- NOTES -

AWS (Amazon Web Services)

- A cloud computing platform.
- First product: **S3**, launched in **2006**.

Client-Server Model

Many clients use services from a **centralized server**.

Cloud Computing

- A computing service made available **over the internet**.
- It follows a **pay-as-you-go model** for delivering IT resources.

Deployment Models

➤ **Cloud-Based Deployment**

- This model allows building new applications or moving existing ones to the cloud.
- Different service levels (high to low) require different infrastructure, architecture, and management.
- **Example**: A company application using virtual servers, databases, and networking components entirely hosted in the cloud.

➤ **On-Premises Deployment / Private Cloud**

- Resources are deployed using **virtualization** and **resource management tools**.
- **Example**: Applications running on technology that remains within an organization's **on-premises data center**.

➤ **Hybrid Deployment**

- Connects **cloud resources** with **on-premises infrastructure**.
- **Example**: Suitable when working with **sensitive data** or under **government regulations**.

---

5. Why Choose Cloud Computing?

- Cost Savings
- Scalability
- Security
- Flexibility

Cloud Benefits

1. Payment Model

- Flexible and based on **variable expenses**.
- No upfront investment in data centers or servers.
- Pay-as-you-go model – use services from the cloud and consume as needed.

2. Fewer Operations

- No need for own data centers and servers.
- Reduces operational efforts and lets you focus on applications and customers.

3. Flexible Capacity

- No over-provisioning or under-provisioning.
- Pay only for what you use.

4. Economies of Scale

- Sharing infrastructure **reduces overall cost**.

5. Increased Speed & Flexibility

- Easier and faster deployment or development.
- Instant resource access.
- Rapid testing and utilization.

6. Global Reach

- AWS has **data centers worldwide**.
- Allows deploying applications globally.
- Enables **low-latency access** by selecting data centers near your clients.

AWS EC2 – Virtual Cloud Server

- **EC2 (Elastic Compute Cloud)**: Virtual server in AWS cloud.
- Easily scale capacity **up or down**.
- **On-demand access** to resources.
- **No upfront cost**, and **secure** environment.

Getting Started with AWS EC2

You can get started with EC2 in three steps.

---

1. Launch

- Start by selecting a template with basic configurations.
- The configuration includes: Operating system, Application server, Applications
- Choose the instance type and hardware configuration.
- Specify security settings to control traffic in and out of your instance.

---

2. Connect

- Multiple ways to connect to an instance.
- Programs and applications support various connection methods for data exchange.
- Users can login, access the computer desktop.

---

3. Use

- Once connected, you can use the instance.
- Execute commands to: Install software. Add storage, Copy and organize files

---

AWS EC2 Instance Types

There are different **instance types**. Which one to use depends on your **specific needs**.

More About EC2 Instance Types

- Different types are **optimized for different tasks**.
- Select an instance type based on **compute, memory, or storage requirements**.

➤ **General Purpose Instance**

- Balances **compute, memory**, and **networking resources**.
- Suitable for various purposes, such as:
  - Application servers
  - Gaming servers
  - Backend servers for companies
  - Small and medium databases
- Best used when there's a **balance between resources**.

➤ **Compute Optimized Instances**

- Ideal for **high-performance and compute-intensive** tasks.
- Best suited for:
  - Application servers
  - Gaming servers
  - Web applications
- Designed for situations where there is a **high compute need**.

➤ **Memory Optimized Instances**

- Delivers **fast processing** of **large dataset workloads**.
- Memory serves as a **temporary storage area** for:
  - Loading from storage
  - Holding and processing data
- Ideal when **huge amounts of data need preloading** before execution.

➤ **Accelerated Computing Instances**

- Uses **hardware accelerators** to **boost data processing**.
- Best suited for:
  - **Graphics applications**
  - **Streaming**

➤ **Storage Optimized Instances**

- Designed for workloads with **large datasets on local storage**.
- Best suited for:
  - Large file systems
  - Data warehouses
  - Online transaction systems
- Delivers **high input/output performance**.

AWS EC2 Pricing

- With **AWS EC2**, you **pay only for the compute time** you use.
- It offers **multiple pricing options** to suit different needs.

➤ **On-Demand Instances**

- Best for **short-term workloads**.
- **No upfront cost** or minimum purchase requirement.
- Instances run **until manually stopped**.
- **Pay-as-you-go** model — pay only for what you use.

➤ **AWS EC2 Savings Plan**

- Requires a **1-year or 3-year usage commitment**.
- Provides **discounted rates** for committed usage.
- If usage exceeds commitment, cost **reverts to on-demand rates**.
- *AWS Cost Explorer* helps plan and monitor usage and spending.

➤ **Reserved Instances**

- Reserve instances for **1-year or 3-year terms**.
- **Higher discounts** for longer commitments (3-year offers the most).
- Ideal for **steady-state workloads**.

➤ **Spot Instances**

- Suitable for **flexible workloads** that can handle interruptions.
- Offers up to **90% cost savings**.
- AWS utilizes unused capacity, passing the savings on to you.

➤ **Dedicated Hosts**

- Physical servers **dedicated entirely to you**.
- Allows use of **existing VM software licenses**.
- **Most expensive option**, ideal for **regulatory or licensing needs**.

• AWS EC2 Scaling

- Scaling is about using only the **resources you need**.
- Provides the **flexibility to grow freely**.
- Important to have an **architecture** that can handle **changes in demand**.
- A **scalable architecture** ensures you only **pay for resources** used at a given time.

• **AWS EC2 Auto Scaling**

- Servers may receive **more requests than they can handle**, causing **timeouts or outages**.
- **EC2 Auto Scaling** automatically **adds or removes instances** based on demand.
- It **automates capacity** adjustment to match traffic.

➤ **Scaling Approaches:**

- **Dynamic Scaling** – Responds to changing demand.
- **Predictive Scaling** – Schedules instance count based on predicted demand.
- Both can be **combined** to scale more effectively.

• **Auto Scaling Group Configuration**

- Acts as a **buffer layer** over EC2 instances.
- Automatically **adds instances** when needed and **terminates** them when idle.
- You can configure a **group of instances** with:
  - **Minimum capacity** – Always running.
  - **Desired capacity** – Defaults to minimum if not set.
  - **Maximum capacity** – Upper limit for instance scaling.
- Helps maintain a **dynamic, predictable, and cost-effective architecture**.

• **Elastic Load Balancing (ELB)**

- Distributes incoming **application traffic** across multiple instances.
- Acts as a **single point of contact** for web traffic.
- **Prevents any single resource** from getting overloaded.
- Automatically **scales** to handle **traffic spikes** without raising hourly costs.

➤ **Traffic Allocation**

- ELB allocates traffic among **available resources**.
- Works during both **high and low demand periods**.

• **Messaging and Queuing**

➤ **Monolithic Applications**

- Built with **tightly coupled components** (e.g., DB, UI, Server).
- If one component fails, the **entire service may crash**.

➤ **Microservices Architecture**

- Components are **loosely coupled** and can communicate independently.
- Failure of one component **does not affect the entire system**.
- Promotes **resilience and maintainability**.

➤ **AWS Services Supporting Integration:**

- **AWS SNS** – Simple Notification Service
- **AWS SQS** – Simple Queue Service

• **AWS SNS – Simple Notification Service**

- A **cloud-based notification** service.
- Fully managed **publish-subscribe** messaging.
- Supports **event-driven** architecture.
- Enables communication between **distributed systems** and **microservices**.
- Also supports **app-to-user** messaging via:
  - SMS
  - Push notifications
  - Email

• **Message Endpoints Supported by AWS SNS:**

- **HTTP/HTTPS**
- **Email / Email-JSON**
- **AWS SQS**
- **Applications**
- **AWS Lambda**
- **SMS** (Region-dependent)

• **Difference between AWS SQS and AWS SNS**

- **SNS**: Push-based **notification** system.
  - Delivers messages to **subscribed endpoints** automatically.
- **SQS**: Pull-based **queuing** system.
  - Receivers must **pull messages** and process/delete them.
- **SNS and SQS** can be used **together** effectively.

• **Cloud Queue Service – AWS SQS**

- AWS SQS (**Simple Queue Service**) is a **message queuing** service.
- Used to **exchange and store messages** between software components.
- **Messages are added to a queue**.
- **Services/users pick up** the messages from the queue.
- Once processed, **messages are deleted** from the queue.
- **Subscribers** to SQS can be:
  - Web applications
  - Servers
  - Emails
  - Serverless functions
  - Other cloud services

• **Serverless Computing**

- With serverless, **you don't manage servers** – only focus on **code**.
- Cloud provider handles **infrastructure management**.
- Helps in **scaling** and **cost reduction**.

• **Difference: EC2 vs Serverless**

Feature	AWS EC2	Serverless (e.g., Lambda)
Type	Virtual Servers	Abstracted compute service
Setup	Provision, deploy, manage	Upload code only
Maintenance	Manual server ops	Handled by AWS
Billing	Pay for uptime	Pay per execution time
Scalability	Manual or auto with configs	Auto-scaled by default

• **Serverless Cloud Compute – AWS Lambda**

- AWS Lambda is a **serverless compute service**.
- Lets you **run code without managing servers**.
- You only **pay for compute time** (when code runs).
- Focus stays on building applications, not infrastructure.

➤ **How AWS Lambda Works:**

1. Deploy your code to Lambda
2. Make it **trigger-ready**
3. Code runs **only when triggered**
4. **Billing starts only during execution**

➤ **Analogy:**

- Like a car – you **only consume fuel while driving**.
- Lambda only charges when the function is **in execution**.

➤ **Uses of AWS Lambda:**

- Build & deploy apps
- Monitor & maintain apps
- Run backend code

➤ **Popular Back-End Languages Supported:**

- Node.js
- Python
- Java
- Kotlin
- C#

• **Containers in the AWS Cloud**

- Containers help **deploy & manage applications**.
- Pack code + dependencies in a **single isolated object**.
- Ideal for **microservices architectures**.
- Removes external dependencies.
- Ensures **consistent behavior across environments**.

➤ **Benefits of Containerized Approach:**

- Easier **debugging**
- **Isolation** ensures no outside interference
- Consistency during **deployment and updates**

• **Containers and Scalability**

- Containers support large-scale environments:
  - **Tens of hosts, hundreds of containers**
- Must plan for **operations at scale**
- Use **container orchestration services** to manage:

➤ **Examples:**

- **AWS Elastic Container Service (ECS)**
- **AWS Elastic Kubernetes Service (EKS)**

• **Elastic Container Service – AWS ECS**

- ECS helps you **run and manage containerized applications**.
- Built for **scale and high performance**.

• **AWS ECS and Docker**

- ECS supports **Docker**, a platform to **build, test, and deploy** apps.
- AWS supports both:
  - **Open Source Docker**
  - **Enterprise Docker Editions**
- Use **API calls** to launch and stop Docker containers.

Kubernetes on AWS – AWS EKS (Elastic Kubernetes Service)

- **AWS EKS** is a **managed Kubernetes** service by AWS.
- Lets you **run Kubernetes clusters on AWS** or on-premise.
- Designed for **scalability and flexibility**.
- **Always updated** with the latest Kubernetes features.

➤ **What is Kubernetes?**

- An **open-source** system for managing containerized applications.
- Provides **deployment**, **scaling**, and **management** of containers.
- Backed by a **large open-source community**.

➤ **EKS Deployment Options:**

- Cloud deployment (fully managed)
- On-premises deployment
- With your own DevOps tools

Serverless Compute for Containers – AWS Fargate

- Fargate allows you to **deploy containers without managing servers**.
- **Serverless approach** for container workloads.
- **Pay-as-you-go** pricing model – only pay when containers are running.
- Focus on **building apps**, not infrastructure.

Introduction to AWS Infrastructure

- AWS has **global infrastructure** with **Data Centers worldwide**.
- You can **deploy apps globally** or in a **specific location**.

➤ **Benefits of Global Infrastructure:**

- **Low latency** if Data Center is near the user
- **High availability** – if one Data Center fails, others take over
- **Fast content delivery**

Introduction to AWS Regions

- A **Region** is a physical location in the world with **multiple Availability Zones (AZs)**.
- AWS offers **many global regions** to choose from.

➤ **Reasons to Choose a Region:**

- **Data regulations** (compliance, privacy laws)
- **Customer proximity** (reduce latency)
- **Service availability** (not all services in every region)
- **Pricing differences** between regions

Government Regulations and Data Governance

- Some countries require **data residency** within national borders.
- Choose regions accordingly to stay **compliant** with these laws.

Customer Proximity Example:

- If your company is in **Seattle** and customers are in **Norway**, you can:
  - Deploy backend in Seattle
  - Run front-end or caching systems in **Norwegian regions** for faster response

---

Region Availability Services

- Not all AWS services are available in every region.
- New services are **gradually rolled out** to various regions.

AWS Pricing

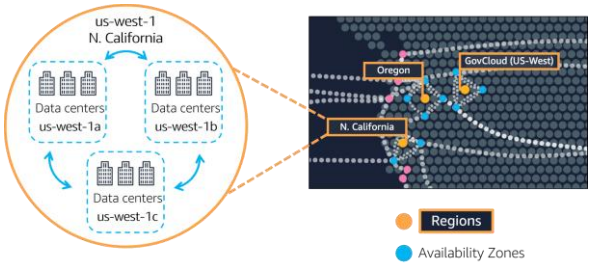
- Prices can **vary by region**.
- Pricing depends on:
  - Resource usage
  - Region
  - Service type
- Detailed pricing info is covered in later modules.

Introduction to Availability Zones (AZs)

- AZ = **One or more isolated Data Centers** within a region.
- Located **miles apart** to avoid failures from natural disasters.
- Still **close enough** to maintain **low-latency communication**.

➤ **Availability Zones Illustration:**

- Example region: **N. California**
  - Contains: **US-West-1A, US-West-1B, US-West-1C**
- Zones are built to be **fault-tolerant** and **highly available**.
- Illustrated as blue circles within regional orange circles (e.g., N. California, Oregon, GovCloud).



Edge Locations: Introduction

- **Edge Location** is a data center used to deliver content quickly to users.
- It is the **nearest site** to the users.

Fast Delivery using CloudFront

- AWS uses **CloudFront** at Edge Locations.
- CloudFront stores **cached copies** of content.
- This results in **faster delivery** of the data.

What is Cache?

- **Caching** helps software deliver content faster and more affordably.
- Cache = fast storage that **copies & stores** data (e.g., **RAM**).

- Purpose: Deliver content quickly without accessing slow main storage.
- Data is delivered from the **Edge Location** closest to the user.

How Cache Works

- Cache saves parts of the data (subset).
- On **first request**, data is stored in the cache.
- On **subsequent requests**, data is delivered **faster** from cache at Edge Location.

AWS Cloud Resource Provisioning

• Ways to Interact with AWS Services:

1. **AWS Management Console**
   - Web-based interface to manage services.
   - Also available as a **mobile app** (for monitoring, billing).
2. **AWS CLI (Command Line Interface)**
   - Save time with API requests.
   - Manage multiple services via scripts.
   - Supported on **Windows, macOS, Linux**.
3. **SDKs (Software Development Kits)**
   - Access services using **API** in your preferred language.
   - Can integrate into **existing or new** apps.
   - Supported languages: **Java, C++, .NET, Python**, etc.

How to Provision AWS Resources

1. AWS Elastic Beanstalk

- Provide **code + configuration**.
- Beanstalk handles:
  - Capacity adjustment
  - Load balancing
  - Auto-scaling
  - Health monitoring

2. AWS CloudFormation

- Use code to **define infrastructure**.
- Handles configuration and provisioning.
- Manages **resource dependencies**.
- No need for manual setup via Console.

AWS Elastic Beanstalk

• Cloud Orchestration Service

- Web infrastructure **management service**.
- Automates:
  - Deployment
  - Scaling
  - Configuration
  - Provisioning

• Services Managed by Beanstalk:

- **EC2, S3, RDS, DynamoDB, SimpleDB**

• Supported Platforms:

- **Docker**
- **Go**
- **Java / Java with Tomcat**
- **.NET**
- **Node.js**
- **PHP**
- **Python**
- **Ruby**

AWS CloudFormation

• What is CloudFormation?

- **Infrastructure-as-Code** service.
- Create **templates** to describe required AWS resources.
- Automatically **configures and provisions** everything.
- Handles **inter-resource dependencies**.

AWS Networking – Overview

AWS Cloud Connectivity

- AWS offers multiple network connectivity options depending on setup and requirements.

AWS Virtual Private Cloud (VPC)

- VPC: Isolated section of the AWS Cloud to launch resources in a virtual network.
- Organizes resources into subnets.
- Subnets: Sections within a VPC for organizing resources with specific security or operational needs.

Internet Gateway

- Acts as a door between the VPC and the Internet.
- Required for allowing public traffic to enter/exit the VPC.
- Without it, public access is not possible.

Virtual Private Gateway

- Enables access to private resources in the VPC.
- Adds encryption and security layers.
- Facilitates VPN (Virtual Private Network) setup.
- Restricts traffic to approved networks only.

AWS Direct Connect

- Provides a dedicated private connection between on-premises data center and AWS VPC.
- Offers:
    - Higher bandwidth
    - Lower latency
    - Increased security

AWS Cloud Subnet and Access

Subnets

- Allow grouping of resources in a VPC.
- Two types:
    - Public Subnet: Accessible by the internet (e.g., website frontends).
    - Private Subnet: Only accessible via private network (e.g., databases).

Communication

- Public and private subnets can communicate securely within the same VPC.

Network Traffic in a VPC

- Data is sent and received as Packets.
- Packets must pass permission checks before entering subnets.

Network Access Control Lists (ACLs)

- Stateless firewalls at the subnet level.
- Control inbound and outbound traffic.
- Follow a list of rules:
    - Allowed: Pass through
    - Denied: Dropped
- Stateless: No memory of past requests.

Security Groups

- Firewalls specific to EC2 instances.
- Control inbound and outbound traffic at the instance level.
- Default: All inbound traffic denied, all outbound traffic allowed.
- Stateful: Remembers previous packet decisions.

Configuration

- Both ACLs and Security Groups can be custom-configured with rules.

Global Networking – DNS

Domain Name System (DNS)

- Maps domain names to IP addresses.
- Works like a phonebook for websites.

AWS Route 53

- AWS's DNS web service.
- Routes users to applications hosted on AWS or external resources.
- Features:
    - DNS management
    - Domain registration and transfer
    - Traffic routing

Route 53 and CloudFront

- CloudFront (CDN) and Route 53 can be used together to deliver content efficiently and globally.

Architecture Flow:

1. User requests content.
2. Route 53 resolves the domain to an IP address.
3. CloudFront sends the request to the nearest edge location.
4. Load Balancer forwards the request to an EC2 instance in an Auto Scaling group.
5. Response is returned to the user.

AWS Storage and Databases

Concepts Covered:

- Basic storage and database concepts

Storage Services

- AWS EBS (Elastic Block Store): Block storage for EC2.
- AWS S3 (Simple Storage Service): Scalable object storage.
- AWS EFS (Elastic File System): Shared file storage for Linux instances.
- Storage solutions vary based on workload needs.

Database Services

- AWS RDS: Managed relational database (e.g., MySQL, PostgreSQL).
- AWS DynamoDB: Fully managed NoSQL database.
- AWS Redshift: Data warehouse for analytics.
- AWS DMS (Database Migration Service): Migrate databases to AWS with minimal downtime.

AWS Instance Stores

- **Instance Store**
    - Acts as a **physical hard drive**.
    - Provides **temporary storage** for EC2 instances.
    - **Data persists only during the lifetime** of the instance.
    - **Reboot** → Data **persists**.

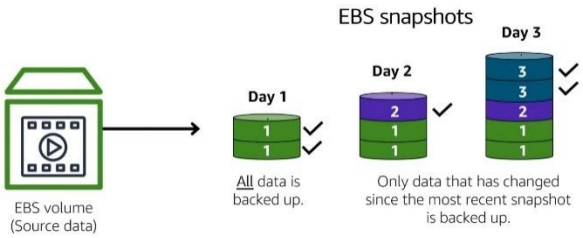
- **Hibernate / Terminate** → Data is **lost**.
- If EC2 starts on a different host → Instance store **does not follow**.
- **Not recommended** to store important data.
- Best used for **temporary files** or data that can be recreated.

AWS EBS (Elastic Block Store)

- Provides **persistent storage volumes** for EC2 instances.
- Attach volume to EC2 after creation.
- **Data remains** even if the instance is **stopped or terminated**.
- Use **EBS Snapshots** for backups.

EBS Snapshots

- **Incremental backups**
- First snapshot → full backup.
- Next backups → only changed blocks.
- **Storage efficient** – no duplicate data.
- Deleting a snapshot only removes **unique blocks**.

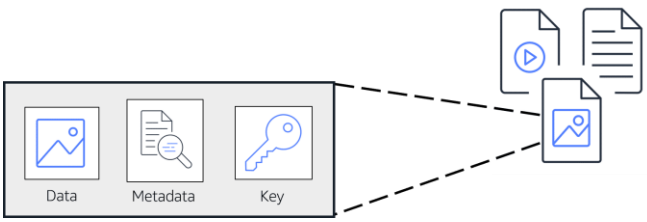


AWS S3 (Simple Storage Service)

- Object-level **cloud storage service**.
- Can upload **any type of file** (up to **5 TB**)
- Set **access permissions** to objects.
- **Unlimited storage capacity**

Object Storage Structure:

- **Data** – actual file
- **Metadata** – file info
- **Key** – unique ID for each object



AWS S3 Storage Classes

Class	Description
S3 Standard	For **frequently accessed** data High availability (≥ 3 AZs) Most expensive
S3 Standard-IA	For **infrequent access** High availability Lower storage cost, **higher retrieval cost**
S3 One Zone-IA	Stored in **1 AZ** Cheaper than S3 Standard & IA
S3 Intelligent-Tiering	For **unknown or changing access patterns** Automatically shifts data between tiers
S3 Glacier	For **archiving** Slower retrieval (few minutes) Very cheap
S3 Glacier Deep Archive	Lowest cost Retrieval within **12 hours** For **long-term archiving**

Comparison: AWS EBS vs AWS S3

Feature	AWS EBS	AWS S3
Data Format	**Blocks**	**Objects**
Max Size	16 TiB/block	5 TB/object
Modifiable	Yes	No (must reupload)
Speed	Faster	Slower
Durability	N/A	Very high (no corruption over time)

AWS EFS (Elastic File System)

- Cloud-based **file system**.
- Data accessed using **file paths**.
- Stored in **multiple AZs** → highly available.
- **Auto-scaling** without app disruption.
- Ideal for **shared access** by multiple services.

● AWS RDS – Relational Database Service

- **RDS (Relational Database Service)** is a cloud-based service by AWS that automates database tasks.
- Allows running **relational databases** in AWS Cloud.
- **Supported Database Engines**:
  - Amazon Aurora
  - PostgreSQL
  - MySQL
  - MariaDB
  - Oracle Database

- Microsoft SQL Server
- **Key Features**:
  - Data encryption (in transit and at rest)
  - Automates administrative tasks
  - Saves time for developers to focus on application features

---

● **What is a Relational Database?**

- A **relational database** stores data in tables and links it with other data.
- Uses **SQL (Structured Query Language)** for storing and querying data.

Example Table:

ID	Product Name	Price
1	T-shirt	$20
2	Jeans	$35
3	Shoes	$50

- SQL helps maintain **data consistency** and **clarity**.

● **Amazon Aurora**

- **High-performance relational database** built for enterprises.
- Provides **high availability** and **scalability**.
- **Performance**:
  - 5x faster than MySQL
  - 3x faster than PostgreSQL
- **Storage**:
  - 6 copies of data across **3 Availability Zones**
  - Backed up on **Amazon S3**
  - Ensures **data durability and availability**

● **AWS DynamoDB – Non-relational Cloud Database**

- **DynamoDB** is a **serverless, NoSQL database**.
- High performance, no need to manage servers.

What is a Non-Relational Database?

- Uses **key-value pair** structure.
- Keys identify items, values store attributes.
- Items can have **different attributes**.

Example:

ID Data

1 Name: Katherine Smith
Profession: Photographer

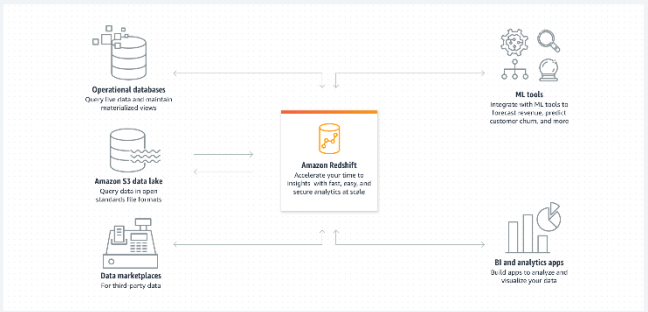
2 Name: Joe Doe
Age: 35
Height: 190cm

● **Comparing AWS RDS and DynamoDB**

AWS RDS	AWS DynamoDB
Supports complex data	Supports simple structured data
Stores data in tables	Stores data in documents
More expensive	Cost-effective
Slower performance	High-speed performance

● **AWS Redshift – Big Data Analytics**

- **AWS Redshift** is used for **big data analytics**.
- Gathers and analyzes data from multiple sources.
- Helps discover **connections and insights** from data.
- Powered by:
  - **SQL**
  - **AWS-designed hardware**
  - **Machine learning**
- Ideal for **large and complex datasets** that go beyond relational DB limits.



● **AWS DMS – Database Migration Service**

- **AWS DMS** is used to **migrate databases**.
- **Source Database**: Where the data is currently stored.
- **Target Database**: Where the data will be moved.

Key Points:

- Source DB remains operational during migration.
- Simple to use, low cost, reliable.
- Supports a wide range of databases.

Use Cases:

- Testing applications against real data safely
- Merging multiple databases into one
- Sending data to different data sources

AWS Additional Database Services

AWS DocumentDB

- Document-based NoSQL database.
- Compatible with MongoDB.
- Ideal use cases:
  - Content Management Systems (CMS)

- User profiling
- Cataloging

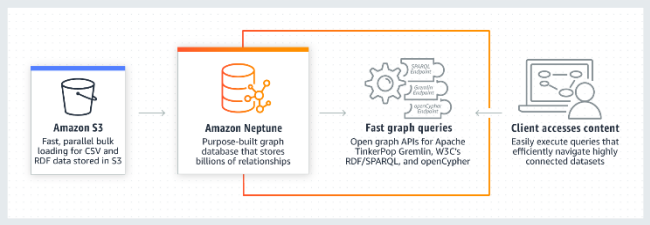
- Example document:

The following is an example of a document inside a document-based database.

```
{
 name: "Jason",
 age: "29",
 city: "New York"
 profession: "Accountant"
}
```

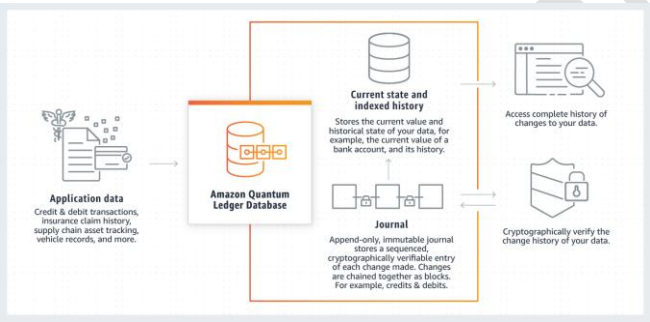
AWS Neptune

- Graph database service.
- Best for:
  - Financial records
  - Supply chain systems
  - Centralized digital records



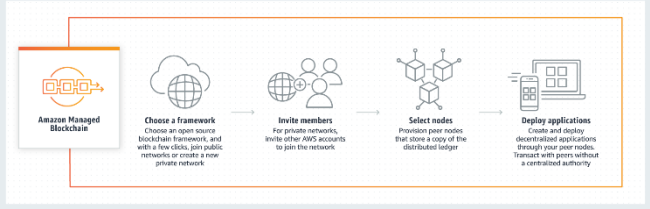
AWS QLDB (Quantum Ledger Database)

- Ledger database service.
- Maintains a cryptographically verifiable transaction log.
- Use cases:
  - Financial records
  - Supply chains
  - Immutable history of changes



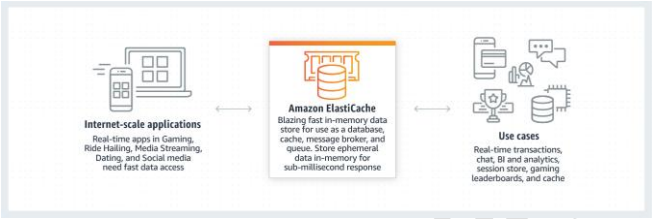
AWS Managed Blockchain

- Managed blockchain network using open-source frameworks.
- Supported frameworks:
  - Ethereum
  - Hyperledger Fabric
- Enables:
  - Creation and joining of blockchain networks
  - Management with minimal setup



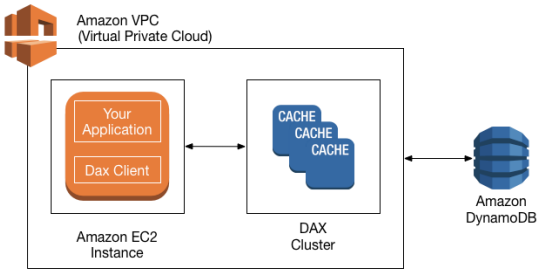
AWS ElastiCache

- Adds caching layers on top of databases.
- Boosts read performance by storing frequently accessed data in-memory.
- Supported engines:
  - Redis
  - Memcached



AWS DynamoDB Accelerator (DAX)

- In-memory cache specifically for DynamoDB.
- Reduces response times from milliseconds to microseconds.
- Fully managed, secure, and highly available.



AWS Cloud Security

Topics Covered

- Shared responsibility model
- IAM & access control
- AWS Organizations
- Compliance and security services

Shared Responsibility Model

Responsibility Area	AWS	Customer (You)
Physical infrastructure	Responsible	Not responsible
Network & hardware	Responsible	Configure use
Operating system & apps	Not responsible	Responsible
Identity & access	Not responsible	Responsible
Data encryption	Partial (server-side)	Responsible (client-side, app)

- AWS secures the cloud infrastructure.
- Customer secures everything built inside the cloud (data, configuration, etc.).

AWS Identity and Access Management (IAM)

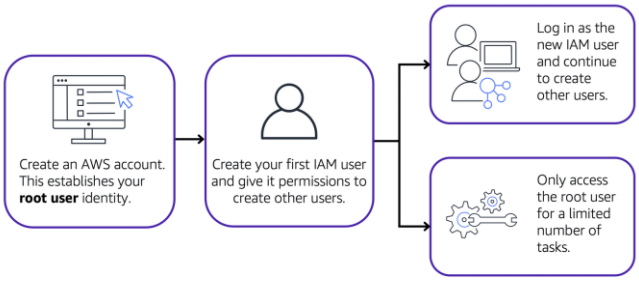
IAM Features

- AWS Account Root User
- IAM Users

- IAM Groups
- IAM Policies
- IAM Roles
- Multi-Factor Authentication (MFA)

AWS Account Root User

- Created when an AWS account is created.
- Has full control over all resources.
- Best practices:
  - Avoid daily use
  - Use only for tasks requiring root access
  - Create IAM users for operational use

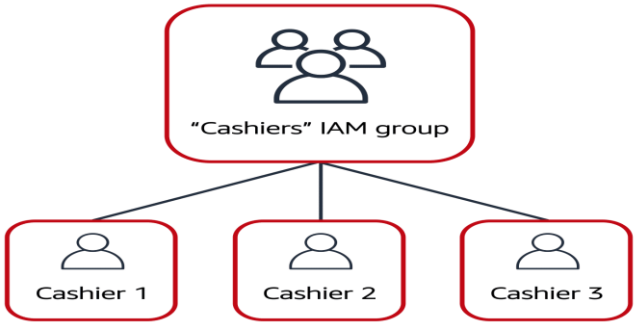


IAM Users

- Represents individuals or applications.
- Created with a username and credentials.
- No permissions by default.
- Permissions are granted through IAM policies.

IAM Groups

- A collection of IAM users.
- Policies assigned to the group apply to all members.



IAM Policies

- JSON documents that define permissions.
- Control access to AWS resources.
- Follow the principle of least privilege.
- Example:

```
{
  "Version": "2012-10-17",
  "Statement": {
    "Effect": "Allow",
    "Action": ["s3:ListObject", "s3:GetObject"],
    "Resource": ["arn:aws:s3::: awsdoc-example-bucket",
"arn:aws:s3::: awsdoc-example-bucket/*"]
  }
}
```

IAM Roles

- Provides temporary access to services/resources.
- Must be assumed by a trusted entity (user, service, app).
- Useful for temporary tasks, automation, or cross-account access.

Multi-Factor Authentication (MFA)

- Adds an extra layer of login security.
- Uses additional authentication factors (e.g., OTP, security keys).
- Recommended for both root and IAM users.

AWS Organizations

What is AWS Organizations?

- A management tool for multiple AWS accounts.
- Root user manages the organization.
- Uses Service Control Policies (SCPs) to set permissions at the org level.

Organizational Units (OUs)

- Logical groupings of accounts with similar permissions.
- SCPs applied at the OU level affect all accounts within it.

AWS Cloud Compliance

AWS Artifact

- Provides on-demand access to AWS security and compliance reports.
- Two main components:
  - AWS Artifact Reports – Official compliance documents
  - AWS Artifact Agreements – Legal usage agreements for sensitive data

Customer Compliance Center

- A resource hub for compliance best practices.
- Includes:
  - Audit guides and security checklists
  - Case studies and real-world compliance solutions
  - Support for common compliance frameworks

- **Denial-of-Service Attacks (DoS) & Distributed Denial-of-Service Attacks (DDoS)**
  - **Denial-of-Service (DoS) Attack**
    - Attempts to make a website or application inaccessible.
    - Caused by flooding the network with excessive traffic.
    - Originates from a **single source**.
  - **Distributed Denial-of-Service (DDoS) Attack**
    - Similar goal as DoS: make applications or websites inaccessible.
    - Attack originates from **multiple sources**.

- May involve **bots** (infected systems that flood the app/site with traffic).

- **AWS Shield**
  - Protection service provided by AWS against DoS & DDoS attacks.
  - Two types:
    - **Standard**: Free, protects all AWS users.
    - **Advanced**: Paid, provides detailed insights and advanced mitigation features.

• Additional Security Services

- **AWS KMS (Key Management Service)**
  - Manages cryptographic keys for data encryption and decryption.
  - You control access and permissions using IAM.
  - **Encryption** = locking data
  - **Decryption** = unlocking data
- **AWS WAF (Web Application Firewall)**
  - Monitors and filters network requests.
  - Uses **Access Control Lists (ACLs)** to allow/block traffic.
- **Amazon Inspector**
  - Analyzes apps for vulnerabilities and compliance issues.
  - Provides security findings and improvement suggestions.
- **Amazon GuardDuty**
  - **Threat detection** service for AWS infrastructure.
  - Continuously monitors network activity.
  - Reports threats and gives **fix recommendations**.

• AWS Monitoring and Analytics Tools

AWS CloudWatch

- Cloud-based monitoring & management tool.
- **Web-based interface** to track resource performance and usage.
- **Alarms**:
  - Triggered based on custom metrics & thresholds.
  - Example: Automatically stop idle resources.
- **Dashboard**:
  - Visual overview of AWS resources.
  - Multiple dashboards can be created as needed.

AWS CloudTrail

- Logs all API calls & user actions in your AWS environment.
- Tracks:
  - Identity
  - Time
  - IP address
  - Actions performed
- **CloudTrail Events**:
  - Help answer: Who? What? When? How?
  - Provide contextual insights into events.
- **CloudTrail Insights**:
  - Detects unusual or abnormal API activities.

AWS TrustedAdvisor

- Real-time, web-based recommendation tool.
- Evaluates AWS usage based on 5 categories:
  - **Cost Optimization**
  - **Performance**
  - **Security**
  - **Fault Tolerance**
  - **Service Limits**
- Returns:
  - Recommendations
  - Learning resources to adopt AWS best practices
- **Dashboard Symbols**:
  - ✅ Green Check: No issues
  - ⚠️ Orange Triangle: Investigate
  - ⛔ Red Circle: Action Required

AWS Pricing and Support

---

1. Consolidated Billing (Cloud Services)

- AWS allows **management of multiple accounts** from a **central location**.
- Centralized billing results in:
  - **One single bill** across multiple accounts.
  - Simplified **cost tracking**.
  - Shared **volume discounts** across accounts.
- **Core Idea**: Merging multiple bills into one.
- **Limit**: Maximum of **4 accounts** per organization (can request increase via AWS Support).

---

2. AWS Free Tier Account

AWS Free Tier allows users to explore services **free of cost** for a limited time.

Free Tier Offerings:

- **Always Free** – Services that remain free with usage limits.
- **12 Months Free** – Available for new accounts, free for 12 months.
- **Trials** – Short-term free trials for selected services.

---

3. AWS Pricing Models

➤ **Pay for What You Use**

- **No long-term contracts** or licensing required.
- Pay only for actual resource usage.

➤ **Pay Less When You Reserve**

- **Commit** in advance for future usage.
- **Discounted pricing**, but payment is required even if resources are unused.
- Ideal when future usage is predictable.

➤ **Volume-Based Discount (Use More, Pay Less)**

- As usage increases, **per-unit pricing decreases**.
- Cross a threshold to unlock **lower prices**.

---

4. AWS Pricing Calculator

- Helps estimate costs for AWS services.
- Allows **grouping estimates** to reflect organizational structure (e.g., cost centers).
- **Sharable links** for collaboration.
- **Detailed breakdowns** by resource and configuration.
- Assists in decision-making:
    - Choosing **regions**
    - Selecting **instance types** based on needs

---

5. AWS Billing Dashboard

- A tool to manage and monitor AWS billing and usage.

Features:

- Pay AWS bills
- Monitor **usage and spending**
    - View spending by **day**, **month**, or **year-to-date**
    - Track **remaining Free Tier usage**
- **Create saving plans**
- Generate **cost and usage reports**
- **Compare billing periods**

---

6. Pricing Examples

- **AWS Lambda Pricing**
- **AWS EC2 Pricing**
- **AWS S3 Pricing**

---

7. Benefits of Consolidated Billing

- **Single invoice** for all accounts.
- **Easier tracking** and managing of costs.
- **Shared discounts** across all accounts.
- Default account limit: **4 accounts per org** (contact AWS to increase).

AWS Budget

- **Purpose**: Plan usage, cost, and instance reservations.
- **Update Frequency**: Updated 3 times/day.
- **Features**:
    - Create **custom alerts**.
    - Alerts notify when usage exceeds forecast.
- **Overview**:
    - Compare actual vs. budgeted usage.
    - Compare across various dimensions.

AWS Cost Explorer

- **Tool**: Visualize, understand & manage AWS costs.
- **Used for**:
    - Creating reports.
    - Analyzing expenses.
- **Dashboard Features**:
    - Web-based dashboard.
    - Filter by period, service, usage, etc.
- **Benefits**:
    - Understand cost drivers.
    - Gain insights for **cost optimization**.

AWS Support Plans

- **4 Plans**:
    - **Basic** (Free)
    - **Developer** (Low cost)
    - **Business** (Mid-tier)
    - **Enterprise** (High-end)

Basic Support

- Default, free option.
- Access to whitepapers, documentation, communities.
- Limited contact with AWS.

Developer Support

- Includes **Basic** +
    - Best practices.
    - Client-side tools.
    - Architecture support.

Business Support

- Includes **Developer** +

- Use-case guidance.
- Full Trusted Advisor checks.
- Limited 3rd-party software support.

Enterprise Support

- Includes **Business** +
  - Application architecture guidance.
  - Project assessments.
  - **Technical Account Manager (TAM)**

Technical Account Manager (TAM)

- Primary AWS contact.
- Helps in design, scaling, and architecture.
- Has access to AWS-wide expertise.

AWS Marketplace

- **Purpose**: Discover, test, and buy software running on AWS.
- **Features**:
  - Detailed product info: pricing, support, reviews.
  - Vendors list their software.
- **Categories**:
  - Business Applications
  - Data & Analytics
  - DevOps
  - Infrastructure Software
  - Internet of Things (IoT)
  - Machine Learning
  - Migration
  - Security

Cloud Migration & Innovation

What You'll Learn:

- AWS Cloud migration strategy.
- AWS Cloud Adoption Framework (CAF).
- Migration benefits and key factors.
- AWS innovative solutions overview.

AWS CAF (Cloud Adoption Framework)

- **Purpose**: Guide for migrating applications to AWS cloud.
- **6 Perspectives**:
  1. **Business**
  2. **People**
  3. **Governance**
  4. **Platform**
  5. **Security**
  6. **Operations**

Business Perspective

- Align IT & business investment.
- Roles: Budget Owners, Finance/Business Managers, Strategists.

People Perspective

- Evaluate cloud skills & roles.
- Roles: HR, Managers, Staffing teams.

Governance Perspective

- Minimize risks & ensure process readiness.
- Roles: CIO, Architects, Analysts, Managers.

Platform Perspective

- Deploy & migrate solutions to cloud.
- Roles: CTO, Solutions Architects, IT Managers.

Security Perspective

- Ensure **security objectives**: visibility, control, auditability.
- Roles: CISO, Security Analysts, IT Security Teams.

Operations Perspective

- Ensure smooth cloud operations.
- Roles: Ops Managers, Support Managers.
- Focus: day-to-day, quarter-to-quarter, year-to-year planning.

Cloud Migration Strategies

What Are Migration Strategies?

Migration strategies are **plans** that help you move your **applications into the cloud** efficiently and securely.

Six Common Cloud Migration Strategies (The 6 R's)

---

1. Rehosting (Lift-and-Shift)

- Move applications to the cloud **without any changes**.
- Quick migration approach.
- Suitable for large-scale migrations with tight timelines.

---

2. Replatforming (Lift, Tinker, and Shift)

- Move applications with **minimal optimizations** for the cloud.
- No core architecture changes.
- Improves performance and compatibility with cloud environments.

---

3. Refactoring (Re-architecting)

- **Change the application's core architecture** and environment.
- Ideal for applications needing **scalability**, **performance improvements**, or **cloud-native features**.

---

4. Repurchasing

- Shift from traditional applications to **SaaS (Software-as-a-Service)** solutions.
- Example: Moving from a custom CRM to Salesforce.

---

5. Retaining

- Keep **critical business applications** in their current environment.
- Often used when apps require refactoring before migrating or due to dependencies.

---

6. Retiring

- **Decommission or remove** obsolete or unnecessary applications.
- Helps reduce cost and complexity.

AWS Snow Family

What is AWS Snow Family?

- A group of **physical devices** used to **transport data into or out of AWS**.
- Used when transferring **large datasets** (up to **exabytes**).
- One **exabyte** = 1,000,000,000,000 MB.

Devices in AWS Snow Family:

- **AWS Snowcone**
- **AWS Snowball**
- **AWS Snowmobile**

AWS Snowcone

- **Smallest** and **secure** device in the Snow Family.
- Specs:
  - 8 TB of storage
  - 4 GB of memory
  - 2 CPUs
- Used for **small data transfers** or **edge computing**.

AWS Snowball

- Comes in two variants:

Type	Use Case	Storage	Memory & CPU
Storage Optimized	Large-scale data migrations	80 TB HDD + 1 TB SSD (block volume)	–
Compute Optimized	High compute resource requirements	42 TB HDD + 7.68 TB NVMe SSD	208 GiB memory, 52 vCPUs

---

AWS Snowmobile

- **Truck-sized device** for **massive data migration**.
- Transfers up to **100 petabytes** of data.
- One **petabyte** = 1,000,000,000 MB.
- Ideal for **extremely large-scale data transfers**.

Innovation with AWS Cloud

AWS Services help you to:

- Evaluate your current business state.
- Determine your target state.
- Solve existing business problems.

AWS Offers Solutions Through:

- Machine Learning (ML)
- Artificial Intelligence (AI)
- Serverless Applications

Machine Learning (ML)

- AWS provides **Amazon SageMaker** for ML development.
- SageMaker reduces development time and complexity.
- With ML, you can:
  - Predict future outcomes.
  - Solve complex problems.
  - Analyze large datasets.

Artificial Intelligence (AI)

- AI is software capable of performing complex human tasks.
- AWS AI-powered services include:
  - **Amazon Lex** – Voice and text chatbots.
  - **Amazon Transcribe** – Converts speech to text.
  - **Amazon Comprehend** – Detects patterns in text.
  - **Amazon Fraud Detector** – Identifies fraudulent activity.

Serverless Applications

- **AWS Lambda** is used for running serverless applications.
- AWS manages the server backend.
- Benefits:
  - Focus more on development.
  - Less overhead for administration.

AWS Cloud Journey

What You'll Learn

- AWS Well-Architected Framework overview.
- Benefits of cloud computing.

AWS Well-Architected Framework

- A tool that uses best practices to improve cloud-based applications.
- Focuses on five pillars:
  1. Operational Excellence
  2. Security

3. Reliability
4. Performance Efficiency
5. Cost Optimization

Pillars of the AWS Well-Architected Framework

1. Operational Excellence

- Ability to manage and monitor systems efficiently.
- Key aspects:
  - Small, reversible changes.
  - System disruption predictions.
  - Automated code tasks.
  - Clear documentation.

2. Security

- Protects systems and data at all levels.
- Covers:
  - Data at rest and in transit.
  - Automated application of best security practices.

3. Reliability

- Ensures minimum disruption to systems.
- Supports:
  - Automatic system recovery.
  - On-demand resource allocation.
  - Improved availability.

4. Performance Efficiency

- Efficient use of computing resources.
- Provides scalability as per demand.

5. Cost Optimization

- Operates cloud services at the lowest cost.
- Activities include:
  - Cost analysis.
  - Use of managed services.
  - Pay-as-you-go billing.

Benefits of the AWS Cloud

What Are the Benefits of the AWS Cloud?

There are **six key benefits** of using the AWS Cloud:

---

1. Trade Upfront Expense for Variable Expense

- Pay **only for what you use**.
- Avoid large upfront investments in infrastructure (like servers or data centers).

---

2. Benefit from Massive Economies of Scale

- AWS serves millions of customers.
- This large scale allows **lower variable costs** and **reduced pay-as-you-go rates**.

---

3. Stop Guessing Capacity

- No need to estimate capacity in advance.
- Pay only for the resources you actually consume.
- Helps **reduce capacity-related costs**.

---

4. Increase Speed and Agility

- Deploy applications **quickly and easily**.
- Faster experimentation and innovation.

---

5. Stop Spending Money Running and Maintaining Data Centers

- AWS manages the infrastructure for you.
- You can focus more on **application development** and **customer experience**.

---

6. Go Global in Minutes

- Easily deploy applications to **multiple regions worldwide**.
- **Low latency** and fast access for customers across the globe.

AWS Cloud Practitioner Exam Preparation Summary

W3Schools offers a **free AWS Cloud Practitioner Tutorial**, which serves as a **comprehensive study guide** for the **AWS Certified Cloud Practitioner exam**. This resource is accessible to everyone, whether or not you plan to take the actual exam. Using this tutorial can **boost your preparation and improve your chances of passing**.

Exam Topics

The exam is divided into **four key subject areas**:

1. **Cloud Concepts**
2. **Security and Compliance**
3. **Technology**
4. **Billing and Pricing**

For a more detailed breakdown, it's advised to **review the official AWS Exam Guide**.

Weight of Each Subject

Here's an approximate distribution:

Subject	Estimated Weight
Cloud Concepts	26%
Security and Compliance	25%
Technology	33%
Billing and Pricing	16%
Total	**100%**

Note: Some exam questions may cover more than one topic area.

Recommended Background

Although not mandatory, it's suggested that candidates have a **basic understanding of AWS Cloud and IT services**. Ideally, you should have around **six months of hands-on experience** using AWS Cloud. However, with thorough preparation, you can take the exam earlier.

Exam Format and Passing Criteria

- **Number of Questions:** 65 (Multiple Choice)
- **Time Limit:** 90 minutes
- **Passing Score:** 70% or higher
- **Tip:** Always answer every question—even if you're unsure—since blank responses are counted as incorrect.
- You can **review and change answers** before submitting.
- Your **score and pass/fail result** are shown immediately after completing the exam.