**AWS Cheatsheets**

**EC2 Cheatsheet**

* Elastic Compute Cloud (EC2) is a cloud computing service. With Amazon EC2 you launch virtual server instances on the AWS cloud.
* Configure your EC2 by choosing your OS, storage, memory, network throughput
* EC2 comes in a variety of Instance Types specialized for different roles:
* **General-purpose:** balance compute, memory, and networking resources
* **Compute Optimized:** Ideal for compute-bound applications that benefit from the high-performance processor
* **Memory-Optimized:** fast performance for workloads that process large data sets in memory
* **Accelerated Optimized:** hardware accelerators, or co-processors
* **Storage optimized:** high, sequential read and write access to very large data sets on local storage
* Instance sizes generally double in price and key attributes
* You can access your EC2 instance with SSH via keypair given by AWS. A key pair consists of a public key that AWS stores, and a private key file that you store. For Windows AMIs, the private key file is required to obtain the password used to log into your instance.
* Placement Groups let you to choose the logical placement of your instances to optimize for communication, performance or durability. Placement groups are free
* Userdata a script that will be automatically run when launching an EC2 instance.
* Metadata metadata about the current instance. You access this metadata via local endpoint when ssh into the EC2 instance. eg. curl http://169.254.169.254/latest/meta-data metadata could be the instance id, current IP or public IP, etc.
* User data is data that is supplied by the user at instance launch in the form of a script.
* Instance user data is available at http://169.254.169.254/latest/user-data.
* User data and metadata are not encrypted.
* **Instances Profiles:** This is an IAM role that you can use to pass role information to an EC2 instance when the instance starts
* EC2 compute units (ECU) provide the relative measure of the integer processing power of an Amazon EC2 instance.

### EC2 Pricing

* EC2 has for 4 pricing models On-demand, Spot, Reserved Instances (RI) and Dedicated
* You are limited to running up to a total of 20 On-Demand instances across the instance family, purchasing 20 Reserved Instances, and requesting Spot Instances per your dynamic spot limit per region (by default).

**On-demand (least commitment)**

* Low cost and flexible
* only pay per hour with no commitment
* Use case: short term, spiky, unpredictable workloads, first-time apps. Good for dev/test
* Ideal when your workloads cannot be interrupted. In addition, ideal for auto-scaling groups.

**Reserved Instances (RI) up to 75% off (Best long-term value)**

* Use case: stay state or predictable usage, ideal for applications that need the reserved capacity
* Can resell unused reserved instances (Reserved Instance Marketplace)
* Reduced Pricing is based on Term x Class Offering x Payment Option
* If you don’t need your RI’s, you can try to sell them on the Reserved Instance Marketplace
* Payment Terms: 1 year or 3 years
* Payment Options: All Upfront, Partial Upfront, and No upfront
* Can be shared across multiple accounts within Consolidated Billing
* Billed whether running or not
* Can switch AZ within the same region
* Class Offerings:
  + Standard: Up to 75% reduced pricing compared to on-demand. Cannot change RI Attributes
  + Convertible: Up to 54% reduced pricing compared to on-demand. Allows you to change RI Attributes if greater or equal in value
  + Scheduled: You reserve instances for specific time periods eg. once a week for a few hours. Saving vary. Scheduled = reserved for specific periods of time, accure charges hourly, billed in monthly increments over the term (1 year)

**Spot Pricing up to 90% off (Biggest Savings)**

* Request spare computing capacity
* Flexible start and end times
* Use Case: Can handle interruptions (server randomly stopping and starting). ideal For non-critical background jobs. Perfect for testing.
* You can use Spot Instances for various stateless, fault-tolerant, or flexible applications such as big data, containerized workloads, CI/CD, web servers, high-performance computing (HPC), and other test & development workloads.
* Spot Instances receive a two-minute interruption notice when these instances are about to be reclaimed by EC2, because EC2 needs the capacity back.
* Instances can be terminated by AWS at anytime
* To reduce the impact of interruptions and optimize Spot Instances, diversify and run your application across multiple capacity pools.
* To further reduce the impact of interruptions, you can also set up Spot Instances and Spot Fleets to respond to an interruption notice by stopping or hibernating rather than terminating instances when capacity is no longer available.
* If your instance is terminated by AWS, you don’t get charged for partial hour of usage
* If you terminate an instance you will still be charged for any hour that it ran.
* New pricing model: The Spot price is determined by long-term trends in supply and demand for EC2 spare capacity.

**Dedicated Hosting (Most Expensive)**

* Dedicated servers
* Complete isolation
* Can be on-demand or reserved (up to 70% off)
* Predictable performance
* Use case: When you need a guarantee of isolation hardware (enterprise requirements). In addition, Good for regulatory compliance or licensing requirements.

**Saving Plan:**

* Savings Plans is a flexible pricing model that helps you save up to 72 percent on Amazon EC2, AWS Fargate, and AWS Lambda usage.
* Savings Plans provide you lower prices for your Amazon EC2 usage, Fargate, and Lambda in exchange for a commitment to a consistent usage amount (measured in $/hour) for a one or three-year term.
* For example, if you commit to $10/hour of compute usage, your usage is charged at your Savings Plans rate up to $10. Any usage beyond your Savings Plans commitment is charged at your regular On-Demand rates.
* There are two types of Saving Plan:
  + **Compute Savings Plans** provide the most flexibility and prices that are up to 66 percent off of On-Demand rates. These plans automatically apply to your EC2 instance usage, **regardless of instance family** (for example, M5, C5, etc.), instance sizes (for example, c5.large, c5.xlarge, etc.), Region (for example, us-east-1, us-east-2, etc.), operating system (for example, Windows, Linux, etc.), or tenancy (for example, Dedicated, default, dedicated host). They also apply to your **Fargate and Lambda** usage.
  + **EC2 Instance Savings Plans** provide savings up to 72 percent off On-Demand, in exchange for a commitment to a specific **instance family in a chosen AWS Region** (for example, M5 in Virginia). With an EC2 Instance Savings Plan, you can change your instance size **within the instance family**.

For the latest information about Saving Plan and Comparing of Savings Plans and Reserved Instance , see [Click here](https://docs.aws.amazon.com/savingsplans/latest/userguide/what-is-savings-plans.html) .

### EC2 AMI

* Amazon Machine Image (AMI) provides the information required to launch an instance.
* AMIs are region-specific, if you need to use an AMI in another region you can copy an AMI into the destination region via Copy AMI
* You can create an AMI from an existing EC2 instance that’s either running or stopped
* Community AMI are free AMIs maintained by the community
* AWS Marketplace free or paid subscription AMIs maintained by vendors
* AMIs have AMI ID. The same AMI eg (Amazon Linux 2) will vary in both AMI ID options eg. Architecture options in different regions
* An AMI holds the following information:
  + A template for the root volume for instance (EBS Snapshot or Instance Store template) eg. an operating system, an application server, and applications
  + Launch permissions that control which AWS accounts can use the AMI to launch instances
  + A block device mapping that specifies the volumes to attach to the instance when it’s launched

### EC2 Networking

**IP Addresses**

* There are three types of IP address that can be assigned to an Amazon EC2 instance:
  + Public – a public address that is assigned automatically to instances in public subnets and reassigned if the instance is stopped/started.
  + Private – Private address assigned automatically to all instances.
  + Elastic IP – a public address that is static.
* Public IPv4 addresses are lost when the instance is stopped but private addresses (IPv4 and IPv6) are retained.
* Public IPv4 addresses are retained if you restart the instance.
* Elastic IPs are retained when the instance is stopped.
* Elastic IP addresses are static public IP addresses that can be remapped (moved) between instances.
* All accounts are limited to 5 elastic IPs per region by default.
* AWS charges for elastic IPs when they’re not being used.
* An Elastic IP address is for use in a specific region only.
* Public IP addresses are assigned for instances in public subnets (VPC).
* Public IP addresses are always assigned for instances in EC2-Classic.
* DNS records for elastic IP’s can be configured by filling out a form.
* You can choose whether secondary IP addresses can be reassigned.
* You can assign or remove IP addresses from EC2 instances while they are running or stopped.
* All IP addresses (IPv4 and IPv6) remain attached to the network interface when detached or reassigned to another instance.

**Elastic Network Interfaces**

* An elastic network interface (referred to as a network interface in this documentation) is a logical networking component in a VPC that represents a virtual network card.
* You can create and configure network interfaces in your account and attach them to instances in your VPC.
* eth0 is the primary network interface and cannot be moved or detached.
* By default, eth0 is the only Elastic Network Interface (ENI) created with an EC2 instance when launched.
* You can add additional interfaces to EC2 instances (number dependent on instances family/type).
* An ENI is bound to an AZ and you can specify which subnet/AZ you want the ENI to be added in.
* You can specify which IP address within the subnet to configure or leave it be auto-assigned.
* You can only add one extra ENI when launching but more can be attached later.
* ENIs can be “hot attached” to running instances.
* ENIs can be “warm-attached” when the instance is stopped.
* ENIs can be “cold-attached” when the instance is launched.
* If you add a second interface AWS will not assign a public IP address to eth0 (you would need to add an Elastic IP).
* Default interfaces are terminated with instance termination.
* Manually added interfaces are not terminated by default.
* You can change the termination behavior.

**Enhanced Networking – Elastic Network Adapter (ENA)**

* Enhanced networking provides higher bandwidth, higher packet-per-second (PPS) performance, and consistently lower inter-instance latencies.
* Enhanced networking is enabled using an Elastic Network Adapter (ENA).
* If your packets-per-second rate appears to have reached its ceiling, you should consider moving to enhanced networking because you have likely reached the upper thresholds of the VIF driver.
* AWS currently supports enhanced networking capabilities using SR-IOV.
* SR-IOV provides direct access to network adapters, provides higher performance (packets-per-second) and lower latency.
* Only available for certain instance types. Must launch an HVM AMI with the appropriate drivers.

**Elastic Fabric Adapter (EFA)**

* An Elastic Fabric Adapter is an AWS Elastic Network Adapter (ENA) with added capabilities.
* An EFA can still handle IP traffic but also supports an important access model commonly called OS bypass.
* This model allows the application (most commonly through some user-space middleware) to access the network interface without having to get the operating system involved with each message.
* Elastic Fabric Adapter (EFA) is a network interface for Amazon EC2 instances that enables customers to run applications requiring high levels of inter-node communications at scale on AWS.
* Its custom-built operating system (OS) bypass hardware interface enhances the performance of inter-instance communications, which is critical to scaling these applications.
* With EFA, **High-Performance Computing (HPC)** applications using the **Message Passing Interface (MPI)** and **Machine Learning (ML)** applications can scale to thousands of CPUs or GPUs.
* As a result, you get the application performance of on-premises HPC clusters with the on-demand elasticity and flexibility of the AWS cloud.
* EFA is available as an optional EC2 networking feature that you can enable on any supported EC2 instance at no additional cost.

**Comparision of ENI, ENA and EFA**

**When to use ENI:**

* This is the basic adapter type for when you don’t have any high-performance requirements.
* Can use with all instance types.

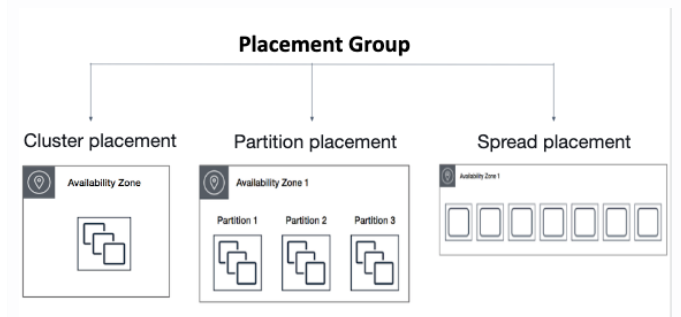
**When to use ENA:**

* Good for use cases that require higher bandwidth and lower inter-instance latency.
* Supported for limited instance types (HVM only).

**When to use EFA:**

* High-Performance Computing.
* MPI and ML use cases.
* Tightly coupled applications.
* Can use with all instance types.

### Placement Group



**Placement groups** are a logical grouping of instances in one of the following configurations.

**Cluster** – clusters instances into a low-latency group in a single AZ:

A cluster placement group is a logical grouping of instances within a single Availability Zone. Cluster placement groups are recommended for applications that benefit from low network latency, high network throughput, or both, and if the majority of the network traffic is between the instances in the group.

**Spread** – spreads instances across underlying hardware (can span AZs):

A spread placement group is a group of instances that are each placed on distinct underlying hardware. Spread placement groups are recommended for applications that have a small number of critical instances that should be kept separate from each other.

**Partition** — divides each group into logical segments called partitions:

Amazon EC2 ensures that each partition within a placement group has its own set of racks. Each rack has its own network and power source. No two partitions within a placement group share the same racks, allowing you to isolate the impact of hardware failure within your application. Partition placement groups can be used to deploy large distributed and replicated workloads, such as HDFS, HBase, and Cassandra, across distinct racks.

### EC2 Migration

VM Import/Export is a tool for migrating VMware, Microsoft, XEN VMs to the Cloud.

Can also be used to convert EC2 instances to VMware, Microsoft or XEN VMs.

Supported for:

* Windows and Linux.
* VMware ESX VMDKs and (OVA images for export only).
* Citrix XEN VHD.
* Microsoft Hyper-V VHD.

Can only be used via the API or CLI (not the console).

Stop the VM before generating VMDK or VHD images.

AWS has a VM connector plugin for vCenter:

* Allows migration of VMs to S3.
* Then converts into an EC2 AMI.
* Progress can be tracked in vCenter.

### EC2 Autoscaling

* An Autoscaling Group (ASG) is a collection of EC2 instances grouped for scaling and management
* Auto Scaling is a region-specific service.
* Auto Scaling can span multiple AZs within the same AWS region.
* There is no additional cost for Auto Scaling, you just pay for the resources (EC2 instances) provisioned.
* Auto Scaling works with ELB, CloudWatch, and CloudTrail.
* Scale up is when you increase the size of an instance (eg. updating launch configuration with large size)
* Size of an ASG is based on a Min, Max, and Desired Capacity
* **Target Scaling policy** scales based on when a target value for the metric is breached eg. Average CPU utilization exceeds 75%
* Triggered by an event of scaling action to either launch or terminate instances.
* **Simple Scaling policy** with steps is the new version of the simple scaling policy and allows you to create steps based on escalation alarm values
* **In scheduled Action** you determine a specific event or time to trigger auto-scaling.
* **Desire capacity** is how many EC2 instances you want to ideally run
* An ASG will always launch instances to meet minimum capacity
* **Health check** determine the current state of an instance you want to ideally run
* An ASG will always launch instances to meet minimum capacity
* **Cooldown Period:** The cooldown period is a configurable setting for your Auto Scaling group that helps to ensure that it doesn’t launch or terminate additional instances before the previous scaling activity takes effect. The default cooldown period which is 300 seconds is applied when you create your Auto Scaling group.
* Health checks can be run against either an ELB or the EC2 instance
* The ELBs must be in the same region.
* When an Autoscaling launches a new instance it uses a **Launch configuration** which holds the configuration values fort hat instances eg. AMI, Instance type, Role
* Launch Configurations cannot be edited and must be cloned or a new one created
* Launch Configurations must be manually updated by editing the autoscaling settings
* There are four scaling options:
  + **Maintain** – keep a specific or minimum number of instances running.
  + **Manual**– use maximum, minimum, or a specific number of instances.
  + **Scheduled** – increase or decrease the number of instances based on a schedule.
  + **Dynamic** – scale based on real-time system metrics (e.g. CloudWatch metrics).
* Can also scale based on an Amazon Simple Queue Service (SQS) queue.
  + Uses a custom metric that’s sent to Amazon CloudWatch that measures the number of messages in the queue per EC2 instance in the Auto Scaling group.
  + Then use a target tracking policy that configures your Auto Scaling group to scale based on the custom metric and a set target value. CloudWatch alarms invoke the scaling policy.
  + Use a custom “backlog per instance” metric to track not just the number of messages in the queue but the number available for retrieval.
  + Can base off the SQS Metric “ApproximateNumberOfMessages”.

### EC2 Elastic Load Balance (ELB)

* There are three Elastic Load Balancers(ELB): Network Load Balancer (NLB), Application Load Balancer (ALB) and Classic Load Balancer (CLB
* ALB must have at least 2 AZs
* For NLB only one subnet must be specified (recommended to add at least 2)
* For CLB you don’t need to specify any subnets unless you have “Enable advanced VPC configuration” enabled in which case you must specify two
* ELBs can be internet-facing or internal-only
* Security groups control the ports and protocols that can reach the front end listener
* Target Groups: Target groups are a logical grouping of targets (EC2 instances or ECS)
* A single target can be in multiple target groups
* A target group can only be associated with one load balancer
* ELB cannot go cross-region. You must create on per region
* Application Load Balancer (ALB) has listeners, Rules and Target Groups to route traffic
  + Operates at the request level
  + Routes based on the content of the request (layer7)
  + Supports path-based routing, host-based routing, query string parametry based routing, and source IP based routing
  + Supports IP address, Lambda functions, and containers as targets
* Network Load Balancer (NLB) use listeners and Target Groups to route traffic
  + Operates at the connection level
  + Routes connections based on IP protocol data (layer)
  + Offers ultra high performance, low latency, and TLS offloading at scale
  + can have static IP/Elastic IP
  + Supports UDP and static IP addresses as targets
* Classic Load balancer (CLB) use listeners and EC2 instances are directly registered as targets to CLB
  + Old generation; not recommended for new applications
  + Performs routing at Layer 4 and Layer 7
  + Use for existing applications running in EC2-Classic
* Application Load Balancer is for HTTP and HTTPS traffic and the name implies it good for Web Applications
* NLB is for TCP/UDP is good high network throughput eg. Video Games
* CLB is legacy and its recommended to use ALB or NLB
* Use X-forwarded-for (XFF) to get original IP of incoming traffic passing through ELB
* You can attach Web Application Firewall (WAF) to ALB but not to NLB or CLB
* You can attach Amazon Certification Manager SSL to any of the ELB for SSL
* ALB has advanced Request Routing rules where you can route based on subdomain header, path, and other HTTP or HTTPS information
* Sticky session can be enabled for CLB or ALB and session are remembered via Cookie

### Cross-Zone Load Balancing

With cross-zone load balancing, each load balancer node for your Classic Load Balancer distributes requests evenly across the registered instances in all enabled Availability Zones. If cross-zone load balancing is disabled, each load balancer node distributes requests evenly across the registered instances in its Availability Zone only.

* Cross-zone load balancing is enabled by default for CLB and ALB but not for NLB (when created through the console).
* Cross-zone load balancing is NOT enabled by default if the CLB is created from the CLI or API.
* You can enable or disable cross-zone load balancing on the CLB and NLB at any time.
* For the ALB, cross-zone load balancing is always on and cannot be disabled.
* When cross-zone load balancing is enabled, each load balancer node distributes traffic across the registered targets in all enabled Availability Zones.
* When cross-zone load balancing is disabled, each load balancer node distributes traffic across the registered targets in its Availability Zone only.
* Connection draining is enabled by default and provides a period of time for existing connections to close cleanly.
* When connection draining is in action a CLB will be in the status “InService: Instance deregistration currently in progress”.

## S3 Cheatsheet

### Simple Storage Service (S3)

* Simple Storage Service (S3) object-based storage. Store unlimited amount of data without worry about underlying storage infrastructure
* S3 is a persistent, highly durable data store.
* Use Cases
  + **Backup and Storage** – Provide data backup and storage services for others.
  + **Application Hosting** – Provide services that deploy, install, and manage web applications.
  + **Media Hosting** – Build a redundant, scalable, and highly available infrastructure that hosts video, photo, or music uploads and downloads.
  + **Software Delivery** – Host your software applications that customers can download.
  + **Static Website** – you can configure a static website to run from an S3 bucket.
* S3 is a universal namespace so names must be unique globally.
* An S3 bucket is region-specific.
* S3 replicates data across at least 3 AZs to ensure 99.99% availability and 11’ 9 durability
* Object contains your data (they’re like files)
* Object can be size anywhere from 0 bytes up to 5 Terabytes
* Buckets contain objects. The bucket can also contain folders which can in turn can contain objects.
* Bucket names are unique across all AWS accounts. Like a domain name.
* When you upload a file to S3 successfully you'll receive an HTTP 200 code
* **Lifecycle Management** objects can be moved between storage classes or objects can be deleted automatically based on a schedule
* **Versioning**, objects are giving a version ID. When objects are uploaded the old ones are kept. You can access any object version. When you delete an object the previous object is restored. Once versioning turned on it cannot be turned off, only suspended.
* MFA delete enforce DELETE operations to require an MFA token in order to delete an object. Must have versioning turned on to use. Can only turn on MFA delete from the AWS CLI. The root account is only allowed to delete objects
* All new buckets are private by default
* **Server Access Logging** provides detailed records for the request that are made to your bucket. By default, S3 does not collect server access logs.
* **Static Website Hosting** allows you to host a static website on Amazon S3.
* **Access control List (ACL)** enables you to manage access to buckets and objects. The ACL defines which AWS accounts (grantees) or pre-defined S3 groups are granted access and the type of access.
* ACLs are the legacy method (not deprecated) where you grant access to objects and buckets with actions
* Access auditing can be configured by configuring an Amazon S3 bucket to create access log records for all requests made against it
* Bucket Policies are JSON documents that let you write complex control access
* Security in Transit uploading files is done over SSL
* SSE stands for Server Side Encryption. S3 has 3 options for SSE
* **SSE-AES**: S3 handles the key,
  + Each object is encrypted with a unique key.
  + Encryption key is encrypted with a master key.
  + AWS regularly rotate the master key.
  + Uses AES 256 algorithm.
* **SSE-KMS**: Envelope encryption via AWS KMS and you manage the keys
  + KMS uses Customer Master Keys (CMKs) to encrypt.
  + Can use the automatically created CMK key.
  + OR you can select your own key (gives you control for the management of keys).
  + An envelope key protects your keys.
  + Chargeable.
* **SSE-C**: Customer provided key (you manage keys)
  + Client manages the keys, S3 manages encryption.
  + AWS does not store the encryption keys.
  + If keys are lost data cannot be decrypted.
* **Client-Side Encryption**: Clients must encrypt their own files before uploading them to S3.
* **Cross-Region Replication (CRR)** allows you to replicate files across regions for greater durability. You must have versioning turned on in the source and destination bucket. You can have CRR replicate to the bucket in another AWS Account
* **Transfer Acceleration** provides faster and secure uploads from anywhere in the World. Data is uploaded via a distinct url to an Edge Location. Data is then transported to your S3 bucket via the AWS backbone network. Transfer Acceleration leverages Amazon CloudFront’s globally distributed AWS Edge Locations.
* **Pre signed URLs** is a URL generated via the AWS CLI and SDK. It provides temporary access to write or download object data. Pre signed URLs are commonly used to access private objects.
* **Object Tags:** S3 object tags are key-value pairs applied to S3 objects which can be created, updated, or deleted at any time during the lifetime of the object.
* **Events:** Amazon S3 event notifications enable you to run workflows, send alerts, or perform other actions in response to changes in your objects stored in S3. you can enable certain S3 bucket events to send a notification message to a destination whenever the events occur. It can trigger SNS, SQS, Lambda.
* S3 has 6 different Storage Class
  + **Standart**, 99.99% availability, 11 9’ durability. Replicated across at least three AZs
  + **Intelligent Tiering** uses ML to analyze your object usage and determine the appropriate storage class. Data is moved to the most cost-effective access tier, without any performance impact or added overhead
  + **Standart Infrequent Accessed (IA)**, cheaper if you access files less than once a month. An additional retrieval fee is applied. 50% less than Standart (reduced availability)
  + **One Zone IA**, objects only exist one AZ. Availability (is 99.5%). but cheaper than Standart IA by 20% less (Reduce durability). Data could get destroyed. A retrieval fee is applied.
  + **Glacier** for long-term cold storage. Retrieval of data can take minutes to hours but the off is very cheap storage
  + **Glacier Deep Archive** the lowest cost storage class. Data retrieval time is 12 hours.

**S3 Glacier** The storage classes of Amazon S3 Glacier stands to be cheaper in comparison to the other option available in S3 storage classes and the major reason behind this is that instant data access is given from customer side in this case. As a result, it just takes some minutes or hours for data retrieval in the case of Glacier instead of the S3 objects retrieval in sub-second latency.

For the time of retrieval, users are provided with 3 options by storage class of classic S3 Glacier. These are:

**Bulk:** retrieving bulk requests stands to be the slowest option and the data is accessible after the duration of about 5-12 hours.For the purpose of restoring data in huge amounts that are not immediately in demand, it stands to be the most affordable option available.

**Standard:** Customers get access to the data in the span of 3-5 hours with the option of standard retrievals.

**Expedited:** accessing the data is possible in expedited retrieval within some minutes. This option is the fastest of all but is expensive as well.

Just one option is there in the case of S3 Glacier Deep Archive for the time of retrieval. About 12-48 hours are needed for retrieving data from S3 Glacier Deep Archive. Cost announcements have not been done yet for this.

**Minimum Storage Duration**

The design of S3 Glacier is intended for storage for the long term and therefore users are charged by AWS in case data is deleted by them quickly after it is stored in Glacier.

90 days is the minimum duration of storage in case of classic S3 Glacier. If an object is deleted before the span of 90 days of its placement in S3 Glacier, a prorated fee is charged that is $0.012/GB. In the case object is deleted after 45 days of its placement, the charged fee is half of the prorated fee.

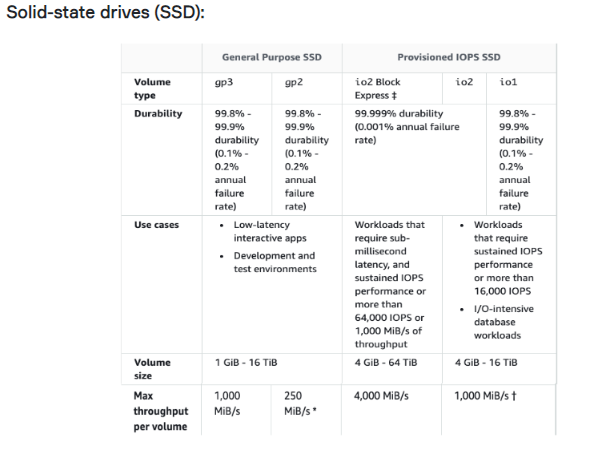
180 days is the minimum duration for storage in the case of S3 Glacier Deep Archive and the structure of pricing is alike S3 Glacier.

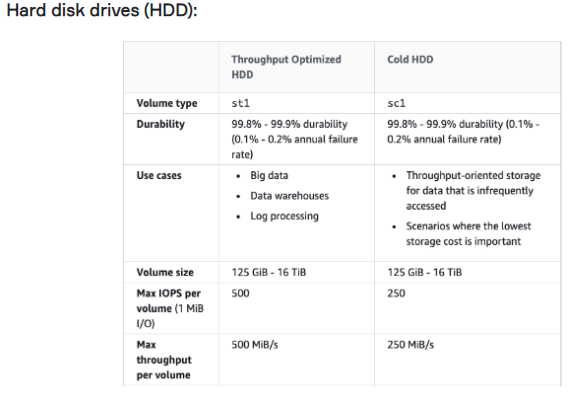


## EBS Cheatsheet

* Elastic Block Storage (EBS) is a virtual hard disk. Snapshot is a point-in-time copy of that disk
* Volumes exist on EBS. Snapshot exists on S3
* Snapshots are incremental, only made since the last snapshot are moved to S3.
* Initial Snapshots of EC2 instance will make take longer to create than subsequent Snapshots
* If taking a Snapshot of a root volüme, the EC2 instance should be stopped before snapshotting.
* You can take Snapshot while the instance is still running
* You can create AMIs from Volumes, or from Snapshots.
* EBS Volumes are durable, a block-level storage device that you can attach to a single EC2 instance.
* EBS volumes do not need to be attached to an instance.
* You can attach multiple EBS volumes to an instance
* You cannot attach an EBS volume to multiple instances (use Elastic File Store instead --> Biggest differences between EFS and EBS). (Exception:16 Nitro-based instances in the same Availability Zone)
* Volume sizes and types can be upgraded without downtime (except for magnetic standard).
* Volumes always exist in the same AZ as the EC2 instance.
* **Instance store** volumes are a temporary storage type located on a disc that is physically attached to a host machine.
* **Snapshots** capture a point-in-time state of an instance. it can be used to migrate a system to a new AZ or region.
* Snapshots are stored on **Amazon S3**.
* You can resize volumes by restoring snapshots with different sizes (configured when taking the snapshot).
* Instance store volumes (ephemeral) cannot be stopped. If the host fails then you lost your data
* EBS backed instances can be stopped and you will not lose any data
* By default root volumes are deleted on termination
* EBS volumes can have termination protection (don’t delete the volume on termination)
* You can encrypt both the boot and data volumes of an EC2 instance. When you create an encrypted EBS volume and attach it to a supported instance type, the following types of data are encrypted:
  + Data at rest inside the volume.
  + All data moving between the volume and the instance.
  + All snapshots created from the volume.
  + All volumes created from those snapshots.
* You cannot share a snapshot if it has been encrypted.
* Unencrypted snapshots can be shared with other AWS accounts or made public
* **To migrate volumes** between AZ’s create a snapshot then create a volume in another AZ from the snapshot (possible to change size and type).

### EBS Volume Types





**Instance Store**

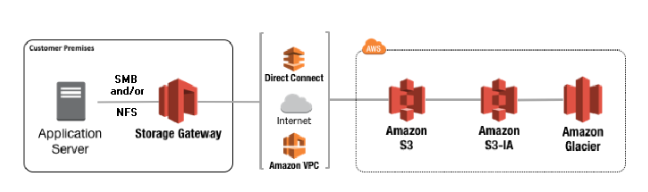
* An instance store provides temporary (non-persistent) block-level storage for your instance.
* This is different to EBS which provides persistent storage but is also a block storage service that can be a root or additional volume.
* Instance store storage is located on disks that are physically attached to the host computer.
* Instance store is ideal for temporary storage of information that changes frequently, such as buffers, caches, scratch data, and other temporary content, or for data that is replicated across a fleet of instances, such as a load-balanced pool of web servers.
* You can specify instance store volumes for an instance only when you launch it.
* You can’t detach an instance store volume from one instance and attach it to a different instance.
* The instance type determines the size of the instance store available and the type of hardware used for the instance store volumes.
* Instance store volumes are included as part of the instance’s usage cost.
* This is a good option when you need storage with very low latency, but you don’t need the data to persist when the instance terminates or you can take advantage of fault-tolerant architectures.
* **EXAM TIP:** Instance stores offer very high performance and low latency. As long as you can afford to lose an instance, i.e. you are replicating your data, these can be a good solution for high performance/low latency requirements. Look out for questions that mention distributed or replicated databases that need high I/O. Also, remember that the cost of instance stores is included in the instance charges so it can also be more cost-effective than EBS Provisioned IOPS.

## EFS Cheatsheet

* EFS is a fully-managed service that makes it easy to set up and scale file storage in the Amazon Cloud.
* Elastic File System (EFS) supports the **Network File System (NFS)** protocol
* Can configure mount-points in one, or many, AZs.
* You pay GB of storage per month
* EFS can scale up to petabyte size storage
* Volumes will shrink and grow to meet current data stored (elastic)
* Can support thousands of concurrent connections over NFS
* Your data is stored across multiple AZs within a region
* Multiple EC2 instances can mount to a single EFS (as long as they are all in the same VPC). it can concurrently connect 1 to 1000s of EC2 instances, from multiple AZs.
* Creates month points in all your VPC subnets so you can mount from anywhere within your VPC
* Provides Read After Write consistency
* EFS can be mounted from on-premises systems ONLY if using **Direct Connect or a VPN connection.**
* Good for big data and analytics, media processing workflows, content management, web serving, home directories etc.
* There are two performance modes:
  + **“General Purpose”** performance mode is appropriate for most file systems.
  + **“Max I/O”** performance mode is optimized for applications where tens, hundreds, or thousands of EC2 instances are accessing the file system.
* EFS offers the ability to encrypt data at rest and in transit. Encryption keys are managed by the AWS Key Management Service (KMS).
* **EFS File Sync** provides a fast and simple way to securely sync existing file systems into Amazon EFS.
* EFS File Sync copies files and directories into Amazon EFS at speeds up to 5x faster than standard Linux copy tools, with simple setup and management in the AWS Console.
* EFS File Sync securely and efficiently copies files over the internet or an AWS Direct Connect connection.
* Amazon EFS is well suited to support a broad spectrum of use cases from highly parallelized, scale-out workloads that require the highest possible throughput to single-threaded, latency-sensitive workloads. Use cases such as lift-and-shift enterprise applications, big data analytics, web serving and content management, application development and testing, media and entertainment workflows, database backups, and container storage.

## Storage Gateway Cheatsheet

### Elastic File Storage (EFS)



* The AWS Storage Gateway service enables hybrid storage between on-premises environments and the AWS Cloud.
* It provides low-latency performance by caching frequently accessed data on-premises while storing data securely and durably in Amazon cloud storage services.
* Provides local storage resources backed by AWS S3 and Glacier.
* Often used in disaster recovery preparedness to sync data to AWS.
* **Useful in cloud migrations.**
* There are three Storage Gateways: File Gateway, Volume Getaway, Tape Gateway
* **File Gateway** lets S3 act as a local file system using NFS or SMB, extends your local hard drive to S3
* **Volume Gateway** is used for backups and has two types: Stored and Cached
* **Stored Volume Gateway**, the entire dataset is stored on-site and is asynchronously backed up to S3 (EBS point-in-time snapshots). Snapshots are incremental and compressed.
* **Stored volumes** are 1 GB to 16 TB in size
* **Cached Volume Gateway** caches the frequently used files on-premise, primary data is stored on S3
* Cached Volumes are 1GB to 32GB in size
* **Tape Gateway** backups virtual tapes to S3 Glacier for long archive storage

## Amazon FX Cheatsheet

### Elastic File Storage (EFS)

* Amazon FSx provides you with the native compatibility of third-party file systems with feature sets for workloads such as Windows-based storage, high-performance computing (HPC), machine learning, and electronic design automation (EDA).
* You don’t have to worry about managing file servers and storage, as Amazon FSx automates time-consuming administration tasks such as hardware provisioning, software configuration, patching, and backups.
* Amazon FSx integrates the file systems with cloud-native AWS services, making them even more useful for a broader set of workloads.
* Amazon FSx provides you with two file systems to choose from:
  + Amazon FSx for Windows File Server for Windows-based applications
  + Amazon FSx for Lustre for compute-intensive workloads.
* **Amazon FSx for Windows File Server** provides a fully managed native Microsoft Windows file system so you can easily move your Windows-based applications that require shared file storage to AWS. Built on Windows Server, Amazon FSx provides the compatibility and features that your Microsoft applications rely on, including full support for the SMB protocol, Windows NTFS, and Microsoft Active Directory (AD) integration. If you read Windows file server, think Amazon FSx for Windows File server.
* **Amazon FSx for Lustre** is a high-performance file system optimized for fast processing of workloads such as machine learning, high-performance computing (HPC), video processing, financial modeling, and electronic design automation (EDA).

## Cloudformation Cheatsheet

### Cloudformation

* When being asked to automate the provisioning of resources think Cloudformation
* When infrastructure as code (IaC) is mentioned think Cloudformation
* Cloudformation can be written in either JSON or YAML
* Elastic Beanstalk is more focused on deploying applications on EC2 (PaaS).
* When Cloudformation encounters an error it will rollback with ROLLBACK\_IN\_PROGRESS
* Cloudformation templates larger than 51,200bytes (0.05 MB) are too large to upload directly and must be imported into Cloudformation via an S3 bucket.
* NestedStacks helps you break up your Cloudformation template into smaller reusable templates that can be composed into larger templates
* At least one resources under resources: must be defined for a Cloudformation template to be valid
* **Metadata:** extra information about your template
* **Description:** a description of what the template is supposed to do
* **Parameters** is how you get user inputs into templates
* **Outputs** are values you can use to import into other stacks
* **Mappings:** map keys to values, just like a lookup table
* **Resources** defines the resources you want to provision, at least one resources is required
* **Conditions** are whether resources are created or properties are assigned

## Elastic Beanstalk Cheatsheet

### Elastic Beanstalk

* AWS Elastic Beanstalk can be used to **quickly deploy and manage applications** in the AWS Cloud.
* Stores your application files and, optionally, server log files in Amazon S3.
* Elastic Beanstalk handles the deployment, from capacity provisioning, load balancing, autoscaling to application health monitoring
* When you want to run a web application but you don’t want to have to think about the underlying infrastructure, you can use Elastic Beanstalk
* It costs nothing to use Elastic Beanstalk (only the resources it provisions eg. RDS, ELB, EC2)
* Recommended for test or development apps. Not recommended for production use
* It can also be used with a Docker. So, you can run dockerized environments on Elastic Beanstalk.
* You can choose from the following pre-configured platforms: Java, .Net, PHP, Node.js, Python, Ruby.

## Database Cheatsheet

### Amazon Aurora

* When you need a fully managed **Postgres or Mysql** database that needs to scale, automatic backups, high availability and fault tolerance think Aurora
* Aurora is an AWS proprietary database. That’s why They ask at least one question in every exam.
* Can run Mysql and Postgres database engines
* Mysql is 5x faster over regular Mysql
* Postgres is 3x faster over regular Postgres
* 1/10 the cost over its competitors with similar performance and availability options.
* Replicates 6 copies for your database across 3 AZs
* Allowed up to 15 aurora replicas
* Aurora database can span multiple regions via **Aurora Global Database**
* Aurora Serverless allows you to stop and start Aurora and scale automatically while keeping cost low
* Aurora Serverless is ideal for new projects with infrequent database usage
* There are two types of replication: **Aurora replica** (up to 15), **MySQL Read Replica** (up to 5).
* You can create read replicas for an Amazon Aurora database in up to five AWS regions. This capability is available for Amazon Aurora with MySQL compatibility.
* **Cross-region read replicas** allow you to improve your disaster recovery posture, scale read operations in regions closer to your application users, and easily migrate from one region to another.
* Cross-region replicas provide fast local reads to your users.
* For globally distributed applications you can use Global Database, where a single Aurora database can span multiple AWS regions to enable fast local reads and quick disaster recovery.
* **Global Database** uses storage-based replication to replicate a database across multiple AWS Regions, with a typical latency of less than 1 second.
* Amazon Aurora Multi-Master is a new feature of the Aurora MySQL-compatible edition that adds the ability to scale out write performance across multiple Availability Zones, allowing applications to direct read/write workloads to multiple instances in a database cluster and operate with higher availability.
* Amazon Aurora Serverless is an on-demand, auto-scaling configuration for Amazon Aurora. The database automatically starts up, shuts down, and scales capacity up or down based on application needs.
* Aurora Auto Scaling dynamically adjusts the number of Aurora Replicas provisioned for an Aurora DB cluster using single-master replication.
* Amazon Aurora’s backup capability enables point-in-time recovery for your instance.

## Database Cheatsheet

### RDS

* Relational Database Service (RDS) is the AWS Solution for relational databases.
* RDS instance are managed by AWS, you cannot SSH into the VM running the database
* RDS is an **Online Transaction Processing (OLTP)** type of database.
* There are 6 relational database options currently available on AWS; Aurora, Mysql, MariaDB, Postgres, Oracle, Microsoft SQL Server.
* You can define the maintenance window or AWS will schedule a 30-minute window.
* Automated backups and patching applied in customer-defined maintenance windows
* **Multi-AZ** is an option you can turn on which makes an exact copy of your database in another AZ that is only on standby
* For Multi-AZ AWS automatically synchronizes changes in the database over to the standby copy
* Multi-AZ has Automatic failover protection if one AZ goes down failover will occur and the standby slave will be promoted to master
* **Read Replicas** allow you to run multiple copies of your database, these copies only allows read (no writes) and is intended to reduce the workload of your primary databases to improve performance
* Read replicas use Asynchronous replication
* You must have automatic backups enabled to use Read Replicas
* If a source DB instance is deleted without deleting the replicas each replica becomes a standalone single-AZ DB instance.
* You can encrypt your Amazon RDS instances and snapshots at rest by enabling the encryption option for your Amazon RDS DB instance.
* You cannot encrypt an existing DB, you need to create a snapshot, copy it, encrypt the copy, then build an encrypted DB from the snapshot.
* A DB subnet group is a collection of subnets (typically private) that you create in a VPC and that you then designate for your DB instances.
* Each DB subnet group should have subnets in at least two Availability Zones in a given region
* You can only scale RDS up (compute and storage).
* Amazon RDS uses EBS volumes (never uses instance store) for DB and logs storage.
* DB Snapshots are user-initiated and enable you to back up your DB instance in a known state as frequently as you wish, and then restore it to that specific state.
* Snapshots remain on S3 until manually deleted.
* There are three storage types available: **General Purpose (SSD), Provisioned IOPS (SSD), and Magnetic.**

**Storage Type of RDS**

**General Purpose (SSD):**

* Use for Database workloads with moderate I/O requirements.
* Cost effective.

**Provisioned IOPS (SSD):**

* Use for I/O intensive workloads.
* Low latency and consistent I/O.

**Magnetic:**

* Not recommended anymore, available for backward compatibility.
* Doesn’t allow you to scale storage when using the SQL Server database engine

## Database Cheatsheet

### DynamoDB

* DynamoDB is a fully managed **NoSQL key/value** and documents database
* Applications that contain large amounts of data but require predictable read and write performance while scaling is a good fit for DynamoDB
* DynamoDB scales with whatever read and write capacity you specify per second
* DynamoDB can be set to have eventually consistent Reads (default) which data is returned immediately but data can be inconsistent and Strong Consistent Reads which will always read from the leader partition since it always has an up-to-date copy.
* Strong consistent reads will until data are inconsistent. Data will never be inconsistent but latency will be higher. Copies of data will be consistent with a guarantee of 1 second.
* DynamoDB stores 3 copies of data on SSD drives across 3 regions.
* **DynamoDB Streams** help you to keep a list of item level changes or provide a list of item level changes that have taken place in the last 24hrs.
* **Amazon DynamoDB Accelerator (DAX)** is a fully managed, highly available, in-memory cache for DynamoDB that delivers up to a 10x performance improvement.
* DAX does all the heavy lifting required to add in-memory acceleration to your DynamoDB tables, without requiring developers to manage cache invalidation, data population, or cluster management.
* ElastiCache can be used in front of DynamoDB for the performance of reads on infrequently changed data
* RedShift complements DynamoDB with advanced business intelligence
* Amazon DynamoDB global tables provide a fully managed solution for deploying a multi-region, multi-master database. When you create a global table, you specify the AWS regions where you want the table to be available. DynamoDB global tables are ideal for massively scaled applications, with globally dispersed users.
* **A global table** is a collection of one or more replica tables, all owned by a single AWS account. With a global table, each replica table stores the same set of data items. DynamoDB does not support partial replication of only some of the items.
* **A replica table** (or replica, for short) is a single DynamoDB table that functions as a part of a global table. Each replica stores the same set of data items. Any given global table can only have one replica table per region.
* **DynamoDB auto scaling** uses the AWS Application Auto Scaling service to dynamically adjust provisioned throughput capacity on your behalf, in response to actual traffic patterns. This enables a table or a global secondary index to increase its provisioned read and write capacity to handle sudden increases in traffic, without throttling.
* Multi-AZ NoSQL data store with Cross-Region Replication option.
* Priced on throughput, rather than compute
* DynamoDB supports two kinds of secondary indexes:
  + Global secondary index – An index with a partition key and sort key that can be different from those on the table.
  + Local secondary index – An index that has the same partition key as the table, but a different sort key.
* You can search using one of the following methods:
  + Query operation – find items in a table or a secondary index using only the primary keys attributes.
  + Scan operation – reads every item in a table or a secondary index and by default will return all items.

## Database Cheatsheet

### Redshift

* Amazon Redshift is a fast, fully managed data warehouse that makes it simple and cost-effective to analyze all your data using standard SQL and existing Business Intelligence (BI) tools.
* RedShift is a **SQL-based data WAREHOUSE** used for analytics applications.
* RedShift is an **Online Analytics Processing (OLAP)** type of DB.
* RedShift is ideal for processing large amounts of data for business intelligence.
* Data can be loaded from S3, EMR, DynamoDB, or multiple data sources on remote hosts
* Redshift can handle petabytes worth of data. Redshift is for Data Warehousing
* RedShift uses replication and continuous backups to enhance availability and improve durability and can automatically recover from component and node failures.
* Redshift can only run in a 1 AZ (Single AZ)
* Redshift can run via a single node or multiple-node (cluster)
* A single node is 160 GB in size
* RedShift is 10x faster than a traditional SQL DB.
* A multi-node is comprised of a leader node and multiple compute nodes
* You are bill per hour for each node (excluding leader node in multi-node)
* You are not billed fort he leader node
* You can have up to 128 compute nodes
* Redshift has two kinds of Node Type; **Dense Compute and Dense Storage**
* Redshift attempts to backup 3 copies of your data, the original, on compute node and on S3
* Similar data is stored on disk sequentially for faster reads
* Redshift database can be encrypted via KMS or CloudHSM
* Backup Retention is default to 1 day and can be increased to a maximum of 35 days
* Redshift can **asynchronously back up** your snapshot to another region delivered to S3
* Redshift uses Massively Parallel Processing (MPP) to distribute queries and data across all loads
* In case of an empty table, when importing Redshift will sample data to create a schema
* **RedShift uses columnar data storage;**
  + Data is stored sequentially in columns instead of rows.
  + Columnar based DB is ideal for data warehousing and analytics.
  + Requires fewer I/Os which greatly enhances performance.

## Database Cheatsheet

### Elasticache

* Elasticache is a managed in-memory caching service.
* The in-memory caching provided by ElastiCache can be used to significantly improve latency and throughput for many read-heavy application workloads or compute-intensive workloads
* Elasticache can launch either Memcache or Redis.
* **Memcache** is a simple key/value store preferred for caching HTML fragments and is arguably faster than Redis. Memcache is not persistent and cannot be used as a data store but employed cache the contents of a DB.
* **Redis** has richer data types and operations. Great for the leaderboard, geospatial data or keeping track of unread notifications. Data is persistent with Redis. Scales by adding shards, not nodes.
* Most frequently identical queries are stored in the cache
* Resources only within the same VPC may connect to Elasticache to ensure low latencies.
* Use cases:
  + **Web session store:** In case with load-balanced web servers, store web session information in Redis so if a server is lost, the session info is not lost and another web server can pick it up
  + **Database caching:** Use Memcached in front of AWS RDS to cache popular queries to offload work from RDS and return results faster to the user
  + **Leaderboards:** Use Redis to provide a live leaderboard for millions of users of your mobile app
  + **Streaming data dashboards:** Provide a landing spot for streaming sensor data on the factory floor, providing live real-time dashboards displays

## Database Cheatsheet

### AWS Neptune

* The core of Amazon Neptune is a purpose-built, high-performance **graph database engine**.
* Amazon Neptune supports popular **graph query languages** like Property Graph and W3C's RDF, and their respective query languages Apache TinkerPop Gremlin and SPARQL.
* Allows you to easily build queries that efficiently navigate highly connected datasets
* It has read replica, point-in-time recovery, continuous backup to Amazon S3, and replication across Availability Zones
* Optimized for storing billions of **relationships and querying the graph** with milliseconds latency.
* Use case:
  + **Social Networking:** AWS Neptune can easily process user's interactions like comments, follows, and likes in a social network application through highly interactive queries
  + **Recommendation Engines:** Amazon Neptune can build apps for suggesting personalized and relevant products based on the relationship between information such as customer's interest and purchase history
  + **Knowlege Graphs:** We can create a knowledge graph for search engines that will enable users to quickly discover new information
  + **Identity Graphs:** it can be used as a graph database to easily link and update user profile data for ad-targeting, personalization, and analytics

## Database Cheatsheet

### AWS Database Migration Service

* AWS Database Migration Service helps you migrate databases to AWS quickly and securely.
* Use along with the Schema Conversion Tool (SCT) to migrate databases to AWS RDS or EC2-based databases.
* The source database remains fully operational during the migration, minimizing downtime to applications that rely on the database.
* The AWS Database Migration Service can migrate your data to and from the most widely used commercial and open-source databases.
* Schema Conversion Tool can copy database schemas for homogenous migrations (same database) and convert schemas for heterogeneous migrations (different databases).
* DMS is used for smaller, simpler conversions and also supports MongoDB and DynamoDB.
* SCT is used for larger, more complex datasets like data warehouses.
* DMS has replication functions for on-premise to AWS or to Snowball or S3.
* The RDS service includes the following:
  + Security and patching of the DB instances.
  + Automated backup for the DB instances.
  + Software updates for the DB engine.
  + Easy scaling for storage and compute.
  + Multi-AZ option with synchronous replication.
  + Automatic failover for Multi-AZ option.
  + Read replicas option for read-heavy workloads.

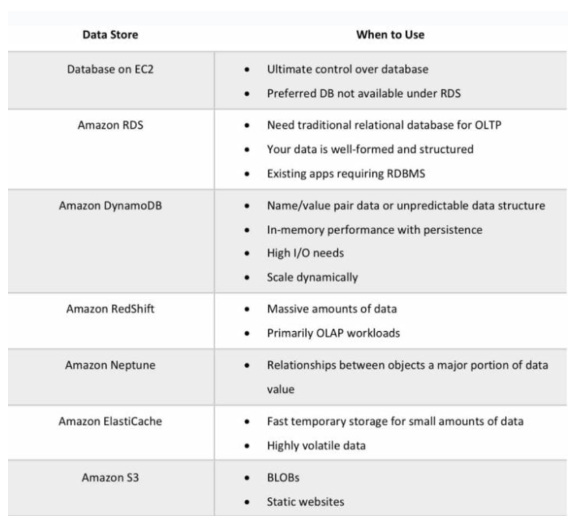
## Database Cheatsheet

### High Availability Approaches for Databases

* If possible, choose DynamoDB over RDS because of inherent fault tolerance.
* If DynamoDB can’t be used, choose Aurora because of redundancy and automatic recovery features.
* If Aurora can’t be used, choose Multi-AZ RDS.
* Frequent RDS snapshots can protect against data corruption or failure and they won’t impact the performance of Multi-AZ deployment.
* Regional replication is also an option, but will not be strongly consistent.
* If the database runs on EC2, you have to design the HA yourself.

## Database Cheatsheet

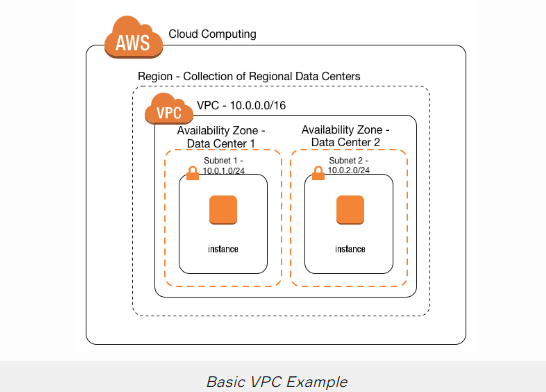
### Comparison of Databases and Storage Solutions



## Network Cheatsheet

### VPC General

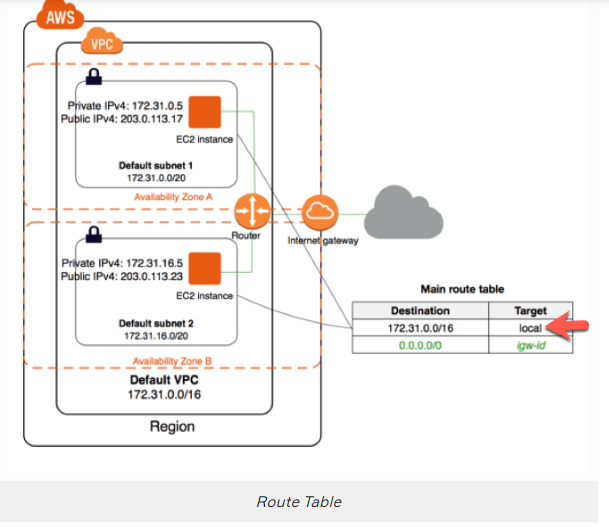
* Amazon **Virtual Private Cloud (VPC)** lets you provision a logically isolated section of the Amazon Web Services (AWS) cloud where you can launch AWS resources in a virtual network that you define.
* Provides complete control over the virtual networking environment including a selection of IP ranges, creation of subnets, and configuration of route tables and gateways.
* A VPC is logically isolated from other VPCs on AWS.
* A default VPC is automatically created for each AWS account the first time Amazon EC2 resources are provisioned.
* VPCs are Region Specific and they do not span regions
* You can create up to 5 VPCs per region
* Every region comes with a default VPC
* You can have 200 subnets per VPC
* You can use IPv4 CIDR block and in addition to IPv6 CIDR blocks (the address of the VPC)
* Cost free: VPC’s, Route Tables, NACL’s, Security Groups, Internet Gateway’s, VPC Peering’s are free of charge
* NAT Gateways, NAT Instance, VPN Gateway, Customer Gateways have cost.
* VPC can assign DNS hostnames to your AWS resources.



## Network Cheatsheet

### Route Tables

* The VPC route tables (router) perform routing between AZs within a region.
* The VPC router connects different AZs together and connects the VPC to the Internet Gateway.
* Each subnet has a Route Table. Route tables use to forward traffic within the VPC.
* Route tables also have entries to external destinations.
* Each subnet can only be associated with one route table.
* Can assign one route table to multiple subnets.
* Cannot delete the main route table.



## Network Cheatsheet

### Subnets

* Types of subnet:
  + If a subnet’s traffic is routed to an internet gateway, the subnet is known as a **public subnet**.
  + If a subnet doesn’t have a route to the internet gateway, the subnet is known as a **private subnet**.
  + If a subnet doesn’t have a route to the internet gateway, but has its traffic routed to a virtual private gateway for a VPN connection, the subnet is known as a **VPN-only subnet**.
* You cannot create additional CIDR blocks that overlap with existing CIDR blocks.
* 5 IP addresses are reserved by AWS in every subnet. These are the first 4 and last 1 IP address on their CIDR block.

Internet Gateway

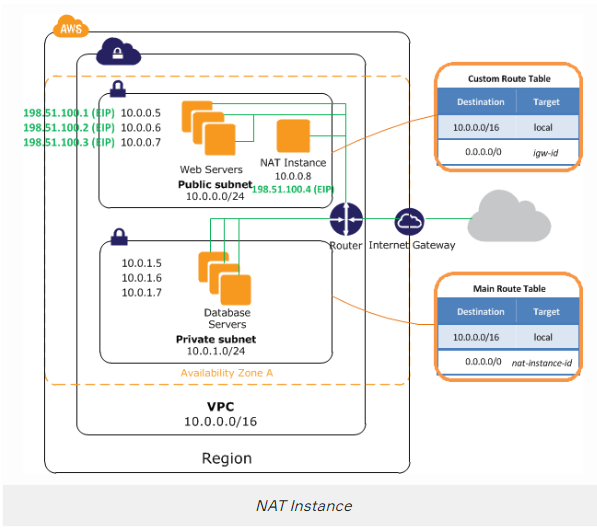
* An Internet Gateway is a horizontally scaled, redundant, and highly available VPC component that allows communication between instances in your VPC and the internet.
* Internet Gateways (IGW) must be created and then attached to a VPC, be added to a route table, and then associated with the relevant subnet(s).
* If your subnet is associated with a route to the Internet, then it is a public subnet.
* Can only attach 1 IGW to a VPC at a time.
* We can mention 4 different kinds of Gateways
  + **Internet gateway (IGW)** – AWS VPC side of the connection to the public Internet.
  + **Virtual private gateway (VPG)** – VPC endpoint on the AWS side.
  + **Customer Gateway (CGW)** – representation of the customer end of the connection.
  + **Egress-only Internet Gateway** Provides outbound Internet access for IPv6 addressed instances.

## Network Cheatsheet

### NAT Instance & NAT Gateway

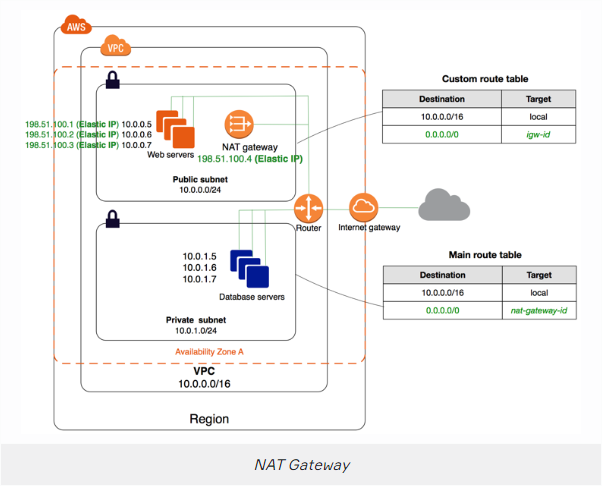
**NAT Instance:**

* NAT Instance is managed by the owner of the account.
* When creating a NAT instance you must **disable source and destination check** on the instance
* NAT instances must exist in a public subnet with a route to an Internet Gateway.
* You must have a route of the private subnet to the NAT instance
* NAT instances need to be assigned to security groups.
* Can use it as a bastion (jump) host.
* The size of a NAT instance determines how much traffic can be handled
* High availability(HA) can be achieved using Autoscaling Groups, multiple subnets in different AZs, and automate failover between them using a script



**NAT Gateway:**

* NAT gateways are fully managed by AWS.
* NAT Gateways are redundant inside an Availability Zone (can survive failure of EC2 instance)
* You can only have 1 NAT Gateway inside 1 AZ (cannot span AZs)
* Starts at 5 GBPS and scales all the way up to 45 Gbps
* Uses an Elastic IP address for the public IP.
* NAT Gateways are the preferred setup for enterprise systems.
* There is no requirement to patch NAT Gateways, and there is no need to disable Source/Destination checks fort he NAT Gateway (unlike NAT instances)
* NAT Gateways are automatically assigned a public IP address
* Not associated with any security groups.
* Route Tables for the NAT Gateway MUST be uploaded
* Resource in multiple AZs sharing a Gateway will löse internet access if the Gateway goes down unless you create a Gateway in each AZ and configure route tables accordingly.



## Network Cheatsheet

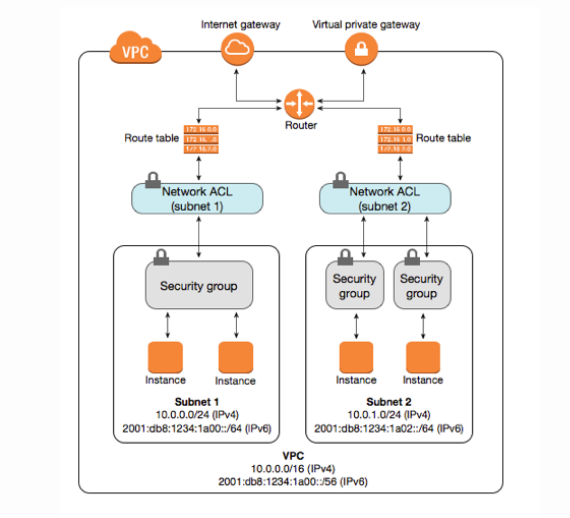
### Security Group & Network Access List (NACL)

**Security Groups:**

* Security Groups acts as a firewall at the instance level
* Useless allowed specifically, all inbound traffic is blocked by default
* All outbound traffic from the instance is allowed by default
* You can specific for the source to be either an IP range, single IP address, or another security group
* Unlike NACL, you can only assign permit rules in a security group, cannot assign deny rules.
* Security Groups are **STATEFULL** (if traffic is allowed inbound it is also allowed outbound)
* Any changes to a Security Group apply immediately.
* You can use security group names as the source or destination in other security groups.
* We can assign multiple security groups to EC2 instances.
* Security groups can include multiple EC2 instances
* We can not block any specific IP address. For this purpose, we need Network Access List (NACL) tables of subnets.
* You can have 16 Security Groups associated with an ENI (default is 5)

**Network Access List (NACL):**

* Network ACL’s function at the subnet level.
* With NACLs you can have permit and deny rules.
* Network ACLs contain a numbered list of rules that are evaluated in order from the lowest number until the explicit deny.
* Network ACLs have separate inbound and outbound rules and each rule can allow or deny traffic.
* Network ACLs are **STATELESS** so responses are subject to the rules for the direction of traffic.
* NACLs only apply to traffic that is ingress or egress to the subnet not to traffic within the subnet.
* A VPC automatically comes with a default network ACL which allows all inbound/outbound traffic.
* A custom NACL denies all traffic both inbound and outbound by default.
* All subnets must be associated with a network ACL.
* You can create custom NACL’s. By default, each custom network ACL denies all inbound and outbound traffic until you add rules.
* You can associate a network ACL with multiple subnets; however a subnet can only be associated with one network ACL at a time.
* NACLs are the preferred option for blocking specific IPs or ranges.
* Security groups cannot be used to block specific ranges of IPs.
* NACL is the first line of defense, the security group is the second line.
* Changes to NACLs take effect immediately like Security Groups.



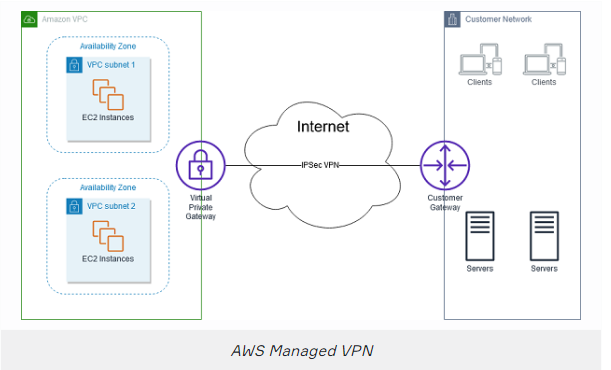
## Network Cheatsheet

### VPC Connectivity

* There are several methods of connecting to a VPC. These include:
  + AWS Managed VPN.
  + AWS Direct Connect.
  + AWS Direct Connect plus a VPN.
  + AWS VPN CloudHub.
  + Software VPN.
  + Transit VPC.
  + VPC Peering.
  + AWS PrivateLink.
  + VPC Endpoints.

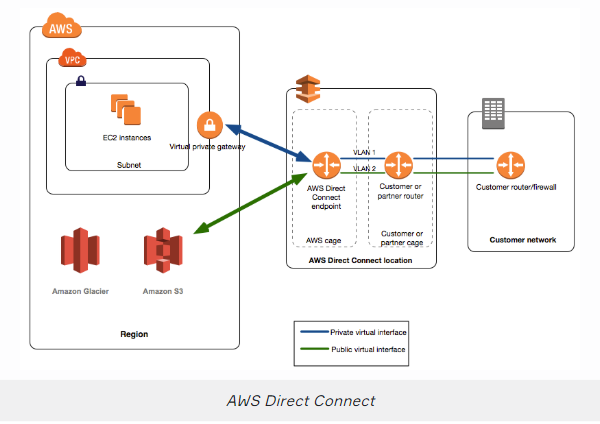
**AWS Managed VPN:**

* AWS Managed IPSec VPN connector over your existing internet
* Use case: If you need to establish a secure tunneled connection with a quick and usually simple way to a VPC or redundant link for Direct Connect or other VPN
* A Virtual Private Gateway (VGW) is required on the AWS side
* A customer Gateway is required on the customer side
* An Internet routable IP address is required on the customer gateway.
* Two tunnels per connection must be configured for redundancy.



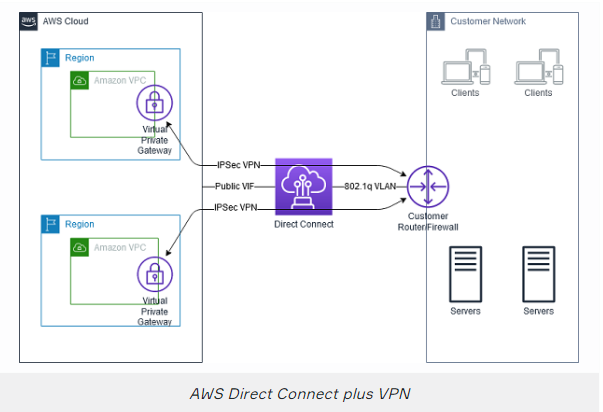
**AWS Direct Connect:**

* AWS Direct Connect makes it easy to establish a dedicated connection from an on-premises network to Amazon VPC.
* Using AWS Direct Connect, you can establish private connectivity between AWS and your data center, office, or collocated environment.
* This private connection can reduce network costs, increase bandwidth throughput, and provide a more consistent network experience than internet-based connections.
* AWS Direct Connect lets you establish 1 Gbps or 10 Gbps dedicated network connections (or multiple connections) between AWS networks and one of the AWS Direct Connect locations.
* AWS Direct Connect does not encrypt your traffic that is in transit.
* Use case: When you require a large network link into AWS or lots of resources and services being provided on AWS to your corporate users
* Pros of Direct connect: More predictable network performance; potential bandwidth cost reduction; up to 10 Gbps provisioned connections; supports BGP peering and routing
* Cons of Direct Connection: May require additional telecom and hosting provider relationship and/or network circuits; high cost



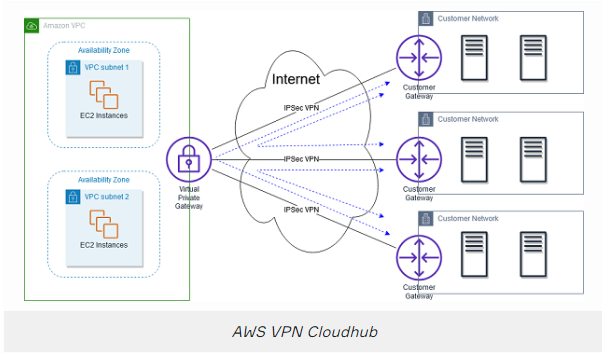
**AWS Direct Connect Plus VPN:**

* With AWS Direct Connect plus VPN, you can combine one or more AWS Direct Connect dedicated network connections with the Amazon VPC VPN.
* **Use case:** Need the added security of encrypted tunnels over Direct Connect.
* This solution combines the AWS managed benefits of the VPN solution with low latency, increased bandwidth, more consistent benefits of the AWS Direct Connect solution, and an end-to-end, secure IPsec connection.
* This combination provides an IPsec-encrypted private connection that also reduces network costs, increases bandwidth throughput, and provides a more consistent network experience than internet-based VPN connections.



**AWS VPN CloudHub:**

* The AWS VPN CloudHub operates on a simple hub-and-spoke model that you can use with or without a VPC.
* Use case: Link remote offices for back up or primary WAN access to AWS resources and each other -**Use this design** if you have multiple branch offices and existing Internet connections and would like to implement a convenient, potentially low cost hub-and-spoke model for primary or backup connectivity between these remote offices.
* VPN CloudHub is used for hardware-based VPNs and allows you to configure your branch offices to go into a VPC and then connect that to the corporate DC (hub and spoke topology with AWS as the hub).
* Uses a **Border Gateway Protocol (BGP)**.
* Branches can talk to each other (and provides redundancy).
* Can have Direct Connect connections.
* Hourly rates plus data egress charges.

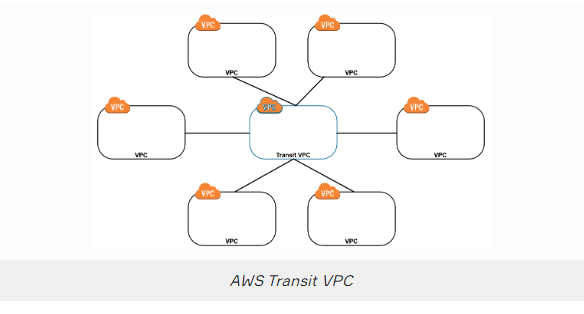


**Software VPN:**

* Amazon VPC offers you the flexibility to fully manage both sides of your Amazon VPC connectivity by creating a VPN connection between your remote network and a software VPN appliance running in your Amazon VPC network.
* **Use case:** You must manage both ends of the VPN connection for compliance reasons or you want to use a VPN option not supported by AWS
* This option is recommended if you must manage both ends of the VPN connection either for compliance purposes or for leveraging gateway devices that are not currently supported by Amazon VPC’s VPN solution.

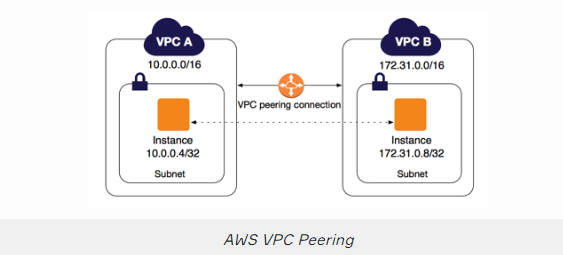
**Transit VPC:**

* Building on the Software VPN design mentioned above, you can create a global transit network on AWS. If you need to connect different VPC’s and handle them in one point think transit VPC.
* **Use case:** Locations and VPC-deployed assets across multiple regions that need to communicate with one another
* A transit VPC is a common strategy for connecting multiple, geographically disperse VPCs and remote networks in order to create a global network transit center.
* A transit VPC simplifies network management and minimizes the number of connections required to connect multiple VPCs and remote networks.



**VPC Peering:**

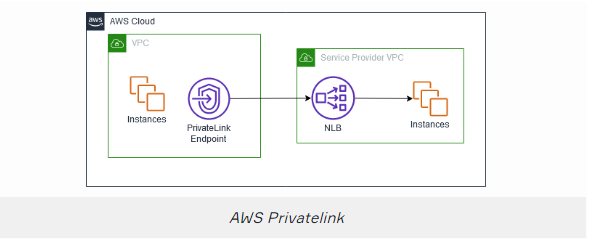
* A VPC peering connection is a networking connection between two VPCs that enables you to route traffic between them using private IPv4 addresses or IPv6 addresses.
* **Use Case:** Multiple VPS’s need to communicate or access each other’s resources
* Instances in either VPC can communicate with each other as if they are within the same network.
* You can create a VPC peering connection between your own VPCs, or with a VPC in another AWS account.
* The VPCs can be in different regions (also known as an inter-region VPC peering connection).
* Data sent between VPCs in different regions is encrypted (traffic charges apply)
* Can only have one peering connection between any two VPCs at a time.
* Can peer with other accounts (within or between regions).
* You can create multiple VPC peering connections for each VPC that you own, but transitive peering relationships are not supported.
* Must update route tables to configure routing.
* Need to accept the pending access request in the peered VPC.



**AWS PrivateLink:**

* AWS PrivateLink simplifies the security of data shared with cloud-based applications by eliminating the exposure of data to the public Internet.
* Provides private connectivity between VPCs, AWS services, and on-premises applications, securely on the Amazon network
* Makes it easy to connect services across different accounts and VPCs to significantly simplify the network architecture.
* **Use case:** Keep private subnets truly by using the AWS backbone to reach other AWS or Marketplace services rather than the public internet.

**EXAM TIP:** Know the difference between AWS PrivateLink and ClassicLink. ClassicLink allows you to link EC2-Classic instances to a VPC in your account, within the same region. EC2-Classic is an old platform from before VPCs were introduced and is not available to accounts created after December 2013. However, ClassicLink may come up in exam questions as a possible (incorrect) answer so you need to know what it is.



**VPC Endpoints:**

* An Interface endpoint uses AWS PrivateLink and is an elastic network interface (ENI) with a private IP address that serves as an entry point for traffic destined to a supported service.
* Using PrivateLink you can connect your VPC to supported AWS services, services hosted by other AWS accounts (VPC endpoint services), and supported AWS Marketplace partner services.
* An interface VPC endpoint (interface endpoint) enables you to connect to services powered by AWS PrivateLink.
* Gateway endpoints are only available for S3 and DynamoDB
* **Use case:** You can reach S3 and DynamoDB in a private subnet without any internet connection

## Network Cheatsheet

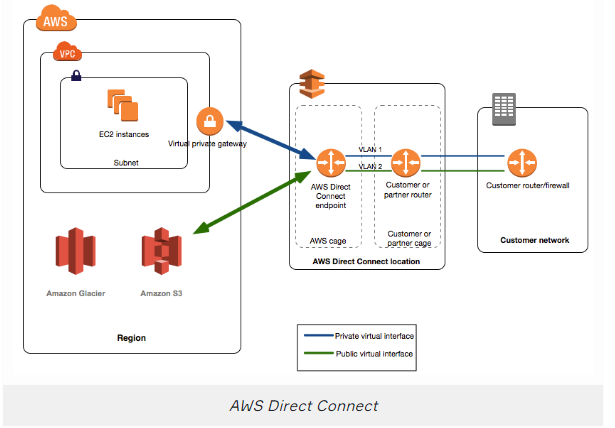
### VPC Flow Logs

* Flow Logs capture information about the IP traffic going to and from network interfaces in a VPC.
* Flow log data is stored using Amazon CloudWatch Logs.
* VPC Flow Logs can be sent directly to an S3 bucket which allows you to retrieve and analyze these logs yourself.
* You can’t change the configuration of a flow log after it’s been created.

## Direct Connect Cheatsheet

### Direct Connect General

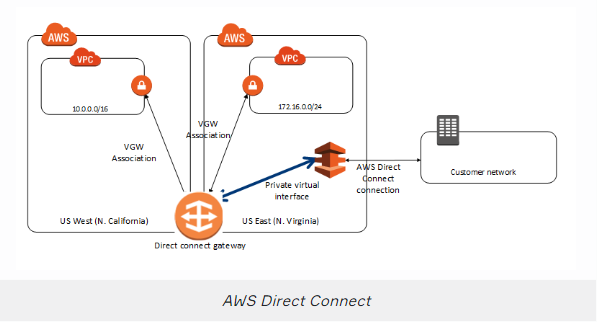
* AWS Direct Connect is a network service that provides an alternative to using the Internet to connect a customer’s on-premise sites to AWS.
* Data is transmitted through a private network connection between AWS and a customer’s datacenter or corporate network.
* **Benefits:**
  + Reduce cost when using large volumes of traffic.
  + Increase reliability (predictable performance).
  + Increase bandwidth (predictable bandwidth).
  + Decrease latency.
* Each AWS Direct Connect connection can be configured with one or more virtual interfaces (VIFs).
* Public VIFs allow access to public services such as S3, EC2, and DynamoDB.
* Private VIFs allow access to your VPC.
* Must use public IP addresses on public VIFs.
* Must use private IP addresses on private VIFs.
* Cannot do layer 2 over Direct Connect (L3 only).
* From Direct Connect you can connect to all AZs within the region.
* You can establish IPSec connections over public VIFs to remote regions.
* You can bind multiple ports for higher bandwidth.
* Virtual interfaces are configured to connect to either AWS public services (e.g. EC2/S3) or private services (e.g. VPC based resources).
* Available in 1Gbps and 10Gbps.
* Speeds of 50Mbps, 100Mbps, 200Mbps, 300Mbps, 400Mbps, and 500Mbps can be purchased through AWS Direct Connect Partners.
* Route tables need to be updated to point to a Direct Connect connection.
* Can aggregate up to 4 Direct Connect ports into a single connection using Link Aggregation Groups (LAG)
* Technical requirements for connecting virtual interfaces:
  + A public or private Autonomous System Number (ASN.) If you are using a public ASN you must own it. If you are using a private ASN, it must be in the 64512 to 65535 range.
  + A new unused VLAN tag that you select.
  + Private Connection (VPC) – The VPC Virtual Private Gateway (VGW) ID.
  + Public Connection – Public IPs (/30) allocated by you for the BGP session.



## Direct Connect Cheatsheet

### Direct Connect Gateway

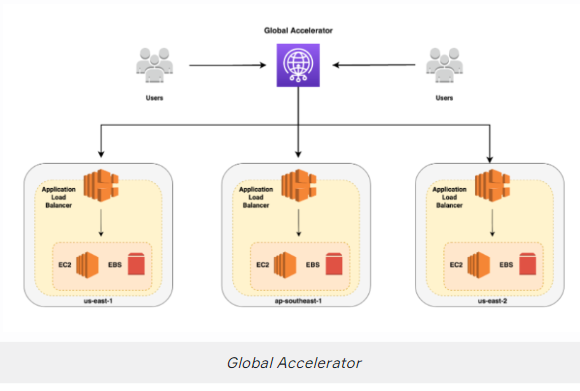
* Direct Connect Gateway enables you to interface with VPCs in any AWS Region (except AWS China Region).
* You associate an AWS Direct Connect gateway with either of the following gateways:
  + A transit gateway when you have multiple VPCs in the same region.
  + A virtual private gateway.
* Can share private virtual interface to interface with more than one Virtual Private Clouds (VPCs) reducing the number of BGP sessions.
* A Direct Connect gateway is a globally available resource.
* You can create the Direct Connect gateway in any public Region and access it from all other public Regions.



## Global Accelerator Cheatsheet

### AWS Global Accelerator

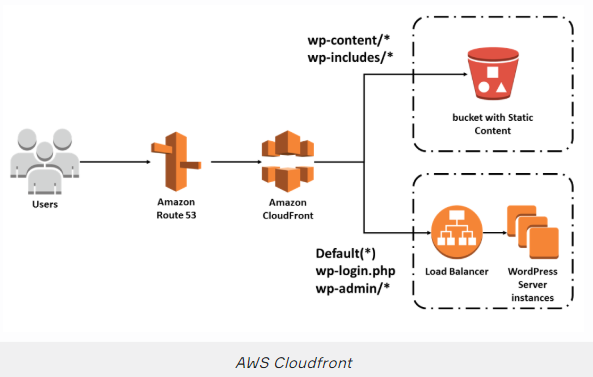
* AWS Global Accelerator is a service that improves the availability and performance of applications with local or global users.
* It provides static IP addresses that act as a fixed entry point to application endpoints in a single or multiple AWS Regions, such as Application Load Balancers, Network Load Balancers or EC2 instances.
* Uses the AWS global network to optimize the path from users to applications, improving the performance of TCP and UDP traffic.
* AWS Global Accelerator continually monitors the health of application endpoints and will detect an unhealthy endpoint and redirect traffic to healthy endpoints in less than 1 minute.
* For stateful applications where you need to consistently route users to the same endpoint, you can choose to direct all requests from a user to the same endpoint, regardless of the port and protocol.
* By default, AWS Global Accelerator is protected by AWS Shield Standard, which minimizes application downtime and latency from denial of service attacks by using always-on network flow monitoring and automated in-line mitigation.



## Cloudfront Cheatsheet

### Cloudfront General

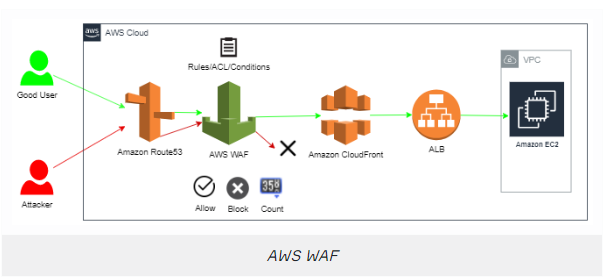
* Cloudfront is a **CDN (Content Distribution Network)**. It makes the website load fast by serving cached content that is nearby
* CloudFront is a good choice for the distribution of frequently accessed static content that benefits from edge delivery—like popular website images, videos, media files, or software downloads.
* An **Edge Location** is the location where content is cached (separate to AWS regions/AZs).
* An **origin** is the origin of the files that the CDN will distribute.
* Origins can be either an S3 bucket, an EC2 instance, an Elastic Load Balancer, or Route 53 – can also be external (non-AWS).
* When using Amazon S3 as an origin you place all of your objects within the bucket.
* You can make objects publicly available or use CloudFront **signed URLs**.
* Edge Locations aren’t just read-only, you can write to them eg. PUT objects
* **TTL (Time to Live)** defines how long until the cache expires (refresh cache)
* When you invalidate your cache, you are forcing it to immediately expire (refreshes cached data)
* Refreshing the cache cost Money because of transfer costs to update Edge Locations
* Origin is the address of where the original copies of your files reside eg. S3, EC2, ELB, Route 53
* Distribution defines a collection of Edge Locations and behavior on how it should handle your cached content
* **Origin Identity Access (OAI)** is used to access private S3 buckets
* Access to cached content can be protected via **Signed URL or Signed Cookies**
* Blacklists and whitelists can be used for geography – you can only use one at a time.
* There are two options available for geo-restriction (geo-blocking):
  + Use the CloudFront geo-restriction feature (use for restricting access to all files in a distribution and at the country level).
  + Use a 3rd party geo-location service (use for restricting access to a subset of the files in a distribution and for finer granularity at the country level).
* Lamda@Edge allows you to pass each request through a Lambda to change the behavior of the response



## Cloudfront Cheatsheet

### AWS WAF

* AWS WAF stands for Web Application Firewall that lets you monitor HTTP and HTTPS requests that are forwarded to CloudFront and lets you control access to your content.
* With AWS WAF you can shield access to content based on conditions in a web access control list (web ACL) such as:
  + Origin IP address.
  + Values in query strings.
* CloudFront responds to requests with the requested content or an HTTP 403 status code (forbidden).
* Need to associate the relevant distribution with the web ACL.



## Cloudfront Cheatsheet

### Lambda@Edge

* Lambda@Edge allows you to run code across AWS locations globally without provisioning or managing servers, responding to end-users at the lowest network latency.
* Lambda@Edge lets you run Node.js and Python Lambda functions to customize content that CloudFront delivers, executing the functions in AWS locations closer to the viewer.
* The functions run in response to CloudFront events, without provisioning or managing servers. You can use Lambda functions to change CloudFront requests and responses at the following points:
* You just upload your Node.js code to AWS Lambda and configure your function to be triggered in response to an Amazon CloudFront request.
* The code is then ready to execute across AWS locations globally when a request for content is received, and scales with the volume of CloudFront requests globally.

## Route 53 Cheatsheet

### Some Definitions

* **Top-Level Domain** example.com last part of the domain
* **Second Level Domain** example.CO.UK second last part of the domain
* **Domain Registrar** 3rd party company who you register domains through
* **Name Server (NS):** The server(s) which contain the DNS records for a domain
* **Start of Authority (SOA)** contains information about the DNS zone and associated DNS records. Amazon Route 53 automatically creates the Name Server (NS) and Start of Authority (SOA) records for the hosted zones.
* **A record** is a DNS record that directly converts a domain name into an IP address
* **CNAME** is record DNS record which lets you convert a domain name into another domain name
* **Time to Live (TTL)**, the time that a DNS record will be cached for
* **A hosted zone** is a collection of records for a specified domain.
* There are two types of zones:
  + **Public host zone** – determines how traffic is routed on the Internet.
  + **Private hosted zone for VPC** – determines how traffic is routed within VPC (resources are not accessible outside the VPC).

### Route 53

* Route 53 is a DNS provider, register and manage domains, create record sets. Think Godaddy or NameCheap
* Route 53 can be used to route Internet traffic for domains registered with another domain registrar (any domain).
* Route 53 is located alongside all edge locations.
* **The Alias record** is a Route 53 specific record type.
* Alias records are used to map resource record sets in your hosted zone to Amazon Elastic Load Balancing load balancers, Amazon CloudFront distributions, AWS Elastic Beanstalk environments, or Amazon S3 buckets that are configured as websites.
* Alias records work like a CNAME record in that you can map one DNS name (e.g. example.com) to another ‘target’ DNS name (e.g. elb1234.elb.amazonaws.com)
* **Simple Routing** – Default routing policy, multiple address result in a random endpoint selection
  + An A record is associated with one or more IP addresses.
  + Does not support health checks.
* **Weighted Routing** – Split up traffic based on different “weights2 assigned (percentages)
  + Similar to simple but you can specify a weight per IP address.
  + You create records that have the same name and type and assign each record a relative weight.
  + Numerical value that favors one IP over another.
  + To stop sending traffic to a resource you can change the weight of the record to 0.
* **Latency-based Routing** – Directs traffic based on region, for lowest possible latency for users
  + AWS maintains a database of latency from different parts of the world.
  + Focussed on improving performance by routing to the region with the lowest latency.
  + You create latency records for your resources in multiple EC2 locations.
* **Failover Routing** – Primary site in one location, secondary data recovery site in another. (change on health check)
  + Failover to a secondary IP address.
  + Associated with a health check.
  + Used for active-passive.
  + Routes only when the resource is healthy.
  + Can be used with ELB.
  + When used with Alias records set Evaluate Target Health to “Yes” and do not use health checks.
* **Geolocation Routing** – Route traffic based on the geographic location of request origin.
  + Caters to different users in different countries and different languages.
  + Contains users within particular geography and offers them a customized version of the workload based on their specific needs.
  + Geolocation can be used for localizing content and presenting some or all of your website in the language of your users.
  + Can also protect distribution rights.
  + Can be used for spreading load evenly between regions.
  + If you have multiple records for overlapping regions, Route 53 will route to the smallest geographic region.
  + You can create a default record for IP addresses that do not map to a geographic location.
* **Geo-proximity Routing** – Use for routing traffic based on the location of resources and, optionally, shift traffic from resources in one location to resources in another.
* **Traffic flow** – visual editor, for chaining routing policies, can verision policy records for easy rollback. It is Global Traffic Management (GTM) services. Traffic flow policies allow you to create routing configurations for resources using routing types such as failover and geolocation.
* **AWS Alias Record** – AWS's smart DNS record, detects changed IPs for AWS resources and adjust automatically
* **Route53 Resolver** – Lets you regionally route DNS queries between your VPCs an your network Hybrid Environments (on-premise). Connectivity needs to be established between your on-premises DNS infrastructure and AWS through a Direct Connect (DX) or a Virtual Private Network (VPN).
* Health checks can be created to monitor and automatically over endpoints. You can have health checks monitör other health checks.

## AWS Lambda Cheatsheet

### AWS Lambda

* Lambda is a serverless function. You upload your code and it runs without you managing or provisioning any servers
* Lambda is serverless which means your application still runs on servers, but all the server management is done by AWS.
* Lambda is a good fit for short-running tasks where you don’t need to customize the os environment. If you need long-running tasks (>15 mins) and a custom OS environment than consider using Fargate
* There are 7 runtime language environments officially supported by Lambda: Ruby, Python, Java, NodeJs, C#, Powershell, and Go
* You pay invocation ( The duration and the amount of memory used) rounded up to the nearest 100 milliseconds and you based on the amount of request. The First 1M requests per month are free.
* You can adjust the duration timeout for up to **15 mins** and memory up to 3008 MB.
* You can trigger Lambda from the SDK, API calls or multiple AWS services eg. S3, API Gateway, DynamoDB, etc., or Lambda triggers these AWS services.
* Lambda by default run in No VPC, To interact with some services you need to have your Lambda in the same VPC
* Lambda can scale to 1000 concurrent functions in seconds. (1000 is the default, you can increase with AWS Service limit increase)
* Lambda has Cold Stars. If the function has not been recently executed, there will be a delay
* Functions can access:
  + AWS services or non-AWS services.
  + AWS services running in VPCs (e.g. RedShift, Elasticache, RDS instances).
  + Non-AWS services running on EC2 instances in an AWS VPC.

**Supported AWS event sources include:**

* Amazon S3.
* Amazon DynamoDB.
* Amazon Kinesis Data Streams.
* Amazon Simple Notification Service.
* Amazon Simple Email Service.
* Amazon Simple Queue Service.
* Amazon Cognito.
* AWS CloudFormation.
* Amazon CloudWatch Logs.
* Amazon CloudWatch Events.
* AWS CodeCommit.
* Scheduled Events (powered by Amazon CloudWatch Events).
* AWS Config.
* Amazon Alexa.
* Amazon Lex.
* Amazon API Gateway.
* AWS IoT Button.
* Amazon CloudFront.
* Amazon Kinesis Data Firehose.
* Other Event Sources: Invoking a Lambda Function On Demand.

## API Gateway Cheatsheet

### API Gateway

* API Gateway is a solution for creating secure APIs in your cloud environment at any scale
* Create APIs that act as a front door for applications to access data, business logic, or functionality from back-end services
* API Gateway throttles API endpoints at 10.000 requests per second (can be increased via service request through AWS support)
* Together with Lambda, API Gateway forms the app-facing part of the AWS serverless infrastructure.
* CloudFront is used as the public endpoint for API Gateway.
* Stages allow you to have multiple published versions of your API eg. prod, staging, QA.
* Each stage has an invoke URL which is the endpoint you use to interact with your API
* The API endpoint type can be edge-optimized, regional, or private, depending on where the majority of your API traffic originates from.
* An edge-optimized API endpoint is best for geographically distributed clients. API requests are routed to the nearest CloudFront Point of Presence (POP). This is the default endpoint type for API Gateway REST APIs.
* A regional API endpoint is intended for clients in the same region.
* A private API endpoint is an API endpoint that can only be accessed from your Amazon Virtual Private Cloud (VPC) using an interface VPC endpoint, which is an endpoint network interface (ENI) that you create in your VPC.
* You can use a custom domain for your invoke URL eg. api.exampro.co
* You need to publish your API via deploy API. You choose which stage you want to publish your API
* You defined multiple methods on your resources eg GET, POST, DELETE
* CORS issues are common with API Gateway, CORS can be enabled on all individual endpoints
* Caching improves latency and reduces the number of calls made to your endpoint
* Same origin policies help to prevent XSS attacks
* Same origin policies ignore tools like postman or curl
* Can enable Cross-Origin Resource Sharing (CORS) for multiple domain use with Javascript/AJAX. CORS is always enforced by the client.
* You can require authorization to your API via AWS Cognito or a custom Lambda
* The Amazon API Gateway logs (near real-time) back-end performance metrics such as API calls, latency, and error rates to CloudWatch.
* Amazon API Gateway is integrated with AWS CloudTrail to give a full auditable history of the changes to your REST APIs.
* With Amazon API Gateway, you only pay when your APIs are in use.
* There are no minimum fees or upfront commitments.
* API Gateway provides several features that assist with creating and managing APIs:
  + Metering – Define plans that meter and restrict third-party developer access to APIs.
  + Security – API Gateway provides multiple tools to authorize access to APIs and control service operation access.
  + Resiliency – Manage traffic with throttling so that backend operations can withstand traffic spikes.
  + Operations Monitoring – API Gateway provides a metrics dashboard to monitor calls to services.
  + Lifecycle Management – Operate multiple API versions and multiple stages for each version simultaneously so that existing applications can continue to call previous versions after new API versions are published.

## Cloudwatch Cheatsheet

### Cloudwatch

* Cloudwatch is a collection of monitoring services. It has Dashboards, Events, Alarms, Logs, and Metrics
* **Cloudwatch Logs:** Log data from AWS services. eg. CPU Utilization
* **Cloudwatch Metrics:** Represents a time-ordered set of data points. A variable to monitor eg. CPU utilization over time
* **Cloudwatch Events:** Trigger an event based on a condition eg. ever hour take a snapshot of the server
* **Cloudwatch Alarms:** Triggers notification based on metrics when a defined threshold is breached
* **Cloudwatch Dashboards:** Create visualization based on metrics
* EC2 monitors at 5 mins intervals and at Detailed Monitoring 1-minute intervals
* Most other service monitors at 1-minute intervals, with intervals of 1,3,5 minutes
* Logs must belong to a Log Group
* **Cloudwatch Agent** needs to be installed on the EC2 host to track **Memory Usage** and disc size
* You can stream custom log files eg. production logs
* Custom metrics allow you to track high-resolution metrics a sub-minute interval all the way down to 1 second.

## Cloudtrail Cheatsheet

### Cloudtrail

* AWS CloudTrail is a web service that records activity made on your account
* Governance, compliance, operational auditing, and risk auditing are keywords relating to Cloudtrail
* When you need to know who to do think Cloudtrail
* Cloudtrail by default logs event data fort he past 90s days via Event History
* To track beyond 90 days you need to create Trail
* To ensure logs have not been tampered with you need to turn on Log File Validation
* Cloudtrail logs can be encrypted using KMS (Key Management Service)
* Cloudtrail can be set to log across all AWS accounts in an Organization and all regions in an account
* Cloudtrail logs can be streamed to Cloudwatch logs
* You can also enable encryption using SSE KMS for additional security.
* Trails are output on an S3 bucket that you specify
* CloudTrail log files are encrypted using S3 Server Side Encryption (SSE).
* Cloudtrail logs two kinds of events: Management Events and Data events
* Management events provide insight into management operations that are performed on resources in your AWS account. These are also known as control plane operations. Management events can also include non-API events that occur in your account.
* Data Events provide insight into the resource operations performed on or within a resource. These are also known as data plane operations.
* Data Events are disabled bu default when creating a Trail
* Trail logs in S3 can be analyzed using Athena

## Cloudtrail Cheatsheet

### Cloudwatch vs Cloudtrail

* For performance monitoring, Cloudwatch; for auditing, Cloudtrail
* Log events across AWS services – think operations and Cloudwatch; Log API activity across AWS services – think activities and Cloudtrail
* For high-level comprehensive monitoring and eventing, Cloudwatch; for more low-level granular, Cloudtrail
* Cloudwatch asks how Cloudtrail asks who.

## AWS Config Cheatsheet

### AWS Config

* AWS Config is a fully managed service that provides you with an AWS resource inventory, configuration history, and configuration change notifications to enable security and governance.
* With AWS Config you can discover existing AWS resources, export a complete inventory of your AWS resources with all configuration details, and determine how a resource was configured at any point in time.
* These capabilities enable compliance auditing, security analysis, resource change tracking, and troubleshooting.
* Allow you to assess, audit, and evaluate *configurations* of your AWS resources.
* Creates a baseline of various configuration settings and files and can then track variations against that baseline.
* AWS CloudTrail records user API activity on your account and allows you to access information about this activity. But, AWS Config records point-in-time configuration details for your AWS resources as Configuration Items (CIs).
* You can use an **AWS Config** to answer **What did my AWS resource look like?** at a point in time. However, you can use **AWS CloudTrail** to answer **Who made an API call to modify this resource**.

## AWS OpsWorks Cheatsheet

### AWS OpsWorks

* AWS OpsWorks is a configuration management service that provides managed instances of Chef, Puppet, and **Ansible** tree very popular automation platforms.
* Automates how applications are configured, deployed, and managed.
* Provide configuration management to deploy code, automate tasks, configure instances, perform upgrades etc.
* OpsWorks lets you use Chef and Puppet to automate how servers are configured, deployed and managed across your Amazon EC2 instances or on-premises compute environments.
* OpsWorks is an automation platform that transforms infrastructure into code.
* OpsWorks consists of Stacks and Layers:
  + Stack are collections of resources needed to support a service or application.
  + Stacks are containers of resources (EC2, RDS, etc.) that you want to manage collectively.
  + Every Stack contains one or more Layers and Layers automate the deployment of packages.
  + Stacks can be cloned – but only within the same region.
  + Layers represent different components of the application delivery hierarchy.
  + EC2 instances, RDS instances, and ELBS are examples of Layers.
* OpsWorks is a global service. But when you create a stack, you must specify a region, and that stack can only control resources in that region.

## AWS System Manager Cheatsheet

### AWS System Manager

* AWS Systems Manager allows you to centralize operational data from multiple AWS services and automate tasks across your AWS resources.
* You can create logical groups of resources such as applications, different layers of an application stack, or production versus development environments.
* With Systems Manager, you can select a resource group and view its recent API activity, resource configuration changes, related notifications, operational alerts, software inventory, and patch compliance status.
* Systems Manager provides a central place to view and manage your AWS resources, so you can have complete visibility and control over your operations.
* SSM Agent installed by default on recent AWS-provided base AMIs for Linux and Windows.
* Manages AWS-based and on-premises based systems via the agent.
* Collects information about your instances and the software installed on them, helping you to understand your system configurations and installed applications.
* Lets you scan your managed instances for patch compliance and configuration inconsistencies.
* allows you to safely automate common and repetitive IT operations and management tasks across AWS resources.
* Provides you safe, secure remote management of your instances at scale without logging into your servers, replacing the need for bastion hosts, SSH, or remote PowerShell.
* Helps you select and deploy an operating system and software patches automatically across large groups of Amazon EC2 or on-premises instances.
* Lets you schedule windows of time to run administrative and maintenance tasks across your instances.
* Distributor is an AWS Systems Manager feature that enables you to securely store and distribute software packages in your organization.
* Provides a centralized store to manage your configuration data, whether plain-text data such as database strings or secrets such as passwords.

## AWS Trusted Advisor Cheatsheet

### AWS Trusted Advisor

* Trusted Advisor inspects the AWS environment to make recommendations for system performance, saving money, availability and closing security gaps Trusted Advisor checks the following four categories
* **Cost Optimization**
  + Recommendations that can potentially save money by highlighting unused resources and opportunities to reduce your bill.
* **Security**
  + Identification of security settings and gaps, inline with the best practices, that could make the AWS solution less secure
* **Fault Tolerance**
  + Recommendations that help increase the resiliency and availability of the AWS solution by highlighting redundancy shortfalls, current service limits, and over-utilized resources.
* **Performance**
  + Recommendations that can help to improve the speed and responsiveness of the applications
* **Service Limits**
  + Checks for service usage that is more than 80% of the service limit.
  + Values are based on a snapshot, so the current usage might differ.
  + Limit and usage data can take up to 24 hours to reflect any changes

## AWS Organization Cheatsheet

### AWS Organizations

* AWS Organizations helps you centrally govern your environment as you grow and scale your workloads on AWS.
* Organization helps you to centrally manage to the bill; control access, compliance, and security; and share resources across your AWS accounts.
* Using AWS Organizations, you can automate account creation, create groups of accounts to reflect your business needs, and apply policies for these groups for governance.
* You can also simplify billing by setting up a single payment method for all of your AWS accounts.
* Through integrations with other AWS services, you can use Organizations to define central configurations and resource sharing across accounts in your organization.
* AWS Organizations is available to all AWS customers at no additional charge.
* Available in two feature sets; Consolidated billing and All features.
* By default, organizations support consolidated billing features.
* With consolidated billing, you can see a combined view of charges incurred by all your accounts.
* Can also take advantage of pricing benefits from aggregated usage, such as volume discounts for Amazon EC2 and Amazon S3.
* Can help with cost control through volume discounts.
* Accounts can be migrated between organizations.
* You must have root or IAM access to both the member and master accounts.
* Use the AWS Organizations console for just a few accounts.
* Use the AWS Organizations API or AWS Command Line Interface (AWS CLI) if there are many accounts to migrate.
* You can use resource groups to organize your AWS resources.
* Resource groups make it easier to manage and automate tasks on large numbers of resources at one time.

## AWS Resource Access Manager Cheatsheet

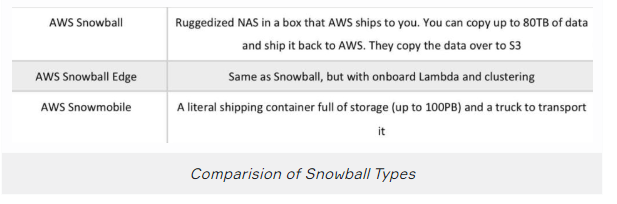
### AWS Resource Access Manager

* AWS Resource Access Manager (RAM) is a service that enables you to easily and securely share AWS resources with any AWS account or within your AWS Organization.
* You can share AWS Transit Gateways, Subnets, AWS License Manager configurations, and Amazon Route 53 Resolver rules resources with RAM.
* RAM eliminates the need to create duplicate resources in multiple accounts, reducing the operational overhead of managing those resources in every single account you own.
* You can create resources centrally in a multi-account environment, and use RAM to share those resources across accounts in three simple steps:
* **Key benefits:**
  + **Reduce Operational Overhead**– Procure AWS resources centrally, and use RAM to share resources such as subnets or License Manager configurations with other accounts. This eliminates the need to provide duplicate resources in every account in a multi-account environment.
  + **Improve Security and Visibility** – RAM leverages existing policies and permissions set in AWS Identity and Access Management (IAM) to govern the consumption of shared resources. RAM also provides comprehensive visibility into shared resources to set alarms and visualize logs through integration with Amazon CloudWatch and AWS CloudTrail.
  + **Optimize Costs** – Sharing resources such as AWS License Manager configurations across accounts allows you to leverage licenses in multiple parts of your company to increase utilization and optimize costs.

## Migration Cheatsheet

### AWS Snowball

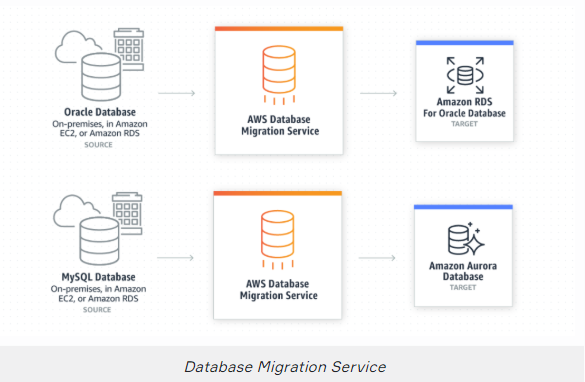
* Scale data transport solution for transferring data into or out of AWS.
* Snowball and Snowball Edge are a rugged container which contains a storage device.
* Snowball and Snowball Edge are for petascale migration. The snowmobile is for exabyte-scale migration
* Low Cost: Thousands of dollars to transfer 100 TB over high-speed internet, with Snowball it’s 1/5th
* Speed 100 TB over 100 days to transfer over high-speed internet, Snowball takes less than a week
* AWS Snowball Client is software that is installed on a local computer and is used to identify, compress, encrypt, and transfer data.
* Uses 256-bit encryption (managed with the AWS KMS) and tamper-resistant enclosures with TPM.
* Snowball must be ordered from and returned to the same region.
* To speed up data transfer it is recommended to run simultaneous instances of the AWS Snowball Client in multiple terminals and transfer small files as batches.
* Snowball can import to S3 (also Glacier) or export from S3.
* Snowball comes in two sizes:
  + 50 TB (42 TB of usage space)
  + 80 TB ( 72 TB of usage space)
* Snowmobile comes in one size: 100 PB
* You can both export or import data using Snowball or Snowmobile
* Snowball Edge can undertake local processing and edge-computing workloads
* Snowball Edge can use in a cluster in groups of 5 to 10 devices



## Migration Cheatsheet

### AWS Database Migration Service

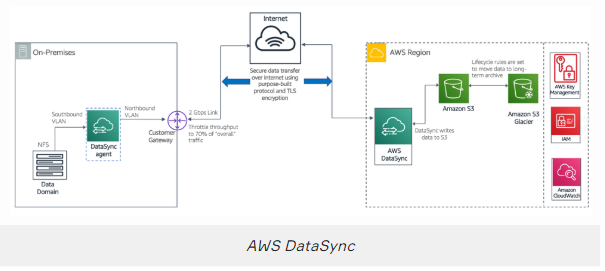
* AWS Database Migration Service (DMS) helps you migrate databases to AWS quickly and securely.
* The source database remains **fully operational during the migration**, minimizing downtime to applications that rely on the database.
* The AWS Database Migration Service can migrate your data to and from the most widely used commercial and open-source databases.
* Supported migration paths include:
  + On-premises and EC2 databases to Amazon RDS or Amazon Aurora.
  + Homogeneous migrations such as Oracle to Oracle.
  + Heterogeneous migrations between different database platforms, such as Oracle or Microsoft SQL Server to Amazon Aurora.
* With AWS Database Migration Service, you can continuously replicate your data with high availability and consolidate databases into a petabyte-scale data warehouse by streaming data to Amazon Redshift and Amazon S3.
* When migrating databases to Amazon Aurora, Amazon Redshift, Amazon DynamoDB or Amazon DocumentDB (with MongoDB compatibility) you can use DMS free for six months.
* Use along with the Schema Conversion Tool (SCT) to migrate databases to AWS RDS or EC2-based databases.
* The AWS Database Migration Service can migrate your data to and from the most widely used commercial and open-source databases.
* Schema Conversion Tool can copy database schemas for homogenous migrations (same database) and convert schemas for heterogeneous migrations (different databases).
* DMS is used for smaller, simpler conversions and also supports MongoDB and DynamoDB.
* SCT is used for larger, more complex datasets like data warehouses.
* DMS has replication functions for on-premise to AWS or to Snowball or S3.



## Migration Cheatsheet

### AWS DataSync

* AWS DataSync makes it simple and fast to move large amounts of data online between on-premises storage and Amazon S3 or Amazon Elastic File System (Amazon EFS).
* Manual tasks related to data transfers can slow down migrations and burden IT operations.
* DataSync eliminates or automatically handles many of these tasks, including scripting copy jobs, scheduling and monitoring transfers, validating data, and optimizing network utilization.
* The DataSync software agent connects to your Network File System (NFS) and Server Message Block (SMB) storage, so you don’t have to modify your applications.
* DataSync can transfer hundreds of terabytes and millions of files over the internet or AWS Direct Connect links.
* You can use DataSync to migrate active data sets or archives to AWS, transfer data to the cloud for timely analysis and processing, or replicate data to AWS for business continuity.
* DataSync can copy data between Network File System (NFS) or Server Message Block (SMB) file servers, all Amazon Simple Storage Service (Amazon S3) storage classes, and Amazon Elastic File System (Amazon EFS) file systems.
* All data is encrypted in transit with Transport Layer Security (TLS).
* DataSync supports using default encryption for S3 buckets using Amazon S3-Managed Encryption Keys (SSE-S3), and Amazon EFS file system encryption of data at rest.
* Task scheduling enables you to configure periodically executing a task, to detect and copy changes from your source storage system to the destination.
* You can schedule your tasks using the AWS DataSync Console or AWS Command Line Interface (CLI), without needing to write and run scripts to manage repeated transfers.
* The DataSync agent connects to your existing storage systems using the industry-standard NFS and SMB protocols.
* The agent transfers data rapidly and deposits it your designated Amazon S3 bucket or Amazon EFS file system.
* When copying data to Amazon S3, DataSync automatically converts each file to be a single S3 object in a 1:1 relationship, and preserves POSIX metadata as Amazon S3 object metadata.
* When you copy objects that contain file system metadata back to file formats, the original file metadata that DataSync copied to S3 is restored.
* Similarly, when Amazon EFS is the destination for your data, DataSync preserves existing directory structures and file metadata.
* DataSync supports VPC endpoints (powered by AWS PrivateLink) in order to move files directly into your Amazon VPC.
* The following diagram depicts how AWS DataSync works:



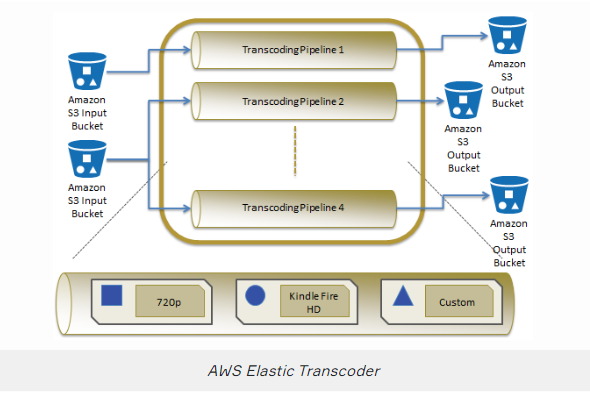
### AWS Server Migration Service (SMS)

* AWS Server Migration Service (SMS) is an agent-less service that makes it easier and faster for you to migrate thousands of on-premises workloads to AWS.
* AWS SMS allows you to automate, schedule, and track incremental replications of live server volumes, making it easier for you to coordinate large-scale server migrations.
* Automates migration of on-premises VMware vSphere or Microsoft Hyper-V/SCVMM virtual machines to AWS.
* Replicates VMs to AWS, syncing volumes and creating periodic AMIs.
* Minimizes cutover downtime by syncing VMs incrementally.
* Supports Windows and Linux VMs only (just like AWS)

## AWS Elastic Transcoder Cheatsheet

### AWS Elastic Transcoder

* Amazon Elastic Transcoder is a highly scalable, easy to use, and cost-effective way for developers and businesses to convert (or “transcode”) video and audio files from their source format into versions that will playback on devices like smartphones, tablets, and PCs.
* Supports a wide range of input and output formats, resolutions, bitrates, and frame rates.
* Also offers features for automatic video bit rate optimization, generation of thumbnails, overlay of visual watermarks, caption support, DRM packaging, progressive downloads, encryption, and more.
* Picks up files from an input S3 bucket and saves the output to an output S3 bucket.
* Uses a JSON API, and SDKs are provided for Python, Node.js, Java, .NET, PHP, and Ruby.
* Provides transcoding presets for popular formats.
* You are charged based on the duration of the content and the resolution or format of the media.



## Kinesis Cheatsheet

* Amazon Kinesis is the AWS solution for **collecting, processing, and analyzing streaming data** in the cloud. When you need **“REAL TIME”** think Kinesis
* Data is processed in “shards” – with each shard able to ingest 1000 records per second.
* There are four types of Kinesis service; Kinesis Data Streams, Kinesis Firehose, Kinesis Video Streams, Kinesis Data Analytics.
* **Kinesis Data Streams** enables you to build custom applications that process or analyze streaming data for specialized needs. Kinesis Data Streams enables real-time processing of streaming big data. It is useful for rapidly moving data off data producers and then continues processing the data.
* **Use case:** Accelerated log and data feed intake, real-time metrics and reporting, real-time data analytics, complex stream processing.
* **Kinesis Firehose** is the easiest way to load streaming data into data stores and analytics tools.
* Captures, transforms, and loads streaming data.
* Enables near real-time analytics with existing business intelligence tools and dashboards.
* Kinesis Data Streams can be used as the source(s) to Kinesis Data Firehose.
* Firehose Destinations include S3, Reshift, Elastisearch, or Splunk
* **Kinesis Data Analytics** is the easiest way to process and analyze real-time, streaming data. It can be used as standard SQL queries to process Kinesis data streams.
* Allows you to perform queries in real-time. Needs a Kinesis Data Streams/Firehose as the input and output
* **Use cases:** Generate time-series analytics, feed real-time dashboards, create real-time alerts, and notifications.
* **Kinesis Video Analytics** makes it easy to securely stream video from connected devices to AWS for analytics, machine learning (ML), and other processing.
* Securely ingest and stores video and audio encoded data to consumers such as SageMaker, Recognition, or other services to apply Machine learning and video processing

## Amazon EMR Cheatsheet

* Amazon Elastic Map Reduce(EMR) is a web service that enables businesses, researchers, data analysts, and developers to easily and cost-effectively process vast amounts of data.
* EMR is used as a hosted Hadoop framework running on Amazon EC2 and Amazon S3.
* Managed Hadoop framework for processing huge amounts of data.
* Also support Apache Spark, HBase, Presto, and Flink.
* Most commonly used for log analysis, financial analysis, or extract, translate, and loading (ETL) activities.
* A cluster is a collection of EC2 instances (maybe a hundred) provisioned by EMR to run your Steps.
* EMR is a good place to deploy Apache Spark, an open-source distributed processing used for big data workloads that utilize in-memory caching and optimized query execution.
* EMR launches all nodes for a given cluster in the same Amazon EC2 Availability Zone.
* If they ask to **grab the AWS logs and analyze** them think **EMR**.

## Amazon Glue Cheatsheet

* AWS Glue is a fully-managed, pay-as-you-go, **extract, transform, and load (ETL)** service that automates the time-consuming steps of **data preparation for analytics**.
* AWS Glue automatically discovers and profiles data via the Glue Data Catalog, recommends, and generates ETL code to transform your source data into target schemas.
* AWS Glue runs the ETL jobs on a fully managed, scale-out Apache Spark environment to load your data into its destination.
* AWS Glue also allows you to set up, orchestrate, and monitor complex data flows.
* You can create and run an ETL job with a few clicks in the AWS Management Console.
* Simply point AWS Glue to your data stored on AWS, and AWS Glue discovers data and stores the associated metadata (e.g. table definition and schema) in the AWS Glue Data Catalog.
* AWS Glue crawlers connect to a source or target data store, progress through a prioritized list of classifiers to determine the schema for the data, and then creates metadata in the AWS Glue Data Catalog.
* AWS Glue helps clean and prepare data for analysis by providing a Machine Learning Transform called FindMatches for deduplication and finding matching records.
* **Use Cases:**
  + Use AWS Glue to discover properties of data, transform it, and prepare it for analytics.
  + Glue can automatically discover both structured and semi-structured data stored in data lakes on Amazon S3, data warehouses in Amazon Redshift, and various databases running on AWS.
  + It provides a unified view of data via the Glue Data Catalog that is available for ETL, querying, and reporting using services like Amazon Athena, Amazon EMR, and Amazon Redshift Spectrum.
  + Glue automatically generates Scala or Python code for ETL jobs that you can further customize using tools you are already familiar with.

## Athena Cheatsheet

* Amazon Athena is an **interactive query** service that makes it easy to **analyze data in Amazon S3** using standard **SQL**.
* Athena is **serverless**, so there is no infrastructure to manage, and you pay only for the queries that you run.
* Athena is easy to use – simply point to your data in Amazon S3, define the schema, and start querying using standard SQL.
* Amazon Athena uses Presto with full standard SQL support and works with a variety of standard data formats, including CSV, JSON, ORC, Apache Parquet, and Avro.
* Amazon Athena uses a managed Data Catalog to store information and schemas about the databases and tables that you create for your data stored in Amazon S3.
* Athena automatically executes queries in parallel, so that you get query results in seconds, even on large datasets.
* Compare the Use Cases:
  + **Amazon Redshift** provides the fastest query performance for enterprise reporting and business intelligence workloads, particularly those involving extremely complex SQL with multiple joins and sub-queries.
  + **Amazon EMR** makes it simple and cost-effective to run highly distributed processing frameworks such as Hadoop, Spark, and Presto when compared to on-premises deployments. Amazon EMR is flexible – you can run custom applications and code, and define specific compute, memory, storage, and application parameters to optimize your analytic requirements.
  + **Amazon Athena** provides the easiest way to run interactive queries (ad-hoc queries) for data in S3 without the need to set up or manage any servers.
  + **AWS Glue** transforms and moves data to various destinations. Used to prepare and load data for analytics. The data source can be S3, Redshift, or other databases. Glue data catalog can be queried by Athena, EMR, and RedShift

## IAM Cheatsheet

* **Identity Access Management (IAM)** is used to managed access to users, groups and resources
* IAM is a universal system (applied to all regions at the same time). IAM is a free service.
* A root account is the account initially created when AWS is set up (full administrator)
* New IAM accounts have no permission by default until granted
* Permission must be explicitly granted to allow a user to access an AWS service.
* **IAM is universal (global)** and does not apply to regions.
* It is a best practice to not use the root account for anything other than billing.
* Power-user access allows all permissions except the management of groups and users in IAM.
* New users get assigned an **Access Key Id** and **Secret Access Key** when first created when you give them **programmatic access**
* Access Keys are only shown once when created. If lost they must be deleted/recreated again
* The Access Key ID and Secret Access Key are not the same as a password and cannot be used to login to the AWS console.
* MFA uses an authentication device that continually generates random, six-digit, single-use authentication codes.
* Always set up MFA for root Accounts
* Users must enable MFA on their own, Administrator cannot turn it on for each user
* IAM allows your set password policies to set minimum password requirements or rotate passwords
* IAM Identities as **Users, Groups, and Roles**
* **IAM Users**, end-users who log into the console or interact with AWS resources programmatically
  + The root account has full administrative permissions and these cannot be restricted.
  + You can have up to 5000 users per AWS account.
* **IAM Groups** have required permissions. This permissions can be automatically attached to users who are members of the group.
  + A group is not an identity and cannot be identified as a principal in an IAM policy.
* **IAM policies** JSON documents that grant permission for a specific user, group, or role to access services.
  + IAM policies are stored in IAM as JSON documents and specify the permissions that are allowed or denied.
  + The most restrictive policy is applied.
  + IAM policies can be set as a user (identity) based policies or resource-based policies.
  + Managed policies are policies provided by AWS and cannot be edited.
  + **Customer Managed Policies** are standalone identity–based policies that you create and which you can attach to multiple users, groups, or roles in your AWS account
  + Inline Policies are policies that are directly attached to a user
* **IAM roles** associate permissions to a role and then assign this to users or groups. You can define a role to the person who works into the on-premise environment.
  + A role can be assigned to a federated user who signs in using an external identity provider.
* **AWS Security Token Service (STS)** is a web service that enables you to request temporary, limited-privilege credentials for IAM users or for users that you authenticate (federated users).
  + By default, AWS STS is available as a global service, and all AWS STS requests go to a single endpoint at https://sts.amazonaws.com
  + You can optionally send your AWS STS requests to endpoints in any region (can reduce latency).
  + The advantage of this service that you do not have to distribute or embed long-term AWS security credentials with an application. The temporary security credentials have a limited lifetime, so you do not have to rotate them or explicitly revoke them when they’re no longer needed.
* **Cross-Account Access** lets users from one AWS account access resources in another. To make a request in a different account the resource in that account must have an attached resource-based policy with the permissions you need or you must assume a role (identity-based policy) within that account with the permissions you need.

## AWS Directory Service Cheatsheet

* **AWS Cloud Directory:** Cloud-native directory to share and control access to hierarchical data between the application
* **Use case:** Cloud applications that need hierarchical data with the complex relationship
* **AWS Cognito:** Sign-up and sign-in functionality that scales to millions of users and federated to public social media services like Google, Facebook etc.
* **Use case:** Develop consumer apps or SaaS
* **AWS Directory Service for Microsoft Active Directory:** AWS- managed full Microsoft AD running on Windows Server 2012 R2. Best choice if you have more than 5000 users and/or need a trust relationship set up.
* Runs on a Windows Server.
* On-premise users and groups can access resources in either domain using SSO.
* You can set up trust relationships to extend authentication from on-premises Active Directories into the AWS cloud.
* **Use case:** Enterprises that want hosted Microsoft AD or you need LDAP for Linux apps
* **AD Connector:** Allows on-premises users to log into AWS services with their existing AD credentials. Also allows EC2 instances to join AD domain
* AD Connector is a directory gateway for redirecting directory requests to your on-premises Active Directory.
* Connects your existing on-premise AD to AWS.
* The VPC must be connected to your on-premise network via VPN or Direct Connect.
* When users log in to AWS applications AD connector forwards sign-in requests to your on-premise AD DCs.
* **Use case:** Single sign-on for on-premises employees and for adding EC2 instances to the domain
* **Simple AD:** Low scale, low cost, AD implementation based on Samba
* Simple AD is generally the least expensive option.
* The best choice for less than 5000 users and doesn’t need advanced AD features.
* Powered by SAMBA 4 Active Directory compatible server.
* You can also sign on to the AWS management console with Simple AD user accounts to manage AWS resources.
* It has two options that are small and large. Smal supports up to 500 users, large supports up to 5000 user.
* **Use case:** Simple user directory or you need LDAP compatibility.

## AWS KMS Cheatsheet

### Encryption

**Encryption in flight (SSL)**

* Data is encrypted before sending and decrypted after receiving
* SSL certificates help with encryption (HTTPS)
* Encryption in flight ensures no MITM (man in the middle attack) can happen

**Server-side encryption at rest**

* Data is encrypted after being received by the server
* Data is decrypted before being sent
* It is stored in an encrypted form thanks to a key (usually a data key)
* The encryption/decryption keys must be managed somewhere and the server must have access to it

**Client-side encryption**

* Data is encrypted by the client and never decrypted by the server, instead, data will be decrypted by a receiving client
* The server should not be able to decrypt the data
* Could leverage Envelope Encryption

### KMS

* Anytime you hear “encryption” for an AWS service, it’s most likely KMS
* **AWS Key Management Store (KMS)** is a managed service that enables you to easily encrypt your data.
* AWS KMS provides a highly available key storage, management, and auditing solution for you to encrypt data within your own applications and control the encryption of stored data across AWS services.
* Never ever store your secrets in plaintext, especially in your code!
* Encrypted secrets can be stored in the code/environment variables
* KMS can only help in encrypting up to 4KB of data per call
* AWS KMS allows you to centrally manage and securely store your keys. These are known as customer master keys (CMK).
* Three types of Customer Master Keys (CMK):
  + AWS Managed Service Default CMK: free
  + User Keys created in KMS: $1 / month
  + User Keys imported (must be 256-bit symmetric key): $1 / month
* You can generate CMKs in KMS, in an **AWS CloudHSM cluster**, or import them from your own key management infrastructure.
* You set usage policies on these keys that determine which users can use them to encrypt and decrypt data and under which conditions.
* KMS is tightly integrated into many AWS services like Lambda, S3, EBS, EFS, DynamoDB, SQS etc.
* AWS services encrypt your data and store an encrypted copy of the data key along with the data it protects.
* When a service needs to decrypt your data they request KMS to decrypt the data key using your master key.
* If the user requesting data from the AWS service is authorized to decrypt under your master key policy, the service will receive the decrypted data key from KMS with which it can decrypt your data and return it in plaintext.
* All requests to use your master keys are logged in AWS CloudTrail so you can understand who used which key under which context and when they used it.
* KMS differs from Secrets Manager as its purpose-built for encryption key management.
* The AWS KMS custom key store feature combines the controls provided by AWS CloudHSM with the integration and ease of use of AWS KMS.
* You can configure your own CloudHSM cluster and authorize KMS to use it as a dedicated key store for your keys rather than the default KMS key store.

## AWS CloudHSM Cheatsheet

### AWS CloudHSM

* AWS CloudHSM is a cloud-based hardware security module (HSM) that enables you to easily generate and use your own encryption keys on the AWS Cloud
* CloudHSM is standards-compliant and enables you to export all of your keys to most other commercially-available HSMs, subject to your configurations
* CloudHSM complements existing data protection solutions and allows you to protect your encryption keys within HSMs that are designed and validated to government standards for secure key management.
* CloudHSM allows you to securely generate, store and manage cryptographic keys used for data encryption in a way that keys are accessible only by you.
* You can use the CloudHSM service to support a variety of use cases and applications, such as;
  + Database encryption,
  + Digital Rights Management (DRM),
  + Public Key Infrastructure (PKI),
  + Authentication and Authorization,
  + Document signing,
  + Transaction processing.
* Must be within a VPC and can be accessed via VPC Peering.
* **KMS** ----> AWS manages the **software for encryption**
* **CloudHSM** ---> AWS provisions **encryption hardware**

## Amazon Cognito Cheatsheet

* Cognito is a decentralized managed authentication system. When you need to easily add authentication to your mobile and desktop app think Cognito
* Your users can sign in directly with a user name and password, or through a third party such as Facebook, Amazon, or Google.
* The two main components of AWS Cognito are user pools and identity pools; user pools are user directories that provide sign-up and sign-in options for your app users, identity pools enable you to grant your users access to other AWS services.
* **User Pools** are user directory that allows users to authenticate using OAuth to IpD such as Facebook, Google, Amazon to connect to web-applications. Cognito user pool is in itself an IpD
* User Pools use JSON Web Token (JWT)s to persist authentication
* **Identity Pools** provide temporary AWS credentials to access service eg. S3, DynamoDB
* **Cognito Sync** can sync user data and preference across device with on-line code (powered by SNS)
* **Web Identity Federation** exchange identity and security information between an identity provider (IdP) and an application
* **Identity Provider (IdP):** A trusted provider of your user identity that lets you use to authenticate to access other services. eg Facebook, Google, Amazon
* OIDC is a type of Identity Provider that uses OAuth
* SAML is a type of identity Provider which is used for single sign-on

## AWS WAF, AWS Shield, and AWS Firewall Manager

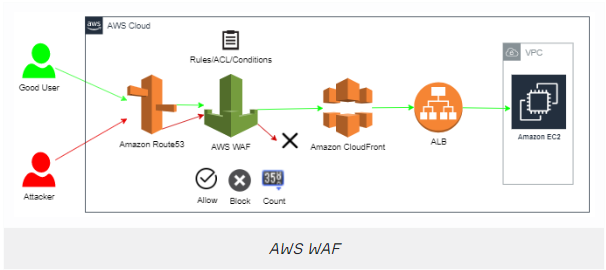
### AWS WAF, AWS Shield, and AWS Firewall Manager

* **AWS Web Application Firewall (WAF)** and **AWS Shield** help to protect your AWS resources from web exploits and Distributed Denial of Service (DDoS) attacks. If the question talks about **DDoS attacks** think WAF and Shield.
  + AWS WAF is a web application firewall that helps protect your web applications from common web exploits that could affect application availability, compromise security, or consume excessive resources.
  + For **additional protection** against DDoS attacks, use **AWS Shield**. Get 24/7 support from our DDoS response team and detailed visibility into DDoS events,
  + While AWS WAF can mitigate DDoS attacks at **layer 7** of the OSI reference model, AWS Shield protects web services from DDoS attacks at **layer 3 and 4** of the OSI reference model.
* As for **AWS Firewall Manager**, It simplifies your administration and maintenance tasks across multiple accounts and resources for AWS WAF rules, AWS Shield Advanced protection, and Amazon VPC security groups. The Firewall Manager service automatically applies your rules and other security protections across your accounts and resources, even as you add new accounts and resources.

## AWS WAF, AWS Shield, and AWS Firewall Manager

### AWS WAF

* AWS Web Application Firewall **(WAF)** helps protect web applications from attacks by allowing you to configure rules that allow, block, or monitor (count) web requests based on conditions that you define.
* WAF gives you control over which traffic to allow or block to your web applications by defining customizable web security rules.
* When AWS services receive requests for websites, the requests are forwarded to AWS WAF for inspection against defined rules.
* Pricing is based on how many rules you deploy and how many web requests your web application receives.
* Tightly integrated with Amazon CloudFront and the Application Load Balancer (ALB), services.
* When you use AWS WAF on Amazon CloudFront, rules run in all AWS Edge Locations, located around the world close to end-users.
* Web traffic filtering: AWS WAF lets you create rules to filter web traffic based on conditions that include IP addresses, HTTP headers, and body, or custom URIs.



* AWS Shield is a managed DDoS protection service that safeguards applications running on AWS.
* Provides always-on detection and automatic inline mitigations that minimize application downtime and latency, so there is no need to engage AWS Support to benefit from DDoS protection.
* There are two tiers of AWS Shield: Standard and Advanced.
* AWS Shield Standard defends against the most common, frequently occurring network and transport layer DDoS attacks that target your website or applications.
* When using AWS Shield Standard with Amazon CloudFront and Amazon Route 53, you receive comprehensive availability protection against all known infrastructure (Layer 3 and 4) attacks.
* AWS Shield Advanced can be used for higher levels of protection against attacks targeting your applications running on EC2, ELB, Amazon CloudFront, and Amazon Route 53 resources.

## CLI&SDK Cheatsheet

### CLI&SDK

* **CLI** stands for Command Line Interface
* **SDK** stands for Software Development Kit
* The AWS CLI lets you interact with AWS from anywhere by simply a command line
* The AWS SDK is a set of API libraries that let you integrate AWS services into your applications
* Programmatic Acces must be enabled per user via the IAM console to use CLI or SDK
* The command **“aws configure”** is used to set your AWS credentials up for CLI
* The CLI is installed via a Python script
* Credentials get stored in a plain text file (whenever possible use roles instead of AWS credentials)
* The SDK is available fort he following programming languages
  + C++
  + Go
  + Java
  + Javascript
  + .NET
  + NodeJs
  + PHP
  + Python
  + Ruby

## SQS Cheatsheet

### AWS SQS

* **Simple Queue Service (SQS)** is a queuing service using messages with a queue.
* SQS is used for application integration, it lets distributed/decoupled services or applications to talk to each other
* To read SQS use need to pull the queue using the AWS SDK.
* SQS uses pull-based (polling) not push-based.
* Messages can be kept in the queue from 1 minute to 14 days (default is 4 days).
* SQS supports both **Standart and First-in-first-out (FIFO)** queues
* Standart allows nearly unlimited messages per second, does not guarantee the order of delivery, always delivers at least once, you must protect again duplicate messages being processed
* FIFO queues preserve the exact order in which messages are sent and received.
* FIFO queues are available in limited regions currently.
* FIFO queues provide exactly-once processing, which means that each message is delivered once and remains available until a consumer processes it and deletes it.
* By default FIFO queues support up to 300 messages per second.
* There are two kinds of polling Short(Default) and Long Polling
* Short polling returns messages immediately, even if the message queue being polled is empty. It means short polling does not wait for messages to appear in the queue
* Long polling waits until a message arrives in the queue, or the long poll timeout expires. It eliminates false empty responses by querying all servers. SQS waits until a message is available in the queue before sending a response.
* In the majority of cases Long polling is preferred over short polling
* Visibility time-out is the period of time that messages are invisible in the SQS queue
* Messages will be deleted from the queue after a job has processed (before visibility timeout expires)
* If visibility timeout expires then a job will become visible to the queue
* The default visibility time-out is 30 seconds. The timeout can be 0 seconds to a maximum of 12 hours.
* Message size between 1 byte to 256 kb, Extended Client Library for Java can increase to 2 GB
* Differencies between Kinesis Data Streams and SQS is that Kinesis Data Streams routes related records to the same record processor, SQS sends messaging semantics such as message-level ack/fail visibility

## AWS SWF Cheatsheet

### AWS SWF

* **Amazon Simple Workflow (SWF)** helps developers build, run, and scale background jobs that have parallel or sequential steps.
* Amazon SWF is a fully-managed state tracker and task coordinator in the cloud.
* Create distributed asynchronous systems as workflows.
* Supports both sequential and parallel processing.
* If you need to recover or retry if a task fails, Amazon SWF can help you.
* Tracks the state of your workflow which you interact and update via API.
* Best suited for human-enabled workflows like an order fulfillment system or for procedural requests.
* AWS recommends that for new applications customers consider Step Functions instead of SWF
* SWF enables applications for a range of use cases, including media processing, web application back-ends, business process workflows, and analytics pipelines, to be designed as coordination of tasks.
* SWF has a completion time of up to 1 year for workflow executions.
* SWF uses a task-oriented API.
* Workers are programs that interact with Amazon SWF to get tasks, process received tasks, and return the results.
* **The decider** is a program that controls the coordination of tasks, i.e. their ordering, concurrency, and schedule according to the application logic.
* Step Functions is recommended to be used for new applications, except:
  + If you need external signals to intervene in the processes
  + If you need child processes that return values to parent processes

## Step Functions Cheatsheet

### AWS Step Functions

* AWS Step Functions lets you coordinate multiple AWS services into serverless workflows so you can build and update apps quickly.
* You can quickly build and run state machines to execute the steps of your application in a reliable and scalable fashion.
* It translates your workflow into a state machine diagram
* Create tasks, sequential steps, parallel steps, branching paths or timers.
* Amazon State Language declarative JSON.
* Apps can interact and update the stream via Step Function API.
* Visual interface describes the flow and real-time status.
* Detailed logs of each step execution.

## Amazon MQ Cheatsheet

### Amazon MQ

* Amazon MQ is a managed message broker service for Apache ActiveMQ that makes it easy to set up and operate message brokers in the cloud so you can migrate your messaging and applications without rewriting code.
* Message brokers allow different software systems–often using different programming languages, and on different platforms–to communicate and exchange information.
* SQS and SNS are “cloud-native” services, and they’re using proprietary protocols from AWS. Traditional applications running from on-premise may use open protocols such as MQTT, AMQP, STOMP, Openwire, WSS. When migrating to the cloud, instead of re-engineering the application to use SQS and SNS, we can use Amazon MQ
* Amazon MQ supports industry-standard APIs and protocols so you can migrate messaging and applications without rewriting code
* With Amazon MQ, you can use the AWS Management Console, AWS CloudFormation, the Command Line Interface (CLI), or simple API calls to launch a production-ready message broker in minutes.
* Fully managed and highly available within a region.
* Amazon MQ stores your messages redundantly across multiple Availability Zones (AZs).
* ActiveMQ API and support for JMS, NMS, MQTT, and WebSockets.
* Use SQS if you’re creating a new application from scratch. Use MQ if you want an easy low-hassle path to migrate from existing message brokers to AWS.
* Amazon MQ provides encryption of your messages at rest and in transit.

## SNS Cheatsheet

### AWS SWF

* **Amazon Simple Notification Service (SNS)** is a highly available, durable, secure, fully managed pub/sub messaging service.
* SNS is for Application Integration. It allows decoupled services and apps to communicate with each other
* Topic is a logical point and communication channel
* Uses simple APIs and easy integration with applications.
* A topic is able to deliver to multiple protocols
* You can encrypt topics via KMS
* Publishers use the AWS API via AWS CLI or SDK to push messages to a topic. Many AWS services integrate with SNS and act as publishers
* Subscriptions subscribe to topics. When a topic receives a message it automatically and immediately pushes messages to subscribers.
* All messages published to SNS are stored redundantly across multiple AZs.
* The following protocols:
  + HTTP and HTTPS create webhooks into your web-application
  + Email / Email-JSON messages are sent to registered addresses as email (text-based or JSON-object).
  + Amazon-SQS place SNS message into SQS queen
  + AWS Lambda triggers a lambda function
  + SMS send a text message
  + Platform application endpoints Mobile Push eg. Apple, Goole, Microsoft Baidu notification system

## SNS Cheatsheet

### Comparision of Application Integration Services

* **SNS:** Setup, operate, and send notifications from the cloud
  + Use case example: Send an email notification when Cloudwatch is triggered
* **Step Functions:** Out-of-the-box coordination of AWS service components with visual workflow
  + Use case example: Order processing workflow
* **SWF:** Amazon Simple Workflow (Amazon SWF) helps developers build, run, and scale background jobs that have parallel or sequential steps. You can think of Amazon SWF as a fully-managed state tracker and task coordinator in the cloud.
  + Use case example: Human enabled workflows like an order fulfillment system or for procedural request. AWS recommends that for **new applications customer** consider **Step functions** instead of SWF
* **SQS:** Messaging queue; store and forward patterns
  + Use case example: Building distributed/decoupled applications
* **Amazon MQ:** Managed message broker service for Apache ActiveMQ that makes it easy to set up and operate message brokers in the cloud.
  + Use case example: Easy low-hassle path to migrate from existing message brokers to AWS

## Amazon Workspaces Cheatsheet

### Amazon Workspaces

* Amazon WorkSpaces is a fully managed, secure cloud desktop service.
* WorkSpaces allows customers to easily provision cloud-based desktops that allow end-users to access documents and applications.
* Supported client-side devices include Windows and Mac computers, Chromebooks, iPads, Fire tablets, Android tablets, and Chrome and Firefox web browsers.
* WorkSpaces offers bundles that come with a Windows 7 or Windows 10 desktop experience, powered by Windows Server 2008 R2 and Windows Server 2016 respectively.
* By default users can personalize their workspaces but this can be locked down.
* Workspaces are persistent.
* You do not need an AWS account to login to workspaces.

## ECR & EKS Cheatsheet

### ECS

* **Amazon Elastic Container Service (ECS)** is a highly scalable, high-performance container management service that supports Docker containers and allows you to easily run applications on a managed cluster of Amazon EC2 instances.
* Amazon ECS eliminates the need for you to install, operate, and scale your own cluster management infrastructure.
* Using API calls you can launch and stop container-enabled applications, query the complete state of clusters, and access many familiar features like security groups, Elastic Load Balancing, EBS volumes, and IAM roles.
* Amazon ECS can be used to schedule the placement of containers across clusters based on resource needs and availability requirements.
* There is no additional charge for Amazon ECS. You pay for AWS resources (e.g. EC2 instances or EBS volumes) you create to store and run your application.
* Possible to use Elastic Beanstalk to handle the provisioning of an Amazon ECS cluster, balancing load, auto-scaling, monitoring, and placing your containers across your cluster.
* Alternatively, use ECS directly for more fine-grained control for customer application architectures.
* It is possible to associate a service on Amazon ECS to an Application Load Balancer (ALB) for the Elastic Load Balancing (ELB) service.
* ECS provides Blox, a collection of open-source projects for container management and orchestration. Blox makes it easy to consume events from Amazon ECS, store the cluster state locally and query the local data store through APIs.

**AWS Fargate**

* You can use Fargate with ECS to run containers without having to manage server sor clusters of EC2 instances.
* You no longer have to provision, configure, or scale a cluster of virtual machines to run containers
* Fargate only supports container images hosted on Elastic Container Registry (ECR) or Docker Hub

## ECR & EKS Cheatsheet

### ECR

Amazon Elastic Container Registry (ECR) is a managed Docker registry service. - ECR is account level Registry and regional service. - The EC2 Container Instance should have IAM permissions to access ECR - ECR only supports private images and needs authentication from an AWS account

### EKS

* Amazon also provides the Elastic Container Service for Kubernetes (Amazon EKS) which can be used to deploy, manage, and scale containerized applications using Kubernetes on AWS.
* Compatible with upstream Kubernetes so it’s easy to lift and shift from other Kubernetes deployments
* “Pods” are containers collocated with one another and can have shared access to each other

**Fargate Launch Type**

* The Fargate launch type allows you to run your containerized applications without the need to provision and manage the backend infrastructure. Just register your task definition and Fargate launches the container for you.
* Fargate Launch Type is a serverless infrastructure managed by AWS.
* Fargate only supports container images hosted on Elastic Container Registry (ECR) or Docker Hub.

**EC2 Launch Type**

* The EC2 launch type allows you to run your containerized applications on a cluster of Amazon EC2 instances that you manage.
* Private repositories are only supported by the EC2 Launch Type.

## CodeBuild & CodeDeploy & Code Pipeline & CodeCommit Cheetsheet

### AWS CodeCommit & AWS CodeBuild & AWS CodeDeploy & AWS CodePipeline

**AWS CodeCommit** – A fully-managed source control service that hosts secure Git-based repositories. CodeCommit makes it easy for teams to collaborate on code in a secure and highly scalable ecosystem. This solution uses CodeCommit to create a repository to store the application and deployment codes.

**AWS CodeBuild** – A fully managed continuous integration service that compiles source code, runs tests, and produces software packages that are ready to deploy, on a dynamically created build server. This solution uses CodeBuild to build and test the code, which we deploy later.

**AWS CodeDeploy** – A fully managed deployment service that automates software deployments to a variety of compute services such as Amazon EC2, AWS Fargate, AWS Lambda, and your on-premises servers. This solution uses CodeDeploy to deploy the code or application onto a set of EC2 instances running CodeDeploy agents.

**AWS CodePipeline** – A fully managed continuous delivery service that helps you automate your release pipelines for fast and reliable application and infrastructure updates. This solution uses CodePipeline to create an end-to-end pipeline that fetches the application code from CodeCommit, builds and tests using CodeBuild, and finally deploys using CodeDeploy.