The Complete AWS Services Guide

A Comprehensive Reference for Enterprise Cloud Architecture

Document Version: 1.0
Last Updated: November 29, 2025
Audience: AWS Beginners to Intermediate Practitioners

Overview

Amazon Web Services (AWS) is a comprehensive cloud computing platform with 300+ services. However, to become an effective AWS practitioner, you only need to master approximately 50 core services. This guide explains the essential services using a practical e-commerce application architecture.

Part 1: Domain and Traffic Management

Amazon Route 53 - Domain and DNS Management

What Is It?

Amazon Route 53 is AWS's fully managed Domain Name System (DNS) and traffic management service. It serves as the internet entry point for applications by translating human-readable domain names into IP addresses.

Definition: DNS (Domain Name System)

DNS is a hierarchical naming system that converts domain names (like amazon.com) into Internet Protocol (IP) addresses that computers understand. Without DNS, users would need to remember numerical IP addresses instead of domain names.

Example:

- User types: amazon.com
- DNS resolves to: 205.251.242.103
- User's browser connects to that IP address

Key Features

Domain Registration

- Register and manage domain names directly through AWS
- Transfer existing domains to AWS

Health Checks

- Continuously monitor endpoints
- Ensure only healthy resources receive traffic
- Automatic failover if primary endpoint fails

Traffic Routing Policies

- **Geographic Routing**: Route North American users to US servers, European users to EU servers
- **Latency-Based Routing**: Always route to lowest-latency endpoint
- **Failover Routing**: Automatic failover to backup infrastructure
- **Weighted Routing**: Distribute traffic using percentage weights (10% new version, 90% old)
- **Multi-Value Answer**: Return multiple IP addresses for client-side load balancing

DNSSEC (Domain Name System Security Extensions)

DNSSEC protects DNS records from spoofing attacks:

- Adds cryptographic signatures to DNS responses
- Prevents attackers from redirecting traffic to malicious sites
- Ensures user visits the legitimate website

Real-World Use Case

E-commerce platform (like Amazon.com) with global traffic:

1. Route 53 registers domain amazon.com
2. Performs continuous health checks on servers
3. Routes North American users to us-east-1 data center
4. Routes European users to eu-west-1 data center
5. Routes Asian users to ap-northeast-1 data center
6. If primary data center fails, automatically routes to backup

Pricing Model

Component	Cost
Hosted Zone	$0.50 per month
DNS Queries	$0.40 per million queries
Health Checks	$0.50 per health check per month

Table 1: Table 1: Route 53 Pricing

Part 2: Static Content and Content Delivery

Amazon S3 - Object Storage

What Is It?

Amazon S3 is a massively scalable object storage service designed to store and retrieve any amount of data from anywhere on the web.

Object Storage vs Block Storage

Aspect	Object Storage (S3)	Block Storage (EBS)
Use Case	Files, images, videos	Databases, OS
Access Pattern	HTTP retrieval	Random read/write
Scalability	Unlimited	Fixed size
Sharing	Easily shared	Single instance

Table 2: Table 2: Object vs Block Storage

Key Features

- **Universal Access**: Store and retrieve via HTTP/HTTPS from anywhere
- **Auto-Scaling**: Automatically scales to any volume
- **Event Triggers**: Invoke Lambda functions when objects uploaded
- **Versioning**: Maintain multiple versions for recovery
- **Encryption**: Server-side encryption for data protection
- **Lifecycle Policies**: Auto-archive or delete old objects
- **Access Control**: Fine-grained permissions per object

Real-World E-Commerce Example

Website needs to serve:

- Product images (high-resolution photos)
- CSS stylesheets for website styling
- JavaScript files for functionality
- User-uploaded product reviews with images

All stored in S3 buckets and served to users.

S3 Storage Classes

Storage Class	Cost/GB/Month	Use Case
S3 Standard	$0.023	Frequently accessed
S3 Intelligent-Tiering	$0.0125	Variable access patterns
S3 Standard-IA	$0.0125	Infrequent access
S3 Glacier	$0.004	Archive/backup

Table 3: Table 3: S3 Storage Class Pricing

Amazon CloudFront - Content Delivery Network

What Is It?

Amazon CloudFront is a Content Delivery Network (CDN) that caches content globally across 500+ edge locations worldwide.

Definition: CDN (Content Delivery Network)

A CDN is a distributed network of servers strategically located worldwide. Instead of all users downloading from a single origin server, the CDN caches copies near users:

- Reduces latency dramatically
- Improves page load speed
- Reduces origin server load

The Problem CloudFront Solves

Without CloudFront:

- S3 bucket in Virginia (us-east-1)
- User in Tokyo downloads 5MB image
- Latency: 500-1000ms
- Download time: 5-10 seconds

With CloudFront:

- S3 bucket in Virginia (origin)
- CloudFront caches content in Tokyo edge
- User in Tokyo downloads from nearby edge location
- Latency: 50-100ms
- Download time: 0.5-1 second (10x faster!)

How CloudFront Works

1. User requests content from CloudFront URL
2. Request routed to nearest edge location
3. If content in cache and fresh, return immediately
4. If cache miss, CloudFront fetches from origin (S3)
5. CloudFront caches content
6. Future requests served from cache

CloudFront Key Features

- **500+ Edge Locations**: Worldwide presence ensures low latency
- **Cache Behaviors**: Define caching rules for URL patterns
- **Custom Origins**: Cache from EC2, ELB, or non-AWS servers
- **SSL/TLS Termination**: Encrypt user-to-edge traffic
- **DDoS Protection**: Integrated AWS Shield protection
- **Geo-Restriction**: Block/allow by country
- **Lambda@Edge**: Run functions at edge locations

Performance Improvement

Metric	Without CloudFront	With CloudFront
Average Latency	500-1000ms	50-100ms
Time to First Byte	800-1500ms	100-200ms
Page Load Time	5-10 seconds	1-2 seconds

Table 4: Table 4: CloudFront Performance Impact

Pricing

- **Data Transfer Out**: $0.085/GB (US/EU)
- **HTTP Requests**: $0.0075 per 10,000 requests
- **HTTPS Requests**: $0.01 per 10,000 requests

Part 3: API and Load Balancing

Elastic Load Balancer (ELB)

What Is It?

Elastic Load Balancer automatically distributes incoming application traffic across multiple targets to ensure no single server becomes overloaded.

Definition: Load Balancing

Load balancing distributes network traffic across multiple servers to:

- Prevent bottlenecks on single servers
- Ensure high availability (if server fails, others handle requests)
- Enable horizontal scaling (add servers as demand increases)

Three Types of Load Balancers

1. **Application Load Balancer (ALB)**: HTTP/HTTPS, microservices (Layer 7)
2. **Network Load Balancer (NLB)**: Ultra-high performance, non-HTTP (Layer 4)
3. **Classic Load Balancer**: Legacy applications (being phased out)

Key Features

- **Auto-Scaling Groups**: Automatically add/remove instances based on demand
- **Health Checks**: Remove unhealthy instances from rotation
- **Sticky Sessions**: Route user's subsequent requests to same target
- **Cross-Zone**: Distribute traffic across multiple availability zones
- **Path-Based Routing**: /images/* to image server, /api/* to API server

Real-World Black Friday Example

E-commerce site receives 10,000 requests/second during Black Friday:

- Each server handles: 1,000 requests/second
- Without load balancer: Server crashes after 1 second
- With load balancer + 10 servers: 10,000 requests/second evenly distributed (1,000 each)
- Result: Website stays online

Load Balancing Algorithms

Algorithm	Description
Round Robin	Cycles through servers sequentially
Least Outstanding Requests	Sends to server with fewest pending requests
IP Hash	Routes same client IP always to same server

Table 5: Table 5: Load Balancing Algorithms

AWS API Gateway

What Is It?

API Gateway is a fully managed service to create, publish, maintain, monitor, and secure APIs at scale.

Definition: API (Application Programming Interface)

An API is a set of protocols for building applications. REST APIs use HTTP verbs:

- GET: Retrieve data
- POST: Create data
- PUT: Update data
- DELETE: Remove data

Key Features

- **RESTful APIs**: Traditional request-response APIs
- **WebSocket APIs**: Bi-directional real-time communication
- **Rate Limiting**: Limit to 100 requests/minute per user
- **Authentication**: OAuth, IAM, Cognito integration
- **Caching**: Cache responses to reduce load
- **Request/Response Transformation**: Modify requests/responses
- **CORS**: Enable cross-origin requests

Advantage Over Load Balancer

API Gateway can route directly to AWS services without compute instances:

Instead of: API → Load Balancer → EC2 → DynamoDB

You can do: API → DynamoDB directly

Cost Comparison

Component	With ELB	With API Gateway
API Gateway	No	$3.50 per million requests
EC2 Instances (10×)	$0.0116/hour each	Not needed
Total for 100M requests/month	Higher	Lower

Table 6: Table 6: Cost: ELB vs API Gateway

Part 4: Security and Authentication

AWS Certificate Manager

What Is It?

AWS Certificate Manager provisions, manages, and deploys SSL/TLS certificates for use with AWS services.

Definition: SSL/TLS

- **SSL (Secure Sockets Layer)**: Outdated encryption protocol
- **TLS (Transport Layer Security)**: Modern replacement

SSL/TLS encrypts data in transit so attackers cannot intercept and read sensitive information.

How SSL/TLS Protects Data

Without SSL:

- User logs in with password "SuperSecret123"
- Traffic unencrypted
- Attacker intercepts: reads password immediately
- Attacker can steal account

With SSL:

- User logs in with password "SuperSecret123"

- Traffic encrypted with cryptographic key
- Attacker intercepts: sees gibberish encryption
- Attacker cannot decrypt without key
- Password stays secret

SSL Handshake Process

1. User visits https://example.com
2. Browser requests certificate from server
3. Server sends SSL certificate with public key
4. Browser verifies certificate is valid
5. Browser and server establish encrypted connection
6. All subsequent traffic encrypted

Certificate Manager Features

- **Free Certificates**: AWS provides certificates for free
- **Automatic Renewal**: Renews before expiration automatically
- **Wildcard Certificates**: One cert covers *.example.com
- **AWS Service Integration**: Works with CloudFront, ELB, API Gateway

Real-World E-Commerce Example

Online store must protect payment information:

- Certificate Manager provisions certificate for shop.example.com
- Automatic renewal every 90 days
- CloudFront uses certificate to encrypt user-to-edge traffic
- User sees "secure" padlock icon
- Customer payment information encrypted in transit

AWS WAF (Web Application Firewall)

What Is It?

AWS WAF protects web applications from common attacks like SQL injection, cross-site scripting, and DDoS.

Definition: SQL Injection

SQL injection is a cyber attack where malicious SQL code injected into input fields:

Example attack:
Login form username field: admin' OR '1'='1

This manipulates SQL query to return unauthorized data or modify database.

Legitimate query:
SELECT * FROM users WHERE username = 'admin'

Malicious query:
SELECT * FROM users WHERE username = 'admin' OR '1'='1'

The `OR '1'='1'` part is always true, so query returns ALL users without checking password.

Definition: XSS (Cross-Site Scripting)

XSS injects malicious JavaScript into web pages. When users visit, malicious script runs:

- Steals session cookies
- Redirects to phishing site
- Captures keystrokes
- Defaces website

WAF Features

- **IP Reputation**: Block known malicious IPs
- **Bot Control**: Detect and block bot traffic
- **SQL Injection Protection**: Detect malicious SQL patterns
- **XSS Protection**: Detect JavaScript injection attempts
- **Rate Limiting**: Block IPs exceeding request thresholds
- **Geo-Blocking**: Block traffic from specific countries
- **Custom Rules**: Create application-specific rules

Real-World Attack Prevention

Attacker attempts SQL injection:

1. Enters SQL injection code in login form
2. WAF detects malicious pattern
3. Request blocked
4. Attack logged for investigation
5. Database remains safe

Pricing

- **Web ACL**: $5.00 per month
- **Rules**: $1.00 per rule per month
- **Requests**: $0.60 per million requests

AWS Shield - DDoS Protection

What Is It?

AWS Shield is a DDoS (Distributed Denial of Service) protection service with free and paid tiers.

Definition: DDoS Attack

DDoS attack overwhelms website with traffic from many sources simultaneously:

Normal traffic: 1,000 requests/second
DDoS attack: 1,000,000 requests/second from 100,000 attacking computers
Result: Servers crash, legitimate users cannot access website

How DDoS Attack Works

1. Attacker compromises 100,000 computers (botnet)
2. Attacker instructs all to visit target.com simultaneously
3. 100,000 computers each send 1,000 requests/second
4. Website receives 100 million requests/second
5. Servers overwhelmed, crash
6. Legitimate users see "website down" error

Two Tiers

Feature	Standard (Free)	Advanced ($3,000/month)
DDoS Protection	Basic	Comprehensive
24/7 Support	No	Yes
Cost Protection	No	AWS refunds costs
Attack Notifications	No	Yes (real-time)
Attack Analytics	No	Detailed reports

Table 7: Table 7: AWS Shield Tiers

When to Use Advanced

Standard (Free): Small apps, can tolerate downtime
Advanced: Mission-critical e-commerce, banking, healthcare

Amazon Cognito - User Authentication

What Is It?

Amazon Cognito manages user sign-up, sign-in, and access control for applications.

Definition: Authentication vs Authorization

- **Authentication**: Who are you? (username + password)
- **Authorization**: What can you do? (permissions)

Cognito Workflow

1. User visits application
2. Application redirects to Cognito sign-in page
3. User enters email and password
4. Cognito verifies credentials against user pool database
5. If valid, Cognito returns authentication token (JWT)
6. User's browser stores token in storage
7. Browser includes token with each API request
8. API Gateway verifies token is valid
9. If valid, request allowed; if invalid, request denied

Key Features

- **User Pools**: Store and manage user credentials
- **Multi-Factor Authentication**: SMS or authenticator app
- **Social Login**: Sign-in with Facebook, Google, Apple
- **Custom Attributes**: Store custom user data
- **User Groups**: Organize users with different permissions
- **Passwordless Auth**: Email links or SMS codes instead of passwords

Real-World E-Commerce Example

Customer authentication flow:

1. Customer visits shop.example.com
2. Redirected to Cognito sign-in page
3. Enters email: customer@example.com and password
4. Cognito verifies credentials
5. Returns JWT token: eyJhbGciOiJIUzI1NiIs...
6. Browser stores token
7. Customer clicks "View Orders" button
8. API call includes token header
9. API Gateway verifies token
10. Customer's orders retrieved and displayed

Authorization: Role-Based Permissions

- **Admin**: Can view all orders, refund orders, manage inventory, view analytics
- **Customer**: Can view own orders only, cannot refund
- **Guest**: Can browse products only, cannot place orders

Pricing

- **Monthly Active Users (MAU)**: First 50,000 free
- **Additional MAU**: $0.015 per MAU
- **Advanced Security**: Additional charges

Part 5: Compute Services

EC2 (Elastic Compute Cloud)

What Is It?

EC2 provides resizable compute capacity in the cloud. It's a virtual server you rent from AWS.

Definition: Virtualization

Virtualization allows one physical server to run multiple isolated virtual machines. Each VM thinks it has its own CPU, memory, and storage but shares physical hardware.

Common Instance Types

Type	Use Case	Cost/hour
t3.micro	Testing, low traffic	$0.0116
t3.small	Small websites	$0.023
m6i.large	Medium applications	$0.077
c6i.xlarge	High CPU workloads	$0.085
r6i.2xlarge	Memory-intensive (databases)	$0.252

Table 8: Table 8: EC2 Instance Types

Your Responsibilities

You must manage:

- Operating system installation and patching
- Security updates and software maintenance
- Performance monitoring and troubleshooting
- Instance scaling (launch/terminate)
- Backup and disaster recovery

Advantages

- Full control over environment
- Highly flexible - install any software
- Cost-effective for sustained workloads
- Auto-scaling groups for handling traffic spikes

Disadvantages

- Management overhead
- Must monitor and patch regularly
- Need to manage auto-scaling rules
- Not ideal for sporadic workloads

Real-World Example: Black Friday

E-commerce needs 5 servers for Black Friday surge:

1. Launch 5 EC2 t3 instances
2. Install web application on each
3. Configure load balancer
4. Monitor during traffic surge
5. After Black Friday, terminate to save costs

AWS Lightsail

What Is It?

Lightsail is a simplified cloud platform perfect for developers new to cloud computing.

Comparison: Lightsail vs EC2

Factor	Lightsail	EC2
Ease of Use	Very simple	Moderate
Templates	Yes (WordPress, apps)	No
Management	Minimal	High
Cost	Fixed $3.50-$160/month	Variable
Scalability	Limited	Unlimited
Best For	Blogs, small sites	Enterprise apps

Table 9: Table 9: Lightsail vs EC2

Real-World Example: Personal Blog

With Lightsail:

- Click "Deploy WordPress"
- 5 minutes later: Live blog
- No server management

With EC2:

- Launch instance
- Install OS
- Install web server
- Install WordPress
- Configure firewall
- Time: hours of setup

Pricing

Fixed monthly pricing:

- **$3.50/month**: 512MB RAM, 20GB SSD
- **$5.00/month**: 1GB RAM, 40GB SSD
- **$10.00/month**: 2GB RAM, 80GB SSD

Elastic Container Service (ECS)

What Is It?

ECS is container orchestration service for running Docker containers in AWS.

Definition: Docker Containers

Docker containers package application code, libraries, and dependencies:

- Runs on developer's laptop
- Runs on test server
- Runs on production server
- **Same behavior everywhere** (no "works on my machine" problems)

Container vs Virtual Machine

Aspect	Virtual Machine	Container
Includes OS	Yes	No
Size	1-2GB+	10-100MB
Startup Time	Minutes	Seconds
Overhead	High	Low
Density	5-10 per server	100s per server

Table 10: Table 10: Container vs VM

ECS Launch Types

EC2 Launch Type:

- Run containers on EC2 instances you manage
- Lower cost for sustained workloads
- You manage EC2 instances

Fargate Launch Type (Serverless):

- AWS manages infrastructure
- Focus only on application code
- Higher cost for low utilization

ECS Components

- **Cluster**: Logical grouping of resources
- **Task**: Running instance of container
- **Service**: Ensures desired number of tasks run
- **Task Definition**: How to run container (image, CPU, memory)

Real-World Microservices Example

1. Product Service: 3 containers running
2. Order Service: 2 containers running
3. User Service: 1 container running
4. ECS ensures if one crashes, replaces it automatically

AWS Lambda - Serverless Compute

What Is It?

Lambda is serverless compute where you write code and AWS manages infrastructure.

Definition: Serverless

Serverless means:

- No servers to manage
- No instances to provision
- No infrastructure to maintain
- Pay only for execution time (not idle time)

How Lambda Works

1. Write function code (Python, Node.js, Java, Go, C#, Ruby)
2. Upload code to Lambda
3. Configure trigger (API, S3, SQS, etc.)
4. When trigger event occurs:
    - AWS provisions compute
    - Loads your code
    - Executes function
    - Returns result
    - Deallocates resources
    - You pay only for execution time

Pricing Model

Charge	Cost
Free Tier	1,000,000 requests/month free
Requests	$0.20 per 1,000,000 requests
Execution Time	$0.0000166667 per GB-second

Table 11: Table 11: Lambda Pricing

Real-World Example: S3 Image Processing

Workflow when user uploads product image:

1. User uploads image to S3
2. S3 automatically triggers Lambda function
3. Lambda function:
    - Downloads image
    - Resizes to create thumbnail
    - Optimizes image quality
    - Uploads thumbnails back to S3
4. Lambda completes and deallocates
5. No servers running when idle
6. You pay only for processing time

Advantages

- No server management
- Automatic scaling (1 to 1 million requests)
- Pay per execution
- Tight AWS service integration
- Perfect for event-driven workloads

Disadvantages

- Cold starts (slight delay on first execution)
- Max 15-minute execution time
- Limited disk space (/tmp 500MB)
- Not ideal for long-running processes

Part 6: Storage and Data

EFS (Elastic File System)

What Is It?

EFS is managed network file system mountable by multiple EC2 instances simultaneously.

EFS vs EBS

Feature	EBS	EFS
Connected to	One EC2 instance	Multiple instances
Type	Block storage	Network file system
Performance	Very high	High
Scaling	Fixed size	Auto-grows
Use Case	OS, databases	Shared files

Table 12: Table 12: EBS vs EFS

Real-World Example: Video Cluster

Video processing cluster needs shared library:

- **Without EFS**: 10 instances × 500GB each = 5TB cost
- **With EFS**: All 10 instances share 500GB = 500GB cost
- **Savings**: 90% reduction

EBS (Elastic Block Store)

What Is It?

EBS provides persistent block storage volumes for EC2 instances (like hard drive).

Volume Types

Type	Technology	Use Case	Max IOPS
gp3	SSD	General purpose	16,000
io1	SSD	High-perf databases	64,000
st1	HDD	Big data/Hadoop	500

Table 13: Table 13: EBS Volume Types

Features

- **Snapshots**: Backups of volumes
- **Encryption**: Encrypt data at rest
- **Auto-Scaling**: Increase size without downtime

Part 7: Secrets and Configuration

AWS Secrets Manager

What It Is?

Secrets Manager securely stores sensitive credentials (API keys, passwords, OAuth tokens).

Why Not Store Secrets in Code

- If committed to GitHub, attackers find it immediately
- Difficult to rotate credentials
- Credentials visible in code history
- Cannot have different secrets per environment
- Risk of accidental exposure in logs

How Secrets Manager Works

1. Store secret (encrypted)
2. Application requests secret by name
3. Application must have IAM permission
4. Secret returned (decrypted)
5. Application uses temporarily in memory
6. Secret never stored in code or logs

Real-World Example

- Production database password stored in Secrets Manager
- Lambda function needs to connect to database
- Lambda requests secret by name: "prod/rds/password"
- Secrets Manager returns password
- Lambda connects to database
- No password hardcoded in function code
- Password can be rotated without changing Lambda

Features

- **Encryption**: KMS encryption at rest
- **Rotation**: Automatically rotate on schedule
- **Audit Logging**: CloudTrail logs all access
- **Versioning**: Keep old versions for rollback

Pricing

- **Per Secret**: $0.40 per secret per month
- **API Calls**: $0.05 per 10,000 calls

AWS AppConfig - Feature Flags

What It Is?

AppConfig manages feature flags and configuration centrally.

Definition: Feature Flags

Feature flags are toggles in code enabling/disabling features without redeploying:

1. Test new features with subset of users
2. Quickly disable problematic features
3. A/B test different versions
4. Gradual rollout of features

Real-World Example

New checkout process being tested:

- Dev: NEW_CHECKOUT_ENABLED = true (test new feature)
- Prod: NEW_CHECKOUT_ENABLED = false (use stable)
- Friday 9pm: Bugs discovered → set flag to false
- Result: Instantly disable without redeploying code

Benefits

- Change behavior without redeploying
- Test with specific users
- Emergency disable if issues found
- Roll back instantly

Part 8: Databases

RDS (Relational Database Service)

What It Is?

RDS is managed service for relational databases. AWS handles installation, backup, and patching.

Supported Engines

- MySQL
- PostgreSQL
- Oracle
- Microsoft SQL Server
- MariaDB

What AWS Manages

- Database installation and configuration
- Security patches and updates
- Automated backups daily
- Multi-AZ replication for high availability
- Monitoring and alerting
- Encryption at rest and in transit

Multi-AZ Deployment

Multi-AZ provides high availability:

1. Primary database in AZ-A (us-east-1a)
2. Standby replica in AZ-B (us-east-1b)
3. Changes replicated synchronously
4. If AZ-A fails, automatic failover to AZ-B
5. Failover typically 60-120 seconds

6. Applications automatically reconnect

Real-World Example

E-commerce database storing customer orders:

- Lightning strikes us-east-1a data center
- Primary database lost
- Automatic failover to standby in us-east-1b
- Customer doesn't notice outage
- No order data lost (continuous replication)

Instance Pricing

Instance Type	Cost/hour	Cost/month
db.t3.micro	$0.017	$12.24
db.t3.small	$0.033	$23.76
db.m5.large	$0.192	$138.24

Table 14: Table 14: RDS Pricing

Aurora - AWS-Optimized Database

What It Is?

Aurora is AWS's proprietary relational database engine built for cloud, compatible with MySQL/PostgreSQL.

Aurora Features

- **Performance**: 3x faster than MySQL
- **Serverless Option**: Auto-scale from zero to thousands
- **Read Replicas**: Create read-only copies in other regions
- **Global Database**: Multi-region database
- **Auto-Scaling Storage**: Grows to 64TB automatically

Aurora Serverless

Traditional database planning:

- Estimate peak users
- Provision for peak
- Pay for peak 24/7
- Cost: High fixed costs during low-traffic times

Aurora Serverless:

- Automatically scales based on demand
- Zero capacity during low traffic
- Scales to thousands during peak
- Pay only for capacity used

Pricing

- **db.r6g.large**: $0.29/hour
- **Storage**: $1 per GB-month
- **Serverless**: $0.06 per ACU-hour

DynamoDB - NoSQL Database

What It Is?

DynamoDB is fully managed NoSQL key-value database for ultra-high performance.

SQL vs NoSQL

Aspect	SQL (RDS)	NoSQL (DynamoDB)
Data Model	Tables/rows	Key-value
Schema	Fixed	Flexible
Queries	Complex SQL	Simple lookups
Scaling	Vertical	Horizontal

Table 15: Table 15: SQL vs NoSQL

Real-World Use Case

Mobile gaming app tracking scores:

- Millions of users
- Millions of score updates per second
- RDS becomes bottleneck
- DynamoDB handles millions of writes/second automatically

Capacity Modes

Mode	Best For	Pricing
On-Demand	Unpredictable	$1.25 per million writes
Provisioned	Predictable	Fixed hourly rate

Table 16: Table 16: DynamoDB Capacity Modes

DocumentDB - MongoDB Compatible

What It Is?

DocumentDB is managed MongoDB-compatible database for flexible JSON documents.

When to Use DocumentDB

- Complex hierarchical data
- Nested arrays and objects
- Flexible schema (different documents, different fields)
- Rich queries on document contents

vs DynamoDB

- **DynamoDB**: Simple key lookups (fast but limited)
- **DocumentDB**: Complex queries on documents (flexible)

Neptune - Graph Database

What It Is?

Neptune is managed graph database for relationship data.

Definition: Graph Database

Graph databases store nodes (entities) and edges (relationships), optimized for relationship queries.

Real-World Use Case: Social Network

Who are friends of my friends?

Traditional SQL: Complex multi-join query

Neptune: Direct graph traversal - instant results

Use Cases

- Social networks (friend connections)
- Recommendation engines (product relationships)
- Fraud detection (transaction patterns)
- Identity management (role hierarchies)

Part 9: Caching

ElastiCache

What It Is?

ElastiCache is managed in-memory data store for caching frequently accessed data.

How Caching Works

First request:

1. Check cache - miss
2. Query database (slow - 200ms)
3. Store in cache
4. Return to user

Second request:

1. Check cache - hit!
2. Return immediately (5ms)
3. 40x faster

Supported Engines

Engine	Features	Use Case
Redis	Rich types, persistence	Session storage
Memcached	Simple key-value, in-memory	Query caching

Table 17: Table 17: Caching Engines

Problem: Data Loss

If cache crashes, all data lost. Application must:

1. Detect miss
2. Query database
3. Reload cache
4. "Thundering herd" - many simultaneous database requests

MemoryDB

What It Is?

MemoryDB is Redis-compatible database with persistent storage, solving ElastiCache data loss problem.

Key Difference

Aspect	ElastiCache	MemoryDB
Persistence	No (lost on crash)	Yes (survives crashes)
Recovery	Minutes	Seconds
Cost	Lower	Higher
Use Case	Cache layer	Primary data store

Table 18: Table 18: ElastiCache vs MemoryDB

Real-World Example

E-commerce shopping carts:

- **ElastiCache**: Cache crashes = customers lose carts (bad!)
- **MemoryDB**: Carts restored automatically from persistent storage

---

Part 10: Artificial Intelligence

Amazon Bedrock

What It Is?

Bedrock is managed service accessing foundation models through single API.

Definition: Foundation Models

Pre-trained AI models (billions of parameters) that can:

- Generate text
- Summarize documents
- Translate languages
- Generate images
- Answer questions

Available Models

- Claude (Anthropic)
- Llama (Meta)
- Mistral
- DALL-E (images)
- Titan (Amazon)

Real-World Use Cases

1. **Customer Service Chatbot**: Answer common questions automatically
2. **Document Processing**: Extract information from PDFs
3. **Content Generation**: Generate product descriptions
4. **Image Generation**: Create marketing images
5. **Code Generation**: Generate code snippets

Pricing

- **Claude Input**: $0.003 per 1,000 tokens
- **Claude Output**: $0.015 per 1,000 tokens
- **Llama**: Lower cost alternatives

SageMaker - Machine Learning Platform

What It Is?

SageMaker is managed ML platform for building, training, deploying models.

ML Workflow

1. Prepare data (SageMaker Processing)
2. Label data (SageMaker Ground Truth)
3. Train model (SageMaker Training)
4. Evaluate model (SageMaker Studio)
5. Deploy model (SageMaker Hosting)
6. Monitor performance (CloudWatch)

Real-World Example

Predict customer churn:

1. Collect historical customer data
2. Label who churned vs retained
3. Train ML model
4. Deploy as API
5. For new customers, predict churn probability
6. Offer retention incentives to at-risk customers

Part 11: Messaging and Coordination

SNS (Simple Notification Service)

What It Is?

SNS is managed pub/sub messaging service broadcasting messages to multiple subscribers.

Definition: Pub/Sub

Pub/sub decouples publishers from subscribers:

- **Publisher**: Sends message (doesn't know receivers)
- **Subscriber**: Receives messages it subscribes to
- **Topic**: Central message hub

Real-World E-Commerce Example

Order placement triggering multiple actions:

Traditional (tightly coupled):
Order placed → Update inventory → Create shipment → Log analytics → Send email
If any step fails, entire order fails

Pub/Sub (loosely coupled):
Order placed → SNS topic "OrderPlaced"
├─ Inventory service receives → updates stock
├─ Shipping service receives → creates shipment
├─ Analytics service receives → logs event
└─ Email service receives → sends confirmation
Services fail independently without affecting others

SNS Features

- **Message Filtering**: Receive only matching messages
- **Fan-Out**: One topic, many subscribers
- **Email**: Direct email notifications
- **SMS**: Send text messages
- **Lambda**: Trigger functions
- **Message Attributes**: Add metadata

Pricing

- **Published Messages**: $0.50 per million
- **Subscriptions**: $1 per million HTTP subs
- **Email**: $2 per 100,000 emails

SQS (Simple Queue Service)

What It Is?

SQS is managed message queue for asynchronous processing.

Queue vs Topic

SQS Queue	SNS Topic
Processed one at a time	Broadcast to many
Pull model (consumer pulls)	Push model (pushed to subscribers)
Messages deleted after processing	Same message to all
Good for jobs	Good for notifications

Table 19: Table 19: SQS vs SNS

Queue Types

Standard Queue:

- Best effort ordering
- At-least-once delivery

FIFO Queue:

- Guaranteed ordering
- Exactly once

Real-World Example

E-commerce processing hundreds of orders/second:

1. Order service receives order
2. Places order message in SQS queue
3. Multiple order processors pull messages
4. Each processes order independently
5. If processor crashes, message re-queued
6. No lost orders even if processor fails

Features

- **Visibility Timeout**: Message hidden while processing (30s default)
- **Dead Letter Queues**: Failed messages moved here
- **Message Deduplication**: Prevent duplicates (FIFO)
- **Message Grouping**: FIFO ordering per group (FIFO)
- **Batching**: Send/receive 10 messages per call

Pricing

- **Requests**: $0.40 per million
- **FIFO**: $0.50 per million

Part 12: Key Terminology Reference

Security Terms

SSL (Secure Sockets Layer)

Cryptographic protocol providing secure communication. Now superseded by TLS but term still commonly used.

TLS (Transport Layer Security)

Modern replacement for SSL providing:

- Encryption (confidentiality)
- Authentication (verify identity)
- Integrity (detect tampering)

DNSSEC (Domain Name System Security Extensions)

Cryptographic security for DNS:

- Adds digital signatures to DNS responses
- Prevents DNS spoofing
- Ensures user visits legitimate site

DDoS (Distributed Denial of Service)

Cyber attack using many sources overwhelming target with traffic, making service unavailable.

Infrastructure Terms

IAM (Identity and Access Management)

AWS service managing users, groups, and permissions controlling who can do what.

VPC (Virtual Private Cloud)

Logically isolated network within AWS providing:

- Network isolation
- Firewall rules (Security Groups)
- Private IP address ranges
- VPN connectivity

Availability Zone (AZ)

Isolated physical data center within AWS region providing:

- High availability
- Disaster recovery
- Fault tolerance

Region

Geographic area containing multiple AZs (e.g., us-east-1 has 6 AZs)

Data Terms

IOPS (Input/Output Per Second)

Number of read/write operations per second. Higher IOPS = faster database.

Throughput

Amount of data transferred per second (measured in Mbps or GBps).

Latency

Time delay between request and response (measured in milliseconds).

Conclusion

This comprehensive guide covers 25+ essential AWS services for building modern cloud applications. While AWS offers 300+ services, mastering these core services enables you to:

- Build scalable web applications
- Process data at scale
- Implement AI/ML solutions
- Secure infrastructure
- Achieve high availability
- Optimize costs

Recommended Next Steps

1. Choose a service (Lambda, RDS, or DynamoDB)
2. Complete AWS tutorials and hands-on labs
3. Build a small project (e.g., serverless API)
4. Explore pricing and cost optimization
5. Study for AWS certification

Free Resources

- AWS Free Tier: 12 months free for new accounts
- AWS Tutorials: https://aws.amazon.com/getting-started/
- AWS Certification: Solutions Architect, Developer, DevOps paths

---

References

[1] "AWS Explained: The Most Important AWS Services To Know". Be A Better Dev, November 3, 2025. https://www.youtube.com/watch?v=OGYEXGy8ca4

[2] Amazon Web Services (AWS). AWS Official Documentation. https://aws.amazon.com/documentation/

[3] AWS Training and Certification. AWS Fundamentals and Best Practices. https://aws.amazon.com/training/

Document Version: 1.0
Last Updated: November 29, 2025
Total Services Covered: 25+
Estimated Reading Time: 2-3 hours

This comprehensive guide provides foundational knowledge of AWS services. For production implementations, consult official AWS documentation and security best practices.