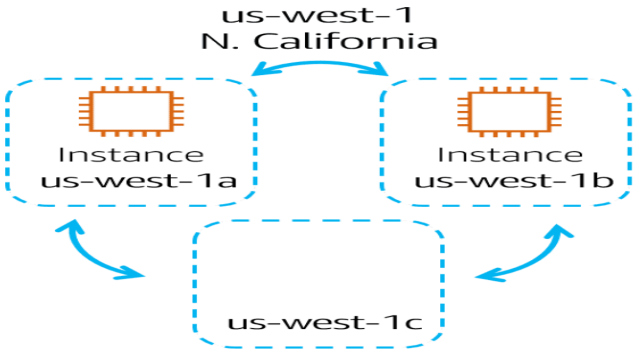
AWS=On demand cloud facility.

The following are the components that make up the AWS infrastructure:

- **Availability Zones**: **one or more Data centers within a region with redundant power, network**. If one az is affected due to disaster than other az is not effected in a region
- **Region** : Physical location where machine are stored.(collection of 2 more AZ)
- **Edge locations** : caches static data by CDN for fasten delivery to your location

  An **edge location** is a site that Amazon CloudFront uses to store cached copies of your content closer to your customers for faster delivery.

- **Regional Edge Caches**



best practice is to run applications across at least two Availability Zones in a egion. In this example, you might choose to run a second Amazon EC2 instance in s-west-1b.

global specific service:(route 53. Manages the traffic by routing the traffic).. Services offered

Region specific service: Help to host something. Tightly coupled with a region

whenever natural calamity happens then aws keep data in two or more AZ. so if one AZ goes down then another can work.

AWS service: Management and deployment, application services, foundational services

Selecting a region:1.compliance with data gov and legal req as sometime it is not allowed to reside data outside country

2 pricing(can vary on govt tax rule)

3. proximity to your customers(for faster delivery of req)

4. available service with a region

Root user: when we make a new account to aws then we login as root user. through email. Root user can give permission to different user(IAM) to access aws account with different permission Root user will charge for aws account. Single entity having all permission. Root user can only add product features

IAM user: access through Accountid, username and password. Permission of IAm users can vary(it can have full access or rare access).

By default, when you create a new IAM user in AWS, it has no permissions associated with it

MFA: Multi Factor Authentication.

Two factor authentication. First credential login. second one login system(random numeric code on a personal device)

IAM: enables you to manage access to AWS services and resources securely. Using IAM, you can create and manage AWS users and groups, and use permissions to allow and deny their access to AWS resources.

Principle: role(according to this access is grant) - have a policy document.(JSON DOCUMENT details of whether to allow the resource.

request: object bundled to access a resource according to authentication , authorization

Amazon map policy document with resource then according it request is granting
Authentication **confirms that users are who they say they are**. Authorization gives those users permission to access a resource

Policy:
AWS Policy(Effect, action, condition) can be defined on IAM users and AWS resources
An **IAM policy** is a document that allows or denies permissions to AWS services and resources.

An **IAM group** is a collection of IAM users. When you assign an IAM policy to a group, all users in the group are granted permissions specified by the policy.

An **IAM role i**s an identity that you can assume to gain temporary access to permissions.

Before an IAM user, application, or service can assume an IAM role, they must be granted permissions to switch to the role. When someone assumes an IAM role, they abandon all previous permissions that they had under a previous role and assume the permissions of the new role.

6 Type of Police

1.Identity policies: Attached to users, groups roles
2.Resource Policies: Attached to resource eg(Bucket policy)
3.Permission boundary: max permission to identity\
4.Acces control list: Enabling sharing of data on request on demand by acl to access data by other principle
5 Session Policies

JSON Policies structure
1 version
2 statement:
- 2.1 sid.(potion)-statement id
- 2.2 effect(deny or allow)
- 2.3 principle
- 2.4 Action(list of actions to allow or deny)

2.5 Resources
2.6 Condition(optional)

we can have multiple statement and multiple policies

AWS Organizations to consolidate and manage multiple AWS accounts within a central location.
In AWS Organizations, you can centrally control permissions for the accounts in your organization by using **service control policies (SCPs)**. SCPs enable you to place restrictions on the AWS services, resources, and individual API actions that users and roles in each account can access.
you can group accounts into **organizational units (OUs)** to make it easier to manage accounts with similar business or security requirements.

Amazon Virtual Private Cloud(VPC):

Launch resource to virtual network

Region based

VPC1

AZ

SUBNET 1
SUBNET 2

VPC2

AZ

SUBNET 1
SUBNET 2

INTENET GATEWAY- ACCESS PUBLIC SUBNET .
public subnet : access to net and lan
external subnet: access to internet

Dynamo db: non relational db, scalable, available durable
Table : fundamental component.

cloud based : everything on cloud , migrate existing as well, build new as well on cloud, have the option to build application on low level architecture(need it guide to manage) aur high level(no need of maintenance)

on premise deployment(private cloud dep)resources are deployed on premises by using virtualization and resource management tools.

Hybrid : Connect cloud-based resources to on-premises infrastructure.

Benefits: low cost , variable expenses, pay as you go, speed, can access app anywhere

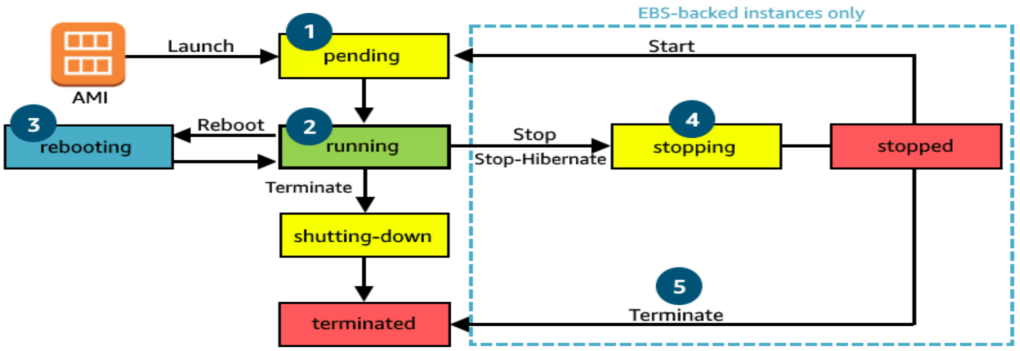
An **Amazon Machine Image (AMI)** provides the information required to launch an instance. You must specify an AMI when you launch an instance. You can launch multiple instances from a single AMI when you need multiple instances with the same configuration.

EC2 for computes is highly flexible, cost effective, and quick when you compare it to running your own servers on premises in a data center that you own.

AWS already built and secured the data centers. AWS has already bought the servers, racked and stacked them, and they are already online ready to be used.

EC2 is build on top of physical server machine and these are share different virtual instance
This idea of sharing underlying hardware is called **multi tenancy.** T**he hypervisor** is responsible for coordinating this multitenancy and it is managed by AWS. hypervisor also isolate diff virtual machine from each other to make it secure.
You can configure instance acc to you ca, choose your own os type.
So what type of requests make it to your server and if they are publicly or privately accessible is something you decide.

EC2instance is combination of virtual cpu, memory, storage, instance storage , gpu so we should choose ec2 instance acc our requirement.
by default these ec2 instance are place in vpc and are public.



pending state: no billing start, setting up instance
rebooting: same rebooting your pc
terminate: lost your instance permanent

Amazon EC2 instance types are optimized for different tasks. When selecting an instance type, consider the specific needs of your workloads and applications

1.**General purpose**: when we need equalized resource for memory, computing,networking i.e not focus mainly on single need,
eg gaming server

2. **Compute bound**: we done compute bound task need high processing
high-performance web servers, compute-intensive applications servers, and dedicated gaming servers. , batch workload.

3. **Memory optimized instance type**: designed to deliver fast performance for workloads that process large datasets in memory. In computing, memory is a temporary storage area. preloading large data from storage before giving to cpu.

4.**Accelerate computing instances**: use hardware accelerators, or coprocessors, to perform some functions more efficiently than is possible in software running on CPUs computing instances are ideal for workloads such as graphics applications, game streaming, and application streaming.

5 **Storage optimized** : are designed for workloads that require high, sequential read and write access to large datasets on local storage.online transaction, file system

Aws pricing:

1. **On demand**: pay only when you use, use in short-term, irregular workloads that cannot be interrupted. No upfront costs or minimum contracts apply.

For applications extending more than a year this becomes costly.

2. **Savings Plans** enable you to reduce your compute costs by committing to a consistent amount of compute usage for a 1-year or 3-year term. This term commitment results in savings of up to 66% over On-Demand costs.
Any usage up to the commitment is charged at the discounted plan rate (for example, $10 an hour). Any usage beyond the commitment is charged at regular On-Demand rates.

3.Reserved Instances are a billing discount applied to the use of On-Demand Instances in your account. You can purchase Standard Reserved and Convertible Reserved Instances for a 1-year or 3-year term, and Scheduled Reserved Instances for a 1-year term. You realize greater cost savings with the 3-year option.

At the end of a Reserved Instance term, you can continue using the Amazon EC2 instance without interruption. However, you are charged On-Demand rates until you do one of the following:

- Terminate the instance.
- Purchase a new Reserved Instance that matches the instance attributes (instance type, Region, tenancy, and platform).

4. **Spot instance** :used for can withstand interruptions. Spot Instances use unused Amazon EC2 computing capacity and offer you cost savings at up to 90% off of On-Demand prices. in this instance can be taken back when need for time period with warning to complete your task early.

5.**Dedicated** :physical servers with Amazon EC2 instance capacity that is fully dedicated to your use. are expensive.

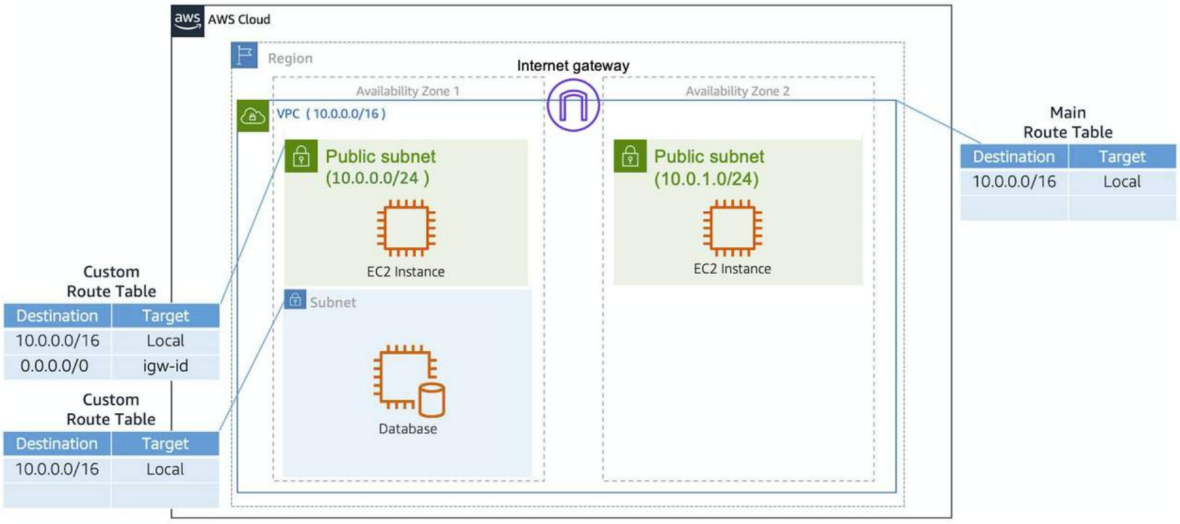
Amazon provide auto scaling on increasing demand and paying according to your need by EC2 auto scaling by adding or removing more instances on demand

two methods of auto scaling
1 .**on Demand**
2 **predictive scaling**
To scale faster, you can use dynamic scaling and predictive scaling together.

The **minimum capacity** is the number of Amazon EC2 instances that launch immediately after you have created the Auto Scaling group.

you can set desired instance and max instance as well but you pay only for what you use

RouteTable
A route table contains a set of rules, called routes, that are used to determine where network traffic is directed.



- The **destination** is a range of IP addresses where you want your traffic to go. In the example of sending a letter, you need a destination to route the letter to the appropriate place. The same is true for routing traffic. In this case, the destination is the VPC network's IP range.
- The **target** is the connection through which to send the traffic. In this case, the traffic is routed through the local VPC network.
- The **destination** is a range of IP addresses where you want your traffic to go. In the example of sending a letter, you need a destination to route the letter to the appropriate place. The same is true for routing traffic. In this case, the destination is the VPC network's IP range.
- The **target** is the connection through which to send the traffic. In this case, the traffic is routed through the local VPC network.
- The **destination** is a range of IP addresses where you want your traffic to go. In the example of sending a letter, you need a destination to route the letter to the appropriate place. The same is true for routing traffic. In this case, the destination is the VPC network's IP range.
- The **target** is the connection through which to send the traffic. In this case, the traffic is routed through the local VPC network.

Elastic Load Balancing is the AWS service that automatically distributes incoming application traffic across multiple resources, such as Amazon EC2 instances.
ELB is single point of contact for upcoming requests which then distribute to diff ec2 instance.

Monolithic: application or component are tightly coupled in a single system so if one fail whole system goes down
Microservice: application components are loosely coupled. In this case, if a single component fails, the other components continue to work because they are communicating with each other.

We use microservices in Aws

AWS use **msg queue** to buffer the request so that no request is lost.

Amazon Simple Notification Service (Amazon SNS) and Amazon Simple Queue Service (Amazon SQS).

Amazon Simple Notification Service (Amazon SNS) is a publish/subscribe service. Using Amazon SNS topics, a publisher publishes messages to subscribers, emails.

In Amazon SNS, subscribers can be web servers, email addresses, AWS Lambda functions, or several other options.Subscribers are of different type.

Amazon Simple Queue Service (Amazon SQS) is a message queuing service.

Using Amazon SQS, you can send, store, and receive messages between software components, without losing messages or requiring other services to be available. In Amazon SQS, an application sends messages into a **queue**. A user or service retrieves a message from the queue, processes it, and then deletes it from the queue.

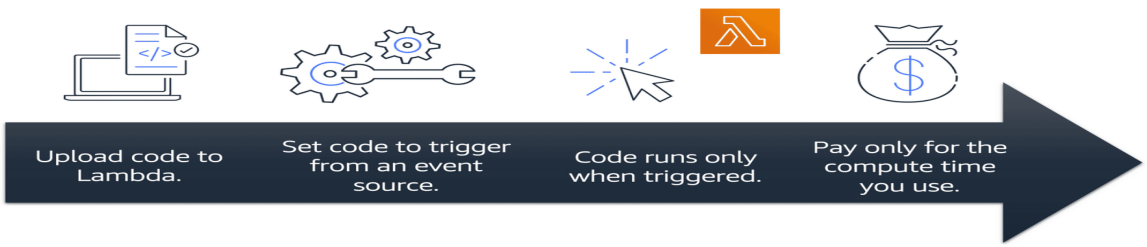
With EC2 we need to manage instances like scaling the service, patching instances when new software packages come out.

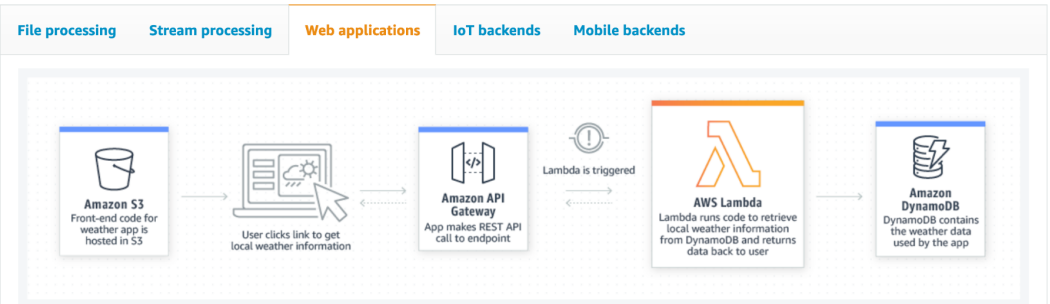
to solve this we have **serverless means that your code runs on servers, but you do not need to provision or manage these servers, automatic scaling take place**

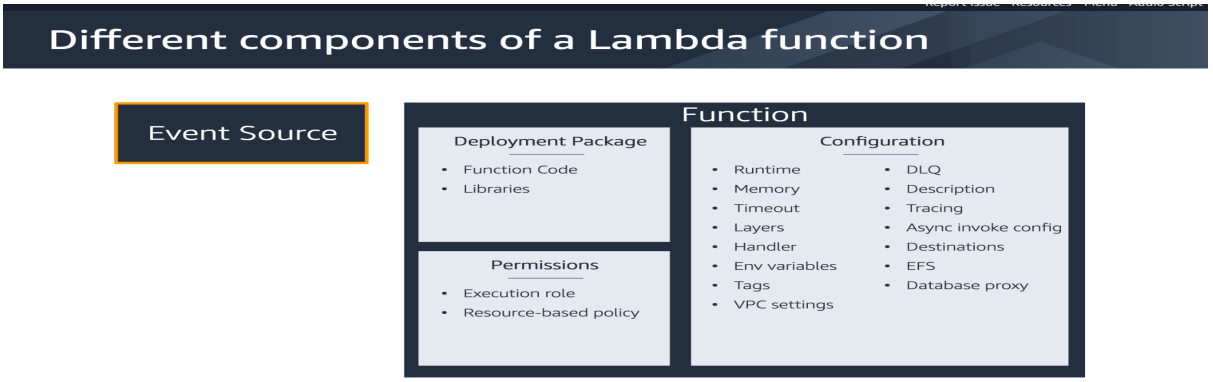
An AWS service for serverless computing is **AWS Lambda**.

AWS Lambda is a service that lets you run code without needing to provision or manage servers./serverless.

Serverless event driven computer service that lets you run code virtually any type of application or backend service without managing service.







Lambda execution env: env where function run. Config max execution duration . Provide temp disk space, cpu power proportion amount of memory configured
The configuration of a Lambda function consists of information that describes how the function should run. In the configuration, you specify network placement, environment variables, memory, invocation type, permission sets, and other configurations. To dive deeper into these configurations, check out the resources section.

Triggers describe when a Lambda function should run.

The AWS Lambda function handler is the method in your function code that processes events. When your function is invoked, Lambda runs the handler method. When the handler exits or returns a response, it becomes available to handle another event.

A function handler can be any name; however, the default on the Lambda console is lambda_function.lambda_handler. This name reflects the function name as lambda_handler, and the file where the handler code is stored in lambda_function.py.

Containers provide you with a standard way to package your application's code and dependencies into a single object. You can also use containers for processes and workflows in which there are essential requirements for security, reliability, and scalability.

Amazon Elastic Container Service (Amazon ECS) is a highly scalable, high-performance container management system that enables you to run and scale containerized applications on AWS.

Amazon ECS supports Docker containers. Docker is a software platform that enables you to build, test, and deploy applications quickly
Containers share the same operating system and kernel as the host they exist on, whereas virtual machines contain their own operating system.
Amazon Elastic Kubernetes Service (Amazon EKS) is a fully managed service that you can use to run Kubernetes on AWS.

software development kits (SDKs). SDKs make it easier for you to use AWS services through an API designed for your programming language or platform. SDKs enable you to use AWS services with your existing applications or create entirely new applications that will run on AWS.

Kubernetes is open-source software that enables you to deploy and manage containerized applications at scale.

Ways to interact with aws services

The **AWS Management Console** is a web-based interface for accessing and managing AWS services. You can quickly access recently used services and search for other services by name, keyword, or acronym. The console includes wizards and automated workflows that can simplify the process of completing tasks.
But to start service, or monitor you need to manually click different buttons every time

ASW command line interface (cli) for making api call.AWS CLI enables you to control multiple AWS services directly from the command line within one tool.automate the actions that your services and applications perform through scripts. For example, you can use commands to launch an Amazon EC2 instance, connect an Amazon EC2 instance to a specific Auto Scaling group, and more
.
software development kits (SDKs). SDKs make it easier for you to use AWS services through an API designed for your programming language or platform. SDKs enable you to use AWS services with your existing applications or create entirely new applications that will run on AWS.

With **AWS Elastic Beanstalk**, you provide code and configuration settings, and Elastic Beanstalk deploys the resources necessary to perform the following tasks:

- Adjust capacity
- Load balancing
- Automatic scaling
- Application health monitoring

AWS CloudFormation

With **AWS CloudFormation**, you can treat your infrastructure as code. This means that you can build an environment by writing lines of code instead of using the AWS Management Console to individually provision resources.

AWS CloudFormation provisions your resources in a safe, repeatable manner, enabling you to frequently build your infrastructure and applications without having to perform manual actions. It determines the right operations to perform when managing your stack and rolls back changes automatically if it detects errors.

AWS Output: allow to use aws resource in your own data centers

Amazon virtual private cloud(vpc)allow to set boundaries around your AWS resources .
A **subnet** is a section of a VPC that can contain resources such as Amazon EC2 instances.

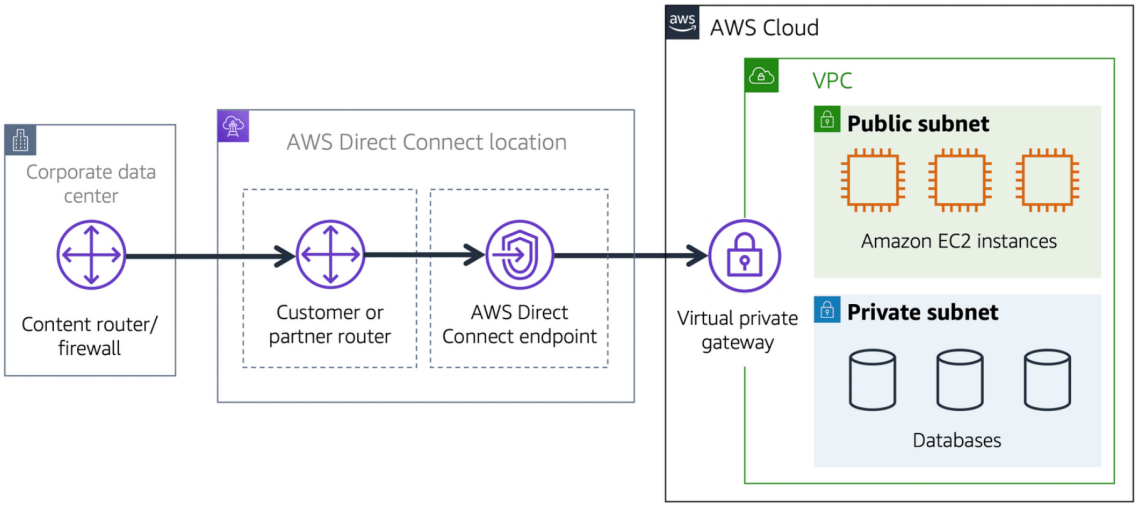
Internet gateway

An internet gateway is a connection between a VPC and the internet
The virtual private gateway is the component that allows protected internet traffic to enter into the VPC

A virtual private gateway enables you to establish a virtual private network (VPN) connection between your VPC and a private network, such as an on-premises data center or internal corporate network. A virtual private gateway allows traffic into the VPC only if it is coming from an approved network.

AWS Direct Connect

enables you to establish a dedicated private connection between your data center and a VPC.



Public Subnet: contain resource that public can access.
Private Subnet: contain resource hat should be accessible only through your private network, such as a database that contains customers' personal information and order histories.
A **packet** is a unit of data sent over the internet or a network.
The VPC component that checks packet permissions for subnets is a **network access control list (ACL)**.(virtual firewall). They are stateless and they remember nothing and check packets that cross the subnet border each way: inbound and outbound.

Default network ACL: allows all inbound and outbound traffic, but you can modify it by adding your own rules

Custom Network ACL: ll inbound and outbound traffic is denied until you add rules to specify which traffic to allow.

The VPC component that checks packet permissions for an Amazon EC2 instance is a **security group**.
By default a security group denies all inbound traffic and allows all outbound traffic. You can add custom rules to configure which traffic to allow or deny. S**tateful** packet filtering. They remember previous decisions made for incoming packets.

A NAT gateway is a Network Address Translation (NAT) service. You can use a NAT gateway so that instances in a private subnet can connect to services outside your VPC but external services cannot initiate a connection with those instances.

Amazon Route 53: DNS service. Connects user requests to infrastructure running in AWS (such as Amazon EC2 instances and load balancers). It can route users to infrastructure outside of AWS.It also register new domain names directly in Route 53.

Amazon route s3 and cloudfront works together

client send req data from a particular company site. Amazon route s3 convert company site to ip add and send back to client . Then this ip add is taken by cloudfront and search fpr nearest edge location and connect to load balancer to send req packet to amazon ec2 instance

Block-level storage volumes behave like physical hard drives.
While file storage treats files as a singular unit, block storage splits files into fixed-size chunks of data called blocks that have their own addresses. Since each block is addressable, blocks can be retrieved efficiently.

Instance store provide temporary block level storage for an amazon ec2 instance. It is a disk **attached directly to host computer of EC2 instance** and have life time as that of instance. As on stopping an instance and running it again it start on different host and hence all data is lost. It is faster.
It's also ideal for temporary storage of information that changes frequently, such as buffers, caches, scratch data, and other temporary content.

Amazon Elastic Block Storage: block level storage for ec2 instance that remain available after termination of ec2 instance. it stores data in single AZ. It is connected externally to ec2 instance and can be connected to more than 1 instance ec2 but not for all ebs and instance must be in same AZ.
or EBS volumes, the maximum amount of storage you can have is 16 TB.
To create an EBS volume, you define the configuration (such as volume size and type) and provision it. After you create an EBS volume, it can attach to an Amazon EC2 instance.
Because EBS volumes are for data that needs to persist, it's important to back up the data. You can take incremental backups of EBS volumes by creating Amazon EBS snapshots.

An **EBS snapshot** is an incremental backup(that means on next backup only change content is backed up)
Amazon EBS is meant for data that changes frequently and needs to persist through instance stops, terminations, or hardware failures. Amazon EBS has two types of volumes – SSD-backed volumes and HDD-backed volumes.

Amazon Simple Storage Service (Amazon S3)

provide object level storage means data is stored in forme of object.
Any type of data can be stored, unlimited capacity , but one object can be of max 5 TB, can manage permission for visibility and access, can track changes as well.
a standalone **storage solution that isn't tied to compute.** It enables you to retrieve your data from anywhere on the web.

Amazon S3 storage classes

pay only for you use.
Different S3 storage classes
1Standard S3 : requently accessed data, Stores data in a minimum of three Availability Zones, provide high avaialbity of object , highest cost.

S3 Standard-Infrequent Access (S3 Standard-IA)

- deal for infrequently accessed data
- Similar to S3 Standard but has a lower storage price and higher retrieval price
- data in a minimum of three Availability Zones.
- same avaialbirty as standard s3

S3 One Zone-Infrequent Access (S3 One Zone-IA)

- Stores data in a single Availability Zone

- Has a lower storage price than S3 Standard-IA

S3 Intelligent-Tiering

–

- Ideal for data with unknown or changing access patterns
- Requires a small monthly monitoring and automation fee per object

S3 Glacier(archieved data lesser used data old photos)

- Low-cost storage designed for data archiving
- Able to retrieve objects within a few minutes to hours

S3 Glacier Deep Archive

–

- Lowest-cost object storage class ideal for archiving
- Able to retrieve objects within 12 hours

S3 bucket

they are 99 .999 999 999% durable,

Amazon Elastic File System (Amazon EFS) is a scalable file system used with AWS Cloud services and on-premises resources(through aws direct connect). a storage server uses block storage with a local file system to organize files. Clients access data through file paths. used when large nouber of resource service need to access the data, it is auto scalable (shrink aur expand)without effecting application.Regional service, stores data in multiple AZ,

Amazon Relational Database Service (Amazon RDS) is a service that enables you to run relational databases in the AWS Cloud.automates tasks such as hardware provisioning, database setup, patching, and backups. Amazon RDS database engines offer encryption at rest (protecting data while it is stored) and encryption in transit (protecting data while it is being sent and received).

Amazon RDS database engines

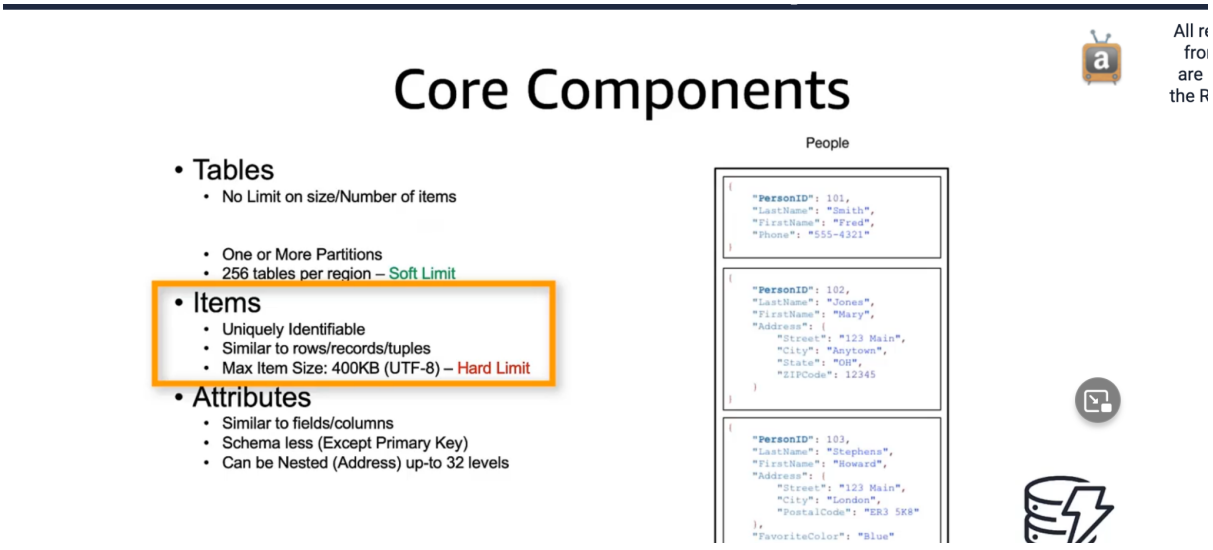
- Amazon Aurora
- PostgreSQL
- MySQL
- MariaDB
- Oracle Database
- Microsoft SQL Server

> **Amazon Aurora** is an enterprise-class relational database. It is compatible with standard. MySQL and PostgreSQL relational databases.5 time faster than mysql and 3 times faster than postgrelsql. Replicated copies to diff az and also done backup in s3 bucket.

In a **nonrelational database**, you create tables(not in form of rows and column but in other form may be in key value) A table is a place where you can store and query data.

Amazon DynamoDB is a key-value database service. It delivers single-digit millisecond performance at any scale.

DynamoDB is **serverless**, which means that you do not have to provision, patch, or manage servers.You also do not have to install, maintain, or operate software. auto scaling.



DynamoDB also offers encryption at rest, which eliminates the operational burden and complexity involved in protecting sensitive data. For more information, see DynamoDB Encryption at Rest.

Amazon Redshift is a data warehousing service that you can use for big data analytics. It offers the ability to collect data from many sources and helps you to understand relationships and trends across your data.

AWS Database Migration Service (AWS DMS) enables you to migrate relational databases, nonrelational databases, and other types of data stores.

With AWS DMS, you move data between a source database and a target database(can be of same type or diff) source can be data on your prmise, on ec2 instance on one amazon rds). target can be db on ec2 instance or oamazon rds.

source data remain operational during migration.

when data of source and target are diff. then there are mainly two steps.

1 conversion of schema using AWS Schema Conversion Too

2 actual transfer of database.

we can consolidate serval db in single central db.

Enabling developers to test applications against production data without affecting production users

other db category

1. **Amazon DocumentDB** is a document database service that supports MongoDB workloads.

2. **Amazon Neptune** is a graph database service.

You can use Amazon Neptune to build and run applications that work with highly connected datasets, such as recommendation engines, fraud detection, and knowledge graphs.

3. **Amazon Managed Blockchain** is a service that you can use to create and manage blockchain networks with open-source frameworks

4.**Amazon ElastiCache** is a service that adds caching layers on top of your databases to help improve the read times of common requests.

5.Amazon DynamoDB Accelerator (DAX) is an in-memory cache for DynamoDB.

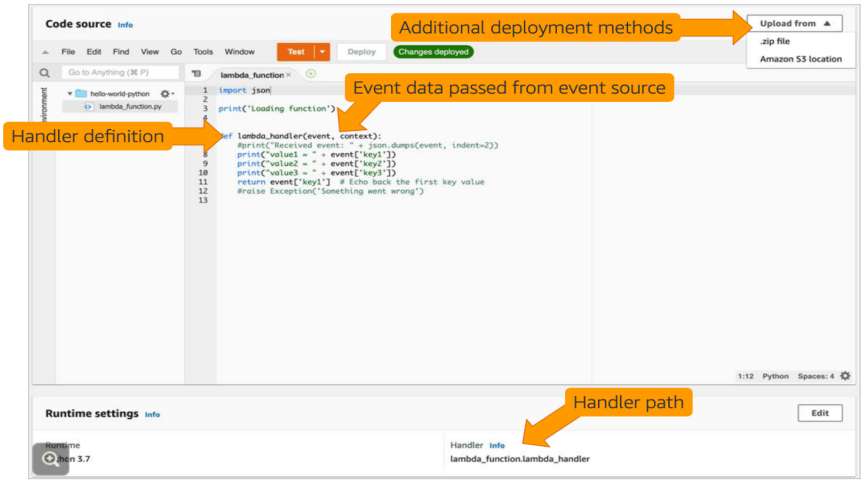
It helps improve response times from single-digit milliseconds to microseconds.

Database Type	Use Cases	AWS Service
Relational	Traditional applications, ERP, CRM, e-commerce	Amazon RDS, Amazon Aurora, Amazon Redshift
Key-value	High-traffic web apps, e-commerce systems, gaming applications	Amazon DynamoDB
In-memory	Caching, session management, gaming leaderboards, geospatial applications	Amazon ElastiCache for Memcached, Amazon ElastiCache for Redis
Document	Content management, catalogs, user profiles	Amazon DocumentDB (with MongoDB compatibility)
Wide column	High-scale industrial apps for equipment maintenance, fleet management, and route optimization	Amazon Keyspaces (for Apache Cassandra)
Graph	Fraud detection, social networking, recommendation engines	Amazon Neptune
Time series	IoT applications, DevOps, industrial telemetry	Amazon Timestream
Ledger	Systems of record, supply chain, registrations, banking transactions	Amazon QLDB

The shared responsibility model divides into customer responsibilities (commonly referred to as “security in the cloud”) and AWS responsibilities (commonly referred to as “security of the cloud”).

Lamda details

Function code



Lambda permission model : execution role : define action that lambda function can perform.
Resource based policy: service who can invoke lamda fn. Identify based permission : caller principle role with permission to invoke lambda

Invoke opn : invoke lambda function require caller to have permission for lambda.
Invocation type: synchronous : requestresponse
Asynchronous: event

5xx : error need to redo
Raise exception time out

Lambda can invoke fn multiple time on encountering error
In push based event resource : no automatically retrieve on error
In poll based : stream based (Dynamo db) retry till success
Non stream based : msg will return to queue

Automatically scaling to increase traffic
Invocation limit : account concurrency limit can be distributed on different fn. If reaching the limit then fn is throttle.
Version: immutable snapshot of lambda fn
$Latest version: version which is mutable
Alias pointer to version
Can be changed to point to diff version

The act of collecting, analyzing, and using data to make decisions or answer questions about your IT resources and systems is called **monitoring.**

Amazon CloudWatch is a monitoring and observability service that collects data like metrics, logs, network traffic, events and more.

You can use CloudWatch to:

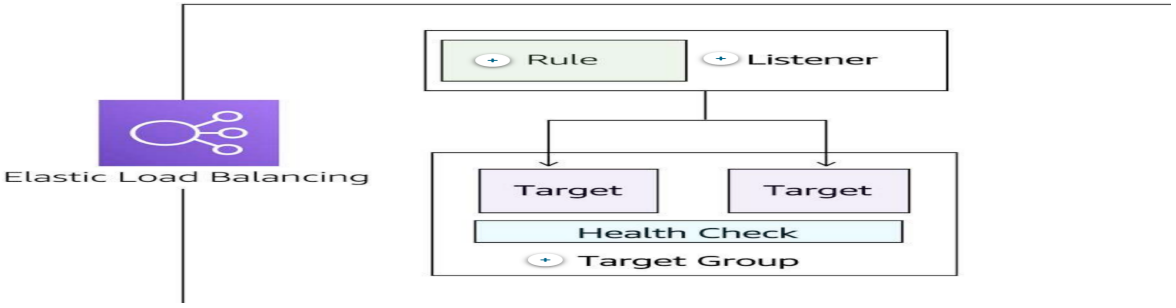
- Detect anomalous behavior in your environments
- Set alarms to alert you when something is not right
- Visualize logs and metrics with the AWS Management Console
- Take automated actions like scaling

Custom metrics allows you to publish your own metrics to CloudWatch.

If you want to gain more granular visibility, you can use high-resolution custom metrics, which enable you to collect custom metrics down to a 1-second resolution. This means that you can send one data point per second per custom metric.

CloudWatch can also be the centralized place for logs to be stored and analyzed, using Amazon CloudWatch Logs. CloudWatch Logs can monitor, store, and access your log files from applications running on Amazon EC2 instances, AWS Lambda functions, and other sources.
You can create CloudWatch alarms to automatically initiate actions based on sustained state changes of your metrics. You configure when alarms are triggered and the action that is performed.

ELB components



lListener: client connect to listener(refered as client side). to define listner need port and protocol depending on load balancer type. more than listner can be connect to load balancer

Rule: to associate target to listner rules ae defined. these are condition that can be source ip and condition which target group to send traffic

target: type of backend where you want traffic to dirwct like lamda fn, ec2 instance

load balancing
1 application lb(http, https)
2 network lb(tcp, udp, tls)

Step Function: help to add resilient workflow automation to your application-without writitng code.Workflows built with Step Functions include built-in error handling, parameter passing, recommended security settings, and state management, reducing the amount of code you have to write and maintain.
The workflows you build with Step Functions are called **state machines**, and each step of your workflow is called a **state**.A task is a state in a workflow that represents a single unit of work that another AWS service performs. Each step in a workflow is a state.
Through Step Functions' graphical console, you see your application's workflow as a series of event-driven steps.

The Amazon EC2 Auto Scaling service can take care of that task by automatically creating and removing EC2 instances based on metrics from Amazon CloudWatch.(horizontal scaling, active-active)

Configure EC2 Auto Scaling components

Three main components of EC2 Auto Scaling are as follows:

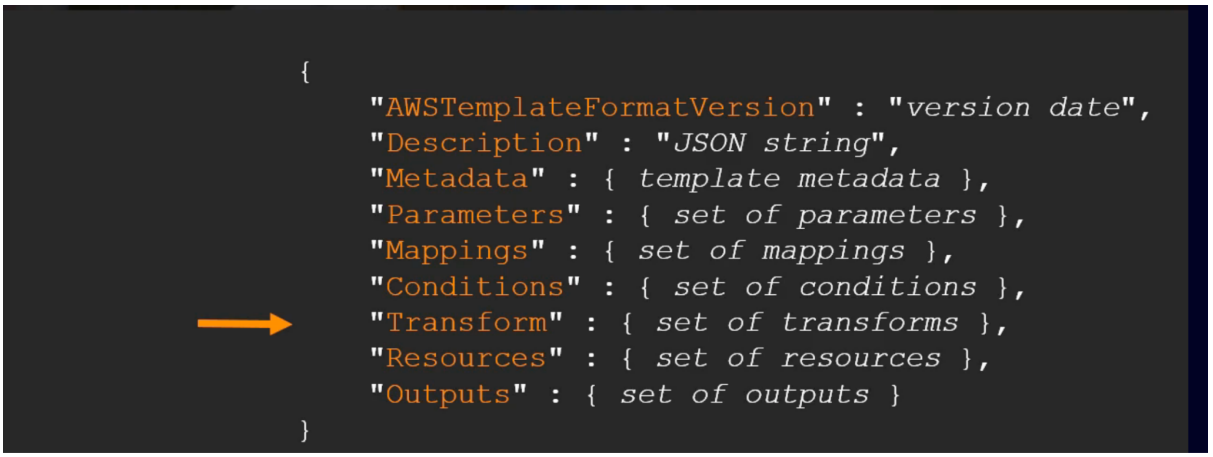
- **Launch template or configuration:** What resource should be automatically scaled?
- **EC2 Auto Scaling Group:** Where should the resources be deployed?
- **Scaling policies:** When should the resources be added or removed?

CloudFormation can help you manage your AWS resources, especially resources that depend on each other. You can use CloudFormation to group your resources into **stacks**, using declarative **templates**. CloudFormation can also help you manage creating, updating,

and deleting the resources within a stack. You can create resources in parallel, if possible, or create them in specific orders, if they depend on each other.

What are the basic technical concepts of CloudFormation?

- **Resources –** Any of the things you can create within AWS, which includes things like Amazon Simple Storage Service (Amazon S3) buckets, Amazon Elastic Compute Cloud (Amazon EC2) instances, or Amazon Simple Queue Service (Amazon SQS) queues.
- **Templates** – Text-based (JSON or YAML) descriptions of CloudFormation stacks that you can use to define all of your resources, including which resources depend on each other.
- **Stack** – A collection of AWS resources that you can manage as a single unit.
- **StackSet** – A named set of stacks that use the same template, but applied across different accounts and Regions. You can create, update, or delete stacks across multiple accounts and Regions with a single operation
- **change set:** can be generated before updating stack. Allow to see changes and its impacton your running resources



```
{
    "Resources" : {
        "Logical ID" : {
            "Type" : "Resource type",
            "Properties" : {
                … set of properties ..
            }
        }
    }
}
```

eg(json)

```
{
    "Resources" : {
        "MyEC2Instance" : {
            "Type" : "AWS::EC2::Instance",
            "Properties" : {
                "ImageId" : "ami-43874721",
                "InstanceType" : "t2.micro",
            }
        }
    }
}
```

intrinsic fn: built in fn to manage stack

eg:join

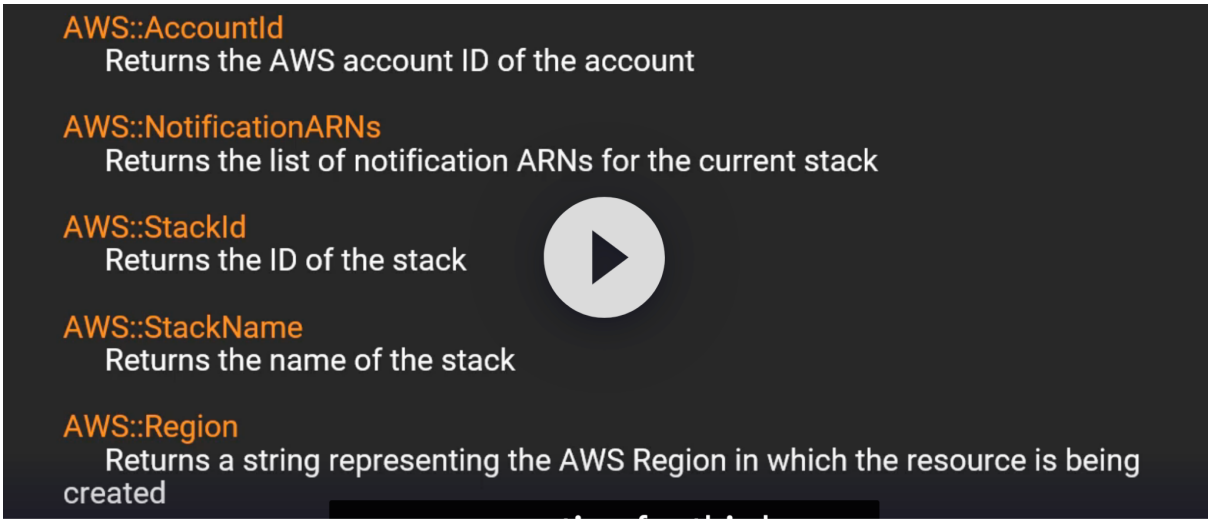
JSON

```
{"Fn::Join" : ["delimiter", [ comma-delimited list of values ]]}

{"Fn::Join" : [ ":", [ "a", "b", "c" ] ]}
```

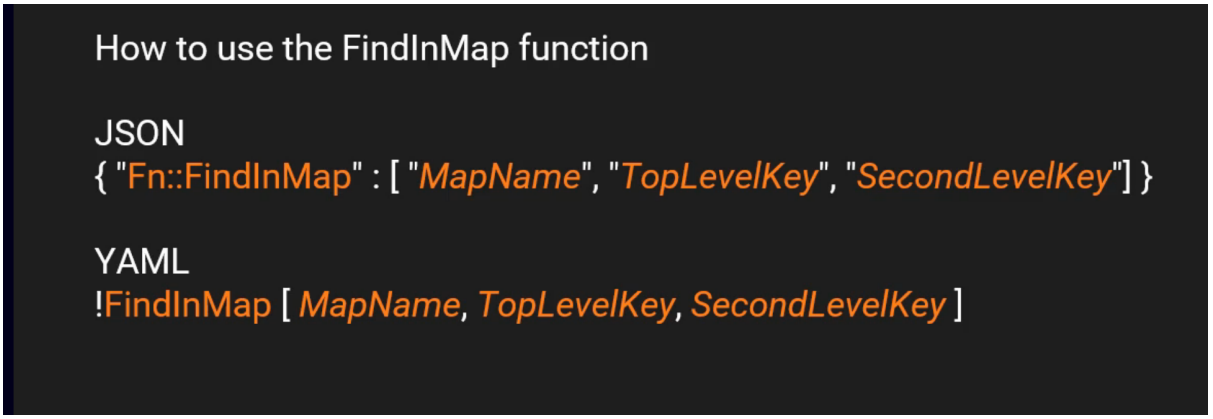
YAML

```
!Join [ delimiter, [ comma-delimited list of values ] ]

!Join [ ":", [ a, b, c ] ]  = "a:b:c"
```

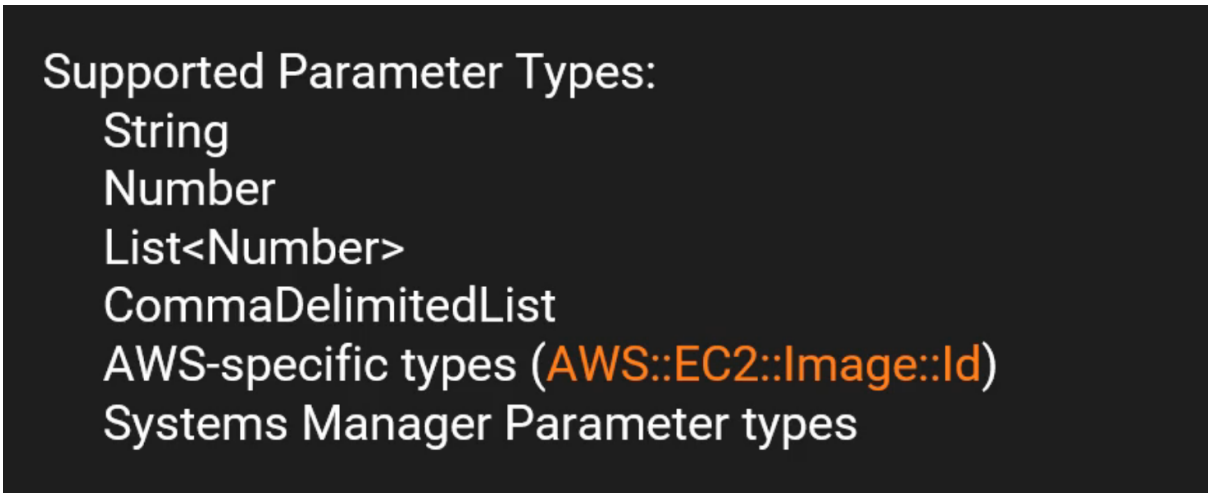
Pseudo Parameter: paramater that are predefined by cloud formation
use Ref intrinsic fn to referenc the paramater

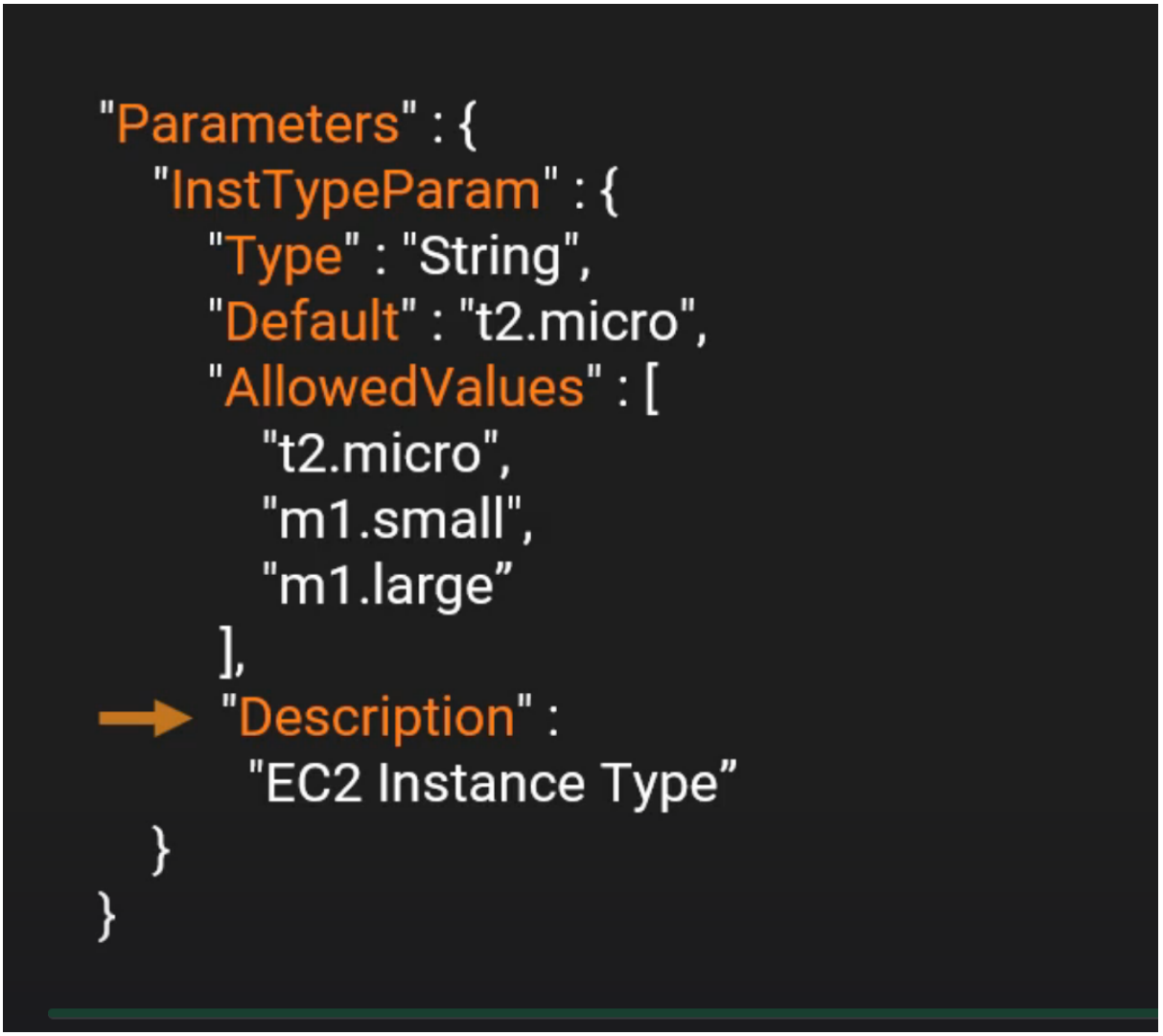


Mapping: use input value to determine output value

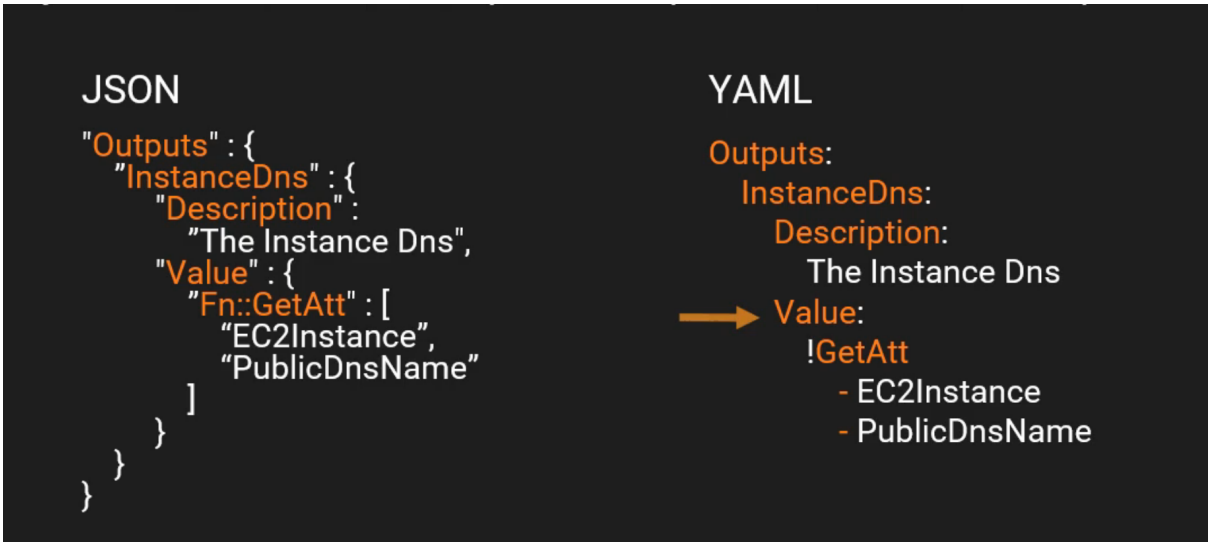


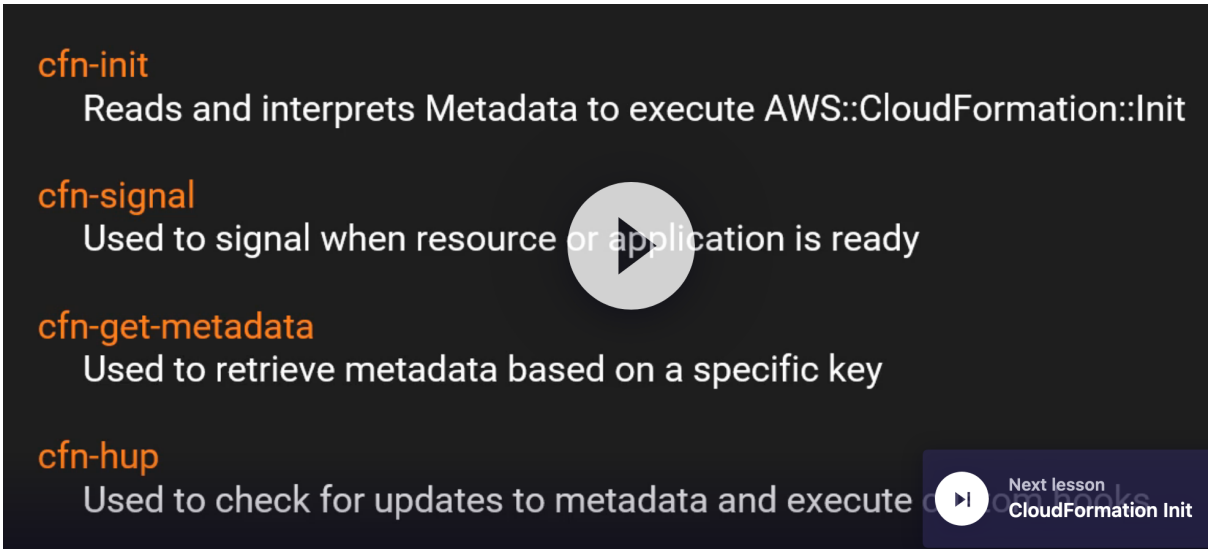
Input Paramater : to input custom value to our template





Output enable to get access to info about resources within stack




cfn-init
Reads and interprets Metadata to execute AWS::CloudFormation::Init
cfn-signal
Used to signal when resource or application is ready
cfn-get-metadata
Used to retrieve metadata based on a specific key
cfn-hup
Used to check for updates to metadata and execute
Next lesson
CloudFormation Init