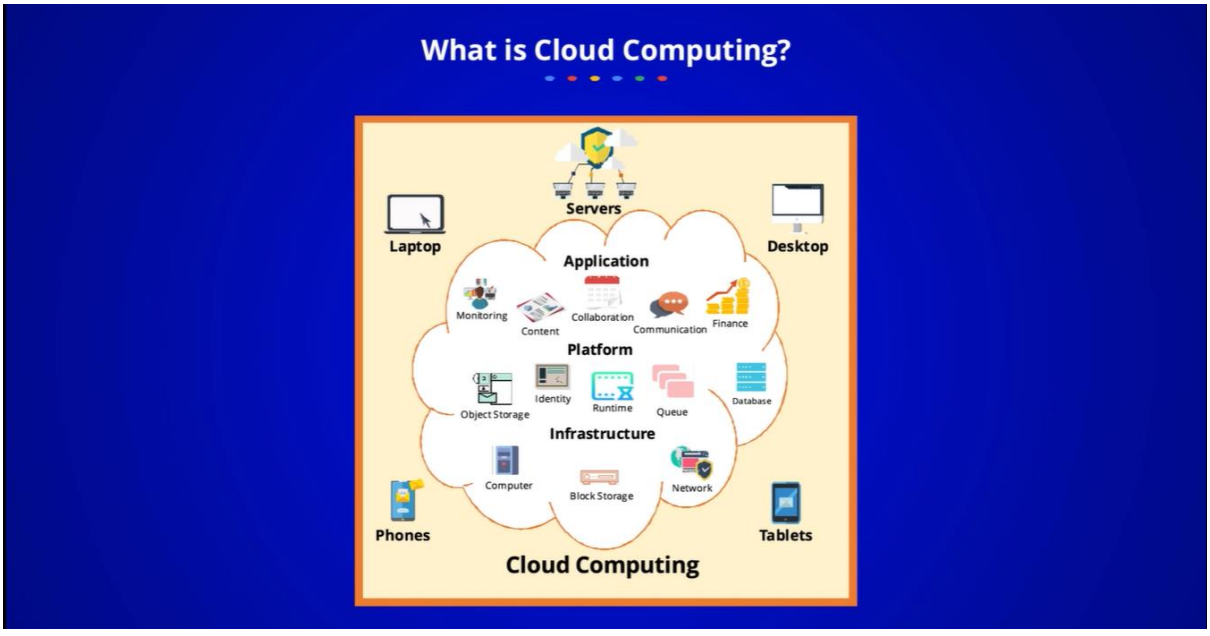
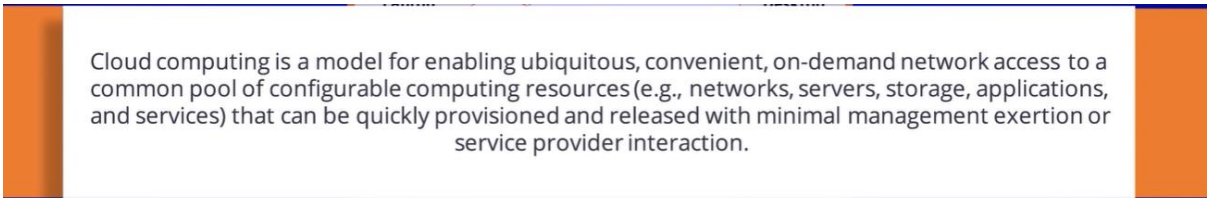
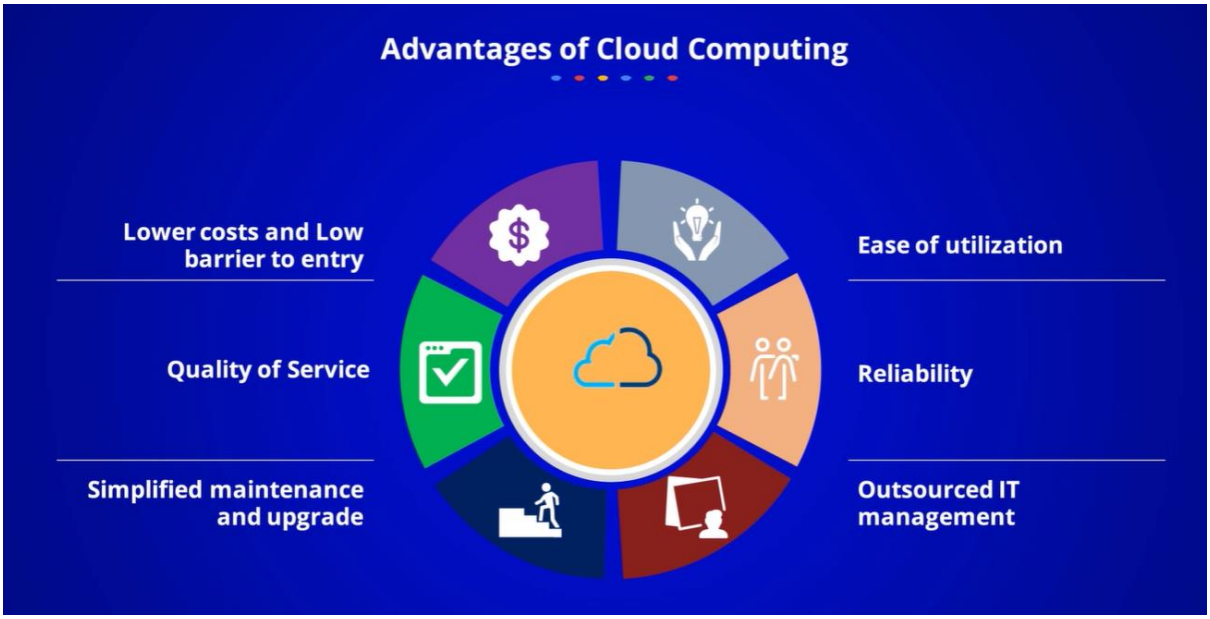
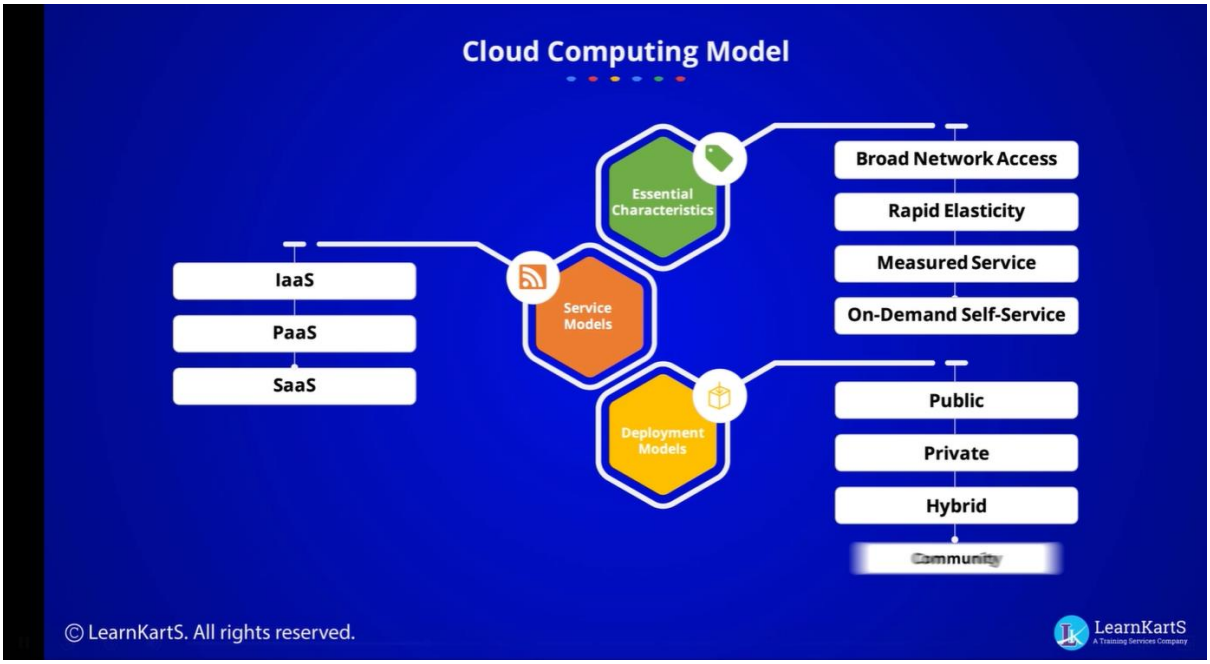
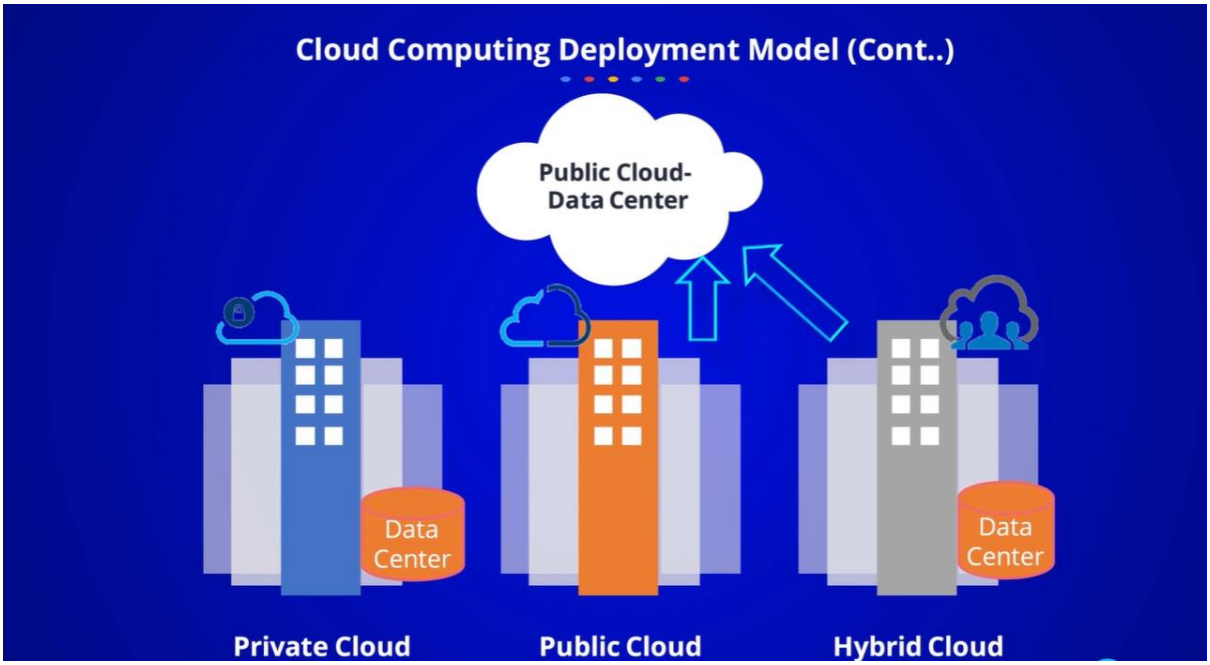
Cloud computing introduction, advantages and types.

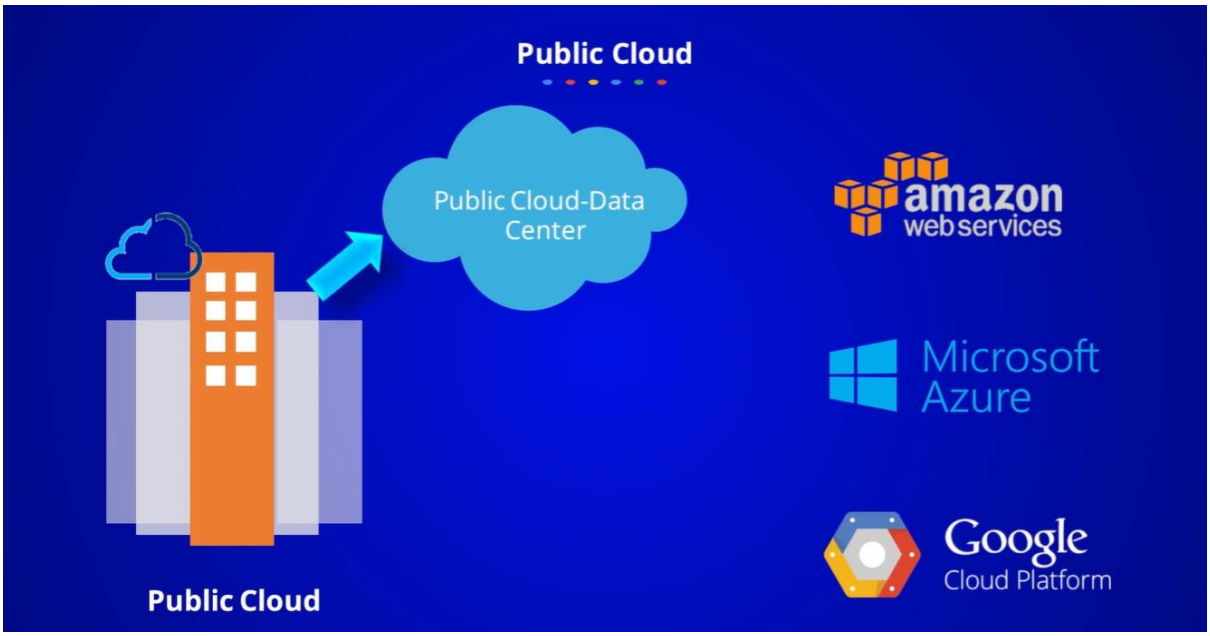


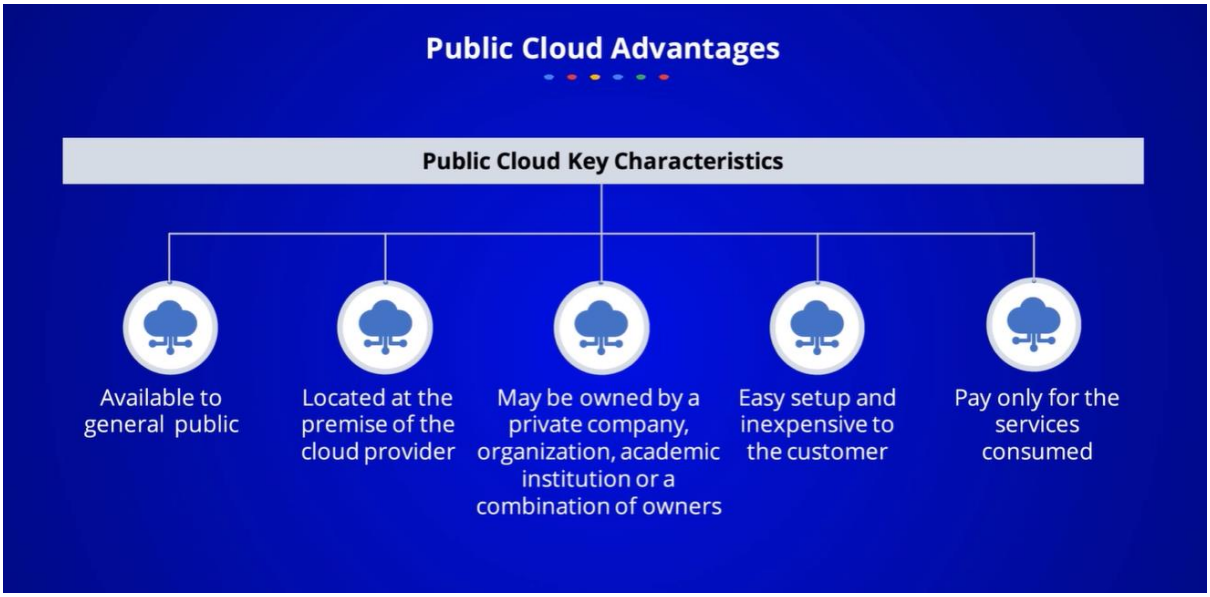


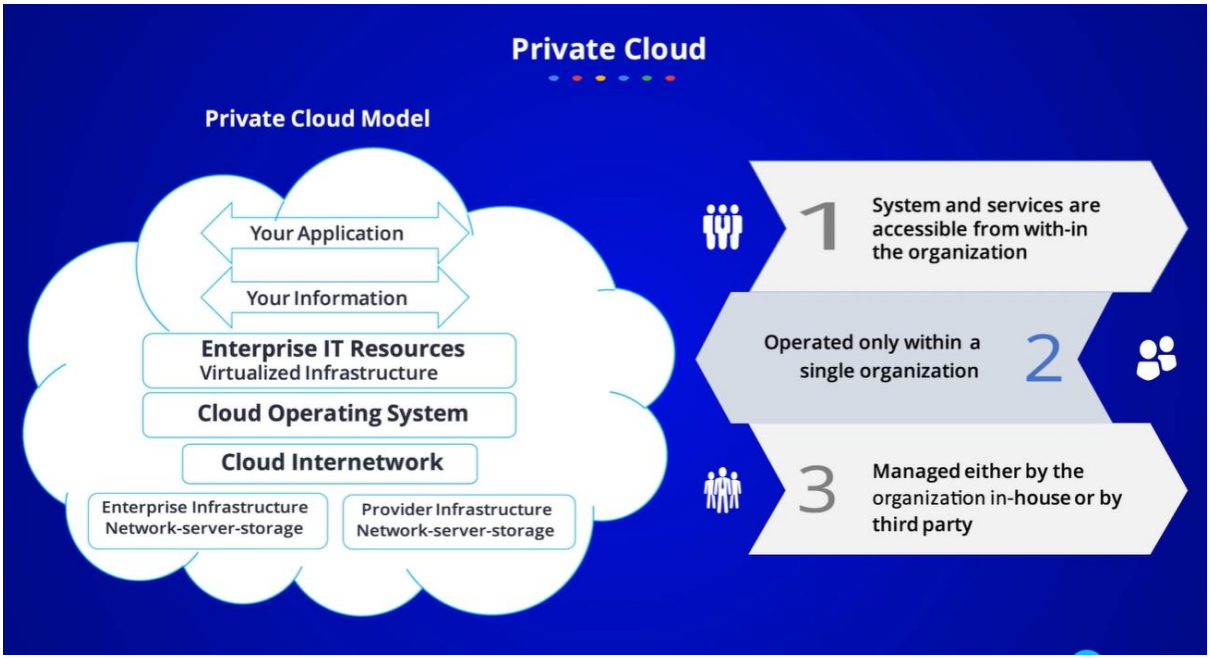

Advantages of Cloud Computing
Lower costs and Low barrier to entry
Quality of Service
Simplified maintenance and upgrade
Ease of utilization
Reliability
Outsourced IT management


Cloud Computing Model
Essential Characteristics
Broad Network Access
Rapid Elasticity
Measured Service
On-Demand Self-Service
Service Models
IaaS
PaaS
SaaS
Deployment Models
Public
Private
Hybrid
Community
© LearnKartS. All rights reserved.
LearnKartS


Cloud Computing Deployment Model (Cont..)
Public Cloud-
Data Center
Data
Center
Data
Center
Private Cloud
Public Cloud
Hybrid Cloud


Public Cloud
Public Cloud-Data
Center
amazon
web services
Microsoft
Azure
Google
Cloud Platform
Public Cloud


Public Cloud Advantages
Public Cloud Key Characteristics
Available to general public
Located at the premise of the cloud provider
May be owned by a private company, organization, academic institution or a combination of owners
Easy setup and inexpensive to the customer
Pay only for the services consumed


Private Cloud
Private Cloud Model
Your Application
Your Information
Enterprise IT Resources
Virtualized Infrastructure
Cloud Operating System
Cloud Internetwork
Enterprise Infrastructure
Network-server-storage
Provider Infrastructure
Network-server-storage
1
System and services are accessible from with-in the organization
Operated only within a single organization
2
3
Managed either by the organization in-house or by third party

Private Cloud Disadvantages

Restricted Area of Operation

Private cloud is only accessible with-in the organization hence cannot be deployed globally.

Limited Scalability

Scalability is limited to the resources which are available with-in organization

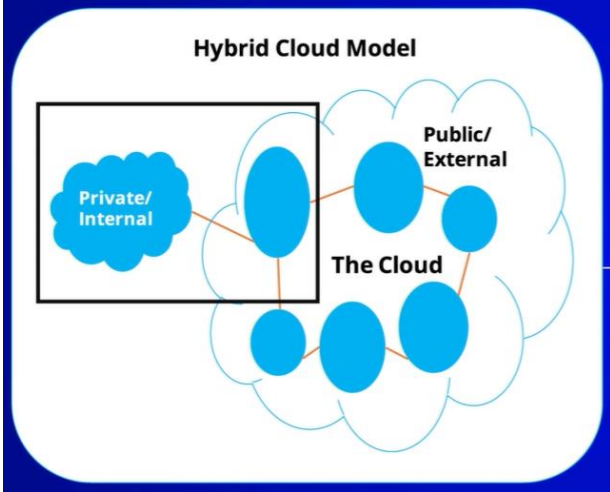
High Priced

Organizations need to purchase hardware to deploy a Private cloud

Additional Skills

Maintaining cloud infrastructure required additional expertise with-in organization

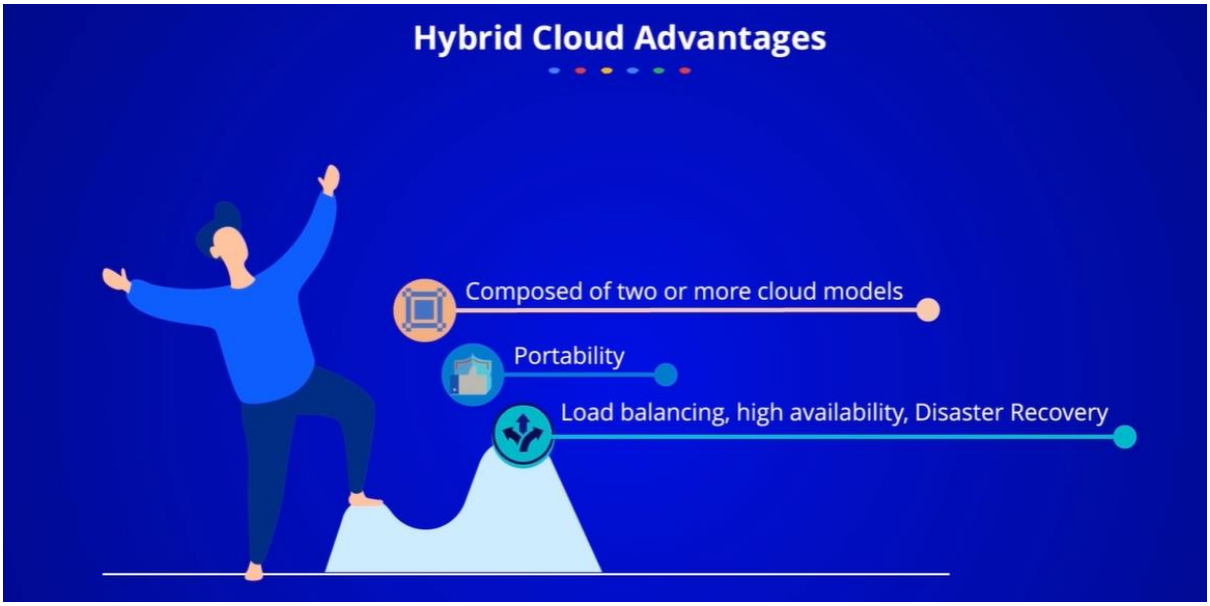
Hybrid Cloud



Mixture of **public** and **private** cloud.

Non-critical activities are performed using public cloud

Critical activities are performed using private cloud

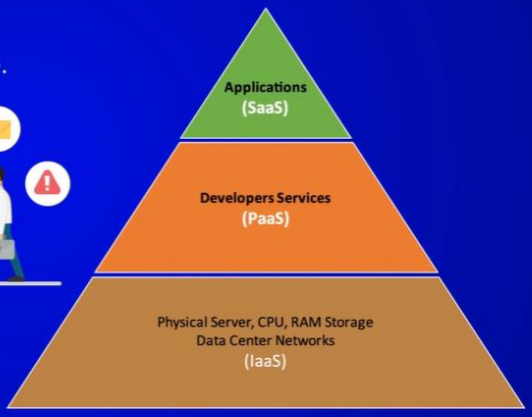

Hybrid Cloud Advantages
Composed of two or more cloud models
Portability
Load balancing, high availability, Disaster Recovery

Cloud Service Categories

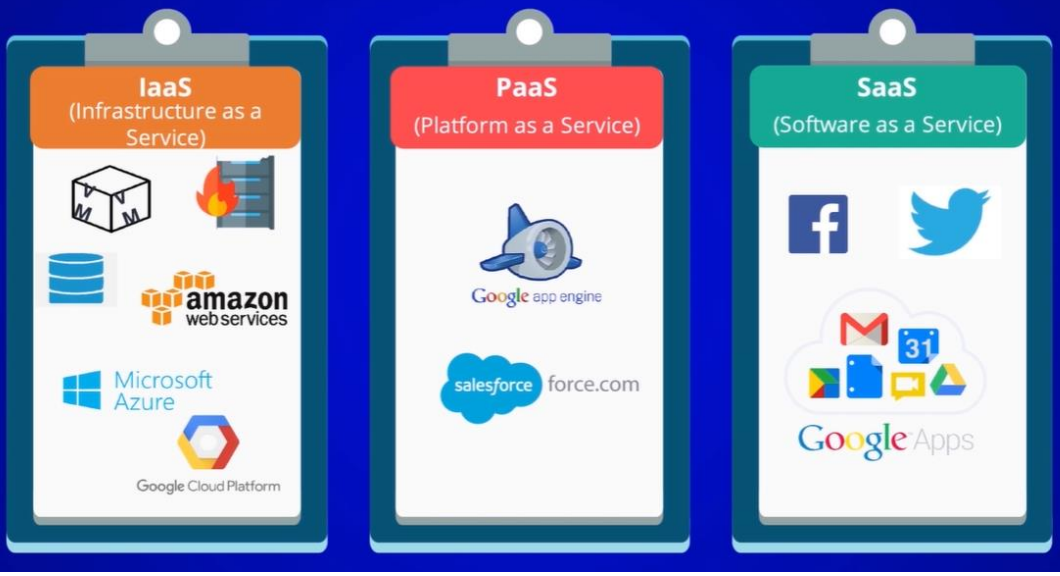
IAAS : Provides Basic Building Block for Cloud IT

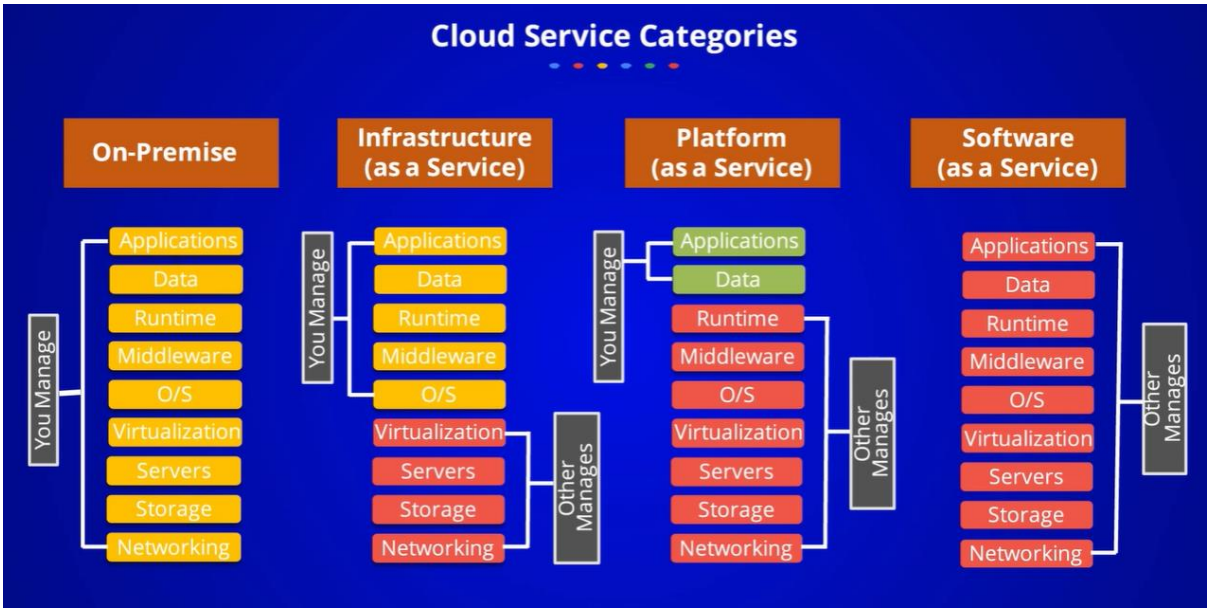
PAAS : Manages its own underlying infrastructure.

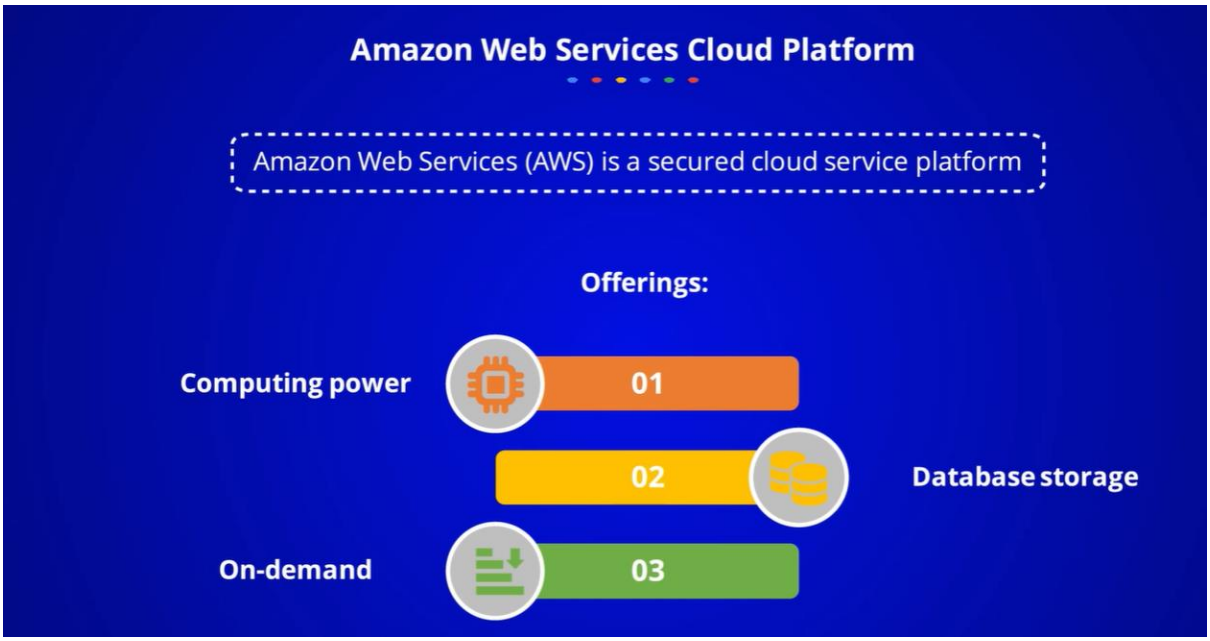
SAAS : Offers its complete product as a webservice.

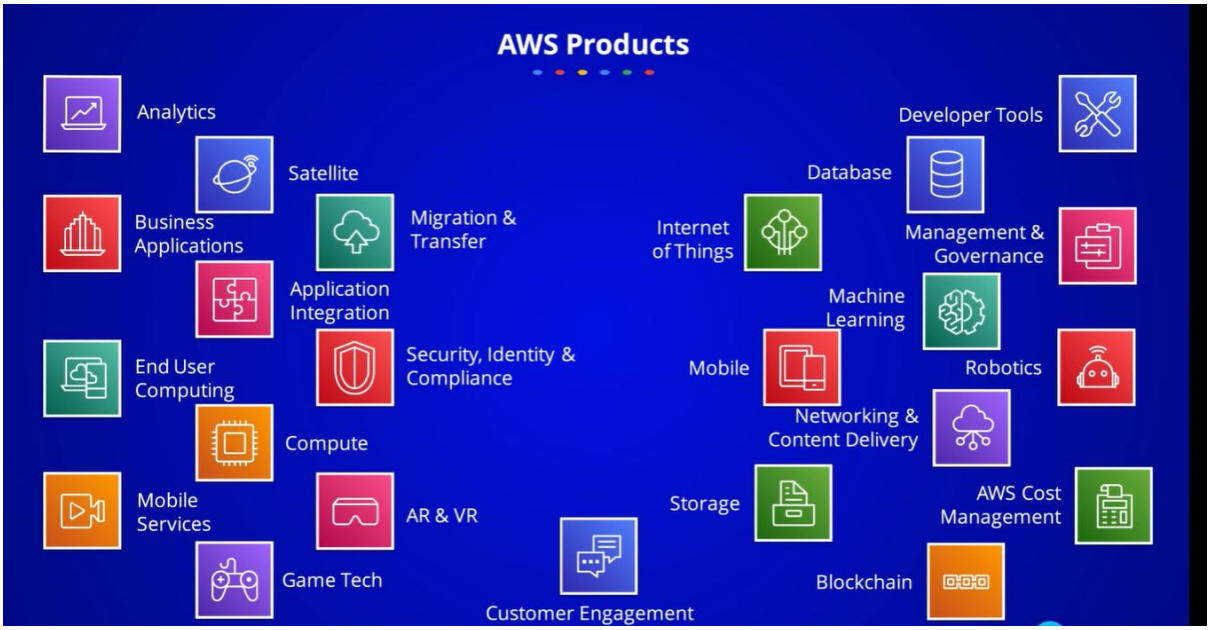


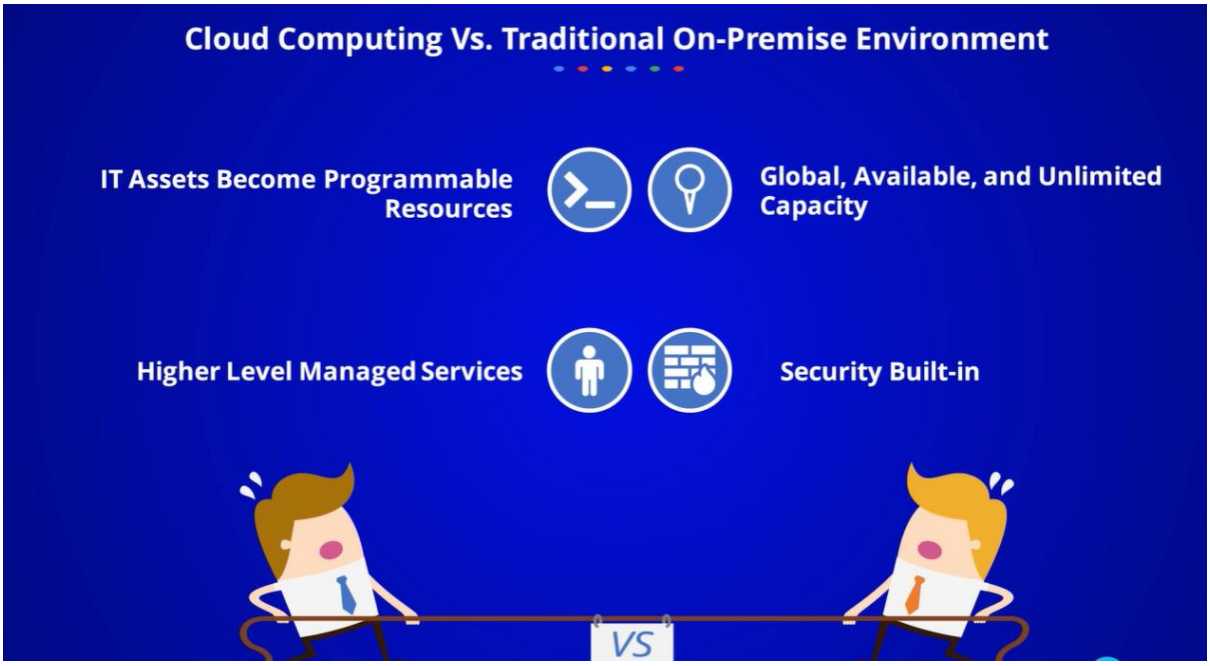
Cloud Service Categories

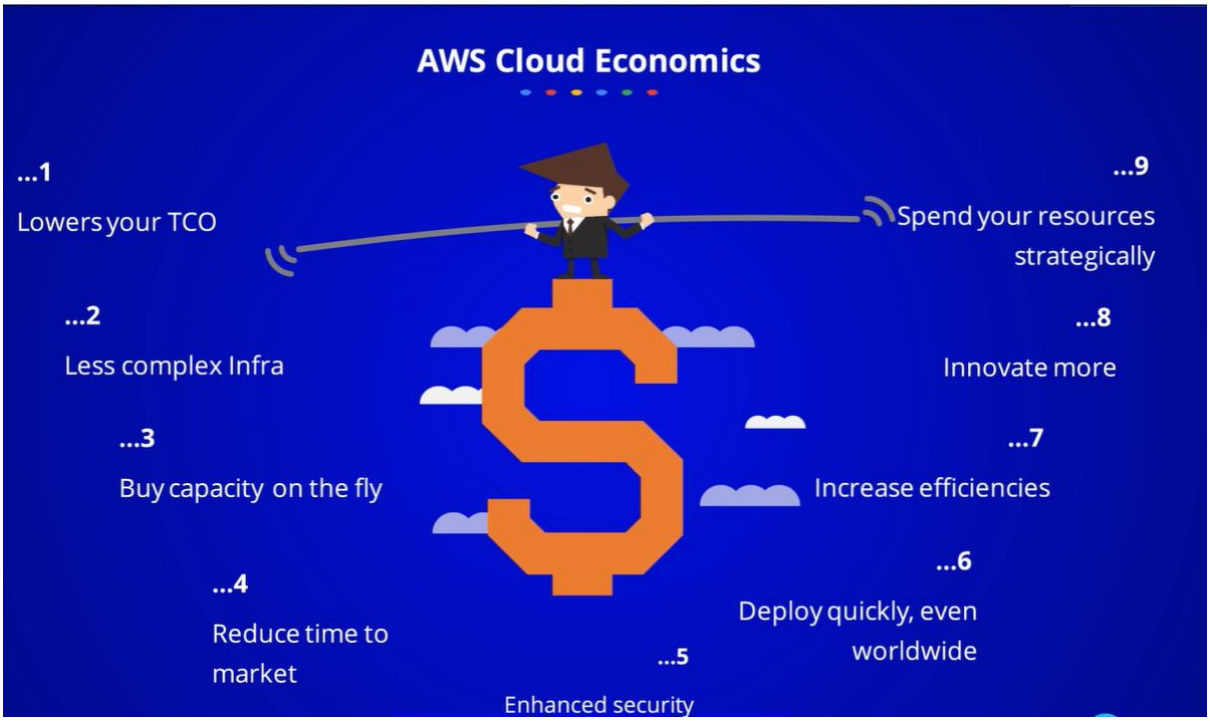


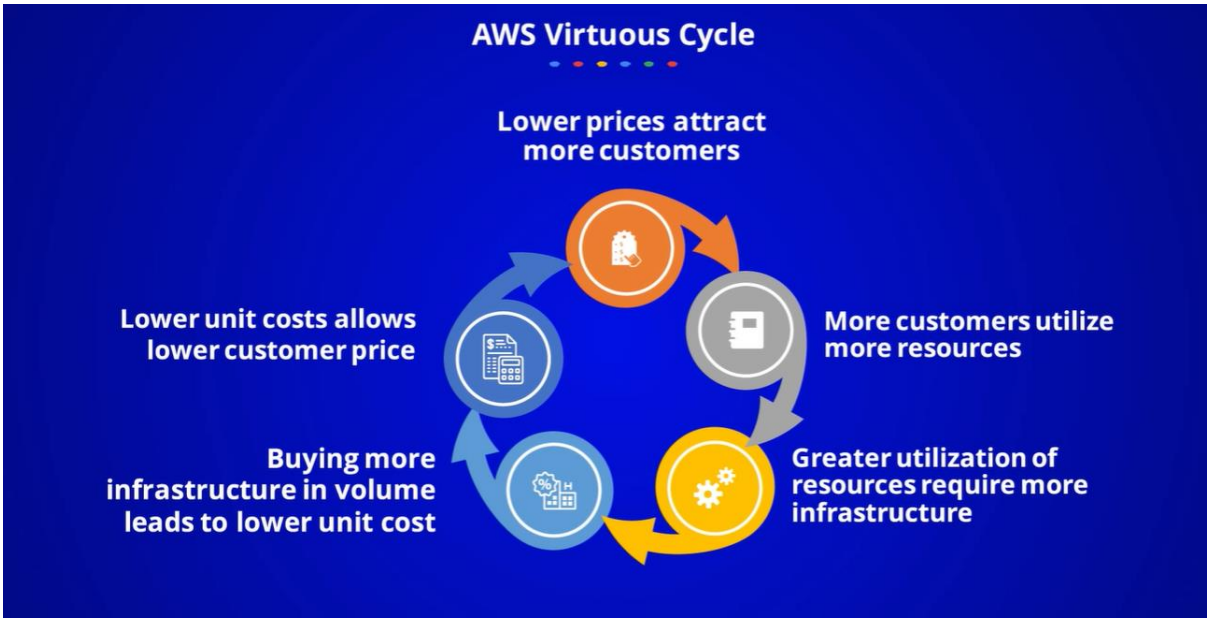

Cloud Service Categories
On-Premise
Infrastructure (as a Service)
Platform (as a Service)
Software (as a Service)
You Manage
Applications
Data
Runtime
Middleware
O/S
Virtualization
Servers
Storage
Networking
Other Manages

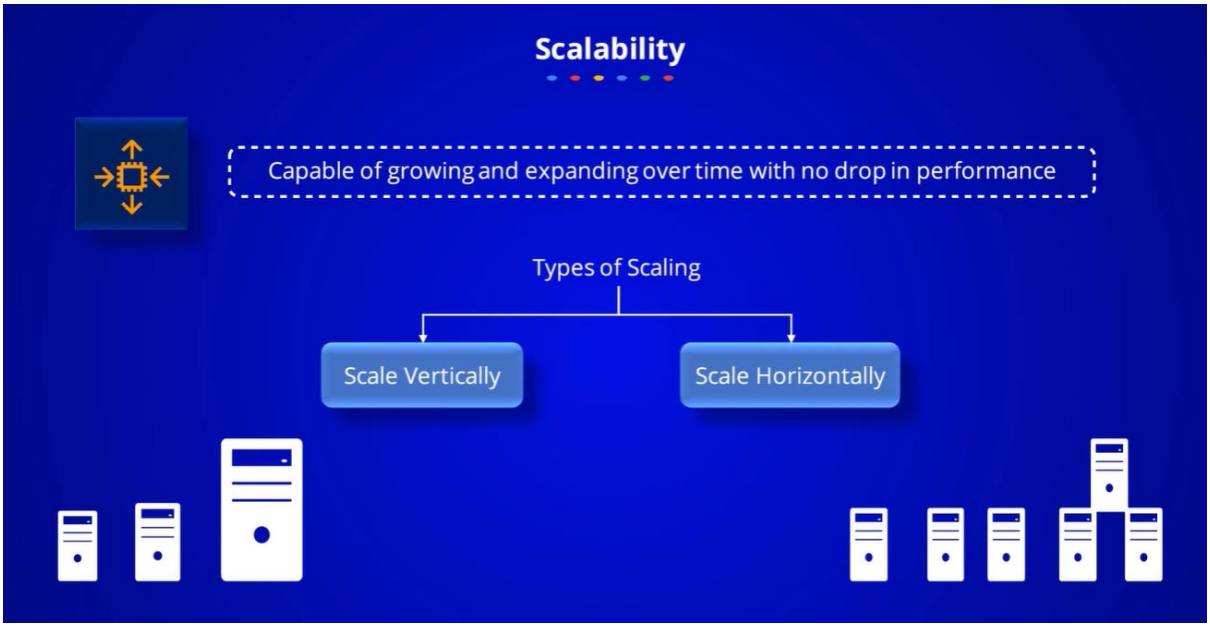

Amazon Web Services Cloud Platform
Amazon Web Services (AWS) is a secured cloud service platform
Offerings:
Computing power
01
02
Database storage
On-demand
03


AWS Products
Analytics
Satellite
Business Applications
Migration & Transfer
Application Integration
End User Computing
Security, Identity & Compliance
Compute
Mobile Services
AR & VR
Game Tech
Customer Engagement
Developer Tools
Database
Internet of Things
Management & Governance
Machine Learning
Mobile
Robotics
Networking & Content Delivery
Storage
AWS Cost Management
Blockchain


Cloud Computing Vs. Traditional On-Premise Environment
IT Assets Become Programmable Resources
Global, Available, and Unlimited Capacity
Higher Level Managed Services
Security Built-in
VS


AWS Cloud Economics
...1
Lowers your TCO
...2
Less complex Infra
...3
Buy capacity on the fly
...4
Reduce time to market
...5
Enhanced security
...6
Deploy quickly, even worldwide
...7
Increase efficiencies
...8
Innovate more
...9
Spend your resources strategically


AWS Virtuous Cycle
Lower prices attract more customers
More customers utilize more resources
Greater utilization of resources require more infrastructure
Buying more infrastructure in volume leads to lower unit cost
Lower unit costs allows lower customer price



Scale Vertically : adding more servers and added high ram and storage.

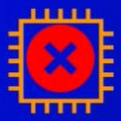
Scale horizontally: scale adding more machine to pool of resources to exiting machine.

Disposable Resources Instead of Fixed Servers

Auto Scaling permits EC2 instances to be disposed of when not utilized and new instances to be provisioned in seconds to satisfy the higher demand.

RDS database scales into several nodes and any node can be discarded without impact on DB availability

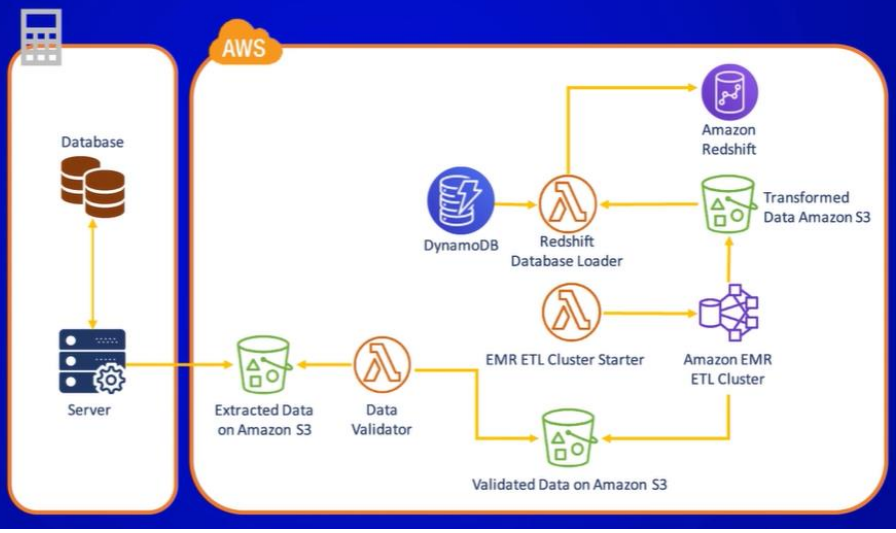
CloudWatch

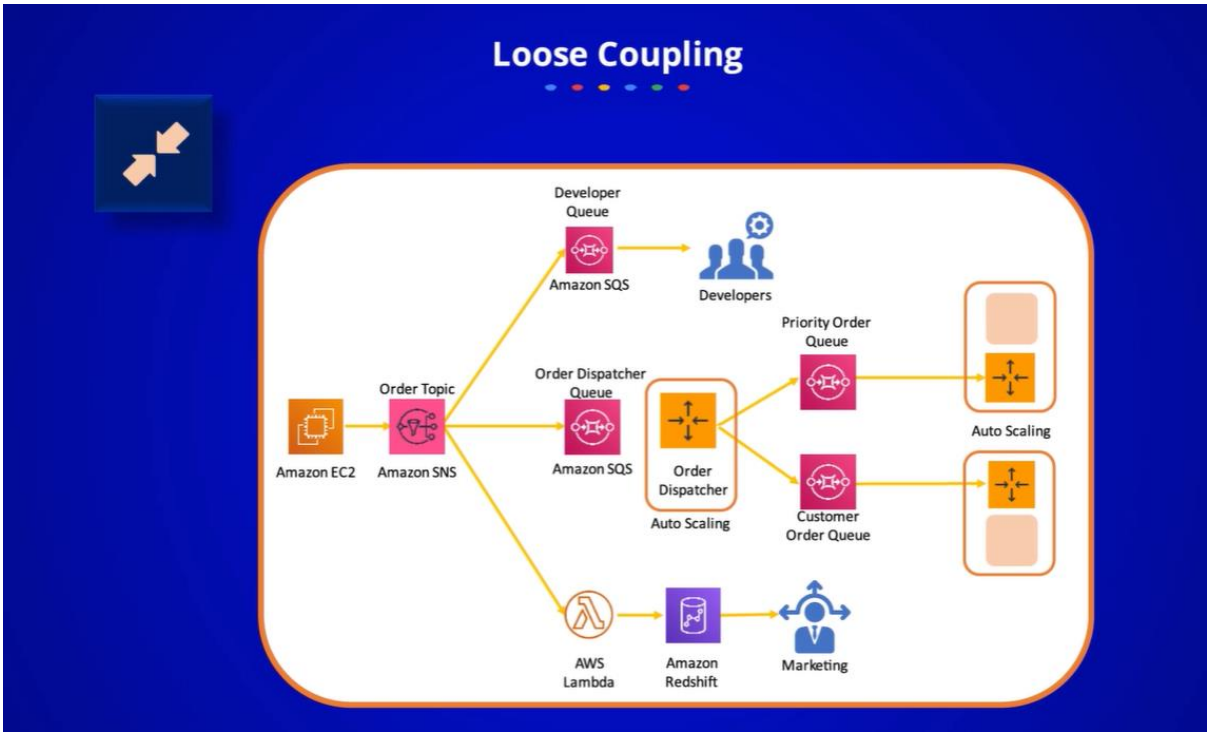


Instance crashes

Replacement automatically launches

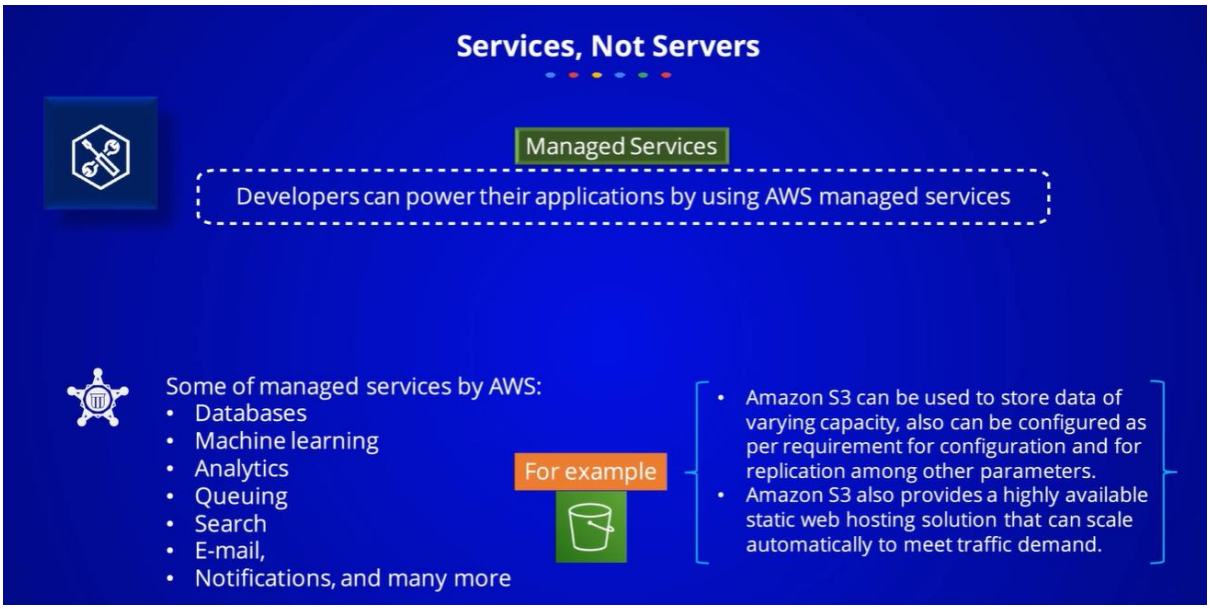
Automation





Loose coupling artichecture.

Reduce dependences and for change reduce failler and not casecate to another compound .



Services, Not Servers

Server-less Architectures

Server-less architectures reduce the operational complexity of running applications.

Both Event-driven and synchronous services can be built without managing any server infrastructure.

For example

Your code can be uploaded to AWS Lambda compute service that runs the code on your behalf.

Removing Single Points of Failure

A system needs to be highly available to withstand any failure of the individual or multiple components (e.g., hard disks, servers, network links, etc.).

Introduce Redundancy

Automated Detection of Failure

Durable Data Storage

Automated Multi-Data Center Resilience

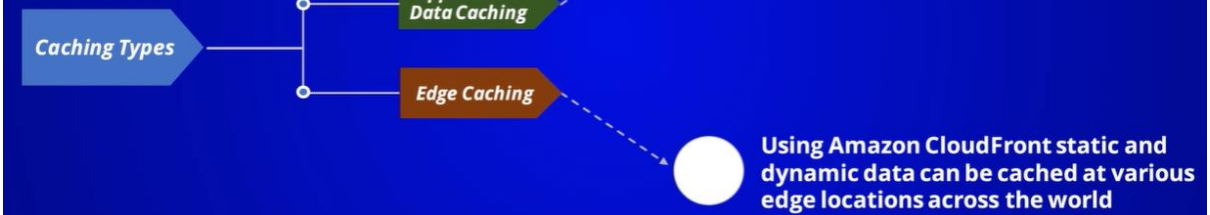
Caching

To store previously examined data, caching is used

Improves application performance

Increases the cost efficiency of implementation

Caching Types



Application data can be stored in the cache for subsequent requests to:

- Improve latency for end users
- Reduce the load on back-end systems

Using Amazon CloudFront static and dynamic data can be cached at various edge locations across the world

Security

Businesses have legal obligations to keep client data secure

How to achieve security

Utilize AWS Features for Defense in Depth

- Use VPC for Isolation and WAF for Application protection

Shared Security Responsibility to AWS

- Security of the Cloud-AWS
- Security for the Data-Customer

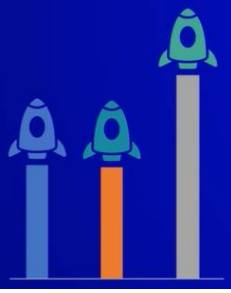
Reduce Privileged Access

- To avoid a breach of security

Real-Time Auditing

- Continuously monitor and automate controls to minimize security risk exposures

Why AWS for Big Data-Challenges and Reasons



Big Data requires significant compute capacity that can vary in size.

Big Data on AWS meet your needs, without having to wait for additional hardware or being required to over invest to provision enough capacity.

Big Data on AWS lets you provision more capacity and compute in a matter of minutes, meaning that your big data applications grow and shrink as demand dictates

Why AWS for Big Data-Challenges and Reasons

Amazon S3

AWS Glue

AWS IoT

Lets you to store data, orchestrate jobs and lets connected devices interact.

Amazon Kinesis Data Firehose

Lets you load streaming data continuously

AWS Database Migration

Lets you migrate data easily. AWS

AWS Direct Connect

Lets you have scalable private connections between On-premise and Cloud.

Databases in AWS

Relational Databases

Non-Relational Databases

Data Warehousing in AWS

Benefits of AWS Database Services

Fully managed services
AWS handles installs,
patching, restarts

Easy to scale
Grow as you need

Pay only for
what you use
No up-front cost

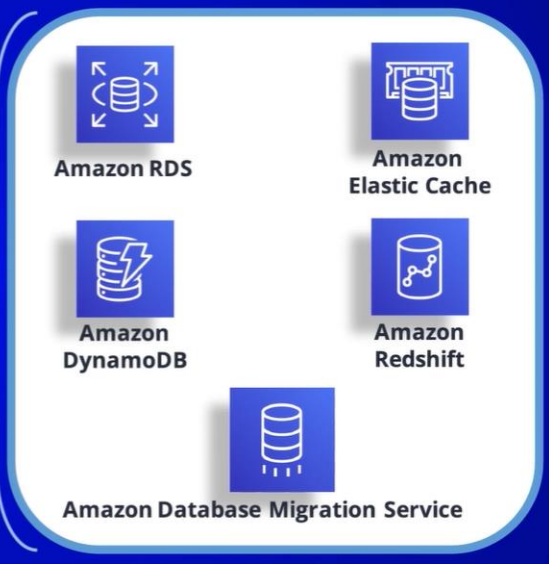
Designed for use with
other AWS services

Databases in AWS

AWS Database Services

Scalable High Performance
Application Storage in the Cloud

- Deployment & Administration
- Application Services
- Compute | Storage | **Database**
- Networking
- AWS Global Infrastructure



Databases in AWS

Amazon Relational Database Service (Amazon RDS) is used to setup, operate and scale relational database in cloud.

Automates time-consuming administrative task with cost efficiency and resizable capacity:

- **hardware provisioning,**
- **database setup,**
- **patching and backups.**

Allows to focus on following tasks for the application:

- **fast performance,**
- **high availability,**
- **security and compatibility.**

Databases in AWS

Amazon RDS is available on multiple database instance types which one can choose from according to the specific application requirement- optimized for memory, performance or I/O.

There are 6 well known database engines available with Amazon RDS:

Amazon Aurora

PostgreSQL

MySQL

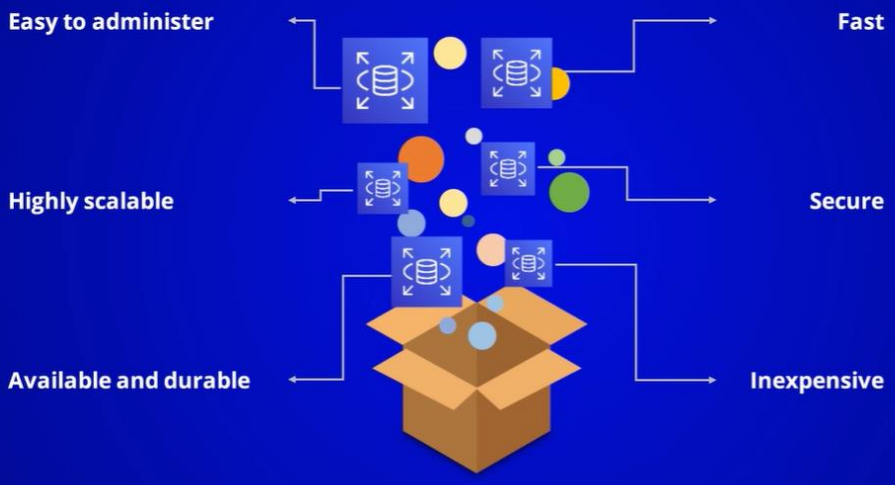
MariaDB

ORACLE

Microsoft SQL Server

Databases in AWS

Relational Databases



Databases in AWS

Relational Databases

Relational Databases Use Cases

Ecommerce applications:
Provides a managed database offering

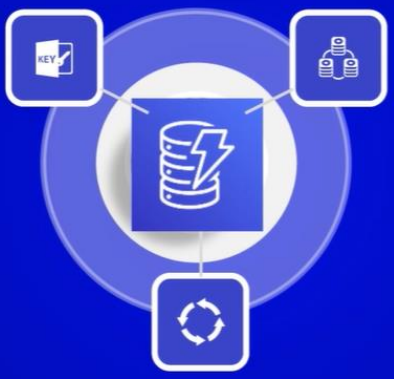
Mobile and online games
Amazon RDS manages the database infrastructure.

Web and mobile applications :
Amazon RDS does not have any licensing constraints.

Databases in AWS

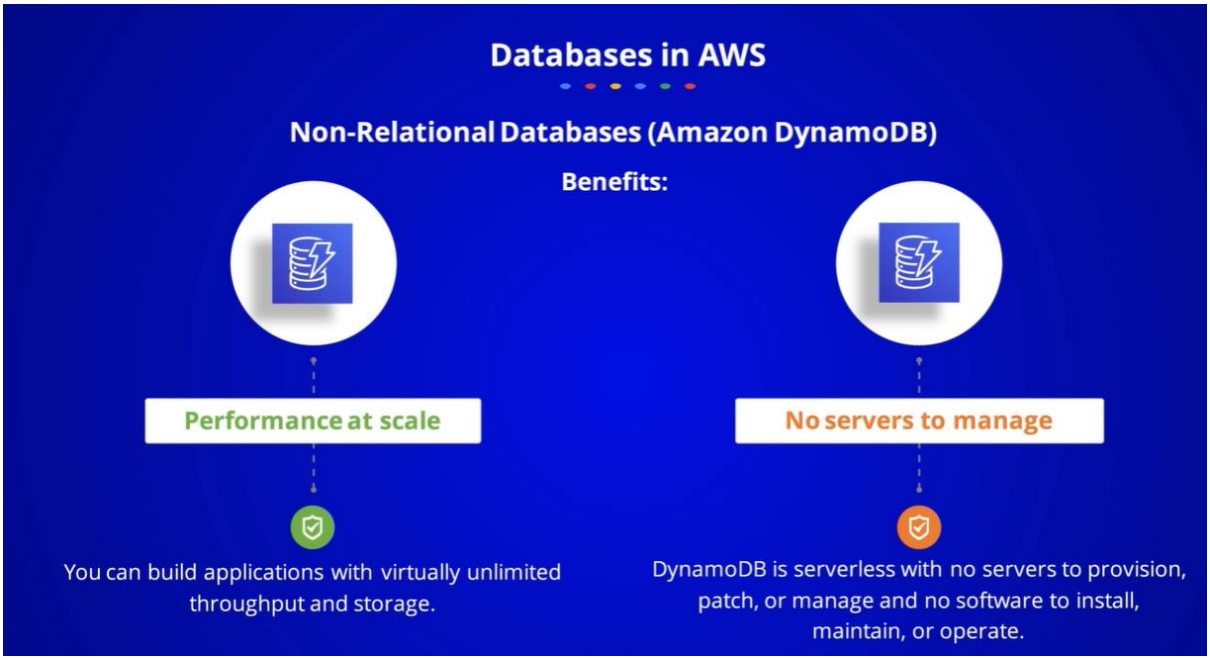
Non-Relational Databases (Amazon DynamoDB)

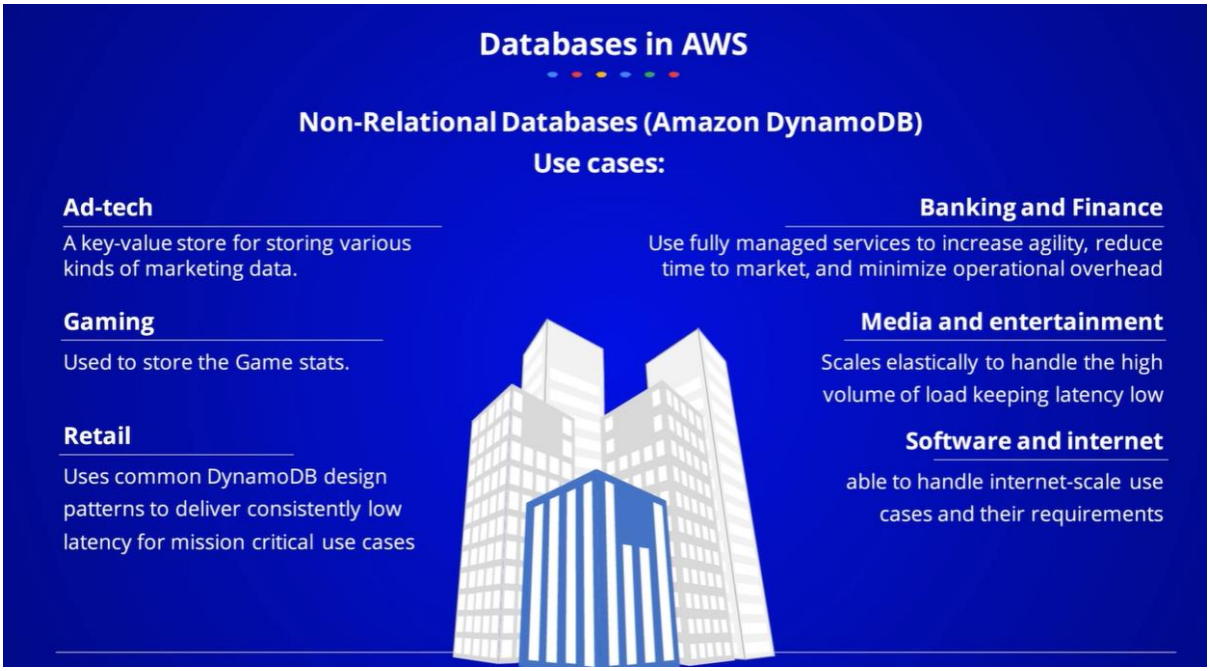
A key-value and document database



A fully managed, multi-region, multi-master, durable database

It can handle 10 trillion requests per day



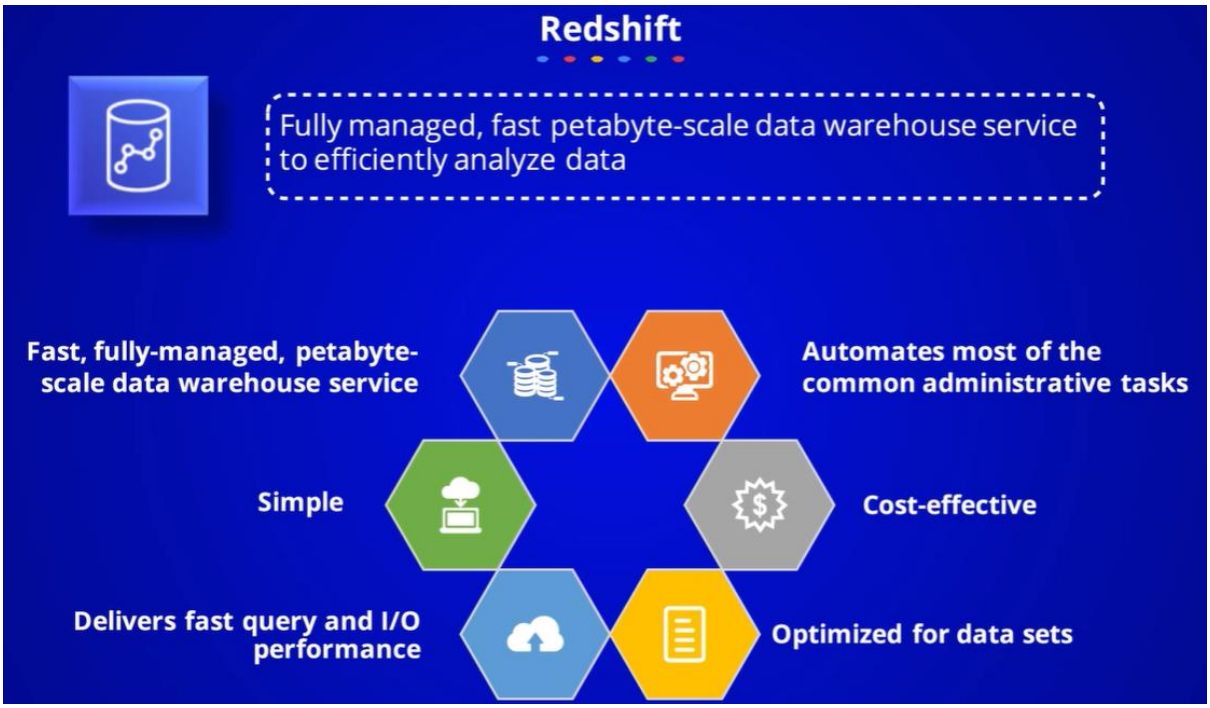


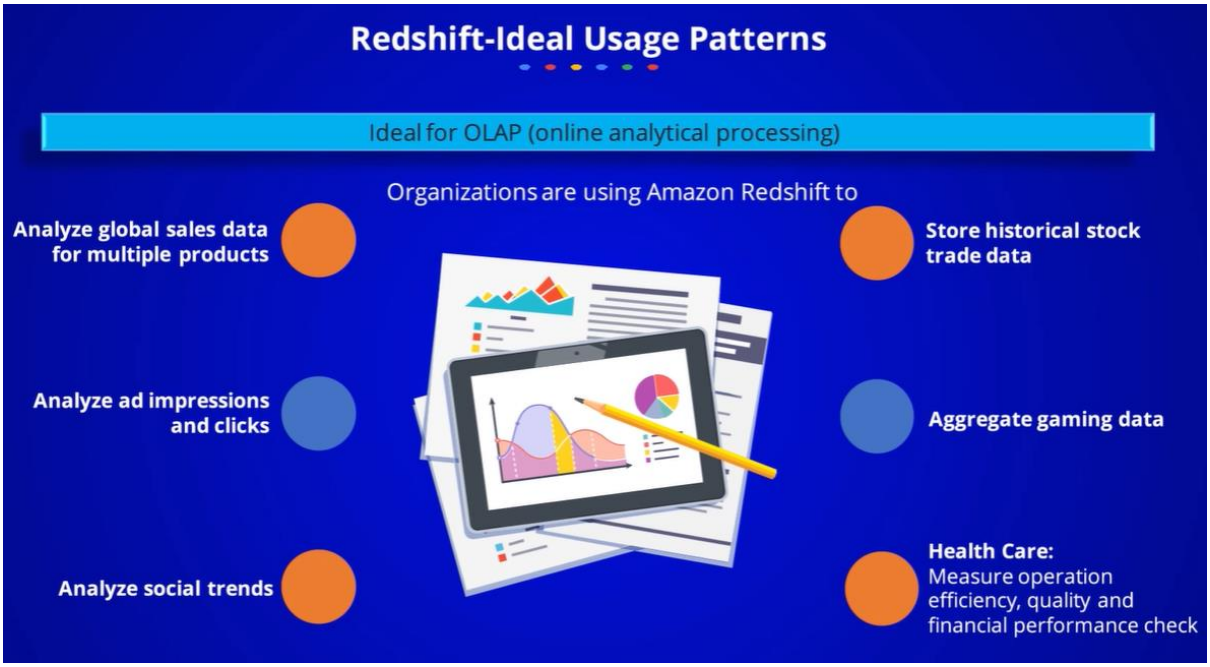
Redshift spl relations databases are data wasehouse


Data Warehousing in AWS
Amazon Redshift
Data warehouse:
Fast, fully managed and cost-effective
It gives you petabyte scale data warehousing and exabyte scale data lake analytics together in one service
It is up to ten times faster than traditional on-premises data warehouses.


Data Warehousing in AWS
Amazon Redshift
Key Features:
Maximize your ROI
Amazon Redshift is a cost effective cloud data warehouse that costs less than 1/10th the cost of traditional data warehouses.
Setup, Deploy, and Manage with ease
Fast, fully managed, simple and cost-effective data warehouse

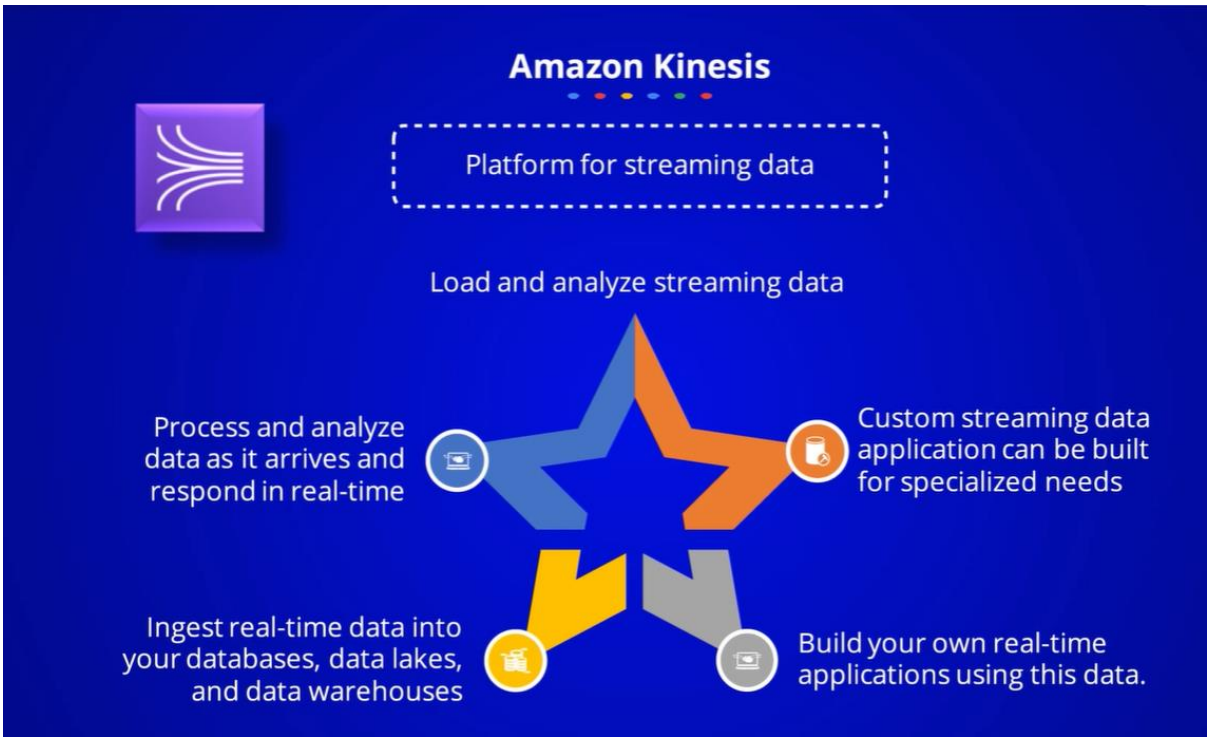

Data Warehousing in AWS
Amazon Redshift
Log Analysis: Redshift is cost effective, it's possible to store raw, event-level data without getting killed on storage cost.
Business Applications real-time analytics: Companies uses Redshift under the covers, and offer analytics in a SaaS model to their customers.
Mission-critical Workloads: Companies use Redshift for Mission-critical workloads through time-sensitive apps where the database stays up.


Redshift
Fully managed, fast petabyte-scale data warehouse service to efficiently analyze data
Fast, fully-managed, petabyte-scale data warehouse service
Automates most of the common administrative tasks
Simple
Cost-effective
Delivers fast query and I/O performance
Optimized for data sets



Kinesis

It’s a platform for steaming data and to collect and process and analysis real time steaming data.



Amazon Kinesis

Currently there are 4 pieces of the Kinesis platform that can be utilized based on use case:

Amazon Kinesis Data Streams

Enables to build custom applications that process or analyze streaming data.

Amazon Kinesis Video Streams

Enables you to build custom applications that process or analyze streaming video

Amazon Kinesis Data Firehose

Enables to deliver real-time streaming data to AWS destinations such as Amazon S3, Amazon Redshift, Amazon Kinesis Analytics, and Amazon Elasticsearch Service.

Amazon Kinesis Data Analytics

Enables to process and analyze streaming data with standard SQL.

Amazon Kinesis-Ideal Usage Patterns

Real-time data analytics

- Analyzing website clickstream data
- Customer engagement analytics

Log and data feed intake and processing

- Producers push data to Kinesis stream directly
- Prevents log data loss in case of failure (front end or application failure)

Real-time metrics and reporting

- Enables KPIs and metrics generation in real-time for reporting/dashboards

Amazon EMR

Highly distributed computing framework to process and store data quickly, easily and in cost effective manner.

Uses Apache Hadoop to distribute your data and processing across a resizable cluster of Amazon EC2 instances

Amazon EMR does all the work involved with provisioning, managing, and maintaining the infrastructure and software of a Hadoop cluster.

Amazon EMR-Ideal Usage Patterns

Log processing and analytics

Genomics

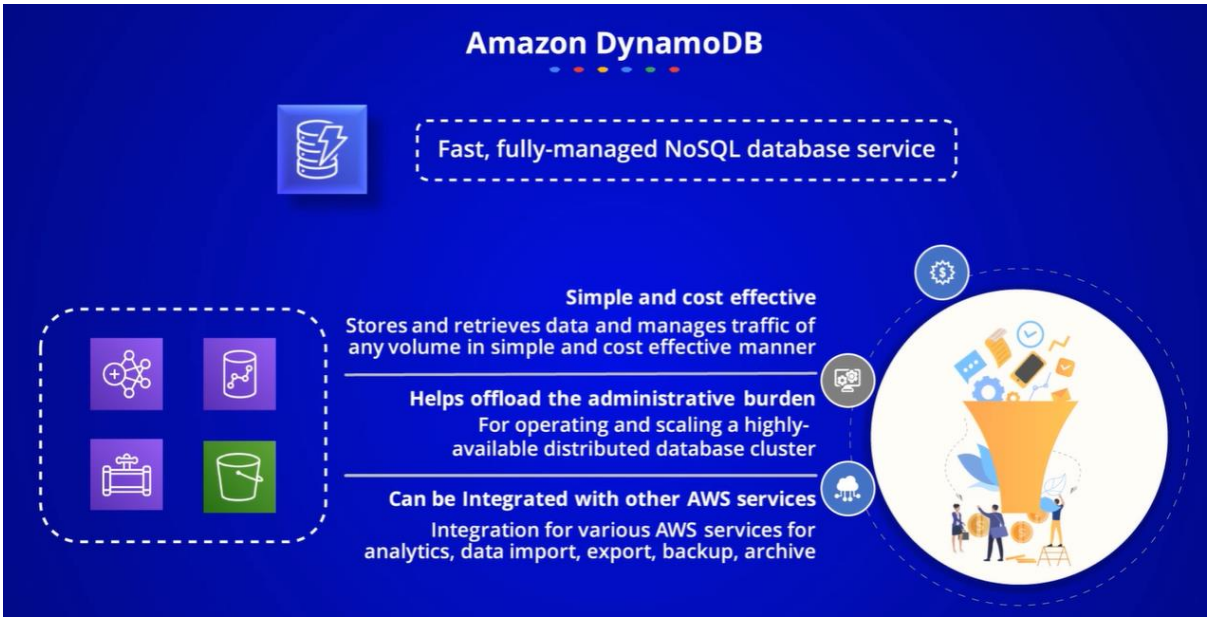
Ad targeting and click stream analytics

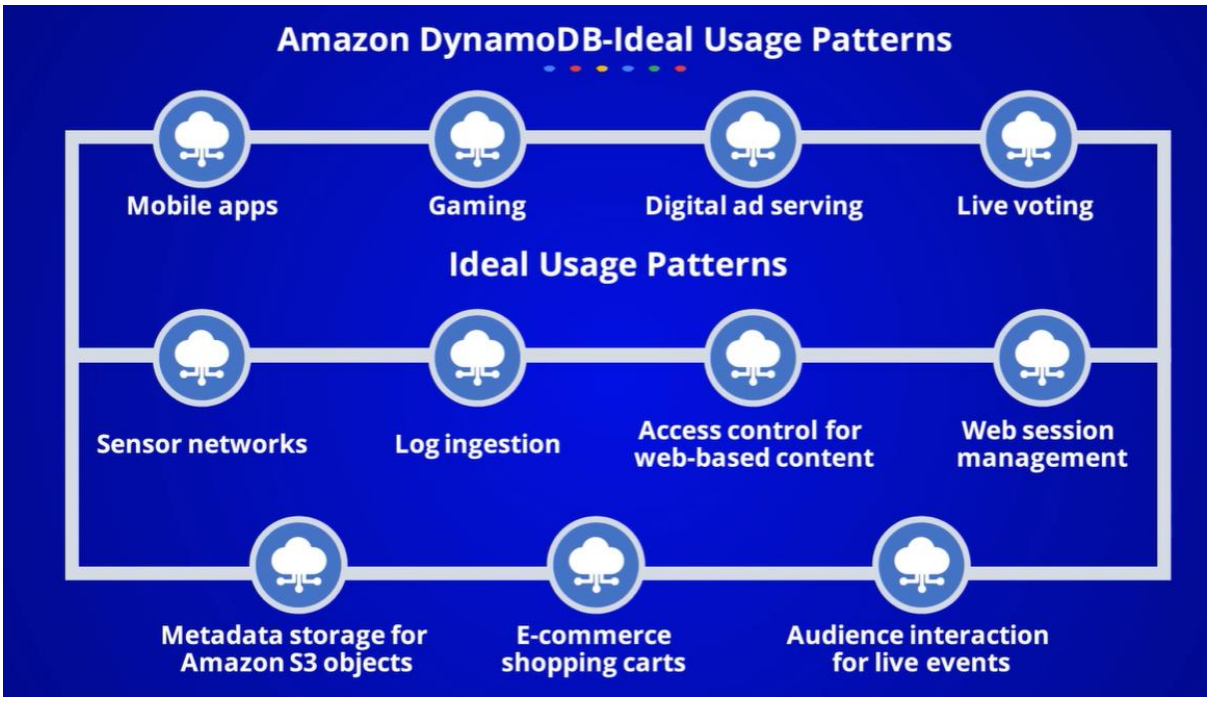
Large extract, transform, and load (ETL) data movement

Risk modeling and threat analytics

Predictive analytics

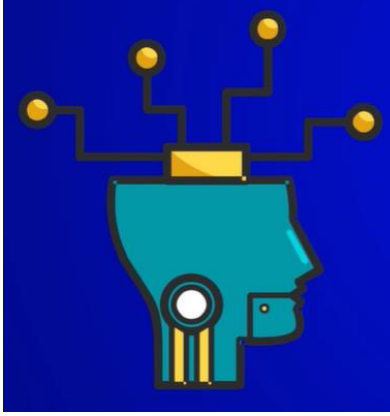
Ad hoc data mining and analytics


Amazon DynamoDB
Fast, fully-managed NoSQL database service
Simple and cost effective
Stores and retrieves data and manages traffic of any volume in simple and cost effective manner
Helps offload the administrative burden
For operating and scaling a highly-available distributed database cluster
Can be Integrated with other AWS services
Integration for various AWS services for analytics, data import, export, backup, archive


Amazon DynamoDB-Ideal Usage Patterns
Mobile apps
Gaming
Digital ad serving
Live voting
Ideal Usage Patterns
Sensor networks
Log ingestion
Access control for web-based content
Web session management
Metadata storage for Amazon S3 objects
E-commerce shopping carts
Audience interaction for live events

Amazon Machine Learning

Amazon ML is a service that makes it easy for anyone to use predictive analytics and machine-learning technology.



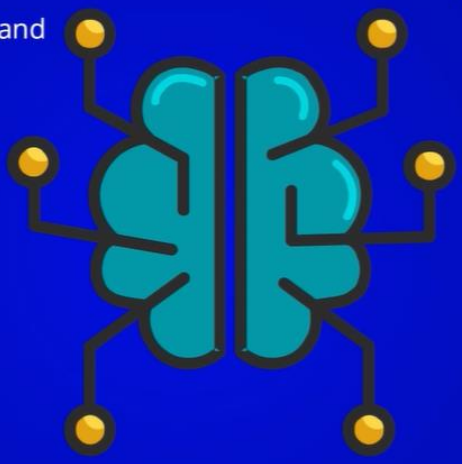
Provides visualization tool/wizards to create ML models
Learning of complex ML algorithms and technology is not required

Can create ML models based on data stored in AWS:
- S3
- Redshift
- RDS

Amazon Machine Learning-Ideal Usage Patterns

Predict user activity

Forecast product demand



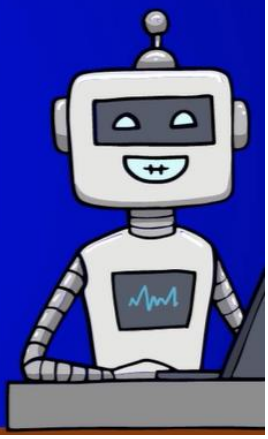
Personalize application content

Listen to social media

Enable applications to flag suspicious transactions

AWS Lambda

Let you run code directly on AWS without any provisioning/configuring



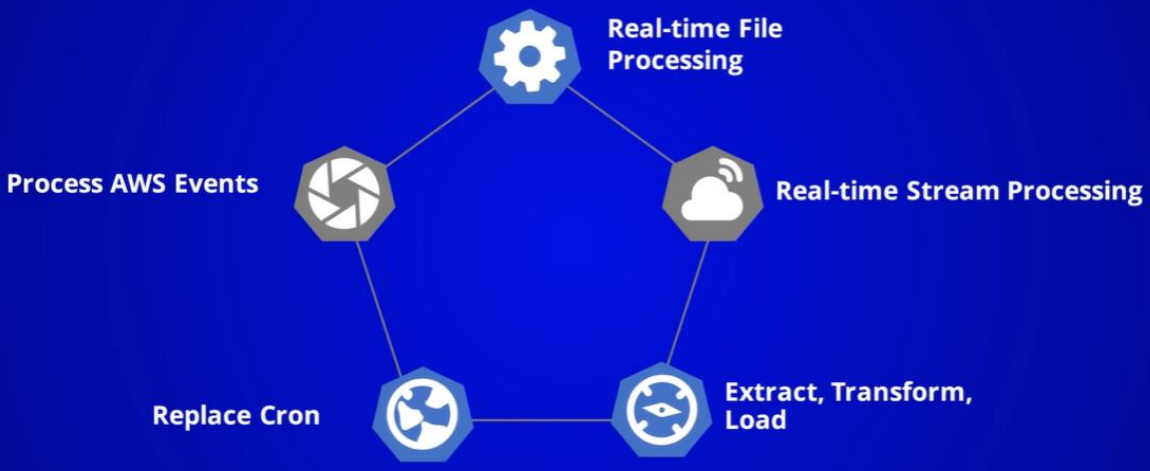
01. Pay only for the compute time you consume

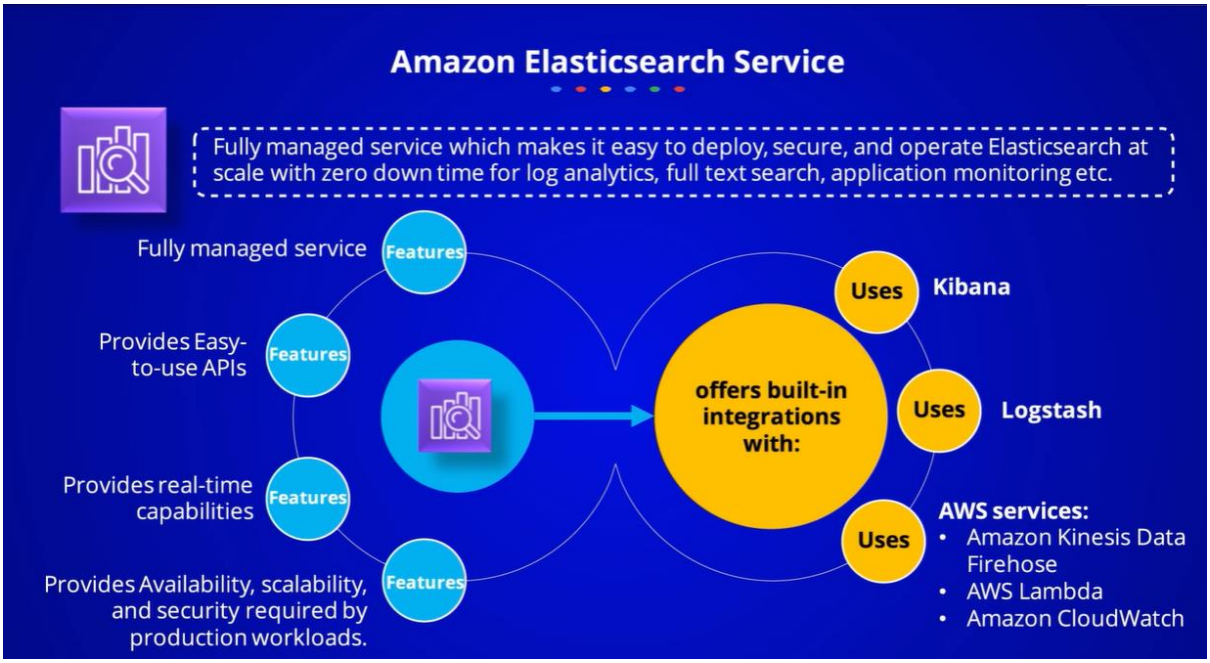
02. Zero Administration: Run code on any application type or backend service with no administration

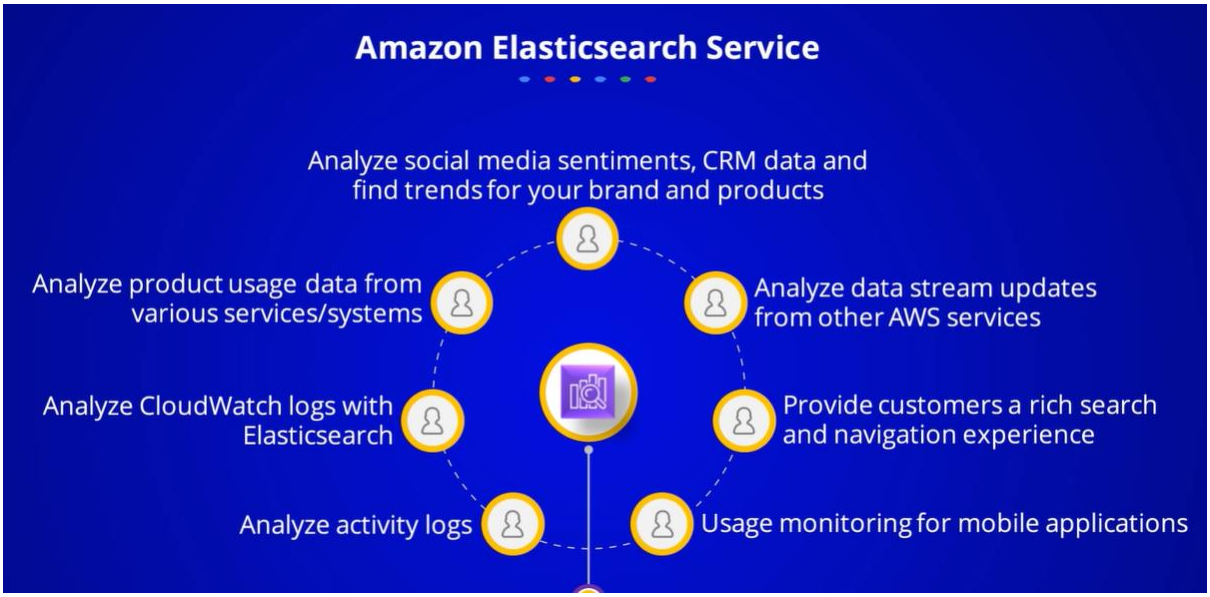
03. Just upload your code: Lambda takes care of everything else to run you code along with managing scalability and availability

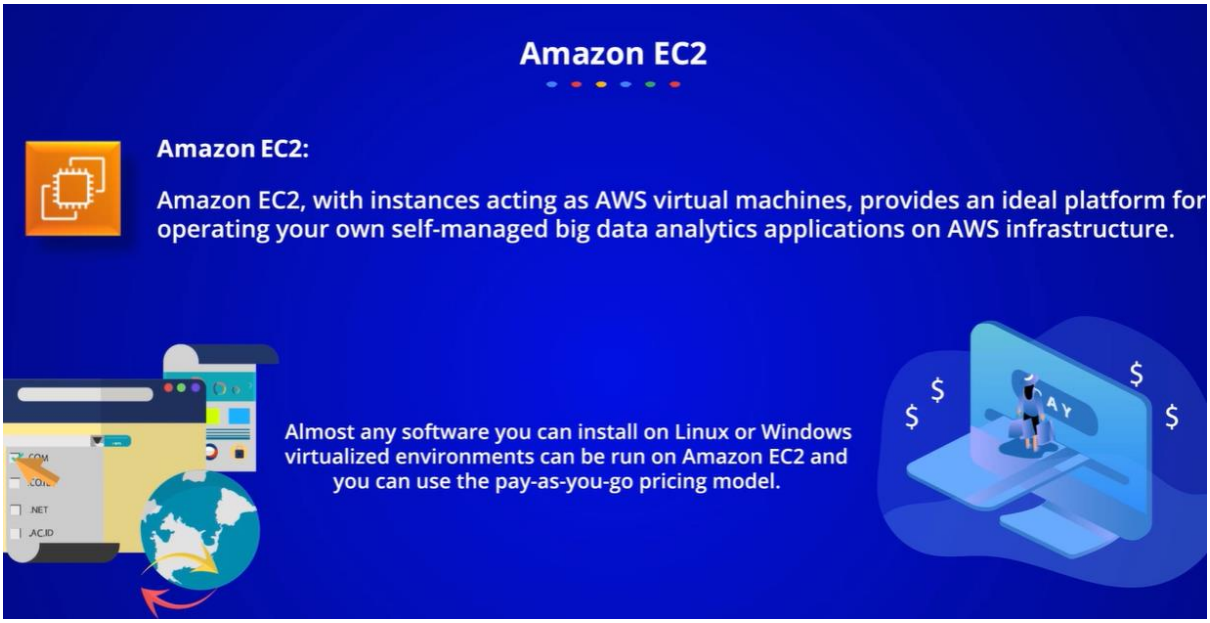
04. Automatic triggering of code: Can be configured to trigger from other AWS services or from any other app (web/mobile)

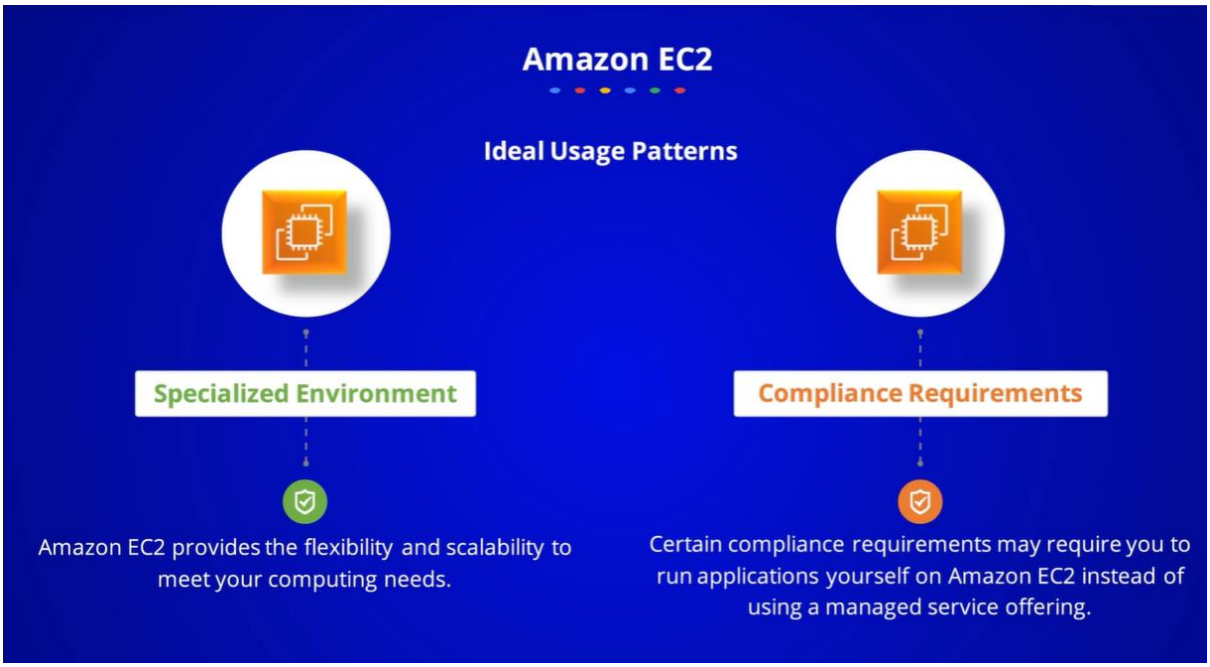
AWS Lambda-Ideal Usage Pattern




Amazon Elasticsearch Service
Fully managed service which makes it easy to deploy, secure, and operate Elasticsearch at scale with zero down time for log analytics, full text search, application monitoring etc.
Fully managed service
Features
Provides Easy-to-use APIs
Features
Provides real-time capabilities
Features
Provides Availability, scalability, and security required by production workloads.
Features
offers built-in integrations with:
Uses
Kibana
Uses
Logstash
Uses
AWS services:
• Amazon Kinesis Data Firehose
• AWS Lambda
• Amazon CloudWatch


Amazon Elasticsearch Service
Analyze social media sentiments, CRM data and find trends for your brand and products
Analyze product usage data from various services/systems
Analyze data stream updates from other AWS services
Analyze CloudWatch logs with Elasticsearch
Provide customers a rich search and navigation experience
Analyze activity logs
Usage monitoring for mobile applications



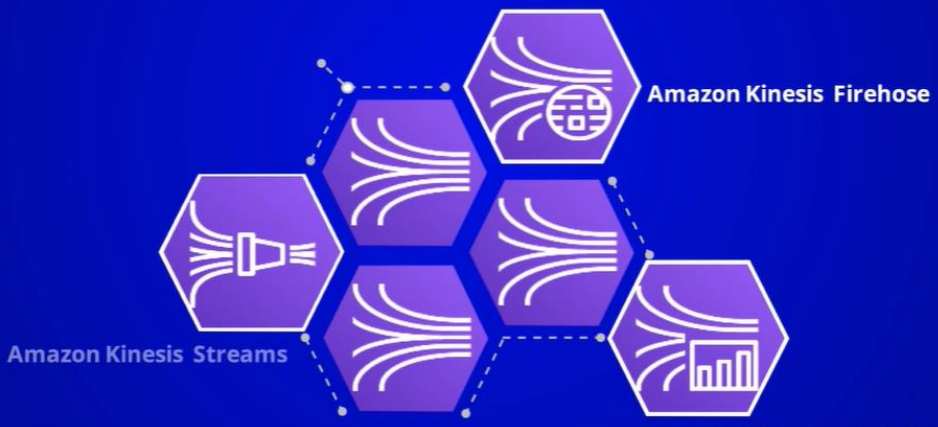


Its used to process collecy and analyze real time and streaming data.

Kinesis Stream Introduction

Amazon Kinesis is a service which is used to process, collect, and analyze real-time and streaming data.

Amazon Kinesis contains three different types of services:

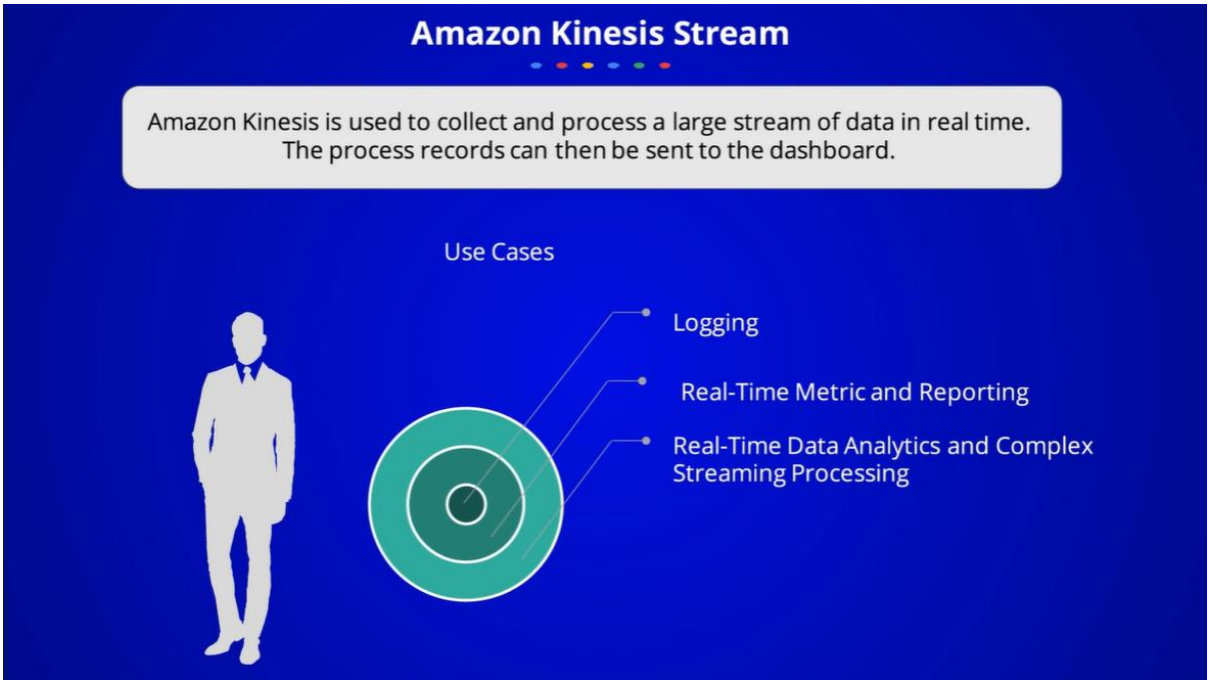


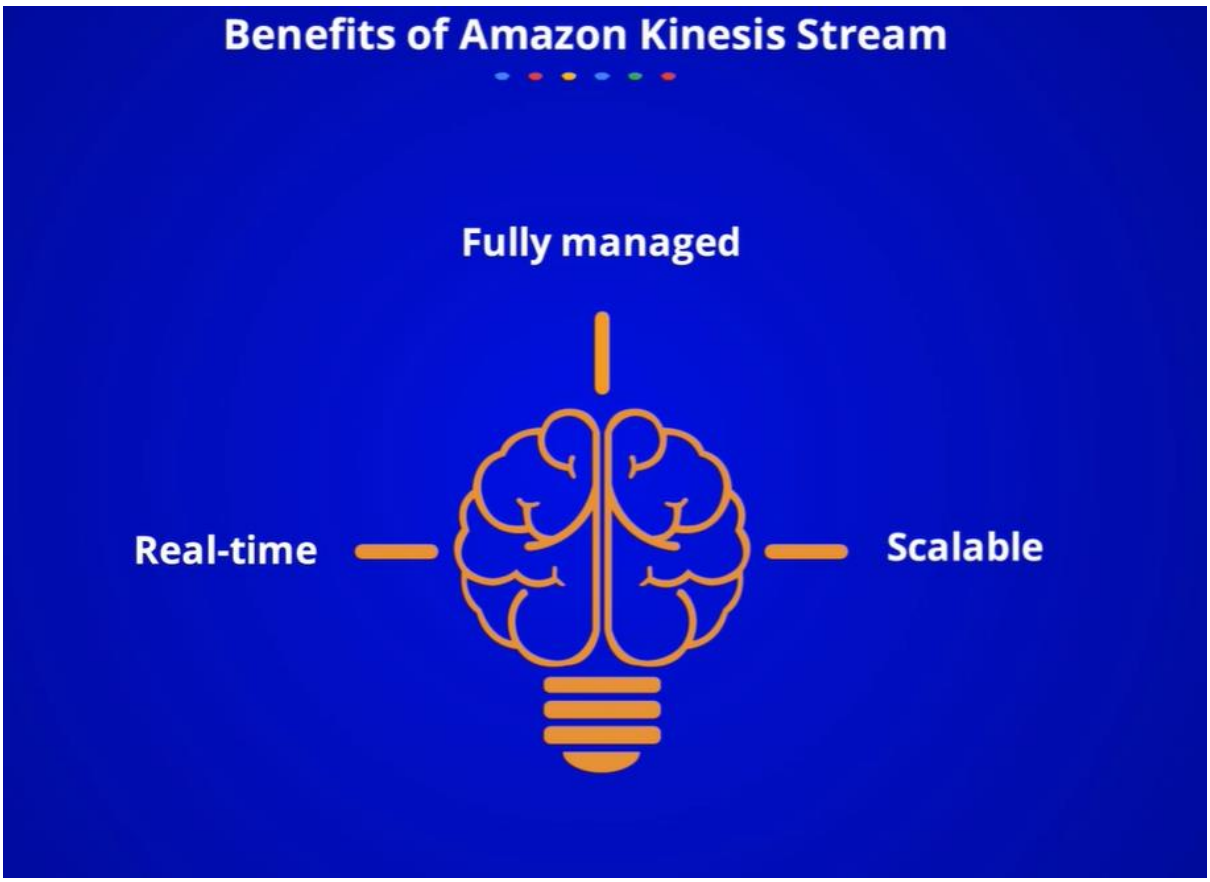
Benefits of Amazon Kinesis Stream

Kinesis Stream is used for collecting and processing a large number of data records in real time.

Kinesis Firehose is used to load large streaming data into AWS services such as RedShift.

Kinesis Analytics is used to process and analyze a large amount of data in real time





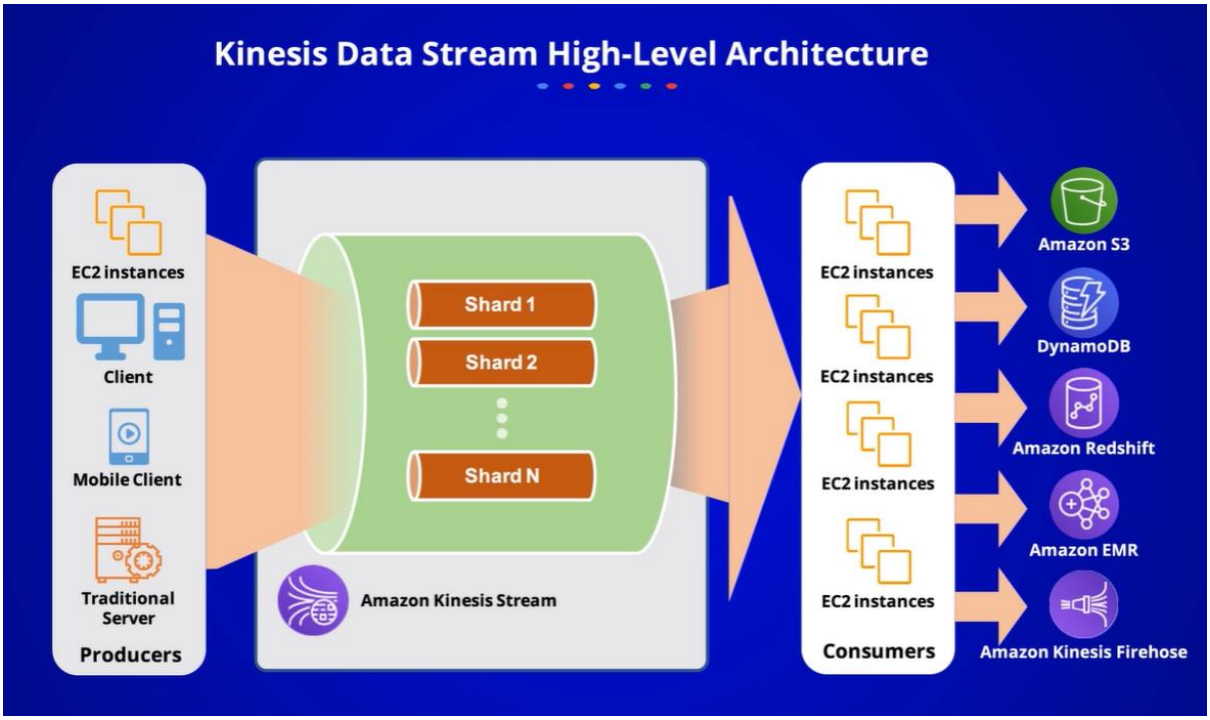
Logging

Real time metric and reporting

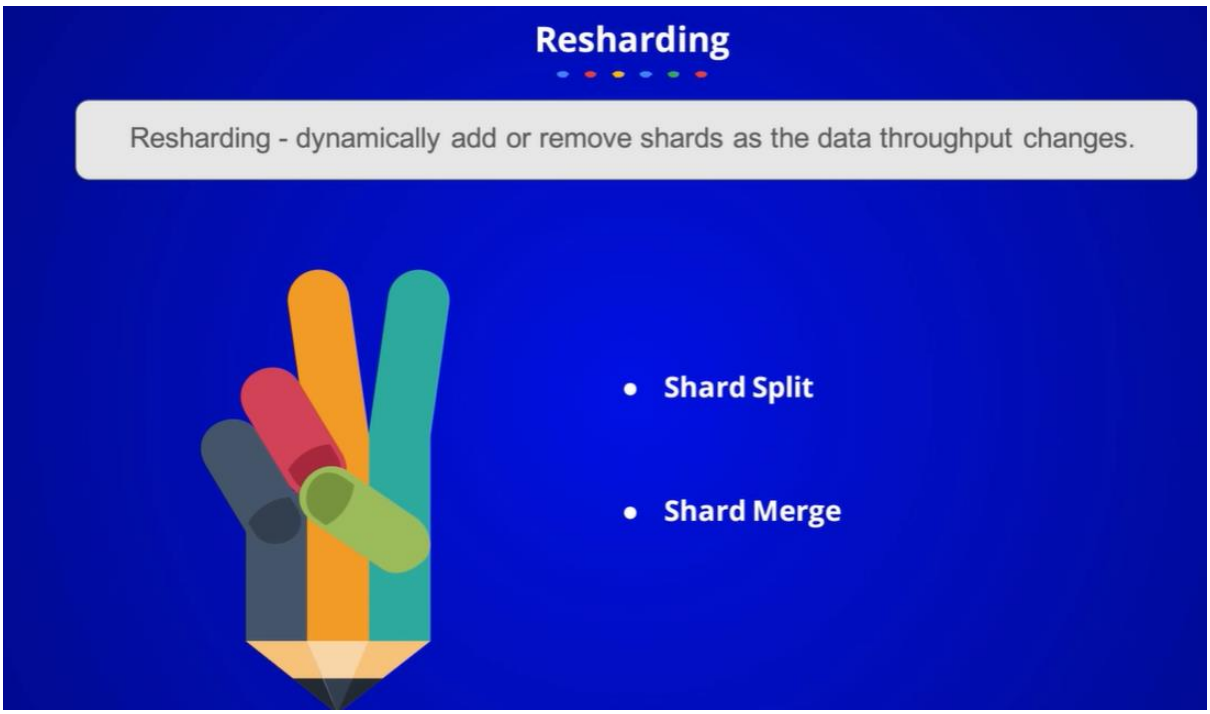
Real time data analystics and complex steaming processing.

Real time, full manage scalable.

- Producer App: add record into streaming.
- Consumer App: read data after the stream and record reading less then one sec.



Producer continuously push the code to kinesis stream, and consumers such as consumer app running consumers app to store in s3, dynamodb, redshift and emr, firehose.

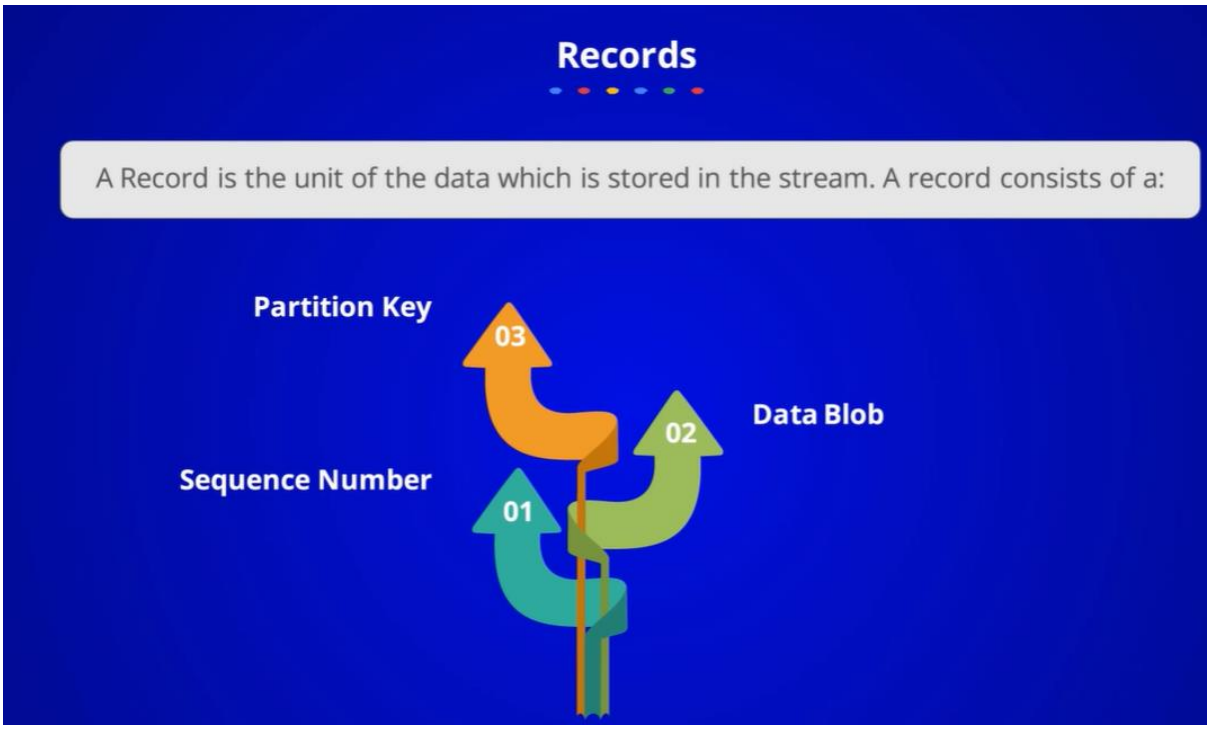


Two types of reshards

Shard split :

Shared split (single shared divided shared.)

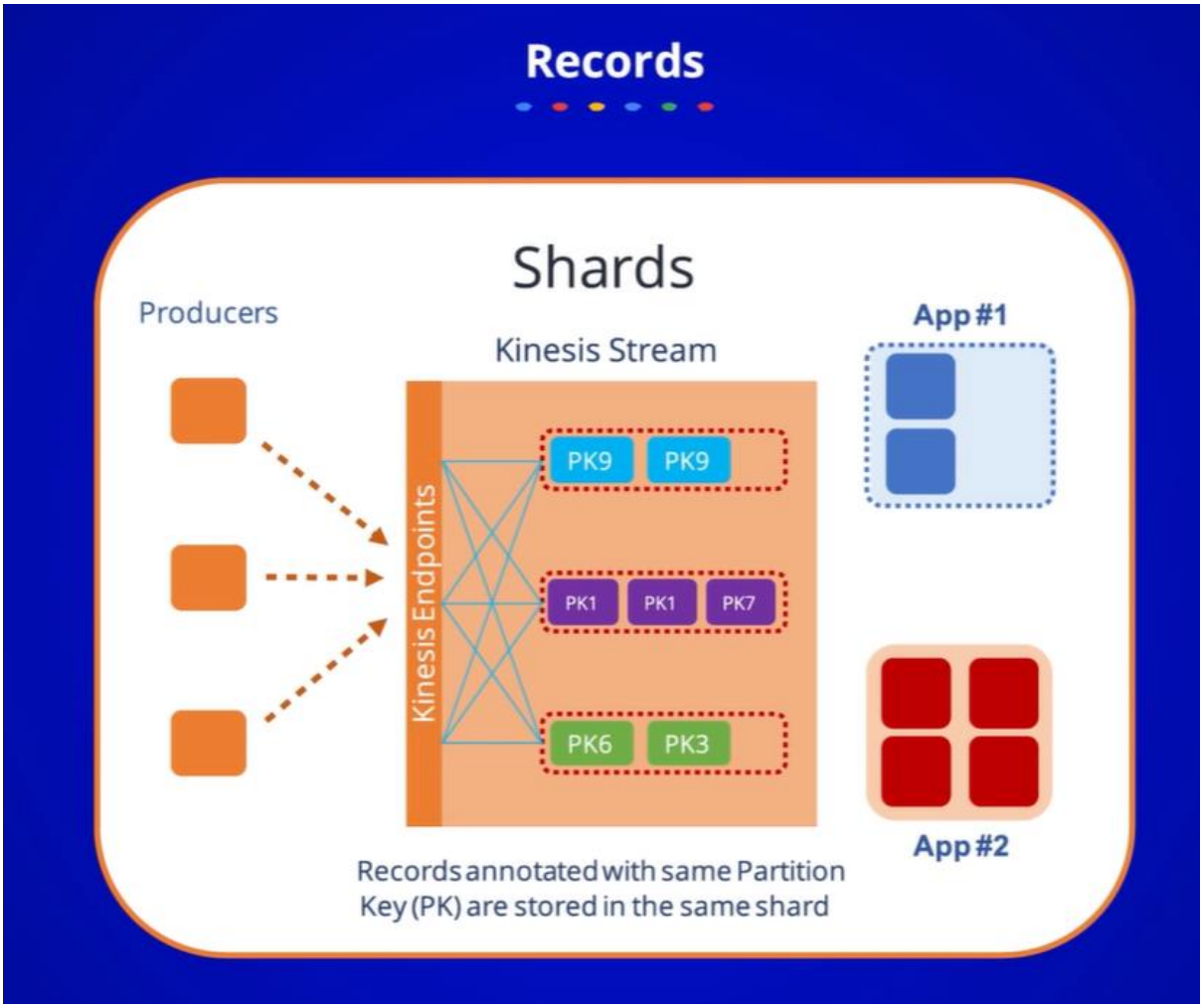
Shard merge(Shared merge two shared in single shard)



- Partition key(256 characters) is used stream distributed data across shards.

Kinesis data stream separated records belong to stream into multiple shards used partition key each associated data determine records which data record belong

- Sequence number:increase over time and specific shard within team.
- Data blog: size equal Or less then 1 Mb.



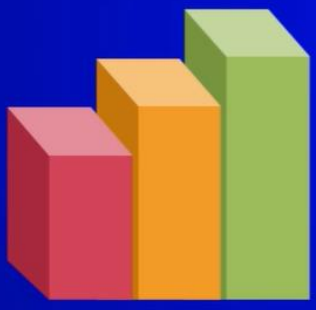
We have three shards contain difference partition keys in each shards and orgizied are belong to sequence no is unique identifiers for records insert into shards.

Data blog: actual data data producer add to stream and max size 1Mb.

- Retention period: max duration after which data added into stream is expired
- Default retention period is 24 hrs by defaults and it can be increased to seven day if required.

Data Producer

Data Producer puts the data records to Kinesis Stream, and this could be something like the web server, a mobile client.



- Amazon Kinesis Stream API
- Amazon Kinesis Producer Library (KPL)
- Amazon Kinesis Agent

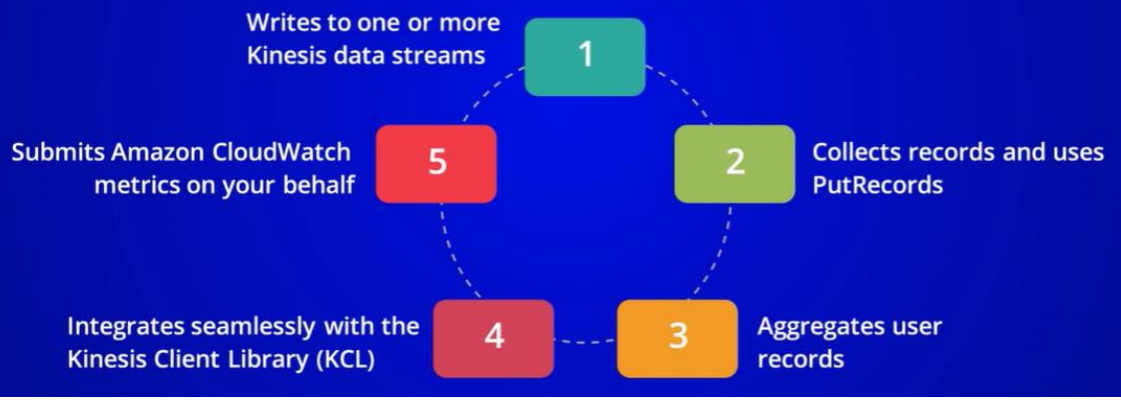
Amazon Kinesis Stream API

- It is a managed service that scales elastically for real-time processing of streaming big data.
- You can develop producers by using the AWS SDK for Java.
- PutRecord (Sends Single data record to the stream at a time.)
- PutRecords (Sends Multiple data records to the stream at a time.)

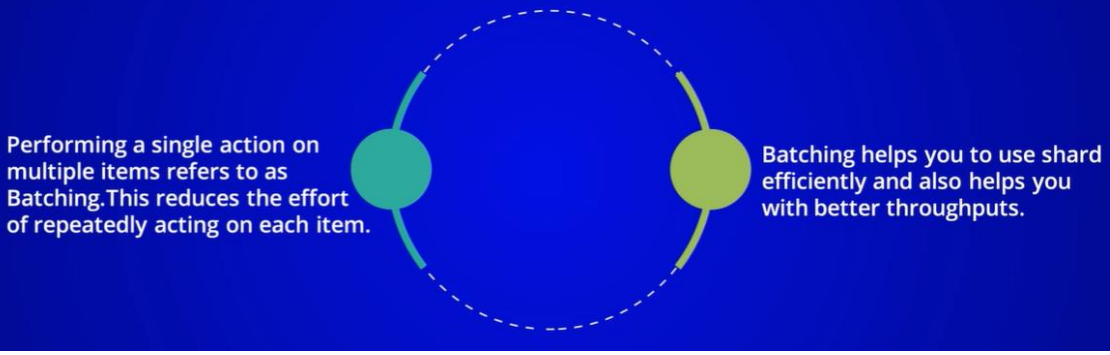
EXAM TIP: You should use PutRecords for most of the applications because it will give you a higher throughput for your data producer.

Amazon Kinesis Producer Library

The KPL is the configurable library which is used to create producer applications that allow the developers to achieve high-write throughput to a Kinesis Stream.



KPL Batching



KPL Batching

The KPL supports two types of batching:

- Aggregation
- Collection

Two types of batching.

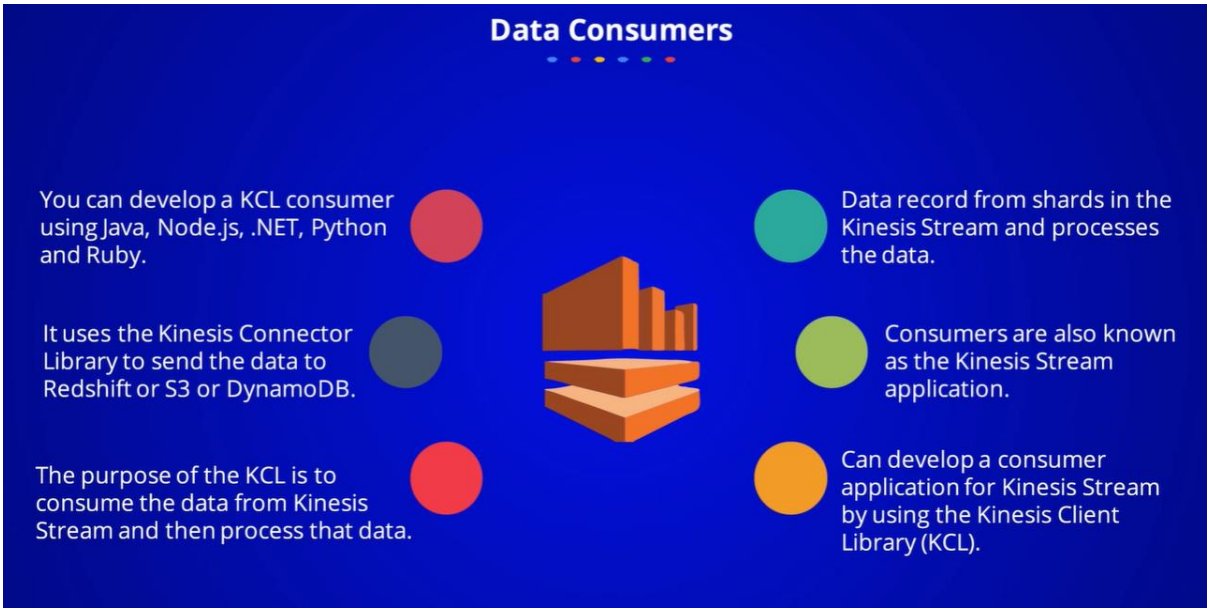
- Aggregation (multiple record into single stream record)
- Collection(using API operation put the record send to multiple kinesis data stream record, one or more shards in kensis data stream.
  By default data stream turn on.

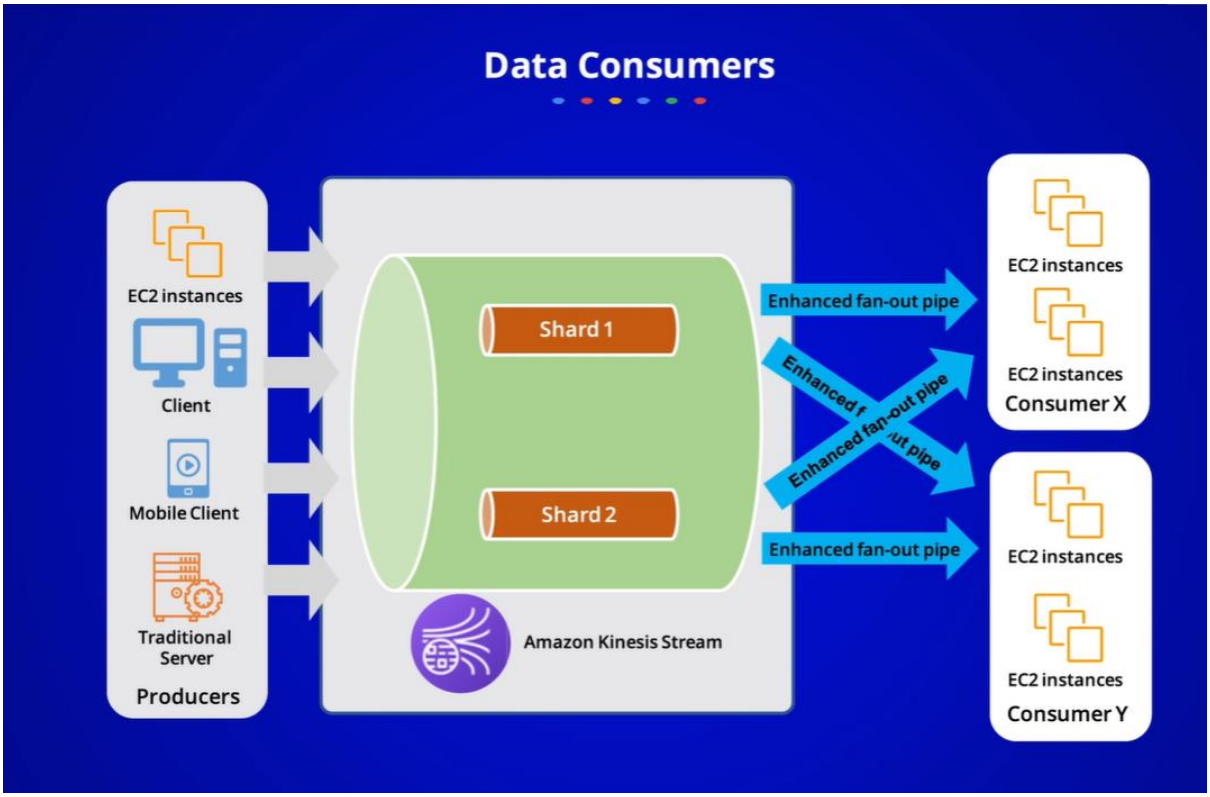
Glacier,s3, redshift and elastic search service.

Kinesis analytics, emr,lambda,ec2 finally go to visualization tools.

Data Consumers:

Consumer used to push data respectively from data stream.



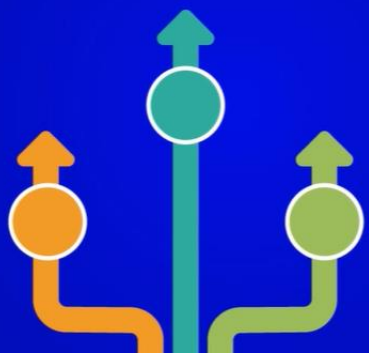


Kinesis Stream Emitting Data to AWS Services

Consumers can emit data to S3, DynamoDB, Elasticsearch, Redshift, and EMR. Since multiple applications can consume data from the stream, you might have a case where some other data may be sent to services like RedShift, but the rest of the data may be archiving in S3 and use object lifecycle process to archive into the glacier.

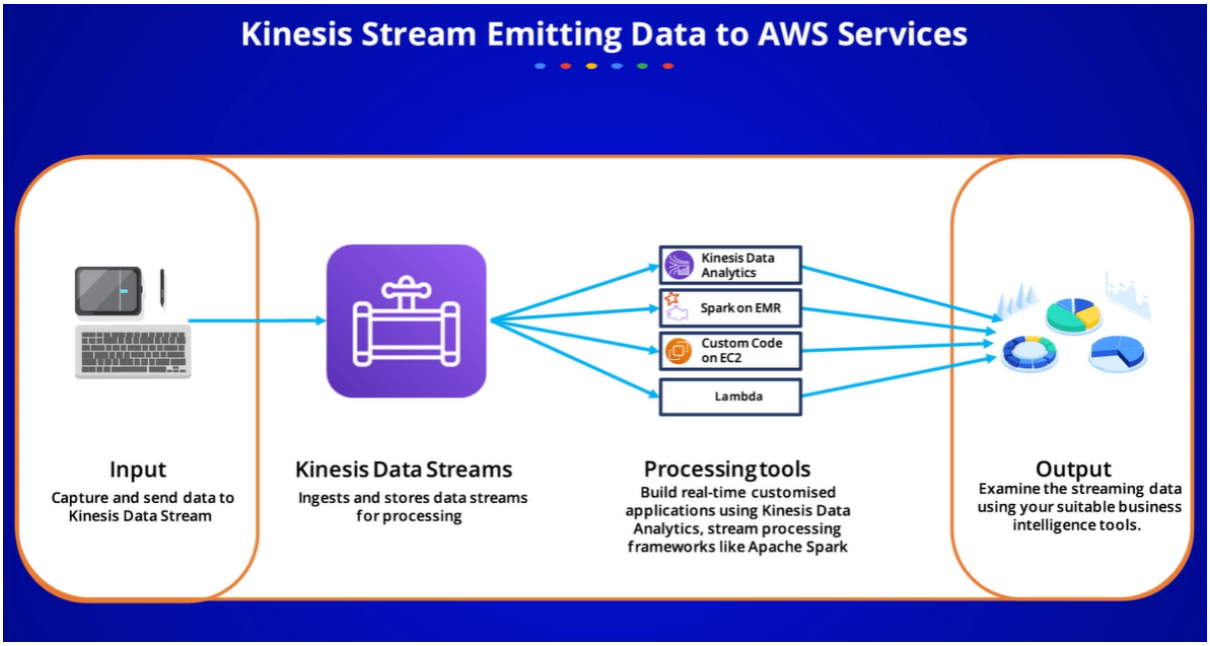
Kinesis Stream Emitting Data to AWS Services

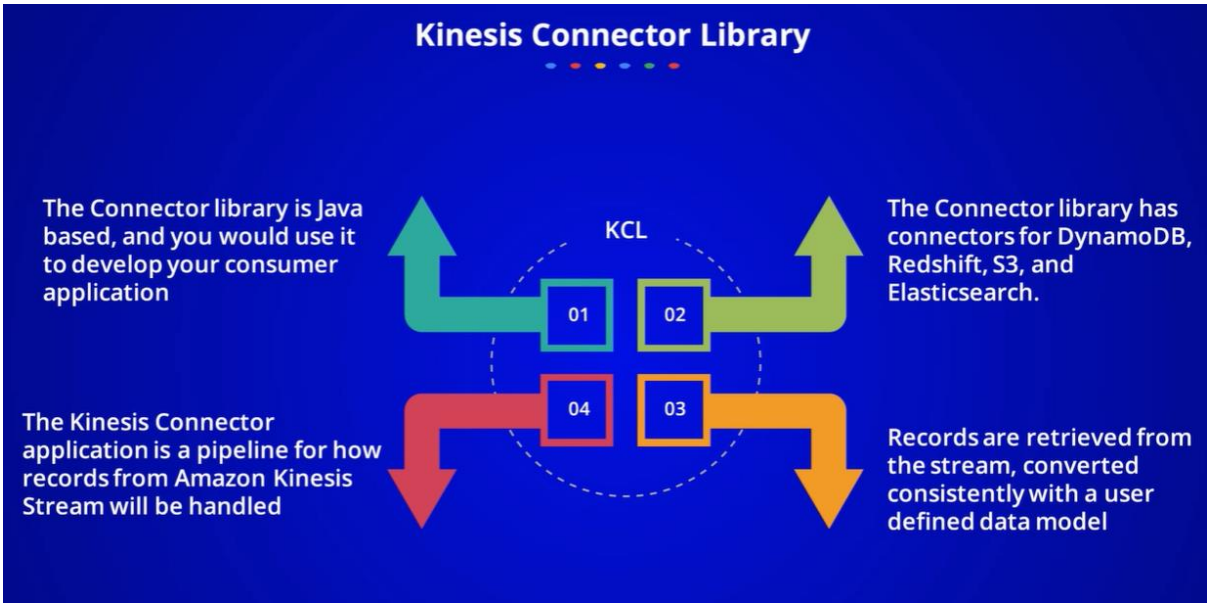
DynamoDB-Data in the stream could be sent to DynamoDB.

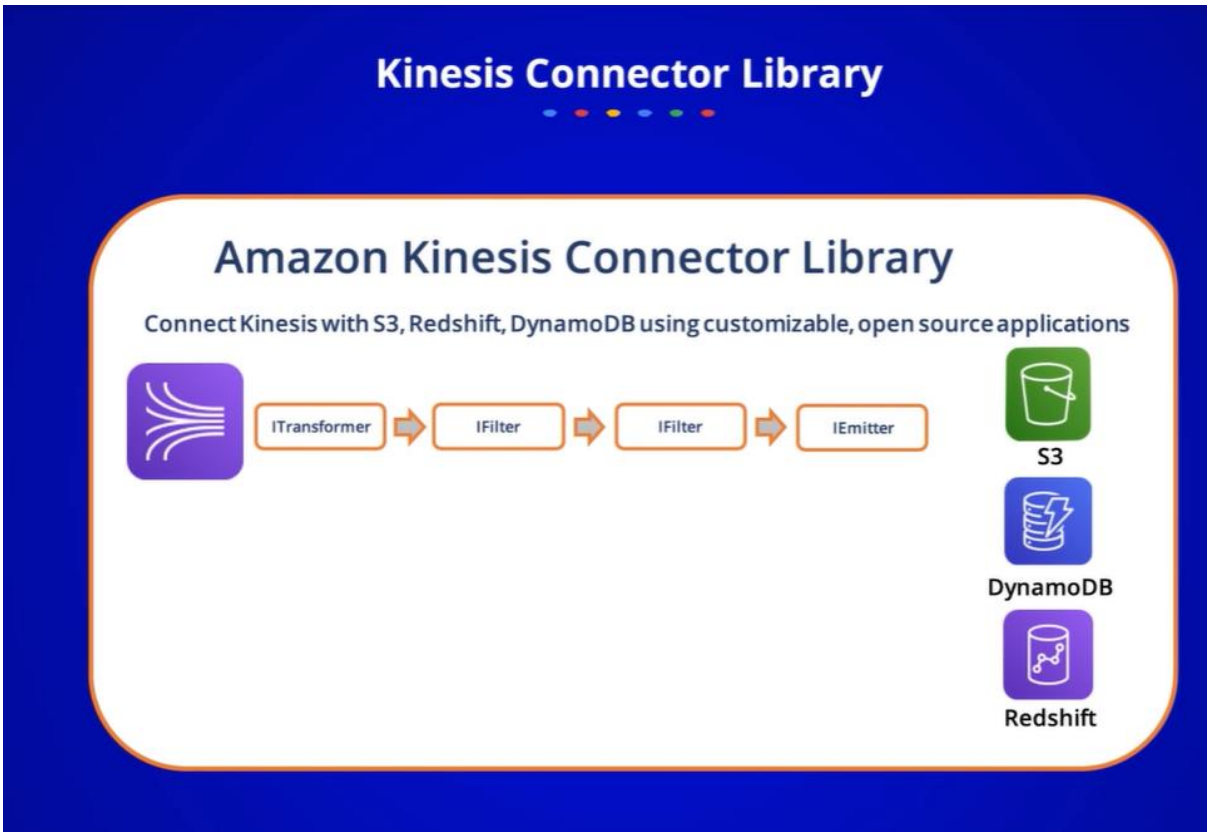


Redshift-If Data needs to be added to data warehouses send it to Redshift.

Elastic search-if you need to make the data from the stream searchable, then send this data to Elasticsearch using the KCL.


Kinesis Stream Emitting Data to AWS Services
Kinesis Data Analytics
Spark on EMR
Custom Code on EC2
Lambda
Input
Capture and send data to Kinesis Data Stream
Kinesis Data Streams
Ingests and stores data streams for processing
Processing tools
Build real-time customised applications using Kinesis Data Analytics, stream processing frameworks like Apache Spark
Output
Examine the streaming data using your suitable business intelligence tools.





Consumer can emitted data to s3, dynamo db, elastic search, redshift and emr.

Since multiple app can consumer data from stream you might have case where some other data may be sent to services like redshift but rest of data may archiving in s3. And use object life cycle process to archive into glacier.

Use case

- Dynamo Db data in stream could be sent dynamo db

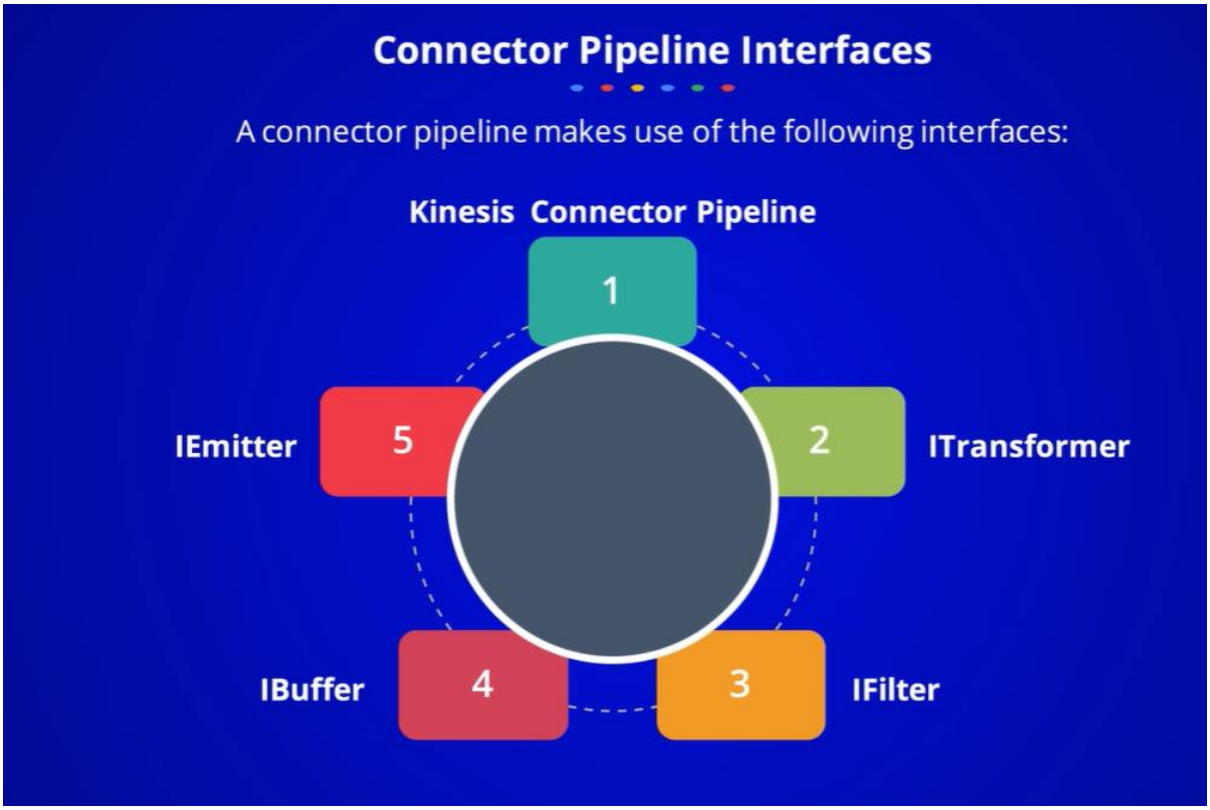
- Like gaming app and dashboard-> read data

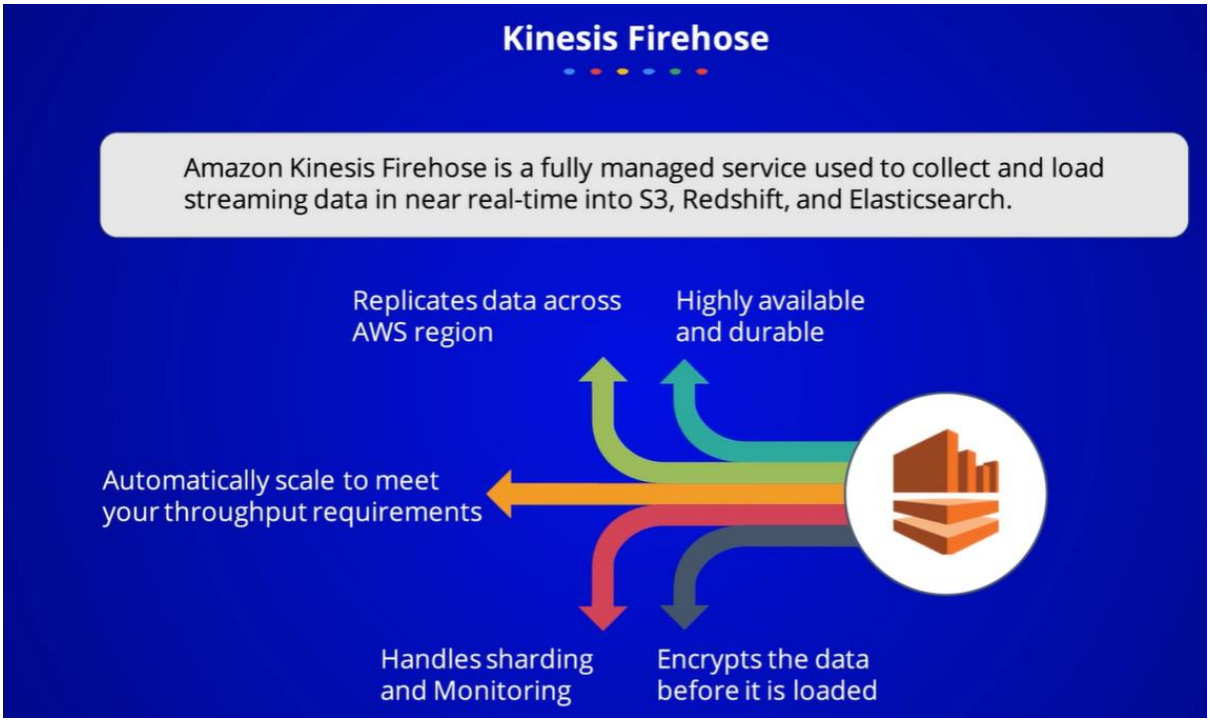
Redshift : if data needs to be added to data warehouse send it to redshift.

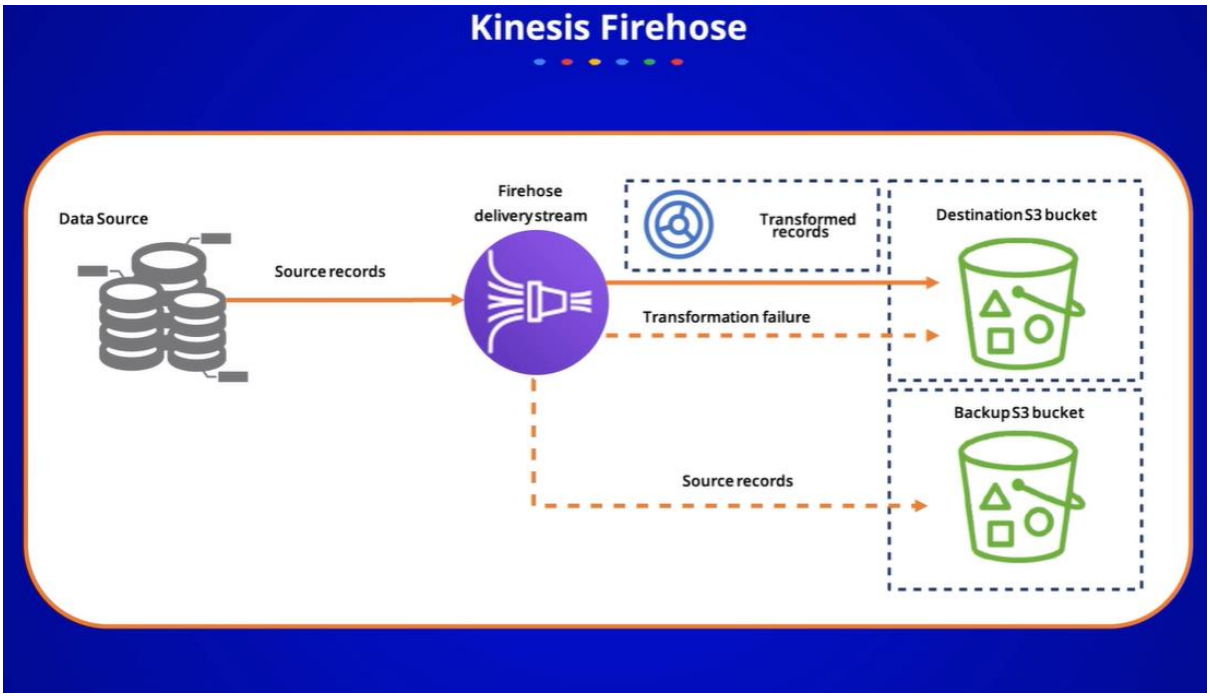
Elastic search: if you need to make data from steam searchable then send this data to elastic search using kcl.

Capture and send data to kiness data stream

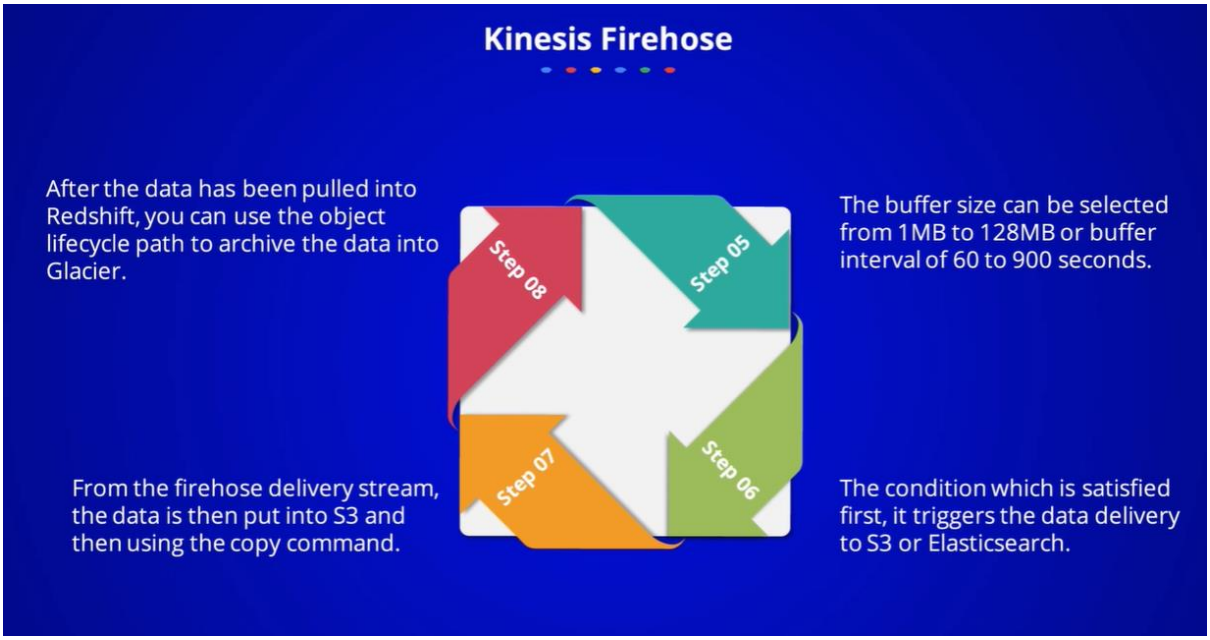
Kinesis data stream(ingests and stores data streams for processing)





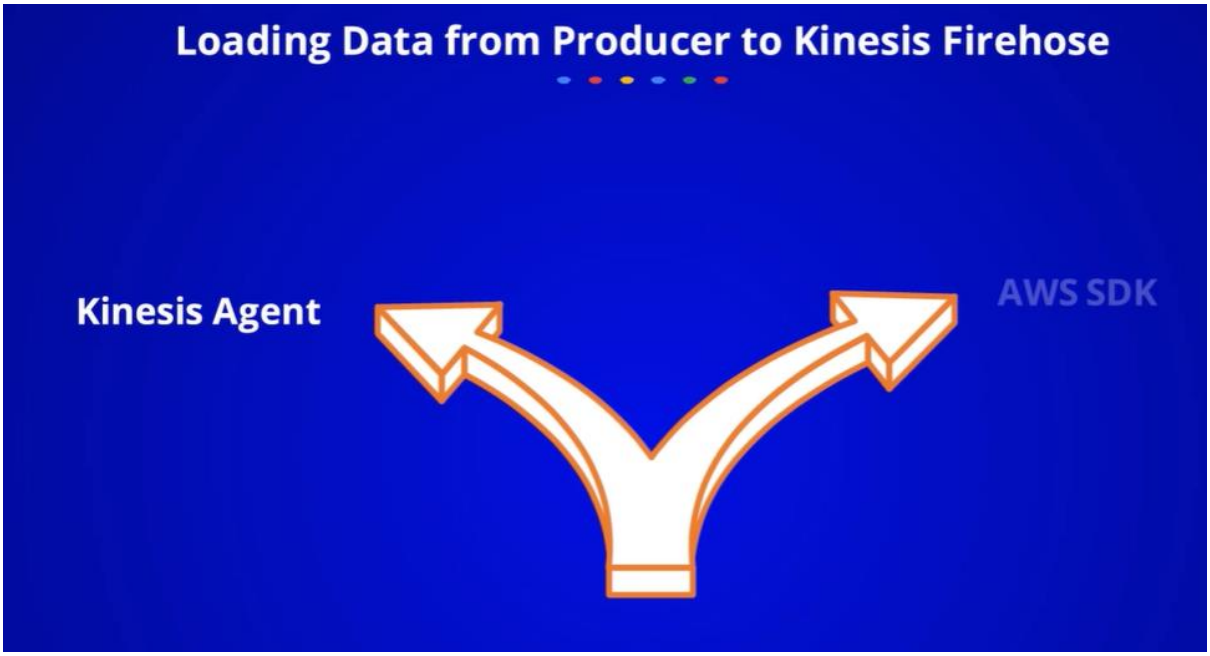


- No need application manage by resource
- We can config data source send source records to fireshose delivery stream.
- It automatically delivery data to destination also configure transformed records.
- If you destination configure s3 delivery s3 bucket.



Web-server(producers): generate logs which send data to kinesis fire hose delivery stream in form of records.

- Each record can be as large as 10000kb
- For each delivery stream you will set buffer size or buffer interval.
- Firehose buffer incoming streaming data to certain size of certain period before delivering it to s3 or elastic search.
- Buffer size can be selected from 1mb to 128mb or buffer interval of 60 to 900 sec.
- Condition which is satisfied first it triggers data delivery to s3 or elastic search.
- From firehose delivery stream the data is then put into s3 and the using copy command.
- After data has been pulled into redshift, you can use the object lifecycle path to archive the data into glacier.



Two types
Kinesis agent
Aws sdk

Agents

1) Standalone java software application offers easy to collect data into data stream and stream kinesis firehose agent continuously set of file and seniers data into stream.
2) Allows you to collect and send data to firehose.
3) Can install it on linux based web server log server or data base web servers.
4) Will also help in cloudwatch metrics.
5) Can preprocess records from monitored files this involved multiple line in single line converting delimeters to json format and converting records to log format to json format. File monitoring destination stream for data
6) Agent can ve monitoring multiple tree and write multiple streams it also preprocess covert records for send to stream or delivery stream.

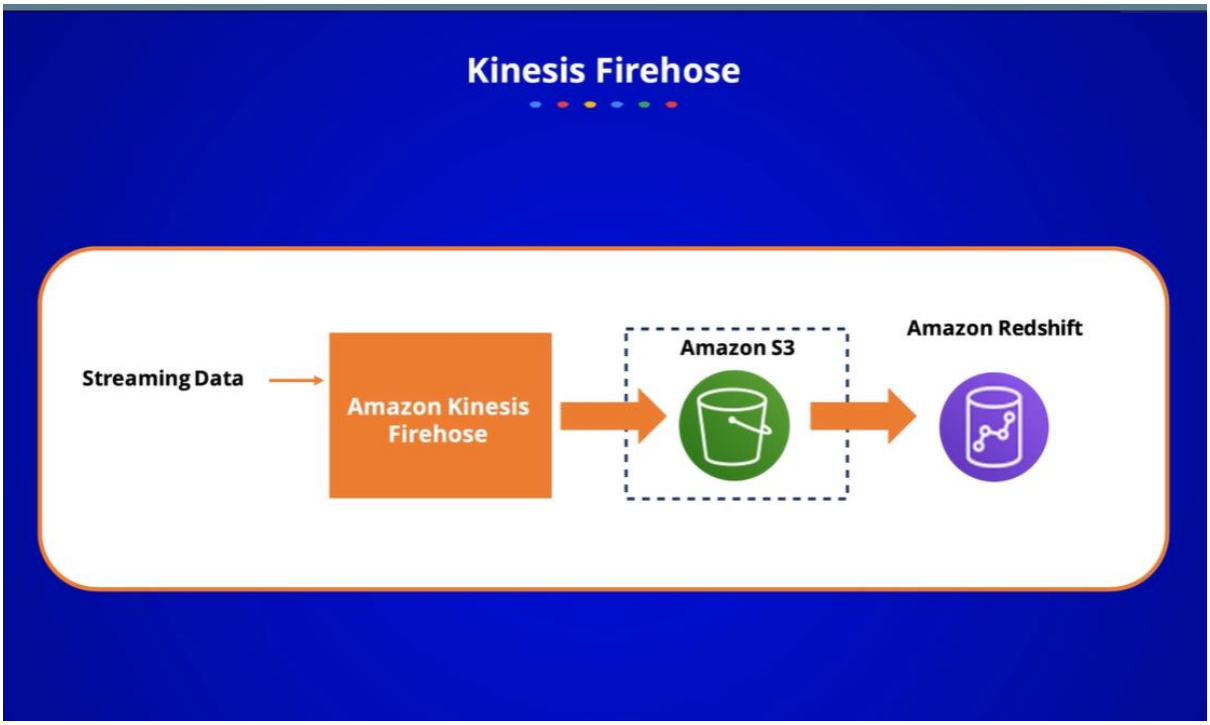
Firehost agent have Two types of records

* **Put records**.

Single data record into amazon kiness firehose delivery stream.

***put record batch**

Use multiple record into delivery stream in single call it can achieve high through put then writing single records



- Don’t write any code just create delivery stream and configure destination data
- Client data into stream using aws API calls and put records batch and data automatically data sent to destination.
- When configure say stream Amazon s3 and firehose and data directly to s3.
- Amazon red shift (destination data shift to Amazon s3 and then red shift copy command execute load data to red shift.

Workout

Put and get records from kinesis data stream

Go to cli

Aws

Step1

```
aws: error: too few arguments
CHMB119868:Downloads samuelasirvat_r$ aws kinesis create-stream --stream-name Foo --shard-count 1
```

Step2

```
CHMB119868:Downloads samuelasirvat_r$ aws kinesis describe-stream --stream-name Foo
{
    "StreamDescription": {
        "KeyId": null,
        "EncryptionType": "NONE",
        "StreamStatus": "CREATING",
        "StreamName": "Foo",
        "Shards": [],
        "StreamARN": "arn:aws:kinesis:us-east-1:559483139917:stream/Foo",
        "EnhancedMonitoring": [
            {
                "ShardLevelMetrics": []
            }
        ],
        "StreamCreationTimestamp": 1572504543.0,
        "RetentionPeriodHours": 24
    }
}
```

Step3 checking stream active using previous cmd

```
CHMB119868:Downloads samuelasirvat_r$ aws kinesis describe-stream --stream-name Foo
{
    "StreamDescription": {
        "KeyId": null,
        "EncryptionType": "NONE",
        "StreamStatus": "ACTIVE",
        "StreamName": "Foo",
        "Shards": [
            {
                "ShardId": "shardId-000000000000",
                "HashKeyRange": {
                    "EndingHashKey": "340282366920938463463374607431768211455",
                    "StartingHashKey": "0"
                },
                "SequenceNumberRange": {
                    "StartingSequenceNumber": "49600945676829605166972608541704302084528788124637069314"
                }
            }
        ],
        "StreamARN": "arn:aws:kinesis:us-east-1:559483139917:stream/Foo",
        "EnhancedMonitoring": [
            {
                "ShardLevelMetrics": []
            }
        ],
        "StreamCreationTimestamp": 1572504543.0,
        "RetentionPeriodHours": 24
    }
}
```

Step4: check stream name

```
CHMB119868:Downloads samuelasirvat_r$ aws kinesis list-streams
{
    "StreamNames": [
        "Foo"
    ]
}
```

Step5: put the record using partitions

```
CHMB119868:Downloads samuelasirvat_r$ aws kinesis put-record --stream-name Foo --partition-key 123 --data testdata
{
    "ShardId": "shardId-000000000000",
    "SequenceNumber": "49600945676829605166972608706057767261137630198377742338"
}
```

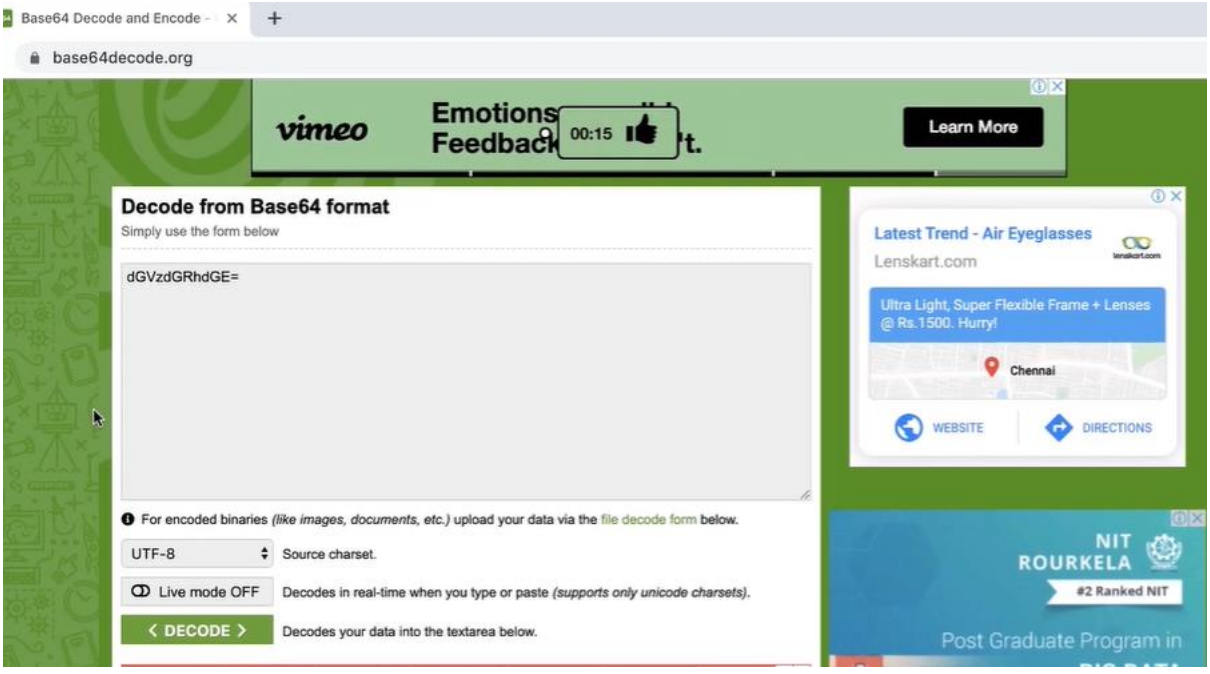
Step6:

```
CHMB119868:Downloads samuelasirvat_r$ aws kinesis get-shard-iterator --shard-id shardId-000000000000 --shard-iterator-type TRIM_HORIZON --stream-name Foo
{
    "ShardIterator": "AAAAAAAAAAH0zPVU4pRH2ocY5sjOYPbNntbU8eC9KtvpTZvCq9JTaKtzGCl5jpadoJ2VCVxxmcvL4mHcBtjQNYlw0hPSRbSIyduYXe6nzfFBwlX1clLjQULdq3S+4uCIKkjwYUeX
g2609F9YONRJMi5YDo39lRilHpVKZXif1LNQo0KvbLflIH+apnC9u/qIBn+V7QN22876GzJ5g9kKlT4TJRtTEL4H"
}
```

Step7

```
CHMB119868:Downloads samuelasirvat_r$ aws kinesis get-records --shard-iterator AAAAAAAAAAH0zPVU4pRH2ocY5sjOYPbNntbU8eC9KtvpTZvCq9JTaKtzGCl5jpadoJ2VCVxxmcvL4mH
cBtjQNYlw0hPSRbSIyduYXe6nzfFBwlX1clLjQULdq3S+4uCIKkjwYUeXg2609F9YONRJMi5YDo39lRilHpVKZXif1LNQo0KvbLflIH+apnC9u/qIBn+V7QN22876GzJ5g9kKlT4TJRtTEL4H
{
    "Records": [
        {
            "Data": "dGVzdGRhdGE=",
            "PartitionKey": "123",
            "ApproximateArrivalTimestamp": 1572504625.14,
            "SequenceNumber": "49600945676829605166972608706057767261137630198377742338"
        }
    ],
    "NextShardIterator": "AAAAAAAAAAE8TeXj3p+GMpe8YEpZKv6/vGta9/dcSNUHlKR3VR3zVsqo7V73v+VyzuEBb+T38UnNjTDFsiiy4p0C1wGIiPiv856wabqCicIZtfQ2YCQDOg73unZQH7quMlm6
ehWeKoYzRAuzqij1Q6g83pdyZtmODKZQ4fPpjI8zkBK9R5UhU2GK9dyGh6f361YHgnULBCUvpjj8YYO+R5EUoMJhdjEe",
    "MillisBehindLatest": 0
}
```

Step8: decoding



Result: testdata (user enter the data)

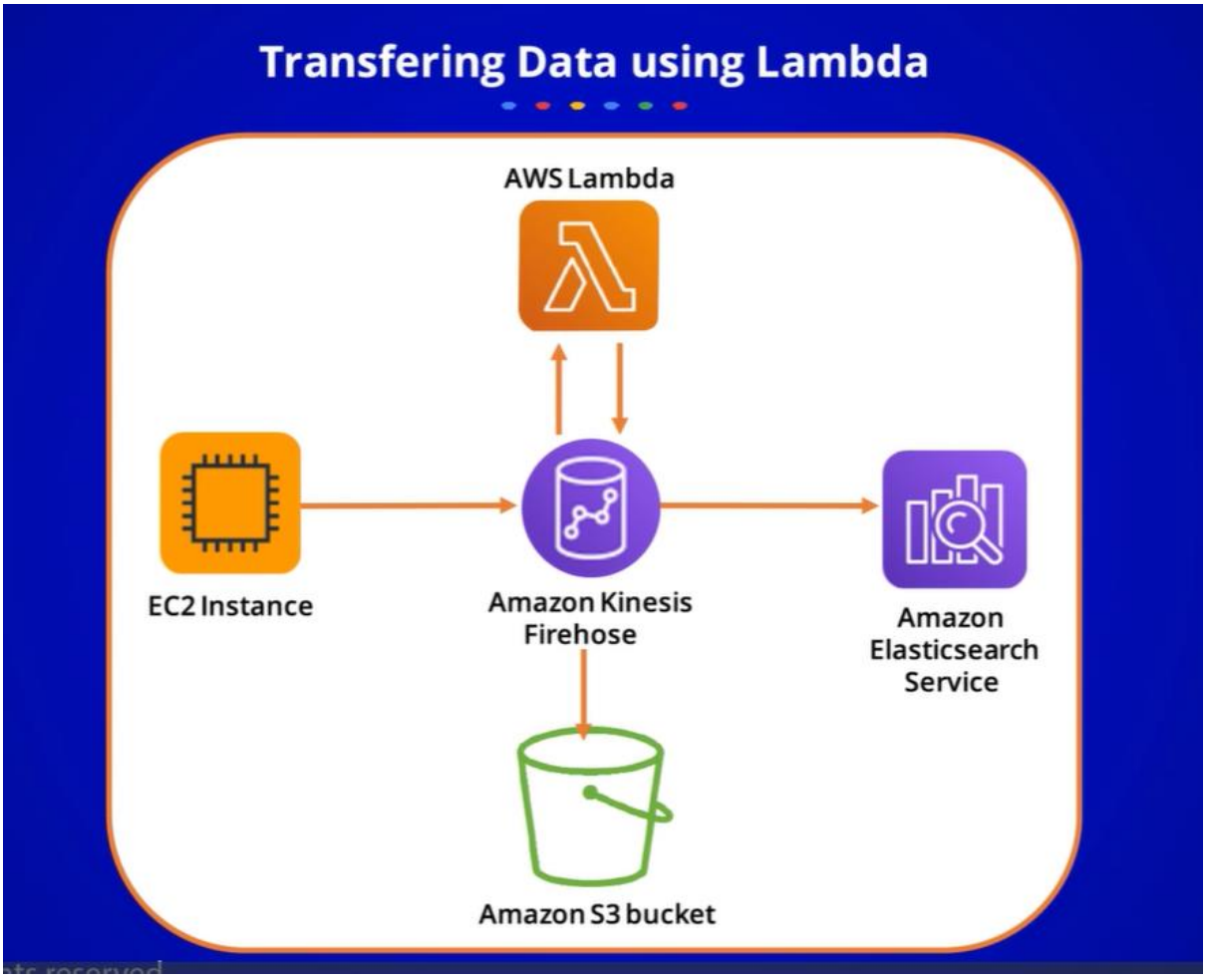
Step 9: delete stream name

```
CHMB119868:Downloads samuelasirvat_r$ aws kinesis delete-stream --stream-name Foo
CHMB119868:Downloads samuelasirvat_r$
```

Step9.1

```
CHMB119868:Downloads samuelasirvat_r$ aws kinesis describe-stream --stream-name Foo
{
    "StreamDescription": {
        "KeyId": null,
        "EncryptionType": "NONE",
        "StreamStatus": "DELETING",
        "StreamName": "Foo",
        "Shards": [],
        "StreamARN": "arn:aws:kinesis:us-east-1:559483139917:stream/Foo",
        "EnhancedMonitoring": [
            {
                "ShardLevelMetrics": []
            }
        ],
        "StreamCreationTimestamp": 1572504543.0,
        "RetentionPeriodHours": 24
    }
}
```

- **Lambda example of service less computing which allows you run code without provision managing servers.**
- **Lambda you can run code anytype of app backend server all zero admission**
- **Just uploaded code lambda take care any things required to run scala code with high availability.**
- **You setup target automatically other aws services and call directly from any web or mobile app**



Flow:

- **Firehose incoming data buffering upto 3mb or buffering size specific delivery stream.**
- **Firehose involved specific lambda function each buffer batch sync.**

- **Transform data from lambda to firehouse for buffering**
- **Transfer data send to destination with specific buffering size and buffering interval reach.**

Failure handling

- **If data transformation fails unsuccessfully processed records are sent to your s3 bucket in processing failed folder.**
- **Fire hose retries invocation for three time an discard batch of records if retry is not successes**
- **Retry delivery period is 24 hours**
- **Invocation errors and records that have status of processing failed can be emitted to cloud watch logs.**
- **If data transfer fails to unsuccessful process record send to s3 bucket into process fail folder.**

Data flow:

For s3 destination streaming records are delivered on your s3 bucket if data transformation is enabled, you could optionally backup supply data to any other s3 bucket.

Red shift destination:

Streaming records is kept in s3 bucket. To load data s3 bucket kinesis. Data issues an amazon red shift copy command into red shift cluster

In call transformation is enable you many optionally backup supply information every other Amazon s3 bucket.

Data delivery frequency in s3:

- **In s3 buffer size (1mb to 128mb)**
- **Buffer interval 60 -900 sec**
- **Fire hose is capable of raising the buffers size dynamically to catch upto issues where data delivery to specific app is following behind.**
- **Retry delivery to your destination for upto 24 hrs.**

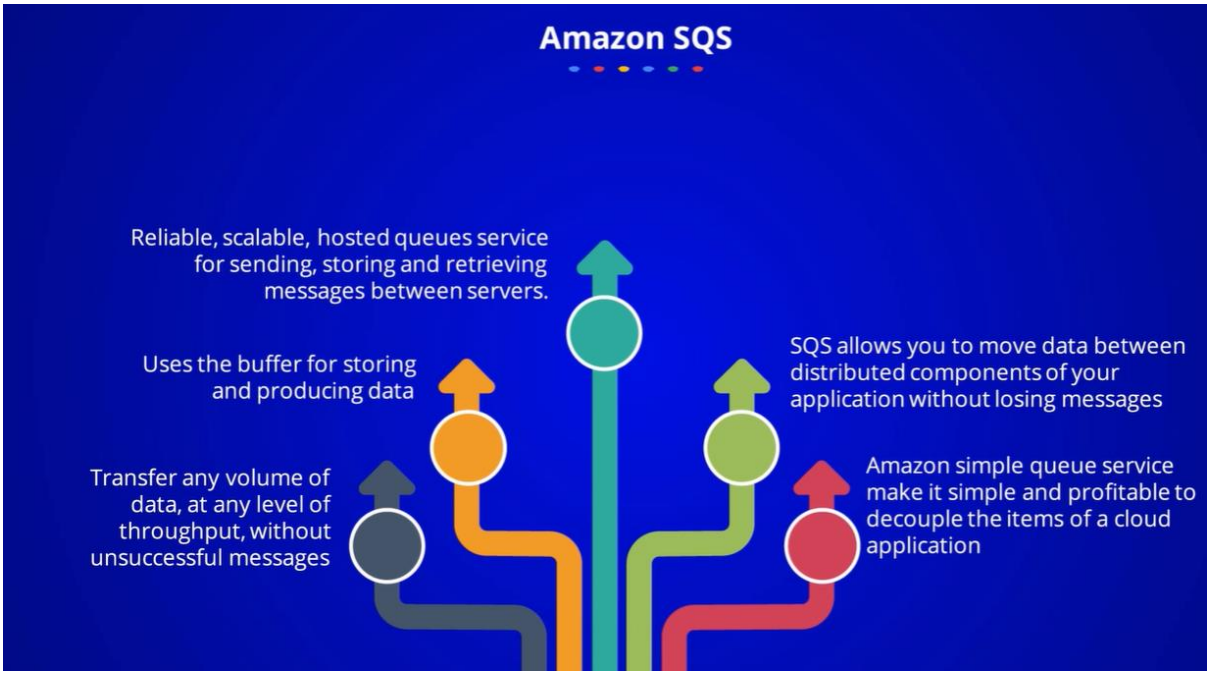
Data delivery frequency in Red Shift

- **Red Shift frequency will depend on how fast red shift cluster finishes copy command.**
- **Retry in red shift 0 to 7200 sec.**
- **Fire hose will issues new copy cmd automatically when previous copy cmd has completed running.**

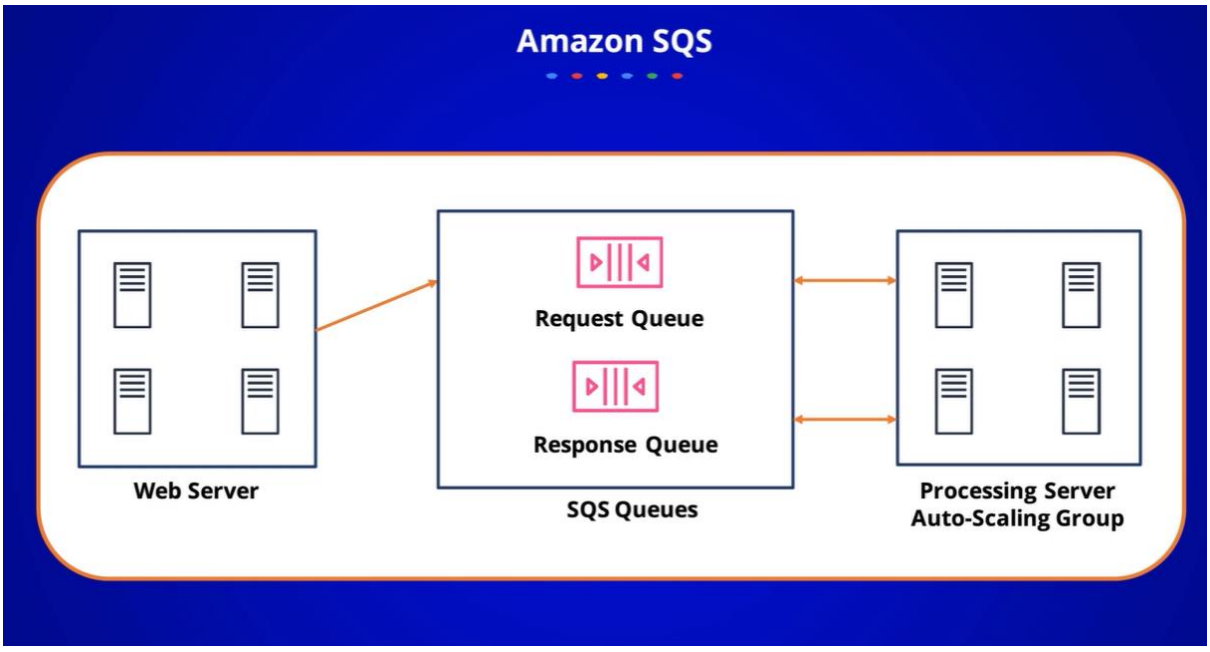
Data delivery frequency in elastic search

- **Elastic search also depends on buffer size (1Mb to 100Mb) and buffer interval (60- 90 sec)**
- **Fire hose can raise size of buffer dynamically. If data delivery to elastic search is following behind.**

- **You can also specific retry duration from zero to 7200 sec.**

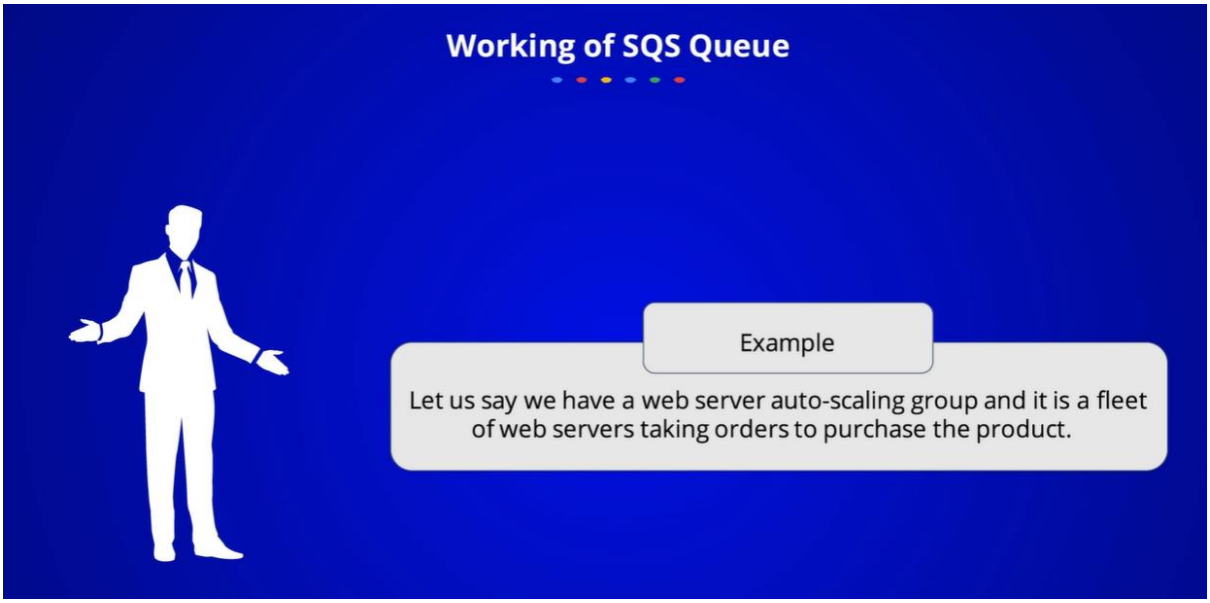


- Standards for simple queue services its reailable, scalable, post queue sending storing and reterving message between server.

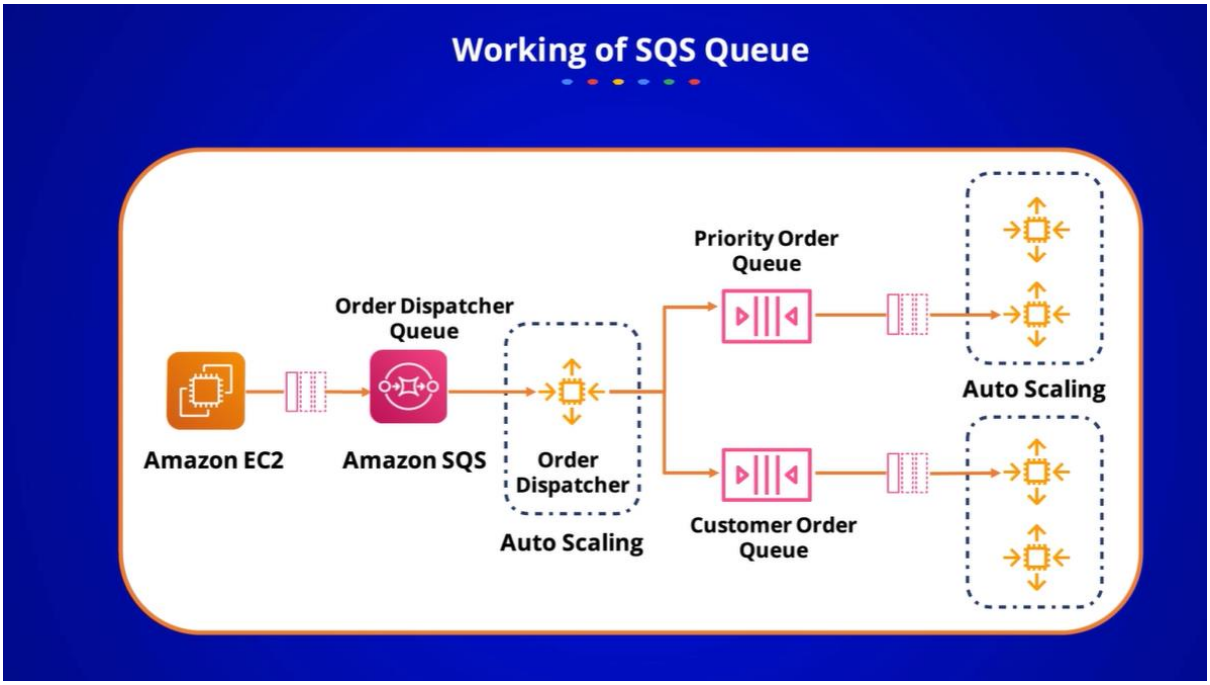


- Web server sub multiple reader/write which mean many complent share in single queue.

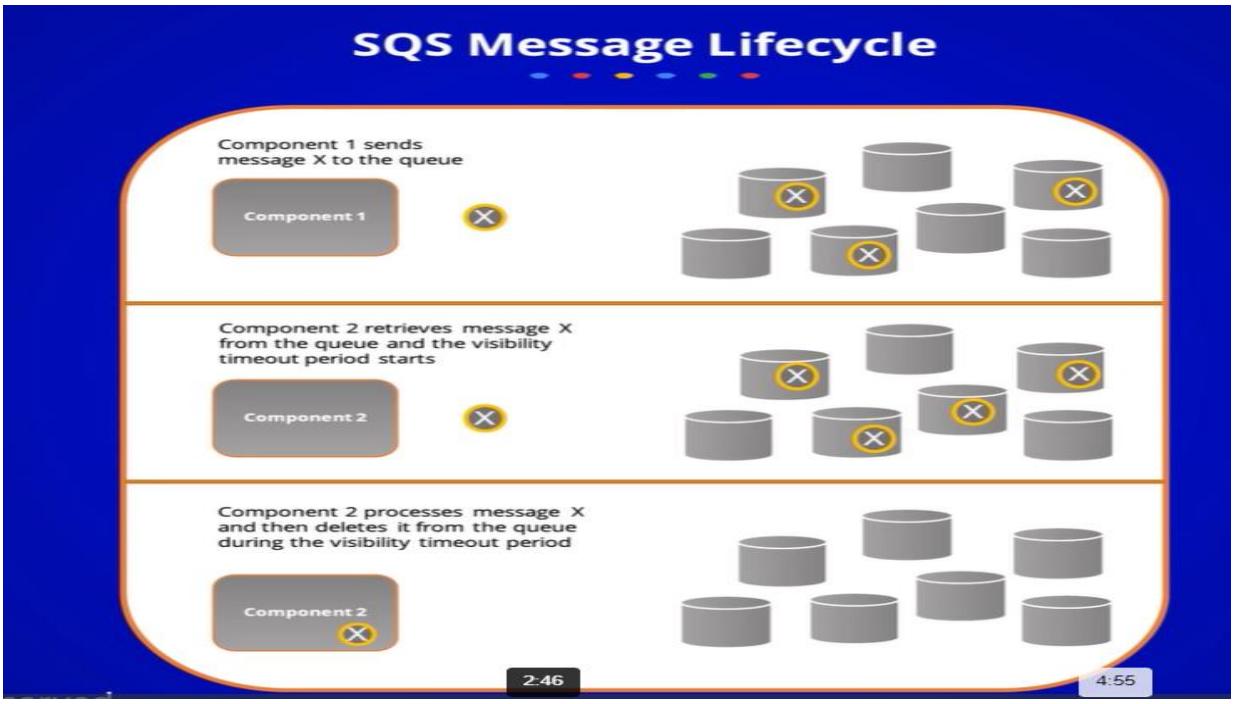
- Aws works fifo concept first in first out an fifo queue order send in massage send once it remains available contain customer predict need that data
- Queue contain many region massage can be contain queue upto 14 dayd this massage send read simultaneously
- Sqs following long following its reduce minus cost while retrieve msg quickly and possible
- When queue is empty long fall request wait up to 20 sec next msg arrived.

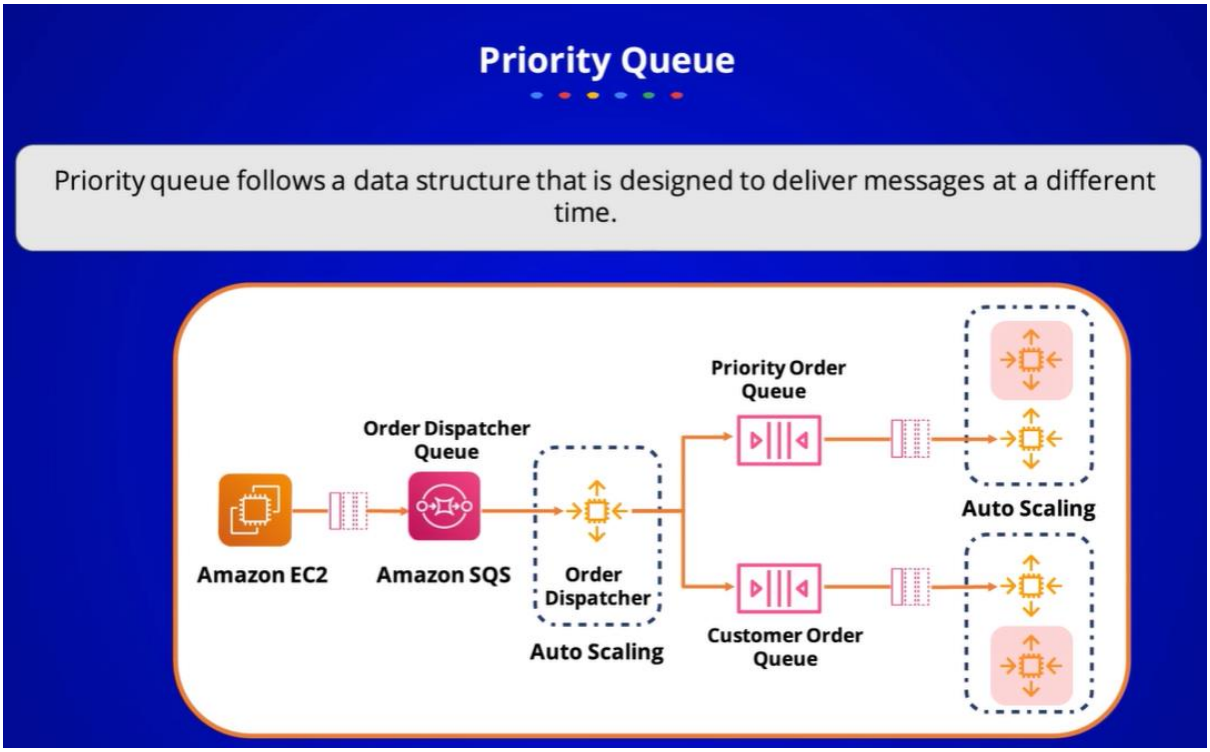


- Web app place order into a sqs request queue this order set in queue until picked one free processing server. The server process server then black process to queue then customer.
- Queue can determine load on application. You can use length of collect determined queue and could watch determined load and could watch integrate with sqs queue to collect and view analyst metrics and you can several different discussion based on data.



When web application place order sqs queue this order pickup one of the free processing server. The process order send back to priority order queue and send to customer order queue. The advantage of queue auto scaling scale back based on item sqs queue

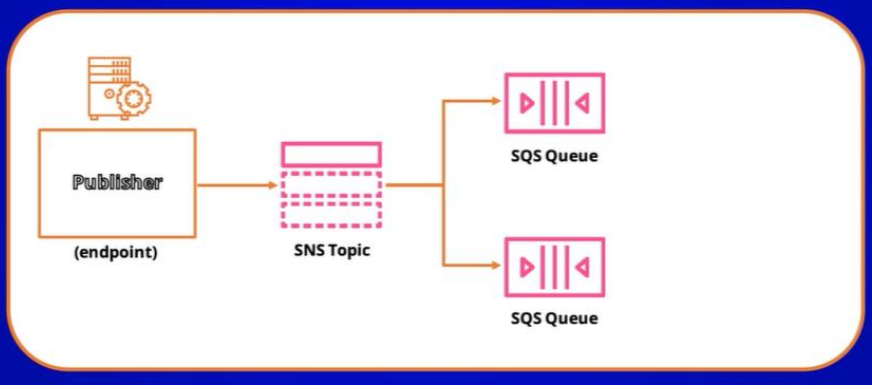




Massage process two types high priority and low priority based on queue.

Priority Queue

The fan-out scenario involves SNS to send multiple topics using the SQS queue at the same time to decouple the processing further. It includes SNS messages which are sent to multiple topics, are replicated and pushed back in SQS queue to allow parallel processing.



AWS IoT

The AWS IoT service is a managed cloud platform that connects devices easily and securely to interact with cloud applications and other devices.

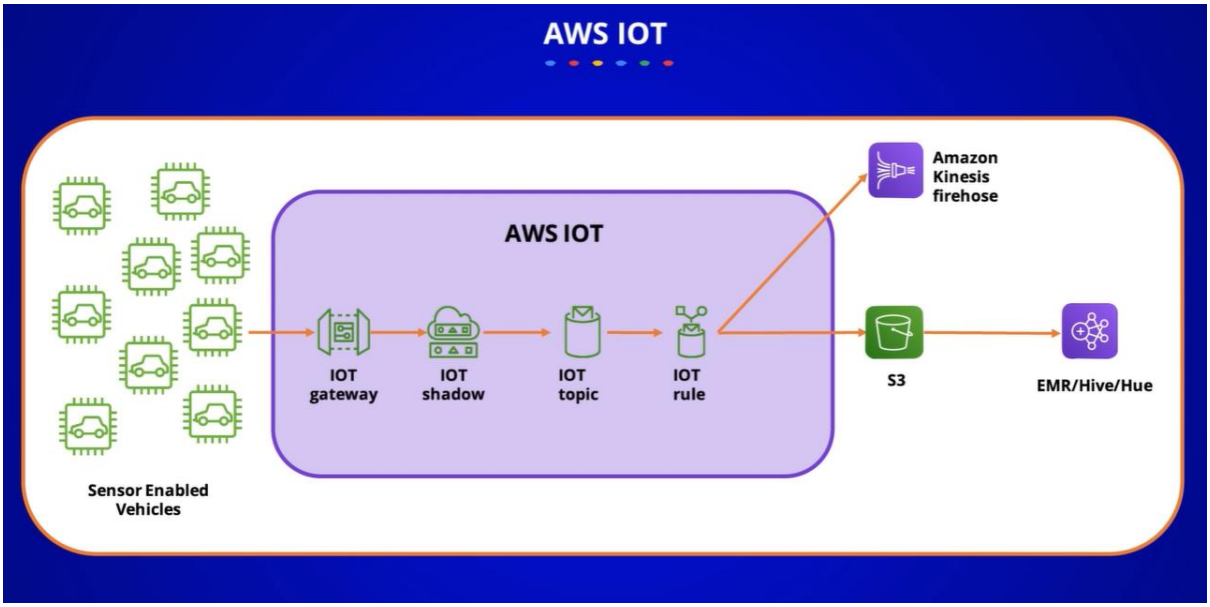
It support billions of devices and trillions of messages and process messages to other AWS services and other devices.

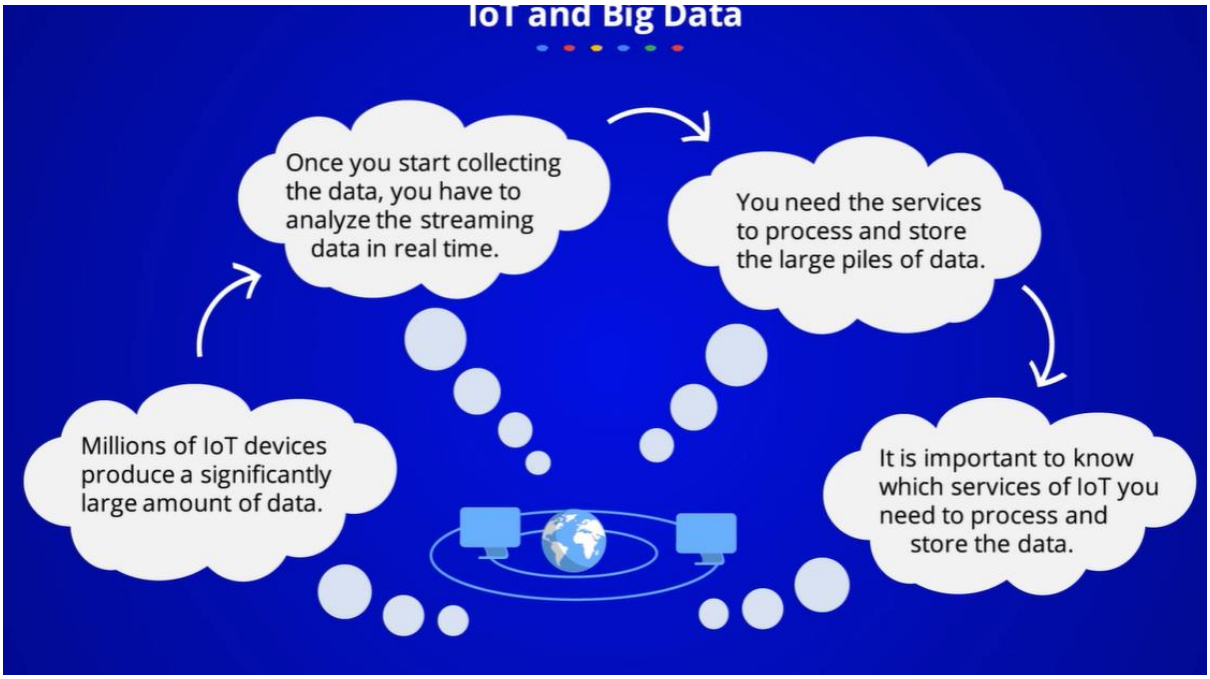
01

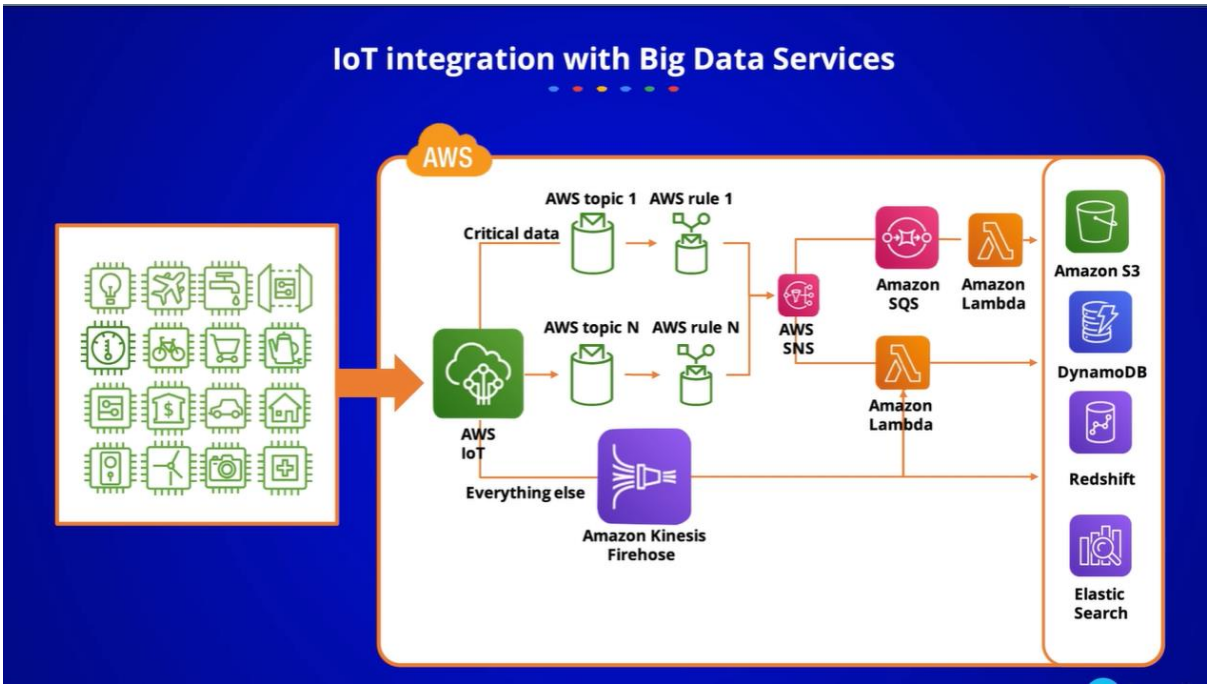
02

You can create applications that permit your customers to govern IoT devices from their phones or tablets.

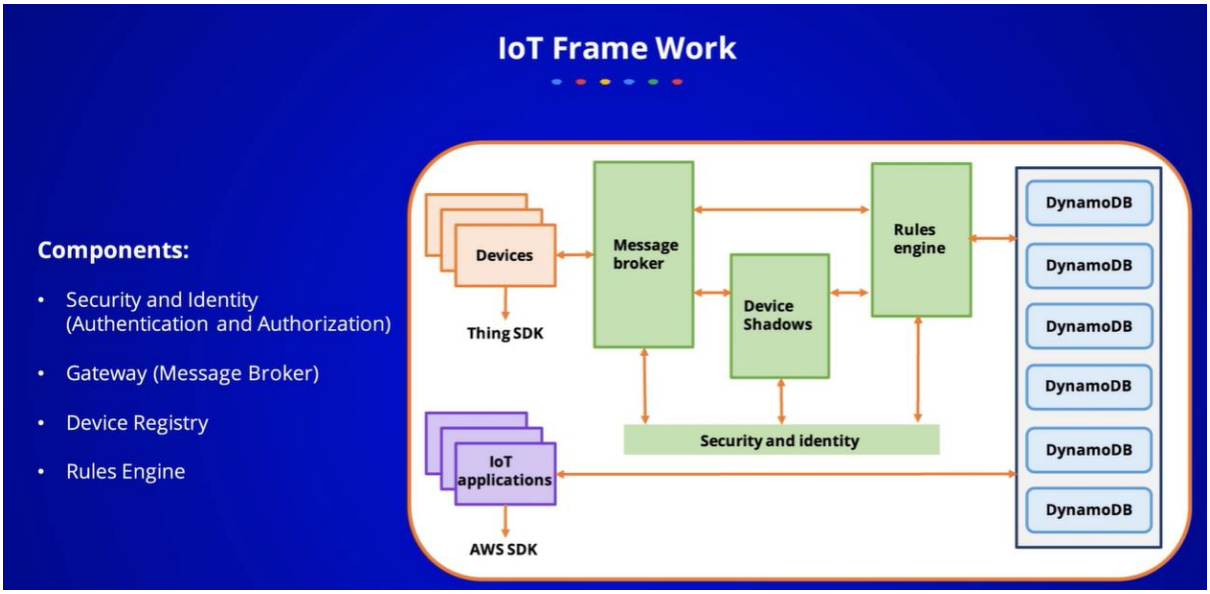
AWS IoT offers at ease, bi-directional communication among net-linked devices which include sensors, actuators, embedded micro-controllers, or smart home equipment and the AWS Cloud.


AWS IOT
Amazon Kinesis firehose
AWS IOT
IOT gateway
IOT shadow
IOT topic
IOT rule
S3
EMR/Hive/Hue
Sensor Enabled Vehicles

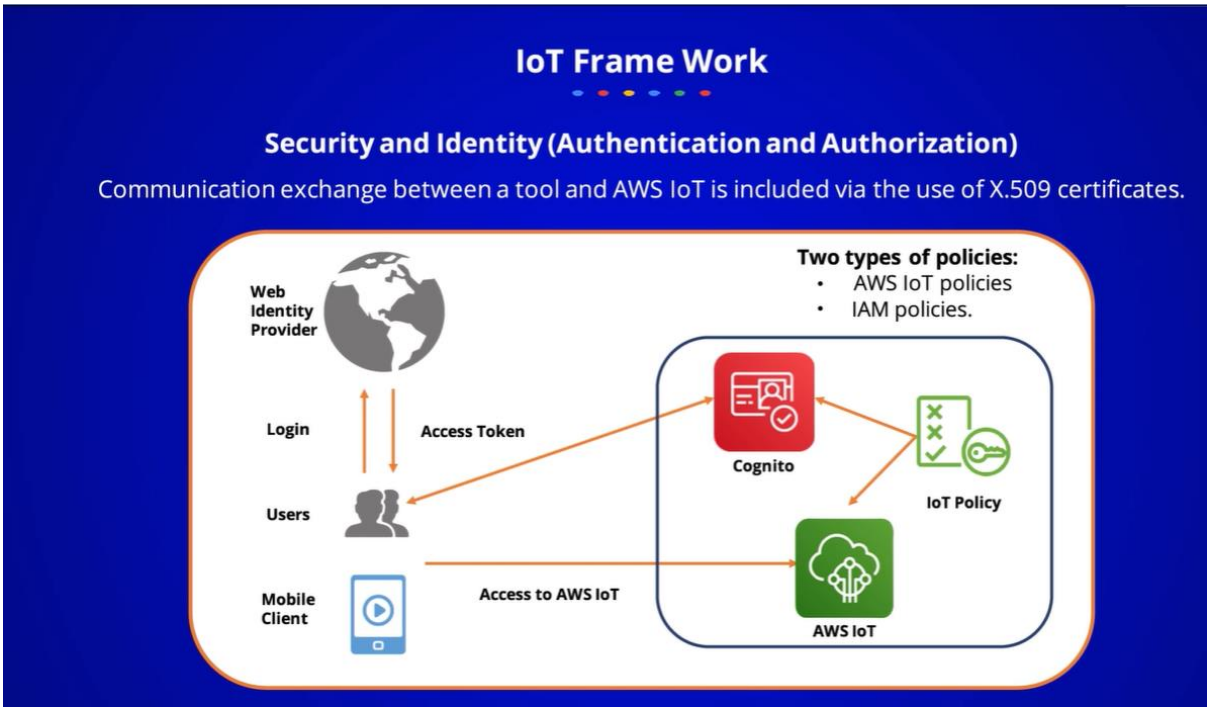

IoT and Big Data
Once you start collecting the data, you have to analyze the streaming data in real time.
You need the services to process and store the large piles of data.
Millions of IoT devices produce a significantly large amount of data.
It is important to know which services of IoT you need to process and store the data.



Once received massage to IoT device. IoT device create rule and action data to elastic search service domain . data kinesis firehose stream write data into dynamo table even though Amazon machine learn base and production machine learning model. IoT machine model allow to change cloud watch alarm capture cloud watch metrics write data to s3 buckets, sqs queue sns push noticification and finally invoked lambda function.



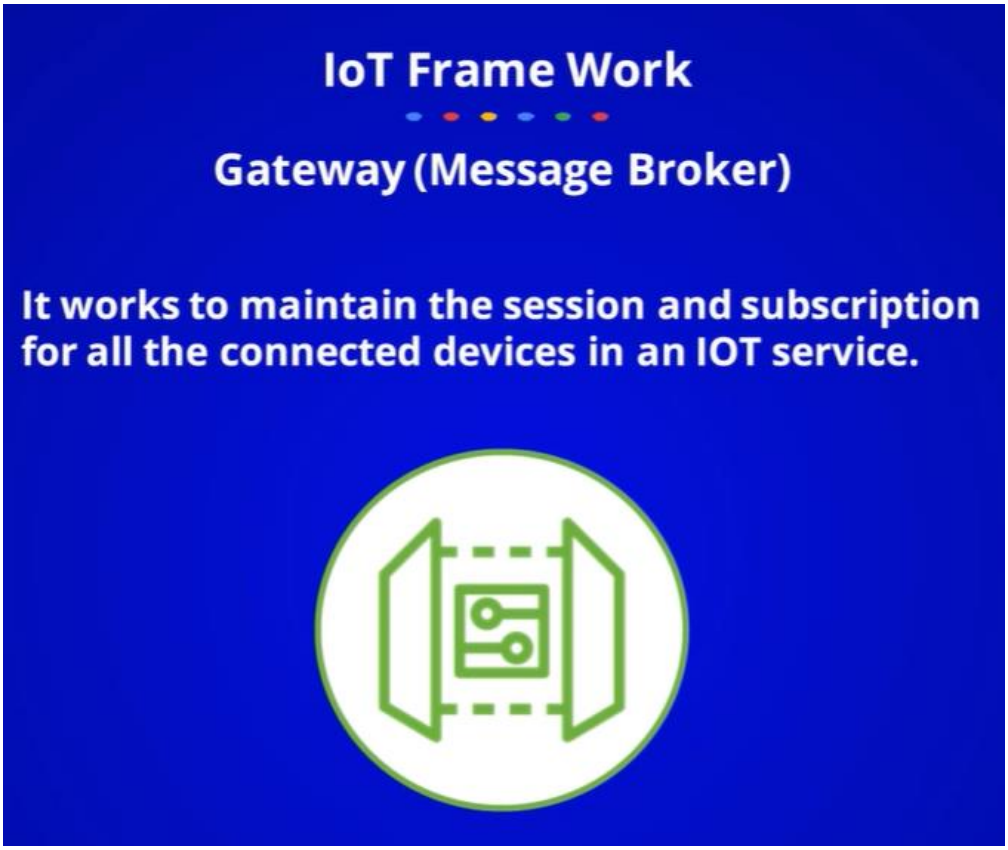
One device send to IoT device. Iot device having permission to write data into Elastic search data domain.



Aws IoT cerificified and register and copy to our device. When device communicating aws IOT present certification identity credential.

Aws Iot use IAM policies use for user group and role. For mobile application authentication against IOT.

Authorisation two type of policy aws IoT policies and Iam policies.



Gateway communicate one to one and one to many devices

IoT Frame Work

Device Registry and Device Shadow

The device registry is a central location used for storing attributes

JSON document

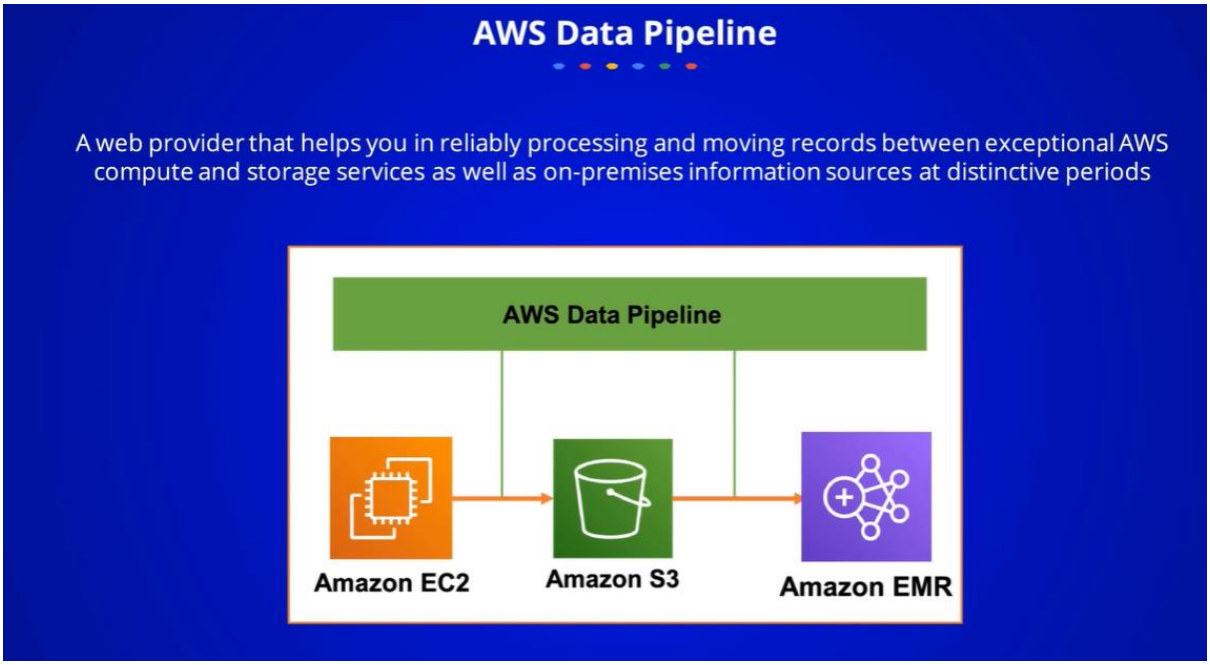
Store and retrieve the current state for a thing

It acts as a message channel to send commands reliably to a thing and stores the last known state in the AWS service.

IoT Frame Work

The Rule Engine collects the statistics dispatched to the IoT cloud

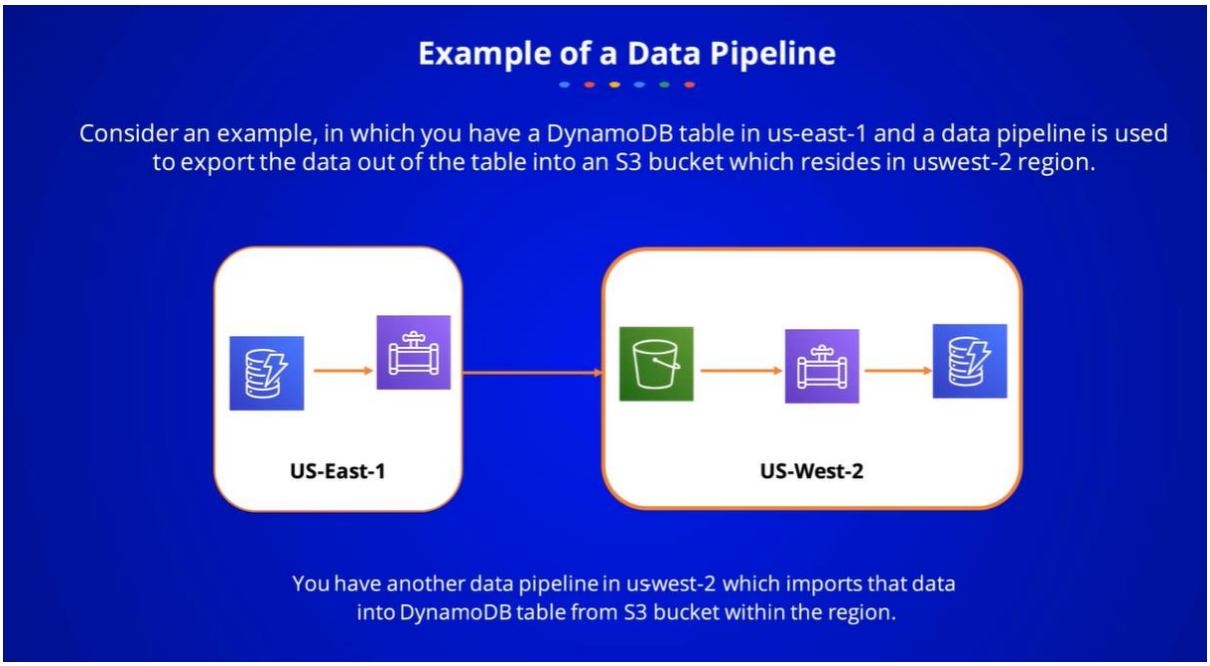
Performs actions based totally on factors which might be a gathered data and routes them to AWS endpoints

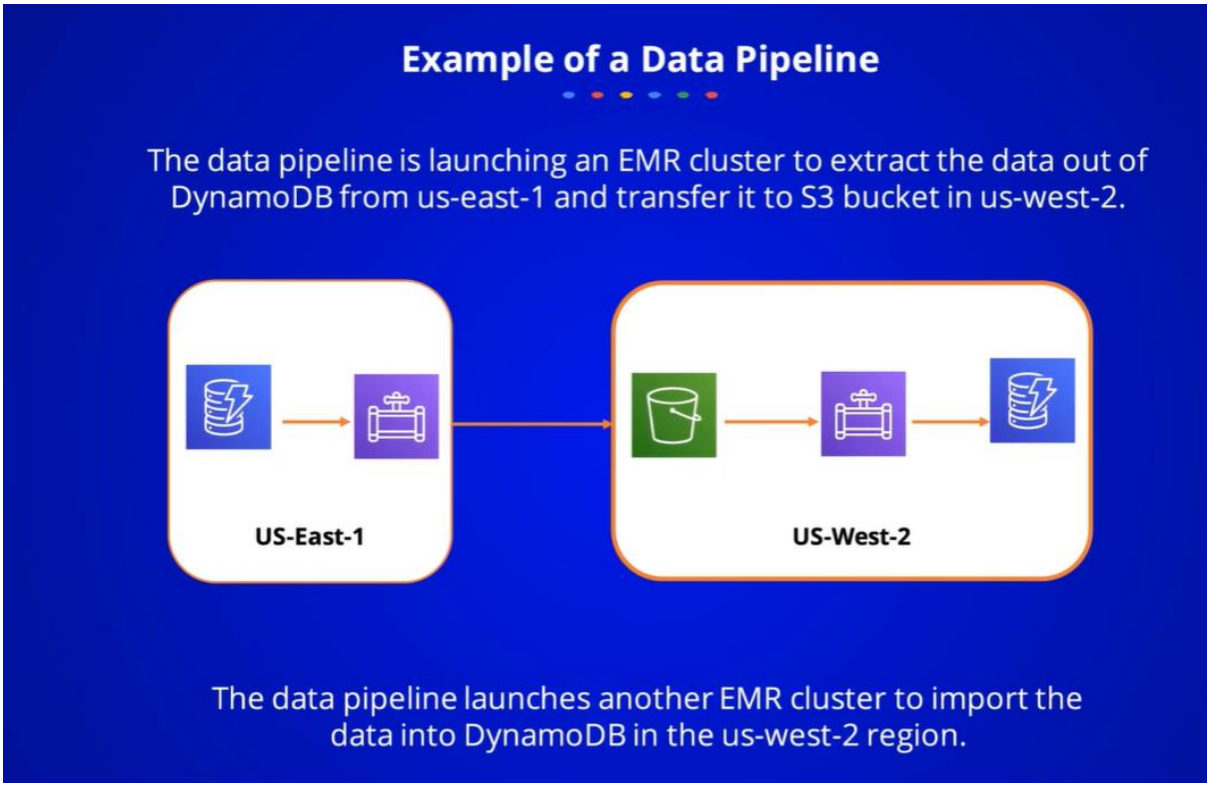


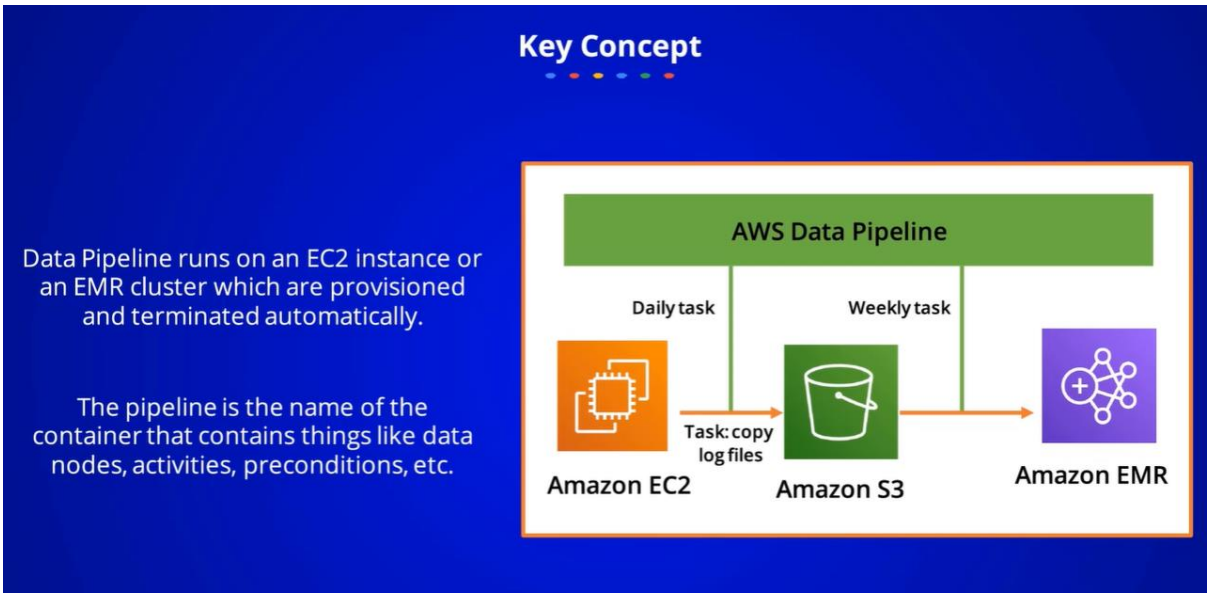
Data pipeline allows to create ETL work flow automate process moment of data schedule interval and terminate etl resource after work completed.

One of future data pipeline is ability to move data across region and you can copy entire dynamo db tables to another region or incremental copy table to another region.

You can reduce time frequency copy the table data pipeline help you reliable process move data between difference aws compute and store data services.





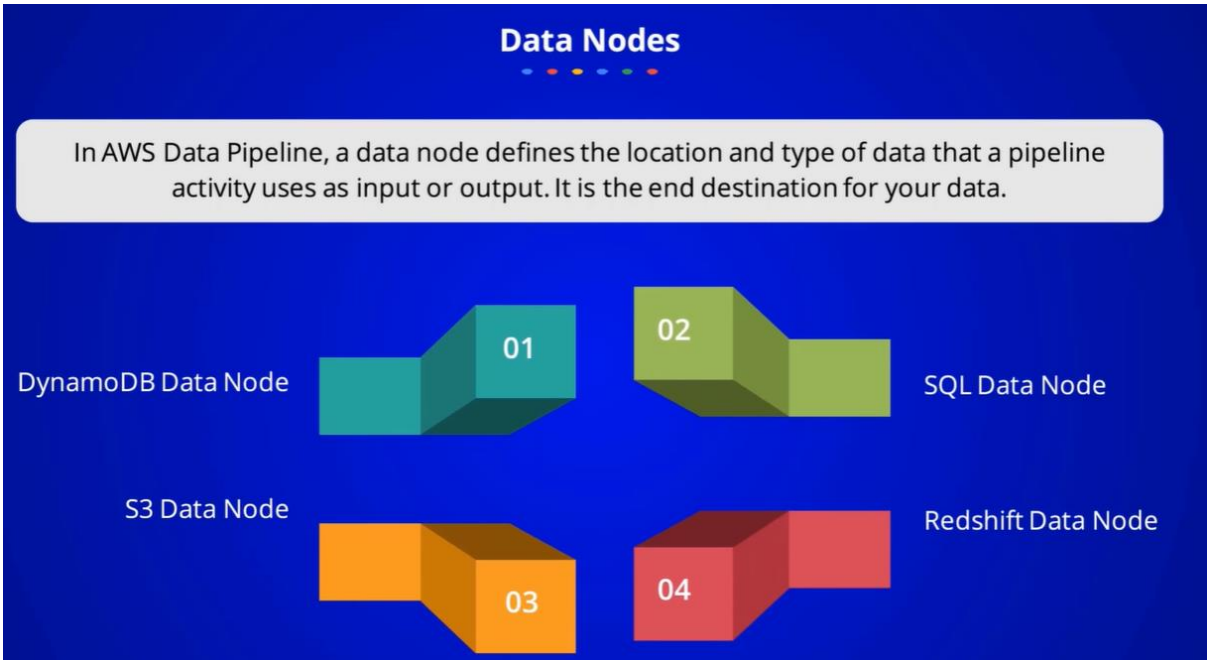


One of best benefits pipe line that can be scheduled all components move one location to another.

And pipe line run on premise run aws provide task run package install on own premise host.

As soon as install packet poll the pipe line works carrier out while time run select activity on premise source as example executing db store to process data base done. Aws pipe line issues to task run.

Data pipeline integrated with on premise environment.

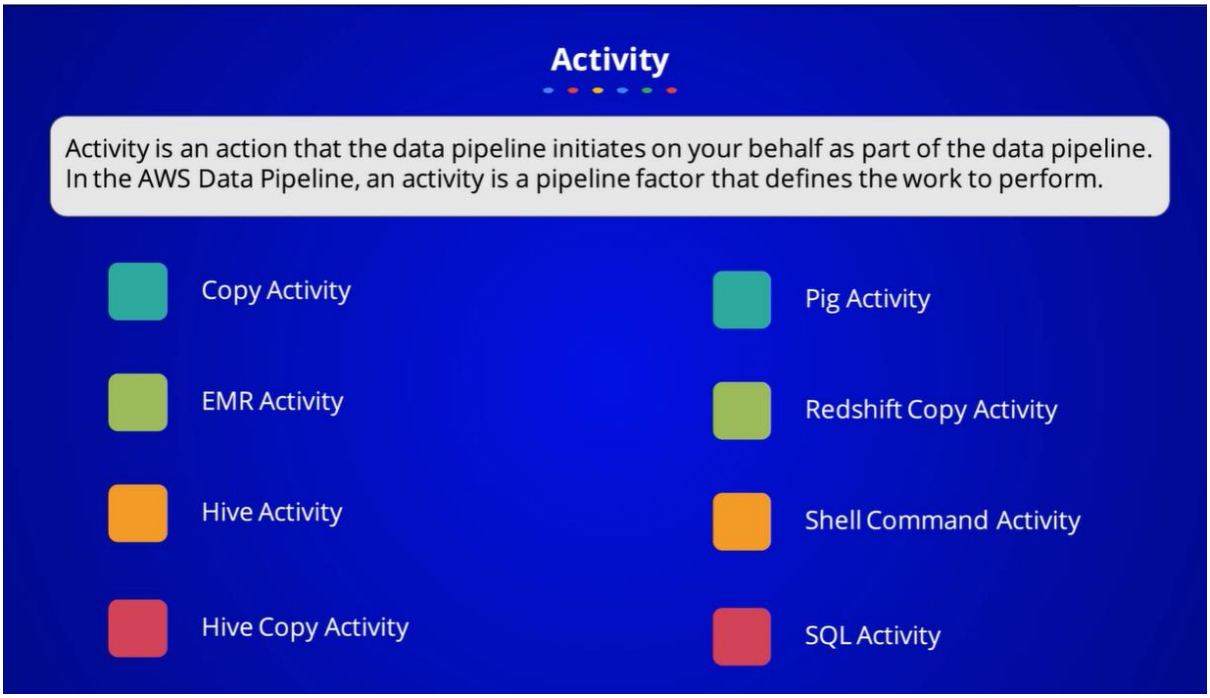


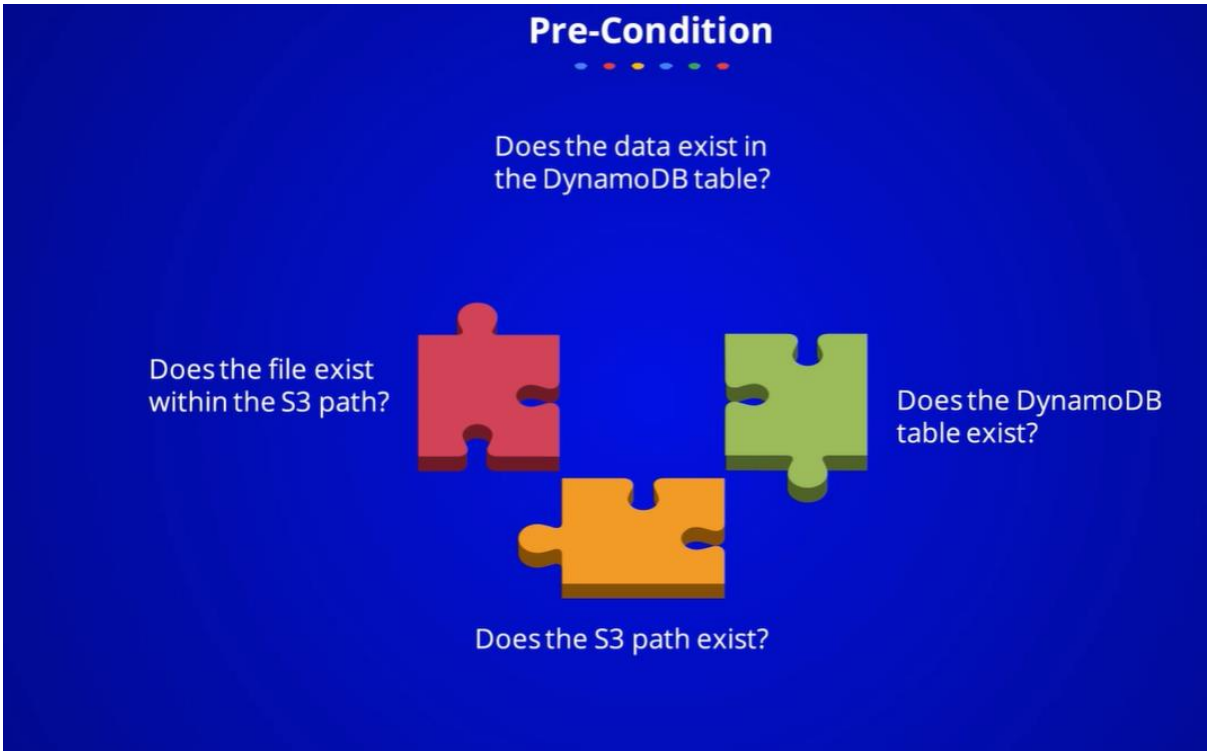
dynamo db table contain data high activity or emr activity use.

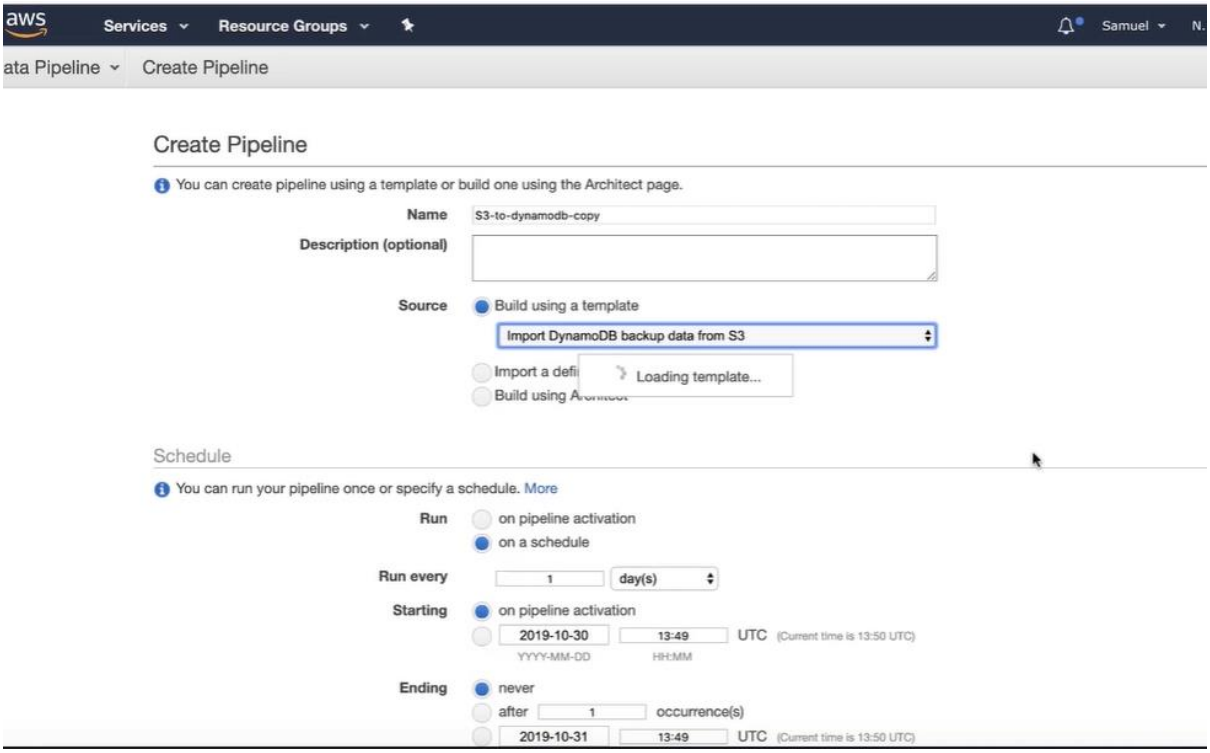
Sql data note: sql databases query represent data for pipe line activity use.

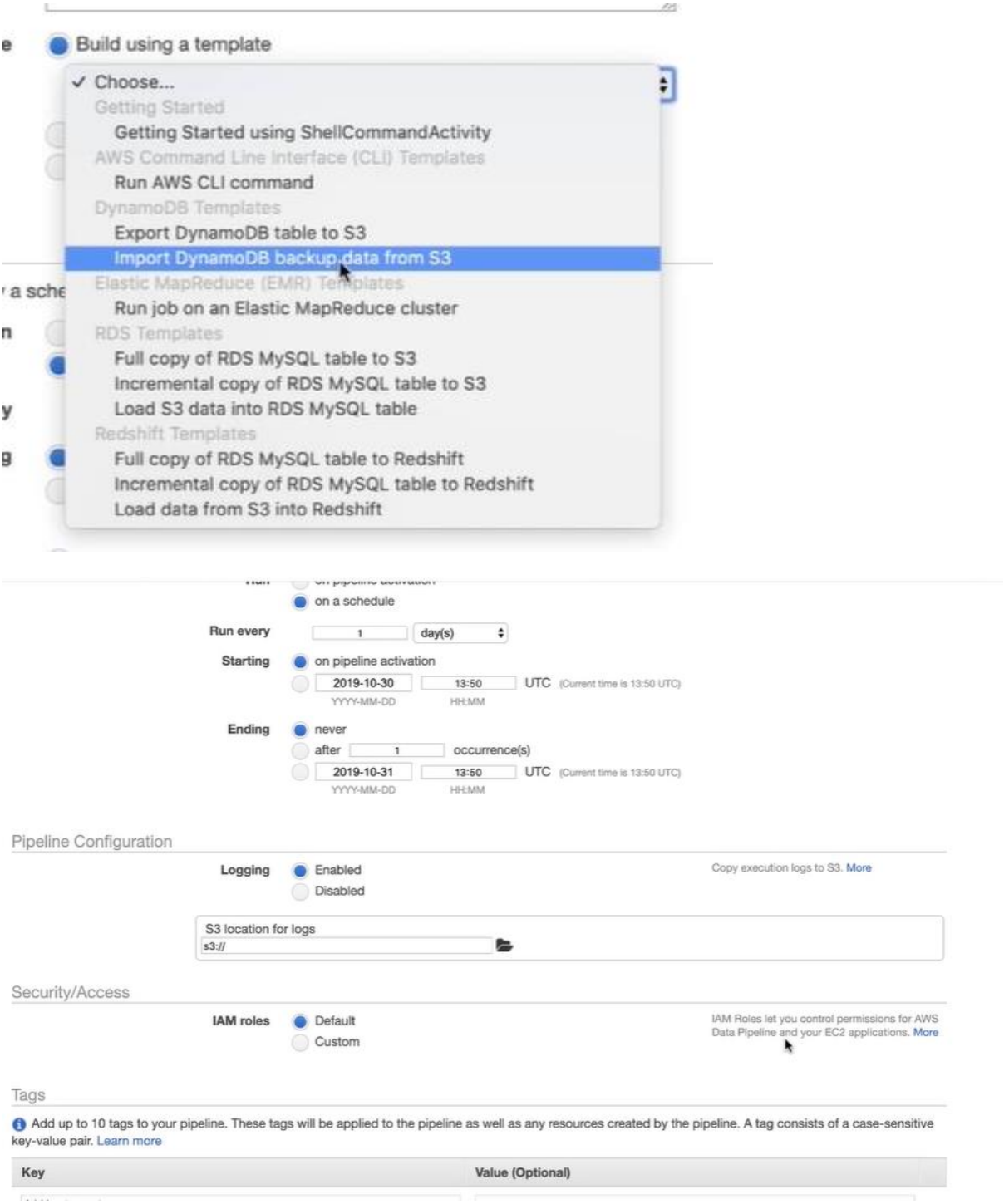
Redshift data note: amazon redshift table contain redshift copy activity to use.

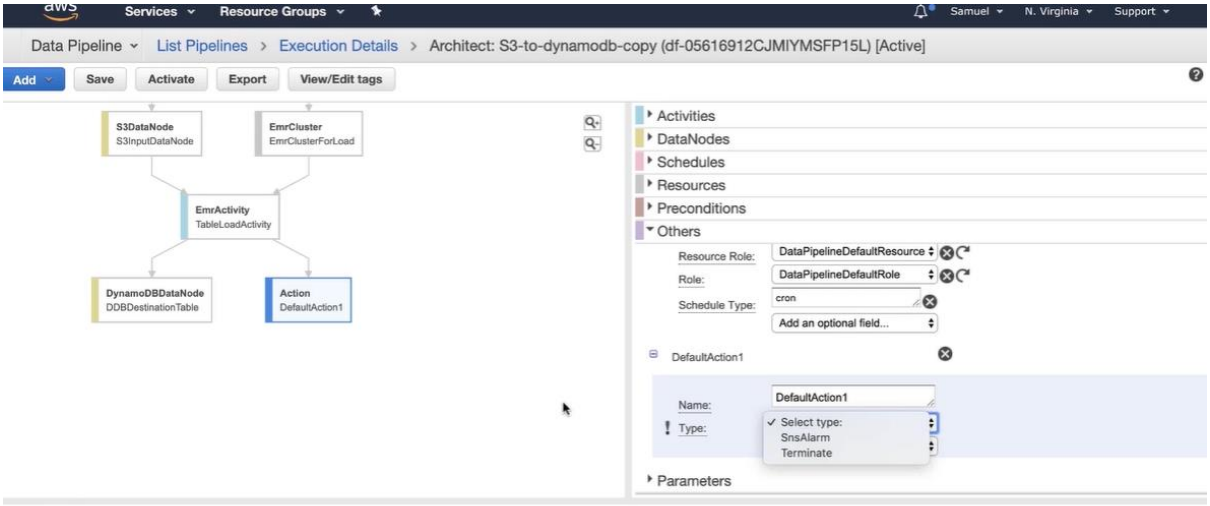
S3 data node: amzon location that contain one or more file contain for pipeline activity use.



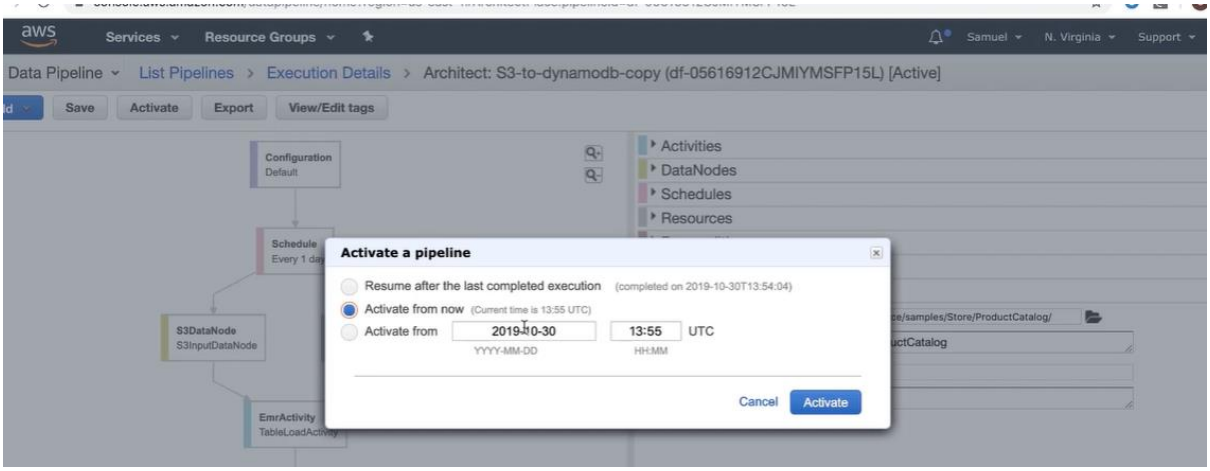

Pre-Condition
Does the data exist in the DynamoDB table?
Does the file exist within the S3 path?
Does the DynamoDB table exist?
Does the S3 path exist?

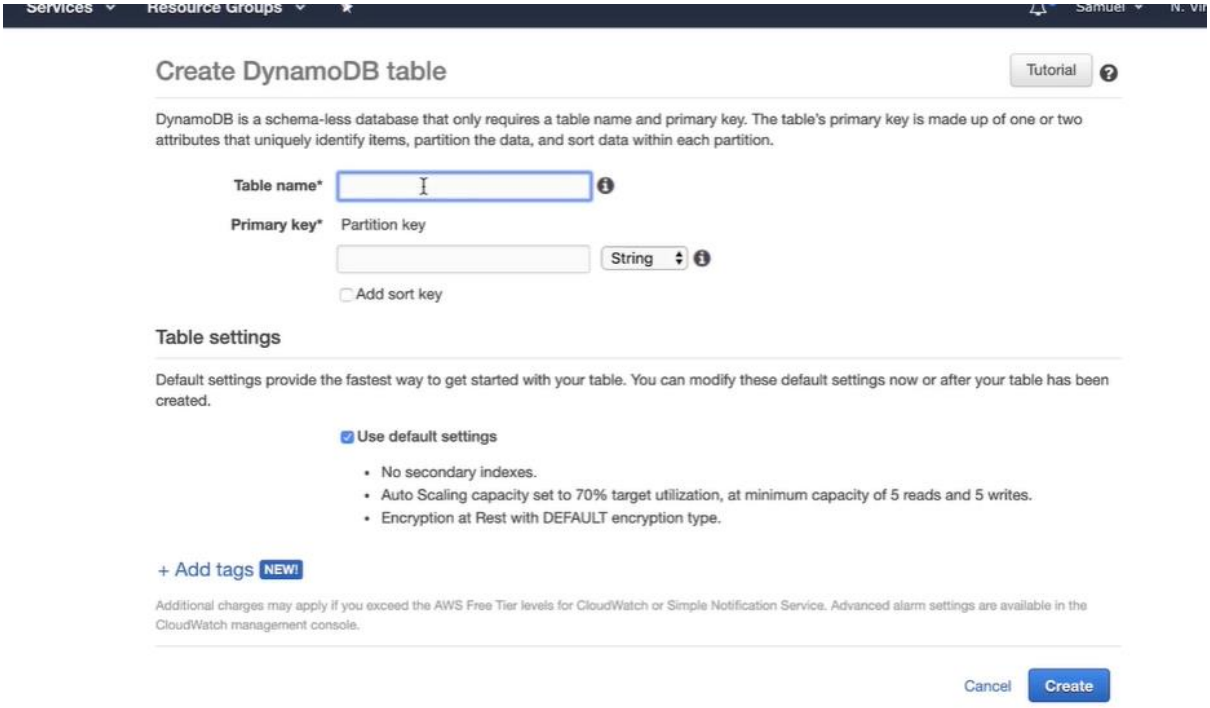

aws
Services
Resource Groups
Samuel
ata Pipeline
Create Pipeline
Create Pipeline
You can create pipeline using a template or build one using the Architect page.
Name
S3-to-dynamodb-copy
Description (optional)
Source
Build using a template
Import DynamoDB backup data from S3
Loading template...
Schedule
You can run your pipeline once or specify a schedule. More
Run
on pipeline activation
on a schedule
Run every
1
day(s)
Starting
on pipeline activation
2019-10-30
13:49
UTC
(Current time is 13:50 UTC)
YYYY-MM-DD
HH:MM
Ending
never
after
1
occurrence(s)
2019-10-31
13:49
UTC
(Current time is 13:50 UTC)

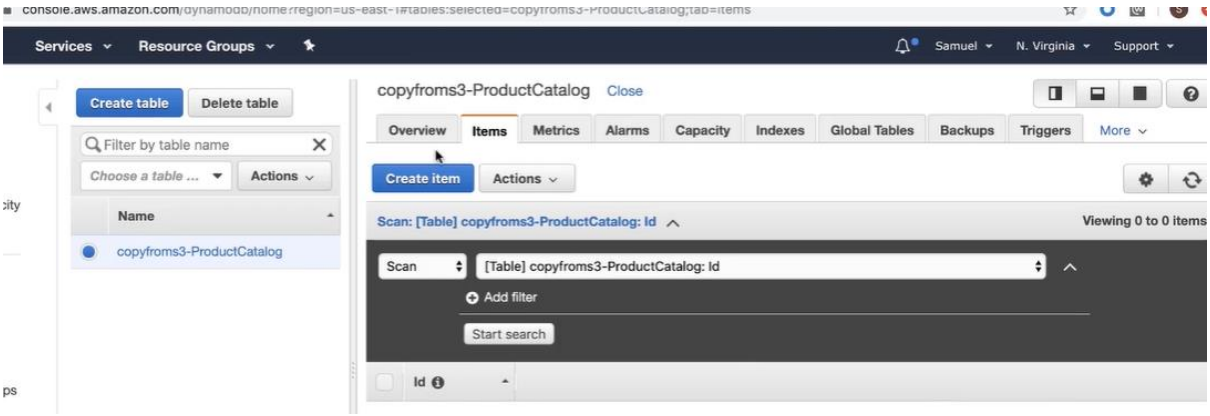

Build using a template
Choose...
Getting Started
Getting Started using ShellCommandActivity
AWS Command Line Interface (CLI) Templates
Run AWS CLI command
DynamoDB Templates
Export DynamoDB table to S3
Import DynamoDB backup data from S3
Elastic MapReduce (EMR) Templates
Run job on an Elastic MapReduce cluster
RDS Templates
Full copy of RDS MySQL table to S3
Incremental copy of RDS MySQL table to S3
Load S3 data into RDS MySQL table
Redshift Templates
Full copy of RDS MySQL table to Redshift
Incremental copy of RDS MySQL table to Redshift
Load data from S3 into Redshift
on a schedule
Run every
1
day(s)
Starting
on pipeline activation
2019-10-30
13:50
UTC
(Current time is 13:50 UTC)
YYYY-MM-DD
HH:MM
Ending
never
after
1
occurrence(s)
2019-10-31
13:50
UTC
(Current time is 13:50 UTC)
YYYY-MM-DD
HH:MM
Pipeline Configuration
Logging
Enabled
Disabled
Copy execution logs to S3. More
S3 location for logs
s3://
Security/Access
IAM roles
Default
Custom
IAM Roles let you control permissions for AWS Data Pipeline and your EC2 applications. More
Tags
Add up to 10 tags to your pipeline. These tags will be applied to the pipeline as well as any resources created by the pipeline. A tag consists of a case-sensitive key-value pair. Learn more
Key
Value (Optional)

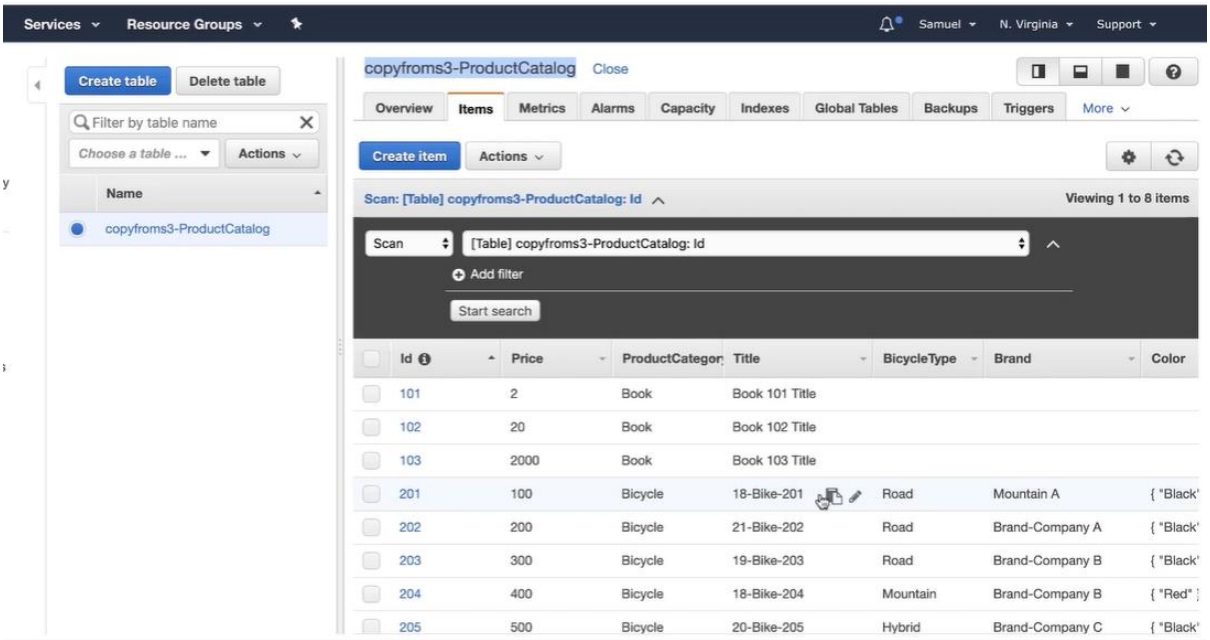


rrors/Warnings




Services
Resource Groups
Create DynamoDB table
Tutorial
DynamoDB is a schema-less database that only requires a table name and primary key. The table's primary key is made up of one or two attributes that uniquely identify items, partition the data, and sort data within each partition.
Table name*
Primary key*
Partition key
String
Add sort key
Table settings
Default settings provide the fastest way to get started with your table. You can modify these default settings now or after your table has been created.
Use default settings
No secondary indexes.
Auto Scaling capacity set to 70% target utilization, at minimum capacity of 5 reads and 5 writes.
Encryption at Rest with DEFAULT encryption type.
+ Add tags
NEW!
Additional charges may apply if you exceed the AWS Free Tier levels for CloudWatch or Simple Notification Service. Advanced alarm settings are available in the CloudWatch management console.
Cancel
Create


Services
Resource Groups
Samuel
N. Virginia
Support
Create table
Delete table
Filter by table name
Choose a table ...
Actions
Name
copyfroms3-ProductCatalog
copyfroms3-ProductCatalog
Close
Overview
Items
Metrics
Alarms
Capacity
Indexes
Global Tables
Backups
Triggers
More
Create item
Actions
Scan: [Table] copyfroms3-ProductCatalog: Id
Viewing 0 to 0 items
Scan
[Table] copyfroms3-ProductCatalog: Id
Add filter
Start search
Id


Services
Resource Groups
Samuel
N. Virginia
Support
Create table
Delete table
Filter by table name
Choose a table ...
Actions
Name
copyfroms3-ProductCatalog
copyfroms3-ProductCatalog
Close
Overview
Items
Metrics
Alarms
Capacity
Indexes
Global Tables
Backups
Triggers
More
Create item
Actions
Scan: [Table] copyfroms3-ProductCatalog: Id
Viewing 1 to 8 items
Scan
[Table] copyfroms3-ProductCatalog: Id
Add filter
Start search
Id
Price
ProductCategor
Title
BicycleType
Brand
Color
101 2 Book Book 101 Title
102 20 Book Book 102 Title
103 2000 Book Book 103 Title
201 100 Bicycle 18-Bike-201 Road Mountain A { "Black'
202 200 Bicycle 21-Bike-202 Road Brand-Company A { "Black'
203 300 Bicycle 19-Bike-203 Road Brand-Company B { "Black'
204 400 Bicycle 18-Bike-204 Mountain Brand-Company B { "Red"
205 500 Bicycle 20-Bike-205 Hybrid Brand-Company C { "Black'

Lesson 4:

AWS Big Data Storage Services

Learning Objective

After completing this lesson, you should be able to understand:

1. Amazon Glacier and Big Data
2. Introduction to AWS DynamoDB
3. DynamoDB in AWS Eco-System
4. Snowball & AWS Big Data
5. AWS DMS
6. AWS Aurora in Big Data

Amazon Glacier

Exceptionally low-cost service of AWS which offers secure and substantial storage for data archiving and backup.

It is optimized to hold expenses low for the data which is accessed infrequently, and for which retrieval time of numerous hours is appropriate.

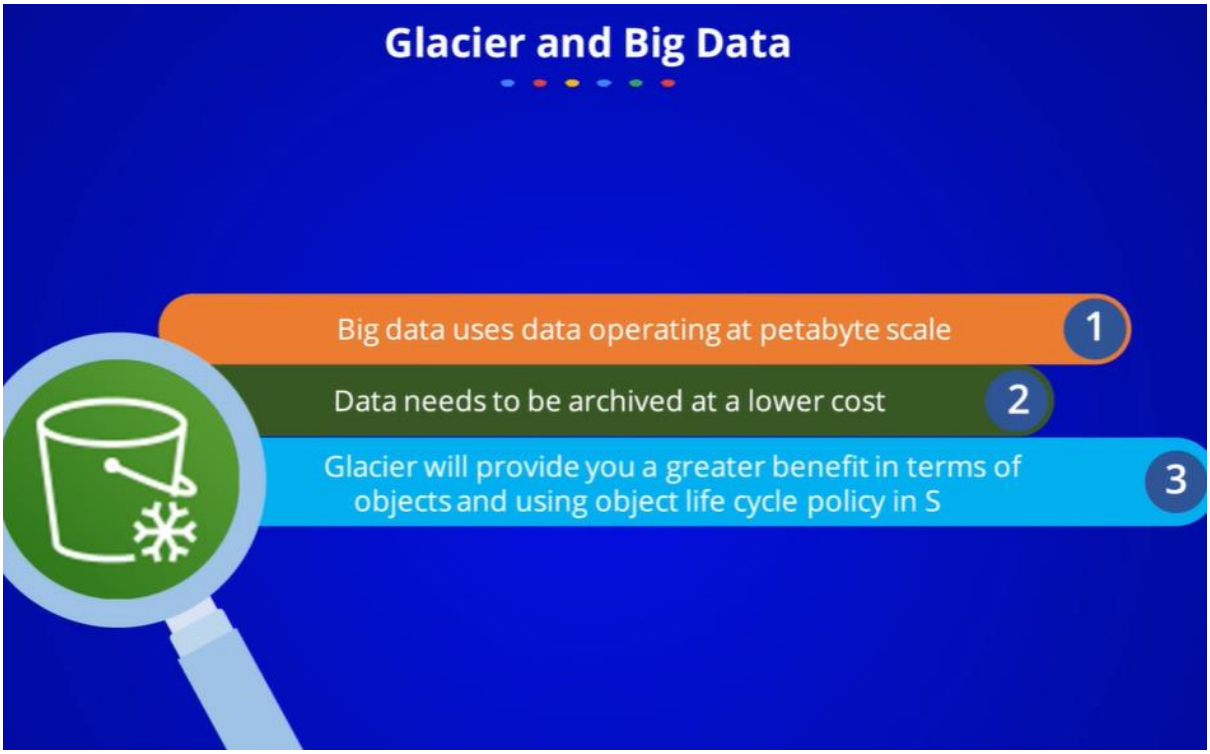
Amazon Glacier

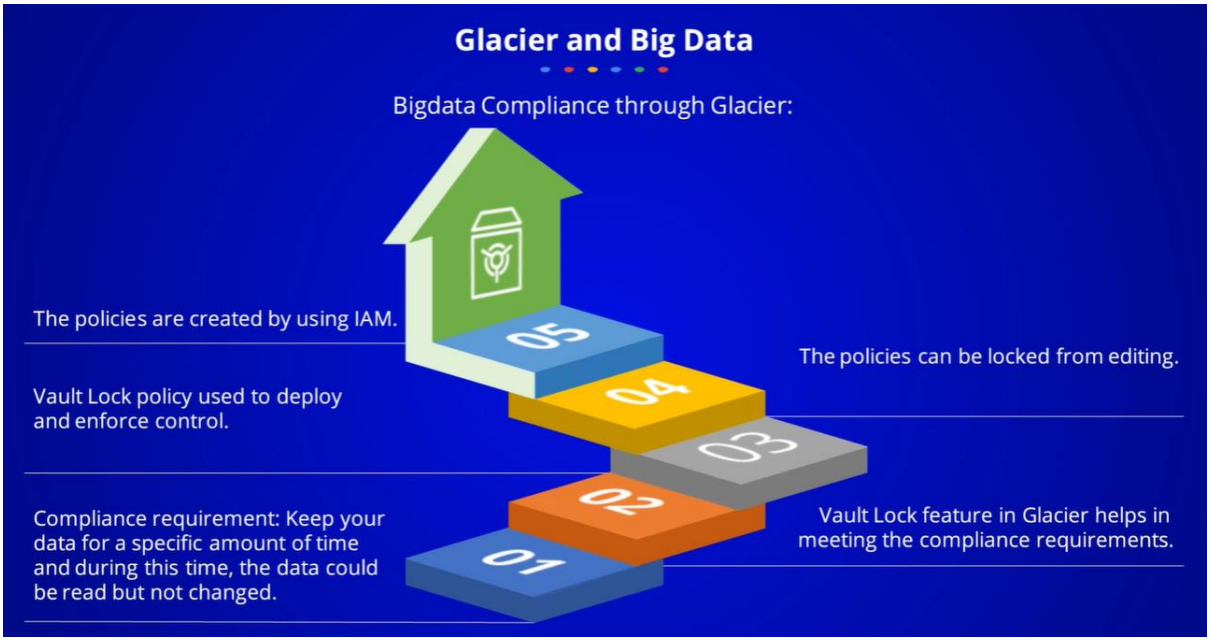
Current archiving Problem in Traditional storage

- Companies generally over-pay for archival of data.
- Provision for an unexpected increase

Amazon Glacier game Changer

- You do not pay anything upfront
- Pay a very low fee for storage
- Scale your utilization up or down as needed


Glacier and Big Data
Big data uses data operating at petabyte scale
1
Data needs to be archived at a lower cost
2
Glacier will provide you a greater benefit in terms of objects and using object life cycle policy in S
3


Glacier and Big Data
Bigdata Compliance through Glacier:
The policies are created by using IAM.
Vault Lock policy used to deploy and enforce control.
Compliance requirement: Keep your data for a specific amount of time and during this time, the data could be read but not changed.
05
04
03
02
01
The policies can be locked from editing.
Vault Lock feature in Glacier helps in meeting the compliance requirements.

Glacier and Big Data

Compliance Storage with Vault Lock

Amazon Glacier Vault Lock gives you the ease of setting compliance controls on individual vaults and enforce them via a lockable policy.

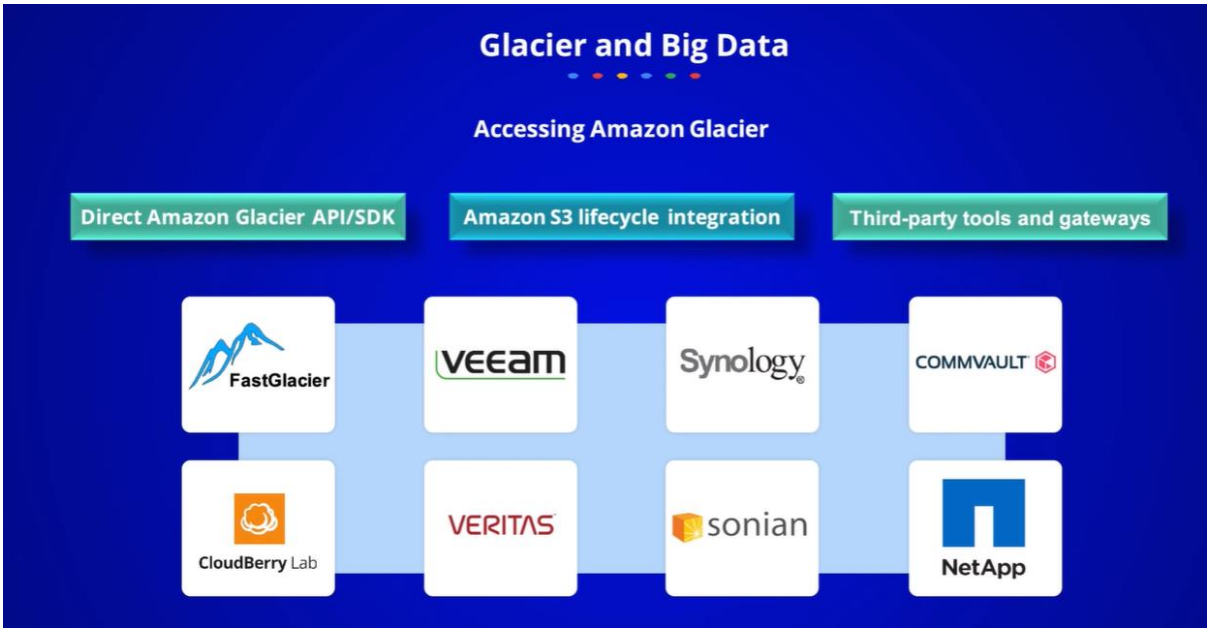
- Time-based retention
  MFA authentication
- Control govern all records in a vault
- Immutable policy
  Two-step locking

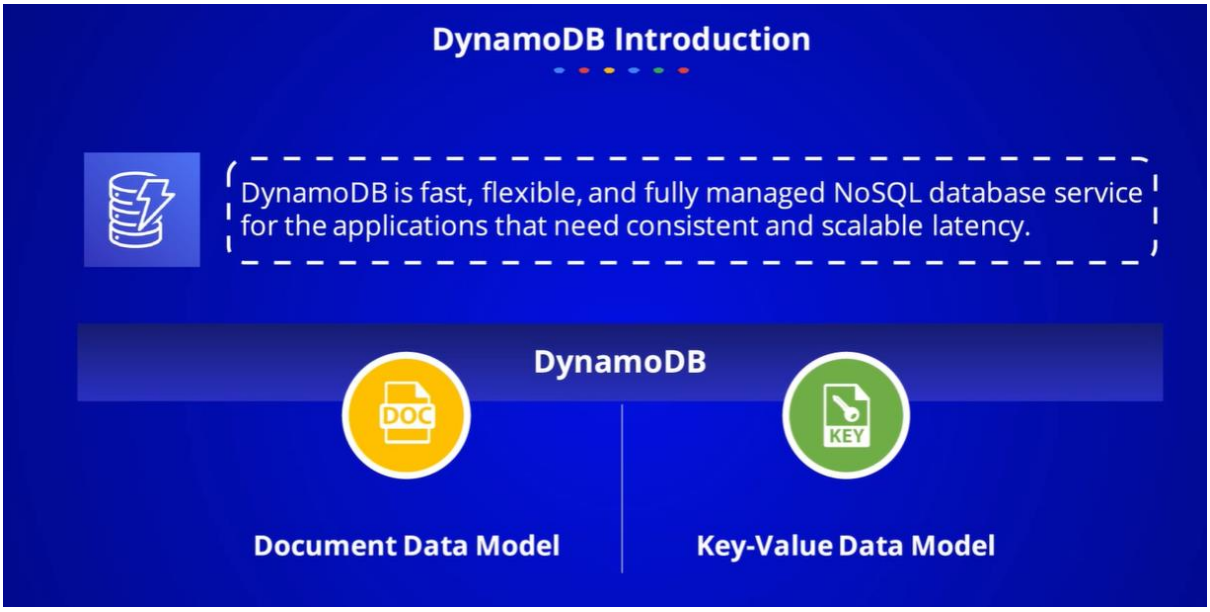
Glacier and Big Data

Management features: Vault Lock

- Non-overwrite, non-erasable records
- Time-based retention with "ArchieveAgeInDays" control
- Policy lockdown (strong governance)
- Legal hold with vault-level tags
- Optionally configure third-party access and provides temporary access

EXAM TIP: The Vault Policies are implemented by using IAM.


Glacier and Big Data
Accessing Amazon Glacier
Direct Amazon Glacier API/SDK
Amazon S3 lifecycle integration
Third-party tools and gateways
FastGlacier
veeam
Synology
COMMVAULT
CloudBerry Lab
VERITAS
sonian
NetApp

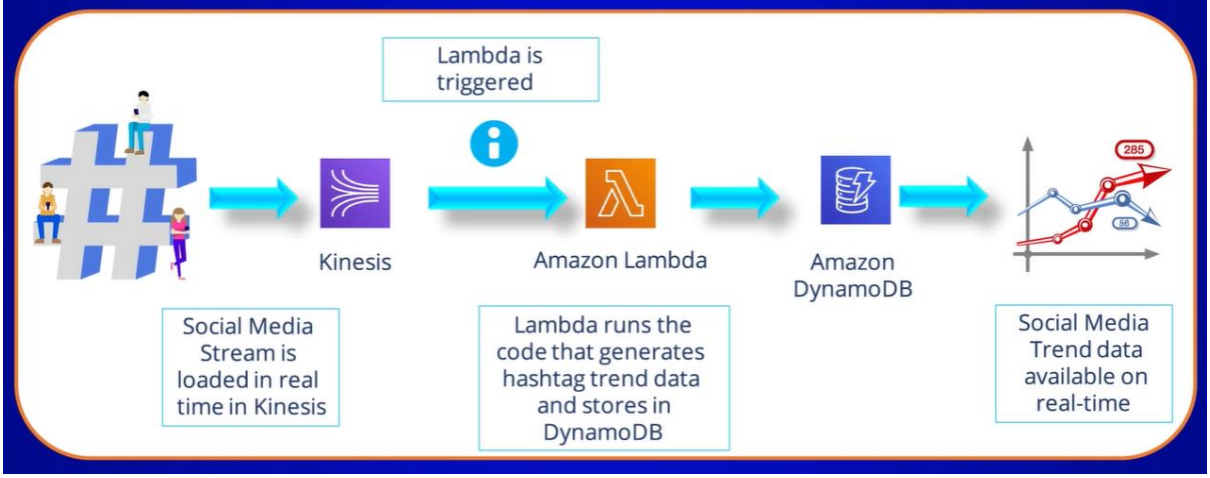

DynamoDB Introduction
DynamoDB is fast, flexible, and fully managed NoSQL database service for the applications that need consistent and scalable latency.
DynamoDB
DOC
KEY
Document Data Model
Key-Value Data Model

DynamoDB Introduction

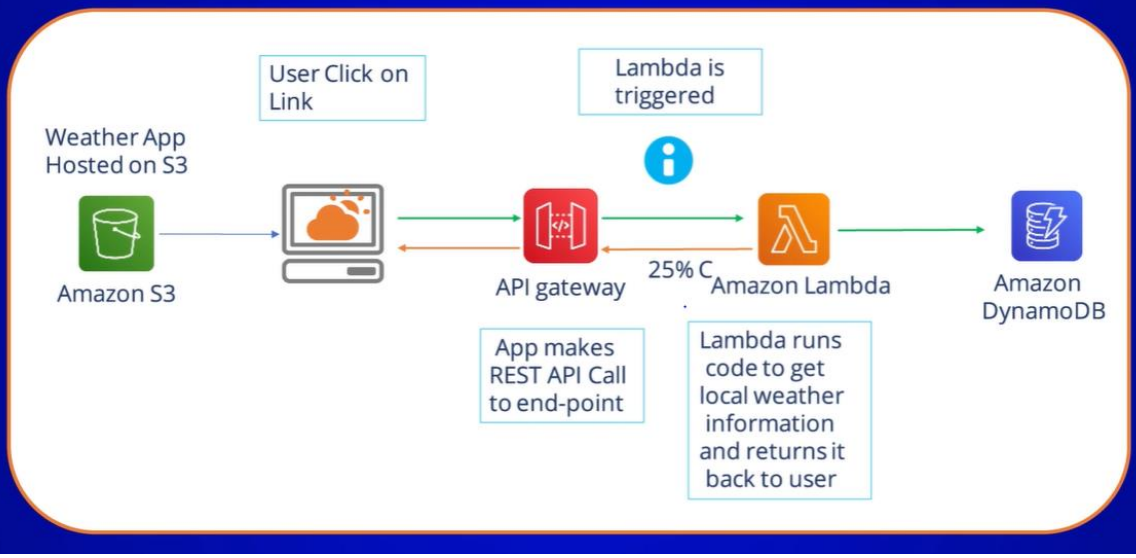
Use cases

- Mobile and web application
- Gaming
- Internet of Things
- Live online voting
- Session management
- Storing S3 object metadata in the DynamoDB table.

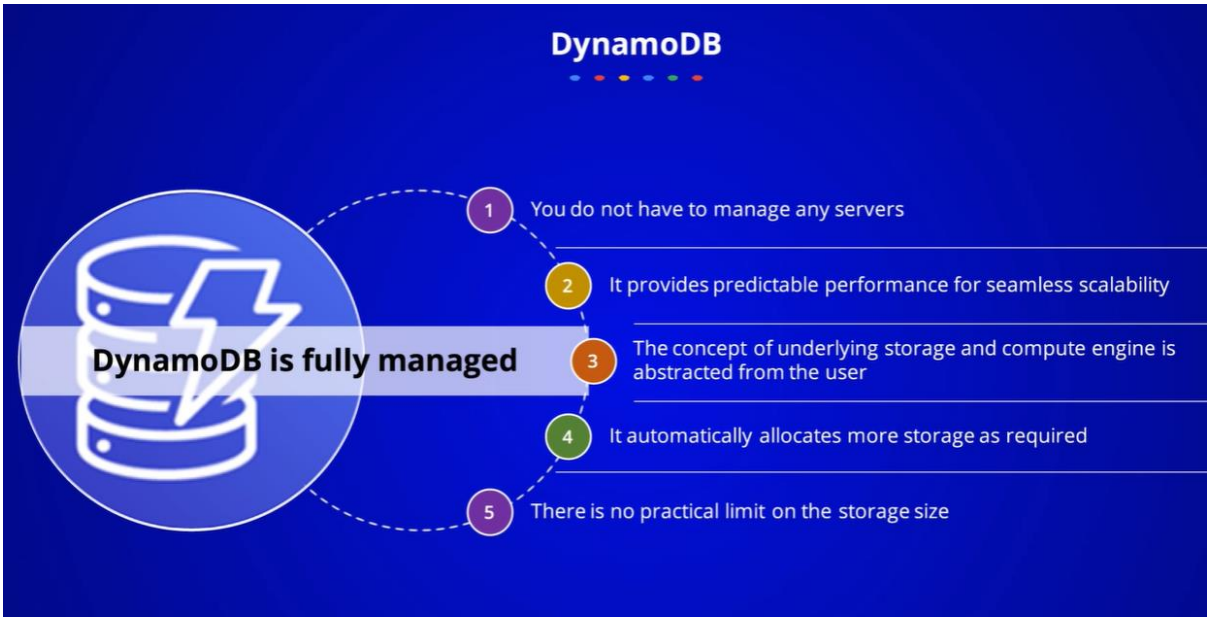
Analysis of Streaming Social Media Data

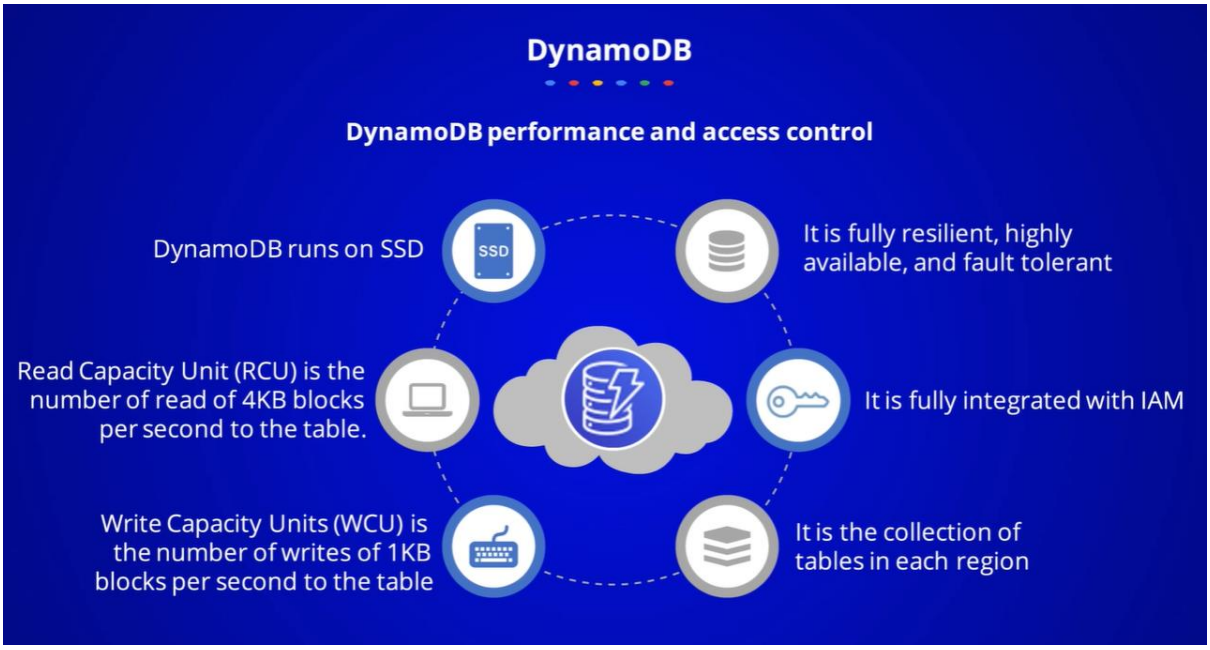


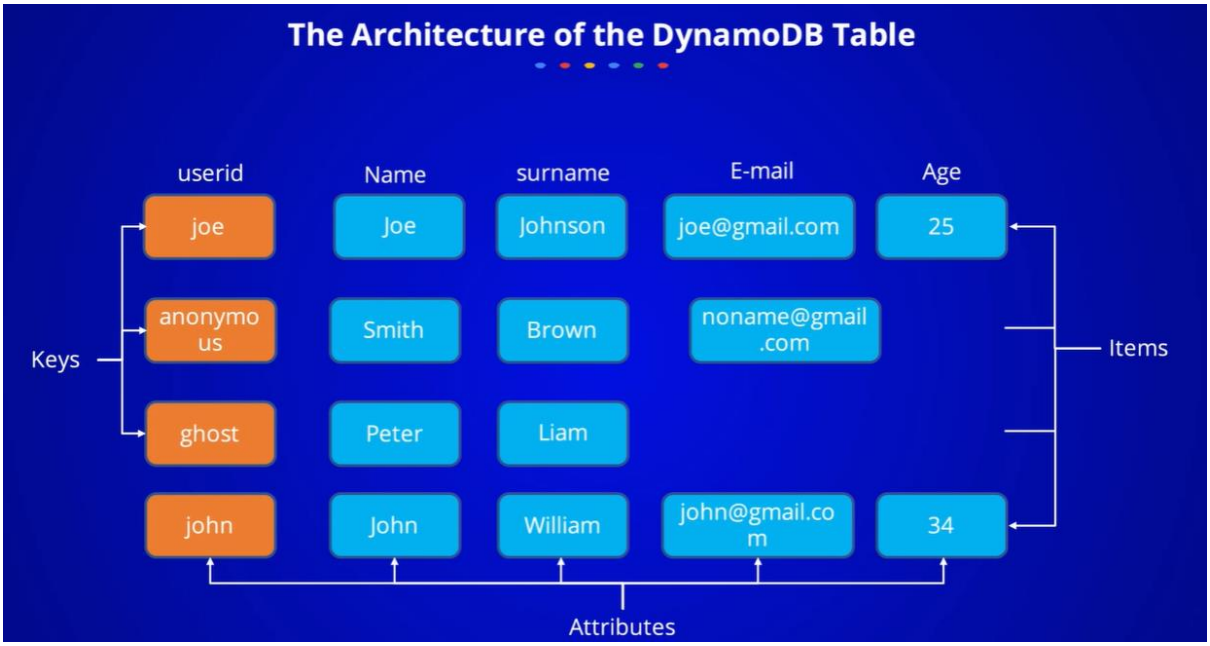
Weather Application

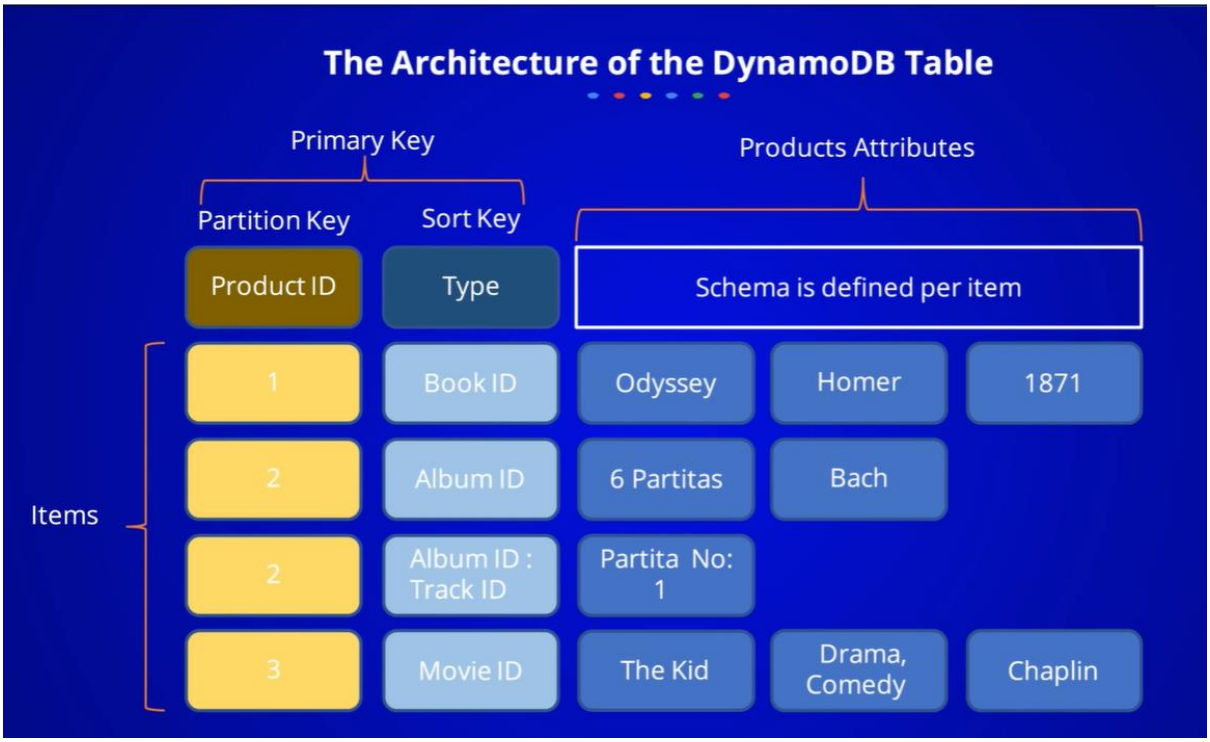


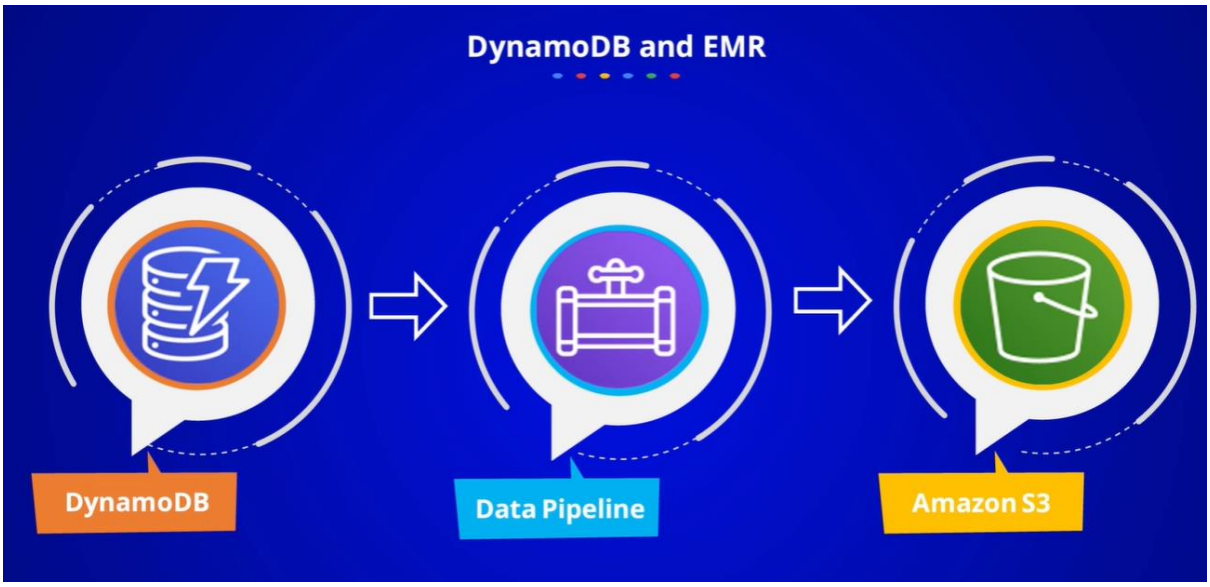
DynamoDB




DynamoDB
DynamoDB performance and access control
DynamoDB runs on SSD
SSD
It is fully resilient, highly available, and fault tolerant
Read Capacity Unit (RCU) is the number of read of 4KB blocks per second to the table.
It is fully integrated with IAM
Write Capacity Units (WCU) is the number of writes of 1KB blocks per second to the table
It is the collection of tables in each region


The Architecture of the DynamoDB Table
userid
Name
surname
E-mail
Age
joe
Joe
Johnson
joe@gmail.com
25
anonymous
Smith
Brown
noname@gmail.com
ghost
Peter
Liam
john
John
William
john@gmail.com
34
Keys
Items
Attributes





Using copy cmd directly copy from dynamo db to redshift.

Dynamodb export data using data pipeline emr cluster, emr running dynamo db table write into s3 bucket. If s3 want write data to dynamo db using data pipeline emr cluster write with dynamo db.