Big Data Architectural Patterns and Best Practices on AWS

Julien Simon
Principal Technical Evangelist
@julsimon

What to Expect from the Session

Big data challenges

Architectural principles

How to simplify big data processing

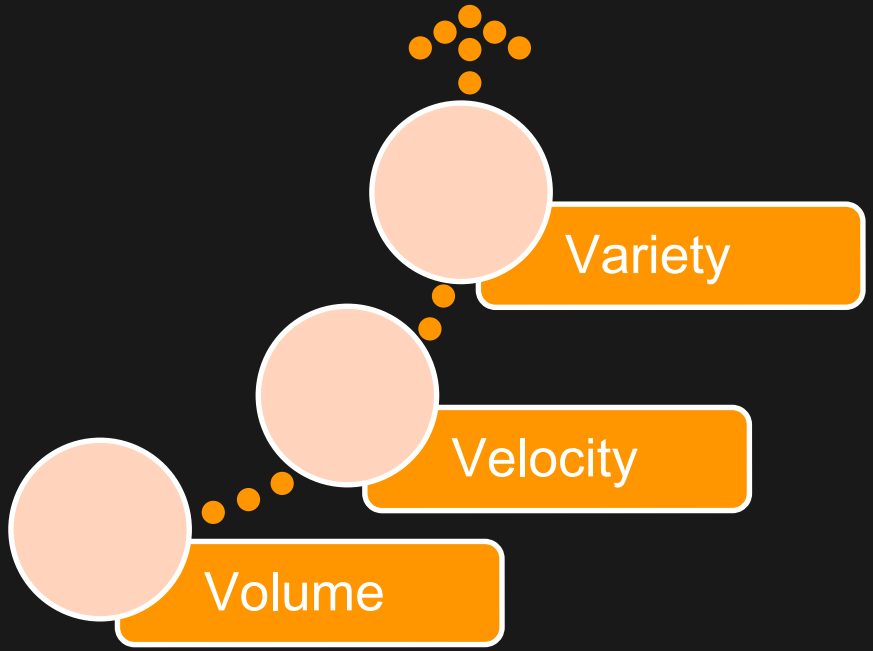
What technologies should you use?

- Why?
- How?

Reference architecture

Design patterns

Ever Increasing Big Data



Big Data Evolution

Batch processing

Stream processing

Artificial intelligence

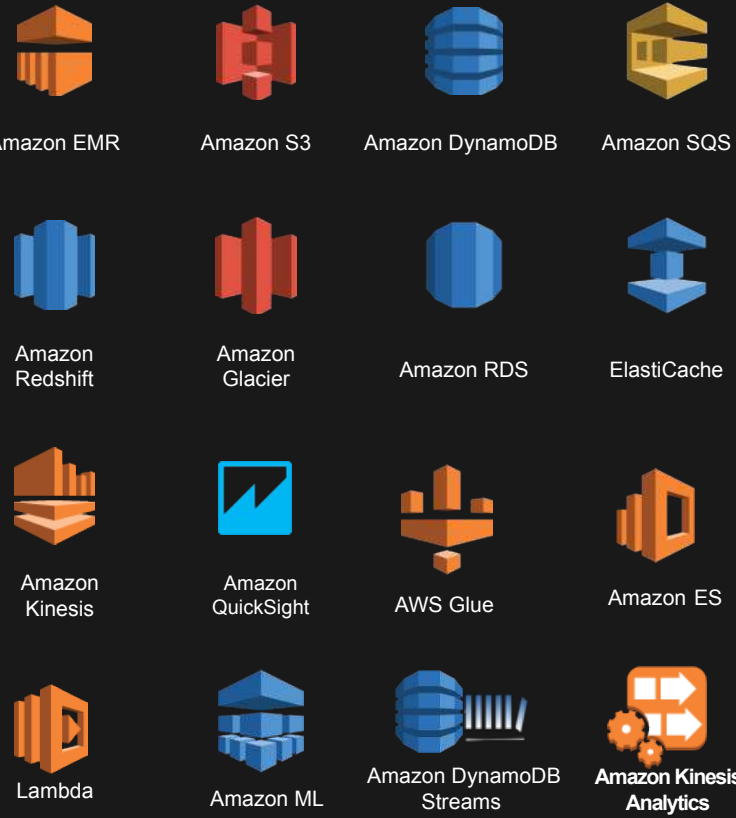
Cloud Services Evolution

Virtual machines

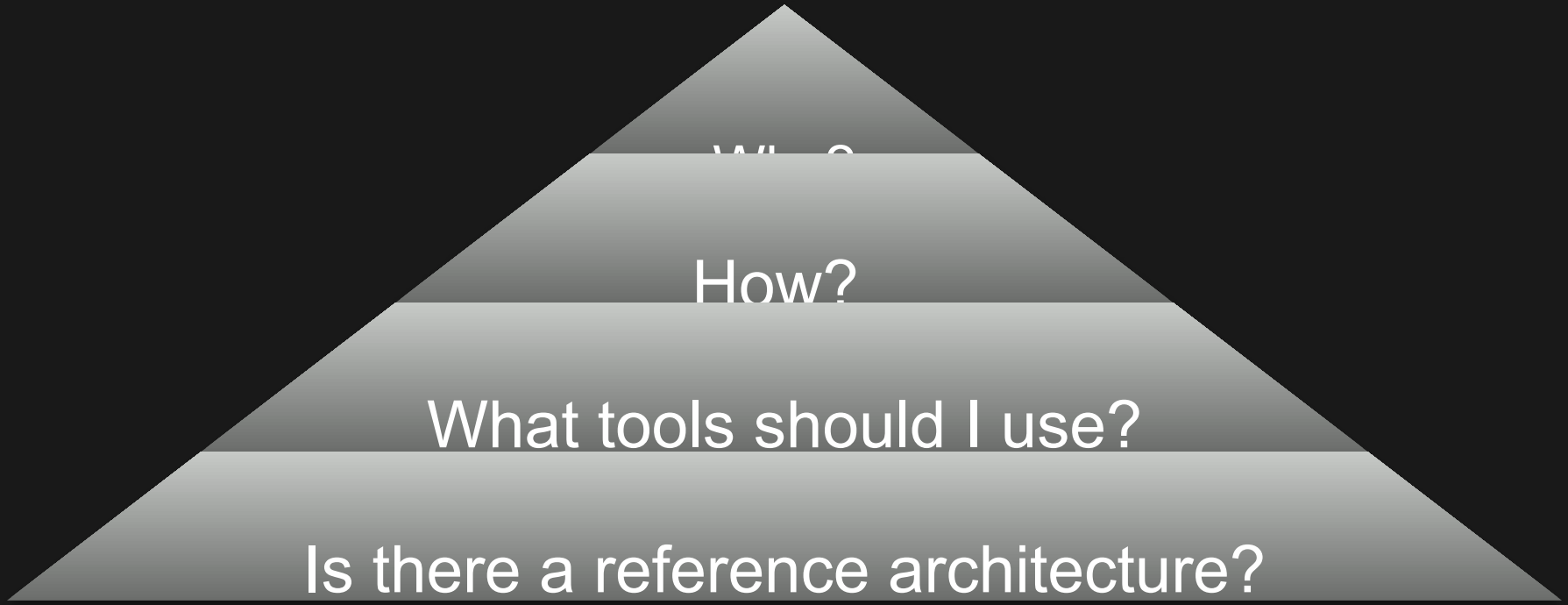
Managed services

Serverless

Plethora of Tools



Big Data Challenges



Architectural Principles

Build decoupled systems

- Data → Store → Process → Store → Analyze → Answers

Use the right tool for the job

- Data structure, latency, throughput, access patterns

Leverage managed and serverless services

- Scalable/elastic, available, reliable, secure, no/low admin

Use log-centric design patterns

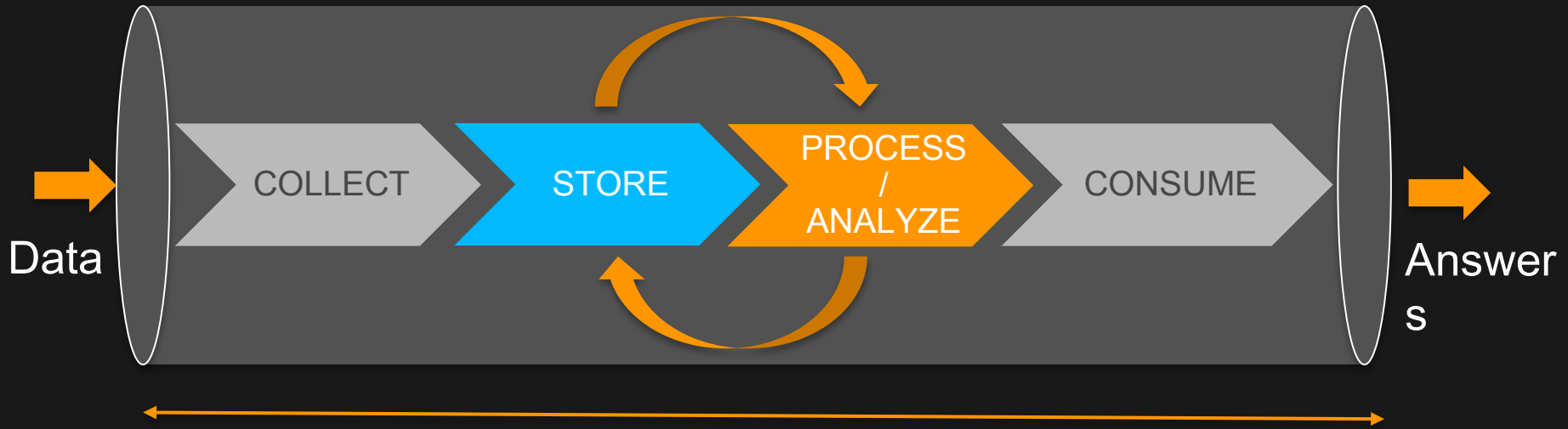
- Immutable logs (data lake), materialized views

Be cost-conscious

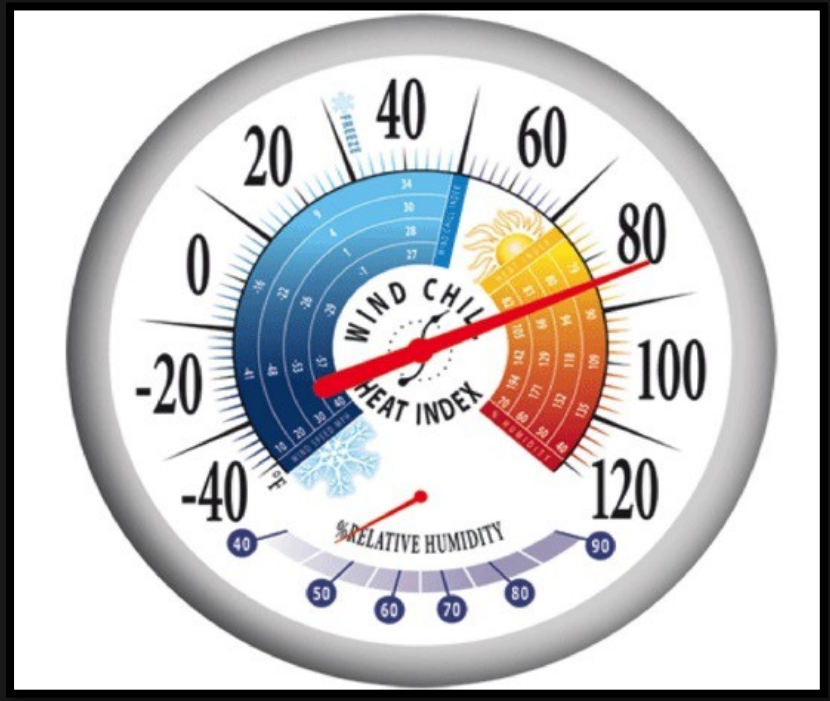
- Big data ≠ big cost

AI/ML enable your applications

Simplify Big Data Processing



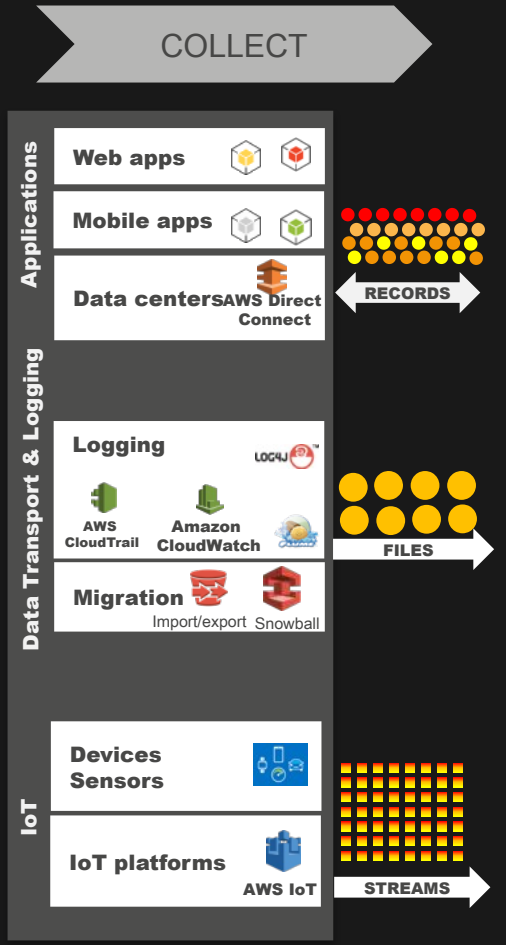
What Is the Temperature of Your Data?



Data Characteristics: Hot, Warm, Cold

	Hot	Warm	Cold
Volume	MB–GB	GB–TB	PB–EB
Item size	B–KB	KB–MB	KB–TB
Latency	ms	ms, sec	min, hrs
Durability	Low–high	High	Very high
Request rate	Very high	High	Low
Cost/GB	$$-$	$-¢¢	¢

Type of Data



Data structures

Database records

Transactions

Media files

Log files

Files

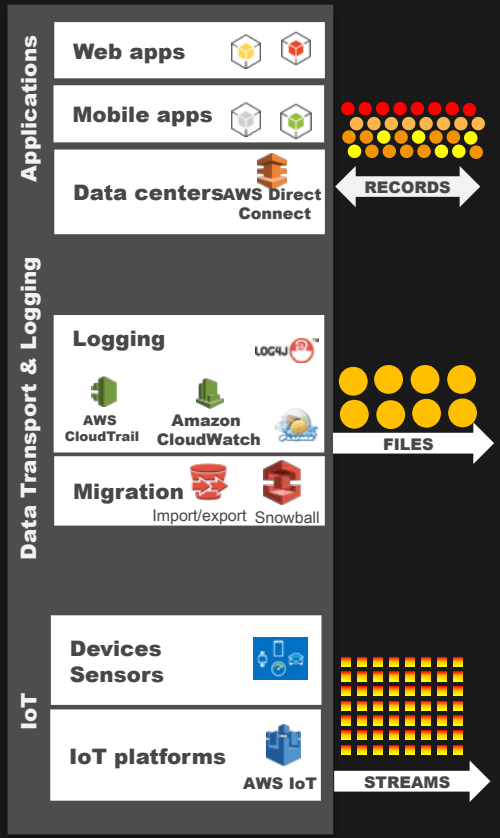
Data streams

Events

Type of Data

COLLECT

STORE



Data structures

Database records

Media files

Log files

Data streams

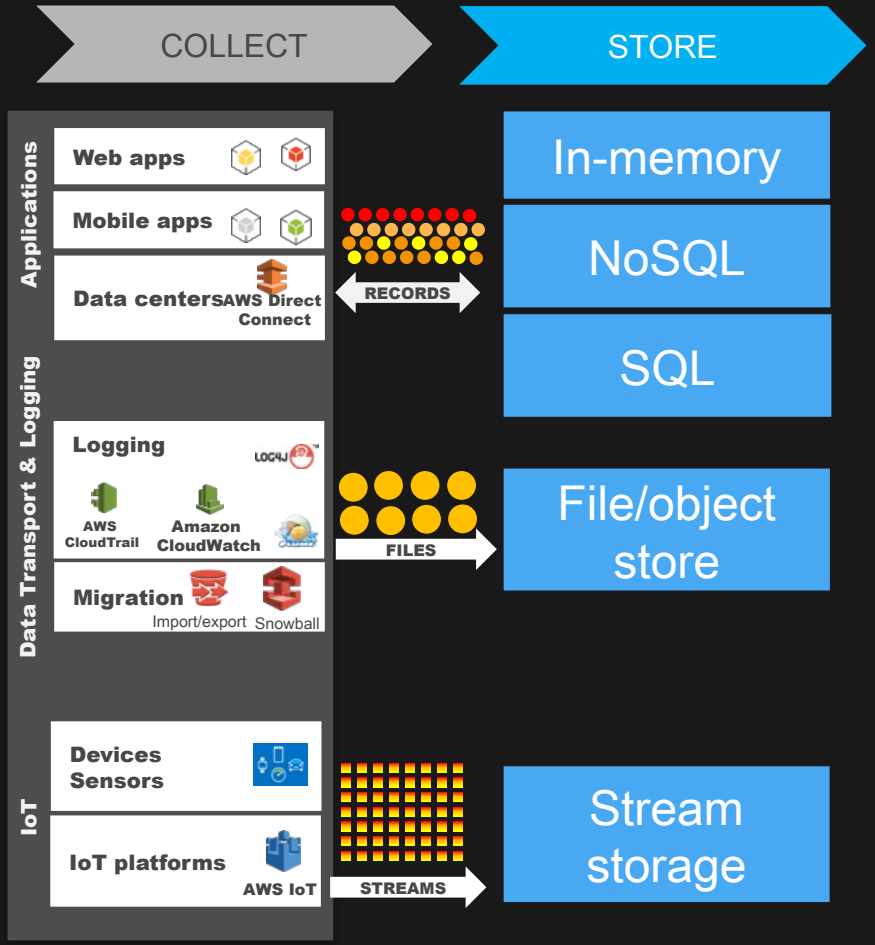
In-memory

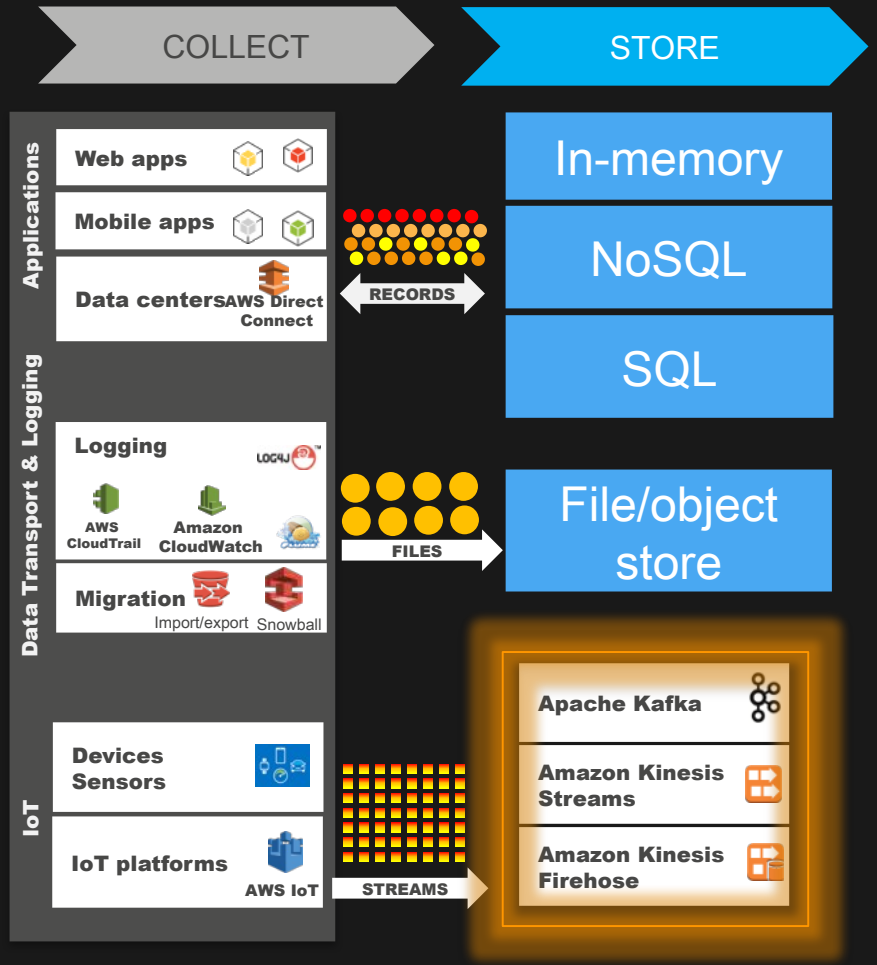
NoSQL

SQL

File/object store

Stream storage





Stream Storage

Apache Kafka

- High throughput distributed streaming platform

Amazon Kinesis Streams

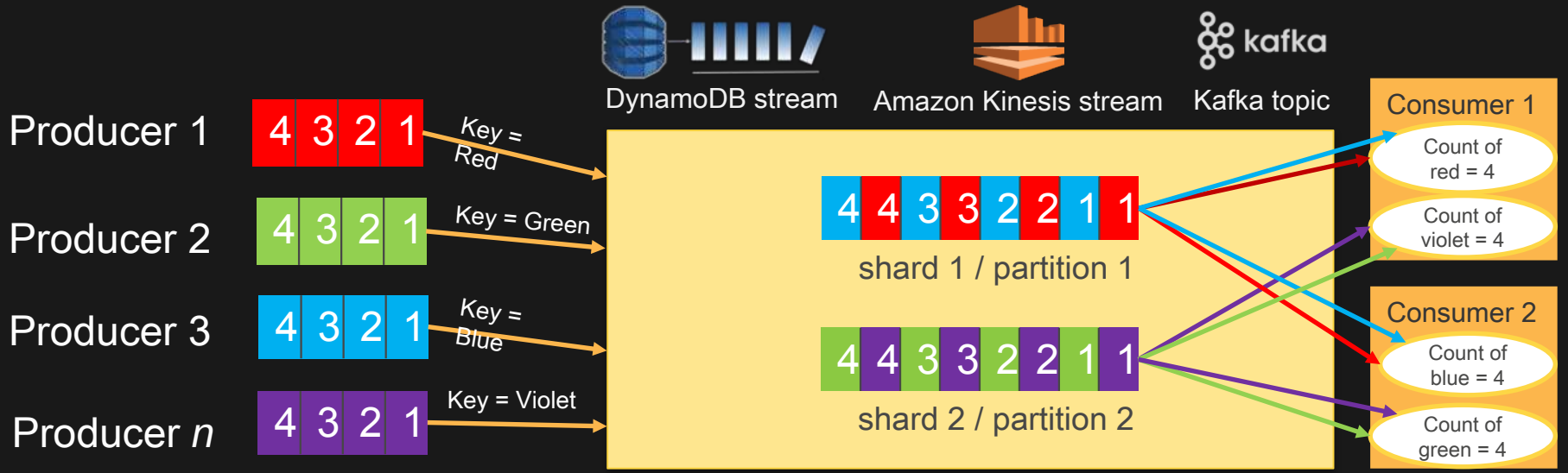
- Managed stream storage

Amazon Kinesis Firehose

- Managed data delivery

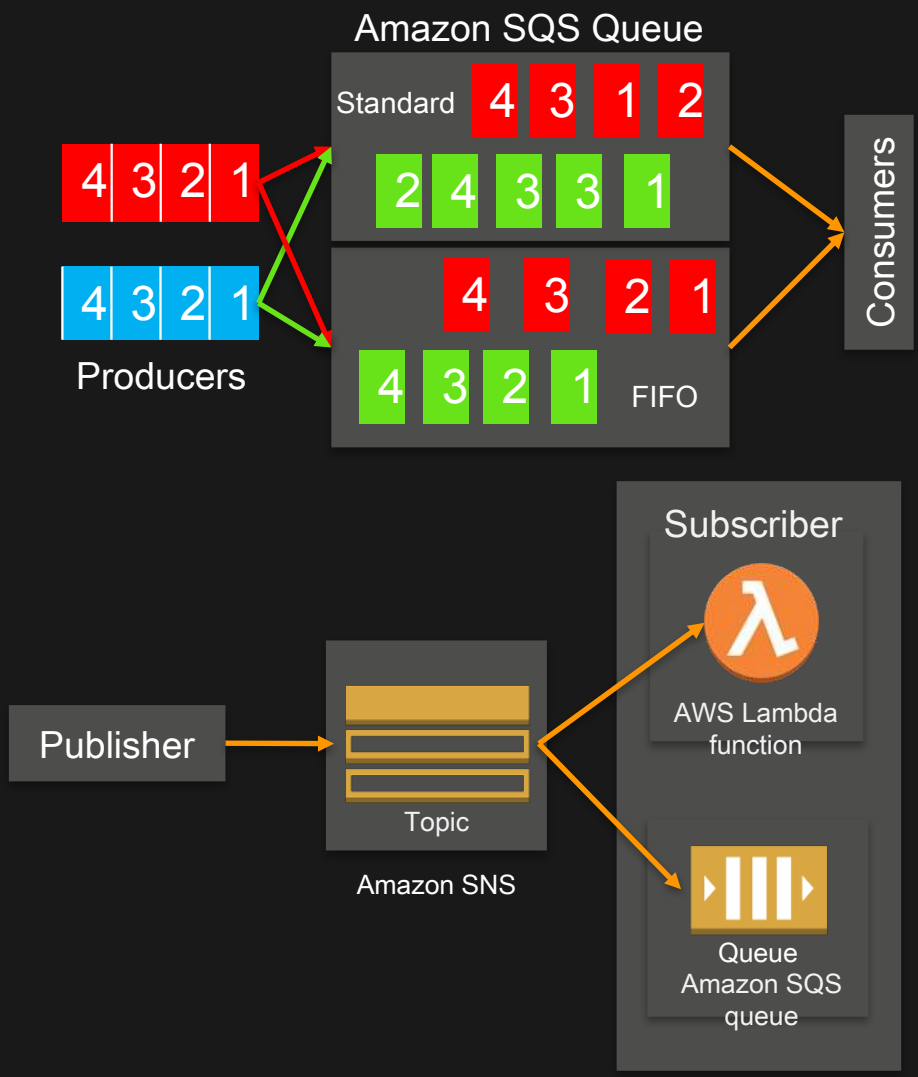
Why Stream Storage?

- Decouple producers & consumers
- Persistent buffer
- Collect multiple streams
- Preserve client ordering
- Parallel consumption
- Streaming MapReduce



What About Amazon SQS?

- Decouple producers & consumers
- Persistent buffer
- Collect multiple streams
- **No** client ordering (standard)
  - FIFO queue preserves client ordering
- **No** streaming MapReduce
- **No** parallel consumption
  - Amazon SNS can publish to multiple SNS subscribers (queues or Lambda functions)

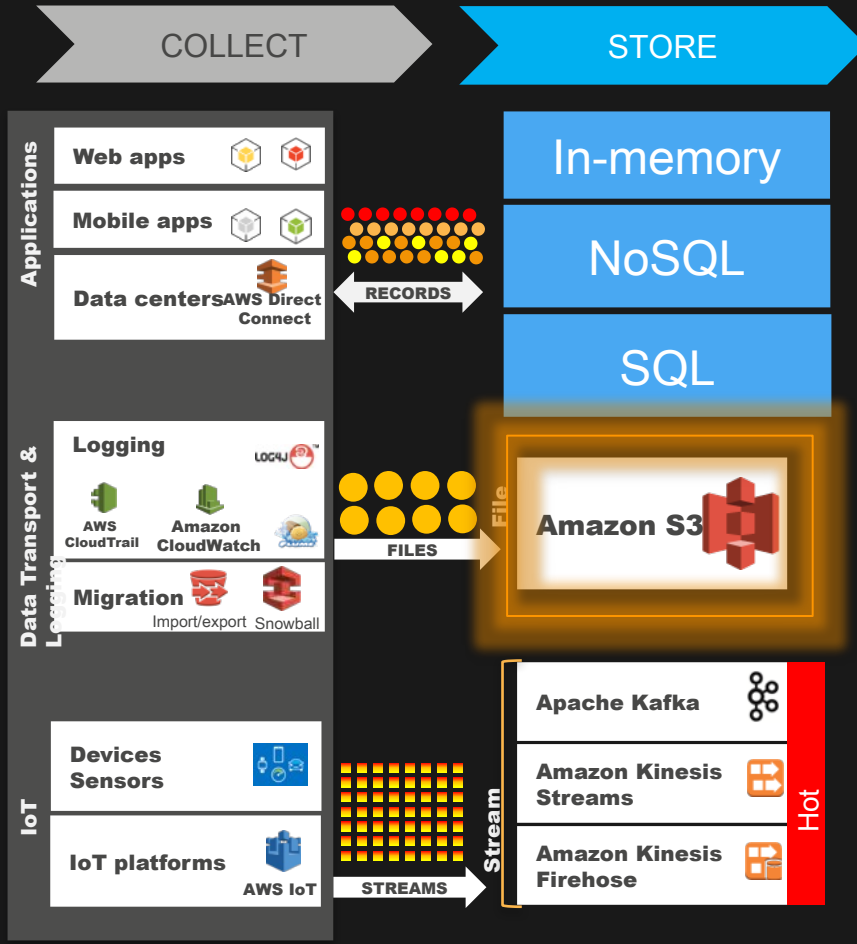


Which Stream/Message Storage Should I Use?

	Amazon Kinesis Streams	Amazon Kinesis Firehose	Apache Kafka (on Amazon EC2)	Amazon SQS (Standard)	Amazon SQS (FIFO)
AWS managed	Yes	Yes	No	Yes	Yes
Guaranteed ordering	Yes	No	Yes	No	Yes
Delivery (deduping)	At least once	At least once	At least/At most/exactly once	At least once	Exactly once
Data retention period	7 days	N/A	Configurable	14 days	14 days
Availability	3 AZ	3 AZ	Configurable	3 AZ	3 AZ
Scale / throughput	No limit / ~ shards	No limit / automatic	No limit / ~ nodes	No limits / automatic	300 TPS / queue
Parallel consumption	Yes	No	Yes	No	No
Stream MapReduce	Yes	N/A	Yes	N/A	N/A
Row/object size	1 MB	Destination row/object size	Configurable	256 KB	256 KB
Cost	Low	Low	Low (+admin)	Low-medium	Low-medium

Hot Warm

File/Object Storage



Amazon S3

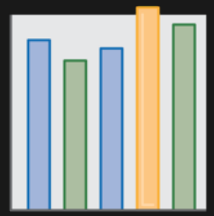
Managed object storage service built to store and retrieve any amount of data

Use Amazon S3 as Your Persistent File Store

- Natively supported by big data frameworks (Spark, Hive, Presto, etc.)
- Decouple storage and compute
  - No need to run compute clusters for storage (unlike HDFS)
  - Can run transient Amazon EMR clusters with Amazon EC2 Spot Instances
  - Multiple & heterogeneous analysis clusters and services can use the same data
- Designed for 99.999999999% durability
- No need to pay for data replication within a region
- Secure: SSL, client/server-side encryption at rest
- Low cost

What About HDFS & Data Tiering?

- Use HDFS for hottest datasets (e.g. iterative read on the same datasets)
- Use Amazon S3 Standard for frequently accessed data
- Use Amazon S3 Standard – IA for less frequently accessed data
- Use Amazon Glacier for archiving cold data
- Use S3 Analytics to optimize tiering strategy



S3 Analytics

Cache & Database

Amazon ElastiCache

- Managed Memcached or Redis service

Amazon DynamoDB Accelerator (DAX)

- Managed in-memory cache for DynamoDB

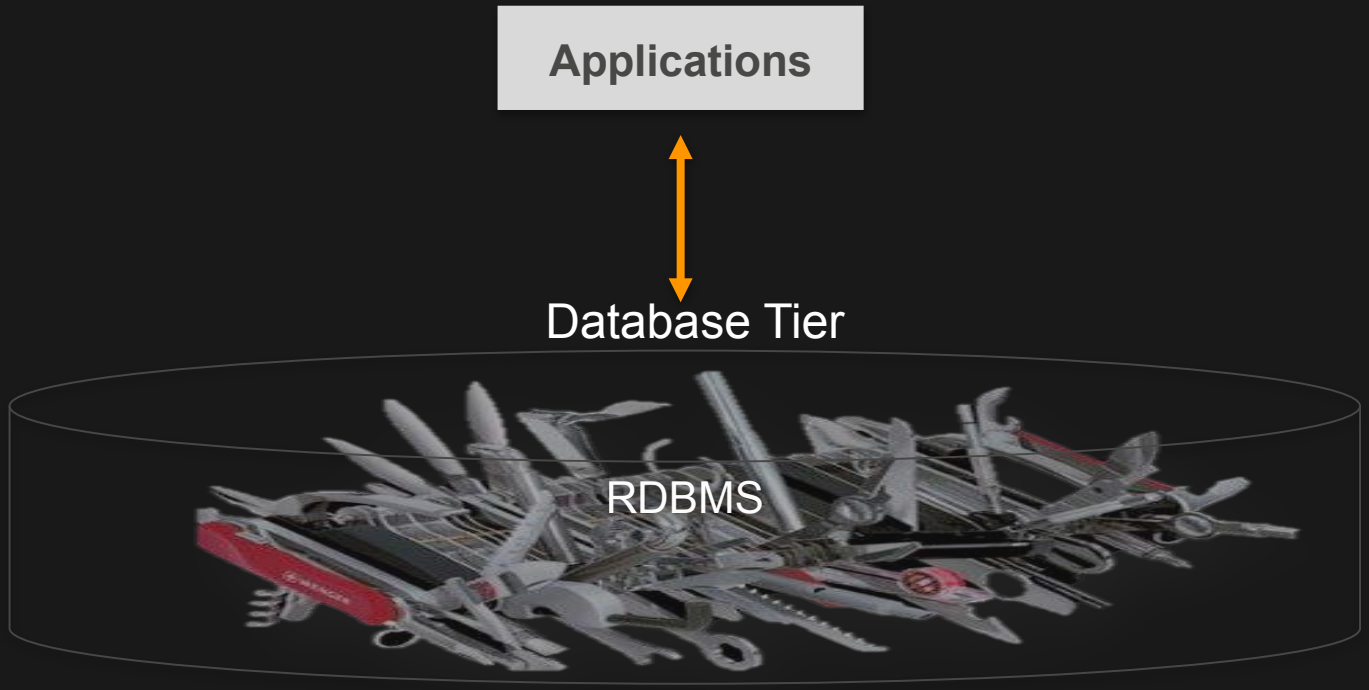
Amazon DynamoDB

- Managed NoSQL database service

Amazon RDS

- Managed relational database service

Anti-Pattern



Best Practice: Use the Right Tool for the Job

Applications

Database Tier

In-memory

Amazon ElastiCache

Amazon DynamoDB Acclerator

redis

SAP HANA

GraphDB

JanusGraph

neo4j

Apache TinkerPop

NoSQL

Amazon DynamoDB

APACHE HBASE

cassandra

mongoDB

SQL

Amazon RDS/Aurora

MySQL

ORACLE DATABASE

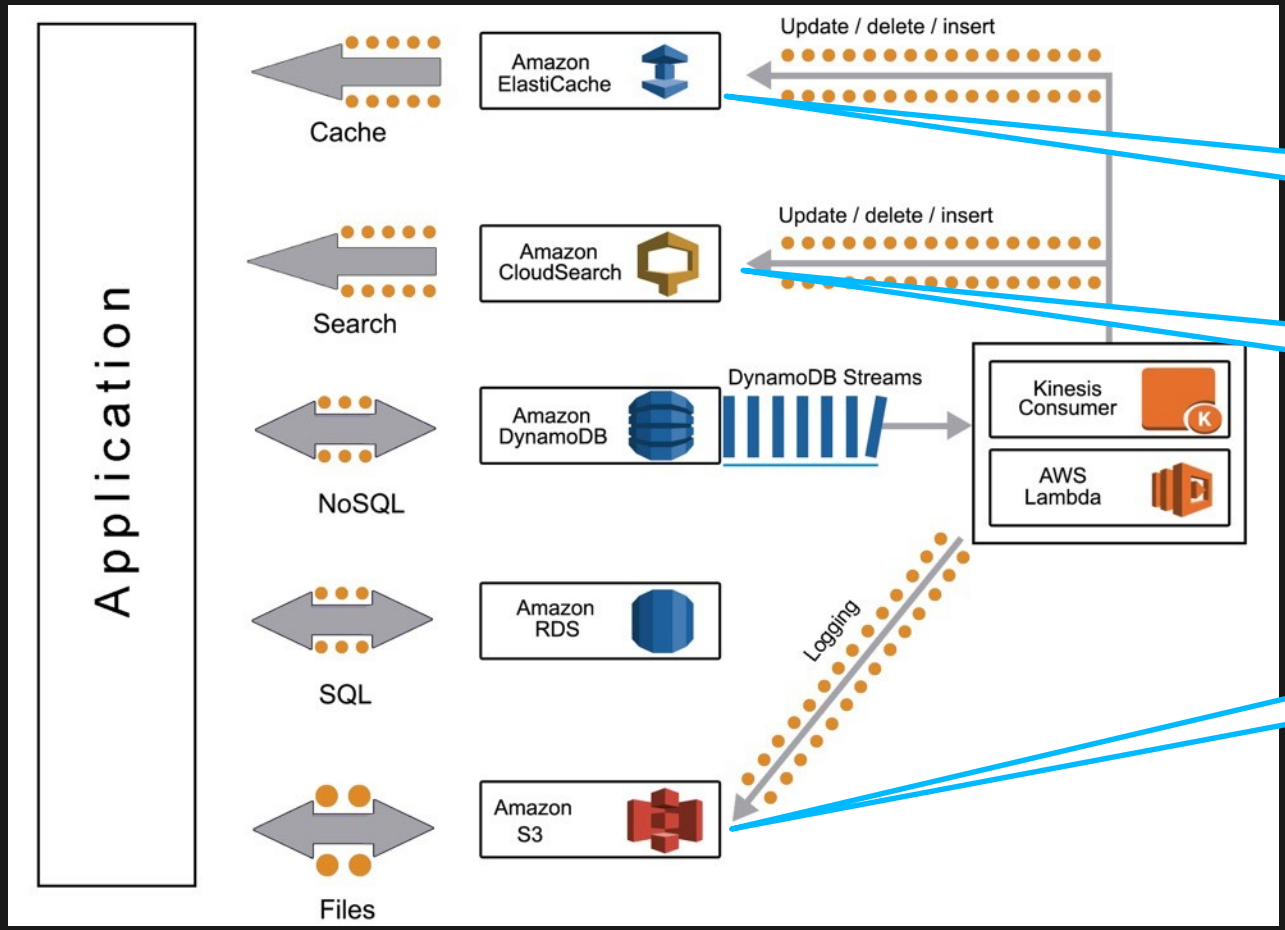
SQL Server 2017

Search

Amazon ES

Amazon CloudSearch

Materialized Views and Immutable Log



Which Data Store Should I Use?

Data structure → Fixed-schema, JSON, Key/Value,

Access patterns → Store data in the format you will access it

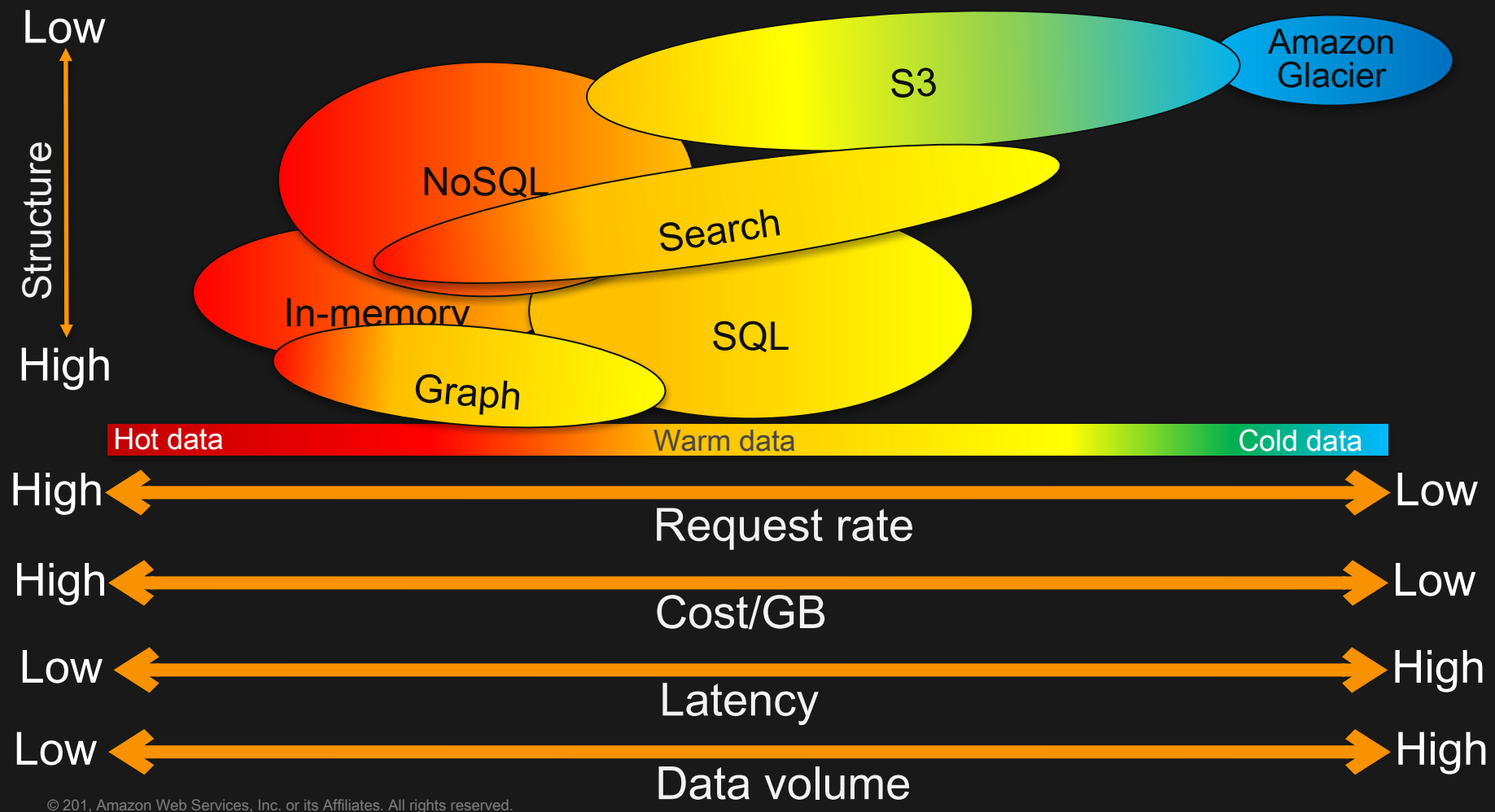
Data characteristics → Hot, warm, cold

Cost → Right cost

Data Structure and Access Patterns

Data Structure	What to use?
Fixed schema	SQL, NoSQL
Schema-free (JSON)	NoSQL, Search
Key/Value	In-memory, NoSQL
Graph	GraphDB

Access Patterns	What to use?
Put/Get (key, value)	In-memory, NoSQL
Simple relationships → 1:N, M:N	NoSQL
Multi-table joins, transaction, SQL	SQL
Faceting, Search	Search
Graph traversal	GraphDB



Which Data Store Should I Use?

	Amazon ElastiCache	Amazon DAX	Amazon DynamoDB	Amazon RDS (Aurora)	Amazon ES	Amazon S3	Amazon Glacier
Average Latency	µs-ms	µs-ms	ms	ms, sec	ms,sec	ms,sec,min (~ size)	hrs
Typical Data Volume	GB	GB	GB–TBs (no limit)	GB–TB (64 TB max)	GB–TB	MB–PB (no limit)	GB–PB (no limit)
Typical Item Size	B-KB	KB (400 KB max)	KB (400 KB max)	KB (64 KB max)	B-KB (2 GB max)	KB-TB (5 TB max)	GB (40 TB max)
Request Rate	High – very high	High – very high	Very high (no limit)	High	High	Low – high (no limit)	Very low
Storage Cost GB/Month	$$	$$	¢¢	¢¢	¢¢	¢	¢4/10
Durability	Low - moderate	NA	Very high	Very high	High	Very high	Very high
Availability	High 2 AZ	High 3 AZ	Very high 3 AZ	Very high 3 AZ	High 2 AZ	Very high 3 AZ	Very high 3 AZ

Hot data — Warm data — Cold data

Example: Should I use Amazon S3 or Amazon DynamoDB?

"I'm currently scoping out a project. The design calls for **many small files**, perhaps up to a **billion during peak**. The **total size** would be on the order of **1.5 TB per month**…"

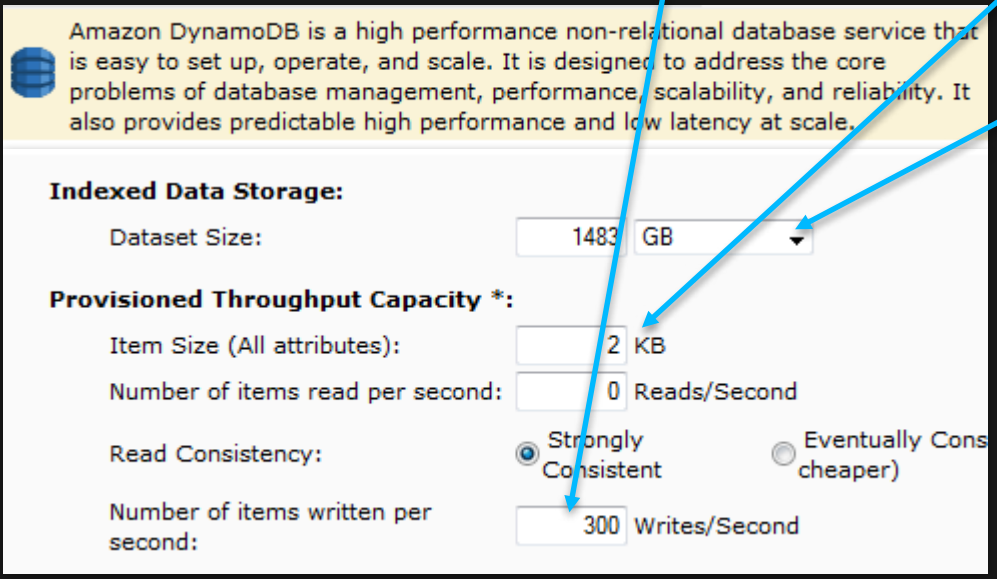
Request rate (Writes/sec)	Object size (Bytes)	Total size (GB/month)	Objects per month
300	2048	1483	777,600,000

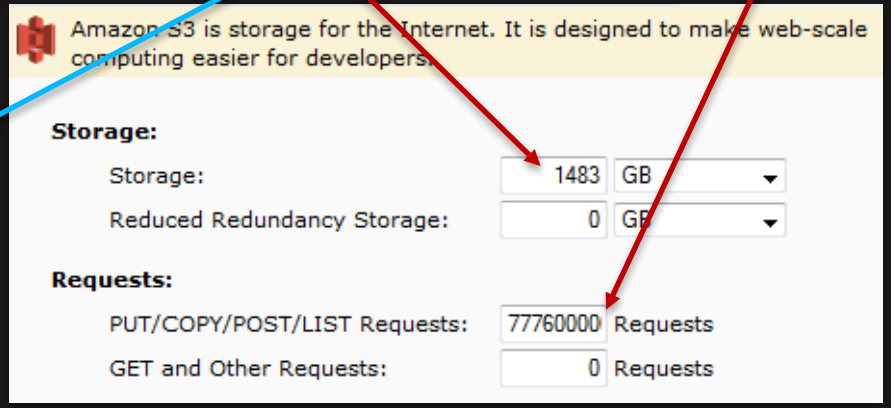
Amazon S3 or Amazon DynamoDB?

aws SIMPLE MONTHLY CALCULATOR

https://calculator.s3.amazonaws.com/index.html

Request rate (writes/sec)	Object size (bytes)	Total size (GB/month)	Objects per month
300	2,048	1,483	777,600,000



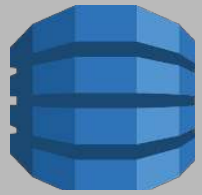


Amazon S3 or Amazon DynamoDB?

aws SIMPLE MONTHLY CALCULATOR

Amazon S3 Standard	
Storage	$34
Put/list requests	$3,888
Total	$3,922

Amazon DynamoDB	
Provisioned throughput	$273
Indexed data storage	$383
Total	$656



Amazon DynamoDB Wins!

	Request rate (writes/sec)	Object size (bytes)	Total size (GB/month)	Objects per month
Scenario 1	300	2,048	1,483	777,600,000
Scenario 2	300	32,768	23,730	777,600,000

Amazon S3 Wins!

Amazon S3 Standard	
Storage	$545
Put/List Requests	$3,888
Total	$4,433

Amazon DynamoDB	
Provisioned Throughput	$4,556
Indexed Data Storage	$5,944
Total	$10,500

Predictive Analytics

Amazon AI

API-driven Services

- Amazon Lex – Chatbots
- Amazon Polly & Transcribe – Speech
- Amazon Rekognition – Image analysis
- Amazon Translate & Comprehend -- NLP

Developers

Managed ML Platforms

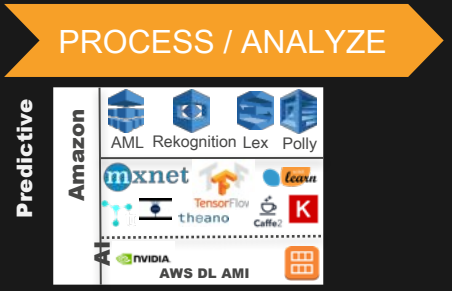
- Amazon SageMaker
- Apache Spark ML on EMR

Data scientists

AWS Deep Learning AMI

- Pre-installed with Apache MXNet, TensorFlow, Caffe, Theano, Torch, Microsoft Cognitive Toolkit, and Keras; plus DL tools/drivers

Deep learning experts



PROCESS / ANALYZE

Interactive and Batch Analytics

Amazon ES

- Managed Service for Elasticsearch

Amazon Redshift and Amazon Redshift Spectrum

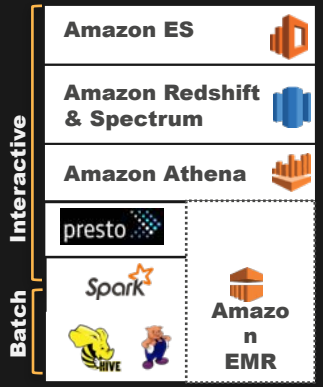
- Managed Data Warehouse
- Spectrum enables querying Amazon S3

Amazon Athena

- Serverless Interactive Query Service

Amazon EMR

- Managed Hadoop Framework for running Apache Spark, Flink, Presto, Tez, Hive, Pig, HBase, etc.



PROCESS / ANALYZE

Stream/Real-time Analytics

Spark Streaming on Amazon EMR

Amazon Kinesis Analytics

- Managed Service for running SQL on Streaming data

Amazon KCL

- Amazon Kinesis Client Library

AWS Lambda

- Run code Serverless (without provisioning or managing servers)
- Services such as S3 can publish events to Lambda
- Lambda can pool event from a Kinesis

Which Analytics Should I Use?

Batch

- Takes minutes to hours
- Example: Daily/weekly/monthly reports
- Amazon EMR (MapReduce, Hive, Pig, Spark)

Interactive

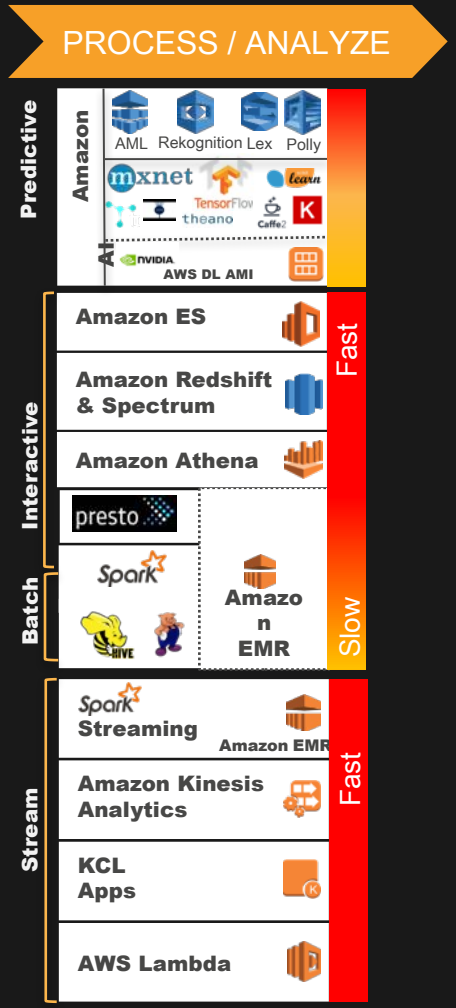
- Takes seconds
- Example: Self-service dashboards
- Amazon Redshift, Amazon Athena, Amazon EMR (Presto, Spark)

Stream

- Takes milliseconds to seconds
- Example: Fraud alerts, 1 minute metrics
- Amazon EMR (Spark Streaming), Amazon Kinesis Analytics, KCL, AWS Lambda, etc.

Predictive

- Takes milliseconds (real-time) to minutes (batch)
- Example: Fraud detection, Forecasting demand, Speech recognition
- Amazon AI (Lex, Polly, ML, Amazon Rekognition), Amazon EMR (Spark ML), Deep Learning AMI (MXNet, TensorFlow, Theano, Torch, CNTK, and Caffe)



Which Stream Processing Technology Should I Use?

	Amazon EMR (Spark Streaming)	KCL Application	Amazon Kinesis Analytics	AWS Lambda
Managed Service	Yes	No (EC2 + Auto Scaling)	Yes	Yes
Serverless	No	No	Yes	Yes
Scale/Throughput	No limits / ~ nodes	No limits / ~ nodes	No limits / automatic	No limits / automatic
Availability	Single AZ	Multi-AZ	Multi-AZ	Multi-AZ
Programming Languages	Java, Python, Scala	Java, others via MultiLangDaemon	ANSI SQL with extensions	Node.js, Java, Python
Sliding Window Functions	Build-in	App needs to implement	Built-in	No
Reliability	KCL and Spark	Managed by KCL	Managed by Analytics	Managed by AWS Lambda

Fast

Which Analytics Tool Should I Use?

	Amazon Redshift	Amazon Redshift Spectrum	Amazon Athena	Amazon EMR		
				Presto	Spark	Hive
Use case	Optimized for data warehousing	Query S3 data from Amazon Redshift	Interactive Queries over S3 data	Interactive Query	General purpose	Batch
Scale/Throughput	~Nodes	~Nodes	Automatic	~ Nodes		
Managed Service	Yes	Yes	Yes, Serverless	Yes		
Storage	Local storage	Amazon S3	Amazon S3	Amazon S3, HDFS		
Optimization	Columnar storage, data compression, and zone maps	AVRO, PARQUET TEXT, SEQ RCFILE, ORC, etc.	AVRO, PARQUET TEXT, SEQ RCFILE, ORC, etc.	Framework dependent		
Metadata	Amazon Redshift Catalog	Glue Catalog	Glue Catalog	Glue Catalog or Hive Meta-store		
Auth/Access controls	IAM, Users, groups, and access controls	IAM, Users, groups, and access controls	IAM	IAM, LDAP & Kerberos		
UDF support						

Fastest — Fast — Slow

What About Extract Transform and Load?

Data Integration Partners

Reduce the effort to move, cleanse, synchronize, manage, and automatize data-related processes.

alteryx | ATTUNITY | informatica | ironSource .atom | MATILLION | snapLogic | bryte | talend | alooma

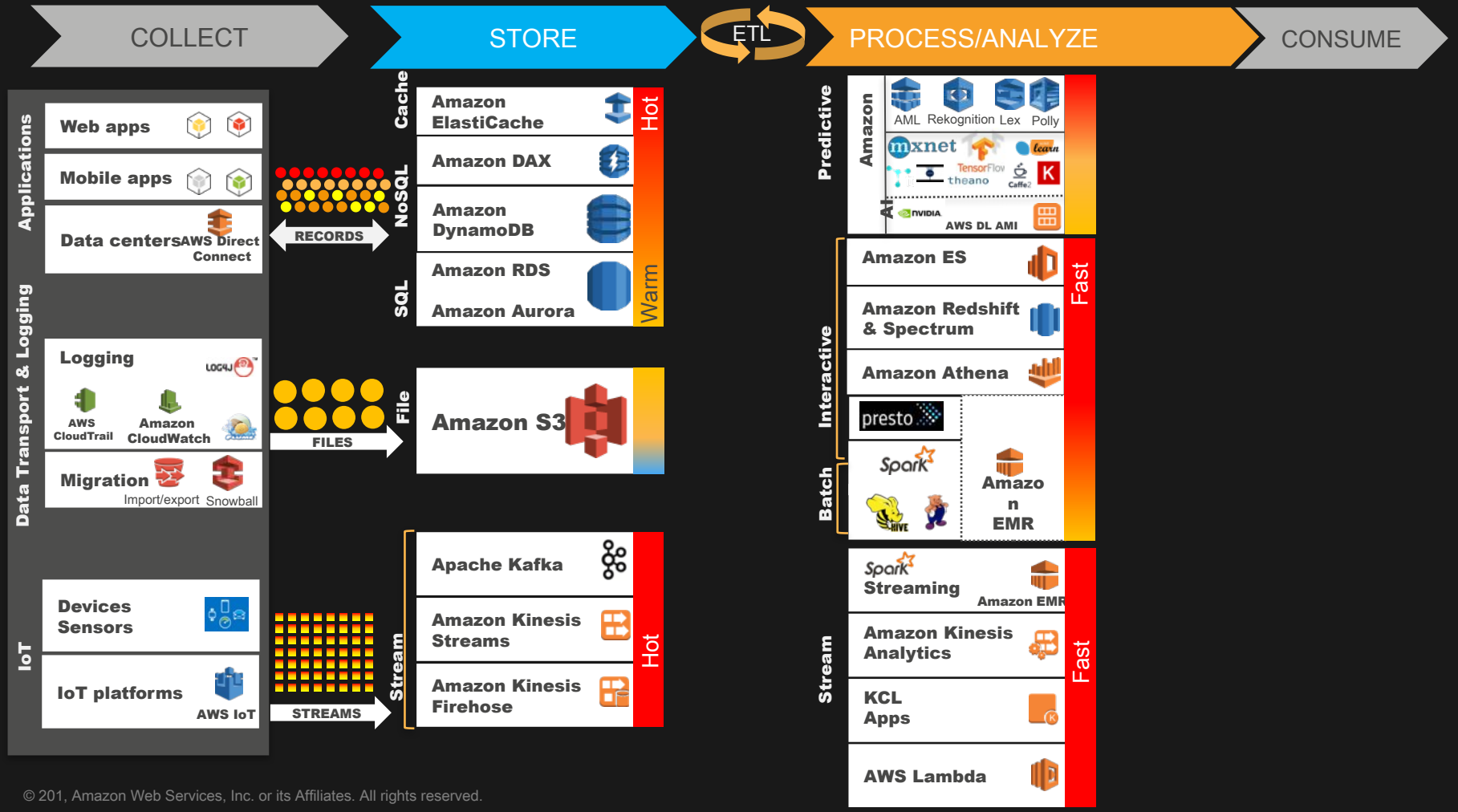
AWS Glue

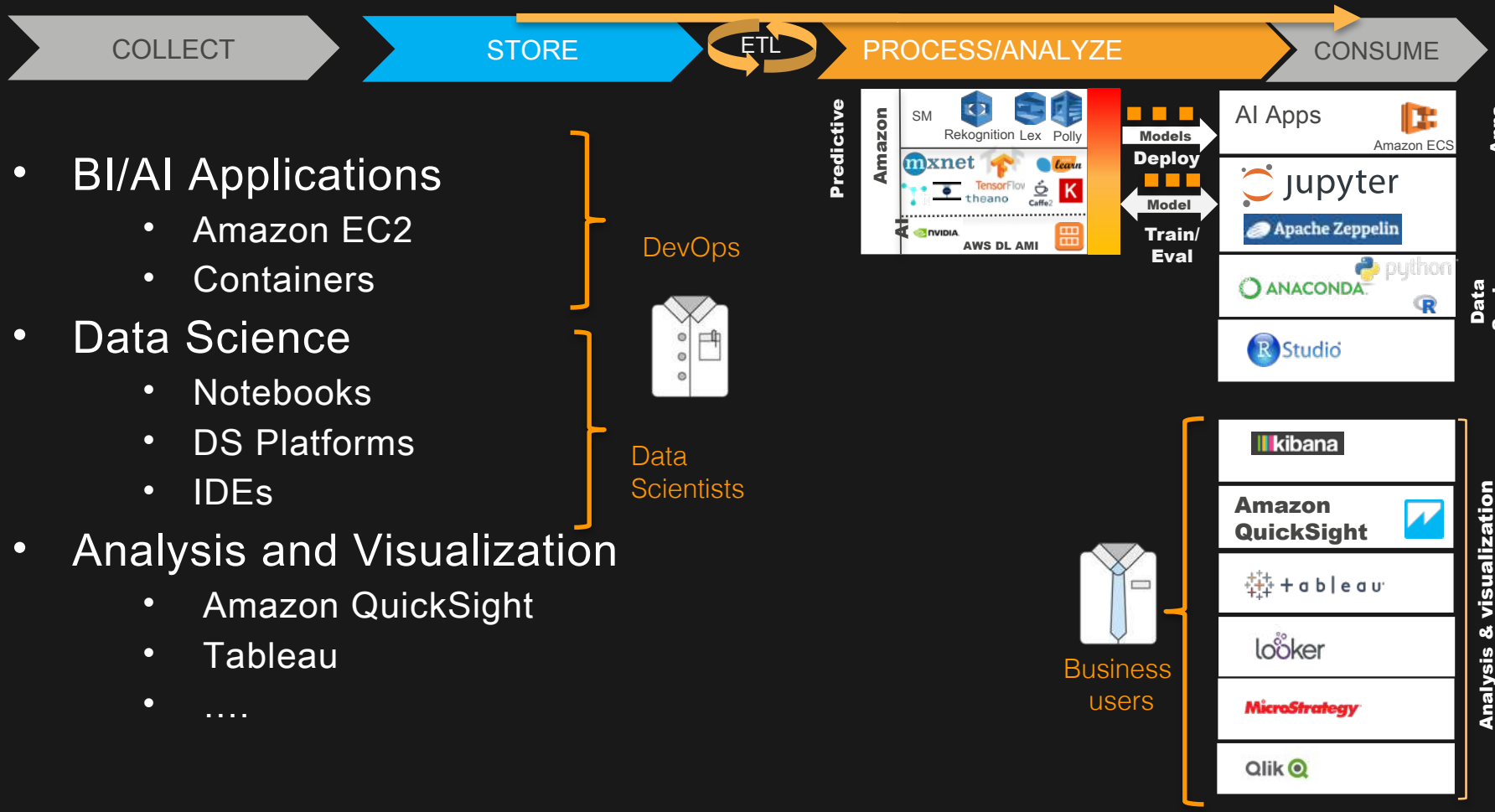
Data Catalog

Job Authoring

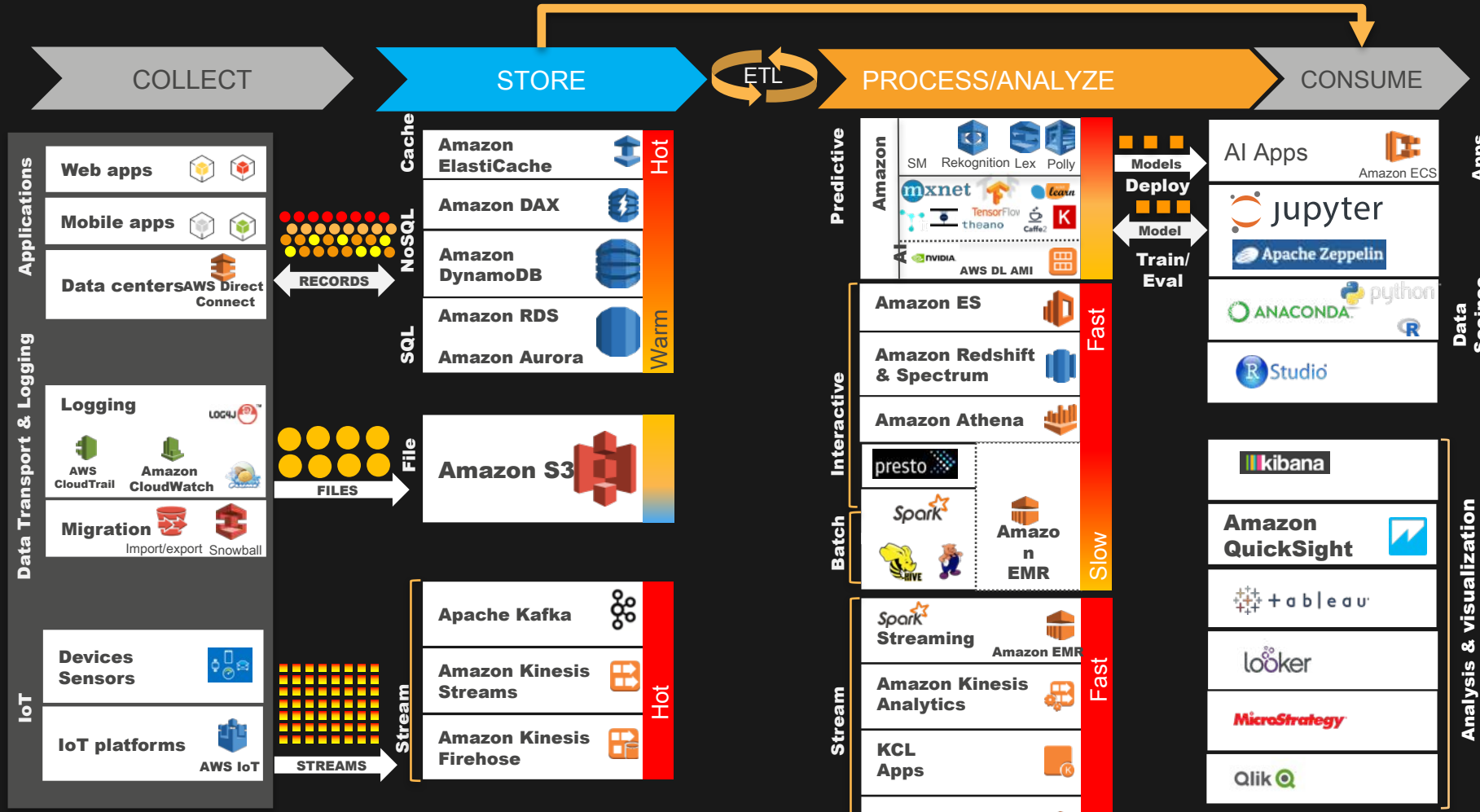
Job Execution

AWS Glue is a fully managed (serverless) ETL service that makes it simple and cost-effective to categorize your data, clean it, enrich it, and move it reliably between various data stores.



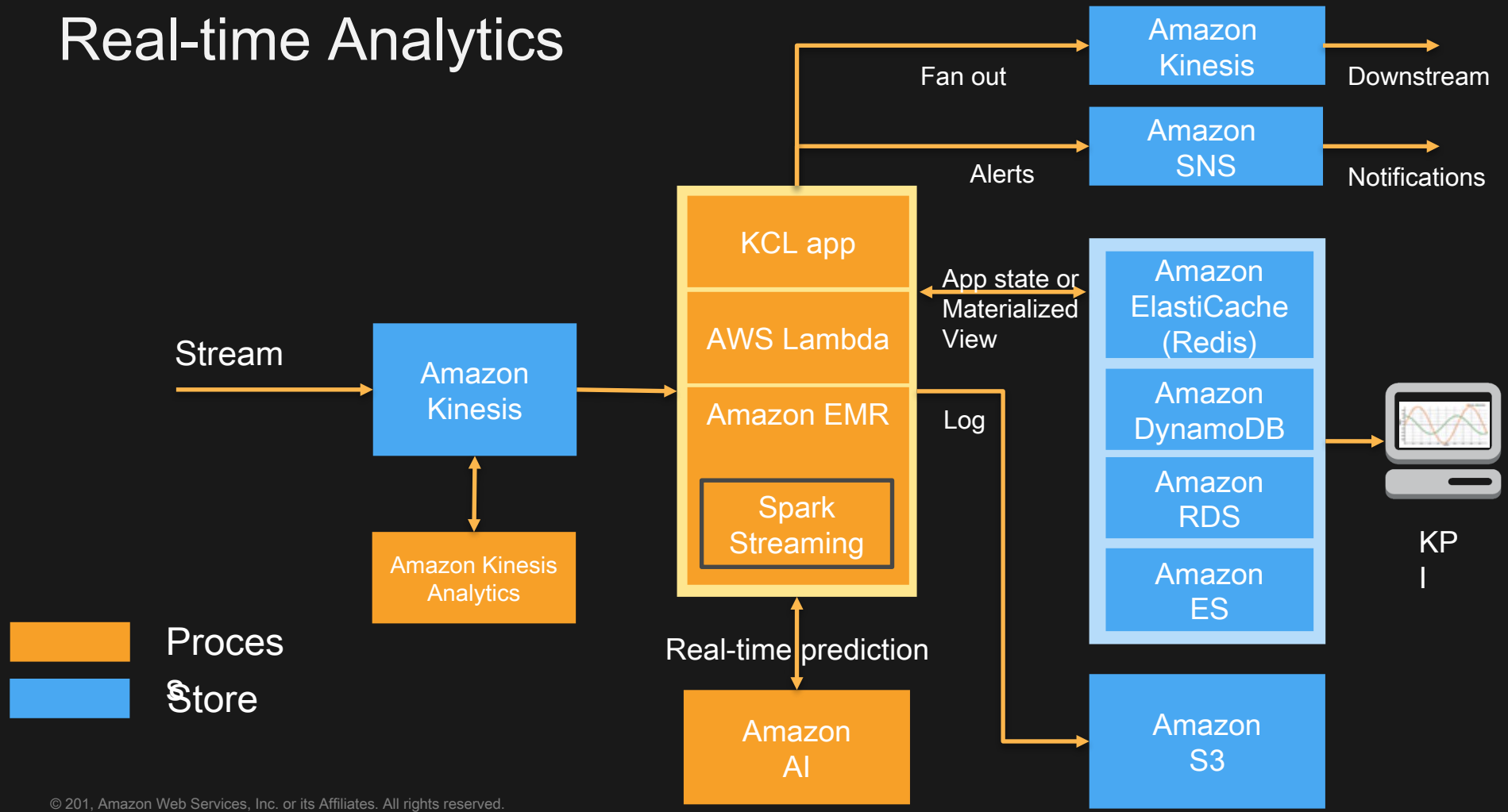


Putting It All Together

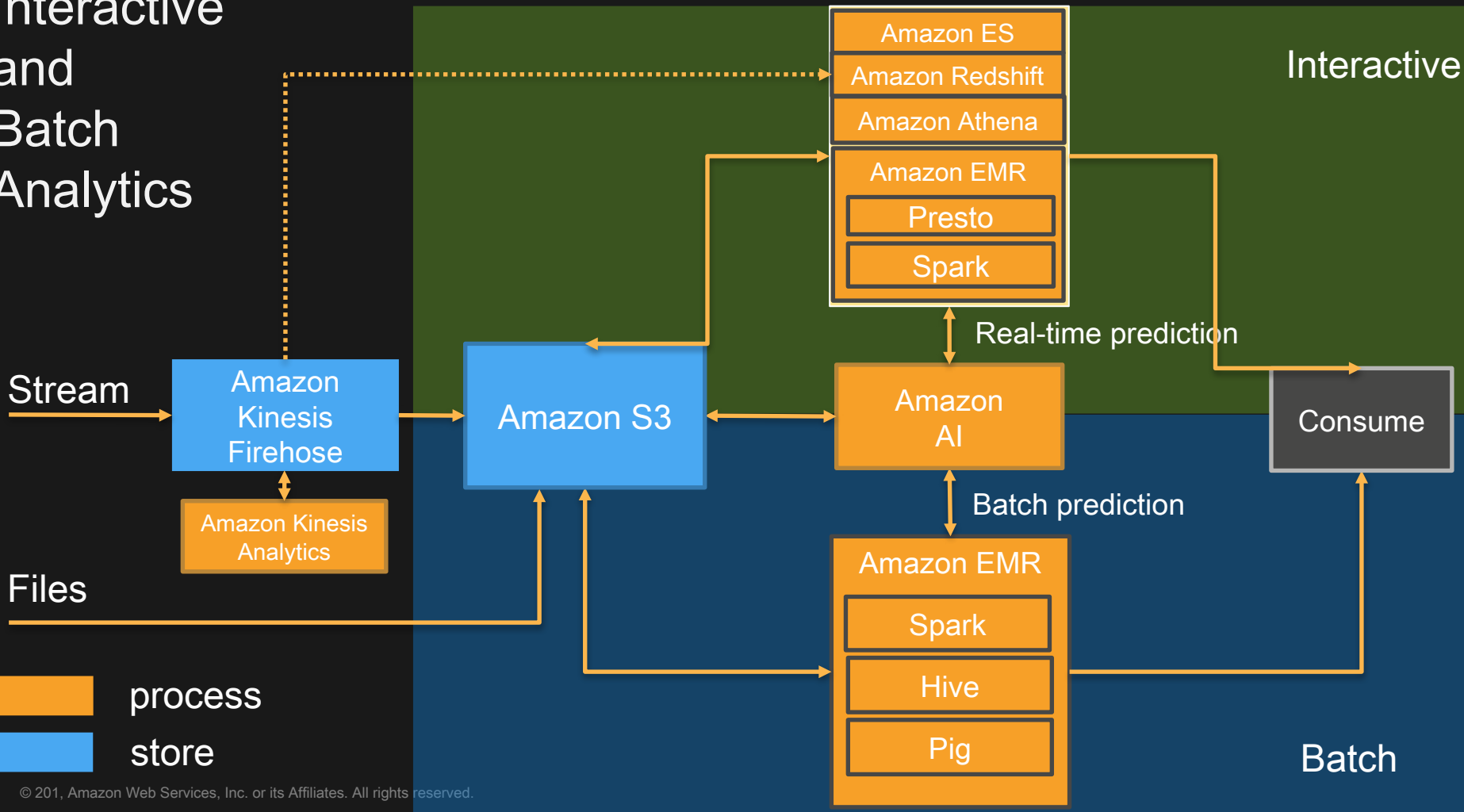


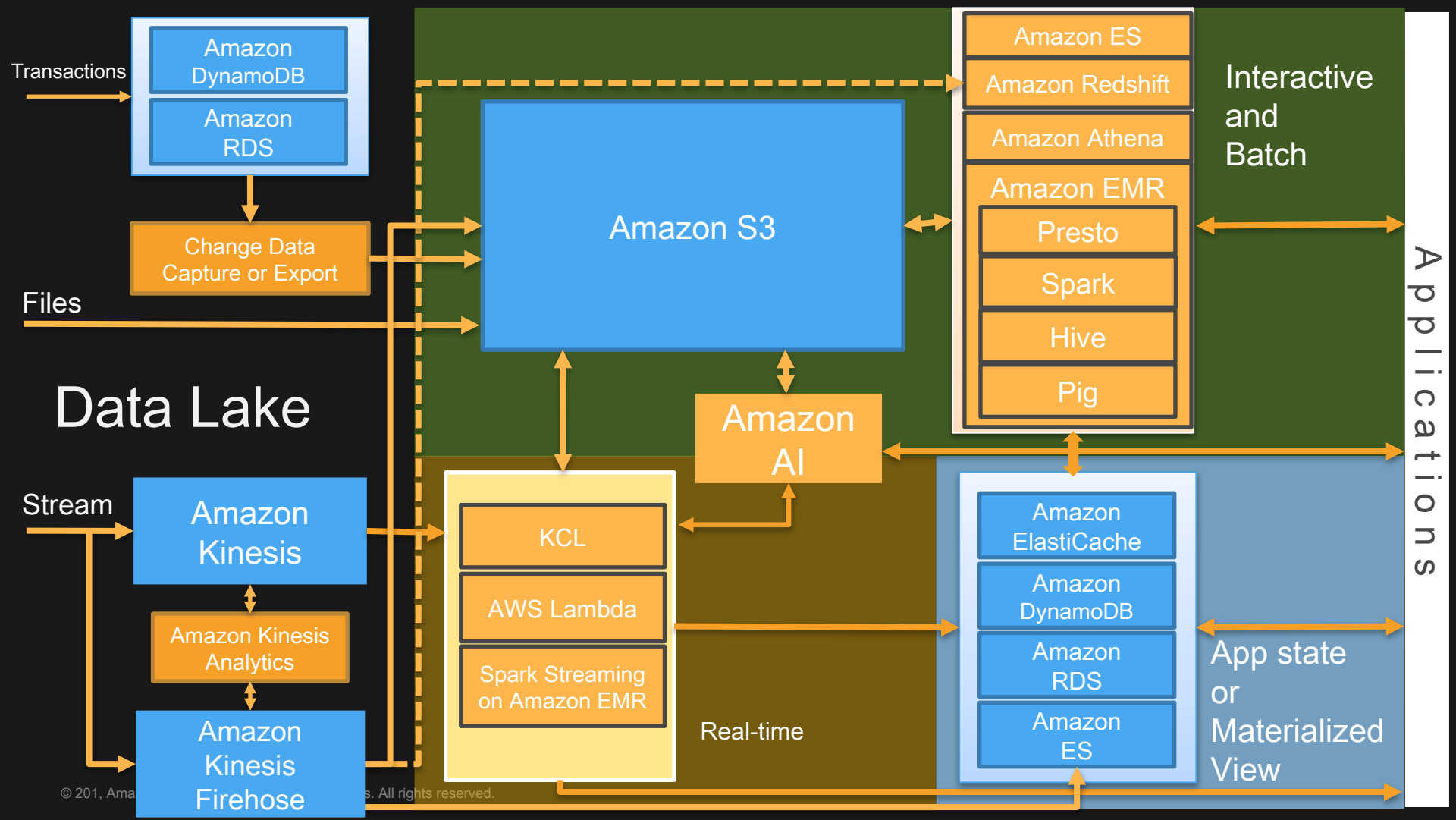
Design Patterns

Real-time Analytics



Interactive and Batch Analytics



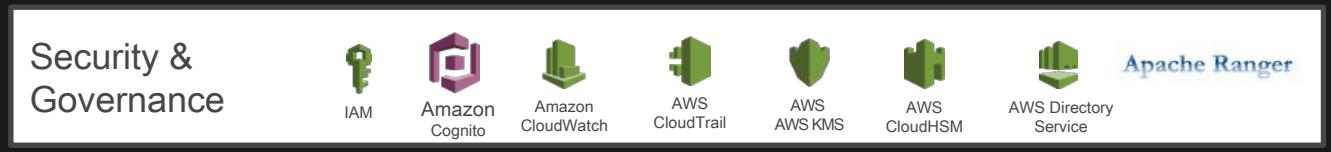


What about Metadata?

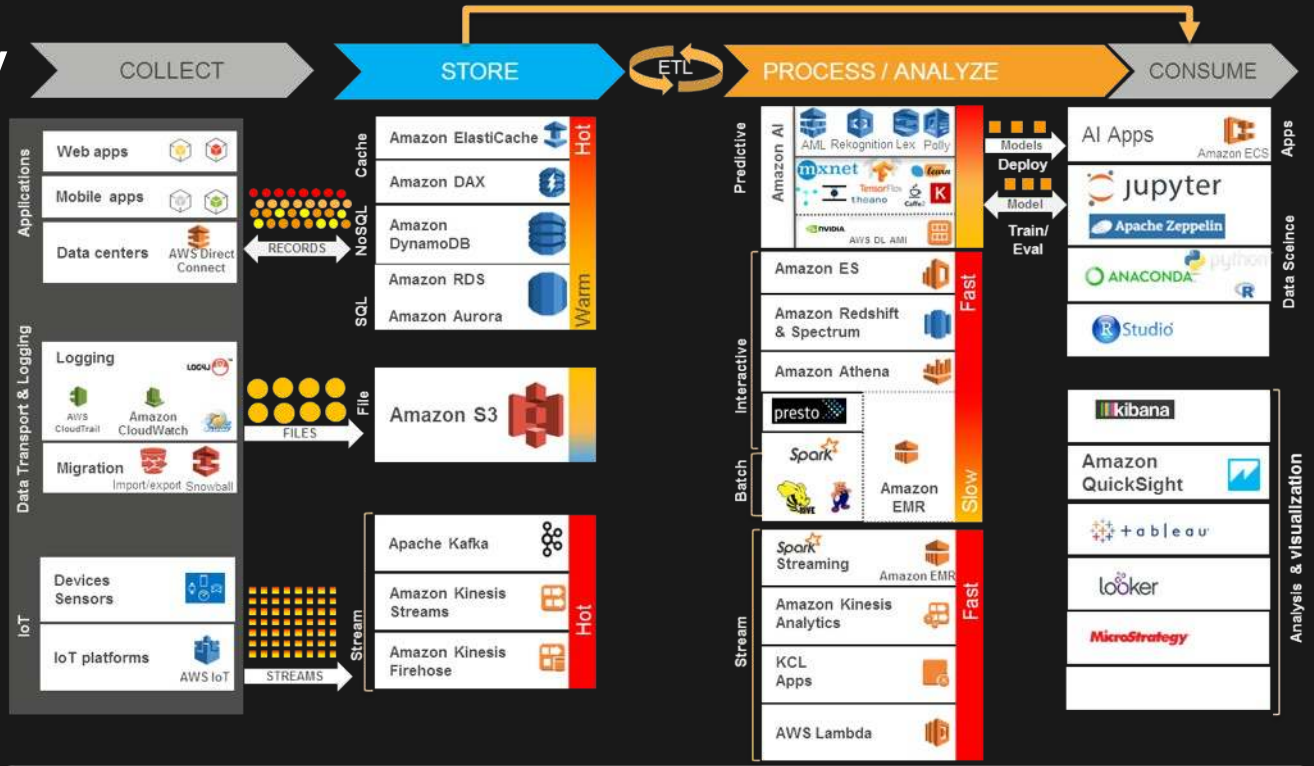
- Glue Catalog
  - Hive Metastore compliant
  - Crawlers - Detect new data, schema, partitions
  - Search - Metadata discovery
  - Amazon Athena, Amazon EMR, and Amazon Redshift Spectrum compatible
- Hive Metastore (Presto, Spark, Hive, Pig)
  - Can be hosted on Amazon RDS

Security & Governance

- AWS Identity and Access Management (IAM)
- Amazon Cognito
- Amazon CloudWatch & AWS CloudTrail
- AWS KMS
- AWS Directory Service
- Apache Ranger



Summary



Data lake Reference Architecture

Summary

Build decoupled systems

- Data → Store → Process → Store → Analyze → Answers

Use the right tool for the job

- Data structure, latency, throughput, access patterns

Leverage AWS managed and serverless services

- Scalable/elastic, available, reliable, secure, no/low admin

Use log-centric design patterns

- Immutable logs, data lake, materialized views

Be cost-conscious

- Big data ≠ Big cost

AI/ML enable your applications

Resources

https://aws.amazon.com/big-data/

https://aws.amazon.com/whitepapers/

https://aws.amazon.com/blogs/big-data/

https://aws.amazon.com/blogs/big-data/aws-big-data-analytics-sessions-at-reinvent-2017/

Thank you!

Julien Simon
Principal Technical Evangelist
@julsimon