

Ejercicio de investigación

Python

1.- Defina una lista y una tupla en Python. Proporcione algunos ejemplos.

Una Lista es una colección ordenada y modificable (mutable) de elementos. Los elementos dentro de una lista pueden ser de diferentes tipos de datos (enteros, cadenas de texto, otros objetos, etc.).

Python

Creación de una Lista

```
frutas = ["manzana", "banana", "cereza"]
```

```
print(f"Lista inicial: {frutas}")
```

1. Mutabilidad: Añadir un elemento

```
frutas.append("naranja")
```

```
print(f"Lista después de añadir: {frutas}")
```

2. Mutabilidad: Modificar un elemento (el índice 1 es "banana")

```
frutas[1] = "uva"
```

```
print(f"Lista después de modificar: {frutas}")
```

Una Tupla es una colección ordenada e inmodificable (immutable) de elementos. Al igual que las listas, puede contener diferentes tipos de datos.

Creación de una Tupla

```
coordenadas = (10.5, 25.3) print(f"Tupla inicial: {coordenadas}")
```

```
# Intentar modificar generaría un error (TypeError) # coordenadas[0] = 12.0
```

```
# ¡Esto causaría un error!
```

Ejemplo de desempaquetado de tuplas

```
latitud, longitud = coordenadas print(f"Latitud: {latitud}, Longitud: {longitud}")
```

2.- ¿Qué es un espacio de nombres en Python?

Un espacio de nombres (o namespace) en Python es un sistema que garantiza que todos los nombres (variables, funciones, clases, módulos, etc.) en un programa son únicos y pueden ser identificados sin ambigüedad.

3. ¿Cuál es la diferencia entre una variable local y una variable global?

La diferencia principal entre una variable local y una global en Python (o en la mayoría de los lenguajes de programación) radica en su alcance (scope), es decir, dónde pueden ser accedidas y modificadas dentro de un programa.

4. ¿Qué es un IDE ? Mencione algunos IDE comunes que podrían utilizarse con Python.

Un IDE significa Entorno de Desarrollo Integrado (Integrated Development Environment). Es una aplicación de software que proporciona un conjunto integral de herramientas y funcionalidades para ayudar a los programadores a escribir, compilar/interpretar, depurar y ejecutar código de manera eficiente y centralizada. En esencia, es la herramienta principal que un desarrollador utiliza para trabajar.

PyCharm (JetBrains)

Visual Studio Code

Spyder

5. ¿Qué son los módulos en Python? Proporcione algunos ejemplos.

Un módulo en Python es simplemente un archivo que contiene código Python. El archivo debe tener la extensión .py.

Un módulo puede definir funciones, clases y variables. Su objetivo principal es agrupar código relacionado de una manera lógica y organizada, permitiendo la reutilización y evitando que un único archivo de programa se vuelva demasiado grande y difícil de manejar.

Archivo: saludos.py

ejemplos

Archivo: saludos.py

```
VERSION = "1.0"
```

```
def saludar_formal(nombre):
```

```
    """Retorna un saludo formal."""
```

```
    return f"Estimado/a {nombre}, bienvenido/a."
```

```
def saludar_informal(nombre):
```

```
    """Retorna un saludo informal."""
```

```
    return f"Hola {nombre}, ¿cómo estás?"
```

ejemplo 2

Archivo: main.py

Importamos el módulo

```
import saludos
```

Usamos las funciones

```
saludo1 = saludos.saludar_formal("Ana")
```

```
saludo2 = saludos.saludar_informal("Pedro")
```

Usamos la variable

```
version_modulo = saludos.VERSION
```

```
print(saludo1)
```

```
print(saludo2)
```

```
print(f"Usando la versión: {version_modulo}")
```

6. ¿Cuál es la diferencia entre una matriz y una lista?

La diferencia entre una matriz y una lista en Python se centra principalmente en su estructura, su tipo de datos y las operaciones que se pueden realizar sobre ellas de manera eficiente.

En el contexto de Python, cuando hablamos de "matriz" generalmente nos referimos a un objeto del paquete NumPy (numpy.array), que es la forma estándar de manejar datos numéricos multidimensionales.

7. ¿Qué son los operadores? Proporcione algunos ejemplos.

Los operadores son símbolos especiales que realizan una operación (o acción) en uno o más valores. Estos valores sobre los que actúa el operador se llaman operandos.

| Operador | Nombre | Ejemplo | Resultado |
|----------|--------|---------|-----------|
|----------|--------|---------|-----------|

| | | | |
|---|------|-------|---|
| + | Suma | 5 + 3 | 8 |
|---|------|-------|---|

| | | | |
|---|-------|--------|---|
| - | Resta | 10 - 4 | 6 |
|---|-------|--------|---|

| | | | |
|---|----------------|-------|----|
| * | Multiplicación | 4 * 6 | 24 |
|---|----------------|-------|----|

| | | | |
|---|----------|--------|---------------------|
| / | División | 10 / 3 | 3.333... (Flotante) |
|---|----------|--------|---------------------|

Bases de datos

1.- ¿Cuál es la diferencia entre una base de datos relacional y una no relacional ?

Bases de Datos Relacionales vs. No Relacionales (NoSQL)

Característica

Base de Datos Relacional (SQL)

Base de Datos No Relacional (NoSQL)

Estructura

Estructura fija (esquema rígido). Los datos se almacenan en tablas con filas y columnas predefinidas.

Estructura dinámica (esquema flexible). Los datos se almacenan de diversas formas (documentos, clave-valor, grafos).

Relaciones

Fuertes y definidas (a través de claves).

Débiles o inexistentes.

Escalabilidad

Escalabilidad vertical (aumentar capacidad del servidor).

Escalabilidad horizontal (añadir más servidores).

Ejemplos

MySQL, PostgreSQL, SQL Server, Oracle.

MongoDB (Documentos), Redis (Clave-Valor), Neo4j (Grafos).

2. ¿Qué son los Índices?

Un índice es una estructura de datos especial asociada a una tabla que mejora drásticamente la velocidad de la recuperación de datos (consultas SELECT). Es similar al índice alfabético de un libro:

Permite que el motor de la base de datos encuentre filas sin tener que examinar cada fila de la tabla (lo que se conoce como exploración de tabla o table scan).

Se crea sobre una o más columnas de una tabla.

Aunque acelera las lecturas, ralentiza ligeramente las operaciones de escritura (INSERT, UPDATE, DELETE), ya que el índice también debe ser actualizado.

3.- Claves Principales y Secundarias (Foráneas)

Clave Principal (Primary Key):

Es una columna o conjunto de columnas que identifica de forma única cada registro (fila) en una tabla.

Debe contener valores únicos y no puede contener valores nulos (NULL).

Garantiza la integridad de la entidad.

Clave Secundaria / Foránea (Foreign Key):

Es una columna o conjunto de columnas en una tabla que hace referencia a la Clave Principal de otra tabla.

Establece la relación entre dos tablas.

Garantiza la integridad referencial; asegura que la relación entre los datos sea válida.

4. Uniones Internas y Externas (JOINS)

Las uniones se utilizan para combinar filas de dos o más tablas basándose en una columna relacionada.

Unión Interna (INNER JOIN):

Devuelve solo las filas que tienen valores coincidentes en ambas tablas.

Es la unión más común.

Unión Externa (OUTER JOIN):

Left Outer Join (LEFT JOIN): Devuelve todas las filas de la tabla izquierda y las filas coincidentes de la tabla derecha. Si no hay coincidencia, devuelve NULL para las columnas de la tabla derecha.

Right Outer Join (RIGHT JOIN): Es lo opuesto a la izquierda; devuelve todas las filas de la tabla derecha y las coincidentes de la izquierda.

Full Outer Join (FULL JOIN): Devuelve todas las filas cuando hay una coincidencia en una de las tablas, rellenando con NULL donde no hay coincidencia.

5.- Diferencia entre DROP TABLE y TRUNCATE TABLE

Característica

DROP TABLE

TRUNCATE TABLE

Estructura

Elimina la tabla completa (estructura, datos e índices).

Elimina solo los datos, la estructura se mantiene.

Metadatos

Elimina los metadatos de la tabla del diccionario de datos.

Mantiene los metadatos de la tabla.

Auto-Incremento

Se restablece.

Se restablece.

Uso

Cuando la tabla ya no es necesaria.

Para vaciar una tabla grande de forma rápida, manteniendo su diseño

6.- Tipos de Datos de SQL Comunes

Los tipos de datos definen qué tipo de valor puede almacenar una columna. Los más comunes son:

Numéricos:

INT: Números enteros.

DECIMAL/NUMERIC: Números con punto flotante de precisión exacta.

FLOAT/REAL: Números de punto flotante de precisión aproximada.

Texto/Cadenas:

VARCHAR(n): Cadena de longitud variable (máximo n caracteres).

CHAR(n): Cadena de longitud fija (rellenada con espacios hasta n caracteres).

TEXT: Cadenas largas.

Fecha y Hora:

DATE: Almacena sólo la fecha.

TIME: Almacena solo la hora.

DATETIME/TIMESTAMP: Almacena tanto la fecha como la hora.

7.- Cláusulas WHERE y HAVING

Ambas cláusulas se utilizan para filtrar datos, pero operan en diferentes momentos y sobre diferentes conjuntos de resultados:

WHERE:

Función: Filtra filas individuales antes de que se agrupen los datos.

Uso: Se aplica a columnas y se utiliza para condiciones de registro (ej. WHERE edad > 25).

Ubicación: Viene antes de la cláusula GROUP BY.

No puede usar funciones de agregación (como SUM(), COUNT()).

HAVING:

Función: Filtra grupos de filas después de que se hayan creado con la cláusula GROUP BY.

Uso: Se utiliza para aplicar condiciones a los resultados de las funciones de agregación (ej. HAVING COUNT(pedido) > 10).

Ubicación: Viene después de la cláusula GROUP BY.

Puede usar funciones de agregación.

Ejemplo Sencillo:

SQL

Fundamentos de AWS Cloud: primera parte

1. Definición de IaaS, PaaS y SaaS

Estos son los tres modelos fundamentales de servicio de la nube, que difieren en la cantidad de gestión que el proveedor realiza y la que el usuario debe asumir:

2.- Ventajas de la Informática en la Nube

Costos Reducidos (Opex vs. Capex): Se paga solo por lo que se usa (pay-as-you-go), eliminando la necesidad de grandes inversiones iniciales en hardware (CAPEX).

Escalabilidad: Se puede aumentar o disminuir rápidamente la capacidad de los recursos (cómputo, almacenamiento) según la demanda, automáticamente si es necesario.

Agilidad e Innovación: Permite desplegar recursos y aplicaciones mucho más rápido que en un entorno local, acelerando el desarrollo y las pruebas.

Respaldo y Recuperación ante Desastres: Los datos se replican en múltiples ubicaciones, lo que simplifica la copia de seguridad y la recuperación de forma rápida en caso de fallos.

Alcance Global: Se pueden implementar aplicaciones en múltiples regiones geográficas con facilidad, mejorando la latencia para usuarios en todo el mundo.

Mantenimiento Cero de Infraestructura: El proveedor de la nube es responsable del mantenimiento del hardware, la seguridad física y las actualizaciones de la infraestructura base.

3.- Definición de Región de AWS y Zona de Disponibilidad

Región de AWS: Una ubicación geográfica física y aislada donde AWS agrupa sus centros de datos. Cada región es un área completamente separada y aislada de las demás para garantizar la máxima tolerancia a fallos y estabilidad. Su propósito es la separación geográfica y la soberanía de los datos.

Zona de Disponibilidad (AZ): Uno o varios centros de datos discretos, con energía, refrigeración y redes redundantes e independientes, ubicados dentro de una Región. Las AZ están interconectadas con baja latencia. Su propósito es permitir a los clientes operar aplicaciones de alta disponibilidad y tolerancia a fallos al distribuir recursos entre ellas.

4.- Enumeración de Regiones de AWS

AWS tiene actualmente 33 regiones geográficas a nivel mundial (hasta noviembre de 2025). Algunas de las más relevantes incluyen:

América del Norte: N. Virginia (us-east-1), Ohio (us-east-2), N. California (us-west-1), Oregón (us-west-2), Canadá (Central).

América del Sur: São Paulo (sa-east-1).

Europa: Irlanda (eu-west-1), Frankfurt (eu-central-1), Londres (eu-west-2), París, Milán.

Asia Pacífico: Sídney, Tokio, Seúl, Bombay, Singapur.

Medio Oriente y África: Ciudad del Cabo, Baréin.

5. Categorías de Servicios de AWS

Los más de 200 servicios de AWS se agrupan en las siguientes categorías principales:

Cómputo (Compute): Servidores virtuales y contenedores (ej. EC2, Lambda).

Almacenamiento (Storage): Bases de datos, data lakes y almacenamiento de archivos (ej. S3, EBS).

Bases de Datos (Databases): Servicios de bases de datos relacionales y NoSQL (ej. RDS, DynamoDB).

Redes y Entrega de Contenido (Networking & Content Delivery): Gestión de tráfico y conectividad (ej. VPC, Route 53, CloudFront).

Seguridad, Identidad y Cumplimiento (Security, Identity & Compliance): Gestión de accesos y protección de datos (ej. IAM, Cognito).

Desarrollador (Developer Tools): Herramientas para CI/CD y monitoreo (ej. CodeCommit, CloudWatch).

Aprendizaje Automático (Machine Learning): Servicios para crear y desplegar modelos de ML (ej. SageMaker).

6. Almacenamiento de Objetos vs. Almacenamiento en Bloques

Característica

Almacenamiento de Objetos (Object Storage)

Almacenamiento en Bloques (Block Storage)

Acceso

Accedido a través de API HTTP (por URL), no a través del sistema de archivos del SO.

Accedido a través de un sistema de archivos del Sistema Operativo.

Uso Principal

Almacenamiento web, backups, data lakes. Archivos no estructurados.

Discos duros virtuales para máquinas virtuales (VM).

Escalabilidad

Prácticamente ilimitada.

Limitada al tamaño del disco duro asignado a la VM.

Ejemplo AWS

Amazon S3.

Amazon EBS (Elastic Block Store).

7. Dos Servicios de Cómputo de AWS

Amazon Elastic Compute Cloud (EC2):

Definición: Proporciona servidores virtuales redimensionables (instancias) en la nube. Permite a los usuarios tener control total sobre la instancia, incluyendo el sistema operativo y las stacks de software. Es la columna vertebral de la infraestructura de cómputo IaaS de AWS.

AWS Lambda:

Definición: Es un servicio de cómputo sin servidor (serverless). Permite ejecutar código en respuesta a eventos (cambios de datos, peticiones HTTP, etc.) sin tener que provisionar ni gestionar servidores. El usuario solo paga por el tiempo de cómputo consumido.

8. Dos Servicios de Almacenamiento de AWS

Amazon Simple Storage Service (S3):

Definición: Ofrece almacenamiento de objetos escalable, seguro y duraderos (99.999999999%). Los datos se almacenan en buckets (cubos) y son accesibles a través de Internet mediante una API. Es el servicio estándar para data lakes, almacenamiento de backups y contenido web estático.

Amazon Elastic Block Store (EBS):

Definición: Proporciona volúmenes de almacenamiento en bloques persistentes que se pueden adjuntar a instancias EC2. Funcionan como un disco duro de red para la VM, lo que permite instalar sistemas operativos, bases de datos y aplicaciones que necesitan un sistema de archivos tradicional.

Fundamentos de AWS Cloud: segunda parte

1. Modelo de Responsabilidad Compartida de AWS

El modelo establece la línea divisoria de la responsabilidad de seguridad entre AWS y el cliente:

AWS es responsable de la seguridad de la nube (Security of the Cloud): Protege la infraestructura subyacente que ejecuta los servicios (hardware, software, redes e instalaciones).

El cliente es responsable de la seguridad en la nube (Security in the Cloud): Es responsable de la configuración de seguridad de los servicios que utiliza (ej. configuración de firewalls, cifrado de datos, gestión de acceso y parches del sistema operativo de sus instancias EC2).

¡Excelente! Aquí tienes el desarrollo breve y conciso de los ejercicios de investigación sobre Fundamentos de AWS Cloud, el AWS Well-Architected Framework, AWS CloudFormation y la Facturación de AWS.

2. Rol de AWS Identity and Access Management (IAM)

Un Rol de IAM es una identidad de IAM que se puede asumir. A diferencia de un usuario (que tiene credenciales a largo plazo), un rol no tiene credenciales permanentes. Los roles se utilizan para:

Delegación: Otorgar permisos temporales a usuarios, aplicaciones, o servicios de AWS para acceder a recursos específicos.

Ejemplo: Un rol que permite a una instancia EC2 acceder y escribir datos en un bucket de S3.

3. Política de AWS Identity and Access Management (IAM)

Una Política de IAM es un documento (en formato JSON) que define los permisos de una identidad.

Especifica lo que el recurso puede (o no puede) hacer.

Contiene elementos como: Effect (Allow/Deny), Action (qué operación se permite, ej. s3:GetObject), Resource (sobre qué recurso aplica) y Principal (quién puede realizar la acción).

4. Amazon Machine Image (AMI)

Una AMI es una plantilla que contiene la configuración de software (sistema operativo, servidor de aplicaciones y aplicaciones) necesaria para lanzar una instancia EC2.

Se utiliza para crear copias idénticas y consistentes de servidores.

Esencialmente, define la plantilla de la máquina virtual (VM).

5. Tipos de Instancias de Amazon EC2

Las instancias EC2 se agrupan en familias optimizadas para diferentes casos de uso:

Tipo de Instancia

Enfoque

Caso de Uso Típico

General Purpose (T/M)

Equilibrio de cómputo, memoria y red.

Servidores web, entornos de desarrollo/prueba, bases de datos pequeñas.

Compute Optimized (C)

Alto rendimiento de CPU.

Gaming, procesamiento por lotes, servidores web de alto tráfico.

Memory Optimized (R/X/Z)

Alto rendimiento de memoria (RAM).

Bases de datos de alto rendimiento, análisis de datos en memoria, cachés.

Storage Optimized (I/D)

Alto rendimiento de almacenamiento local.

Almacenes de datos, grandes bases de datos NoSQL, almacenamiento de registro.

Accelerated Computing (P/G/F)

Hardware acelerador (GPU, FPGA).

Aprendizaje automático (ML), gráficos, análisis de big data de alto rendimiento.

6. Amazon Virtual Private Cloud (VPC)

Amazon VPC es un servicio que permite aprovisionar una sección lógicamente aislada de la Nube de AWS donde se pueden lanzar recursos de AWS en una red virtual que se define.

Permite control total sobre el entorno de red virtual, incluyendo rangos de direcciones IP, subredes, tablas de ruteo y gateways de red.

7. Subred Pública y Subred Privada

Ambas son subdivisiones dentro de una VPC, pero se distinguen por su acceso a Internet:

Subred Pública:

Tiene una Ruta a un Internet Gateway (IGW) en su tabla de ruteo.

Los recursos en ella (ej. servidores web) pueden enviar y recibir tráfico directamente desde Internet.

Subred Privada:

No tiene una ruta directa a un Internet Gateway.

Los recursos en ella (ej. bases de datos) están aislados de Internet para mayor seguridad. Solo pueden acceder a Internet a través de un NAT Gateway (para actualizaciones y parches).

AWS Well-Architected Framework

1. Los Cinco Pilares de Well-Architected Framework (WAF)

WAF proporciona orientación para diseñar sistemas en la nube robustos, de alto rendimiento y rentables. Sus cinco pilares son:

Excelencia Operativa

Seguridad

Fiabilidad

Eficiencia del Rendimiento

Optimización de Costos

2.- 3 Áreas de Excelencia Operativa en la Nube

La Excelencia Operativa se centra en el buen funcionamiento y el monitoreo de los sistemas. Las tres áreas son:

Organización: Definir, documentar y refinar los procedimientos operativos.

Preparación: Diseñar la carga de trabajo y sus cambios con anticipación y planificar los despliegues.

Operación: Ejecutar tareas, monitorear el rendimiento y responder a eventos.

4.- Principios de Diseño que Aumentan la Fiabilidad

- Probar los Procedimientos de Recuperación: Simular fallos regularmente (no esperar a que ocurran).
- Recuperarse Automáticamente de Fallos: Configurar la curación automática (ej. Auto Scaling Group) y no depender de la intervención humana.
- Escalar Horizontalmente para Aumentar la Disponibilidad: Usar múltiples recursos en lugar de un único recurso grande para evitar un único punto de fallo.
- Gestionar el Cambio con Automatización: Usar versiones y despliegues controlados para minimizar el riesgo de introducir errores.

5. Áreas para Lograr la Eficiencia del Rendimiento

La eficiencia del rendimiento se logra centrándose en:

- Democratizar las Tecnologías Avanzadas: Usar servicios gestionados que aprovechan tecnologías modernas (ej. *serverless*, *machine learning*).
- Mecanismos sin Servidor (*Serverless*): Eliminar la necesidad de gestionar la infraestructura y escalar automáticamente (ej. Lambda).
- Ir Global en Minutos: Desplegar recursos en múltiples regiones para reducir la latencia para los usuarios.
- Experimentar con Menos Riesgo: Probar diferentes tipos de instancias y arquitecturas de manera rápida y económica.

6. Enfoques para Utilizar los Recursos de AWS de Forma Rentable

- Adoptar un Modelo de Consumo: Pagar solo por los recursos que se utilizan realmente (eliminar la capacidad ociosa).
- Medir la Eficacia General: Analizar métricas y uso para justificar el gasto y el dimensionamiento correcto.
- Analizar y Atribuir el Gasto: Usar etiquetas de costos (tags) para asignar gastos a equipos o proyectos.
- Usar Dimensionamiento Correcto (*Right Sizing*): Evaluar constantemente los tipos de instancias y eliminar recursos infrautilizados o *zombies*.

Ejercicio de Investigación de AWS CloudFormation

1. ¿Qué es el Aprovisionamiento de la Configuración?

Es el proceso de configurar e instalar el software, las dependencias y la configuración de una máquina después de que la infraestructura subyacente (VM, red) ha sido creada (ej. instalar un servidor Apache en una instancia EC2 ya lanzada).

2. ¿Qué es la Administración de la Configuración?

Es el proceso de mantener y gestionar el estado del software y la configuración de un sistema a lo largo del tiempo para asegurar que se mantenga en un estado deseado y consistente.

Herramientas Comunes: Ansible, Chef, Puppet, SaltStack.

3. ¿Qué es la Integración Continua (CI)?

CI es una práctica de desarrollo de software donde los desarrolladores integran los cambios de código en un repositorio central frecuentemente (varias veces al día). Cada integración es verificada por una construcción automatizada (build) y pruebas automatizadas para detectar errores rápidamente.

4. ¿Qué es la Entrega Continua (CD)?

CD es la extensión de CI. Es la práctica de garantizar que los cambios de código verificados se puedan implementar de forma segura y rápida en un entorno de producción (o staging) después de que hayan pasado las pruebas automatizadas.

5. ¿Qué es AWS CloudFormation?

AWS CloudFormation es un servicio que permite a los usuarios modelar y aprovisionar recursos de AWS e infraestructura de terceros en la nube como código (Infrastructure as Code - IaC). Utiliza archivos de plantilla declarativos para describir todos los recursos necesarios.

3 Ventajas de CloudFormation:

IaC: Permite el versionamiento, la revisión y la reutilización de la infraestructura.

Aprovisionamiento Estandarizado: Garantiza que los entornos sean idénticos (ej. Dev, Test, Prod).

Gestión de Dependencias: Gestiona el orden de creación y eliminación de recursos automáticamente.

6. ¿Qué son JSON y YAML?

Ambos son formatos de serialización de datos utilizados para escribir plantillas de CloudFormation y configuraciones.

| Característica | JSON (JavaScript Object Notation) | YAML (YAML Ain't Markup Language) |
|--------------------|--|--|
| Sintaxis | Usa llaves (<code>{ }</code>), corchetes (<code>[]</code>), comas y dos puntos. | Usa indentación (espacios) para definir la estructura. |
| Legibilidad | Menos legible para humanos, más legible para máquinas. | Más legible para humanos y más conciso. |
| Uso en CFN | Formato original, estricto con la sintaxis. | Formato más moderno y preferido por su concisión. |

7. ¿Qué es una Pila (Stack) en AWS CloudFormation?

Una Pila es una colección de recursos de AWS que se administran como una sola unidad.

Una plantilla de CloudFormation describe los recursos. Cuando ejecutas la plantilla, CloudFormation crea la pila, gestionando automáticamente la creación, actualización y eliminación de todos los recursos definidos en ella.

Ejercicio de Investigación de Facturación de AWS

¡Excelente! Aquí tienes el desarrollo breve y conciso de los ejercicios de investigación sobre Fundamentos de AWS Cloud, el AWS Well-Architected Framework, AWS CloudFormation y la Facturación de AWS.

9. Ejercicio de Investigación de Fundamentos de AWS Cloud

1. Modelo de Responsabilidad Compartida de AWS

El modelo establece la línea divisoria de la responsabilidad de seguridad entre AWS y el cliente:

AWS es responsable de la seguridad de la nube (Security of the Cloud): Protege la infraestructura subyacente que ejecuta los servicios (hardware, software, redes e instalaciones).

El cliente es responsable de la seguridad en la nube (Security in the Cloud): Es responsable de la configuración de seguridad de los servicios que utiliza (ej. configuración de firewalls, cifrado de datos, gestión de acceso y parches del sistema operativo de sus instancias EC2).

Shutterstock

2. Rol de AWS Identity and Access Management (IAM)

Un Rol de IAM es una identidad de IAM que se puede asumir. A diferencia de un usuario (que tiene credenciales a largo plazo), un rol no tiene credenciales permanentes. Los roles se utilizan para:

Delegación: Otorgar permisos temporales a usuarios, aplicaciones, o servicios de AWS para acceder a recursos específicos.

Ejemplo: Un rol que permite a una instancia EC2 acceder y escribir datos en un bucket de S3.

3. Política de AWS Identity and Access Management (IAM)

Una Política de IAM es un documento (en formato JSON) que define los permisos de una identidad.

Especifica lo que el recurso puede (o no puede) hacer.

Contiene elementos como: Effect (Allow/Deny), Action (qué operación se permite, ej. s3:GetObject), Resource (sobre qué recurso aplica) y Principal (quién puede realizar la acción).

4. Amazon Machine Image (AMI)

Una AMI es una plantilla que contiene la configuración de software (sistema operativo, servidor de aplicaciones y aplicaciones) necesaria para lanzar una instancia EC2.

Se utiliza para crear copias idénticas y consistentes de servidores.

Esencialmente, define la plantilla de la máquina virtual (VM).

5. Tipos de Instancias de Amazon EC2

Las instancias EC2 se agrupan en familias optimizadas para diferentes casos de uso:

| Tipo de Instancia | Enfoque | Caso de Uso Típico |
|-------------------------------|---|--|
| General Purpose (T/M) | Equilibrio de cómputo, memoria y red. | Servidores web, entornos de desarrollo/prueba, bases de datos pequeñas. |
| Compute Optimized (C) | Alto rendimiento de CPU. | Gaming, procesamiento por lotes, servidores web de alto tráfico. |
| Memory Optimized (R/X/Z) | Alto rendimiento de memoria (RAM). | Bases de datos de alto rendimiento, análisis de datos en memoria, cachés. |
| Storage Optimized (I/D) | Alto rendimiento de almacenamiento local. | Almacenes de datos, grandes bases de datos NoSQL, almacenamiento de registro. |
| Accelerated Computing (P/G/F) | Hardware acelerador (GPU, FPGA). | Aprendizaje automático (ML), gráficos, análisis de big data de alto rendimiento. |

6. Amazon Virtual Private Cloud (VPC)

Amazon VPC es un servicio que permite aprovisionar una sección lógicamente aislada de la Nube de AWS donde se pueden lanzar recursos de AWS en una red virtual que se define.

Permite control total sobre el entorno de red virtual, incluyendo rangos de direcciones IP, subredes, tablas de ruteo y gateways de red.

7. Subred Pública y Subred Privada

Ambas son subdivisiones dentro de una VPC, pero se distinguen por su acceso a Internet:

Subred Pública:

Tiene una Ruta a un Internet Gateway (IGW) en su tabla de ruteo.

Los recursos en ella (ej. servidores web) pueden enviar y recibir tráfico directamente desde Internet.

Subred Privada:

No tiene una ruta directa a un Internet Gateway.

Los recursos en ella (ej. bases de datos) están aislados de Internet para mayor seguridad. Solo pueden acceder a Internet a través de un NAT Gateway (para actualizaciones y parches).

11. Ejercicio de Investigación de AWS Well-Architected Framework

1. Los Cinco Pilares de Well-Architected Framework (WAF)

WAF proporciona orientación para diseñar sistemas en la nube robustos, de alto rendimiento y rentables. Sus cinco pilares son:

Getty Images

Excelencia Operativa

Seguridad

Fiabilidad

Eficiencia del Rendimiento

Optimización de Costos

2. 3 Áreas de Excelencia Operativa en la Nube

La Excelencia Operativa se centra en el buen funcionamiento y el monitoreo de los sistemas. Las tres áreas son:

Organización: Definir, documentar y refinar los procedimientos operativos.

Preparación: Diseñar la carga de trabajo y sus cambios con anticipación y planificar los despliegues.

Operación: Ejecutar tareas, monitorear el rendimiento y responder a eventos.

3. Principios de Diseño para Reforzar la Seguridad del Sistema

Implementar una Identidad Sólida: Usar IAM y Roles para aplicar el principio de mínimo privilegio.

Habilitar la Trazabilidad: Monitorear, alertar y auditar acciones y cambios en el entorno (ej. CloudTrail, CloudWatch).

Aplicar Seguridad en Todas las Capas: Utilizar firewalls (VPC), cifrado (Storage/Transfer), WAF, etc.

Automatizar las Mejores Prácticas de Seguridad: Integrar controles de seguridad como código.

4. Principios de Diseño que Aumentan la Fiabilidad

Probar los Procedimientos de Recuperación: Simular fallos regularmente (no esperar a que ocurran).

Recuperarse Automáticamente de Fallos: Configurar la curación automática (ej. Auto Scaling Group) y no depender de la intervención humana.

Escalar Horizontalmente para Aumentar la Disponibilidad: Usar múltiples recursos en lugar de un único recurso grande para evitar un único punto de fallo.

Gestionar el Cambio con Automatización: Usar versiones y despliegues controlados para minimizar el riesgo de introducir errores.

5. Áreas para Lograr la Eficiencia del Rendimiento

La eficiencia del rendimiento se logra centrándose en:

Democratizar las Tecnologías Avanzadas: Usar servicios gestionados que aprovechan tecnologías modernas (ej. serverless, machine learning).

Mecanismos sin Servidor (Serverless): Eliminar la necesidad de gestionar la infraestructura y escalar automáticamente (ej. Lambda).

Ir Global en Minutos: Desplegar recursos en múltiples regiones para reducir la latencia para los usuarios.

Experimentar con Menos Riesgo: Probar diferentes tipos de instancias y arquitecturas de manera rápida y económica.

6. Enfoques para Utilizar los Recursos de AWS de Forma Rentable

Adoptar un Modelo de Consumo: Pagar solo por los recursos que se utilizan realmente (eliminar la capacidad ociosa).

Medir la Eficacia General: Analizar métricas y uso para justificar el gasto y el dimensionamiento correcto.

Analizar y Atribuir el Gasto: Usar etiquetas de costos (tags) para asignar gastos a equipos o proyectos.

Usar Dimensionamiento Correcto (Right Sizing): Evaluar constantemente los tipos de instancias y eliminar recursos infrautilizados o zombies.

Ejercicio de Investigación de AWS CloudFormation

1. ¿Qué es el Aprovisionamiento de la Configuración?

Es el proceso de configurar e instalar el software, las dependencias y la configuración de una máquina después de que la infraestructura subyacente (VM, red) ha sido creada (ej. instalar un servidor Apache en una instancia EC2 ya lanzada).

2. ¿Qué es la Administración de la Configuración?

Es el proceso de mantener y gestionar el estado del software y la configuración de un sistema a lo largo del tiempo para asegurar que se mantenga en un estado deseado y consistente.

Herramientas Comunes: Ansible, Chef, Puppet, SaltStack.

3. ¿Qué es la Integración Continua (CI)?

CI es una práctica de desarrollo de software donde los desarrolladores integran los cambios de código en un repositorio central frecuentemente (varias veces al día). Cada integración es verificada por una construcción automatizada (build) y pruebas automatizadas para detectar errores rápidamente.

4. ¿Qué es la Entrega Continua (CD)?

CD es la extensión de CI. Es la práctica de garantizar que los cambios de código verificados se puedan implementar de forma segura y rápida en un entorno de producción (o staging) después de que hayan pasado las pruebas automatizadas.

5. ¿Qué es AWS CloudFormation?

AWS CloudFormation es un servicio que permite a los usuarios modelar y aprovisionar recursos de AWS e infraestructura de terceros en la nube como código (Infrastructure as Code - IaC). Utiliza archivos de plantilla declarativos para describir todos los recursos necesarios.

3 Ventajas de CloudFormation:

IaC: Permite el versionamiento, la revisión y la reutilización de la infraestructura.

Aprovisionamiento Estandarizado: Garantiza que los entornos sean idénticos (ej. Dev, Test, Prod).

Gestión de Dependencias: Gestiona el orden de creación y eliminación de recursos automáticamente.

6. ¿Qué son JSON y YAML?

Ambos son formatos de serialización de datos utilizados para escribir plantillas de CloudFormation y configuraciones.

| | | |
|----------------|-----------------------------------|-----------------------------------|
| Característica | JSON (JavaScript Object Notation) | YAML (YAML Ain't Markup Language) |
|----------------|-----------------------------------|-----------------------------------|

| | | |
|----------|--|--|
| Sintaxis | Usa llaves ({}), corchetes ([]), comas y dos puntos. | Usa indentación (espacios) para definir la estructura. |
|----------|--|--|

| | | |
|-------------|--|---|
| Legibilidad | Menos legible para humanos, más legible para máquinas. | Más legible para humanos y más conciso. |
|-------------|--|---|

| | | |
|------------|---|---|
| Uso en CFN | Formato original, estricto con la sintaxis. | Formato más moderno y preferido por su concisión. |
|------------|---|---|

7. ¿Qué es una Pila (Stack) en AWS CloudFormation?

Una Pila es una colección de recursos de AWS que se administran como una sola unidad.

Una plantilla de CloudFormation describe los recursos. Cuando ejecutas la plantilla, CloudFormation crea la pila, gestionando automáticamente la creación, actualización y eliminación de todos los recursos definidos en ella.

Ejercicio de Investigación de Facturación de AWS

1. Tipos de Planes de Soporte de AWS

AWS ofrece varios niveles de soporte técnico y guía:

Basic (Básico): Incluido de forma gratuita. Acceso a documentación, whitepapers y foros de la comunidad.

Developer (Desarrollador): Para entornos de desarrollo. Soporte por correo electrónico, respuesta rápida en horas de oficina.

Business (Empresarial): Para cargas de trabajo críticas. Soporte 24/7 por teléfono, chat o correo electrónico, respuesta rápida en minutos.

Enterprise On-Ramp / Enterprise (Empresarial): Para cargas de trabajo de misión crítica. Incluye un Administrador Técnico de Cuentas (TAM) y soporte de respuesta más rápida.

2. Calculadora de Costo Mensual de AWS

La Calculadora de Costo Mensual de AWS es una herramienta gratuita que permite a los usuarios estimar el costo mensual de los servicios de AWS que planean utilizar.

Permite modelar la carga de trabajo, seleccionar regiones, y experimentar con diferentes modelos de precios (ej. instancias reservadas) para obtener una proyección detallada de la factura antes de lanzar los recursos.

3. Modelos de Precios de Amazon EC2

Amazon EC2 ofrece diferentes modelos para optimizar los costos según el compromiso y la necesidad:

On-Demand (Bajo Demanda):

Definición: El usuario paga por la capacidad de cómputo por hora o por segundo sin compromisos a largo plazo.

Uso: Cargas de trabajo de corta duración, impredecibles o en desarrollo. Es el más caro, pero flexible.

Savings Plans / Reserved Instances (Planes de Ahorro / Instancias Reservadas):

Definición: El usuario se compromete a una cantidad constante de uso (por 1 o 3 años) a cambio de un descuento significativo (hasta 72%).

Uso: Cargas de trabajo estables o predictibles que se ejecutarán continuamente a largo plazo.

Spot Instances (Instancias de Spot):

Definición: Permite comprar la capacidad EC2 no utilizada de AWS con descuentos de hasta el 90% sobre el precio bajo demanda. AWS puede reclamar esta capacidad con poca antelación si la necesita.

Uso: Cargas de trabajo tolerantes a fallos (que se pueden interrumpir) como procesamiento por lotes, análisis de big data o tareas de CI/CD.

Dedicated Hosts (Hosts Dedicados):

Definición: Un servidor físico EC2 que es dedicado para el uso exclusivo del cliente.

Uso: Necesidades regulatorias o de licencia de software que requieren control total sobre la colocación del servidor físico.