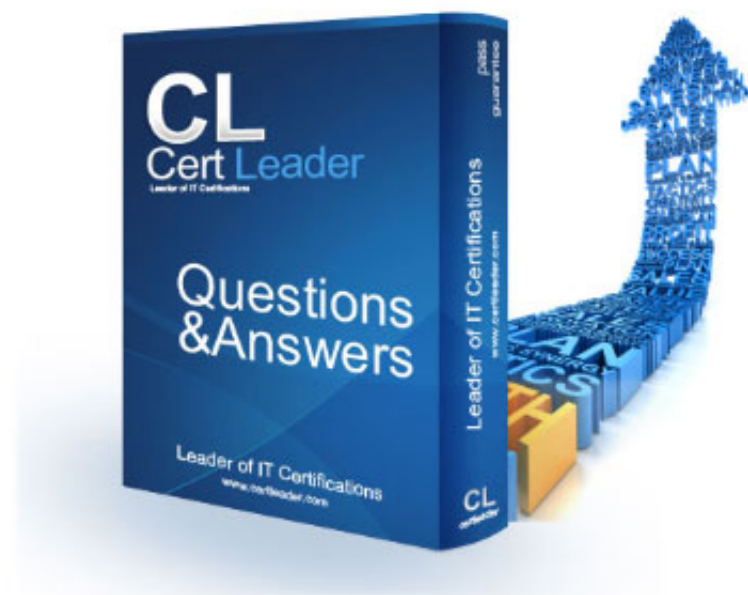


AI-200 Dumps

Developing AI Cloud Solutions on Azure

<https://www.certleader.com/AI-200-dumps.html>



NEW QUESTION 1

You are investigating high latency in an AI search application that processes millions of requests daily. Telemetry is stored in Azure Monitor Logs.

You must create a KQL query that correlates information from the AppRequests table and the AppDependencies table. The query must meet the following requirements:

Include only data from the last 24 hours.

Filter for failed requests only.

Calculate the average duration of dependencies, grouped by operation.

The query must follow best practice to optimize the performance by minimizing the initial data scan.

You need to create the query.

In which order should perform the actions? To answer, move all actions from the list of actions to the answer area and arrange them in the correct order.

Actions

⋮ Summarize average dependency duration by operation.

⋮ Join the dependencies table.

⋮ Select the requests table.

⋮ Filter the failed requests.

⋮ Apply a time filter.

Answer Area

- A. Mastered
- B. Not Mastered

Answer: A

Explanation:

Verified Answer:1) Select the requests table; 2) apply the 24-hour time filter; 3) filter failed requests; 4) join the dependencies table; 5) summarize average dependency duration by operation.

Detailed Explanation:KQL performs best when high-selectivity filters are applied as early as possible, especially the time predicate that limits the amount of data scanned. Starting from AppRequests, restricting to the last 24 hours, and then filtering failures reduces the left-side dataset before the join. The dependency table is then correlated with those requests, and aggregation is performed last to calculate average dependency duration by operation. Joining or summarizing before the time and failure filters would process more data than necessary.

Study Guide Alignment:Security and operations: Key Vault, App Configuration, managed identity, OpenTelemetry, Azure Monitor, and KQL-based troubleshooting.

Official Microsoft Learn References:AI-200 Study Guide | Optimize log queries in Azure Monitor

NEW QUESTION 2

You are developing a microservices-based application that uses Azure Container Apps. The application consists of several containerized services that handle tasks, such as processing orders, managing inventory, and generating reports. You deploy a new revision of the processing orders app.

Processing orders must be triggered by a web request and must always be available based on incoming web requests.

You need to validate that the replica is ready to handle incoming requests.

What should you implement?

- A. HTTP liveness probe
- B. HTTP startup probe
- C. TCP readiness probe
- D. HTTP readiness probe
- E. TCP liveness probe

Answer: D

Explanation:

Detailed Explanation:A readiness probe answers whether a running replica is currently ready to receive traffic. For a web-triggered order-processing service, an HTTP readiness probe can test the application endpoint and keep the replica out of the routing pool until it is able to handle requests. Liveness probes detect whether the process should be restarted, and startup probes protect slow-starting containers during initialization. A TCP readiness probe can test socket acceptance, but the HTTP readiness probe is the more application-aware choice for the HTTP service described.

Study Guide Alignment:Containerized Azure workloads: registry builds, App Service containers, Container Apps revision/scaling behavior, and AKS deployment choices.

Official Microsoft Learn References:AI-200 Study Guide | Health probes in Azure Container Apps

NEW QUESTION 3

You are developing a microservices-based application that uses Azure Container Apps The application consists of several containerized services that handle tasks, such as processing orders, managing inventory, and generating reports.

You must secure the container apps. All apps must reside in the same virtual network, share the same Dapr configuration, and share the same logging location.

Apps must support the configuration of the amount of memory and compute resources available to containers.

You need to configure the Azure Container App

How should you complete the CLI command? To answer, select the appropriate options in the answer area.

NOTE: Each correct selection is worth one point.

-n MyContainerApp -g MyResourceGroup --location eastus2

▼

env

create

add-on

▼

env

create

add-on

-n MyContainerApp -g MyResourceGroup --location eastus2

▼

--enable-mtls

--internal-only

--enable-workload-profiles

A. Mastered
B. Not Mastered

Answer: A

Explanation:

Verified Answer: `az containerapp env create ... --enable-workload-profiles`.

Detailed Explanation: The shared boundary for Container Apps is the managed environment. Apps in one environment can share the environment's virtual network integration, observability destination, and Dapr-related environment configuration. Enabling workload profiles provides selectable compute profiles and resource sizing options for workloads in that environment. The CLI must therefore create a Container Apps environment and enable workload profiles. Creating an individual container app alone would not establish the common environment-level network, logging, and resource-profile boundary required by the scenario.

Study Guide Alignment: Containerized Azure workloads: registry builds, App Service containers, Container Apps revision/scaling behavior, and AKS deployment choices.

Official Microsoft Learn References: AI-200 Study Guide | Azure Container Apps environments

NEW QUESTION 4

You need to address the known issue resulting from vector similarity queries. Which two actions should you perform? Each correct answer presents part of the solution. Choose two. NOTE: Each correct selection is worth one point.

A. Add a composite index on the vector fields and metadata properties of the container.
B. Reduce the numeric precision of the vector embeddings where supported.
C. Set the account consistency level to Strong.
D. Change the vector index type from flat to quantizedFlat or diskANN.

Answer: BD

Explanation:

The objective is to reduce RU consumption from vector similarity queries. Microsoft guidance identifies vector numeric precision and vector-index selection as major cost and performance levers. Using a lower supported vector precision can reduce storage and processing cost, while quantizedFlat and DiskANN are designed to reduce latency and RU consumption compared with flat search for appropriate data sizes. Strong consistency would increase resource cost rather than solve vector-search efficiency. A regular composite index is not a substitute for the vector index. Option B should be read as reducing vector numeric precision, not changing a generic indexing precision setting.

Study Guide Alignment: AI data-management workloads: Cosmos DB, PostgreSQL, caching, vector storage, vector retrieval, consistency, and connection optimization.

Official Microsoft Learn References: AI-200 Study Guide | Vector search in Azure Cosmos DB | Optimize Cosmos DB vector search performance

NEW QUESTION 5

You are provisioning and configuring a Service Bus processor for AI batch jobs. The processor must connect to an existing queue, register handlers for message and error processing, and then begin receiving messages. You need to provision and configure the Service Bus processor for message and error handling. Which four actions should you perform in sequence? To answer, move the appropriate actions from the list of actions to the answer area and arrange them in the correct order.

⋮ Dead-letters failed messages.

⋮ Create the Service Bus client.

⋮ Create the Service Bus processor for the queue.

⋮ Register message and error handlers.

⋮ Start the message processor.

Answer Area

A. Mastered
B. Not Mastered

Answer: A

Explanation:

The Leader of IT Certification

visit - <https://www.certleader.com>

The processor depends on a namespace client/connection, so the client is created first. Next, the queue-specific processor or receiver is created from that client. Message and error callbacks must be registered before processing starts so incoming messages and failures have defined handlers. Starting the processor is therefore the final step. Dead-lettering failed messages is an optional application settlement decision and is not a prerequisite for constructing and starting the processor.

Study Guide Alignment: Azure service integration: Service Bus, Event Grid, Azure Functions triggers /bindings, and event-driven processing.

Official Microsoft Learn References: AI-200 Study Guide | Service Bus message transfers, locks, and settlement

NEW QUESTION 6

A company is implementing a publish-subscribe (Pub/Sub) messaging component by using Azure Service Bus. You are developing the first subscription application.

In the Azure portal you see that messages are being sent to the subscription for each topic. You create and initialize a subscription client object by supplying the correct details, but the subscription application is still not consuming the messages.

You need to ensure that the subscription client processes all messages. Which code segment should you use?

- A. `subscriptionClient = new SubscriptionClient(ServiceBusConnectionString, TopicName, SubscriptionName);`
- B. `subscriptionClient.RegisterMessageHandler(ProcessMessagesAsync, messageHandlerOptions);`
- C. `await subscriptionClient.AddRuleAsync(new RuleDescription(RuleDescription.DefaultRuleName, new TrueFilter()));`
- D. `await subscriptionClient.CloseAsync();`

Answer: B

Explanation:

Creating a subscription client establishes the client object but does not start application-level message processing. The message handler must be registered so received subscription messages are delivered to the application callback. Adding a default TrueFilter is only necessary when subscription rules require adjustment and does not start the receiver; the default subscription rule already accepts messages in a normal subscription. Closing the client would stop processing rather than enable it.

Study Guide Alignment: Azure service integration: Service Bus, Event Grid, Azure Functions triggers /bindings, and event-driven processing.

Official Microsoft Learn References: AI-200 Study Guide | Service Bus message transfers, locks, and settlement

NEW QUESTION 7

You are implementing an application by using Azure Event Grid to push near-real-time information to customers. You have the following requirements:

- You must send events to thousands of customers that include hundreds of various event types.
- The events must be filtered by event type before processing.
- Authentication and authorization must be handled by using Microsoft Entra ID.
- The events must be published to a single endpoint

You need to implement Azure Event Grid

Solution: Publish events to a system topic. Create an event subscription for each customer.

Does the solution meet the goal?

- A. Yes
- B. No

Answer: B

Explanation:

Explanation

System topics represent events emitted by Azure services and are not the correct resource for an application publishing its own hundreds of event types to thousands of customers through one custom endpoint. Event Grid domains are intended for exactly this type of application-level multi-tenant topic organization and expose one publishing endpoint for many topics. Therefore a system topic with a subscription per customer does not satisfy the stated publishing model, even though Event Grid supports filtering and secure delivery in other contexts.

Study Guide Alignment: Azure service integration: Service Bus, Event Grid, Azure Functions triggers/bindings, and event-driven processing.

Official Microsoft Learn References: AI-200 Study Guide | Azure Event Grid event domains

Verification completed using official Microsoft Learn sources only. Original exhibits have been retained unchanged.

NEW QUESTION 8

You are reviewing the message processing code in a backend worker service. You review the following Python code that initializes and starts a Service Bus processor

```
from azure.servicebus import ServiceBusClient, ServiceBusReceiveMode

@1 import json
@2 from azure.servicebus import ServiceBusClient, ServiceBusReceiveMode
@3 from azure.identity import DefaultAzureCredential
@4 credential = DefaultAzureCredential()
@5 with client.get_queue_receiver(
@6     queue_name="inference-requests",
```

Service Bus processor behavior and SDK defaults

Statement	Yes	No
Messages are removed from the queue as soon as they are received by the processor.	☐	☐
If message processing fails due to a transient service error, the message can be retried by another consumer.	☐	☐
Messages with invalid JSON payloads are retried until the maximum delivery count is reached.	☐	☐

- A. Mastered
- B. Not Mastered

Answer: A

Explanation:

The default Service Bus receive behavior is PeekLock rather than receive-and-delete, so merely receiving a message does not remove it from the queue; it must be settled successfully. If processing fails and the message remains unsettled or the lock is lost, it can become available for another delivery. Invalid JSON, however, is an application-level content problem: Service Bus does not inspect the payload as JSON and automatically retry it to MaxDeliveryCount solely because deserialization fails. The application's settlement/error path determines what happens.

Study Guide Alignment: Azure service integration: Service Bus, Event Grid, Azure Functions triggers/bindings, and event-driven processing.

Official Microsoft Learn References: AI-200 Study Guide | Service Bus message transfers, locks, and settlement | Service Bus messages, routing, and correlation

NEW QUESTION 9

You process Azure Service Bus messages that require a dependent external API call.

If the API is temporarily unavailable, you must delay processing of the message without incrementing the delivery count. You need to find a way to process the message when the API is available while keeping the message accessible. Which message action should you perform?

- A. Defer
- B. Complete
- C. Dead-letter
- D. Abandon

Answer: A

Explanation:

Deferral is intended for a message that cannot be processed now because a particular dependency is temporarily unavailable, while the application wants to keep the message in the main queue for later retrieval by sequence number. Complete would permanently remove it, dead-letter would move it aside as a failed message, and abandon immediately releases the lock for redelivery. Deferral therefore best matches the requirement to postpone the individual message while keeping it available for deliberate later processing.

Study Guide Alignment: Azure service integration: Service Bus, Event Grid, Azure Functions triggers/bindings, and event-driven processing.

Official Microsoft Learn References: AI-200 Study Guide | Service Bus message deferral | Service Bus message transfers, locks, and settlement

NEW QUESTION 10

You are using the Python SDK to develop a containerized AI application that retrieves the value of a runtime setting stored in an Azure App Configuration resource. The application will run in Azure in the security context of a managed identity.

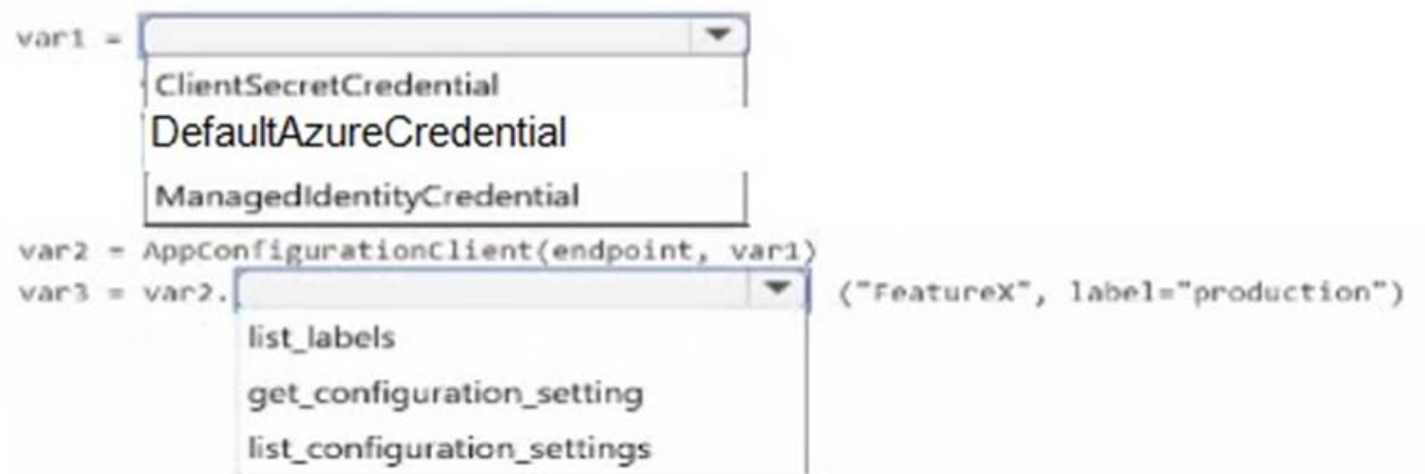
The application must work in a local development environment without any code changes.

You need to complete the code that implements the retrieval.

How should you complete the code? To answer, select the appropriate options in the answer area.

NOTE: Each correct selection is worth one point.

SDK-based configuration retrieval



- A. Mastered
- B. Not Mastered

Answer: A

Explanation:

DefaultAzureCredential is designed to support the same application code across local development and Azure-hosted environments. Locally it can use developer credentials, while in Azure it can use the workload's managed identity. The App Configuration client's get_configuration_setting method retrieves a specific key/label pair, which matches the code shown. Using ManagedIdentityCredential directly would not provide the same local-development credential chain without code or environment-specific handling.

Study Guide Alignment: Security and operations: Key Vault, App Configuration, managed identity, OpenTelemetry, Azure Monitor, and KQL-based troubleshooting.

Official Microsoft Learn References: AI-200 Study Guide | Azure App Configuration Python client | Managed identities for Azure resources

NEW QUESTION 10

You are developing a solution that uses several Azure Service Bus queues. You create an Azure Event Grid subscription for the Azure Service Bus namespace.

You use Azure Functions as subscribers to process the messages.

You need to ensure that events can be sent to Azure Event Grid from the queues. The solution must use the principle of least privilege and minimize costs.

Which Azure Service Bus role and tier level should you use? To answer, select the appropriate options in the answer area.

NOTE: Each correct selection is worth one point.

Answer Area

Configuration	Value
Tier level	Basic Standard Premium
Role	Azure Service Bus Data Receiver Azure Service Bus Data Sender Azure Service Bus Data Owner

- A. Mastered
- B. Not Mastered

Answer: A

Explanation:

Microsoft documents Event Grid integration for Service Bus queues and topics as requiring a Service Bus Premium namespace. The Azure Function subscriber must then receive messages from the queue; the least-privilege built-in data-plane role for that purpose is Azure Service Bus Data Receiver. Data Sender does not permit receiving, and Data Owner is broader than necessary. Premium therefore satisfies the feature prerequisite, while Data Receiver satisfies the stated least-privilege requirement.

Study Guide Alignment: Azure service integration: Service Bus, Event Grid, Azure Functions triggers

/bindings, and event-driven processing.

Official Microsoft Learn References: AI-200 Study Guide | Service Bus to Event Grid integration | Authenticate Service Bus with Microsoft Entra ID

NEW QUESTION 12

You are designing an Azure Function app that exposes a public API. The solution must:

Validate incoming request data and return results immediately to the caller. Support Microsoft Entra ID authentication.

Scale automatically under variable load. You need to implement a trigger. Which trigger should you implement?

- A. HTTP
- B. Azure Queue storage
- C. Azure Event Grid
- D. Service Bus topic

Answer: A

Explanation:

An HTTP trigger is the only option that directly accepts a synchronous external request and can return a response immediately to the caller. Azure Functions uses HTTP triggers to implement serverless APIs and webhooks, while Microsoft Entra authentication can be applied to the Function App or API surface. Queue, Event Grid, and Service Bus triggers are asynchronous event-processing mechanisms and do not provide the request/response semantics required by this public API scenario.

Study Guide Alignment: Azure service integration: Service Bus, Event Grid, Azure Functions triggers

/bindings, and event-driven processing.

Official Microsoft Learn References: AI-200 Study Guide | Azure Functions HTTP trigger

NEW QUESTION 13

You are developing several microservices to run on Azure Container Apps. External HTTP ingress traffic has been enabled for the Microservices.

A deployed microservice must be updated to allow users to test new features. You have the following requirements:

* Enable and maintain a single URL for the updated microservice to provide to test users.

* Update the microservice that corresponds to the current microservice version. You need to configure Azure Container Apps.

Which features should you configure? To answer, select the appropriate options in the answer area. NOTE: Each correct selection is worth one point.

Answer Area

Requirement:	Feature
Single URL for test users	Revision Label Revision mode Container image Container registry
Current microservice activation	Revision Label Revision mode Container image Container registry

- A. Mastered
- B. Not Mastered

Answer: A

Explanation:

Azure Container Apps revision labels provide a stable label-specific URL that can be reassigned to a different revision, which is appropriate for a consistent test endpoint. Revision mode controls

whether one or multiple revisions can be active at the same time and therefore governs how the current and updated application versions are activated. The container image and registry identify what is deployed, but they do not provide the stable test URL or the application-level revision activation behavior requested. Study Guide Alignment: Containerized Azure workloads: registry builds, App Service containers, Container Apps revision/scaling behavior, and AKS deployment choices.

Official Microsoft Learn References: AI-200 Study Guide | Azure Container Apps revisions | Azure Container Apps revision labels

NEW QUESTION 17

An ACA app processes messages from an Azure Storage queue.

The app must scale automatically based on messages in a specific Azure Storage queue by using a Kubernetes Event-driven Autoscaler (KEDA) custom scale rule. You need to configure the required scale rule values.

Which two values should you configure? Each correct answer presents part of the solution. Choose two. NOTE: Each correct selection is worth one point.

- A. Polling interval
- B. Trigger type
- C. Maximum replicas
- D. Queue name

Answer: BD

Explanation:

A KEDA custom scale rule must identify what scaler to use and which resource within that scaler should be observed. For Azure Storage queues, the trigger type selects the Storage Queue scaler and the queue name identifies the specific queue whose message count drives scaling. Polling interval and maximum replicas are general scaling controls, but they do not define the required queue-specific trigger. Therefore trigger type and queue name are the essential rule values tested here.

Study Guide Alignment: Containerized Azure workloads: registry builds, App Service containers, Container Apps revision/scaling behavior, and AKS deployment choices.

Official Microsoft Learn References: AI-200 Study Guide | Set scaling rules in Azure Container Apps

NEW QUESTION 22

You have an Azure web app that uses Azure Cosmos DB as a data store. You create a Cosmos DB container by running the following PowerShell script:

```
$resourceGroupName = "testResourceGroup"
$accountName = "testCosmosAccount"
$databaseName = "testDatabase"
$containerName = "testContainer"
$partitionKeyPath = "/EmployeeId"
$autoscaleMaxThroughput = 5000
```

```
New-AzCosmosDBSqlContainer `
    -ResourceGroupName $resourceGroupName `
    -AccountName $accountName `
    -DatabaseName $databaseName `
    -Name $containerName `
    -PartitionKeyKind Hash `
    -PartitionKeyPath $partitionKeyPath `
    -AutoscaleMaxThroughput $autoscaleMaxThroughput
```

You create the following queries that target the container:

```
SELECT * FROM c WHERE c.EmployeeId > '12345'
SELECT * FROM c WHERE c.UserId = '12345'
```

For each of the following statements, select Yes if the statement is true. Otherwise, select No. NOTE: Each correct selection is worth one point.

Answer Ares

Statements	Yes	No
The minimum throughput for the container is 400 RU/s	☐	☐
The first query statement is an in-partition query.	☐	☐
The second query statement is a cross-partition query.	☐	☐

- A. Mastered
B. Not Mastered

Answer: A

Explanation:

The question maps directly to the AI-200 objective “Develop AI solutions by using Azure Cosmos DB for NoSQL,” which includes running queries and optimizing query performance and Request Unit (RU) consumption.

Statement 1: No — The minimum throughput is not 400 RU/s.

The PowerShell command provisions the container with:

-AutoscaleMaxThroughput 5000

Azure Cosmos DB autoscale operates between approximately 10% and 100% of the configured maximum throughput . Microsoft documentation specifically gives the example of an autoscale container provisioned with 5,000 RU/s scaling between 500 RU/s and 5,000 RU/s . Therefore, for this container, the minimum operating autoscale throughput is 500 RU/s , not 400 RU/s.

Therefore:
??The minimum throughput for the container is 400 RU/s.?? # No
Statement 2: No — The first query is not an in-partition query. The container uses: /EmployeeId as its partition key. The first query is:
SELECT * FROM c WHERE c.EmployeeId > ' 12345 '
Although the query references the partition key, it uses a range predicate (>) , not an equality predicate. Microsoft explicitly states that a range filter on a partition key is not scoped to a single physical partition . To qualify as an in-partition query, the filter must identify the applicable partition, typically through an equality predicate such as:
WHERE c.EmployeeId = ' 12345 '
Microsoft ' s documentation provides essentially the same example: a query using DeviceId > ... against a container partitioned by DeviceId is not an in-partition query .
Therefore:
??The first query statement is an in-partition query.?? # No
Statement 3: Yes — The second query is a cross-partition query. The second query is:
SELECT * FROM c WHERE c.UserId = ' 12345 '
The container ' s partition key is /EmployeeId, not /UserId. Because the query contains no filter on the partition key , Azure Cosmos DB cannot route it to one logical partition. It must fan out the query across the applicable physical partitions and combine the results.
Microsoft describes this behavior directly: when a query does not contain a filter on the partition key, it must execute across the partitions.
Therefore:
??The second query statement is a cross-partition query.?? # Yes

NEW QUESTION 26

You deploy a private container image from Azure Container Registry (ACR) to App Service. App Service must authenticate to ACR to pull the image. The solution must NOT store static registry credentials. You need to configure the secure image pull authentication. Which configurations should you use? To answer, select the appropriate options in the answer area.
NOTE: Each correct selection is worth one point.
Configure secure container image pull

Requirement	Configuration
Authenticate App Service to ACR	Assign the Container Registry Repository Reader role to a managed identity. Configure a webhook on ACR Enable the admin user.
Avoid storing static credentials	Embed credentials in Dockerfile. Store the registry password in application settings. Use managed identity with role assignment.

- A. Mastered
- B. Not Mastered

Answer: A

Explanation:

App Service can pull a private image from Azure Container Registry by using a managed identity rather than registry username/password credentials. The identity must be granted an appropriate pull-only registry role at the registry scope, preserving least privilege. A webhook does not authenticate an image pull, and enabling the ACR admin account or storing a registry password would introduce static credentials. The managed-identity approach satisfies both secure authentication and credential elimination requirements.
Study Guide Alignment: Containerized Azure workloads: registry builds, App Service containers, Container Apps revision/scaling behavior, and AKS deployment choices.
Official Microsoft Learn References: AI-200 Study Guide | Configure a custom container for App Service | Managed identities for Azure resources

NEW QUESTION 27

You are reviewing an Azure Function app that processes incoming order requests for a company. The function must:

- Accept order submissions from an external client application.
- Require controlled access for security
- Return a response containing the processed request payload.

You review the following code segment:

```
01 import azure.functions as func
02 app = func.FunctionApp(http_auth_level=func.AuthLevel.FUNCTION)
03 @app.route(route="processorder", methods=["POST"])
04 def processorder(req: func.HttpRequest) -> func.HttpResponse:
05     result = req.get_body().decode("utf-8")
06     return func.HttpResponse(result)
```

For each of the following statements, select Yes if the statement is true. Otherwise, select No.
NOTE: Each correct selection is worth one point.

Code-level analysis of HTTP trigger configuration

Statement	Yes	No
The function requires a function key for invocation.	☐	☐
The function supports HTTP GET requests.	☐	☐
The response returns the request body.	☐	☐
The function validates the request body before returning a response.	☐	☐

- A. Mastered
B. Not Mastered

Answer: A

Explanation:

auth_level=FUNCTION` means invocation requires a function key unless a stronger platform authentication layer is configured, so the first statement is true. The route explicitly permits POST, not GET. The function reads `req.get_body()` and returns that value in the HttpResponseMessage, so the response contains the request body. No schema, type, required-field, or other payload validation is performed before the response is constructed, making the final statement false.

Study Guide Alignment: Azure service integration: Service Bus, Event Grid, Azure Functions triggers /bindings, and event-driven processing.

Official Microsoft Learn References: AI-200 Study Guide | Azure Functions HTTP trigger

NEW QUESTION 32

You develop a message-processing service deployed to Azure Container Apps. The service reads messages from an Azure Service Bus queue. The solution must minimize costs by ensuring NO compute resources are consumed when the queue is empty. You need to configure scaling for the service. Which two actions should you perform? Each correct answer presents part of the solution. NOTE: Each correct selection is worth one point.

- A. A.Increase the scaling rule to allow for the maximum running replica count.B.Configure the scaling rule to allow for the termination of all active replicas.C.Configure a Kubernetes Event-driven Autoscaler rule that monitors queue length.D.Enable HTTP ingress concurrency scaling.

Answer: BC

NEW QUESTION 36

You configure ACR Tasks to automate image builds. Container images must rebuild when: Application updates occur. Base image updates occur, such as when the underlying OS image is updated. Regular scheduled rebuilds are required. You need to configure ACR Tasks to support automated image rebuilds. Which three triggers should you configure? Each correct answer presents part of the solution. NOTE: Each correct selection is worth one point.

- A. A.Timer triggerB.Source code commit triggerC.Registry event triggerD.Base image update triggerE.Webhook notification trigger

Answer: ABD

Explanation:

To support automated container image rebuilds based on your requirements, you need to configure source triggers, base image triggers, and timer triggers in Azure Container Registry (ACR) Tasks. Timer Triggers: Automates regular scheduled rebuilds. This uses a cron schedule expression to run image builds at specific intervals (e.g., weekly or monthly) regardless of code changes. Source Triggers: Automates rebuilds when application updates occur. This monitors changes in your source code repository (like GitHub or Azure Repos) and fires a build when code is committed. Base Image Triggers: Automates rebuilds when the underlying OS or framework image updates. ACR tracks dependencies and automatically kicks off a new build when your defined FROM image changes in the public registry or your private registry. Reference: <https://oneuptime.com/blog/post/2026-02-16-how-to-set-up-acr-tasks-for-automated-containerimage-builds-on-git-commit/view>

NEW QUESTION 41

DRAG DROP - You are developing several microservices to run on Azure Container Apps. The microservices must allow HTTPS access by using a custom domain. You need to configure the custom domain in Azure Container Apps. In which order should you perform the actions? To answer, move all actions from the list of actions to the answer area and arrange them in the correct order.

Actions

Add the custom domain name.

Bind the certificate.

Enable ingress.

Validate the custom domain name.

Add DNS records to the domain provider.

Answer area

1.

2.

3.

4.

5.

- A. Mastered
- B. Not Mastered

Answer: A

Explanation:

Answer area

1.

Enable ingress.

2.

Add DNS records to the domain provider.

3.

Validate the custom domain name.

4.

Add the custom domain name.

5.

Bind the certificate.

NEW QUESTION 42

You have an Event Grid subscription that triggers an Azure Function.
You need to prevent loss of events in case the endpoint returns an HTTP 400 response.
Which action should you perform?

- A. Implement optimistic batchin
- B. Implement asynchronous handshake validatio
- C. Configure a dead-letter destinatio
- D. Configure a retry policy.

Answer: C

Explanation:

Configuring a dead-letter destination is exactly the right step to take.

By default, when an endpoint returns an HTTP 400 (Bad Request) or HTTP 413 (Payload Too Large) response, Azure Event Grid immediately stops delivery attempts and drops the message. It treats these specific errors as non-transient, client-side issues, meaning it skips its standard 24-hour retry policy.

Configuring a dead-letter destination ensures that these dropped events are safely preserved for future troubleshooting and reprocessing.

Reference:

<https://turbo360.com/blog/azure-event-grid-dead-letter-monitoring>

NEW QUESTION 43

You plan to deploy a container to an Azure App Service API app named api1. You host the source code for api1 in a GitHub repository. The container uses the API key at runtime to connect to a backend service.

The container must be able to retrieve the API key at runtime without exposing it in the source repository or Git commit history. You need to ensure that the API key remains outside of Git commit history and is available to the container at runtime. Solution: Store the API key as a GitHub repository secret. Does the solution meet the goal?

- A. Yes
- B. No

Answer: B

Explanation:

Correct:

* Store the API key in Azure Key Vault and reference it from an App Service application setting.

Storing the API key in Azure Key Vault and referencing it via App Service application settings is the recommended, secure approach. This strategy completely removes sensitive credentials from your GitHub repository and Git commit history while injecting them safely into your container environment at runtime.

Incorrect:

* Embed the API key as a hardcoded environment variable in the Dockerfile.

* Store the API key as a GitHub repository secret.

Reference:

<https://www.qservicesit.com/full-stack-applications-on-azure>

NEW QUESTION 44

You deploy an AI application across multiple Azure regions.

The application must be able to view writes across all regions within a predictable time window.

You need to determine the appropriate consistency level.

What are two consistency levels you can use to achieve the goal? Each correct answer presents a complete solution.

NOTE: Each correct selection is worth one point.

- A. Session
- B. Strong
- C. Bounded staleness
- D. Eventual

Answer: BC

Explanation:

The two appropriate consistency levels for a globally distributed AI application needing to view writes within a predictable time window are Bounded Staleness and Strong.

Bounded Staleness: This provides a guarantee that reads will lag behind writes by a user-specified "staleness window" (either in time or number of operations). It delivers high availability and predictable read consistency across all regions, balancing performance with strict read-order guarantees.

Strong: This provides linearizable and absolute consistency, ensuring that writes are seen synchronously across all regions. It features an exact zero-second predictable time window, though it comes at the expense of higher latency for multi-region operations due to the time required to replicate globally.

Reference:

<https://learn.microsoft.com/en-us/azure/cosmos-db/consistency-levels>

NEW QUESTION 49

.....

Thank You for Trying Our Product

* 100% Pass or Money Back

All our products come with a 90-day Money Back Guarantee.

* One year free update

You can enjoy free update one year. 24x7 online support.

* Trusted by Millions

We currently serve more than 30,000,000 customers.

* Shop Securely

All transactions are protected by VeriSign!

100% Pass Your AI-200 Exam with Our Prep Materials Via below:

<https://www.certleader.com/AI-200-dumps.html>