1. Data Engineering - 20%
2. Exploratory Data Analysis - 24%
3. Modeling - 36%
4. ML Implementations and Operations - 20%

**1. Data Engineering**

**1.1. S3**

* S3 allows people to store objects (files) in buckets (directories).
* Buckets must have a globally unique name.
* Objects have keys (file path): buckets/objects. Max size: 5TB. Support any data format.
* Object Tags: key/value pairs. Useful for security.

**1.1.2. S3 Data Partitioning**

* Useful for range queries.
* Service eg: Kinesis.

**1.1.3. S3 Storage Classes**



**1.1.4. S3 Durability and Availability**

* Durability: How many times object is going to be lost by Amazon S3. Durability is 99.99999999999% across all S3 services.
* Availability: How readily available service is. Different S3 classes have different availability.

**1.1.5. S3 Standard**

* Frequently accessed data. Availability 99.99%, low latency, high throughput.
* Use cases: Big data analytics, mobile & gaming applications, content distribution.

**1.1.6. S3 Infrequent Access**

* Less frequently accessed data, when required rapidly accessed. Lower cost than S3 standard. 99.9% availability.
* Use cases: Disaster recovery, Backups.

**1.1.7. S3 One Zone - IA**

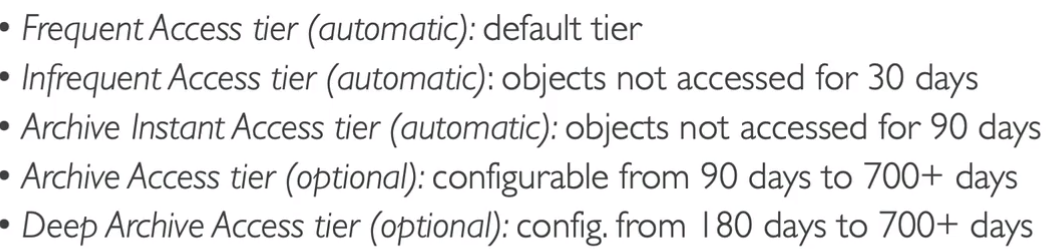
* Availability 99.5%. 1 Availability Zone.
* Use cases: Bring secondary copies of on-premises data, recreating data.

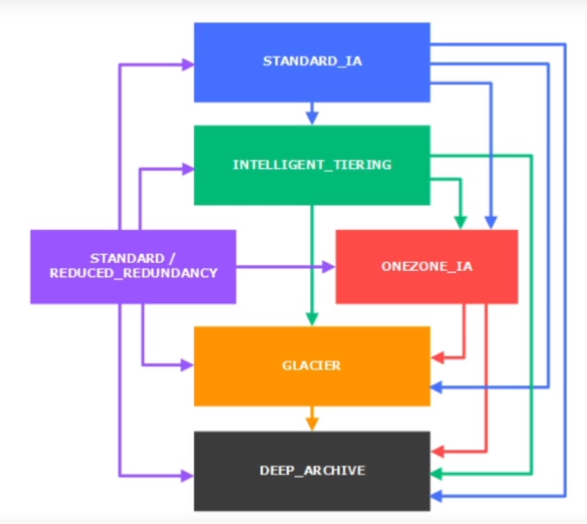
**1.1.8. S3 Glacier**

* Low cost. Archiving/Backup. Price for storage + Object retrieval cost.
* Amazon S3 Glacier Instant retrieval - Retrieval in milliseconds.Min Storage duration 90 Days.
* Amazon S3 Glacier Flexible retrieval - Retrieval in Expedited (1-5 mins), Standard (3-5 hrs), Bulk (5-12 hrs) - Free. Min Storage duration 90 Days.
* Amazon S3 Glacier Deep archive - Retrieval in Standard (12 hrs), Bulk (48 hrs). Long-term Storage duration min 180 Days.

**1.1.9. S3 Automatic Tiering**

Cost for only monthly monitoring and auto-tiering access tiers.





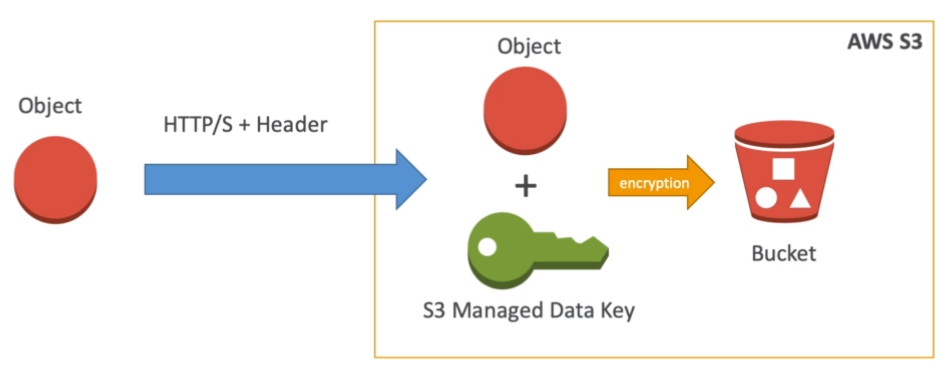
**1.1.10. S3 Lifecycle rules**

* Automatic rules to be applies on objects in a bucket.
* Transition actions - Move objects from 1 storage class to another, Expiration actions - Delete objects as per condition (in days, old versions, incomplete data upload), Rules as per prefixes such as mp3 directory objects, rules per object tags (eg: objects with tag Department: Finance).

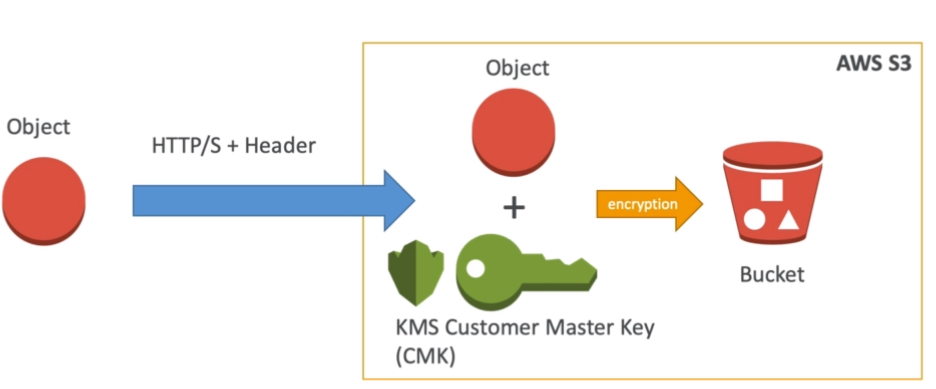
**1.1.11. S3 Encryption for Objects**

* SSE-S3 - using keys handled and managed by AWS.

SSE (Server Side Encryption)



* SSE-KMS - Key Management Service to manage encryption keys. (Access control, audit trail).



* SSE-C - Manage own encryption keys.
* Client-side encryption.

**1.1.12. S3 Security**

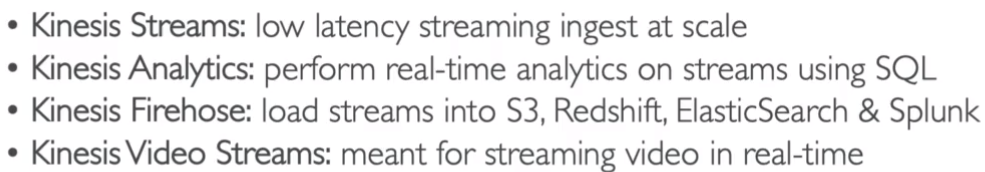
* User Based - IAM (Identity Access Management) which user have access to which API.
* Resource Based - Bucket Policy, Object Access Control List, Bucket Access Control List.
* S3 Default Encryption - Encrypt every object sent to the bucket.
* Networking - Avoid public access of traffic through VPC (Virtual Private Cloud) Endpoint gateway.
* Logging and auditing of access. (eg. CloudTrail)
* Tagged based + IAM policy + Bucket policy. (eg. Allow IAM user acces to tag object classification = PHI)

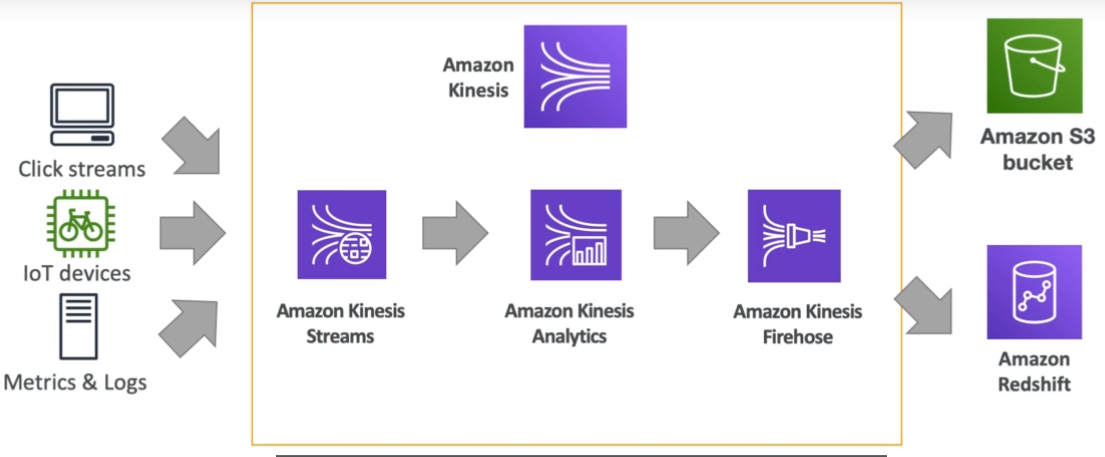
**1.1.13. S3 Bucket Policies**

* JSON-based policies with key Resources - buckets/objects, Actions - set of APIs to allow/deny, Principal - account/user to apply the policy.
* Use cases: grand public access, force encryption of the object at upload, grant access to another account.

**1.2. Kinesis**

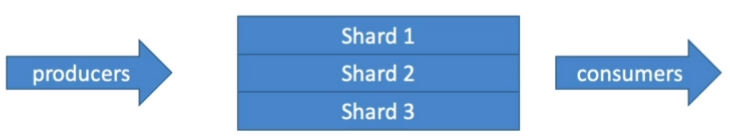
* Alternative to Apache Kafka, used for logs, metrics, IoT device information, clickstreams, real-time Big Data, and stream processing frameworks (Spark, NiFi, etc).
* Replication to 3 AZ automatically.

with video analytics.



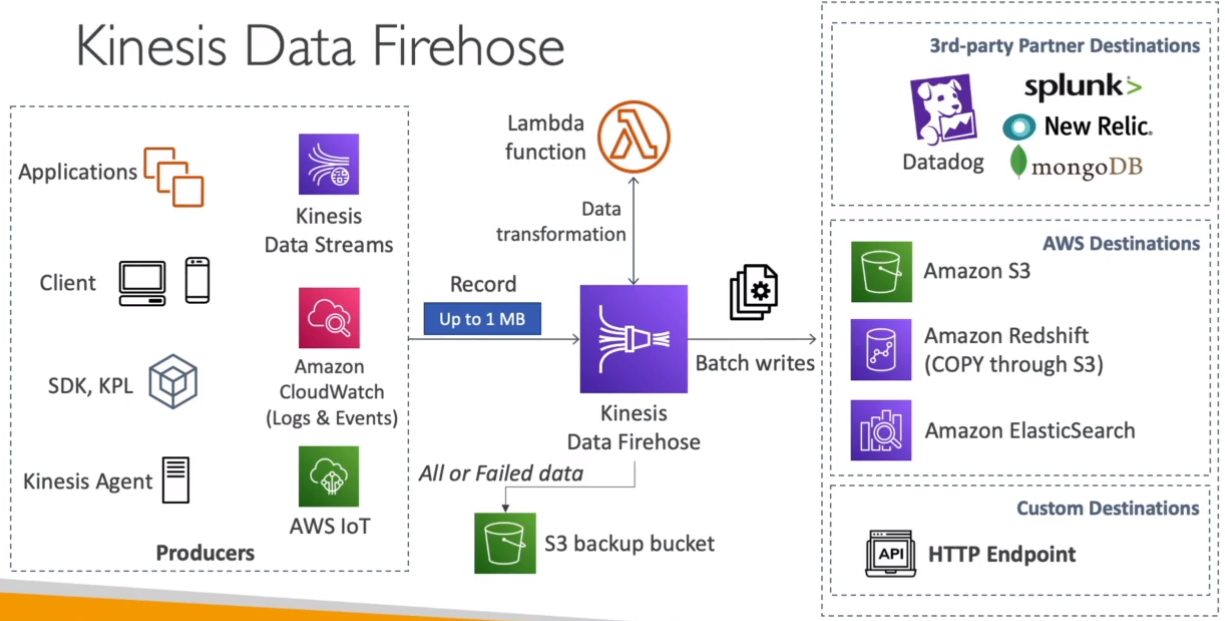
**1.2.1. Kinesis Streams**

Real-time Streams (200 ms classic, 70 ms enhanced latency) are divided into ordered shards or partitions. Custom code for consumer/producer supported.

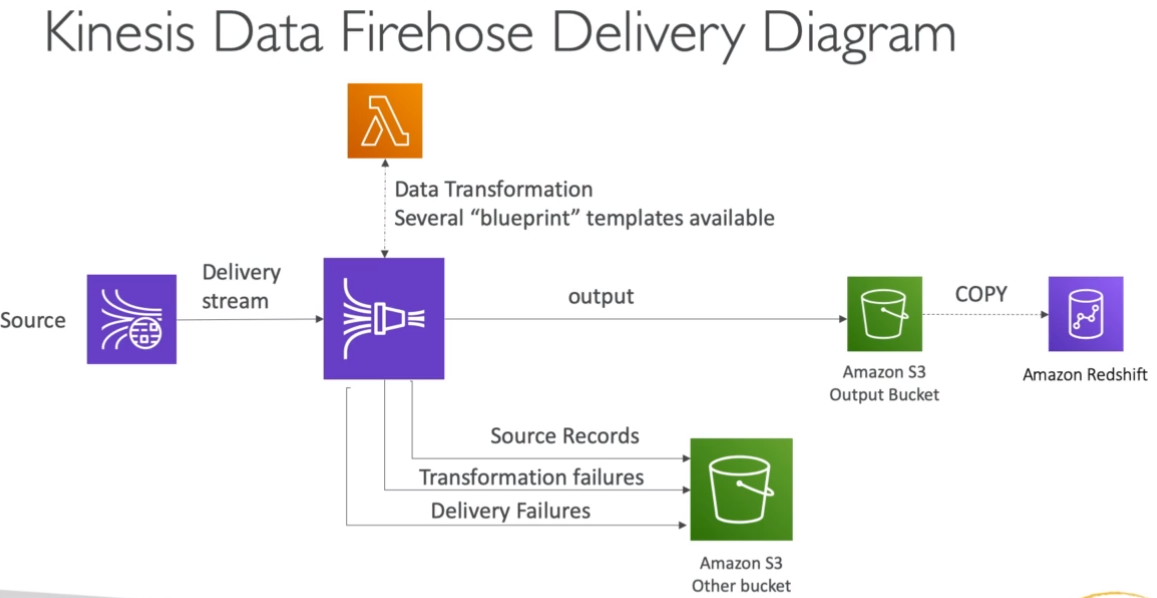


* Retention in 24 hrs default, up to 365 days with the ability to replay/reprocess the data, therefore multiple applications can consume the same stream at that time being.
* Data insertions are immutable (no deletion).
* Used for a small amount of data up to 1MB.
* Capacity Modes:
  + Provisioned - The number of shards to be provisioned is chosen, with manual or API-based scaling. Each shard gets 1MB/s or 1000 records/s IN throughput, and 2MB/s OUT throughput. Pay per shards provisioned/hr.
  + On-demand - No capacity management of shards. Default capacity provisioned (4MB/s). Automatic scaling based on observed throughput peak during 30 days. Pay/stream per hour & data in/out per GB.
* Limit: Producer - 1MB/s or 1000 msgs/s at write per shard, Consumer classic - 2 MB/s per at read per shard & 5 API calls/s per shard.

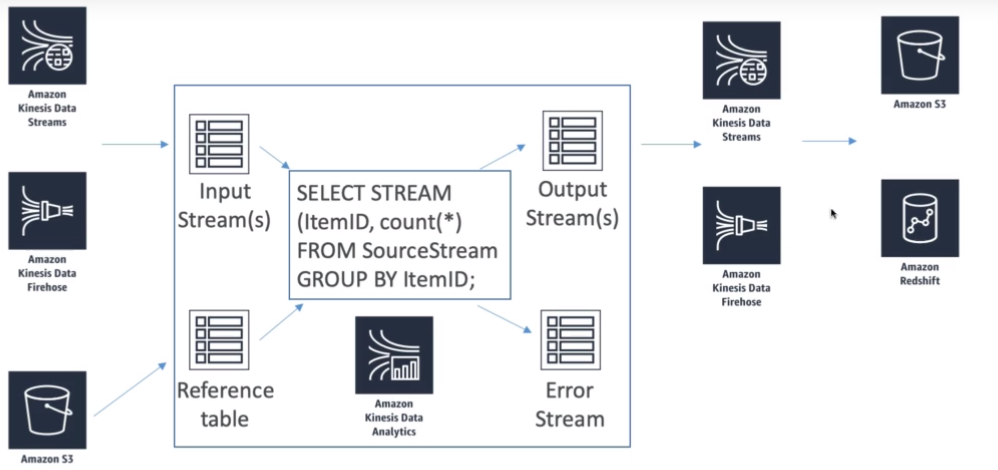
**1.2.2. Kinesis Data Firehose**

****

* Fully managed delivery/ingestion service for massive near real-time data streams (60 secs latency for a nonfull batch) with auto-scaling.
* Data ingestion into Redshift (data warehouse), Amazon S3, Elastic Search (index-based), Splunk (3rd party).
* Supports many data formats, data conversion to CSV/Parquet/JSON/ORC (only S3), transformation using AWS Lambda (eg: CSV=>JSON), and compression when the target is S3 (GZiP, ZiP, Snappy).
* No data storage for replay capability.
* Pay for the amount of data going through Firehose.



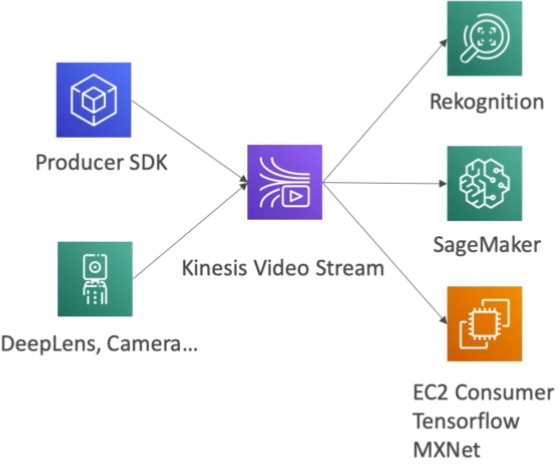
**1.2.3. Kinesis Analytics (Real-time analytics)**

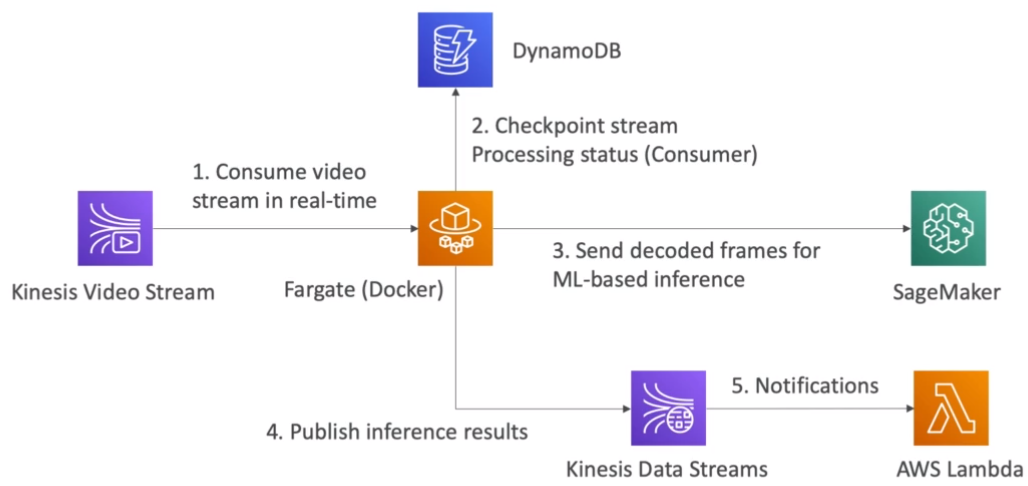


* Use cases: Streaming ETL (Extract, Transform, Load) - select columns, simple transformation on streams, Continuous metric generation - Live leaderboard of a mobile game, Responsive analytics - find the unusual patterns for certain criteria and build alerts for the same.
* Serverless service with auto-scale, pay only for resources consumed, using IAM permission for access control on the source(s) & destination(s), SQL(eg: for Kinesis firehose)/Flink for computations, Lambda for pre-processing, discover schema features.
* For machine learning, we can apply 2 models directly: Random Cut Forest (Everchanging model) - Used for anomaly detection based on recent history (Eg: detect anomalous subway ridership during NYC marathon), Hotspots - locate and return info about relatively dense regions in data (Eg: a collection of overheated servers in data center).

**1.2.4. Kinesis Video Streams**

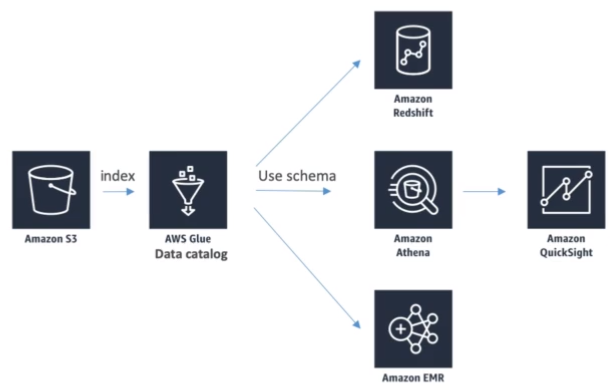
* Producers can be security cameras, body-worn cameras, AWS Deeplens, smartphone cameras, audio feeds, images, RADAR data, and RTSP cameras.
* Consumers can be custom (MXNet, Tensorflow), AWS Sagemaker, Amazon Rekognition video, etc.
* One producer per video stream.
* Video playback capability.
* Retention of 1 hr to 10 yrs.





**1.3. Glue Data Catalog**

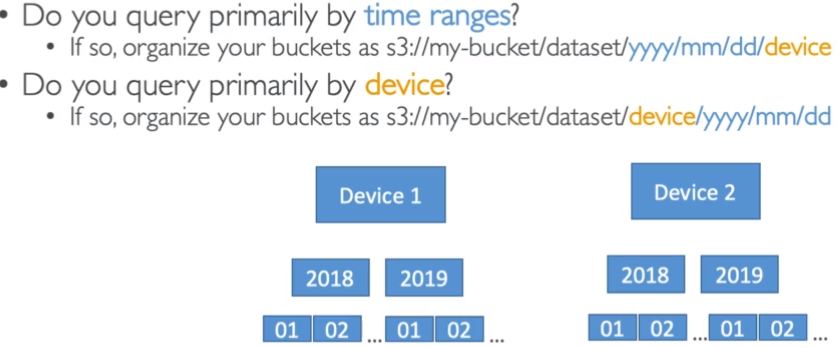
* Metadata repository for all the tables with automated schema inference and schema versioning.
* Can integrate with Athena and Redshift spectrum for schema and data discovery.
* Glue crawlers help in building Glue Data Catalog.



**1.3.1. Glue Crawlers**

* Crawlers go through the data to infer schemas and partitions(Eg: from S3 based on how your data is organized).

Example: Devices send sensor data hourly.



* They can work with different data formats such as CSV, JSON, Parquet, relational store, and different data sources such as S3, Amazon Redshift, and Amazon RDS.
* The crawler can run on schedule or On Demand.
* IAM Role access/credentials are required.

**1.3.2. Glue ETL (Extract, Transform, Load)**

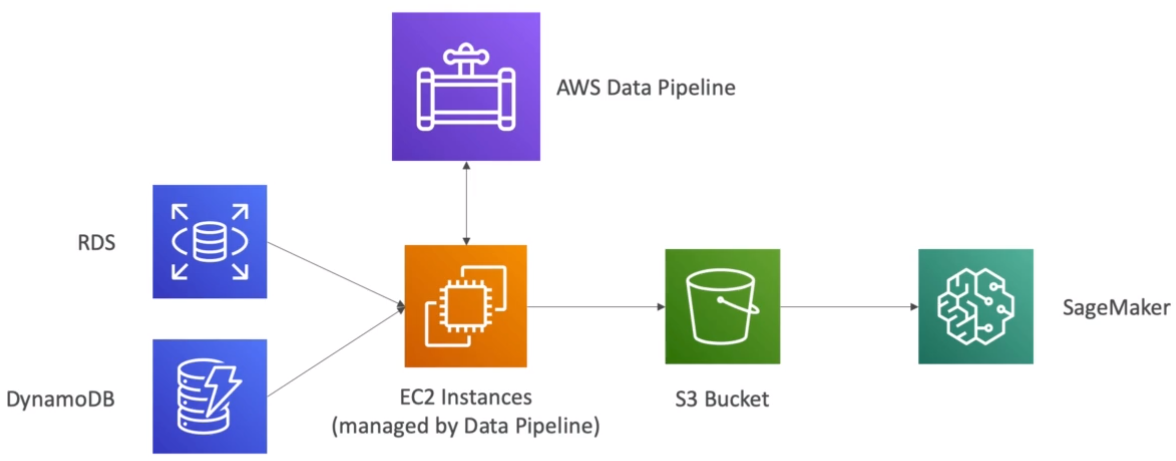
* Fully managed, cost-effective, pay only for resources consumed service that allows transformation, cleaning, and enriching data.
* Can either generate ETL code in Scala/Python or provide your own Spark/PySpark scripts.
* Resources belong to AWS so we might not have an easy access to some of the resources.
* Target can be S3, JDBC (RDS, Redshift), or Glue Data catalog.
* Can run spark jobs on serverless spark platform, Glue Scheduler can be used to schedule the jobs, and Glue Triggers to run jobs based on events.
* Transformation such as Bundled (drop fields/null fields, filter records using the function, join data, map to add/delete/lookup fields), ML (FindML matches for approximate records even if they do not have a common unique ID or are exactly same), format conversions (JSON, CSV, Parquet, ORC, Avro, XML), Spark (eg: KMeans algorithm).

**1.4. AWS ML Data stores**

* Redshift - Big Data database needed to be provisioned in advance.
  + Data warehouse technology with SQL analytics (OLAP - Online analytical processing) feature.
  + Used for running SQL queries on a massive amount of data in parallel, and do some analytics on them.
  + By default, we have to load data from S3 to Redshift, but we can use the Redshift Spectrum to run SQL queries directly on S3.
  + Columnar-based - Data is organized in columns.
* RDS, Aurora -
  + Relation store with SQL, OLTP (Online Transactional Processing) feature.
  + Row-based - Data is organized in rows.
  + Used when we want to store data regarding the model for exports.
  + Provisioning servers in advance is a must.
* DynamoDB -
  + NoSQL (Not Only SQL) data store.
  + Provisioning read/write capacity is a must.
  + Serverless.
  + Useful to store machine learning models (eg. model output) served by applications.
* S3
* OpenSearch (ElasticSearch) -
  + Indexing of data.
  + Search amongst data points.
  + Clickstream analytics.
  + No machine learning direct integration.
* Elastic cache - Used for caching in case there is a frequent requirement of certain data.

**1.5. AWS Data pipeline**

* Data pipeline is an ETL service to move data from one place to another with the capability to do retries and send notifications for failures.
* Orchestrator that manages task dependencies. (ETL happens in EC2/EMR instances)
* Destinations can include S3, RDS, DynamoDB, Redshift, and EMR.
* Sources may be on-premises.
* High availability.
* Flexibility to manage the environment, resources required to run the code, and the code itself.
* Example -

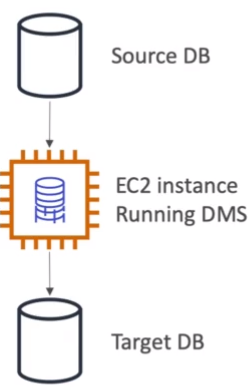


**1.6. AWS Batch**

* Serverless service (no need to manage clusters) that runs any computing and non-ETL related batch jobs as Docker images.
* Do not need to provision EC2 or Spot instances.
* Figures out the optimal quantity and type of instances based on volumes of jobs and requirements automatically.
* Pay for underlying instances created.
* Schedule batch jobs using Cloudwatch events, or orchestrate batch jobs using the Step function.

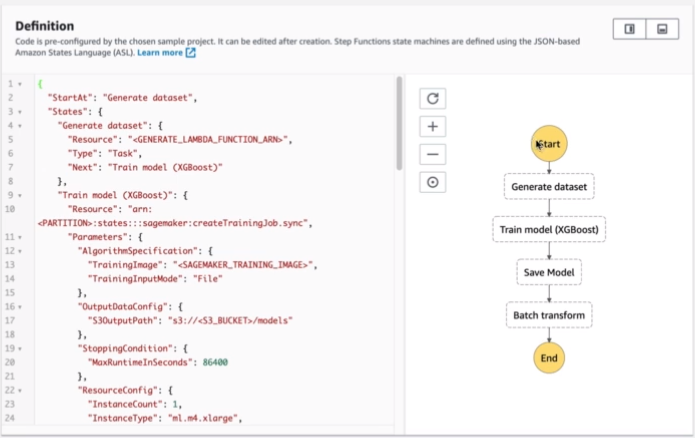
**1.7. AWS DMS (Database Migration Service)**

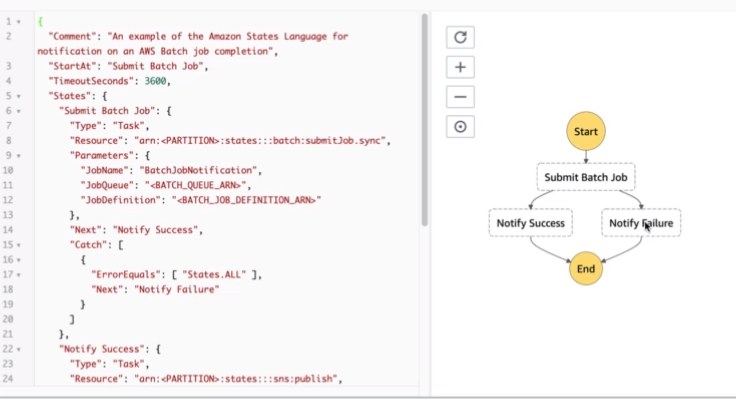
* Resilient and self-healing service that provides a quick and secured way of migrating databases to AWS.
* Source database remains available during the migration.
* Supports homogeneous migrations (eg. Oracle to Oracle) and some heterogeneous migrations (eg. MSSQL Server to Aurora).
* Uses Continuous Data replication using CDC (Continuous Data capture), and EC2 instances must be created for replication tasks.
* No transformation.



**1.8. AWS step Functions**

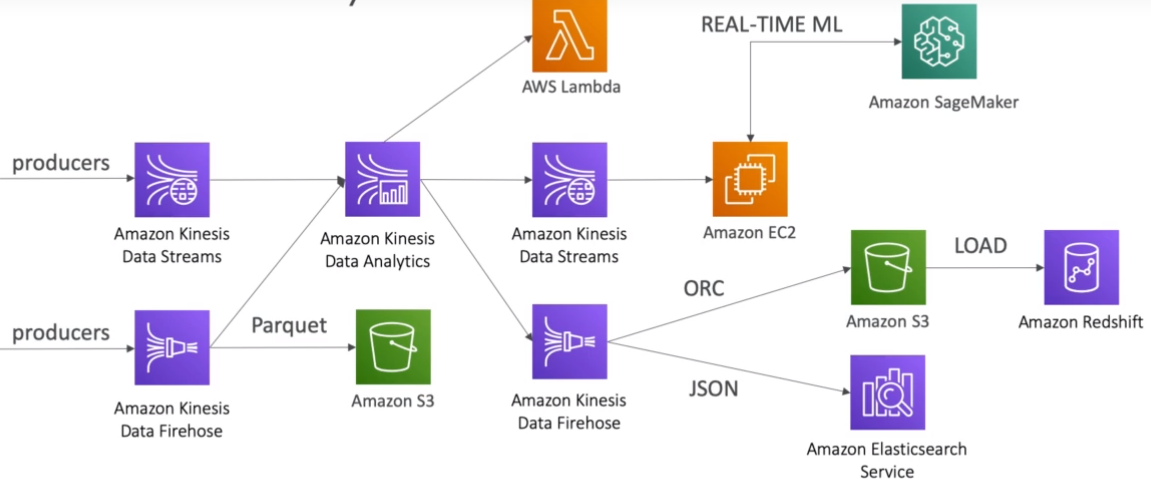
* Orchestrator using which we can make sure that the service workflows works well.
* Use to design workflows with easy visualizations, and advanced error handling and retry mechanism outside the code.
* Has audit of history of workflows.
* Ability to ‘Wait’ arbitrary amount of time.
* Max execution time of a state machine (workflow) is 1 year.



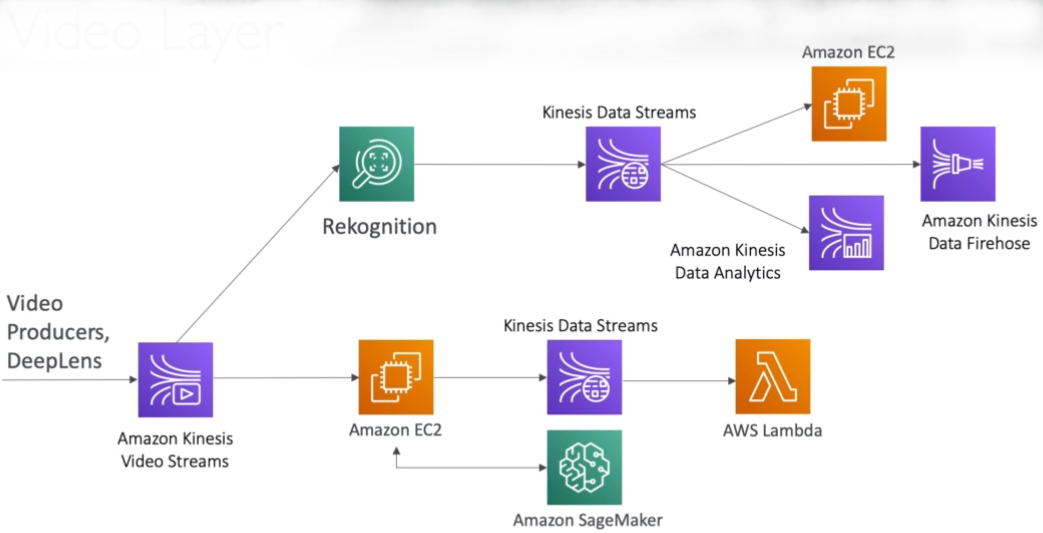
****

**1.9. Full Data-Engineering Pipeline**

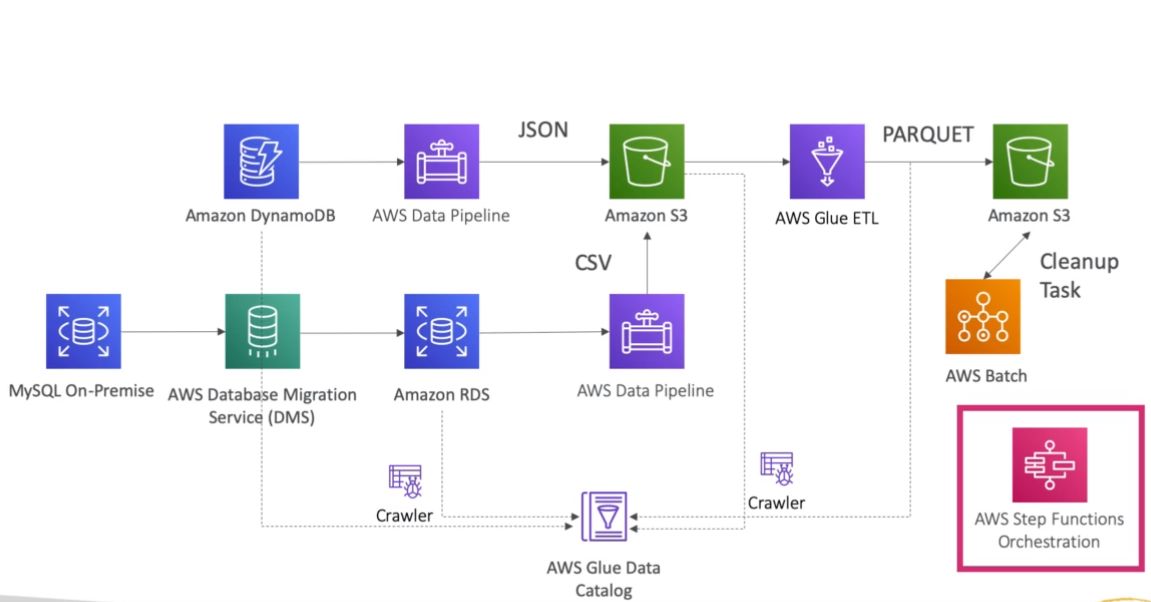
* Real Time Layer



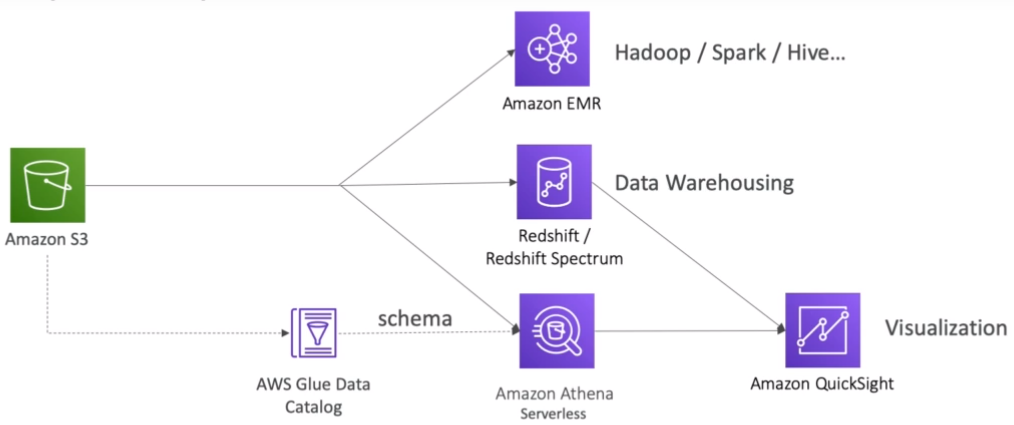
* Video Layer



* Batch Layer



* Analytics Layer



**1.10. Summary**

* Amazon S3: Object Storage for your data
* VPC Endpoint Gateway: Privately access your S3 bucket without going through the public internet
* Kinesis Data Streams: real-time data streams, need capacity planning, real-time applications
* Kinesis Data Firehose: near real-time data ingestion to S3, Redshift, ElasticSearch, Splunk
* Kinesis Data Analytics: SQL transformations on streaming data
* Kinesis Video Streams: real-time video feeds
* Glue Data Catalog & Crawlers: Metadata repositories for schemas and datasets in your account
* Glue ETL: ETL Jobs as Spark programs, run on a serverless Spark Cluster
* DynamoDB: NoSQL store
* Redshift: Data Warehousing for OLAP, SQL language
* Redshift Spectrum: Redshift on data in S3 (without the need to load it first in Redshift)
* RDS / Aurora: Relational Data Store for OLTP, SQL language
* ElasticSearch: index for your data, search capability, clickstream analytics
* ElastiCache: data cache technology
* Data Pipelines: Orchestration of ETL jobs between RDS, DynamoDB, S3. Runs on EC2 instances
* Batch: batch jobs run as Docker containers - not just for data, manages EC2 instances for you
* DMS: Database Migration Service, 1-to-1 CDC replication, no ETL
* Step Functions: Orchestration of workflows, audit, retry mechanisms

**2. Exploratory Data Analytics**