**AWS Certified Developer Associate – Results – PT1**

Top of Form

Question 1: **Incorrect**

A Developer is designing a fault-tolerant application that will use Amazon EC2 instances and an Elastic Load Balancer. The Developer needs to ensure that if an EC2 instance fails session data is not lost. How can this be achieved?

* **Enable Sticky Sessions on the Elastic Load Balancer**

**(Incorrect)**

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

**(Correct)**

* **Use an EC2 Auto Scaling group to automatically launch new instances**
* **Use Amazon SQS to save session data**

**Explanation**

For this scenario the key requirement is to ensure the data is not lost. Therefore, the data must be stored in a durable data store outside of the EC2 instances. Amazon DynamoDB is a suitable solution for storing session data. DynamoDB has a session handling capability for multiple languages as in the below example for PHP:

“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.”

Therefore, the best answer is to use DynamoDB to store the session data.

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

**INCORRECT:** "Enable Sticky Sessions on the Elastic Load Balancer" is incorrect. Sticky sessions attempts to direct a user that has reconnected to the application to the same EC2 instance that they connected to previously. However, this does not ensure that the session data is going to be available.

**INCORRECT:** "Use an EC2 Auto Scaling group to automatically launch new instances" is incorrect as this does not provide a solution for storing the session data.

**INCORRECT:** "Use Amazon SQS to save session data" is incorrect as Amazon SQS is not suitable for storing session data.

**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/>

Bottom of Form

Top of Form

Question 2: **Incorrect**

A Developer is writing an imaging microservice on AWS Lambda. The service is dependent on several libraries that are not available in the Lambda runtime environment.

Which strategy should the Developer follow to create the Lambda deployment package?

* **Create a ZIP file with the source code and all dependent libraries**

**(Correct)**

* **Create a ZIP file with the source code and a buildspec.yml file that installs the dependent libraries on AWS Lambda**

**(Incorrect)**

* **Create a ZIP file with the source code and a script that installs the dependent libraries at runtime**
* **Create a ZIP file with the source code and an appspec.yml file. Add the libraries to the appspec.yml file and upload to Amazon S3. Deploy using CloudFormation**

**Explanation**

A deployment package is a ZIP archive that contains your function code and dependencies. You need to create a deployment package if you use the Lambda API to manage functions, or if you need to include libraries and dependencies other than the AWS SDK.

You can upload the package directly to Lambda, or you can use an Amazon S3 bucket, and then upload it to Lambda. If the deployment package is larger than 50 MB, you must use Amazon S3.

**CORRECT:**"Create a ZIP file with the source code and all dependent libraries" is the correct answer.

**INCORRECT:** "Create a ZIP file with the source code and a script that installs the dependent libraries at runtime" is incorrect as the Developer should not run a script at runtime as this will cause latency. Instead, the Developer should include the dependent libraries in the ZIP package.

**INCORRECT:** "Create a ZIP file with the source code and an appspec.yml file. Add the libraries to the appspec.yml file and upload to Amazon S3. Deploy using CloudFormation" is incorrect. The appspec.yml file is used with CodeDeploy, you cannot add libraries into it, and it is not deployed using CloudFormation.

**INCORRECT:** "Create a ZIP file with the source code and a buildspec.yml file that installs the dependent libraries on AWS Lambda" is incorrect as the buildspec.yml file is used with CodeBuild for compiling source code and running tests. It cannot be used to install dependent libraries within Lambda.

**References:**

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

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

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

Bottom of Form

Top of Form

Question 3: **Correct**

A company runs a legacy application that uses an XML-based SOAP interface. The company needs to expose the functionality of the service to external customers and plans to use Amazon API Gateway.

How can a Developer configure the integration?

* **Create a RESTful API using Amazon API Gateway. Pass the incoming JSON to the SOAP interface through an Application Load Balancer.**
* **Create a SOAP API using Amazon API Gateway. Pass the incoming JSON to the SOAP interface through a Network Load Balancer.**
* **Create a SOAP API using Amazon API Gateway. Transform the incoming JSON into a valid XML message for the SOAP interface using AWS Lambda.**
* **Create a RESTful API using Amazon API Gateway. Transform the incoming JSON into a valid XML message for the SOAP interface using mapping templates.**

**(Correct)**

**Explanation**

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.

**CORRECT:**"Create a RESTful API using Amazon 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 using Amazon API Gateway. Pass the incoming JSON to the SOAP interface through an Application Load Balancer" is incorrect. The API Gateway cannot process the XML SOAP data and cannot pass it through an ALB.

**INCORRECT:** "Create a SOAP API using Amazon API Gateway. Transform the incoming JSON into a valid XML message for the SOAP interface using AWS Lambda" is incorrect. API Gateway does not support SOAP APIs.

**INCORRECT:** "Create a SOAP API using Amazon API Gateway. Pass the incoming JSON to the SOAP interface through a Network Load Balancer" is incorrect. API Gateway does not support SOAP APIs.

**References:**

[https://docs.aws.amazon.com/apigateway/latest/Developerguide/request-response-data-mappings.html](https://docs.aws.amazon.com/apigateway/latest/developerguide/request-response-data-mappings.html)

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

[https://digitalcloud.training/amazon-api-gateway/](https://digitalcloud.training/certification-training/aws-developer-associate/aws-networking-and-content-delivery/amazon-api-gateway/)

Bottom of Form

Top of Form

Question 4: **Correct**

A company is migrating to the AWS Cloud and needs to build a managed Public Key Infrastructure (PKI) using AWS services. The solution must support the following features:

     - IAM integration.

     - Auditing with AWS CloudTrail.

     - Private certificates.

     - Subordinate certificate authorities (CAs).

Which solution should the company use to meet these requirements?

* **AWS Private Certificate Authority.**

**(Correct)**

* **AWS Key Management Service.**
* **AWS Secrets Manager.**
* **AWS Certificate Manager.**

**Explanation**

An AWS Private CA hierarchy provides strong security and restrictive access controls for the most-trusted root CA at the top of the trust chain, while allowing more permissive access and bulk certificate issuance for subordinate CAs lower on the chain.

With AWS Private CA, you can create private certificates to identify resources and protect data. You can create versatile certificate and CA configurations to identify and protect your resources, including servers, applications, users, devices, and containers.

The service offers direct integration with AWS IAM, and you can control access to AWS Private CA with IAM policies.

**CORRECT:**"AWS Private Certificate Authority" is the correct answer (as explained above.)

**INCORRECT:** "AWS Certificate Manager" is incorrect. This service is used for issuing SSL/TLS certificates and is not suitable as a private CA for building a PKI with subordinate CAs.

**INCORRECT:** "AWS Key Management Service" is incorrect. This service is used to create and manage the encryption keys used for encrypting data at rest.

**INCORRECT:** "AWS Secrets Manager" is incorrect. This service is used for storing secret information such as database connections strings and passwords with API access.

**References:**

<https://aws.amazon.com/private-ca/features/>

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

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

Bottom of Form

Top of Form

Question 5: **Incorrect**

A serverless application uses an AWS Lambda function to process Amazon S3 events. The Lambda function executes 20 times per second and takes 20 seconds to complete each execution.

How many concurrent executions will the Lambda function require?

* **5**
* **40**
* **400**

**(Correct)**

* **20**

**(Incorrect)**

**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.

To calculate the concurrency requirements for the Lambda function simply multiply the number of executions per second (20) by the time it takes to complete the execution (20).

Therefore, for this scenario, the calculation is 20 x 20 = 400.

**CORRECT:**"400" is the correct answer.

**INCORRECT:** "5" is incorrect. Please use the formula above to calculate concurrency requirements.

**INCORRECT:** "40" is incorrect. Please use the formula above to calculate concurrency requirements.

**INCORRECT:** "20" is incorrect. Please use the formula above to calculate concurrency 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/>

Bottom of Form

Top of Form

Question 6: **Correct**

A Developer has recently created an application that uses an AWS Lambda function, an Amazon DynamoDB table, and also sends notifications using Amazon SNS. The application is not working as expected and the Developer needs to analyze what is happening across all components of the application.

What is the BEST way to analyze the issue?

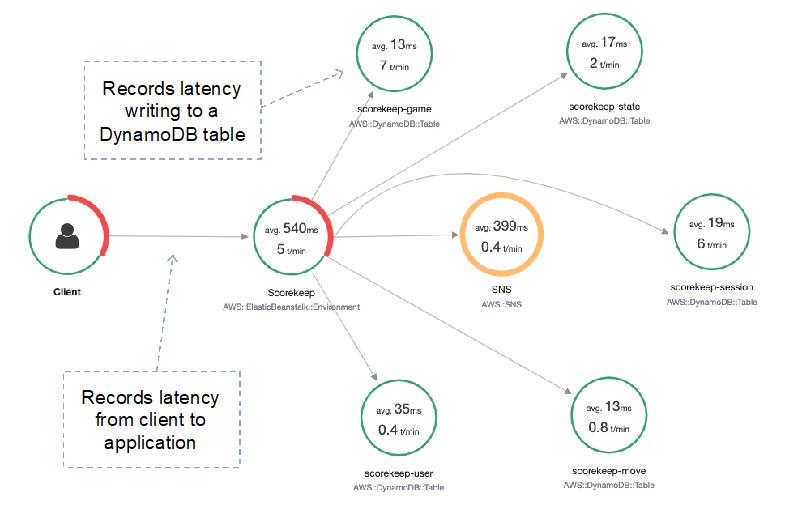
* **Create an Amazon CloudWatch Events rule**
* **Assess the application with Amazon Inspector**
* **Monitor the application with AWS Trusted Advisor**
* **Enable X-Ray tracing for the Lambda function**

**(Correct)**

**Explanation**

AWS X-Ray makes it easy for developers to analyze the behavior of their production, distributed applications with end-to-end tracing capabilities. You can use X-Ray to identify performance bottlenecks, edge case errors, and other hard to detect issues.

AWS X-Ray provides an end-to-end, cross-service view of requests made to your application. It gives you an application-centric view of requests flowing through your application by aggregating the data gathered from individual services in your application into a single unit called a trace. You can use this trace to follow the path of an individual request as it passes through each service or tier in your application so that you can pinpoint where issues are occurring.



AWS X-Ray will assist the developer with visually analyzing the end-to-end view of connectivity between the application components and how they are performing using a Service Map. X-Ray also provides aggregated data about the application.

**CORRECT:**"Enable X-Ray tracing for the Lambda function" is the correct answer.

**INCORRECT:** "Create an Amazon CloudWatch Events rule" is incorrect as this feature of CloudWatch is used to trigger actions based on changes in the state of AWS services.

**INCORRECT:** "Assess the application with Amazon Inspector" is incorrect. Amazon Inspector is an automated security assessment service that helps improve the security and compliance of applications deployed on AWS.

**INCORRECT:** "Monitor the application with AWS Trusted Advisor" is incorrect. **AWS Trusted Advisor** is an online tool that provides you real time guidance to help you provision your resources following AWS best practices.

**References:**

<https://aws.amazon.com/xray/features/>

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

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

Bottom of Form

Top of Form

Question 7: **Correct**

A Developer is creating a web application that will be used by employees working from home. The company uses a SAML directory on-premises for storing user information. The Developer must integrate with the SAML directory and authorize each employee to access only their own data when using the application.

Which approach should the Developer take?

* **Use an Amazon Cognito identity pool, federate with the SAML provider, and use a trust policy with an IAM condition key to limit employee access.**

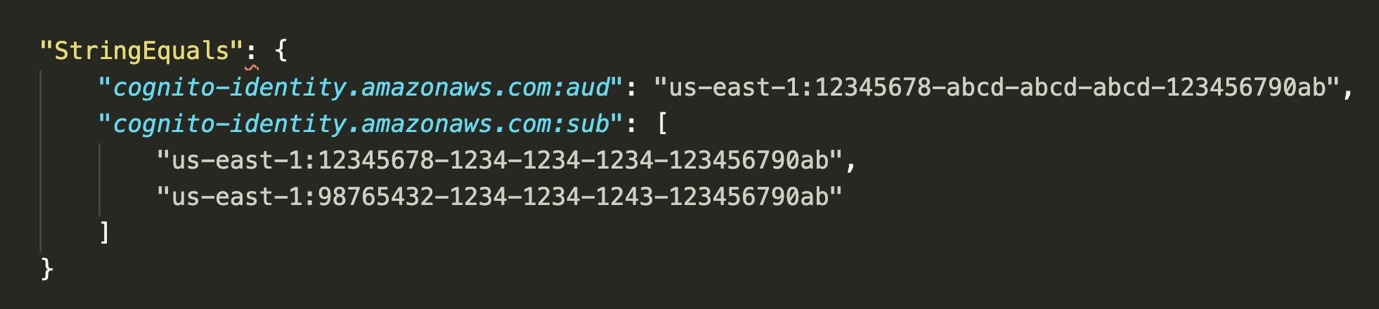
**(Correct)**

* **Create the application within an Amazon VPC and use a VPC endpoint with a trust policy to grant access to the employees.**
* **Use Amazon Cognito user pools, federate with the SAML provider, and use user pool groups with an IAM policy.**
* **Create a unique IAM role for each employee and have each employee assume the role to access the application so they can access their personal data only.**

**Explanation**

Amazon Cognito leverages IAM roles to generate temporary credentials for your application's users. Access to permissions is controlled by a role's trust relationships.

In this example the Developer must limit access to specific identities in the SAML directory. The Developer can create a trust policy with an IAM condition key that limits access to a specific set of app users by checking the value of cognito-identity.amazonaws.com:sub:



**CORRECT:**"Use an Amazon Cognito identity pool, federate with the SAML provider, and use a trust policy with an IAM condition key to limit employee access" is the correct answer.

**INCORRECT:** "Use Amazon Cognito user pools, federate with the SAML provider, and use user pool groups with an IAM policy" is incorrect. A user pool can be used to authenticate but the identity pool is used to provide authorized access to AWS services.

**INCORRECT:** "Create the application within an Amazon VPC and use a VPC endpoint with a trust policy to grant access to the employees" is incorrect. You cannot provide access to an on-premises SAML directory using a VPC endpoint.

**INCORRECT:** "Create a unique IAM role for each employee and have each employee assume the role to access the application so they can access their personal data only" is incorrect. This is not an integration into the SAML directory and would be very difficult to manage.

**References:**

[https://docs.aws.amazon.com/cognito/latest/Developerguide/role-trust-and-permissions.html](https://docs.aws.amazon.com/cognito/latest/developerguide/role-trust-and-permissions.html)

[https://docs.aws.amazon.com/cognito/latest/Developerguide/iam-roles.html](https://docs.aws.amazon.com/cognito/latest/developerguide/iam-roles.html)

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

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

Bottom of Form

Top of Form

Question 8: **Incorrect**

A Developer is setting up a code update to Amazon ECS using AWS CodeDeploy. The Developer needs to complete the code update quickly. Which of the following deployment types should the Developer use?

* **Canary**
* **In-place**

**(Incorrect)**

* **Blue/green**

**(Correct)**

* **Linear**

**Explanation**

CodeDeploy provides two deployment type options – in-place and blue/green. Note that AWS Lambda and Amazon ECS deployments cannot use an in-place deployment type.

The Blue/green deployment type on an Amazon ECS compute platform works like this:

Traffic is shifted from the task set with the original version of an application in an Amazon ECS service to a replacement task set in the same service.

You can set the traffic shifting to linear or canary through the deployment configuration.

The protocol and port of a specified load balancer listener is used to reroute production traffic.

During a deployment, a test listener can be used to serve traffic to the replacement task set while validation tests are run.

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

**INCORRECT:** "Canary" is incorrect as this is a traffic shifting option, not a deployment type. Traffic is shifted in two increments.

**INCORRECT:** "Linear" is incorrect as this is a traffic shifting option, not a deployment type. Traffic is shifted in two increments.

**INCORRECT:** "In-place" is incorrect as AWS Lambda and Amazon ECS deployments cannot use an in-place deployment type.

**References:**

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

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

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

Bottom of Form

Top of Form

Question 9: **Incorrect**

A business operates a web app on Amazon EC2 instances utilizing a bespoke Amazon Machine Image (AMI). They employ AWS CloudFormation for deploying their app, which is currently active in the us-east-1 Region. However, their goal is to extend the deployment to the us-west-1 Region.

During an initial attempt to create an AWS CloudFormation stack in us-west-1, the action fails, and an error message indicates that the AMI ID does not exist. A developer is tasked with addressing this error through a method that minimizes operational complexity.

Which action should the developer take?

* **Create a new AMI in the us-west-1 Region and update the CloudFormation template with the new AMI ID.**
* **Use AWS Lambda to create an AMI in the us-west-1 Region during stack creation.**
* **Copy the AMI from the us-east-1 Region to the us-west-1 Region and use the new AMI ID in the CloudFormation template.**

**(Correct)**

* **Modify the CloudFormation template to refer to the AMI in us-east-1 Region.**

**(Incorrect)**

**Explanation**

This is the best option as it allows the developer to use the same AMI in a different region with minimal effort and maintenance.

**CORRECT:**"Copy the AMI from the us-east-1 Region to the us-west-1 Region and use the new AMI ID in the CloudFormation template" is the correct answer (as explained above.)

**INCORRECT:** "Create a new AMI in the us-west-1 Region and update the CloudFormation template with the new AMI ID" is incorrect.

This is incorrect as creating a new AMI would be operationally complex and time-consuming.

**INCORRECT:** "Modify the CloudFormation template to refer to the AMI in us-east-1 Region" is incorrect.

AMIs are regional resources and cannot be used directly in other regions.

**INCORRECT:** "Use AWS Lambda to create an AMI in the us-west-1 Region during stack creation" is incorrect.

This process would add unnecessary complexity and the new AMI would not be identical to the original one.

**References:**

<https://docs.aws.amazon.com/AWSEC2/latest/UserGuide/CopyingAMIs.html>

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

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

Bottom of Form

Top of Form

Question 10: **Incorrect**

A company is using an AWS Step Functions state machine. When testing the state machine errors were experienced in the Step Functions task state machine. To troubleshoot the issue a developer requires that the state input be included along with the error message in the state output.

Which coding practice can preserve both the original input and the error for the state?

* **Use ResultPath in a Catch statement to include the original input with the error.**

**(Correct)**

* **Use InputPath in a Catch statement to include the original input with the error.**

**(Incorrect)**

* **Use ErrorEquals in a Retry statement to include the original input with the error.**
* **Use OutputPath in a Retry statement to include the original input with the error.**

**Explanation**

A Step Functions execution receives a JSON text as input and passes that input to the first state in the workflow. Individual states receive JSON as input and usually pass JSON as output to the next state.

In the Amazon States Language, these fields filter and control the flow of JSON from state to state:

• InputPath

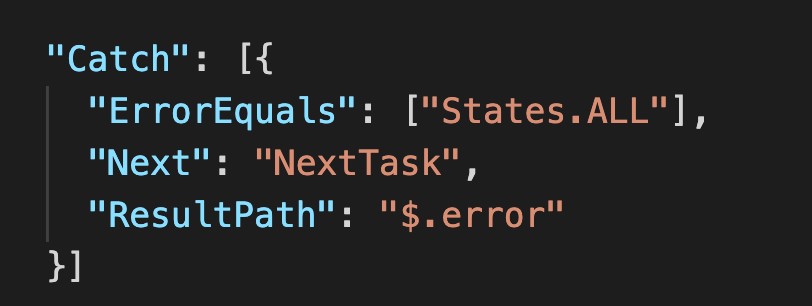
• OutputPath

• ResultPath

• Parameters

• ResultSelector

Use ResultPath to combine a task result with task input, or to select one of these. The path you provide to ResultPath controls what information passes to the output. Use ResultPath in a Catch to include the error with the original input, instead of replacing it. The following code is an example of this tactic:



**CORRECT:**"Use ResultPath in a Catch statement to include the original input with the error" is the correct answer (as explained above.)

**INCORRECT:** "Use InputPath in a Catch statement to include the original input with the error" is incorrect.

You can use InputPath to select a portion of the state input.

**INCORRECT:** "Use ErrorEquals in a Retry statement to include the original input with the error" is incorrect.

A retry is used to attempt to retry the process that caused the error based on the retry policy described by ErrorEquals.

**INCORRECT:** "Use OutputPath in a Retry statement to include the original input with the error" is incorrect.

OutputPath enables you to select a portion of the state output to pass to the next state. This enables you to filter out unwanted information and pass only the portion of JSON that you care about.

**References:**

<https://docs.aws.amazon.com/step-functions/latest/dg/input-output-resultpath.html>

<https://docs.aws.amazon.com/step-functions/latest/dg/tutorial-handling-error-conditions.html>

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

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

Bottom of Form

Top of Form

Question 11: **Incorrect**

A developer is setting up the primary key for an Amazon DynamoDB table that logs a company's purchases from various suppliers. Each transaction is recorded with these attributes: supplierId, transactionTime, item, and unitCost.

Which primary key configuration will be valid in this case?

* **A composite primary key with supplierId as the partition key and unitCost as the sort key.**
* **A composite primary key with supplierId as the partition key and transactionTime as the sort key.**

**(Correct)**

* **A composite primary key with supplierId as the partition key and item as the sort key.**

**(Incorrect)**

* **A simple primary key with supplierId serving as the partition key.**

**Explanation**

A composite primary key with supplierId as the partition key and transactionTime as the sort key ensures each entry is unique. It is extremely unlikely for the same supplier to make two separate deliveries at the exact same millisecond.

**CORRECT:**"A composite primary key with supplierId as the partition key and transactionTime as the sort key" is the correct answer (as explained above.)

**INCORRECT:** "A simple primary key with supplierId serving as the partition key" is incorrect.

This simple primary key (only partition key) will not ensure uniqueness because multiple purchases can be made from the same supplier.

**INCORRECT:** "A composite primary key with supplierId as the partition key and item as the sort key" is incorrect.

Using supplierId and item as a composite key would not guarantee uniqueness, as a supplier can deliver the same item multiple times.

**INCORRECT:** "A composite primary key with supplierId as the partition key and unitCost as the sort key" is incorrect.

A composite key using supplierId and unitCost would not ensure uniqueness either, as the same supplier can supply multiple items with the same unit cost.

**References:**

<https://repost.aws/knowledge-center/dynamodb-create-composite-key>

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

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

Bottom of Form

Top of Form

Question 12: **Correct**

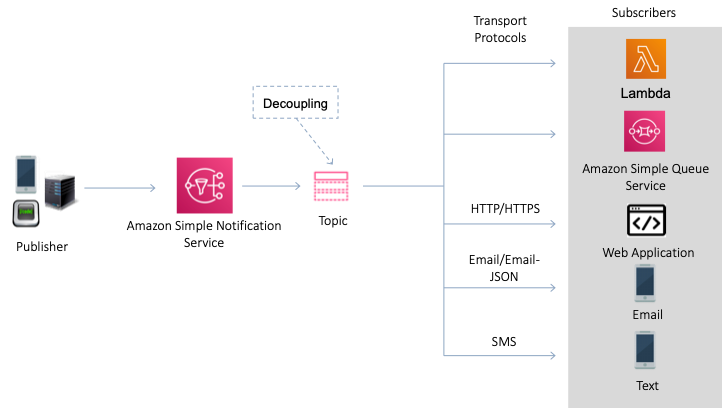
An application needs to generate SMS text messages and emails for a large number of subscribers. Which AWS service can be used to send these messages to customers?

* **Amazon SES**
* **Amazon SWF**
* **Amazon SQS**
* **Amazon SNS**

**(Correct)**

**Explanation**

Amazon Simple Notification Service (Amazon SNS) is a web service that coordinates and manages the delivery or sending of messages to subscribing endpoints or clients. In Amazon SNS, there are two types of clients—publishers and subscribers—also referred to as producers and consumers.



Publishers communicate asynchronously with subscribers by producing and sending a message to a topic, which is a logical access point and communication channel.

Subscribers (that is, web servers, email addresses, Amazon SQS queues, AWS Lambda functions) consume or receive the message or notification over one of the supported protocols (that is, Amazon SQS, HTTP/S, email, SMS, Lambda) when they are subscribed to the topic.

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

**INCORRECT:** "Amazon SES" is incorrect as this service only sends email, not SMS text messages.

**INCORRECT:** "Amazon SQS" is incorrect as this is a hosted message queue for decoupling application components.

**INCORRECT:** "Amazon SWF" is incorrect as the Simple Workflow Service is used for orchestrating multi-step workflows.

**References:**

<https://docs.aws.amazon.com/sns/latest/dg/welcome.html>

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

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

Bottom of Form

Top of Form

Question 13: **Correct**

An application uses an Amazon RDS database. The company requires that the performance of database reads is improved, and they want to add a caching layer in front of the database. The cached data must be encrypted, and the solution must be highly available.

Which solution will meet these requirements?

* **Amazon ElastiCache for Redis in cluster mode.**

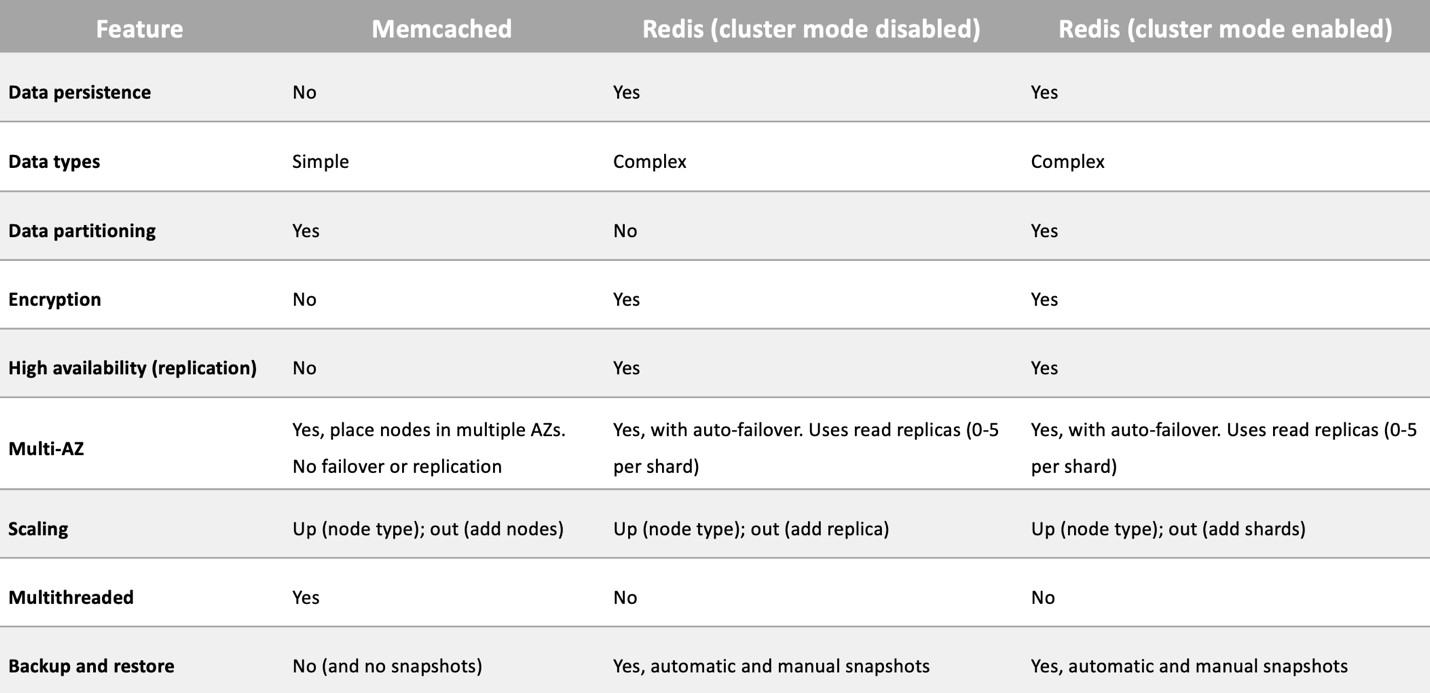
**(Correct)**

* **Amazon ElastiCache for Memcached.**
* **Amazon DynamoDB Accelerator (DAX).**
* **Amazon CloudFront with multiple origins.**

**Explanation**

Amazon ElastiCache is an in-memory database cache that can be used in front of Amazon RDS. The key to answering this question is to know the differences between ElastiCache Memcached and ElastiCache Redis. To support both encryption and high availability we must use ElastiCache Redis with cluster mode enabled.

You can see the differences between the different engines and configuration options for ElastiCache in the table below:



**CORRECT:**"Amazon ElastiCache for Redis in cluster mode" is the correct answer (as explained above.)

**INCORRECT:** "Amazon ElastiCache for Memcached" is incorrect.

The Memcached engine does not support encryption or high availability.

**INCORRECT:** "Amazon CloudFront with multiple origins" is incorrect.

You cannot configure an Amazon RDS as an origin for Amazon RDS. Also, what would the second origin be anyway? There’s only one database!

**INCORRECT:** "Amazon DynamoDB Accelerator (DAX)" is incorrect.

DynamoDB DAX can be used to increase the performance of DynamoDB tables and offload read requests. It cannot be used in front of an Amazon RDS database.

**References:**

<https://docs.aws.amazon.com/AmazonElastiCache/latest/red-ug/Replication.Redis-RedisCluster.html>

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

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

Bottom of Form

Top of Form

Question 14: **Correct**

A Developer is building a three-tier web application that must be able to handle a minimum of 10,000 requests per minute. The requirements state that the web tier should be completely stateless while the application maintains session state data for users.

How can the session state data be maintained externally, whilst keeping latency at the LOWEST possible value?

* **Implement a shared Amazon EFS file system solution across the underlying Amazon EC2 instances, then implement session handling at the application level to leverage the EFS file system for session data storage**
* **Create an Amazon RedShift instance, then implement session handling at the application level to leverage a database inside the RedShift database instance for session data storage**
* **Create an Amazon DynamoDB table, then implement session handling at the application level to leverage the table for session data storage**
* **Create an Amazon ElastiCache Redis cluster, then implement session handling at the application level to leverage the cluster for session data storage**

**(Correct)**

**Explanation**

It is common to use key/value stores for storing session state data. The two options presented in the answers are Amazon DynamoDB and Amazon ElastiCache Redis. Of these two, ElastiCache will provide the lowest latency as it is an in-memory database.

Therefore, the best answer is to create an Amazon ElastiCache Redis cluster, then implement session handling at the application level to leverage the cluster for session data storage

**CORRECT:**"Create an Amazon ElastiCache Redis cluster, then implement session handling at the application level to leverage the cluster for session data storage" is the correct answer.

**INCORRECT:** "Create an Amazon DynamoDB table, then implement session handling at the application level to leverage the table for session data storage" is incorrect as though this is a good solution for storing session state data, the latency will not be as low as with ElastiCache.

**INCORRECT:** "Create an Amazon RedShift instance, then implement session handling at the application level to leverage a database inside the RedShift database instance for session data storage" is incorrect. RedShift is a data warehouse that is used for OLAP use cases, not for storing session state data.

**INCORRECT:** "Implement a shared Amazon EFS file system solution across the underlying Amazon EC2 instances, then implement session handling at the application level to leverage the EFS file system for session data storage" is incorrect. For session state data a key/value store such as DynamoDB or ElastiCache will provide better performance.

**References:**

<https://aws.amazon.com/caching/session-management/>

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

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

Bottom of Form

Top of Form

Question 15: **Correct**

A critical application runs on an Amazon EC2 instance. A Developer has configured a custom Amazon CloudWatch metric that monitors application availability with a data granularity of 1 second. The Developer must be notified within 30 seconds if the application experiences any issues.

What should the Developer do to meet this requirement?

* **Specify an Amazon SNS topic for alarms when issuing the put-metric-data AWS CLI command.**
* **Use Amazon CloudWatch Logs Insights and trigger an Amazon Eventbridge rule to send a notification.**
* **Configure a high-resolution CloudWatch alarm and use Amazon SNS to send the alert.**

**(Correct)**

* **Use a default CloudWatch metric, configure an alarm, and use Amazon SNS to send the alert.**

**Explanation**

If you set an alarm on a high-resolution metric, you can specify a high-resolution alarm with a period of 10 seconds or 30 seconds, or you can set a regular alarm with a period of any multiple of 60 seconds. There is a higher charge for high-resolution alarms.

Amazon SNS can then be used to send notifications based on the CloudWatch alarm.

**CORRECT:**"Configure a high-resolution CloudWatch alarm and use Amazon SNS to send the alert" is the correct answer.

**INCORRECT:** "Specify an Amazon SNS topic for alarms when issuing the put-metric-data AWS CLI command" is incorrect. You cannot specify an SNS topic with this CLI command.

**INCORRECT:** "Use Amazon CloudWatch Logs Insights and trigger an Amazon Eventbridge rule to send a notification" is incorrect. Logs Insights cannot be used for alarms or alerting based on custom CloudWatch metrics.

**INCORRECT:** "Use a default CloudWatch metric, configure an alarm, and use Amazon SNS to send the alert" is incorrect. There is no default metric that would monitor the application uptime and the resolution would be lower.

**References:**

<https://docs.aws.amazon.com/AmazonCloudWatch/latest/monitoring/AlarmThatSendsEmail.html#high-resolution-alarms>

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

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

Bottom of Form

Top of Form

Question 16: **Correct**

An organization is launching a new service that will use an IoT device. How can secure communication protocols be established over the internet to ensure the security of the IoT devices during the launch?

* **Use IoT Core to provide TLS secured communications to AWS from the IoT devices by issuing X.509 certificates.**
* **Use IoT Greengrass to enable TLS secured communications to AWS from the IoT devices by issuing X.509 certificates.**
* **Use AWS Certificate Manager (ACM) to provide TLS secured communications to IoT devices and deploy X.509 certificates in the IoT environment.**

**(Correct)**

* **Use AWS Private Certificate Authority (CA) to provide TLS secured communications to the IoT devices and deploy X.509 certificates in the IoT environment.**

**Explanation**

AWS Certificate Manager (ACM) is used to provision X.509 certificates for TLS/SSL secured communications. It can be used to create certificates for use with many AWS services and applications. It is compatible with IoT devices and applications such as IoT Core and IoT Greengrass.

**CORRECT:**"Use AWS Certificate Manager (ACM) to provide TLS secured communications to IoT devices and deploy X.509 certificates in the IoT environment" is the correct answer (as explained above.)

**INCORRECT:** "Use AWS Private Certificate Authority (CA) to provide TLS secured communications to the IoT devices and deploy X.509 certificates in the IoT environment" is incorrect. AWS Private Certificate Authority cannot be used over the internet.

**INCORRECT:** "Use IoT Greengrass to enable TLS secured communications to AWS from the IoT devices by issuing X.509 certificates" is incorrect. AWS IoT Greengrass is not a certificate authority.

**INCORRECT:** "Use IoT Core to provide TLS secured communications to AWS from the IoT devices by issuing X.509 certificates" is incorrect. AWS IoT Core is not a certificate authority.

**References**:

<https://docs.aws.amazon.com/iot/latest/developerguide/device-certs-your-own.html>

Bottom of Form

Top of Form

Question 17: **Incorrect**

A developer is running queries on Hive-compatible partitions in Athena using DDL but is facing time out issues. What is the most effective and efficient way to prevent this from continuing to happen?

* **Export the data into DynamoDB to perform queries in a more flexible schema.**
* **Use the MSCK REPAIR TABLE command to update the metadata in the catalog.**

**(Correct)**

* **Export the data into a JSON document to clean any errors and upload the cleaned data into S3.**
* **Use the ALTER TABLE ADD PARTITION command to update the column names.**

**(Incorrect)**

**Explanation**

The MSCK REPAIR TABLE command scans Amazon S3 for Hive compatible partitions that were added to the file system after the table was created. It compares the partitions in the table metadata and the partitions in S3. If new partitions are present in the S3 location that you specified when you created the table, it adds those partitions to the metadata and to the Athena table. MSK REPAIR TABLE can work better than DDL if have more than a few thousand partitions and DDL is facing timeout issues.

**CORRECT:**"Use the MSCK REPAIR TABLE command to update the metadata in the catalog" is the correct answer (as explained above.)

**INCORRECT:** "Use the ALTER TABLE ADD PARTITION command to update the column names" is incorrect. This DDL command is used to add one or more partition columns.

**INCORRECT:** "Export the data into DynamoDB to perform queries in a more flexible schema" is incorrect. DynamoDB is a NoSQL table.

**INCORRECT:** "Export the data into a JSON document to clean any errors and upload the cleaned data into S3" is incorrect. This is not an efficient or effective way to reduce DDL time out issues.

**References**:

<https://docs.aws.amazon.com/athena/latest/ug/msck-repair-table.html>

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

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

Bottom of Form

Top of Form

Question 18: **Incorrect**

A developer is working on an application that must save hundreds of sensitive files. The application needs to encrypt each file using a unique key before storing it.

What should the developer do to implement this in the application?

* **Utilize the AWS Key Management Service (KMS) Encrypt API to secure the data, storing both the encrypted data key and the actual data.**

**(Incorrect)**

* **Use a crypto library to generate a unique encryption key for the application, employ the encryption key to secure the data, and store the encrypted data.**
* **Upload the data to an Amazon S3 bucket employing server-side encryption with a key from AWS KMS.**
* **Use the AWS KMS GenerateDataKey API to acquire a data key, use the data key to encrypt the data, and store both the encrypted data key and the data.**

**(Correct)**

**Explanation**

The AWS KMS GenerateDataKey API returns a plaintext version of the key and a copy of the key encrypted under a KMS key. The application can use the plaintext key to encrypt data, and then discard it from memory as soon as possible to reduce potential exposure.

**CORRECT:**"Use the AWS KMS GenerateDataKey API to acquire a data key, use the data key to encrypt the data, and store both the encrypted data key and the data" is the correct answer (as explained above.)

**INCORRECT:** "Utilize the AWS Key Management Service (KMS) Encrypt API to secure the data, storing both the encrypted data key and the actual data" is incorrect.

The AWS KMS Encrypt API could be used, but it doesn't generate a unique key for each file. It encrypts data under a specified KMS key, which isn't the requirement here.

**INCORRECT:** "Use a crypto library to generate a unique encryption key for the application, employ the encryption key to secure the data, and store the encrypted data" is incorrect.

Although a cryptography library can be used to generate a key and encrypt the data, it puts the onus of secure key management on the application. It's a less secure and more complex solution compared to using KMS.

**INCORRECT:** "Upload the data to an Amazon S3 bucket employing server-side encryption with a key from AWS KMS" is incorrect.

The requirement is to encrypt data within the application prior to storage. Uploading to an S3 bucket using server-side encryption doesn't fulfill this requirement.

**References:**

<https://docs.aws.amazon.com/kms/latest/APIReference/API_GenerateDataKey.html>

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

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

Bottom of Form

Top of Form

Question 19: **Incorrect**

A Developer is creating a new web application that will be deployed using AWS Elastic Beanstalk from the AWS Management Console. The Developer is about to create a source bundle which will be uploaded using the console.

Which of the following are valid requirements for creating the source bundle? (Select TWO.)

* **Must not include a parent folder or top-level directory.**

**(Correct)**

* **Must not exceed 512 MB.**

**(Correct)**

* **Must include the cron.yaml file.**
* **Must consist of one or more ZIP files.**

**(Incorrect)**

* **Must include a parent folder or top-level directory.**

**(Incorrect)**

**Explanation**

When you use the AWS Elastic Beanstalk console to deploy a new application or an application version, you'll need to upload a source bundle. Your source bundle must meet the following requirements:

  •  Consist of a single ZIP file or WAR file (you can include multiple WAR files inside your ZIP file)

  •  Not exceed 512 MB

  •  Not include a parent folder or top-level directory (subdirectories are fine)

If you want to deploy a worker application that processes periodic background tasks, your application source bundle must also include a cron.yaml file, but in other cases it is not required.

**CORRECT:**"Must not include a parent folder or top-level directory" is a correct answer.

**CORRECT:**"Must not exceed 512 MB" is also a correct answer.

**INCORRECT:** "Must include the cron.yaml file" is incorrect. As mentioned above, this is not required in all cases.

**INCORRECT:** "Must include a parent folder or top-level directory" is incorrect. A parent folder or top-level directory must NOT be included.

**INCORRECT:** "Must consist of one or more ZIP files" is incorrect. You bundle into a single ZIP or WAR file.

**References:**

<https://docs.aws.amazon.com/elasticbeanstalk/latest/dg/applications-sourcebundle.html>

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

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

Bottom of Form

Top of Form

Question 20: **Correct**

An e-commerce web application that shares session state on-premises is being migrated to AWS. The application must be fault tolerant, natively highly scalable, and any service interruption should not affect the user experience.

What is the best option to store the session state?

* **Enable session stickiness using elastic load balancers**
* **Store the session state in Amazon S3**
* **Store the session state in Amazon CloudFront**
* **Store the session state in Amazon ElastiCache**

**(Correct)**

**Explanation**

There are various ways to manage user sessions including storing those sessions locally to the node responding to the HTTP request or designating a layer in your architecture which can store those sessions in a scalable and robust manner. Common approaches used include utilizing Sticky sessions or using a Distributed Cache for your session management.

In this scenario, a distributed cache is suitable for storing session state data. ElastiCache can perform this role and with the Redis engine replication is also supported. Therefore, the solution is fault-tolerant and natively highly scalable.

**CORRECT:**"Store the session state in Amazon ElastiCache" is the correct answer.

**INCORRECT:** "Store the session state in Amazon CloudFront" is incorrect as CloudFront is not suitable for storing session state data, it is used for caching content for better global performance.

**INCORRECT:** "Store the session state in Amazon S3" is incorrect as though you can store session data in Amazon S3 and replicate the data to another bucket, this would result in a service interruption if the S3 bucket was not accessible.

**INCORRECT:** "Enable session stickiness using elastic load balancers" is incorrect as this feature directs sessions from a specific client to a specific EC2 instances. Therefore, if the instance fails the user must be redirected to another EC2 instance and the session state data would be lost.

**References:**

<https://aws.amazon.com/caching/session-management/>

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

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

Bottom of Form

Top of Form

Question 21: **Correct**

A company runs many microservices applications that use Docker containers. The company are planning to migrate the containers to Amazon ECS. The workloads are highly variable and therefore the company prefers to be charged per running task.

Which solution is the BEST fit for the company’s requirements?

* **Amazon ECS with the Fargate launch type**

**(Correct)**

* **An Amazon ECS Cluster with Auto Scaling**
* **An Amazon ECS Service with Auto Scaling**
* **Amazon ECS with the EC2 launch type**

**Explanation**

The key requirement is that the company should be charged per running task. Therefore, the best answer is to use Amazon ECS with the Fargate launch type as with this model AWS charge you for running tasks rather than running container instances.

The Fargate launch type allows you to run your containerized applications without the need to provision and manage the backend infrastructure. You just register your task definition and Fargate launches the container for you. The Fargate Launch Type is a serverless infrastructure managed by AWS.

**CORRECT:**"Amazon ECS with the Fargate launch type" is the correct answer.

**INCORRECT:** "Amazon ECS with the EC2 launch type" is incorrect as with this launch type you pay for running container instances (EC2 instances).

**INCORRECT:** "An Amazon ECS Service with Auto Scaling" is incorrect as this does not specify the launch type. You can run an ECS Service on the Fargate or EC2 launch types.

**INCORRECT:** "An Amazon ECS Cluster with Auto Scaling" is incorrect as this does not specify the launch type. You can run an ECS Cluster on the Fargate or EC2 launch types.

**References:**

<https://docs.aws.amazon.com/AmazonECS/latest/developerguide/AWS_Fargate.html>

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

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

Bottom of Form

Top of Form

Question 22: **Correct**

To reduce the cost of API actions performed on an Amazon SQS queue, a Developer has decided to implement long polling. Which of the following modifications should the Developer make to the API actions?

* **Set the SetQueueAttributes with a MessageRetentionPeriod of 60**
* **Set the ReceiveMessage API with a WaitTimeSeconds of 20**

**(Correct)**

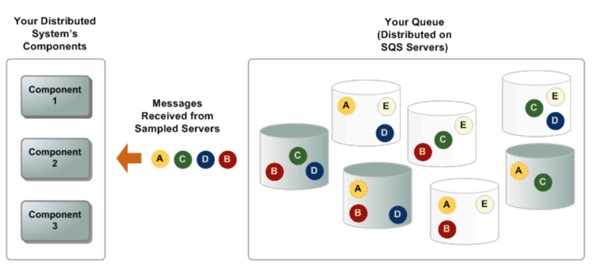
* **Set the SetQueueAttributes API with a DelaySeconds of 20**
* **Set the ReceiveMessage API with a VisibilityTimeout of 30**

**Explanation**

The process of consuming messages from a queue depends on whether you use short or long polling. By default, Amazon SQS uses short polling, querying only a subset of its servers (based on a weighted random distribution) to determine whether any messages are available for a response. You can use long polling to reduce your costs while allowing your consumers to receive messages as soon as they arrive in the queue.

When you consume messages from a queue using short polling, Amazon SQS samples a subset of its servers (based on a weighted random distribution) and returns messages from only those servers. Thus, a particular ReceiveMessage request might not return all of your messages. However, if you have fewer than 1,000 messages in your queue, a subsequent request will return your messages. If you keep consuming from your queues, Amazon SQS samples all of its servers, and you receive all of your messages.

The following diagram shows the short-polling behavior of messages returned from a standard queue after one of your system components makes a receive request. Amazon SQS samples several of its servers (in gray) and returns messages A, C, D, and B from these servers. Message E isn't returned for this request but is returned for a subsequent request.



When the wait time for the ReceiveMessage API action is greater than 0, *long polling* is in effect. The maximum long polling wait time is 20 seconds. Long polling helps reduce the cost of using Amazon SQS by eliminating the number of empty responses (when there are no messages available for a ReceiveMessage request) and false empty responses (when messages are available but aren't included in a response).

Long polling occurs when the WaitTimeSeconds parameter of a ReceiveMessage request is set to a value greater than 0 in one of two ways:

* The ReceiveMessage call sets WaitTimeSeconds to a value greater than 0.
* The ReceiveMessage call doesn’t set WaitTimeSeconds, but the queue attribute ReceiveMessageWaitTimeSeconds is set to a value greater than 0.

**Therefore, the Developer should**set the ReceiveMessage API with a WaitTimeSeconds of 20.

**CORRECT:**"Set the ReceiveMessage API with a WaitTimeSeconds of 20" is the correct answer.

**INCORRECT:** "Set the SetQueueAttributes API with a DelaySeconds of 20" is incorrect as this would be used to configure a delay queue where the delivery of messages in the queue is delayed.

**INCORRECT:** "Set the ReceiveMessage API with a VisibilityTimeout of 30" is incorrect as this would configure the visibility timeout which is the length of time a message that has been received is invisible.

**INCORRECT:** "Set the SetQueueAttributes with a MessageRetentionPeriod of 60" is incorrect as this would configure how long messages are retained in the queue.

**References:**

<https://docs.aws.amazon.com/AWSSimpleQueueService/latest/SQSDeveloperGuide/sqs-short-and-long-polling.html>

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

<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/>

Bottom of Form

Top of Form

Question 23: **Correct**

A company has a large Amazon DynamoDB table which they scan periodically so they can analyze several attributes. The scans are consuming a lot of provisioned throughput. What technique can a Developer use to minimize the impact of the scan on the table's provisioned throughput?

* **Define a range key on the table**
* **Prewarm the table by updating all items**
* **Use parallel scans**
* **Set a smaller page size for the scan**

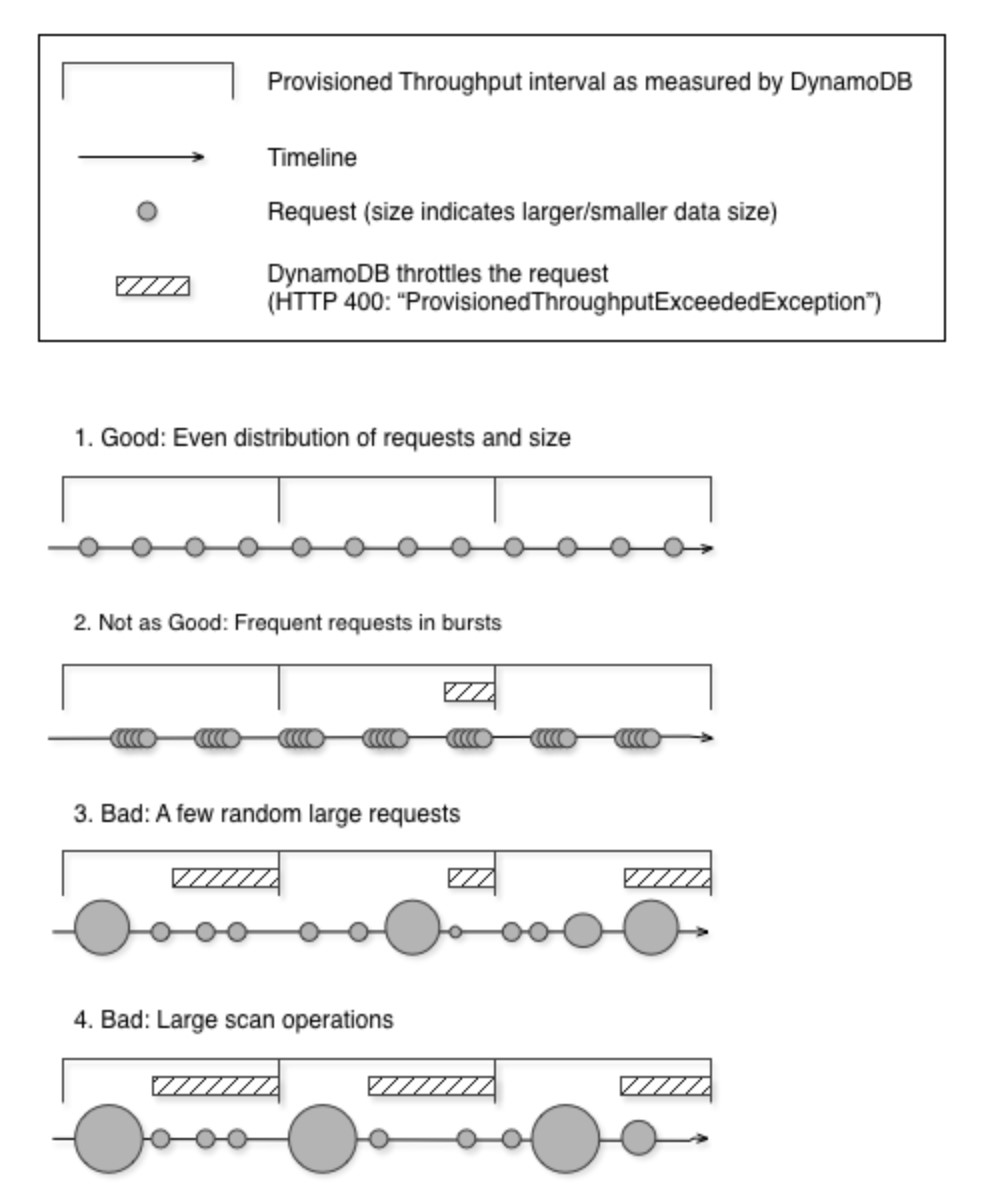
**(Correct)**

**Explanation**

In general, Scan operations are less efficient than other operations in DynamoDB. A Scan operation always scans the entire table or secondary index. It then filters out values to provide the result you want, essentially adding the extra step of removing data from the result set.

If possible, you should avoid using a Scan operation on a large table or index with a filter that removes many results. Also, as a table or index grows, the Scan operation slows. The Scan operation examines every item for the requested values and can use up the provisioned throughput for a large table or index in a single operation.

The following diagram illustrates the impact of a sudden spike of capacity unit usage by Query and Scan operations, and its impact on your other requests against the same table.



Instead of using a large Scan operation, you can use the following techniques to minimize the impact of a scan on a table's provisioned throughput.

**Reduce page size**

Because a Scan operation reads an entire page (by default, 1 MB), you can reduce the impact of the scan operation by setting a smaller page size. The Scan operation provides a *Limit* parameter that you can use to set the page size for your request. Each Query or Scan request that has a smaller page size uses fewer read operations and creates a "pause" between each request.

**Isolate scan operations**

DynamoDB is designed for easy scalability. As a result, an application can create tables for distinct purposes, possibly even duplicating content across several tables. You want to perform scans on a table that is not taking "mission-critical" traffic. Some applications handle this load by rotating traffic hourly between two tables—one for critical traffic, and one for bookkeeping. Other applications can do this by performing every write on two tables: a "mission-critical" table, and a "shadow" table.

Therefore, the best option to reduce the impact of the scan on the table's provisioned throughput is to set a smaller page size for the scan.

**CORRECT:**"Set a smaller page size for the scan" is the correct answer.

**INCORRECT:** "Use parallel scans" is incorrect as this will return results faster but place more burden on the table’s provisioned throughput.

**INCORRECT:** "Define a range key on the table" is incorrect. A range key is a composite key that includes the hash key and another attribute. This is of limited use in this scenario as the table is being scanned to analyze multiple attributes.

**INCORRECT:** "Prewarm the table by updating all items" is incorrect as updating all items would incur significant costs in terms of provisioned throughput and would not be advantageous.

**References:**

<https://docs.aws.amazon.com/amazondynamodb/latest/developerguide/bp-query-scan.html>

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

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

Bottom of Form

Top of Form

Question 24: **Correct**

A developer is writing an application for a company. The program needs to access and read the file named "secret-data.xlsx" located in the root directory of an Amazon S3 bucket named "DATA-BUCKET". The company's security policies mandate the enforcement of the principle of least privilege for the IAM policy associated with the application.

Which IAM policy statement will comply with these security stipulations?

* **{"Effect": "Allow", "Action": "s3:ListBucket", "Resource": "arn:aws:s3:::DATA-BUCKET"}**
* **{"Effect": "Allow", "Action": "s3:GetObject", "Resource": "arn:aws:s3:::DATA-BUCKET/secret-data.xlsx"}**

**(Correct)**

* **{"Effect": "Allow", "Action": "s3:\*", "Resource": "arn:aws:s3:::DATA-BUCKET/\*"}**
* **{"Effect": "Allow", "Action": "s3:GetObject", "Resource": "arn:aws:s3:::DATA-BUCKET/\*"}**

**Explanation**

This statement provides the minimal permission necessary for the application to read the specific "secret-data.xlsx" file from the specified S3 bucket.

**CORRECT:**"{"Effect": "Allow", "Action": "s3:GetObject", "Resource": "arn:aws:s3:::DATA-BUCKET/secret-data.xlsx"}" is the correct answer (as explained above.)

**INCORRECT:** "{"Effect": "Allow", "Action": "s3:\*", "Resource": "arn:aws:s3:::DATA-BUCKET/\*"}" is incorrect.

This policy statement gives permission for all S3 actions on all objects in the bucket, which goes against the principle of least privilege.

**INCORRECT:** "{"Effect": "Allow", "Action": "s3:ListBucket", "Resource": "arn:aws:s3:::DATA-BUCKET"}" is incorrect.

This policy permits the application to list all objects in the bucket, but it doesn't grant read permission for "secret-data.xlsx".

**INCORRECT:** "{"Effect": "Allow", "Action": "s3:GetObject", "Resource": "arn:aws:s3:::DATA-BUCKET/\*"}" is incorrect.

This statement provides read access for all objects in the bucket, not just the "secret-data.xlsx" file, contradicting the principle of least privilege.

**References:**

<https://docs.aws.amazon.com/AmazonS3/latest/userguide/example-bucket-policies.html>

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

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

Bottom of Form

Top of Form

Question 25: **Correct**

An organization is hosting a website on an Amazon EC2 instance in a public subnet. The website should allow public access for HTTPS traffic on TCP port 443 but should only accept SSH traffic on TCP port 22 from a corporate address range accessible over a VPN.

Which security group configuration will support both requirements?

* **Allow traffic to both port 443 and port 22 from 0.0.0.0/0 and 192.168.0.0/16.**
* **Allow traffic to port 22 from 0.0.0.0/0 and allow traffic to port 443 from 192.168.0.0/16.**
* **Allow traffic to port 443 from 0.0.0.0/0 and allow traffic to port 22 from 192.168.0.0/16.**

**(Correct)**

* **Allow traffic to both port 443 and port 22 from the VPC CIDR block.**

**Explanation**

Allowing traffic from 0.0.0.0/0 to port 443 will allow any traffic from the internet to access the website. Limiting the IP address to 192.168.0.0/16 for port 22 will only allow local organizational traffic.

**CORRECT**: "Allow traffic to port 443 from 0.0.0.0/0 and allow traffic to port 22 from 192.168.0.0/16" is the correct answer (as explained above.)

**INCORRECT**: “Allow traffic to port 22 from 0.0.0.0/0 and allow traffic to port 443 from 192.168.0.0/16" is incorrect. This will allow traffic from the Internet to port 22 and allow traffic to port 443 from the corporate address block only (192.168.0.0/16).

**INCORRECT**: "Allow traffic to both port 443 and port 22 from the VPC CIDR block" is incorrect. This would not satisfy either requirement as internet-based users will not be able to access the website and corporate users will not be able to manage the instance via SSH.

**INCORRECT**: "Allow traffic to both port 443 and port 22 from 0.0.0.0/0 and 192.168.0.0/16" is incorrect. This does not satisfy the requirement to restrict access to port 22 to only the corporate address block.

**References**:

<https://docs.aws.amazon.com/vpc/latest/userguide/VPC_SecurityGroups.html>

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

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

Bottom of Form

Top of Form

Question 26: **Correct**

An organization handles data that requires high availability in its relational database. The main headquarters for the organization is in Virginia with smaller offices located in California. The main headquarters uses the data more frequently than the smaller offices. How should the developer configure their databases to meet high availability standards?

* **Create an Aurora database with the primary database in Virginia and specify the failover to the Aurora replica in another AZ in Virginia.**

**(Correct)**

* **Create an Aurora database with the primary database in Virginia and specify the failover to the Aurora replica in another AZ in California.**
* **Create an Athena database with the primary database in Virginia and specify the failover to the Athena replica in another AZ in Virginia.**
* **Create a DynamoDB database with the primary database in Virginia and specify the failover to the DynamoDB replica in another AZ in Virginia.**

**Explanation**

Aurora is a relational database that provides high availability by allowing customers to create up to 15 database replications in different Availability Zones. It also allows you to specify which Aurora replica can be promoted to the primary database should the primary database become unavailable. Selecting the AZ that is closest to the main headquarters should not negatively impact the smaller offices but changing the primary database to California could negatively impact the main headquarters.

**CORRECT:**"Create an Aurora database with the primary database in Virginia and specify the failover to the Aurora replica in another AZ in Virginia" is the correct answer (as explained above.)

**INCORRECT:** "Create an Aurora database with the primary database in Virginia and specify the failover to the Aurora replica in another AZ in California" is incorrect. It could create some latency issues for the main headquarters in Virginia.

**INCORRECT:** "Create a DynamoDB database with the primary database in Virginia and specify the failover to the DynamoDB replica in another AZ in Virginia" is incorrect. DynamoDB is not a relational database.

**INCORRECT:** "Create an Athena database with the primary database in Virginia and specify the failover to the Athena replica in another AZ in Virginia" is incorrect. Athena analyzes data but is not a database service.

**References**:

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

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

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

Bottom of Form

Top of Form

Question 27: **Correct**

A Developer is creating a serverless application that uses an Amazon DynamoDB table. The application must make idempotent, all-or-nothing operations for multiple groups of write actions.

Which solution will meet these requirements?

* **Update the items in the table using the BatchWriteltem operation and configure idempotency at the table level.**
* **Update the items in the table using the TransactWriteltems operation to group the changes.**

**(Correct)**

* **Enable DynamoDB streams and capture new images. Update the items in the table using the BatchWriteltem.**
* **Create an Amazon SQS FIFO queue and use the SendMessageBatch operation to group the changes.**

**Explanation**

TransactWriteItems is a synchronous and idempotent write operation that groups up to 25 write actions in a single all-or-nothing operation. These actions can target up to 25 distinct items in one or more DynamoDB tables within the same AWS account and in the same Region. The aggregate size of the items in the transaction cannot exceed 4 MB. The actions are completed atomically so that either all of them succeed or none of them succeeds.

A TransactWriteItems operation differs from a BatchWriteItem operation in that all the actions it contains must be completed successfully, or no changes are made at all. With a BatchWriteItem operation, it is possible that only some of the actions in the batch succeed while the others do not.

**CORRECT:**"Update the items in the table using the TransactWriteltems operation to group the changes" is the correct answer.

**INCORRECT:** "Update the items in the table using the BatchWriteltem operation and configure idempotency at the table level" is incorrect. As explained above, the TransactWriteItems operation must be used.

**INCORRECT:** "Enable DynamoDB streams and capture new images. Update the items in the table using the BatchWriteltem" is incorrect. DynamoDB streams will not assist with making idempotent write operations.

**INCORRECT:** "Create an Amazon SQS FIFO queue and use the SendMessageBatch operation to group the changes" is incorrect. Amazon SQS should not be used as it does not assist and this solution is supposed to use a DynamoDB table

**References:**

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

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

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

Bottom of Form

Top of Form

Question 28: **Correct**

A web application runs on a fleet of Amazon EC2 instances in an Auto Scaling group behind an Application Load Balancer (ALB). A developer needs a store for session data so it can be reliably served across multiple requests.

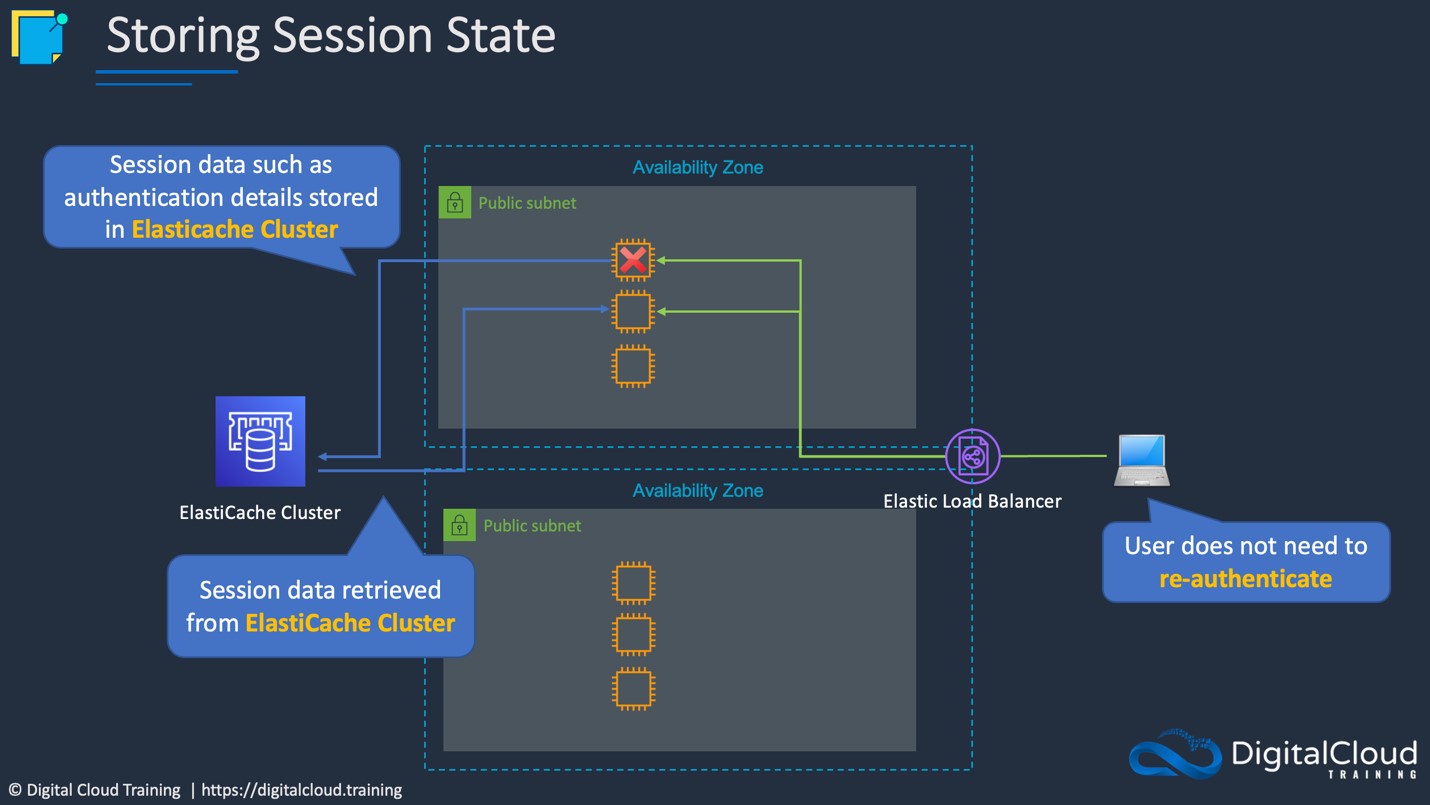
Where is the best place to store the session data?

* **Write the data to the root of the filesystem.**
* **Write the data to the local instance store volumes.**
* **Write the data to a shared Amazon EBS volume.**
* **Write the data to an Amazon ElastiCache cluster.**

**(Correct)**

**Explanation**

ElastiCache is a good solution for storing session state data as it has very low latency and high performance. DynamoDB is often used for the same purpose. In this case the session data can be written to the ElastiCache cluster and can then be easily retrieved from subsequent sessions on the same or a different EC2 instance. This decouples the data from the individual instance so if an instance fails, the data is not lost.



**CORRECT:**"Write the data to an Amazon ElastiCache cluster" is the correct answer (as explained above.)

**INCORRECT:** "Write the data to a shared Amazon EBS volume" is incorrect.

You cannot share an EBS volume except under specific circumstances and even then, you must share the volume from an EC2 instance which could fail.

**INCORRECT:** "Write the data to the root of the filesystem" is incorrect.

This will result in data loss if the instance fails.

**INCORRECT:** "Write the data to the local instance store volumes" is incorrect.

This will result in data loss if the instance fails.

**References:**

<https://aws.amazon.com/caching/session-management/>

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

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

Bottom of Form

Top of Form

Question 29: **Correct**

A developer is looking to verify that redirects are performing as expected. What is the most efficient way that the developer can access the web logs and perform an analysis on them?

* **Store the logs in EBS and use Athena to run SQL queries.**
* **Store the logs in a S3 bucket and use Athena to run SQL queries.**

**(Correct)**

* **Store the logs in an Instance Store and use Athena to run SQL queries.**
* **Store the logs in EFS and use Athena to run SQL queries.**

**Explanation**

Logs can be stored in an S3 bucket to be retained for review and inspection. Athena is an interactive query service that can work directly with S3 and run ad-hoc SQL queries.

**CORRECT:**"Store the logs in a S3 bucket and use Athena to run SQL queries" is the correct answer (as explained above.)

**INCORRECT:** "Store the logs in EBS and use Athena to run SQL queries" is incorrect. As explained above log records are stored in S3 and Athena is compatible to perform queries directly in S3 buckets.

**INCORRECT:** "Store the logs in an Instance Store and use Athena to run SQL queries" is incorrect. Instance stores are ephemeral and temporarily store data for EC2 instances. They cannot be used with Athena.

**INCORRECT:** "Store the logs in EFS and use Athena to run SQL queries" is incorrect. As explained above log records are stored in S3 and Athena is compatible to perform queries directly in S3 buckets.

**References**:

<https://docs.aws.amazon.com/athena/latest/ug/what-is.html>

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

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

Bottom of Form

Top of Form

Question 30: **Incorrect**

A startup is developing a prototype for a news aggregator application. This application will display the latest news for a specific industry and provide a RESTful API endpoint that clients can invoke. Where feasible, the application should leverage AWS's caching features to reduce the load on the backend service. The backend of the application is expected to handle a modest amount of traffic, primarily during testing periods.

Which method would be the most cost-effective for the developer to implement this REST endpoint?

* **Deploy an AWS Elastic Beanstalk application and use Amazon DynamoDB for caching services.**
* **Launch an Amazon EC2 instance, host the application on it, and employ Amazon ElastiCache for caching purposes.**

**(Incorrect)**

* **Construct an Amazon ECS service for the application, using Amazon ElastiCache for caching.**
* **Utilize AWS API Gateway with an AWS Lambda function as the backend and enable caching in API Gateway.**

**(Correct)**

**Explanation**

AWS API Gateway, combined with AWS Lambda, offers a serverless solution where costs are directly related to usage. API Gateway also provides a caching feature which can help reduce the load on the backend.

**CORRECT:**"Utilize AWS API Gateway with an AWS Lambda function as the backend and enable caching in API Gateway" is the correct answer (as explained above.)

**INCORRECT:** "Launch an Amazon EC2 instance, host the application on it, and employ Amazon ElastiCache for caching purposes" is incorrect.

Using an Amazon EC2 instance would incur constant costs, regardless of whether the instance is in use or idle. Additionally, employing AWS ElastiCache could increase the costs unnecessarily, especially for a prototype application.

**INCORRECT:** "Construct an Amazon ECS service for the application, using Amazon ElastiCache for caching" is incorrect.

Amazon ECS is overkill for a simple prototype application like this. It requires more resources and management than a serverless solution.

**INCORRECT:** "Deploy an AWS Elastic Beanstalk application and use Amazon DynamoDB for caching services" is incorrect.

While AWS Elastic Beanstalk simplifies application deployment, it does not inherently provide caching capabilities. Amazon DynamoDB is a database service, not a caching service, making it unsuitable for this requirement.

**References:**

<https://docs.aws.amazon.com/apigateway/latest/developerguide/api-gateway-caching.html>

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

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

Bottom of Form

Top of Form

Question 31: **Incorrect**

A Developer is writing code to run in a cron job on an Amazon EC2 instance that sends status information about the application to Amazon CloudWatch.

Which method should the Developer use?

* **Use the unified CloudWatch agent to publish custom metrics.**

**(Incorrect)**

* **Use the AWS CLI put-metric-data command.**

**(Correct)**

* **Use the CloudWatch console with detailed monitoring.**
* **Use the AWS CLI put-metric-alarm command.**

**Explanation**

The put-metric-data command publishes metric data points to Amazon CloudWatch. CloudWatch associates the data points with the specified metric. If the specified metric does not exist, CloudWatch creates the metric.

**CORRECT:**"Use the AWS CLI put-metric-data command" is the correct answer.

**INCORRECT:** "Use the AWS CLI put-metric-alarm command" is incorrect. This command creates or updates an alarm and associates it with the specified metric, metric math expression, or anomaly detection model.

**INCORRECT:** "Use the unified CloudWatch agent to publish custom metrics" is incorrect. It is not necessary to use the unified CloudWatch agent. In this case the Developer can use the AWS CLI with the cron job.

**INCORRECT:** "Use the CloudWatch console with detailed monitoring" is incorrect. You cannot collect custom metric data using the CloudWatch console with detailed monitoring. Detailed monitoring sends data at 1-minute rather than 5-minute frequencies but will not collect custom data.

**References:**

<https://docs.aws.amazon.com/cli/latest/reference/cloudwatch/put-metric-data.html>

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

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

Bottom of Form

Top of Form

Question 32: **Correct**

The Lambda function needs to write this data to an Amazon DynamoDB table. After deploying the function, the developer notices that the write operations to the DynamoDB table occasionally fail due to throttling.

What should the developer do to reduce the likelihood of these throttling issues without significantly over-provisioning the DynamoDB table's write capacity?

* **Convert the DynamoDB table to an on-demand capacity mode to automatically adjust its write capacity to match the workload.**
* **Implement exponential backoff in the Lambda function's error handling code to retry failed write operations.**

**(Correct)**

* **Use Amazon SQS to queue data before writing it to the DynamoDB table, ensuring a steady and controlled write rate.**
* **Increase the Lambda function's timeout setting to allow for retries in case of throttling.**

**Explanation**

Implementing exponential backoff is a best practice for handling retries in distributed systems, especially when dealing with AWS services like DynamoDB. Exponential backoff involves gradually increasing the delay between retry attempts to reduce the load on the service and the likelihood of subsequent throttling.

This approach allows the Lambda function to manage intermittent spikes more effectively in demand without requiring significant over-provisioning of the DynamoDB table's write capacity.

**CORRECT:**"Implement exponential backoff in the Lambda function's error handling code to retry failed write operations" is the correct answer (as explained above.)

**INCORRECT:** "Increase the Lambda function's timeout setting to allow for retries in case of throttling" is incorrect.

This issue is not directly related to handling throttling issues; increasing the timeout setting allows the function to run longer but does not address the root cause of the throttling.

**INCORRECT:** "Convert the DynamoDB table to an on-demand capacity mode to automatically adjust its write capacity to match the workload" is incorrect.

This solution could potentially solve the issue by automatically scaling the table's capacity, but it might lead to higher costs due to uncontrolled scaling and does not represent a coding or architectural strategy to manage write throughput efficiently.

**INCORRECT:** "Use Amazon SQS to queue data before writing it to the DynamoDB table, ensuring a steady and controlled write rate" is incorrect.

This solution introduces additional complexity and potential latency without directly addressing the issue of handling write retries when throttling occurs. While SQS can help decouple components and manage load, the key to solving the immediate problem of write throttling is to implement a retry strategy that respects DynamoDB's adaptive capacity and avoids overwhelming the table with rapid, repeated write requests.

**References:**

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

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

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

Bottom of Form

Top of Form

Question 33: **Correct**

An application uses an Amazon DynamoDB table that is 50 GB in size and provisioned with 10,000 read capacity units (RCUs) per second. The table must be scanned during non-peak hours when normal traffic consumes around 5,000 RCUs. The Developer must scan the whole table in the shortest possible time whilst ensuring the normal workload is not affected.

How would the Developer optimize this scan cost-effectively?

* **Increase read capacity units during the scan operation.**
* **Use sequential scans and apply a FilterExpression.**
* **Use the Parallel Scan API operation and limit the rate.**

**(Correct)**

* **Use sequential scans and set the ConsistentRead parameter to false.**

**Explanation**

To make the most of the table’s provisioned throughput, the Developer can use the Parallel Scan API operation so that the scan is distributed across the table’s partitions. This will help to optimize the scan to complete in the fastest possible time. However, the Developer will also need to apply rate limiting to ensure that the scan does not affect normal workloads.

**CORRECT:**"Use the Parallel Scan API operation and limit the rate" is the correct answer.

**INCORRECT:** "Use sequential scans and apply a FilterExpression" is incorrect. A FilterExpression is a string that contains conditions that DynamoDB applies after the Scan operation, but before the data is returned to you. This will not assist with speeding up the scan or preventing it from affecting normal workloads.

**INCORRECT:** "Increase read capacity units during the scan operation" is incorrect. There are already more RCUs provisioned than are needed during the non-peak hours. The key here is to use what is available for cost-effectiveness whilst ensuing normal workloads are not affected.

**INCORRECT:** "Use sequential scans and set the ConsistentRead parameter to false" is incorrect. This setting would turn off consistent reads making the scan eventually consistent. This will not satisfy the requirements of the question.

**References:**

[https://aws.amazon.com/blogs/Developer/rate-limited-scans-in-amazon-dynamodb/](https://aws.amazon.com/blogs/developer/rate-limited-scans-in-amazon-dynamodb/)

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

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

Bottom of Form

Top of Form

Question 34: **Correct**

A company is deploying a microservices application on AWS Fargate using Amazon ECS. The application has environment variables that must be passed to a container for the application to initialize.

How should the environment variables be passed to the container?

* **Use advanced container definition parameters and define environment variables under the environment parameter within the service definition.**
* **Use standard container definition parameters and define environment variables under the secrets parameter within the task definition.**
* **Use advanced container definition parameters and define environment variables under the environment parameter within the task definition.**

**(Correct)**

* **Use standard container definition parameters and define environment variables under the WorkingDirectory parameter within the service definition.**

**Explanation**

When you register a task definition, you must specify a list of container definitions that are passed to the Docker daemon on a container instance.

The developer should use advanced container definition parameters and define environment variables to pass to the container.

**CORRECT:**"Use advanced container definition parameters and define environment variables under the environment parameter within the task definition" is the correct answer (as explained above.)

**INCORRECT:** "Use advanced container definition parameters and define environment variables under the environment parameter within the service definition" is incorrect.

The task definition is the correct place to define the environment variables to pass to the container.

**INCORRECT:** "Use standard container definition parameters and define environment variables under the secrets parameter within the task definition" is incorrect.

Advanced container definition parameters must be used to pass the environment variables to the container. The environment parameter should also be used.

**INCORRECT:** "Use standard container definition parameters and define environment variables under the WorkingDirectory parameter within the service definition" is incorrect.

Advanced container definition parameters must be used to pass the environment variables to the container. The environment parameter should also be used.

**References:**

<https://docs.aws.amazon.com/AmazonECS/latest/userguide/task_definition_parameters.html#container_definition_environment>

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

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

Bottom of Form

Top of Form

Question 35: **Incorrect**

An application deployed on AWS Elastic Beanstalk experienced increased error rates during deployments of new application versions, resulting in service degradation for users. The Development team believes that this is because of the reduction in capacity during the deployment steps. The team would like to change the deployment policy configuration of the environment to an option that maintains full capacity during deployment while using the existing instances.

Which deployment policy will meet these requirements while using the existing instances?

* **Immutable**

**(Incorrect)**

* **Rolling with additional batch**

**(Correct)**

* **All at once**
* **Rolling**

**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 rolling with additional batch launches a new batch to ensure capacity is not reduced and then updates the existing instances. Therefore, this is the best option to use for these requirements.

**CORRECT:**“Rolling with additional batch” is the correct answer.

**INCORRECT:** “Rolling” is incorrect as this will only use the existing instances without introducing an extra batch and therefore this will reduce the capacity of the application while the updates are taking place.

**INCORRECT:** “All at once” is incorrect as this will run the updates on all instances at the same time causing a total outage.

**INCORRECT:** “Immutable” is incorrect as this installs the updates on new instances, not existing instances.

**References:**

<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/>

Bottom of Form

Top of Form

Question 36: **Incorrect**

A developer has deployed an application on AWS Lambda. The application uses Python and must generate and then upload a file to an Amazon S3 bucket. The developer must implement the upload functionality with the least possible change to the application code.

Which solution BEST meets these requirements?

* **Use the AWS SDK for Python that is installed in the Lambda execution environment.**

**(Correct)**

* **Include the AWS SDK for Python in the Lambda function code.**
* **Use the AWS CLI that is installed in the Lambda execution environment.**
* **Make an HTTP request directly to the S3 API to upload the file.**

**(Incorrect)**

**Explanation**

The best practice for Lambda development is to bundle all dependencies used by your Lambda function, including the AWS SDK. However, since this question specifically requests that the least possible changes are made to the application code, the developer can instead use the SDK for Python that is installed in the Lambda environment to upload the file to Amazon S3.

**CORRECT:**"Use the AWS SDK for Python that is installed in the Lambda execution environment" is the correct answer (as explained above.)

**INCORRECT:** "Include the AWS SDK for Python in the Lambda function code" is incorrect.

This is the best practice for deployment. However, in this case the developer must minimize changes to code and including the SDK as a dependency in the code would require potential updates to existing Python code.

**INCORRECT:** "Make an HTTP request directly to the S3 API to upload the file" is incorrect.

AWS supports uploads to S3 using the console, AWS SDKs, REST API, and the AWS CLI.

**INCORRECT:** "Use the AWS CLI that is installed in the Lambda execution environment" is incorrect.

The AWS CLI is not installed in the Lambda execution environment.

**References:**

<https://aws.amazon.com/blogs/compute/upcoming-changes-to-the-python-sdk-in-aws-lambda/>

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

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

Bottom of Form

Top of Form

Question 37: **Incorrect**

An ecommerce company manages a storefront that uses an Amazon API Gateway API which exposes an AWS Lambda function. The Lambda functions processes orders and stores the orders in an Amazon RDS for MySQL database. The number of transactions increases sporadically during marketing campaigns, and then goes close to zero during quite times.

How can a developer increase the elasticity of the system MOST cost-effectively?

* **Migrate from Amazon RDS to Amazon Aurora MySQL. Use an Aurora Auto Scaling policy to scale read replicas based on average connections of Aurora Replicas.**

**(Correct)**

* **Migrate from Amazon RDS to Amazon Aurora MySQL. Use an Aurora Auto Scaling policy to scale read replicas based on average CPU utilization.**
* **Create an Amazon SQS queue. Publish transactions to the queue and set the queue to invoke the Lambda function. Set the reserved concurrency of the Lambda function to be equal to the max number of database connections.**

**(Incorrect)**

* **Create an Amazon SNS topic. Publish transactions to the topic configure an SQS queue as a destination. Configure Lambda to process transactions from the queue.**

**Explanation**

The most efficient solution would be to use Aurora Auto Scaling and configure the scaling events to happen based on target metric. The metric to use is **Average connections of Aurora Replicas** which will create a policy based on the average number of connections to Aurora Replicas.

This will ensure that the Aurora replicas scale based on actual numbers of connections to the replicas which will vary based on how busy the storefront is and how many transactions are being processed.

**CORRECT:**"Migrate from Amazon RDS to Amazon Aurora MySQL. Use an Aurora Auto Scaling policy to scale read replicas based on average connections of Aurora Replicas" is the correct answer (as explained above.)

**INCORRECT:** "Migrate from Amazon RDS to Amazon Aurora MySQL. Use an Aurora Auto Scaling policy to scale read replicas based on average CPU utilization" is incorrect.

The better metric to use for this situation would be the number of connections to Aurora Replicas as that is the metric that has the closest correlation to the number of transactions being executed.

**INCORRECT:** "Create an Amazon SNS topic. Publish transactions to the topic configure an SQS queue as a destination. Configure Lambda to process transactions from the queue" is incorrect.

This is highly inefficient. There is no need for an SNS topic in this situation.

**INCORRECT:** "Create an Amazon SQS queue. Publish transactions to the queue and set the queue to invoke the Lambda function. Set the reserved concurrency of the Lambda function to be equal to the max number of database connections" is incorrect.

This would be less cost effective as you would be paying for the reserved concurrency at all times.

**References:**

<https://docs.aws.amazon.com/AmazonRDS/latest/AuroraUserGuide/Aurora.Integrating.AutoScaling.html>

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

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

Bottom of Form

Top of Form

Question 38: **Correct**

A business is providing its clients read-only permissions to items within an Amazon S3 bucket, utilizing IAM permissions to limit access to this S3 bucket. Clients are only permitted to access their specific files. Regulatory compliance necessitates the enforcement of in-transit encryption during communication with Amazon S3.

What solution will fulfill these criteria?

* **Enable the Amazon S3 Transfer Acceleration feature to ensure encryption during transit.**
* **Activate Amazon S3 server-side encryption to enforce encryption during transit.**
* **Assign IAM roles enforcing SSL/TLS encryption to each customer.**
* **Update the S3 bucket policy to include a condition that requires aws:SecureTransport for all actions.**

**(Correct)**

**Explanation**

By adding a condition to the S3 bucket policy that requires aws:SecureTransport, you are mandating that all interactions with the bucket must be encrypted in transit using SSL/TLS.

**CORRECT:**"Update the S3 bucket policy to include a condition that requires aws:SecureTransport for all actions" is the correct answer (as explained above.)

**INCORRECT:** "Activate Amazon S3 server-side encryption to enforce encryption during transit" is incorrect.

Server-side encryption for S3 secures the data at rest, not during transit.

**INCORRECT:** "Assign IAM roles enforcing SSL/TLS encryption to each customer" is incorrect.

Although IAM roles are utilized to manage access to AWS resources, they do not inherently mandate SSL/TLS encryption for interactions with the resources.

**INCORRECT:** "Enable the Amazon S3 Transfer Acceleration feature to ensure encryption during transit" is incorrect.

While Amazon S3 Transfer Acceleration does use SSL/TLS, it is designed to expedite the transfer of data over long distances between a user and an S3 bucket, not to mandate encryption in transit for all interactions.

**References:**

<https://docs.aws.amazon.com/AmazonS3/latest/userguide/example-bucket-policies.html#example-bucket-policies-use-case-2>

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

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

Bottom of Form

Top of Form

Question 39: **Incorrect**

A critical application is hosted on AWS exposed by an HTTP API through Amazon API Gateway. The API is integrated with an AWS Lambda function and the application data is housed in an Amazon RDS for PostgreSQL DB instance, featuring 2 vCPUs and 16 GB of RAM.

The company has been receiving customer complaints about occasional HTTP 500 Internal Server Error responses from some API calls during unpredictable peak usage times. Amazon CloudWatch Logs has recorded "connection limit exceeded" errors. The company wants to ensure resilience in the application, with no unscheduled downtime for the database.

Which solution would best fit these requirements?

* **Implement auto-scaling for the RDS instance based on connection count.**

**(Incorrect)**

* **Double the RAM and vCPUs of the RDS instance.**
* **Use AWS Lambda to create a connection pool for the RDS instance.**
* **Use Amazon RDS Proxy and update the Lambda function to connect to the proxy.**

**(Correct)**

**Explanation**

Amazon RDS Proxy is designed to improve application scalability and resilience by pooling and sharing database connections, reducing the CPU and memory overhead on the database. It handles the burst in connections seamlessly and improves the application's ability to scale.

**CORRECT:**"Use Amazon RDS Proxy and update the Lambda function to connect to the proxy" is the correct answer (as explained above.)

**INCORRECT:** "Double the RAM and vCPUs of the RDS instance" is incorrect.

Merely augmenting the number of vCPUs and RAM for the RDS instance might not resolve the issue as it doesn't directly tackle the problem of too many connections.

**INCORRECT:** "Implement auto-scaling for the RDS instance based on connection count" is incorrect.

Auto-scaling in response to connection count is not a feature provided by RDS.

**INCORRECT:** "Use AWS Lambda to create a connection pool for the RDS instance" is incorrect.

The best solution is to use RDS proxy which is designed for creating a pool of connections. Lambda may not be the best solution for this problem as it could be costly and has a limitation in execution time.

**References:**

<https://aws.amazon.com/blogs/compute/using-amazon-rds-proxy-with-aws-lambda/>

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

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

Bottom of Form

Top of Form

Question 40: **Correct**

A company has an application that logs all information to Amazon S3. Whenever there is a new log file, an AWS Lambda function is invoked to process the log files. The code works, gathering all of the necessary information. However, when checking the Lambda function logs, duplicate entries with the same request ID are found.

What is the BEST explanation for the duplicate entries?

* **There was an S3 outage, which caused duplicate entries of the same log file**
* **The application stopped intermittently and then resumed**
* **The S3 bucket name was specified incorrectly**
* **The Lambda function failed, and the Lambda service retried the invocation with a delay**

**(Correct)**

**Explanation**

From the AWS documentation:

“When an error occurs, your function may be invoked multiple times. Retry behavior varies by error type, client, event source, and invocation type. For example, if you invoke a function asynchronously and it returns an error, Lambda executes the function up to two more times. For more information, see Retry Behavior.

For asynchronous invocation, Lambda adds events to a queue before sending them to your function. If your function does not have enough capacity to keep up with the queue, events may be lost. Occasionally, your function may receive the same event multiple times, even if no error occurs. To retain events that were not processed, configure your function with a dead-letter queue.”

Therefore, the most likely explanation is that the function failed, and Lambda retried the invocation.

**CORRECT:**"The Lambda function failed, and the Lambda service retried the invocation with a delay" is the correct answer.

**INCORRECT:** "The S3 bucket name was specified incorrectly" is incorrect. If this was the case all attempts would fail but this is not the case.

**INCORRECT:** "There was an S3 outage, which caused duplicate entries of the same log file" is incorrect. There cannot be duplicate log files in Amazon S3 as every object must be unique within a bucket. Therefore, if the same log file was uploaded twice it would just overwrite the previous version of the file. Also, if a separate request was made to Lambda it would have a different request ID.

**INCORRECT:** "The application stopped intermittently and then resumed" is incorrect. The issue is duplicate entries of the same request ID.

**References:**

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

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

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

Bottom of Form

Top of Form

Question 41: **Incorrect**

An application serves customers in several different geographical regions. Information about the location users connect from is written to logs stored in Amazon CloudWatch Logs. The company needs to publish an Amazon CloudWatch custom metric that tracks connections for each location.

Which approach will meet these requirements?

* **Create a CloudWatch metric filter to extract metrics from the log files with location as a dimension.**

**(Correct)**

* **Create a CloudWatch Logs Insights query to extract the location information from the logs and to create a custom metric with location as a dimension.**

**(Incorrect)**

* **Stream data to an Amazon Elasticsearch cluster in near-real time and export a custom metric.**
* **Configure a CloudWatch Events rule that creates a custom metric from the CloudWatch Logs group.**

**Explanation**

You can search and filter the log data coming into CloudWatch Logs 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.

When you create a metric from a log filter, you can also choose to assign dimensions and a unit to the metric. In this case, the company can assign a dimension that uses the location information.

**CORRECT:**"Create a CloudWatch metric filter to extract metrics from the log files with location as a dimension" is the correct answer.

**INCORRECT:** "Create a CloudWatch Logs Insights query to extract the location information from the logs and to create a custom metric with location as a dimension" is incorrect. You cannot create a custom metric through CloudWatch Logs Insights.

**INCORRECT:** "Configure a CloudWatch Events rule that creates a custom metric from the CloudWatch Logs group" is incorrect. You cannot create a custom metric using a CloudWatch Events rule.

**INCORRECT:** "Stream data to an Amazon Elasticsearch cluster in near-real time and export a custom metric" is incorrect. This is not a valid way of creating a custom metric in CloudWatch.

**References:**

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

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

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

Bottom of Form

Top of Form

Question 42: **Incorrect**

A developer is responsible for a business critical application that uses Amazon DynamoDB as its main data repository. This DynamoDB table holds millions of records and handles high volumes of requests. The developer must implement near-real time processing on the records as soon as they are inserted or modified in the DynamoDB table.

What's the most efficient way to introduce this capability with MINIMUM modification to the existing application code?

* **Use AWS Lambda triggered by DynamoDB Streams to process the documents.**

**(Correct)**

* **Modify the application code to add processing logic after each DynamoDB write operation.**
* **Use Amazon SQS to queue incoming data and process it using Amazon EC2 instances.**
* **Set up an Amazon Kinesis Data Stream to process updates from the DynamoDB table.**

**(Incorrect)**

**Explanation**

AWS Lambda can be triggered by DynamoDB Streams to automatically process changes to the DynamoDB table, ensuring near-real-time processing without substantial changes to the application code.

**CORRECT:**"Use AWS Lambda triggered by DynamoDB Streams to process the documents" is the correct answer (as explained above.)

**INCORRECT:** "Use Amazon SQS to queue incoming data and process it using Amazon EC2 instances" is incorrect.

Amazon SQS would require significant changes to the application code to integrate the queuing and processing mechanism. Moreover, it would not guarantee near-real-time processing.

**INCORRECT:** "Modify the application code to add processing logic after each DynamoDB write operation" is incorrect.

Modifying the application code to include processing logic after each write operation would entail significant changes to the code and may also slow down write operations.

**INCORRECT:** "Set up an Amazon Kinesis Data Stream to process updates from the DynamoDB table" is incorrect.

Amazon Kinesis Data Streams can be used to capture the changes, but consumers are required to perform the processing, and this is not mentioned in the solution. The simplest solution is to use DynamoDB streams with Lambda.

**References:**

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

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

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

Bottom of Form

Top of Form

Question 43: **Incorrect**

A developer is using AWS CodeBuild to build an application into a Docker image. The buildspec file is used to run the application build. The developer needs to push the Docker image to an Amazon ECR repository only upon the successful completion of each build.

* **Add a post\_build phase to the buildspec file that uses the commands block to push the Docker image.**

**(Correct)**

* **Add a post\_build phase to the buildspec file that uses the artifacts sequence to find the build artifacts and push to Amazon ECR.**

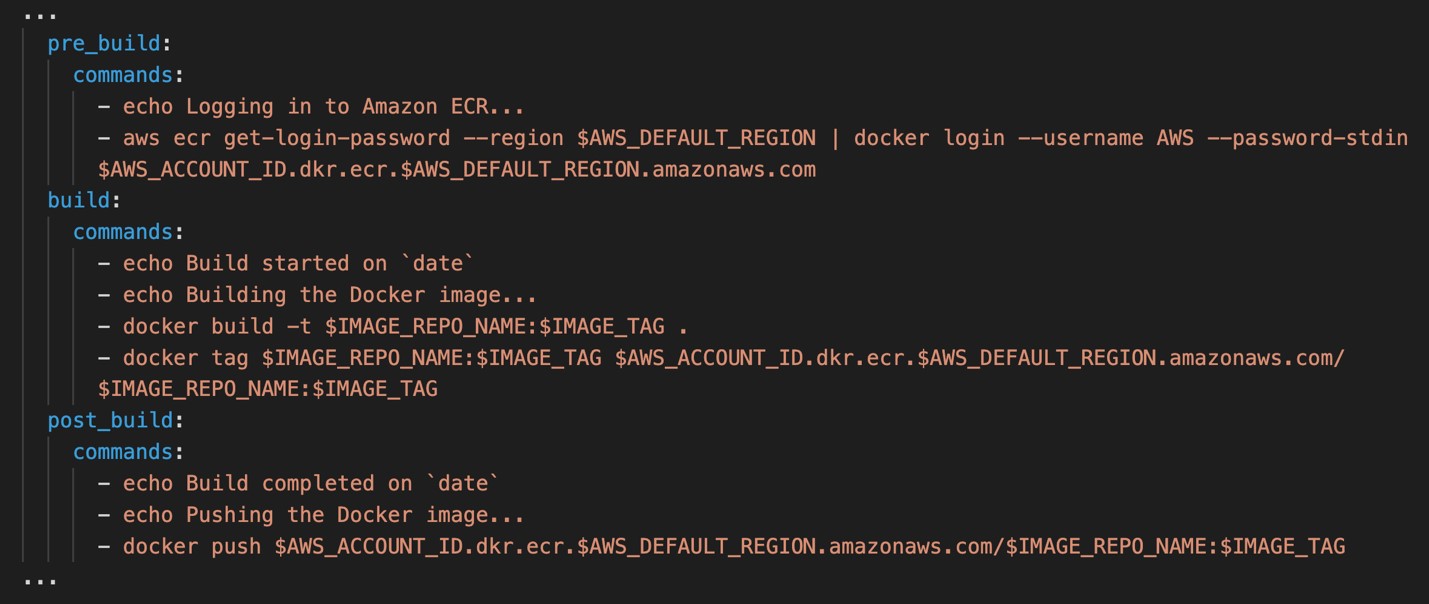
**(Incorrect)**

* **Add a post\_build phase to the buildspec file that uses the finally block to push the Docker image.**
* **Add an install phase to the buildspec file that uses the commands block to push the Docker image.**

**Explanation**

The post\_build phase is an optional sequence. It represents the commands, if any, that CodeBuild runs after the build. For example, you might use Maven to package the build artifacts into a JAR or WAR file, or you might push a Docker image into Amazon ECR. Then you might send a build notification through Amazon SNS.

Here is an example of a buildspec file with a post\_build phase that pushes a Docker image to Amazon ECR:



**CORRECT:**"Add a post\_build phase to the buildspec file that uses the commands block to push the Docker image" is the correct answer (as explained above.)

**INCORRECT:** "Add a post\_build phase to the buildspec file that uses the finally block to push the Docker image" is incorrect.

Commands specified in a finally block are run after commands in the commands block. The commands in a finally block are run even if a command in the commands block fails. This would not be ideal as this would push the image to ECR even if commands in previous sequences failed.

**INCORRECT:** "Add an install phase to the buildspec file that uses the commands block to push the Docker image" is incorrect.

These are commands that are run during installation. The develop would want to push the image only after all installations have succeeded. Therefore, the post\_build phase should be used.

**INCORRECT:** "Add a post\_build phase to the buildspec file that uses the artifacts sequence to find the build artifacts and push to Amazon ECR" is incorrect.

The artifacts sequence is not required if you are building and pushing a Docker image to Amazon ECR, or you are running unit tests on your source code, but not building it.

**References:**

<https://docs.aws.amazon.com/codebuild/latest/userguide/sample-docker.html>

<https://docs.aws.amazon.com/codebuild/latest/userguide/build-spec-ref.html>

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

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

Bottom of Form

Top of Form

Question 44: **Incorrect**

A Developer has used a third-party tool to build, bundle, and package a software package on-premises. The software package is stored in a local file system and must be deployed to Amazon EC2 instances.

How can the application be deployed onto the EC2 instances?

* **Use AWS CodeDeploy and point it to the local file system to deploy the software package.**

**(Incorrect)**

* **Create a repository using AWS CodeCommit to automatically trigger a deployment to the EC2 instances.**
* **Upload the bundle to an Amazon S3 bucket and specify the S3 location when doing a deployment using AWS CodeDeploy.**

**(Correct)**

* **Use AWS CodeBuild to commit the package and automatically deploy the software package.**

**Explanation**

AWS CodeDeploy can deploy software packages using an archive that has been uploaded to an Amazon S3 bucket. The archive file will typically be a .zip file containing the code and files required to deploy the software package.

**CORRECT:**"Upload the bundle to an Amazon S3 bucket and specify the S3 location when doing a deployment using AWS CodeDeploy" is the correct answer.

**INCORRECT:** "Use AWS CodeDeploy and point it to the local file system to deploy the software package" is incorrect. You cannot point CodeDeploy to a local file system running on-premises.

**INCORRECT:** "Create a repository using AWS CodeCommit to automatically trigger a deployment to the EC2 instances" is incorrect. CodeCommit is a source control system. In this case the source code has already been package using a third-party tool.

**INCORRECT:** "Use AWS CodeBuild to commit the package and automatically deploy the software package" is incorrect. CodeBuild does not commit packages (CodeCommit does) or deploy the software. It is a build service.

**References:**

<https://docs.aws.amazon.com/codedeploy/latest/userguide/tutorials-windows-upload-application.html>

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

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

Bottom of Form

Top of Form

Question 45: **Incorrect**

A company has created a set of APIs using Amazon API Gateway and exposed them to partner companies. The APIs have caching enabled for all stages. The partners require a method of invalidating the cache that they can build into their applications.

What can the partners use to invalidate the API cache?

* **They can pass the HTTP header Cache-Control: max-age=0**

**(Correct)**

* **They can use the query string parameter INVALIDATE\_CACHE**
* **They can invoke an AWS API endpoint which invalidates the cache**

**(Incorrect)**

* **They must wait for the TTL to expire**

**Explanation**

You can enable API caching in Amazon API Gateway to cache your endpoint's responses. With caching, you can reduce the number of calls made to your endpoint and also improve the latency of requests to your API.

When you enable caching for a stage, API Gateway caches responses from your endpoint for a specified time-to-live (TTL) period, in seconds. API Gateway then responds to the request by looking up the endpoint response from the cache instead of making a request to your endpoint. The default TTL value for API caching is 300 seconds. The maximum TTL value is 3600 seconds. TTL=0 means caching is disabled.

A client of your API can invalidate an existing cache entry and reload it from the integration endpoint for individual requests. The client must send a request that contains the Cache-Control: max-age=0 header.

The client receives the response directly from the integration endpoint instead of the cache, provided that the client is authorized to do so. This replaces the existing cache entry with the new response, which is fetched from the integration endpoint.

To grant permission for a client, attach a policy of the following format to an IAM execution role for the user.



This policy allows the API Gateway execution service to invalidate the cache for requests on the specified resource (or resources).

Therefore, as described above the solution is to get the partners to pass the HTTP header Cache-Control: max-age=0.

**CORRECT:**"They can pass the HTTP header Cache-Control: max-age=0" is the correct answer.

**INCORRECT:** "They can use the query string parameter INVALIDATE\_CACHE" is incorrect. This is not a valid method of invalidating the cache with API Gateway.

**INCORRECT:** "They must wait for the TTL to expire" is incorrect as this is not true, you do not need to wait as you can pass the HTTP header Cache-Control: max-age=0 whenever you need to in order to invalidate the cache.

**INCORRECT:** "They can invoke an AWS API endpoint which invalidates the cache" is incorrect. This is not a valid method of invalidating the cache with API Gateway.

**References:**

<https://docs.aws.amazon.com/apigateway/latest/developerguide/api-gateway-caching.html>

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

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

Bottom of Form

Top of Form

Question 46: **Incorrect**

A Developer will be launching several Docker containers on a new Amazon ECS cluster using the EC2 Launch Type. The containers will all run a web service on port 80.

What is the EASIEST way the Developer can configure the task definition to ensure the web services run correctly and there are no port conflicts on the host instances?

* **Specify port 80 for the container port and a unique port number for the host port**
* **Specify a unique port number for the container port and port 80 for the host port**

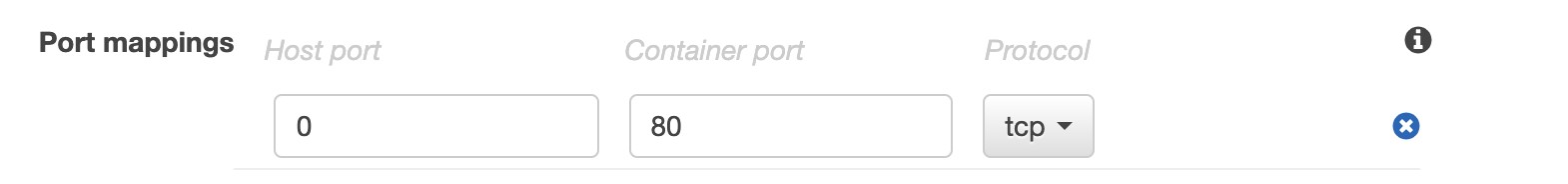
**(Incorrect)**

* **Leave both the container port and host port configuration blank**
* **Specify port 80 for the container port and port 0 for the host port**

**(Correct)**

**Explanation**

Port mappings allow containers to access ports on the host container instance to send or receive traffic. Port mappings are specified as part of the container definition. The container port is the port number on the container that is bound to the user-specified or automatically assigned host port. The host port is the port number on the container instance to reserve for your container.



As we cannot have multiple services bound to the same host port, we need to ensure that each container port mapping uses a different host port. The easiest way to do this is to set the host port number to 0 and ECS will automatically assign an available port. We also need to assign port 80 to the container port so that the web service is able to run.

**CORRECT:**"Specify port 80 for the container port and port 0 for the host port" is the correct answer.

**INCORRECT:** "Specify port 80 for the container port and a unique port number for the host port" is incorrect as this is more difficult to manage as you have to manually assign the port number.

**INCORRECT:** "Specify a unique port number for the container port and port 80 for the host port" is incorrect as the web service on the container needs to run on pot 80 and you can only bind one container to port 80 on the host so this would not allow more than one container to work.

**INCORRECT:** "Leave both the container port and host port configuration blank" is incorrect as this would mean that ECS would dynamically assign both the container and host port. As the web service must run on port 80 this would not work correctly.

**References:**

<https://docs.aws.amazon.com/AmazonECS/latest/APIReference/API_PortMapping.html>

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

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

Bottom of Form

Top of Form

Question 47: **Correct**

A review of Amazon CloudWatch metrics shows that there are a high number of reads taking place on a primary database built on Amazon Aurora with MySQL. What can a developer do to improve the read scaling of the database? (Select TWO.)

* **Create a separate Aurora MySQL cluster and configure binlog replication.**

**(Correct)**

* **Create a duplicate Aurora primary database to process read requests.**
* **Create Aurora Replicas in a global S3 bucket as the primary read source.**
* **Create a duplicate Aurora database cluster to process read requests.**
* **Create Aurora Replicas in same cluster as the primary database instance.**

**(Correct)**

**Explanation**

Aurora Replicas can help improve read scaling because it synchronously updates data with the primary database (within 100 ms). Aurora Replicas are created in the same DB cluster within a Region. With Aurora MySQL you can also enable binlog replication to another Aurora DB cluster which can be in the same or a different Region.

**CORRECT:**"Create Aurora Replicas in same cluster as the primary database instance" is the correct answer (as explained above.)

**CORRECT:**"Create a separate Aurora MySQL cluster and configure binlog replication" is also a correct answer (as explained above.)

**INCORRECT:** "Create a duplicate Aurora database cluster to process read requests" is incorrect. A duplicate Aurora database cluster would be a separate database with read and write capability and would not help with read scaling.

**INCORRECT:** "Create a duplicate Aurora primary database to process read requests" is incorrect. A duplicate Aurora primary database would be for read and write requests and would not help with read scaling.

**INCORRECT:** "Creating read replicas of Aurora in a S3 global bucket as the primary read source" is incorrect. S3 is an object storage service. It cannot be used to host databases.

**References**:

<https://docs.aws.amazon.com/AmazonRDS/latest/AuroraUserGuide/Aurora.Managing.Performance.html#Aurora.Managing.Performance.ReadScaling>

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

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

Bottom of Form

Top of Form

Question 48: **Incorrect**

A Development team would use a GitHub repository and would like to migrate their application code to AWS CodeCommit.  
What needs to be created before they can migrate a cloned repository to CodeCommit over HTTPS?

* **A public and private SSH key file**
* **A GitHub secure authentication token**

**(Incorrect)**

* **An Amazon EC2 IAM role with CodeCommit permissions**
* **A set of Git credentials generated with IAM**

**(Correct)**

**Explanation**

AWS CodeCommit is a managed version control service that hosts private Git repositories in the AWS cloud. To use CodeCommit, you configure your Git client to communicate with CodeCommit repositories. As part of this configuration, you provide IAM credentials that CodeCommit can use to authenticate you. IAM supports CodeCommit with three types of credentials:

Git credentials, an IAM -generated user name and password pair you can use to communicate with CodeCommit repositories over HTTPS.

SSH keys, a locally generated public-private key pair that you can associate with your IAM user to communicate with CodeCommit repositories over SSH.

AWS access keys, which you can use with the credential helper included with the AWS CLI to communicate with CodeCommit repositories over HTTPS.

In this scenario the Development team need to connect to CodeCommit using HTTPS so they need either AWS access keys to use the AWS CLI or Git credentials generated by IAM. Access keys are not offered as an answer choice so the best answer is that they need to create a set of Git credentials generated with IAM

**CORRECT:**"A set of Git credentials generated with IAM" is the correct answer.

**INCORRECT:** "A GitHub secure authentication token" is incorrect as they need to authenticate to AWS CodeCommit, not GitHub (they have already accessed and cloned the repository).

**INCORRECT:** "A public and private SSH key file" is incorrect as these are used to communicate with CodeCommit repositories using SSH, not HTTPS.

**INCORRECT:** "An Amazon EC2 IAM role with CodeCommit permissions" is incorrect as you need the Git credentials generated through IAM to connect to CodeCommit.

**References:**

<https://docs.aws.amazon.com/IAM/latest/UserGuide/id_credentials_ssh-keys.html>

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

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

Bottom of Form

Top of Form

Question 49: **Correct**

A developer is updating an Amazon ECS app that uses an ALB with two target groups and a single listener. The developer has an AppSpec file in an S3 bucket and an AWS CodeDeploy deployment group tied to the ALB and AppSpec file. The developer needs to use an AWS Lambda function for update validation before deployment.

Which solution meets these requirements?

* **Attach a listener to the deployment group. Update the AppSpec file to link the Lambda function to the BeforeAllowTraffic lifecycle hook.**
* **Add a listener to the ALB. Update the AppSpec file to link the Lambda function to the AfterAllowTraffic lifecycle hook.**
* **Attach a listener to the deployment group. Update the AppSpec file to link the Lambda function to the AfterAllowTraffic lifecycle hook.**
* **Add a listener to the ALB. Update the AppSpec file to link the Lambda function to the BeforeAllowTraffic lifecycle hook.**

**(Correct)**

**Explanation**

The AppSpec file allows a developer to specify scripts to be run at different lifecycle hooks. The BeforeAllowTraffic lifecycle event occurs before the updated task set is moved to the target group that is receiving live traffic. So, any validation before production deployment should be configured at this lifecycle event. The listener is configured at the ALB level, not at the deployment group.

**CORRECT:**"Add a listener to the ALB. Update the AppSpec file to link the Lambda function to the BeforeAllowTraffic lifecycle hook" is the correct answer (as explained above.)

**INCORRECT:** "Add a listener to the ALB. Update the AppSpec file to link the Lambda function to the AfterAllowTraffic lifecycle hook" is incorrect.

This is incorrect because AfterAllowTraffic lifecycle event occurs after the updated task set is moved to the target group that is receiving live traffic. Validation should be performed before the updated task set receives live traffic, not after.

**INCORRECT:** "Attach a listener to the deployment group. Update the AppSpec file to link the Lambda function to the BeforeAllowTraffic lifecycle hook" is incorrect.

This is incorrect because the listener is not attached to the deployment group. It is configured at the ALB level.

**INCORRECT:** "Attach a listener to the deployment group. Update the AppSpec file to link the Lambda function to the AfterAllowTraffic lifecycle hook" is incorrect.

Listeners are added to the ALB, not the deployment group, and validation should occur before the updated task set receives live traffic.

**References:**

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

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

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

Bottom of Form

Top of Form

Question 50: **Correct**

A developer is updating an Amazon Aurora MySQL database to allow more clients to connect. What database parameter needs to be updated to support a higher number of client connections?

* **max\_allowed\_packet**
* **max\_join\_size**
* **max\_user\_connections**
* **max\_connections**

**(Correct)**

**Explanation**

The maximum number of connections allowed to an Aurora MySQL DB instance is determined by the max\_connections parameter in the instance-level parameter group for the DB instance.

You can increase the maximum number of connections to your Aurora MySQL DB instance by scaling the instance up to a DB instance class with more memory, or by setting a larger value for the max\_connections parameter in the DB parameter group for your instance, up to 16,000.

**CORRECT:**"max\_connections" is the correct answer (as explained above.)

**INCORRECT:** "max\_allowed\_packet" is incorrect. This parameter sets the maximum size of one packet or any generated or intermediate string.

**INCORRECT:** "max\_join\_size" is incorrect. This option is used to set a limit on the maximum number of row accesses.

**INCORRECT:** "max\_user\_connections" is incorrect. This option limits the number of simultaneous connections that the user can make.

**References**:

<https://aws.amazon.com/premiumsupport/knowledge-center/rds-mysql-max-connections/>

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

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

Bottom of Form

Top of Form

Question 51: **Correct**

A customer requires a serverless application with an API which mobile clients will use. The API will have both and AWS Lambda function and an Amazon DynamoDB table as data sources. Responses that are sent to the mobile clients must contain data that is aggregated from both of these data sources.

The developer must minimize the number of API endpoints and must minimize the number of API calls that are required to retrieve the necessary data.

Which solution should the developer use to meet these requirements?

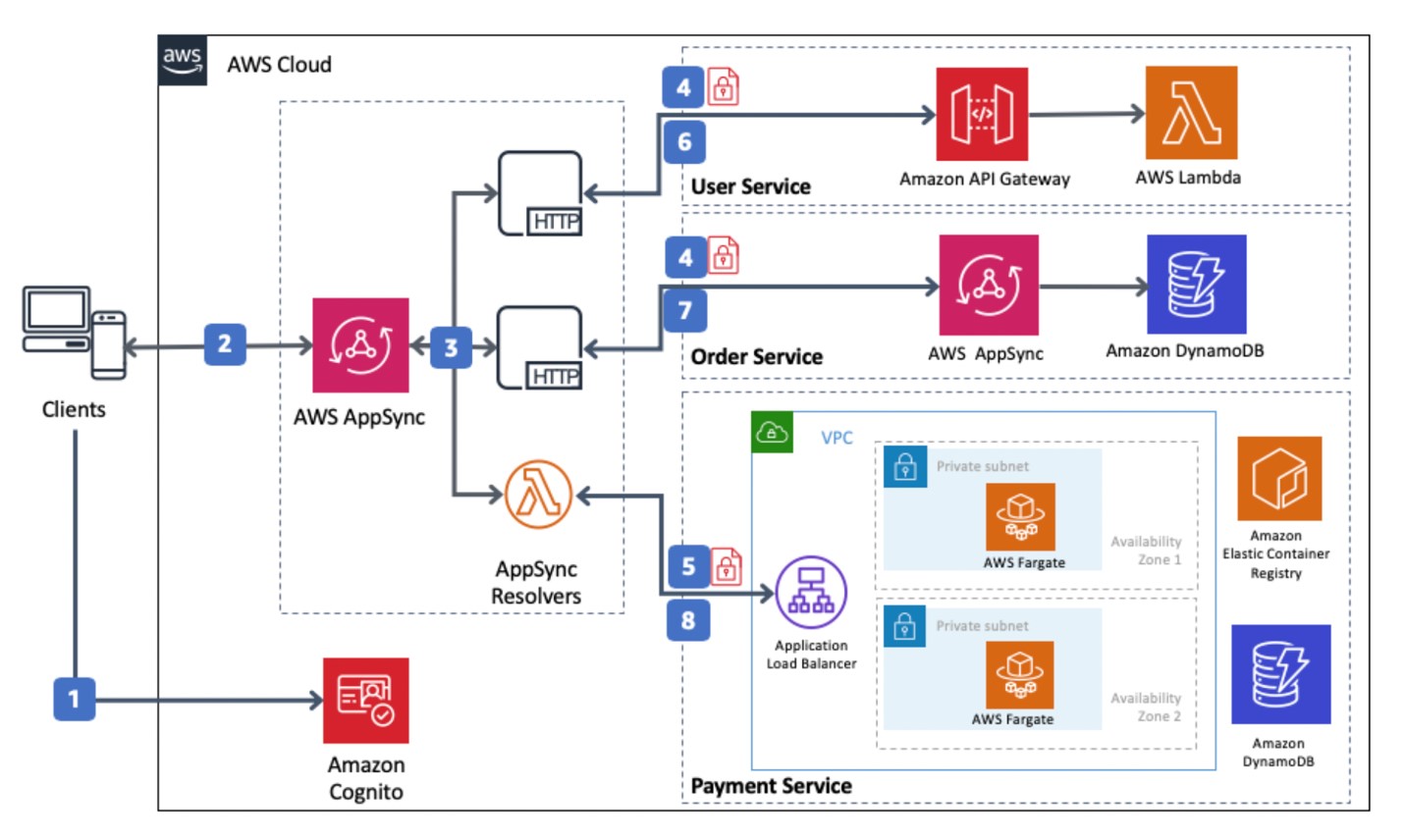
* **REST API on Amazon API Gateway**
* **GraphQL API on an Amazon EC2 instance**
* **REST API on AWS Elastic Beanstalk**
* **GraphQL API on AWS AppSync**

**(Correct)**

**Explanation**

GraphQL APIs built with AWS AppSync give front-end developers the ability to query multiple databases, microservices, and APIs from a single GraphQL endpoint. This would not be possible with a REST API running on API Gateway which would have a single target for each API endpoint.

The example diagram below depicts a solution that includes AWS AppSync in front of mobile clients. The services then connect to Lambda and DynamoDB via API Gateway and AppSync.



**CORRECT:**"GraphQL API on AWS AppSync" is the correct answer (as explained above.)

**INCORRECT:** "REST API on Amazon API Gateway" is incorrect.

A REST API is not suitable as the question asks to reduce the number of API endpoints. With a REST API there is a single target such as a Lambda per API endpoint so more endpoints would be required.

**INCORRECT:** "GraphQL API on an Amazon EC2 instance" is incorrect.

This would not be a serverless solution and the question states that the solution must be serverless.

**INCORRECT:** "REST API on AWS Elastic Beanstalk" is incorrect.

As explained above.

**References:**

<https://aws.amazon.com/blogs/mobile/appsync-microservices/>

Bottom of Form

Top of Form

Question 52: **Incorrect**

An organization is selling memorabilia that is illegal in specific countries. How can a developer restrict access to the website to countries where the memorabilia are illegal?

* **Create a Web ACL in AWS Shield with a rule that matches the specified countries and blocks access.**
* **Create a Web ACL in AWS WAF with a rule that matches the specified countries and triggers an SNS notification.**

**(Incorrect)**

* **Create a Web ACL in AWS WAF with a rule that matches the specified countries and blocks access.**

**(Correct)**

* **Create a Web ACL in AWS Shield with a rule that matches the specified countries and triggers an SNS notification.**

**Explanation**

AWS WAF can be used to set up a WEB ACL that can be used to block statements that originate from a specific country.

**CORRECT:**"Create a Web ACL in AWS WAF with a rule that matches the specified countries and blocks access" is the correct answer (as explained above.)

**INCORRECT:** "Create a Web ACL in AWS Shield with a rule that matches the specified countries and blocks access" is incorrect. AWS Shield is used to protect from DDoS attacks.

**INCORRECT:** "Create a Web ACL in AWS WAF with a rule that matches the specified countries and triggers an SNS notification" is incorrect. This will not block access to specific countries.

**INCORRECT:** " Create a Web ACL in AWS Shield with a rule that matches the specified countries and triggers an SNS notification" is incorrect. This will not block access to specific countries.

**References**:

<https://aws.amazon.com/premiumsupport/knowledge-center/waf-allow-block-country-geolocation/>

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

<https://digitalcloud.training/aws-waf-shield/>

Bottom of Form

Top of Form

Question 53: **Correct**

A developer must identify the public IP addresses of clients connecting to Amazon EC2 instances behind a public Application Load Balancer (ALB). The EC2 instances run an HTTP server that logs all requests to a log file.

How can the developer ensure the client public IP addresses are captured in the log files on the EC2 instances?

* **Install the Amazon CloudWatch Logs agent on the EC2 instances and configure logging.**
* **Install the AWS X-Ray daemon on the EC2 instances and configure request logging.**
* **Configure the HTTP server to add the x-forwarded-for request header to the logs.**

**(Correct)**

* **Configure the HTTP server to add the x-forwarded-proto request header to the logs.**

**Explanation**

The X-Forwarded-For request header is automatically added and helps you identify the IP address of a client when you use an HTTP or HTTPS load balancer.

Because load balancers intercept traffic between clients and servers, your server access logs contain only the IP address of the load balancer. To see the IP address of the client, use the X-Forwarded-For request header.

The HTTP server may need to be configured to include the x-forwarded-for request header in the log files. Once this is done, the logs will contain the public IP addresses of the clients.

**CORRECT:**"Configure the HTTP server to add the x-forwarded-for request header to the logs" is the correct answer (as explained above.)

**INCORRECT:** "Configure the HTTP server to add the x-forwarded-proto request header to the logs" is incorrect.

This request header identifies the protocol (HTTP or HTTPS).

**INCORRECT:** "Install the AWS X-Ray daemon on the EC2 instances and configure request logging" is incorrect.

X-Ray is used for tracing applications; it will not help identify the public IP addresses of clients.

**INCORRECT:** "Install the Amazon CloudWatch Logs agent on the EC2 instances and configure logging" is incorrect.

The Amazon CloudWatch Logs agent will send application and system logs to CloudWatch Logs. This does not help to capture the client IP addresses of connections.

**References:**

<https://docs.aws.amazon.com/elasticloadbalancing/latest/application/x-forwarded-headers.html#x-forwarded-for>

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

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

Bottom of Form

Top of Form

Question 54: **Correct**

An application asynchronously invokes an AWS Lambda function. The application has recently been experiencing occasional errors that result in failed invocations. A developer wants to store the messages that resulted in failed invocations such that the application can automatically retry processing them.

What should the developer do to accomplish this goal with the LEAST operational overhead?

* **Configure a redrive policy on an Amazon SQS queue. Set the dead-letter queue as an event source to the Lambda function.**

**(Correct)**

* **Configure an Amazon S3 bucket as a destination for failed invocations. Configure event notifications to trigger the Lambda function to process the events.**
* **Configure logging to an Amazon CloudWatch Logs group. Configure Lambda to read failed invocation events from the log group.**
* **Configure Amazon EventBridge to send the messages to Amazon SNS to initiate the Lambda function again.**

**Explanation**

Amazon SQS supports *dead-letter queues* (DLQ), which other queues (*source queues*) can target for messages that can't be processed (consumed) successfully. Dead-letter queues are useful for debugging your application or messaging system because they let you isolate unconsumed messages to determine why their processing doesn't succeed.

The *redrive policy* specifies the *source queue*, the *dead-letter queue*, and the conditions under which Amazon SQS moves messages from the former to the latter if the consumer of the source queue fails to process a message a specified number of times.

You can set your DLQ as an event source to the Lambda function to drain your DLQ. This will ensure that all failed invocations are automatically retried.

**CORRECT:**"Configure a redrive policy on an Amazon SQS queue. Set the dead-letter queue as an event source to the Lambda function" is the correct answer (as explained above.)

**INCORRECT:** "Configure logging to an Amazon CloudWatch Logs group. Configure Lambda to read failed invocation events from the log group" is incorrect.

The information in the logs may not be sufficient for processing the event. This is not an automated or ideal solution.

**INCORRECT:** "Configure Amazon EventBridge to send the messages to Amazon SNS to initiate the Lambda function again" is incorrect.

Amazon EventBridge can be configured as a failure destination and can send to SNS. SNS can also be configured with Lambda as a target. However, this solution requires more operational overhead compared to using a DLQ.

**INCORRECT:** "Configure an Amazon S3 bucket as a destination for failed invocations. Configure event notifications to trigger the Lambda function to process the events" is incorrect.

S3 is not a supported failure destination. Supported destinations are Amazon SNS, Amazon SQS, and Amazon EventBridge.

**References:**

<https://aws.amazon.com/blogs/compute/introducing-aws-lambda-destinations/>

<https://docs.aws.amazon.com/eventbridge/latest/userguide/eb-rule-dlq.html>

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

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

Bottom of Form

Top of Form

Question 55: **Incorrect**

A company is deploying a static website hosted from an Amazon S3 bucket. The website must support encryption in-transit for website visitors.

Which combination of actions must the Developer take to meet this requirement? (Select TWO.)

* **Configure the S3 bucket with an SSL/TLS certificate.**
* **Configure an Amazon CloudFront distribution with an SSL/TLS certificate.**

**(Correct)**

* **Configure an Amazon CloudFront distribution with an AWS WAF WebACL.**
* **Create an AWS WAF WebACL with a secure listener.**

**(Incorrect)**

* **Create an Amazon CloudFront distribution. Set the S3 bucket as an origin.**

**(Correct)**

**Explanation**

Amazon S3 static websites use the HTTP protocol only and you cannot enable HTTPS. To enable HTTPS connections to your S3 static website, use an Amazon CloudFront distribution that is configured with an SSL/TLS certificate. This will ensure that connections between clients and the CloudFront distribution are encrypted in-transit as per the requirements.

**CORRECT:**"Create an Amazon CloudFront distribution. Set the S3 bucket as an origin" is a correct answer.

**CORRECT:**"Configure an Amazon CloudFront distribution with an SSL/TLS certificate" is also a correct answer.

**INCORRECT:** "Create an AWS WAF WebACL with a secure listener" is incorrect. You cannot configure a secure listener on a WebACL.

**INCORRECT:** "Configure an Amazon CloudFront distribution with an AWS WAF WebACL" is incorrect. This will not enable encrypted connections.

**INCORRECT:** "Configure the S3 bucket with an SSL/TLS certificate" is incorrect. You cannot manually add SSL/TLS certificates to Amazon S3, and it is not possible to directly configure an S3 bucket that is configured as a static website to accept encrypted connections.

**References:**

<https://aws.amazon.com/premiumsupport/knowledge-center/cloudfront-serve-static-website/>

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

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

Bottom of Form

Top of Form

Question 56: **Incorrect**

A Developer needs to scan a full DynamoDB 50GB table within non-peak hours. About half of the strongly consistent RCUs are typically used during non-peak hours and the scan duration must be minimized.

How can the Developer optimize the scan execution time without impacting production workloads?

* **Use sequential scans**
* **Increase the RCUs during the scan operation**

**(Incorrect)**

* **Use parallel scans while limiting the rate**

**(Correct)**

* **Change to eventually consistent RCUs during the scan operation**

**Explanation**

Performing a scan on a table consumes a lot of RCUs. A Scan operation always scans the entire table or secondary index. It then filters out values to provide the result you want, essentially adding the extra step of removing data from the result set. To reduce the amount of RCUs used by the scan so it doesn’t affect production workloads whilst minimizing the execution time, there are a couple of recommendations the Developer can follow.

Firstly, the *Limit* parameter can be used to reduce the page size. The Scan operation provides a *Limit* parameter that you can use to set the page size for your request. Each Query or Scan request that has a smaller page size uses fewer read operations and creates a "pause" between each request.

Secondly, the Developer can configure parallel scans. With parallel scans the Developer can maximize usage of the available throughput and have the scans distributed across the table’s partitions.

A parallel scan can be the right choice if the following conditions are met:

The table size is 20 GB or larger.

The table's provisioned read throughput is not being fully used.

Sequential Scan operations are too slow.

Therefore, to optimize the scan operation the Developer should use parallel scans while limiting the rate as this will ensure that the scan operation does not affect the performance of production workloads and still have it complete in the minimum time.

**CORRECT:**"Use parallel scans while limiting the rate" is the correct answer.

**INCORRECT:** "Use sequential scans" is incorrect as this is slower than parallel scans and the Developer needs to minimize scan execution time.

**INCORRECT:** "Increase the RCUs during the scan operation" is incorrect as the table is only using half of the RCUs during non-peak hours so there are RCUs available. You could increase RCUs and perform the scan faster, but this would be more expensive. The better solution is to use parallel scans with the limit parameter.

**INCORRECT:** "Change to eventually consistent RCUs during the scan operation" is incorrect as this does not provide a solution for preventing impact to the production workloads. The limit parameter should be used to ensure the tables RCUs are not fully used.

**References:**

<https://docs.aws.amazon.com/amazondynamodb/latest/developerguide/bp-query-scan.html#QueryAndScanGuidelines.ParallelScan>

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

[https://digitalcloud.training/amazon-dynamodb/](https://digitalcloud.training/certification-training/aws-developer-associate/aws-database/amazon-dynamodb/)

Bottom of Form

Top of Form

Question 57: **Incorrect**

A Developer has created a task definition that includes the following JSON code:

1. "placementStrategy": [
2. {
3. "field": "attribute:ecs.availability-zone",
4. "type": "spread"
5. },
6. {
7. "field": "instanceId",
8. "type": "spread"
9. }
10. ]

What is the effect of this task placement strategy?

* **It distributes tasks evenly across Availability Zones and then distributes tasks randomly across instances within each Availability Zone**
* **It distributes tasks evenly across Availability Zones and then distributes tasks evenly across the instances within each Availability Zone**

**(Correct)**

* **It distributes tasks evenly across Availability Zones and then distributes tasks evenly across distinct instances within each Availability Zone**

**(Incorrect)**

* **It distributes tasks evenly across Availability Zones and then bin packs tasks based on memory within each Availability Zone**

**Explanation**

A *task placement strategy* is an algorithm for selecting instances for task placement or tasks for termination. Task placement strategies can be specified when either running a task or creating a new service.

Amazon ECS supports the following task placement strategies:

binpack

Place tasks based on the least available amount of CPU or memory. This minimizes the number of instances in use.

random

Place tasks randomly.

spread

Place tasks evenly based on the specified value. Accepted values are instanceId (or host, which has the same effect), or any platform or custom attribute that is applied to a container instance, such as attribute:ecs.availability-zone.

You can specify task placement strategies with the following actions: [CreateService](https://docs.aws.amazon.com/AmazonECS/latest/APIReference/API_CreateService.html), [UpdateService](https://docs.aws.amazon.com/AmazonECS/latest/APIReference/API_UpdateService.html), and [RunTask](https://docs.aws.amazon.com/AmazonECS/latest/APIReference/API_RunTask.html). You can also use multiple strategies together as in the example JSON code provided with the question.

**CORRECT:**"It distributes tasks evenly across Availability Zones and then distributes tasks evenly across the instances within each Availability Zone" is the correct answer.

**INCORRECT:** "It distributes tasks evenly across Availability Zones and then bin packs tasks based on memory within each Availability Zone" is incorrect as it does not use the binpack strategy.

**INCORRECT:** "It distributes tasks evenly across Availability Zones and then distributes tasks evenly across distinct instances within each Availability Zone" is incorrect as it does not spread tasks across distinct instances (use a task placement constraint).

**INCORRECT:** "It distributes tasks evenly across Availability Zones and then distributes tasks randomly across instances within each Availability Zone" is incorrect as it does not use the random strategy.

**References:**

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

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

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

Bottom of Form

Top of Form

Question 58: **Correct**

A Developer is deploying an application in a microservices architecture on Amazon ECS. The Developer needs to choose the best task placement strategy to MINIMIZE the number of instances that are used. Which task placement strategy should be used?

* **binpack**

**(Correct)**

* **spread**
* **random**
* **weighted**

**Explanation**

A *task placement strategy* is an algorithm for selecting instances for task placement or tasks for termination. Task placement strategies can be specified when either running a task or creating a new service.

Amazon ECS supports the following task placement strategies:

**binpack** - Place tasks based on the least available amount of CPU or memory. This minimizes the number of instances in use.

**random** - Place tasks randomly.

**spread** - Place tasks evenly based on the specified value. Accepted values are instanceId (or host, which has the same effect), or any platform or custom attribute that is applied to a container instance, such as attribute:ecs.availability-zone. Service tasks are spread based on the tasks from that service. Standalone tasks are spread based on the tasks from the same task group.

The **binpack** task placement strategy is the most suitable for this scenario as it minimizes the number of instances used which is a requirement for this solution.

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

**INCORRECT:** "random" is incorrect as this would assign tasks randomly to EC2 instances which would not result in minimizing the number of instances used.

**INCORRECT:** "spread" is incorrect as this would spread the tasks based on a specified value. This is not used for minimizing the number of instances used.

**INCORRECT:** "weighted" is incorrect as this is not an ECS task placement strategy. Weighted is associated with Amazon Route 53 routing policies.

**References:**

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

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

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

Bottom of Form

Top of Form

Question 59: **Correct**

A Development team is involved with migrating an on-premises MySQL database to Amazon RDS. The database usage is very read-heavy. The Development team wants re-factor the application code to achieve optimum read performance for queries.

How can this objective be met?

* **Add a connection string to use a read replica on an Amazon EC2 instance**
* **Add a connection string to use an Amazon RDS read replica for read queries**

**(Correct)**

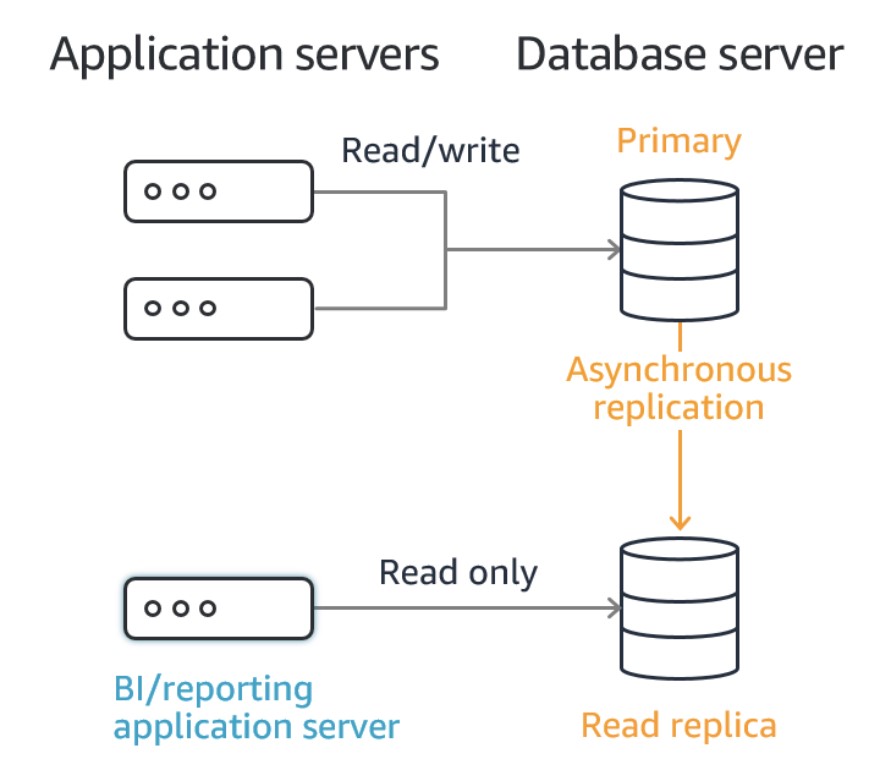
* **Add database retries to the code and vertically scale the Amazon RDS database**
* **Use Amazon RDS with a multi-AZ deployment**

**Explanation**

Amazon RDS uses the MariaDB, MySQL, Oracle, and PostgreSQL DB engines' built-in replication functionality to create a special type of DB instance called a Read Replica from a source DB instance. Updates made to the source DB instance are asynchronously copied to the Read Replica.

You can reduce the load on your source DB instance by routing read queries from your applications to the Read Replica. Using Read Replicas, you can elastically scale out beyond the capacity constraints of a single DB instance for read-heavy database workloads.

In the image below a primary Amazon RDS database server allows reads and writes while a Read Replica can be used for running read-only workloads such as BI/reporting. This reduces the load on the primary database.



It is necessary to add logic to your code to direct read traffic to the Read Replica and write traffic to the primary database. Therefore, in this scenario the Development team will need to “Add a connection string to use an Amazon RDS read replica for read queries”.

**CORRECT:**"Add a connection string to use an Amazon RDS read replica for read queries" is the correct answer.

**INCORRECT:** "Add database retries to the code and vertically scale the Amazon RDS database" is incorrect as this is not a good way to scale reads as you will likely hit a ceiling at some point in terms of cost or instance type. Scaling reads can be better implemented with horizontal scaling using a Read Replica.

**INCORRECT:** "Use Amazon RDS with a multi-AZ deployment" is incorrect as this creates a standby copy of the database in another AZ that can be failed over to in a failure scenario. This is used for DR not (at least not primarily) used for scaling performance. It is possible for certain RDS engines to use a multi-AZ standby as a read replica however the requirements in this solution do not warrant this configuration.

**INCORRECT:** "Add a connection string to use a read replica on an Amazon EC2 instance" is incorrect as Read Replicas are something you create on Amazon RDS, not on an EC2 instance.

**References:**

<https://docs.aws.amazon.com/AmazonRDS/latest/UserGuide/USER_ReadRepl.html>

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

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

Bottom of Form

Top of Form

Question 60: **Correct**

A developer is partitioning data using Athena to improve performance when performing queries. What are two things the analyst can do that would counter any benefit of using partitions? (Select TWO.)

* **Creating partitions directly from data source.**
* **Skewing data heavily to one partition value.**

**(Correct)**

* **Storing the data in S3.**
* **Segmenting data too finely.**

**(Correct)**

* **Using a Hive-style partition format.**

**Explanation**

There is a cost associated with partitioning data. A higher number of partitions can also increase the overhead from retrieving and processing the partition metadata. Multiple smaller files can counter the benefit of using partitioning. If your data is heavily skewed to one partition value, and most queries use that value, then the overhead may wipe out the initial benefit.

**CORRECT:**"Segmenting data too finely" is a correct answer (as explained above.)

**CORRECT:** "Skewing data heavily to one partition value" is a correct answer (as explained above.)

**INCORRECT:** "Storing the data in S3" is incorrect. Data must be stored in S3 buckets.

**INCORRECT:** "Creating partitions directly from data source " is incorrect. Athena can pull data directly from the S3 source.

**INCORRECT:** "Using a Hive-style partition format" is incorrect. Athena is compatible with Hive-style partition formats.

**References**:

<https://aws.amazon.com/blogs/big-data/top-10-performance-tuning-tips-for-amazon-athena/>

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

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

Bottom of Form

Top of Form

Question 61: **Incorrect**

A company has implemented AWS CodePipeline to automate its release pipelines. The Development team is writing an AWS Lambda function that will send notifications for state changes of each of the actions in the stages.

Which steps must be taken to associate the Lambda function with the event source?

* **Create an Amazon CloudWatch Events rule that uses CodePipeline as an event source**

**(Correct)**

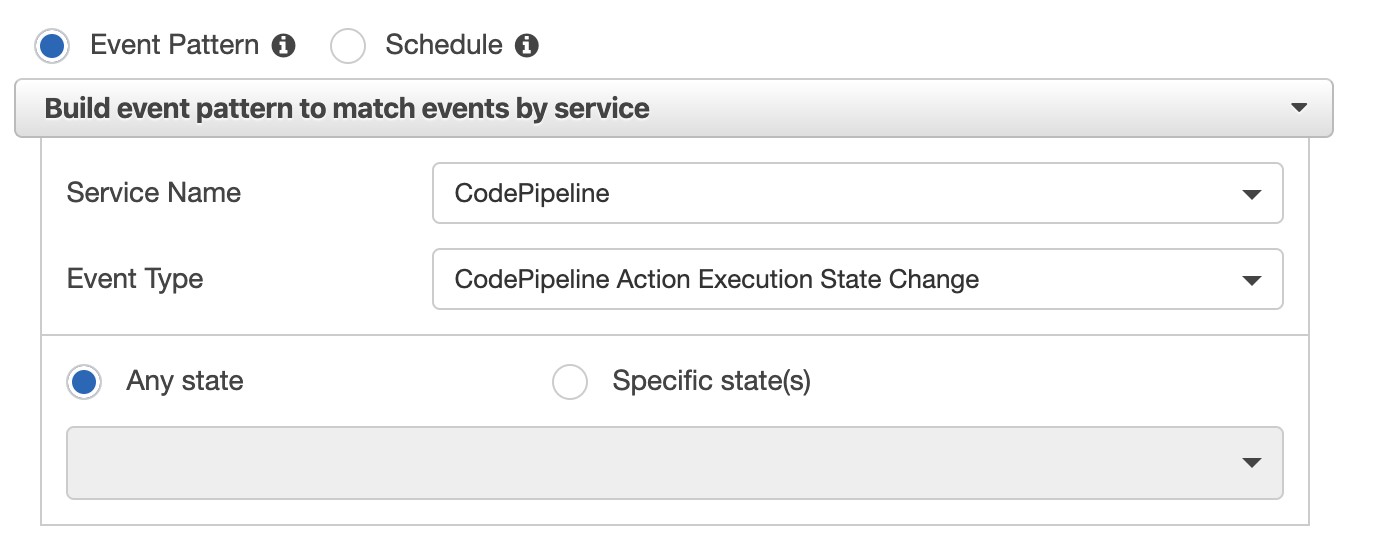
* **Create a trigger that invokes the Lambda function from the Lambda console by selecting CodePipeline as the event source**
* **Create an event trigger and specify the Lambda function from the CodePipeline console**
* **Create an Amazon CloudWatch alarm that monitors status changes in CodePipeline and triggers the Lambda function**

**(Incorrect)**

**Explanation**

[Amazon CloudWatch Events](https://docs.aws.amazon.com/AmazonCloudWatch/latest/DeveloperGuide/WhatIsCloudWatchEvents.html) help you to respond to state changes in your AWS resources. When your resources change state, they automatically send events into an event stream. You can create rules that match selected events in the stream and route them to your AWS Lambda function to take action.

AWS CodePipeline can be configured as an event source in CloudWatch Events and can then send notifications using as service such as Amazon SNS.



Therefore, the best answer is to create an Amazon CloudWatch Events rule that uses CodePipeline as an event source.

**CORRECT:**"Create an Amazon CloudWatch Events rule that uses CodePipeline as an event source" is the correct answer.

**INCORRECT:** "Create a trigger that invokes the Lambda function from the Lambda console by selecting CodePipeline as the event source" is incorrect as CodePipeline cannot be configured as a trigger for Lambda.

**INCORRECT:** "Create an event trigger and specify the Lambda function from the CodePipeline console" is incorrect as CodePipeline cannot be configured as a trigger for Lambda.

**INCORRECT:** "Create an Amazon CloudWatch alarm that monitors status changes in CodePipeline and triggers the Lambda function" is incorrect as CloudWatch Events is used for monitoring state changes.

**References:**

<https://docs.aws.amazon.com/AmazonCloudWatch/latest/events/WhatIsCloudWatchEvents.html>

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

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

Bottom of Form

Top of Form

Question 62: **Incorrect**

A Developer is deploying an AWS Lambda update using AWS CodeDeploy. In the appspec.yaml file, which of the following is a valid structure for the order of hooks that should be specified?

* **BeforeAllowTraffic > AfterAllowTraffic**

**(Correct)**

* **BeforeInstall > AfterInstall > AfterAllowTestTraffic > BeforeAllowTraffic > AfterAllowTraffic**

**(Incorrect)**

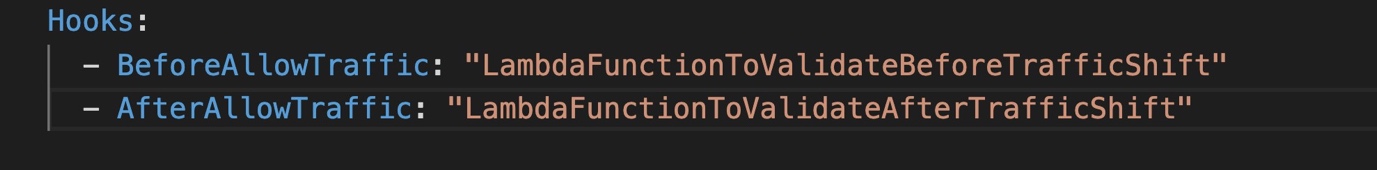
* **BeforeInstall > AfterInstall > ApplicationStart > ValidateService**
* **BeforeBlockTraffic > AfterBlockTraffic > BeforeAllowTraffic > AfterAllowTraffic**

**Explanation**

The content in the 'hooks' section of the AppSpec file varies, depending on the compute platform for your deployment. The 'hooks' section for an EC2/On-Premises deployment contains mappings that link deployment lifecycle event hooks to one or more scripts.

The 'hooks' section for a Lambda or an Amazon ECS deployment specifies Lambda validation functions to run during a deployment lifecycle event. If an event hook is not present, no operation is executed for that event. This section is required only if you are running scripts or Lambda validation functions as part of the deployment.

The following code snippet shows a valid example of the structure of hooks for an AWS Lambda deployment:



Therefore, in this scenario a valid structure for the order of hooks that should be specified in the appspec.yml file is: BeforeAllowTraffic > AfterAllowTraffic

**CORRECT:**"BeforeAllowTraffic > AfterAllowTraffic" is the correct answer.

**INCORRECT:** "BeforeInstall > AfterInstall > ApplicationStart > ValidateService" is incorrect as this would be valid for Amazon EC2.

**INCORRECT:** "BeforeInstall > AfterInstall > AfterAllowTestTraffic > BeforeAllowTraffic > AfterAllowTraffic" is incorrect as this would be valid for Amazon ECS.

**INCORRECT:** "BeforeBlockTraffic > AfterBlockTraffic > BeforeAllowTraffic > AfterAllowTraffic" is incorrect as this is a partial listing of hooks for Amazon EC2 but is incomplete.

**References:**

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

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

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

Bottom of Form

Top of Form

Question 63: **Correct**

A team of Developers require read-only access to an Amazon DynamoDB table. The Developers have been added to a group. What should an administrator do to provide the team with access whilst following the principal of least privilege?

* **Create a customer managed policy with read/write access to DynamoDB for all resources. Attach the policy to the group**
* **Assign the AWSLambdaDynamoDBExecutionRole AWS managed policy to the group**
* **Create a customer managed policy with read only access to DynamoDB and specify the ARN of the table for the “Resource” element. Attach the policy to the group**

**(Correct)**

* **Assign the AmazonDynamoDBReadOnlyAccess AWS managed policy to the group**

**Explanation**

The key requirement is to provide read-only access to the team for a specific DynamoDB table. Therefore, the AWS managed policy cannot be used as it will provide access to all DynamoDB tables in the account which does not follow the principal of least privilege.

Therefore, a customer managed policy should be created that provides read-only access and specifies the ARN of the table. For instance, the resource element might include the following ARN:

arn:aws:dynamodb:us-west-1:515148227241:table/exampletable

This will lock down access to the specific DynamoDB table, following the principal of least privilege.

**CORRECT:**"Create a customer managed policy with read only access to DynamoDB and specify the ARN of the table for the “Resource” element. Attach the policy to the group" is the correct answer.

**INCORRECT:** "Assign the AmazonDynamoDBReadOnlyAccess AWS managed policy to the group" is incorrect as this will provide read-only access to all DynamoDB tables in the account.

**INCORRECT:** "Assign the AWSLambdaDynamoDBExecutionRole AWS managed policy to the group" is incorrect as this is a role used with AWS Lambda.

**INCORRECT:** "Create a customer managed policy with read/write access to DynamoDB for all resources. Attach the policy to the group" is incorrect as read-only access should be provided, not read/write.

**References:**

<https://docs.aws.amazon.com/amazondynamodb/latest/developerguide/using-identity-based-policies.html>

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

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

Bottom of Form

Top of Form

Question 64: **Correct**

A Development team wants to run their container workloads on Amazon ECS. Each application container needs to share data with another container to collect logs and metrics.

What should the Development team do to meet these requirements?

* **Create two pod specifications. Make one to include the application container and the other to include the other container. Link the two pods together**
* **Create one task definition. Specify both containers in the definition. Mount a shared volume between those two containers**

**(Correct)**

* **Create two task definitions. Make one to include the application container and the other to include the other container. Mount a shared volume between the two tasks**
* **Create a single pod specification. Include both containers in the specification. Mount a persistent volume to both containers**

**Explanation**

Amazon ECS tasks support Docker volumes. To use data volumes, you must specify the volume and mount point configurations in your task definition. Docker volumes are supported for the EC2 launch type only.

To configure a Docker volume, in the task definition volumes section, define a data volume with name and DockerVolumeConfiguration values. In the containerDefinitions section, define multiple containers with mountPoints values that reference the name of the defined volume and the containerPath value to mount the volume at on the container.

The containers should both be specified in the same task definition. Therefore, the Development team should create one task definition, specify both containers in the definition and then mount a shared volume between those two containers

**CORRECT:**"Create one task definition. Specify both containers in the definition. Mount a shared volume between those two containers" is the correct answer.

**INCORRECT:** "Create two pod specifications. Make one to include the application container and the other to include the other container. Link the two pods together" is incorrect as pods are a concept associated with the Elastic Kubernetes Service (EKS).

**INCORRECT:** "Create two task definitions. Make one to include the application container and the other to include the other container. Mount a shared volume between the two tasks" is incorrect as a single task definition should be created with both containers.

**INCORRECT:** "Create a single pod specification. Include both containers in the specification. Mount a persistent volume to both containers" is incorrect as pods are a concept associated with the Elastic Kubernetes Service (EKS).

**References:**

<https://docs.aws.amazon.com/AmazonECS/latest/developerguide/docker-volumes.html>

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

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

Bottom of Form

Top of Form

Question 65: **Incorrect**

An application on-premises uses Linux servers and a relational database using PostgreSQL. The company will be migrating the application to AWS and require a managed service that will take care of capacity provisioning, load balancing, and auto-scaling.

Which combination of services should the Developer use? (Select TWO.)

* **AWS Elastic Beanstalk**

**(Correct)**

* **AWS Lambda with CloudWatch Events**
* **Amazon RDS with PostrgreSQL**

**(Correct)**

* **Amazon EC2 with PostgreSQL**
* **Amazon EC2 with Auto Scaling**

**(Incorrect)**

**Explanation**

The company require a managed service therefore the Developer should choose to use Elastic Beanstalk for the compute layer and Amazon RDS with the PostgreSQL engine for the database layer.

AWS Elastic Beanstalk will handle all capacity provisioning, load balancing, and auto-scaling for the web front-end and Amazon RDS provides push-button scaling for the backend.

**CORRECT:**"AWS Elastic Beanstalk" is a correct answer.

**CORRECT:**"Amazon RDS with PostrgreSQL" is also a correct answer.

**INCORRECT:** "Amazon EC2 with Auto Scaling" is incorrect as though these services will be used to provide the automatic scalability required for the solution, they still need to be managed. The questions asks for a managed solution and Elastic Beanstalk will manage this for you. Also, there is no mention of a load balancer so connections cannot be distributed to instances.

**INCORRECT:** "Amazon EC2 with PostgreSQL" is incorrect as the question asks for a managed service and therefore the database should be run on Amazon RDS.

**INCORRECT:** "AWS Lambda with CloudWatch Events" is incorrect as there is no mention of refactoring application code to run on AWS Lambda.

**References:**

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

<https://aws.amazon.com/rds/postgresql/>

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

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

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