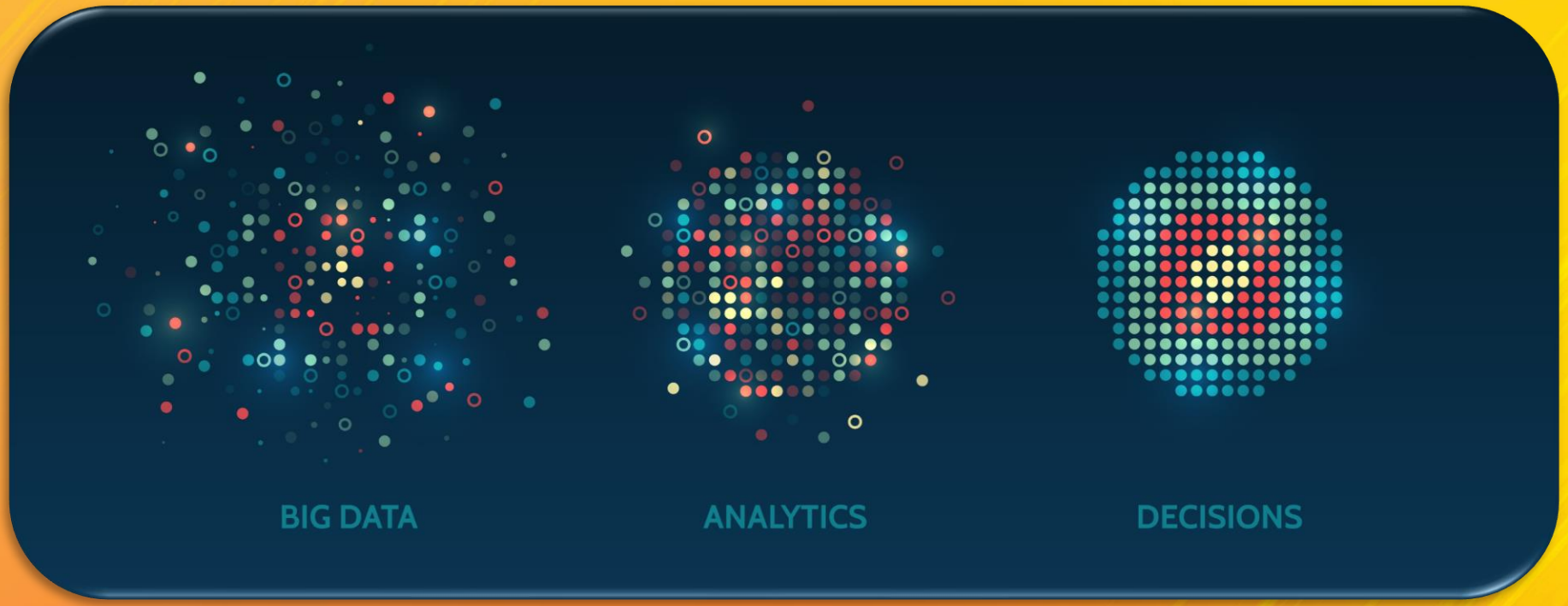
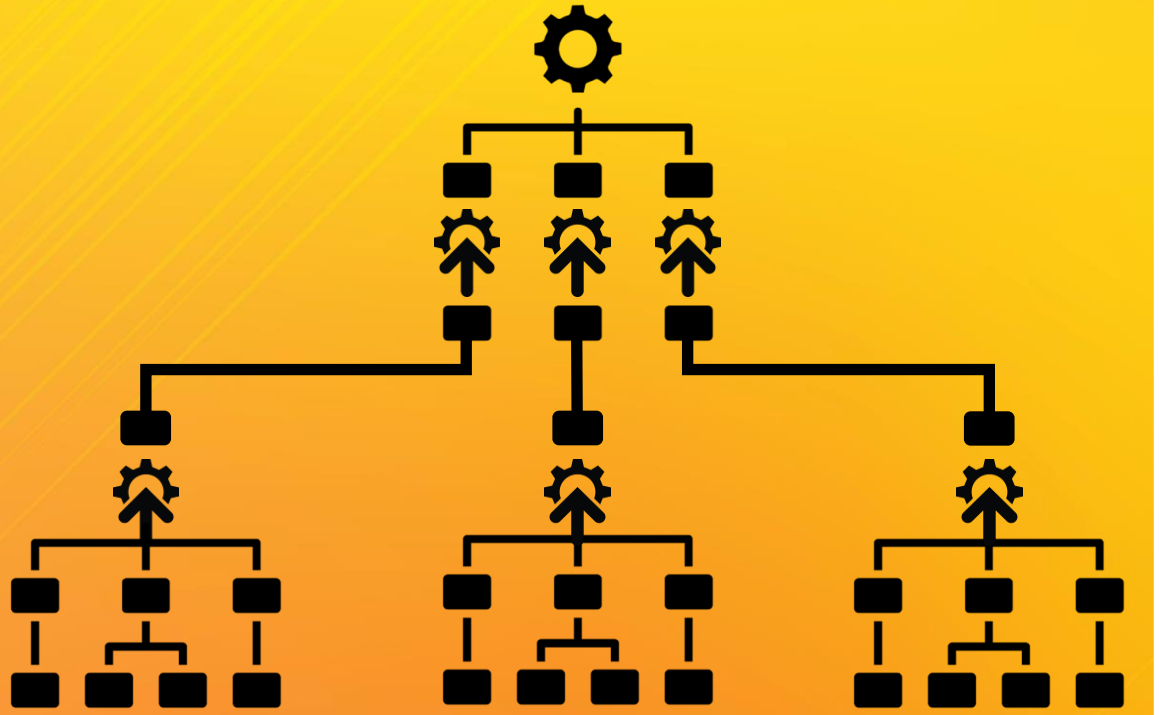
Working with Big Data in AWS



Big Data

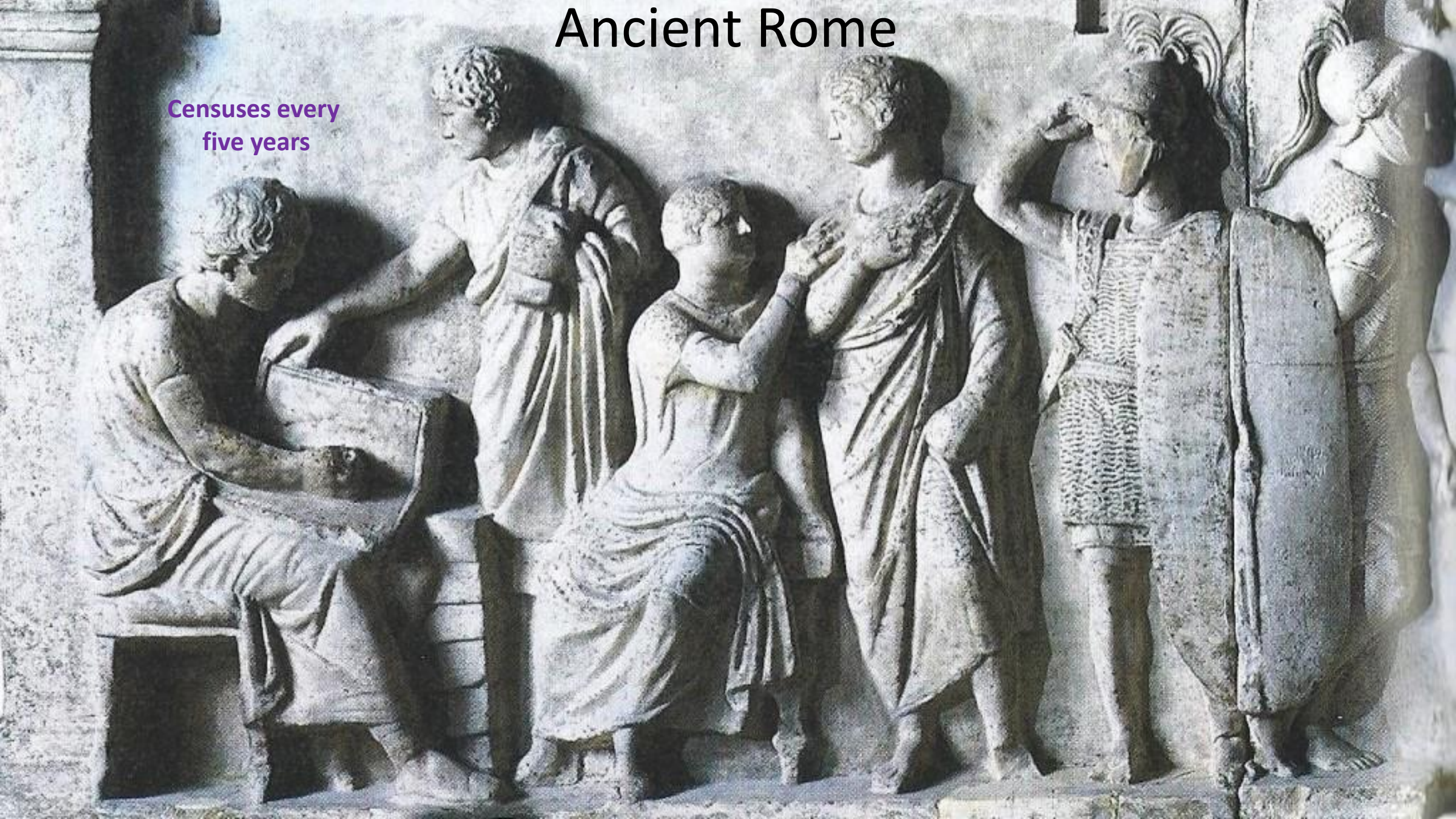
What is Big Data?

- Big data primarily refers to data sets that are too large or complex to be dealt with by traditional data-processing application software.
- While there are plenty of definitions for big data, most of them include the concept of what's commonly known as "three V's" of big data:



Map Reduce

Ancient Rome



Censuses every five years

Ancient Rome
Rome
1
Ephesus
2
Antioch
3
Carthage
4
Alexandria
7
Judaea
6

Census Bureau
Rome
1
Carthage
4
Ephesus
2
Antioch
3
Ancient Rome
Alexandria
7
Judaea
6

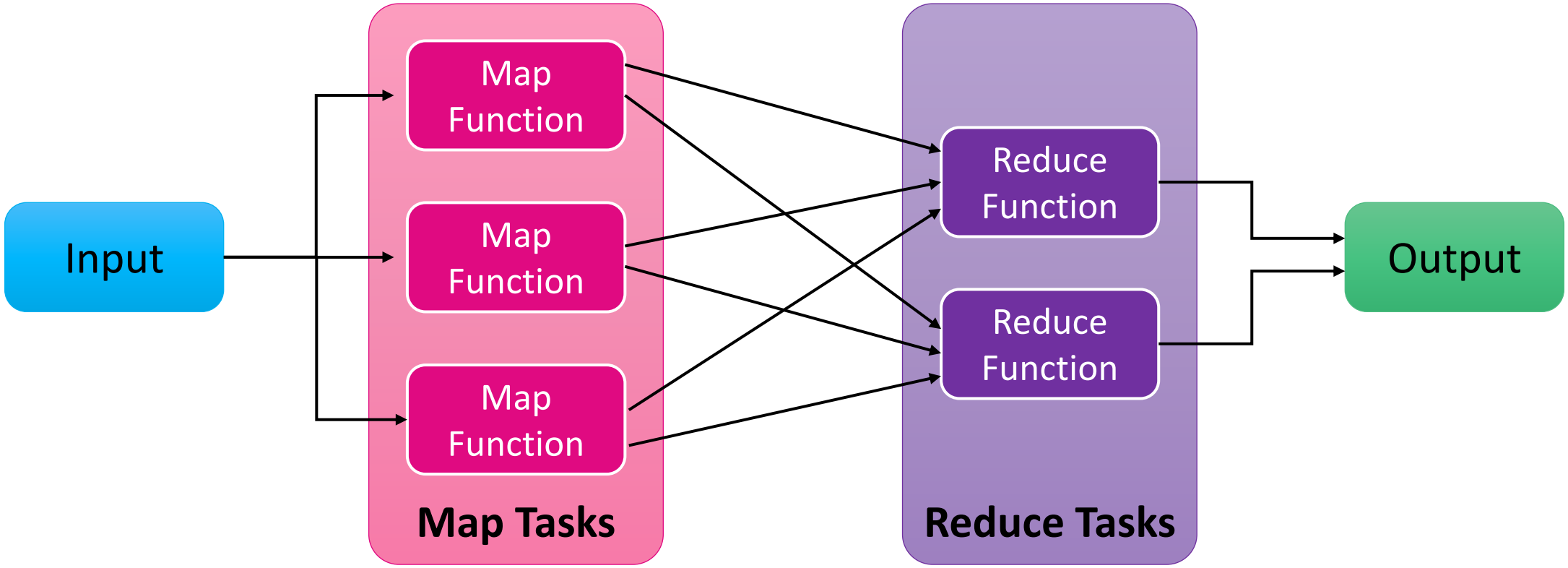
Census Bureau
Rome
1
Carthage
4
Ephesus
2
Antioch
3
Ancient Rome
Alexandria
7
Judaea
6

Census Bureau
Rome
1
Carthage
4
Ephesus
2
Antioch
3
Ancient Rome
Alexandria
7
Judaea
6

Census Bureau
Rome
1
Ephesus
2
Antioch
3
Carthage
4
Ancient Rome
Alexandria
7
Judaea
6

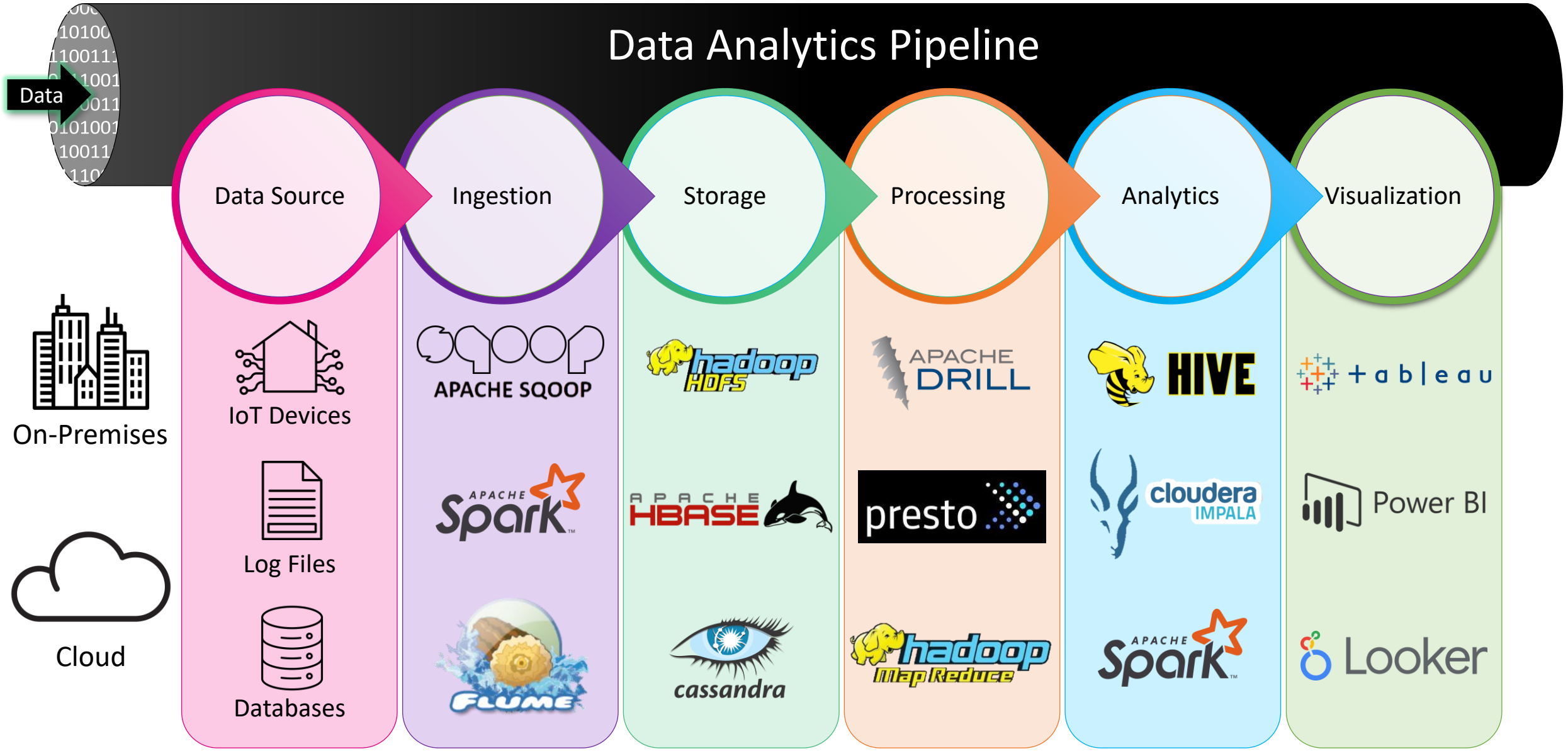
What is MapReduce?

- MapReduce is a programming framework that allows us to perform distributed and parallel processing on large data sets in a distributed environment.



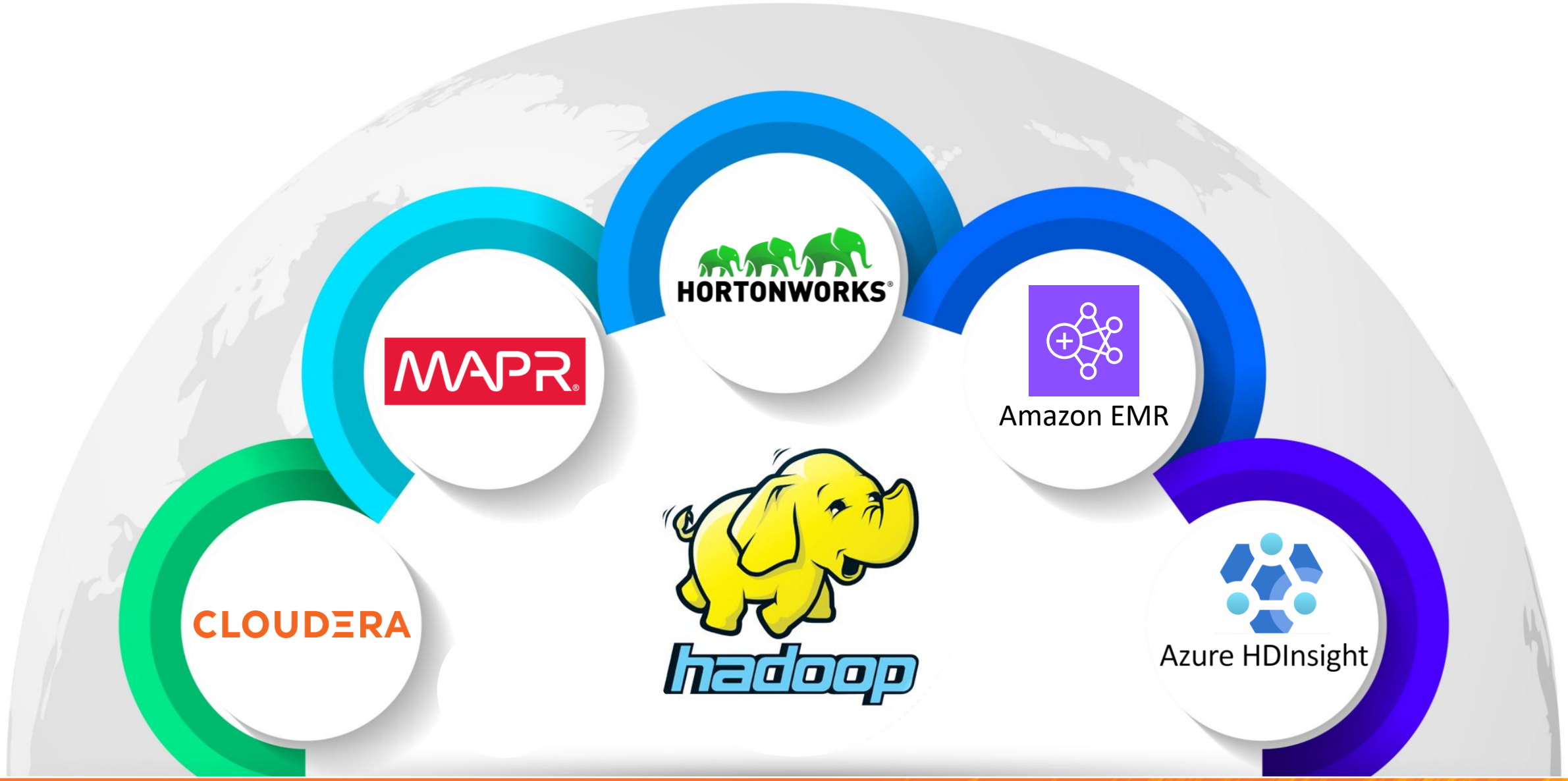
Big Data Ecosystem

Big Data Eco System



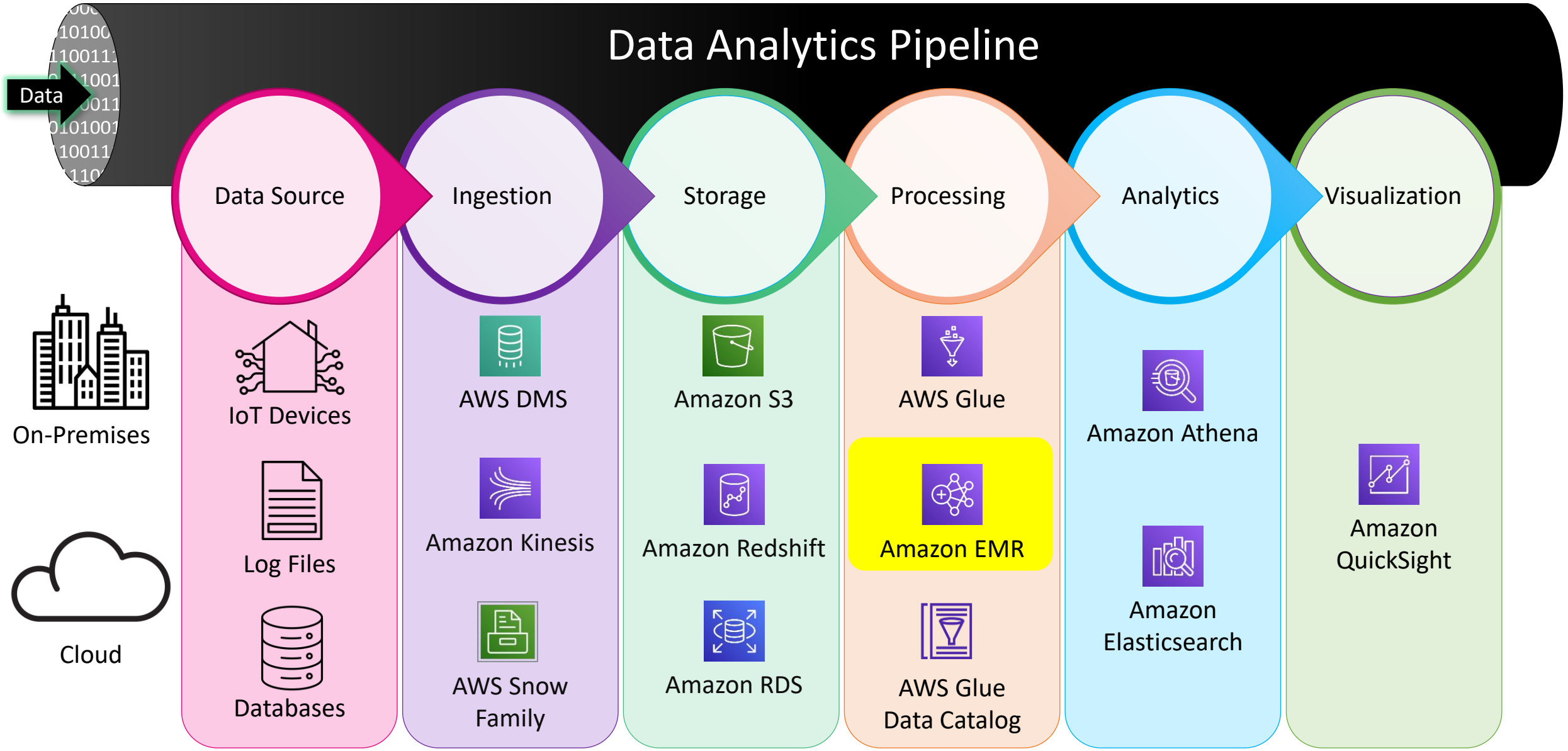
And many more…

Various Hadoop Distribution / Vendors



Amazon EMR

Data Analytics on AWS



Amazon EMR

- Amazon EMR (previously called Amazon Elastic MapReduce) is a managed cluster platform that simplifies running big data frameworks, such as Apache Hadoop and Apache Spark.
- With Amazon EMR you can set up a cluster to process and analyze data with big data frameworks in just a few minutes.

Less admin time to manage, and support Hadoop clusters

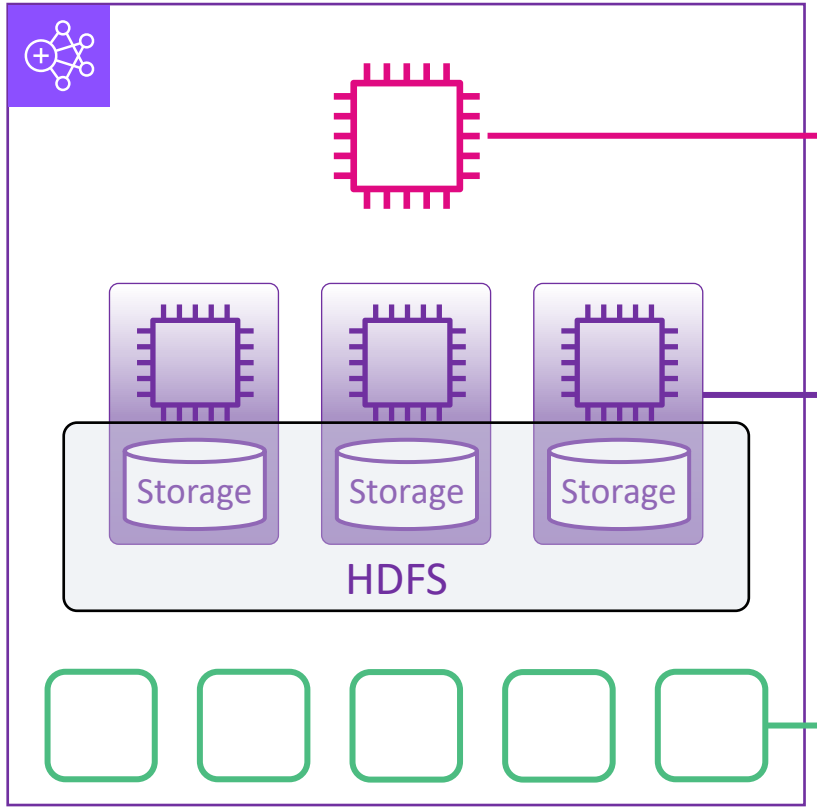
No up-front costs—hardware acquisition, installation

Save on operating costs—data center space, power, cooling

Business Value: Cost of delays, Risk Premium, Competitive abilities, governance, etc.

Amazon EMR architecture

- The central component of Amazon EMR is the cluster. A cluster is a collection of Amazon EC2 instances. Each instance in the cluster is called a node.
- Amazon EMR also installs different software components on each node type, giving each node a role in a distributed application like Apache Hadoop.



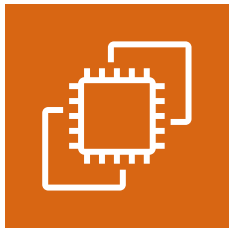
Primary node - A node that manages the cluster by running software components to coordinate the distribution of data and tasks among other nodes for processing. The primary node manages the cluster and the runs the YARN ResourceManager service to manage resources for applications.

Core node - Core nodes run the Data Node daemon to coordinate data storage as part of the Hadoop Distributed File System (HDFS). They also run the Task Tracker daemon and perform other parallel computation tasks on data that installed applications require.

Task node: A node with software components that only runs tasks and does not store data in HDFS. You can use task nodes to add power to perform parallel computation tasks on data. Task nodes are optional.

Amazon EMR Flexible deployment options

Amazon EMR on EC2

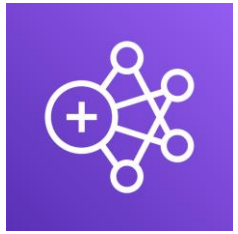


Run workloads with the maximum flexibility on your own Amazon EC2 instances

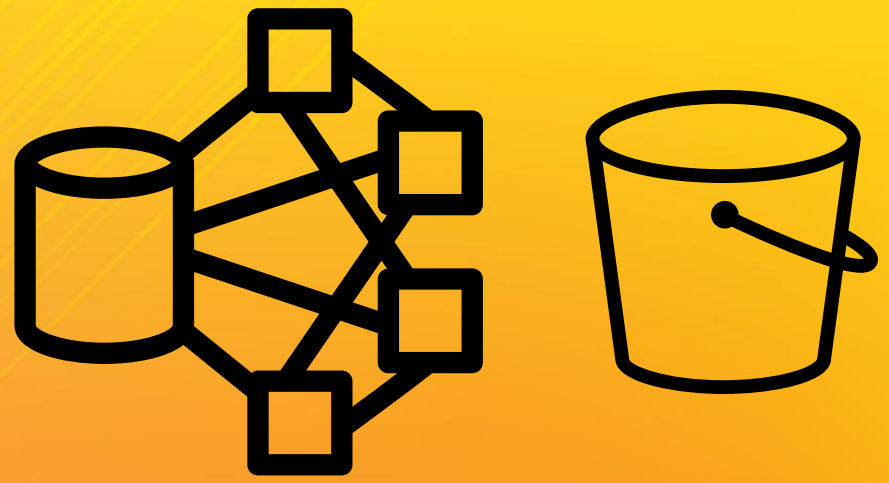
Amazon EMR on EKS

Automate provisioning, management, and scaling on Amazon EKS

Amazon EMR Serverless

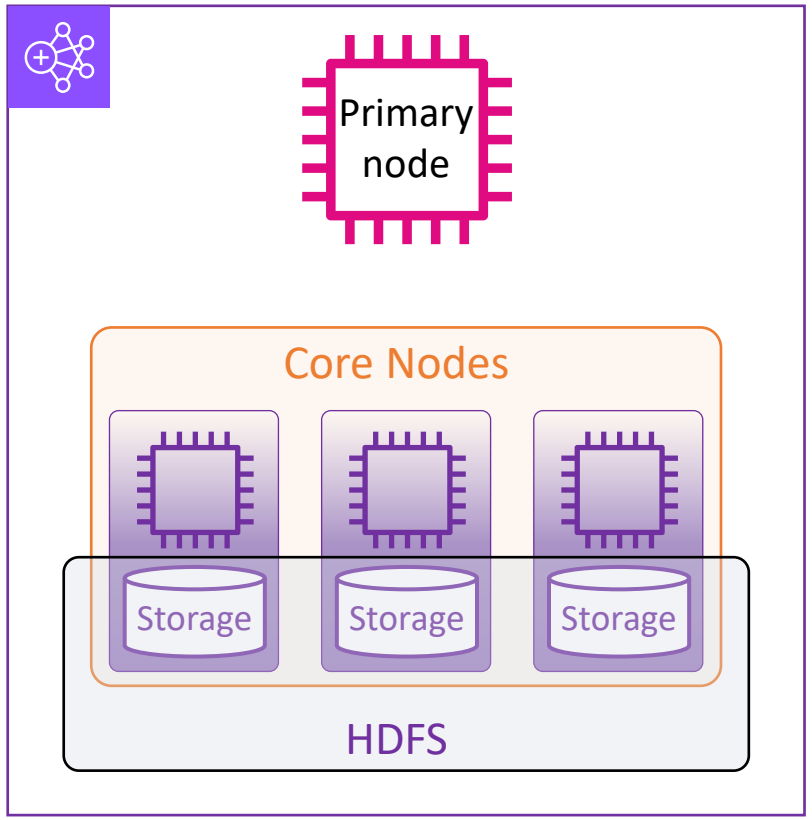


Easiest and simplest way to run workloads with Amazon EMR



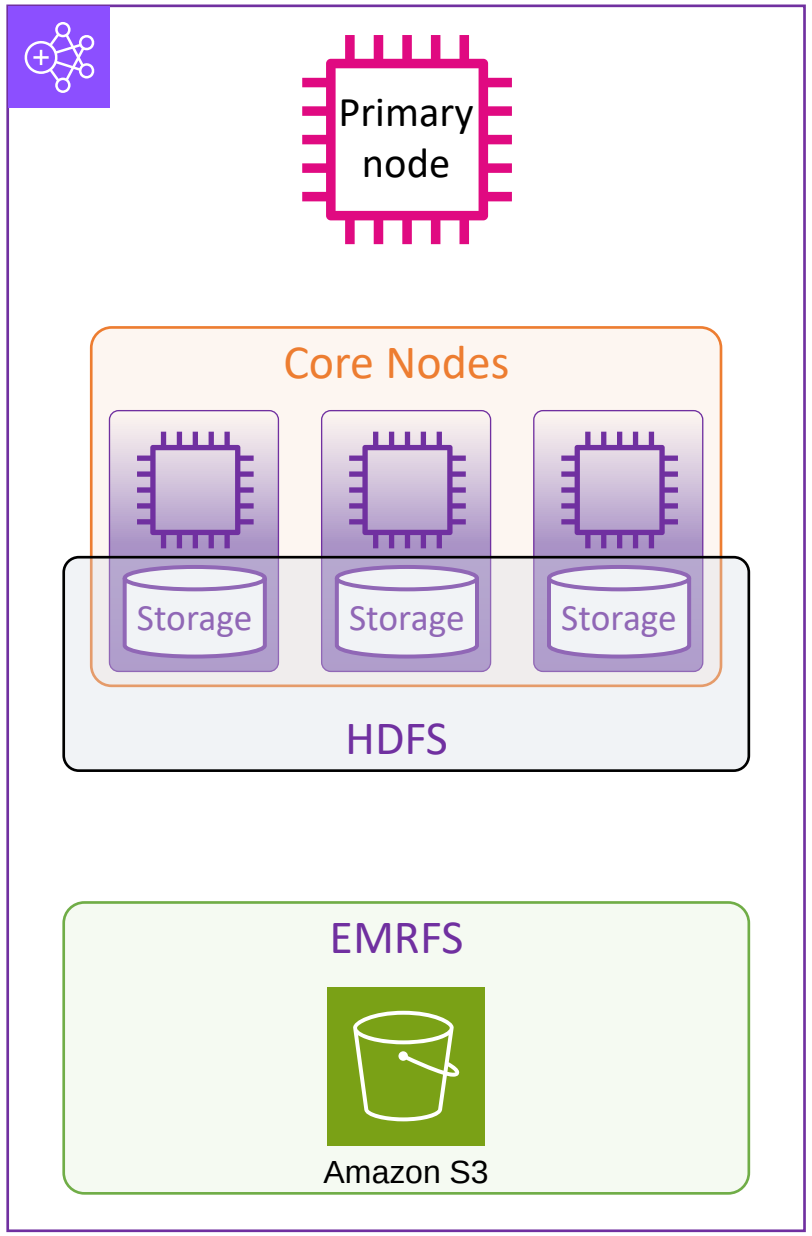
Storage Options for Amazon EMR

Hadoop Distributed File System (HDFS)

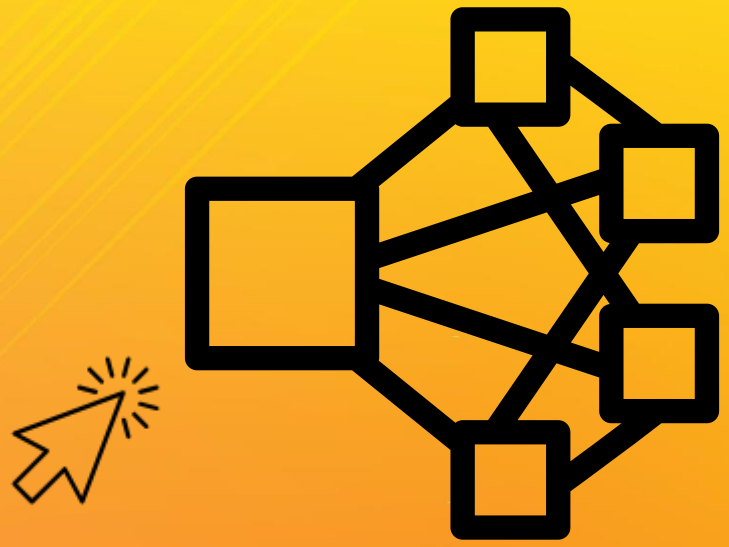


- HDFS is a distributed, scalable, and portable file system for Hadoop.
- Advantage
  - Fast
  - Data awareness between the Hadoop cluster nodes managing the clusters and the Hadoop cluster nodes managing the individual steps.
- Disadvantage
  - It is ephemeral storage which is reclaimed when the cluster ends.
- Suitable for
  - Caching the results produced by intermediate job-flow steps.
  - Iterative reads on the same dataset or Disk I/O intensive workloads.

EMR File System (EMRFS)



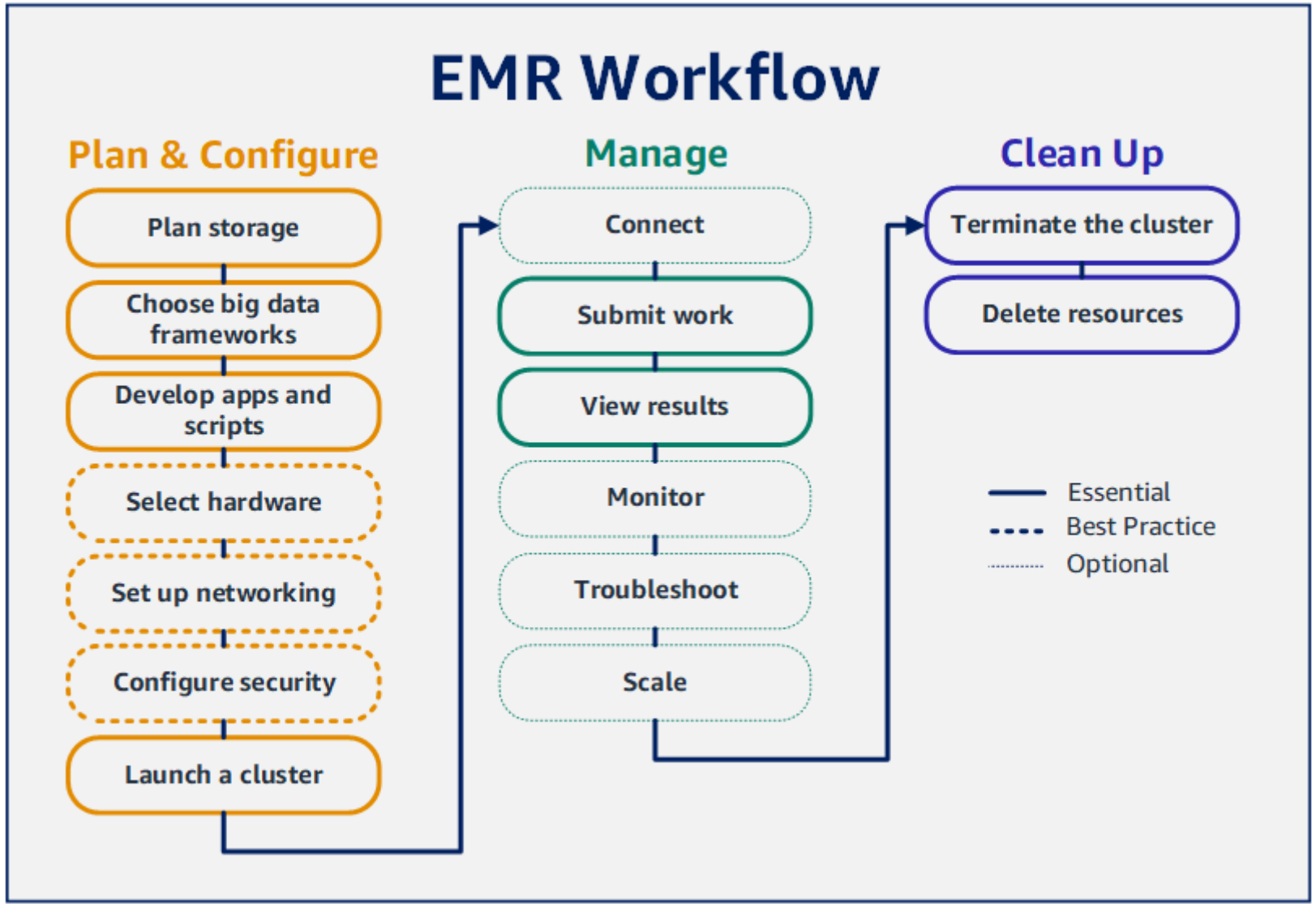
- EMRFS is a connector which allows Hadoop to use Amazon S3 as a storage layer.
- Advantage
  - Decouples storage from compute, so you don't need to provision core nodes specifically to store data.
  - Retain data after shutting down the cluster.
- Disadvantage
  - May not be as performant as HDFS.
- Suitable for
  - Large datasets.
  - Making data available to multiple clusters.
  - Workloads which read the dataset one time in each run.

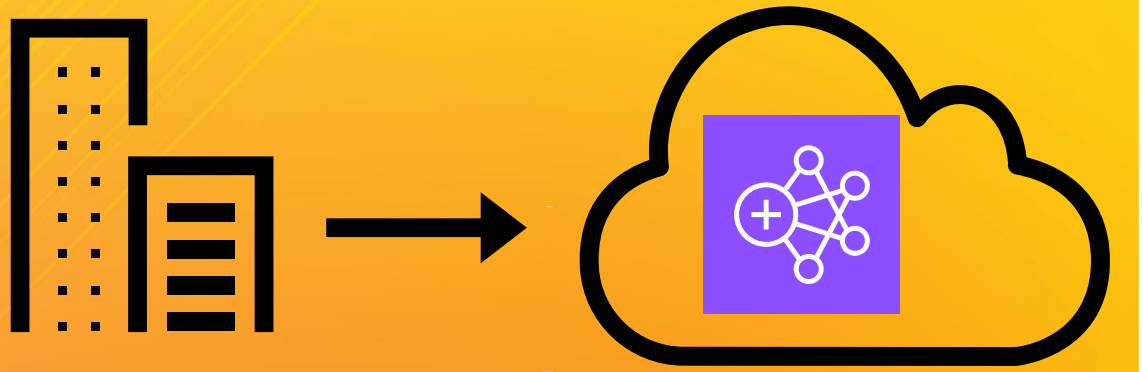


Getting started with EMR

Getting started with Amazon EMR

- Tutorial: Getting started with Amazon EMR





Migrating to Amazon EMR

How to migrate?

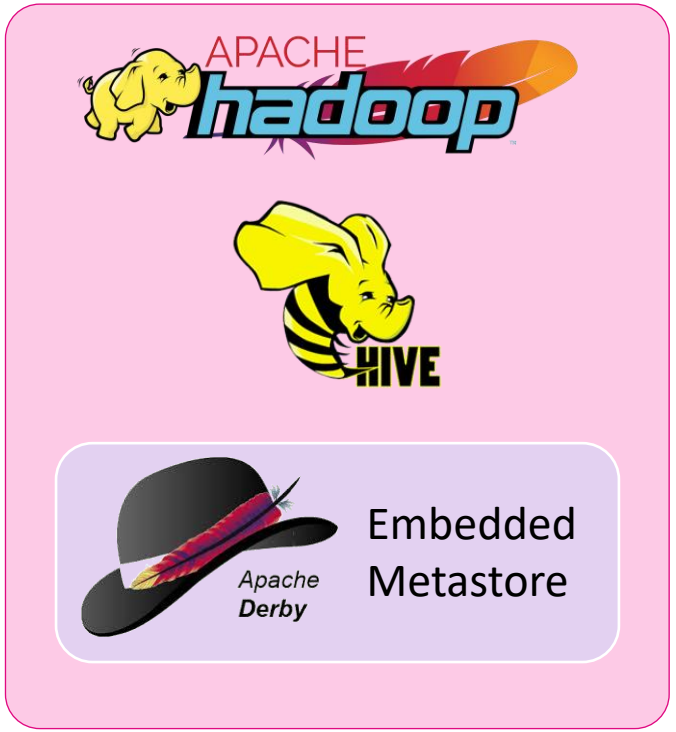
Analytics Migrations

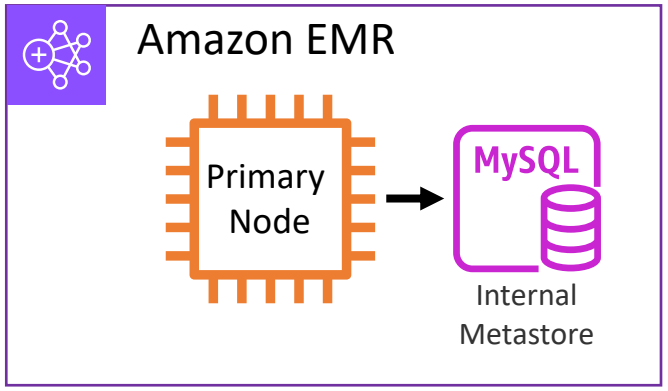
Save time and cost by migrating to fully managed analytics solutions

External Hive Metastore

- Apache Hive is a distributed, fault-tolerant data warehouse system that enables analytics at a massive scale.
- Hive Metastore (HMS) provides a central repository that stores the metadata (Schema, Locations etc.) for Hive tables and relations.
- A whole ecosystem of tools, open-source and otherwise, are built around the Hive Metastore.
- It is and therefore it is a critical component of many data analytics architectures.
- By default, the Hive Metastore stores all Hive metadata in an embedded Apache Derby database in the filesystem.

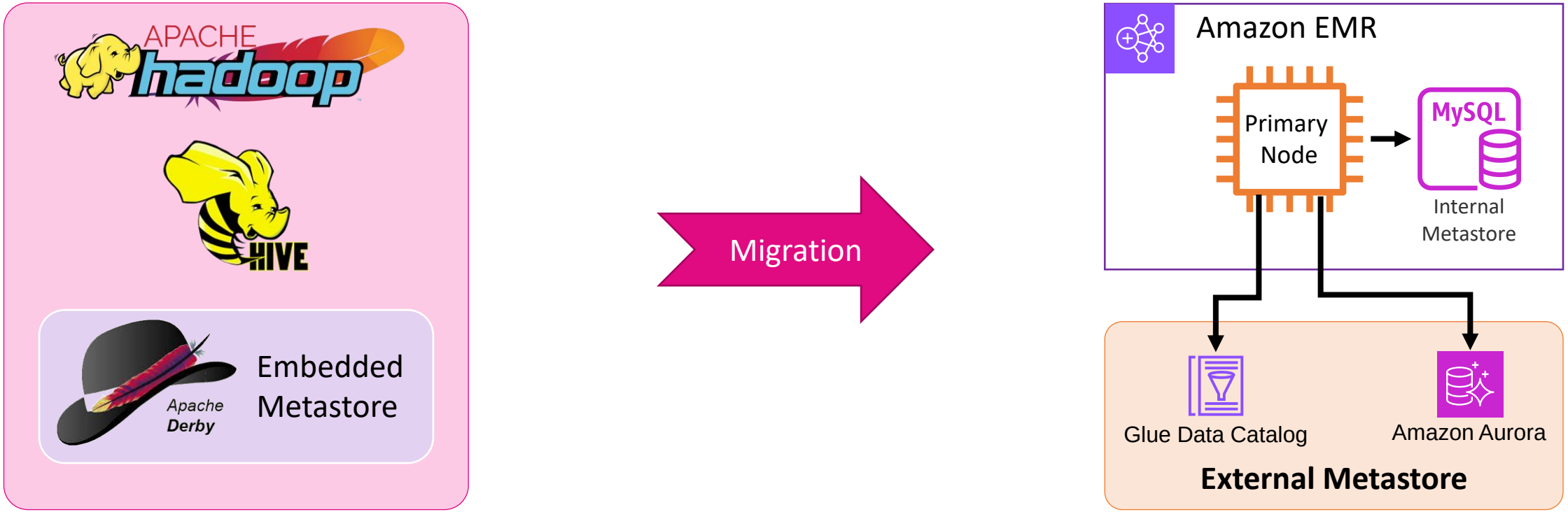
Migrating to Amazon EMR





- In Amazon EMR, Hive records metastore information in a MySQL database on the primary node's file system.
- When a cluster terminates, all cluster nodes shut down, including the primary node. When this happens, local data is lost because node file systems use ephemeral storage.

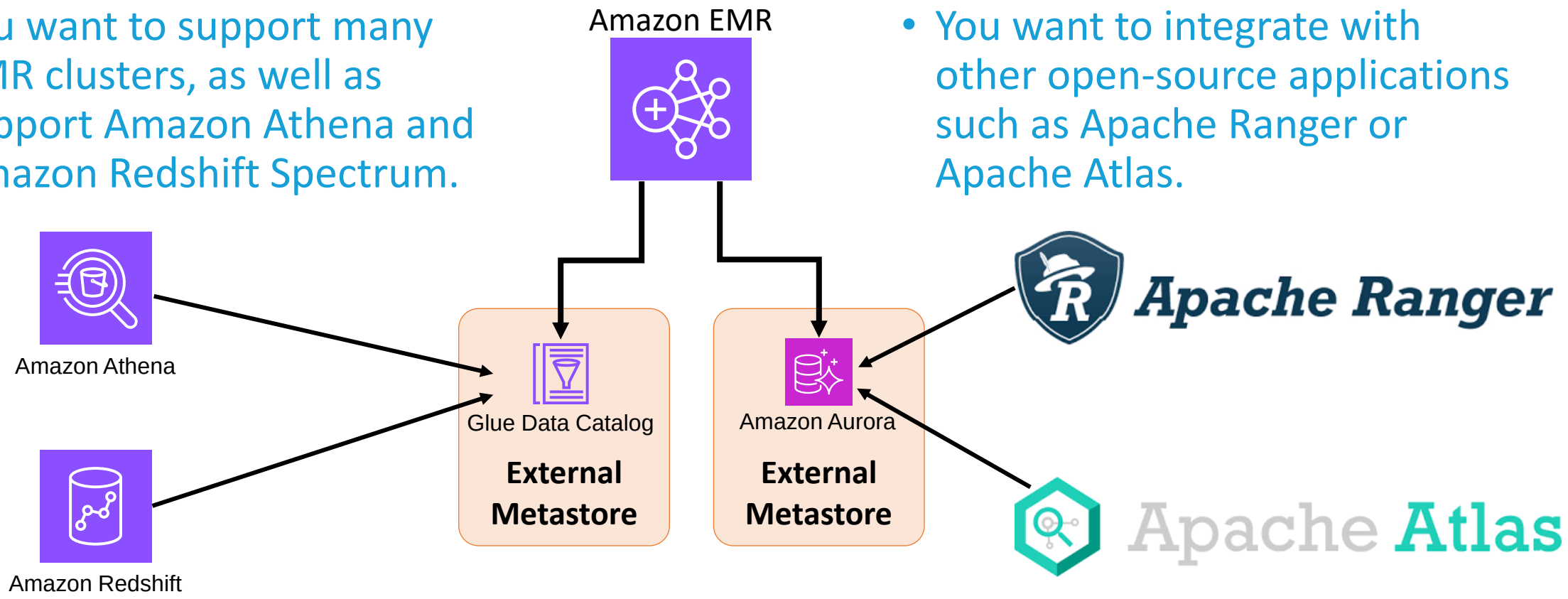
Migrating to Amazon EMR



- If you need the metastore to persist, you must create an external metastore that exists outside the cluster.
- You have two options for an external metastore:
  - AWS Glue Data Catalog
  - Amazon RDS or Amazon Aurora

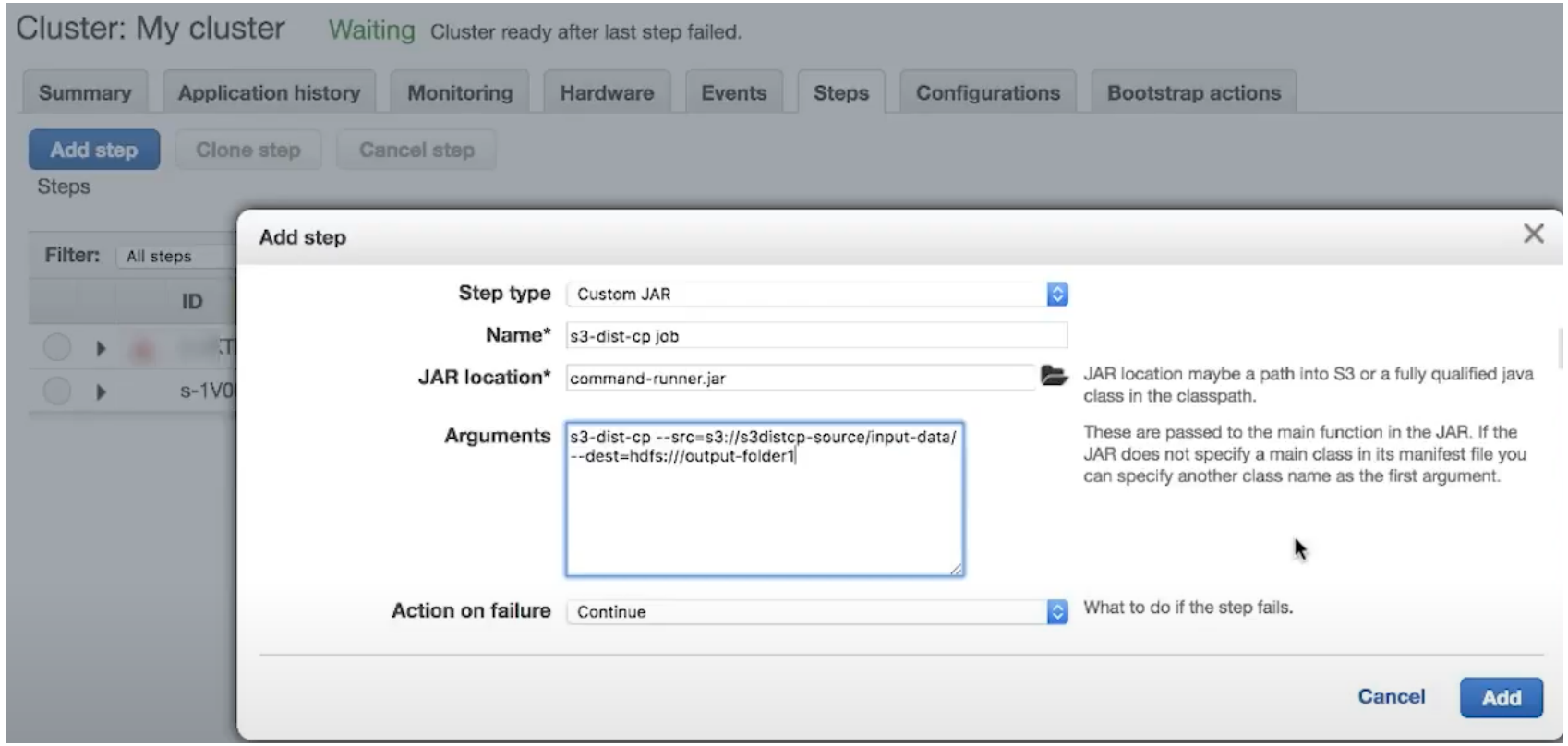
Which one to choose?

- Use AWS Glue Data Catalog if:
  - You want a fully managed database.
  - You want to support many EMR clusters, as well as support Amazon Athena and Amazon Redshift Spectrum.
- Use Amazon RDS or Amazon Aurora if:
  - You want full control of your Hive metastore.
  - You want to integrate with other open-source applications such as Apache Ranger or Apache Atlas.



S3DistCp

- S3DistCp is an application which enables copying large amounts of data between Amazon S3 and HDFS. It is installed on EMR cluster by default.
- S3DistCp is an extension of the open-source tool, DistCp, so it creates a MapReduce job underneath it.

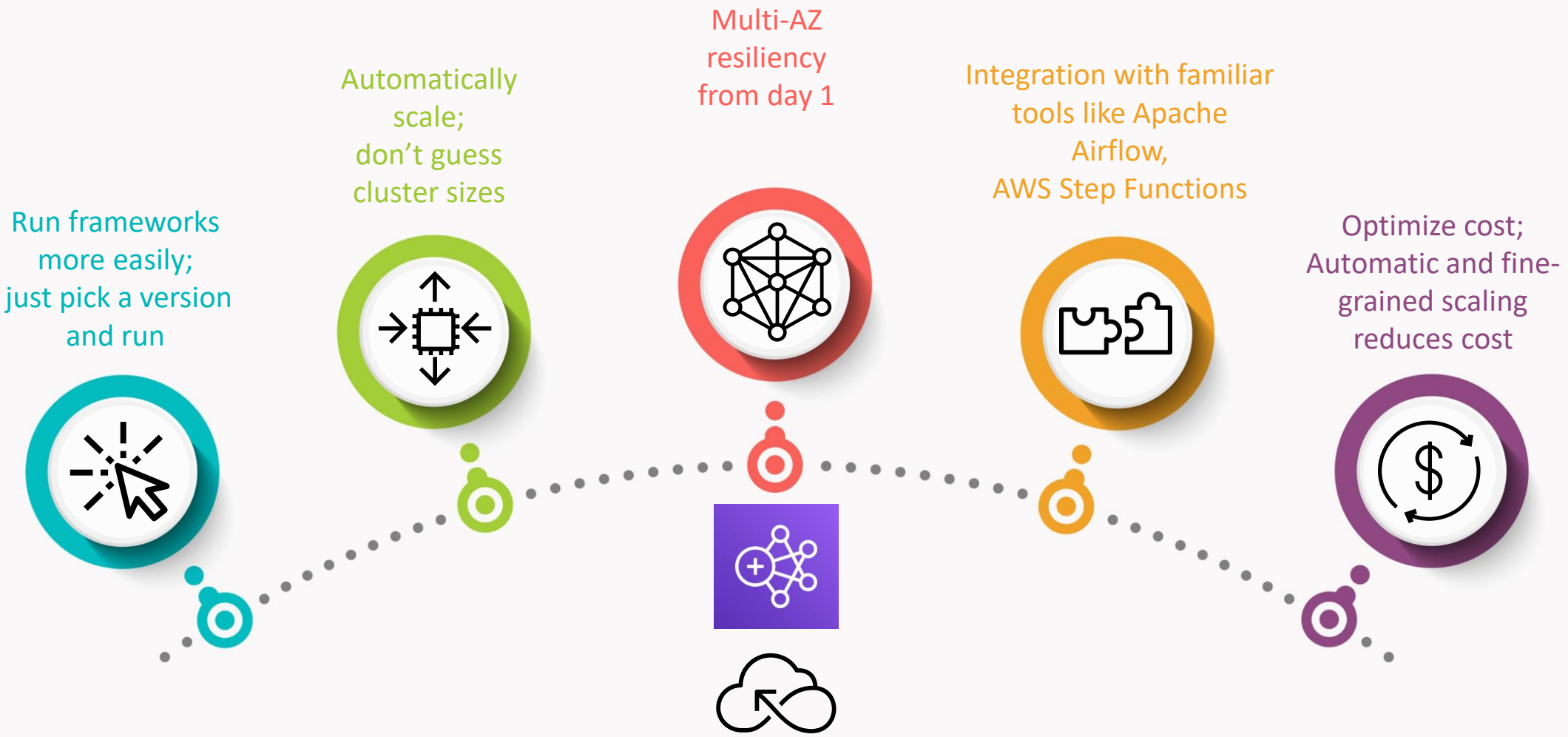


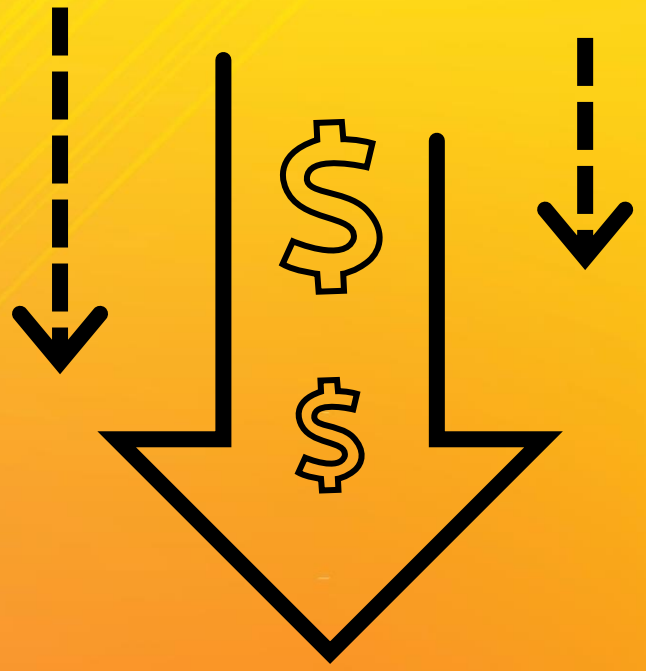
- You can run S3DistCp using EMR step, or execute the command in a terminal on EMR primary node.

Amazon EMR Serverless

Amazon EMR Serverless

- All the benefits of EMR without managing clusters and servers

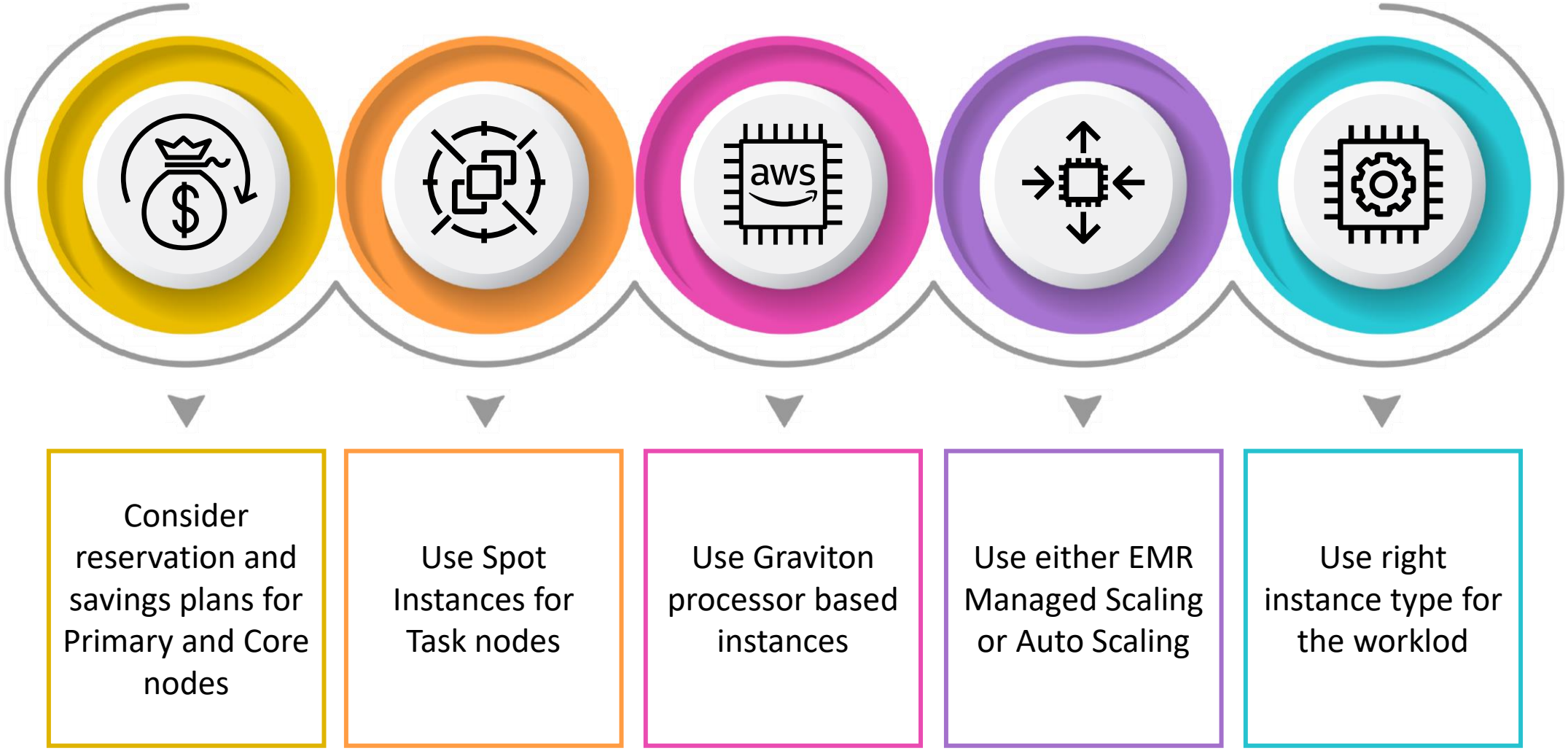




Cost Optimization

Optimizing Amazon EMR on Amazon EC2 Pricing

- The Amazon EMR price is added to the Amazon EC2 price.



Amazon EMR on EC2

Streaming Services in AWS (Not an exhaustive list)

Stream Producer

- Amazon Kinesis Producer Library (KPL)
- AWS SDK
- Amazon Kinesis Agent
- Amazon MSK Connect
- Other AWS Services

Stream Storage

- Amazon Kinesis Data Streams
- Amazon Managed Streaming for Apache Kafka (Amazon MSK)

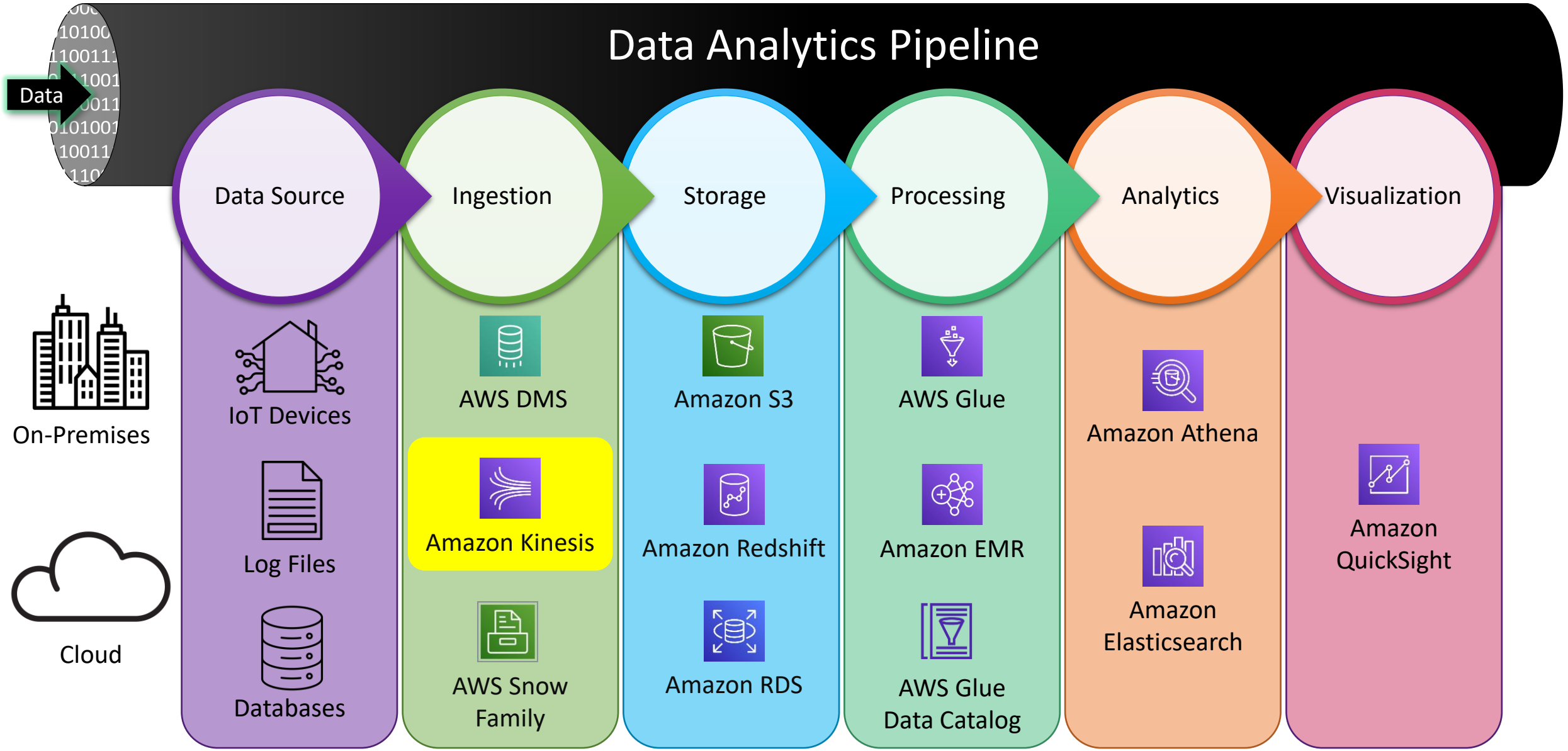
Stream Consumer

- Amazon Data Firehose
- Amazon Managed Service for Apache Flink
- AWS Glue
- Amazon EMR
- Custom Apps on EC2, Containers, Lambda using KCL

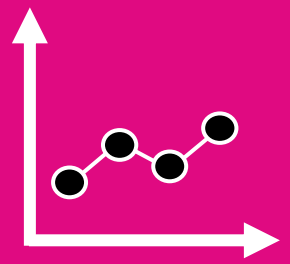
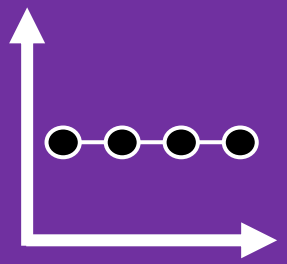
Stream Destination

- Amazon S3
- Amazon Redshift
- Amazon OpenSearch Service
- Amazon DynamoDB
- 3rd Party Services

Data Analytics on AWS

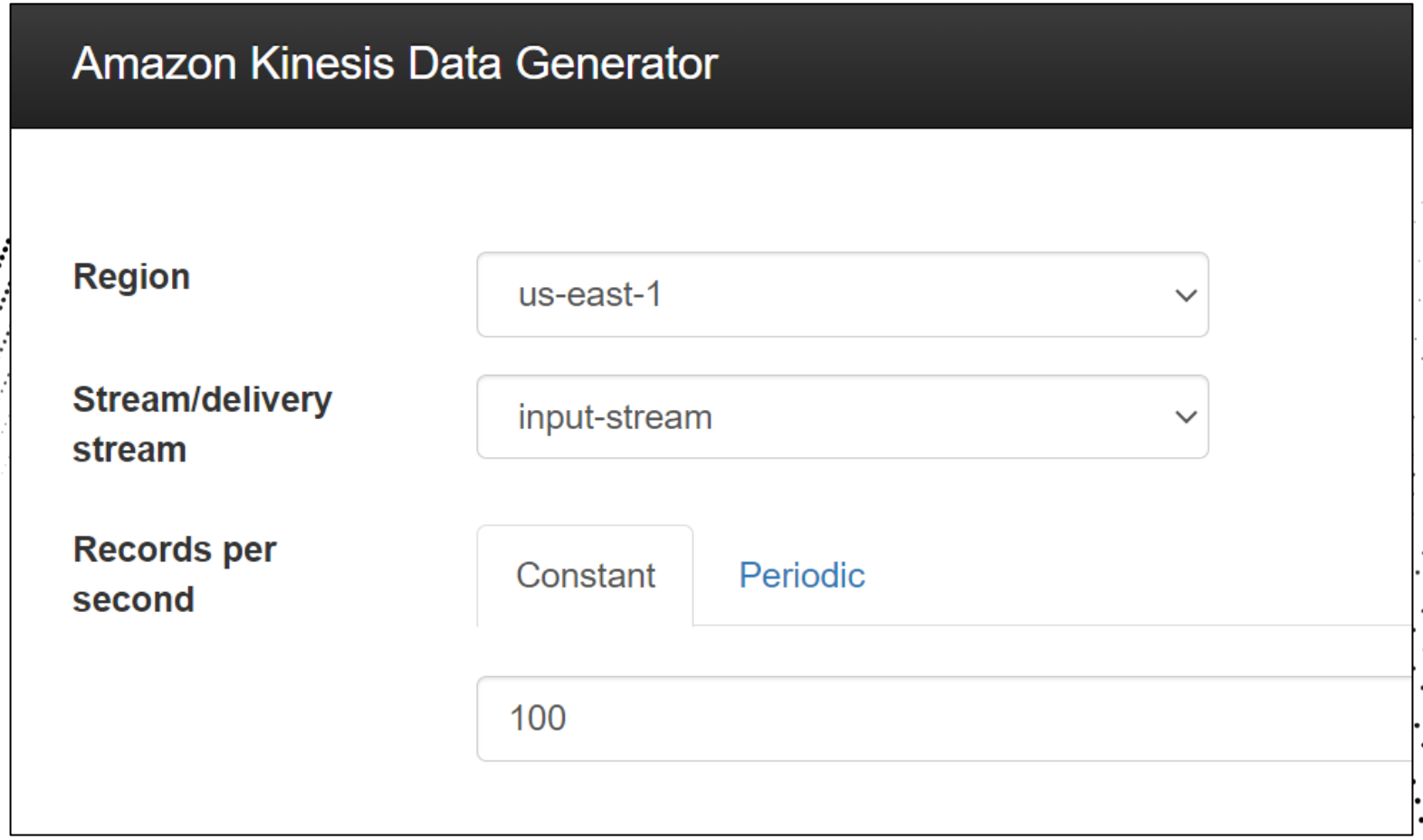


Capacity modes

On Demand	Provisioned
	
New streams with unknown traffic	Predictable traffic
Unpredictable traffic	Traffic is consistent
Prefer the ease of hands-free management	Need tight control of shards
Lower cost of ownership	Predictable costs
Limited to 200 MB/s of write and 400 MB/s of read throughput	No capacity limits for a provisioned stream

Amazon Kinesis Data Generator

- The KDG makes it simple to send test data to your Amazon Kinesis stream or Amazon Kinesis Firehose delivery stream.



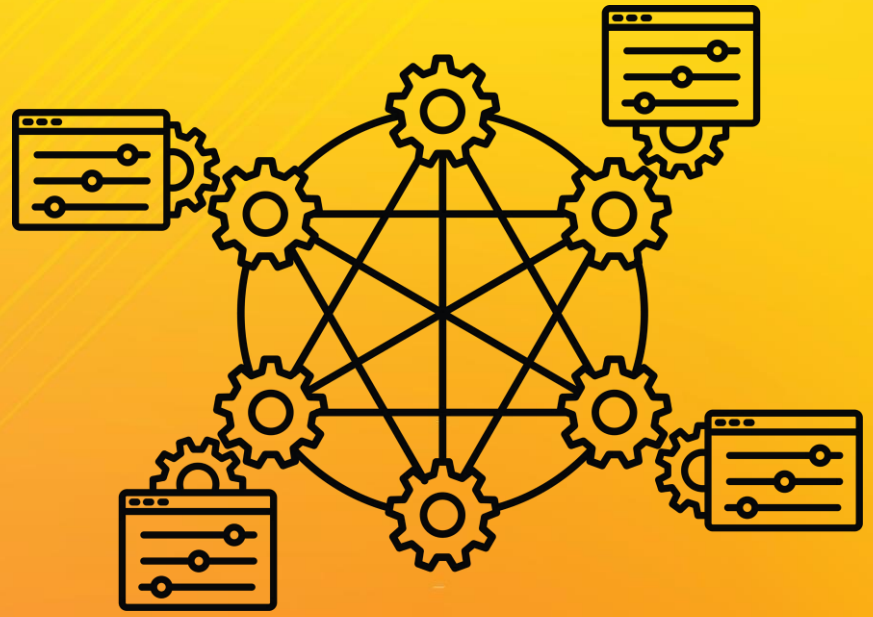
Processing data stored in Kinesis Data Streams (2)

- You can read data from Amazon Kinesis Data Streams data streams using one of the following third-party options that integrate with Kinesis Data Streams:
  - Apache Flink
  - Adobe Experience Platform
  - Apache Druid
  - Apache Spark
  - Databricks
  - Kafka Confluent Platform
  - Kinesumer
  - Talend

Comparison

	Shared throughput	Enhanced fan-out
When to use	• Total number of consuming applications is low (< 3) • Consumers are not latency-sensitive	• Multiple consumer applications • Low-latency requirements for data processing
Shard Read Throughput	• Fixed at a total of 2 MB/sec per shard. If there are multiple consumers reading from the same shard, they all share this throughput. The sum of the throughputs they receive from the shard doesn't exceed 2 MB/sec.	• Scales as consumers register to use enhanced fan-out. Each consumer registered to use enhanced fan-out receives its own read throughput per shard, up to 2 MB/sec, independently of other consumers.
Message propagation delay	• An average of around 200 ms if you have one consumer reading from the stream. This average goes up to around 1000 ms if you have five consumers.	• Typically an average of 70 ms whether you have one consumer or five consumers.
Cost	• N/A	• There is a data retrieval cost and a consumer-shard hour cost.
Record delivery model	• Pull model over HTTP using GetRecords.	• Kinesis Data Streams pushes the records to you over HTTP/2 using SubscribeToShard.

	Amazon Kinesis Data Streams	Amazon Data Firehose
Objective	• Streaming data ingestion and storage	• Streaming data transfer and format conversion
Scaling	• Provisioned mode – Manual • On demand mode – Automated	• Automated
Data Retention	• Configuration (1 day to 365 days)	• Not applicable
Processing	• Real-time processing capabilities with almost 200ms latency for classic tasks and almost 70ms latency for enhanced fan-out tasks.	• Near real-time processing capabilities, depending on the buffer size or minimum buffer time of 60 seconds.
Replay Capabilities	• Supported	• Unsupported
Parallel Consumption	• Supported	• Unsupported
Max message size	• 1 MB	• 1 MB • 10 MB if your data source is Amazon MSK
Data Producers (Partial list)	• Amazon Kinesis Agent • AWS SDK • Amazon Kinesis Producer Library (KPL)	• Amazon Kinesis Data Stream • Amazon MSK • Direct Put
Data Consumers (Partial list)	• Managed Apache Flink • Amazon Data Firehose • Amazon Kinesis Client Library (KCL)	• Amazon OpenSearch • Amazon Redshift • Amazon S3 • HTTP Endpoints & 3rd Party Services



Map Reduce

Apache Flink on AWS

Amazon Managed Service for Apache Flink

Fully managed stream processing

Build and run Apache Flink applications, without setting up infrastructure and managing resources and clusters.

AWS managed scaling

Amazon EMR

Flexibility with less management

Amazon EMR supports Apache Flink as a YARN application.

Amazon EKS

Full flexibility with complete control

Apache Flink natively supports Kubernetes. You can self-host Apache Flink on Amazon EKS or Amazon EC2.

Customer managed scaling

Amazon Managed Flink Studio

Stream inspection & visualization

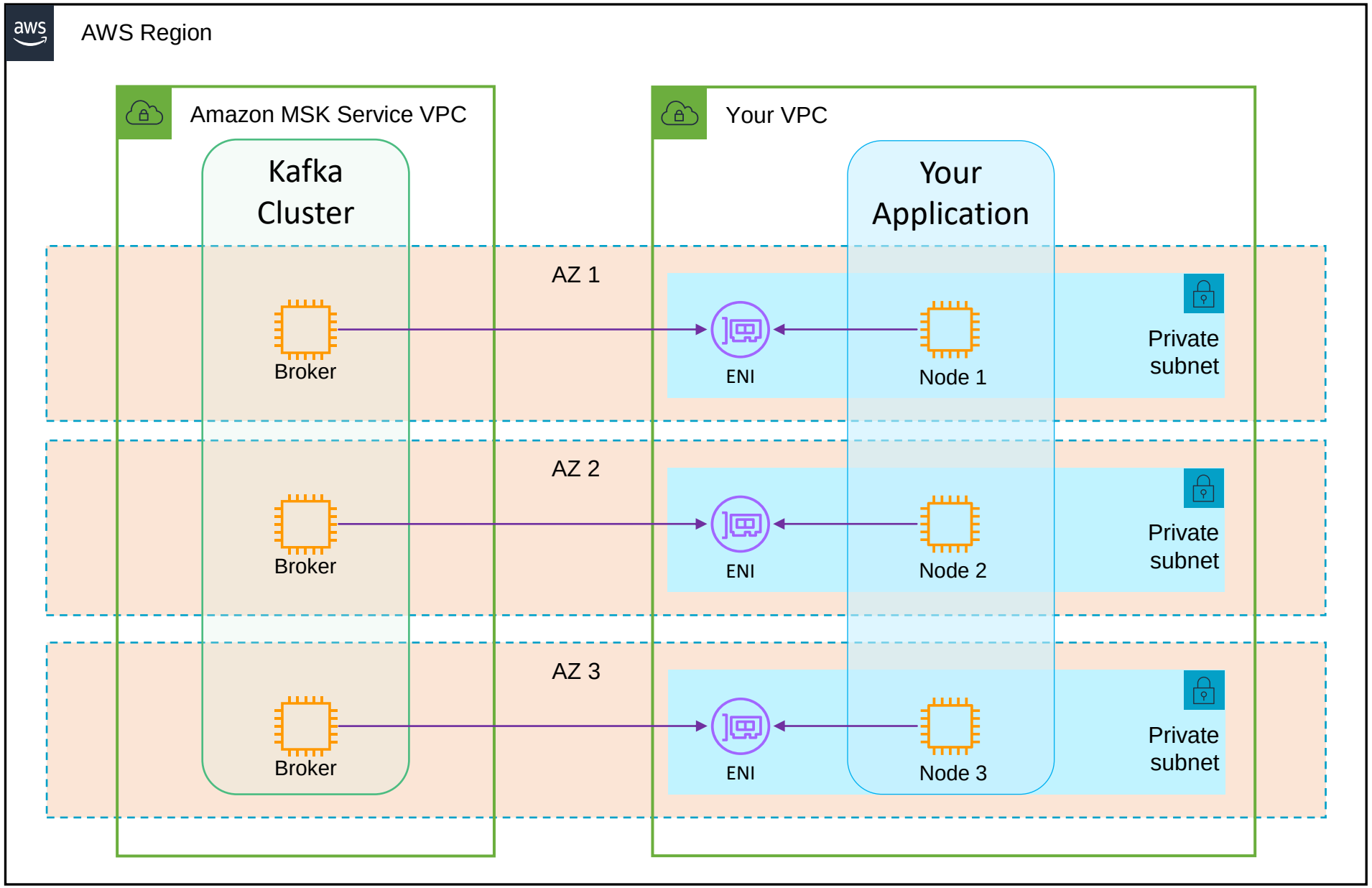
Simple build and run environment

Serverless, rapid development of stream processing applications

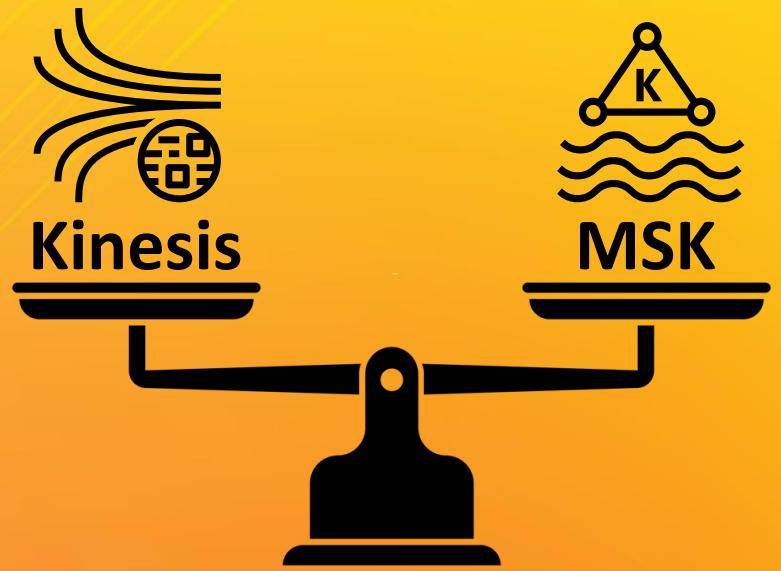
Process using SQL, Python, or Scala

Integrates with AWS Glue Data Catalog

Amazon MSK Connectivity



	Amazon MSK Provisioned Cluster	Amazon MSK Serverless
Creating a new cluster	Select the number of brokers, size of the instances, storage capacity, broker instance type.	No capacity configuration needed.
Workload type and use-case	Predictable workload. Stable, have more control, large workloads.	Unpredictable workload. Offload capacity management.
Scaling	Manually adjust the capacity configuration to adhere to the traffic increases.	Automatically provisions and manages capacity needed during traffic increases.
Monitoring	Manually monitor overload/idle brokers.	Auto-scales on demand so do not need to constantly monitor brokers and storage.
Configurations	Allows broker level configurations	Allows a few topic level configurations
Pricing dimensions	On-demand hourly pricing for brokers and storage	Pay-per-use pricing based on cluster hours, partition hours, storage



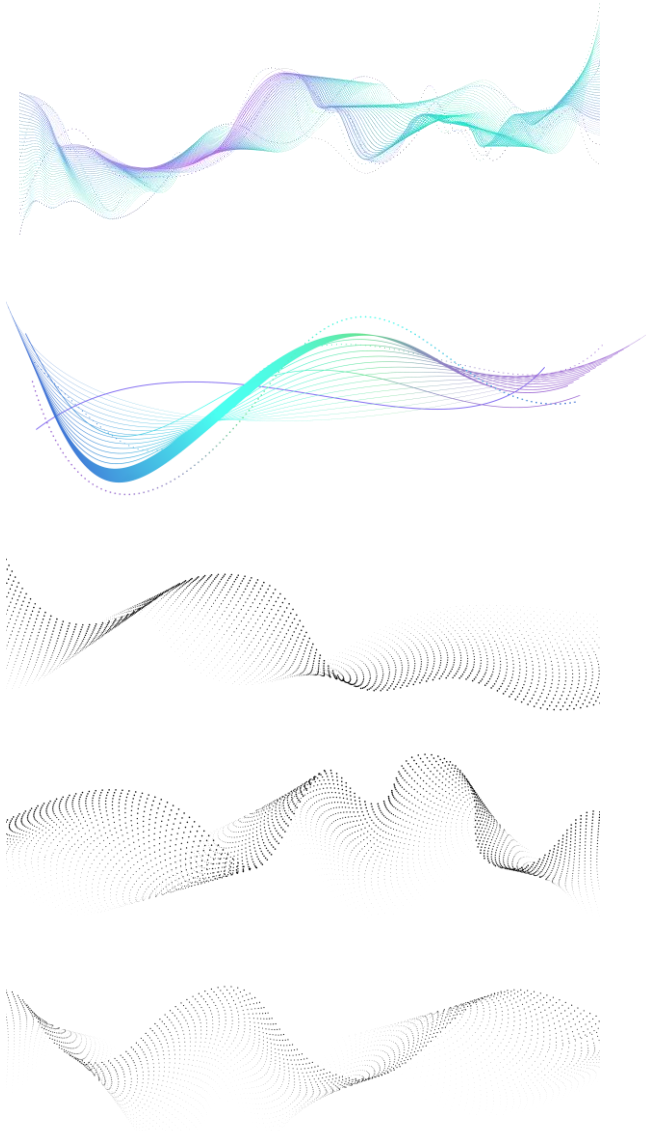
Comparison

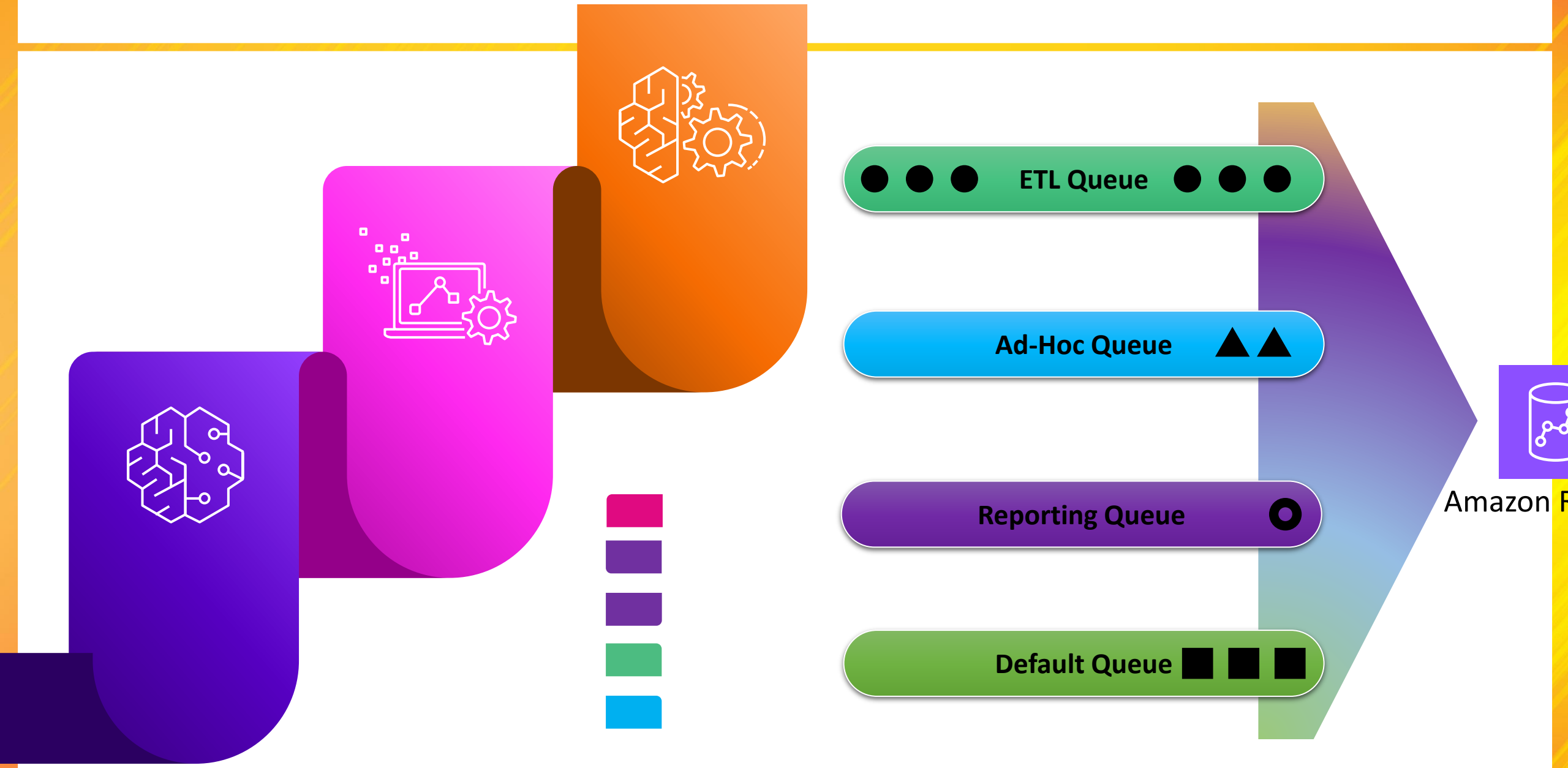
Similarity	AWS managed infrastructure
	Parallel consumption
	Guaranteed ordering: Kinesis – Shard level, MSK – Partition level
	Support Managed Service for Apache Flink

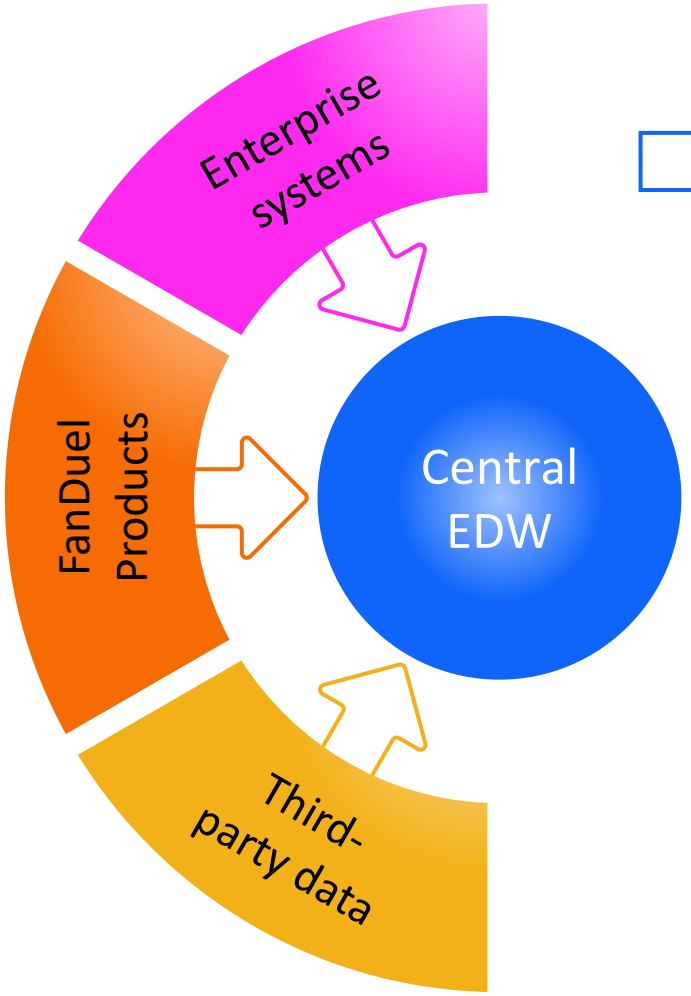
	Amazon Kinesis Data Stream	Amazon MSK
Differences	Serverless	Provisioned and Serverless
	Proprietary to AWS	Based on open source Apache Kafka
	3 AZ replication	Configurable replication
	Suitable for cloud native application	Suitable for on-prem Kafka migration to cloud
	1 MB message/blob size (blob)	Configurable message/blob size
	Integrates natively with AWS services	Supports integration with Kafka ecosystem
	Low (70ms with Enhance Fan Out)	Lowest

Using Lambda with Kinesis Data Streams

- https://docs.aws.amazon.com/lambda/latest/dg/with-kinesis-example.html




ETL Queue
Ad-Hoc Queue
Reporting Queue
Default Queue
Amazon R



Sportsbook Online	Retail	Casino
Racing	Daily Fantasy	Skill Gaming
F2P	Canada	Fraud & Payments
CS/VIP	Risk & Trading	Marketing
Product	Finance	Tax
Legal Compliance	Partnerships	HR

Risk and trading customer insights & personalization delayed

7 hours

Critical insights blocked

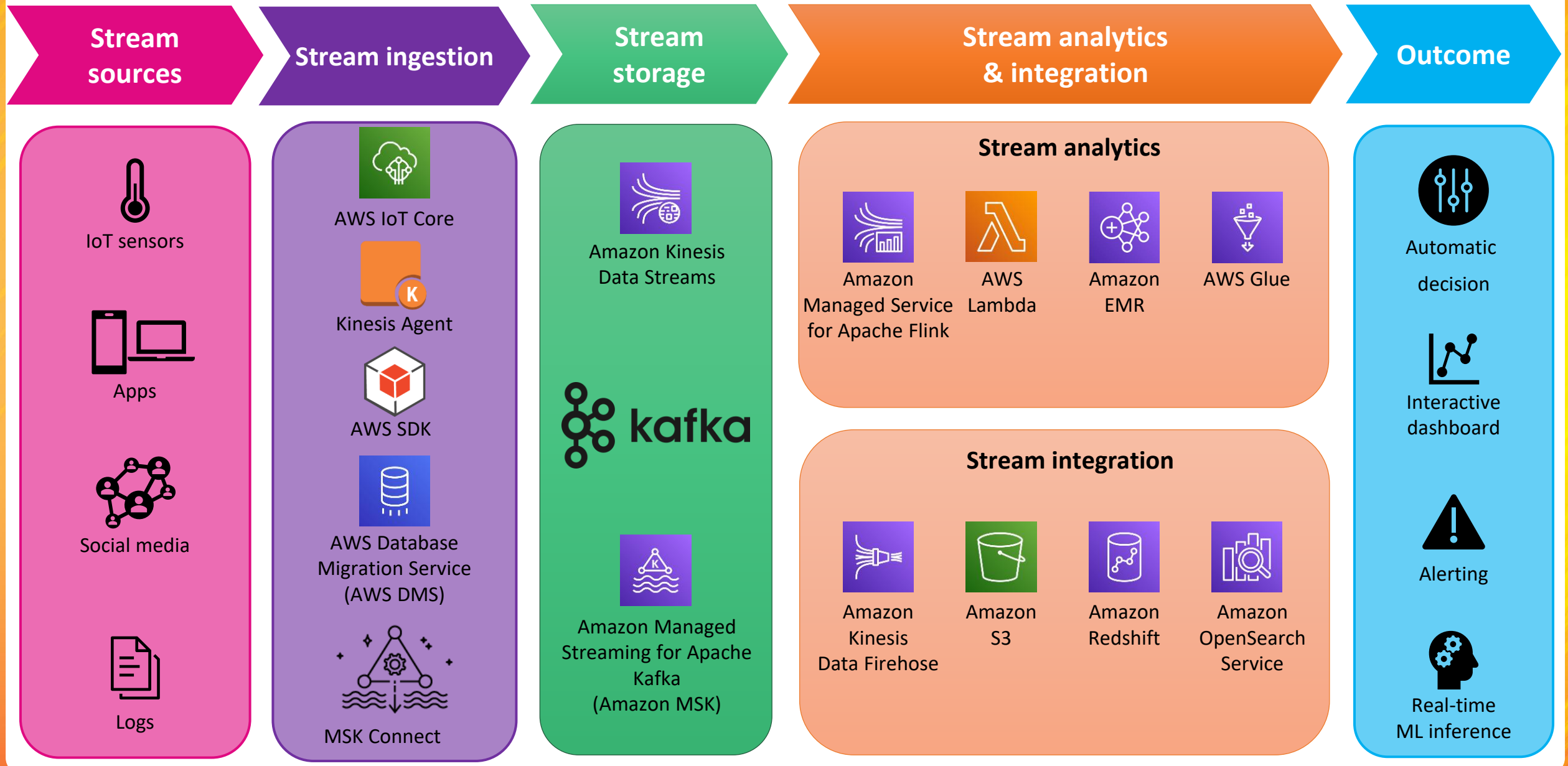
Key metrics (estimated revenue, segmentation, risk profiling) blocked

$50K

Business value loss

No other queries run until it is done

Streaming data pipeline



Data analysis
2020
Monthly
5.8k
7.1k
XRP
$ 230.30
+6.44%
01
BTC
$ 1148.41
+ 9.31%
02
LTC
$42.32
-2.19%
03
ETH
$ 440.17
+6.11%
04
9:30 10:00 10:30 11:00 11:30 12:00 12:30
M T W T F S S
64%
58%
62%
76%
Lorem ipsum dolor sit amet, consectetur adipiscing elit. Vivamus nec eros enim. Aliquam erat volutpat.
5371
10%
14%
21%
10%
13%
17%
15%
Dolor sit
Consectetur
Donec tortor
Nunc dapibus
Phasellus nibh
Consectetur
Nunc
+47%
ETH Ethereum
BTC Bitcoin
XRP Ripple
LTC Litecoin
BCH Bitcoin Cash
EOS EOS
ZEC Zcash
0x ZRX
TRX TRON
XLM Stellar
IOT IOTA
XRP Ripple
LINK Chainlink
DASH Dash