**AWS Certified Developer Associate – PT 3 – Results**

Attempt 1: Failed (72% required to pass)

49% correct(32/65)

2 hours 10 minutes

May 27, 2024 at 10:45 AM

**Domains**

**AWS Storage (2 questions)**

50%

50%

**AWS Database (12 questions)**

58%

42%

**AWS Compute (14 questions)**

21%

79%

**AWS Security, Identity, & Compliance (9 questions)**

44%

56%

**AWS Networking & Content Delivery (10 questions)**

80%

20%

**AWS Management & Governance (5 questions)**

60%

40%

**AWS Developer Tools (9 questions)**

33%

67%

**AWS Application Integration (4 questions)**

75%

25%

Correct Wrong Skipped/Unanswered

**Question 1Incorrect**

An application is running on an Amazon EC2 Linux instance. The instance needs to make AWS API calls to several AWS services. What is the MOST secure way to provide access to the AWS services with MINIMAL management overhead?

**Store the credentials in the ~/.aws/credentials file**

**Your answer is incorrect**

**Use AWS KMS to store and retrieve credentials**

**Store the credentials in AWS CloudHSM**

**Correct answer**

**Use EC2 instance profiles**

Overall explanation

An instance profile is a container for an IAM role that you can use to pass role information to an EC2 instance when the instance starts. Using an instance profile you can attach an IAM Role to an EC2 instance that the instance can then assume in order to gain access to AWS services.

**CORRECT:**"Use EC2 instance profiles" is the correct answer.

**INCORRECT:** "Use AWS KMS to store and retrieve credentials" is incorrect as KMS is used to manage encryption keys.

**INCORRECT:** "Store the credentials in AWS CloudHSM" is incorrect as CloudHSM is also used to manage encryption keys. It is similar to KMS but uses a dedicated hardware device that is not multi-tenant.

**INCORRECT:** "Store the credentials in the ~/.aws/credentials file" is incorrect as this is not the most secure option. The credentials file is associated with the AWS CLI and used for passing credentials in the form of an access key ID and secret access key when making programmatic requests from the command line.

**References:**

<https://docs.aws.amazon.com/IAM/latest/UserGuide/id_roles_use_switch-role-ec2_instance-profiles.html>

**Save time with our AWS cheat sheets:**

<https://digitalcloud.training/amazon-ec2/>

**Domain**

AWS Security, Identity, & Compliance

**Question 2Incorrect**

An application is instrumented to generate traces using AWS X-Ray and generates a large amount of trace data. A Developer would like to use filter expressions to filter the results to specific key-value pairs added to custom subsegments.

How should the Developer add the key-value pairs to the custom subsegments?

**Your answer is incorrect**

**Add metadata to the custom subsegments**

**Add the key-value pairs to the Trace ID**

**Setup sampling for the custom subsegments**

**Correct answer**

**Add annotations to the custom subsegments**

Overall explanation

You can record additional information about requests, the environment, or your application with annotations and metadata. You can add annotations and metadata to the segments that the X-Ray SDK creates, or to custom subsegments that you create.

**Annotations** are key-value pairs with string, number, or Boolean values. Annotations are indexed for use with [filter expressions](https://docs.aws.amazon.com/xray/latest/devguide/xray-console-filters.html). Use annotations to record data that you want to use to group traces in the console, or when calling the [GetTraceSummaries](https://docs.aws.amazon.com/xray/latest/api/API_GetTraceSummaries.html) API.

**Metadata** are key-value pairs that can have values of any type, including objects and lists, but are not indexed for use with filter expressions. Use metadata to record additional data that you want stored in the trace but don't need to use with search.

Annotations can be used with filter expressions, so this is the best solution for this requirement. The Developer can add annotations to the custom subsegments and will then be able to use filter expressions to filter the results in AWS X-Ray.

**CORRECT:**"Add annotations to the custom subsegments" is the correct answer.

**INCORRECT:** "Add metadata to the custom subsegments" is incorrect as though you can add metadata to custom subsegments it is not indexed and cannot be used with filters.

**INCORRECT:** "Add the key-value pairs to the Trace ID" is incorrect as this is not something you can do.

**INCORRECT:** "Setup sampling for the custom subsegments " is incorrect as this is a mechanism used by X-Ray to send only statistically significant data samples to the API.

**References:**

<https://docs.aws.amazon.com/xray/latest/devguide/xray-sdk-java-segment.html>

**Save time with our AWS cheat sheets:**

<https://digitalcloud.training/aws-developer-tools/>

**Domain**

AWS Developer Tools

**Question 3Incorrect**

A developer is planning the deployment of a new version of an application to AWS Elastic Beanstalk. The new version of the application should be deployed only to new EC2 instances.

Which deployment methods will meet these requirements? (Select TWO.)

**Your selection is incorrect**

**Rolling with additional batch**

**Correct selection**

**Blue/green**

**All at once**

**Your selection is correct**

**Immutable**

**Rolling**

Overall explanation

AWS Elastic Beanstalk provides several options for how deployments are processed, including deployment policies and options that let you configure batch size and health check behavior during deployments.

All at once:

• Deploys the new version to all instances simultaneously.

Rolling:

• Update a few instances at a time (bucket), and then move onto the next bucket once the first bucket is healthy (downtime for 1 bucket at a time).

Rolling with additional batch:

• Like Rolling but launches new instances in a batch ensuring that there is full availability.

Immutable:

• Launches new instances in a new ASG and deploys the version update to these instances before swapping traffic to these instances once healthy.

• Zero downtime.

Blue / Green deployment:

• Zero downtime and release facility.

• Create a new “stage” environment and deploy updates there.

The immutable and blue/green options both provide zero downtime as they will deploy the new version to a new version of the application. These are also the only two options that will ONLY deploy the updates to new EC2 instances.

**CORRECT:**"Immutable" is the correct answer.

**CORRECT:**"Blue/green" is the correct answer.

**INCORRECT:** "All-at-once" is incorrect as this will deploy the updates to existing instances.

**INCORRECT:** "Rolling" is incorrect as this will deploy the updates to existing instances.

**INCORRECT:** "Rolling with additional batch" is incorrect as this will launch new instances but will also update the existing instances as well (which is not allowed according to the requirements).

**References:**

<https://docs.aws.amazon.com/elasticbeanstalk/latest/dg/using-features.rolling-version-deploy.html>

<https://docs.aws.amazon.com/elasticbeanstalk/latest/dg/using-features.deploy-existing-version.html>

**Save time with our AWS cheat sheets:**

<https://digitalcloud.training/aws-elastic-beanstalk/>

**Domain**

AWS Compute

**Question 4Correct**

In the process of developing an application, a software engineer deploys an Amazon API Gateway REST API within the us-west-2 Region. The plan is to use Amazon CloudFront and a custom domain name for the API, using an SSL/TLS certificate acquired from a third-party provider.

What is the appropriate strategy for configuring the custom domain name?

**Your answer is correct**

**Import the third-party SSL/TLS certificate to AWS Certificate Manager (ACM), link it with the custom domain name in API Gateway, and then create an alias (A) record in Route 53 for the custom domain name.**

**Use Route 53 to create a simple routing policy with the custom domain name directly pointed to the API Gateway.**

**Directly install the third-party SSL/TLS certificate on the API Gateway and establish a CNAME record in Route 53 for the custom domain name.**

**Use the third-party SSL/TLS certificate directly with Amazon CloudFront, and bypass API Gateway, creating an alias (A) record in Route 53.**

Overall explanation

The developer needs to import the SSL/TLS certificate into AWS Certificate Manager (ACM) to use a custom domain name with Amazon API Gateway. After that, the developer can link this certificate to the custom domain name in the API Gateway. Finally, set an alias (A) record in Route 53 to point to the API Gateway's domain name.

**CORRECT:**"Import the third-party SSL/TLS certificate to AWS Certificate Manager (ACM), link it with the custom domain name in API Gateway, and then create an alias (A) record in Route 53 for the custom domain name" is the correct answer (as explained above.)

**INCORRECT:** "Use Route 53 to create a simple routing policy with the custom domain name directly pointed to the API Gateway" is incorrect.

You cannot point a custom domain name directly to the API Gateway using a simple routing policy in Route 53 without setting up the custom domain name in API Gateway first.

**INCORRECT:** "Directly install the third-party SSL/TLS certificate on the API Gateway and establish a CNAME record in Route 53 for the custom domain name" is incorrect.

API Gateway doesn't support direct installation of third-party SSL/TLS certificates. Also, for API Gateway, an alias record is preferred over a CNAME record in Route 53.

**INCORRECT:** "Use the third-party SSL/TLS certificate directly with Amazon CloudFront, and bypass API Gateway, creating an alias (A) record in Route 53" is incorrect.

Amazon CloudFront is inherently used and managed by API Gateway, and you cannot bypass API Gateway using Amazon CloudFront directly for a custom domain. Also, an SSL/TLS certificate needs to be managed via ACM.

**References:**

<https://docs.aws.amazon.com/apigateway/latest/developerguide/how-to-custom-domains.html>

<https://docs.aws.amazon.com/Route53/latest/DeveloperGuide/routing-to-api-gateway.html>

**Save time with our AWS cheat sheets:**

<https://digitalcloud.training/aws-certificate-manager/>

**Domain**

AWS Networking & Content Delivery

**Question 5Correct**

A Developer is using AWS SAM to create a template for deploying a serverless application. The Developer plans deploy an AWS Lambda function and an Amazon DynamoDB table using the template.

Which resource types should the Developer specify? (Select TWO.)

**Your selection is correct**

**AWS::Serverless::SimpleTable**

**AWS::Serverless::Application**

**AWS::Serverless:API**

**AWS::Serverless:LayerVersion**

**Your selection is correct**

**AWS::Serverless:Function**

Overall explanation

A **serverless application** is a combination of Lambda functions, event sources, and other resources that work together to perform tasks. Note that a serverless application is more than just a Lambda function—it can include additional resources such as APIs, databases, and event source mappings.

AWS SAM templates are an extension of AWS CloudFormation templates, with some additional components that make them easier to work with.

To create a Lambda function using an AWS SAM template the Developer can use the AWS::Serverless::Function resource type. The AWS::Serverless::Function resource type can be used to Create a Lambda function, IAM execution role, and event source mappings that trigger the function.

To create a DynamoDB table using an AWS SAM template the Developer can use the AWS::Serverless::SimpleTable resource type which creates a DynamoDB table with a single attribute primary key. It is useful when data only needs to be accessed via a primary key.

**CORRECT:**"AWS::Serverless:Function" is a correct answer.

**CORRECT:**"AWS::Serverless:SimpleTable" is also a correct answer.

**INCORRECT:** "AWS::Serverless::Application" is incorrect as this embeds a serverless application from the [AWS Serverless Application Repository](https://serverlessrepo.aws.amazon.com/applications) or from an Amazon S3 bucket as a nested application.

**INCORRECT:** "AWS::Serverless:LayerVersion" is incorrect as this creates a Lambda LayerVersion that contains library or runtime code needed by a Lambda Function.

**INCORRECT:** "AWS::Serverless:API" is incorrect as this creates a collection of Amazon API Gateway resources and methods that can be invoked through HTTPS endpoints.

**References:**

<https://docs.aws.amazon.com/serverless-application-model/latest/developerguide/what-is-sam.html>

**Save time with our AWS cheat sheets:**

<https://digitalcloud.training/aws-sam/>

**Domain**

AWS Compute

**Question 6Correct**

A company needs to ingest several terabytes of data every hour from a large number of distributed sources. The messages are delivered continually 24 hrs a day. Messages must be delivered in real time for security analysis and live operational dashboards.

Which approach will meet these requirements?

**Use the Amazon S3 API to write messages to an S3 bucket, then process the messages by using Amazon RedShift**

**Send the messages to an Amazon SQS queue, then process the messages by using a fleet of Amazon EC2 instances**

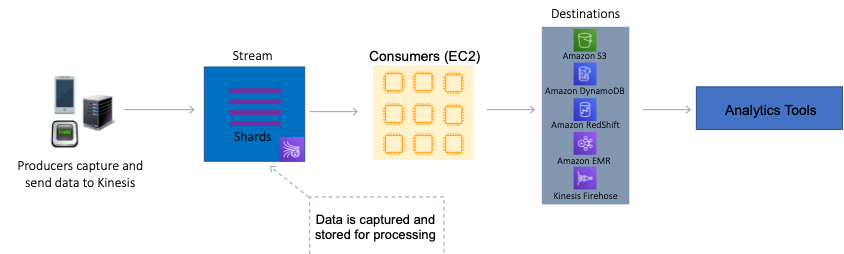
**Your answer is correct**

**Use Amazon Kinesis Data Streams with Kinesis Client Library to ingest and deliver messages**

**Use AWS Data Pipeline to automate the movement and transformation of data**

Overall explanation

You can use Amazon Kinesis Data Streams to collect and process large [streams](https://aws.amazon.com/streaming-data/) of data records in real time. You can create data-processing applications, known as *Kinesis Data Streams applications*. A typical Kinesis Data Streams application reads data from a *data stream* as data records.



These applications can use the Kinesis Client Library, and they can run on Amazon EC2 instances. You can send the processed records to dashboards, use them to generate alerts, dynamically change pricing and advertising strategies, or send data to a variety of other AWS services.

This scenario is an ideal use case for Kinesis Data Streams as large volumes of real time streaming data are being ingested. Therefore, the best approach is to use Amazon Kinesis Data Streams with Kinesis Client Library to ingest and deliver messages

**CORRECT:**"Use Amazon Kinesis Data Streams with Kinesis Client Library to ingest and deliver messages" is the correct answer.

**INCORRECT:** "Send the messages to an Amazon SQS queue, then process the messages by using a fleet of Amazon EC2 instances" is incorrect as this is not an ideal use case for SQS because SQS is used for decoupling application components, not for ingesting streaming data. It would require more cost (lots of instances to process data) and introduce latency. Also, the message size limitations could be an issue.

**INCORRECT:** "Use the Amazon S3 API to write messages to an S3 bucket, then process the messages by using Amazon RedShift" is incorrect as RedShift does not process messages from S3. RedShift is a data warehouse which is used for analytics.

**INCORRECT:** "Use AWS Data Pipeline to automate the movement and transformation of data" is incorrect as the question is not asking for transformation of data. The scenario calls for a solution for ingesting and processing the real time streaming data for analytics and feeding some data into a system that generates an operational dashboard.

**References:**

<https://docs.aws.amazon.com/streams/latest/dev/introduction.html>

**Save time with our AWS cheat sheets:**

<https://digitalcloud.training/amazon-kinesis/>

**Domain**

AWS Application Integration

**Question 7Correct**

A programmer is creating an application that requires signed requests (Signature Version 4) for invoking other AWS services. Having constructed a canonical request, created the string to sign, and calculated the signing information, which strategies can the programmer apply to finalize a signed request? (Select TWO.)

**Add the signature to a query string parameter named "Signature-Token".**

**Your selection is correct**

**Incorporate the signature into an HTTP header called "Authorization".**

**Your selection is correct**

**Insert the signature into a query string parameter referred to as "X-Amz-Signature".**

**Append the signature to a query string parameter known as "X-Amz-Credentials".**

**Embed the signature in an HTTP header labelled "Authorization-Key".**

Overall explanation

When sending a request to AWS services using Signature Version 4, the signature should be included in the "Authorization" HTTP header or as a parameter "X-Amz-Signature" in the query string.

**CORRECT:**"Incorporate the signature into an HTTP header called "Authorization"" is a correct answer (as explained above.)

**CORRECT:**"Insert the signature into a query string parameter referred to as "X-Amz-Signature"" is also a correct answer (as explained above.)

**INCORRECT:** "Append the signature to a query string parameter known as "X-Amz-Credentials"" is incorrect.

"X-Amz-Credentials" is a query string parameter, but it's not meant for adding signatures. It's used to include access key ID and scoped credential details.

**INCORRECT:** "Embed the signature in an HTTP header labelled "Authorization-Key"" is incorrect.

"Authorization-Key" is not a recognized HTTP header for adding AWS Signature Version 4.

**INCORRECT:** "Add the signature to a query string parameter named "Signature-Token"" is incorrect.

"Signature-Token" is not a recognized query string parameter for adding AWS Signature Version 4.

**References:**

<https://docs.aws.amazon.com/IAM/latest/UserGuide/reference_aws-signing.html>

**Domain**

AWS Security, Identity, & Compliance

**Question 8Correct**

A Developer has added a Global Secondary Index (GSI) to an existing Amazon DynamoDB table. The GSI is used mainly for read operations whereas the primary table is extremely write-intensive. Recently, the Developer has noticed throttling occurring under heavy write activity on the primary table. However, the write capacity units on the primary table are not fully utilized.

What is the best explanation for why the writes are being throttled on the primary table?

**There are insufficient write capacity units on the primary table**

**The Developer should have added an LSI instead of a GSI**

**There are insufficient read capacity units on the primary table**

**Your answer is correct**

**The write capacity units on the GSI are under provisioned**

Overall explanation

Some applications might need to perform many kinds of queries, using a variety of different attributes as query criteria. To support these requirements, you can create one or more global secondary indexes and issue Query requests against these indexes in Amazon DynamoDB.

When items from a primary table are written to the GSI they consume write capacity units. It is essential to ensure the GSI has sufficient WCUs (typically, at least as many as the primary table). If writes are throttled on the GSI, the main table will be throttled (even if there’s enough WCUs on the main table). LSIs do not cause any special throttling considerations.

In this scenario, it is likely that the Developer assumed that the GSI would need fewer WCUs as it is more read-intensive and neglected to factor in the WCUs required for writing data into the GSI. Therefore, the most likely explanation is that the write capacity units on the GSI are under provisioned

**CORRECT:**"The write capacity units on the GSI are under provisioned" is the correct answer.

**INCORRECT:** "There are insufficient read capacity units on the primary table" is incorrect as the table is being throttled due to writes, not reads.

**INCORRECT:** "The Developer should have added an LSI instead of a GSI" is incorrect as a GSI has specific advantages and there was likely good reason for adding a GSI. Also, you cannot add an LSI to an existing table.

**INCORRECT:** "There are insufficient write capacity units on the primary table" is incorrect as the question states that the WCUs are underutilized.

**References:**

<https://docs.aws.amazon.com/amazondynamodb/latest/developerguide/GSI.html>

**Domain**

AWS Networking & Content Delivery

**Question 9Correct**

An engineer wishes to share a software project they've developed with their team members for review. The shared application code needs to be preserved over time, with multiple versions and batch modifications being tracked. What is the most appropriate AWS service for this purpose?

**AWS CodeStar**

**AWS CodeDeploy**

**AWS Cloud9**

**Your answer is correct**

**AWS CodeCommit**

Overall explanation

AWS CodeCommit is a fully managed source control service that hosts secure Git-based repositories. It's designed to collaborate on code, maintain version control, and keep track of changes. This makes it an optimal choice for this scenario.

**CORRECT:**"AWS CodeCommit" is the correct answer (as explained above.)

**INCORRECT:** "AWS CodeStar" is incorrect.

AWS CodeStar enables you to quickly develop, build, and deploy applications on AWS. It isn't focused on source control and versioning, which is the primary need in this scenario.

**INCORRECT:** "AWS Cloud9" is incorrect.

AWS Cloud9 is a cloud-based integrated development environment (IDE) that lets you write, run, and debug your code with just a browser. While it allows for code sharing, it does not offer the robust version control and change tracking of CodeCommit.

**INCORRECT:** "AWS CodeDeploy" is incorrect.

AWS CodeDeploy is a deployment service that enables developers to automate software deployments to a variety of compute services such as Amazon EC2, AWS Fargate, AWS Lambda, and your on-premises servers.

**References:**

<https://aws.amazon.com/codecommit/>

**Save time with our AWS cheat sheets:**

<https://digitalcloud.training/aws-developer-tools/>

**Domain**

AWS Developer Tools

**Question 10Incorrect**

A company runs a decoupled application that uses an Amazon SQS queue. The messages are processed by an AWS Lambda function. The function is not keeping up with the number of messages in the queue. A developer noticed that though the application can process multiple messages per invocation, it is only processing one at a time.

How can the developer configure the application to process messages more efficiently?

**Call the ChangeMessageVisibility API for the queue and set MessageRetentionPeriod to a value greater than the default of 1.**

**Correct answer**

**Call the ReceiveMessage API to set MaxNumberOfMessages to a value greater than the default of 1.**

**Call the ReceiveMessage API to set MaximumMessageSize to a value greater than the default of 1.**

**Your answer is incorrect**

**Call the SetQueueAttributes API for the queue and set MaxNumberOfMessages to a value greater than the default of 1.**

Overall explanation

The ReceiveMessage API retrieves one or more messages (up to 10), from the specified queue. The MaxNumberOfMessages specifies the maximum number of messages to return. Amazon SQS never returns more messages than this value (however, fewer messages might be returned). Valid values: 1 to 10. Default: 1.

Changing the MaxNumberOfMessages using the ReceiveMessage API to a value greater than 1 will therefore enable the application to process more messages in a single invocation, leading to greater efficiency.

**CORRECT:**"Call the ReceiveMessage API to set MaxNumberOfMessages to a value greater than the default of 1" is the correct answer (as explained above.)

**INCORRECT:** "Call the ReceiveMessage API to set MaximumMessageSize to a value greater than the default of 1" is incorrect.

MaximumMessageSize specifies the maximum bytes a message can contain before SQS rejects it.

**INCORRECT:** "Call the ChangeMessageVisibility API for the queue and set MessageRetentionPeriod to a value greater than the default of 1" is incorrect.

ChangeMessageVisibility changes the visibility timeout of a specified message in a queue to a new value.

**INCORRECT:** "Call the SetQueueAttributes API for the queue and set MaxNumberOfMessages to a value greater than the default of 1" is incorrect.

MaxNumberOfMessages is configured using the ReceiveMessage API, not the SetQueueAttributes API.

**References:**

<https://docs.aws.amazon.com/AWSSimpleQueueService/latest/APIReference/API_ReceiveMessage.html>

**Save time with our AWS cheat sheets:**

<https://digitalcloud.training/aws-application-integration-services/>

**Domain**

AWS Application Integration

**Question 11Incorrect**

An engineer is constructing an AWS Lambda function and intends to log specific key events that transpire during the function's execution. To correlate the events with a particular function invocation, the engineer is looking to incorporate a unique identifier.

The following code segment has been added to the Lambda function:

function handler (event, context) {

}

**Correct answer**

**Use context.awsRequestId within the function to fetch the unique identifier associated with each invocation.**

**Use event.requestId to obtain the unique identifier for each function execution.**

**Your answer is incorrect**

**Use context.invocationId to extract the unique identifier tied to each function run.**

**Use context.lambdaId to get the unique identifier corresponding to each function invocation.**

Overall explanation

The **context** object in a Lambda function provides metadata about the function and the current invocation, including a unique identifier for the request, **awsRequestId**, which can be used to correlate logs from a specific invocation.

**CORRECT:**"Use **context.awsRequestId** within the function to fetch the unique identifier associated with each invocation" is the correct answer (as explained above.)

**INCORRECT:** "Use **event.requestId** to obtain the unique identifier for each function execution" is incorrect.

**event.requestId** is incorrect as the event object does not contain a property called **requestId**.

**INCORRECT:** "Use **context.invocationId** to extract the unique identifier tied to each function run" is incorrect.

**context.invocationId** is not valid as there is no such property in the context object of a Lambda function.

**INCORRECT:** "Use **context.lambdaId** to get the unique identifier corresponding to each function invocation" is incorrect.

**context.lambdaId** is not a valid property within the context object for a Lambda function.

**References:**

<https://docs.aws.amazon.com/lambda/latest/dg/nodejs-context.html>

**Save time with our AWS cheat sheets:**

<https://digitalcloud.training/aws-lambda/>

**Domain**

AWS Compute

**Question 12Correct**

An application is using Amazon DynamoDB as its data store and needs to be able to read 100 items per second as strongly consistent reads. Each item is 5 KB in size.  
What value should be set for the table's provisioned throughput for reads?

**500 Read Capacity Units**

**Your answer is correct**

**200 Read Capacity Units**

**50 Read Capacity Units**

**250 Read Capacity Units**

Overall explanation

With provisioned capacity mode, you specify the number of data reads and writes per second that you require for your application.

**Read capacity unit (RCU):**

• Each API call to read data from your table is a read request.

• Read requests can be strongly consistent, eventually consistent, or transactional.

• For items up to 4 KB in size, one RCU can perform one *strongly consistent* read request per second.

• Items larger than 4 KB require additional RCUs.

• For items up to 4 KB in size, one RCU can perform two *eventually consistent* read requests per second.

• *Transactional* read requests require two RCUs to perform one read per second for items up to 4 KB.

• For example, a strongly consistent read of an 8 KB item would require two RCUs, an eventually consistent read of an 8 KB item would require one RCU, and a transactional read of an 8 KB item would require four RCUs.

**Write capacity unit (WCU):**

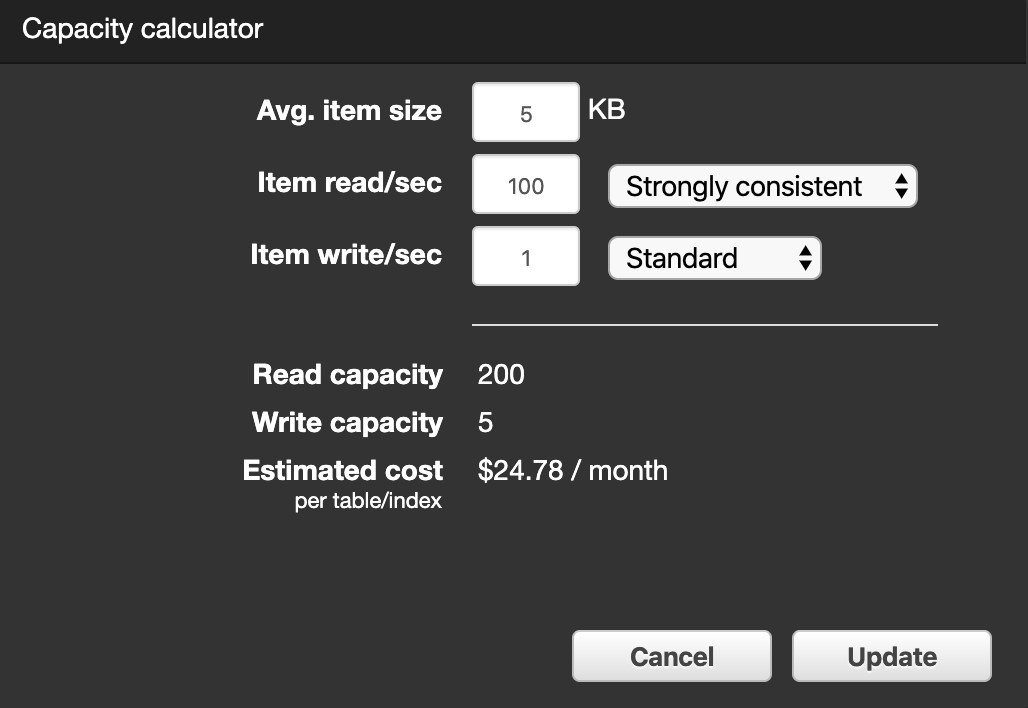
• Each API call to write data to your table is a write request.

• For items up to 1 KB in size, one WCU can perform one*standard* write request per second.

• Items larger than 1 KB require additional WCUs.

• *Transactional* write requests require two WCUs to perform one write per second for items up to 1 KB.

• For example, a standard write request of a 1 KB item would require one WCU, a standard write request of a 3 KB item would require three WCUs, and a transactional write request of a 3 KB item would require six WCUs.



**To determine the number of RCUs required to handle 100 strongly consistent reads per/second with an average item size of 5KB, perform the following steps:**

**1. Determine the average item size by rounding up the next multiple of 4KB (5KB rounds up to 8KB).**

**2. Determine the RCU per item by dividing the item size by 4KB (8KB/4KB = 2).**

3. **Multiply the value from step 2 with the number of reads required per second (2x100 = 200).**

**CORRECT:**"200 Read Capacity Units" is the correct answer.

**INCORRECT:** "50 Read Capacity Units" is incorrect.

**INCORRECT:** "250 Read Capacity Units" is incorrect.

**INCORRECT:** "500 Read Capacity Units" is incorrect.

**References:**

<https://aws.amazon.com/dynamodb/pricing/provisioned/>

**Save time with our AWS cheat sheets:**

<https://digitalcloud.training/amazon-dynamodb/>

**Domain**

AWS Database

**Question 13Incorrect**

A company is planning to use AWS CodeDeploy to deploy a new AWS Lambda function

What are the MINIMUM properties required in the 'resources' section of the AppSpec file for CodeDeploy to deploy the function successfully?

**Your answer is incorrect**

**TaskDefinition, LoadBalancerInfo, and ContainerPort**

**Correct answer**

**name, alias, currentversion, and targetversion**

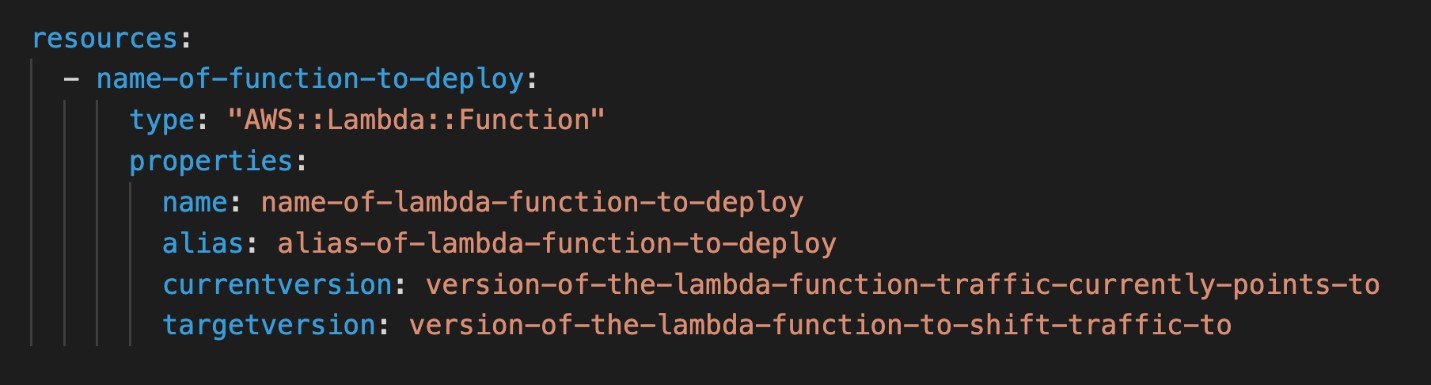
**TaskDefinition, PlatformVersion, and ContainerName**

**name, alias, PlatformVersion, and type**

Overall explanation

The content in the 'resources' section of the AppSpec file varies, depending on the compute platform of your deployment. The 'resources' section for an AWS Lambda deployment contains the name, alias, current version, and target version of a Lambda function.

Here is an example of a 'resources' section with the minimum required properties:



**CORRECT:**"name, alias, currentversion, and targetversion" is the correct answer (as explained above.)

**INCORRECT:** "name, alias, PlatformVersion, and type" is incorrect (as explained above.)

**INCORRECT:** "TaskDefinition, LoadBalancerInfo, and ContainerPort" is incorrect.

These properties are related to ECS deployments.

**INCORRECT:** "TaskDefinition, PlatformVersion, and ContainerName" is incorrect.

These properties are related to ECS deployments.

**References:**

<https://docs.aws.amazon.com/codedeploy/latest/userguide/reference-appspec-file-structure-resources.html>

**Save time with our AWS cheat sheets:**

<https://digitalcloud.training/aws-developer-tools/>

**Domain**

AWS Developer Tools

**Question 14Incorrect**

A developer plan to deploy an application on Amazon ECS that uses the AWS SDK to make API calls to Amazon DynamoDB. In the development environment the application was configured with access keys. The application is now ready for deployment to a production cluster.

How should the developer configure the application to securely authenticate to AWS services?

**Correct answer**

**Configure an ECS task IAM role for the application to use.**

**Add the necessary AWS service permissions to an ECS instance profile.**

**Your answer is incorrect**

**Add environment variables pointing to new access key credentials.**

**Configure the credentials file with a new access key/secret access key.**

Overall explanation

Your Amazon ECS tasks can have an IAM role associated with them. The permissions granted in the IAM role are assumed by the containers running in the task. The following explain the benefits of using IAM roles with your tasks.

• **Credential Isolation:** A container can only retrieve credentials for the IAM role that is defined in the task definition to which it belongs; a container never has access to credentials that are intended for another container that belongs to another task.

• **Authorization:** Unauthorized containers cannot access IAM role credentials defined for other tasks.

• **Auditability:** Access and event logging is available through CloudTrail to ensure retrospective auditing. Task credentials have a context of taskArn that is attached to the session, so CloudTrail logs show which task is using which role.

**CORRECT:**"Configure an ECS task IAM role for the application to use" is the correct answer (as explained above.)

**INCORRECT:** "Add the necessary AWS service permissions to an ECS instance profile" is incorrect.

The privileges assigned to instance profiles on the Amazon ECS instances are available to all tasks running on the instance. This is not secure and AWS recommend that you limit the permissions you assign to the instance profile.

**INCORRECT:** "Configure the credentials file with a new access key/secret access key" is incorrect.

Access keys are not a secure way of providing authentication. It is better to use roles that obtain temporary security permissions using the AWS STS service.

**INCORRECT:** "Add environment variables pointing to new access key credentials" is incorrect.

As above, access keys should not be used, IAM roles should be used instead.

**References:**

<https://docs.aws.amazon.com/AmazonECS/latest/developerguide/task-iam-roles.html>

**Save time with our AWS cheat sheets:**

<https://digitalcloud.training/aws-iam/>

**Domain**

AWS Compute

**Question 15Incorrect**

A Developer is creating an AWS Lambda function that will process data from an Amazon Kinesis data stream. The function is expected to be invoked 50 times per second and take 100 seconds to complete each request.

What MUST the Developer do to ensure the functions runs without errors?

**Implement exponential backoff in the function code**

**Correct answer**

**Contact AWS and request to increase the limit for concurrent executions**

**No action is required as AWS Lambda can easily accommodate this requirement**

**Your answer is incorrect**

**Increase the concurrency limit for the function**

Overall explanation

Concurrency is the number of requests that your function is serving at any given time. When your function is invoked, Lambda allocates an instance of it to process the event. When the function code finishes running, it can handle another request. If the function is invoked again while a request is still being processed, another instance is allocated, which increases the function's concurrency.

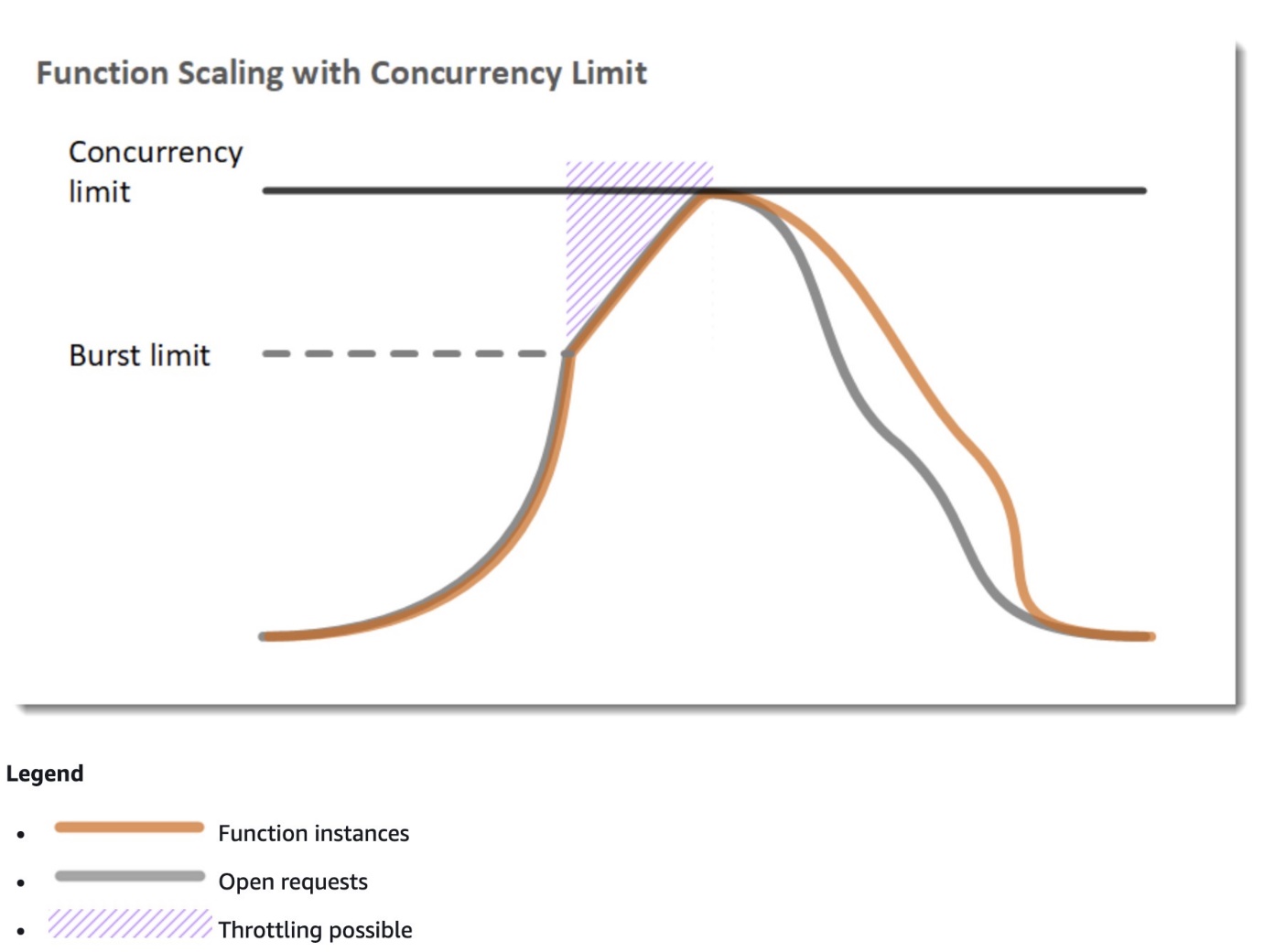
Concurrency is subject to a Regional limit that is shared by all functions in a Region. For an initial burst of traffic, your functions' cumulative concurrency in a Region can reach an initial level of between 500 and 3000, which varies per Region:

• **3000** – US West (Oregon), US East (N. Virginia), Europe (Ireland)

• **1000** – Asia Pacific (Tokyo), Europe (Frankfurt)

• **500** – Other Regions

After the initial burst, your functions' concurrency can scale by an additional 500 instances each minute. This continues until there are enough instances to serve all requests, or until a concurrency limit is reached. When requests come in faster than your function can scale, or when your function is at maximum concurrency, additional requests fail with a throttling error (429 status code).



The function continues to scale until the account's concurrency limit for the function's Region is reached. The function catches up to demand, requests subside, and unused instances of the function are stopped after being idle for some time. Unused instances are frozen while they're waiting for requests and don't incur any charges.

The regional concurrency limit starts at 1,000. You can increase the limit by submitting a request in the Support Center console.

**Calculating concurrency requirements for this scenario**

To calculate the concurrency requirements for this scenario, simply multiply the invocation requests per second (50) with the average execution time in seconds (100). This calculation is 50 x 100 = 5,000.

Therefore, 5,000 concurrent executions is over the default limit and the Developer will need to request in the AWS Support Center console.

**CORRECT:**"Contact AWS and request to increase the limit for concurrent executions" is the correct answer.

**INCORRECT:** "No action is required as AWS Lambda can easily accommodate this requirement" is incorrect as by default the AWS account will be limited. Lambda can easily scale to this level of demand however the account limits must first be increased.

**INCORRECT:** "Increase the concurrency limit for the function" is incorrect as the default account limit of 1,000 concurrent executions will mean you can only assign up to 900 executions to the function (100 must be left unreserved). This is insufficient for this requirement to the account limit must be increased.

**INCORRECT:** "Implement exponential backoff in the function code" is incorrect. Exponential backoff means configuring the application to wait longer between API calls, slowing the demand. However, this is not a good resolution to this issue as it will have negative effects on the application. The correct choice is to raise the account limits so the function can concurrently execute according to its requirements.

**References:**

<https://docs.aws.amazon.com/lambda/latest/dg/invocation-scaling.html>

**Save time with our AWS cheat sheets:**

<https://digitalcloud.training/aws-lambda/>

**Domain**

AWS Compute

**Question 16Incorrect**

A company is creating an application that will require users to access AWS services and allow them to reset their own passwords. Which of the following would allow the company to manage users and authorization while allowing users to reset their own passwords?

**Amazon Cognito user pools and AWS KMS**

**Amazon Cognito identity pools and AWS STS**

**Correct answer**

**Amazon Cognito user pools and identity pools**

**Your answer is incorrect**

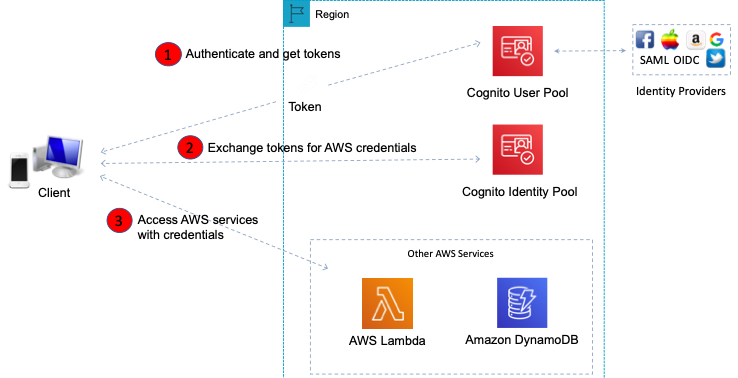
**Amazon Cognito identity pools and AWS IAM**

Overall explanation

There are two key requirements in this scenario. Firstly the company wants to manage user accounts using a system that allows users to reset their own passwords. The company also wants to authorize users to access AWS services.

The first requirement is provided by an Amazon Cognito User Pool. With a Cognito user pool you can add sign-up and sign-in to mobile and web apps and it also offers a user directory so user accounts can be created directly within the user pool. Users also have the ability to reset their passwords.

To access AWS services you need a Cognito Identity Pool. An identity pool can be used with a user pool and enables a user to obtain temporary limited-privilege credentials to access AWS services.



Therefore, the best answer is to use Amazon Cognito user pools and identity pools.

**CORRECT:**"Amazon Cognito user pools and identity pools" is the correct answer.

**INCORRECT:** "Amazon Cognito identity pools and AWS STS" is incorrect as there is no user directory in this solution. A Cognito user pool is required.

**INCORRECT:** "Amazon Cognito identity pools and AWS IAM" is incorrect as a Cognito user pool should be used as the directory source for creating and managing users. IAM is used for accounts that are used to administer AWS services, not for application user access.

**INCORRECT:** "Amazon Cognito user pools and AWS KMS" is incorrect as KMS is used for encryption, not for authentication to AWS services.

**References:**

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

**Save time with our exam-specific cheat sheets:**

<https://docs.aws.amazon.com/cognito/latest/developerguide/cognito-identity.html>

**Domain**

AWS Security, Identity, & Compliance

**Question 17Incorrect**

A team of Developers are building a continuous integration and delivery pipeline using AWS Developer Tools. Which services should they use for running tests against source code and installing compiled code on their AWS resources? (Select TWO.)

**Your selection is correct**

**AWS CodeBuild for running tests against source code**

**AWS Cloud9 for running tests against source code**

**Correct selection**

**AWS CodeDeploy for installing compiled code on their AWS resources**

**Your selection is incorrect**

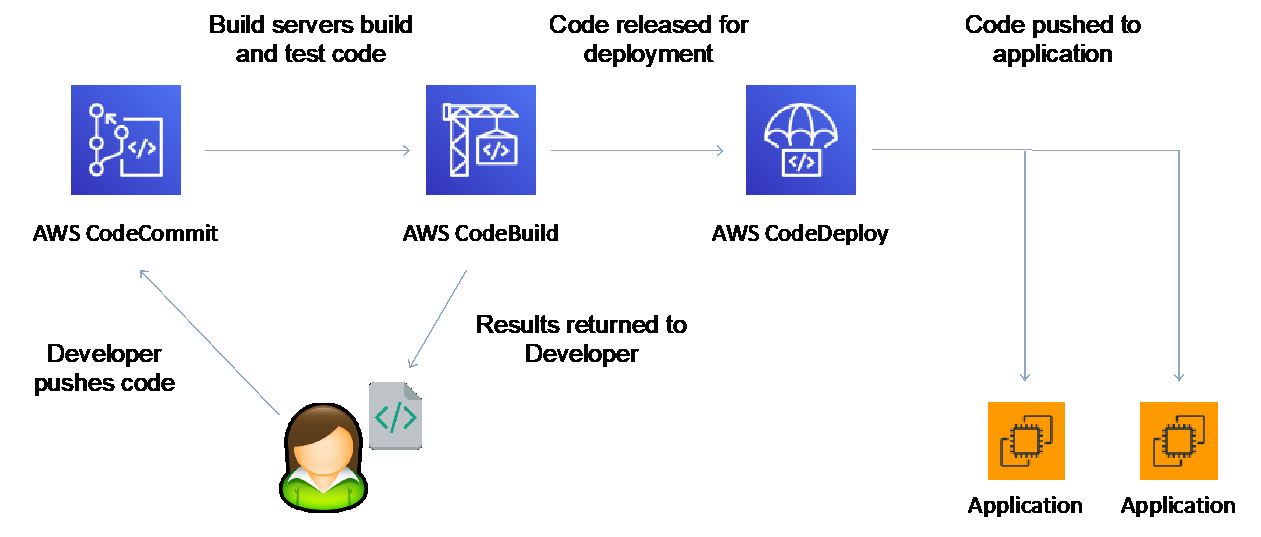
**AWS CodePipeline for running tests against source code**

**AWS CodeCommit for installing compiled code on their AWS resources**

Overall explanation

AWS CodeBuild is a fully managed build service in the cloud. CodeBuild compiles your source code, runs unit tests, and produces artifacts that are ready to deploy. CodeBuild eliminates the need to provision, manage, and scale your own build servers. It provides pre-packaged build environments for popular programming languages and build tools such as Apache Maven, Gradle, and more.

CodeDeploy is a deployment service that automates application deployments to Amazon EC2 instances, on-premises instances, serverless Lambda functions, or Amazon ECS services.



**CORRECT:**"AWS CodeBuild for running tests against source code" is a correct answer.

**CORRECT:**"AWS CodeDeploy for installing compiled code on their AWS resources" is also a correct answer.

**INCORRECT:** "AWS CodePipeline for running tests against source code" is incorrect. AWS CodePipeline is a fully managed continuous delivery service that helps you automate your release pipelines for fast and reliable application and infrastructure updates. This service works with the other Developer Tools to create a pipeline.

**INCORRECT:** "AWS CodeCommit for installing compiled code on their AWS resources" is incorrect as AWS CodeCommit is a fully-managed [source control](https://aws.amazon.com/devops/source-control/) service that hosts secure Git-based repositories.

**INCORRECT:** "AWS Cloud9 for running tests against source code" is incorrect as AWS Cloud9 is a cloud-based integrated development environment (IDE) that lets you write, run, and debug your code with just a browser.

**References:**

<https://docs.aws.amazon.com/codebuild/latest/userguide/welcome.html>

<https://docs.aws.amazon.com/codedeploy/latest/userguide/welcome.html>

**Save time with our AWS cheat sheets:**

<https://digitalcloud.training/aws-developer-tools/>

**Domain**

AWS Developer Tools

**Question 18Correct**

A company needs a version control system for collaborative software development. The solution must include support for batches of changes across multiple files and parallel branching.

Which AWS service will meet these requirements?

**Your answer is correct**

**AWS CodeCommit**

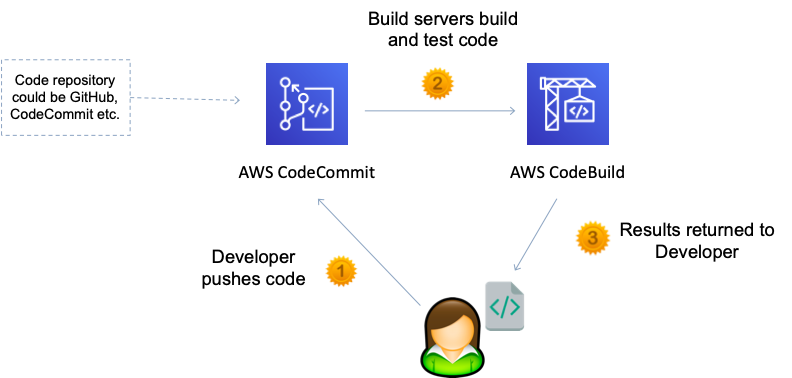
**AWS CodeBuild**

**Amazon S3**

**AWS CodePipeline**

Overall explanation

AWS CodeCommit is a version control service hosted by Amazon Web Services that you can use to privately store and manage assets (such as documents, source code, and binary files) in the cloud.



CodeCommit is optimized for team software development. It manages batches of changes across multiple files, which can occur in parallel with changes made by other developers.

**CORRECT:**"AWS CodeCommit" is the correct answer.

**INCORRECT:** "AWS CodeBuild" is incorrect as it is a fully managed continuous integration service that compiles source code, runs tests, and produces software packages that are ready to deploy.

**INCORRECT:** "AWS CodePipeline" is incorrect as it is a fully managed [continuous delivery](https://aws.amazon.com/devops/continuous-delivery/) service that helps you automate your release pipelines for fast and reliable application and infrastructure updates.

**INCORRECT:** "Amazon S3" is incorrect. Amazon S3 versioning supports the recovery of past versions of files, but it's not focused on collaborative file tracking features that software development teams need.

**References:**

<https://docs.aws.amazon.com/codecommit/latest/userguide/welcome.html>

**Save time with our AWS cheat sheets:**

<https://digitalcloud.training/aws-developer-tools/>

**Domain**

AWS Developer Tools

**Question 19Correct**

An application developer is crafting a new software product. To streamline the registration process, they want new users to be able to set up their accounts using their existing social media profiles.

Which AWS service or feature would be the most appropriate for achieving this goal?

**Your answer is correct**

**Amazon Cognito User Pools**

**AWS Managed Microsoft AD**

**AWS Identity and Access Management (IAM)**

**AWS Security Token Service**

Overall explanation

Amazon Cognito User Pools provide a secure and scalable directory to manage users. User Pools also support the ability for users to sign in through third-party identity providers, like social media platforms, which makes it the right choice for this requirement.

**CORRECT:**"Amazon Cognito User Pools" is the correct answer (as explained above.)

**INCORRECT:** "AWS Security Token Service" is incorrect.

AWS Security Token Service (STS) is primarily used to grant temporary, limited-privilege credentials for IAM users or for users that you authenticate (federated users), which is not related to the registration of application users via social media.

**INCORRECT:** "AWS Identity and Access Management (IAM)" is incorrect.

AWS IAM manages access to AWS services and resources securely. While IAM can federate with external identity providers, it is not designed to manage application user registration and social sign-in functionality directly.

**INCORRECT:** "AWS Managed Microsoft AD" is incorrect.

AWS Managed Microsoft AD is a service that is used to enable directory-aware workloads and AWS resources to use managed Active Directory in AWS. It's used for traditional enterprise applications and does not directly support social media-based user registration.

**References:**

<https://repost.aws/knowledge-center/cognito-user-pools-identity-pools>

**Save time with our AWS cheat sheets:**

<https://digitalcloud.training/amazon-cognito/>

**Domain**

AWS Security, Identity, & Compliance

**Question 20Correct**

Messages produced by an application must be pushed to multiple Amazon SQS queues. What is the BEST solution for this requirement?

**Create an Amazon SWF workflow that receives the messages and pushes them to multiple SQS queues**

**Your answer is correct**

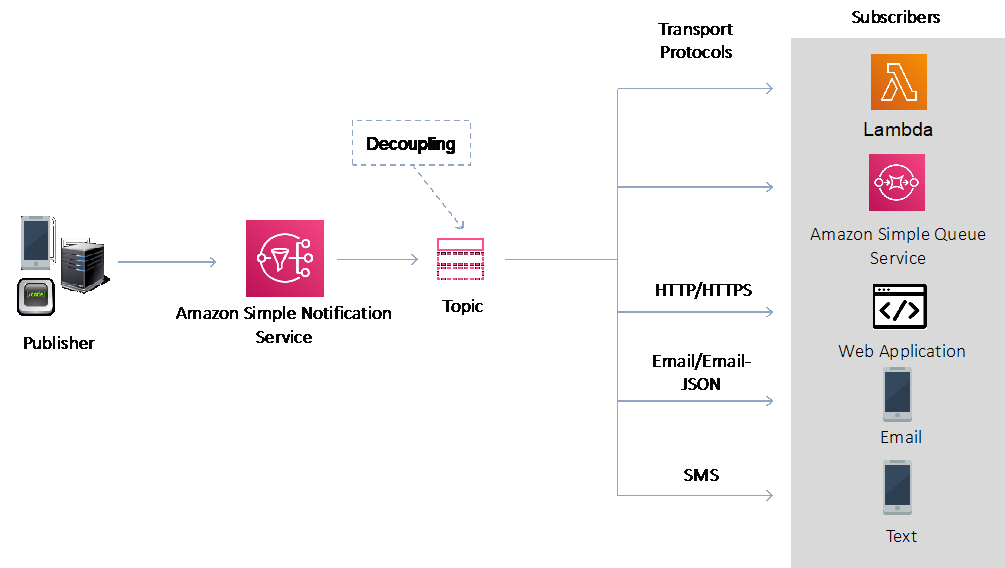
**Publish the messages to an Amazon SNS topic and subscribe each SQS queue to the topic**

**Publish the messages to an Amazon SQS queue and configure an AWS Lambda function to duplicate the message into multiple queues**

**Create and AWS Step Functions state machine that uses multiple Lambda functions to process and push the messages into multiple SQS queues**

Overall explanation

[Amazon SNS](https://aws.amazon.com/sns/) works closely with Amazon Simple Queue Service (Amazon SQS). Both services provide different benefits for developers. Amazon SNS allows applications to send time-critical messages to multiple subscribers through a “push” mechanism, eliminating the need to periodically check or “poll” for updates.



When you subscribe an Amazon SQS queue to an Amazon SNS topic, you can publish a message to the topic and Amazon SNS sends an Amazon SQS message to the subscribed queue. The Amazon SQS message contains the subject and message that were published to the topic along with metadata about the message in a JSON document.

**CORRECT:**"Publish the messages to an Amazon SNS topic and subscribe each SQS queue to the topic" is the correct answer.

**INCORRECT:** "Publish the messages to an Amazon SQS queue and configure an AWS Lambda function to duplicate the message into multiple queues" is incorrect as this seems like an inefficient solution. By using SNS we can eliminate the initial queue and Lambda function.

**INCORRECT:** "Create an Amazon SWF workflow that receives the messages and pushes them to multiple SQS queues" is incorrect as this is not a workable solution. Amazon SWF is not suitable for pushing messages to SQS queues.

**INCORRECT:** Create and AWS Step Functions state machine that uses multiple Lambda functions to process and push the messages into multiple SQS queues"" is incorrect as this is an inefficient solution and there is not mention on how the functions will be invoked with the message data

**References:**

<https://docs.aws.amazon.com/sns/latest/dg/sns-sqs-as-subscriber.html>

**Save time with our AWS cheat sheets:**

<https://digitalcloud.training/aws-application-integration-services/>

**Domain**

AWS Application Integration

**Question 21Correct**

A Developer is storing sensitive documents in Amazon S3. The documents must be encrypted at rest and company policy mandates that the encryption keys must be rotated annually. What is the EASIEST way to achieve this?

**Import a custom key into AWS KMS with annual rotation enabled**

**Export a key from AWS KMS to encrypt the data**

**Encrypt the data before sending it to Amazon S3**

**Your answer is correct**

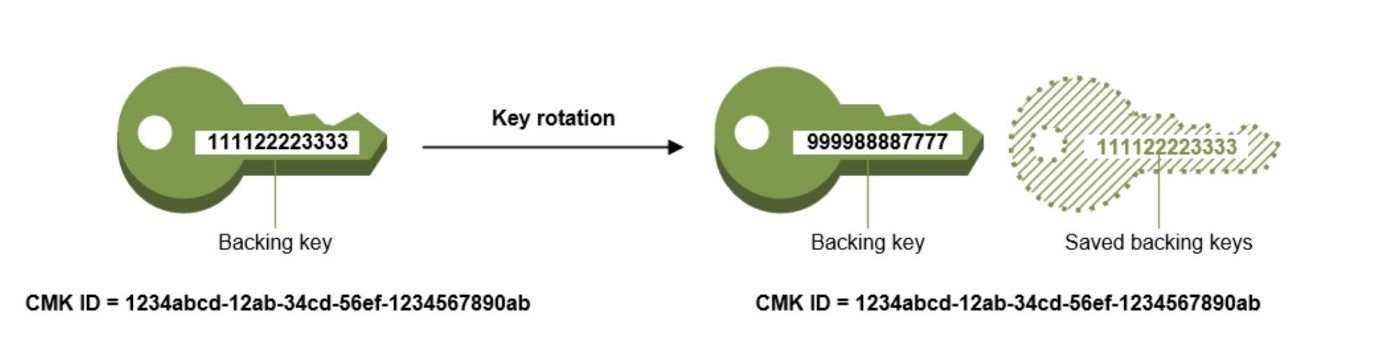
**Use AWS KMS with automatic key rotation**

Overall explanation

Cryptographic best practices discourage extensive reuse of encryption keys. To create new cryptographic material for your AWS Key Management Service (AWS KMS) customer master keys (CMKs), you can create new CMKs, and then change your applications or aliases to use the new CMKs. Or, you can enable automatic key rotation for an existing [customer managed CMK](https://docs.aws.amazon.com/kms/latest/developerguide/concepts.html#customer-cmk).

When you enable *automatic key rotation* for a customer managed CMK, AWS KMS generates new cryptographic material for the CMK every year. AWS KMS also saves the CMK's older cryptographic material in perpetuity so it can be used to decrypt data that it encrypted. AWS KMS does not delete any rotated key material until you [delete the CMK](https://docs.aws.amazon.com/kms/latest/developerguide/deleting-keys.html).

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



Therefore, the easiest way to meet this requirement is to use AWS KMS with automatic key rotation.

**CORRECT:**"Use AWS KMS with automatic key rotation" is the correct answer.

**INCORRECT:** "Encrypt the data before sending it to Amazon S3" is incorrect as that requires managing your own encryption infrastructure which is not the easiest way to achieve the requirements.

**INCORRECT:** "Import a custom key into AWS KMS with annual rotation enabled" is incorrect as when you import key material into AWS KMS you are still responsible for the key material while allowing KMS to use a copy of it. Therefore, this is not the easiest solution as you must manage the key materials.

**INCORRECT:** "Export a key from AWS KMS to encrypt the data" is incorrect as when you export a data encryption key you are then responsible for using it and managing it.

**References:**

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

**Save time with our AWS cheat sheets:**

<https://digitalcloud.training/aws-kms/>

**Domain**

AWS Security, Identity, & Compliance

**Question 22Incorrect**

The source code for an application is stored in a file named index.js that is in a folder along with a template file that includes the following code:

1. AWSTemplateFormatVersion: '2010-09-09'
2. Transform: 'AWS::Serverless-2016-10-31'
3. Resources:
4. LambdaFunctionWithAPI:
5. Type: AWS::Serverless::Function
6. Properties:
7. Handler: index.handler
8. Runtime: nodejs12.x

What does a Developer need to do to prepare the template so it can be deployed using an AWS CLI command?

**Run the aws cloudformation compile command to base64 encode and embed the source file into a modified CloudFormation template**

**Run the aws lambda zip command to package the source file together with the CloudFormation template and deploy the resulting zip archive**

**Your answer is incorrect**

**Run the aws serverless create-package command to embed the source file directly into the existing CloudFormation template**

**Correct answer**

**Run the aws cloudformation package command to upload the source code to an Amazon S3 bucket and produce a modified CloudFormation template**

Overall explanation

The template shown is an AWS SAM template for deploying a serverless application. This can be identified by the template header: *Transform: 'AWS::Serverless-2016-10-31'*

The Developer will need to package and then deploy the template. To do this the source code must be available in the same directory or referenced using the “codeuri” parameter. Then, the Developer can use the “aws cloudformation package” or “sam package” commands to prepare the local artifacts (local paths) that your AWS CloudFormation template references.

The command uploads local artifacts, such as source code for an AWS Lambda function or a Swagger file for an AWS API Gateway REST API, to an S3 bucket. The command returns a copy of your template, replacing references to local artifacts with the S3 location where the command uploaded the artifacts.

Once that is complete the template can be deployed using the “aws cloudformation deploy” or “sam deploy” commands. Therefore, the next step in this scenario is for the Developer to run the “aws cloudformation” package command to upload the source code to an Amazon S3 bucket and produce a modified CloudFormation template. An example of this command is provided below:

*aws cloudformation package --template-file /path\_to\_template/template.json --s3-bucket bucket-name --output-template-file packaged-template.json*

**CORRECT:**"Run the aws cloudformation package command to upload the source code to an Amazon S3 bucket and produce a modified CloudFormation template" is the correct answer.

**INCORRECT:** "Run the aws cloudformation compile command to base64 encode and embed the source file into a modified CloudFormation template" is incorrect as the Developer should run the “aws cloudformation package” command.

**INCORRECT:** "Run the aws lambda zip command to package the source file together with the CloudFormation template and deploy the resulting zip archive" is incorrect as the Developer should run the “aws cloudformation package” command which will automatically copy the relevant files to Amazon S3.

**INCORRECT:** "Run the aws serverless create-package command to embed the source file directly into the existing CloudFormation template" is incorrect as the Developer has the choice to run either “aws cloudformation package” or “sam package”, but not “aws serverless create-package”.

**References:**

<https://docs.aws.amazon.com/cli/latest/reference/cloudformation/package.html>

**Save time with our AWS cheat sheets:**

<https://digitalcloud.training/aws-sam/>

**Domain**

AWS Management & Governance

**Question 23Incorrect**

A Development team has deployed several applications running on an Auto Scaling fleet of Amazon EC2 instances. The Operations team have asked for a display that shows a key performance metric for each application on a single screen for monitoring purposes.

What steps should a Developer take to deliver this capability using Amazon CloudWatch?

**Create a custom alarm with a unique metric name for each application**

**Your answer is incorrect**

**Create a custom dimension with a unique metric name for each application**

**Correct answer**

**Create a custom namespace with a unique metric name for each application**

**Create a custom event with a unique metric name for each application**

Overall explanation

A *namespace* is a container for CloudWatch metrics. Metrics in different namespaces are isolated from each other, so that metrics from different applications are not mistakenly aggregated into the same statistics.

Therefore, the Developer should create a custom namespace with a unique metric name for each application. This namespace will then allow the metrics for each individual application to be shown in a single view through CloudWatch.

**CORRECT:**"Create a custom namespace with a unique metric name for each application" is the correct answer.

**INCORRECT:** "Create a custom dimension with a unique metric name for each application" is incorrect as a dimension further clarifies what a metric is and what data it stores.

**INCORRECT:** "Create a custom event with a unique metric name for each application" is incorrect as an event is not used to organize metrics for display.

**INCORRECT:** "Create a custom alarm with a unique metric name for each application" is incorrect as alarms are used to trigger actions when a threshold is reached, this is not relevant to organizing metrics for display.

**References:**

<https://docs.aws.amazon.com/AmazonCloudWatch/latest/monitoring/cloudwatch_concepts.html>

**Save time with our AWS cheat sheets:**

[https://digitalcloud.training/amazon-cloudwatch/](https://digitalcloud.training/certification-training/aws-developer-associate/aws-management-and-governance/amazon-cloudwatch/)

**Domain**

AWS Management & Governance

**Question 24Correct**

An application reads data from Amazon S3 and makes 55,000 read requests per second. A Developer must design the storage solution to ensure the performance requirements are met cost-effectively.

How can the storage be optimized to meet these requirements?

**Move the files to Amazon DynamoDB. Index the files with S3 metadata.**

**Create at least 10 S3 buckets and split the files across the buckets.**

**Your answer is correct**

**Create at least 10 prefixes and split the files across the prefixes.**

**Move the files to Amazon EFS. Index the files with S3 metadata.**

Overall explanation

To avoid throttling in Amazon S3 you must ensure you do not exceed certain limits on a per-prefix basis. You can send 3,500 PUT/COPY/POST/DELETE or 5,500 GET/HEAD requests per second per prefix in an Amazon S3 bucket. There are no limits to the number of prefixes that you can have in your bucket.

In this case the Developer would need to split the files across at least 10 prefixes in a single Amazon S3 bucket. The application should then read the files across the prefixes in parallel.

**CORRECT:**"Create at least 10 prefixes and split the files across the prefixes" is the correct answer.

**INCORRECT:** "Create at least 10 S3 buckets and split the files across the buckets" is incorrect. Performance is improved based on splitting reads across prefixes, not buckets.

**INCORRECT:** "Move the files to Amazon EFS. Index the files with S3 metadata" is incorrect. This is not cost-effective.

**INCORRECT:** "Move the files to Amazon DynamoDB. Index the files with S3 metadata" is incorrect. This is not cost-effective.

**References:**

<https://aws.amazon.com/premiumsupport/knowledge-center/s3-request-limit-avoid-throttling/>

**Save time with our AWS cheat sheets:**

<https://digitalcloud.training/amazon-s3-and-glacier/>

**Domain**

AWS Storage

**Question 25Correct**

A website is running on a single Amazon EC2 instance. A Developer wants to publish the website on the Internet and is creating an A record on Amazon Route 53 for the website’s public DNS name.

What type of IP address MUST be assigned to the EC2 instance and used in the A record to ensure ongoing connectivity?

**Private IP address**

**Dynamic IP address**

**Public IP address**

**Your answer is correct**

**Elastic IP address**

Overall explanation

In Amazon Route 53 when you create an A record you must supply an IP address for the resource to connect to. For a public hosted zone this must be a public IP address.

There are three types of IP address that can be assigned to an Amazon EC2 instance:

• Public – public address that is assigned automatically to instances in public subnets and reassigned if instance is stopped/started.

• Private – private address assigned automatically to all instances.

• Elastic IP – public address that is static.

To ensure ongoing connectivity the Developer needs to use an Elastic IP address for the EC2 instance and DNS A record as this is the only type of static, public IP address you can assign to an Amazon EC2 instance.

**CORRECT:**"Elastic IP address" is the correct answer.

**INCORRECT:** "Public IP address" is incorrect as though this is a public IP address, it is not static and will change every time the EC2 instance restarts. Therefore, connectivity would be lost until you update the Route 53 A record.

**INCORRECT:** "Dynamic IP address" is incorrect as a dynamic IP address is an IP address that will change over time. For this scenario a static, public address is required.

**INCORRECT:** "Private IP address" is incorrect as a public IP address is required for the public DNS A record.

**References:**

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

<https://docs.aws.amazon.com/Route53/latest/DeveloperGuide/ResourceRecordTypes.html>

**Save time with our AWS cheat sheets:**

<https://digitalcloud.training/amazon-ec2/>

<https://digitalcloud.training/amazon-route-53/>

**Domain**

AWS Networking & Content Delivery

**Question 26Incorrect**

A developer must deploy an update to Amazon ECS using AWS CodeDeploy. The deployment should expose 10% of live traffic to the new version. Then after a period of time, route all remaining traffic to the new version.

Which ECS deployment should the company use to meet these requirements?

**Correct answer**

**Blue/green with canary**

**Blue/green with all at once**

**Your answer is incorrect**

**Rolling update**

**Blue/green with linear**

Overall explanation

The *blue/green* deployment type uses the blue/green deployment model controlled by CodeDeploy. This deployment type enables you to verify a new deployment of a service before sending production traffic to it.

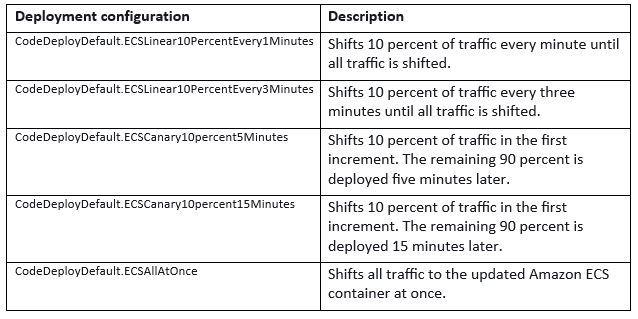
There are three ways traﬃc can shift during a blue/green deployment:

• **Canary** — Traﬃc is shifted in two increments. You can choose from predeﬁned canary options that specify the percentage of traﬃc shifted to your updated task set in the ﬁrst increment and the interval, in minutes, before the remaining traﬃc is shifted in the second increment.

• **Linear** — Traﬃc is shifted in equal increments with an equal number of minutes between each increment. You can choose from predeﬁned linear options that specify the percentage of traﬃc shifted in each increment and the number of minutes between each increment.

• **All-at-once** — All traﬃc is shifted from the original task set to the updated task set all at once.

The best choice for this use case would be to use the canary traffic shifting strategy. You can see the predefined canary options in the table below:



**CORRECT:**"Blue/green with canary" is the correct answer (as explained above.)

**INCORRECT:** "Blue/green with linear" is incorrect.

With this option traffic is shifted in equal increments with an equal amount of time between increments.

**INCORRECT:** "Blue/green with all at once" is incorrect.

With this option all traffic is shifted at once.

**INCORRECT:** "Rolling update" is incorrect.

This is a native ECS deployment model. It does not deploy in two increments with 10% first.

**References:**

<https://docs.aws.amazon.com/AmazonECS/latest/developerguide/deployment-type-bluegreen.html>

**Save time with our AWS cheat sheets:**

<https://digitalcloud.training/aws-developer-tools/>

**Domain**

AWS Developer Tools

**Question 27Incorrect**

An application has been instrumented to use the AWS X-Ray SDK to collect data about the requests the application serves. The Developer has set the user field on segments to a string that identifies the user who sent the request.

How can the Developer search for segments associated with specific users?

**Correct answer**

**By using the GetTraceSummaries API with a filter expression**

**Use a filter expression to search for the user field in the segment annotations**

**Your answer is incorrect**

**By using the GetTraceGraph API with a filter expression**

**Use a filter expression to search for the user field in the segment metadata**

Overall explanation

A **segment document** conveys information about a segment to X-Ray. A segment document can be up to 64 kB and contain a whole segment with subsegments, a fragment of a segment that indicates that a request is in progress, or a single subsegment that is sent separately. You can send segment documents directly to X-Ray by using the [PutTraceSegments](https://docs.aws.amazon.com/xray/latest/api/API_PutTraceSegments.html) API.

Example minimally complete segment:

1. {
2. "name" : "example.com",
3. "id" : "70de5b6f19ff9a0a",
4. "start\_time" : 1.478293361271E9,
5. "trace\_id" : "1-581cf771-a006649127e371903a2de979",
6. "end\_time" : 1.478293361449E9
7. }

A subset of segment fields are indexed by X-Ray for use with filter expressions. For example, if you set the user field on a segment to a unique identifier, you can search for segments associated with specific users in the X-Ray console or by using the GetTraceSummaries API.

**CORRECT:**"By using the GetTraceSummaries API with a filter expression" is the correct answer.

**INCORRECT:** "By using the GetTraceGraph API with a filter expression" is incorrect as this API action retrieves a service graph for one or more specific trace IDs.

**INCORRECT:** "Use a filter expression to search for the user field in the segment metadata" is incorrect as the user field is not part of the segment metadata and metadata is not is not indexed for search.

**INCORRECT:** "Use a filter expression to search for the user field in the segment annotations" is incorrect as the user field is not part of the segment annotations.

**References:**

<https://docs.aws.amazon.com/xray/latest/devguide/xray-api-segmentdocuments.html>

**Save time with our AWS cheat sheets:**

<https://digitalcloud.training/aws-developer-tools/>

**Domain**

AWS Developer Tools

**Question 28Correct**

A company runs an application on a fleet of web servers running on Amazon EC2 instances. The web servers are behind an Elastic Load Balancer (ELB) and use an Amazon DynamoDB table for storing session state. A Developer has been asked to implement a mechanism for automatically deleting session state data that is older than 24 hours.

What is the SIMPLEST solution to this requirement?

**Add an attribute with the expiration time; name the attribute ItemExpiration**

**Each day, create a new table to hold session data; delete the previous day's table**

**Your answer is correct**

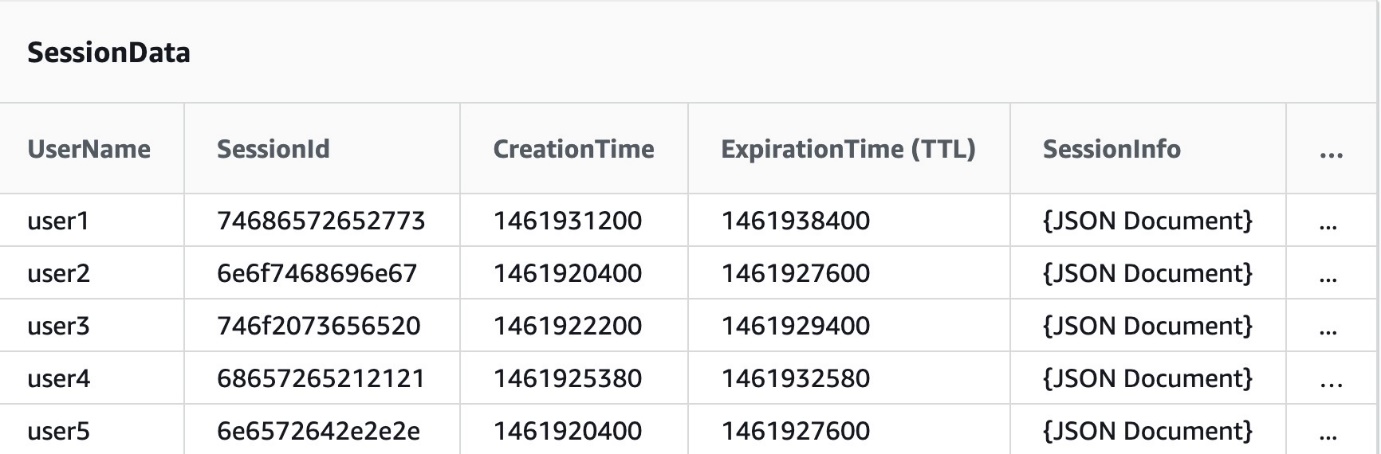
**Add an attribute with the expiration time; enable the Time To Live feature based on that attribute**

**Write a script that deletes old records; schedule the scripts as a cron job on an Amazon EC2 instance**

Overall explanation

Time to Live (TTL) for Amazon DynamoDB lets you define when items in a table expire so that they can be automatically deleted from the database. With TTL enabled on a table, you can set a timestamp for deletion on a per-item basis, allowing you to limit storage usage to only those records that are relevant.

TTL is useful if you have continuously accumulating data that loses relevance after a specific time period (for example, session data, event logs, usage patterns, and other temporary data). If you have sensitive data that must be retained only for a certain amount of time according to contractual or regulatory obligations, TTL helps you ensure that it is removed promptly and as scheduled.



When Time to Live (TTL) is enabled on a table in Amazon DynamoDB, a background job checks the TTL attribute of items to determine whether they are expired.

DynamoDB compares the current time, in [epoch time format](https://en.wikipedia.org/wiki/Unix_time), to the value stored in the user-defined Number attribute of an item. If the attribute’s value is in the epoch time format, is less than the current time, and is not older than 5 years, the item is deleted.

Processing takes place automatically, in the background, and doesn't affect read or write traffic to the table. In addition, deletes performed via TTL are not counted towards capacity units or request units. TTL deletes are available at no additional cost.

For this requirement, the Developer must add an attribute to each item with the expiration time in epoch format and then enable the Time To Live (TTL) feature based on that attribute.

**CORRECT:**"Add an attribute with the expiration time; enable the Time To Live feature based on that attribute" is the correct answer.

**INCORRECT:** "Each day, create a new table to hold session data; delete the previous day's table" is incorrect. This solution would delete some data that is not 24 hours old as it would have to run at a specific time.

**INCORRECT:** "Write a script that deletes old records; schedule the scripts as a cron job on an Amazon EC2 instance" is incorrect. This is not an elegant solution and would also cost more as it requires RCUs/WCUs to delete the items.

**INCORRECT:** "Add an attribute with the expiration time; name the attribute ItemExpiration" is incorrect as this is not a complete solution. You also need to enable the TTL feature on the table.

**References:**

<https://docs.aws.amazon.com/amazondynamodb/latest/developerguide/howitworks-ttl.html>

**Save time with our AWS cheat sheets:**

<https://digitalcloud.training/amazon-dynamodb/>

**Domain**

AWS Database

**Question 29Incorrect**

An application is being migrated into the cloud. The application is stateless and will run on a fleet of Amazon EC2 instances. The application should scale elastically. How can a Developer ensure that the number of instances available is sufficient for current demand?

**Create a task definition and use an AWS Fargate cluster**

**Correct answer**

**Create a launch configuration and use Amazon EC2 Auto Scaling**

**Your answer is incorrect**

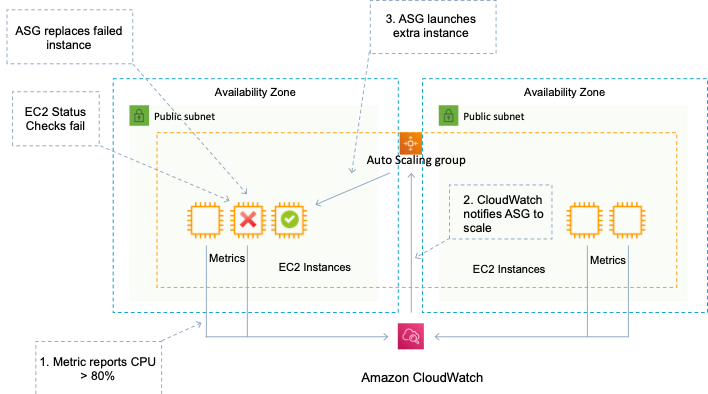
**Create a task definition and use an Amazon ECS cluster**

**Create a launch configuration and use Amazon CodeDeploy**

Overall explanation

Amazon EC2 Auto Scaling helps you maintain application availability and allows you to automatically add or remove EC2 instances according to conditions you define. You can use the fleet management features of EC2 Auto Scaling to maintain the health and availability of your fleet.

You can also use the dynamic and predictive scaling features of EC2 Auto Scaling to add or remove EC2 instances. Dynamic scaling responds to changing demand and predictive scaling automatically schedules the right number of EC2 instances based on predicted demand. Dynamic scaling and predictive scaling can be used together to scale faster.



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. Include the ID of the Amazon Machine Image (AMI), the instance type, a key pair, one or more security groups, and a block device mapping. If you've launched an EC2 instance before, you specified the same information in order to launch the instance.

You can specify your launch configuration with multiple Auto Scaling groups. However, you can only specify one launch configuration for an Auto Scaling group at a time, and you can't modify a launch configuration after you've created it. To change the launch configuration for an Auto Scaling group, you must create a launch configuration and then update your Auto Scaling group with it.

Therefore, the Developer should create a launch configuration and use Amazon EC2 Auto Scaling.

**CORRECT:**"Create a launch configuration and use Amazon EC2 Auto Scaling" is the correct answer.

**INCORRECT:** "Create a launch configuration and use Amazon CodeDeploy" is incorrect as CodeDeploy is not used for auto scaling of Amazon EC2 instances.

**INCORRECT:** "Create a task definition and use an Amazon ECS cluster" is incorrect as the migrated application will be running on Amazon EC2 instances, not containers.

**INCORRECT:** "Create a task definition and use an AWS Fargate cluster" is incorrect as the migrated application will be running on Amazon EC2 instances, not containers.

**References:**

<https://docs.aws.amazon.com/autoscaling/ec2/userguide/what-is-amazon-ec2-auto-scaling.html>

<https://docs.aws.amazon.com/autoscaling/ec2/userguide/LaunchConfiguration.html>

**Save time with our AWS cheat sheets:**

<https://digitalcloud.training/amazon-ec2-auto-scaling/>

**Domain**

AWS Compute

**Question 30Correct**

A company wants to implement authentication for its new REST service using Amazon API Gateway. To authenticate the calls, each request must include HTTP headers with a client ID and user ID. These credentials must be compared to authentication data in an Amazon DynamoDB table.

What MUST the company do to implement this authentication in API Gateway?

**Modify the integration requests to require the credentials, then grant API Gateway access to the authentication table**

**Create a model that requires the credentials, then grant API Gateway access to the authentication table**

**Your answer is correct**

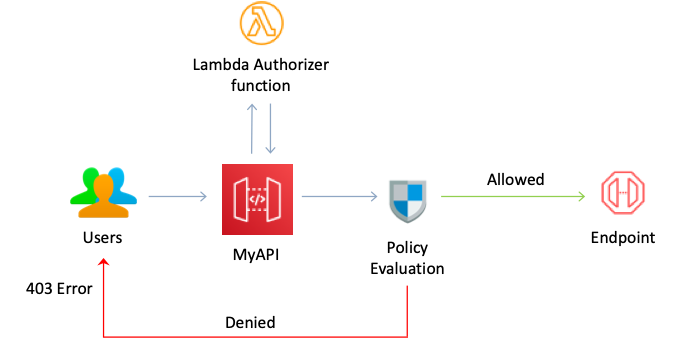
**Implement an AWS Lambda authorizer that references the DynamoDB authentication table**

**Implement an Amazon Cognito authorizer that references the DynamoDB authentication table**

Overall explanation

A *Lambda authorizer* (formerly known as a *custom authorizer*) is an API Gateway feature that uses a Lambda function to control access to your API.

A Lambda authorizer is useful if you want to implement a custom authorization scheme that uses a bearer token authentication strategy such as OAuth or SAML, or that uses request parameters to determine the caller's identity.



When a client makes a request to one of your API's methods, API Gateway calls your Lambda authorizer, which takes the caller's identity as input and returns an IAM policy as output.

There are two types of Lambda authorizers:

• A *token-based* Lambda authorizer (also called a TOKEN authorizer) receives the caller's identity in a bearer token, such as a JSON Web Token (JWT) or an OAuth token.

• A *request parameter-based* Lambda authorizer (also called a REQUEST authorizer) receives the caller's identity in a combination of headers, query string parameters, [stageVariables](https://docs.aws.amazon.com/apigateway/latest/developerguide/api-gateway-mapping-template-reference.html" \l "stagevariables-template-reference), and [$context](https://docs.aws.amazon.com/apigateway/latest/developerguide/api-gateway-mapping-template-reference.html#context-variable-reference) variables.

• For WebSocket APIs, only request parameter-based authorizers are supported.

In this scenario, the authentication is using headers in the request and therefore the request parameter-based Lambda authorizer should be used.

**CORRECT:**"Implement an AWS Lambda authorizer that references the DynamoDB authentication table" is the correct answer.

**INCORRECT:** "Create a model that requires the credentials, then grant API Gateway access to the authentication table" is incorrect as a model defines the structure of the incoming payload using the JSON Schema.

**INCORRECT:** "Modify the integration requests to require the credentials, then grant API Gateway access to the authentication table" is incorrect as API Gateway will not authorize directly using the table information, an authorizer should be used.

**INCORRECT:** "Implement an Amazon Cognito authorizer that references the DynamoDB authentication table" is incorrect as a Lambda authorizer should be used in this example as the authentication data is being passed in request headers.

**References:**

<https://docs.aws.amazon.com/apigateway/latest/developerguide/apigateway-use-lambda-authorizer.html>

**Save time with our AWS cheat sheets:**

<https://digitalcloud.training/aws-lambda/>

**Domain**

AWS Networking & Content Delivery

**Question 31Correct**

A developer is creating a microservices application that includes and AWS Lambda function. The function generates a unique file for each execution and must commit the file to an AWS CodeCommit repository.

How should the developer accomplish this?

**Your answer is correct**

**Use an AWS SDK to instantiate a CodeCommit client. Invoke the PutFile method to add the file to the repository and execute a commit with CreateCommit.**

**Send a message to an Amazon SQS queue with the file attached. Configure an AWS Step Function as a destination for messages in the queue. Configure the Step Function to add the new file to the repository and commit the change.**

**Upload the new file to an Amazon S3 bucket. Create an AWS Step Function to accept S3 events. Use AWS Lambda functions in the Step Function, to add the file to the repository and commit the change.**

**After the new file is created in Lambda, use CURL to invoke the CodeCommit API. Send the file to the repository and automatically commit the change.**

Overall explanation

The developer can instantiate a CodeCommit client using the AWS SDK. This provides the ability to programmatically work with the AWS CodeCommit repository. The PutFile method is used to add or modify a single file in a specified repository and branch. The CreateCommit method creates a commit for changes to a repository.

**CORRECT:**"Use an AWS SDK to instantiate a CodeCommit client. Invoke the PutFile method to add the file to the repository and execute a commit with CreateCommit" is the correct answer (as explained above.)

**INCORRECT:** "Send a message to an Amazon SQS queue with the file attached. Configure an AWS Step Function as a destination for messages in the queue. Configure the Step Function to add the new file to the repository and commit the change" is incorrect.

A Step Function cannot be a destination for messages in an SQS queue. There would need to be another Lambda function or other method to trigger the state machine and pass the information across. This would be a highly inefficient solution.

**INCORRECT:** "After the new file is created in Lambda, use CURL to invoke the CodeCommit API. Send the file to the repository and automatically commit the change" is incorrect.

CURL cannot be used to work with the CodeCommit API. The developer must use the AWS SDK.

**INCORRECT:** "Upload the new file to an Amazon S3 bucket. Create an AWS Step Function to accept S3 events. Use AWS Lambda functions in the Step Function, to add the file to the repository and commit the change" is incorrect.

Step Functions is not a supported destination for Amazon S3 event notifications. Supported destinations are SNS topics, SQS queues, Lambda functions, and EventBridge event buses.

**References:**

<https://docs.aws.amazon.com/AWSJavaSDK/latest/javadoc/com/amazonaws/services/codecommit/AWSCodeCommitClient.html>

**Save time with our AWS cheat sheets:**

<https://digitalcloud.training/aws-developer-tools/>

**Domain**

AWS Developer Tools

**Question 32Incorrect**

A developer is creating an AWS Serverless Application Model (AWS SAM) template. It includes several AWS Lambda functions, an Amazon S3 bucket, and an Amazon CloudFront distribution. One Lambda function, running on Lambda@Edge, is integrated with the CloudFront distribution, while the S3 bucket serves as an origin for the distribution.

However, upon deploying the AWS SAM blueprint in the us-west-1 Region, the stack's creation fails.

What could be the possible reason for this failure?

**AWS SAM templates are not supported in the us-west-1 Region.**

**AWS Lambda functions integrated with CloudFront cannot be deployed using AWS SAM templates.**

**Your answer is incorrect**

**Amazon S3 buckets serving as origins for CloudFront must be created in a separate Region from the CloudFront distribution.**

**Correct answer**

**Lambda@Edge functions can only be deployed in the us-east-1 Region.**

Overall explanation

Lambda@Edge functions can only be created in the us-east-1 Region. If you want to deploy such functions, they must be done in this specific region.

**CORRECT:**"Lambda@Edge functions can only be deployed in the us-east-1 Region" is the correct answer (as explained above.)

**INCORRECT:** "AWS SAM templates are not supported in the us-west-1 Region" is incorrect.

This is not a restriction, and hence not a reason for failure.

**INCORRECT:** "Amazon S3 buckets serving as origins for CloudFront must be created in a separate Region from the CloudFront distribution" is incorrect.

Amazon S3 buckets serving as origins for CloudFront distributions can be in the same region as the distribution. Therefore, this would not cause a failure.

**INCORRECT:** "AWS Lambda functions integrated with CloudFront cannot be deployed using AWS SAM templates" is incorrect.

AWS Lambda functions can be integrated with Amazon CloudFront and deployed using AWS SAM templates. This statement is incorrect and would not lead to a stack creation failure.

**References:**

<https://docs.aws.amazon.com/AmazonCloudFront/latest/DeveloperGuide/lambda-edge-how-it-works.html>

**Save time with our AWS cheat sheets:**

<https://digitalcloud.training/aws-sam/>

**Domain**

AWS Compute

**Question 33Incorrect**

An application uses an Auto Scaling group of Amazon EC2 instances, an Application Load Balancer (ALB), and an Amazon Simple Queue Service (SQS) queue. An Amazon CloudFront distribution caches content for global users. A Developer needs to add in-transit encryption to the data by configuring end-to-end SSL between the CloudFront Origin and the end users.

How can the Developer meet this requirement? (Select TWO.)

**Add a certificate to the Auto Scaling Group**

**Correct selection**

**Configure the Viewer Protocol Policy**

**Create an Origin Access Identity (OAI)**

**Your selection is incorrect**

**Create an encrypted distribution**

**Your selection is correct**

**Configure the Origin Protocol Policy**

Overall explanation

For web distributions, you can configure CloudFront to require that viewers use HTTPS to request your objects, so that connections are encrypted when CloudFront communicates with viewers. You also can configure CloudFront to use HTTPS to get objects from your origin, so that connections are encrypted when CloudFront communicates with your origin.

If you configure CloudFront to require HTTPS both to communicate with viewers and to communicate with your origin, here's what happens when CloudFront receives a request for an object:

1. A viewer submits an HTTPS request to CloudFront. There's some SSL/TLS negotiation here between the viewer and CloudFront. In the end, the viewer submits the request in an encrypted format.

2. If the object is in the CloudFront edge cache, CloudFront encrypts the response and returns it to the viewer, and the viewer decrypts it.

3. If the object is not in the CloudFront cache, CloudFront performs SSL/TLS negotiation with your origin and, when the negotiation is complete, forwards the request to your origin in an encrypted format.

4. Your origin decrypts the request, encrypts the requested object, and returns the object to CloudFront.

5. CloudFront decrypts the response, re-encrypts it, and forwards the object to the viewer. CloudFront also saves the object in the edge cache so that the object is available the next time it's requested.

6. The viewer decrypts the response.

To enable SSL between the origin and the distribution the Developer can configure the Origin Protocol Policy. Depending on the domain name used (CloudFront default or custom), the steps are different. To enable SSL between the end-user and CloudFront the Viewer Protocol Policy should be configured.

**CORRECT:**"Configure the Origin Protocol Policy" is a correct answer.

**CORRECT:**"Configure the Viewer Protocol Policy" is also a correct answer.

**INCORRECT:** "Create an Origin Access Identity (OAI)" is incorrect as this is a special user used for securing objects in Amazon S3 origins.

**INCORRECT:** "Add a certificate to the Auto Scaling Group" is incorrect as you do not add certificates to an ASG. The certificate should be located on the ALB listener in this scenario.

**INCORRECT:** "Create an encrypted distribution" is incorrect as there’s no such thing as an encrypted distribution

**References:**

<https://docs.aws.amazon.com/AmazonCloudFront/latest/DeveloperGuide/using-https-viewers-to-cloudfront.html>

<https://docs.aws.amazon.com/AmazonCloudFront/latest/DeveloperGuide/using-https-cloudfront-to-custom-origin.html>

**Save time with our AWS cheat sheets:**

<https://digitalcloud.training/amazon-cloudfront/>

**Domain**

AWS Networking & Content Delivery

**Question 34Correct**

A Developer wants to debug an application by searching and filtering log data. The application logs are stored in Amazon CloudWatch Logs. The Developer creates a new metric filter to count exceptions in the application logs. However, no results are returned from the logs.

What is the reason that no filtered results are being returned?

**The log group for CloudWatch Logs should be first streamed to Amazon Elasticsearch Service before filtering returns the results**

**Your answer is correct**

**CloudWatch Logs only publishes metric data for events that happen after the filter is created**

**Metric data points to logs groups can be filtered only after they are exported to an Amazon S3 bucket**

**A setup of the Amazon CloudWatch interface VPC endpoint is required for filtering the CloudWatch Logs in the VPC**

Overall explanation

After the CloudWatch Logs agent begins publishing log data to Amazon CloudWatch, you can begin searching and filtering the log data by creating one or more metric filters. Metric filters define the terms and patterns to look for in log data as it is sent to CloudWatch Logs.

CloudWatch Logs uses these metric filters to turn log data into numerical CloudWatch metrics that you can graph or set an alarm on. You can use any type of CloudWatch statistic, including percentile statistics, when viewing these metrics or setting alarms.

Filters do not retroactively filter data. Filters only publish the metric data points for events that happen after the filter was created. Filtered results return the first 50 lines, which will not be displayed if the timestamp on the filtered results is earlier than the metric creation time.

Therefore, the filtered results are not being returned as CloudWatch Logs only publishes metric data for events that happen after the filter is created.

**CORRECT:**"CloudWatch Logs only publishes metric data for events that happen after the filter is created" is the correct answer.

**INCORRECT:** "A setup of the Amazon CloudWatch interface VPC endpoint is required for filtering the CloudWatch Logs in the VPC" is incorrect as a VPC endpoint is not required.

**INCORRECT:** "The log group for CloudWatch Logs should be first streamed to Amazon Elasticsearch Service before filtering returns the results" is incorrect as you do not need to stream the results to Elasticsearch.

**INCORRECT:** "Metric data points to logs groups can be filtered only after they are exported to an Amazon S3 bucket" is incorrect as it is not necessary to export the logs to an S3 bucket.

**References:**

<https://docs.aws.amazon.com/AmazonCloudWatch/latest/logs/MonitoringLogData.html>

**Save time with our AWS cheat sheets:**

<https://digitalcloud.training/amazon-cloudwatch/>

**Domain**

AWS Management & Governance

**Question 35Incorrect**

A Developer is designing a cloud native application. The application will use several AWS Lambda functions that will process items that the functions read from an event source. Which AWS services are supported for Lambda event source mappings? (Select THREE.)

**Your selection is incorrect**

**Amazon Simple Notification Service (SNS)**

**Correct selection**

**Amazon Simple Queue Service (SQS)**

**Your selection is correct**

**Amazon DynamoDB**

**Another Lambda function**

**Your selection is incorrect**

**Amazon Simple Storage Service (S3)**

**Correct selection**

**Amazon Kinesis**

Overall explanation

An event source mapping is an AWS Lambda resource that reads from an event source and invokes a Lambda function. You can use event source mappings to process items from a stream or queue in services that don't invoke Lambda functions directly. Lambda provides event source mappings for the following services.

**Services That Lambda Reads Events From**

• [Amazon Kinesis](https://docs.aws.amazon.com/lambda/latest/dg/with-kinesis.html)

• [Amazon DynamoDB](https://docs.aws.amazon.com/lambda/latest/dg/with-ddb.html)

• [Amazon Simple Queue Service](https://docs.aws.amazon.com/lambda/latest/dg/with-sqs.html)

An event source mapping uses permissions in the function's [execution role](https://docs.aws.amazon.com/lambda/latest/dg/lambda-intro-execution-role.html) to read and manage items in the event source. Permissions, event structure, settings, and polling behavior vary by event source.

**CORRECT:**"Amazon Kinesis, Amazon DynamoDB, and Amazon Simple Queue Service (SQS)" are the correct answers.

**INCORRECT:** "Amazon Simple Notification Service (SNS)" is incorrect as SNS should be used as destination for asynchronous invocation.

**INCORRECT:** "Amazon Simple Storage Service (S3)" is incorrect as Lambda does not read from Amazon S3, you must configure the event notification on the S3 side.

**INCORRECT:** "Another Lambda function" is incorrect as another function should be invoked asynchronously.

**References:**

<https://docs.aws.amazon.com/lambda/latest/dg/invocation-eventsourcemapping.html>

**Save time with our AWS cheat sheets:**

<https://digitalcloud.training/aws-lambda/>

**Domain**

AWS Compute

**Question 36Incorrect**

An application uses Amazon API Gateway, an AWS Lambda function and a DynamoDB table. The developer requires that another Lambda function is triggered when an item lifecycle activity occurs in the DynamoDB table.

How can this be achieved?

**Correct answer**

**Enable a DynamoDB stream and trigger the Lambda function synchronously from the stream**

**Enable a DynamoDB stream and trigger the Lambda function asynchronously from the stream**

**Configure an Amazon CloudTrail API alarm that sends a message to an Amazon SQS queue. Configure the Lambda function to poll the queue and invoke the function synchronously**

**Your answer is incorrect**

**Configure an Amazon CloudWatch alarm that sends an Amazon SNS notification. Trigger the Lambda function asynchronously from the SNS notification**

Overall explanation

Amazon DynamoDB is integrated with AWS Lambda so that you can create *triggers*—pieces of code that automatically respond to events in DynamoDB Streams. With triggers, you can build applications that react to data modifications in DynamoDB tables.

If you enable DynamoDB Streams on a table, you can associate the stream Amazon Resource Name (ARN) with an AWS Lambda function that you write. Immediately after an item in the table is modified, a new record appears in the table's stream. AWS Lambda polls the stream and invokes your Lambda function synchronously when it detects new stream records.

**CORRECT:**"Enable a DynamoDB stream and trigger the Lambda function synchronously from the stream" is the correct answer.

**INCORRECT:** "Enable a DynamoDB stream and trigger the Lambda function asynchronously from the stream" is incorrect as the invocation should be synchronous.

**INCORRECT:** "Configure an Amazon CloudWatch alarm that sends an Amazon SNS notification. Trigger the Lambda function asynchronously from the SNS notification" is incorrect as you cannot configure a CloudWatch alarm that notifies based on item lifecycle events. It is better to use DynamoDB streams and integrate Lambda.

**INCORRECT:** "Configure an Amazon CloudTrail API alarm that sends a message to an Amazon SQS queue. Configure the Lambda function to poll the queue and invoke the function synchronously" is incorrect. There is no such alarm that notifies from Amazon CloudTrail relating to item lifecycle events.

**References:**

<https://docs.aws.amazon.com/amazondynamodb/latest/developerguide/Streams.Lambda.html>

**Save time with our AWS cheat sheets:**

<https://digitalcloud.training/amazon-dynamodb/>

**Domain**

AWS Database

**Question 37Correct**

An engineer is constructing a web-based application that uses Amazon DynamoDB for storing data. The data is distributed across two tables: 'authors' and 'books'. The 'authors' table uses 'authorName' as its partition key, while the 'books' table has 'bookTitle' as the partition key and 'authorName' as the sort key.

The application requires the ability to fetch multiple books and authors simultaneously in a single database operation for effective performance. The engineer is seeking a solution that maximizes application efficiency and reduces network traffic.

What strategy should the engineer employ to achieve these requirements?

**First query the 'books' table using 'bookTitle' as a key condition, then separately query the 'authors' table using 'authorName'.**

**Execute individual GetItem operations for each book and author to be retrieved.**

**Use a DynamoDB Scan operation to fetch the required items from both tables.**

**Your answer is correct**

**Utilize the DynamoDB BatchGetItem operation to fetch multiple items from both tables in a single network round trip.**

Overall explanation

The BatchGetItem operation in DynamoDB permits the retrieval of multiple items from one or more tables in a single operation, thereby minimizing network traffic and improving overall application performance.

**CORRECT:**"Utilize the DynamoDB BatchGetItem operation to fetch multiple items from both tables in a single network round trip" is the correct answer (as explained above.)

**INCORRECT:** "Execute individual GetItem operations for each book and author to be retrieved" is incorrect.

Executing individual GetItem operations for each book and author would increase network traffic and slow down application performance.

**INCORRECT:** "Use a DynamoDB Scan operation to fetch the required items from both tables" is incorrect.

A DynamoDB Scan operation retrieves every item in the table and then filters to provide the required result, making it inefficient in this context.

**INCORRECT:** "First query the 'books' table using 'bookTitle' as a key condition, then separately query the 'authors' table using 'authorName'" is incorrect.

Querying the 'books' and 'authors' tables separately would increase the number of network operations and negatively affect the application's performance.

**References:**

<https://docs.aws.amazon.com/amazondynamodb/latest/APIReference/API_BatchGetItem.html>

**Save time with our AWS cheat sheets:**

<https://digitalcloud.training/amazon-dynamodb/>

**Domain**

AWS Database

**Question 38Correct**

A company is creating a serverless application that uses AWS Lambda functions. The developer has written the code to initialize the AWS SDK outside of the Lambda handler function.

What is PRIMARY benefit of this action?

**Creates a new SDK instance for each invocation.**

**Improves readability and reduces complexity.**

**It minimizes the deployment package size.**

**Your answer is correct**

**Takes advantage of execution environment reuse.**

Overall explanation

You should initialize SDK clients and database connections outside of the function handler, and cache static assets locally in the /tmp directory. Subsequent invocations processed by the same instance of your function can reuse these resources. This saves cost by reducing function run time.

The primary benefit of this technique is to take advantage of execution environment reuse to improve the performance of your function.

**CORRECT:**"Takes advantage of execution environment reuse" is the correct answer (as explained above.)

**INCORRECT:** "Creates a new SDK instance for each invocation" is incorrect.

This is the opposite of what we are trying to achieve here.

**INCORRECT:** "It minimizes the deployment package size" is incorrect.

This technique does not affect the deployment package size.

**INCORRECT:** "Improves readability and reduces complexity" is incorrect.

It may improve readability but that is debatable. This is not the primary reason you would use this technique.

**References:**

<https://docs.aws.amazon.com/lambda/latest/dg/best-practices.html>

**Save time with our AWS cheat sheets:**

<https://digitalcloud.training/aws-lambda/>

**Domain**

AWS Compute

**Question 39Incorrect**

An application will use AWS Lambda and an Amazon RDS database. The Developer needs to secure the database connection string and enable automatic rotation every 30 days. What is the SIMPLEST way to achieve this requirement?

**Your answer is incorrect**

**Store a SecureString in Systems Manager Parameter Store and enable automatic rotation every 30 days**

**Store the connection string as an encrypted environment variable in Lambda and create a separate function that rotates the connection string every 30 days**

**Store the connection string in an encrypted Amazon S3 bucket and use a scheduled CloudWatch Event to update the connection string every 30 days**

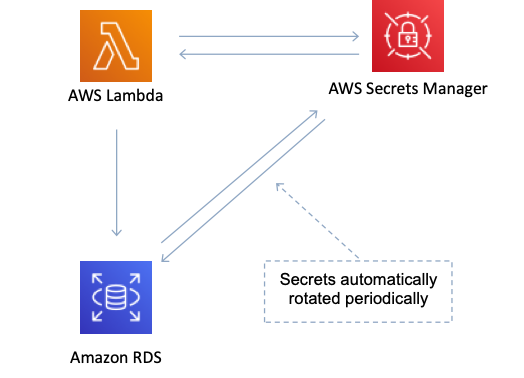
**Correct answer**

**Store a secret in AWS Secrets Manager and enable automatic rotation every 30 days**

Overall explanation

AWS Secrets Manager encrypts secrets at rest using encryption keys that you own and store in AWS Key Management Service (KMS). When you retrieve a secret, Secrets Manager decrypts the secret and transmits it securely over TLS to your local environment.

With AWS Secrets Manager, you can rotate secrets on a schedule or on demand by using the Secrets Manager console, AWS SDK, or AWS CLI.



For example, to rotate a database password, you provide the database type, rotation frequency, and master database credentials when storing the password in Secrets Manager. Secrets Manager natively supports rotating credentials for databases hosted on Amazon RDS and Amazon DocumentDB and clusters hosted on Amazon Redshift.

**CORRECT:**"Store a secret in AWS Secrets Manager and enable automatic rotation every 30 days" is the correct answer.

**INCORRECT:** "Store a SecureString in Systems Manager Parameter Store and enable automatic rotation every 30 days" is incorrect as SSM Parameter Store does not support automatic key rotation.

**INCORRECT:** "Store the connection string as an encrypted environment variable in Lambda and create a separate function that rotates the connection string every 30 days" is incorrect as this is not the simplest solution. In this scenario using AWS Secrets Manager would be easier to implement as it provides native features for rotating the secret.

**INCORRECT:** "Store the connection string in an encrypted Amazon S3 bucket and use a scheduled CloudWatch Event to update the connection string every 30 days" is incorrect. There is no native capability of CloudWatch to update connection strings so you would need some other service such as a Lambda function to execute and rotate the connection string which is missing from this answer.

**References:**

<https://aws.amazon.com/secrets-manager/features/>

**Domain**

AWS Security, Identity, & Compliance

**Question 40Incorrect**

A company wants a serverless solution for phased release of static websites hosted on various version control systems. Deployments should be triggered by Git branch merges and all data exchange should be over HTTPS.

Which option offers the LOWEST operational overhead?

**Deploy websites on separate Amazon EC2 instances for each environment, use AWS CodeDeploy for automation, and link it with the version control systems.**

**Your answer is incorrect**

**Use AWS Elastic Beanstalk for hosting and use AWS CodeStar to manage deployments and workflows.**

**Use Amazon S3 to host the websites, create a manual script to deploy changes when there are code merges in the version control systems.**

**Correct answer**

**Use AWS Amplify for hosting, connect corresponding repository branches, and initiate deployments by merging changes to the needed branch.**

Overall explanation

AWS Amplify is designed to host static websites with continuous deployment linked to Git repositories. It allows you to connect branches in your repository with environments in Amplify, and to automatically deploy changes when you merge code to those branches. This solution requires the least operational overhead among the options.

**CORRECT:**"Use AWS Amplify for hosting, connect corresponding repository branches, and initiate deployments by merging changes to the needed branch" is the correct answer (as explained above.)

**INCORRECT:** "Deploy websites on separate Amazon EC2 instances for each environment, use AWS CodeDeploy for automation, and link it with the version control systems" is incorrect.

While Amazon EC2 could host the websites and AWS CodeDeploy could manage deployments, this is not a serverless solution as requested. The EC2 instances would need to run continuously, creating more operational overhead.

**INCORRECT:** "Use Amazon S3 to host the websites, create a manual script to deploy changes when there are code merges in the version control systems" is incorrect.

While Amazon S3 can host static websites, creating a manual script to deploy changes adds unnecessary complexity and operational overhead compared to a service like Amplify that automates this process.

**INCORRECT:** "Use AWS Elastic Beanstalk for hosting and use AWS CodeStar to manage deployments and workflows" is incorrect.

AWS Elastic Beanstalk is typically used for dynamic, multi-tier web applications, and AWS CodeStar provides a unified user interface, enabling you to easily manage your software development activities in one place. However, this combination could be more complex and require more operational overhead than AWS Amplify for a static website.

**References:**

<https://docs.aws.amazon.com/amplify/latest/userguide/welcome.html>

**Domain**

AWS Developer Tools

**Question 41Incorrect**

A Developer is designing a fault-tolerant environment where client sessions will be saved. How can the Developer ensure that no sessions are lost if an Amazon EC2 instance fails?

**Use Elastic Load Balancer connection draining to stop sending requests to failing instances**

**Correct answer**

**Use Amazon DynamoDB to perform scalable session handling**

**Use Amazon SQS to save session data**

**Your answer is incorrect**

**Use sticky sessions with an Elastic Load Balancer target group**

Overall explanation

The **DynamoDB Session Handler** is a custom session handler for PHP that allows developers to use Amazon DynamoDB as a session store. Using DynamoDB for session storage alleviates issues that occur with session handling in a distributed web application by moving sessions off of the local file system and into a shared location. DynamoDB is fast, scalable, easy to setup, and handles replication of your data automatically.

**CORRECT:**"Use Amazon DynamoDB to perform scalable session handling" is the correct answer.

**INCORRECT:** "Use sticky sessions with an Elastic Load Balancer target group" is incorrect as this involves maintaining session state data on the EC2 instances which means that data is lost if an instance fails.

**INCORRECT:** "Use Amazon SQS to save session data" is incorrect as SQS is not used for session data, it is used for application component decoupling.

**INCORRECT:** "Use Elastic Load Balancer connection draining to stop sending requests to failing instances" is incorrect as this does not solve the problem of ensuring the session data is available, the data will be on the failing instance and will be lost.

**References:**

<https://docs.aws.amazon.com/aws-sdk-php/v2/guide/feature-dynamodb-session-handler.html>

**Save time with our AWS cheat sheets:**

<https://digitalcloud.training/amazon-dynamodb/>

**Domain**

AWS Database

**Question 42Incorrect**

A company is running a web application on Amazon EC2 behind an Elastic Load Balancer (ELB). The company is concerned about the security of the web application and would like to secure the application with SSL certificates. The solution should not have any performance impact on the EC2 instances.

What steps should be taken to secure the web application? (Select TWO.)

**Your selection is correct**

**Add an SSL certificate to the Elastic Load Balancer**

**Your selection is incorrect**

**Configure Server-Side Encryption with KMS managed keys**

**Install SSL certificates on the EC2 instances**

**Configure the Elastic Load Balancer with SSL passthrough**

**Correct selection**

**Configure the Elastic Load Balancer for SSL termination**

Overall explanation

The requirements clearly state that we cannot impact the performance of the EC2 instances at all. Therefore, we will not be able to add certificates to the EC2 instances as that would place a burden on the CPU when encrypting and decrypting data.

We are therefore left with configuring SSL on the Elastic Load Balancer itself. For this we need to add an SSL certificate to the ELB and then configure the ELB for SSL termination.

You can create an HTTPS listener, which uses encrypted connections (also known as *SSL offload*). This feature enables traffic encryption between your load balancer and the clients that initiate SSL or TLS sessions.

To use an HTTPS listener, you must deploy at least one SSL/TLS server certificate on your load balancer. The load balancer uses a server certificate to terminate the front-end connection and then decrypt requests from clients before sending them to the targets.

This is the most secure solution we can created without adding any performance impact to the EC2 instances.

**CORRECT:**"Add an SSL certificate to the Elastic Load Balancer" is a correct answer.

**CORRECT:**"Configure the Elastic Load Balancer for SSL termination" is also a correct answer.

**INCORRECT:** "Configure the Elastic Load Balancer with SSL passthrough" is incorrect as this would be used to forward encrypted packets directly to the EC2 instance for termination but we do not want to add SSL certificates to the EC2 instances due to the extra processing required.

**INCORRECT:** "Install SSL certificates on the EC2 instances" is incorrect as we do not want to add SSL certificates to the EC2 instances due to the extra processing required.

**INCORRECT:** "Configure Server-Side Encryption with KMS managed keys" is incorrect as this applies to Amazon S3, not ELB.

**References:**

<https://docs.aws.amazon.com/elasticloadbalancing/latest/application/create-https-listener.html>

**Save time with our AWS cheat sheets:**

<https://digitalcloud.training/aws-elastic-load-balancing-aws-elb/>

**Domain**

AWS Compute

**Question 43Incorrect**

To include objects defined by the AWS Serverless Application Model (SAM) in an AWS CloudFormation template, in addition to Resources, what section MUST be included in the document root?

**Conditions**

**Globals**

**Correct answer**

**Transform**

**Your answer is incorrect**

**Properties**

Overall explanation

The primary differences between AWS SAM templates and AWS CloudFormation templates are the following:

• **Transform declaration.** The declaration Transform: AWS::Serverless-2016-10-31 is required for AWS SAM templates. This declaration identifies an AWS CloudFormation template as an AWS SAM template.

• **Globals section.**The Globals section is unique to AWS SAM. It defines properties that are common to all your serverless functions and APIs. All the AWS::Serverless::Function, AWS::Serverless::Api, and AWS::Serverless::SimpleTable resources inherit the properties that are defined in the Globals section.

• **Resources section.**In AWS SAM templates the Resources section can contain a combination of AWS CloudFormation resources and AWS SAM resources.

Of these three sections, only the Transform section and Resources sections are required; the Globals section is optional.

**CORRECT:**"Transform" is the correct answer.

**INCORRECT:** "Globals" is incorrect as this is not a required section.

**INCORRECT:** "Conditions" is incorrect as this is an optional section.

**INCORRECT:** "Properties" is incorrect as this is not a section in a template, it is used within a resource.

**References:**

<https://docs.aws.amazon.com/serverless-application-model/latest/developerguide/sam-specification-template-anatomy.html>

**Save time with our AWS cheat sheets:**

<https://digitalcloud.training/aws-sam/>

**Domain**

AWS Compute

**Question 44Correct**

A legacy service has an XML-based SOAP interface. The Developer wants to expose the functionality of the service to external clients with the Amazon API Gateway. Which technique will accomplish this?

**Create a RESTful API with the API Gateway; transform the incoming XML into a valid message for the SOAP interface using mapping templates**

**Create a RESTful API with the API Gateway; pass the incoming JSON to the SOAP interface through an Application Load Balancer**

**Your answer is correct**

**Create a RESTful API with the API Gateway; transform the incoming JSON into a valid XML message for the SOAP interface using mapping templates**

**Create a RESTful API with the API Gateway; pass the incoming XML to the SOAP interface through an Application Load Balancer**

Overall explanation

Amazon API Gateway is an AWS service for creating, publishing, maintaining, monitoring, and securing REST and WebSocket APIs at any scale. API developers can create APIs that access AWS or other web services as well as data stored in the AWS Cloud.

In API Gateway, an API's method request can take a payload in a different format from the corresponding integration request payload, as required in the backend. Similarly, the backend may return an integration response payload different from the method response payload, as expected by the frontend.

API Gateway lets you use mapping templates to map the payload from a method request to the corresponding integration request and from an integration response to the corresponding method response.

If an existing legacy service returns XML-style data, you can use the API Gateway to transform the output to JSON as part of your modernization effort. The API Gateway can be configured to transform the output of legacy services from XML to JSON, allowing them to make a move that is seamless and non-disruptive. The transformation is specified using JSON-Schema.

Therefore, the technique the Developer should use is to create a RESTful API with the API Gateway and transform the incoming JSON into a valid XML message for the SOAP interface using mapping templates.

**CORRECT:**"Create a RESTful API with the API Gateway; transform the incoming JSON into a valid XML message for the SOAP interface using mapping templates" is the correct answer.

**INCORRECT:** "Create a RESTful API with the API Gateway; pass the incoming JSON to the SOAP interface through an Application Load Balancer" is incorrect as we don’t need an ALB to do this, we can use a mapping template within the API Gateway which will be more cost-efficient.

**INCORRECT:** "Create a RESTful API with the API Gateway; pass the incoming XML to the SOAP interface through an Application Load Balancer" is incorrect as the incoming data will be JSON, not XML as the Developer needs to publish a modern application interface. A mapping template should also be used in place of the ALB.

**INCORRECT:** "Create a RESTful API with the API Gateway; transform the incoming XML into a valid message for the SOAP interface using mapping templates" is incorrect as the incoming data will be JSON, not XML as the Developer needs to publish a modern application interface.

**References:**

<https://docs.aws.amazon.com/apigateway/latest/developerguide/models-mappings.html>

**Save time with our AWS cheat sheets:**

<https://digitalcloud.training/amazon-api-gateway/>

**Domain**

AWS Networking & Content Delivery

**Question 45Incorrect**

A financial application is hosted on an Auto Scaling group of EC2 instance with an Elastic Load Balancer. A Developer needs to capture information about the IP traffic going to and from network interfaces in the VPC.

How can the Developer capture this information?

**Capture the information using a Network ACL**

**Your answer is incorrect**

**Capture the information directly into Amazon CloudWatch Logs**

**Create a flow log in the VPC and publish data to Amazon CloudTrail**

**Correct answer**

**Create a flow log in the VPC and publish data to Amazon S3**

Overall explanation

VPC Flow Logs is a feature that enables you to capture information about the IP traffic going to and from network interfaces in your VPC. Flow log data can be published to Amazon CloudWatch Logs or Amazon S3. After you've created a flow log, you can retrieve and view its data in the chosen destination.

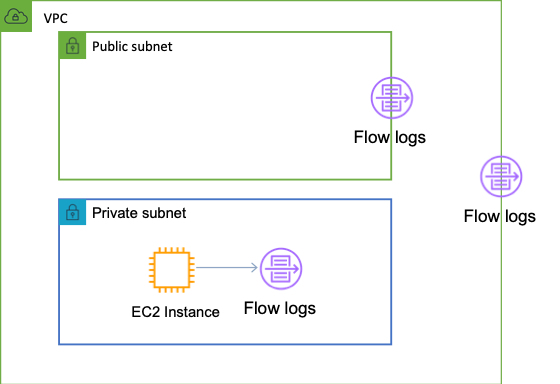
Flow logs can help you with a number of tasks, such as:

• Diagnosing overly restrictive security group rules

• Monitoring the traffic that is reaching your instance

• Determining the direction of the traffic to and from the network interfaces

As you can see in the image below, you can create a flow log for a VPC, a subnet, or a network interface. If you create a flow log for a subnet or VPC, each network interface in that subnet or VPC is monitored.



Therefore, the Developer should create a flow log in the VPC and publish data to Amazon S3. The Developer could also choose CloudWatch Logs as a destination for publishing the data, but this is not presented as an option.

**CORRECT:**"Create a flow log in the VPC and publish data to Amazon S3" is the correct answer.

**INCORRECT:** "Capture the information directly into Amazon CloudWatch Logs" is incorrect as you cannot capture this information directly into CloudWatch Logs. You would need to capture with a flow log and then publish to CloudWatch Logs.

**INCORRECT:** "Capture the information using a Network ACL" is incorrect as you cannot capture data using a Network ACL as it is a subnet-level firewall.

**INCORRECT:** "Create a flow log in the VPC and publish data to Amazon CloudTrail" is incorrect as you cannot publish data from a flow log to CloudTrail. Amazon CloudTrail captures information about API calls.

**References:**

<https://docs.aws.amazon.com/vpc/latest/userguide/flow-logs.html>

**Save time with our AWS cheat sheets:**

<https://digitalcloud.training/amazon-vpc/>

**Domain**

AWS Networking & Content Delivery

**Question 46Correct**

A company is releasing an updated version of its APIs for its new mobile application, which uses Amazon API Gateway. The developers aim to gradually and seamlessly roll out the new version of APIs.

What is the MOST straightforward method for them to introduce the new API version to a subset of users through API Gateway?

**Use an Amazon Route 53 failover routing policy to divert a certain percentage of traffic to the updated API version.**

**Your answer is correct**

**Utilize the canary release deployment feature in API Gateway. Configure the canarySettings to redirect a portion of the API traffic.**

**Deploy the new API in a separate VPC and use Amazon CloudFront to distribute the API traffic between the old and new versions.**

**Develop a custom Lambda function to control the API traffic distribution between the two API versions.**

Overall explanation

The canary release deployment in API Gateway enables developers to roll out API changes gradually. By configuring the canarySettings, a percentage of the API traffic can be redirected to the new version, allowing for cautious and controlled rollout.

**CORRECT:**"Utilize the canary release deployment feature in API Gateway. Configure the canarySettings to redirect a portion of the API traffic" is the correct answer (as explained above.)

**INCORRECT:** "Use an Amazon Route 53 failover routing policy to divert a certain percentage of traffic to the updated API version" is incorrect.

Amazon Route 53 failover routing policy is used for routing internet traffic to a resource when the primary resource becomes unavailable. It is not a suitable solution for this scenario.

**INCORRECT:** "Deploy the new API in a separate VPC and use Amazon CloudFront to distribute the API traffic between the old and new versions" is incorrect.

Deploying the new API in a separate VPC and using Amazon CloudFront for traffic distribution could work, but this approach is more complex and does not provide an out-of-the-box traffic control mechanism like canary releases.

**INCORRECT:** "Develop a custom Lambda function to control the API traffic distribution between the two API versions" is incorrect.

Developing a custom Lambda function to control API traffic distribution can be overly complicated and might not provide the desired level of control, especially when compared to built-in solutions like canary releases.

**References:**

<https://docs.aws.amazon.com/apigateway/latest/developerguide/canary-release.html>

**Save time with our AWS cheat sheets:**

<https://digitalcloud.training/amazon-api-gateway/>

**Domain**

AWS Networking & Content Delivery

**Question 47Correct**

A company is deploying a new serverless application with an AWS Lambda function. A developer ran some test invocations using the AWS CLI. The function is invoking correctly and returning a success message, but not log data is being generated in Amazon CloudWatch Logs. The developer waited for 15 minutes but the log data is not showing up.

What is the most likely explanation for this issue?

**The Lambda function does not have any explicit log statements for the log data to send it to CloudWatch Logs.**

**Your answer is correct**

**The function execution role does not have permission to write log data to CloudWatch Logs.**

**The function configuration does not have CloudWatch Logs configured as a success destination.**

**A log group and log stream has not been configured for the function in CloudWatch Logs.**

Overall explanation

AWS Lambda automatically monitors Lambda functions on your behalf, reporting metrics through Amazon CloudWatch. To help you troubleshoot failures in a function, after you set up permissions, Lambda logs all requests handled by your function and automatically stores logs generated by your code through Amazon CloudWatch Logs.

You can insert logging statements into your code to help you validate that your code is working as expected. Lambda automatically integrates with CloudWatch Logs and pushes all logs from your code to a CloudWatch Logs group associated with a Lambda function, which is named /aws/lambda/*<function name>*.

It can take 5-10 minutes for logs to show up after a function invocation. If your Lambda function code is executing, but you don't see any log data being generated after several minutes, this could mean that your execution role for the Lambda function didn't grant permissions to write log data to CloudWatch Logs.

**CORRECT:**"The function execution role does not have permission to write log data to CloudWatch Logs" is the correct answer (as explained above.)

**INCORRECT:** "The Lambda function does not have any explicit log statements for the log data to send it to CloudWatch Logs" is incorrect.

You do need to have logging statements in your code to send meaningful data to CloudWatch Logs. However, the most likely cause of having nothing show up is that the permissions were not assigned.

**INCORRECT:** "The function configuration does not have CloudWatch Logs configured as a success destination" is incorrect.

CloudWatch Logs is not configured as a destination in a Lambda function.

**INCORRECT:** "A log group and log stream has not been configured for the function in CloudWatch Logs" is incorrect.

The log group and log stream are automatically created as long as permissions are assigned.

**References:**

<https://docs.aws.amazon.com/lambda/latest/dg/lambda-monitoring.html>

**Save time with our AWS cheat sheets:**

<https://digitalcloud.training/aws-lambda/>

**Domain**

AWS Management & Governance

**Question 48Incorrect**

AWS CodeBuild builds code for an application, creates a Docker image, pushes the image to Amazon Elastic Container Registry (ECR), and tags the image with a unique identifier.

If the Developers already have AWS CLI configured on their workstations, how can the Docker images be pulled to the workstations?

**Correct answer**

**Run the output of the following: aws ecr get-login-password, and then run: docker pull REPOSITORY URI : TAG**

**Run the output of the following: aws ecr get-download-url-for-layer, and then run: docker pull REPOSITORY URI : TAG**

**Run the following: docker pull REPOSITORY URI : TAG**

**Your answer is incorrect**

**Run the following: aws ecr get-login-password, and then run: docker pull REPOSITORY URI : TAG**

Overall explanation

If you would like to run a Docker image that is available in Amazon ECR, you can pull it to your local environment with the docker pull command. You can do this from either your default registry or from a registry associated with another AWS account.

Docker CLI does not support standard AWS authentication methods, so client authentication must be handled so that ECR knows who is requesting to push or pull an image. To do this you can issue the aws ecr get-login-password AWS CLI command and then use the output to login using docker login and then issue a docker pull command specifying the image name using registry/repository[:tag].

**CORRECT:**"Run the output of the following: aws ecr get-login-password, and then run docker pull REPOSITORY URI : TAG" is the correct answer.

**INCORRECT:** "Run the following: docker pull REPOSITORY URI : TAG" is incorrect as the Developers first need to authenticate before they can pull the image.

**INCORRECT:** "Run the following: aws ecr get-login-password, and then run: docker pull REPOSITORY URI : TAG" is incorrect. The Developers need to not just run the login command but run the output of the login command which contains the authentication token required to log in.

**INCORRECT:** "Run the output of the following: aws ecr get-download-url-for-layer, and then run docker pull REPOSITORY URI : TAG" is incorrect as this command retrieves a pre-signed Amazon S3 download URL corresponding to an image layer.

**References:**

<https://docs.aws.amazon.com/AmazonECR/latest/userguide/Registries.html#registry_auth>

<https://docs.aws.amazon.com/AmazonECR/latest/userguide/docker-pull-ecr-image.html>

**Save time with our AWS cheat sheets:**

<https://digitalcloud.training/amazon-ecs-and-eks/>

**Domain**

AWS Compute

**Question 49Incorrect**

A customer requires a schema-less, key/value database that can be used for storing customer orders. Which type of AWS database is BEST suited to this requirement?

**Amazon S3**

**Amazon RDS**

**Your answer is incorrect**

**Amazon ElastiCache**

**Correct answer**

**Amazon DynamoDB**

Overall explanation

Amazon DynamoDB is a fully managed NoSQL database service that provides fast and predictable performance with seamless scalability. It is a non-relational (schema-less), key-value type of database. This is the most suitable solution for this requirement.

**CORRECT:**"Amazon DynamoDB" is the correct answer.

**INCORRECT:** "Amazon RDS" is incorrect as this a relational database that has a schema.

**INCORRECT:** "Amazon ElastiCache" is incorrect as this is a key/value database but it is used to cache the contents of other databases (including DynamoDB and RDS) for better performance for reads.

**INCORRECT:** "Amazon S3" is incorrect as this is an object-based storage system not a database. It is a key/value store but DynamoDB is a better fit for a customer order database.

**References:**

<https://docs.aws.amazon.com/amazondynamodb/latest/developerguide/Introduction.html>

**Save time with our AWS cheat sheets:**

<https://digitalcloud.training/amazon-dynamodb/>

**Domain**

AWS Database

**Question 50Incorrect**

A Developer needs to create an instance profile for an Amazon EC2 instance using the AWS CLI. How can this be achieved? (Select THREE.)

**Your selection is correct**

**Run the aws iam add-role-to-instance-profile command**

**Your selection is correct**

**Run the aws iam create-instance-profile command**

**Your selection is incorrect**

**Run the CreateInstanceProfile API**

**Run the AssignInstanceProfile API**

**Run the AddRoleToInstanceProfile API**

**Correct selection**

**Run the aws ec2 associate-instance-profile command**

Overall explanation

To add a role to an Amazon EC2 instance using the AWS CLI you must first create an instance profile. Then you need to add the role to the instance profile and finally assign the instance profile to the Amazon EC2 instance.

The following example commands would achieve this outcome:

1. aws iam create-instance-profile --instance-profile-name EXAMPLEPROFILENAME
2. aws iam add-role-to-instance-profile --instance-profile-name EXAMPLEPROFILENAME --role-name EXAMPLEROLENAME
3. aws ec2 associate-iam-instance-profile --iam-instance-profile Name=EXAMPLEPROFILENAME --instance-id i-012345678910abcde

**CORRECT:**"Run the aws iam create-instance-profile command" is a correct answer.

**CORRECT:**"Run the aws iam add-role-to-instance-profile command" is a correct answer.

**CORRECT:**"Run the aws ec2 associate-instance-profile command" is a correct answer.

**INCORRECT:** "Run the CreateInstanceProfile API" is incorrect as this is an API action, not an AWS CLI command.

**INCORRECT:** "Run the AddRoleToInstanceProfile API" is incorrect as this is an API action, not an AWS CLI command.

**INCORRECT:** "Run the AssignInstanceProfile API" is incorrect as this is an API action, not an AWS CLI command.

**References:**

<https://aws.amazon.com/premiumsupport/knowledge-center/attach-replace-ec2-instance-profile/>

**Save time with our AWS cheat sheets:**

<https://digitalcloud.training/aws-iam/>

**Domain**

AWS Compute

**Question 51Correct**

An application writes items to an Amazon DynamoDB table. As the application scales to thousands of instances, calls to the DynamoDB API generate occasional ThrottlingException errors. The application is coded in a language incompatible with the AWS SDK.

How should the error be handled?

**Pass API calls through Amazon API Gateway**

**Use Amazon SQS as an API message bus**

**Your answer is correct**

**Add exponential backoff to the application logic**

**Send the items to DynamoDB through Amazon Kinesis Data Firehose**

Overall explanation

Exponential backoff can improve an application's reliability by using progressively longer waits between retries. When using the AWS SDK, this logic is built‑in. However, in this case the application is incompatible with the AWS SDK so it is necessary to manually implement exponential backoff.

**CORRECT:**"Add exponential backoff to the application logic" is the correct answer.

**INCORRECT:** "Use Amazon SQS as an API message bus" is incorrect as SQS requires instances or functions to pick up and process the messages and put them in the DynamoDB table. This is unnecessary cost and complexity and will not improve performance.

**INCORRECT:** "Pass API calls through Amazon API Gateway" is incorrect as this is not a suitable method of throttling the application. Exponential backoff logic in the application is a better solution.

**INCORRECT:** "Send the items to DynamoDB through Amazon Kinesis Data Firehose" is incorrect as DynamoDB is not a destination for Kinesis Data Firehose.

**References:**

<https://aws.amazon.com/premiumsupport/knowledge-center/dynamodb-table-throttled/>

**Save time with our AWS cheat sheets:**

<https://digitalcloud.training/amazon-dynamodb/>

**Domain**

AWS Database

**Question 52Correct**

The following permissions policy is applied to an IAM user account:

1. {
2. "Version": "2012-10-17",
3. "Statement": [{
4. "Effect": "Allow",
5. "Action": "sqs:\*",
6. "Resource": "arn:aws:sqs:\*:513246782345:staging-queue\*"
7. }]
8. }

Due to this policy, what Amazon SQS actions will the user be able to perform?

**Your answer is correct**

**The user will be able to use all Amazon SQS actions, but only for queues with names begin with the string “staging-queue“**

**The user will be able to create a queue named “staging-queue“**

**The user will be able to apply a resource-based policy to the Amazon SQS queue named “staging-queue”**

**The user will be granted cross-account access from account number “513246782345” to queue “staging-queue”**

Overall explanation

The policy allows the user to use all Amazon SQS actions, but only with queues whose names are prefixed with the literal string “staging-queue”. This policy is useful to provide a queue creator the ability to use Amazon SQS actions. Any user who has permissions to create a queue must also have permissions to use other Amazon SQS actions in order to do anything with the created queues.

**CORRECT:**"The user will be able to use all Amazon SQS actions, but only for queues with names begin with the string “staging-queue“" is the correct answer.

**INCORRECT:** "The user will be able to create a queue named “staging-queue“" is incorrect as this policy provides the permissions to perform SQS actions on an existing queue.

**INCORRECT:** "The user will be able to apply a resource-based policy to the Amazon SQS queue named “staging-queue”" is incorrect as this is a single operation and the permissions policy allows all SQS actions.

**INCORRECT:** "The user will be granted cross-account access from account number “513246782345” to queue “staging-queue”" is incorrect as this is not a policy for granting cross-account access. The account number and queue relate to the same account.

**References:**

<https://docs.aws.amazon.com/AWSSimpleQueueService/latest/SQSDeveloperGuide/sqs-overview-of-managing-access.html>

**Save time with our AWS cheat sheets:**

<https://digitalcloud.training/aws-application-integration-services/>

**Domain**

AWS Application Integration

**Question 53Incorrect**

A Developer is creating a DynamoDB table for storing application logs. The table has 5 write capacity units (WCUs). The Developer needs to configure the read capacity units (RCUs) for the table. Which of the following configurations represents the most efficient use of throughput?

**Strongly consistent reads of 5 RCUs reading items that are 4 KB in size**

**Correct answer**

**Eventually consistent reads of 15 RCUs reading items that are 1 KB in size**

**Strongly consistent reads of 15 RCUs reading items that are 1KB in size**

**Your answer is incorrect**

**Eventually consistent reads of 5 RCUs reading items that are 4 KB in size**

Overall explanation

Among these options, the most cost-efficient in terms of maximizing the number of read operations per RCU is:

· **Eventually consistent reads of 15 RCUs reading items that are 1 KB in size.**

This choice utilizes the RCU allocation most effectively because:

· It leverages the lower RCU consumption of eventually consistent reads.

· It takes advantage of the smaller item size (1 KB), which requires fewer RCUs for the same number of operations compared to items up to 4 KB.

**CORRECT:**"Strongly consistent reads of 15 RCUs reading items that are 1KB in size" is correct as described above.

**INCORRECT:** "Eventually consistent reads of 15 RCUs reading items that are 1 KB in size" is incorrect as described above.

**INCORRECT:** "Strongly consistent reads of 5 RCUs reading items that are 4 KB in size" is incorrect as described above.

**INCORRECT:** "Eventually consistent reads of 5 RCUs reading items that are 4 KB in size" is incorrect as described above.

**References:**

<https://docs.amazonaws.cn/en_us/amazondynamodb/latest/developerguide/ProvisionedThroughput.html>

**Save time with our AWS cheat sheets:**

<https://digitalcloud.training/amazon-dynamodb/>

**Domain**

AWS Database

**Question 54Correct**

A Development team are creating a new REST API that uses Amazon API Gateway and AWS Lambda. To support testing there need to be different versions of the service. What is the BEST way to provide multiple versions of the REST API?

**Create an AWS Lambda authorizer to route API clients to the correct API version**

**Create an API Gateway resource policy to isolate versions and provide context to the Lambda functions**

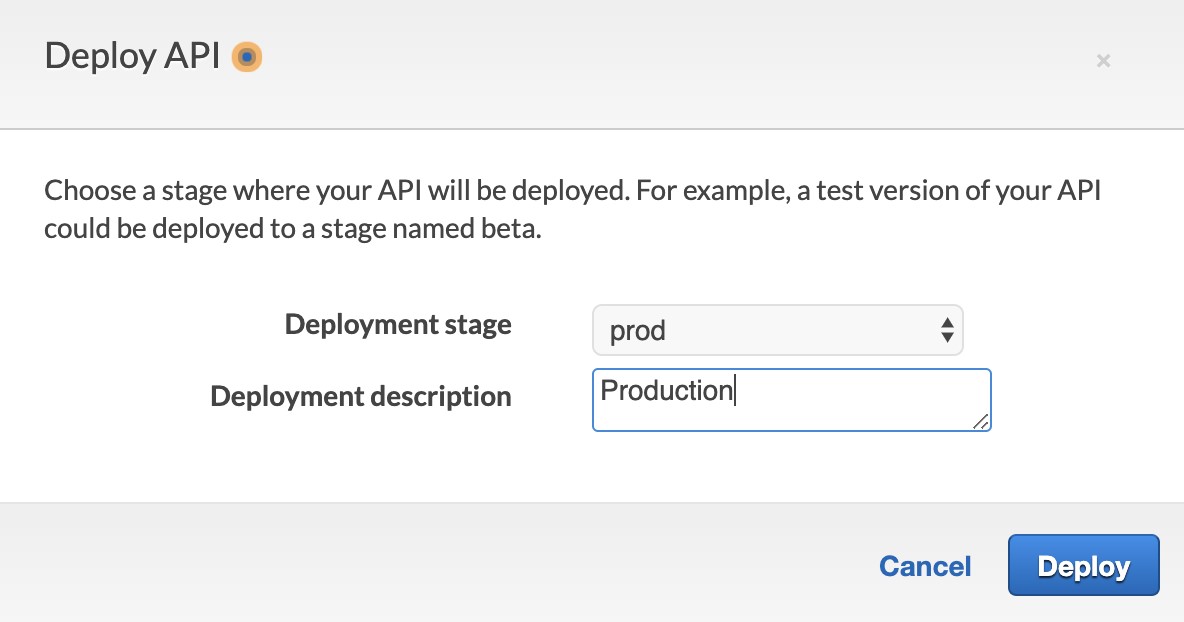
**Deploy an HTTP Proxy integration and configure the proxy with API versions**

**Your answer is correct**

**Deploy the API versions as unique stages with unique endpoints and use stage variables to provide further context**

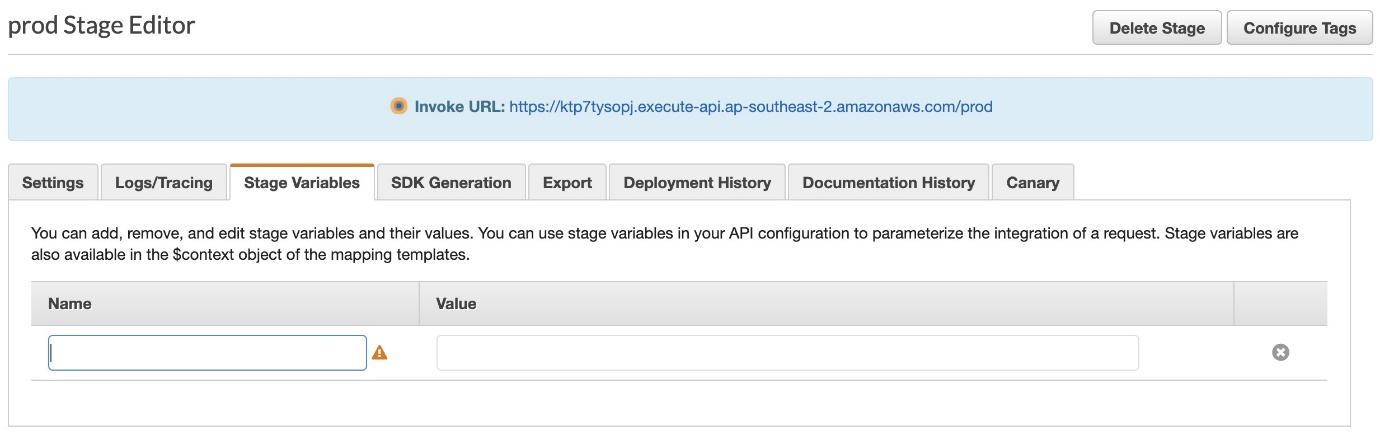
Overall explanation

A stage is a named reference to a deployment, which is a snapshot of the API. You use a Stage to manage and optimize a particular deployment. For example, you can set up stage settings to enable caching, customize request throttling, configure logging, define stage variables or attach a canary release for testing. APIs are deployed to stages:



Stage variables are name-value pairs that you can define as configuration attributes associated with a deployment stage of a REST API. They act like environment variables and can be used in your API setup and mapping templates.

With deployment stages in API Gateway, you can manage multiple release stages for each API, such as alpha, beta, and production. Using stage variables you can configure an API deployment stage to interact with different backend endpoints. For example, your API can pass a GET request as an HTTP proxy to the backend web host (for example, [http://example.com](http://example.com/)).



In this case, the backend web host is configured in a stage variable so that when developers call your production endpoint, API Gateway calls example.com. When you call your beta endpoint, API Gateway uses the value configured in the stage variable for the beta stage, and calls a different web host (for example, beta.example.com). Similarly, stage variables can be used to specify a different AWS Lambda function name for each stage in your API.

Therefore, for this scenario the Developers can deploy the API versions as unique stages with unique endpoints and use stage variables to provide further context such as connections to different backend services.

**CORRECT:**"Deploy the API versions as unique stages with unique endpoints and use stage variables to provide further context" is the correct answer.

**INCORRECT:** "Create an API Gateway resource policy to isolate versions and provide context to the Lambda functions" is incorrect. API Gateway [resource policies](https://docs.aws.amazon.com/apigateway/latest/developerguide/apigateway-resource-policies.html) are JSON policy documents that you attach to an API to control whether a specified principal (typically, an IAM user or role) can invoke the API.

**INCORRECT:** "Create an AWS Lambda authorizer to route API clients to the correct API version" is incorrect. A *Lambda authorizer* (formerly known as a *custom authorizer*) is an API Gateway feature that uses a Lambda function to control access to your API. This is not used for routing API clients to different versions.

**INCORRECT:** "Deploy an HTTP Proxy integration and configure the proxy with API versions" is incorrect. The HTTP proxy integration allows a client to access the backend HTTP endpoints with a streamlined integration setup on a single API method. This is not used for providing multiple versions of the API, use stages and stage variables instead.

**References:**

<https://docs.aws.amazon.com/apigateway/latest/developerguide/set-up-stages.html>

<https://docs.aws.amazon.com/apigateway/latest/developerguide/stage-variables.html>

**Save time with our AWS cheat sheets:**

<https://digitalcloud.training/amazon-api-gateway/>

**Domain**

AWS Networking & Content Delivery

**Question 55Correct**

An application runs on a fleet of Amazon EC2 instances and stores data in a Microsoft SQL Server database hosted on Amazon RDS. The developer wants to avoid storing database connection credentials the application code. The developer would also like a solution that automatically rotates the credentials.

What is the MOST secure way to store and access the database credentials?

**Your answer is correct**

**Use AWS Secrets Manager to store the credentials. Retrieve the credentials from Secrets Manager as needed.**

**Use AWS Systems Manager Parameter store to store the credentials. Enable automatic rotation of the credentials.**

**Store the credentials in an encrypted source code repository. Retrieve the credentials from AWS CodeCommit as needed.**

**Create an IAM role that has permissions to access the database. Attach the role to the EC2 instance.**

Overall explanation

AWS Secrets Manager can be used for secure storage of secrets such as database connection credentials. Automatic rotation is supported for several RDS database types including Microsoft SQL Server. This is the most secure solution for storing and retrieving the credentials.

**CORRECT:**"Use AWS Secrets Manager to store the credentials. Retrieve the credentials from Secrets Manager as needed" is the correct answer (as explained above.)

**INCORRECT:** "Use AWS Systems Manager Parameter store to store the credentials. Enable automatic rotation of the credentials" is incorrect.

With SSM Parameter Store you cannot enable automatic rotation. You can rotate the credentials but you would need to configure your own Lambda function.

**INCORRECT:** "Create an IAM role that has permissions to access the database. Attach the role to the EC2 instance" is incorrect.

RDS for SQL Server does support windows authentication using a managed Microsoft AD with IAM roles for permissions to the AD service, but this is not described in the solution.

**INCORRECT:** "Store the credentials in an encrypted source code repository. Retrieve the credentials from AWS CodeCommit as needed" is incorrect.

This is not a solution that is suitable for retrieving database connection credentials and it does not support automatic rotation.

**References:**

<https://docs.aws.amazon.com/secretsmanager/latest/userguide/rotating-secrets.html>

**Save time with our AWS cheat sheets:**

<https://digitalcloud.training/aws-secrets-manager/>

**Domain**

AWS Security, Identity, & Compliance

**Question 56Incorrect**

A Developer has created an Amazon Cognito user pool and configured a domain for it. The Developer wants to add sign-up and sign-in pages to an app with a company logo.

What should the Developer do to meet these requirements?

**Create a REST API using Amazon API Gateway and add a Cognito authorizer. Upload the company logo to a stage in the API.**

**Your answer is incorrect**

**Create a custom login page that includes the company logo and upload it to Amazon Cognito. Specify the login page in the app client settings.**

**Correct answer**

**Customize the Amazon Cognito hosted web UI and add the company logo.**

**Upload the company logo to an Amazon S3 bucket. Specify the S3 object path in the app client settings in Amazon Cognito.**

Overall explanation

When you create a user pool in Amazon Cognito and then configure a domain for it, Amazon Cognito automatically provisions a hosted web UI to let you add sign-up and sign-in pages to your app. You can add a custom logo or customize the CSS for the hosted web UI.

**CORRECT:**"Customize the Amazon Cognito hosted web UI and add the company logo" is the correct answer.

**INCORRECT:** "Create a REST API using Amazon API Gateway and add a Cognito authorizer. Upload the company logo to a stage in the API" is incorrect. There is no need to add a REST API to this solution.

**INCORRECT:** "Upload the company logo to an Amazon S3 bucket. Specify the S3 object path in the app client settings in Amazon Cognito" is incorrect. This is not required as the hosted web UI can be used.

**INCORRECT:** "Create a custom login page that includes the company logo and upload it to Amazon Cognito. Specify the login page in the app client settings" is incorrect. This is not required as the hosted web UI can be used.

**References:**

<https://aws.amazon.com/premiumsupport/knowledge-center/cognito-hosted-web-ui/>

**Save time with our AWS cheat sheets:**

<https://digitalcloud.training/amazon-cognito/>

**Domain**

AWS Security, Identity, & Compliance

**Question 57Incorrect**

A development team manage a high-traffic e-Commerce site with dynamic pricing that is updated in real-time. There have been incidents where multiple updates occur simultaneously and cause an original editor’s updates to be overwritten. How can the developers ensure that overwriting does not occur?

**Use batch operations**

**Your answer is incorrect**

**Use atomic counters**

**Correct answer**

**Use conditional writes**

**Use concurrent writes**

Overall explanation

By default, the DynamoDB write operations (PutItem, UpdateItem, DeleteItem) are *unconditional*: Each operation overwrites an existing item that has the specified primary key.

DynamoDB optionally supports conditional writes for these operations. A conditional write succeeds only if the item attributes meet one or more expected conditions. Otherwise, it returns an error. Conditional writes are helpful in many situations. For example, you might want a PutItem operation to succeed only if there is not already an item with the same primary key. Or you could prevent an UpdateItem operation from modifying an item if one of its attributes has a certain value.

Conditional writes can be ***idempotent*** if the conditional check is on the same attribute that is being updated. This means that DynamoDB performs a given write request only if certain attribute values in the item match what you expect them to be at the time of the request.

For example, suppose that you issue an UpdateItem request to increase the Price of an item by 3, but only if the Price is currently 20. After you send the request, but before you get the results back, a network error occurs, and you don't know whether the request was successful. Because this conditional write is idempotent, you can retry the same UpdateItem request, and DynamoDB updates the item only if the Price is currently 20.

The following example shows how to use the condition-expression parameter to achieve a conditional write with idempotence:

1. aws dynamodb update-item \
2. --table-name ProductCatalog \
3. --key '{"Id":{"N":"1"}}' \
4. --update-expression "SET Price = :newval" \
5. --condition-expression "Price = :currval" \
6. --expression-attribute-values file://expression-attribute-values.json

For this scenario, conditional writes with idempotence will mean that each writer can check the current price and update the price only if the price matches that price. If the price is updated by another writer before the write is made, it will fail as the item price has changed and will not reflect the expected price.

**CORRECT:**"Use conditional writes" is the correct answer.

**INCORRECT:** "Use concurrent writes" is incorrect as writing concurrently to the same items is exactly what we want to avoid.

**INCORRECT:** "Use atomic counters" is incorrect. An atomic counter is a numeric attribute that is incremented, unconditionally, without interfering with other write requests. This is used for cases such as tracking visitors to a website. This does not prevent recent updated from being overwritten.

**INCORRECT:** "Use batch operations" is incorrect. Batch operations can reduce the number of network round trips from your application to DynamoDB. However, this does not solve the problem of preventing recent updates from being overwritten.

**References:**

<https://docs.aws.amazon.com/amazondynamodb/latest/developerguide/WorkingWithItems.html#WorkingWithItems.ConditionalUpdate>

<https://docs.aws.amazon.com/amazondynamodb/latest/developerguide/Expressions.ConditionExpressions.html>

**Save time with our AWS cheat sheets:**

<https://digitalcloud.training/amazon-dynamodb/>

**Domain**

AWS Database

**Question 58Correct**

A Developer is creating a database solution using an Amazon ElastiCache caching layer. The solution must provide strong consistency to ensure that updates to product data are consistent between the backend database and the ElastiCache cache. Low latency performance is required for all items in the database.

Which cache writing policy will satisfy these requirements?

**Add a short duration TTL value to each write.**

**Invalidate the cache for each database write.**

**Your answer is correct**

**Use a write-through caching strategy.**

**Use a lazy-loading caching strategy.**

Overall explanation

The write-through strategy adds data or updates data in the cache whenever data is written to the database. The advantages of write-through are as follows:

- Data in the cache is never stale. Because the data in the cache is updated every time it's written to the database, the data in the cache is always current.

- Write penalty vs. read penalty.

Every write involves two trips:

  1. A write to the cache

  2. A write to the database

Which adds latency to the process. That said, end users are generally more tolerant of latency when updating data than when retrieving data. There is an inherent sense that updates are more work and thus take longer.

**CORRECT:**"Use a write-through caching strategy" is the correct answer.

**INCORRECT:** "Use a lazy-loading caching strategy" is incorrect. *Lazy loading* is a caching strategy that loads data into the cache only when necessary. This will not ensure strong consistency between the database and the cache.

**INCORRECT:** "Add a short duration TTL value to each write" is incorrect. A TTL specifies the number of seconds until the key expires. This will not ensure strong consistency between the database and the cache.

**INCORRECT:** "Invalidate the cache for each database write" is incorrect. This will allow the cache to be updated when an item is next read but will not ensure the best performance for all items in the database.

**References:**

<https://docs.aws.amazon.com/AmazonElastiCache/latest/mem-ug/Strategies.html>

**Save time with our AWS cheat sheets:**

<https://digitalcloud.training/amazon-elasticache/>

**Domain**

AWS Database

**Question 59Correct**

An application uses both Amazon EC2 instances and on-premises servers. The on-premises servers are a critical component of the application, and a developer wants to collect metrics and logs from these servers. The developer would like to use Amazon CloudWatch.

How can the developer accomplish this?

**Write a batch script that uses system utilities to collect performance metrics and application logs. Upload the metrics and logs to CloudWatch.**

**Your answer is correct**

**Install the CloudWatch agent on the on-premises servers and specify IAM credentials with permissions to CloudWatch.**

**Install the CloudWatch agent on the on-premises servers and specify an IAM role with permissions to CloudWatch.**

**Install an AWS SDK on the on-premises servers that automatically sends logs to CloudWatch.**

Overall explanation

You can download the CloudWatch agent package using either Systems Manager Run Command or an Amazon S3 download link. You then install the agent and specify the IAM credentials to use. The IAM credentials are an access key and secret access key of an IAM user that has permissions to Amazon CloudWatch.

Once this has been completed the on-premises servers will automatically send metrics and log files to Amazon CloudWatch and can be centrally monitored along with AWS services.

**CORRECT:**Install the CloudWatch agent on the on-premises servers and specify IAM credentials with permissions to CloudWatch" is the correct answer (as explained above.)

**INCORRECT:** "Install the CloudWatch agent on the on-premises servers and specify an IAM role with permissions to CloudWatch" is incorrect.

You cannot specify a role with an on-premises server so you must use access keys instead.

**INCORRECT:** "Write a batch script that uses system utilities to collect performance metrics and application logs. Upload the metrics and logs to CloudWatch" is incorrect.

The CloudWatch agent would be a better solution and you must have permissions to send this information to CloudWatch.

**INCORRECT:** "Install an AWS SDK on the on-premises servers that automatically sends logs to CloudWatch" is incorrect.

The CloudWatch agent would be a better solution and you must have permissions to send this information to CloudWatch.

**References:**

<https://docs.aws.amazon.com/AmazonCloudWatch/latest/monitoring/install-CloudWatch-Agent-on-premise.html>

**Save time with our AWS cheat sheets:**

<https://digitalcloud.training/amazon-cloudwatch/>

**Domain**

AWS Management & Governance

**Question 60Correct**

An Amazon API Gateway API developer aims to integrate request validation in a production setting but wants to test it before deployment. Which of the following methods offers the least operational overhead for testing via a tool by sending test requests?

**Clone the existing API, add the request validation, run the tests, then modify the original API to include request validation before deploying to production.**

**First export the current API to an OpenAPI file, create and modify a new API by importing the OpenAPI file and adding request validation, test it, then modify and deploy the original API.**

**Create a new API from scratch with the necessary resources, methods, and request validation, run the tests, then modify and deploy the original API.**

**Your answer is correct**

**Modify the existing API to include request validation, deploy this to a new API Gateway stage, test it, then deploy it to the production stage.**

Overall explanation

The primary concern here is reducing operational overhead. The correct answer is the most straightforward and efficient because it leverages the functionality of Amazon API Gateway stages.

By modifying the existing API to add request validation and deploying the updated API to a new API Gateway stage, it allows for testing in a controlled environment before deploying to production. It provides the least operational overhead because it does not involve creating a new API or exporting/importing an OpenAPI file.

**CORRECT:**"Modify the existing API to include request validation, deploy this to a new API Gateway stage, test it, then deploy it to the production stage" is the correct answer (as explained above.)

**INCORRECT:** "Clone the existing API, add the request validation, run the tests, then modify the original API to include request validation before deploying to production" is incorrect.

Cloning the API or exporting it to an OpenAPI file and importing into a new API both involve unnecessary steps that increase operational overhead.

**INCORRECT:** "First export the current API to an OpenAPI file, create and modify a new API by importing the OpenAPI file and adding request validation, test it, then modify and deploy the original API" is incorrect.

Cloning the API or exporting it to an OpenAPI file and importing into a new API both involve unnecessary steps that increase operational overhead.

**INCORRECT:** "Create a new API from scratch with the necessary resources, methods and request validation, run the tests, then modify and deploy the original API" is incorrect.

Creating a new API from scratch can be time-consuming and requires extra effort which again adds to operational overhead.

**References:**

<https://docs.aws.amazon.com/apigateway/latest/developerguide/stages.html>

**Save time with our AWS cheat sheets:**

<https://digitalcloud.training/amazon-api-gateway/>

**Domain**

AWS Networking & Content Delivery

**Question 61Correct**

A Developer has completed some code updates and needs to deploy the updates to an Amazon Elastic Beanstalk environment. The environment includes twelve Amazon EC2 instances and there can be no reduction in application performance and availability during the update.

Which deployment policy is the most cost-effective choice to suit these requirements?

**Immutable**

**Your answer is correct**

**Rolling with additional batch**

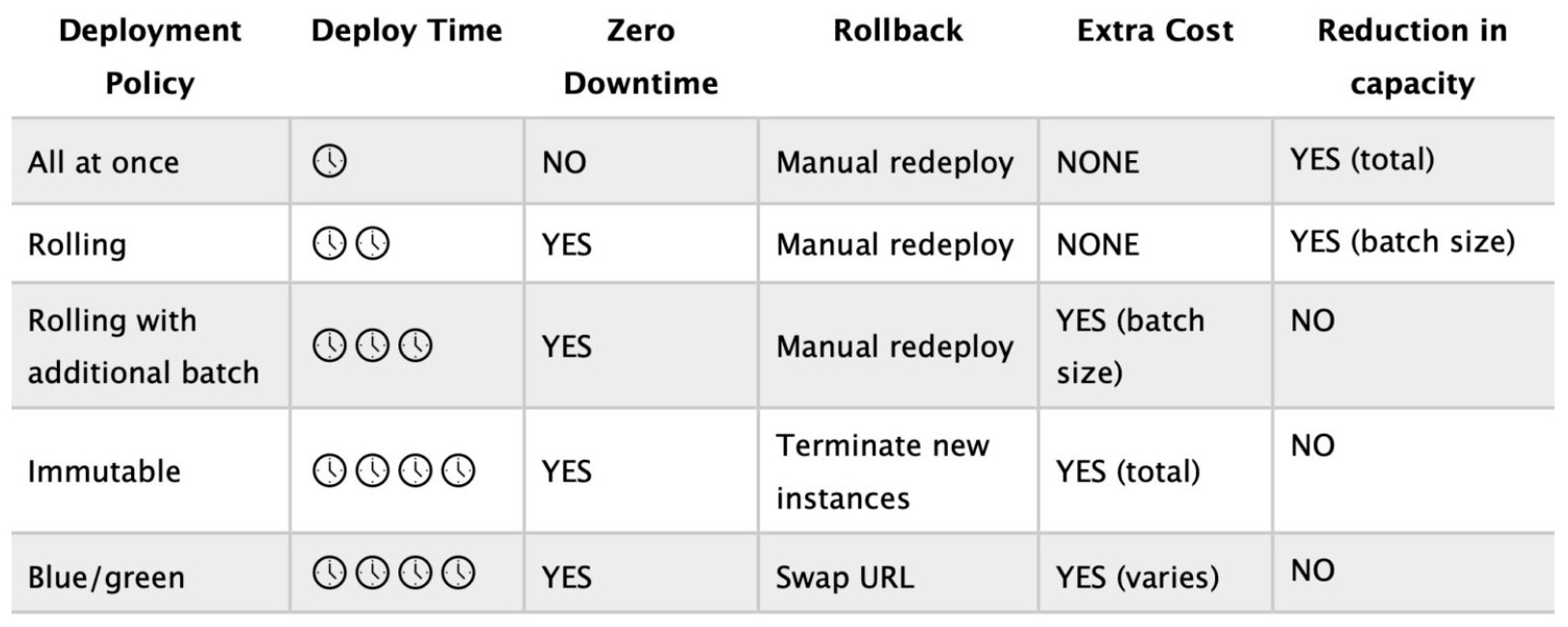
**Rolling**

**All at once**

Overall explanation

AWS Elastic Beanstalk provides several options for how deployments are processed, including deployment policies and options that let you configure batch size and health check behavior during deployments.

Each deployment policy has advantages and disadvantages and it’s important to select the best policy to use for each situation. The following tables summarizes the different deployment policies:



The “rolling with additional batch” policy will add an additional batch of instances, updates those instances, then move onto the next batch.

Rolling with additional batch:

Like Rolling but launches new instances in a batch ensuring that there is full availability.

Application is running at capacity.

Can set the bucket size.

Application is running both versions simultaneously.

Small additional cost.

Additional batch is removed at the end of the deployment.

Longer deployment.

Good for production environments.

For this scenario there can be no reduction in application performance and availability during the update. The question also asks for the most cost-effective choice.

Therefore, the “rolling with additional batch” is the best choice as it will ensure fully availability of the application but minimize cost as the additional batch size can be kept small.

**CORRECT:**"Rolling with additional batch" is the correct answer.

**INCORRECT:** "Rolling" is incorrect as this will result in a reduction in capacity as there is no additional batch of instances introduced to the environment. This is a better choice if speed is required and a reduction in capacity of a batch size is acceptable.

**INCORRECT:** "All at once" is incorrect as this will take the application down and cause a complete outage of the application during the update.

**INCORRECT:** "Immutable" is incorrect as this is the most expensive option as it doubles capacity with a whole new set of instances attached to a new ASG.

**References:**

<https://docs.aws.amazon.com/elasticbeanstalk/latest/dg/using-features.deploy-existing-version.html>

<https://docs.aws.amazon.com/elasticbeanstalk/latest/dg/using-features.rolling-version-deploy.html>

**Save time with our AWS cheat sheets:**

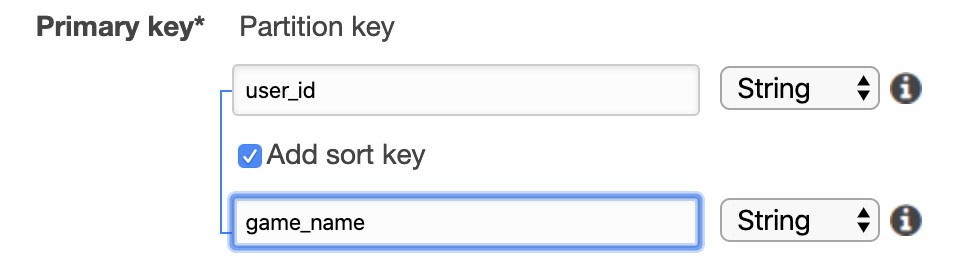
<https://digitalcloud.training/aws-elastic-beanstalk/>

**Domain**

AWS Compute

**Question 62Correct**

A gaming company is building an application to track the scores for their games using an Amazon DynamoDB table. Each item in the table is identified by a partition key  
(user\_id) and a sort key (game\_name). The table also includes the attribute “TopScore”. The table design is shown below:



A Developer has been asked to write a leaderboard application to display the highest achieved scores for each game (game\_name), based on the score identified in the “TopScore” attribute.

What process will allow the Developer to extract results MOST efficiently from the DynamoDB table?

**Your answer is correct**

**Create a global secondary index with a partition key of “game\_name” and a sort key of “TopScore” and get the results based on the score attribute**

**Create a local secondary index with a partition key of “game\_name” and a sort key of “TopScore” and get the results based on the score attribute**

**Use a DynamoDB scan operation to retrieve the scores for “game\_name” using the “TopScore” attribute, and order the results based on the score attribute**

**Create a global secondary index with a partition key of “user\_id” and a sort key of “game\_name” and get the results based on the “TopScore” attribute**

Overall explanation

In an Amazon DynamoDB table, the primary key that uniquely identifies each item in the table can be composed not only of a partition key, but also of a sort key.

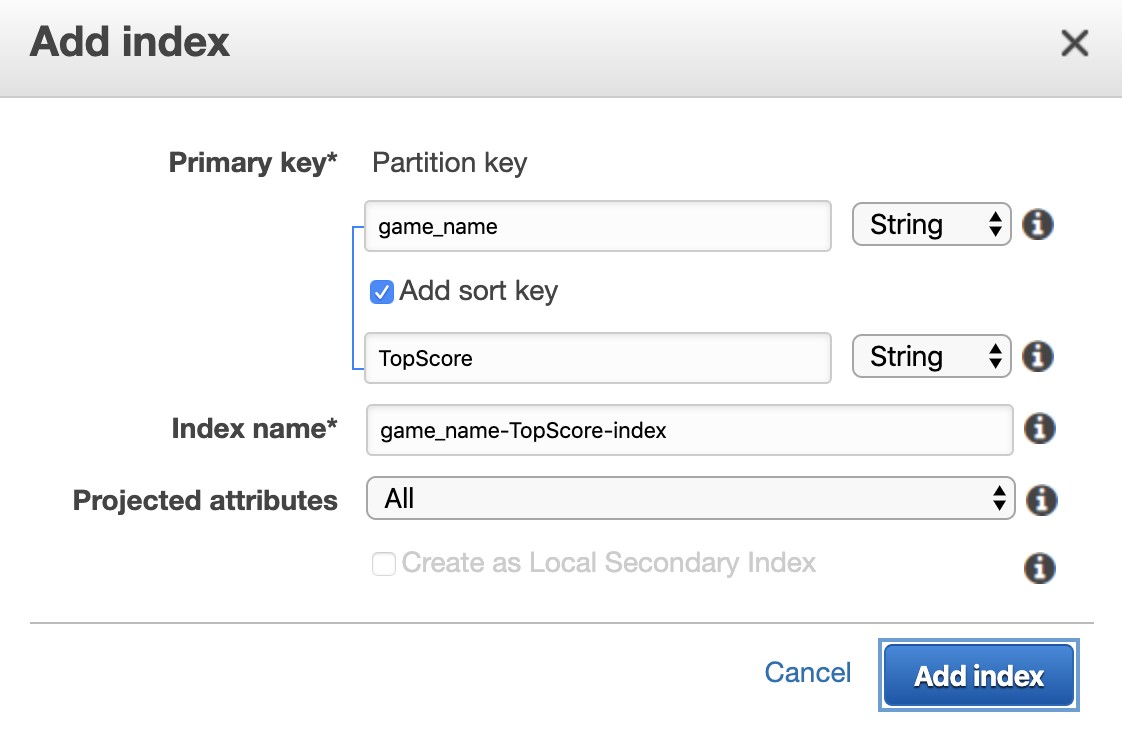
Well-designed sort keys have two key benefits:

- They gather related information together in one place where it can be queried efficiently. Careful design of the sort key lets you retrieve commonly needed groups of related items using range queries with operators such as begins\_with, between, >, <, and so on.

- Composite sort keys let you define hierarchical (one-to-many) relationships in your data that you can query at any level of the hierarchy.

To speed up queries on non-key attributes, you can create a global secondary index. A global secondary index contains a selection of attributes from the base table, but they are organized by a primary key that is different from that of the table. The index key does not need to have any of the key attributes from the table. It doesn't even need to have the same key schema as a table.

For this scenario we need to identify the top achieved score for each game. The most efficient way to do this is to create a global secondary index using “game\_name” as the partition key and “TopScore” as the sort key. We can then efficiently query the global secondary index to find the top achieved score for each game.



**CORRECT:**"Create a global secondary index with a partition key of “game\_name” and a sort key of “TopScore” and get the results based on the score attribute" is the correct answer.

**INCORRECT:** "Create a local secondary index with a partition key of “game\_name” and a sort key of “TopScore” and get the results based on the score attribute" is incorrect. With a local secondary index you can have a different sort key but the partition key is the same.

**INCORRECT:** "Use a DynamoDB scan operation to retrieve the scores for “game\_name” using the “TopScore” attribute, and order the results based on the score attribute" is incorrect. This would be inefficient as it scans the whole table. First, we should create a global secondary index, and then use a query to efficiently retrieve the data.

**INCORRECT:** "Create a global secondary index with a partition key of “user\_id” and a sort key of “game\_name” and get the results based on the score attribute" is incorrect as with a global secondary index you have a different partition key and sort key. Also, we don’t need “user\_id”, we need “game\_name” and “TopScore”.

**References:**

<https://docs.aws.amazon.com/amazondynamodb/latest/developerguide/bp-sort-keys.html>

<https://docs.aws.amazon.com/amazondynamodb/latest/developerguide/GSI.html>

**Save time with our AWS cheat sheets:**

<https://digitalcloud.training/amazon-dynamodb/>

**Domain**

AWS Database

**Question 63Incorrect**

A company uses an Amazon S3 bucket to store a large number of sensitive files relating to eCommerce transactions. The company has a policy that states that all data written to the S3 bucket must be encrypted.

How can a Developer ensure compliance with this policy?

**Correct answer**

**Create an S3 bucket policy that denies any S3 Put request that does not include the x-amz-server-side-encryption**

**Create a bucket policy that denies the S3 PutObject request with the attribute x-amz-acl having values public-read, public-read-write, or authenticated-read**

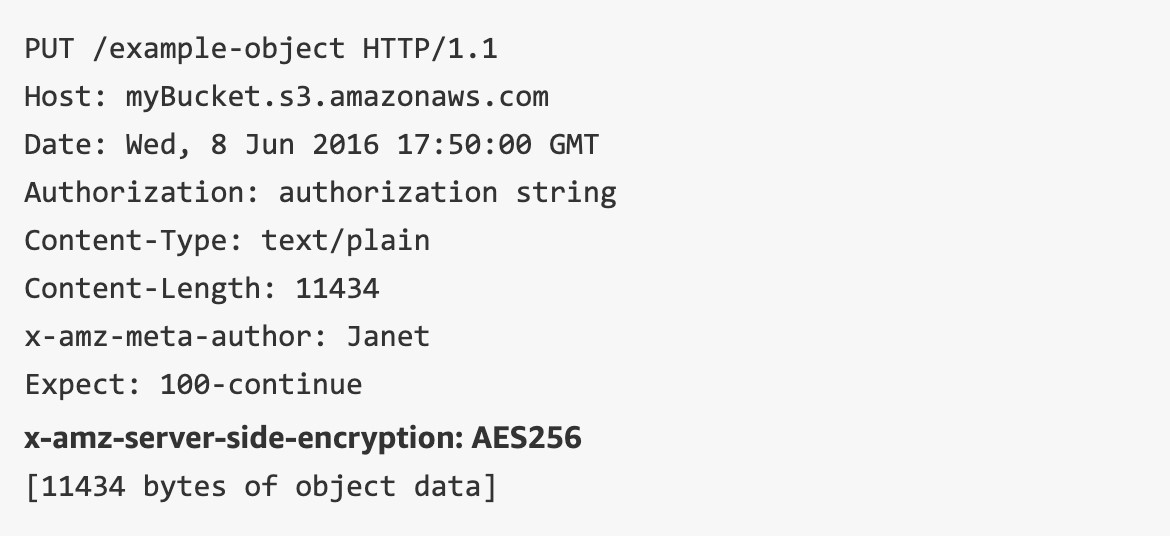
**Create an Amazon CloudWatch alarm that notifies an administrator if unencrypted objects are uploaded to the S3 bucket**

**Your answer is incorrect**

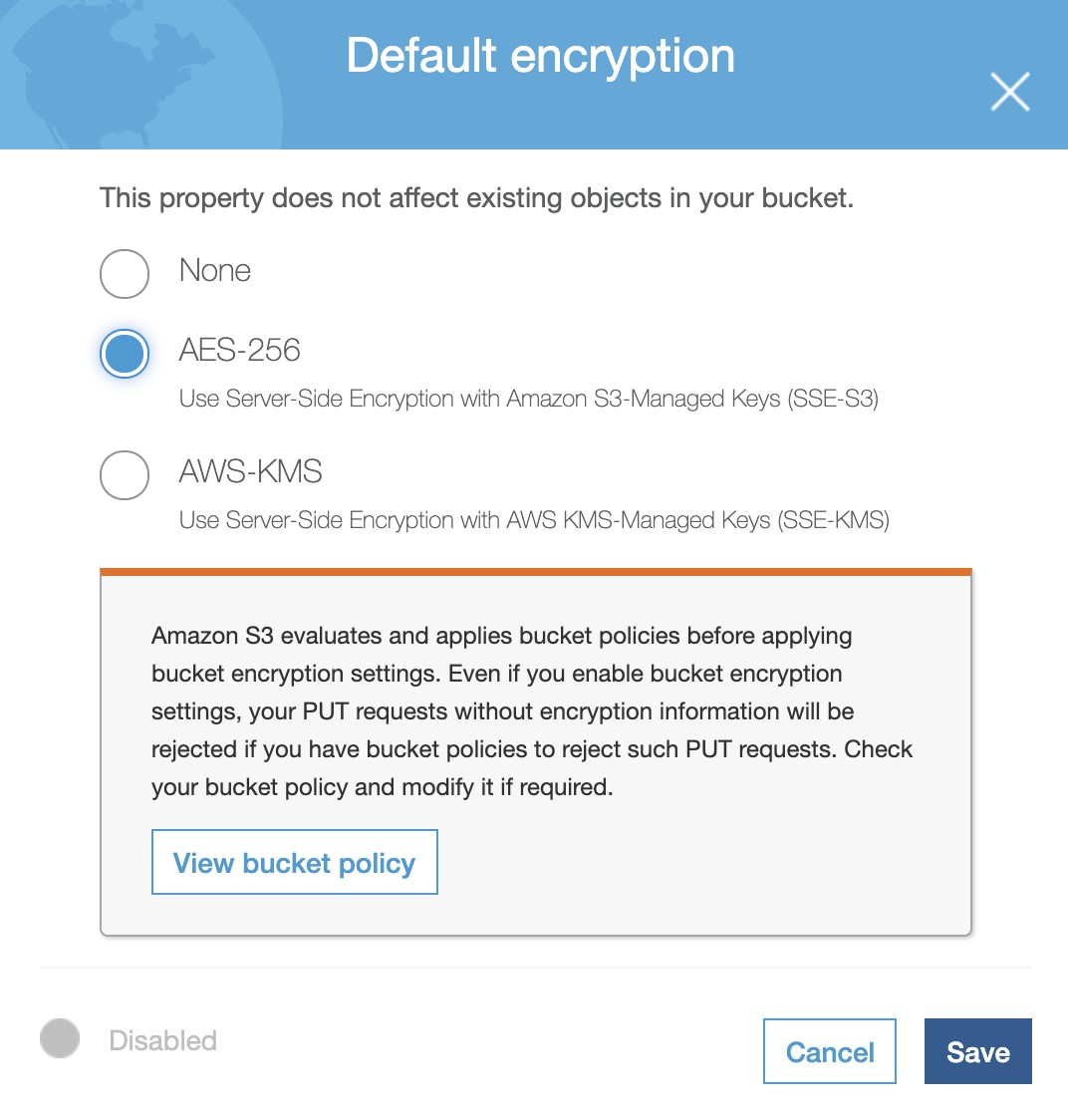
**Enable Server-Side Encryption with Amazon S3-Managed Keys (SSE-S3) on the Amazon S3 bucket**

Overall explanation

To encrypt an object at the time of upload, you need to add a header called x-amz-server-side-encryption to the request to tell S3 to encrypt the object using SSE-C, SSE-S3, or SSE-KMS. The following code example shows a Put request using SSE-S3.



Enabling encryption on an S3 bucket does not enforce encryption however, so it is still necessary to take extra steps to force compliance with the policy. As the message in the image below states, bucket policies are applied before encryption settings so PUT requests without encryption information can be rejected by a bucket policy:



Therefore, we need to create an S3 bucket policy that denies any S3 Put request that do not include the x-amz-server-side-encryption header. There are two possible values for the x-amz-server-side-encryption header: AES256, which tells S3 to use S3-managed keys, and aws:kms, which tells S3 to use AWS KMS–managed keys.

**CORRECT:**"Create an S3 bucket policy that denies any S3 Put request that does not include the x-amz-server-side-encryption" is the correct answer.

**INCORRECT:** "Create a bucket policy that denies the S3 PutObject request with the attribute x-amz-acl having values public-read, public-read-write, or authenticated-read" is incorrect. This policy means that authenticated users cannot upload objects to the bucket if the objects have public permissions.

**INCORRECT:** "Enable Server-Side Encryption with Amazon S3-Managed Keys (SSE-S3) on the Amazon S3 bucket" is incorrect as this will enable default encryption but will not enforce encryption on the S3 bucket. You do still need to enable default encryption on the bucket, but this alone will not enforce encryption.

**INCORRECT:** "Create an Amazon CloudWatch alarm that notifies an administrator if unencrypted objects are uploaded to the S3 bucket" is incorrect. This is operationally difficult to manage and only notifies, it does not prevent.

**References:**

<https://aws.amazon.com/blogs/security/how-to-prevent-uploads-of-unencrypted-objects-to-amazon-s3/>

**Save time with our AWS cheat sheets:**

<https://digitalcloud.training/amazon-s3-and-glacier/>

**Domain**

AWS Storage

**Question 64Correct**

An eCommerce application uses an Amazon RDS database with Amazon ElastiCache in front. Stock volume data is updated dynamically in listings as sales are made. Customers have complained that occasionally the stock volume data is incorrect, and they end up purchasing items that are out of stock. A Developer has checked the front end and indeed some items display the incorrect stock count.

What could be causing this issue?

**Your answer is correct**

**The cache is not being invalidated when the stock volume data is changed.**

**The stock volume data is being retrieved using a write-through ElastiCache cluster.**

**The Amazon RDS database has insufficient IOPS provisioned for its EBS volumes.**

**The Amazon RDS database is deployed as Multi-AZ and the standby is inconsistent.**

Overall explanation

Amazon ElastiCache is being used to cache data from the Amazon RDS database to improve performance when performing queries. In this case the cache has stale stock volume data stored and is returning this information when customers are purchasing items.

The resolution is to ensure that the cache is invalidated whenever the stock volume data is changed. This can be done in the application layer.

**CORRECT:**"The cache is not being invalidated when the stock volume data is changed" is the correct answer.

**INCORRECT:** "The stock volume data is being retrieved using a write-through ElastiCache cluster" is incorrect. If this was the case the data would not be stale.

**INCORRECT:** "The Amazon RDS database is deployed as Multi-AZ and the standby is inconsistent" is incorrect. Multi-AZ standbys are not used for reading data and the replication is synchronous so it would not be inconsistent.

**INCORRECT:** "The Amazon RDS database has insufficient IOPS provisioned for its EBS volumes" is incorrect. This is not the issue here; the stale data is being retrieved from the ElastiCache database.

**References:**

<https://docs.aws.amazon.com/AmazonElastiCache/latest/red-ug/Strategies.html>

**Save time with our AWS cheat sheets:**

<https://digitalcloud.training/amazon-elasticache/>

**Domain**

AWS Database

**Question 65Incorrect**

An application that processes financial transactions receives thousands of transactions each second. The transactions require end-to-end encryption, and the application implements this by using the AWS KMS GenerateDataKey operation. During operation the application receives the following error message:

*“You have exceeded the rate at which you may call KMS. Reduce the frequency of your calls.*

*(Service: AWSKMS; Status Code: 400; Error Code: ThrottlingException; Request ID: <ID>”*

Which actions are best practices to resolve this error? (Select TWO.)

**Correct selection**

**Create a case in the AWS Support Center to increase the quota for the account.**

**Your selection is incorrect**

**Use Amazon SQS to queue the requests and configure AWS KMS to poll the queue.**

**Correct selection**

**Create a local cache using the AWS Encryption SDK and the LocalCryptoMaterialsCache feature.**

**Create an AWS KMS custom key store and generate data keys through AWS CloudHSM.**

**Your selection is incorrect**

**Call the AWS KMS Encrypt operation directly to allow AWS KMS to encrypt the data.**

Overall explanation

To ensure that AWS KMS can provide fast and reliable responses to API requests from all customers, it throttles API requests that exceed certain boundaries. *Throttling* occurs when AWS KMS rejects an otherwise valid request and returns a ThrottlingException error.

*Data key caching* stores data keys and related cryptographic material in a cache. When you encrypt or decrypt data, the AWS Encryption SDK looks for a matching data key in the cache. If it finds a match, it uses the cached data key rather than generating a new one. Data key caching can improve performance, reduce cost, and help you stay within service limits as your application scales.

Your application can benefit from data key caching if:

• It can reuse data keys.

• It generates numerous data keys.

• Your cryptographic operations are unacceptably slow, expensive, limited, or resource-intensive.

To create an instance of the local cache, use the LocalCryptoMaterialsCache constructor in Java and Python, the getLocalCryptographicMaterialsCache function in JavaScript, or the aws\_cryptosdk\_materials\_cache\_local\_new constructor in C.

Additionally, the developer can request an increase in the quota for AWS KMS which will provide the ability to submit more API calls the AWS KMS.

**CORRECT:**"Create a local cache using the AWS Encryption SDK and the LocalCryptoMaterialsCache feature" is a correct answer (as explained above.)

**CORRECT:**"Create a case in the AWS Support Center to increase the quota for the account" is also a correct answer (as explained above.)

**INCORRECT:** "Call the AWS KMS Encrypt operation directly to allow AWS KMS to encrypt the data" is incorrect.

This will not reduce API calls to AWS KMS. Additionally, there are limits to the maximum size of the data that can be encrypted using this method. The max is 4096 bytes.

**INCORRECT:** "Use Amazon SQS to queue the requests and configure AWS KMS to poll the queue" is incorrect.

KMS cannot be configured to poll and SQS queue.

**INCORRECT:** "Create an AWS KMS custom key store and generate data keys through AWS CloudHSM" is incorrect.

This is an unnecessary step and would incur additional cost. CloudHSM is not beneficial for this specific situation.

**References:**

<https://docs.aws.amazon.com/encryption-sdk/latest/developer-guide/data-key-caching.html>

**Save time with our AWS cheat sheets:**

<https://digitalcloud.training/aws-kms/>

**Domain**

AWS Security, Identity, & Compliance