysco is the leader in selling, marketing, & distributing food

Challenge: Large volumes of data stored in multiple systems

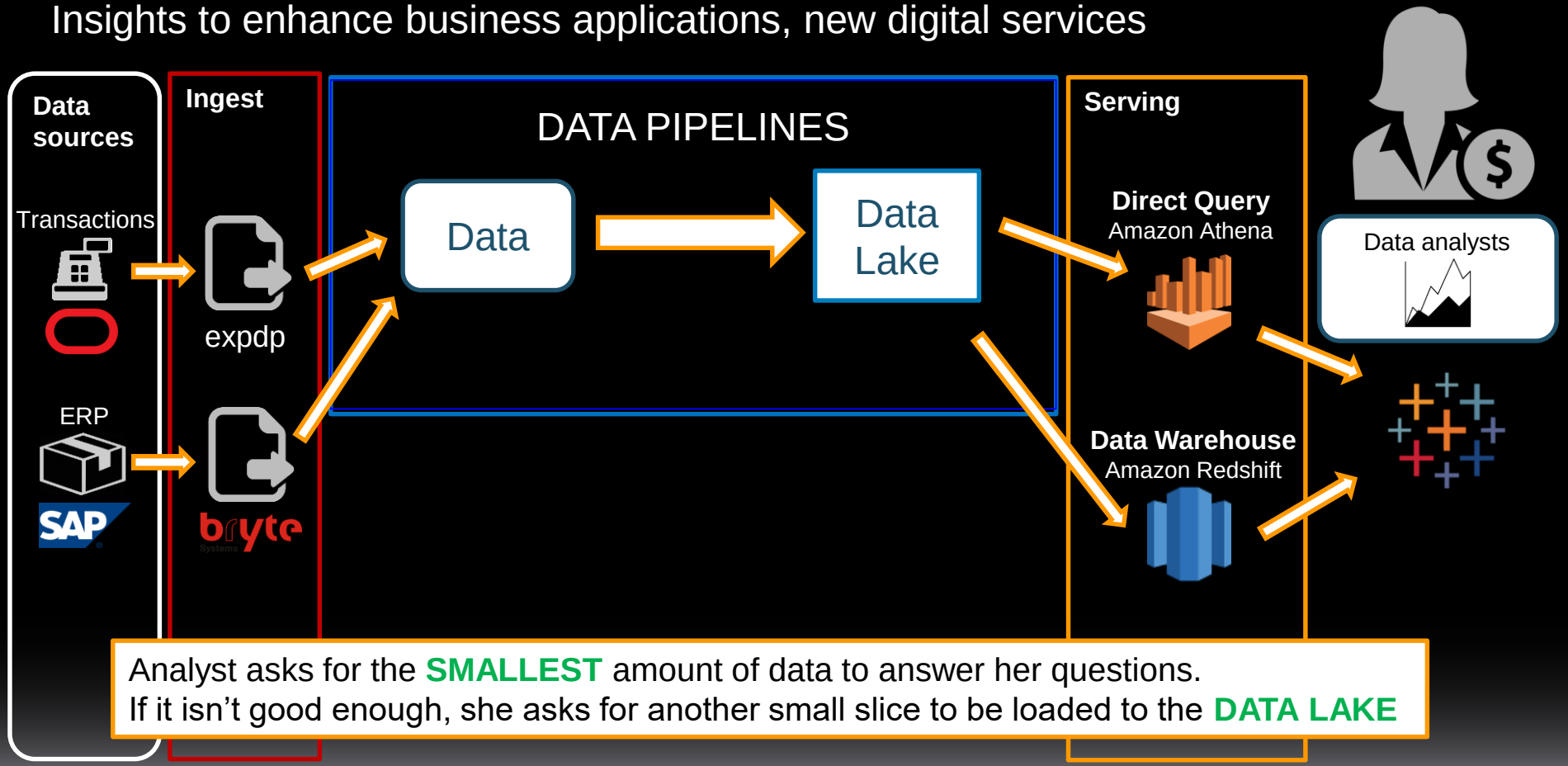
Consolidated data into a single S3 data lake

Redshift Spectrum used by business users for reporting

EMR & Athena used by data scientists

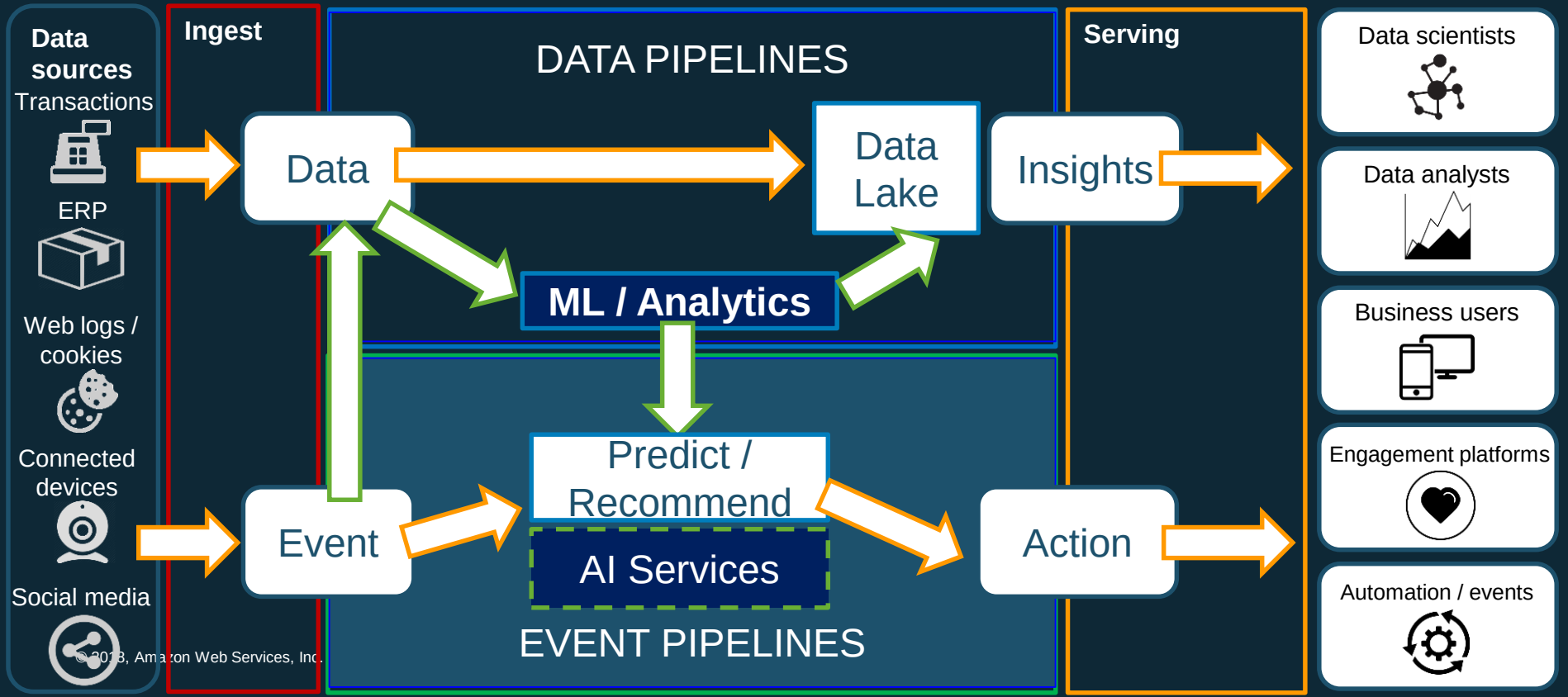
Modern data architecture

Insights to enhance business applications, new digital services



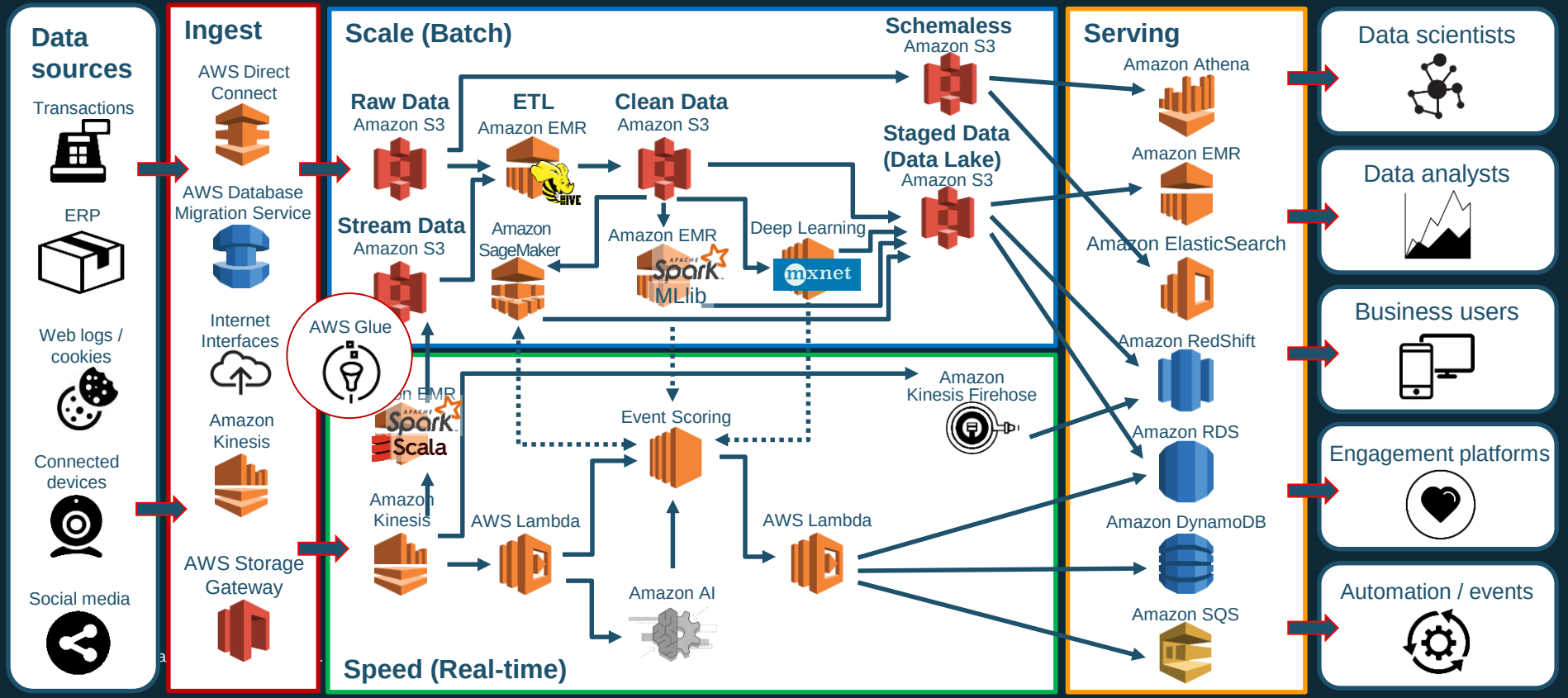
Modern data architecture

Insights to enhance business applications, new digital services



Real-time engagement

Interactive customer experience, event-driven automation, fraud detection



aws SUMMIT

Thank you!