# BIG DATA ON AWS

Ankara Cloud Meetup – July 2016

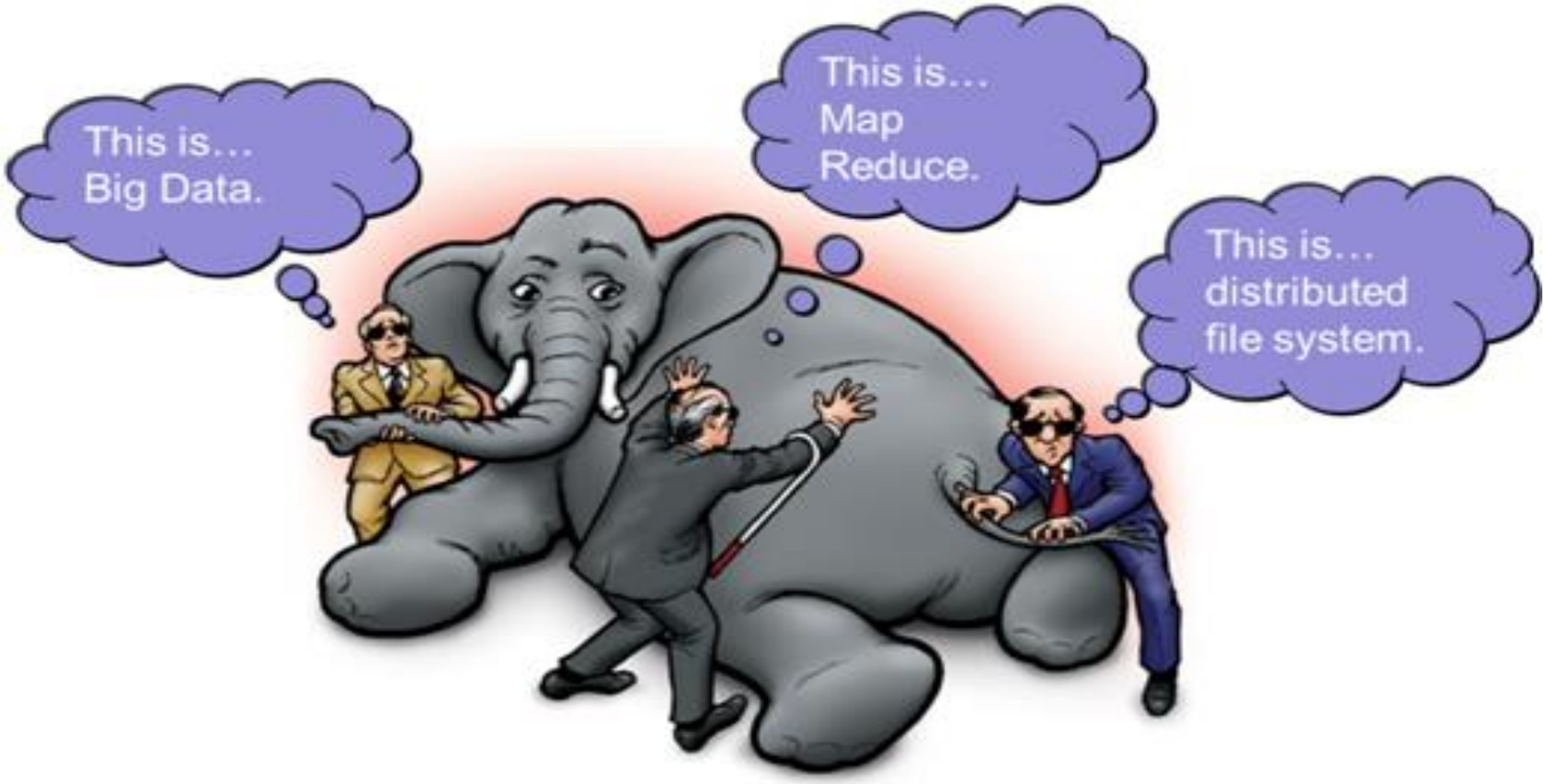
Serkan ÖZAL

# AGENDA

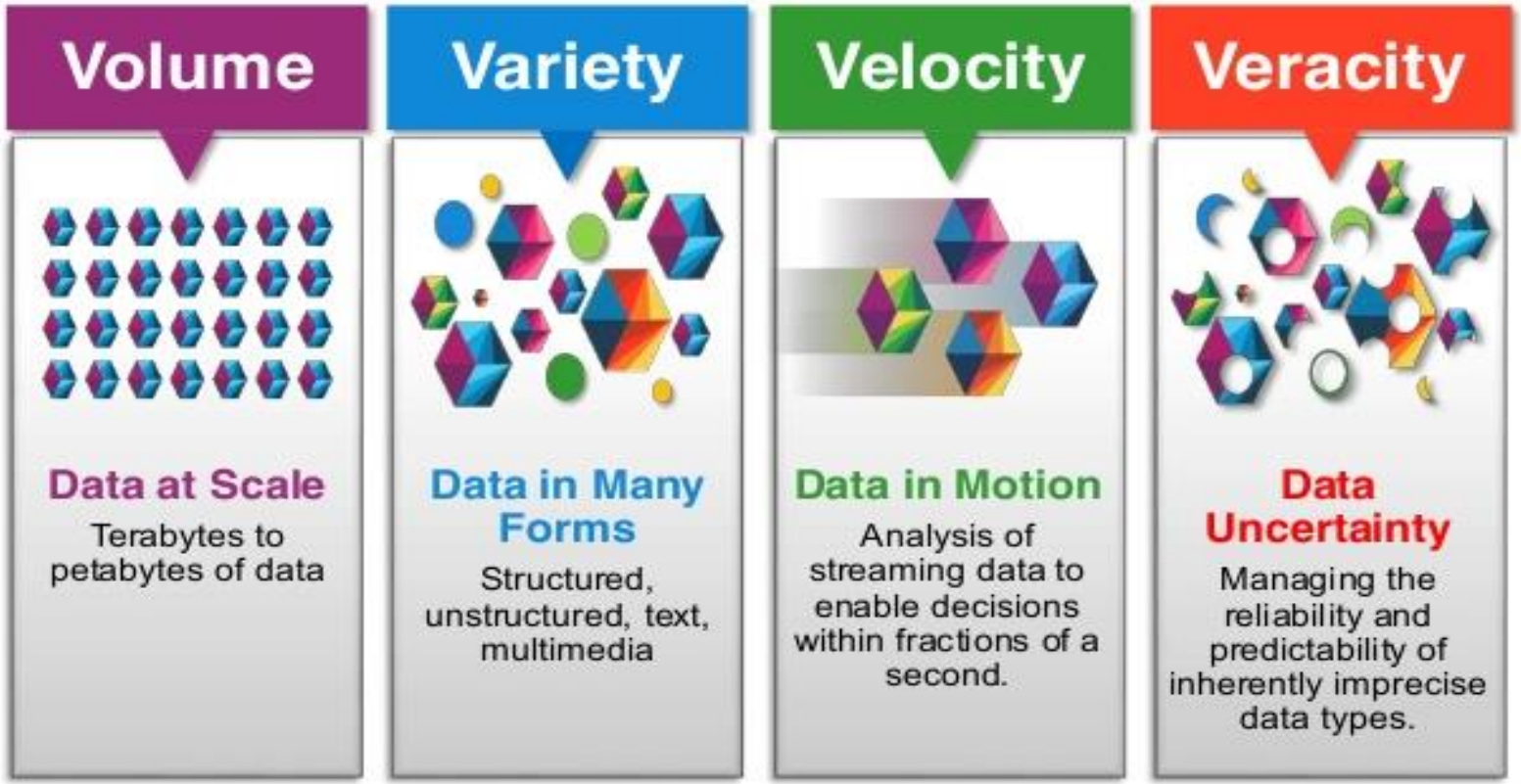
- What is Big Data?
- Big Data Concepts
- Big Data on AWS
- Data Storage on AWS
- Data Analytics & Querying on AWS
- Data Processing on AWS
- Data Flow on AWS
- Demo


Big Data

# WHAT IS BIG DATA???


This is…
Big Data.
This is…
Map
Reduce.
This is…
distributed
file system.

# 4V OF BIG DATA



# BIG DATA CONCEPTS

# BIG DATA CONCEPTS

- Scalability
- Cloud Computing
- Data Storage
- Data Analytics & Querying
- Data Processing
- Data Flow

# BIG DATA ON AWS

# AWS BIG DATA PORTOFILO

# SCALABILITY ON AWS

- Types of scalability:
  - Vertical Scalability: More/Less CPU, RAM, ...
  - Horizontal Scalability: More/Less machine

- Auto scalable by
  - CPU
  - Memory
  - Network
  - Disk
  - …

# CLOUD COMPUTING ON AWS

- Rich service catalog
- «Pay as you go» payment model
- Cloud Computing models:
  - IaaS (Infrastructure as a Service)
    - EC2, S3
  - PaaS (Platform as a Service)
    - Elastic Beanstalk
  - SaaS (Software as a Service)
    - DynamoDB, ElasticSearch

# DATA STORAGE ON AWS

# DATA STORAGE ON AWS

- There are many different data storage services for different purposes
- Key features for storage:
  - Scalability
  - Access Performance
  - Availability
  - Reliability
  - Durability
  - Security
  - Cost
- Data storage services on AWS:
  - S3
  - Glacier

# S3

- Secure, durable and highly-available storage service
- Bucket: Containers for objects
  - Globally unique regardless of the AWS region
- Object: Stored entries
  - Uniquely within a bucket by a name and a version ID
- Supports versioning
- WORM (Write once, read many)
- Consistency models
  - «read-after-write» consistency for puts of new objects
  - «eventual consistency» for puts/deletes of existing objects
  - Updates to a single object are atomic.

# GLACIER

- Storage service optimized for infrequently used data
- Archives are stored in the containers name «vault»s
- Uploading is synchronous operation with replicas
- Downloading is asynchronous operation through retrieval job
- Immutable
- Encrypted by default
- «S3» data can be archived into «Glacier» by life-cycle rules
- 4x cheaper than «S3»

# DATA ANALYTICS & QUERYING ON AWS

# DATA ANALYTICS & QUERYING ON AWS

- There are different data analytics&querying services for different purposes.
  - NoSQL
  - SQL
  - Text search
  - Analytics / Visualising


- Data analytics&querying services on AWS:
  - DynamoDB
  - Redshift
  - RDS
  - Elasticsearch (+ Kibana)
  - CloudSearch
  - QuickSight

# DYNAMODB

- Key-value or document storage based NoSQL database
- Consistency model
  - Eventual consistency by default
  - Strong consistency as optional
- Supports
  - Secondary indexes
  - Batch operations
  - Conditional writes
  - Atomic counters
- «DynamoDB Streaming» for tracking changes in near real time
- Integrated with
  - EMR
  - Elasticsearch
  - Redshift
  - Lambda

# REDSHIFT

- SQL data warehouse solution
- Fault tolerant by
  - Replicating to other nodes in the cluster
  - Backing up to «S3» continuously and automatically
- Scalable
  - Vertically by allowing switching to new cluster
  - Horizontally by adding new nodes to cluster
- Integrated with
  - S3
  - EMR
  - DynamoDB
  - Kinesis
  - Data Pipeline

# RDS

- Makes it easier to set up, operate and scale a relational database in the cloud
- Manages backups, software patching, automatic failure detection and recovery
- Supports
  - MySQL
  - PostgreSQL
  - Oracle
  - Microsoft SQL Server
  - Aurora (Amazon's MySQL based RDMS by 5x performance)

# ELASTICSEARCH

- Makes it easy to deploy, operate and scale «Elasticsearch» in the AWS cloud
- Supports
  - Taking snapshots for backup
  - Restoring from backups
  - Replicating domains across Availability Zones
- Integrated with
  - «Kibana» for data visualization
  - «Logstash» to process logs and load them into «ES»
  - «S3», «Kinesis» and «DynamoDB» for loading data into «ES»

# CLOUDSEARCH

- AWS's managed and high-performance search solution
- Reliable by replicating search domain into multiple AZ
- Automatic monitoring and recovery
- Auto scalable
- Supports popular search features such as
  - Free-text search
  - Faceted search
  - Highlighting
  - Autocomplete suggestion
  - Geospatial search
  - …

On 34 languages

# QUICKSIGHT

- AWS's new very fast, cloud-powered business intelligence (BI) service
- Uses a new, Super-fast, Parallel, In-memory Calculation Engine («SPICE») to perform advanced calculations and render visualizations rapidly
- Supports creating analyze stories and sharing them
- Integrated/Supported data sources:
  - EMR
  - Kinesis
  - S3
  - DynamoDB
  - Redshift
  - RDS
  - 3rd party (i.e. Salesforce)
  - Local file upload
  - …
- Said that 1/10th the cost of traditional BI solutions
- Native Access on available major mobile platforms

# DATA PROCESSING ON AWS

# DATA PROCESSING ON AWS

- Allows processing of huge amount of data in highly scalable way as distributed
- Support both of
  - Batch
  - Stream

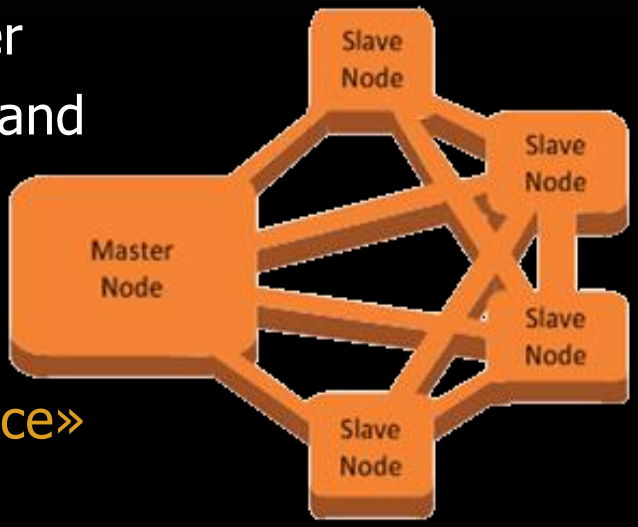
  data processing concepts
- Data processing services on AWS:
  - EMR (Elastic Map-Reduce)
  - Kinesis
  - Lambda
  - Machine Learning

# EMR (ELASTIC MAPREDUCE)

- Big data service/infrastructure to process large amounts of data efficiently in highly-scalable manner
- Supported Hadoop distributions:
  - Amazon
  - MapR
- Low cost by «on-demand», «spot» or «reserved» instances
- Reliable by retrying failed tasks and automatically replacing poorly performing or failed instances
- Elastic by resizing cluster on the fly
- Integrated with
  - S3
  - DynamoDB
  - Data Pipeline
  - Redshift
  - RDS
  - Glacier
  - VPC
  - …

# EMR – CLUSTER COMPONENTS

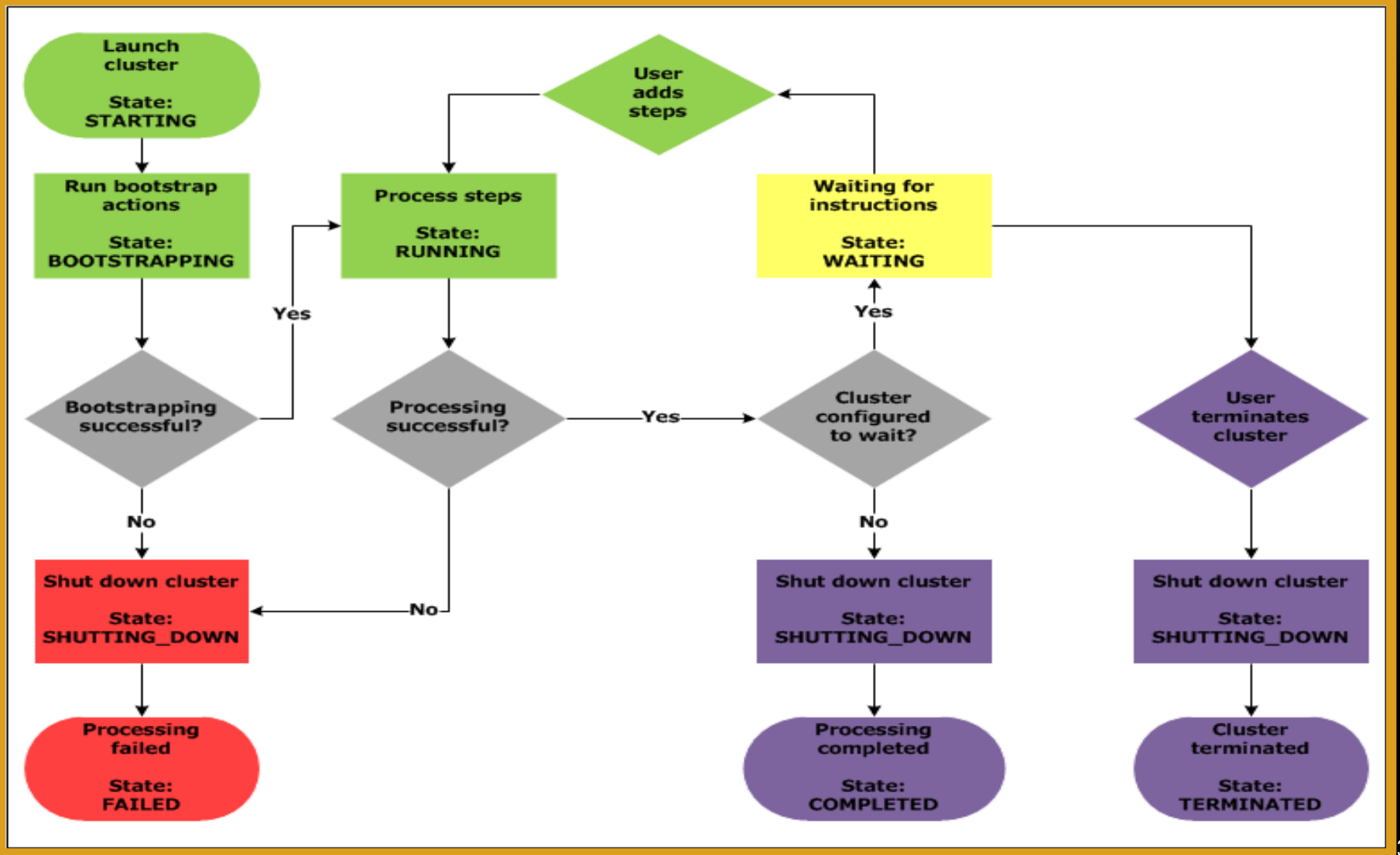
- Master Node
  - Tracks, monitors and manages the cluster
  - Runs the «YARN Resource Manager» and the «HDFS Name Node Service»
- Core Node [Slave Node]
  - Runs tasks and stores data in the HDFS
  - Runs the «YARN Node Manager Service» and «HDFS Data Node Daemon» service
- Task Node [Slave Node]
  - Only runs the tasks
  - Runs the «YARN Node Manager Service»



# EMR – BEST PRACTICES FOR INSTANCE GROUPS

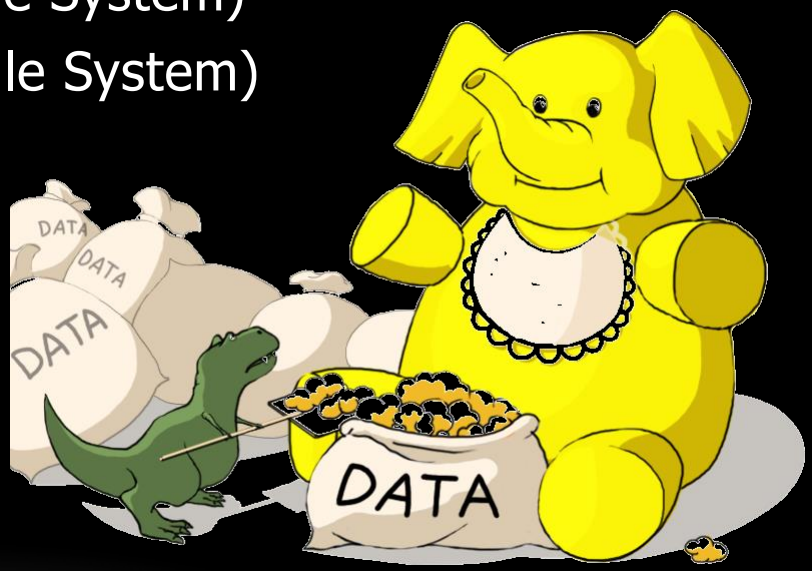
| Project | Master Instance Group | Core Instance Group | Task Instance Group |
|---|---|---|---|
| Long-running clusters | On-Demand | On-Demand | Spot |
| Cost-driven workloads | Spot | Spot | Spot |
| Data-critical workloads | On-Demand | On-Demand | Spot |
| Application testing | Spot | Spot | Spot |

# EMR – CLUSTER LIFECYCLE



# EMR - STORAGE

- The storage layer includes the different file systems for different purposes as follows:
  - HDFS (Hadoop Distributed File System)
  - EMRFS (Elastic MapReduce File System)
  - Local Storage



# EMR - HDFS

- Distributed, scalable file system for Hadoop
- Stores multiple copies to prevent data lost on failure
- It is an ephemeral storage because it is reclaimed when cluster terminates
- Used by the master and core nodes
- Useful for intermediate results during Map-Reduce processing or for workloads which have significant random I/O
- Prefix: «hdfs://»

# EMR - EMRFS

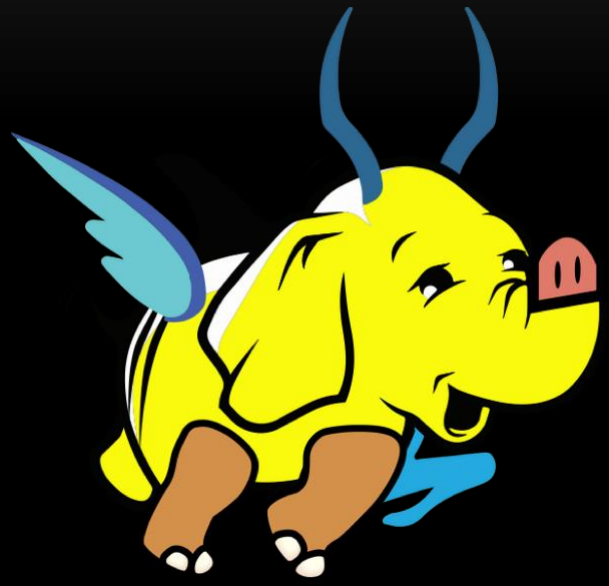
- An implementation of HDFS which allows clusters to store data on «AWS S3»
- Eventually consistent by default on top of «AWS S3»
- Consistent view can be enabled with the following supports:
  - Read-after-write consistency
  - Delete consistency
  - List consistency
- Client/Server side encryption
- Most often, «EMRFS» (S3) is used to store input and output data and intermediate results are stored in «HDFS»
- Prefix: «s3://»

# EMR – LOCAL STORAGE

- Refers to an ephemeral volume directly attached to an EC2 instance
- Ideal for storing temporary data that is continually changing, such as
  - Buffers
  - Caches
  - Scratch data
  - And other temporary content
- Prefix: «file://»

# EMR – DATA PROCESSING FRAMEWORKS

- Hadoop
- Spark
- Hive
- Pig
- HBase
- HCatalog
- Zookeeper
- Presto
- Impala
- Sqoop
- Oozie
- Mahout
- Phoenix
- Tez
- Ganglia
- Hue
- Zeppelin
- …

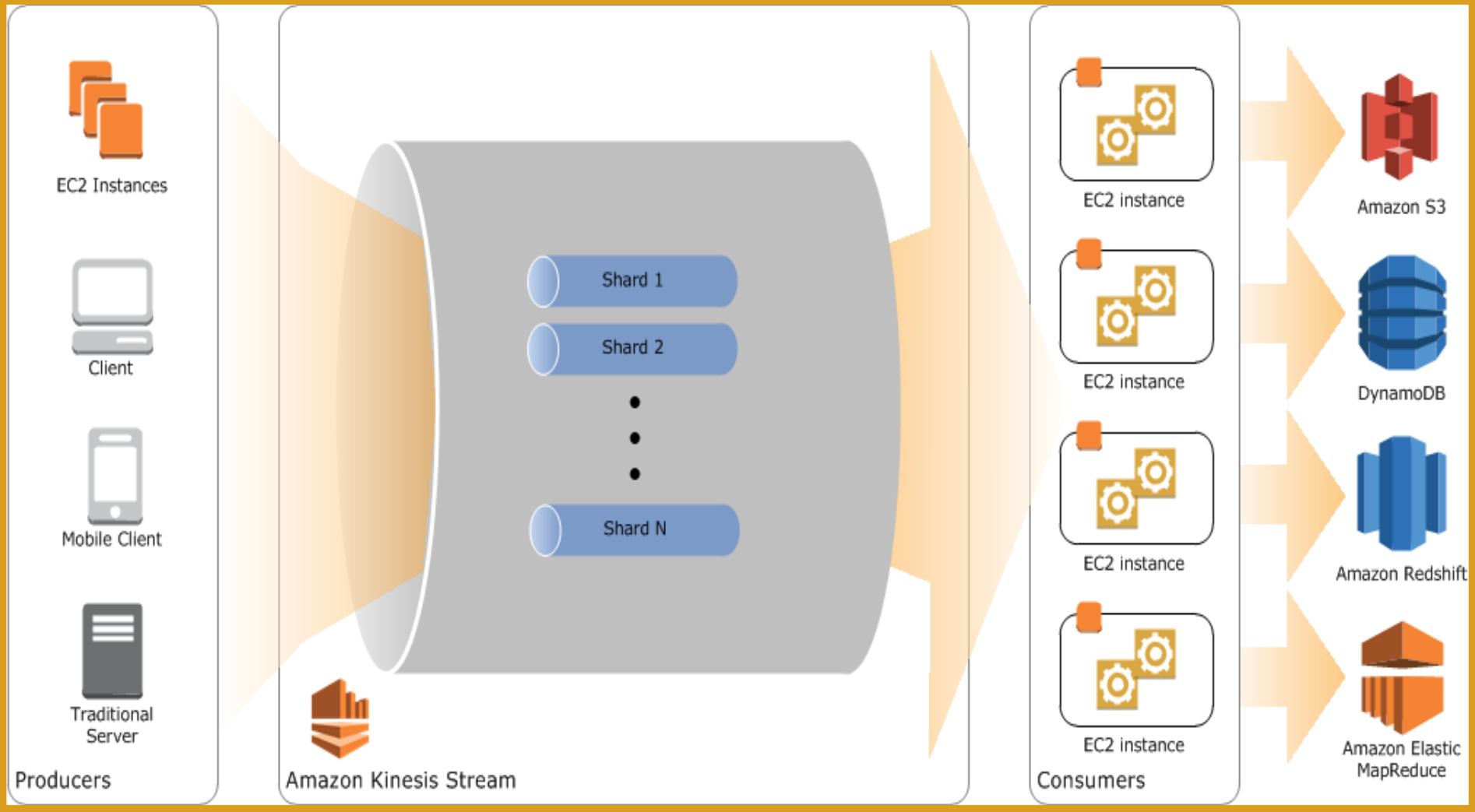


# KINESIS

- Collects and processes large streams of data records in real time
- Reliable by synchronously replicating streaming data across facilities in an AWS Region
- Elastic by increasing shards based on the volume of input data
- Has
  - S3
  - DynamoDB
  - ElasticSearch

  connectors
- Also integrated with «Storm» and «Spark»

# KINESIS – HIGH LEVEL ARCHITECTURE



# KINESIS – KEY CONCEPTS[1]

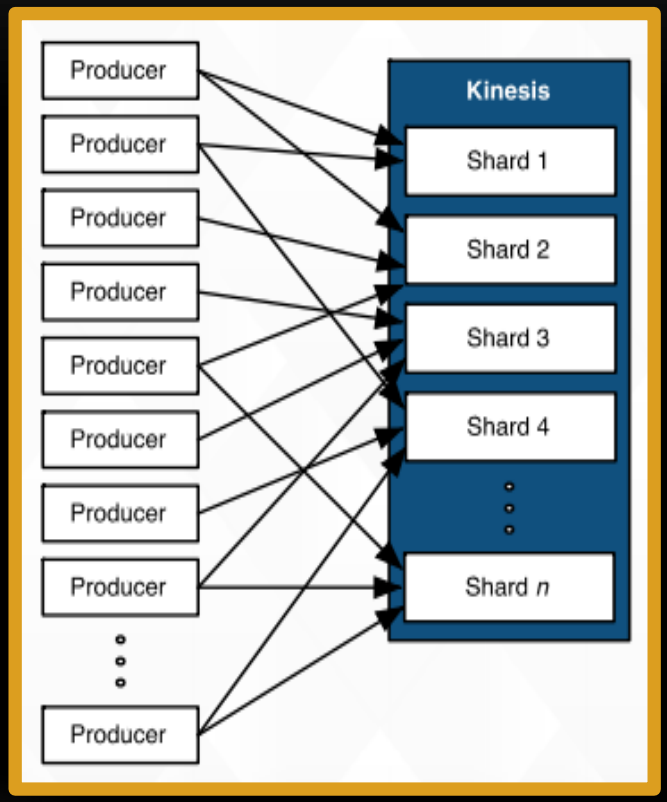
- Streams
  - Ordered sequence of data records
  - All data is stored for 24 hours (can be increased to 7 days)
- Shards
  - It is a uniquely identified group of data records in a stream
  - Kinesis streams are scaled by adding or removing shards
  - Provides 1MB/sec data input and 2MB/sec data output capacity
  - Supports up to 1000 TPS (put records per second)
- Producers
  - Put records into streams
- Consumers
  - Get records from streams and process them

# KINESIS – KEY CONCEPTS[2]

- Partition Key
  - Used to group data by shard within a stream
  - Used to determine which shard a given data record belongs to
- Sequence Number
  - Each record has a unique sequence number in the owned shard
  - Increase over time
  - Specific to a shard within a stream, not across all shards
- Record
  - Unit of data stored
  - Composed of <partition-key, sequence-number, data-blob>
  - Data size can be up to 1 MB
  - Immutable

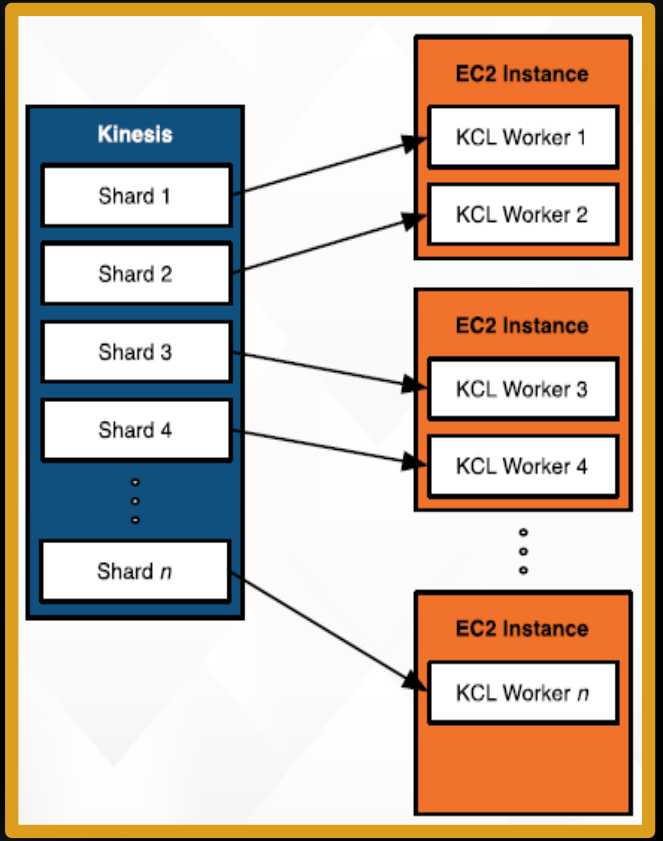
# KINESIS – GETTING DATA IN

- Producers use «PUT» call to store data in a stream
- A partition key is used to distribute the «PUT»s across shards
- A unique sequence # is returned to the producer for each event
- Data can be ingested at 1MB/second or 1000 Transactions/second per shard



# KINESIS – GETTING DATA OUT

- Kinesis Client Library (KCL) simplifies reading from the stream by abstracting you from individual shards
- Automatically starts a worker thread for each shard
- Increases and decreases thread count as number of shards changes
- Uses checkpoints to keep track of a thread's location in the stream
- Restarts threads & workers if they fail



# LAMBDA

- Compute service that run your uploaded code on your behalf using AWS infrastructure
- Stateless
- Auto-scalable
- Pay per use
- Use cases
  - As an event-driven compute service triggered by actions on «S3», «Kinesis», «DynamoDB», «SNS», «CloudWatch»
  - As a compute service to run your code in response to HTTP requests using «AWS API Gateway»
- Supports «Java», «Python» and «Node.js» languages

# MACHINE LEARNING

- Machine learning service for building ML models and generating predictions
- Provides implementations of common ML data transformations
- Supports industry-standard ML algorithms such as
  - binary classification
  - multi-class classification
  - regression
- Supports batch and real-time predictions
- Integrated with
  - S3
  - RDS
  - Redshift

# DATA FLOW ON AWS

# DATA FLOW ON AWS

- Allows transferring data between data storage/processing points

- Data flow services on AWS:
  - Firehose
  - Data Pipeline
  - DMS (Database Migration Service)
  - Snowball

# FIREHOSE

- Makes it easy to capture and load massive volumes of load streaming data into AWS
- Supports endpoints
  - S3
  - Redshift
  - Elasticsearch
- Supports buffering
  - By size: Ranges from 1 MB to 128 MB. Default 5 MB
  - By interval: Ranges from 60 seconds to 900 seconds. Default 5 minutes
- Supports encryption and compression

# DATA PIPELINE

- Automates the movement and transformation of data
- Comprise of workflows of tasks
- The flow can be scheduled
- Has visual designer with drag-and-drop support
- The definition can be exported/imported
- Supports custom activities and preconditions
- Retry flow in case of failure and notifies
- Integrated with
  - EMR
  - S3
  - DynamoDB
  - Redshift
  - RDS

# DATABASE MIGRATION SERVICE

- Helps to migrate databases into AWS easily and securely
- Supports
  - Homogeneous migrations (i.e. Oracle <=> Oracle)
  - Heterogeneous migrations (i.e. Oracle <=> PostgreSQL)
- Also used for:
  - Database consolidation
  - Continuous data replication
- Keep your applications running during migration
  - Start a replication instance
  - Connect source and target databases
  - Let «DMS» load data and keep them in sync
  - Switch applications over to the target at your convenience

# SNOWBALL

- Import/Export service that accelerates transferring large amounts of data into and out of AWS using physical storage appliances, bypassing the Internet
- Data is transported securely as encrypted
- Data is transferred via 10G connection
- Has 50TB or 80TB storage options
- Steps:
  - Job Create
  - Processing
  - In transit to you
  - Delivered to you
  - In transit to AWS
  - At AWS
  - Importing
  - Completed

# DEMO TIME

Source code
and
Instructions
are available at
github.com/serkan-ozal/ankaracloudmeetup-bigdata-demo

# STEPS OF DEMO

- Storing
- Searching
- Batch Processing
- Stream Processing
- Analyzing

# STORING

# [STORING]
# CREATING DELIVERY STREAM

- Create an «AWS S3» bucket to store tweets
- Create an «AWS Firehose» stream to push tweets into to be stored
- Attach created bucket as endpoint to the stream

Then;

- Listened tweet data can be pushed into this stream to be stored

- Instructions:

  **github.com/serkan-ozal/ankaracloudmeetup-bigdata-demo/tree/master/bigdata-demo-storing-firehose**

# [STORING] CRAWLING TWEETS

- Configure stream name to be used by crawler application
- Build «bigdata-demo-storing-crawler» application
- Deploy into «AWS Elasticbeanstalk»

Then;

- Application listens tweets via «Twitter Streaming API»
- Application pushes tweets data into «AWS Firehose» stream
- «AWS Firehose» stream stores tweets on «AWS S3» bucket

- Source code and instructions:

  **github.com/serkan-ozal/ankaracloudmeetup-bigdata-demo/tree/master/bigdata-demo-storing-crawler**

# SEARCHING

# [SEARCHING] CREATING SEARCH DOMAIN

- Create an «AWS Elasticsearch» domain
- Wait until it becomes active and ready to use

Then

- Tweet data can also be indexed to be searched

- Instructions:

  **github.com/serkan-ozal/ankaracloudmeetup-bigdata-demo/tree/master/bigdata-demo-searching-elasticsearch**

# [SEARCHING] INDEXING INTO SEARCH DOMAIN

- Create an «AWS Lambda» function with the given code
- Configure created «AWS S3» bucket as trigger
- Wait until it becomes active and ready to use

Then;

- Stored tweet data will also be indexed to be searched

- Source code and instructions:

  **github.com/serkan-ozal/ankaracloudmeetup-bigdata-demo/tree/master/bigdata-demo-searching-lambda**

# BATCH PROCESSING

# [BATCH PROCESSING] CREATING EMR CLUSTER

- Create an «AWS EMR» cluster
- Wait until it becomes active and ready to use

Then;

- Stored tweet data can be processed as batch

- Instructions:

  **github.com/serkan-ozal/ankaracloudmeetup-bigdata-demo/tree/master/bigdata-demo-batchprocessing-emr**

# [BATCH PROCESSING] CREATING TABLE TO STORE BATCH RESULT

- Create an «AWS DynamoDB» table to save batch results
- Wait until it becomes active and ready to use

Then;

- Analyze result data can be saved into this table

- Instructions:

  **github.com/serkan-ozal/ankaracloudmeetup-bigdata-demo/tree/master/bigdata-demo-batchprocessing-dynamodb**

# [BATCH PROCESSING] PROCESSING TWEETS VIA HADOOP

- Build&upload «bigdata-demo-batchprocessing-hadoop» to «AWS S3»
- Run uploaded jar as «Custom JAR» step on «AWS EMR» cluster
- Wait until step has completed

Then;

- Processed tweet data is on «AWS S3» bucket
- Analyze result is in «AWS DynamoDB» table

- Instructions:

  **github.com/serkan-ozal/ankaracloudmeetup-bigdata-demo/tree/master/bigdata-demo-batchprocessing-hadoop**

# [BATCH PROCESSING] QUERYING TWEETS VIA HIVE

- Upload Hive query file to «AWS S3»
- Run uploaded query as «Hive» step on «AWS EMR» cluster
- Wait until step has completed

Then;

- Query results are dumped to «AWS S3»

- Instructions:

  **github.com/serkan-ozal/ankaracloudmeetup-bigdata-demo/tree/master/bigdata-demo-batchprocessing-hive**

# [BATCH PROCESSING] SCHEDULING WITH DATA PIPILINE

- Import «AWS Data Pipeline» definition
- Activate pipeline

Then;

- Every day, all batch processing jobs (Hadoop +Hive) will be executed automatically

- Instructions:

  **github.com/serkan-ozal/ankaracloudmeetup-bigdata-demo/tree/master/bigdata-demo-batchprocessing-datapipeline**

# STREAM PROCESSING

# [STREAM PROCESSING] CREATING REALTIME STREAM

- Create an «AWS Kinesis» stream to push tweets to be analyzed
- Wait until it becomes active and ready to use

Then;

- Listened tweet data can be pushed into this stream to be analyzed

- Instructions:

  **github.com/serkan-ozal/ankaracloudmeetup-bigdata-demo/tree/master/bigdata-demo-streamprocessing-kinesis**

# [STREAM PROCESSING] PRODUCING STREAM DATA

- Configure stream name to be used by producer application
- Build «bigdata-demo-streamprocessing-kinesis-producer»
- Deploy into «AWS Elasticbeanstalk»

Then;

- Application listens tweets via «Twitter Streaming API»
- Listened tweets are pushed into «AWS Kinesis» stream

- Source code and instructions:

  **github.com/serkan-ozal/ankaracloudmeetup-bigdata-demo/tree/master/bigdata-demo-streamprocessing-kinesis-producer**

# [STREAM PROCESSING] CREATING TABLE TO SAVE STREAM RESULT

- Create an «AWS DynamoDB» table to save realtime results
- Wait until it becomes active and ready to use

Then;

- Analyze result can be saved into this table

- Instructions:

  **github.com/serkan-ozal/ankaracloudmeetup-bigdata-demo/tree/master/bigdata-demo-streamprocessing-dynamodb**

# [STREAM PROCESSING] CONSUMING STREAM DATA

- Configure
  - «AWS Kinesis» stream name for consuming tweets from
  - «AWS DynamoDB» table name for saving results into
- Build «bigdata-demo-streamprocessing-consumer» application
- Deploy into «AWS Elasticbeanstalk»

Then;

- Application consumes tweets from «AWS Kinesis» stream
- Sentiment analyze is applied to tweets via «Stanford NLP»
- Analyze results are saved into «AWS DynamoDB» table in realtime

- Source code and instructions:

  **github.com/serkan-ozal/ankaracloudmeetup-bigdata-demo/tree/master/bigdata-demo-streamprocessing-kinesis-consumer**

# ANALYZING

# [ANALYZING] ANALYTICS VIA QUICKSIGHT

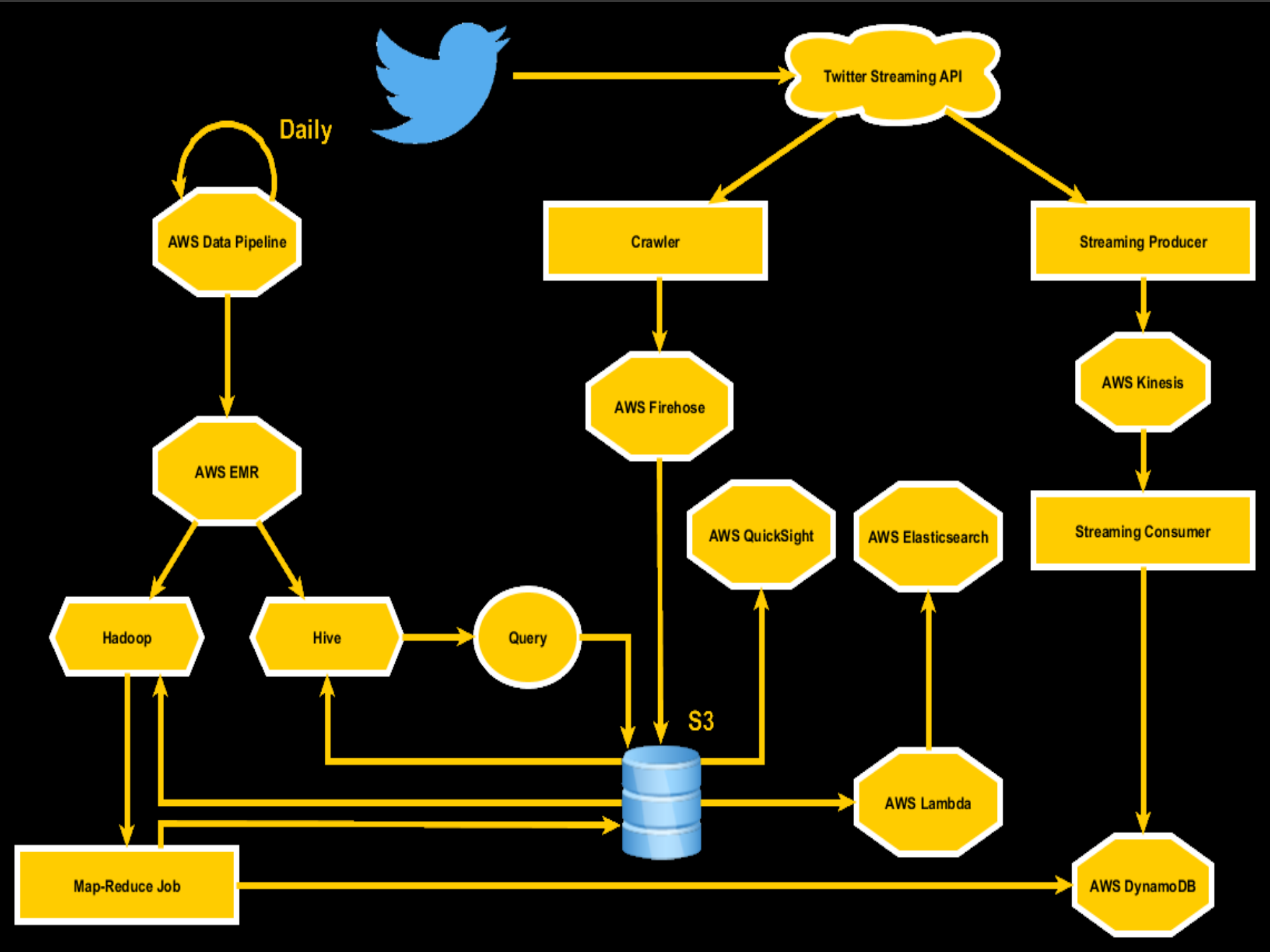
- Create a data source from «AWS S3»
- Create an analyse based on data source from «AWS S3»
- Wait until importing data and analyze has completed

Then;

- You can do your
  - Query
  - Filter
  - Visualization
  - Story (flow of visualisations like presentation)

  stuff on your dataset and share them

# THE BIG PICTURE


Twitter Streaming API
Daily
AWS Data Pipeline
Crawler
Streaming Producer
AWS Firehose
AWS Kinesis
AWS EMR
AWS QuickSight
AWS Elasticsearch
Streaming Consumer
Hadoop
Hive
Query
S3
AWS Lambda
Map-Reduce Job
AWS DynamoDB

# END OF DEMO

# THANKS!!!