Practice Test #2 - AWS Certified Solutions Architect Associate

Pregunta 1:
You have a team of developers in your company, and you would like to ensure they can quickly experiment with AWS Managed Policies by attaching them to their accounts, but you would like to prevent them from doing an escalation of privileges, by granting themselves the `AdministratorAccess` managed policy. How should you proceed?

- **Put the developers into an IAM group, and then define an IAM permission boundary on the group that will restrict the managed policies they can attach to themselves**
- **Attach an IAM policy to your developers, that prevents them from attaching the `AdministratorAccess` policy**
- **Create a Service Control Policy (SCP) on your AWS account that restricts developers from attaching themselves the `AdministratorAccess` policy**
- **For each developer, define an IAM permission boundary that will restrict the managed policies they can attach to themselves(Correcto)**

Explicación
Correct option:

For each developer, define an IAM permission boundary that will restrict the managed policies they can attach to themselves

AWS supports permissions boundaries for IAM entities (users or roles). A permissions boundary is an advanced feature for using a managed policy to set the maximum permissions that an identity-based policy can grant to an IAM entity. An entity's permissions

boundary allows it to perform only the actions that are allowed by both its identity-based policies and its permissions boundaries. Here we have to use an IAM permission boundary. They can only be applied to roles or users, not IAM groups.

Permissions boundaries for IAM entities:

Evaluating Effective Permissions with Boundaries

The permissions boundary for an IAM entity (user or role) sets the maximum permissions that the entity can have. This can change the effective permissions for that user or role. The effective permissions for an entity are the permissions that are granted by all the policies that affect the user or role. Within an account, the permissions for an entity can be affected by identity-based policies, resource-based policies, permissions boundaries, Organizations SCPs, or session policies. For more information about the different types of policies, see Policies and Permissions.

If any one of these policy types explicitly denies access for an operation, then the request is denied. The permissions granted to an entity by multiple permissions types are more complex. For more details about how AWS evaluates policies, see Policy Evaluation Logic.

Identity-based policies with boundaries – Identity-based policies are inline or managed policies that are attached to a user, group of users, or role. Identity-based policies grant permission to the entity, and permissions boundaries limit those permissions. The effective permissions are the intersection of both policy types. An explicit deny in either of these policies overrides the allow.

via - https://docs.aws.amazon.com/IAM/latest/UserGuide/access_policies_boundaries.html

Incorrect options:

Create a Service Control Policy (SCP) on your AWS account that restricts developers from attaching themselves the `AdministratorAccess` policy - Service control policies (SCPs) are one type of policy that you can use to manage your organization. SCPs offer central control over the maximum available permissions for all accounts in your organization, allowing you to ensure your accounts stay within your organization's access control guidelines. SCPs are available only in an organization that has all features enabled. SCPs aren't available if your organization has enabled only the consolidated billing features. Attaching an SCP to an AWS Organizations entity (root, OU, or account) defines a guardrail for what actions the principals can perform. If you consider this option, since AWS Organizations is not mentioned in this question, so we can't apply an SCP.

Attach an IAM policy to your developers, that prevents them from attaching the `AdministratorAccess` policy - This option is incorrect as the developers can remove this policy from themselves and escalate their privileges.

Put the developers into an IAM group, and then define an IAM permission boundary on the group that will restrict the managed policies they can attach to themselves - IAM permission boundary can only be applied to roles or users, not IAM groups. Hence this option is incorrect.

References:

https://docs.aws.amazon.com/IAM/latest/UserGuide/access_policies_boundaries.html

https://docs.aws.amazon.com/organizations/latest/userguide/orgs_manage_policies_scp.html

https://docs.aws.amazon.com/IAM/latest/UserGuide/access_policies_boundaries.html

Pregunta 2:
A developer needs to implement a Lambda function in AWS account A that accesses an Amazon S3 bucket in AWS account B.

As a Solutions Architect, which of the following will you recommend to meet this requirement?

- **The S3 bucket owner should make the bucket public so that it can be accessed by the Lambda function in the other AWS account**
- **Create an IAM role for the Lambda function that grants access to the S3 bucket. Set the IAM role as the Lambda function's execution role and that would give the Lambda function cross-account access to the S3 bucket**
- **Create an IAM role for the Lambda function that grants access to the S3 bucket. Set the IAM role as the Lambda function's execution role. Make sure that the bucket policy also grants access to the Lambda function's execution role(Correcto)**
- **AWS Lambda cannot access resources across AWS accounts. Use Identity federation to work around this limitation of Lambda**

Explicación
Correct option:

Create an IAM role for the Lambda function that grants access to the S3 bucket. Set the IAM role as the Lambda function's execution role. Make sure that the bucket policy also grants access to the Lambda function's execution role

If the IAM role that you create for the Lambda function is in the same AWS account as the bucket, then you don't need to grant Amazon S3 permissions on both the IAM role and the bucket policy. Instead, you can grant the permissions on the IAM role and then verify that the bucket policy doesn't explicitly deny access to the Lambda function role. If the IAM role and the bucket are in different accounts, then you need to grant Amazon S3 permissions on both the IAM role and the bucket policy. Therefore, this is the right way of giving access to AWS Lambda for the given use-case.

Complete list of steps to be followed:

I want my AWS Lambda function to be able to access my Amazon Simple Storage Service (Amazon S3) bucket. How can I do that?

Short Description

Follow these steps:

1. Create an AWS Identity and Access Management (IAM) role for the Lambda function that also grants access to the S3 bucket.
2. Set the IAM role as the Lambda function's execution role.
3. Verify that the bucket policy grants access to the Lambda function's execution role.

Important: If the IAM role that you create for the Lambda function is in the same AWS account as the bucket, then you don't need to grant Amazon S3 permissions on both the IAM role and the bucket policy. Instead, you can grant the permissions on the IAM role and then verify that the bucket policy doesn't explicitly deny access to the Lambda function role. As an example, the following procedure grants Amazon S3 permissions on the IAM role. If the IAM role and the bucket are in different accounts, then you need to grant Amazon S3 permissions on both the IAM role and the bucket policy.

via - https://aws.amazon.com/premiumsupport/knowledge-center/lambda-execution-role-s3-bucket/

Incorrect options:

AWS Lambda cannot access resources across AWS accounts. Use Identity federation to work around this limitation of Lambda - This is an incorrect statement, used only as a distractor.

Create an IAM role for the Lambda function that grants access to the S3 bucket. Set the IAM role as the Lambda function's execution role and that would give the Lambda function cross-account access to the S3 bucket - When the execution role of Lambda and S3 bucket to be accessed are from different accounts, then you need to grant S3 bucket access permissions to the IAM role and also ensure that the bucket policy grants access to the Lambda function's execution role.

The S3 bucket owner should make the bucket public so that it can be accessed by the Lambda function in the other AWS account - Making the S3 bucket public for the given use-case will be considered as a security bad practice. It's usually done for very few use-cases such as hosting a website on S3. Therefore this option is incorrect.

Reference:

https://aws.amazon.com/premiumsupport/knowledge-center/lambda-execution-role-s3-bucket/

Pregunta 3:
A systems administrator has created a private hosted zone and associated it with a Virtual Private Cloud (VPC). However, the DNS queries for the private hosted zone remain unresolved.

As a Solutions Architect, can you identify the Amazon VPC options to be configured in order to get the private hosted zone to work?

- **Fix conflicts between your private hosted zone and any Resolver rule that routes traffic to your network for the same domain name, as it results in ambiguity over the route to be taken**
- **Fix the Name server (NS) record and Start Of Authority (SOA) records that may have been created with wrong configurations**
- **Remove any overlapping namespaces for the private and public hosted zones**
- **Enable DNS hostnames and DNS resolution for private hosted zones(Correcto)**

Explicación
Correct option:

Enable DNS hostnames and DNS resolution for private hosted zones - DNS hostnames and DNS resolution are required settings for private hosted zones. DNS queries for private hosted zones can be resolved by the Amazon-provided VPC DNS server only. As a result, these options must be enabled for your private hosted zone to work.

DNS hostnames: For non-default virtual private clouds that aren't created using the Amazon VPC wizard, this option is disabled by default. If you create a private hosted zone for a domain and create records in the zone without enabling DNS hostnames, private hosted zones aren't enabled. To use a private hosted zone, this option must be enabled.

DNS resolution: Private hosted zones accept DNS queries only from a VPC DNS server. The IP address of the VPC DNS server is the reserved IP address at the base of the VPC IPv4 network range plus two. Enabling DNS resolution allows you to use the VPC DNS server as a Resolver for performing DNS resolution. Keep this option disabled if you're using a custom DNS server in the DHCP Options set, and you're not using a private hosted zone.

Incorrect options:

Remove any overlapping namespaces for the private and public hosted zones - If you have private and public hosted zones that have overlapping namespaces, such as example.com and accounting.example.com, then the Resolver routes traffic based on the most specific match. It won't result in unresolved queries, hence this option is wrong.

Fix the Name server (NS) record and Start Of Authority (SOA) records that may have been created with wrong configurations - When you create a hosted zone, Amazon Route 53 automatically creates a name server (NS) record and a start of authority (SOA) record for the zone for public hosted zone. However, this issue is about the private hosted zone, hence this is an incorrect option.

Fix conflicts between your private hosted zone and any Resolver rule that routes traffic to your network for the same domain name, as it results in ambiguity over the route to be taken - If you have a private hosted zone (example.com) and a Resolver rule that routes traffic to your network for the same domain name, the Resolver rule takes precedence. It won't result in unresolved queries.

References:

https://aws.amazon.com/premiumsupport/knowledge-center/vpc-enable-private-hosted-zone/

https://docs.aws.amazon.com/Route53/latest/DeveloperGuide/hosted-zone-private-considerations.html

https://docs.aws.amazon.com/Route53/latest/DeveloperGuide/hosted-zone-public-considerations.html

Pregunta 4:
Upon a security review of your AWS account, an AWS consultant has found that a few RDS databases are un-encrypted. As a Solutions Architect, what steps must be taken to encrypt the RDS databases?

- **Enable Multi-AZ for the database, and make sure the standby instance is encrypted. Stop the main database to that the standby database kicks in, then disable Multi-AZ**
- **Enable encryption on the RDS database using the AWS Console**
- **Create a Read Replica of the database, and encrypt the read replica. Promote the read replica as a standalone database, and terminate the previous database**
- **Take a snapshot of the database, copy it as an encrypted snapshot, and restore a database from the encrypted snapshot. Terminate the previous database(Correcto)**

Explicación
Correct option:

Take a snapshot of the database, copy it as an encrypted snapshot, and restore a database from the encrypted snapshot. Terminate the previous database

Amazon Relational Database Service (Amazon RDS) makes it easy to set up, operate, and scale a relational database in the cloud. It provides cost-efficient and resizable capacity while automating time-consuming administration tasks such as hardware provisioning, database setup, patching and backups.

You can encrypt your Amazon RDS DB instances and snapshots at rest by enabling the encryption option for your Amazon RDS DB instances. Data that is encrypted at rest includes the underlying storage for DB instances, its automated backups, read replicas, and snapshots.

You can only enable encryption for an Amazon RDS DB instance when you create it, not after the DB instance is created. However, because you can encrypt a copy of an unencrypted DB snapshot, you can effectively add encryption to an unencrypted DB instance. That is, you can create a snapshot of your DB instance, and then create an encrypted copy of that snapshot. So this is the correct option.

Incorrect options:

Create a Read Replica of the database, and encrypt the read replica. Promote the read replica as a standalone database, and terminate the previous database - If the master is not encrypted, the read replicas cannot be encrypted. So this option is incorrect.

Enable Multi-AZ for the database, and make sure the standby instance is encrypted. Stop the main database to that the standby database kicks in, then disable Multi-AZ - Multi-AZ is to help with High Availability, not encryption. So this option is incorrect.

Enable encryption on the RDS database using the AWS Console - There is no direct option to encrypt an RDS database using the AWS Console.

Steps to encrypt an un-encrypted RDS database: Create a snapshot of the un-encrypted database Copy the snapshot and enable encryption for the snapshot Restore the database from the encrypted snapshot Migrate applications to the new database, and delete the old database

Reference:

https://docs.aws.amazon.com/AmazonRDS/latest/UserGuide/Overview.Encryption.html

Pregunta 5:
A cybersecurity company uses a fleet of EC2 instances to run a proprietary application. The infrastructure maintenance group at the company wants to be notified via an email whenever the CPU utilization for any of the EC2 instances breaches a certain threshold.

Which of the following services would you use for building a solution with the LEAST amount of development effort? (Select two)

- **AWS Lambda**
- **Amazon SQS**
- **Amazon CloudWatch(Correcto)**
- **Amazon SNS(Correcto)**
- **AWS Step Functions**

Explicación
Correct options:

Amazon SNS - Amazon Simple Notification Service (SNS) is a highly available, durable, secure, fully managed pub/sub messaging service that enables you to decouple microservices, distributed systems, and serverless applications. Amazon SNS provides topics for high-throughput, push-based, many-to-many messaging.

Amazon CloudWatch - Amazon CloudWatch is a monitoring and observability service built for DevOps engineers, developers, site reliability engineers (SREs), and IT managers. CloudWatch provides you with data and actionable insights to monitor your

applications, respond to system-wide performance changes, optimize resource utilization, and get a unified view of operational health. Amazon CloudWatch allows you to monitor AWS cloud resources and the applications you run on AWS.

You can use CloudWatch Alarms to send an email via SNS whenever any of the EC2 instances breaches a certain threshold. Hence both these options are correct.

Incorrect options:

AWS Lambda - With AWS Lambda, you can run code without provisioning or managing servers. You pay only for the compute time that you consume—there's no charge when your code isn't running. You can run code for virtually any type of application or backend service—all with zero administration. You cannot use AWS Lambda to monitor CPU utilization of EC2 instances or send notification emails, hence this option is incorrect.

Amazon SQS - Amazon SQS Standard offers a reliable, highly scalable hosted queue for storing messages as they travel between computers. Amazon SQS lets you easily move data between distributed application components and helps you build applications in which messages are processed independently (with message-level ack/fail semantics), such as automated workflows. You cannot use SQS to monitor CPU utilization of EC2 instances or send notification emails, hence this option is incorrect.

AWS Step Functions - AWS Step Functions lets you coordinate multiple AWS services into serverless workflows so you can build and update apps quickly. Using Step Functions, you can design and run workflows that stitch together services, such as AWS Lambda, AWS Fargate, and Amazon SageMaker, into feature-rich applications. You cannot use Step Functions to monitor CPU utilization of EC2 instances or send notification emails, hence this option is incorrect.

References:

https://aws.amazon.com/cloudwatch/faqs/

https://aws.amazon.com/sns/

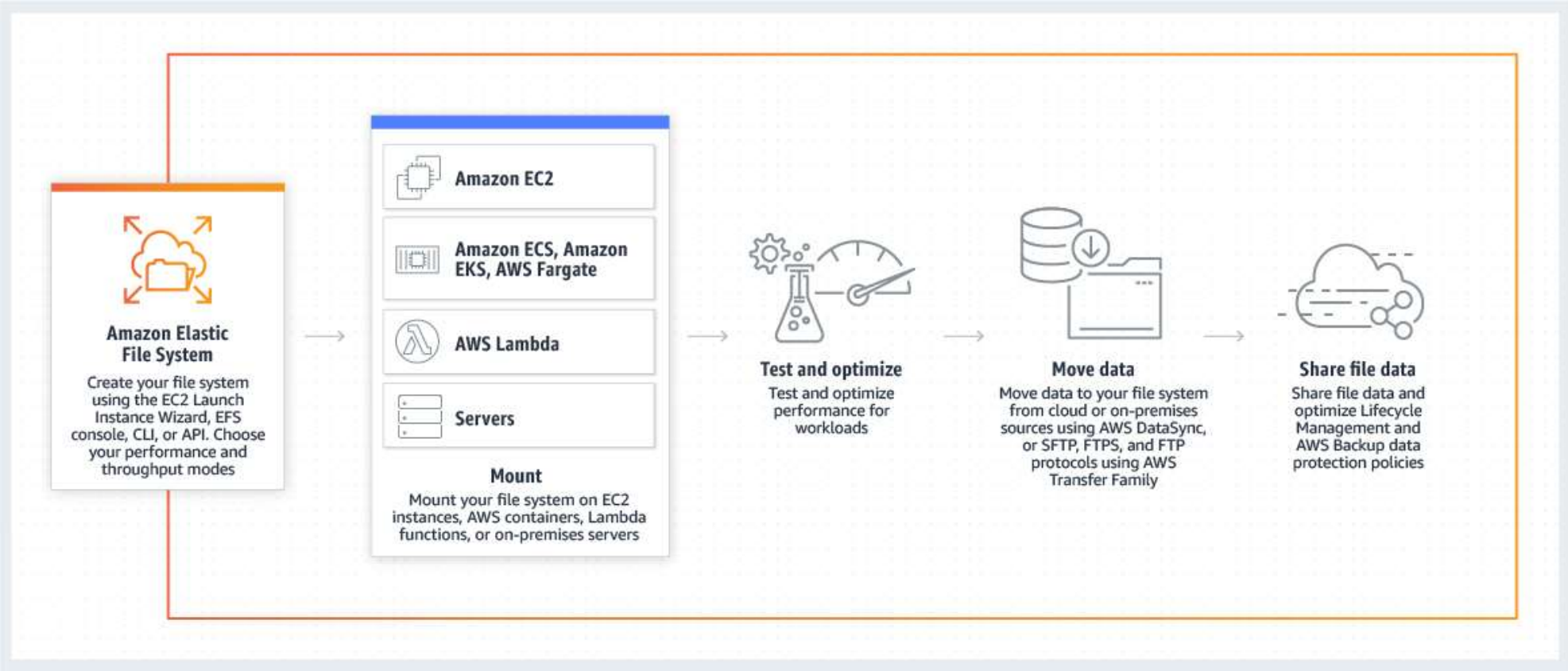
Pregunta 6:
An analytics company wants to improve the performance of its big data processing workflows running on Amazon EFS. Which of the following performance modes should be used for EFS to address this requirement?

- **General Purpose**
- **Max I/O (Correcto)**
- **Provisioned Throughput**
- **Bursting Throughput**
-

Explicación
Correct option:

Max I/O

How EFS Works:



via - https://aws.amazon.com/efs/

Max I/O performance mode is used to scale to higher levels of aggregate throughput and operations per second. This scaling is done with a tradeoff of slightly higher latencies for file metadata operations. Highly parallelized applications and workloads, such as big data analysis, media processing, and genomic analysis, can benefit from this mode.

Performance modes

To support a wide variety of cloud storage workloads, Amazon EFS offers two performance modes, *General Purpose* mode and *Max I/O* mode. You choose a file system's performance mode when you create it, and it cannot be changed.

> **Note**
>
> Max I/O performance mode is not available for file systems using One Zone storage classes.

The two performance modes have no additional costs, so your Amazon EFS file system is billed and metered the same, regardless of your performance mode. For information about file system quotas, see Quotas for Amazon EFS file systems.

> **Note**
>
> An Amazon EFS file system's performance mode can't be changed after the file system is created.

General Purpose performance mode

We recommend the General Purpose performance mode for the majority of your Amazon EFS file systems. General Purpose is ideal for latency-sensitive use cases, like web serving environments, content management systems, home directories, and general file serving. If you don't choose a performance mode when you create your file system, Amazon EFS selects the General Purpose mode for you by default.

Max I/O performance mode

File systems in the Max I/O mode can scale to higher levels of aggregate throughput and operations per second. This scaling is done with a tradeoff of slightly higher latencies for file metadata operations. Highly parallelized applications and workloads, such as big data analysis, media processing, and genomic analysis, can benefit from this mode.

via - https://docs.aws.amazon.com/efs/latest/ug/performance.html

Incorrect options:

Provisioned Throughput

Bursting Throughput

These two options have been added as distractors as these refer to the throughput mode of EFS and not the performance mode. There are two throughput modes to choose from for your file system, Bursting Throughput and Provisioned Throughput. With Bursting Throughput mode, throughput on Amazon EFS scales as the size of your file system in the standard storage class grows. With Provisioned Throughput mode, you can instantly provision the throughput of your file system (in MiB/s) independent of the amount of data stored.

General Purpose - General Purpose performance mode is ideal for latency-sensitive use cases, like web serving environments, content management systems, home directories, and general file serving. If you don't choose a performance mode when you create your file system, Amazon EFS selects the General Purpose mode for you by default.

References:

https://docs.aws.amazon.com/efs/latest/ug/performance.html

https://aws.amazon.com/efs/

Pregunta 7:
An IT company is working on client engagement to build a real-time data analytics tool for the Internet of Things (IoT) data. The IoT data is funneled into Kinesis Data Streams which further acts as the source of a delivery stream for Kinesis Firehose. The engineering team has now configured a Kinesis Agent to send IoT data from another set of devices to the same Firehose delivery stream. They noticed that data is not reaching Firehose as expected.

As a solutions architect, which of the following options would you attribute as the MOST plausible root cause behind this issue?

- **The data sent by Kinesis Agent is lost because of a configuration error**
- **Kinesis Agent cannot write to a Kinesis Firehose for which the delivery stream source is already set as Kinesis Data Streams(Correcto)**
- **Kinesis Agent can only write to Kinesis Data Streams, not to Kinesis Firehose**
- **Kinesis Firehose delivery stream has reached its limit and needs to be scaled manually**

Explicación
Correct option:

Kinesis Agent cannot write to a Kinesis Firehose for which the delivery stream source is already set as Kinesis Data Streams

Amazon Kinesis Data Firehose is the easiest way to reliably load streaming data into data lakes, data stores, and analytics tools. It is a fully managed service that automatically scales to match the throughput of your data and requires no ongoing administration. It can also batch, compress, transform, and encrypt the data before loading it, minimizing the amount of storage used at the destination and increasing security.

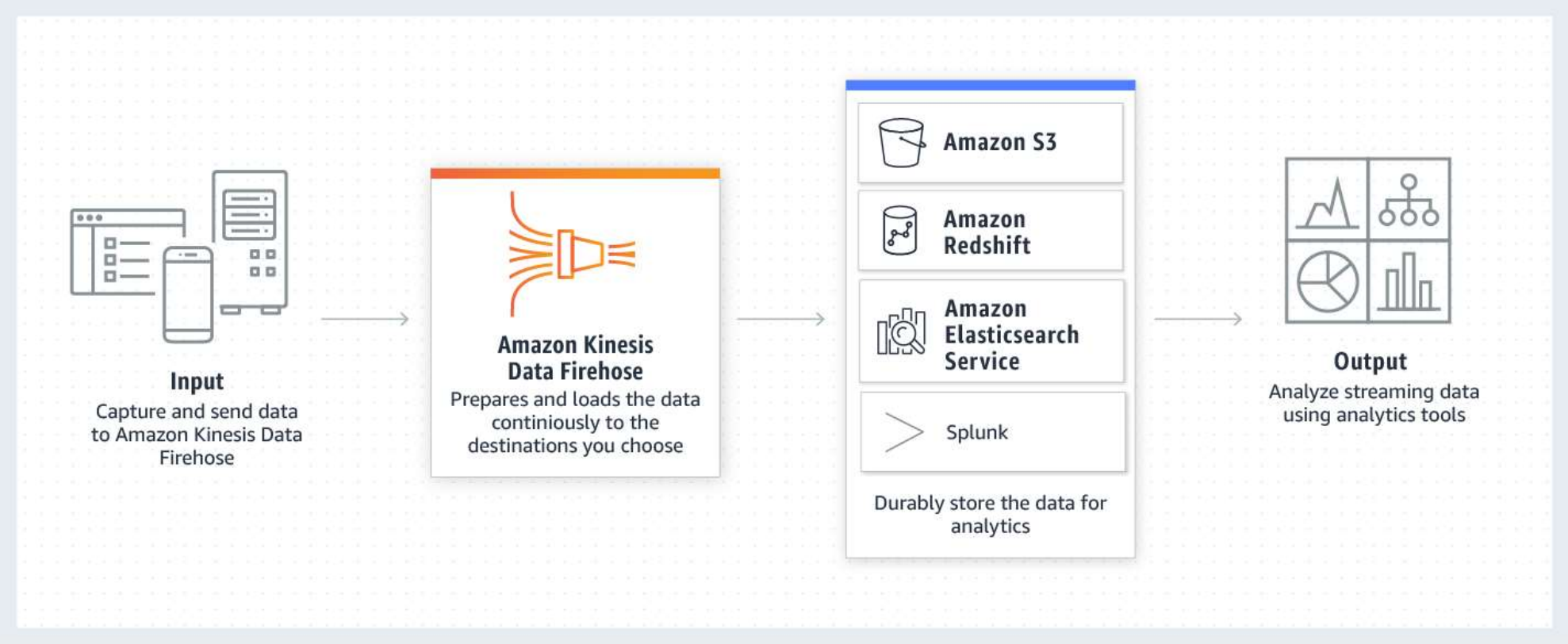
When a Kinesis data stream is configured as the source of a Firehose delivery stream, Firehose's PutRecord and PutRecordBatch operations are disabled and Kinesis Agent cannot write to Firehose delivery stream directly. Data needs to be added to the Kinesis data stream through the Kinesis Data Streams PutRecord and PutRecords operations instead. Therefore, this option is correct.

Incorrect options:

Kinesis Agent can only write to Kinesis Data Streams, not to Kinesis Firehose - Kinesis Agent is a stand-alone Java software application that offers an easy way to collect and send data to Kinesis Data Streams or Kinesis Firehose. So this option is incorrect.

Kinesis Firehose delivery stream has reached its limit and needs to be scaled manually - Kinesis Firehose is a fully managed service that automatically scales to match the throughput of your data and requires no ongoing administration. Therefore this option is not correct.

How Kinesis Firehose works:



via - https://aws.amazon.com/kinesis/data-firehose/

The data sent by Kinesis Agent is lost because of a configuration error - This is a made-up option and has been added as a distractor.

References:

https://aws.amazon.com/kinesis/data-firehose/

https://docs.aws.amazon.com/streams/latest/dev/writing-with-agents.html

https://docs.aws.amazon.com/firehose/latest/dev/writing-with-agents.html

Pregunta 8:
A company is looking at storing their less frequently accessed files on AWS that can be concurrently accessed by hundreds of EC2 instances. The company needs the most cost-effective file storage service that provides immediate access to data whenever needed.

Which of the following options represents the best solution for the given requirements?

- **Amazon Elastic File System (EFS) Standard storage class**
- **Amazon S3 Standard-Infrequent Access (S3 Standard-IA) storage class**
- **Amazon Elastic Block Store (EBS)**
- **Amazon Elastic File System (EFS) Standard–IA storage class(Correcto)**

Explicación
Correct option:

Amazon Elastic File System (EFS) Standard–IA storage class - Amazon EFS is a file storage service for use with Amazon compute (EC2, containers, serverless) and on-premises servers. Amazon EFS provides a file system interface, file system access semantics (such as strong consistency and file locking), and concurrently accessible storage for up to thousands of Amazon EC2 instances.

The Standard–IA storage class reduces storage costs for files that are not accessed every day. It does this without sacrificing the high availability, high durability, elasticity, and POSIX file system access that Amazon EFS provides. AWS recommends Standard-IA storage if you need your full dataset to be readily accessible and want to automatically save on storage costs for files that are less frequently accessed.

Incorrect options:

Amazon S3 Standard-Infrequent Access (S3 Standard-IA) storage class - Amazon S3 is an object storage service. Amazon S3 makes data available through an Internet API that can be accessed anywhere. It is not a file storage service, as is needed in the use case.

Amazon Elastic File System (EFS) Standard storage class - Amazon EFS Standard storage classes are ideal for workloads that require the highest levels of durability and availability. The EFS Standard storage class is used for frequently accessed files. It is the storage class to which customer data is initially written for Standard storage classes. The company is also looking at cutting costs by optimally storing the infrequently accessed data. Hence, EFS standard storage class is not the right solution for the given use case.

Amazon Elastic Block Store (EBS) - Amazon EBS is a block-level storage service for use with Amazon EC2. Amazon EBS can deliver performance for workloads that require the lowest latency access to data from a single EC2 instance. EBS volume cannot be accessed by hundreds of EC2 instances concurrently. It is not a file storage service, as is needed in the use case.

Reference:

https://docs.aws.amazon.com/efs/latest/ug/storage-classes.html

Pregunta 9

Which of the following IAM policies provides read-only access to the S3 bucket `mybucket` and its content?

1.

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

(Correcto)

2.

```
{
  "Version":"2012-10-17",
  "Statement":[
    {
      "Effect":"Allow",
      "Action":[
        "s3:ListBucket",
        "s3:GetObject"
      ],
      "Resource":"arn:aws:s3:::mybucket"
    }
  ]
}
```

3.

```
{
  "Version":"2012-10-17",
```

```
  "Statement":[
    {
      "Effect":"Allow",
      "Action":[
        "s3:ListBucket"
      ],
      "Resource":"arn:aws:s3:::mybucket/*"
    },
    {
      "Effect":"Allow",
      "Action":[
        "s3:GetObject"
      ],
      "Resource":"arn:aws:s3:::mybucket"
    }
  ]
}
```

4.

```
{
  "Version":"2012-10-17",
  "Statement":[
    {
      "Effect":"Allow",
      "Action":[
        "s3:ListBucket",
        "s3:GetObject"
      ],
      "Resource":"arn:aws:s3:::mybucket/*"
```

```
    }
  ]
```

Explicación

Correct option:

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

You manage access in AWS by creating policies and attaching them to IAM identities (users, groups of users, or roles) or AWS resources. A policy is an object in AWS that, when associated with an identity or resource, defines their permissions. AWS evaluates these policies when an IAM principal (user or role) makes a request. Permissions in the policies determine whether the request is allowed or denied. Most policies are stored in AWS as JSON documents. AWS supports six types of policies: identity-based policies, resource-based policies, permissions boundaries, Organizations SCPs, ACLs, and session policies.

`s3:ListBucket` is applied to buckets, so the ARN is in the form `"Resource":"arn:aws:s3:::mybucket"`, without a trailing `/` `s3:GetObject` is applied to objects within the bucket, so the ARN is in the form `"Resource":"arn:aws:s3:::mybucket/*"`, with a trailing `/*` to indicate all objects within the bucket `mybucket`

Therefore, this is the correct option.

```
{
  "Version":"2012-10-17",
  "Statement":[
    {
      "Effect":"Allow",
      "Action":[
        "s3:ListBucket",
        "s3:GetObject"
      ],
      "Resource":"arn:aws:s3:::mybucket"
    }
  ]
}
```

This option is incorrect as it provides read-only access only to the bucket, not its contents.

```
{
  "Version":"2012-10-17",
  "Statement":[
    {
      "Effect":"Allow",
      "Action":[
        "s3:ListBucket",
        "s3:GetObject"
      ],
      "Resource":"arn:aws:s3:::mybucket/*"
```

```
    }
  ]
}
```

This option is incorrect as it provides read-only access only to the bucket contents, not to the bucket itself.

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

This option is incorrect as it provides listing access only to the bucket contents.

References:

https://aws.amazon.com/blogs/security/writing-iam-policies-how-to-grant-access-to-an-amazon-s3-bucket/

https://docs.aws.amazon.com/IAM/latest/UserGuide/access_policies.html

Pregunta 10:
Your company has an on-premises Distributed File System Replication (DFSR) service to keep files synchronized on multiple Windows servers, and would like to migrate to AWS cloud.

What do you recommend as a replacement for the DFSR?

- **Amazon S3**
- **FSx for Lustre**
- **EFS**
- **FSx for Windows(Correcto)**

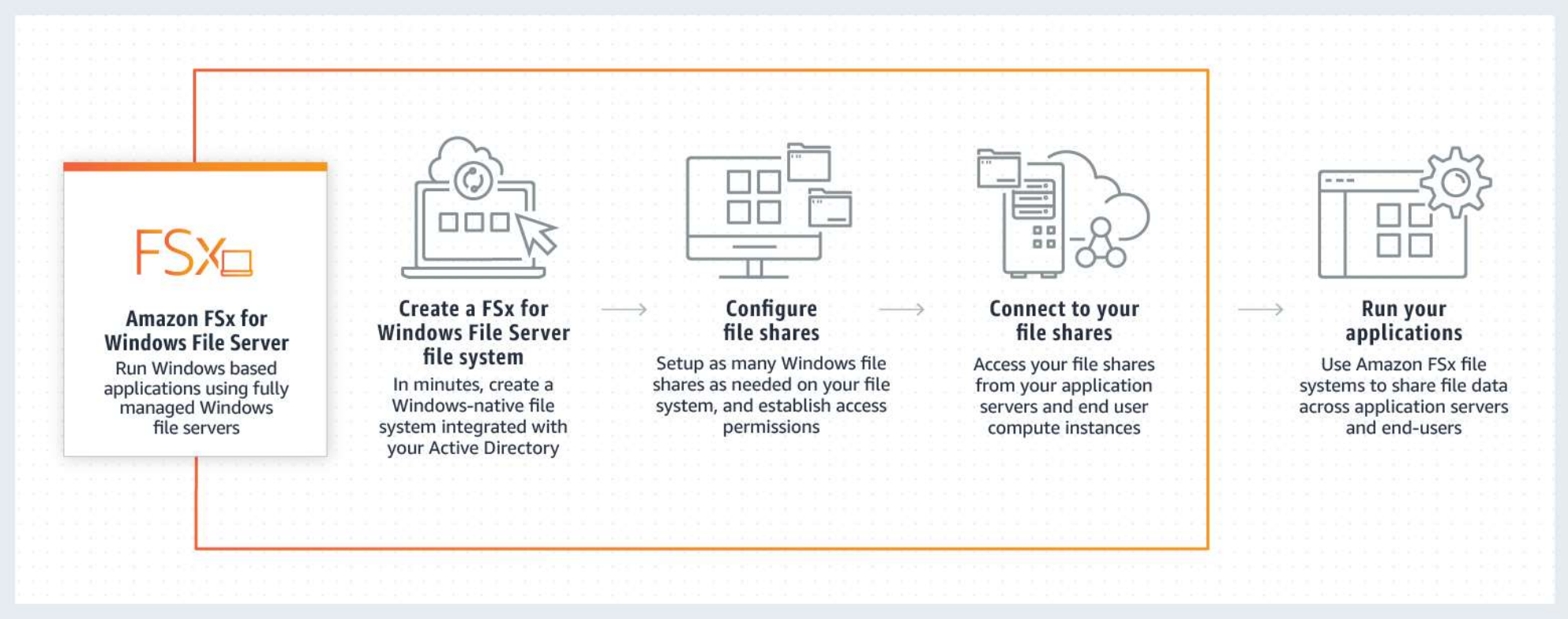
Explicación
Correct option:

FSx for Windows

Amazon FSx for Windows File Server provides fully managed, highly reliable file storage that is accessible over the industry-standard Service Message Block (SMB) protocol. It is built on Windows Server, delivering a wide range of administrative features such as user quotas, end-user file restore, and Microsoft Active Directory (AD) integration. The Distributed File System Replication (DFSR) service is a new multi-master replication engine that is used to keep folders synchronized on multiple servers. Amazon FSx supports the use of Microsoft's Distributed File System (DFS) to organize shares into a single folder structure up to hundreds of PB in size.

FSx for Windows is a perfect distributed file system, with replication capability, and can be mounted on Windows.

How FSx for Windows Works:



via - https://aws.amazon.com/fsx/windows/

Incorrect options:

FSx for Lustre - Amazon FSx for Lustre makes it easy and cost-effective to launch and run the world's most popular high-performance file system. It is used for workloads such as machine learning, high-performance computing (HPC), video processing, and financial modeling. The open-source Lustre file system is designed for applications that require fast storage – where you want

your storage to keep up with your compute. Amazon FSx enables you to use Lustre file systems for any workload where storage speed matters. FSx for Lustre integrates with Amazon S3, making it easy to process data sets with the Lustre file system.

FSx for Lustre is for Linux only, so this option is incorrect.

EFS - Amazon Elastic File System (Amazon EFS) provides a simple, scalable, fully managed elastic NFS file system for use with AWS Cloud services and on-premises resources. It is built to scale on-demand to petabytes without disrupting applications, growing and shrinking automatically as you add and remove files, eliminating the need to provision and manage capacity to accommodate growth.

EFS is a network file system but for Linux only, so this option is incorrect.

Amazon S3 - Amazon Simple Storage Service (Amazon S3) is an object storage service that offers industry-leading scalability, data availability, security, and performance.

Amazon S3 cannot be mounted as a file system on Windows, so this option is incorrect.

References:

https://docs.microsoft.com/en-us/previous-versions/windows/desktop/dfsr/dfsr-overview

https://aws.amazon.com/fsx/windows/

https://aws.amazon.com/fsx/lustre/

Pregunta 11:
A Machine Learning research group uses a proprietary computer vision application hosted on an EC2 instance. Every time the instance needs to be stopped and started again, the application takes about 3 minutes to start as some auxiliary software programs need to be executed so that the application can function. The research group would like to minimize the application boostrap time whenever the system needs to be stopped and then started at a later point in time.

As a solutions architect, which of the following solutions would you recommend for this use-case?

- **Use EC2 Instance Hibernate(Correcto)**
- **Create an AMI and launch your EC2 instances from that**
- **Use EC2 Meta-Data**
- **Use EC2 User-Data**

Explicación
Correct option:

Use EC2 Instance Hibernate

When you hibernate an instance, AWS signals the operating system to perform hibernation (suspend-to-disk). Hibernation saves the contents from the instance memory (RAM) to your Amazon EBS root volume. AWS then persists the instance's Amazon EBS root volume and any attached Amazon EBS data volumes.

When you start your instance:

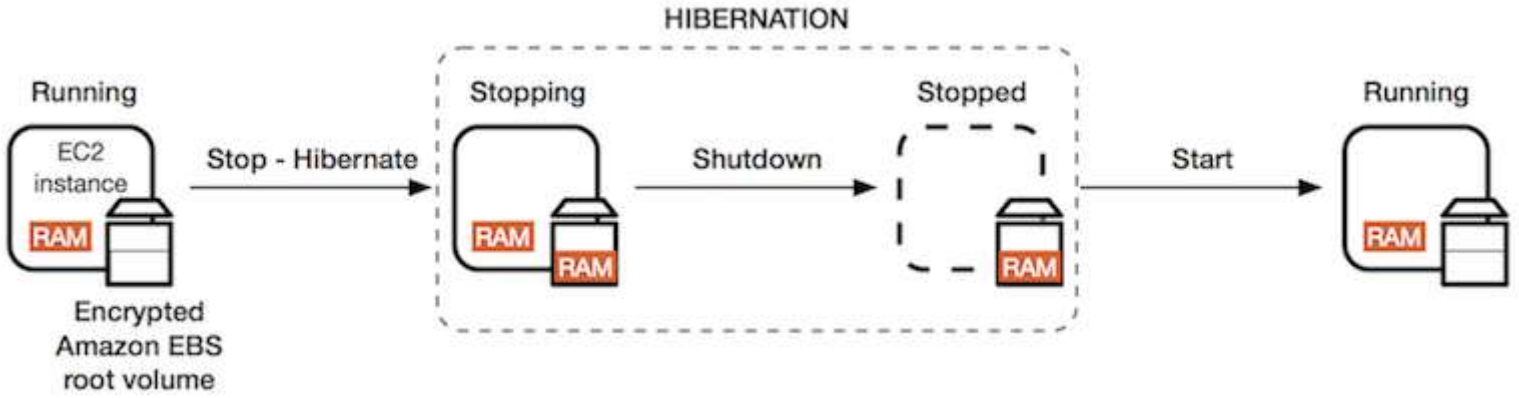
The Amazon EBS root volume is restored to its previous state

The RAM contents are reloaded

The processes that were previously running on the instance are resumed

Previously attached data volumes are reattached and the instance retains its instance ID

Overview of EC2 hibernation:



via - https://docs.aws.amazon.com/AWSEC2/latest/UserGuide/Hibernate.html

By using EC2 hibernate, we have the capability to resume it at any point of time, with the application already launched, thus helping us cut the 3 minutes start time.

Incorrect options:

Use EC2 User-Data - EC2 instance user data is the data that you specified in the form of a configuration script while launching your instance. Here, the problem is that the application takes 3 minutes to launch, no matter what. EC2 user data won't help us because it's just here to help us execute a list of commands, not speed them up.

Use EC2 Meta-Data - EC2 instance metadata is data about your instance that you can use to configure or manage the running instance. Instance metadata is divided into categories, for example, host name, events, and security groups. The EC2 meta-data is a distractor and can only help us determine some metadata attributes on our EC2 instances.

Create an AMI and launch your EC2 instances from that - An Amazon Machine Image (AMI) provides the information required to launch an instance. You must specify an AMI when you launch an instance. You can launch multiple instances from a single AMI when you need multiple instances with the same configuration. You can use different AMIs to launch instances when you need instances with different configurations.

Creating an AMI may help with all the system dependencies, but it won't help us with speeding up the application start time.

References:

https://docs.aws.amazon.com/AWSEC2/latest/UserGuide/Hibernate.html

https://docs.aws.amazon.com/AWSEC2/latest/UserGuide/ec2-instance-metadata.html

https://docs.aws.amazon.com/AWSEC2/latest/UserGuide/AMIs.html

Pregunta 12:
Amazon EC2 Auto Scaling needs to terminate an instance from Availability Zone (AZ) `us-east-1a` as it has the most number of instances amongst the AZs being used currently. There are 4 instances in the AZ `us-east-1a` like so: Instance A has the oldest launch template, Instance B has the oldest launch configuration, Instance C has the newest launch configuration and Instance D is closest to the next billing hour.

Which of the following instances would be terminated per the default termination policy?

- **Instance B(Correcto)**
- **Instance C**
- **Instance D**
- **Instance A**

Explicación
Correct option:

Instance B

Per the default termination policy, the first priority is given to any allocation strategy for On-Demand vs Spot instances. As no such information has been provided for the given use-case, so this criterion can be ignored. The next priority is to consider any instance with the oldest launch template unless there is an instance that uses a launch configuration. So this rules out Instance A. Next, you need to consider any instance which has the oldest launch configuration. This implies Instance B will be selected for termination and Instance C will also be ruled out as it has the newest launch configuration. Instance D, which is closest to the next billing hour, is not selected as this criterion is last in the order of priority.

Please see this note for a deep-dive on the default termination policy:

Default termination policy

The default termination policy is designed to help ensure that your instances span Availability Zones evenly for high availability. The default policy is kept generic and flexible to cover a range of scenarios.

Before Amazon EC2 Auto Scaling selects an instance to terminate, it first determines which Availability Zones have the most instances, and at least one instance that is not protected from scale in.

Within the selected Availability Zone, the default termination policy behavior is as follows:

1. Determine which instances to terminate so as to align the remaining instances to the allocation strategy for the On-Demand or Spot Instance that is terminating. This only applies to an Auto Scaling group that specifies a mixed instances policy, which uses allocation strategies.

   For example, after your instances launch, you change the priority order of your preferred instance types. When a scale-in event occurs, Amazon EC2 Auto Scaling tries to gradually shift the On-Demand Instances away from instance types that are lower priority.
2. Determine whether any of the instances use the oldest launch template or configuration:
   a. [For Auto Scaling groups that use a launch template]

      Determine whether any of the instances use the oldest launch template unless there are instances that use a launch configuration. Amazon EC2 Auto Scaling terminates instances that use a launch configuration before instances that use a launch template.

   b. [For Auto Scaling groups that use a launch configuration]

      Determine whether any of the instances use the oldest launch configuration.
3. After applying all of the above criteria, if there are multiple unprotected instances to terminate, determine which instances are closest to the next billing hour. If there are multiple unprotected instances closest to the next billing hour, terminate one of these instances at random.

   Note that terminating the instance closest to the next billing hour helps you maximize the use of your instances that have an hourly charge. Alternatively, if your Auto Scaling group uses Amazon Linux or Ubuntu, your EC2 usage is billed in one-second increments. For more information, see Amazon EC2 pricing.

via -https://docs.aws.amazon.com/autoscaling/ec2/userguide/as-instance-termination.html

Incorrect options:

Instance A

Instance C

Instance D

These three options contradict the explanation provided above, so these options are incorrect.

Reference:

https://docs.aws.amazon.com/autoscaling/ec2/userguide/as-instance-termination.html

Pregunta 13:
You have multiple AWS accounts within a single AWS Region managed by AWS Organizations and you would like to ensure all EC2 instances in all these accounts can communicate privately. Which of the following solutions provides the capability at the CHEAPEST cost?

- **Create a VPC in an account and share one or more of its subnets with the other accounts using Resource Access Manager(Correcto)**
- **Create a Private Link between all the EC2 instances**
- **Create a Transit Gateway and link all the VPC in all the accounts together**
- **Create a VPC peering connection between all VPCs**

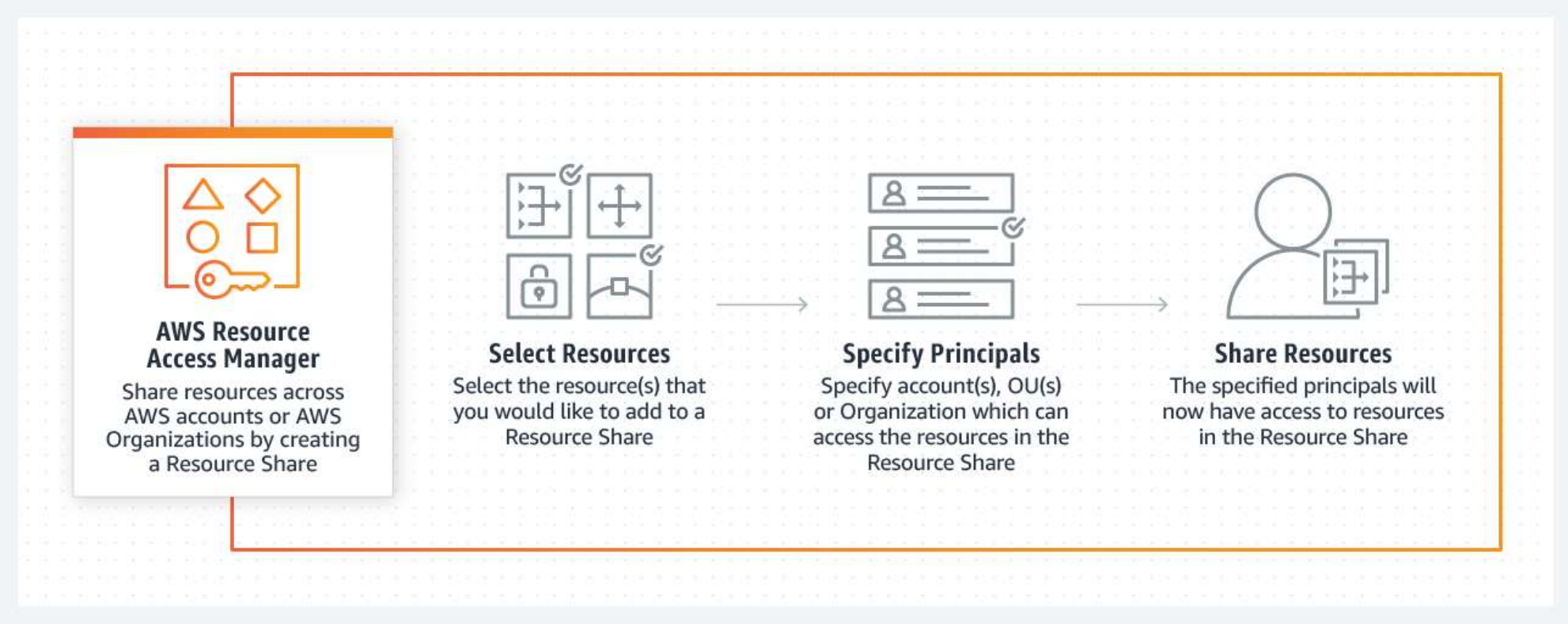
Explicación
Correct option:

Create a VPC in an account and share one or more of its subnets with the other accounts using Resource Access Manager

AWS Resource Access Manager (RAM) is a service that enables you to easily and securely share AWS resources with any AWS account or within your AWS Organization. You can share AWS Transit Gateways, Subnets, AWS License Manager configurations, and Amazon Route 53 Resolver rules resources with RAM. RAM eliminates the need to create duplicate resources in multiple accounts, reducing the operational overhead of managing those resources in every single account you own. You can create resources centrally in a multi-account environment, and use RAM to share those resources across accounts in three simple steps: create a Resource Share, specify resources, and specify accounts. RAM is available to you at no additional charge.

The correct solution is to share the subnet(s) within a VPC using RAM. This will allow all EC2 instances to be deployed in the same VPC (although from different accounts) and easily communicate with one another.

How Resource Access Manager Works:



via - https://aws.amazon.com/ram/

Incorrect options:

Create a Private Link between all the EC2 instances - AWS PrivateLink simplifies the security of data shared with cloud-based applications by eliminating the exposure of data to the public Internet. AWS PrivateLink provides private connectivity between VPCs, AWS services, and on-premises applications, securely on the Amazon network. Private Link is a distractor in this question. Private Link is leveraged to create a private connection between an application that is fronted by an NLB in an account, and an Elastic Network Interface (ENI) in another account, without the need of VPC peering and allowing the connections between the two to remain within the AWS network.

Create a VPC peering connection between all VPCs - A VPC peering connection is a networking connection between two VPCs that enables you to route traffic between them using private IPv4 addresses or IPv6 addresses. Instances in either VPC can communicate with each other as if they are within the same network. You can create a VPC peering connection between your VPCs, or with a VPC in another AWS account. The VPCs can be in different regions (also known as an inter-region VPC peering connection). VPC peering connections will work, but won't efficiently scale if you add more accounts (you'll have to create many connections).

Create a Transit Gateway and link all the VPC in all the accounts together - AWS Transit Gateway is a service that enables customers to connect their Amazon Virtual Private Clouds (VPCs) and their on-premises networks to a single gateway. A Transit Gateway will work but will be an expensive solution. Here we want to minimize cost.

References:

https://aws.amazon.com/ram/

https://aws.amazon.com/privatelink/

https://docs.aws.amazon.com/vpc/latest/peering/what-is-vpc-peering.html

https://aws.amazon.com/transit-gateway/

Pregunta 14:

What does this IAM policy do?

```
{
 "Version": "2012-10-17",
 "Statement": [
  {
   "Sid": "Mystery Policy",
   "Action": [
    "ec2:RunInstances"
   ],
   "Effect": "Allow",
   "Resource": "*",
   "Condition": {
    "IpAddress": {
     "aws:SourceIp": "34.50.31.0/24"
    }
   }
  }
 ]
}
```

- **It allows starting EC2 instances only when they have a Public IP within the `34.50.31.0/24` CIDR block**
- **It allows starting EC2 instances only when they have an Elastic IP within the `34.50.31.0/24` CIDR block**
- **It allows starting EC2 instances only when the IP where the call originates is within the `34.50.31.0/24` CIDR block(Correcto)**

- **It allows starting EC2 instances only when they have a Private IP within the `34.50.31.0/24` CIDR block**

Explicación
Correct option:

It allows starting EC2 instances only when the IP where the call originates is within the `34.50.31.0/24` CIDR block

You manage access in AWS by creating policies and attaching them to IAM identities (users, groups of users, or roles) or AWS resources. A policy is an object in AWS that, when associated with an identity or resource, defines their permissions. AWS evaluates these policies when an IAM principal (user or role) makes a request. Permissions in the policies determine whether the request is allowed or denied. Most policies are stored in AWS as JSON documents. AWS supports six types of policies: identity-based policies, resource-based policies, permissions boundaries, Organizations SCPs, ACLs, and session policies.

Consider the following snippet from the given policy document:

```
"Condition": {
  "IpAddress": {
    "aws:SourceIp": "34.50.31.0/24"
  }
}
```

The `aws:SourceIP` in this condition always represents the IP of the caller of the API. That is very helpful if you want to restrict access to certain AWS API for example from the public IP of your on-premises infrastructure.

Please see this overview of Elastic vs Public vs Private IP addresses:

Elastic IP address - An Elastic IP address is a static IPv4 address designed for dynamic cloud computing. An Elastic IP address is associated with your AWS account. With an Elastic IP address, you can mask the failure of an instance or software by rapidly remapping the address to another instance in your account.

Private IP address - A private IPv4 address is an IP address that's not reachable over the Internet. You can use private IPv4 addresses for communication between instances in the same VPC.

Public IP address - A public IP address is an IPv4 address that's reachable from the Internet. You can use public addresses for communication between your instances and the Internet.

Please note `34.50.31.0/24` is a public IP range, not a private IP range. Private IP ranges are: 192.168.0.0 - 192.168.255.255 (65,536 IP addresses) 172.16.0.0 - 172.31.255.255 (1,048,576 IP addresses) 10.0.0.0 - 10.255.255.255 (16,777,216 IP addresses)

Incorrect options:

It allows starting EC2 instances only when they have a Public IP within the `34.50.31.0/24` CIDR block

It allows starting EC2 instances only when they have an Elastic IP within the `34.50.31.0/24` CIDR block

It allows starting EC2 instances only when they have a Private IP within the `34.50.31.0/24` CIDR block

Each of these three options suggests that the IP addresses of the EC2 instances must belong to the `34.50.31.0/24` CIDR block for the EC2 instances to start. Actually, the policy states that the EC2 instance should start only when the IP where the call originates is within the `34.50.31.0/24` CIDR block. Hence these options are incorrect.

References:

https://docs.aws.amazon.com/IAM/latest/UserGuide/access_policies.html

https://aws.amazon.com/premiumsupport/knowledge-center/iam-restrict-calls-ip-addresses/

https://docs.aws.amazon.com/AWSEC2/latest/UserGuide/using-instance-addressing.html

Pregunta 15:
You would like to use Snowball to move on-premises backups into a long term archival tier on AWS. Which solution provides the MOST cost savings?

- **Create a Snowball job and target an S3 bucket. Create a lifecycle policy to transition this data to Glacier Deep Archive on the same day(Correcto)**
- **Create a Snowball job and target a Glacier Deep Archive Vault**
- **Create a Snowball job and target an S3 bucket. Create a lifecycle policy to transition this data to Glacier on the same day**
- **Create a Snowball job and target a Glacier Vault**

Explicación
Correct option:

Create a Snowball job and target an S3 bucket. Create a lifecycle policy to transition this data to Glacier Deep Archive on the same day

AWS Snowball, a part of the AWS Snow Family, is a data migration and edge computing device that comes in two options. Snowball Edge Storage Optimized devices provide both block storage and Amazon S3-compatible object storage, and 40 vCPUs. They are well suited for local storage and large scale data transfer. Snowball Edge Compute Optimized devices provide 52 vCPUs, block and object storage, and an optional GPU for use cases like advanced machine learning and full-motion video analysis in disconnected environments.

Snowball Edge Storage Optimized is the optimal choice if you need to securely and quickly transfer dozens of terabytes to petabytes of data to AWS. It provides up to 80 TB of usable HDD storage, 40 vCPUs, 1 TB of SATA SSD storage, and up to 40 Gb network connectivity to address large scale data transfer and pre-processing use cases.

The original Snowball devices were transitioned out of service and Snowball Edge Storage Optimized are now the primary devices used for data transfer. You may see the Snowball device on the exam, just remember that the original Snowball device had 80TB of storage space.

For this scenario, you will want to minimize the time spent in S3 Standard for all files to avoid unintended S3 Standard storage charges. To do this, AWS recommends using a zero-day lifecycle policy. From a cost perspective, when using a zero-day lifecycle policy, you are only charged S3 Glacier Deep Archive rates. When billed, the lifecycle policy is accounted for first, and if the destination is S3 Glacier Deep Archive, you are charged S3 Glacier Deep Archive rates for the transferred files.

You can't move data directly from Snowball into Glacier, you need to go through S3 first, and then use a lifecycle policy. So this option is correct.

Incorrect options:

Create a Snowball job and target a Glacier Vault

Create a Snowball job and target a Glacier Deep Archive Vault

Amazon S3 Glacier and S3 Glacier Deep Archive are a secure, durable, and extremely low-cost Amazon S3 cloud storage classes for data archiving and long-term backup. They are designed to deliver 99.999999999% durability and provide comprehensive security and compliance capabilities that can help meet even the most stringent regulatory requirements. Finally, Glacier Deep Archive provides more cost savings than Glacier.

Both these options are incorrect as you can't move data directly from Snowball into a Glacier Vault or a Glacier Deep Archive Vault. You need to go through S3 first and then use a lifecycle policy.

Create a Snowball job and target an S3 bucket. Create a lifecycle policy to transition this data to Glacier on the same day - As Glacier Deep Archive provides more cost savings than Glacier, so you should use Glacier Deep Archive for long term archival for this use-case.

References:

https://aws.amazon.com/snowball/features/

https://aws.amazon.com/glacier/

Pregunta 16
A big data consulting firm needs to set up a data lake on Amazon S3 for a Health-Care client. The data lake is split in raw and refined zones. For compliance reasons, the source data needs to be kept for a minimum of 5 years. The source data arrives in the raw zone and is then processed via an AWS Glue based ETL job into the refined zone. The business analysts run ad-hoc queries only on the data in the refined zone using AWS Athena. The team is concerned about the cost of data storage in both the raw and refined zones as the data is increasing at a rate of 1TB daily in each zone.

As a solutions architect, which of the following would you recommend as the MOST cost-optimal solution? (Select two)

- **Setup a lifecycle policy to transition the raw zone data into Glacier Deep Archive after 1 day of object creation(Correcto)**
- **Setup a lifecycle policy to transition the refined zone data into Glacier Deep Archive after 1 day of object creation**
- **Create a Lambda function based job to delete the raw zone data after 1 day**
- **Use Glue ETL job to write the transformed data in the refined zone using CSV format**
- **Use Glue ETL job to write the transformed data in the refined zone using a compressed file format(Correcto)**

Explicación
Correct options:

Setup a lifecycle policy to transition the raw zone data into Glacier Deep Archive after 1 day of object creation

You can manage your objects so that they are stored cost-effectively throughout their lifecycle by configuring their Amazon S3 Lifecycle. An S3 Lifecycle configuration is a set of rules that define actions that Amazon S3 applies to a group of objects. For example, you might choose to transition objects to the S3 Standard-IA storage class 30 days after you created them, or archive objects to the S3 Glacier storage class one year after creating them.

For the given use-case, the raw zone consists of the source data, so it cannot be deleted due to compliance reasons. Therefore, you should use a lifecycle policy to transition the raw zone data into Glacier Deep Archive after 1 day of object creation.

Please read more about S3 Object Lifecycle Management:

Object lifecycle management

PDF | Kindle | RSS

To manage your objects so that they are stored cost effectively throughout their lifecycle, configure their *Amazon S3 Lifecycle*. An *S3 Lifecycle configuration* is a set of rules that define actions that Amazon S3 applies to a group of objects. There are two types of actions:

- **Transition actions**—Define when objects transition to another storage class. For example, you might choose to transition objects to the S3 Standard-IA storage class 30 days after you created them, or archive objects to the S3 Glacier storage class one year after creating them.

  There are costs associated with the lifecycle transition requests. For pricing information, see Amazon S3 pricing.

- **Expiration actions**—Define when objects expire. Amazon S3 deletes expired objects on your behalf.

  The lifecycle expiration costs depend on when you choose to expire objects. For more information, see Understanding object expiration.

For more information about S3 Lifecycle rules, see Lifecycle configuration elements.

When should I use lifecycle configuration?

Define S3 Lifecycle configuration rules for objects that have a well-defined lifecycle. For example:

- If you upload periodic logs to a bucket, your application might need them for a week or a month. After that, you might want to delete them.
- Some documents are frequently accessed for a limited period of time. After that, they are infrequently accessed. At some point, you might not need real-time access to them, but your organization or regulations might require you to archive them for a specific period. After that, you can delete them.
- You might upload some types of data to Amazon S3 primarily for archival purposes. For example, you might archive digital media, financial and healthcare records, raw genomics sequence data, long-term database backups, and data that must be retained for regulatory compliance.

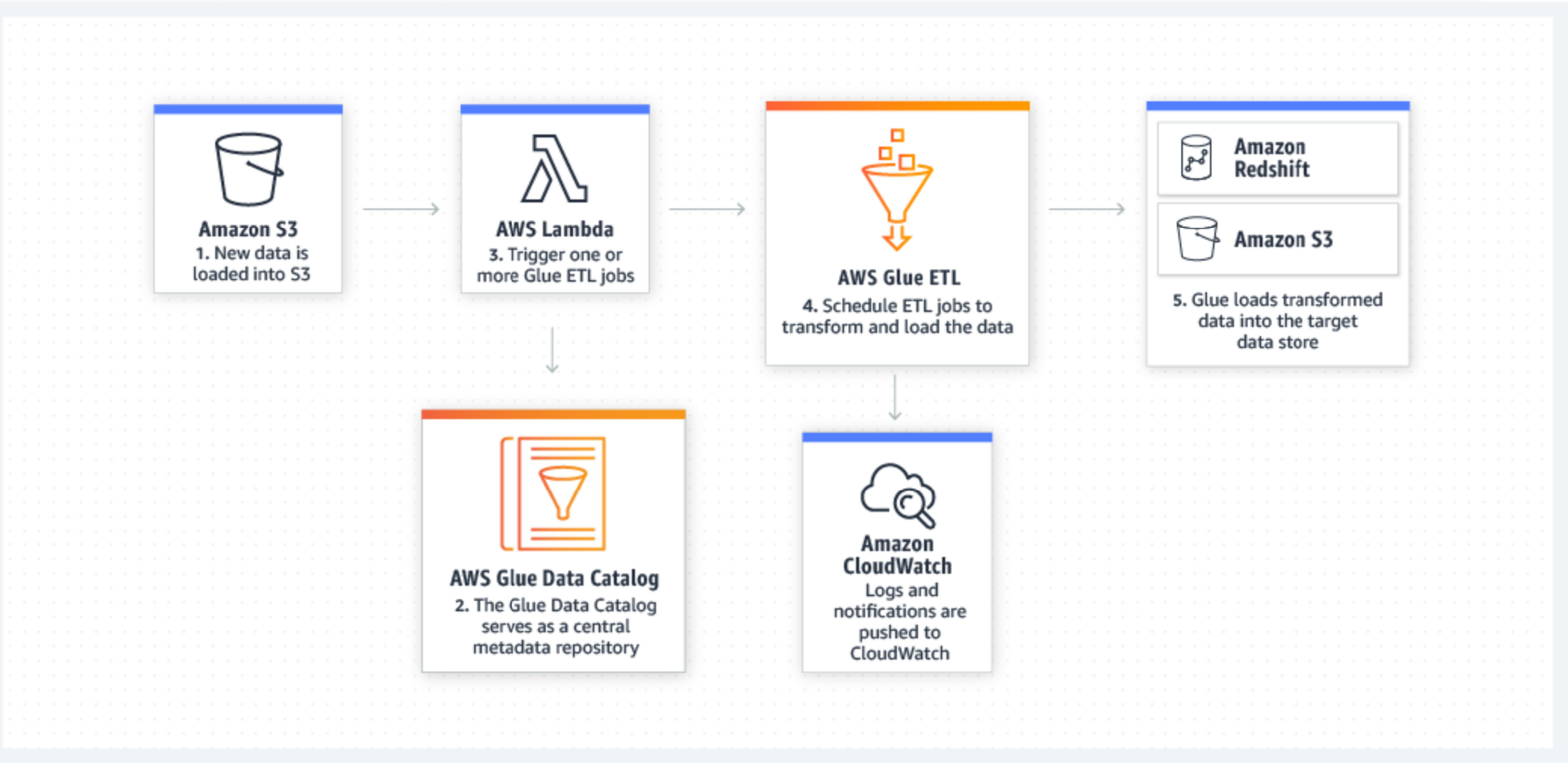
With S3 Lifecycle configuration rules, you can tell Amazon S3 to transition objects to less expensive storage classes, or archive or delete them.

via - https://docs.aws.amazon.com/AmazonS3/latest/dev/object-lifecycle-mgmt.html

Use Glue ETL job to write the transformed data in the refined zone using a compressed file format

AWS Glue is a fully managed extract, transform, and load (ETL) service that makes it easy for customers to prepare and load their data for analytics. You cannot transition the refined zone data into Glacier Deep Archive because it is used by the business analysts for ad-hoc querying. Therefore, the best optimization is to have the refined zone data stored in a compressed format via the Glue job. The compressed data would reduce the storage cost incurred on the data in the refined zone.

Please see this example for a Glue ETL Pipeline:



via - https://aws.amazon.com/glue/

Incorrect options:

Create a Lambda function based job to delete the raw zone data after 1 day - As mentioned in the use-case, the source data needs to be kept for a minimum of 5 years for compliance reasons. Therefore the data in the raw zone cannot be deleted after 1 day.

Setup a lifecycle policy to transition the refined zone data into Glacier Deep Archive after 1 day of object creation - You cannot transition the refined zone data into Glacier Deep Archive because it is used by the business analysts for ad-hoc querying. Hence this option is incorrect.

Use Glue ETL job to write the transformed data in the refined zone using CSV format - It is cost-optimal to write the data in the refined zone using a compressed format instead of CSV format. The compressed data would reduce the storage cost incurred on the data in the refined zone. So, this option is incorrect.

References:

https://docs.aws.amazon.com/AmazonS3/latest/dev/object-lifecycle-mgmt.html

https://aws.amazon.com/glue/

Pregunta 17:
A media company has created an AWS Direct Connect connection for migrating its flagship application to the AWS Cloud. The on-premises application writes hundreds of video files into a mounted NFS file system daily. Post-migration, the company will host the application on an Amazon EC2 instance with a mounted EFS file system. Before the migration cutover, the company must build a process that will replicate the newly created on-premises video files to the EFS file system.

Which of the following represents the MOST operationally efficient way to meet this requirement?

- **Configure an AWS DataSync agent on the on-premises server that has access to the NFS file system. Transfer data over the Direct Connect connection to an S3 bucket by using public VIF. Set up an AWS Lambda function to process event notifications from Amazon S3 and copy the video files from Amazon S3 to the EFS file system**
- **Configure an AWS DataSync agent on the on-premises server that has access to the NFS file system. Transfer data over the Direct Connect connection to an S3 bucket by using an S3 VPC endpoint. Set up an AWS Lambda function to process event notifications from Amazon S3 and copy the video files from Amazon S3 to the EFS file system**
- **Configure an AWS DataSync agent on the on-premises server that has access to the NFS file system. Transfer data over the Direct Connect connection to an AWS PrivateLink interface VPC endpoint for Amazon EFS by using a private VIF. Set up a DataSync scheduled task to send the video files to the EFS file system every 24 hours(Correcto)**

- **Configure an AWS DataSync agent on the on-premises server that has access to the NFS file system. Transfer data over the Direct Connect connection to an AWS VPC peering endpoint for Amazon EFS by using a private VIF. Set up a DataSync scheduled task to send the video files to the EFS file system every 24 hours**

Explicación
Correct option:

Configure an AWS DataSync agent on the on-premises server that has access to the NFS file system. Transfer data over the Direct Connect connection to an AWS PrivateLink interface VPC endpoint for Amazon EFS by using a private VIF. Set up a DataSync scheduled task to send the video files to the EFS file system every 24 hours

AWS DataSync is an online data transfer service that simplifies, automates, and accelerates copying large amounts of data between on-premises storage systems and AWS Storage services, as well as between AWS Storage services.

You can use AWS DataSync to migrate data located on-premises, at the edge, or in other clouds to Amazon S3, Amazon EFS, Amazon FSx for Windows File Server, Amazon FSx for Lustre, Amazon FSx for OpenZFS, and Amazon FSx for NetApp ONTAP.

AWS DataSync:

via - https://aws.amazon.com/datasync/

To establish a private connection between your virtual private cloud (VPC) and the Amazon EFS API, you can create an interface VPC endpoint. You can also access the interface VPC endpoint from on-premises environments or other VPCs using AWS VPN, AWS Direct Connect, or VPC peering.

AWS Direct Connect provides three types of virtual interfaces: public, private, and transit.

AWS Direct Connect VIFs:

Which type of Direct Connect virtual interface should I use to connect different AWS resources?

Last updated: 2022-01-21

AWS Direct Connect provides three types of virtual interfaces: public, private, and transit. How do I determine which type I should use to connect different AWS resources?

Resolution

Use the following Direct Connect virtual interface based on your use case.

Public virtual interface

To connect to AWS resources that are reachable by a public IP address such as an Amazon Simple Storage Service (Amazon S3) bucket or AWS public endpoints, use a public virtual interface. With a public virtual interface, you can:

- Connect to all AWS public IP addresses globally.
- Create public virtual interfaces in any Direct Connect location to receive Amazon's global IP routes.
- Access publicly routable Amazon services in any AWS Region (except the AWS China Region).

Private virtual interface

To connect to your resources hosted in an Amazon Virtual Private Cloud (Amazon VPC) using their private IP addresses, use a private virtual interface. With a private virtual interface, you can:

- Connect VPC resources such as Amazon Elastic Compute Cloud (Amazon EC2) instances or load balancers on your private IP address or endpoint.
- Connect a private virtual interface to a Direct Connect gateway. Then, associate the Direct Connect gateway with one or more virtual private gateways in any AWS Region (except the AWS China Region).
- Connect to multiple Amazon VPCs in any AWS Region (except the AWS China Region), because a virtual private gateway is associated with a single VPC.

Note: For a private virtual interface, AWS advertises the VPC CIDR only over the Border Gateway Protocol (BGP) neighbor. AWS can't advertise or suppress specific subnet blocks in the Amazon VPC for a private virtual interface.

Transit virtual interface

To connect to your resources hosted in an Amazon VPC (using their private IP addresses) through a transit gateway, use a transit virtual interface. With a transit virtual interface, you can:

- Connect multiple Amazon VPCs in the same or different AWS account using Direct Connect.
- Associate up to three transit gateways in any AWS Region when you use a transit virtual interface to connect to a Direct Connect gateway.
- Attach Amazon VPCs in the same AWS Region to the transit gateway. Then, access multiple VPCs in different AWS accounts in the same AWS Region using a transit virtual interface.

Note: For transit virtual interface, AWS advertises only routes that you specify in the allowed prefixes list on the Direct Connect gateway. For a list of all AWS Regions that offer Direct Connect support for AWS Transit Gateway, see AWS Transit Gateway support.

- https://aws.amazon.com/premiumsupport/knowledge-center/public-private-interface-dx/

For the given use case, you can send data over the Direct Connect connection to an AWS PrivateLink interface VPC endpoint for Amazon EFS by using a private VIF.

Using task scheduling in AWS DataSync, you can periodically execute a transfer task from your source storage system to the destination. You can use the DataSync scheduled task to send the video files to the EFS file system every 24 hours.

Incorrect options:

Configure an AWS DataSync agent on the on-premises server that has access to the NFS file system. Transfer data over the Direct Connect connection to an AWS VPC peering endpoint for Amazon EFS by using a private VIF. Set up a DataSync scheduled task to send the video files to the EFS file system every 24 hours - A VPC peering connection is a networking connection between two VPCs that enables you to route traffic between them privately. You cannot use VPC peering to transfer data over the Direct Connect connection from the on-premises systems to AWS. So this option is incorrect.

Configure an AWS DataSync agent on the on-premises server that has access to the NFS file system. Transfer data over the Direct Connect connection to an S3 bucket by using public VIF. Set up an AWS Lambda function to process event notifications from Amazon S3 and copy the video files from Amazon S3 to the EFS file system - You can use a public virtual interface to connect to AWS resources that are reachable by a public IP address such as an Amazon Simple Storage Service (Amazon S3) bucket or AWS public endpoints. Although it is theoretically possible to set up this solution, however, it is not the most operationally efficient solution, since it involves sending data via DataSync to S3 and then in turn using a Lambda function to finally send data to EFS.

Configure an AWS DataSync agent on the on-premises server that has access to the NFS file system. Transfer data over the Direct Connect connection to an S3 bucket by using an S3 VPC endpoint. Set up an AWS Lambda function to process event notifications from Amazon S3 and copy the video files from Amazon S3 to the EFS file system - You can access Amazon S3

from your VPC using gateway VPC endpoints. You cannot use the S3 VPC endpoint to transfer data over the Direct Connect connection from the on-premises systems to S3. So this option is incorrect.

References:

https://aws.amazon.com/datasync/

https://aws.amazon.com/blogs/storage/transferring-files-from-on-premises-to-aws-and-back-without-leaving-your-vpc-using-aws-datasync/

https://docs.aws.amazon.com/efs/latest/ug/efs-vpc-endpoints.html

https://aws.amazon.com/datasync/faqs/

https://aws.amazon.com/premiumsupport/knowledge-center/public-private-interface-dx/

https://docs.aws.amazon.com/datasync/latest/userguide/task-scheduling.html

Pregunta 18:
You would like to store a database password in a secure place, and enable automatic rotation of that password every 90 days. What do you recommend?

- **Key Management Service (KMS)**
- **SSM Parameter Store**
- **Secrets Manager(Correcto)**
- **CloudHSM**

Explicación
Correct option:

"Secrets Manager"

AWS Secrets Manager helps you protect secrets needed to access your applications, services, and IT resources. The service enables you to easily rotate, manage, and retrieve database credentials, API keys, and other secrets throughout their lifecycle. Users and applications retrieve secrets with a call to Secrets Manager APIs, eliminating the need to hardcode sensitive information in plain text. Secrets Manager offers secret rotation with built-in integration for Amazon RDS, Amazon Redshift, and Amazon DocumentDB. The correct answer here is Secrets Manager

Incorrect options:

"KMS" - AWS Key Management Service (AWS KMS) is a service that combines secure, highly available hardware and software to provide a key management system scaled for the cloud. When you use server-side encryption with AWS KMS (SSE-KMS), you can

specify a customer-managed CMK that you have already created. SSE-KMS provides you with an audit trail that shows when your CMK was used and by whom. KMS is an encryption service, it's not a secrets store. So this option is incorrect.

"CloudHSM" - AWS CloudHSM is a cloud-based hardware security module (HSM) that enables you to easily generate and use your encryption keys on the AWS Cloud. With CloudHSM, you can manage your encryption keys using FIPS 140-2 Level 3 validated HSMs. CloudHSM is standards-compliant and enables you to export all of your keys to most other commercially-available HSMs, subject to your configurations. It is a fully-managed service that automates time-consuming administrative tasks for you, such as hardware provisioning, software patching, high-availability, and backups.

CloudHSM is also an encryption service, not a secrets store. So this option is incorrect.

"SSM Parameter Store" - AWS Systems Manager Parameter Store (aka SSM Parameter Store) provides secure, hierarchical storage for configuration data management and secrets management. You can store data such as passwords, database strings, EC2 instance IDs, Amazon Machine Image (AMI) IDs, and license codes as parameter values. You can store values as plain text or encrypted data. You can reference Systems Manager parameters in your scripts, commands, SSM documents, and configuration and automation workflows by using the unique name that you specified when you created the parameter.

SSM Parameter Store can serve as a secrets store, but you must rotate the secrets yourself, it doesn't have an automatic capability for this. So this option is incorrect.

References:

https://aws.amazon.com/secrets-manager/

https://docs.aws.amazon.com/AmazonS3/latest/dev/UsingKMSEncryption.html

https://aws.amazon.com/cloudhsm/

https://docs.aws.amazon.com/systems-manager/latest/userguide/systems-manager-parameter-store.html

https://aws.amazon.com/blogs/mt/the-right-way-to-store-secrets-using-parameter-store/

https://docs.aws.amazon.com/systems-manager/latest/userguide/systems-manager-parameter-store.html

Pregunta 19:
To improve the performance and security of the application, the engineering team at a company has created a CloudFront distribution with an Application Load Balancer as the custom origin. The team has also set up a Web Application Firewall (WAF) with CloudFront distribution. The security team at the company has noticed a surge in malicious attacks from a specific IP address to steal sensitive data stored on the EC2 instances.

As a solutions architect, which of the following actions would you recommend to stop the attacks?

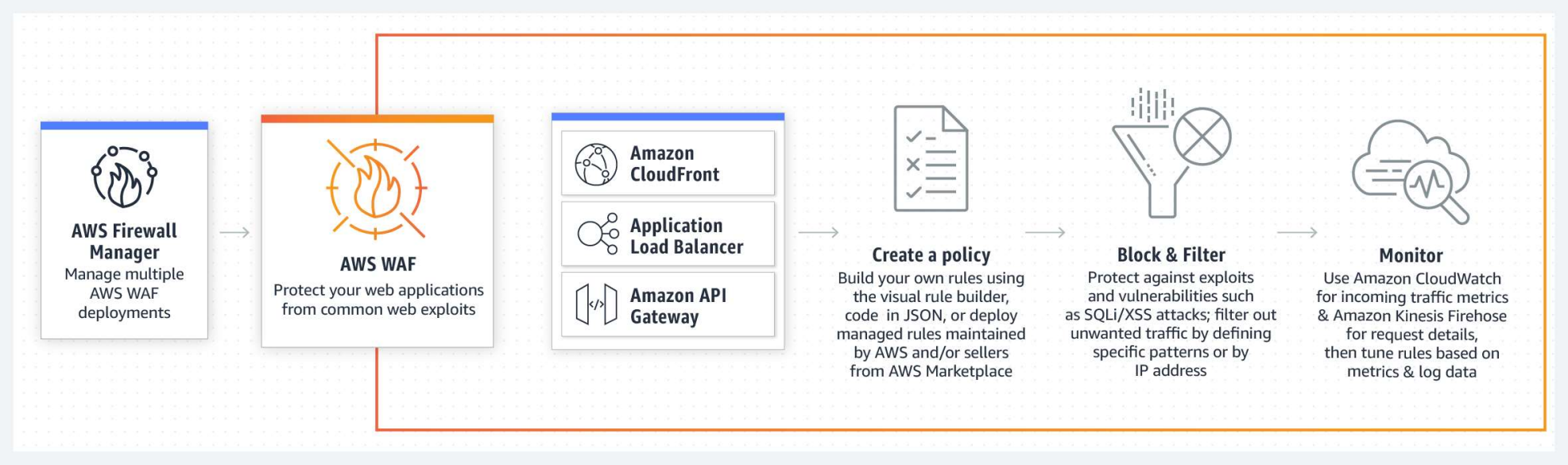
- **Create an IP match condition in the WAF to block the malicious IP address(Correcto)**
- **Create a ticket with AWS support to take action against the malicious IP**
- **Create a deny rule for the malicious IP in the Security Groups associated with each of the instances**
- **Create a deny rule for the malicious IP in the NACL associated with each of the instances**

Explicación
Correct option:

Create an IP match condition in the WAF to block the malicious IP address

AWS WAF is a web application firewall that helps protect your web applications or APIs against common web exploits that may affect availability, compromise security, or consume excessive resources. AWS WAF gives you control over how traffic reaches your applications by enabling you to create security rules that block common attack patterns, such as SQL injection or cross-site scripting, and rules that filter out specific traffic patterns you define.

How WAF Works:



via - https://aws.amazon.com/waf/

If you want to allow or block web requests based on the IP addresses that the requests originate from, create one or more IP match conditions. An IP match condition lists up to 10,000 IP addresses or IP address ranges that your requests originate from. So, this option is correct.

Incorrect options:

Create a deny rule for the malicious IP in the NACL associated with each of the instances - NACLs are not associated with instances. So this option is also ruled out.

Create a deny rule for the malicious IP in the Security Groups associated with each of the instances - You cannot deny rules in Security Groups. So this option is ruled out.

Create a ticket with AWS support to take action against the malicious IP - Managing the security of your application is your responsibility, not that of AWS, so you cannot raise a ticket for this issue.

Reference:

https://docs.aws.amazon.com/waf/latest/developerguide/classic-web-acl-ip-conditions.html

Pregunta 20:
You would like to migrate an AWS account from an AWS Organization A to an AWS Organization B. What are the steps do to it?

- **Send an invite to the new organization. Accept the invite to the new organization from the member account. Remove the member account from the old organization**
- **Open an AWS Support ticket to ask them to migrate the account**
- **Send an invite to the new organization. Remove the member account from the old organization. Accept the invite to the new organization from the member account**
- **Remove the member account from the old organization. Send an invite to the member account from the new Organization. Accept the invite to the new organization from the member account** **(Correcto)**

Explicación
Correct option:

Remove the member account from the old organization. Send an invite to the member account from the new Organization. Accept the invite to the new organization from the member account

AWS Organizations helps you centrally govern your environment as you grow and scale your workloads on AWS. Using AWS Organizations, you can automate account creation, create groups of accounts to reflect your business needs, and apply policies for these groups for governance. You can also simplify billing by setting up a single payment method for all of your AWS accounts. Through integrations with other AWS services, you can use Organizations to define central configurations and resource sharing across accounts in your organization.

To migrate accounts from one organization to another, you must have root or IAM access to both the member and master accounts. Here are the steps to follow: 1. Remove the member account from the old organization 2. Send an invite to the member account from the new Organization 3. Accept the invite to the new organization from the member account

Incorrect options:

Send an invite to the new organization. Accept the invite to the new organization from the member account. Remove the member account from the old organization

Send an invite to the new organization. Remove the member account from the old organization. Accept the invite to the new organization from the member account

These two options contradict the steps described earlier for account migration from one organization to another.

Open an AWS Support ticket to ask them to migrate the account - You don't need to contact AWS support for account migration.

References:

https://aws.amazon.com/organizations/

https://aws.amazon.com/premiumsupport/knowledge-center/organizations-move-accounts/

Pregunta 21:
A financial services company wants a single log processing model for all the log files (consisting of system logs, application logs, database logs, etc) that can be processed in a serverless fashion and then durably stored for downstream analytics. The company wants to use an AWS managed service that automatically scales to match the throughput of the log data and requires no ongoing administration.

As a solutions architect, which of the following AWS services would you recommend solving this problem?

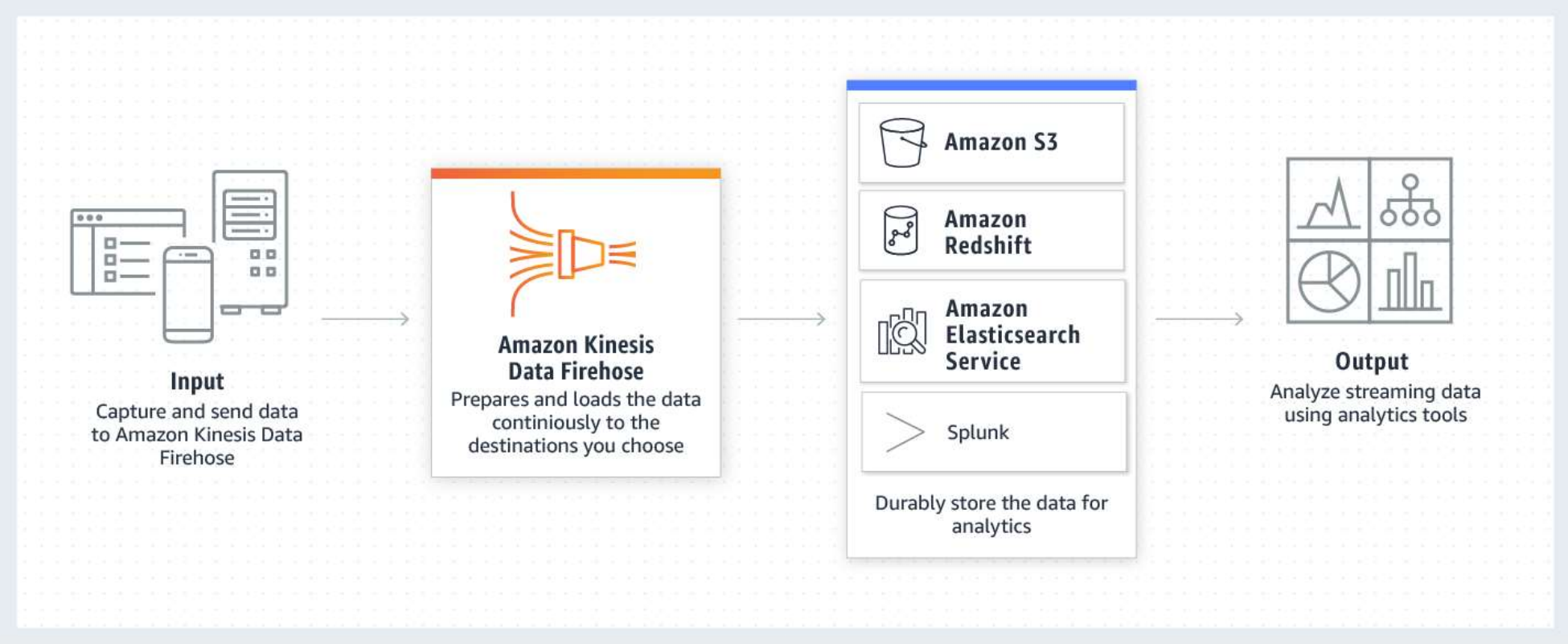
- **Kinesis Data Streams**
- **AWS Lambda**
- **Kinesis Data Firehose(Correcto)**
- **Amazon EMR**

Explicación
Correct option:

Kinesis Data Firehose

Amazon Kinesis Data Firehose is the easiest way to reliably load streaming data into data lakes, data stores, and analytics tools. It can capture, transform, and load streaming data into Amazon S3, Amazon Redshift, Amazon Elasticsearch Service, and Splunk, enabling near real-time analytics with existing business intelligence tools and dashboards you're already using today. It is a fully managed service that automatically scales to match the throughput of your data and requires no ongoing administration. Therefore, this is the correct option.

Please see this overview of how Kinesis Firehose works:



via - https://aws.amazon.com/kinesis/data-firehose/

Incorrect options:

Kinesis Data Streams - Amazon Kinesis Data Streams (KDS) is a massively scalable and durable real-time data streaming service. The throughput of an Amazon Kinesis data stream is designed to scale without limits via increasing the number of shards within a data stream. With Amazon Kinesis Data Streams, you can scale up to a sufficient number of shards (note, however, that you'll need

to provision enough shards ahead of time). As it requires manual administration of shards, it's not the correct choice for the given use-case.

Amazon EMR - Amazon EMR is the industry-leading cloud big data platform for processing vast amounts of data using open source tools such as Apache Spark, Apache Hive, Apache HBase, Apache Flink, Apache Hudi, and Presto. With EMR you can run Petabyte-scale analysis at less than half of the cost of traditional on-premises solutions and over 3x faster than standard Apache Spark. Amazon EMR uses Hadoop, an open-source framework, to distribute your data and processing across a resizable cluster of Amazon EC2 instances.

Using an EMR cluster would imply managing the underlying infrastructure so it’s ruled out.

AWS Lambda - AWS Lambda lets you run code without provisioning or managing servers. It cannot be used for production-grade serverless log analytics.

Reference:

https://aws.amazon.com/kinesis/data-firehose/

Pregunta 22:
A silicon valley based startup has a content management application with the web-tier running on EC2 instances and the database tier running on Amazon Aurora. Currently, the entire infrastructure is located in us-east-1 region. The startup has 90% of its customers in the US and Europe. The engineering team is getting reports of deteriorated application performance from customers in Europe with high application load time.

As a solutions architect, which of the following would you recommend addressing these performance issues? (Select two)

- **Create Amazon Aurora read replicas in the eu-west-1 region(Correcto)**
- **Create Amazon Aurora Multi-AZ standby instance in the eu-west-1 region**
- **Setup another fleet of EC2 instances for the web tier in the eu-west-1 region. Enable failover routing policy in Route 53**
- **Setup another fleet of EC2 instances for the web tier in the eu-west-1 region. Enable latency routing policy in Route 53(Correcto)**
- **Setup another fleet of EC2 instances for the web tier in the eu-west-1 region. Enable geolocation routing policy in Route 53**

Explicación
Correct options:

Setup another fleet of EC2 instances for the web tier in the eu-west-1 region. Enable latency routing policy in Route 53 - Amazon Route 53 is a highly available and scalable cloud Domain Name System (DNS) web service. Use latency based routing when you have resources in multiple AWS Regions and you want to route traffic to the region that provides the lowest latency. To use

latency-based routing, you create latency records for your resources in multiple AWS Regions. When Route 53 receives a DNS query for your domain or subdomain (example.com or acme.example.com), it determines which AWS Regions you've created latency records for, determines which region gives the user the lowest latency, and then selects a latency record for that region. Route 53 responds with the value from the selected record, such as the IP address for a web server.

As customers in Europe are facing performance issues with high application load time, you can use latency based routing to reduce the latency. Hence this is the correct option.

Route 53 Routing Policy Overview:

Choosing a routing policy

PDF | Kindle | RSS

When you create a record, you choose a routing policy, which determines how Amazon Route 53 responds to queries:

- **Simple routing policy** – Use for a single resource that performs a given function for your domain, for example, a web server that serves content for the example.com website.
- **Failover routing policy** – Use when you want to configure active-passive failover.
- **Geolocation routing policy** – Use when you want to route traffic based on the location of your users.
- **Geoproximity routing policy** – Use when you want to route traffic based on the location of your resources and, optionally, shift traffic from resources in one location to resources in another.
- **Latency routing policy** – Use when you have resources in multiple AWS Regions and you want to route traffic to the region that provides the best latency.
- **Multivalue answer routing policy** – Use when you want Route 53 to respond to DNS queries with up to eight healthy records selected at random.
- **Weighted routing policy** – Use to route traffic to multiple resources in proportions that you specify.

via - https://docs.aws.amazon.com/Route53/latest/DeveloperGuide/routing-policy.html

Create Amazon Aurora read replicas in the eu-west-1 region - Amazon Aurora is a MySQL and PostgreSQL-compatible relational database built for the cloud, that combines the performance and availability of traditional enterprise databases with the simplicity and cost-effectiveness of open source databases. Amazon Aurora features a distributed, fault-tolerant, self-healing storage system that auto-scales up to 64TB per database instance.

Amazon Aurora read replicas can be used to scale out reads across regions. This will improve the application performance for users in Europe. Therefore, this is also a correct option for the given use-case.

Incorrect options:

Setup another fleet of EC2 instances for the web tier in the eu-west-1 region. Enable geolocation routing policy in Route 53 - Geolocation routing lets you choose the resources that serve your traffic based on the geographic location of your users, meaning the location that DNS queries originate from. For example, you might want all queries from Europe to be routed to an ELB load balancer in the Frankfurt region. You can also use geolocation routing to restrict the distribution of content to only the locations in which you have distribution rights. You cannot use geolocation routing to reduce latency, hence this option is incorrect.

Setup another fleet of EC2 instances for the web tier in the eu-west-1 region. Enable failover routing policy in Route 53 - Failover routing lets you route traffic to a resource when the resource is healthy or to a different resource when the first resource is unhealthy. The primary and secondary records can route traffic to anything from an Amazon S3 bucket that is configured as a website to a complex tree of records. You cannot use failover routing to reduce latency, hence this option is incorrect.

Create Amazon Aurora Multi-AZ standby instance in the eu-west-1 region - Amazon Aurora Multi-AZ enhances the availability and durability for the database, it does not help in read scaling, so it is not a correct option for the given use-case.

References:

https://docs.aws.amazon.com/Route53/latest/DeveloperGuide/routing-policy.html

https://aws.amazon.com/blogs/aws/new-cross-region-read-replicas-for-amazon-aurora/

Pregunta 23:
A company has recently launched a new mobile gaming application that the users are adopting rapidly. The company uses RDS MySQL as the database. The engineering team wants an urgent solution to this issue where the rapidly increasing workload might exceed the available database storage.

As a solutions architect, which of the following solutions would you recommend so that it requires minimum development and systems administration effort to address this requirement?

- **Enable storage auto-scaling for RDS MySQL(Correcto)**
- **Create read replica for RDS MySQL**
- **Migrate RDS MySQL database to Aurora which offers storage auto-scaling**
- **Migrate RDS MySQL database to DynamoDB which automatically allocates storage space when required**

Explicación
Correct option:

Enable storage auto-scaling for RDS MySQL

If your workload is unpredictable, you can enable storage autoscaling for an Amazon RDS DB instance. With storage autoscaling enabled, when Amazon RDS detects that you are running out of free database space it automatically scales up your storage. Amazon RDS starts a storage modification for an autoscaling-enabled DB instance when these factors apply:

Free available space is less than 10 percent of the allocated storage.

The low-storage condition lasts at least five minutes.

At least six hours have passed since the last storage modification.

The maximum storage threshold is the limit that you set for autoscaling the DB instance. You can't set the maximum storage threshold for autoscaling-enabled instances to a value greater than the maximum allocated storage.

Incorrect options:

Migrate RDS MySQL to Aurora which offers storage auto-scaling - Although Aurora offers automatic storage scaling, this option is ruled out since it involves significant systems administration effort to migrate from RDS MySQL to Aurora. It is much easier to just enable storage auto-scaling for RDS MySQL.

Migrate RDS MySQL database to DynamoDB which automatically allocates storage space when required - This option is ruled out since DynamoDB is a NoSQL database which implies significant development effort to change the application logic to connect and query data from the underlying database. It is much easier to just enable storage auto-scaling for RDS MySQL.

Create read replica for RDS MySQL - Read replicas make it easy to take advantage of supported engines' built-in replication functionality to elastically scale out beyond the capacity constraints of a single DB instance for read-heavy database workloads. You can create multiple read replicas for a given source DB Instance and distribute your application's read traffic amongst them. This option acts as a distractor as read replicas cannot help to automatically scale storage for the primary database.

Reference:

https://docs.aws.amazon.com/AmazonRDS/latest/UserGuide/USER_PIOPS.StorageTypes.html

Pregunta 24:
An e-commerce application uses an Amazon Aurora Multi-AZ deployment for its database. While analyzing the performance metrics, the engineering team has found that the database reads are causing high I/O and adding latency to the write requests against the database.

As an AWS Certified Solutions Architect Associate, what would you recommend to separate the read requests from the write requests?

- **Provision another Amazon Aurora database and link it to the primary database as a read replica**
- **Activate read-through caching on the Amazon Aurora database**
- **Set up a read replica and modify the application to use the appropriate endpoint(Correcto)**
- **Configure the application to read from the Multi-AZ standby instance**

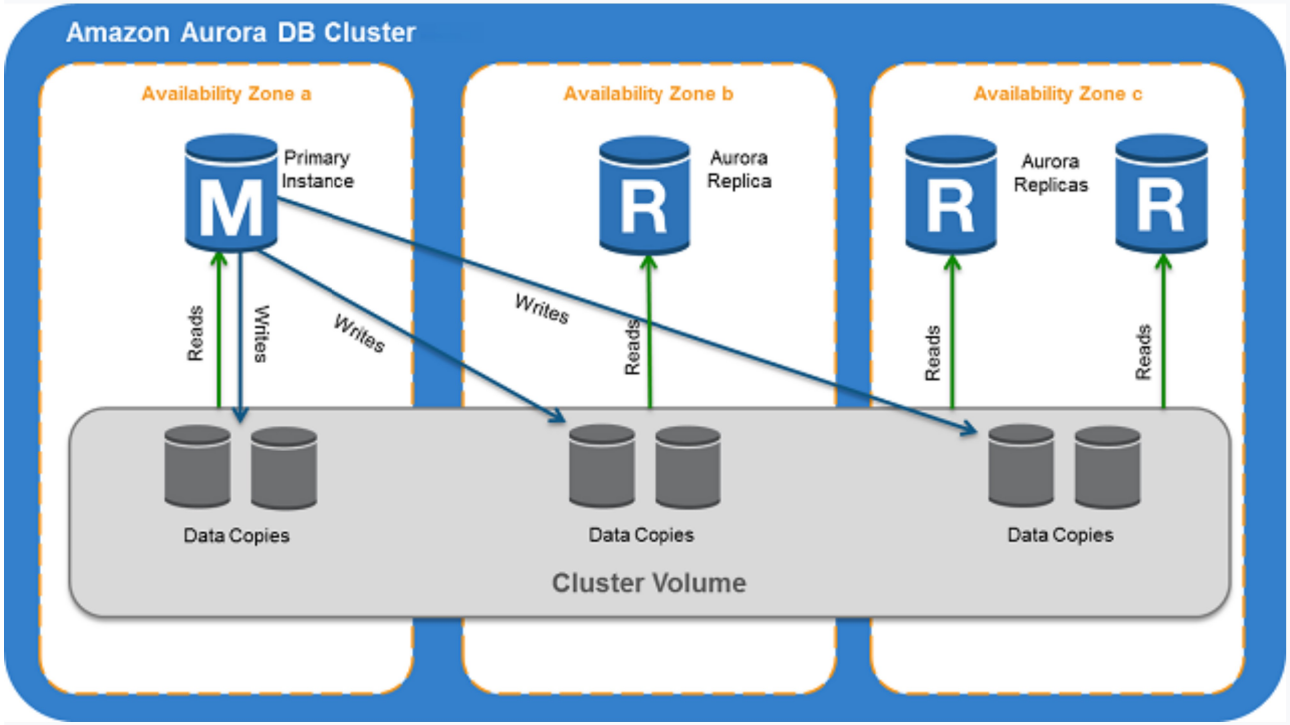
Explicación
Correct option:

Set up a read replica and modify the application to use the appropriate endpoint

An Amazon Aurora DB cluster consists of one or more DB instances and a cluster volume that manages the data for those DB instances. An Aurora cluster volume is a virtual database storage volume that spans multiple Availability Zones, with each Availability Zone having a copy of the DB cluster data. Two types of DB instances make up an Aurora DB cluster:

Primary DB instance – Supports read and write operations, and performs all of the data modifications to the cluster volume. Each Aurora DB cluster has one primary DB instance.

Aurora Replica – Connects to the same storage volume as the primary DB instance and supports only read operations. Each Aurora DB cluster can have up to 15 Aurora Replicas in addition to the primary DB instance. Aurora automatically fails over to an Aurora Replica in case the primary DB instance becomes unavailable. You can specify the failover priority for Aurora Replicas. Aurora Replicas can also offload read workloads from the primary DB instance.



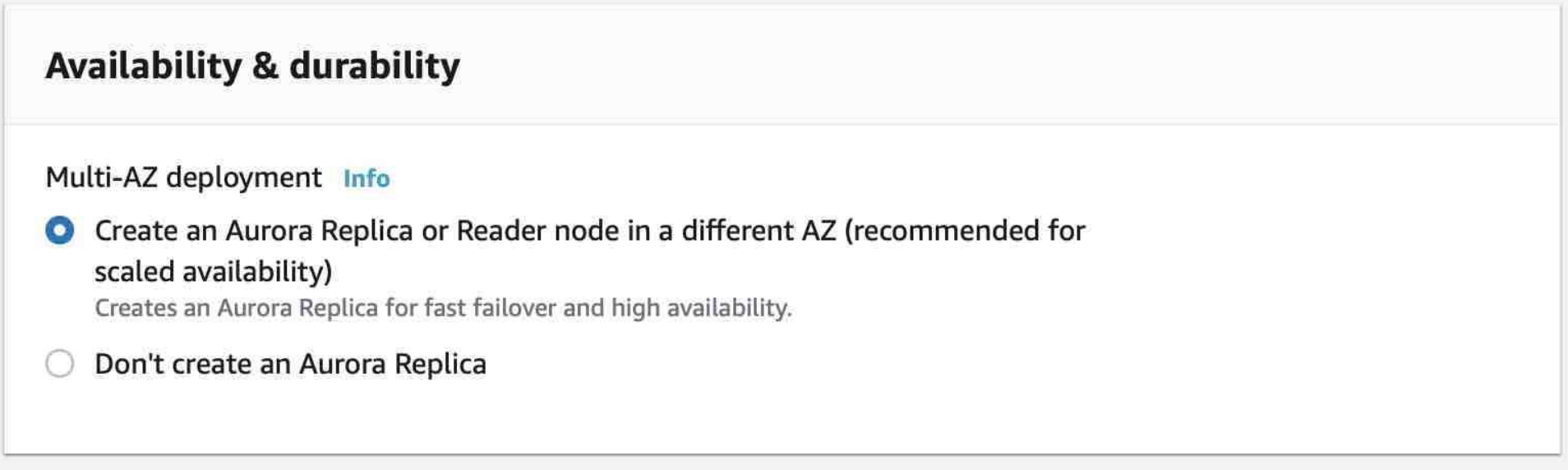
via - https://docs.aws.amazon.com/AmazonRDS/latest/AuroraUserGuide/Aurora.Overview.html

Aurora Replicas have two main purposes. You can issue queries to them to scale the read operations for your application. You typically do so by connecting to the reader endpoint of the cluster. That way, Aurora can spread the load for read-only connections

across as many Aurora Replicas as you have in the cluster. Aurora Replicas also help to increase availability. If the writer instance in a cluster becomes unavailable, Aurora automatically promotes one of the reader instances to take its place as the new writer.

While setting up a Multi-AZ deployment for Aurora, you create an Aurora replica or reader node in a different AZ.

Multi-AZ for Aurora:



You use the reader endpoint for read-only connections for your Aurora cluster. This endpoint uses a load-balancing mechanism to help your cluster handle a query-intensive workload. The reader endpoint is the endpoint that you supply to applications that do reporting or other read-only operations on the cluster. The reader endpoint load-balances connections to available Aurora Replicas in an Aurora DB cluster.

Types of Aurora endpoints

An endpoint is represented as an Aurora-specific URL that contains a host address and a port. The following types of endpoints are available from an Aurora DB cluster.

Cluster endpoint

A *cluster endpoint* (or *writer endpoint*) for an Aurora DB cluster connects to the current primary DB instance for that DB cluster. This endpoint is the only one that can perform write operations such as DDL statements. Because of this, the cluster endpoint is the one that you connect to when you first set up a cluster or when your cluster only contains a single DB instance.

Each Aurora DB cluster has one cluster endpoint and one primary DB instance.

You use the cluster endpoint for all write operations on the DB cluster, including inserts, updates, deletes, and DDL changes. You can also use the cluster endpoint for read operations, such as queries.

The cluster endpoint provides failover support for read/write connections to the DB cluster. If the current primary DB instance of a DB cluster fails, Aurora automatically fails over to a new primary DB instance. During a failover, the DB cluster continues to serve connection requests to the cluster endpoint from the new primary DB instance, with minimal interruption of service.

The following example illustrates a cluster endpoint for an Aurora MySQL DB cluster.

```
mydbcluster.cluster-123456789012.us-east-1.rds.amazonaws.com:3306
```

Reader endpoint

A *reader endpoint* for an Aurora DB cluster provides load-balancing support for read-only connections to the DB cluster. Use the reader endpoint for read operations, such as queries. By processing those statements on the read-only Aurora Replicas, this endpoint reduces the overhead on the primary instance. It also helps the cluster to scale the capacity to handle simultaneous SELECT queries, proportional to the number of Aurora Replicas in the cluster. Each Aurora DB cluster has one reader endpoint.

If the cluster contains one or more Aurora Replicas, the reader endpoint load-balances each connection request among the Aurora Replicas. In that case, you can only perform read-only statements such as SELECT in that session. If the cluster only contains a primary instance and no Aurora Replicas, the reader endpoint connects to the primary instance. In that case, you can perform write operations through the endpoint.

The following example illustrates a reader endpoint for an Aurora MySQL DB cluster.

```
mydbcluster.cluster-ro-123456789012.us-east-1.rds.amazonaws.com:3306
```

Custom endpoint

A *custom endpoint* for an Aurora cluster represents a set of DB instances that you choose. When you connect to the endpoint, Aurora performs load balancing and chooses one of the instances in the group to handle the connection. You define which instances this endpoint refers to, and you decide what purpose the endpoint serves.

An Aurora DB cluster has no custom endpoints until you create one. You can create up to five custom endpoints for each provisioned Aurora cluster. You can't use custom endpoints for Aurora Serverless clusters.

The custom endpoint provides load-balanced database connections based on criteria other than the read-only or read/write capability of the DB instances. For example, you might define a custom endpoint to connect to instances that use a particular AWS instance class or a particular DB parameter group. Then you might tell particular groups of users about this custom endpoint. For example, you might direct internal users to low-capacity instances for report generation or ad hoc (one-time) querying, and direct production traffic to high-capacity instances.

Because the connection can go to any DB instance that is associated with the custom endpoint, we recommend that you make sure that all the DB instances within that group share some similar characteristic. Doing so ensures that the performance, memory capacity, and so on, are consistent for everyone who connects to that endpoint.

This feature is intended for advanced users with specialized kinds of workloads where it isn't practical to keep all the Aurora Replicas in the cluster identical. With custom endpoints, you can predict the capacity of the DB instance used for each connection. When you use custom endpoints, you typically don't use the reader endpoint for that cluster.

The following example illustrates a custom endpoint for a DB instance in an Aurora MySQL DB cluster.

```
myendpoint.cluster-custom-123456789012.us-east-1.rds.amazonaws.com:3306
```

Instance endpoint

An *instance endpoint* connects to a specific DB instance within an Aurora cluster. Each DB instance in a DB cluster has its own unique instance endpoint. So there is one instance endpoint for the current primary DB instance of the DB cluster, and there is one instance endpoint for each of the Aurora Replicas in the DB cluster.

The instance endpoint provides direct control over connections to the DB cluster, for scenarios where using the cluster endpoint or reader endpoint might not be appropriate. For example, your client application might require more fine-grained load balancing based on workload type. In this case, you can configure multiple clients to connect to different Aurora Replicas in a DB cluster to distribute read workloads. For an example that uses instance endpoints to improve connection speed after a failover for Aurora PostgreSQL, see Fast failover with Amazon Aurora PostgreSQL. For an example that uses instance endpoints to improve connection speed after a failover for Aurora MySQL, see MariaDB Connector/J failover support – case Amazon Aurora.

The following example illustrates an instance endpoint for a DB instance in an Aurora MySQL DB cluster.

```
mydbinstance.123456789012.us-east-1.rds.amazonaws.com:3306
```

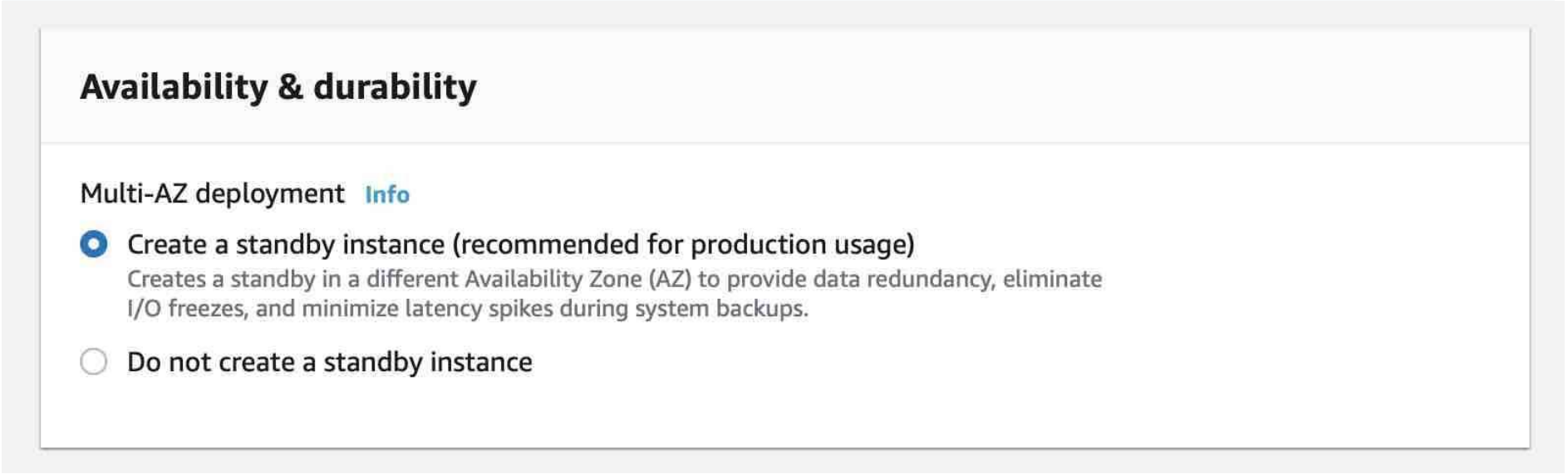
via - https://docs.aws.amazon.com/AmazonRDS/latest/AuroraUserGuide/Aurora.Overview.Endpoints.html

Incorrect options:

Provision another Amazon Aurora database and link it to the primary database as a read replica - You cannot provision another Aurora database and then link it as a read-replica for the primary database. This option is ruled out.

Configure the application to read from the Multi-AZ standby instance - This option has been added as a distractor as Aurora does not have any entity called standby instance. You create a standby instance while setting up a Multi-AZ deployment for RDS and NOT for Aurora.

Multi-AZ for RDS:



Activate read-through caching on the Amazon Aurora database - Aurora does not have built-in support for read-through caching, so this option just serves as a distractor. To implement caching, you will need to integrate something like ElastiCache and that would need code changes for the application.

References:

https://docs.aws.amazon.com/AmazonRDS/latest/AuroraUserGuide/Aurora.Overview.html

https://docs.aws.amazon.com/AmazonRDS/latest/AuroraUserGuide/Concepts.AuroraHighAvailability.html

https://docs.aws.amazon.com/AmazonRDS/latest/AuroraUserGuide/Aurora.Overview.Endpoints.html

Pregunta 25:
An IT company has built a solution wherein a Redshift cluster writes data to an Amazon S3 bucket belonging to a different AWS account. However, it is found that the files created in the S3 bucket using the UNLOAD command from the Redshift cluster are not even accessible to the S3 bucket owner.

What could be the reason for this denial of permission for the bucket owner?

- **When two different AWS accounts are accessing an S3 bucket, both the accounts must share the bucket policies. An erroneous policy can lead to such permission failures**
- **When objects are uploaded to S3 bucket from a different AWS account, the S3 bucket owner will get implicit permissions to access these objects. This issue seems to be due to an upload error that can be fixed by providing manual access from AWS console**
- **By default, an S3 object is owned by the AWS account that uploaded it. So the S3 bucket owner will not implicitly have access to the objects written by the Redshift cluster(Correcto)**
- **The owner of an S3 bucket has implicit access to all objects in his bucket. Permissions are set on objects after they are completely copied to the target location. Since the owner is unable to access the uploaded files, the write operation may be still in progress**

Explicación
Correct option:

By default, an S3 object is owned by the AWS account that uploaded it. So the S3 bucket owner will not implicitly have access to the objects written by Redshift cluster - By default, an S3 object is owned by the AWS account that uploaded it. This is true even when the bucket is owned by another account. Because the Amazon Redshift data files from the UNLOAD command were put into your bucket by another account, you (the bucket owner) don't have default permission to access those files.

To get access to the data files, an AWS Identity and Access Management (IAM) role with cross-account permissions must run the UNLOAD command again. Follow these steps to set up the Amazon Redshift cluster with cross-account permissions to the bucket:

1. From the account of the S3 bucket, create an IAM role (Bucket Role) with permissions to the bucket.
2. From the account of the Amazon Redshift cluster, create another IAM role (Cluster Role) with permissions to assume the Bucket Role.
3. Update the Bucket Role to grant bucket access and create a trust relationship with the Cluster Role.
4. From the Amazon Redshift cluster, run the UNLOAD command using the Cluster Role and Bucket Role.

This solution doesn't apply to Amazon Redshift clusters or S3 buckets that use server-side encryption with AWS Key Management Service (AWS KMS).

Incorrect options:

When objects are uploaded to S3 bucket from a different AWS account, the S3 bucket owner will get implicit permissions to access these objects. This issue seems to be due to an upload error that can be fixed by providing manual access from AWS console - By default, an S3 object is owned by the AWS account that uploaded it. So, the bucket owner will not have any default permissions on the objects. Therefore, this option is incorrect.

The owner of an S3 bucket has implicit access to all objects in his bucket. Permissions are set on objects after they are completely copied to the target location. Since the owner is unable to access the uploaded files, the write operation may be still in progress - This is an incorrect statement, given only as a distractor.

When two different AWS accounts are accessing an S3 bucket, both the accounts must share the bucket policies. An erroneous policy can lead to such permission failures - This is an incorrect statement, given only as a distractor.

Reference:

https://aws.amazon.com/premiumsupport/knowledge-center/s3-access-denied-redshift-unload/

Pregunta 26:
You are establishing a monitoring solution for desktop systems, that will be sending telemetry data into AWS every 1 minute. Data for each system must be processed in order, independently, and you would like to scale the number of consumers to be possibly equal to the number of desktop systems that are being monitored.

What do you recommend?

- **Use an SQS standard queue, and send the telemetry data as is**
- **Use an SQS FIFO queue, and send the telemetry data as is**
- **Use a Kinesis Data Stream, and send the telemetry data with a Partition ID that uses the value of the Desktop ID**
- **Use an SQS FIFO queue, and make sure the telemetry data is sent with a Group ID attribute representing the value of the Desktop ID(Correcto)**

Explicación
Correct option:

Use an SQS FIFO queue, and make sure the telemetry data is sent with a Group ID attribute representing the value of the Desktop ID

Amazon Simple Queue Service (SQS) is a fully managed message queuing service that enables you to decouple and scale microservices, distributed systems, and serverless applications. SQS offers two types of message queues. Standard queues offer maximum throughput, best-effort ordering, and at-least-once delivery. SQS FIFO queues are designed to guarantee that messages are processed exactly once, in the exact order that they are sent.

We, therefore, need to use an SQS FIFO queue. If we don't specify a GroupID, then all the messages are in absolute order, but we can only have 1 consumer at most. To allow for multiple consumers to read data for each Desktop application, and to scale the number of consumers, we should use the "Group ID" attribute. So this is the correct option.

Incorrect options:

Use an SQS FIFO queue, and send the telemetry data as is - This is incorrect because if we send the telemetry data as is then we will not be able to scale the number of consumers to be equal to the number of desktop systems. In this case, each message will have its consumer. So we should use the "Group ID" attribute so that multiple consumers can read data for each Desktop application.

Use an SQS standard queue, and send the telemetry data as is - An SQS standard queue has no ordering capability so that's ruled out.

Use a Kinesis Data Stream, and send the telemetry data with a Partition ID that uses the value of the Desktop ID - Amazon Kinesis Data Streams (KDS) is a massively scalable and durable real-time data streaming service. KDS can continuously capture gigabytes of data per second from hundreds of thousands of sources such as website clickstreams, database event streams, financial transactions, social media feeds, IT logs, and location-tracking events. A Kinesis Data Stream would work and would give us the data for each desktop application within shards, but we can only have as many consumers as shards in Kinesis (which is in practice, much less than the number of producers).

References:

https://aws.amazon.com/blogs/compute/solving-complex-ordering-challenges-with-amazon-sqs-fifo-queues/

https://aws.amazon.com/sqs/faqs/

https://aws.amazon.com/kinesis/data-streams/faqs/

Pregunta 27:
A junior DevOps engineer wants to change the default configuration for EBS volume termination. By default, the root volume of an EC2 instance for an EBS-backed AMI is deleted when the instance terminates.

Which option below helps change this default behavior to ensure that the volume persists even after the instance terminates?

- **Set the DeleteOnTermination attribute to true**
- **Set the DeleteOnTermination attribute to false(Correcto)**
- **Set the TerminateOnDelete attribute to true**
- **Set the TerminateOnDelete attribute to false**

Explicación
Correct option:

Set the DeleteOnTermination attribute to false

An EC2 instance can be launched from either an instance store-backed AMI or an Amazon EBS-backed AMI. Instances that use Amazon EBS for the root device automatically have an Amazon EBS volume attached. By default, the root volume for an AMI backed by Amazon EBS is deleted when the instance terminates.
 The default behavior can be changed to ensure that the volume persists after the instance terminates. To change the default behavior, set the DeleteOnTermination attribute to false using a block device mapping.

Incorrect options:

Set the TerminateOnDelete attribute to true

Set the TerminateOnDelete attribute to false

Both these options are incorrect as there is no such attribute as TerminateOnDelete. These options have been added as distractors.

Set the DeleteOnTermination attribute to true - If you set the DeleteOnTermination attribute to true, then the root volume for an AMI backed by Amazon EBS would be deleted when the instance terminates. Therefore, this option is incorrect.

Reference:

https://docs.aws.amazon.com/AWSEC2/latest/UserGuide/RootDeviceStorage.html

Pregunta 28:
A financial services company has developed its flagship application on AWS Cloud with data security requirements such that the encryption key must be stored in a custom application running on-premises. The company wants to offload the data storage as well as the encryption process to Amazon S3 but continue to use the existing encryption key.

Which of the following S3 encryption options allows the company to leverage Amazon S3 for storing data with given constraints?

- **Client-Side Encryption with data encryption is done on the client-side before sending it to Amazon S3**
- **Server-Side Encryption with Amazon S3-Managed Keys (SSE-S3)**
- **Server-Side Encryption with Customer Master Keys (CMKs) Stored in AWS Key Management Service (SSE-KMS)**
- **Server-Side Encryption with Customer-Provided Keys (SSE-C)** **(Correcto)**

Explicación
Correct option:

Server-Side Encryption with Customer-Provided Keys (SSE-C)

You have the following options for protecting data at rest in Amazon S3:

Server-Side Encryption – Request Amazon S3 to encrypt your object before saving it on disks in its data centers and then decrypt it when you download the objects.

Client-Side Encryption – Encrypt data client-side and upload the encrypted data to Amazon S3. In this case, you manage the encryption process, the encryption keys, and related tools.

For the given use-case, the company wants to manage the encryption keys via its custom application and let S3 manage the encryption, therefore you must use Server-Side Encryption with Customer-Provided Keys (SSE-C).

Please review these three options for Server Side Encryption on S3:

Protecting data using server-side encryption

PDF | Kindle | RSS

Server-side encryption is the encryption of data at its destination by the application or service that receives it. Amazon S3 encrypts your data at the object level as it writes it to disks in its data centers and decrypts it for you when you access it. As long as you authenticate your request and you have access permissions, there is no difference in the way you access encrypted or unencrypted objects. For example, if you share your objects using a presigned URL, that URL works the same way for both encrypted and unencrypted objects. Additionally, when you list objects in your bucket, the list API returns a list of all objects, regardless of whether they are encrypted.

> **Note**
>
> You can't apply different types of server-side encryption to the same object simultaneously.

You have three mutually exclusive options, depending on how you choose to manage the encryption keys.

Server-Side Encryption with Amazon S3-Managed Keys (SSE-S3)

When you use Server-Side Encryption with Amazon S3-Managed Keys (SSE-S3), each object is encrypted with a unique key. As an additional safeguard, it encrypts the key itself with a master key that it regularly rotates. Amazon S3 server-side encryption uses one of the strongest block ciphers available, 256-bit Advanced Encryption Standard (AES-256), to encrypt your data. For more information, see Protecting Data Using Server-Side Encryption with Amazon S3-Managed Encryption Keys (SSE-S3).

Server-Side Encryption with Customer Master Keys (CMKs) Stored in AWS Key Management Service (SSE-KMS)

Server-Side Encryption with Customer Master Keys (CMKs) Stored in AWS Key Management Service (SSE-KMS) is similar to SSE-S3, but with some additional benefits and charges for using this service. There are separate permissions for the use of a CMK that provides added protection against unauthorized access of your objects in Amazon S3. SSE-KMS also provides you with an audit trail that shows when your CMK was used and by whom. Additionally, you can create and manage customer managed CMKs or use AWS managed CMKs that are unique to you, your service, and your Region. For more information, see Protecting Data Using Server-Side Encryption with CMKs Stored in AWS Key Management Service (SSE-KMS).

Server-Side Encryption with Customer-Provided Keys (SSE-C)

With Server-Side Encryption with Customer-Provided Keys (SSE-C), you manage the encryption keys and Amazon S3 manages the encryption, as it writes to disks, and decryption, when you access your objects. For more information, see Protecting data using server-side encryption with customer-provided encryption keys (SSE-C).

via - https://docs.aws.amazon.com/AmazonS3/latest/dev/serv-side-encryption.html

Incorrect options:

Server-Side Encryption with Amazon S3-Managed Keys (SSE-S3) - When you use Server-Side Encryption with Amazon S3-Managed Keys (SSE-S3), each object is encrypted with a unique key. As an additional safeguard, it encrypts the key itself with a master key that it regularly rotates. So this option is incorrect.

Server-Side Encryption with Customer Master Keys (CMKs) Stored in AWS Key Management Service (SSE-KMS) - Server-Side Encryption with Customer Master Keys (CMKs) stored in AWS Key Management Service (SSE-KMS) is similar to SSE-S3. SSE-KMS provides you with an audit trail that shows when your CMK was used and by whom. Additionally, you can create and manage customer-managed CMKs or use AWS managed CMKs that are unique to you, your service, and your Region.

Client-Side Encryption with data encryption is done on the client-side before sending it to Amazon S3 - You can encrypt the data client-side and upload the encrypted data to Amazon S3. In this case, you manage the encryption process, the encryption keys, and related tools.

Reference:

https://docs.aws.amazon.com/AmazonS3/latest/dev/serv-side-encryption.html

Pregunta 29:
An application runs big data workloads on EC2 instances. The application runs 24x7 all round the year and needs at least 20 instances to maintain a minimum acceptable performance threshold and the application needs 300 instances to handle spikes in the workload. Based on historical workloads processed by the application, it needs 80 instances 80% of the time.

As a solutions architect, which of the following would you recommend as the MOST cost-optimal solution so that it can meet the workload demand in a steady state?

- **Purchase 80 spot instances. Use Auto Scaling Group to provision the remaining instances as on-demand instances per the workload demand**
- **Purchase 80 on-demand instances. Provision additional on-demand and spot instances per the workload demand (Use Auto Scaling Group with launch template to provision the mix of on-demand and spot instances)**
- **Purchase 80 reserved instances. Provision additional on-demand and spot instances per the workload demand (Use Auto Scaling Group with launch template to provision the mix of on-demand and spot instances)** **(Correcto)**
- **Purchase 20 on-demand instances. Use Auto Scaling Group to provision the remaining instances as spot instances per the workload demand**

Explicación
Correct option:

Purchase 80 reserved instances. Provision additional on-demand and spot instances per the workload demand (Use Auto Scaling Group with launch template to provision the mix of on-demand and spot instances)

As the steady-state workload demand is 80 instances, we can save on costs by purchasing 80 reserved instances. Based on additional workload demand, we can specify a mix of on-demand and spot instances using Application Load Balancer with a launch template to provision the mix of on-demand and spot instances.

Please see this detailed overview of various types of EC2 instances from a pricing perspective:

On-Demand

With On-Demand instances, you pay for compute capacity by the hour or the second depending on which instances you run. No longer-term commitments or upfront payments are needed. You can increase or decrease your compute capacity depending on the demands of your application and only pay the specified per hourly rates for the instance you use.

On-Demand instances are recommended for:

- Users that prefer the low cost and flexibility of Amazon EC2 without any up-front payment or long-term commitment
- Applications with short-term, spiky, or unpredictable workloads that cannot be interrupted
- Applications being developed or tested on Amazon EC2 for the first time

See On-Demand pricing »

Spot instances

Amazon EC2 Spot instances allow you to request spare Amazon EC2 computing capacity for up to 90% off the On-Demand price. Learn More.

Spot instances are recommended for:

- Applications that have flexible start and end times
- Applications that are only feasible at very low compute prices
- Users with urgent computing needs for large amounts of additional capacity

See Spot pricing »

Savings Plans

Savings Plans are a flexible pricing model that offer low prices on EC2 and Fargate usage, in exchange for a commitment to a consistent amount of usage (measured in $/hour) for a 1 or 3 year term.

Learn more »

Reserved Instances

Reserved Instances provide you with a significant discount (up to 75%) compared to On-Demand instance pricing. In addition, when Reserved Instances are assigned to a specific Availability Zone, they provide a capacity reservation, giving you additional confidence in your ability to launch instances when you need them.

For applications that have steady state or predictable usage, Reserved Instances can provide significant savings compared to using On-Demand instances. See How to Purchase Reserved Instances for more information.

Reserved Instances are recommended for:

- Applications with steady state usage
- Applications that may require reserved capacity
- Customers that can commit to using EC2 over a 1 or 3 year term to reduce their total computing costs

via - https://aws.amazon.com/ec2/pricing/

Incorrect options:

Purchase 20 on-demand instances. Use Auto Scaling Group to provision the remaining instances as spot instances per the workload demand - Provisioning 20 on-demand instances implies that there would be a shortfall of 60 instances 80% of the time. Provisioning all of these 60 instances as spot instances is highly risky as there is no guarantee regarding the availability of the spot instances, which means we may not even meet the steady-state requirement for the workload, so this option is incorrect.

Purchase 80 on-demand instances. Provision additional on-demand and spot instances per the workload demand (Use Auto Scaling Group with launch template to provision the mix of on-demand and spot instances) - Provisioning 80 on-demand instances would end up costlier than the option where we provision 80 reserved instances. So this option is ruled out.

Purchase 80 spot instances. Use Auto Scaling Group to provision the remaining instances as on-demand instances per the workload demand - The option to purchase 80 spot instances is incorrect, as there is no guarantee regarding the availability of the spot instances, which means we may not even meet the steady-state workload.

Reference:

https://aws.amazon.com/ec2/pricing/

Pregunta 30:
Your company has a monthly big data workload, running for about 2 hours, which can be efficiently distributed across multiple servers of various sizes, with a variable number of CPUs. The solution for the workload should be able to withstand server failures.

Which is the MOST cost-optimal solution for this workload?

- **Run the workload on a Spot Fleet(Correcto)**
- **Run the workload on Reserved Instances**
- **Run the workload on Dedicated Hosts**
- **Run the workload on Spot Instances**

Explicación
Correct option:

Run the workload on a Spot Fleet

The Spot Fleet selects the Spot Instance pools that meet your needs and launches Spot Instances to meet the target capacity for the fleet. By default, Spot Fleets are set to maintain target capacity by launching replacement instances after Spot Instances in the fleet are terminated.

A Spot Instance is an unused EC2 instance that is available for less than the On-Demand price. Spot Instances provide great cost efficiency, but we need to select an instance type in advance. In this case, we want to use the most cost-optimal option and leave the selection of the cheapest spot instance to a Spot Fleet request, which can be optimized with the `lowestPrice` strategy. So this is the correct option.

Key Spot Instance Concepts:

Concepts

Before you get started with Spot Instances, you should be familiar with the following concepts:

- *Spot Instance pool* – A set of unused EC2 instances with the same instance type (for example, `m5.large`), operating system, Availability Zone, and network platform.
- *Spot price* – The current price of a Spot Instance per hour.
- *Spot Instance request* – Provides the maximum price per hour that you are willing to pay for a Spot Instance. If you don't specify a maximum price, the default maximum price is the On-Demand price. When the maximum price per hour for your request exceeds the Spot price, Amazon EC2 fulfills your request if capacity is available. A Spot Instance request is either *one-time* or *persistent*. Amazon EC2 automatically resubmits a persistent Spot request after the Spot Instance associated with the request is terminated. Your Spot Instance request can optionally specify a duration for the Spot Instances.
- *Spot Fleet* – A set of Spot Instances that is launched based on criteria that you specify. The Spot Fleet selects the Spot Instance pools that meet your needs and launches Spot Instances to meet the target capacity for the fleet. By default, Spot Fleets are set to *maintain* target capacity by launching replacement instances after Spot Instances in the fleet are terminated. You can submit a Spot Fleet as a one-time *request*, which does not persist after the instances have been terminated. You can include On-Demand Instance requests in a Spot Fleet request.
- *Spot Instance interruption* – Amazon EC2 terminates, stops, or hibernates your Spot Instance when the Spot price exceeds the maximum price for your request or capacity is no longer available. Amazon EC2 provides a Spot Instance interruption notice, which gives the instance a two-minute warning before it is interrupted.

via - https://docs.aws.amazon.com/AWSEC2/latest/UserGuide/using-spot-instances.html

Incorrect options:

Run the workload on Spot Instances - A Spot Instance is an unused EC2 instance that is available for less than the On-Demand price. Because Spot Instances enable you to request unused EC2 instances at steep discounts, you can lower your Amazon EC2 costs significantly. The hourly price for a Spot Instance is called a Spot price. Only spot fleets can maintain target capacity by launching replacement instances after Spot Instances in the fleet are terminated, so spot instances, by themselves, are not the right fit for this use-case.

Run the workload on Reserved Instances - Reserved Instances are less cost-optimized than Spot Instances, and most efficient when used continuously. Here the workload is once a month, so this is not efficient.

Run the workload on Dedicated Hosts - Amazon EC2 Dedicated Hosts allow you to use your eligible software licenses from vendors such as Microsoft and Oracle on Amazon EC2 so that you get the flexibility and cost-effectiveness of using your licenses, but with the resiliency, simplicity, and elasticity of AWS. An Amazon EC2 Dedicated Host is a physical server fully dedicated for your use, so you can help address corporate compliance requirement. They're not particularly cost-efficient. So this option is not correct.

References:

https://docs.aws.amazon.com/AWSEC2/latest/UserGuide/using-spot-instances.html

https://docs.aws.amazon.com/AWSEC2/latest/UserGuide/spot-fleet.html#spot-fleet-allocation-strategy

Pregunta 31:
A Hollywood studio is planning a series of promotional events leading up to the launch of the trailer of its next sci-fi thriller. The executives at the studio want to create a static website with lots of animations in line with the theme of the movie. The studio has hired you as a solutions architect to build a scalable serverless solution.

Which of the following represents the MOST cost-optimal and high-performance solution?

- **Build the website as a static website hosted on Amazon S3. Create a CloudFront distribution with Amazon S3 as the origin. Use Amazon Route 53 to create an alias record that points to your CloudFront distribution** **(Correcto)**
- **Host the website on AWS Lambda. Create a CloudFront distribution with Lambda as the origin**
- **Host the website on an EC2 instance. Create a CloudFront distribution with the EC2 instance as the custom origin**
- **Host the website on an instance in the studio's on-premises data center. Create a CloudFront distribution with this instance as the custom origin**

Explicación
Correct option:

Build the website as a static website hosted on Amazon S3. Create a CloudFront distribution with Amazon S3 as the origin. Use Amazon Route 53 to create an alias record that points to your CloudFront distribution

You can use Amazon S3 to host a static website. On a static website, individual web pages include static content. They might also contain client-side scripts. To host a static website on Amazon S3, you configure an Amazon S3 bucket for website hosting and then upload your website content to the bucket.

Amazon CloudFront is a fast content delivery network (CDN) service that securely delivers data, videos, applications, and APIs to customers globally with low latency, high transfer speeds, all within a developer-friendly environment.

You can use Amazon CloudFront to improve the performance of your website. CloudFront makes your website files (such as HTML, images, and video) available from data centers around the world (called edge locations). When a visitor requests a file from your website, CloudFront automatically redirects the request to a copy of the file at the nearest edge location. This results in faster download times than if the visitor had requested the content from a data center that is located farther away. Therefore, this option is correct.

Hosting a static website on Amazon S3:

Hosting a static website on Amazon S3

PDF | Kindle | RSS

You can use Amazon S3 to host a static website. On a *static* website, individual webpages include static content. They might also contain client-side scripts.

By contrast, a *dynamic* website relies on server-side processing, including server-side scripts such as PHP, JSP, or ASP.NET. Amazon S3 does not support server-side scripting, but AWS has other resources for hosting dynamic websites. To learn more about website hosting on AWS, see Web Hosting.

To configure your bucket for static website hosting, you can use the AWS Management Console without writing any code. You can also create, update, and delete the website configuration *programmatically* by using the AWS SDKs. The SDKs provide wrapper classes around the Amazon S3 REST API. If your application requires it, you can send REST API requests directly from your application.

To host a static website on Amazon S3, you configure an Amazon S3 bucket for website hosting and then upload your website content to the bucket. When you configure a bucket as a static website, you must enable website hosting, set permissions, and create and add an index document. Depending on your website requirements, you can also configure redirects, web traffic logging, and a custom error document.

After you configure your bucket as a static website, you can access the bucket through the AWS Region-specific Amazon S3 website endpoints for your bucket. Website endpoints are different from the endpoints where you send REST API requests. For more information, see Website endpoints.

via - https://docs.aws.amazon.com/AmazonS3/latest/dev/WebsiteHosting.html

Incorrect options:

Host the website on AWS Lambda. Create a CloudFront distribution with Lambda as the origin

With AWS Lambda, you can run code without provisioning or managing servers. You can't host a website on Lambda. Also, you can't have CloudFront in front of Lambda. So this option is incorrect.

Host the website on an EC2 instance. Create a CloudFront distribution with the EC2 instance as the custom origin

Host the website on an instance in the studio's on-premises data center. Create a CloudFront distribution with this instance as the custom origin

Hosting the website on an EC2 instance or a data-center specific instance is ruled out as the studio wants a serverless solution. So both these options are incorrect.

References:

https://docs.aws.amazon.com/AmazonS3/latest/dev/WebsiteHosting.html

https://docs.aws.amazon.com/AmazonS3/latest/dev/website-hosting-custom-domain-walkthrough.html

https://docs.aws.amazon.com/AmazonS3/latest/dev/website-hosting-cloudfront-walkthrough.html

Pregunta 32:
A financial services company has deployed its flagship application on EC2 instances. Since the application handles sensitive customer data, the security team at the company wants to ensure that any third-party SSL/TLS certificates configured on EC2 instances via the AWS Certificate Manager (ACM) are renewed before their expiry date. The company has hired you as an AWS Certified Solutions Architect Associate to build a solution that notifies the security team 30 days before the certificate expiration. The solution should require the least amount of scripting and maintenance effort.

What will you recommend?

- **Leverage AWS Config managed rule to check if any SSL/TLS certificates created via ACM are marked for expiration within 30 days. Configure the rule to trigger an SNS notification to the security team if any certificate expires within 30 days**
- **Leverage AWS Config managed rule to check if any third-party SSL/TLS certificates imported into ACM are marked for expiration within 30 days. Configure the rule to trigger an SNS notification to the security team if any certificate expires within 30 days(Correcto)**
- **Monitor the `days to expiry` CloudWatch metric for certificates created via ACM. Create a CloudWatch alarm to monitor such certificates based on the `days to expiry` metric and then trigger a custom action of notifying the security team**
- **Monitor the `days to expiry` CloudWatch metric for certificates imported into ACM. Create a CloudWatch alarm to monitor such certificates based on the `days to expiry` metric and then trigger a custom action of notifying the security team**

Explicación
Correct option:

Leverage AWS Config managed rule to check if any third-party SSL/TLS certificates imported into ACM are marked for expiration within 30 days. Configure the rule to trigger an SNS notification to the security team if any certificate expires within 30 days

AWS Certificate Manager is a service that lets you easily provision, manage, and deploy public and private Secure Sockets Layer/Transport Layer Security (SSL/TLS) certificates for use with AWS services and your internal connected resources. SSL/TLS certificates are used to secure network communications and establish the identity of websites over the Internet as well as resources on private networks.

AWS Config provides a detailed view of the configuration of AWS resources in your AWS account. This includes how the resources are related to one another and how they were configured in the past so that you can see how the configurations and relationships change over time.

How AWS Config Works

PDF | RSS

When you turn on AWS Config, it first discovers the supported AWS resources that exist in your account and generates a configuration item for each resource.

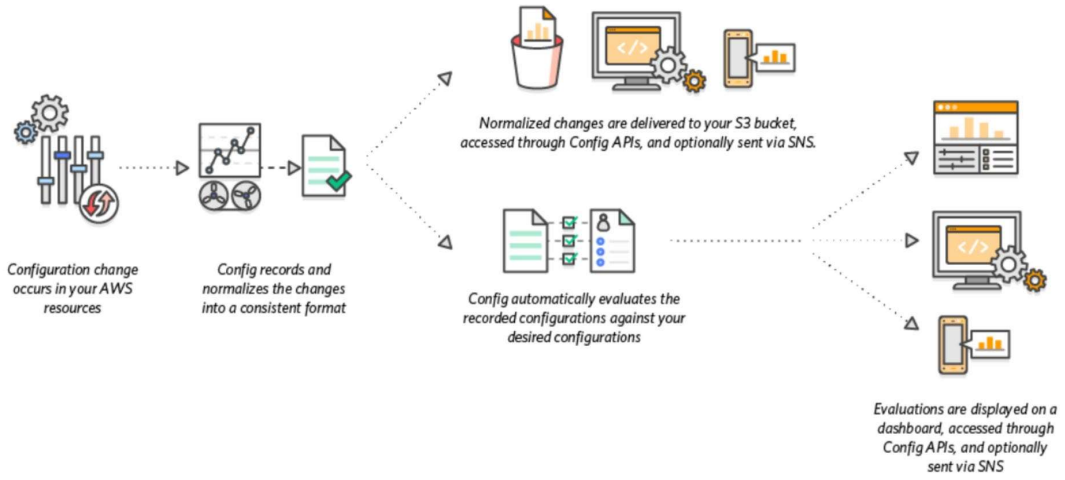
AWS Config also generates configuration items when the configuration of a resource changes, and it maintains historical records of the configuration items of your resources from the time you start the configuration recorder. By default, AWS Config creates configuration items for every supported resource in the region. If you don't want AWS Config to create configuration items for all supported resources, you can specify the resource types that you want it to track.

AWS Config keeps track of all changes to your resources by invoking the Describe or the List API call for each resource in your account. The service uses those same API calls to capture configuration details for all related resources.

For example, removing an egress rule from a VPC security group causes AWS Config to invoke a Describe API call on the security group. AWS Config then invokes a Describe API call on all of the instances associated with the security group. The updated configurations of the security group (the resource) and of each instance (the related resources) are recorded as configuration items and delivered in a configuration stream to an Amazon Simple Storage Service (Amazon S3) bucket.

AWS Config also tracks the configuration changes that were not initiated by the API. AWS Config examines the resource configurations periodically and generates configuration items for the configurations that have changed.

If you are using AWS Config rules, AWS Config continuously evaluates your AWS resource configurations for desired settings. Depending on the rule, AWS Config will evaluate your resources either in response to configuration changes or periodically. Each rule is associated with an AWS Lambda function, which contains the evaluation logic for the rule. When AWS Config evaluates your resources, it invokes the rule's AWS Lambda function. The function returns the compliance status of the evaluated resources. If a resource violates the conditions of a rule, AWS Config flags the resource and the rule as noncompliant. When the compliance status of a resource changes, AWS Config sends a notification to your Amazon SNS topic.



via - https://docs.aws.amazon.com/config/latest/developerguide/how-does-config-work.html

AWS Config provides AWS-managed rules, which are predefined, customizable rules that AWS Config uses to evaluate whether your AWS resources comply with common best practices. You can leverage an AWS Config managed rule to check if any ACM certificates in your account are marked for expiration within the specified number of days. Certificates provided by ACM are automatically renewed. ACM does not automatically renew the certificates that you import. The rule is NON_COMPLIANT if your certificates are about to expire.

acm-certificate-expiration-check

PDF | RSS

Checks if AWS Certificate Manager Certificates in your account are marked for expiration within the specified number of days. Certificates provided by ACM are automatically renewed. ACM does not automatically renew certificates that you import. The rule is NON_COMPLIANT if your certificates are about to expire.

Identifier: ACM_CERTIFICATE_EXPIRATION_CHECK

Trigger type: Configuration changes

AWS Region: All supported AWS regions except China (Beijing), China (Ningxia), Asia Pacific (Osaka), Europe (Milan) Region

Parameters:

daysToExpiration (Optional)
Type: int
Default: 14
Specify the number of days before the rule flags the ACM Certificate as noncompliant.

via - https://docs.aws.amazon.com/config/latest/developerguide/how-does-config-work.html

You can configure AWS Config to stream configuration changes and notifications to an Amazon SNS topic. For example, when a resource is updated, you can get a notification sent to your email, so that you can view the changes. You can also be notified when AWS Config evaluates your custom or managed rules against your resources.

Incorrect options:

Monitor the `days to expiry` CloudWatch metric for certificates imported into ACM. Create a CloudWatch alarm to monitor such certificates based on the `days to expiry` metric and then trigger a custom action of notifying the security team - AWS Certificate Manager (ACM) does not attempt to renew third-party certificates that are imported. Also, an administrator needs to reconfigure missing DNS records for certificates that use DNS validation if the record was removed for any reason after the certificate was issued. Metrics and events provide you visibility into such certificates that require intervention to continue the renewal process. Amazon CloudWatch metrics and Amazon EventBridge events are enabled for all certificates that are managed by ACM. Users can monitor `days to expiry` as a metric for ACM certificates through Amazon CloudWatch. An Amazon EventBridge expiry event is published for any certificate that is at least 45 days away from expiry by default. Users can build alarms to monitor certificates based on days to expiry and also trigger custom actions such as calling a Lambda function or paging an administrator.

It is certainly possible to use the `days to expiry` CloudWatch metric to build a CloudWatch alarm to monitor the imported ACM certificates. The alarm will, in turn, trigger a notification to the security team. But this option needs more configuration effort than directly using the AWS Config managed rule that is available off-the-shelf.

Leverage AWS Config managed rule to check if any SSL/TLS certificates created via ACM are marked for expiration within 30 days. Configure the rule to trigger an SNS notification to the security team if any certificate expires within 30 days

Monitor the `days to expiry` CloudWatch metric for certificates created via ACM. Create a CloudWatch alarm to monitor such certificates based on the `days to expiry` metric and then trigger a custom action of notifying the security team

Any SSL/TLS certificates created via ACM do not need any monitoring/intervention for expiration. ACM automatically renews such certificates. Hence both these options are incorrect.

References:

https://docs.aws.amazon.com/config/latest/developerguide/WhatIsConfig.html

https://docs.aws.amazon.com/config/latest/developerguide/how-does-config-work.html

https://docs.aws.amazon.com/config/latest/developerguide/evaluate-config.html

https://docs.aws.amazon.com/config/latest/developerguide/acm-certificate-expiration-check.html

https://aws.amazon.com/blogs/security/how-to-monitor-expirations-of-imported-certificates-in-aws-certificate-manager-acm/

Pregunta 33:
A silicon valley based startup has a two-tier architecture using EC2 instances for its flagship application. The web servers (listening on port 443), which have been assigned security group A, are in public subnets across two Availability Zones and the MSSQL based database instances (listening on port 1433), which have been assigned security group B, are in two private subnets across two Availability Zones. The DevOps team wants to review the security configurations of the application architecture.

As a solutions architect, which of the following options would you select as the MOST secure configuration? (Select two)

- **For security group A: Add an inbound rule that allows traffic from all sources on port 443. Add an outbound rule with the destination as security group B on port 443**
- **For security group B: Add an inbound rule that allows traffic only from security group A on port 1433(Correcto)**
- **For security group A: Add an inbound rule that allows traffic from all sources on port 443. Add an outbound rule with the destination as security group B on port 1433(Correcto)**
- **For security group B: Add an inbound rule that allows traffic only from all sources on port 1433**
- **For security group B: Add an inbound rule that allows traffic only from security group A on port 443**

Explicación
Correct options:

For security group A: Add an inbound rule that allows traffic from all sources on port 443. Add an outbound rule with the destination as security group B on port 1433

For security group B: Add an inbound rule that allows traffic only from security group A on port 1433

A security group acts as a virtual firewall that controls the traffic for one or more instances. When you launch an instance, you can specify one or more security groups; otherwise, we use the default security group. You can add rules to each security group that allows traffic to or from its associated instances. You can modify the rules for a security group at any time; the new rules are automatically applied to all instances that are associated with the security group. When we decide whether to allow traffic to reach an instance, we evaluate all the rules from all the security groups that are associated with the instance.

The following are the characteristics of security group rules:

By default, security groups allow all outbound traffic.

Security group rules are always permissive; you can't create rules that deny access.

Security groups are stateful

The MOST secure configuration for the given use case is:

For security group A: Add an inbound rule that allows traffic from all sources on port 443. Add an outbound rule with the destination as security group B on port 1433

The above rules make sure that web servers are listening for traffic on all sources on the HTTPS protocol on port 443. The web servers only allow outbound traffic to MSSQL servers in Security Group B on port 1433.

For security group B: Add an inbound rule that allows traffic only from security group A on port 1433. The above rule makes sure that the MSSQL servers only accept traffic from web servers in security group A on port 1433.

Therefore, both of these options are correct.

Incorrect options:

For security group A: Add an inbound rule that allows traffic from all sources on port 443. Add an outbound rule with the destination as security group B on port 443 - As the MSSQL based database instances are listening on port 1433, therefore for security group A, the outbound rule should be added on port 443 with the destination as security group B.

For security group B: Add an inbound rule that allows traffic only from all sources on port 1433 - The inbound rule should allow traffic only from security group A on port 1433. Allowing traffic from all sources will compromise security.

For security group B: Add an inbound rule that allows traffic only from security group A on port 443 - The inbound rule should allow traffic only from security group A on port 1433 because the MSSQL based database instances are listening on port 1433.

Reference:

https://docs.aws.amazon.com/AWSEC2/latest/UserGuide/ec2-security-groups.html

Pregunta 34:
An IT company wants to optimize the costs incurred on its fleet of 100 EC2 instances for the next year. Based on historical analyses, the engineering team observed that 70 of these instances handle the compute services of its flagship application and need to be always available. The other 30 instances are used to handle batch jobs that can afford a delay in processing.

As a solutions architect, which of the following would you recommend as the MOST cost-optimal solution?

- **Purchase 70 reserved instances and 30 on-demand instances**
- **Purchase 70 reserved instances and 30 spot instances** **(Correcto)**
- **Purchase 70 on-demand instances and 30 spot instances**
- **Purchase 70 on-demand instances and 30 reserved instances**

Explicación
Correct option:

Purchase 70 reserved instances and 30 spot instances

As 70 instances need to be always available, these can be purchased as reserved instances for a one-year duration. The other 30 instances responsible for the batch job can be purchased as spot instances. Even if some of the spot instances are interrupted, other spot instances can continue with the job.

Please see this detailed overview of various types of EC2 instances from a pricing perspective:

On-Demand

With On-Demand instances, you pay for compute capacity by the hour or the second depending on which instances you run. No longer-term commitments or upfront payments are needed. You can increase or decrease your compute capacity depending on the demands of your application and only pay the specified per hourly rates for the instance you use.

On-Demand instances are recommended for:

- Users that prefer the low cost and flexibility of Amazon EC2 without any up-front payment or long-term commitment
- Applications with short-term, spiky, or unpredictable workloads that cannot be interrupted
- Applications being developed or tested on Amazon EC2 for the first time

See On-Demand pricing »

Spot instances

Amazon EC2 Spot instances allow you to request spare Amazon EC2 computing capacity for up to 90% off the On-Demand price. Learn More.

Spot instances are recommended for:

- Applications that have flexible start and end times
- Applications that are only feasible at very low compute prices
- Users with urgent computing needs for large amounts of additional capacity

See Spot pricing »

Savings Plans

Savings Plans are a flexible pricing model that offer low prices on EC2 and Fargate usage, in exchange for a commitment to a consistent amount of usage (measured in $/hour) for a 1 or 3 year term.

Learn more »

Reserved Instances

Reserved Instances provide you with a significant discount (up to 75%) compared to On-Demand instance pricing. In addition, when Reserved Instances are assigned to a specific Availability Zone, they provide a capacity reservation, giving you additional confidence in your ability to launch instances when you need them.

For applications that have steady state or predictable usage, Reserved Instances can provide significant savings compared to using On-Demand instances. See How to Purchase Reserved Instances for more information.

Reserved Instances are recommended for:

- Applications with steady state usage
- Applications that may require reserved capacity
- Customers that can commit to using EC2 over a 1 or 3 year term to reduce their total computing costs

via - https://aws.amazon.com/ec2/pricing/

Incorrect options:

Purchase 70 on-demand instances and 30 spot instances

Purchase 70 on-demand instances and 30 reserved instances

Purchasing 70 on-demand instances would be costlier than 70 reserved instances, so these two options are ruled out.

Purchase 70 reserved instances and 30 on-demand instances - Purchasing 30 instances as on-demand instances to handle the batch jobs would not be cost-optimal as these instances don't need to be always available. Spot instances are better at handling such batch jobs. So this option is not correct.

Reference:

https://aws.amazon.com/ec2/pricing/

Pregunta 35:
A big-data consulting firm is working on a client engagement where the ETL workloads are currently handled via a Hadoop cluster deployed in the on-premises data center. The client wants to migrate their ETL workloads to AWS Cloud. The AWS Cloud solution needs to be highly available with about 50 EC2 instances per Availability Zone.

As a solutions architect, which of the following EC2 placement groups would you recommend handling the distributed ETL workload?

- **Partition placement group(Correcto)**
- **Cluster placement group**
- **Both Spread placement group and Partition placement group**
- **Spread placement group**

Explicación
Correct option:

Partition placement group

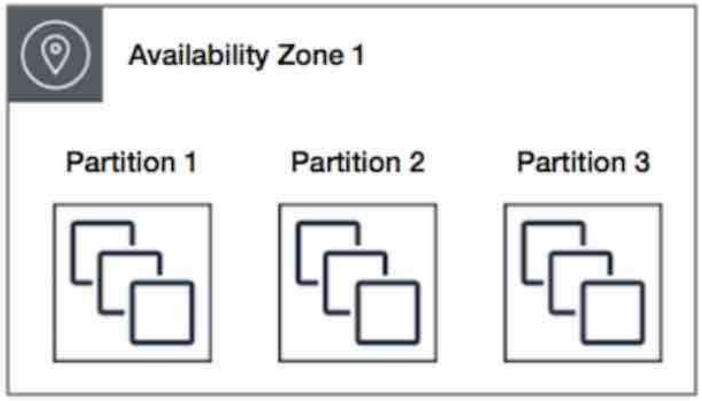
You can use placement groups to influence the placement of a group of interdependent instances to meet the needs of your workload. Depending on the type of workload, you can create a placement group using one of the following placement strategies:

Partition – spreads your instances across logical partitions such that groups of instances in one partition do not share the underlying hardware with groups of instances in different partitions. This strategy is typically used by large distributed and replicated workloads, such as Hadoop, Cassandra, and Kafka. Therefore, this is the correct option for the given use-case.

Partition placement groups

Partition placement groups help reduce the likelihood of correlated hardware failures for your application. When using partition placement groups, Amazon EC2 divides each group into logical segments called partitions. Amazon EC2 ensures that each partition within a placement group has its own set of racks. Each rack has its own network and power source. No two partitions within a placement group share the same racks, allowing you to isolate the impact of hardware failure within your application.

The following image is a simple visual representation of a partition placement group in a single Availability Zone. It shows instances that are placed into a partition placement group with three partitions—**Partition 1**, **Partition 2**, and **Partition 3**. Each partition comprises multiple instances. The instances in a partition do not share racks with the instances in the other partitions, allowing you to contain the impact of a single hardware failure to only the associated partition.



via - https://docs.aws.amazon.com/AWSEC2/latest/UserGuide/placement-groups.html

Incorrect options:

Cluster placement group

Cluster – packs instances close together inside an Availability Zone. This strategy enables workloads to achieve the low-latency network performance necessary for tightly-coupled node-to-node communication that is typical of HPC applications. This is not suited for distributed and replicated workloads such as Hadoop.

Cluster placement groups

A cluster placement group is a logical grouping of instances within a single Availability Zone. A cluster placement group can span peered VPCs in the same Region. Instances in the same cluster placement group enjoy a higher per-flow throughput limit of up to 10 Gbps for TCP/IP traffic and are placed in the same high-bisection bandwidth segment of the network.

The following image shows instances that are placed into a cluster placement group.

via - https://docs.aws.amazon.com/AWSEC2/latest/UserGuide/placement-groups.html

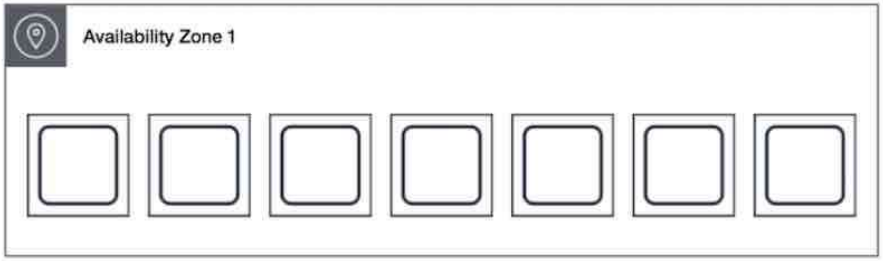
Spread placement group

Spread – strictly places a small group of instances across distinct underlying hardware to reduce correlated failures. This is not suited for distributed and replicated workloads such as Hadoop.

Spread placement groups

A spread placement group is a group of instances that are each placed on distinct racks, with each rack having its own network and power source.

The following image shows seven instances in a single Availability Zone that are placed into a spread placement group. The seven instances are placed on seven different racks.



via - https://docs.aws.amazon.com/AWSEC2/latest/UserGuide/placement-groups.html

Both Spread placement group and Partition placement group - As mentioned earlier, the spread placement group is not suited for distributed and replicated workloads such as Hadoop. So this option is also incorrect.

Reference:

https://docs.aws.amazon.com/AWSEC2/latest/UserGuide/placement-groups.html

Pregunta 36:
Your company has deployed an application that will perform a lot of overwrites and deletes on data and require the latest information to be available anytime data is read via queries on database tables.

As a Solutions Architect, which database technology will you recommend?

- **Amazon Relational Database Service (Amazon RDS)** **(Correcto)**
- **Amazon Simple Storage Service (Amazon S3)**
- **Amazon Neptune**
- **Amazon ElastiCache**

Explicación
Correct option:

Amazon Relational Database Service (Amazon RDS) - Amazon Relational Database Service (Amazon RDS) makes it easy to set up, operate, and scale a relational database in the cloud. It provides cost-efficient and resizable capacity while automating time-consuming administration tasks such as hardware provisioning, database setup, patching, and backups. RDS allows you to create, read, update, and delete records without any item lock or ambiguity. All RDS transactions must be ACID compliant or be Atomic, Consistent, Isolated, and Durable to ensure data integrity.

Atomicity requires that either transaction as a whole is successfully executed or if a part of the transaction fails, then the entire transaction be invalidated. Consistency mandates the data written to the database as part of the transaction must adhere to all defined rules, and restrictions including constraints, cascades, and triggers. Isolation is critical to achieving concurrency control and makes sure each transaction is independent unto itself. Durability requires that all of the changes made to the database be permanent once a transaction is completed. Hence, the best fit is RDS.

Incorrect options:

Amazon ElastiCache - Amazon ElastiCache allows you to seamlessly set up, run, and scale popular open-source compatible in-memory data stores in the cloud. Build data-intensive apps or boost the performance of your existing databases by retrieving data from high throughput and low latency in-memory data stores. Amazon ElastiCache is a popular choice for real-time use cases like Caching, Session Stores, Gaming, Geospatial Services, Real-Time Analytics, and Queuing. ElastiCache could work but it's a better fit as a caching technology to enhance reads.

Amazon Simple Storage Service (Amazon S3) - This option is incorrect as S3 is not a database technology that supports queries on database tables out of the box. It is an object storage service that offers industry-leading scalability, data availability, security, and performance. Your applications can easily achieve thousands of transactions per second in request performance when uploading and retrieving storage from Amazon S3.

After a successful write of a new object or an overwrite of an existing object, any subsequent read request immediately receives the latest version of the object. S3 also provides strong consistency for list operations, so after a write, you can immediately perform a listing of the objects in a bucket with any changes reflected. Strong read-after-write consistency helps when you need to immediately read an object after a write. For example, strong read-after-write consistency when you often read and list immediately after writing objects.

Amazon Neptune - Amazon Neptune is a fast, reliable, fully-managed graph database service that makes it easy to build and run applications that work with highly connected datasets. The core of Amazon Neptune is a purpose-built, high-performance graph database engine optimized for storing billions of relationships and querying the graph with milliseconds latency.

Amazon Neptune is highly available, with read replicas, point-in-time recovery, continuous backup to Amazon S3, and replication across Availability Zones. Neptune is secure with support for HTTPS encrypted client connections and encryption at rest. Neptune is fully managed, so you no longer need to worry about database management tasks such as hardware provisioning, software patching, setup, configuration, or backups. Neptune is a graph database so it's not a good fit.

References:

https://aws.amazon.com/relational-database/

https://aws.amazon.com/rds/

https://aws.amazon.com/neptune/

https://docs.aws.amazon.com/AmazonS3/latest/dev/Introduction.html#ConsistencyModel

Pregunta 37:
A retail company wants to rollout and test a blue-green deployment for its global application in the next 48 hours. Most of the customers use mobile phones which are prone to DNS caching. The company has only two days left for the annual Thanksgiving sale to commence.

As a Solutions Architect, which of the following options would you recommend to test the deployment on as many users as possible in the given time frame?

- **Use Elastic Load Balancer to distribute traffic across deployments**
- **Use Route 53 weighted routing to spread traffic across different deployments**
- **Use AWS Global Accelerator to distribute a portion of traffic to a particular deployment(Correcto)**
- **Use AWS CodeDeploy deployment options to choose the right deployment**

Explicación
Correct option:

Blue/green deployment is a technique for releasing applications by shifting traffic between two identical environments running different versions of the application: "Blue" is the currently running version and "green" the new version. This type of deployment allows you to test features in the green environment without impacting the currently running version of your application. When you're satisfied that the green version is working properly, you can gradually reroute the traffic from the old blue environment to the new green environment. Blue/green deployments can mitigate common risks associated with deploying software, such as downtime and rollback capability.

Use AWS Global Accelerator to distribute a portion of traffic to a particular deployment - AWS Global Accelerator is a network layer service that directs traffic to optimal endpoints over the AWS global network, this improves the availability and performance of

your internet applications. It provides two static anycast IP addresses that act as a fixed entry point to your application endpoints in a single or multiple AWS Regions, such as your Application Load Balancers, Network Load Balancers, Elastic IP addresses or Amazon EC2 instances, in a single or in multiple AWS regions.

AWS Global Accelerator uses endpoint weights to determine the proportion of traffic that is directed to endpoints in an endpoint group, and traffic dials to control the percentage of traffic that is directed to an endpoint group (an AWS region where your application is deployed).

While relying on the DNS service is a great option for blue/green deployments, it may not fit use-cases that require a fast and controlled transition of the traffic. Some client devices and internet resolvers cache DNS answers for long periods; this DNS feature improves the efficiency of the DNS service as it reduces the DNS traffic across the Internet, and serves as a resiliency technique by preventing authoritative name-server overloads. The downside of this in blue/green deployments is that you don't know how long it will take before all of your users receive updated IP addresses when you update a record, change your routing preference or when there is an application failure.

With AWS Global Accelerator, you can shift traffic gradually or all at once between the blue and the green environment and vice-versa without being subject to DNS caching on client devices and internet resolvers, traffic dials and endpoint weights changes are effective within seconds.

Incorrect options:

Use Route 53 weighted routing to spread traffic across different deployments - Weighted routing lets you associate multiple resources with a single domain name (example.com) or subdomain name (acme.example.com) and choose how much traffic is routed to each resource. This can be useful for a variety of purposes, including load balancing and testing new versions of the software. As discussed earlier, DNS caching is a negative behavior for this use case and hence Route 53 is not a good option.

Use Elastic Load Balancer to distribute traffic across deployments - An ELB can distribute traffic across healthy instances. You can also use the ALB weighted target groups feature for blue/green deployments as it does not rely on the DNS service. In addition you don't need to create new ALBs for the green environment. As the use-case refers to a global application, so this option cannot be used for a multi-Region solution which is needed for the given requirement.

Use AWS CodeDeploy deployment options to choose the right deployment - In CodeDeploy, a deployment is the process, and the components involved in the process, of installing content on one or more instances. This content can consist of code, web and configuration files, executables, packages, scripts, and so on. CodeDeploy deploys content that is stored in a source repository, according to the configuration rules you specify. Blue/Green deployment is one of the deployment types that CodeDeploy supports. CodeDeploy is not meant to distribute traffic across instances, so this option is incorrect.

References:

https://aws.amazon.com/blogs/networking-and-content-delivery/using-aws-global-accelerator-to-achieve-blue-green-deployments

https://docs.aws.amazon.com/codedeploy/latest/userguide/deployments.html

https://docs.aws.amazon.com/Route53/latest/DeveloperGuide/routing-policy.html#routing-policy-weighted

Pregunta 38:
A manufacturing company receives unreliable service from its data center provider because the company is located in an area prone to natural disasters. The company is not ready to fully migrate to the AWS Cloud, but it wants a failover environment on AWS in case the on-premises data center fails. The company runs web servers that connect to external vendors. The data available on AWS and on-premises must be uniform.

Which of the following solutions would have the LEAST amount of downtime?

- **Set up a Route 53 failover record. Set up an AWS Direct Connect connection between a VPC and the data center. Run application servers on EC2 in an Auto Scaling group. Run an AWS Lambda function to execute an AWS CloudFormation template to create an Application Load Balancer**
- **Set up a Route 53 failover record. Run an AWS Lambda function to execute an AWS CloudFormation template to launch two EC2 instances. Set up AWS Storage Gateway with stored volumes to back up data to S3. Set up an AWS Direct Connect connection between a VPC and the data center**
- **Set up a Route 53 failover record. Execute an AWS CloudFormation template from a script to provision EC2 instances behind an Application Load Balancer. Set up AWS Storage Gateway with stored volumes to back up data to S3**
- **Set up a Route 53 failover record. Run application servers on EC2 instances behind an Application Load Balancer in an Auto Scaling group. Set up AWS Storage Gateway with stored volumes to back up data to S3 (Correcto)**

Explicación
Correct option:

Set up a Route 53 failover record. Run application servers on EC2 instances behind an Application Load Balancer in an Auto Scaling group. Set up AWS Storage Gateway with stored volumes to back up data to S3

If you have multiple resources that perform the same function, you can configure DNS failover so that Route 53 will route your traffic from an unhealthy resource to a healthy resource.

Elastic Load Balancing is used to automatically distribute your incoming application traffic across all the EC2 instances that you are running. You can use Elastic Load Balancing to manage incoming requests by optimally routing traffic so that no one instance is overwhelmed. Your load balancer acts as a single point of contact for all incoming web traffic to your Auto Scaling group.

AWS Storage Gateway is a hybrid cloud storage service that gives you on-premises access to virtually unlimited cloud storage. It provides low-latency performance by caching frequently accessed data on-premises while storing data securely and durably in Amazon cloud storage services. Storage Gateway optimizes data transfer to AWS by sending only changed data and compressing data. Storage Gateway also integrates natively with Amazon S3 cloud storage which makes your data available for in-cloud processing.

Incorrect options:

Set up a Route 53 failover record. Execute an AWS CloudFormation template from a script to provision EC2 instances behind an Application Load Balancer. Set up AWS Storage Gateway with stored volumes to back up data to S3

Set up a Route 53 failover record. Run an AWS Lambda function to execute an AWS CloudFormation template to launch two EC2 instances. Set up AWS Storage Gateway with stored volumes to back up data to S3. Set up an AWS Direct Connect connection between a VPC and the data center

Set up a Route 53 failover record. Set up an AWS Direct Connect connection between a VPC and the data center. Run application servers on EC2 in an Auto Scaling group. Run an AWS Lambda function to execute an AWS CloudFormation template to create an Application Load Balancer

AWS CloudFormation is a convenient provisioning mechanism for a broad range of AWS and third-party resources. It supports the infrastructure needs of many different types of applications such as existing enterprise applications, legacy applications, applications built using a variety of AWS resources, and container-based solutions.

These three options involve CloudFormation as part of the solution. Now, CloudFormation takes time to provision the resources and hence is not the right solution when LEAST amount of downtime is mandated for the given use case. Therefore, these options are not the right fit for the given requirement.

References:

https://aws.amazon.com/route53/

https://aws.amazon.com/storagegateway/

Pregunta 39:
The engineering manager for a content management application wants to set up RDS read replicas to provide enhanced performance and read scalability. The manager wants to understand the data transfer charges while setting up RDS read replicas.

Which of the following would you identify as correct regarding the data transfer charges for RDS read replicas?

- **There are data transfer charges for replicating data across AWS Regions (Correcto)**
- **There are no data transfer charges for replicating data across AWS Regions**
- **There are data transfer charges for replicating data within the same Availability Zone**
- **There are data transfer charges for replicating data within the same AWS Region**

Explicación
Correct option:

There are data transfer charges for replicating data across AWS Regions

RDS Read Replicas provide enhanced performance and durability for RDS database (DB) instances. They make it easy to elastically scale out beyond the capacity constraints of a single DB instance for read-heavy database workloads.

A read replica is billed as a standard DB Instance and at the same rates. You are not charged for the data transfer incurred in replicating data between your source DB instance and read replica within the same AWS Region.

Q: How much do read replicas cost? When does billing begin and end?

A read replica is billed as a standard DB Instance and at the same rates. Just like a standard DB instance, the rate per "DB Instance hour" for a read replica is determined by the DB instance class of the read replica – please see pricing page for up-to-date pricing. You are not charged for the data transfer incurred in replicating data between your source DB instance and read replica within the same AWS Region.

Billing for a read replica begins as soon as the replica has been successfully created (i.e. when status is listed as "active"). The read replica will continue being billed at standard Amazon RDS DB instance hour rates until you issue a command to delete it.

via - https://aws.amazon.com/rds/faqs/

Incorrect options:

There are data transfer charges for replicating data within the same Availability Zone

There are data transfer charges for replicating data within the same AWS Region

There are no data transfer charges for replicating data across AWS Regions

These three options contradict the explanation provided above, so these options are incorrect.

Reference:

https://aws.amazon.com/rds/faqs/

Pregunta 40:
An engineering team wants to examine the feasibility of the user data feature of Amazon EC2 for an upcoming project.

Which of the following are true about the EC2 user data configuration? (Select two)

- **When an instance is running, you can update user data by using root user credentials**
- **By default, scripts entered as user data are executed with root user privileges(Correcto)**
- **By default, scripts entered as user data do not have root user privileges for executing**
- **By default, user data runs only during the boot cycle when you first launch an instance(Correcto)**
- **By default, user data is executed every time an EC2 instance is re-started**

Explicación
Correct options:

User Data is generally used to perform common automated configuration tasks and even run scripts after the instance starts. When you launch an instance in Amazon EC2, you can pass two types of user data - shell scripts and cloud-init directives. You can also pass this data into the launch wizard as plain text or as a file.

By default, scripts entered as user data are executed with root user privileges - Scripts entered as user data are executed as the root user, hence do not need the sudo command in the script. Any files you create will be owned by root; if you need non-root users to have file access, you should modify the permissions accordingly in the script.

By default, user data runs only during the boot cycle when you first launch an instance - By default, user data scripts and cloud-init directives run only during the boot cycle when you first launch an instance. You can update your configuration to ensure that your user data scripts and cloud-init directives run every time you restart your instance.

Incorrect options:

By default, user data is executed every time an EC2 instance is re-started - As discussed above, this is not a default configuration of the system. But, can be achieved by explicitly configuring the instance.

When an instance is running, you can update user data by using root user credentials - You can't change the user data if the instance is running (even by using root user credentials), but you can view it.

By default, scripts entered as user data do not have root user privileges for executing - Scripts entered as user data are executed as the root user, hence do not need the sudo command in the script.

Reference:

https://docs.aws.amazon.com/AWSEC2/latest/UserGuide/user-data.html

Pregunta 41:
An IT company provides S3 bucket access to specific users within the same account for completing project specific work. With changing business requirements, cross-account S3 access requests are also growing every month. The company is looking for a solution that can offer user level as well as account-level access permissions for the data stored in S3 buckets.

As a Solutions Architect, which of the following would you suggest as the MOST optimized way of controlling access for this use-case?

- **Use Amazon S3 Bucket Policies(Correcto)**
- **Use Access Control Lists (ACLs)**
- **Use Security Groups**
- **Use Identity and Access Management (IAM) policies**

Explicación
Correct option:

Use Amazon S3 Bucket Policies

Bucket policies in Amazon S3 can be used to add or deny permissions across some or all of the objects within a single bucket. Policies can be attached to users, groups, or Amazon S3 buckets, enabling centralized management of permissions. With bucket policies, you can grant users within your AWS Account or other AWS Accounts access to your Amazon S3 resources.

You can further restrict access to specific resources based on certain conditions. For example, you can restrict access based on request time (Date Condition), whether the request was sent using SSL (Boolean Conditions), a requester's IP address (IP Address Condition), or based on the requester's client application (String Conditions). To identify these conditions, you use policy keys.

Types of access control in S3:

- **Bucket Policies.** Bucket policies in Amazon S3 can be used to add or deny permissions across some or all of the objects within a single bucket. Policies can be attached to users, groups, or Amazon S3 buckets, enabling centralized management of permissions. With bucket policies, you can grant users within your AWS Account or other AWS Accounts access to your Amazon S3 resources.

Table 3: Types of access control

Type of Access Control	AWS Account Level Control	User Level Control
IAM Policies	No	Yes
ACLs	Yes	No
Bucket Policies	Yes	Yes

You can further restrict access to specific resources based on certain conditions. For example, you can restrict access based on request time (Date Condition), whether the request was sent using SSL (Boolean Conditions), a requester's IP address (IP Address Condition), or based on the requester's client application (String Conditions). To identify these conditions, you use policy keys. For more information about action-specific policy keys available within Amazon S3, see the Amazon Simple Storage Service Developer Guide.

via - https://d1.awsstatic.com/whitepapers/aws-security-whitepaper.pdf

Incorrect options:

Use Identity and Access Management (IAM) policies - AWS IAM enables organizations with many employees to create and manage multiple users under a single AWS account. IAM policies are attached to the users, enabling centralized control of permissions for users under your AWS Account to access buckets or objects. With IAM policies, you can only grant users within your own AWS account permission to access your Amazon S3 resources. So, this is not the right choice for the current requirement.

Use Access Control Lists (ACLs) - Within Amazon S3, you can use ACLs to give read or write access on buckets or objects to groups of users. With ACLs, you can only grant other AWS accounts (not specific users) access to your Amazon S3 resources. So, this is not the right choice for the current requirement.

Use Security Groups - A security group acts as a virtual firewall for EC2 instances to control incoming and outgoing traffic. S3 does not support Security Groups, this option just acts as a distractor.

Reference:

https://d1.awsstatic.com/whitepapers/aws-security-whitepaper.pdf

Pregunta 42:
You have been hired as a Solutions Architect to advise a company on the various authentication/authorization mechanisms that AWS offers to authorize an API call within the API Gateway. The company would prefer a solution that offers built-in user management.

Which of the following solutions would you suggest as the best fit for the given use-case?

- **Use API Gateway Lambda authorizer**
- **Use AWS_IAM authorization**
- **Use Amazon Cognito Identity Pools**
- **Use Amazon Cognito User Pools** **(Correcto)**

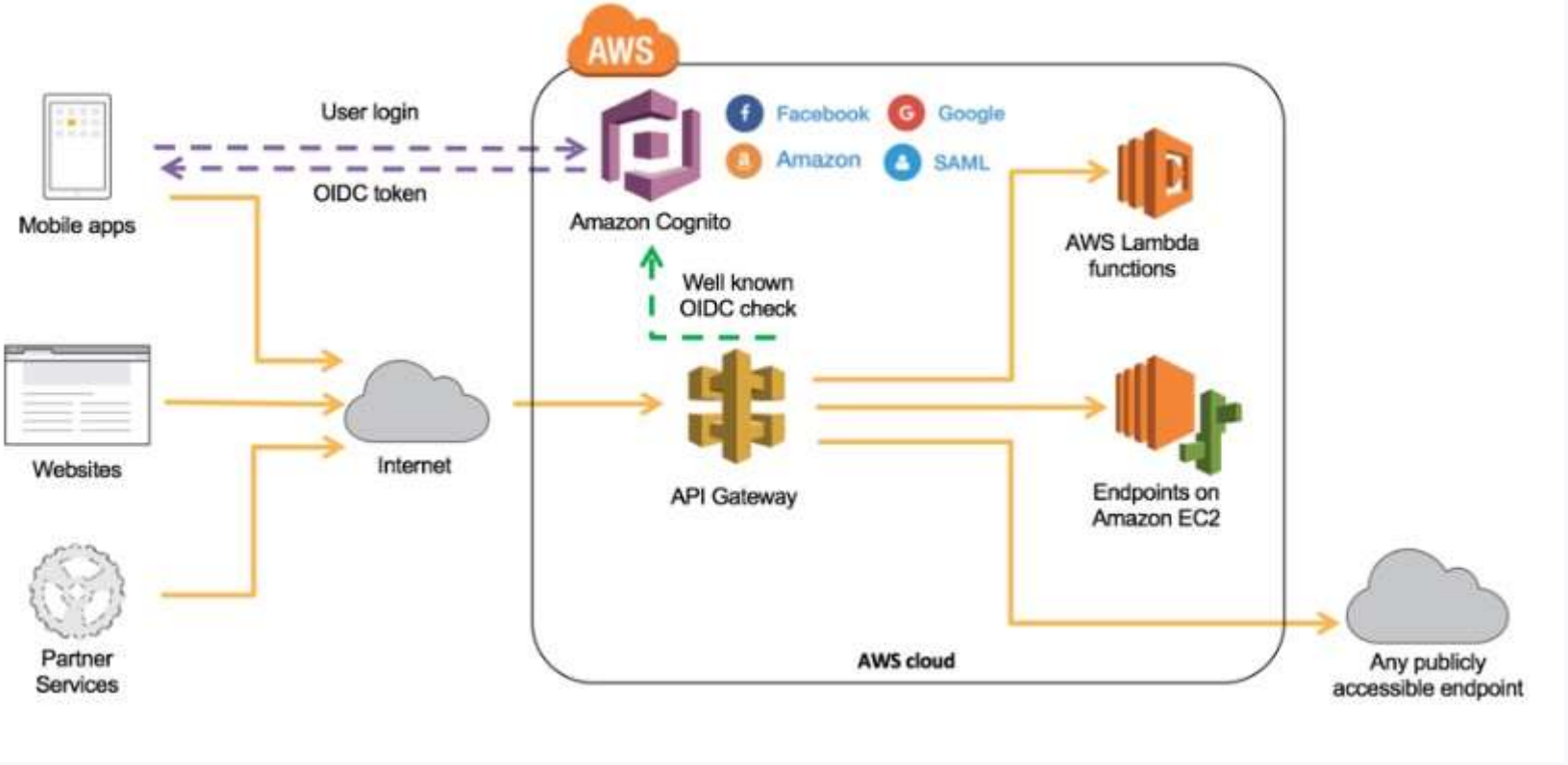
Explicación
Correct option:

Use Amazon Cognito User Pools - A user pool is a user directory in Amazon Cognito. You can leverage Amazon Cognito User Pools to either provide built-in user management or integrate with external identity providers, such as Facebook, Twitter, Google+, and Amazon. Whether your users sign-in directly or through a third party, all members of the user pool have a directory profile that you can access through a Software Development Kit (SDK).

User pools provide: 1. Sign-up and sign-in services. 2. A built-in, customizable web UI to sign in users. 3. Social sign-in with Facebook, Google, Login with Amazon, and Sign in with Apple, as well as sign-in with SAML identity providers from your user pool. 4. User directory management and user profiles. 5. Security features such as multi-factor authentication (MFA), checks for compromised credentials, account takeover protection, and phone and email verification. 6. Customized workflows and user migration through AWS Lambda triggers.

After creating an Amazon Cognito user pool, in API Gateway, you must then create a COGNITO_USER_POOLS authorizer that uses the user pool.

Amazon Cognito User Pools:



via - https://docs.aws.amazon.com/wellarchitected/latest/serverless-applications-lens/identity-and-access-management.html

Incorrect options:

Use AWS_IAM authorization - For consumers who currently are located within your AWS environment or have the means to retrieve AWS Identity and Access Management (IAM) temporary credentials to access your environment, you can use AWS_IAM authorization and add least-privileged permissions to the respective IAM role to securely invoke your API. API Gateway API Keys is not a security mechanism and should not be used for authorization unless it's a public API. It should be used primarily to track a consumer's usage across your API.

Use API Gateway Lambda authorizer - If you have an existing Identity Provider (IdP), you can use an API Gateway Lambda authorizer to invoke a Lambda function to authenticate/validate a given user against your IdP. You can use a Lambda authorizer for custom validation logic based on identity metadata.

A Lambda authorizer can send additional information derived from a bearer token or request context values to your backend service. For example, the authorizer can return a map containing user IDs, user names, and scope. By using Lambda authorizers, your backend does not need to map authorization tokens to user-centric data, allowing you to limit the exposure of such information to just the authorization function.

When using Lambda authorizers, AWS strictly advises against passing credentials or any sort of sensitive data via query string parameters or headers, so this is not as secure as using Cognito User Pools.

In addition, both these options do not offer built-in user management.

Use Amazon Cognito Identity Pools - The two main components of Amazon Cognito are user pools and identity pools. Identity pools provide AWS credentials to grant your users access to other AWS services. To enable users in your user pool to access AWS resources, you can configure an identity pool to exchange user pool tokens for AWS credentials. So, identity pools aren't an authentication mechanism in themselves and hence aren't a choice for this use case.

References:

https://docs.aws.amazon.com/wellarchitected/latest/serverless-applications-lens/identity-and-access-management.html

https://docs.aws.amazon.com/apigateway/latest/developerguide/apigateway-enable-cognito-user-pool.html

https://docs.aws.amazon.com/cognito/latest/developerguide/cognito-user-identity-pools.html

Pregunta 43:
A financial services company wants to store confidential data in Amazon S3 and it needs to meet the following data security and compliance norms:

1. Encryption key usage must be logged for auditing purposes
2. Encryption Keys must be rotated every year
3. The data must be encrypted at rest

Which is the MOST operationally efficient solution?

- **Server-side encryption with AWS KMS (SSE-KMS) customer master keys (CMKs) with manual key rotation**
- **Server-side encryption with AWS KMS (SSE-KMS) customer master keys (CMKs) with automatic key rotation(Correcto)**
- **Server-side encryption (SSE-S3) with automatic key rotation**
- **Server-side encryption with customer-provided keys (SSE-C) with automatic key rotation**

Explicación
Correct option:

Server-side encryption with AWS KMS (SSE-KMS) customer master keys (CMKs) with automatic key rotation

Server-side encryption is the encryption of data at its destination by the application or service that receives it. Amazon S3 encrypts your data at the object level as it writes it to disks in its data centers and decrypts it for you when you access it.

S3 server-side encryption

Protecting data using server-side encryption

PDF | RSS

Server-side encryption is the encryption of data at its destination by the application or service that receives it. Amazon S3 encrypts your data at the object level as it writes it to disks in its data centers and decrypts it for you when you access it. As long as you authenticate your request and you have access permissions, there is no difference in the way you access encrypted or unencrypted objects. For example, if you share your objects using a presigned URL, that URL works the same way for both encrypted and unencrypted objects. Additionally, when you list objects in your bucket, the list API returns a list of all objects, regardless of whether they are encrypted.

> **Note**
>
> You can't apply different types of server-side encryption to the same object simultaneously.

You have three mutually exclusive options, depending on how you choose to manage the encryption keys.

Server-Side Encryption with Amazon S3-Managed Keys (SSE-S3)

When you use Server-Side Encryption with Amazon S3-Managed Keys (SSE-S3), each object is encrypted with a unique key. As an additional safeguard, it encrypts the key itself with a root key that it regularly rotates. Amazon S3 server-side encryption uses one of the strongest block ciphers available, 256-bit Advanced Encryption Standard (AES-256) GCM, to encrypt your data. For objects encrypted prior to AES-GCM, AES-CBC is still supported to decrypt those objects. For more information, see Protecting data using server-side encryption with Amazon S3-managed encryption keys (SSE-S3).

Server-Side Encryption with KMS keys Stored in AWS Key Management Service (SSE-KMS)

Server-Side Encryption with AWS KMS keys (SSE-KMS) is similar to SSE-S3, but with some additional benefits and charges for using this service. There are separate permissions for the use of a KMS key that provides added protection against unauthorized access of your objects in Amazon S3. SSE-KMS also provides you with an audit trail that shows when your KMS key was used and by whom. Additionally, you can create and manage customer managed keys or use AWS managed keys that are unique to you, your service, and your Region. For more information, see Protecting data using server-side encryption with AWS Key Management Service (SSE-KMS).

Server-Side Encryption with Customer-Provided Keys (SSE-C)

With Server-Side Encryption with Customer-Provided Keys (SSE-C), you manage the encryption keys and Amazon S3 manages the encryption, as it writes to disks, and decryption, when you access your objects. For more information, see Protecting data using server-side encryption with customer-provided encryption keys (SSE-C).

via - https://docs.aws.amazon.com/AmazonS3/latest/userguide/serv-side-encryption.html

AWS KMS is a managed service that makes it easy for you to create and control the cryptographic keys that are used to protect your data. AWS KMS keys (KMS keys are also known as customer master key (CMK)) are the primary resource in AWS KMS. You can use a KMS key to encrypt, decrypt, and re-encrypt data. An AWS KMS key is a logical representation of a cryptographic key. A KMS key contains metadata, such as the key ID, key spec, key usage, creation date, description, and key state. Most importantly, it contains a reference to the key material that is used when you run cryptographic operations with the KMS key.

When you enable automatic key rotation for a KMS key, AWS KMS generates new cryptographic material for the KMS key every year.

AWS KMS keys:

Rotating AWS KMS keys

PDF | RSS

Cryptographic best practices discourage extensive reuse of encryption keys. To create new cryptographic material for your customer managed keys, you can create new KMS keys, and then change your applications or aliases to use the new KMS keys. Or, you can enable automatic key rotation for an existing KMS key.

When you enable *automatic key rotation* for a KMS key, AWS KMS generates new cryptographic material for the KMS key every year. AWS KMS saves all previous versions of the cryptographic material in perpetuity so you can decrypt any data encrypted with that KMS key. AWS KMS does not delete any rotated key material until you delete the KMS key. You can track the rotation of key material for your KMS keys in Amazon CloudWatch and AWS CloudTrail.

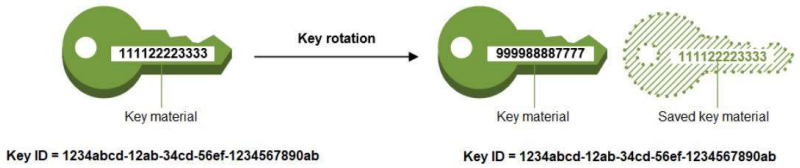
When you use a rotated KMS key to encrypt data, AWS KMS uses the current key material. When you use the rotated KMS key to decrypt ciphertext, AWS KMS uses the version of the key material that was used to encrypt it. You cannot request a particular version of the key material. Because AWS KMS transparently decrypts with the appropriate key material, you can safely use a rotated KMS key in applications and AWS services without code changes.

However, automatic key rotation has no effect on the data that the KMS key protects. It does not rotate the data keys that the KMS key generated or re-encrypt any data protected by the KMS key, and it will not mitigate the effect of a compromised data key.

AWS KMS supports automatic key rotation only for symmetric encryption KMS keys with key material that AWS KMS creates. Automatic rotation is optional for customer managed KMS keys. AWS KMS always rotates the key material for AWS managed KMS keys every year. Rotation of AWS owned KMS keys varies.

> **Note**
>
> The rotation interval for AWS managed keys changed in May 2022. For details, see AWS managed keys.

Key rotation changes only the *key material*, which is the cryptographic secret that is used in encryption operations. The KMS key is the same logical resource, regardless of whether or how many times its key material changes. The properties of the KMS key do not change, as shown in the following image.



via - https://docs.aws.amazon.com/kms/latest/developerguide/rotate-keys.html

For the given use case, you can set up server-side encryption with AWS KMS (SSE-KMS) customer master keys (CMKs) with automatic key rotation.

Incorrect options:

Server-side encryption with AWS KMS (SSE-KMS) customer master keys (CMKs) with manual key rotation - Although it is possible to manually rotate the AWS KMS key, it is not the best fit solution as it is not operationally efficient.

Server-side encryption (SSE-S3) with automatic key rotation - When you use Server-Side Encryption with Amazon S3-Managed Keys (SSE-S3), each object is encrypted with a unique key. As an additional safeguard, it encrypts the key itself with a root key that it regularly rotates. However, with SSE-S3, you cannot log the usage of the encryption key for auditing purposes. So this option is incorrect.

Server-side encryption with customer-provided keys (SSE-C) with automatic key rotation - It is possible to automatically rotate the customer-provided keys but you will need to develop the underlying solution to automate the key rotation. Therefore, this option is not operationally efficient.

References:

https://docs.aws.amazon.com/kms/latest/developerguide/concepts.html#master_keys

https://docs.aws.amazon.com/kms/latest/developerguide/rotate-keys.html

https://docs.aws.amazon.com/AmazonS3/latest/userguide/serv-side-encryption.html

Pregunta 44:
What does this IAM policy do?

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Sid": "Mystery Policy",
      "Action": [
        "ec2:RunInstances"
      ],
      "Effect": "Allow",
      "Resource": "*",
      "Condition": {
        "StringEquals": {
          "aws:RequestedRegion": "eu-west-1"
        }
      }
    }
  ]
}
```

- **It allows running EC2 instances in any region when the API call is originating from the eu-west-1 region**
- **It allows to run EC2 instances in the eu-west-1 region, when the API call is made from the eu-west-1 region**
- **It allows running EC2 instances anywhere but in the eu-west-1 region**

- **It allows running EC2 instances only in the eu-west-1 region, and the API call can be made from anywhere in the world** **(Correcto)**

Explicación
Correct option:

It allows running EC2 instances only in the eu-west-1 region, and the API call can be made from anywhere in the world

You manage access in AWS by creating policies and attaching them to IAM identities (users, groups of users, or roles) or AWS resources. A policy is an object in AWS that, when associated with an identity or resource, defines their permissions. AWS evaluates these policies when an IAM principal (user or role) makes a request. Permissions in the policies determine whether the request is allowed or denied. Most policies are stored in AWS as JSON documents. AWS supports six types of policies: identity-based policies, resource-based policies, permissions boundaries, Organizations SCPs, ACLs, and session policies.

You can use the `aws:RequestedRegion` key to compare the AWS Region that was called in the request with the Region that you specify in the policy. You can use this global condition key to control which Regions can be requested.

`aws:RequestedRegion` represents the target of the API call. So in this example, we can only launch EC2 instances in eu-west-1, and we can do this API call from anywhere.

Incorrect options:

It allows running EC2 instances anywhere but in the eu-west-1 region

It allows running EC2 instances in any region when the API call is originating from the eu-west-1 region

It allows running EC2 instances in the eu-west-1 region when the API call is made from the eu-west-1 region

These three options contradict the earlier details provided in the explanation. To summarize, `aws:RequestedRegion` represents the target of the API call. So, we can only launch EC2 instances in eu-west-1 region and we can do this API call from anywhere. Hence these options are incorrect.

References:

https://docs.aws.amazon.com/IAM/latest/UserGuide/access_policies.html

https://docs.aws.amazon.com/IAM/latest/UserGuide/reference_policies_condition-keys.html

Pregunta 45:
An IT company has an Access Control Management (ACM) application that uses Amazon RDS for MySQL but is running into performance issues despite using Read Replicas. The company has hired you as a solutions architect to address these performance-related challenges without moving away from the underlying relational database schema. The company has branch offices across the world, and it needs the solution to work on a global scale.

Which of the following will you recommend as the MOST cost-effective and high-performance solution?

- **Use Amazon Aurora Global Database to enable fast local reads with low latency in each region** **(Correcto)**
- **Spin up a Redshift cluster in each AWS region. Migrate the existing data into Redshift clusters**
- **Use Amazon DynamoDB Global Tables to provide fast, local, read and write performance in each region**
- **Spin up EC2 instances in each AWS region, install MySQL databases and migrate the existing data into these new databases**

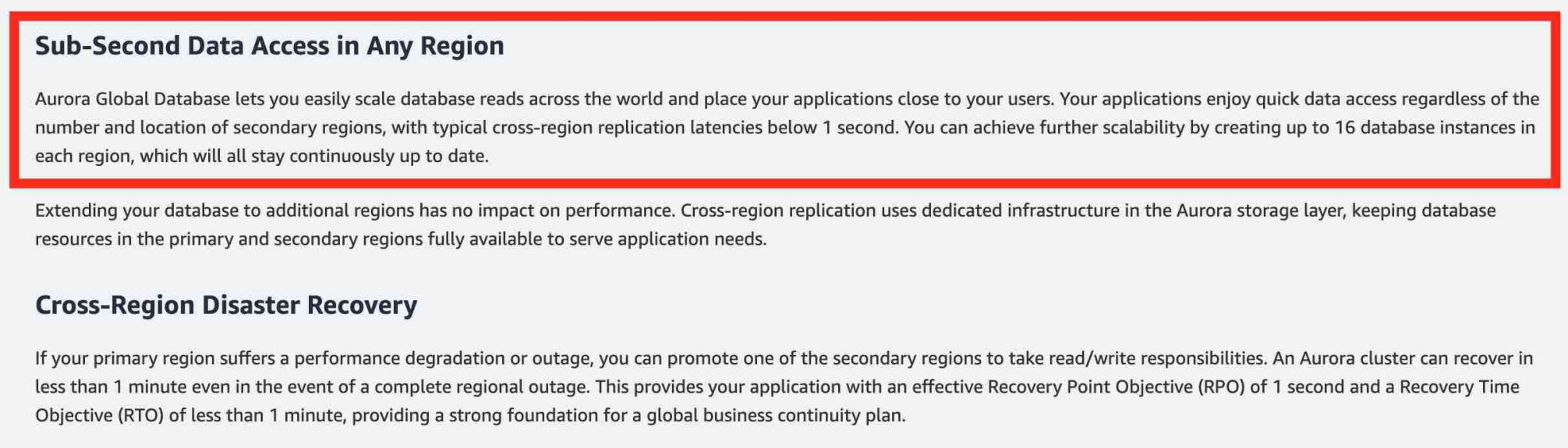
Explicación
Correct option:

Use Amazon Aurora Global Database to enable fast local reads with low latency in each region

Amazon Aurora is a MySQL and PostgreSQL-compatible relational database built for the cloud, that combines the performance and availability of traditional enterprise databases with the simplicity and cost-effectiveness of open source databases. Amazon Aurora features a distributed, fault-tolerant, self-healing storage system that auto-scales up to 64TB per database instance. Aurora is not an in-memory database.

Amazon Aurora Global Database is designed for globally distributed applications, allowing a single Amazon Aurora database to span multiple AWS regions. It replicates your data with no impact on database performance, enables fast local reads with low latency in each region, and provides disaster recovery from region-wide outages. Amazon Aurora Global Database is the correct choice for the given use-case.

Amazon Aurora Global Database Features:



via - https://aws.amazon.com/rds/aurora/global-database/

Incorrect options:

Use Amazon DynamoDB Global Tables to provide fast, local, read and write performance in each region - Amazon DynamoDB is a key-value and document database that delivers single-digit millisecond performance at any scale. It's a fully

managed, multi-region, multi-master, durable database with built-in security, backup and restore, and in-memory caching for internet-scale applications.

Global Tables builds upon DynamoDB's global footprint to provide you with a fully managed, multi-region, and multi-master database that provides fast, local, read, and write performance for massively scaled, global applications. Global Tables replicates your Amazon DynamoDB tables automatically across your choice of AWS regions. Given that the use-case wants you to continue with the underlying schema of the relational database, DynamoDB is not the right choice as it's a NoSQL database.

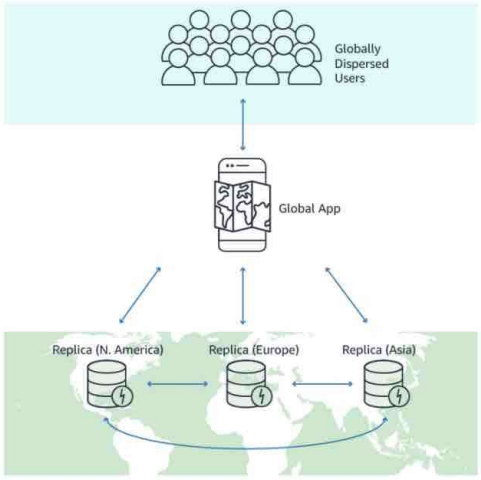
DynamoDB Global Tables Overview:

Global Tables

Multi-Region, Multi-Master tables for fast local performance for globally distributed apps

Global Tables builds upon DynamoDB's global footprint to provide you with a fully managed, multi-region, and multi-master database that provides fast, local, read and write performance for massively scaled, global applications. Global Tables replicates your Amazon DynamoDB tables automatically across your choice of AWS regions.

Global Tables eliminates the difficult work of replicating data between regions and resolving update conflicts, enabling you to focus on your application's business logic. In addition, Global Tables enables your applications to stay highly available even in the unlikely event of isolation or degradation of an entire region.



Global Tables

You can setup Global Tables with just a few clicks in the AWS Management Console. No application changes are required because Global Tables use existing DynamoDB APIs. There are no upfront costs or commitments for using Global Tables, and you pay only for the resources provisioned. Learn more about setting up Global Tables in the DynamoDB Developer Guide.

via - https://aws.amazon.com/dynamodb/global-tables/

Spin up a Redshift cluster in each AWS region. Migrate the existing data into Redshift clusters - Amazon Redshift is a fully-managed petabyte-scale cloud-based data warehouse product designed for large scale data set storage and analysis. Redshift is not suited to be used as a transactional relational database, so this option is not correct.

Spin up EC2 instances in each AWS region, install MySQL databases and migrate the existing data into these new databases - Setting up EC2 instances in multiple regions with manually managed MySQL databases represents a maintenance nightmare and is not the correct choice for this use-case.

References:

https://aws.amazon.com/rds/aurora/global-database/

https://aws.amazon.com/dynamodb/global-tables/

Pregunta 46:
A social photo-sharing web application is hosted on EC2 instances behind an Elastic Load Balancer. The app gives the users the ability to upload their photos and also shows a leaderboard on the homepage of the app. The uploaded photos are stored in S3 and the leaderboard data is maintained in DynamoDB. The EC2 instances need to access both S3 and DynamoDB for these features.

As a solutions architect, which of the following solutions would you recommend as the MOST secure option?

- **Configure AWS CLI on the EC2 instances using a valid IAM user's credentials. The application code can then invoke shell scripts to access S3 and DynamoDB via AWS CLI**
- **Attach the appropriate IAM role to the EC2 instance profile so that the instance can access S3 and DynamoDB(Correcto)**
- **Save the AWS credentials (access key Id and secret access token) in a configuration file within the application code on the EC2 instances. EC2 instances can use these credentials to access S3 and DynamoDB**
- **Encrypt the AWS credentials via a custom encryption library and save it in a secret directory on the EC2 instances. The application code can then safely decrypt the AWS credentials to make the API calls to S3 and DynamoDB**

Explicación
Correct option:

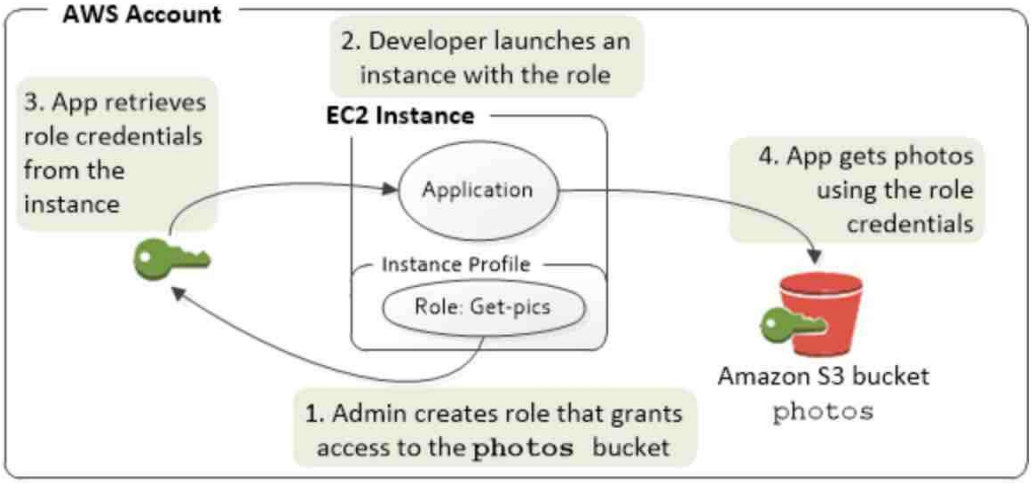
Attach the appropriate IAM role to the EC2 instance profile so that the instance can access S3 and DynamoDB

Applications that run on an EC2 instance must include AWS credentials in their AWS API requests. You could have your developers store AWS credentials directly within the EC2 instance and allow applications in that instance to use those credentials. But developers would then have to manage the credentials and ensure that they securely pass the credentials to each instance and update each EC2 instance when it's time to rotate the credentials.

Instead, you should use an IAM role to manage temporary credentials for applications that run on an EC2 instance. When you use a role, you don't have to distribute long-term credentials (such as a username and password or access keys) to an EC2 instance. The role supplies temporary permissions that applications can use when they make calls to other AWS resources. When you launch an EC2 instance, you specify an IAM role to associate with the instance. Applications that run on the instance can then use the role-supplied temporary credentials to sign API requests. Therefore, this option is correct.

How Do Roles for EC2 Instances Work?

In the following figure, a developer runs an application on an EC2 instance that requires access to the S3 bucket named `photos`. An administrator creates the `Get-pics` service role and attaches the role to the EC2 instance. The role includes a permissions policy that grants read-only access to the specified S3 bucket. It also includes a trust policy that allows the EC2 instance to assume the role and retrieve the temporary credentials. When the application runs on the instance, it can use the role's temporary credentials to access the photos bucket. The administrator doesn't have to grant the developer permission to access the photos bucket, and the developer never has to share or manage credentials.



via - https://docs.aws.amazon.com/IAM/latest/UserGuide/id_roles_use_switch-role-ec2.html

Incorrect options:

Save the AWS credentials (access key Id and secret access token) in a configuration file within the application code on the EC2 instances. EC2 instances can use these credentials to access S3 and DynamoDB

Configure AWS CLI on the EC2 instances using a valid IAM user's credentials. The application code can then invoke shell scripts to access S3 and DynamoDB via AWS CLI

Encrypt the AWS credentials via a custom encryption library and save it in a secret directory on the EC2 instances. The application code can then safely decrypt the AWS credentials to make the API calls to S3 and DynamoDB

Keeping the AWS credentials (encrypted or plain text) on the EC2 instance is a bad security practice, therefore these three options using the AWS credentials are incorrect.

Reference:

https://docs.aws.amazon.com/IAM/latest/UserGuide/id_roles_use_switch-role-ec2.html

Pregunta 47:
A company has many VPC in various accounts, that need to be connected in a star network with one another and connected with on-premises networks through Direct Connect.

What do you recommend?

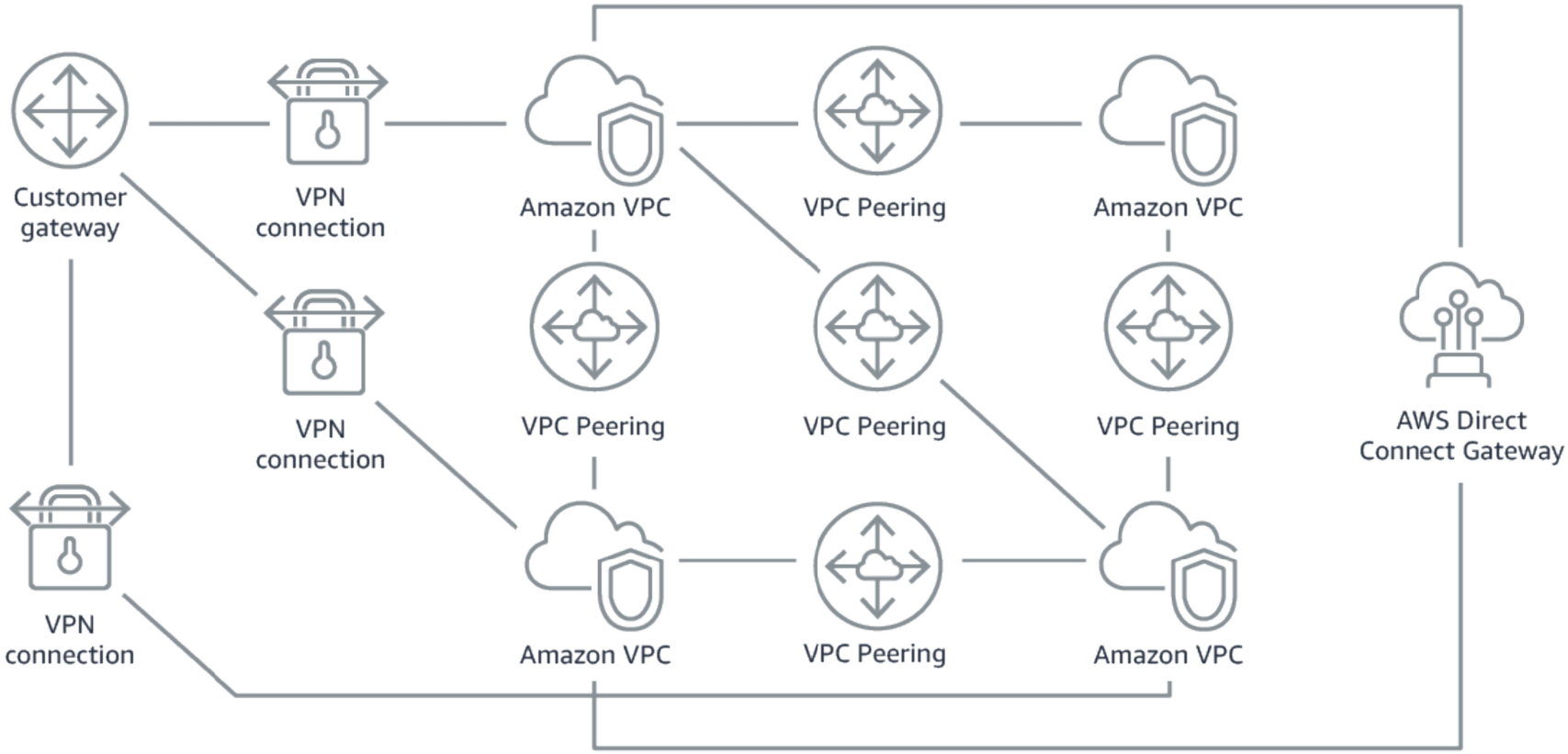
- **Transit Gateway(Correcto)**
- **VPC Peering**
- **Private Link**
- **VPN Gateway**

Explicación
Correct option:

Transit Gateway

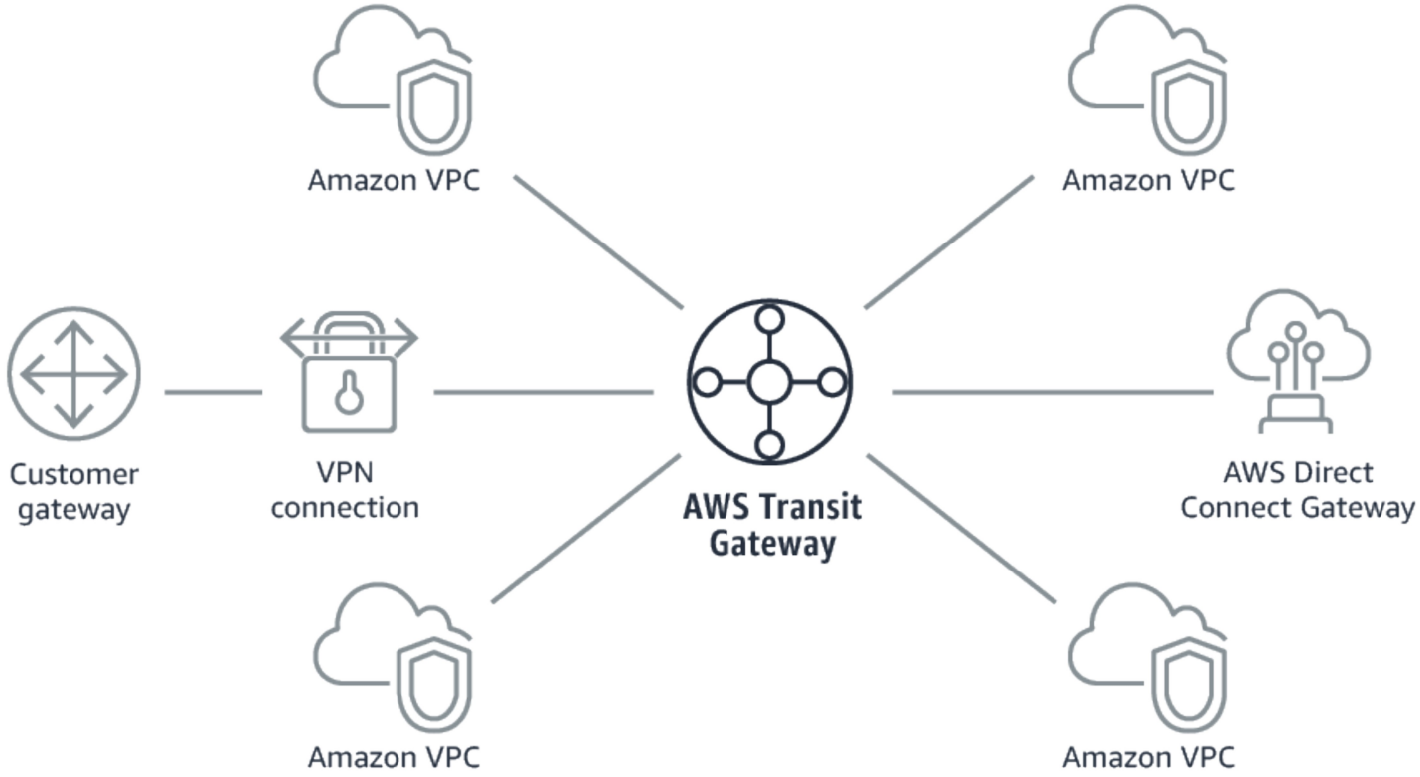
AWS Transit Gateway is a service that enables customers to connect their Amazon Virtual Private Clouds (VPCs) and their on-premises networks to a single gateway. With AWS Transit Gateway, you only have to create and manage a single connection from the central gateway into each Amazon VPC, on-premises data center, or remote office across your network. Transit Gateway acts as a hub that controls how traffic is routed among all the connected networks which act like spokes. So, this is a perfect use-case for the Transit Gateway.

Without Transit Gateway



via - https://aws.amazon.com/transit-gateway/

With Transit Gateway



via - https://aws.amazon.com/transit-gateway/

Incorrect options:

VPC Peering - A VPC peering connection is a networking connection between two VPCs that enables you to route traffic between them using private IPv4 addresses or IPv6 addresses. Instances in either VPC can communicate with each other as if they are within the same network. You can create a VPC peering connection between your VPCs, or with a VPC in another AWS account. The VPCs can be in different regions (also known as an inter-region VPC peering connection). VPC Peering helps connect two VPCs and is not transitive. It would require to create many peering connections between all the VPCs to have them connect. This alone wouldn't work, because we would need to also connect the on-premises data center through Direct Connect and Direct Connect Gateway, but that's not mentioned in this answer.

VPN Gateway - A virtual private gateway (also known as a VPN Gateway) is the endpoint on the VPC side of your VPN connection. You can create a virtual private gateway before creating the VPC itself. VPN Gateway is a distractor here because we haven't mentioned a VPN.

Private Link - AWS PrivateLink simplifies the security of data shared with cloud-based applications by eliminating the exposure of data to the public Internet. AWS PrivateLink provides private connectivity between VPCs, AWS services, and on-premises applications, securely on the Amazon network. Private Link is utilized to create a private connection between an application that is fronted by an NLB in an account, and an Elastic Network Interface (ENI) in another account, without the need of VPC peering, and allowing the connections between the two to remain within the AWS network.

References:

https://aws.amazon.com/transit-gateway/

https://docs.aws.amazon.com/vpc/latest/peering/what-is-vpc-peering.html

https://docs.aws.amazon.com/AWSEC2/latest/APIReference/API_CreateVpnGateway.html

Pregunta 48:
An application is currently hosted on four EC2 instances (behind Application Load Balancer) deployed in a single Availability Zone (AZ). To maintain an acceptable level of end-user experience, the application needs at least 4 instances to be always available.

As a solutions architect, which of the following would you recommend so that the application achieves high availability with MINIMUM cost?

- **Deploy the instances in one Availability Zones. Launch two instances in the Availability Zone**
- **Deploy the instances in two Availability Zones. Launch four instances in each Availability Zone**
- **Deploy the instances in two Availability Zones. Launch two instances in each Availability Zone**
- **Deploy the instances in three Availability Zones. Launch two instances in each Availability Zone(Correcto)**

Explicación
Correct option:

Deploy the instances in three Availability Zones. Launch two instances in each Availability Zone

The correct option is to deploy the instances in three Availability Zones and launch two instances in each Availability Zone. Even if one of the AZs goes out of service, still we shall have 4 instances available and the application can maintain an acceptable level of end-user experience. Therefore, we can achieve high availability with just 6 instances in this case.

Incorrect options:

Deploy the instances in two Availability Zones. Launch two instances in each Availability Zone - When we launch two instances in two AZs, we run the risk of falling below the minimum acceptable threshold of 4 instances if one of the AZs fails. So this option is ruled out.

Deploy the instances in two Availability Zones. Launch four instances in each Availability Zone - When we launch four instances in two AZs, we have to bear costs for 8 instances which is NOT cost-optimal. So this option is ruled out.

Deploy the instances in one Availability Zones. Launch two instances in the Availability Zone - We can't have just two instances in a single AZ as that is below the minimum acceptable threshold of 4 instances.

Pregunta 49:
A weather forecast agency collects key weather metrics across multiple cities in the US and sends this data in the form of key-value pairs to AWS Cloud at a one-minute frequency.

As a solutions architect, which of the following AWS services would you use to build a solution for processing and then reliably storing this data with high availability? (Select two)

- **Redshift**
- **DynamoDB(Correcto)**
- **Lambda(Correcto)**
- **RDS**
- **ElastiCache**

Explicación
Correct options:

Lambda - With AWS Lambda, you can run code without provisioning or managing servers. You pay only for the compute time that you consume—there's no charge when your code isn't running. You can run code for virtually any type of application or backend service—all with zero administration.

DynamoDB - Amazon DynamoDB is a key-value and document database that delivers single-digit millisecond performance at any scale. It's a fully managed, multi-region, multi-master, durable database with built-in security, backup and restore, and in-memory caching for internet-scale applications. DynamoDB is a NoSQL database and it's best suited to store data in key-value pairs.

AWS Lambda can be combined with DynamoDB to process and capture the key-value data from the IoT sources described in the use-case. So both these options are correct.

Incorrect options:

Redshift - Amazon Redshift is a fully-managed petabyte-scale cloud-based data warehouse product designed for large scale data set storage and analysis. You cannot use Redshift to capture data in key-value pairs from the IoT sources, so this option is not correct.

ElastiCache - Amazon ElastiCache allows you to seamlessly set up, run, and scale popular open-Source compatible in-memory data stores in the cloud. Build data-intensive apps or boost the performance of your existing databases by retrieving data from high throughput and low latency in-memory data stores. Amazon ElastiCache is a popular choice for real-time use cases like Caching, Session Stores, Gaming, Geospatial Services, Real-Time Analytics, and Queuing. Elasticache is used as a caching layer in front of relational databases. It is not a good fit to store data in key-value pairs from the IoT sources, so this option is not correct.

RDS - Amazon Relational Database Service (Amazon RDS) makes it easy to set up, operate, and scale a relational database in the cloud. It provides cost-efficient and resizable capacity while automating time-consuming administration tasks such as hardware provisioning, database setup, patching, and backups. Relational databases are not a good fit to store data in key-value pairs, so this option is not correct.

References:

https://aws.amazon.com/dynamodb/

https://aws.amazon.com/lambda/faqs/

Pregunta 50:
A developer has configured inbound traffic for the relevant ports in both the Security Group of the EC2 instance as well as the Network Access Control List (NACL) of the subnet for the EC2 instance. The developer is, however, unable to connect to the service running on the Amazon EC2 instance.

As a solutions architect, how will you fix this issue?

- **Rules associated with Network ACLs should never be modified from command line. An attempt to modify rules from command line blocks the rule and results in an erratic behavior**
- **Network ACLs are stateful, so allowing inbound traffic to the necessary ports enables the connection. Security Groups are stateless, so you must allow both inbound and outbound traffic**
- **Security Groups are stateful, so allowing inbound traffic to the necessary ports enables the connection. Network ACLs are stateless, so you must allow both inbound and outbound traffic(Correcto)**
- **IAM Role defined in the Security Group is different from the IAM Role that is given access in the Network ACLs**

Explicación
Correct option:

Security Groups are stateful, so allowing inbound traffic to the necessary ports enables the connection. Network ACLs are stateless, so you must allow both inbound and outbound traffic - Security groups are stateful, so allowing inbound traffic to the necessary ports enables the connection. Network ACLs are stateless, so you must allow both inbound and outbound traffic.

To enable the connection to a service running on an instance, the associated network ACL must allow both inbound traffic on the port that the service is listening on as well as allow outbound traffic from ephemeral ports. When a client connects to a server, a random port from the ephemeral port range (1024-65535) becomes the client's source port.

The designated ephemeral port then becomes the destination port for return traffic from the service, so outbound traffic from the ephemeral port must be allowed in the network ACL.

By default, network ACLs allow all inbound and outbound traffic. If your network ACL is more restrictive, then you need to explicitly allow traffic from the ephemeral port range.

If you accept traffic from the internet, then you also must establish a route through an internet gateway. If you accept traffic over VPN or AWS Direct Connect, then you must establish a route through a virtual private gateway.

Incorrect options:

Network ACLs are stateful, so allowing inbound traffic to the necessary ports enables the connection. Security Groups are stateless, so you must allow both inbound and outbound traffic - This is incorrect as already discussed.

IAM Role defined in the Security Group is different from the IAM Role that is given access in the Network ACLs - This is a made-up option and just added as a distractor.

Rules associated with Network ACLs should never be modified from command line. An attempt to modify rules from command line blocks the rule and results in an erratic behavior - This option is a distractor. AWS does not support modifying rules of Network ACLs from the command line tool.

Reference:

https://aws.amazon.com/premiumsupport/knowledge-center/resolve-connection-sg-acl-inbound/

Pregunta 51:
The engineering team at a logistics company has noticed that the Auto Scaling group (ASG) is not terminating an unhealthy Amazon EC2 instance.

As a Solutions Architect, which of the following options would you suggest to troubleshoot the issue? (Select three)

- **A custom health check might have failed. ASG does not terminate instances that are set unhealthy by custom checks**
- **The EC2 instance could be a spot instance type, which cannot be terminated by ASG**
- **The instance has failed the ELB health check status(Correcto)**
- **A user might have updated the configuration of ASG and increased the minimum number of instances forcing ASG to keep all instances alive**
- **The health check grace period for the instance has not expired(Correcto)**
- **The instance maybe in Impaired status(Correcto)**

Explicación
Correct options:

The health check grace period for the instance has not expired - Amazon EC2 Auto Scaling doesn't terminate an instance that came into service based on EC2 status checks and ELB health checks until the health check grace period expires.

More on Health check grace period:

Health check grace period

When an instance launches, Amazon EC2 Auto Scaling uses the value of the `HealthCheckGracePeriod` for the Auto Scaling group to determine how long to wait before checking the health status of the instance. Amazon EC2 and Elastic Load Balancing health checks can complete before the health check grace period expires. However, Amazon EC2 Auto Scaling does not act on them until the health check grace period expires.

By default, the health check grace period is `300` seconds when you create an Auto Scaling group from the AWS Management Console. Its default value is `0` seconds when you create an Auto Scaling group using the AWS CLI or an AWS SDK.

To provide ample warm-up time for your instances, ensure that the health check grace period covers the expected startup time for your application, from when an instance comes into service to when it can receive traffic. If you add a lifecycle hook, the grace period does not start until the lifecycle hook actions are completed and the instance enters the `InService` state.

via - https://docs.aws.amazon.com/autoscaling/ec2/userguide/healthcheck.html#health-check-grace-period

The instance maybe in Impaired status - Amazon EC2 Auto Scaling does not immediately terminate instances with an Impaired status. Instead, Amazon EC2 Auto Scaling waits a few minutes for the instance to recover. Amazon EC2 Auto Scaling might also delay or not terminate instances that fail to report data for status checks. This usually happens when there is insufficient data for the status check metrics in Amazon CloudWatch.

The instance has failed the ELB health check status - By default, Amazon EC2 Auto Scaling doesn't use the results of ELB health checks to determine an instance's health status when the group's health check configuration is set to EC2. As a result, Amazon EC2 Auto Scaling doesn't terminate instances that fail ELB health checks. If an instance's status is OutofService on the ELB console, but the instance's status is Healthy on the Amazon EC2 Auto Scaling console, confirm that the health check type is set to ELB.

Incorrect options:

The EC2 instance could be a spot instance type, which cannot be terminated by ASG - This is an incorrect statement. Amazon EC2 Auto Scaling terminates Spot instances when capacity is no longer available or the Spot price exceeds your maximum price.

A user might have updated the configuration of ASG and increased the minimum number of instances forcing ASG to keep all instances alive - This statement is incorrect. If the configuration is updated and ASG needs more number of instances, ASG will launch new, healthy instances and does not keep unhealthy ones alive.

A custom health check might have failed. ASG does not terminate instances that are set unhealthy by custom checks - This statement is incorrect. You can define custom health checks in Amazon EC2 Auto Scaling. When a custom health check determines that an instance is unhealthy, the check manually triggers SetInstanceHealth and then sets the instance's state to Unhealthy. Amazon EC2 Auto Scaling then terminates the unhealthy instance.

References:

https://aws.amazon.com/premiumsupport/knowledge-center/auto-scaling-terminate-instance/

https://aws.amazon.com/premiumsupport/knowledge-center/auto-scaling-instance-how-terminated/

Pregunta 52:
A social media application is hosted on an EC2 server fleet running behind an Application Load Balancer. The application traffic is fronted by a CloudFront distribution. The engineering team wants to decouple the user authentication process for the application, so that the application servers can just focus on the business logic.

As a Solutions Architect, which of the following solutions would you recommend to the development team so that it requires minimal development effort?

- **Use Cognito Authentication via Cognito User Pools for your CloudFront distribution**
- **Use Cognito Authentication via Cognito Identity Pools for your Application Load Balancer**
- **Use Cognito Authentication via Cognito Identity Pools for your CloudFront distribution**
- **Use Cognito Authentication via Cognito User Pools for your Application Load Balancer(Correcto)**

Explicación
Correct option:

Use Cognito Authentication via Cognito User Pools for your Application Load Balancer

Application Load Balancer can be used to securely authenticate users for accessing your applications. This enables you to offload the work of authenticating users to your load balancer so that your applications can focus on their business logic. You can use Cognito User Pools to authenticate users through well-known social IdPs, such as Amazon, Facebook, or Google, through the user pools supported by Amazon Cognito or through corporate identities, using SAML, LDAP, or Microsoft AD, through the user pools supported by Amazon Cognito. You configure user authentication by creating an authenticate action for one or more listener rules.

Authenticate users using an Application Load Balancer

PDF | Kindle | RSS

You can configure an Application Load Balancer to securely authenticate users as they access your applications. This enables you to offload the work of authenticating users to your load balancer so that your applications can focus on their business logic.

The following use cases are supported:

- Authenticate users through an identity provider (IdP) that is OpenID Connect (OIDC) compliant.
- Authenticate users through well-known social IdPs, such as Amazon, Facebook, or Google, through the user pools supported by Amazon Cognito.
- Authenticate users through corporate identities, using SAML, LDAP, or Microsoft AD, through the user pools supported by Amazon Cognito.

---

via - https://docs.aws.amazon.com/elasticloadbalancing/latest/application/listener-authenticate-users.html

Please review the following note to understand the differences between Cognito User Pools and Cognito Identity Pools:

Features of Amazon Cognito

User pools

A user pool is a user directory in Amazon Cognito. With a user pool, your users can sign in to your web or mobile app through Amazon Cognito, or federate through a third-party identity provider (IdP). Whether your users sign in directly or through a third party, all members of the user pool have a directory profile that you can access through an SDK.

User pools provide:

- Sign-up and sign-in services.
- A built-in, customizable web UI to sign in users.
- Social sign-in with Facebook, Google, Login with Amazon, and Sign in with Apple, and through SAML and OIDC identity providers from your user pool.
- User directory management and user profiles.
- Security features such as multi-factor authentication (MFA), checks for compromised credentials, account takeover protection, and phone and email verification.
- Customized workflows and user migration through AWS Lambda triggers.

For more information about user pools, see Getting Started with User Pools and the Amazon Cognito User Pools API Reference.

Identity pools

With an identity pool, your users can obtain temporary AWS credentials to access AWS services, such as Amazon S3 and DynamoDB. Identity pools support anonymous guest users, as well as the following identity providers that you can use to authenticate users for identity pools:

- Amazon Cognito user pools
- Social sign-in with Facebook, Google, Login with Amazon, and Sign in with Apple
- OpenID Connect (OIDC) providers
- SAML identity providers
- Developer authenticated identities

To save user profile information, your identity pool needs to be integrated with a user pool.

For more information about identity pools, see Getting Started with Amazon Cognito Identity Pools (Federated Identities) and the Amazon Cognito Identity Pools API Reference.

via - https://docs.aws.amazon.com/cognito/latest/developerguide/what-is-amazon-cognito.html

Incorrect options:

Use Cognito Authentication via Cognito Identity Pools for your Application Load Balancer - There is no such thing as using Cognito Authentication via Cognito Identity Pools for managing user authentication for the application. Application-specific user authentication can be provided via Cognito User Pools. Amazon Cognito identity pools provide temporary AWS credentials for users who are guests (unauthenticated) and for users who have been authenticated and received a token.

Use Cognito Authentication via Cognito User Pools for your CloudFront distribution - You cannot directly integrate Cognito User Pools with CloudFront distribution as you have to create a separate Lambda@Edge function to accomplish the authentication via Cognito User Pools. This involves additional development effort, so this option is not the best fit for the given use-case.

Use Cognito Authentication via Cognito Identity Pools for your CloudFront distribution - You cannot use Cognito Identity Pools for managing user authentication, so this option is not correct.

References:

https://docs.aws.amazon.com/elasticloadbalancing/latest/application/listener-authenticate-users.html

https://docs.aws.amazon.com/cognito/latest/developerguide/cognito-user-identity-pools.html

https://aws.amazon.com/blogs/networking-and-content-delivery/authorizationedge-using-cookies-protect-your-amazon-cloudfront-content-from-being-downloaded-by-unauthenticated-users/

Pregunta 53:
An IT company is working on a client project to build a Supply Chain Management application. The web-tier of the application runs on an EC2 instance and the database tier is on Amazon RDS MySQL. For beta testing, all the resources are currently deployed in a single Availability Zone. The development team wants to improve application availability before the go-live.

Given that all end users of the web application would be located in the US, which of the following would be the MOST resource-efficient solution?

- **Deploy the web-tier EC2 instances in two regions, behind an Elastic Load Balancer. Deploy the Amazon RDS MySQL database in read replica configuration**
- **Deploy the web-tier EC2 instances in two Availability Zones, behind an Elastic Load Balancer. Deploy the Amazon RDS MySQL database in Multi-AZ configuration(Correcto)**
- **Deploy the web-tier EC2 instances in two regions, behind an Elastic Load Balancer. Deploy the Amazon RDS MySQL database in Multi-AZ configuration**
- **Deploy the web-tier EC2 instances in two Availability Zones, behind an Elastic Load Balancer. Deploy the Amazon RDS MySQL database in read replica configuration**

Explicación
Correct option:

Deploy the web-tier EC2 instances in two Availability Zones, behind an Elastic Load Balancer. Deploy the Amazon RDS MySQL database in Multi-AZ configuration

Elastic Load Balancing automatically distributes incoming application traffic across multiple targets, such as Amazon EC2 instances, containers, IP addresses, and Lambda functions. It can handle the varying load of your application traffic in a single Availability Zone or across multiple Availability Zones. Therefore, deploying the web-tier EC2 instances in two Availability Zones, behind an Elastic Load Balancer would improve the availability of the application.

Amazon RDS Multi-AZ deployments provide enhanced availability and durability for RDS database (DB) instances, making them a natural fit for production database workloads. When you provision a Multi-AZ DB Instance, Amazon RDS automatically creates a primary DB Instance and synchronously replicates the data to a standby instance in a different Availability Zone (AZ). Each AZ runs on its own physically distinct, independent infrastructure, and is engineered to be highly reliable. Deploying the Amazon RDS MySQL database in Multi-AZ configuration would improve availability and hence this is the correct option.

Incorrect options:

Deploy the web-tier EC2 instances in two Availability Zones, behind an Elastic Load Balancer. Deploy the Amazon RDS MySQL database in read replica configuration

Deploy the web-tier EC2 instances in two regions, behind an Elastic Load Balancer. Deploy the Amazon RDS MySQL database in read replica configuration

Amazon RDS Read Replicas provide enhanced performance and durability for RDS database (DB) instances. They make it easy to elastically scale out beyond the capacity constraints of a single DB instance for read-heavy database workloads. Read replicas are meant to address scalability issues. You cannot use read replicas for improving availability, so both these options are incorrect.

Please review this comparison vis-a-vis Multi-AZ vs Read Replica for RDS:

Multi-AZ deployments, multi-region deployments, and read replicas

Amazon RDS Multi-AZ deployments complement multi-region deployments and read replicas. While all three features increase availability and durability by maintaining additional copies of your data, there are differences between them:

Multi-AZ deployments	Multi-Region deployments	Read replicas
Main purpose is high availability	Main purpose is disaster recovery and local performance	Main purpose is scalability
Non-Aurora: synchronous replication; Aurora: asynchronous replication	Asynchronous replication	Asynchronous replication
Non-Aurora: only the primary instance is active; Aurora: all instances are active	All regions are accessible and can be used for reads	All read replicas are accessible and can be used for readscaling
Non-Aurora: automated backups are taken from standby; Aurora: automated backups are taken from shared storage layer	Automated backups can be taken in each region	No backups configured by default
Always span at least two Availability Zones within a single region	Each region can have a Multi-AZ deployment	Can be within an Availability Zone, Cross-AZ, or Cross-Region
Non-Aurora: database engine version upgrades happen on primary; Aurora: all instances are updated together	Non-Aurora: database engine version upgrade is independent in each region; Aurora: all instances are updated together	Non-Aurora: database engine version upgrade is independent from source instance; Aurora: all instances are updated together
Automatic failover to standby (non-Aurora) or read replica (Aurora) when a problem is detected	Aurora allows promotion of a secondary region to be the master	Can be manually promoted to a standalone database instance (non-Aurora) or to be the primary instance (Aurora)

via - https://aws.amazon.com/rds/features/multi-az/

Deploy the web-tier EC2 instances in two regions, behind an Elastic Load Balancer. Deploy the Amazon RDS MySQL database in Multi-AZ configuration - As Elastic Load Balancing does not work across regions, so this option is incorrect.

Reference:

https://aws.amazon.com/rds/features/multi-az/

Pregunta 54:
An HTTP application is deployed on an Auto Scaling Group, is accessible from an Application Load Balancer that provides HTTPS termination, and accesses a PostgreSQL database managed by RDS.

How should you configure the security groups? (Select three)

- **The security group of the ALB should have an inbound rule from anywhere on port 443(Correcto)**
- **The security group of the EC2 instances should have an inbound rule from the security group of the ALB on port 80(Correcto)**
- **The security group of the ALB should have an inbound rule from anywhere on port 80**
- **The security group of RDS should have an inbound rule from the security group of the EC2 instances in the ASG on port 80**
- **The security group of the EC2 instances should have an inbound rule from the security group of the RDS database on port 5432**
- **The security group of RDS should have an inbound rule from the security group of the EC2 instances in the ASG on port 5432(Correcto)**

Explicación
Correct options:

The security group of RDS should have an inbound rule from the security group of the EC2 instances in the ASG on port 5432

The security group of the EC2 instances should have an inbound rule from the security group of the ALB on port 80

The security group of the ALB should have an inbound rule from anywhere on port 443

A security group acts as a virtual firewall that controls the traffic for one or more instances. When you launch an instance, you can specify one or more security groups; otherwise, we use the default security group. You can add rules to each security group that allows traffic to or from its associated instances. You can modify the rules for a security group at any time; the new rules are automatically applied to all instances that are associated with the security group. When we decide whether to allow traffic to reach an instance, we evaluate all the rules from all the security groups that are associated with the instance. The following are the characteristics of security group rules: By default, security groups allow all outbound traffic. Security group rules are always permissive; you can't create rules that deny access. Security groups are stateful

PostgreSQL port = 5432 HTTP port = 80 HTTPS port = 443

The traffic goes like this : The client sends an HTTPS request to ALB on port 443. This is handled by the rule - **The security group of the ALB should have an inbound rule from anywhere on port 443.** The ALB then forwards the request to one of the EC2 instances. This is handled by the rule - **The security group of the EC2 instances should have an inbound rule from the security group of the ALB on port 80.** The EC2 instance further accesses the PostgreSQL database managed by RDS on port 5432. This is handled by the rule - **The security group of RDS should have an inbound rule from the security group of the EC2 instances in the ASG on port 5432.**

Incorrect options:

The security group of the ALB should have an inbound rule from anywhere on port 80 - The client sends an HTTPS request to ALB on port 443 and not on port 80, so this is incorrect.

The security group of the EC2 instances should have an inbound rule from the security group of the RDS database on port 5432 - The security group of the EC2 instances should have an inbound rule from the security group of the ALB and not from the security group of the RDS database, so this option is incorrect.

The security group of RDS should have an inbound rule from the security group of the EC2 instances in the ASG on port 80 - The EC2 instance further accesses the PostgreSQL database managed by RDS on port 5432 and not on port 80, so this option is incorrect.

Reference:

https://docs.aws.amazon.com/AWSEC2/latest/UserGuide/ec2-security-groups.html

Pregunta 55:
A health-care solutions company wants to run their applications on single-tenant hardware to meet regulatory guidelines.

Which of the following is the MOST cost-effective way of isolating their Amazon EC2 instances to a single tenant?

- **Dedicated Instances(Correcto)**
- **On-Demand Instances**
- **Dedicated Hosts**
- **Spot Instances**

Explicación
Correct option:

Dedicated Instances - Dedicated Instances are Amazon EC2 instances that run in a virtual private cloud (VPC) on hardware that's dedicated to a single customer. Dedicated Instances that belong to different AWS accounts are physically isolated at a hardware level, even if those accounts are linked to a single-payer account. However, Dedicated Instances may share hardware with other instances from the same AWS account that are not Dedicated Instances.

A Dedicated Host is also a physical server that's dedicated for your use. With a Dedicated Host, you have visibility and control over how instances are placed on the server.

Differences between Dedicated Hosts and Dedicated Instances:

Differences between Dedicated Hosts and Dedicated Instances

Dedicated Hosts and Dedicated Instances can both be used to launch Amazon EC2 instances onto physical servers that are dedicated for your use.

There are no performance, security, or physical differences between Dedicated Instances and instances on Dedicated Hosts. However, there are some differences between the two. The following table highlights some of the key differences between Dedicated Hosts and Dedicated Instances:

	Dedicated Host	Dedicated Instance
Billing	Per-host billing	Per-instance billing
Visibility of sockets, cores, and host ID	Provides visibility of the number of sockets and physical cores	No visibility
Host and instance affinity	Allows you to consistently deploy your instances to the same physical server over time	Not supported
Targeted instance placement	Provides additional visibility and control over how instances are placed on a physical server	Not supported
Automatic instance recovery	Supported. For more information, see Host recovery.	Supported
Bring Your Own License (BYOL)	Supported	Not supported

via - https://docs.aws.amazon.com/AWSEC2/latest/UserGuide/dedicated-hosts-overview.html#dedicated-hosts-dedicated-instances

Incorrect options:

Spot Instances - A Spot Instance is an unused EC2 instance that is available for less than the On-Demand price. Your Spot Instance runs whenever capacity is available and the maximum price per hour for your request exceeds the Spot price. Any instance present with unused capacity will be allocated. Even though this is cost-effective, it does not fulfill the single-tenant hardware requirement of the client and hence is not the correct option.

Dedicated Hosts - An Amazon EC2 Dedicated Host is a physical server with EC2 instance capacity fully dedicated to your use. Dedicated Hosts allow you to use your existing software licenses on EC2 instances. With a Dedicated Host, you have visibility and control over how instances are placed on the server. This option is costlier than the Dedicated Instance and hence is not the right choice for the current requirement.

On-Demand Instances - With On-Demand Instances, you pay for compute capacity by the second with no long-term commitments. You have full control over its lifecycle—you decide when to launch, stop, hibernate, start, reboot, or terminate it. Hardware isolation is not possible and on-demand has one of the costliest instance charges and hence is not the correct answer for current requirements.

High Level Overview of EC2 Instance Purchase Options:

On-Demand

With On-Demand instances, you pay for compute capacity by the hour or the second depending on which instances you run. No longer-term commitments or upfront payments are needed. You can increase or decrease your compute capacity depending on the demands of your application and only pay the specified per hourly rates for the instance you use.

On-Demand instances are recommended for:

- Users that prefer the low cost and flexibility of Amazon EC2 without any up-front payment or long-term commitment
- Applications with short-term, spiky, or unpredictable workloads that cannot be interrupted
- Applications being developed or tested on Amazon EC2 for the first time

See On-Demand pricing »

Spot instances

Amazon EC2 Spot instances allow you to request spare Amazon EC2 computing capacity for up to 90% off the On-Demand price. Learn More.

Spot instances are recommended for:

- Applications that have flexible start and end times
- Applications that are only feasible at very low compute prices
- Users with urgent computing needs for large amounts of additional capacity

See Spot pricing »

Savings Plans

Savings Plans are a flexible pricing model that offer low prices on EC2 and Fargate usage, in exchange for a commitment to a consistent amount of usage (measured in $/hour) for a 1 or 3 year term.

Dedicated Hosts

A Dedicated Host is a physical EC2 server dedicated for your use. Dedicated Hosts can help you reduce costs by allowing you to use your existing server-bound software licenses, including Windows Server, SQL Server, and SUSE Linux Enterprise Server (subject to your license terms), and can also help you meet compliance requirements. Learn more.

- Can be purchased On-Demand (hourly).
- Can be purchased as a Reservation for up to 70% off the On-Demand price.

See Dedicated pricing »

Reserved Instances

Reserved Instances provide you with a significant discount (up to 75%) compared to On-Demand instance pricing. In addition, when Reserved Instances are assigned to a specific Availability Zone, they provide a capacity reservation, giving you additional confidence in your ability to launch instances when you need them.

For applications that have steady state or predictable usage, Reserved Instances can provide significant savings compared to using On-Demand instances. See How to Purchase Reserved Instances for more information.

Reserved Instances are recommended for:

- Applications with steady state usage
- Applications that may require reserved capacity
- Customers that can commit to using EC2 over a 1 or 3 year term to reduce their total computing costs

via -

https://aws.amazon.com/ec2/pricing/

References:

https://docs.aws.amazon.com/AWSEC2/latest/UserGuide/dedicated-instance.html

https://docs.aws.amazon.com/AWSEC2/latest/UserGuide/instance-purchasing-options.html

Pregunta 56:
A media company is migrating its flagship application from its on-premises data center to AWS for improving the application's read-scaling capability as well as its availability. The existing architecture leverages a Microsoft SQL Server database that sees a heavy read load. The engineering team does a full copy of the production database every 5 hours to populate a dev database. During this period, application users face high latency leading to a bad user experience.

The company is looking at alternate database options and migrating database engines if required. What would you suggest?

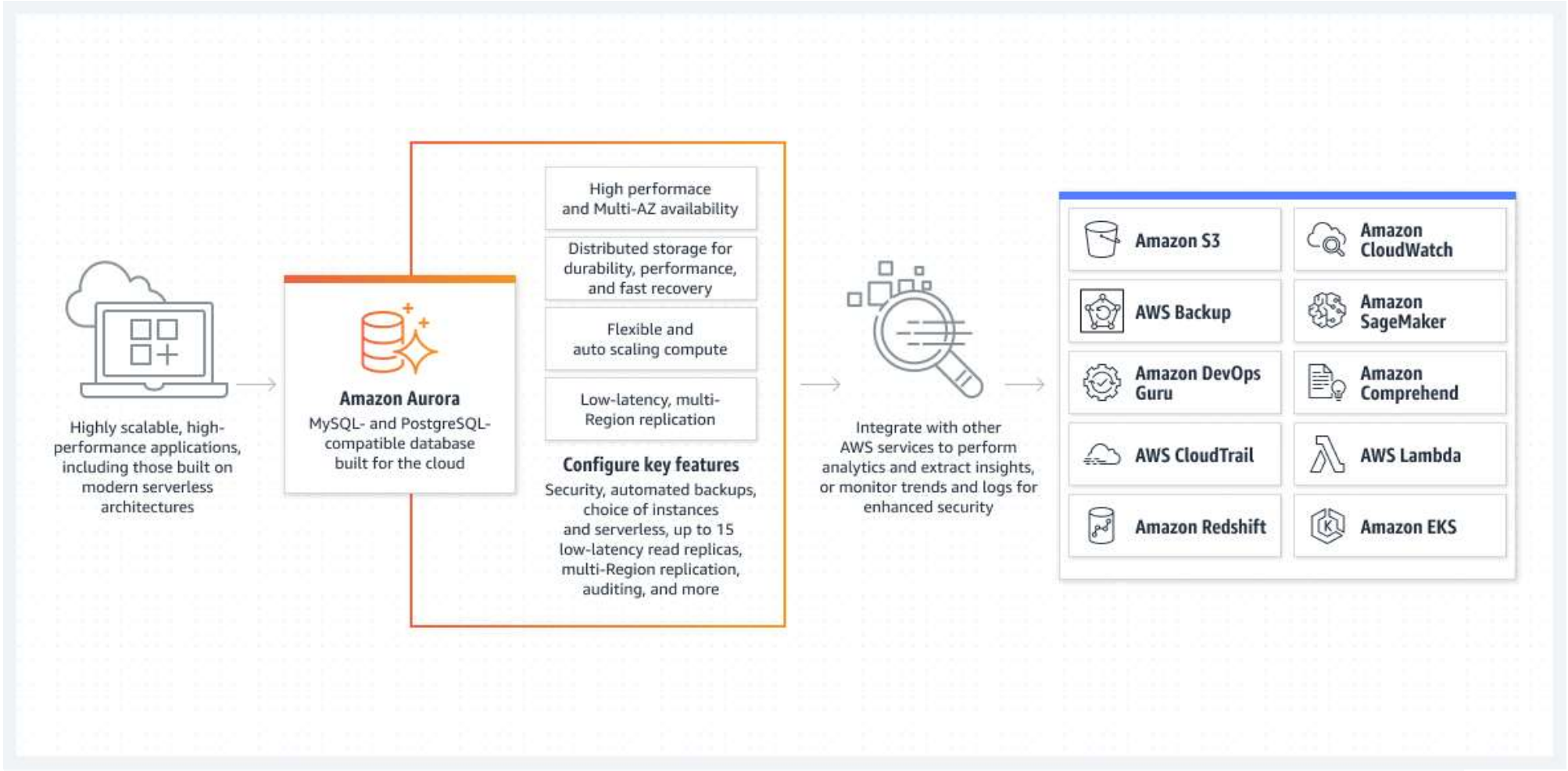
- **Leverage Amazon Aurora MySQL with Multi-AZ Aurora Replicas and create the dev database by restoring from the automated backups of Amazon Aurora(Correcto)**
- **Leverage Amazon RDS for MySQL with a Multi-AZ deployment and use the standby instance as the dev database**
- **Leverage Amazon Aurora MySQL with Multi-AZ Aurora Replicas and restore the dev database via mysqldump**
- **Leverage Amazon RDS for SQL server with a Multi-AZ deployment and read replicas. Use the read replica as the dev database**

Explicación
Correct option:

Leverage Amazon Aurora MySQL with Multi-AZ Aurora Replicas and create the dev database by restoring from the automated backups of Amazon Aurora

Amazon Aurora (Aurora) is a fully managed relational database engine that's compatible with MySQL and PostgreSQL. An Amazon Aurora DB cluster consists of one or more DB instances and a cluster volume that manages the data for those DB instances. An Aurora cluster volume is a virtual database storage volume that spans multiple Availability Zones, with each Availability Zone having a copy of the DB cluster data. Aurora supports Multi-AZ Aurora Replicas that improve the application's read-scaling and availability.

Amazon Aurora Overview:



via - https://aws.amazon.com/rds/aurora/

Aurora backs up your cluster volume automatically and retains restore data for the length of the backup retention period. Aurora backups are continuous and incremental so you can quickly restore to any point within the backup retention period. No performance impact or interruption of database service occurs as backup data is being written.

Backups

Aurora backs up your cluster volume automatically and retains restore data for the length of the *backup retention period*. Aurora backups are continuous and incremental so you can quickly restore to any point within the backup retention period. No performance impact or interruption of database service occurs as backup data is being written. You can specify a backup retention period, from 1 to 35 days, when you create or modify a DB cluster. Aurora backups are stored in Amazon S3.

If you want to retain a backup beyond the backup retention period, you can also take a snapshot of the data in your cluster volume. Because Aurora retains incremental restore data for the entire backup retention period, you only need to create a snapshot for data that you want to retain beyond the backup retention period. You can create a new DB cluster from the snapshot.

> **Note**
>
> - For Amazon Aurora DB clusters, the default backup retention period is one day regardless of how the DB cluster is created.
> - You can't disable automated backups on Aurora. The backup retention period for Aurora is managed by the DB cluster.

via - https://docs.aws.amazon.com/AmazonRDS/latest/AuroraUserGuide/Aurora.Managing.Backups.html

Automated backups occur daily during the preferred backup window. If the backup requires more time than allotted to the backup window, the backup continues after the window ends, until it finishes. The backup window can't overlap with the weekly maintenance window for the DB cluster. Aurora backups are continuous and incremental, but the backup window is used to create a daily system backup that is preserved within the backup retention period. The latest restorable time for a DB cluster is the most recent point at which you can restore your DB cluster, typically within 5 minutes of the current time.

For the given use case, you can create the dev database by restoring from the automated backups of Amazon Aurora.

Incorrect options:

Leverage Amazon Aurora MySQL with Multi-AZ Aurora Replicas and restore the dev database via mysqldump - Restoring the dev database via mysqldump would still result in a significant load on the primary DB, so this option fails to address the given requirement.

Leverage Amazon RDS for MySQL with a Multi-AZ deployment and use the standby instance as the dev database - The standby is there just for handling failover in a Multi-AZ deployment. You cannot access the standby instance and use it as a dev database. Hence this option is incorrect.

Leverage Amazon RDS for SQL server with a Multi-AZ deployment and read replicas. Use the read replica as the dev database - Amazon RDS supports Multi-AZ deployments for Microsoft SQL Server by using either SQL Server Database Mirroring (DBM) or Always On Availability Groups (AGs). Amazon RDS monitors and maintains the health of your Multi-AZ deployment.

Multi-AZ deployments provide increased availability, data durability, and fault tolerance for DB instances. In the event of planned database maintenance or unplanned service disruption, Amazon RDS automatically fails over to the up-to-date secondary DB instance. For SQL Server, I/O activity is suspended briefly during backup for Multi-AZ deployments.

A read replica is only meant to serve read traffic. The primary purpose of the read replica is to replicate the data in the primary DB instance. A read replica cannot be used as a dev database because it does not allow any database write operations.

References:

https://docs.aws.amazon.com/AmazonRDS/latest/AuroraUserGuide/Aurora.Managing.Backups.html

https://docs.aws.amazon.com/AmazonRDS/latest/UserGuide/SQLServer.ReadReplicas.html

Pregunta 57:
You would like to mount a network file system on Linux instances, where files will be stored and accessed frequently at first, and then infrequently. What solution is the MOST cost-effective?

- **Glacier Deep Archive**
- **S3 Intelligent Tiering**
- **EFS IA(Correcto)**
- **FSx for Lustre**

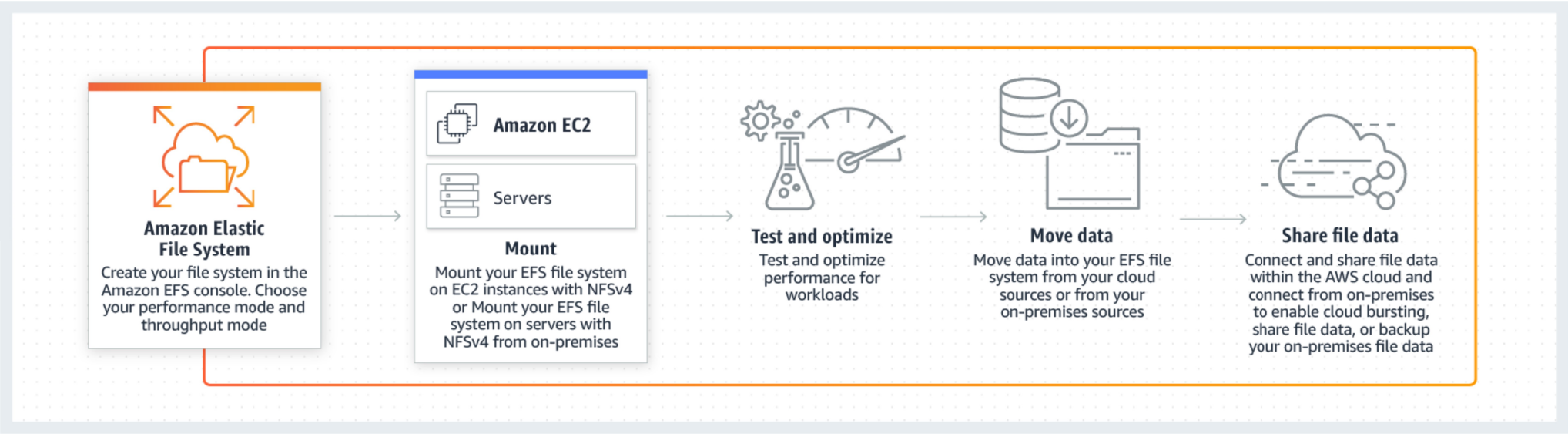
Explicación
Correct option:

EFS IA

Amazon Elastic File System (Amazon EFS) provides a simple, scalable, fully managed elastic NFS file system for use with AWS Cloud services and on-premises resources. Amazon EFS is a regional service storing data within and across multiple Availability Zones (AZs) for high availability and durability.

Amazon EFS Infrequent Access (EFS IA) is a storage class that provides price/performance that is cost-optimized for files, not accessed every day, with storage prices up to 92% lower compared to Amazon EFS Standard. Therefore, this is the correct option.

How EFS works:



via - https://aws.amazon.com/efs/

Incorrect options:

S3 Intelligent Tiering - Amazon Simple Storage Service (Amazon S3) is an object storage service that offers industry-leading scalability, data availability, security, and performance. The S3 Intelligent-Tiering storage class is designed to optimize costs by automatically moving data to the most cost-effective access tier, without performance impact or operational overhead. It works by storing objects in two access tiers: one tier that is optimized for frequent access and another lower-cost tier that is optimized for infrequent access.

You can't mount a network file system on S3 Intelligent Tiering as it's an object storage service, so this option is incorrect.

Glacier Deep Archive - Amazon S3 Glacier and S3 Glacier Deep Archive are a secure, durable, and extremely low-cost Amazon S3 cloud storage classes for data archiving and long-term backup. They are designed to deliver 99.999999999% durability, and provide comprehensive security and compliance capabilities that can help meet even the most stringent regulatory requirements.

You can't mount a network file system on S3 Intelligent Tiering as it's an object storage/archival service, so this option is incorrect.

FSx for Lustre - Amazon FSx for Lustre makes it easy and cost-effective to launch and run the world's most popular high-performance file system. It is used for workloads such as machine learning, high-performance computing (HPC), video processing, and financial modeling. Amazon FSx enables you to use Lustre file systems for any workload where storage speed matters.

FSx for Lustre is a file system better suited for distributed computing for HPC (high-performance computing) and is very expensive

References:

https://aws.amazon.com/efs/

https://aws.amazon.com/efs/features/infrequent-access/

Pregunta 58:
An e-commerce company operates multiple AWS accounts and has interconnected these accounts in a hub-and-spoke style using the AWS Transit Gateway. VPCs have been provisioned across these AWS accounts to facilitate network isolation.

Which of the following solutions would reduce both the administrative overhead and the costs while providing shared access to services required by workloads in each of the VPCs?

- **Use Fully meshed VPC Peers**
- **Use VPCs connected with AWS Direct Connect**

- **Build a shared services VPC(Correcto)**
- **Use Transit VPC to reduce cost and share the resources across VPCs**
-

Explicación
Correct option:

Build a shared services VPC

Consider an organization that has built a hub-and-spoke network with AWS Transit Gateway. VPCs have been provisioned into multiple AWS accounts, perhaps to facilitate network isolation or to enable delegated network administration. When deploying distributed architectures such as this, a popular approach is to build a "shared services VPC, which provides access to services required by workloads in each of the VPCs. This might include directory services or VPC endpoints. Sharing resources from a central location instead of building them in each VPC may reduce administrative overhead and cost.

Centralized VPC Endpoints (multiple VPCs):

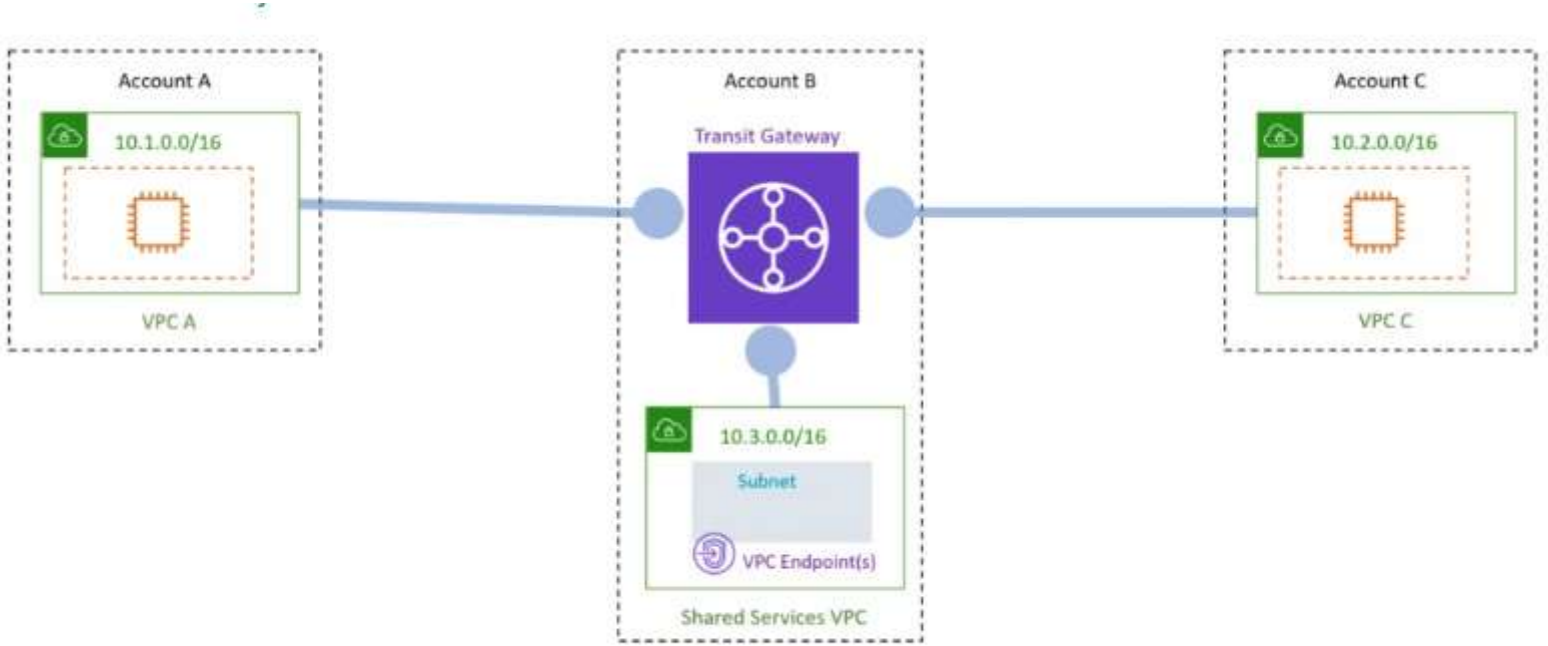


Figure 3: Centralized VPC Endpoints (multiple VPCs)

via - https://aws.amazon.com/blogs/architecture/reduce-cost-and-increase-security-with-amazon-vpc-endpoints/

A VPC endpoint allows you to privately connect your VPC to supported AWS services without requiring an Internet gateway, NAT device, VPN connection, or AWS Direct Connect connection. Endpoints are virtual devices that are horizontally scaled, redundant, and highly available VPC components. They allow communication between instances in your VPC and services without imposing availability risks or bandwidth constraints on your network traffic.

VPC endpoints enable you to reduce data transfer charges resulting from network communication between private VPC resources (such as Amazon Elastic Cloud Compute—or EC2—instances) and AWS Services (such as Amazon Quantum Ledger Database, or

QLDB). Without VPC endpoints configured, communications that originate from within a VPC destined for public AWS services must egress AWS to the public Internet in order to access AWS services. This network path incurs outbound data transfer charges. Data transfer charges for traffic egressing from Amazon EC2 to the Internet vary based on volume. With VPC endpoints configured, communication between your VPC and the associated AWS service does not leave the Amazon network. If your workload requires you to transfer significant volumes of data between your VPC and AWS, you can reduce costs by leveraging VPC endpoints.

Incorrect options:

Use Transit VPC to reduce cost and share the resources across VPCs - Transit VPC uses customer-managed Amazon Elastic Compute Cloud (Amazon EC2) VPN instances in a dedicated transit VPC with an Internet gateway. This design requires the customer to deploy, configure, and manage EC2-based VPN appliances, which will result in additional EC2, and potentially third-party product and licensing charges. Note that this design will generate additional data transfer charges for traffic traversing the transit VPC: data is charged when it is sent from a spoke VPC to the transit VPC, and again from the transit VPC to the on-premises network or a different AWS Region. Transit VPC is not the right choice here.

More on Transit VPC:

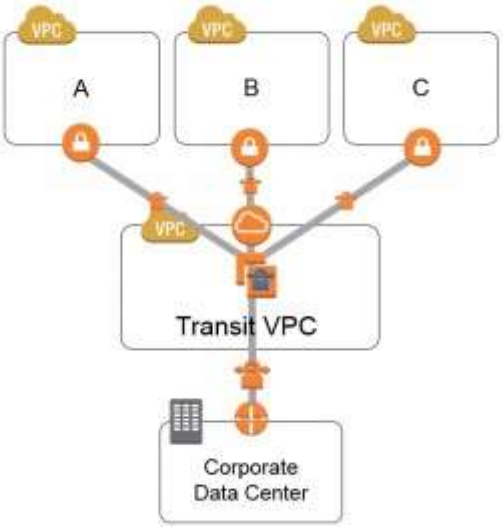
Transit VPC

This approach uses customer-managed Amazon Elastic Compute Cloud (Amazon EC2) VPN instances in a dedicated transit VPC with an Internet gateway. The EC2 instances initiate the VPN connections and route traffic between multiple VPCs and shared-services VPCs. The spoke VPCs can leverage VPC peering to circumvent the transit VPC, providing more scalable, direct access between VPCs.

This design requires the customer to deploy, configure, and manage EC2-based VPN appliances, which will result in additional EC2, and potentially third-party product and licensing charges. Therefore, it is best suited for customers who have already implemented a transit VPC and want to leverage it to manage more advanced connection types, such as inter-region connectivity, or multi-VPC connectivity to on-premises resources.

Note that this design will generate additional data transfer charges for traffic traversing the transit VPC: data is charged when it is sent from a spoke VPC to the transit VPC, and again from the transit VPC to the on-premises network or a different AWS Region.

For additional details on this solution, see the Multiple-VPC VPN Connection Sharing Solution Brief.



via - https://d0.awsstatic.com/aws-answers/AWS_Single_Region_Multi_VPC_Connectivity.pdf

Use Fully meshed VPC Peers - This approach creates multiple peering connections to facilitate the sharing of information between resources in different VPCs. This design connects multiple VPCs in a fully meshed configuration, with peering connections between each pair of VPCs. With this configuration, each VPC has access to the resources in all other VPCs. Each peering connection requires modifications to all the other VPCs' route tables and, as the number of VPCs grows, this can be difficult to maintain. And keep in mind that AWS recommends a maximum of 125 peering connections per VPC. It's complex to manage and isn't a right fit for the current scenario.

More on Fully meshed VPC Peers:

Configuration Details

This design connects multiple VPCs in a fully meshed configuration, with peering connections between each pair of VPCs. With this configuration, each VPC has access to the resources in all other VPCs.

To enable the flow of traffic between VPCs, each VPC route table must contain entries that point to the IP address ranges of all the other VPCs in the fully meshed configuration. This design is more complicated to set up than a partially meshed configuration, but it enables communication across all VPCs in the system.

Considerations

Each peering connection requires modifications to all the other VPCs' route tables and, as the number of VPCs grows, this can be difficult to maintain. And keep in mind that AWS recommends a maximum of 125 peering connections per VPC.

Customers can create VPC peering connections between VPCs in the same account, or with VPCs in a different AWS account, as long as the VPCs are in the same region.

via - https://d0.awsstatic.com/aws-answers/AWS_Single_Region_Multi_VPC_Connectivity.pdf

Use VPCs connected with AWS Direct Connect - This approach is a good alternative for customers who need to connect a high number of VPCs to a central VPC or on-premises resources, or who already have an AWS Direct Connect connection in place. This design also offers customers the ability to incorporate transitive routing into their network design. For example, if VPC A and VPC B are both connected to an on-premises network using AWS Direct Connect connections, then the two VPCs can be connected to each other via AWS Direct Connect. Direct Connect requires physical cables and takes about a month for setting up, this is not an ideal solution for the given scenario.

References:

https://aws.amazon.com/blogs/architecture/reduce-cost-and-increase-security-with-amazon-vpc-endpoints/

https://d0.awsstatic.com/aws-answers/AWS_Single_Region_Multi_VPC_Connectivity.pdf

Pregunta 59:
A company has historically operated only in the `us-east-1` region and stores encrypted data in S3 using SSE-KMS. As part of enhancing its security posture as well as improving the backup and recovery architecture, the company wants to store the encrypted data in S3 that is replicated into the `us-west-1` AWS region. The security policies mandate that the data must be encrypted and decrypted using the same key in both AWS regions.

Which of the following represents the best solution to address these requirements?

- **Create a CloudWatch scheduled rule to invoke a Lambda function to copy the daily data from the source bucket in `us-east-1` region to the destination bucket in `us-west-1` region. Provide AWS KMS key access to the Lambda function for encryption and decryption operations on the data in the source and destination S3 buckets**
- **Create a new S3 bucket in the `us-east-1` region with replication enabled from this new bucket into another bucket in `us-west-1` region. Enable SSE-KMS encryption on the new bucket in `us-east-1` region by using an AWS KMS multi-region key. Copy the existing data from the current S3 bucket in `us-east-1` region into this new S3 bucket in `us-east-1` region(Correcto)**
- **Enable replication for the current bucket in `us-east-1` region into another bucket in `us-west-1` region. Share the existing AWS KMS key from `us-east-1` region to `us-west-1` region**

- **Change the AWS KMS single region key used for the current S3 bucket into an AWS KMS multi-region key. Enable S3 batch replication for the existing data in the current bucket in `us-east-1` region into another bucket in `us-west-1` region**

Explicación
Correct option:

Create a new S3 bucket in the `us-east-1` region with replication enabled from this new bucket into another bucket in `us-west-1` region. Enable SSE-KMS encryption on the new bucket in `us-east-1` region by using an AWS KMS multi-region key. Copy the existing data from the current S3 bucket in `us-east-1` region into this new S3 bucket in `us-east-1` region

AWS KMS supports multi-region keys, which are AWS KMS keys in different AWS regions that can be used interchangeably – as though you had the same key in multiple regions. Each set of related multi-region keys has the same key material and key ID, so you can encrypt data in one AWS region and decrypt it in a different AWS region without re-encrypting or making a cross-region call to AWS KMS.

You can use multi-region AWS KMS keys in Amazon S3. However, Amazon S3 currently treats multi-region keys as though they were single-region keys, and does not use the multi-region features of the key.

Multi-region AWS KMS keys:

Multi-Region keys in AWS KMS

PDF | RSS

AWS KMS supports *multi-Region keys*, which are AWS KMS keys in different AWS Regions that can be used interchangeably – as though you had the same key in multiple Regions. Each set of *related* multi-Region keys has the same key material and key ID, so you can encrypt data in one AWS Region and decrypt it in a different AWS Region without re-encrypting or making a cross-Region call to AWS KMS.

Like all KMS keys, multi-Region keys never leave AWS KMS unencrypted. You can create symmetric or asymmetric multi-Region keys for encryption or signing, create HMAC multi-Region keys for generating and verifying HMAC tags, and create multi-Region keys with imported key material or key material that AWS KMS generates. You must manage each multi-Region key independently, including creating aliases and tags, setting their key policies and grants, and enabling and disabling them selectively. You can use multi-Region keys in all cryptographic operations that you can do with single-Region keys.

Multi-Region keys are a flexible and powerful solution for many common data security scenarios.

Disaster recovery
: In a backup and recovery architecture, multi-Region keys let you process encrypted data without interruption even in the event of an AWS Region outage. Data maintained in backup Regions can be decrypted in the backup Region, and data newly encrypted in the backup Region can be decrypted in the primary Region when that Region is restored.

Global data management
: Businesses that operate globally need globally distributed data that is available consistently across AWS Regions. You can create multi-Region keys in all Regions where your data resides, then use the keys as though they were a single-Region key without the latency of a cross-Region call or the cost of re-encrypting data under a different key in each Region.

Distributed signing applications
: Applications that require cross-Region signature capabilities can use multi-Region asymmetric signing keys to generate identical digital signatures consistently and repeatedly in different AWS Regions.

  If you use certificate chaining with a single global trust store (for a single root certification authority (CA), and Regional intermediate CAs signed by the root CA, you don't need multi-Region keys. However, if your system doesn't support intermediate CAs, such as application signing, you can use multi-Region keys to bring consistency to Regional certifications.

Active-active applications that span multiple Regions
: Some workloads and applications can span multiple Regions in active-active architectures. For these applications, multi-Region keys can reduce complexity by providing the same key material for concurrent encrypt and decrypt operations on data that might be moving across Region boundaries.

You can use multi-Region keys with client-side encryption libraries, such as the AWS Encryption SDK, the DynamoDB Encryption Client, and Amazon S3 client-side encryption. For an example of using multi-Region keys with Amazon DynamoDB global tables and the DynamoDB Encryption Client, see Encrypt global data client-side with AWS KMS multi-Region keys in the AWS Security Blog.

AWS services that integrate with AWS KMS for encryption at rest or digital signatures currently treat multi-Region keys as though they were single-Region keys. They might re-wrap or re-encrypt data moved between Regions. For example, Amazon S3 cross-region replication decrypts and re-encrypts data under a KMS key in the destination Region, even when replicating objects protected by a multi-Region key.

Multi-Region keys are not global. You create a multi-Region primary key and then replicate it into Regions that you select within an AWS partition. Then you manage the multi-Region key in each Region independently. Neither AWS nor AWS KMS ever automatically creates or replicates multi-Region keys into any Region on your behalf. AWS managed keys, the KMS keys that AWS services create in your account for you, are always single-Region keys.

You cannot convert an existing single-Region key to a multi-Region key. This design ensures that all data protected with existing single-Region keys maintain the same data residency and data sovereignty properties.

For most data security needs, the Regional isolation and fault tolerance of Regional resources make standard AWS KMS single-Region keys a best-fit solution. However, when you need to encrypt or sign data in client-side applications across multiple Regions, multi-Region keys might be the solution.

via - https://docs.aws.amazon.com/kms/latest/developerguide/multi-region-keys-overview.html

For the given use case, you must create a new bucket in the `us-east-1` region with replication enabled from this new bucket into another bucket in `us-west-1` region. This would ensure that the data is available in another region for backup and recovery purposes. You should also enable SSE-KMS encryption on the new bucket in `us-east-1` region by using an AWS KMS multi-region key so that the data can be encrypted and decrypted using the same key in both AWS regions. Since the existing data in the current bucket was encrypted using the AWS KMS key restricted to the `us-east-1` region, so data must be copied to the new bucket in `us-east-1` region for replication as well as multi-region KMS key based encryption to kick-in.

To require server-side encryption of all objects in a particular Amazon S3 bucket, you can use a policy. For example, the following bucket policy denies the upload object (s3:PutObject) permission to everyone if the request does not include the x-amz-server-side-encryption header requesting server-side encryption with SSE-KMS.

```
{
  "Version":"2012-10-17",
  "Id":"PutObjectPolicy",
  "Statement":[{
        "Sid":"DenyUnEncryptedObjectUploads",
        "Effect":"Deny",
        "Principal":"*",
        "Action":"s3:PutObject",
        "Resource":"arn:aws:s3:::DOC-EXAMPLE-BUCKET1/*",
        "Condition":{
           "StringNotEquals":{
              "s3:x-amz-server-side-encryption":"aws:kms"
           }
        }
     }
  ]
}
```

The following example IAM policies show statements for using AWS KMS server-side encryption with replication.

In this example, the encryption context is the object ARN. If you use SSE-KMS with an S3 Bucket Key enabled, you must use the bucket ARN as the encryption context.

```
{
   "Version": "2012-10-17",
   "Statement": [{
         "Action": ["kms:Decrypt"],
         "Effect": "Allow",
         "Resource": "List of AWS KMS key ARNs used to encrypt source objects.",
         "Condition": {
            "StringLike": {
               "kms:ViaService": "s3.source-bucket-region.amazonaws.com",
               "kms:EncryptionContext:aws:s3:arn": "arn:aws:s3:::source-bucket-name/key-prefix1/*"
            }
         }
      },

      {
         "Action": ["kms:Encrypt"],
         "Effect": "Allow",
         "Resource": "AWS KMS key ARNs (for the AWS Region of the destination bucket 1). Used to encrypt object replicas created in destination bucket 1.",
         "Condition": {
            "StringLike": {
               "kms:ViaService": "s3.destination-bucket-1-region.amazonaws.com",
               "kms:EncryptionContext:aws:s3:arn": "arn:aws:s3:::destination-bucket-name-1/key-prefix1/*"
            }
         }
      },
      {
         "Action": ["kms:Encrypt"],
         "Effect": "Allow",
```

```
        "Resource": "AWS KMS key ARNs (for the AWS Region of destination bucket 2). Used to encrypt object replicas created in destination bucket 2.",
        "Condition": {
            "StringLike": {
                "kms:ViaService": "s3.destination-bucket-2-region.amazonaws.com",
                "kms:EncryptionContext:aws:s3:arn": "arn:aws:s3:::destination-bucket-2-name/key-prefix1*"
            }
        }
    }
  ]
}
```

Incorrect options:

Change the AWS KMS single region key used for the current S3 bucket into an AWS KMS multi-region key. Enable S3 batch replication for the existing data in the current bucket in us-east-1 region into another bucket in us-west-1 region - S3 batch replication can certainly be used to replicate the existing data in the current bucket in us-east-1 region into another bucket in us-west-1 region.

However, you cannot convert an existing single-Region key to a multi-Region key. This design ensures that all data protected with existing single-Region keys maintain the same data residency and data sovereignty properties. So this option is incorrect.

Enable replication for the current bucket in us-east-1 region into another bucket in us-west-1 region. Share the existing AWS KMS key from us-east-1 region to us-west-1 region - You cannot share an AWS KMS key to another region, so this option is incorrect.

Create a CloudWatch scheduled rule to invoke a Lambda function to copy the daily data from the source bucket in us-east-1 region to the destination bucket in us-west-1 region. Provide AWS KMS key access to the Lambda function for encryption and decryption operations on the data in the source and destination S3 buckets - This option is a distractor as the daily frequency of data replication would result in significant data loss in case of a disaster. In addition, this option involves significant

development effort to create the functionality to reliably replicate the data from source to destination buckets. So this option is not the best fit for the given use case.

References:

https://docs.aws.amazon.com/kms/latest/developerguide/multi-region-keys-overview.html

https://docs.aws.amazon.com/AmazonS3/latest/userguide/replication-config-for-kms-objects.html

Pregunta 60:
A startup has just developed a video backup service hosted on a fleet of EC2 instances. The EC2 instances are behind an Application Load Balancer and the instances are using EBS volumes for storage. The service provides authenticated users the ability to upload videos that are then saved on the EBS volume attached to a given instance. On the first day of the beta launch, users start complaining that they can see only some of the videos in their uploaded videos backup. Every time the users log into the website, they claim to see a different subset of their uploaded videos.

Which of the following is the MOST optimal solution to make sure that users can view all the uploaded videos? (Select two)

- **Write a one time job to copy the videos from all EBS volumes to RDS and then modify the application to use RDS for storing the videos**
- **Write a one time job to copy the videos from all EBS volumes to DynamoDB and then modify the application to use DynamoDB for storing the videos**
- **Write a one time job to copy the videos from all EBS volumes to S3 Glacier Deep Archive and then modify the application to use S3 Glacier Deep Archive for storing the videos**
- **Mount EFS on all EC2 instances. Write a one time job to copy the videos from all EBS volumes to EFS. Modify the application to use EFS for storing the videos(Correcto)**
- **Write a one time job to copy the videos from all EBS volumes to S3 and then modify the application to use Amazon S3 standard for storing the videos(Correcto)**

Explicación
Correct options:

Write a one time job to copy the videos from all EBS volumes to S3 and then modify the application to use Amazon S3 standard for storing the videos

Mount EFS on all EC2 instances. Write a one time job to copy the videos from all EBS volumes to EFS. Modify the application to use EFS for storing the videos

Amazon Elastic Block Store (EBS) is an easy to use, high-performance block storage service designed for use with Amazon Elastic Compute Cloud (EC2) for both throughput and transaction-intensive workloads at any scale.

Amazon Elastic File System (Amazon EFS) provides a simple, scalable, fully managed elastic NFS file system for use with AWS Cloud services and on-premises resources. It is built to scale on-demand to petabytes without disrupting applications, growing and shrinking automatically as you add and remove files, eliminating the need to provision and manage capacity to accommodate growth.

Amazon Simple Storage Service (Amazon S3) is an object storage service that offers industry-leading scalability, data availability, security, and performance.

As EBS volumes are attached locally to the EC2 instances, therefore the uploaded videos are tied to specific EC2 instances. Every time the user logs in, they are directed to a different instance and therefore their videos get dispersed across multiple EBS volumes. The correct solution is to use either S3 or EFS to store the user videos.

Incorrect options:

Write a one time job to copy the videos from all EBS volumes to S3 Glacier Deep Archive and then modify the application to use S3 Glacier Deep Archive for storing the videos - Glacier Deep Archive is meant to be used for long term data archival. It cannot be used to serve static content such as videos or images via a web application. So this option is incorrect.

Write a one time job to copy the videos from all EBS volumes to RDS and then modify the application to use RDS for storing the videos - RDS is a relational database and not the right candidate for storing videos.

Write a one time job to copy the videos from all EBS volumes to DynamoDB and then modify the application to use DynamoDB for storing the videos - DynamoDB is a NoSQL database and not the right candidate for storing videos.

Reference:

https://aws.amazon.com/ebs/

Pregunta 61:
What is true about RDS Read Replicas encryption?

- **If the master database is unencrypted, the read replicas can be either encrypted or unencrypted**
- **If the master database is encrypted, the read replicas can be either encrypted or unencrypted**
- **If the master database is unencrypted, the read replicas are encrypted**
- **If the master database is encrypted, the read replicas are encrypted(Correcto)**

Explicación
Correct option:

If the master database is encrypted, the read replicas are encrypted

Amazon RDS Read Replicas provide enhanced performance and durability for RDS database (DB) instances. They make it easy to elastically scale out beyond the capacity constraints of a single DB instance for read-heavy database workloads. For the MySQL, MariaDB, PostgreSQL, Oracle, and SQL Server database engines, Amazon RDS creates a second DB instance using a snapshot of the source DB instance. It then uses the engines' native asynchronous replication to update the read replica whenever there is a change to the source DB instance. read replicas can be within an Availability Zone, Cross-AZ, or Cross-Region.

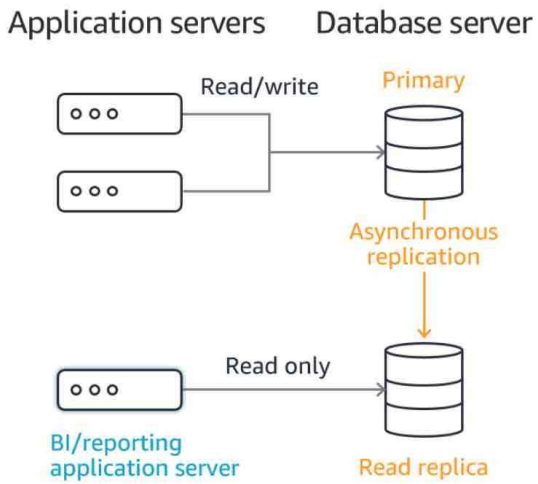
On a database instance running with Amazon RDS encryption, data stored at rest in the underlying storage is encrypted, as are its automated backups, read replicas, and snapshots. Therefore, this option is correct.

RDS Read Replica Overview:

Amazon RDS Read Replicas provide enhanced performance and durability for RDS database (DB) instances. They make it easy to elastically scale out beyond the capacity constraints of a single DB instance for read-heavy database workloads. You can create one or more replicas of a given source DB Instance and serve high-volume application read traffic from multiple copies of your data, thereby increasing aggregate read throughput. Read replicas can also be promoted when needed to become standalone DB instances. Read replicas are available in Amazon RDS for MySQL, MariaDB, PostgreSQL, Oracle, and SQL Server as well as Amazon Aurora.

For the MySQL, MariaDB, PostgreSQL, Oracle, and SQL Server database engines, Amazon RDS creates a second DB instance using a snapshot of the source DB instance. It then uses the engines' native asynchronous replication to update the read replica whenever there is a change to the source DB instance. The read replica operates as a DB instance that allows only read-only connections; applications can connect to a read replica just as they would to any DB instance. Amazon RDS replicates all databases in the source DB instance.

Amazon Aurora futher extends the benefits of read replicas by employing an SSD-backed virtualized storage layer purpose-built for database workloads. Amazon Aurora replicas share the same underlying storage as the source instance, lowering costs and avoiding the need to copy data to the replica nodes. For more information about replication with Amazon Aurora, see the online documentation.



via -https://aws.amazon.com/rds/features/read-replicas/

Incorrect options:

If the master database is encrypted, the read replicas can be either encrypted or unencrypted - If the master database is encrypted, the read replicas are necessarily encrypted, so this option is incorrect.

If the master database is unencrypted, the read replicas can be either encrypted or unencrypted

If the master database is unencrypted, the read replicas are encrypted

If the master database is not encrypted, the read replicas cannot be encrypted, so both these options are incorrect.

References:

https://aws.amazon.com/rds/features/read-replicas/

Pregunta 62:
A retail company wants to share sensitive accounting data that is stored in an Amazon RDS DB instance with an external auditor. The auditor has its own AWS account and needs its own copy of the database.

Which of the following would you recommend to securely share the database with the auditor?

- **Create a snapshot of the database in Amazon S3 and assign an IAM role to the auditor to grant access to the object in that bucket**
- **Create an encrypted snapshot of the database, share the snapshot, and allow access to the AWS Key Management Service (AWS KMS) encryption key(Correcto)**
- **Export the database contents to text files, store the files in Amazon S3, and create a new IAM user for the auditor with access to that bucket**
- **Set up a read replica of the database and configure IAM standard database authentication to grant the auditor access**

Explicación
Correct option:

Create an encrypted snapshot of the database, share the snapshot, and allow access to the AWS Key Management Service (AWS KMS) encryption key

You can share the AWS Key Management Service (AWS KMS) customer master key (CMK) that was used to encrypt the snapshot with any accounts that you want to be able to access the snapshot. You can share AWS KMS CMKs with another AWS account by adding the other account to the AWS KMS key policy.

Making an encrypted snapshot of the database will give the auditor a copy of the database, as required for the given use case.

Incorrect options:

Create a snapshot of the database in Amazon S3 and assign an IAM role to the auditor to grant access to the object in that bucket - RDS stores the DB snapshots in the Amazon S3 bucket belonging to the same AWS region where the RDS instance is located. RDS stores these on your behalf and you do not have direct access to these snapshots in S3, so it's not possible to grant access to the snapshot objects in S3.

Export the database contents to text files, store the files in Amazon S3, and create a new IAM user for the auditor with access to that bucket - This solution is feasible though not optimal. It requires a lot of unnecessary work and is difficult to audit when such bulk data is exported into text files.

Set up a read replica of the database and configure IAM standard database authentication to grant the auditor access - Read Replicas make it easy to elastically scale out beyond the capacity constraints of a single DB instance for read-heavy database workloads. Creating Read Replicas for audit purposes is overkill. Also, the question mentions that the auditor needs to have their own copy of the database, which is not possible with replicas.

Reference:

https://docs.aws.amazon.com/AmazonRDS/latest/UserGuide/USER_ShareSnapshot.html

Pregunta 63:

Consider the following policy associated with an IAM group containing several users:

```
{
   "Version":"2012-10-17",
   "Id":"EC2TerminationPolicy",
   "Statement":[
      {
         "Effect":"Deny",
         "Action":"ec2:*",
         "Resource":"*",
         "Condition":{
            "StringNotEquals":{
               "ec2:Region":"us-west-1"
            }
         }
      },
      {
         "Effect":"Allow",
         "Action":"ec2:TerminateInstances",
         "Resource":"*",
         "Condition":{
            "IpAddress":{
               "aws:SourceIp":"10.200.200.0/24"
            }
         }
      }
   ]
}
```

Which of the following options is correct?

- **Users belonging to the IAM group cannot terminate an EC2 instance in the `us-west-1` region when the user's source IP is 10.200.200.200**
- **Users belonging to the IAM group can terminate an EC2 instance in the `us-west-1` region when the user's source IP is 10.200.200.200(Correcto)**
- **Users belonging to the IAM group can terminate an EC2 instance in the `us-west-1` region when the EC2 instance's IP address is 10.200.200.200**
- **Users belonging to the IAM group can terminate an EC2 instance belonging to any region except the `us-west-1` region when the user's source IP is 10.200.200.200**

Explicación
Correct option:

Users belonging to the IAM group can terminate an EC2 instance in the `us-west-1` region when the user's source IP is 10.200.200.200

The given policy denies all EC2 specification actions on all resources when the region of the underlying resource is not `us-west-1`. The policy allows the terminate EC2 action on all resources when the source IP address is in the CIDR range 10.200.200.0/24, therefore it would allow the user with the source IP 10.200.200.200 to terminate the EC2 instance.

Incorrect options:

Users belonging to the IAM group cannot terminate an EC2 instance in the `us-west-1` region when the user's source IP is 10.200.200.200

Users belonging to the IAM group can terminate an EC2 instance in the `us-west-1` region when the EC2 instance's IP address is 10.200.200.200

Users belonging to the IAM group can terminate an EC2 instance belonging to any region except the `us-west-1` region when the user's source IP is 10.200.200.200

These three options contradict the explanation provided above, so these options are incorrect.

Reference:

https://docs.aws.amazon.com/IAM/latest/UserGuide/reference_policies_evaluation-logic.html

Pregunta 64:
The engineering team at an e-commerce company is working on cost optimizations for EC2 instances. The team wants to manage the workload using a mix of on-demand and spot instances across multiple instance types. They would like to create an Auto Scaling group with a mix of these instances.

Which of the following options would allow the engineering team to provision the instances for this use-case?

- **You can neither use a launch configuration nor a launch template to provision capacity across multiple instance types using both On-Demand Instances and Spot Instances to achieve the desired scale, performance, and cost**
- **You can only use a launch template to provision capacity across multiple instance types using both On-Demand Instances and Spot Instances to achieve the desired scale, performance, and cost(Correcto)**
- **You can use a launch configuration or a launch template to provision capacity across multiple instance types using both On-Demand Instances and Spot Instances to achieve the desired scale, performance, and cost**
- **You can only use a launch configuration to provision capacity across multiple instance types using both On-Demand Instances and Spot Instances to achieve the desired scale, performance, and cost**

Explicación
Correct option:

You can only use a launch template to provision capacity across multiple instance types using both On-Demand Instances and Spot Instances to achieve the desired scale, performance, and cost

A launch template is similar to a launch configuration, in that it specifies instance configuration information such as the ID of the Amazon Machine Image (AMI), the instance type, a key pair, security groups, and the other parameters that you use to launch EC2 instances. Also, defining a launch template instead of a launch configuration allows you to have multiple versions of a template.

With launch templates, you can provision capacity across multiple instance types using both On-Demand Instances and Spot Instances to achieve the desired scale, performance, and cost. Hence this is the correct option.

Incorrect options:

You can only use a launch configuration to provision capacity across multiple instance types using both On-Demand Instances and Spot Instances to achieve the desired scale, performance, and cost

You can use a launch configuration or a launch template to provision capacity across multiple instance types using both On-Demand Instances and Spot Instances to achieve the desired scale, performance, and cost

A launch configuration is an instance configuration template that an Auto Scaling group uses to launch EC2 instances. When you create a launch configuration, you specify information for the instances such as the ID of the Amazon Machine Image (AMI), the instance type, a key pair, one or more security groups, and a block device mapping.

You cannot use a launch configuration to provision capacity across multiple instance types using both On-Demand Instances and Spot Instances. Therefore both these options are incorrect.

You can neither use a launch configuration nor a launch template to provision capacity across multiple instance types using both On-Demand Instances and Spot Instances to achieve the desired scale, performance, and cost - You can use a launch template to provision capacity across multiple instance types using both On-Demand Instances and Spot Instances. So this option is incorrect.

References:

https://docs.aws.amazon.com/autoscaling/ec2/userguide/LaunchTemplates.html

https://docs.aws.amazon.com/autoscaling/ec2/userguide/LaunchConfiguration.html

Pregunta 65:
A company is developing a healthcare application that cannot afford any downtime for database write operations. The company has hired you as an AWS Certified Solutions Architect Associate to build a solution using Amazon Aurora.

Which of the following options would you recommend?

- **Set up an Aurora multi-master DB cluster(Correcto)**
- **Set up an Aurora provisioned DB cluster**
- **Set up an Aurora Global Database cluster**
- **Set up an Aurora serverless DB cluster**

Explicación
Correct option:

Set up an Aurora multi-master DB cluster

In a multi-master cluster, all DB instances can perform write operations. There isn't any failover when a writer DB instance becomes unavailable, because another writer DB instance is immediately available to take over the work of the failed instance. AWS refers to this type of availability as continuous availability, to distinguish it from the high availability (with brief downtime during failover) offered by a single-master cluster. For applications where you can't afford even brief downtime for database write operations, a multi-master cluster can help to avoid an outage when a writer instance becomes unavailable. The multi-master cluster doesn't use the failover mechanism, because it doesn't need to promote another DB instance to have read/write capability.

Overview of Aurora multi-master clusters

Use the following background information to help you choose a multi-master or single-master cluster when you set up a new Aurora cluster. For you to make an informed choice, we recommend that you first understand how you plan to adapt your schema design and application logic to work best with a multi-master cluster.

For each new Amazon Aurora cluster, you can choose whether to create a single-master or multi-master cluster.

Most kinds of Aurora clusters are *single-master* clusters. For example, provisioned, Aurora Serverless, parallel query, and Global Database clusters are all single-master clusters. In a single-master cluster, a single DB instance performs all write operations and any other DB instances are read-only. If the writer DB instance becomes unavailable, a failover mechanism promotes one of the read-only instances to be the new writer.

In a *multi-master cluster*, all DB instances can perform write operations. The notions of a single read/write primary instance and multiple read-only Aurora Replicas don't apply. There isn't any failover when a writer DB instance becomes unavailable, because another writer DB instance is immediately available to take over the work of the failed instance. We refer to this type of availability as *continuous availability*, to distinguish it from the high availability (with brief downtime during failover) offered by a single-master cluster.

Multi-master clusters work differently in many ways from the other kinds of Aurora clusters, such as provisioned, Aurora Serverless, and parallel query clusters. With multi-master clusters, you consider different factors in areas such as high availability, monitoring, connection management, and database features. For example, in applications where you can't afford even brief downtime for database write operations, a multi-master cluster can help to avoid an outage when a writer instance becomes unavailable. The multi-master cluster doesn't use the failover mechanism, because it doesn't need to promote another DB instance to have read/write capability. With a multi-master cluster, you examine metrics related to DML throughput, latency, and deadlocks for all DB instances instead of a single primary instance.

Currently, multi-master clusters require Aurora MySQL version 1, which is compatible with MySQL 5.6. When specifying the DB engine version in the AWS Management Console, AWS CLI, or RDS API, choose `5.6.10a`.

To create a multi-master cluster, you choose **Multiple writers** under **Database features** when creating the cluster. Doing so enables different behavior for replication among DB instances, availability, and performance than other kinds of Aurora clusters. This choice remains in effect for the life of the cluster. Make sure that you understand the specialized use cases that are appropriate for multi-master clusters.

via - https://docs.aws.amazon.com/AmazonRDS/latest/AuroraUserGuide/aurora-multi-master.html#aurora-multi-master-workloads

Incorrect options:

Set up an Aurora serverless DB cluster

Set up an Aurora provisioned DB cluster

Set up an Aurora Global Database cluster

These three options represent Aurora single-master clusters. In a single-master cluster, a single DB instance performs all write operations and any other DB instances are read-only. If the writer DB instance becomes unavailable, a failover mechanism promotes one of the read-only instances to be the new writer. As there is a brief downtime during this failover, so these three options are incorrect for the given use case.

Reference:

https://docs.aws.amazon.com/AmazonRDS/latest/AuroraUserGuide/aurora-multi-master.html#aurora-multi-master-workloads