AWS re:Invent

ANT201

Big Data Analytics Architectural Patterns and Best Practices

Ben Snively
Solutions Architect
Amazon Web Services

AWS re:Invent

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

Customer examples

Demonstrations

Types of big data analytics

Batch/ interactive

Stream processing

Machine learning

Delivery of big data services

Virtualized

Managed services

Serverless/ clusterless/ containerized

Plethora of tools

hadoop
Spark
scikit learn
HIVE
theano
MLlib
K
mxnet
TensorFlow™
APACHE HBASE
cassandra
Caffe2
presto
redis
JanusGraph
MEMCACHED
kafka
kibana
elasticsearch
FLUME
neo4j
MySQL
ORACLE DATABASE
Apache Zeppelin
R Studio
jupyter

Amazon Kinesis Data Analytics

Amazon Athena

Amazon Redshift

Amazon QuickSight

AWS Data Pipeline

Amazon SageMaker

Amazon Comprehend

Amazon EMR

Amazon Elasticsearch Service

AWS Glue

Amazon Polly

Amazon Rekognition

Amazon Translate

Amazon Transcribe

AWS Deep Learning AMIs

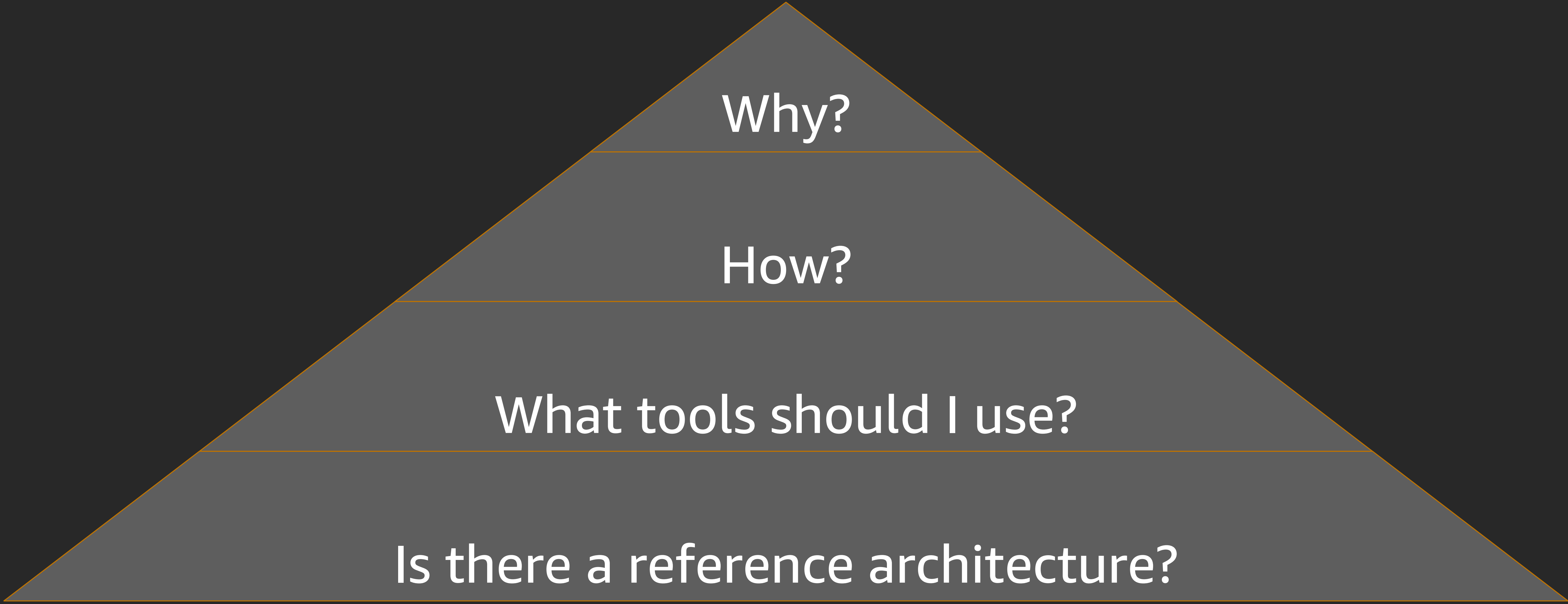
Amazon Aurora

Amazon DynamoDB

Amazon ElastiCache

Amazon Neptune

Big data challenges



Why?

How?

What tools should I use?

Is there a reference architecture?

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 event-journal design patterns

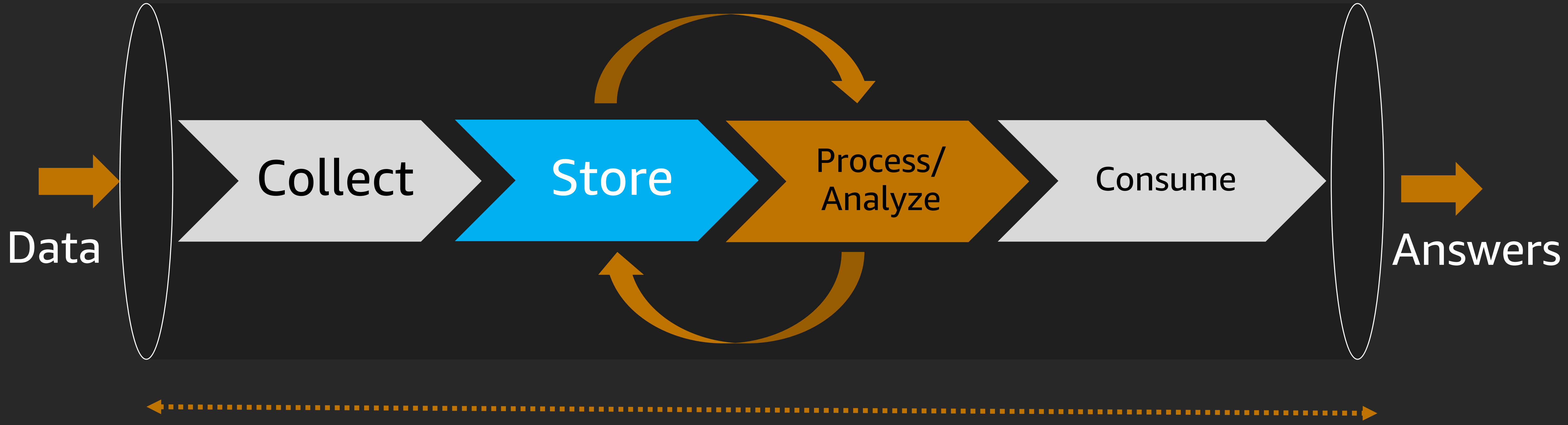
- Immutable datasets (data lake), materialized views

Be cost-conscious

- Big data ≠ big cost

Machine learning (ML) enable your applications

Simplify big data processing



Time to answer (latency)
Throughput
Cost

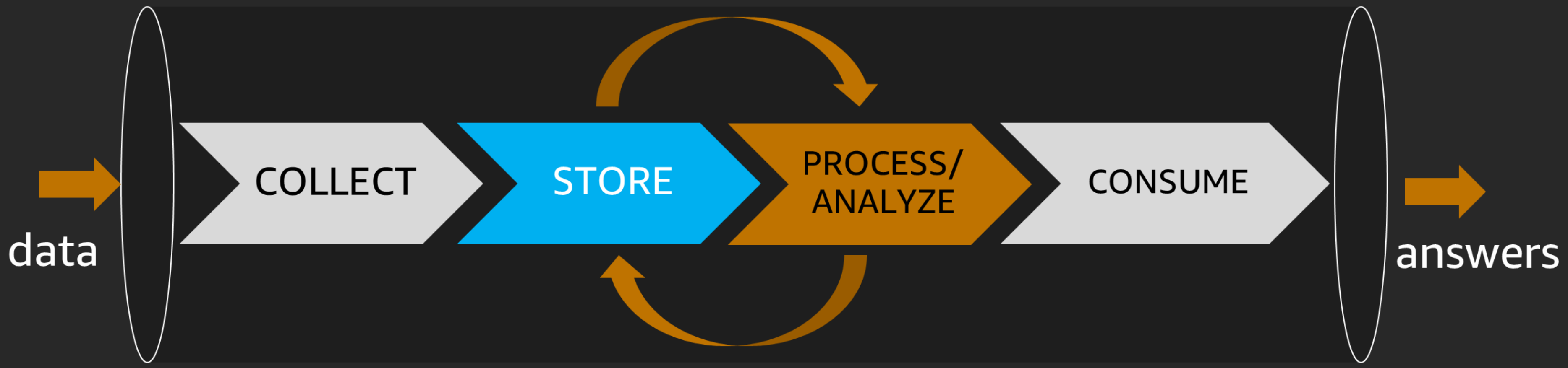
What is the temperature of your data/analytic?

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

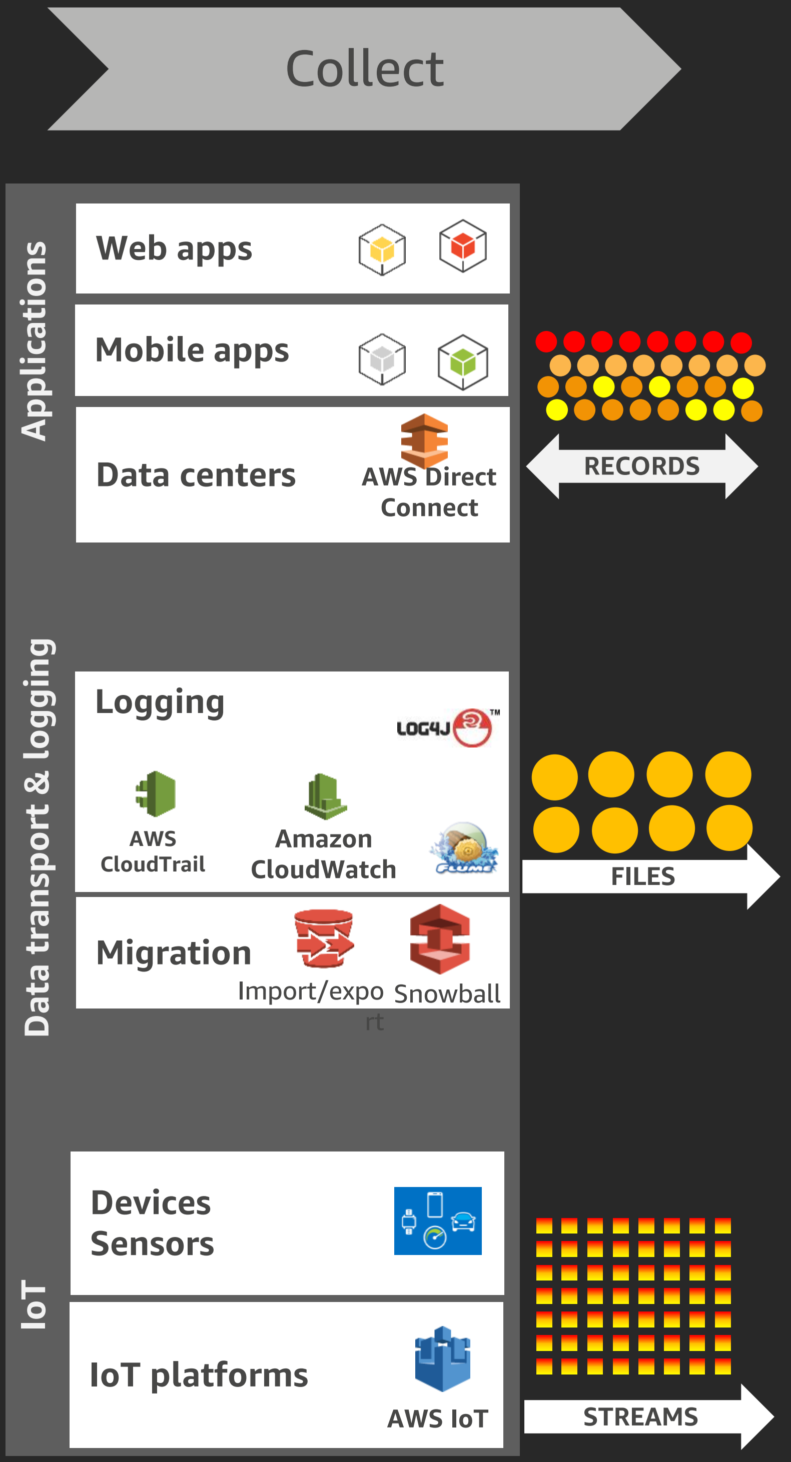
Hot data

Warm data

Cold data



Collect



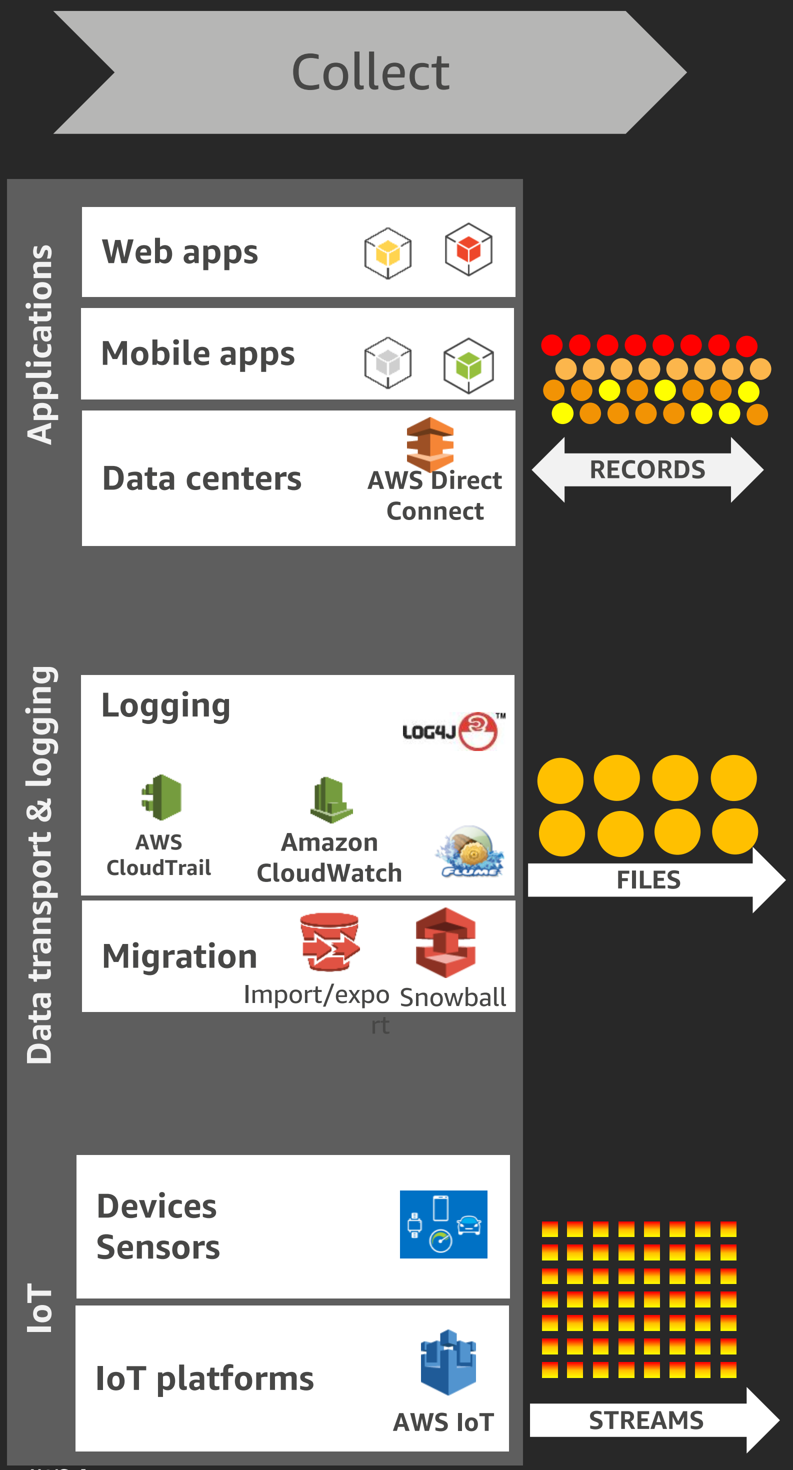
Type of data

Data structures
Database records
→ Transactions

Media files
Log files
→ Files/objects

Data streams
→ Events

Type of data



Data structures

Database records

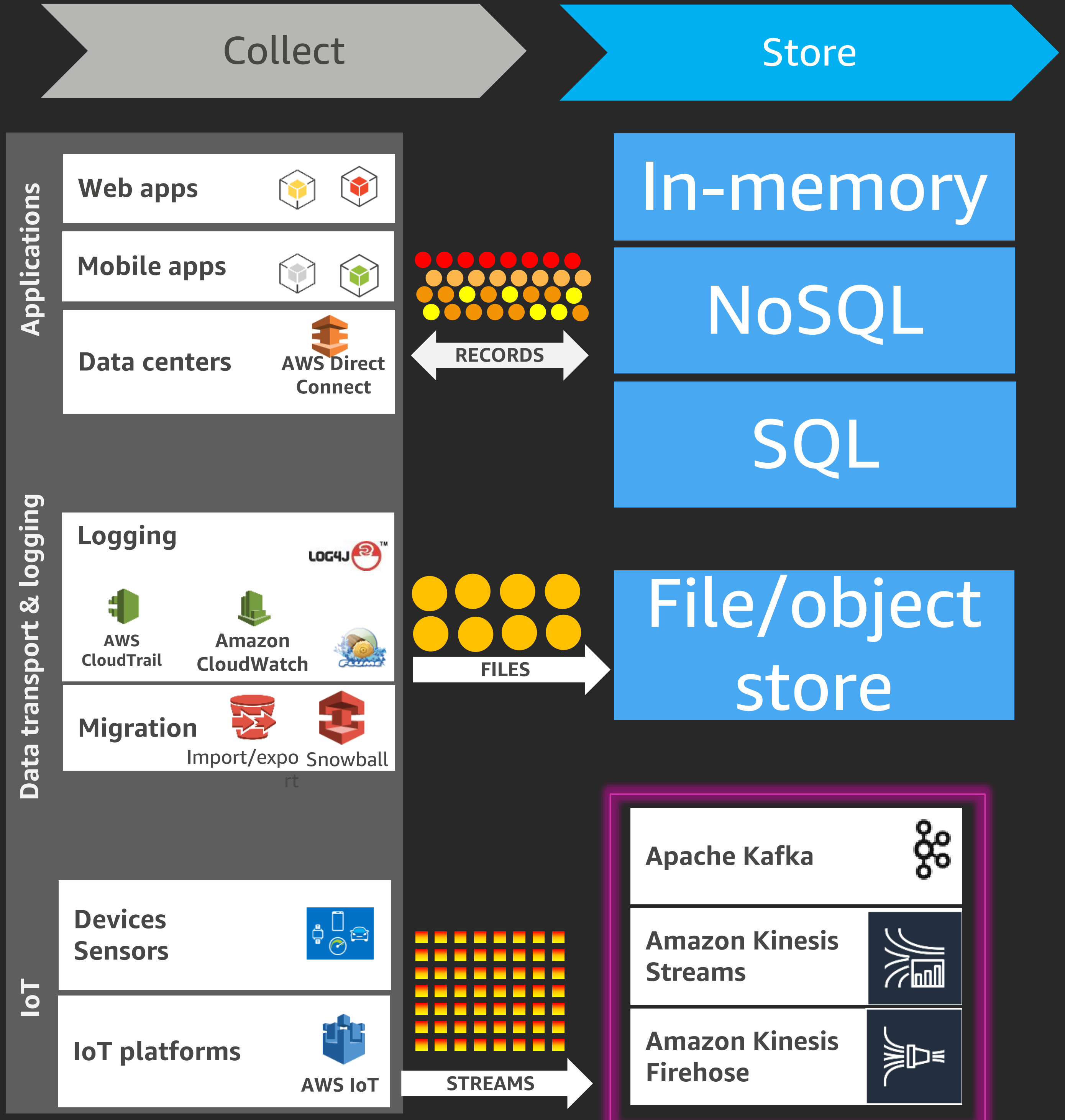
Media files

Log files

Data streams

Store

- In-memory
- NoSQL
- SQL
- File/object store
- Stream storage



Stream storage

Apache Kafka

- High throughput distributed streaming platform

Amazon Kinesis Data Streams

- Managed stream storage

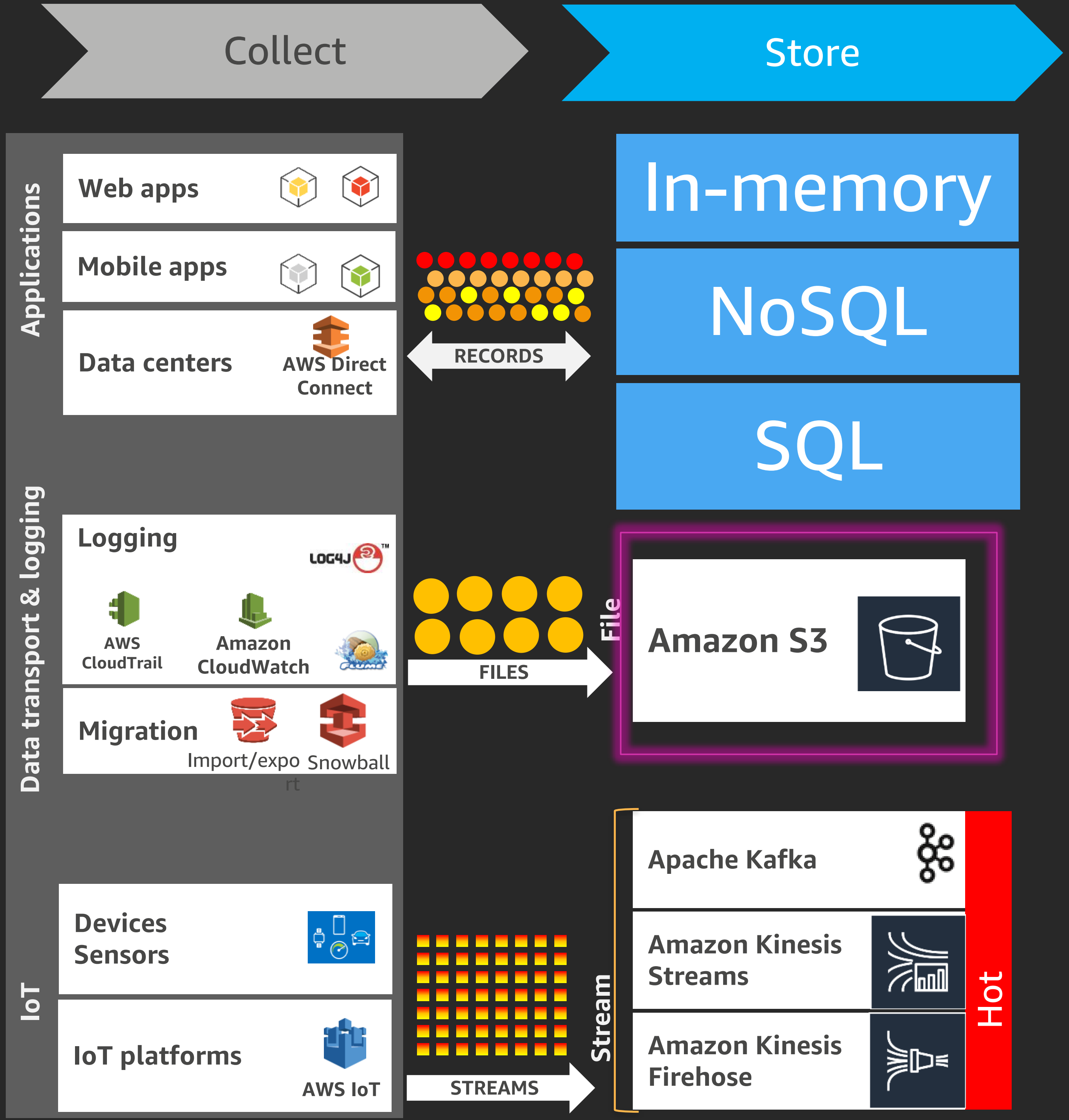
Amazon Kinesis Data Firehose

- Managed data delivery

Which stream/message storage should I use?

	Amazon Kinesis Data Streams	Amazon Kinesis Data Firehose	Apache Kafka (on Amazon Elastic Compute Cloud [Amazon EC2])	Amazon Simple Queue Service (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	Three AZ	Three AZ	Configurable	Three AZ	Three AZ
Scale/throughput	No limit / ~ shards	No limit / automatic	No limit / ~ nodes	No limits / automatic	300 TPS / queue
Multiple consumers	Yes	No	Yes	No	No
Row/object size	1 MB	Destination row/object size	Configurable	256 KB	256 KB
Cost	Low	Low	Low (+admin)	Low-medium	Low-medium

File/object storage



Amazon Simple Storage Service (Amazon S3)

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

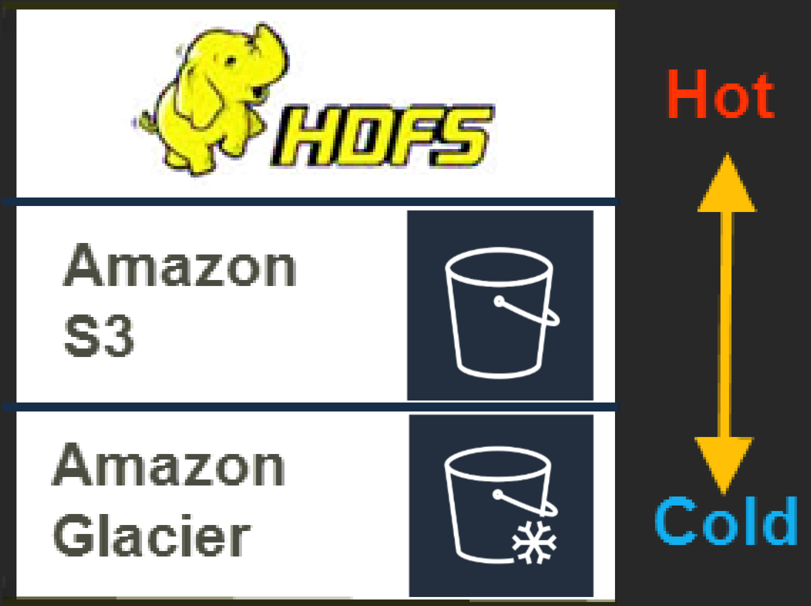
Use Amazon S3 as the storage for your data lake

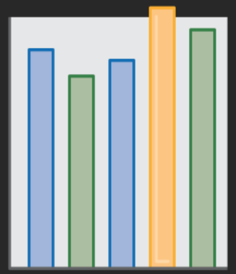
- Natively supported by big data frameworks (Spark, Hive, Presto, and others)
- Decouple storage and compute
  - No need to run compute clusters for storage (unlike HDFS)
  - Can run transient Amazon EMR clusters with Amazon EC2 Spot Instances
  - Multiple & heterogeneous analysis clusters and services 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?

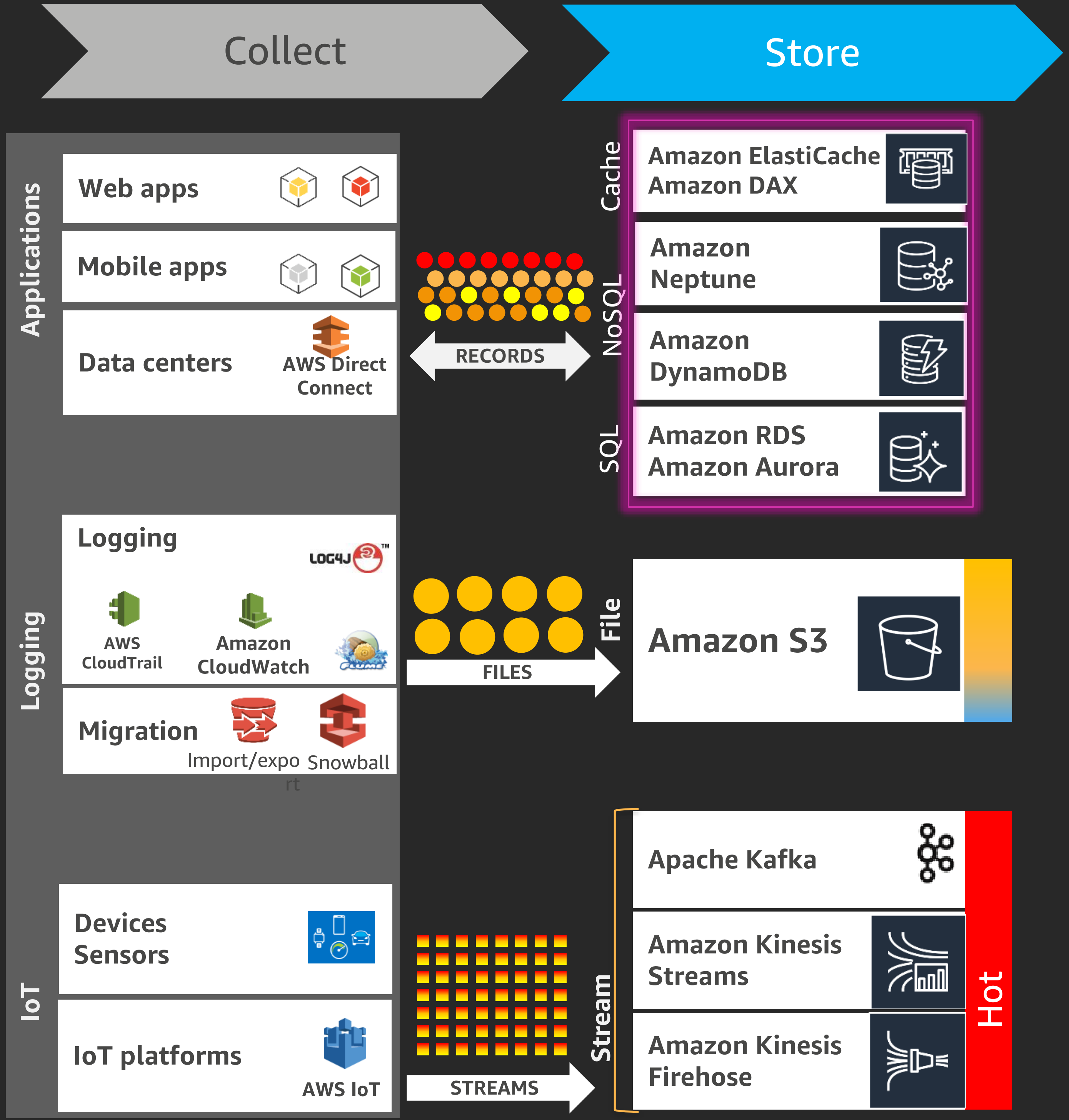
- Use HDFS/local for working data sets
  - (for example, iterative read on the same data sets)
- Use Amazon S3 Standard for frequently accessed data
- Use Amazon S3 Standard–IA for less frequently accessed data
- Use Amazon Glacier for archiving cold data

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 Neptune

- Managed graph DB

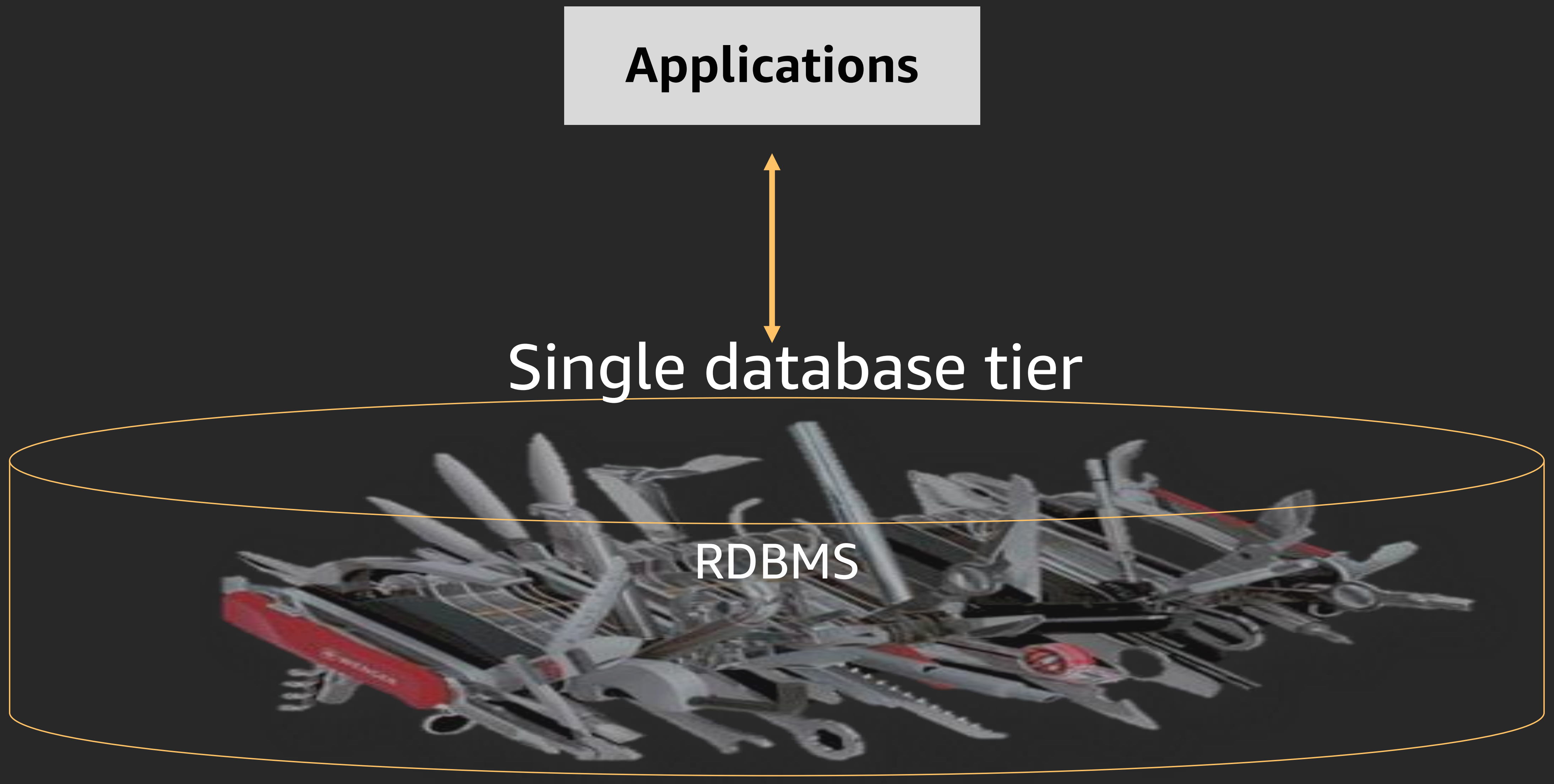
Amazon DynamoDB

- Managed key value/document DB

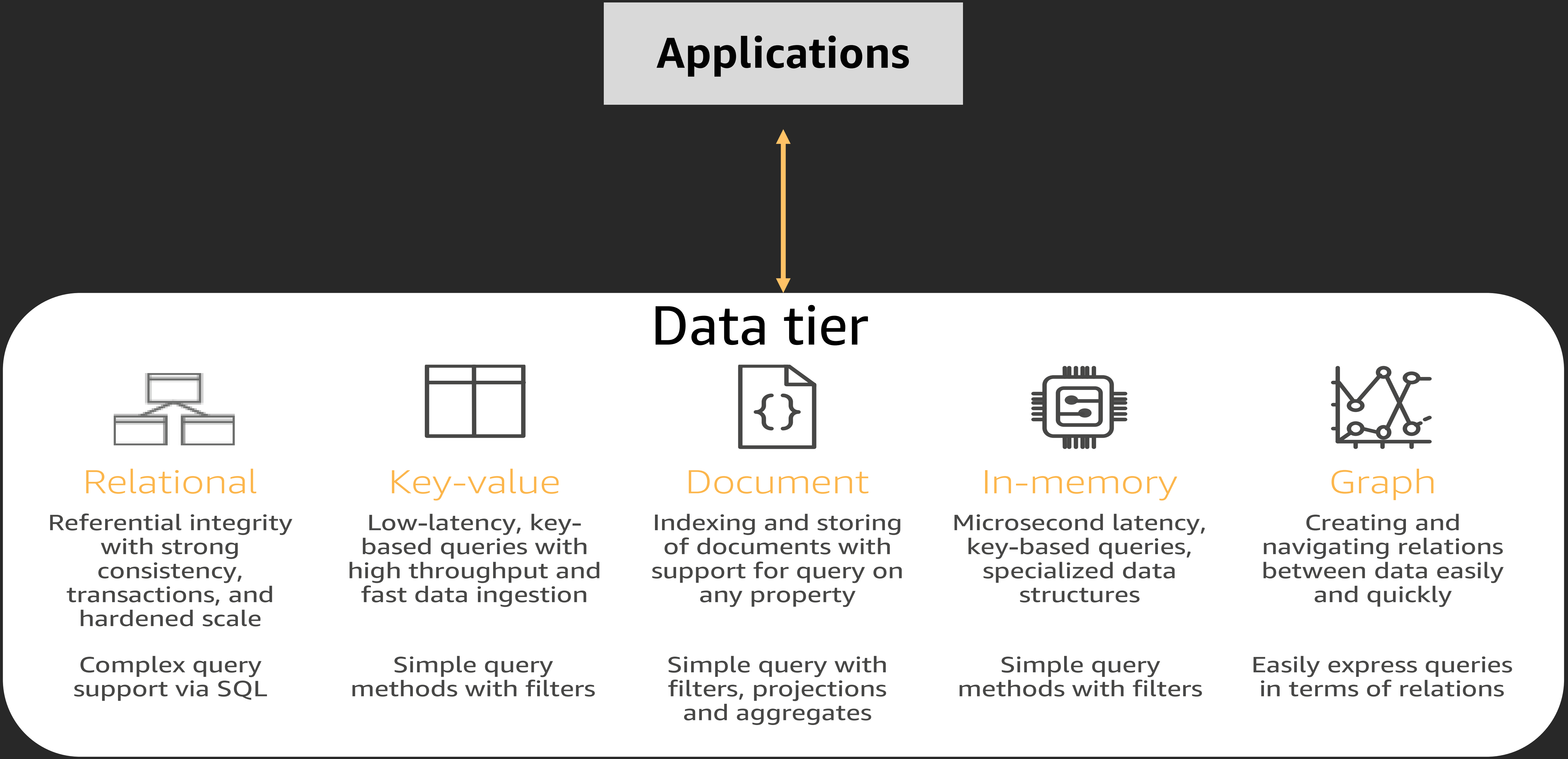
Amazon Relational Database Service (Amazon RDS)

- Managed relational database service

Anti-pattern



Best practice: Use the right tool for the job



Applications

Data tier

Relational	Key-value	Document	In-memory	Graph
Referential integrity with strong consistency, transactions, and hardened scale	Low-latency, key-based queries with high throughput and fast data ingestion	Indexing and storing of documents with support for query on any property	Microsecond latency, key-based queries, specialized data structures	Creating and navigating relations between data easily and quickly
Complex query support via SQL	Simple query methods with filters	Simple query with filters, projections and aggregates	Simple query methods with filters	Easily express queries in terms of relations

Which data store should I use?

Ask yourself some questions

- What is the data structure?
- How will the data be accessed?
- What is the temperature of the data?
- What will the solution cost?

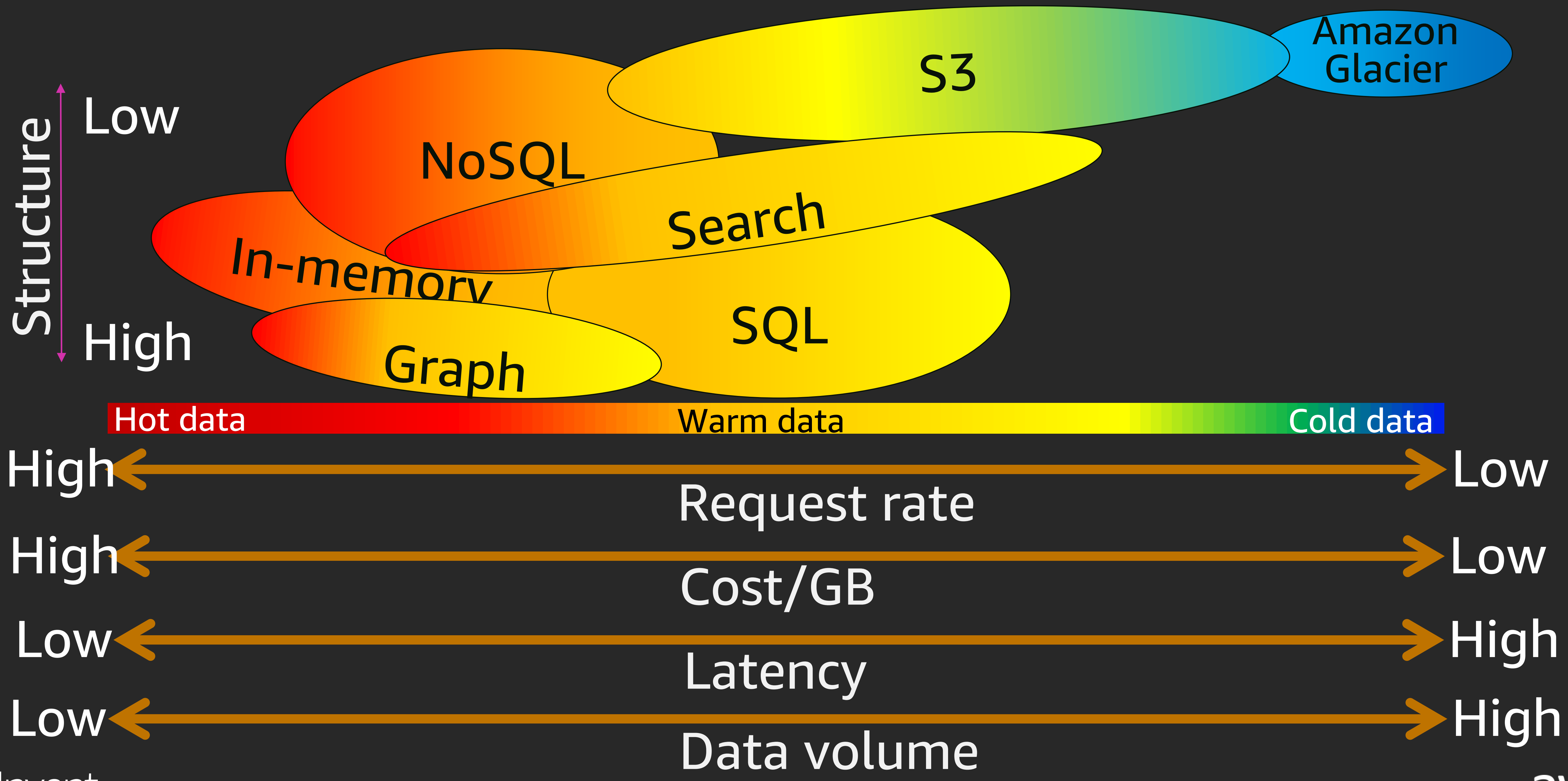
What is the data structure?

Data structure	What to use?
Fixed schema	SQL, NoSQL
Schema-free (JSON)	NoSQL, search
Key/value	In-memory, NoSQL
Graph	GraphDB

How will the data be accessed?

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

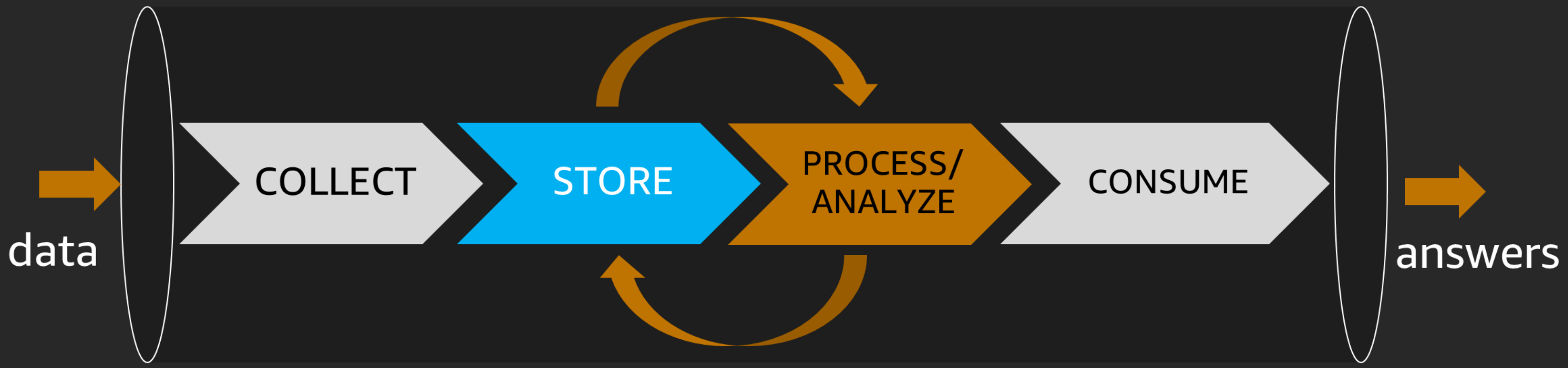
What is the temperature of the data?



Database characteristics

	Amazon ElastiCache	Amazon DynamoDB + DAX	Amazon Aurora	Amazon RDS	Amazon Elasticsearch Service	Amazon Neptune	Amazon S3 + Amazon Glacier
Use cases	In memory caching	K/V lookups, document store	OLTP, transactional	OLTP, transactional	Log analysis, reverse indexing	Graph	File store
Performance	Ultra high request rate, Ultra low latency	Ultra high request rate, ultra low to low latency	Very high request rate, low latency	High request rate, low latency	Medium request rate, low latency	Medium request rate, low latency	High throughput
Shape	K/V	K/V and document	Relational	Relational	Documents	Node/edges	Files
Size	GB	TB, PB (no limits)	GB, mid TB	GB, low TB	GB, TB	GB, mid TB	GB, TB, PB, EB (no limits)
Cost / GB	$$	¢¢ - $$	¢¢	¢¢	¢¢	¢¢	¢- ¢4/10
Availability	2 AZ	3 AZ	3 AZ	2 AZ	1-2 AZ	3 AZ	3 AZ
VPC support	Inside VPC	VPC endpoint	Inside VPC	Inside VPC	Outside or inside VPC	Inside VPC	VPC endpoint

Hot data — Warm data — Cold data



Process/ analyze

Interactive & batch analytics

Process/analyze

Amazon Elasticsearch Service

Managed service for ElasticSearch

Amazon Redshift & Amazon Redshift Spectrum

Managed data warehouse

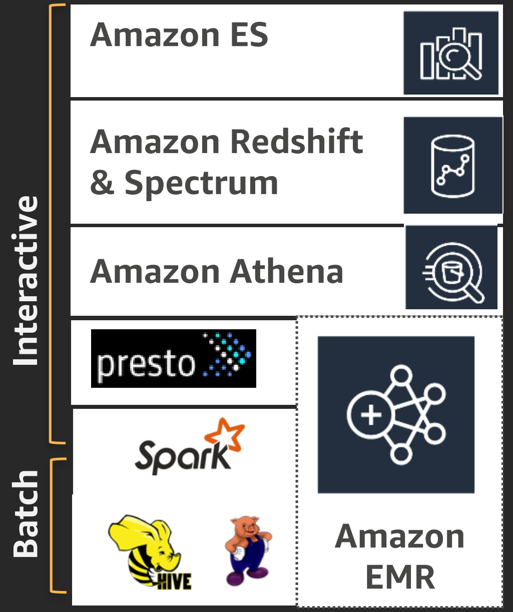
Spectrum enables querying S3

Amazon Athena

Serverless interactive query service

Amazon EMR

Managed Hadoop framework for running Apache Spark, Flink, Presto, Tez, Hive, Pig, HBase, and others



Stream/real-time analytics

Process/analyze

Spark Streaming on Amazon EMR

Amazon Kinesis Data Analytics

- Managed service for running SQL on streaming data

Amazon KCL

- Amazon Kinesis Client Library

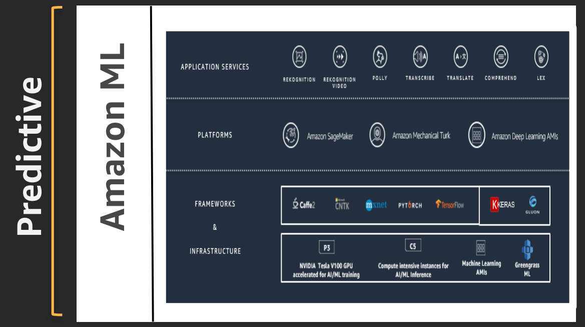
AWS Lambda

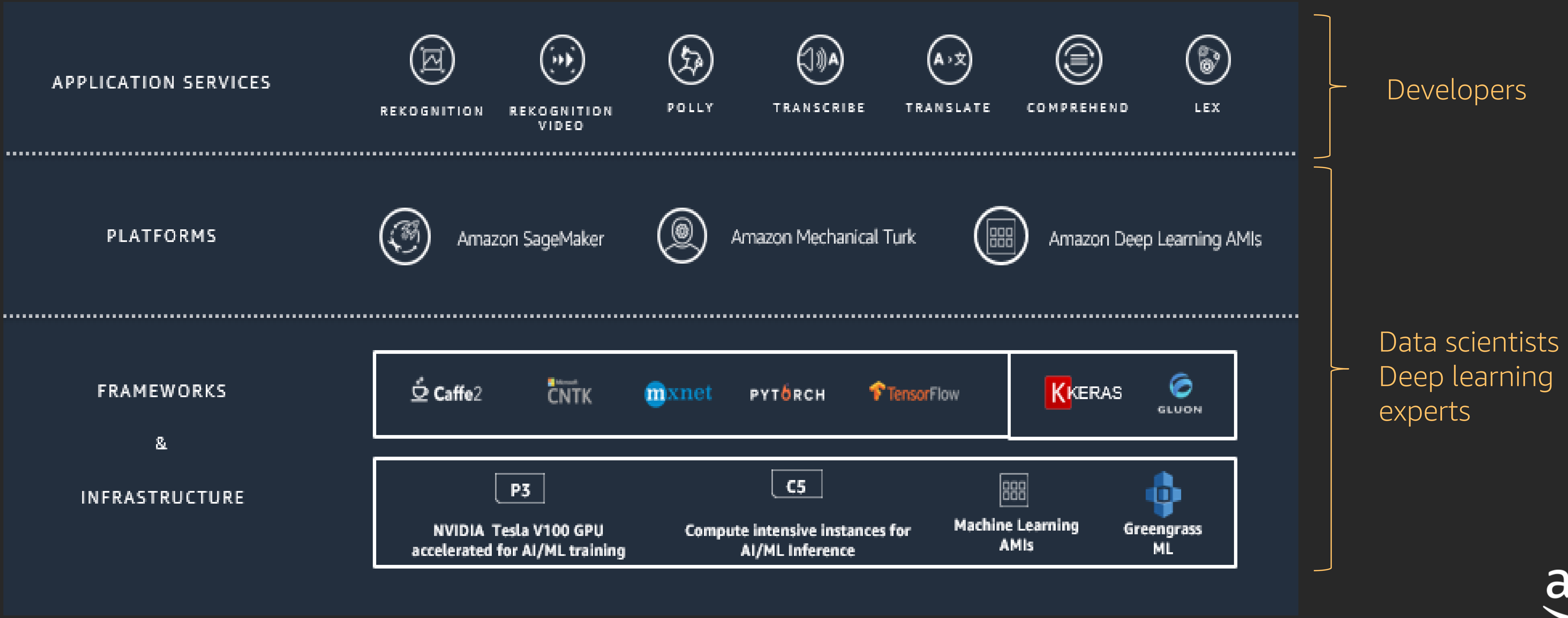
- Run code serverless (without provisioning or managing servers)
- Services such as S3 can publish events to AWS Lambda
- AWS Lambda can pool event from a Kinesis

Stream
Spark Streaming
Amazon Kinesis Analytics
KCL Apps
AWS Lambda

Predictive analytics

Process/analyze





APPLICATION SERVICES

REKOGNITION · REKOGNITION VIDEO · POLLY · TRANSCRIBE · TRANSLATE · COMPREHEND · LEX

PLATFORMS

Amazon SageMaker · Amazon Mechanical Turk · Amazon Deep Learning AMIs

FRAMEWORKS & INFRASTRUCTURE

Caffe2 · CNTK · mxnet · PYTORCH · TensorFlow · KERAS · GLUON

P3 NVIDIA Tesla V100 GPU accelerated for AI/ML training

C5 Compute intensive instances for AI/ML Inference

Machine Learning AMIs

Greengrass ML

Developers

Data scientists
Deep learning experts

aws

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, one-minute metrics

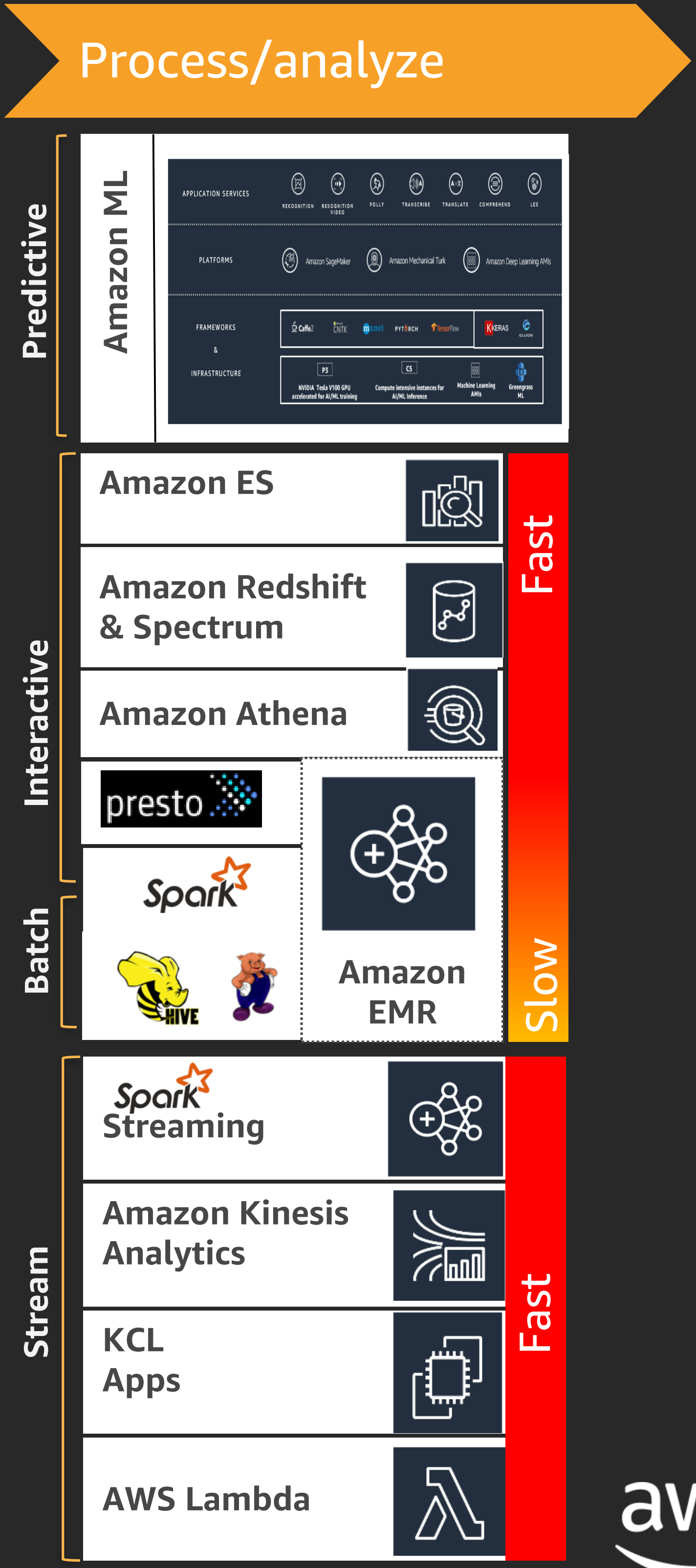
Amazon EMR (Spark Streaming), Amazon Kinesis Data Analytics, Amazon KCL, AWS Lambda, and others

Predictive

Takes milliseconds (real-time) to minutes (batch)

Example: Fraud detection, forecasting demand, speech recognition

Amazon SageMaker, Amazon Polly, Amazon Rekognition, Amazon Transcribe, Amazon Translate, Amazon EMR (Spark ML), Amazon Deep Learning AMI (MXNet, TensorFlow, Theano, Torch, CNTK, and Caffe)



AWS re:Invent

aws

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 through MultiLangDaemon	ANSI SQL with extensions	Node.js, Java, Python, .Net Core
Sliding window functions	Build-in	App needs to implement	Built-in	No
Reliability	KCL and Spark checkpoints	Managed by Amazon KCL	Managed by Amazon Kinesis Data Analytics	Managed by AWS Lambda

	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	AWS Glue catalog	AWS Glue catalog	AWS 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	Yes (Scalar)	Yes (Scalar)	No	Yes		

Fastest Fast Slow

AWS re:Invent

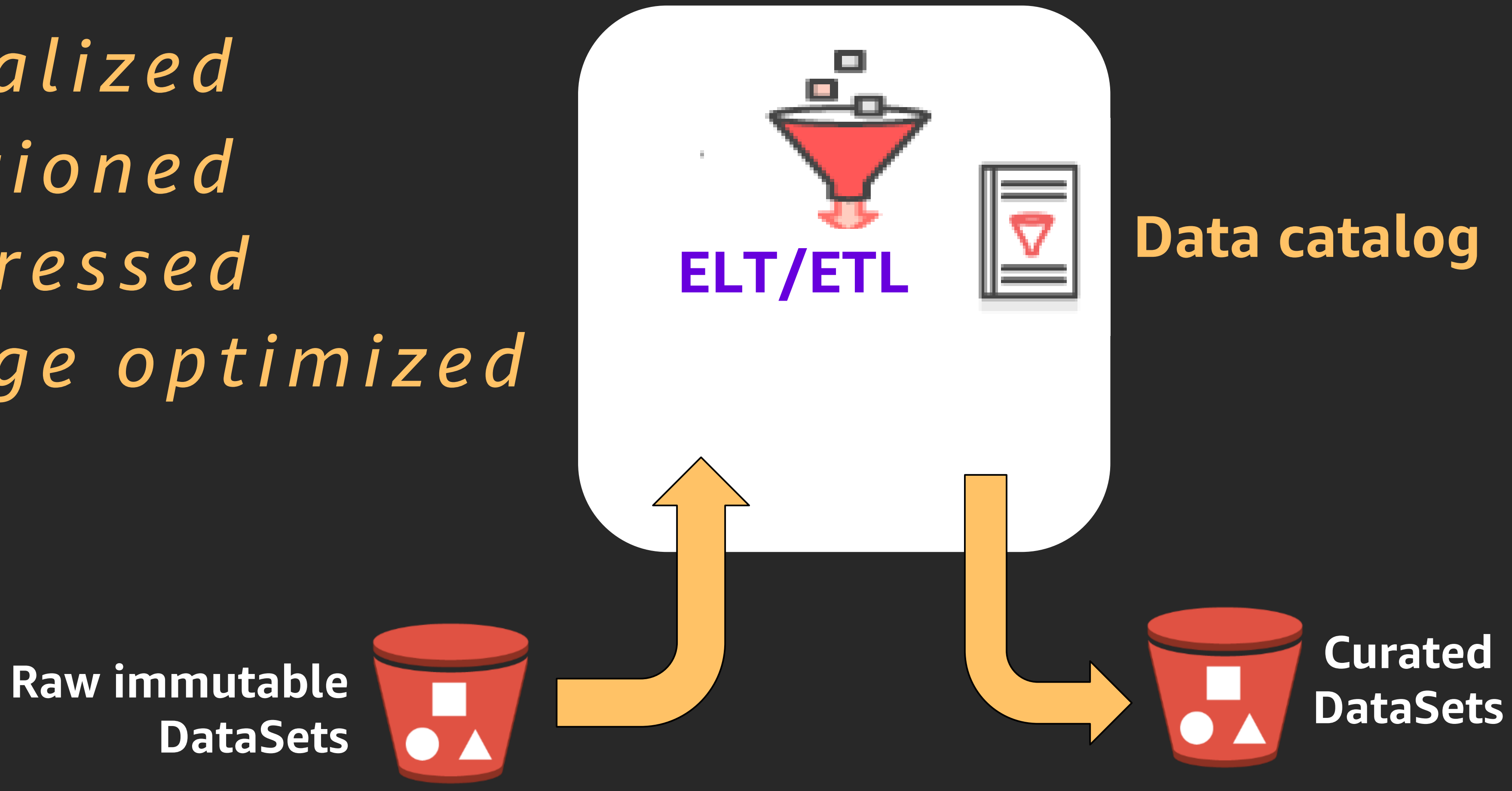
aws

ELT/ETL: Preparing raw data for consumption

Raw data stored in data lake

Preparation

- *Normalized*
- *Partitioned*
- *Compressed*
- *Storage optimized*

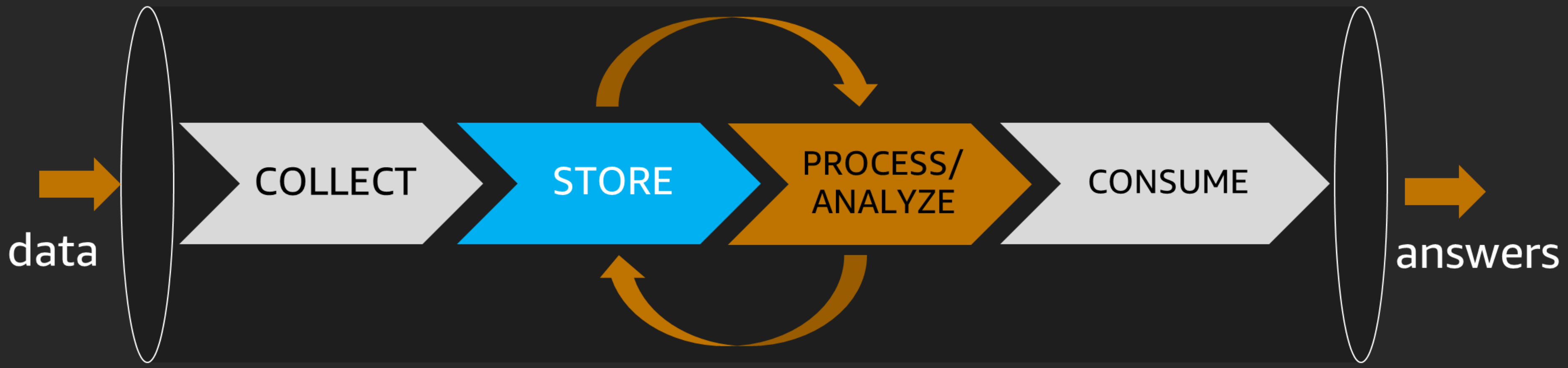


Data lake on AWS

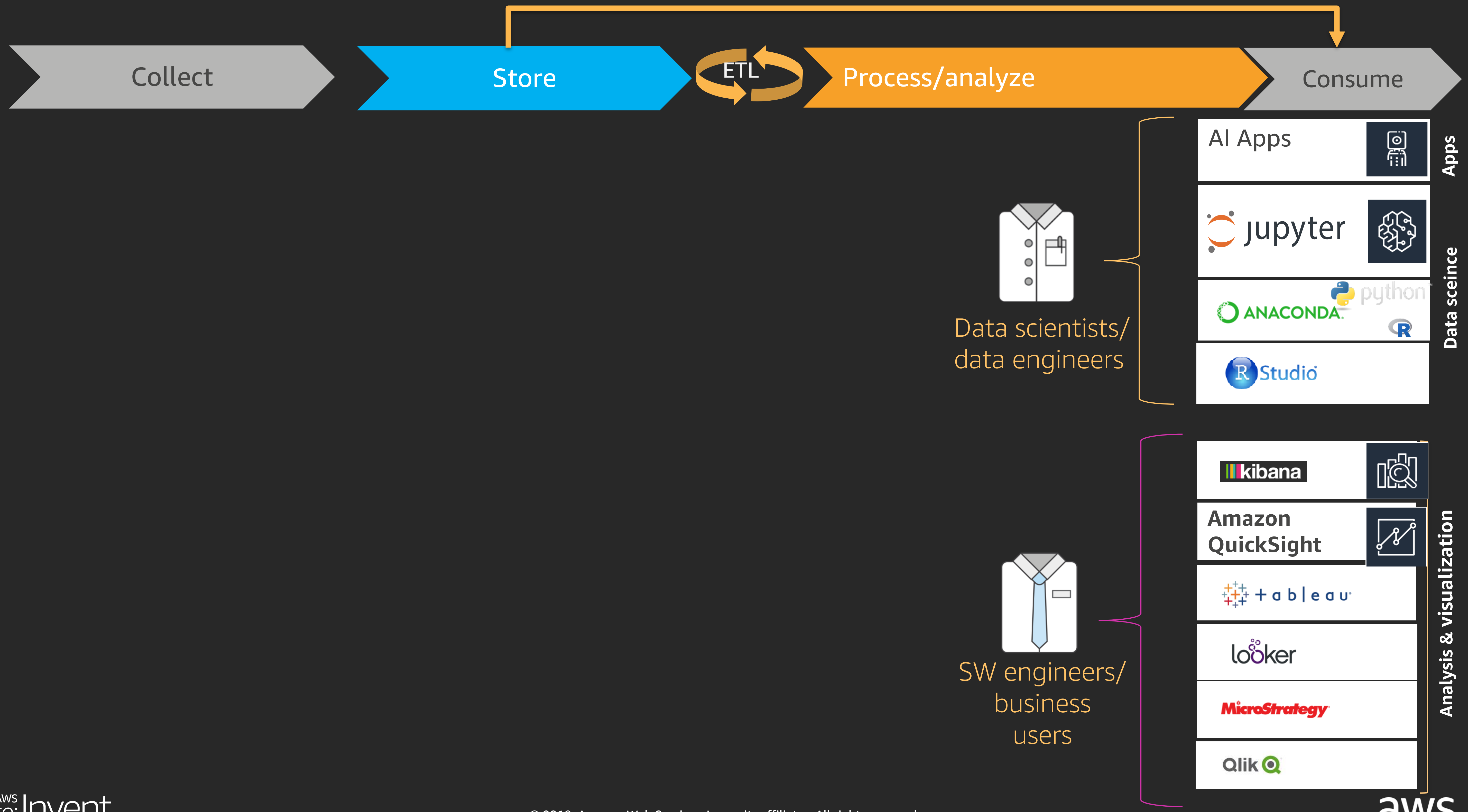
Demonstration—Why ELT?

Which ELT/ETL tool should I use?

	AWS Glue ETL	AWS Data Pipeline	Amazon Data Migration Service (Amazon DMS)	Amazon EMR	Apache NiFi	Partner solution
Use case	Serverless ETL	Data workflow service	Migrate databases (and to/from data lake)	Customize developed Hadoop/Spark ETL	Automate the flow of data between systems	Rich partner ecosystem for ETL
Scale/throughput	~DPUs	~Nodes through EMR cluster	EC2 instance type	~Nodes	Self managed	Self manager or through partner
Managed service	Clusterless	Managed	Managed EC2 on your behalf	Managed EC2 on your behalf	Self managed on Amazon EMR or Marketplace	Self manager or through partner
Data sources	S3, RDBMS, Amazon RedShift, DynamoDB	S3, JDBC, DynamoDB, custom	RDBMS, data warehouses, Amazon S3* (*limited)	Various Hadoop/Spark managed	Various through rich processor framework	Various
Skills needed	Wizard for simple mapping, code snippets for advanced ETL	Wizard and code snippets	Wizard and drag/drop	Hadoop/Spark coding	NiFi processors and some coding	Self manager or through partner



Consume



Collect

Store

ETL

Process/analyze

Consume

Data scientists/ data engineers

SW engineers/ business users

Apps

AI Apps

Data sceince

jupyter

ANACONDA

python

R

RStudio

Analysis & visualization

kibana

Amazon QuickSight

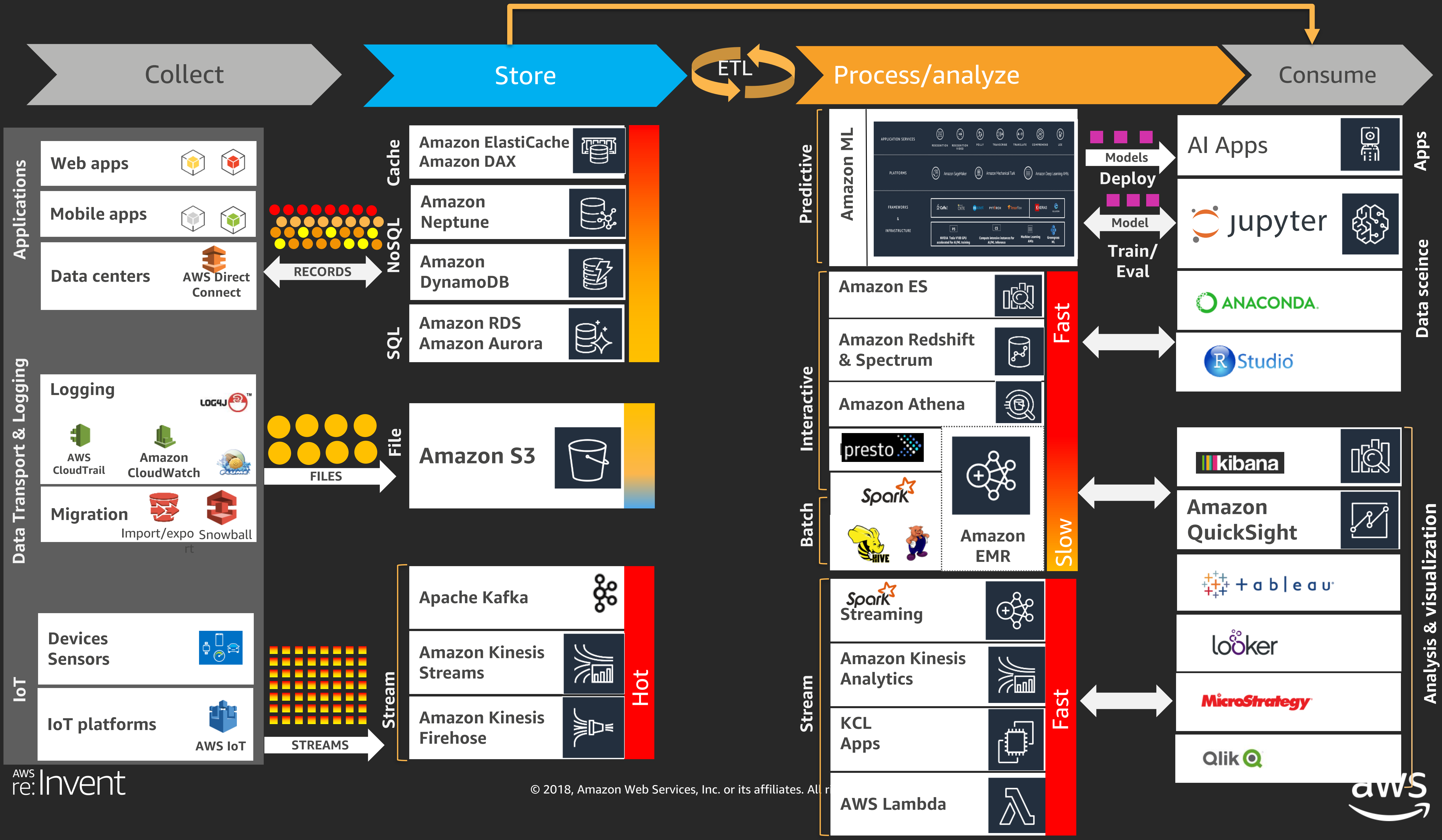
tableau

looker

MicroStrategy

Qlik

Putting it all together



Collect

Store

ETL

Process/analyze

Consume

Applications

Web apps

Mobile apps

Data centers

AWS Direct Connect

Data Transport & Logging

Logging

LOG4J

AWS CloudTrail

Amazon CloudWatch

Migration

Import/export

Snowball

IoT

Devices Sensors

IoT platforms

AWS IoT

RECORDS

FILES

STREAMS

Cache

Amazon ElastiCache
Amazon DAX

NoSQL

Amazon Neptune

Amazon DynamoDB

SQL

Amazon RDS
Amazon Aurora

File

Amazon S3

Stream

Apache Kafka

Amazon Kinesis Streams

Amazon Kinesis Firehose

Hot

Predictive

Amazon ML

APPLICATION SERVICES: REKOGNITION, REKOGNITION VIDEO, POLLY, TRANSCRIBE, TRANSLATE, COMPREHEND, LEX

PLATFORMS: Amazon SageMaker, Amazon Mechanical Turk, Amazon Deep Learning AMIs

FRAMEWORKS & INFRASTRUCTURE: Caffe2, CNTK, mxnet, PYTORCH, TensorFlow, KERAS, GLUON

P3 NVIDIA Tesla V100 GPU accelerated for AI/ML training

C5 Compute intensive instances for AI/ML Inference

Machine Learning AMIs

Greengrass ML

Models

Deploy

Model

Train/ Eval

Interactive

Amazon ES

Amazon Redshift & Spectrum

Amazon Athena

presto

Batch

Spark

HIVE

Amazon EMR

Fast

Slow

Stream

Spark Streaming

Amazon Kinesis Analytics

KCL Apps

AWS Lambda

Fast

Apps

AI Apps

Data sceince

jupyter

ANACONDA.

RStudio

Analysis & visualization

kibana

Amazon QuickSight

+ a b l e a u

looker

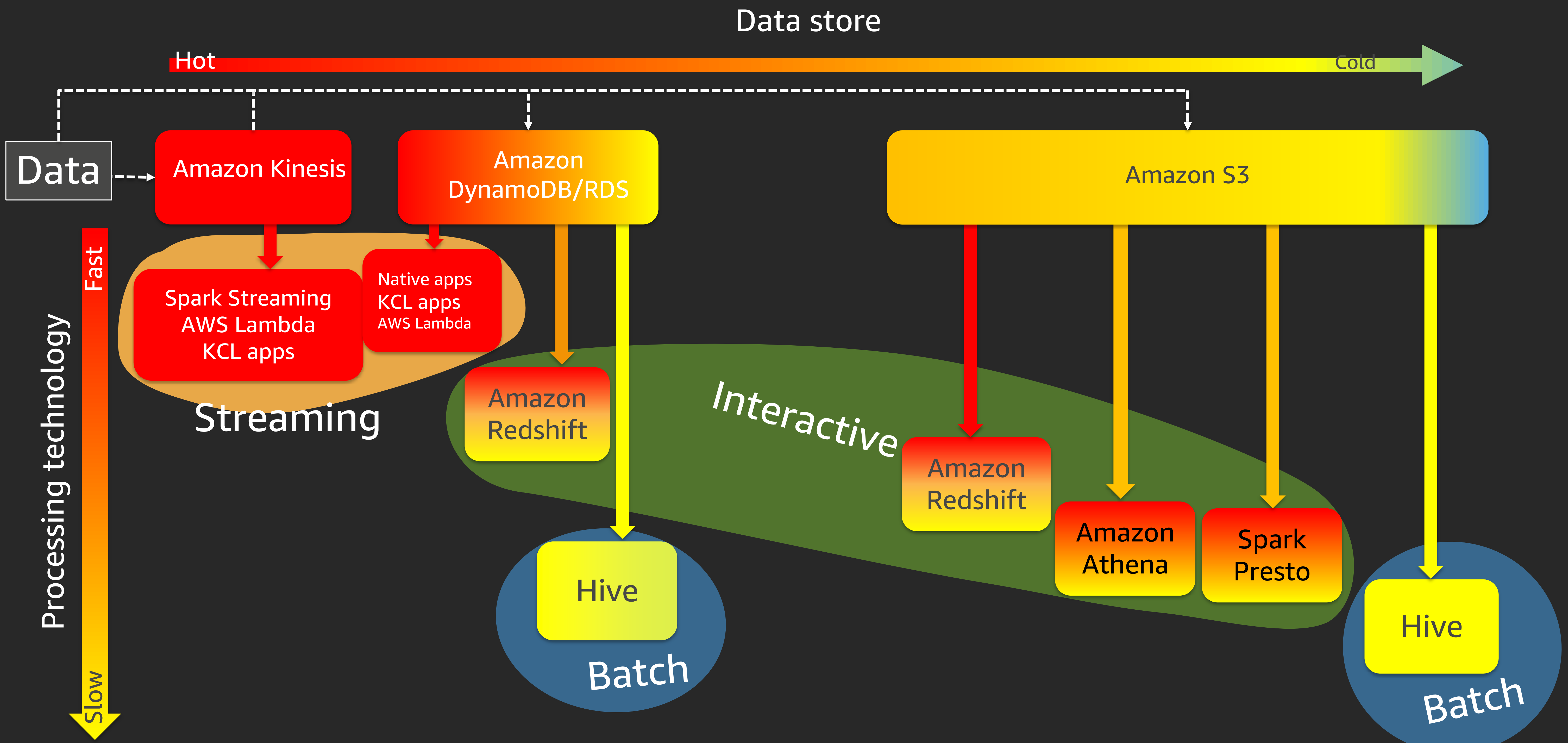
MicroStrategy

Qlik

AWS re:Invent

aws

Design patterns



Data store

Hot

Cold

Data

Amazon Kinesis

Amazon DynamoDB/RDS

Amazon S3

Processing technology

Fast

Slow

Spark Streaming
AWS Lambda
KCL apps

Native apps
KCL apps
AWS Lambda

Streaming

Amazon Redshift

Interactive

Amazon Redshift

Amazon Athena

Spark Presto

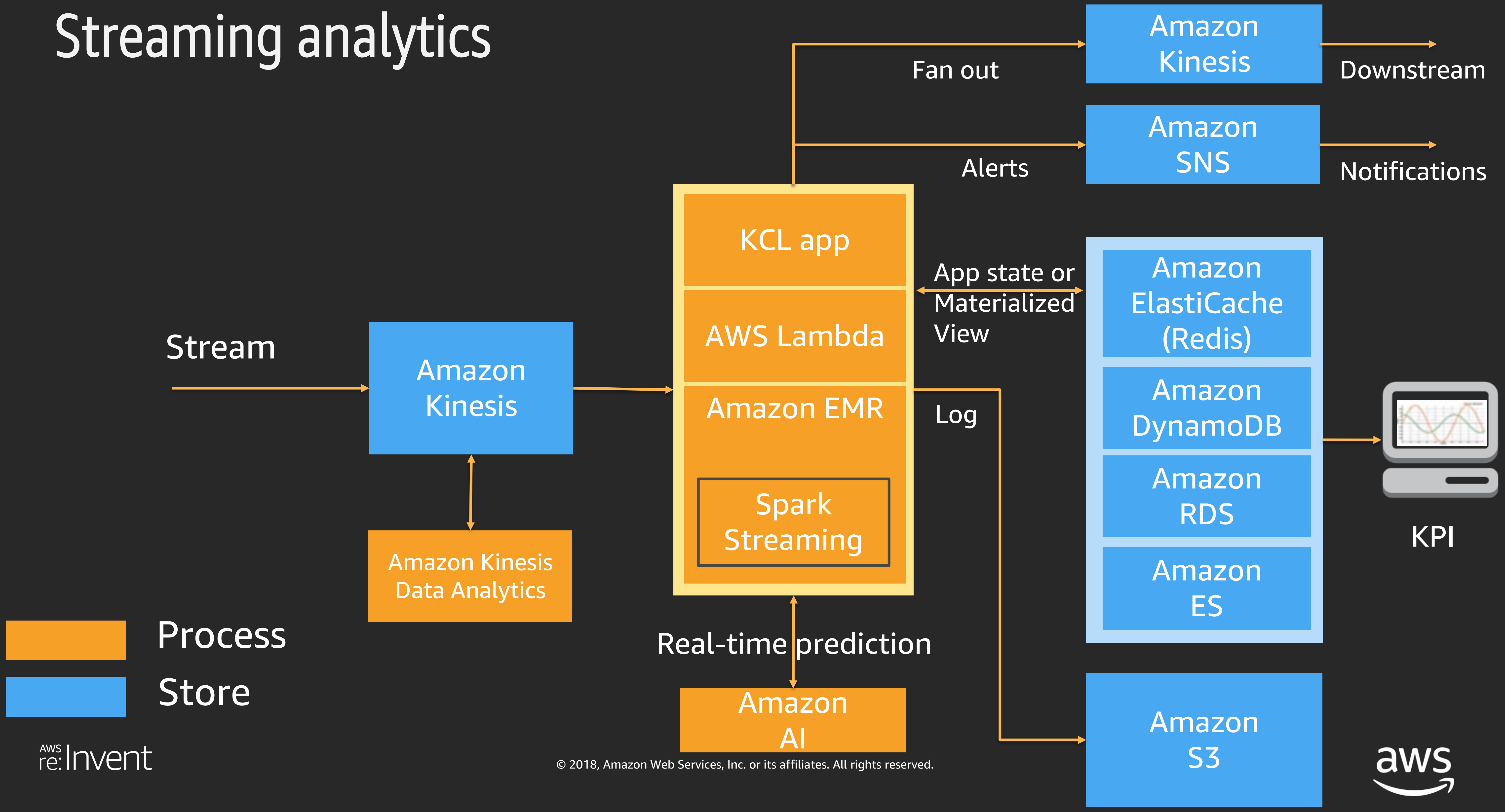
Hive

Batch

Hive

Batch

Streaming analytics



Hearst's serverless data pipeline

HEARST

Media and information company

Television

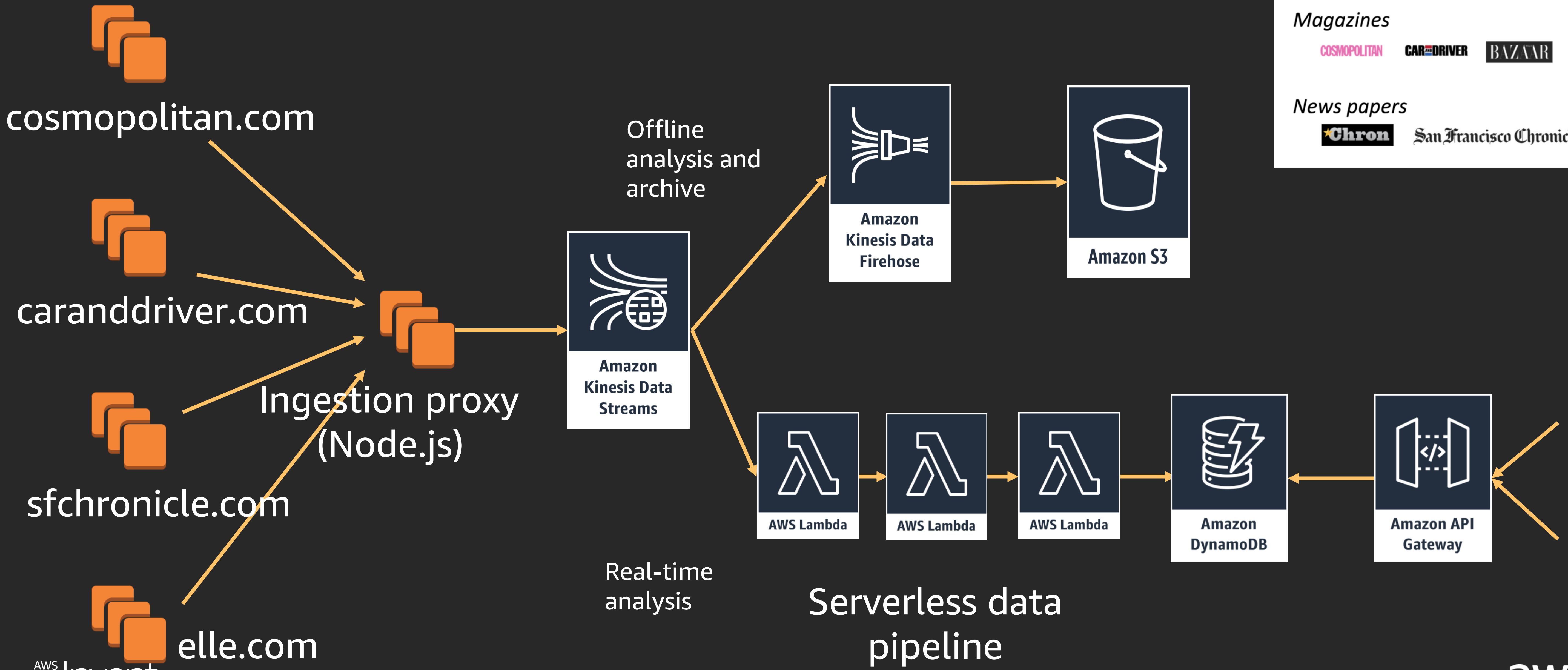
ESPN A&E THE HISTORY CHANNEL LIFETIME

Magazines

COSMOPOLITAN CAR AND DRIVER BAZAAR Bicycling

News papers

Chron San Francisco Chronicle



cosmopolitan.com

caranddriver.com

sfchronicle.com

elle.com

Ingestion proxy (Node.js)

Amazon Kinesis Data Streams

Offline analysis and archive

Amazon Kinesis Data Firehose

Amazon S3

Real-time analysis

AWS Lambda

AWS Lambda

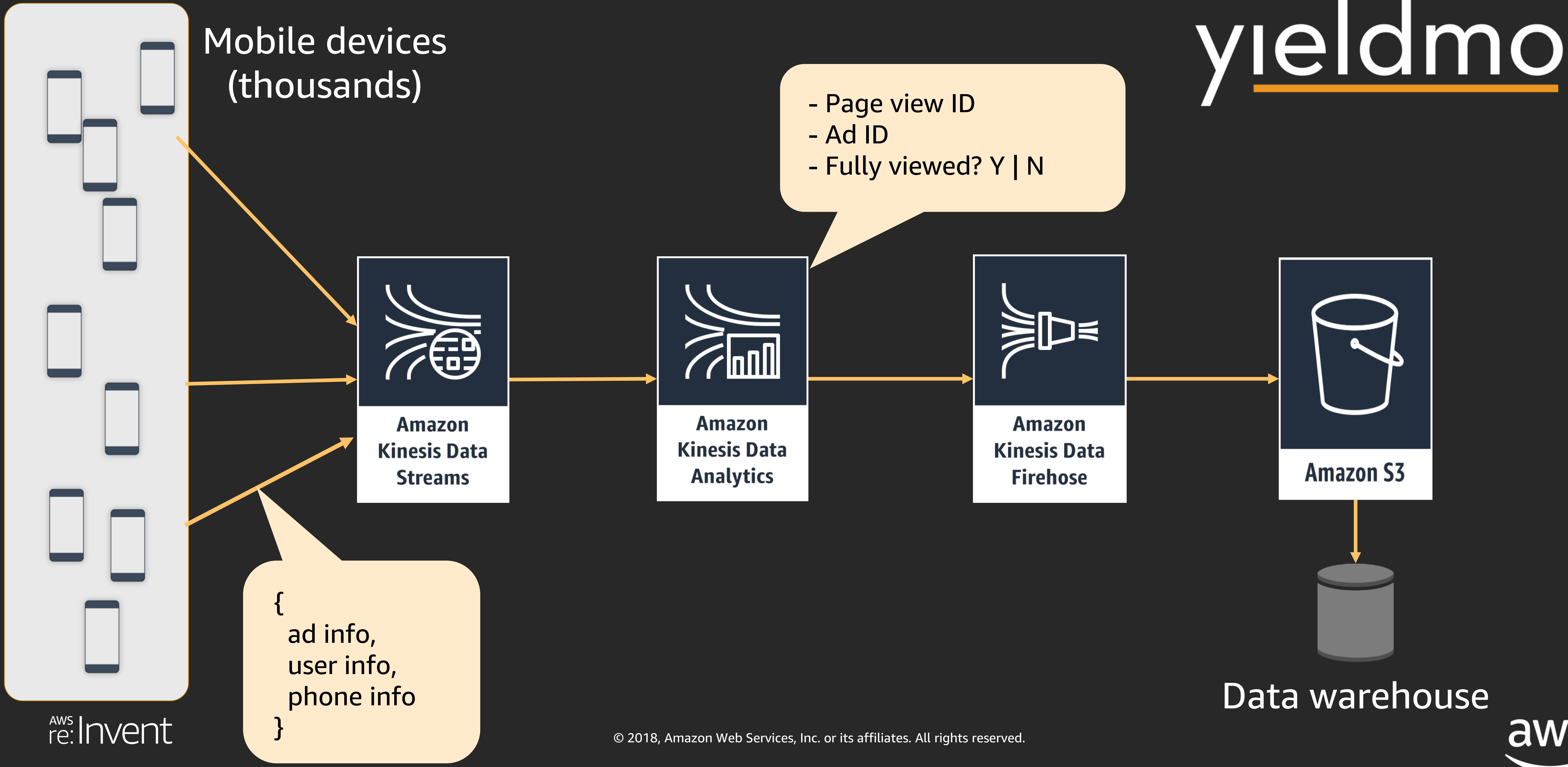
AWS Lambda

Amazon DynamoDB

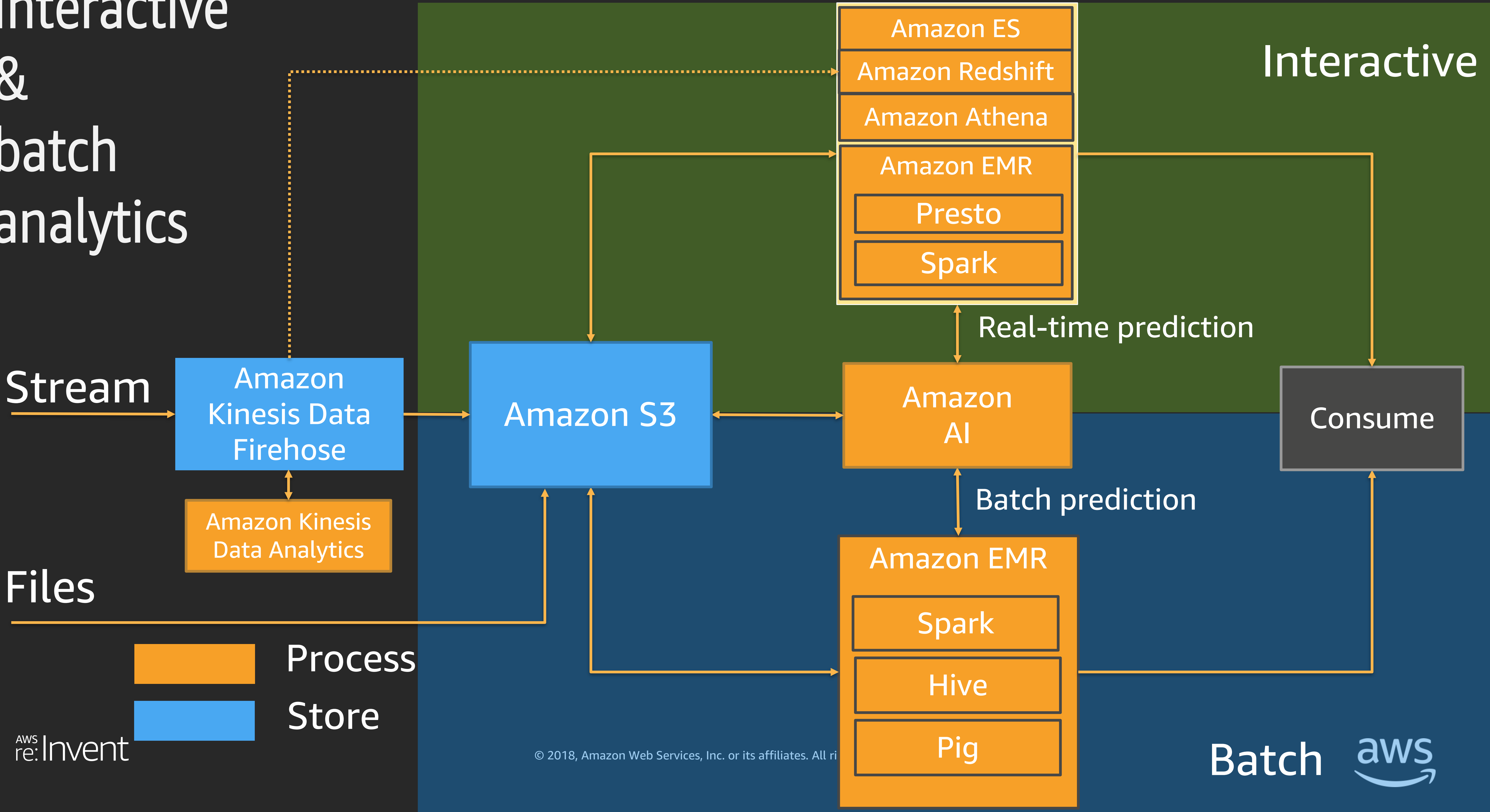
Amazon API Gateway

Serverless data pipeline

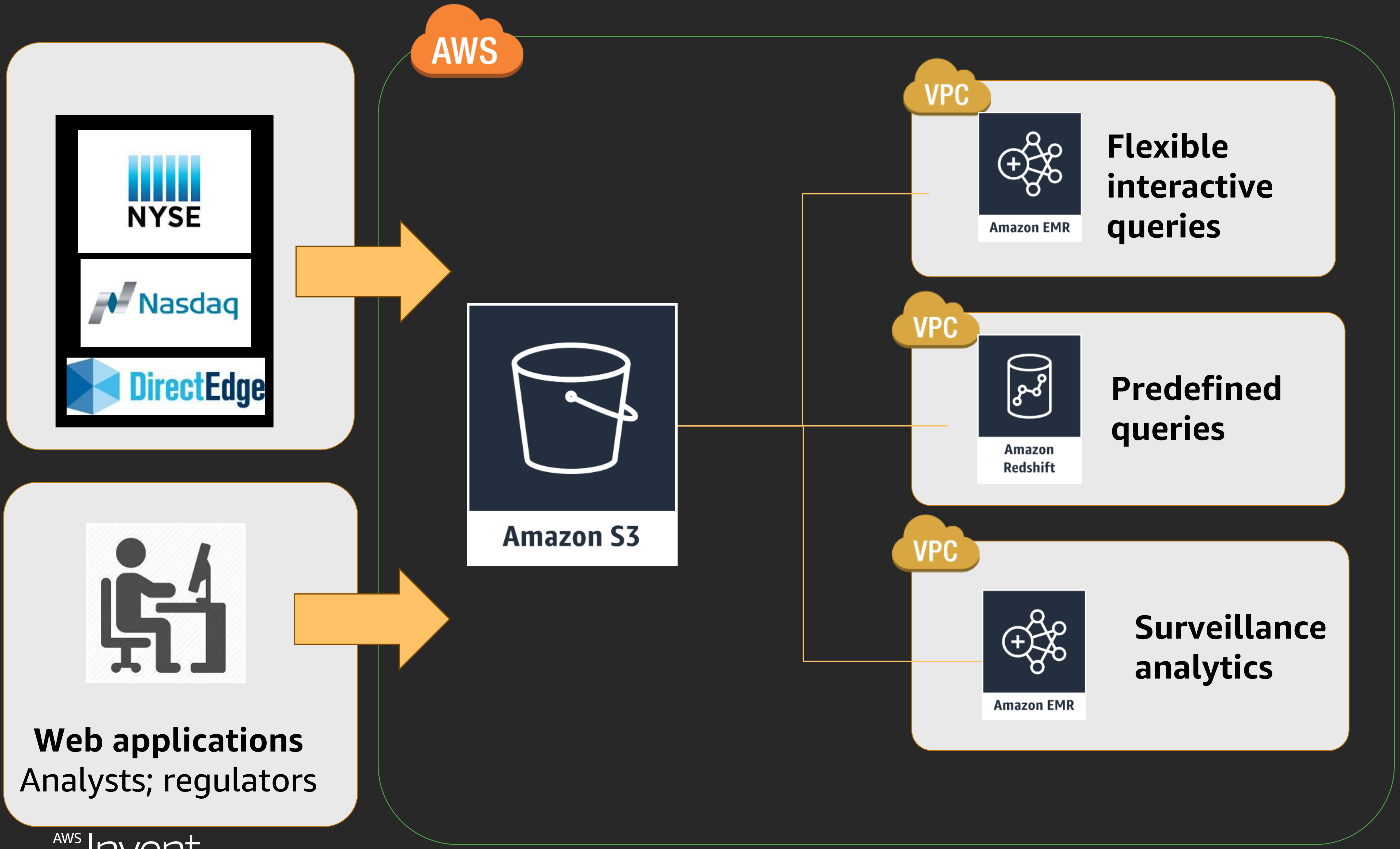
Yieldmo's data ingestion and real-time analysis architecture


yieldmo
Mobile devices (thousands)
{
ad info,
user info,
phone info
}
Amazon Kinesis Data Streams
Amazon Kinesis Data Analytics
- Page view ID
- Ad ID
- Fully viewed? Y | N
Amazon Kinesis Data Firehose
Amazon S3
Data warehouse

Interactive & batch analytics



FINRA: Migrating to AWS

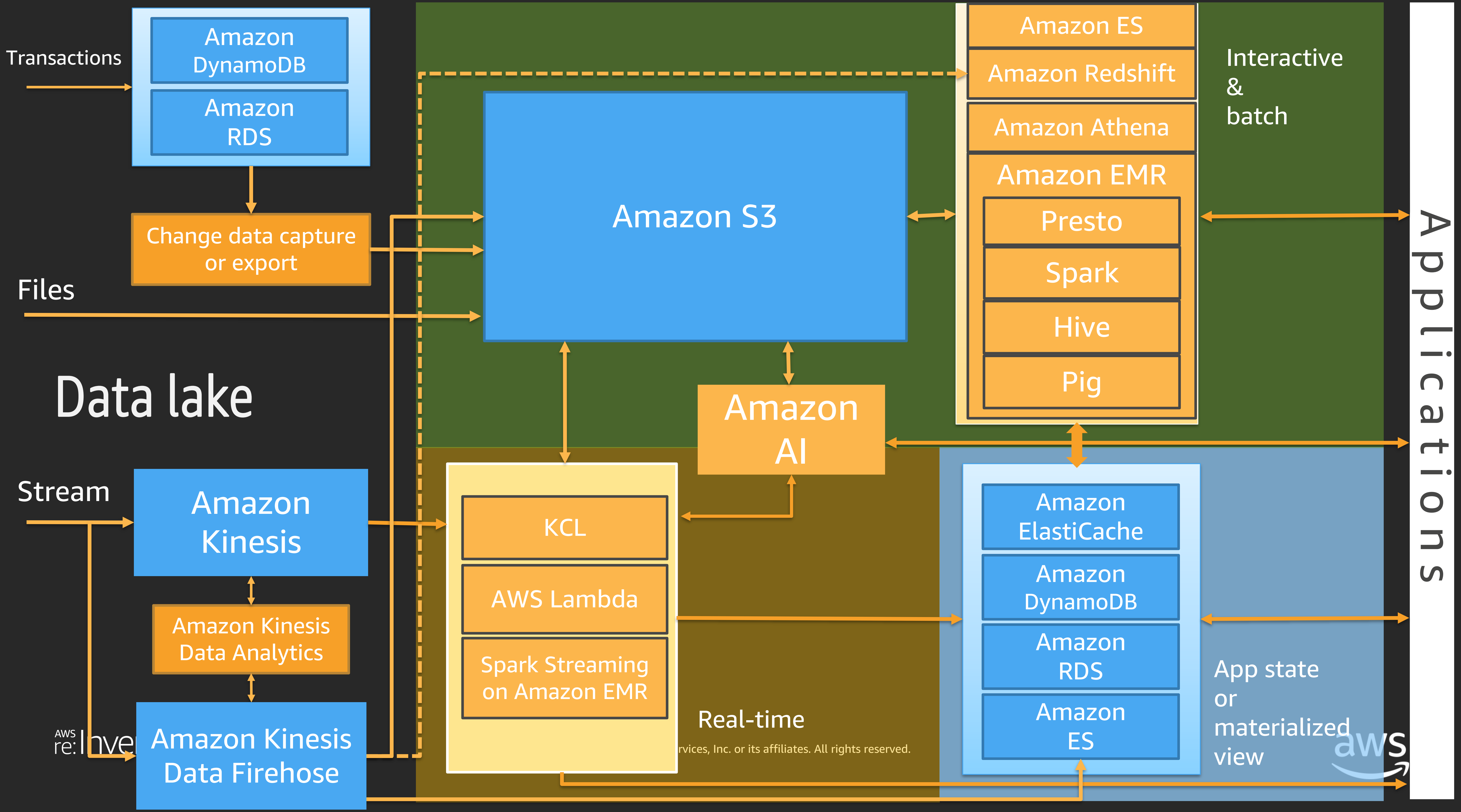


Petabytes of data generated on-premises, brought to AWS, and stored in S3

Thousands of analytical queries performed on Amazon EMR and Amazon Redshift

Stringent security requirements met by leveraging VPC, VPN, encryption at-rest and in-transit, AWS CloudTrail, and database auditing

Data lake



Transactions

Amazon DynamoDB

Amazon RDS

Change data capture or export

Files

Stream

Amazon Kinesis

Amazon Kinesis Data Analytics

Amazon Kinesis Data Firehose

Amazon S3

Amazon AI

Amazon ES

Amazon Redshift

Amazon Athena

Amazon EMR

Presto

Spark

Hive

Pig

Interactive & batch

KCL

AWS Lambda

Spark Streaming on Amazon EMR

Real-time

Amazon ElastiCache

Amazon DynamoDB

Amazon RDS

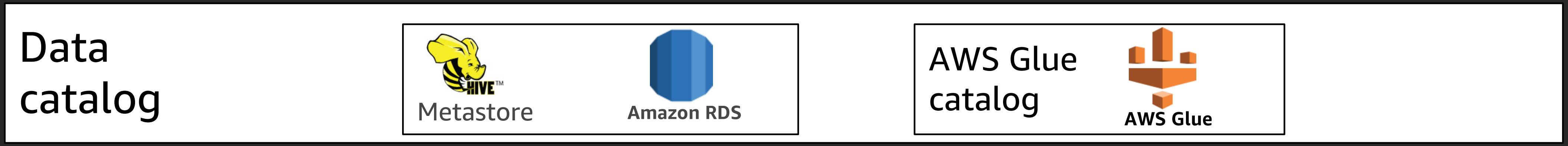
Amazon ES

App state or materialized view

Applications

What about metadata?

- AWS 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



Demonstration—Data Lake Demonstration

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

Thank you!

Ben Snively

AWS re:Invent

aws

Please complete the session survey in the mobile app.