Microsoft

DP-420

DP-420 Designing and Implementing Cloud-Native Applications Using Microsoft Azure Cosmos DB

[Total Questions: 102]

Exam Topic Breakdown

Exam Topic	Number of Questions
Topic 1 : Litware, inc	7
Topic 2 : Misc. Questions	95
TOTAL	102

Topic 1, Litware, inc

Case Study

This is a case study. Case studies are not timed separately. You can use as much exam time as you would like to complete each case. However, there may be additional case studies and sections on this exam. You must manage your time to ensure that you are able to complete all questions included on this exam in the time provided.

To answer the questions included in a case study, you will need to reference information that is provided in the case study. Case studies might contain exhibits and other resources that provide more information about the scenario that is described in the case study. Each question is independent of the other questions in this case study.

At the end of this case study, a review screen will appear. This screen allows you to review your answers and to make changes before you move to the next section of the exam. After you begin a new section, you cannot return to this section.

To start the case study

To display the first question in this case study, click the Next button. Use the buttons in the left pane to explore the content of the case study before you answer the questions. Clicking these buttons displays information such as business requirements, existing environment, and problem statements. If the case study has an All Information tab, note that the information displayed is identical to the information displayed on the subsequent tabs. When you are ready to answer a question, click the Question button to return to the question.

Overview

Litware, Inc. is a United States-based grocery retailer. Litware has a main office and a primary datacenter in Seattle. The company has 50 retail stores across the United States and an emerging online presence. Each store connects directly to the internet.

Existing environment. Cloud and Data Service Environments.

Litware has an Azure subscription that contains the resources shown in the following table.

Name	Type	Description
account1	Azure Cosmos DB Core (SQL) API account	The account has a single read-write region and contains a database named productdb. The productdb database contains two containers named con-product and con-productVendor. The account uses the session default consistency level.
iothub1	Azure IoT hub	The IoT hub collects per-second temperature and humidity telemetry from 5,000 IoT devices in the retail stores. A single telemetry reading generates 1 KB of data.
streamanalytics1	Azure Stream Analytics job	The job processes telemetry data collected from iothub1.

Each container in productdb is configured for manual throughput.

The con-product container stores the company's product catalog data. Each document in con-product includes a con-productvendor value. Most queries targeting the data in con-product are in the following format.

SELECT * FROM con-product p WHERE p.con-productVendor - 'name'

Most queries targeting the data in the con-productVendor container are in the following format

SELECT * FROM con-productVendor pv

ORDER BY pv.creditRating, pv.yearFounded

Existing environment. Current Problems.

Litware identifies the following issues:

Updates to product categories in the con-productVendor container do not propagate automatically to documents in the con-product container.

Application updates in con-product frequently cause HTTP status code 429 "Too many requests". You discover that the 429 status code relates to excessive request unit (RU) consumption during the updates.

Requirements. Planned Changes

Litware plans to implement a new Azure Cosmos DB Core (SQL) API account named account2 that will contain a database named iotdb. The iotdb database will contain two containers named con-iot1 and con-iot2.

Litware plans to make the following changes:

Store the telemetry data in account2.

Configure account1 to support multiple read-write regions.

Implement referential integrity for the con-product container.

Use Azure Functions to send notifications about product updates to different recipients.

Develop an app named App1 that will run from all locations and query the data in account1.

Develop an app named App2 that will run from the retail stores and query the data in account2. App2 must be limited to a single DNS endpoint when accessing account2.

Requirements. Business Requirements

Litware identifies the following business requirements:

Whenever there are multiple solutions for a requirement, select the solution that provides the best performance, as long as there are no additional costs associated.

Ensure that Azure Cosmos DB costs for IoT-related processing are predictable.

Minimize the number of firewall changes in the retail stores.

Requirements. Product Catalog Requirements

Litware identifies the following requirements for the product catalog:

Implement a custom conflict resolution policy for the product catalog data.

Minimize the frequency of errors during updates of the con-product container.

Once multi-region writes are configured, maximize the performance of App1 queries against the data in account1.

Trigger the execution of two Azure functions following every update to any document in the con-product container.

Question #:1 - (Exam Topic 1)

You configure multi-region writes for account1.

You need to ensure that App1 supports the new configuration for account1. The solution must meet the business requirements and the product catalog requirements.

What should you do?

A. Set the default consistency level of accountl to bounded staleness.

B. Create a private endpoint connection.

C. Modify the connection policy of App1.

D. Increase the number of request units per second (RU/s) allocated to the con-product and con-productVendor containers.

Answer: D

Explanation

App1 queries the con-product and con-productVendor containers.

Note: Request unit is a performance currency abstracting the system resources such as CPU, IOPS, and memory that are required to perform the database operations supported by Azure Cosmos DB.

Scenario:

Develop an app named App1 that will run from all locations and query the data in account1.

Once multi-region writes are configured, maximize the performance of App1 queries against the data in account1.

Whenever there are multiple solutions for a requirement, select the solution that provides the best performance, as long as there are no additional costs associated.

Reference:

https://docs.microsoft.com/en-us/azure/cosmos-db/consistency-levels

Question #:2 - (Exam Topic 1)

You need to identify which connectivity mode to use when implementing App2. The solution must support the planned changes and meet the business requirements.

Which connectivity mode should you identify?

A. Direct mode over HTTPS

B. Gateway mode (using HTTPS)

C. Direct mode over TCP

Answer: C

Explanation

Scenario: Develop an app named App2 that will run from the retail stores and query the data in account2. App2 must be limited to a single DNS endpoint when accessing account2.

By using Azure Private Link, you can connect to an Azure Cosmos account via a private endpoint. The private endpoint is a set of private IP addresses in a subnet within your virtual network.

When you're using Private Link with an Azure Cosmos account through a direct mode connection, you can use only the TCP protocol. The HTTP protocol is not currently supported.

Reference:

https://docs.microsoft.com/en-us/azure/cosmos-db/how-to-configure-private-endpoints

Question #:3 - (Exam Topic 1)

You need to select the partition key for con-iot1. The solution must meet the IoT telemetry requirements.

What should you select?

A. the timestamp

B. the humidity

C. the temperature

D. the device ID

Answer: D

Explanation

The partition key is what will determine how data is routed in the various partitions by Cosmos DB and needs to make sense in the context of your specific scenario. The IoT Device ID is generally the "natural" partition key for IoT applications.

Scenario: The iotdb database will contain two containers named con-iot1 and con-iot2.

Ensure that Azure Cosmos DB costs for IoT-related processing are predictable.

Reference:

https://docs.microsoft.com/en-us/azure/architecture/solution-ideas/articles/iot-using-cosmos-db

Question #:4 - (Exam Topic 1)

You need to provide a solution for the Azure Functions notifications following updates to con-product. The solution must meet the business requirements and the product catalog requirements.

Which two actions should you perform? Each correct answer presents part of the solution.

NOTE: Each correct selection is worth one point.

A. Configure the trigger for each function to use a different leaseCollectionPrefix

B. Configure the trigger for each function to use the same leaseCollectionNair.e

C. Configure the trigger for each function to use a different leaseCollectionName

D. Configure the trigger for each function to use the same leaseCollectionPrefix

Answer: A C

Explanation

leaseCollectionPrefix: when set, the value is added as a prefix to the leases created in the Lease collection for this Function. Using a prefix allows two separate Azure Functions to share the same Lease collection by using different prefixes.

Scenario: Use Azure Functions to send notifications about product updates to different recipients.

Trigger the execution of two Azure functions following every update to any document in the con-product container.

Reference:

https://docs.microsoft.com/en-us/azure/azure-functions/functions-bindings-cosmosdb-v2-trigger

Question #:5 - (Exam Topic 1)

You are troubleshooting the current issues caused by the application updates.

Which action can address the application updates issue without affecting the functionality of the application?

A. Enable time to live for the con-product container.

B. Set the default consistency level of account1 to strong.

C. Set the default consistency level of account1 to bounded staleness.

D. Add a custom indexing policy to the con-product container.

Answer: C

Explanation

Bounded staleness is frequently chosen by globally distributed applications that expect low write latencies but require total global order guarantee. Bounded staleness is great for applications featuring group collaboration and sharing, stock ticker, publish-subscribe/queueing etc.

Scenario: Application updates in con-product frequently cause HTTP status code 429 "Too many requests". You discover that the 429 status code relates to excessive request unit (RU) consumption during the updates.

Reference:

https://docs.microsoft.com/en-us/azure/cosmos-db/consistency-levels

Question #:6 - (Exam Topic 1)

You need to implement a solution to meet the product catalog requirements.

What should you do to implement the conflict resolution policy.

A. Remove frequently changed field from the index policy of the con-product container.

B. Disable indexing on all fields in the index policy of the con-product container.

C. Set the default consistency level for account1 to eventual.

D. Create a new container and migrate the product catalog data to the new container.

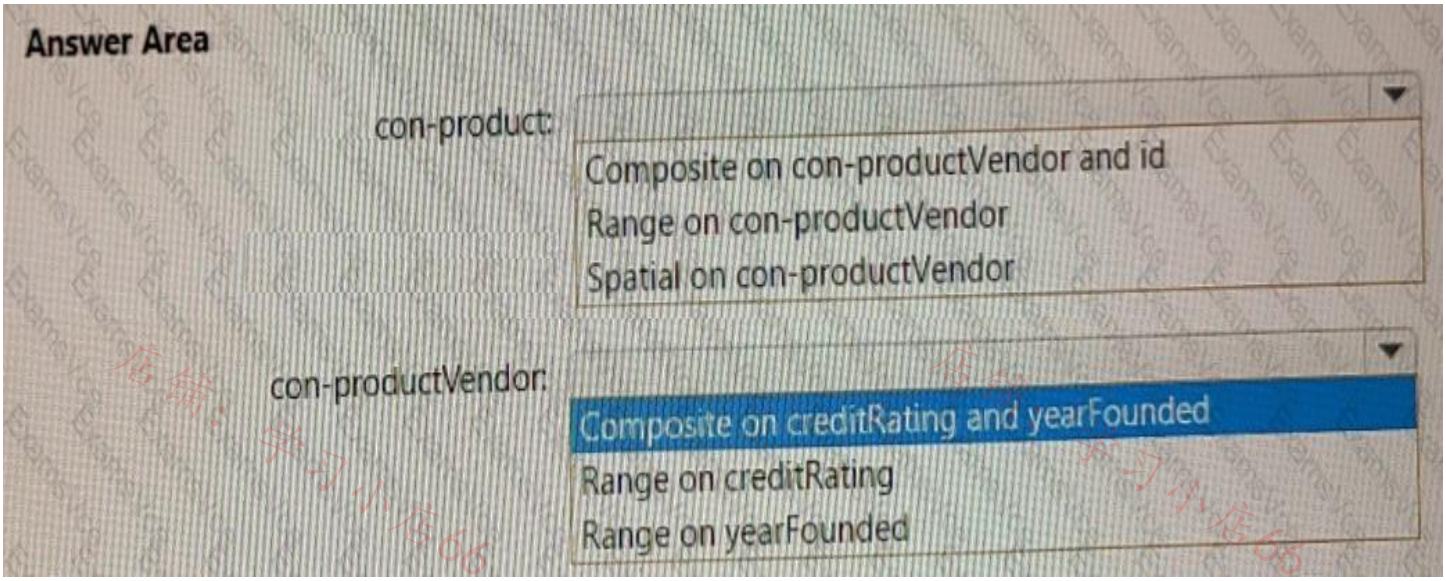
Answer: D

Question #:7 - (Exam Topic 1)

You need to recommend indexes for con-product and con-productVendor. The solution must meet the product catalog requirements and the business requirements.

Which type of index should you recommend for each container? To answer, select the appropriate options in the answer area.

NOTE: Each correct selection is worth one point.



Answer:

Explanation

Answer Area

con-product:

- Composite on con-productVendor and id
- Range on con-productVendor
- Spatial on con-productVendor

con-productVendor:

- Composite in creditRating and yearFounded
- Range on creditRating
- Range on yearFounded

Topic 2, Misc. Questions

Question #:1 - (Exam Topic 2)

You have a container m an Azure Cosmos DB for NoSQL account. The container stores data about families. Data about parents, children, and pets are stored as separate documents.

Each document contains the address of each family. Members of the same family share the same partition key named family Id

You need to update the address for each member of the same family that share the same address. The solution must meet the following requirements:

• Be atomic consistent isolated, and durable (ACID).

• Provide the lowest latency.

What should you do?

A. Update the document of each family member by using a transactional batch operation.

B. Update the document of each family member separately by using a patch operation.

C. Update the document of each family member separately and set the consistency level to strong.

Answer: A

Question #:2 - (Exam Topic 2)

Note: This question is part of a series of questions that present the same scenario. Each question in the series contains a unique solution that might meet the stated goals. Some question sets might have more than one correct solution, while others might not have a correct solution.

After you answer a question in this section, you will NOT be able to return to it. As a result, these questions will not appear in the review screen.

You have an Azure Cosmos DB Core (SQL) API account named account 1 that uses autoscale throughput.

You need to run an Azure function when the normalized request units per second for a container in account1 exceeds a specific value.

Solution: You configure an application to use the change feed processor to read the change feed and you configure the application to trigger the function.

Does this meet the goal?

A. Yes

B. No

Answer: B

Explanation

Instead configure an Azure Monitor alert to trigger the function.

You can set up alerts from the Azure Cosmos DB pane or the Azure Monitor service in the Azure portal.

Reference: https://docs.microsoft.com/en-us/azure/cosmos-db/create-alerts

Question #:3 - (Exam Topic 2)

You have the following query.

```
SELECT * FROM

WHERE c.sensor = "TEMP1"

AND c.value < 22

AND c.timestamp >= 1619146031231
```

You need to recommend a composite index strategy that will minimize the request units (RUs) consumed by the query.

What should you recommend?

A. a composite index for (sensor ASC, value ASC) and a composite index for (sensor ASC, timestamp ASC)

B. a composite index for (sensor ASC, value ASC, timestamp ASC) and a composite index for (sensor DESC, value DESC, timestamp DESC)

C. a composite index for (value ASC, sensor ASC) and a composite index for (timestamp ASC, sensor ASC)

D. a composite index for (sensor ASC, value ASC, timestamp ASC)

Answer: A

Explanation

If a query has a filter with two or more properties, adding a composite index will improve performance.

Consider the following query:

SELECT * FROM c WHERE c.name = “Tim” and c.age > 18

In the absence of a composite index on (name ASC, and age ASC), we will utilize a range index for this query. We can improve the efficiency of this query by creating a composite index for name and age.

Queries with multiple equality filters and a maximum of one range filter (such as >,<, <=, >=, !=) will utilize the composite index.

Reference:
https://azure.microsoft.com/en-us/blog/three-ways-to-leverage-composite-indexes-in-azure-cosmos-db/

Question #:4 - (Exam Topic 2)

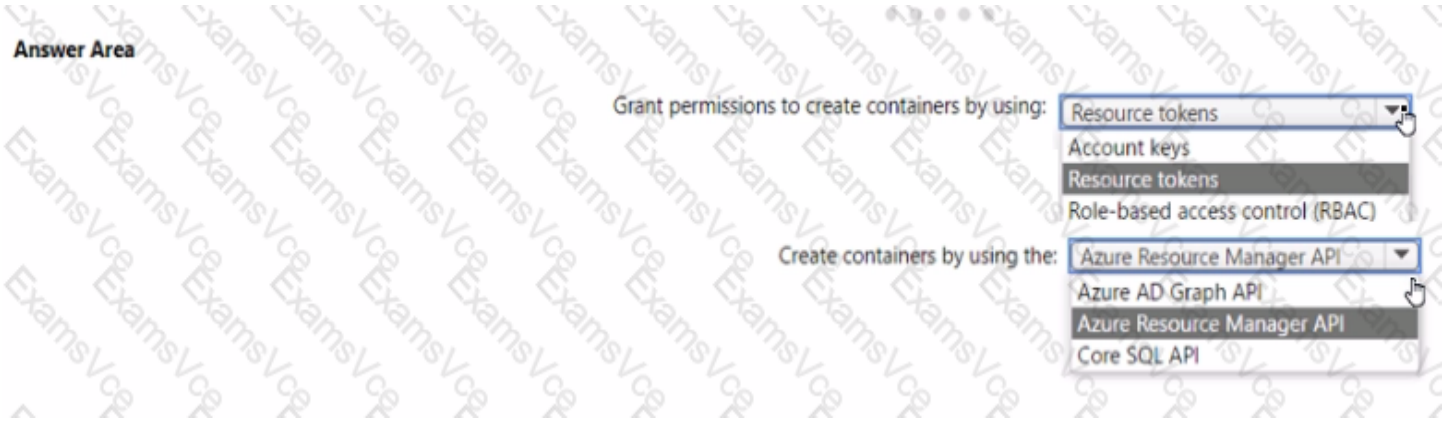
You have an Azure Cosmos DB for NoSQL account named account that has the disablekeyBasedletadatwriteAccess property enabled.

You are developing an app named App1 that will be used by a use1 named DevUser1 to create containers in account1. DevUser1 has a non-privileged user account in the Azure AD tenant.

You need to ensure that DevUser1 can use App1 to create containers in account1.

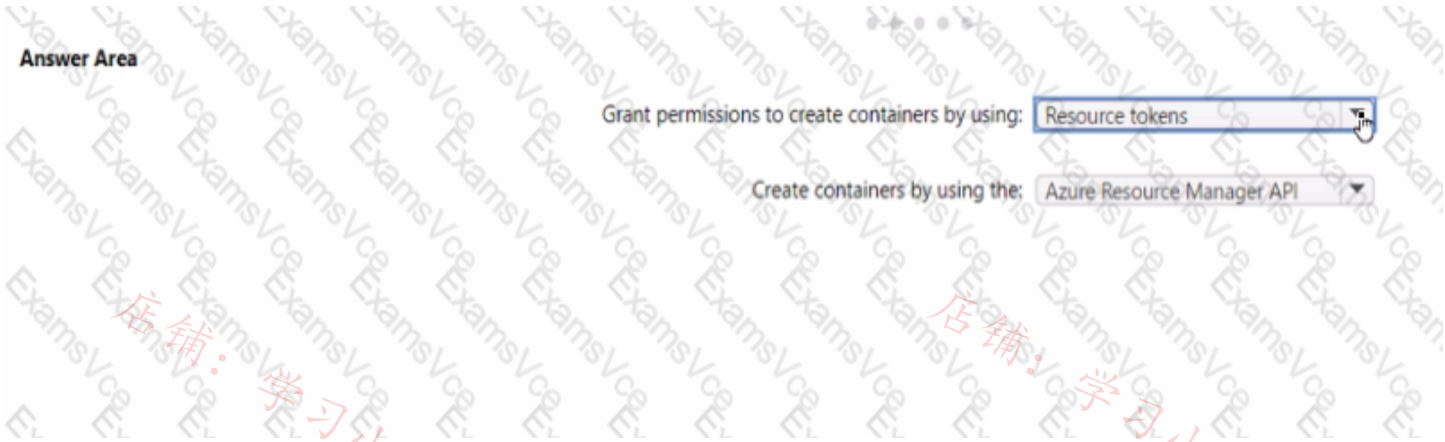
What should you do? To answer, select the appropriate options in the answer area.

NOTE Each correct selection is worth one point.



Answer:

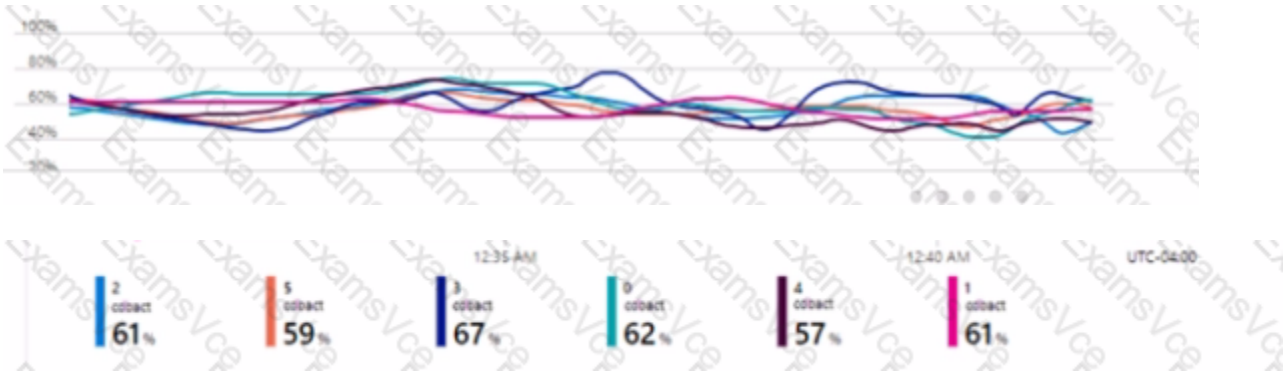
Explanation



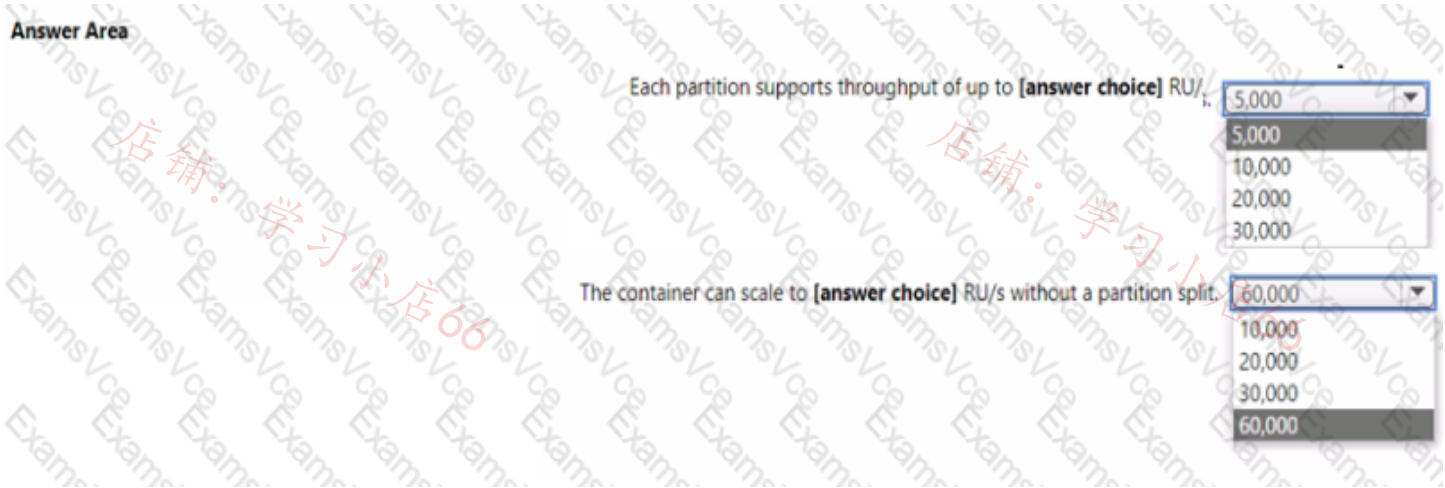
Question #:5 - (Exam Topic 2)

You have a container in an Azure Cosmos DB for NoSQL account. The database that has a manual throughput of 30,000 request units per second (RU/s). The current consumption details are shewn in the following chart.

Normalized RU Consumption (%) By PartitionKeyRangeID

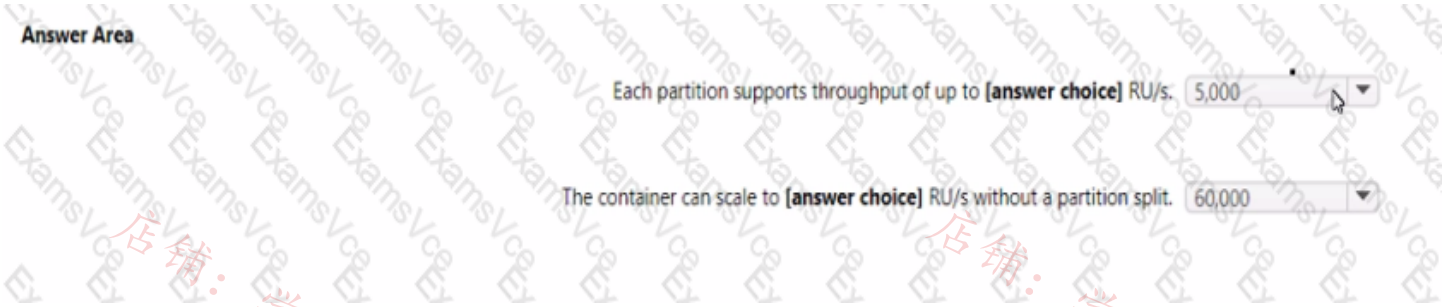


Use the drop-down menus to select the answer choice that answers each question based on the information presented in the graphic. NOTE: Each correct selection is worth one point.



Answer:

Explanation



Question #:6 - (Exam Topic 2)

You maintain a relational database for a book publisher. The database contains the following tables.

Name	Column
Author	authorId **(primary key)**
	fullname
	address
	contactinfo
Book	bookId **(primary key)**
	isbn
	title
	genre
BookauthorInk	authorId **(foreign key)**
	bookId **(foreign key)**

The most common query lists the books for a given authorId.

You need to develop a non-relational data model for Azure Cosmos DB Core (SQL) API that will replace the relational database. The solution must minimize latency and read operation costs.

What should you include in the solution?

A. Create a container for Author and a container for Book. In each Author document, embed booked for each book by the author. In each Book document embed author of each author.

B. Create Author, Book, and Bookauthorlnk documents in the same container.

C. Create a container that contains a document for each Author and a document for each Book. In each Book document, embed authorId.

D. Create a container for Author and a container for Book. In each Author document and Book document embed the data from Bookauthorlnk.

Answer: A

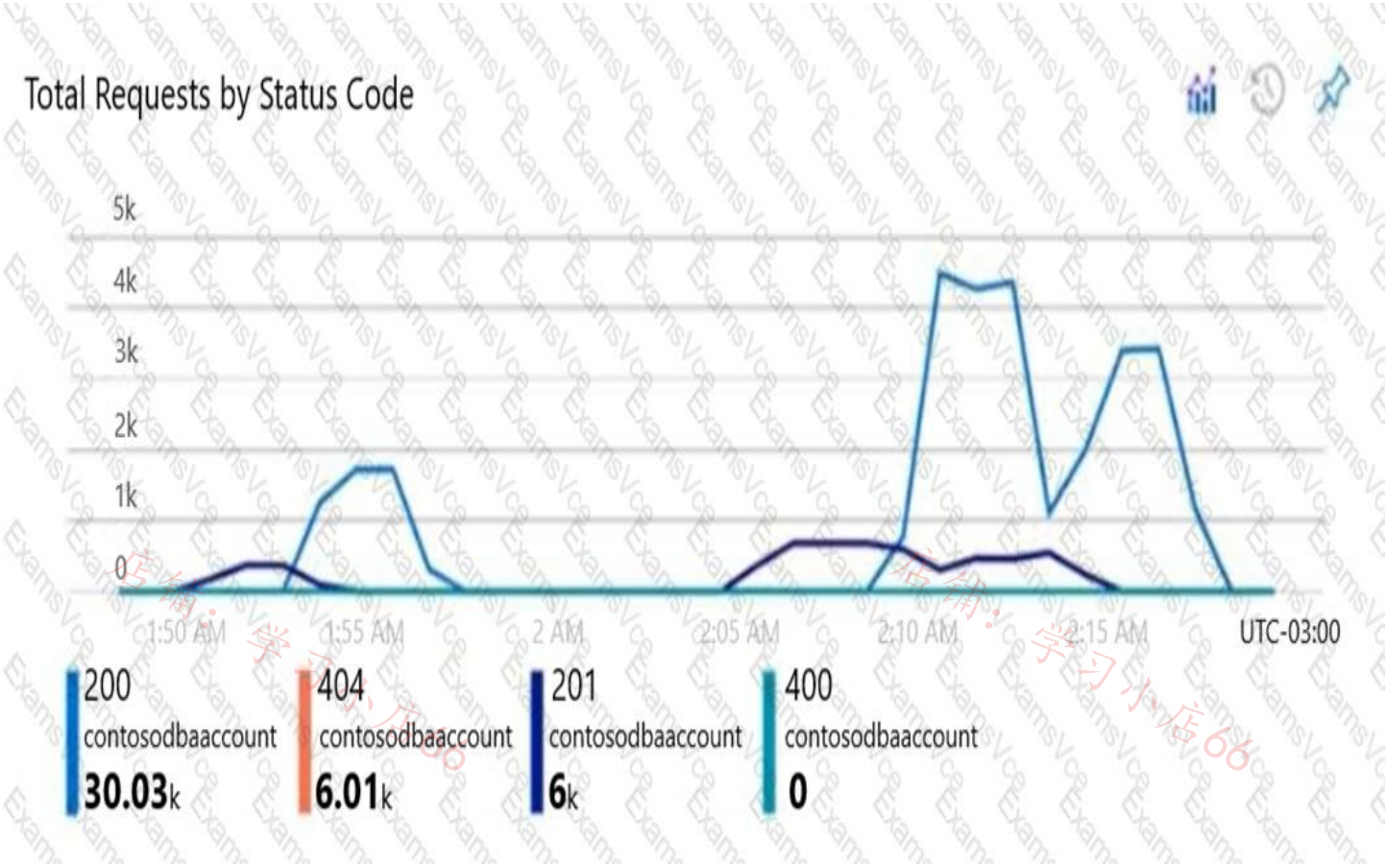
Explanation

Store multiple entity types in the same container.

Question #:7 - (Exam Topic 2)

You have an Azure Cosmos DB Core (SQL) API account used by an application named App1.

You open the Insights pane for the account and see the following chart.



Use the drop-down menus to select the answer choice that answers each question based on the information presented in the graphic.

NOTE: Each correct selection is worth one point.

The HTTP 404 status code is caused by **[answer choice]**

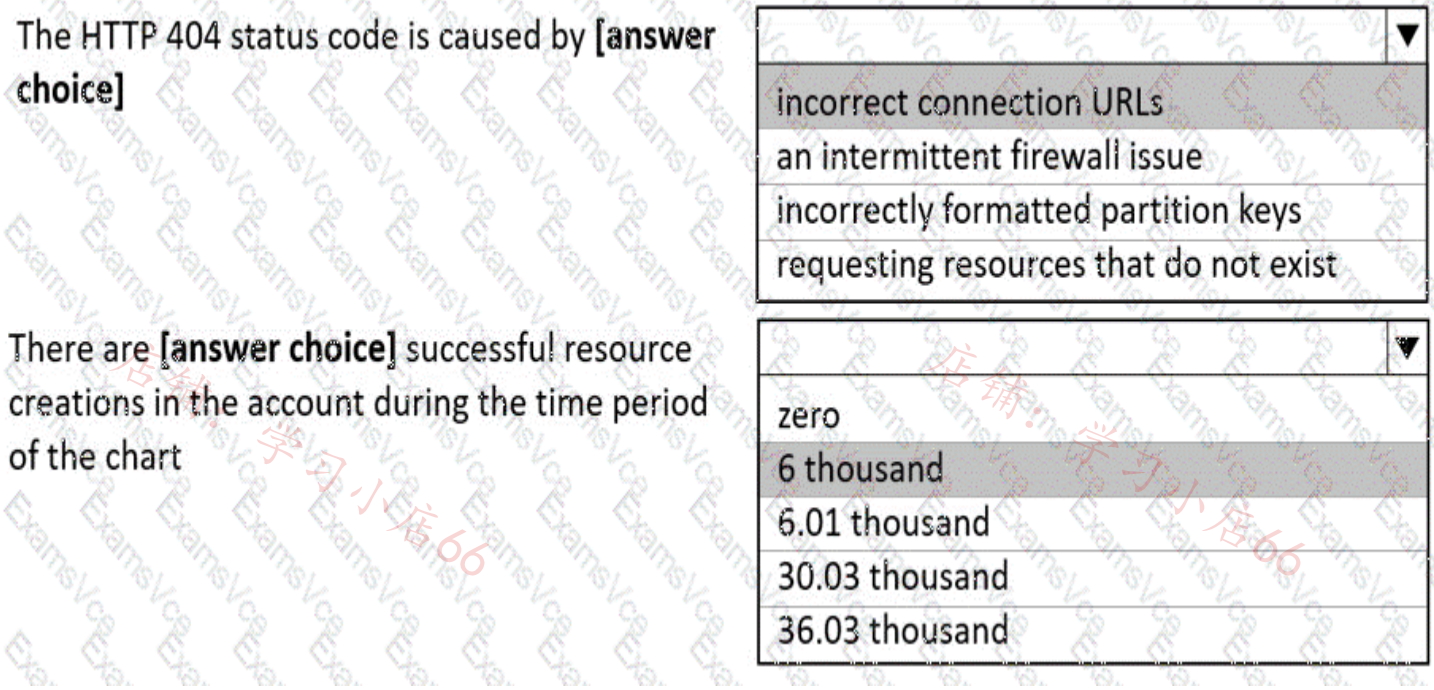
- incorrect connection URLs
- an intermittent firewall issue
- incorrectly formatted partition keys
- requesting resources that do not exist

There are **[answer choice]** successful resource creations in the account during the time period of the chart

- zero
- 6 thousand
- 6.01 thousand
- 30.03 thousand
- 36.03 thousand

Answer:

Explanation



Box 1: incorrect connection URLs

400 Bad Request: Returned when there is an error in the request URI, headers, or body. The response body

will contain an error message explaining what the specific problem is.

The HyperText Transfer Protocol (HTTP) 400 Bad Request response status code indicates that the server cannot or will not process the request due to something that is perceived to be a client error (for example, malformed request syntax, invalid request message framing, or deceptive request routing).

Box 2: 6 thousand

201 Created: Success on PUT or POST. Object created or updated successfully.

Note:

200 OK: Success on GET, PUT, or POST. Returned for a successful response.

404 Not Found: Returned when a resource does not exist on the server. If you are managing or querying an index, check the syntax and verify the index name is specified correctly.

Reference: https://docs.microsoft.com/en-us/rest/api/searchservice/http-status-codes

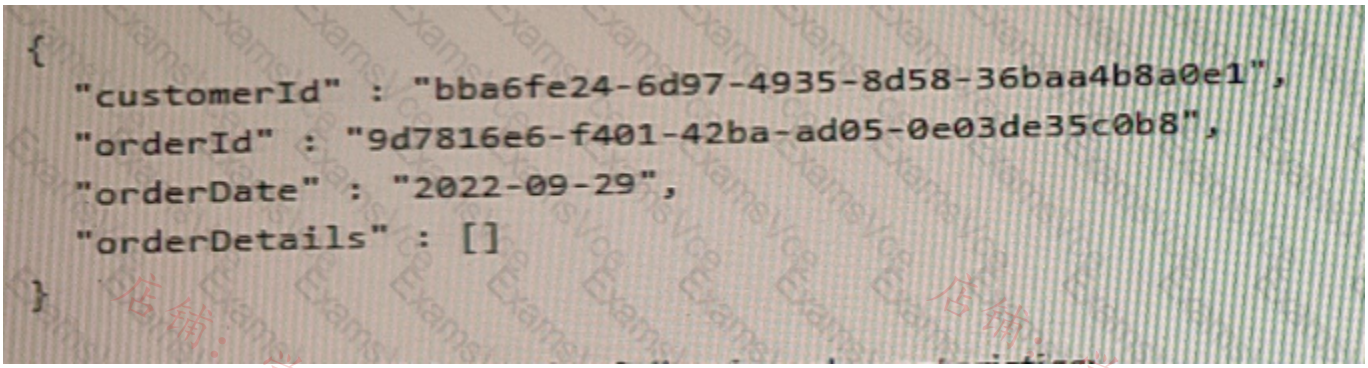
Question #:8 - (Exam Topic 2)

You have a database named db1 in an Azure Cosmos DB for NoSQL

You are designing an application that will use dbl.

In db1, you are creating a new container named coll1 that will store in coll1.

The following is a sample of a document that will be stored in coll1.



The application will have the following characteristics:

• New orders will be created frequently by different customers.

• Customers will often view their past order history.

You need to select the partition key value for coll1 to support the application. The solution must minimize costs.

To what should you set the partition key?

A. id

B. customerId

C. orderDate

D. orderId

Answer: B

Explanation

Based on the characteristics of the application and the provided document structure, the most suitable partition key value for coll1 in the given scenario would be the customerId, Option B.

The application frequently creates new orders by different customers and customers often view their past order history. Using customerId as the partition key would ensure that all orders associated with a particular customer are stored in the same partition. This enables efficient querying of past order history for a specific customer and reduces cross-partition queries, resulting in lower costs and improved performance.

a partition key is a JSON property (or path) within your documents that is used by Azure Cosmos DB to distribute data among multiple partitions**3**. A partition key should have a high cardinality, which means it should have many distinct values, such as hundreds or thousands**1**. A partition key should also align with the most common query patterns of your application, so that you can efficiently retrieve data by using the partition key value**1**.

Based on these criteria, one possible partition key that you could use for coll1 is B. customerId.

This partition key has the following advantages:

- It has a high cardinality, as each customer will have a unique ID**3**.
- It aligns with the query patterns of the application, as customers will often view their past order history**3**.
- It minimizes costs, as it reduces the number of cross-partition queries and optimizes the storage and throughput utilization**1**.

This partition key also has some limitations, such as:

- It may not be optimal for scenarios where orders need to be queried independently from customers or aggregated by date or other criteria**3**.
- It may result in hot partitions or throttling if some customers create orders more frequently than others or have more data than others**1**.
- It may not support transactions across multiple customers, as transactions are scoped to a single logical partition**2**.

Depending on your specific use case and requirements, you may need to adjust this partition key or choose a different one. For example, you could use a synthetic partition key that concatenates multiple properties of an

item**2**, or you could use a partition key with a random or pre-calculated suffix to distribute the workload more evenly**2**.

Question #:9 - (Exam Topic 2)

You need to configure an Apache Kafka instance to ingest data from an Azure Cosmos DB Core (SQL) API account. The data from a container named telemetry must be added to a Kafka topic named iot. The solution must store the data in a compact binary format.

Which three configuration items should you include in the solution? Each correct answer presents part of the solution.

NOTE: Each correct selection is worth one point.

A. "connector.class": "com.azure.cosmos.kafka.connect.source.CosmosDBSourceConnector"

B. "key.converter": "org.apache.kafka.connect.json.JsonConverter"

C. "key.converter": "io.confluent.connect.avro.AvroConverter"

D. "connect.cosmos.containers.topicmap": "iot#telemetry"

E. "connect.cosmos.containers.topicmap": "iot"

F. "connector.class": "com.azure.cosmos.kafka.connect.source.CosmosDBSinkConnector"

Answer: C D F

Explanation

C: Avro is binary format, while JSON is text.

F: Kafka Connect for Azure Cosmos DB is a connector to read from and write data to Azure Cosmos DB. The Azure Cosmos DB sink connector allows you to export data from Apache Kafka topics to an Azure Cosmos DB database. The connector polls data from Kafka to write to containers in the database based on the topics subscription.

D: Create the Azure Cosmos DB sink connector in Kafka Connect. The following JSON body defines config for the sink connector.

Extract:

"connector.class": "com.azure.cosmos.kafka.connect.sink.CosmosDBSinkConnector",

"key.converter": "org.apache.kafka.connect.json.AvroConverter"

"connect.cosmos.containers.topicmap": "hotels#kafka"

Reference:

https://docs.microsoft.com/en-us/azure/cosmos-db/sql/kafka-connector-sink

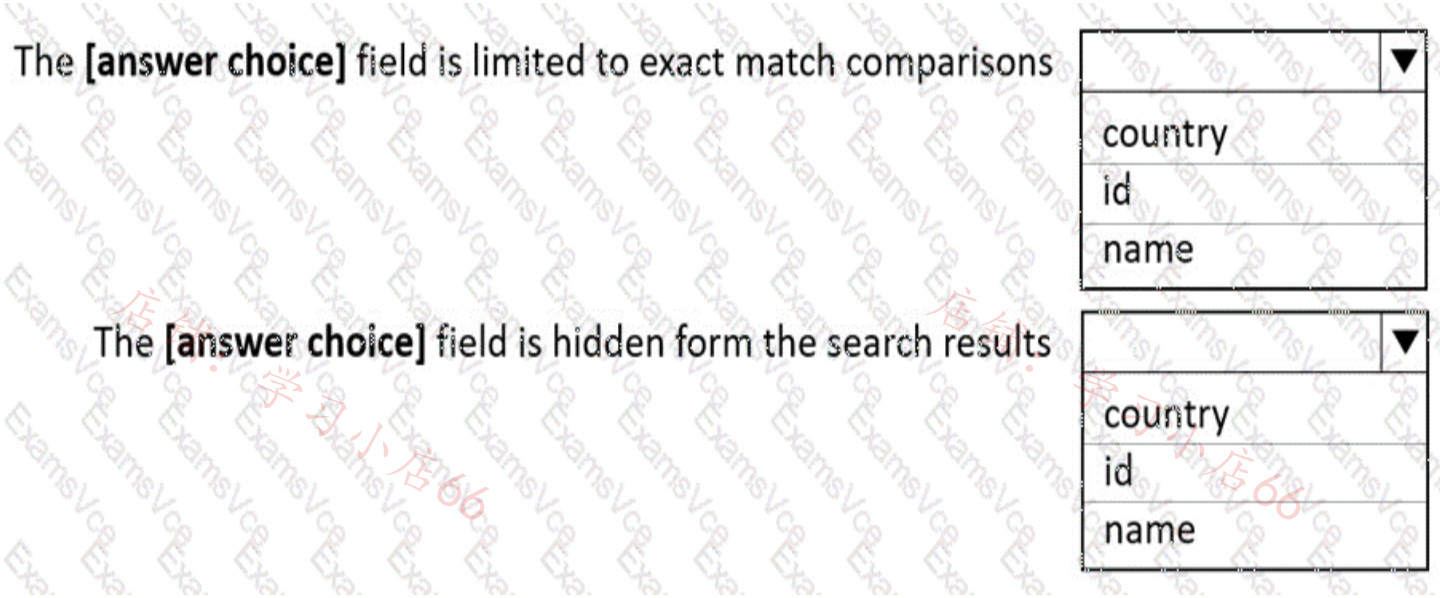
https://www.confluent.io/blog/kafka-connect-deep-dive-converters-serialization-explained/

Question #:10 - (Exam Topic 2)

You configure Azure Cognitive Search to index a container in an Azure Cosmos DB Core (SQL) API account as shown in the following exhibit.

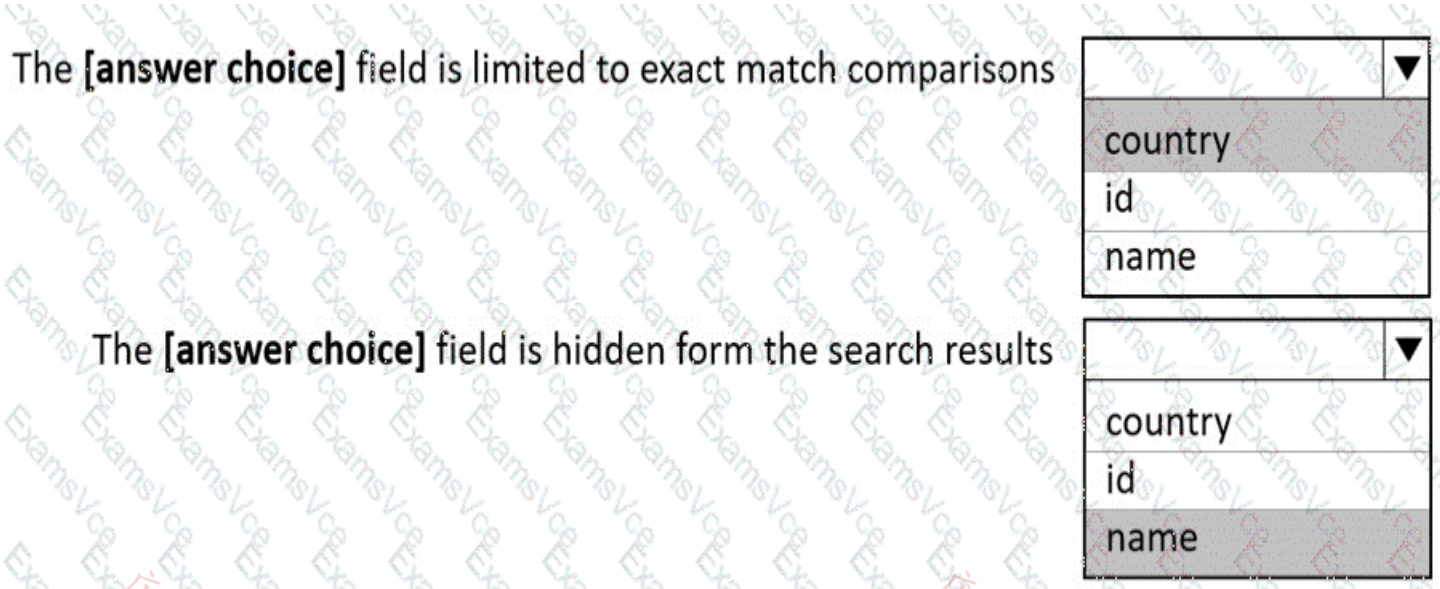
Use the drop-down menus to select the answer choice that completes each statement based on the information presented in the graphic.

NOTE: Each correct selection is worth one point.



Answer:

Explanation



Box 1: country

The country field is filterable.

Note: filterable: Indicates whether to enable the field to be referenced in $filter queries. Filterable differs from searchable in how strings are handled. Fields of type Edm.String or Collection(Edm.String) that are filterable do not undergo lexical analysis, so comparisons are for exact matches only.

Box 2: name

The name field is not Retrievable.

Retrievable: Indicates whether the field can be returned in a search result. Set this attribute to false if you want to use a field (for example, margin) as a filter, sorting, or scoring mechanism but do not want the field to be visible to the end user.

Note: searchable: Indicates whether the field is full-text searchable and can be referenced in search queries.

Reference: https://docs.microsoft.com/en-us/rest/api/searchservice/create-index

Question #:11 - (Exam Topic 2)

You have operational data in an Azure Cosmos OB for NoSQL database.

Database users report that the performance of the database degrades significantly when a business analytics team runs large Apache Spark-based queries against the database.

You need 10 reduce the impact that running the Spark-based queries has on the database users.

What should you implement?

A. Azure Synapse Link

B. a default consistency level of Consistent Prefix

C. a default consistency level of Strong

D. the Spark connector

Answer: A

Question #:12 - (Exam Topic 2)

You have an Azure Cosmos DB Core (SQL) API account.

You configure the diagnostic settings to send all log information to a Log Analytics workspace.

You need to identify when the provisioned request units per second (RU/s) for resources within the account were modified.

You write the following query.

```
AzureDiagnostics
| where Category == "ControlPlaneRequests"
```

What should you include in the query?

A. | where OperationName startswith "AccountUpdateStart"

B. | where OperationName startswith "SqlContainersDelete"

C. | where OperationName startswith "MongoCollectionsThroughputUpdate"

D. | where OperationName startswith "SqlContainersThroughputUpdate"

Answer: A

Explanation

The following are the operation names in diagnostic logs for different operations:

RegionAddStart, RegionAddComplete

RegionRemoveStart, RegionRemoveComplete

AccountDeleteStart, AccountDeleteComplete

RegionFailoverStart, RegionFailoverComplete

AccountCreateStart, AccountCreateComplete

AccountUpdateStart, AccountUpdateComplete

VirtualNetworkDeleteStart, VirtualNetworkDeleteComplete

DiagnosticLogUpdateStart, DiagnosticLogUpdateComplete

Reference: https://docs.microsoft.com/en-us/azure/cosmos-db/audit-control-plane-logs

Question #:13 - (Exam Topic 2)

You have an Azure Cosmos DB Core (SQL) API account that is used by 10 web apps.

You need to analyze the data stored in the account by using Apache Spark to create machine learning models. The solution must NOT affect the performance of the web apps.

Which two actions should you perform? Each correct answer presents part of the solution.

NOTE: Each correct selection is worth one point.

A. In an Apache Spark pool in Azure Synapse, create a table that uses cosmos.olap as the data source.

B. Create a private endpoint connection to the account.

C. In an Azure Synapse Analytics serverless SQL pool, create a view that uses OPENROWSET and the CosmosDB provider.

D. Enable Azure Synapse Link for the account and Analytical store on the container.

E. In an Apache Spark pool in Azure Synapse, create a table that uses cosmos.oltp as the data source.

Answer: A D

Reference:

https://github.com/microsoft/MCW-Cosmos-DB-Real-Time-Advanced-Analytics/blob/main/Hands-on%20lab/H

Question #:14 - (Exam Topic 2)

You have a container in an Azure Cosmos DB Core (SQL) API account. The container stores telemetry data from IoT devices. The container uses telemetryId as the partition key and has a throughput of 1,000 request units per second (RU/s). Approximately 5,000 IoT devices submit data every five minutes by using the same telemetryId value.

You have an application that performs analytics on the data and frequently reads telemetry data for a single IoT device to perform trend analysis.

The following is a sample of a document in the container.

```
{
    "id" : "9ccf1906-2a30-4dc0-9644-2185f5dcbbd7",
    "deviceId" : "bba6fe24-6d97-4935-8d58-36baa4b8a0e1",
    "telemetryId" : "9d7816e6-f401-42ba-ad05-0e03de35c0b8",
    "date" : "2019-05-03",
    "time" : "13:05",
    "temp" : "21"
}
```

You need to reduce the amount of request units (RUs) consumed by the analytics application.

What should you do?

A. Decrease the offerThroughput value for the container.

B. Increase the offerThroughput value for the container.

C. Move the data to a new container that has a partition key of deviceId.

D. Move the data to a new container that uses a partition key of date.

Answer: C

Explanation

The partition key is what will determine how data is routed in the various partitions by Cosmos DB and needs

to make sense in the context of your specific scenario. The IoT Device ID is generally the "natural" partition key for IoT applications.

Reference: https://docs.microsoft.com/en-us/azure/architecture/solution-ideas/articles/iot-using-cosmos-db

Question #:15 - (Exam Topic 2)

You plan to store order data in Azure Cosmos DB for NoSQL account. The data contains information about orders and their associated items.

You need to develop a model that supports order read operations. The solution must minimize the number or requests.

A. Create a single database that contains one container. Store orders and order items in separate documents in the container.

B. Create a single database that contains one container. Create a separate document for each order and embed the order items into the order documents.

C. Create a database for orders and a database for order items.

D. Create a single database that contains a container for order and a container for order items.

Answer: B

Explanation

Azure Cosmos DB is a multi-model database that supports various data models, such as documents, key-value, graph, and column-family**3**. The core content-model of Cosmos DB's database engine is based on atom-record-sequence (ARS), which allows it to store and query different types of data in a flexible and efficient way**3**.

To develop a model that supports order read operations and minimizes the number of requests, you should consider the following factors:

- The size and shape of your data
- The frequency and complexity of your queries
- The latency and throughput requirements of your application
- The trade-offs between storage efficiency and query performance

Based on these factors, one possible model that you could implement is B. Create a single database that contains one container. Create a separate document for each order and embed the order items into the order documents.

This model has the following advantages:

- It stores orders and order items as self-contained documents that can be easily retrieved by order ID**1**.

- It avoids storing redundant data or creating additional containers for order items**1**.
- It allows you to view the order history of a customer with simple queries**1**.
- It leverages the benefits of embedding data, such as reducing the number of requests, improving query performance, and simplifying data consistency**2**.

This model also has some limitations, such as:

- It may not be suitable for some order items that have data that is greater than 2 KB, as it could exceed the maximum document size limit of 2 MB**2**.
- It may not be optimal for scenarios where order items need to be queried independently from orders or aggregated by other criteria**2**.
- It may not support transactions across multiple orders or customers, as transactions are scoped to a single logical partition**2**.

Depending on your specific use case and requirements, you may need to adjust this model or choose a different one. For example, you could use a hybrid data model that combines embedding and referencing data**2**, or you could use a graph data model that expresses entities and relationships as vertices and edges.

Question #:16 - (Exam Topic 2)

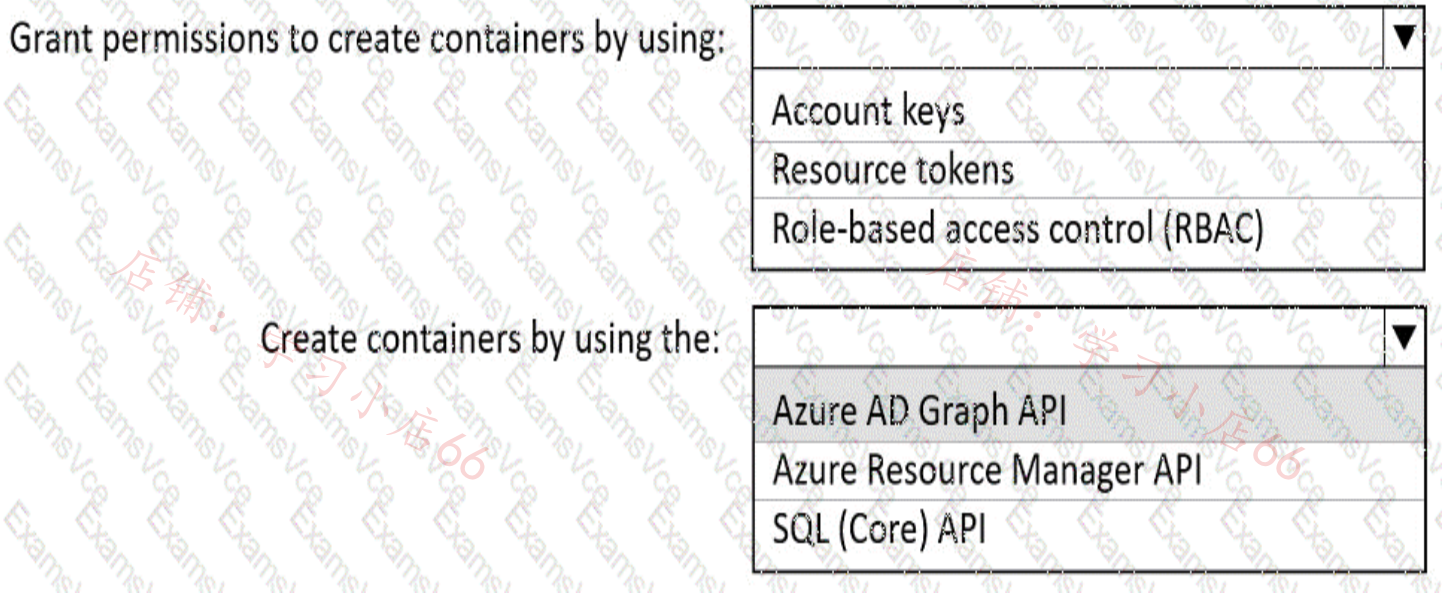
You have an Azure Cosmos DB Core (SQL) API account named account1 that has the disableKeyBasedMetadataWriteAccess property enabled.

You are developing an app named App1 that will be used by a user named DevUser1 to create containers in account1. DevUser1 has a non-privileged user account in the Azure Active Directory (Azure AD) tenant.

You need to ensure that DevUser1 can use App1 to create containers in account1.

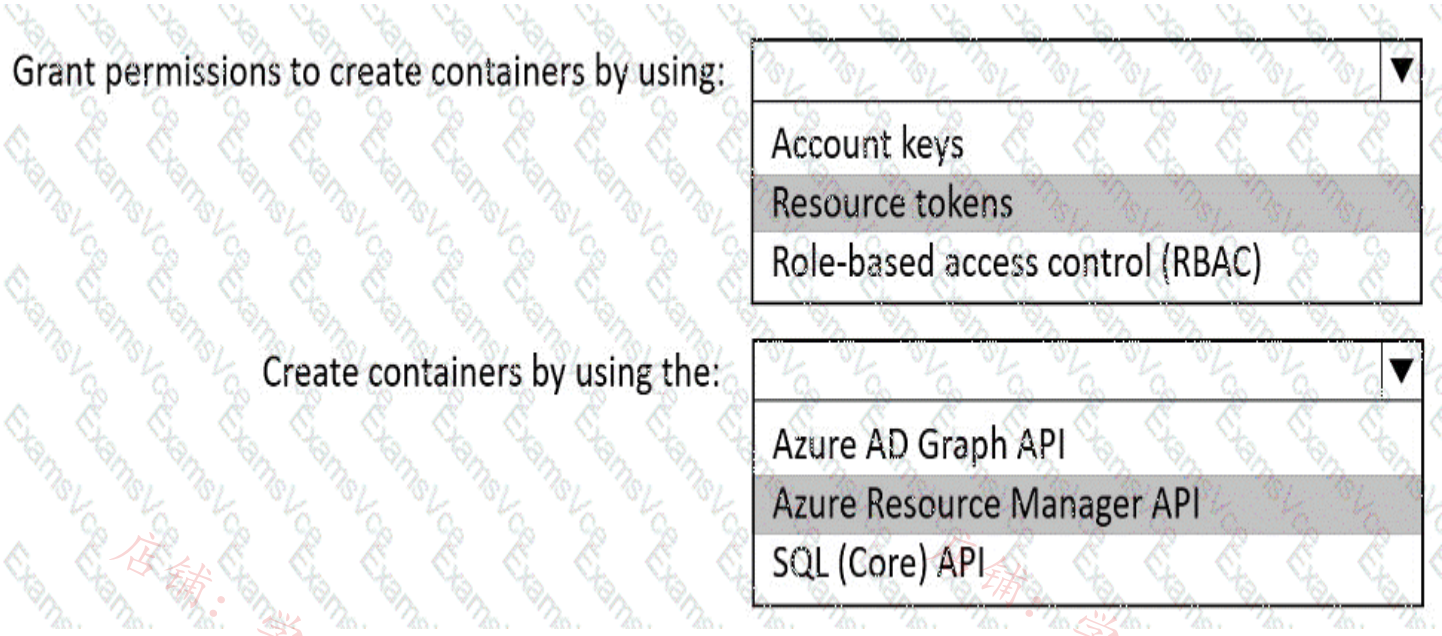
What should you do? To answer, select the appropriate options in the answer area.

NOTE: Each correct selection is worth one point.



Answer:

Explanation



Box 1: Resource tokens

Resource tokens provide access to the application resources within a database. Resource tokens:

Provide access to specific containers, partition keys, documents, attachments, stored procedures, triggers, and UDFs.

Box 2: Azure Resource Manager API

You can use Azure Resource Manager to help deploy and manage your Azure Cosmos DB accounts,

databases, and containers.

Reference:

https://docs.microsoft.com/en-us/azure/cosmos-db/secure-access-to-data

https://docs.microsoft.com/en-us/rest/api/resources/

Question #:17 - (Exam Topic 2)

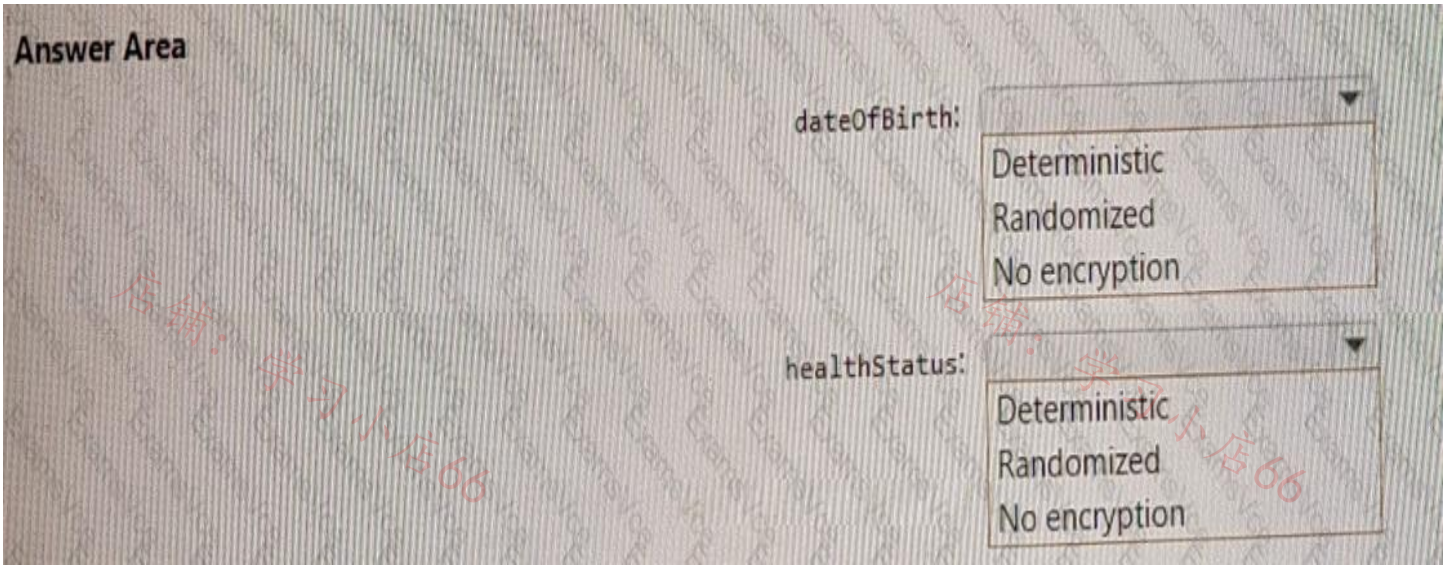
You have an Azure Cosmos DB for NoSQL container. The container contains items that have the following properties.

Property	Data type	Filtered in queries
dateOfBirth	Date	Yes
hasProvidedTaxNumber	Boolean	Yes
healthStatus	String	No

You need to protect the data stored in the container by using Always Encrypted. For each property, you must use the strongest type of encryption and ensure that queries execute properly.

What is the strongest type of encryption that you can apply to each property? To answer, select the appropriate options in the answer area.

NOTE: Each correct selection is worth one point.



Answer:

Explanation

Box 1 = Randomized

Box 2 = Deterministic

Always Encrypted for Azure Cosmos DB supports two types of encryption: deterministic and randomized**1**. Deterministic encryption always produces the same encrypted value for any given plain text value. Randomized encryption produces a different encrypted value for the same plain text value.

For dateOfBirth, randomized encryption is the strongest type of encryption because it provides better protection against statistical analysis and brute-force attacks. Deterministic encryption would not be suitable for dateOfBirth because it could reveal patterns or allow equality comparisons**1**.

For healthStatus, deterministic encryption is the strongest type of encryption because it allows queries to perform equality comparisons and filters on the encrypted property. Randomized encryption would not be suitable for healthStatus because it would prevent any queries on the encrypted property**1**.

Question #:18 - (Exam Topic 2)

You plan to create an Azure Cosmos DB Core (SQL) API account that will use customer-managed keys stored in Azure Key Vault.

You need to configure an access policy in Key Vault to allow Azure Cosmos DB access to the keys.

Which three permissions should you enable in the access policy? Each correct answer presents part of the solution.

NOTE: Each correct selection is worth one point.

A. Wrap Key

B. Get

C. List

D. Update

E. Sign

F. Verify

G. Unwrap Key

Answer: A B G

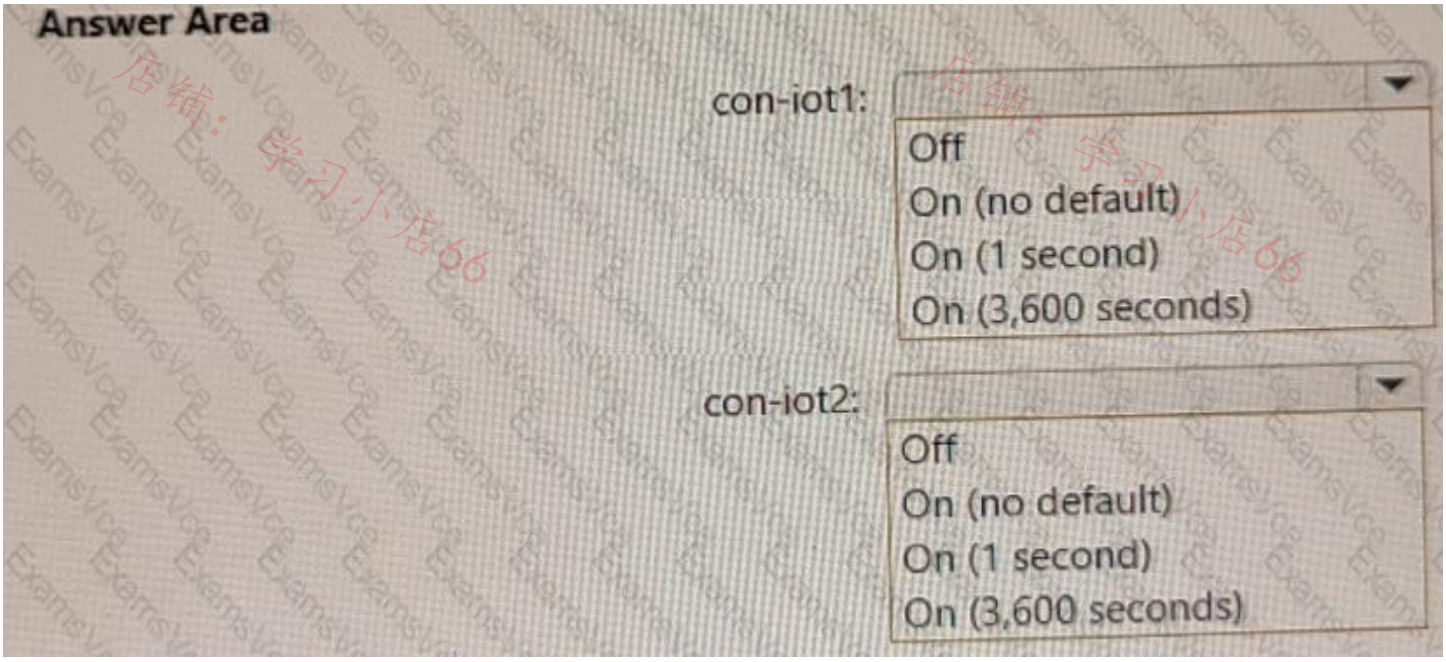
Reference: https://docs.microsoft.com/en-us/azure/cosmos-db/how-to-setup-cmk

Question #:19 - (Exam Topic 2)

You plan to implement con-iot1 and con-iot2.

You need to configure the default Time to Live setting for each container. The solution must meet the loT telemetry requirements.

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



Answer:

Explanation

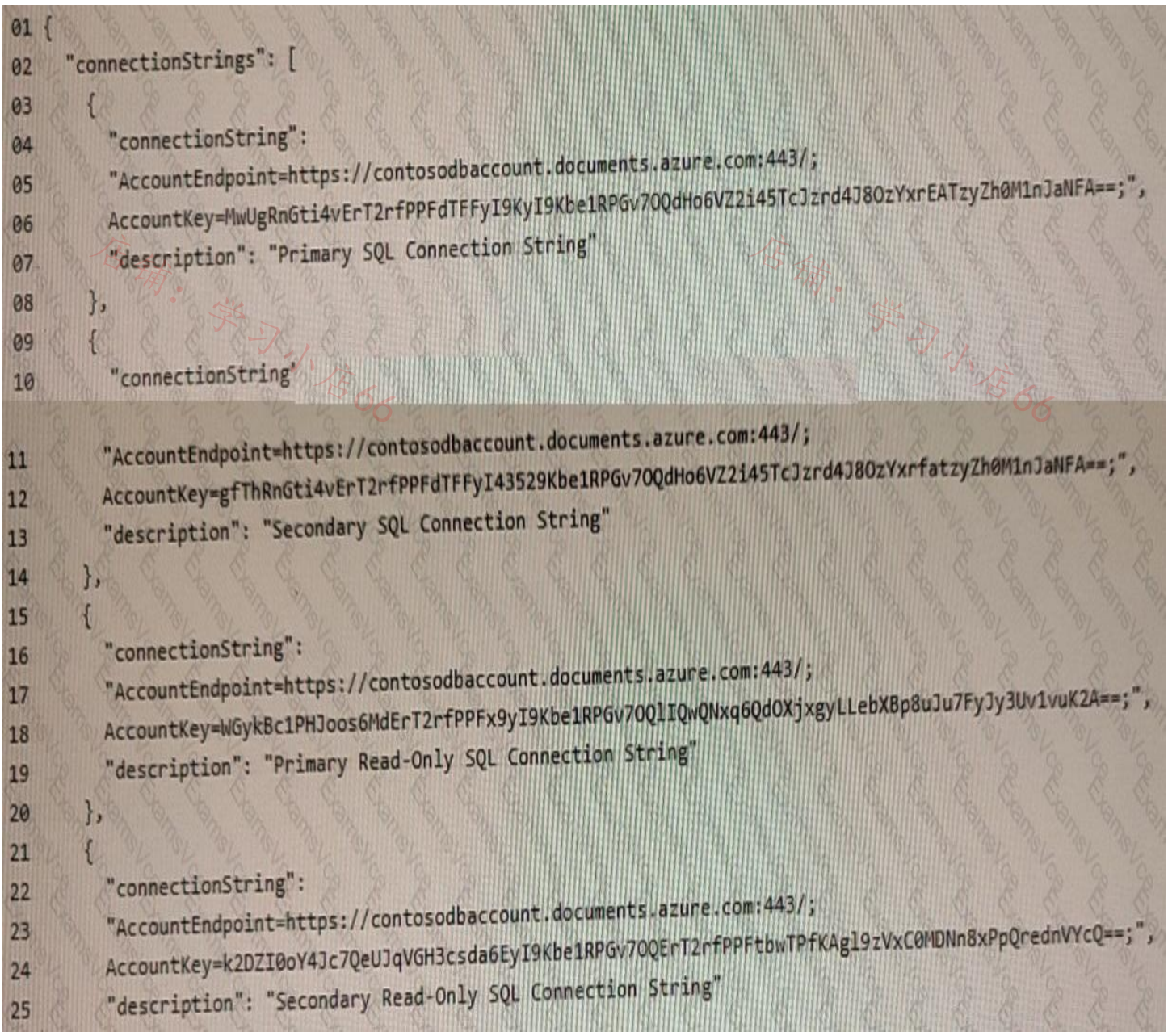
Box 1 = On (no default)For con-iot1, you need to configure the default TTL setting to On (no default), which means that items in this container do not expire by default, but you can override the TTL value on a per-item basis. This meets the requirement of retaining all telemetry data unless overridden**1**.

Box 2 = On (3600 seconds)For con-iot2, you need to configure the default TTL setting to On (3600 seconds), which means that items in this container will expire 3600 seconds (one hour) after their last modified time. This meets the requirement of deleting all telemetry data older than one hour**1**.

Question #:20 - (Exam Topic 2)

You are developing an application that will connect to an Azure Cosmos DB for NoSQL account. The account has a single readme region and one agonal read region. The regions are configured for automatic failover.

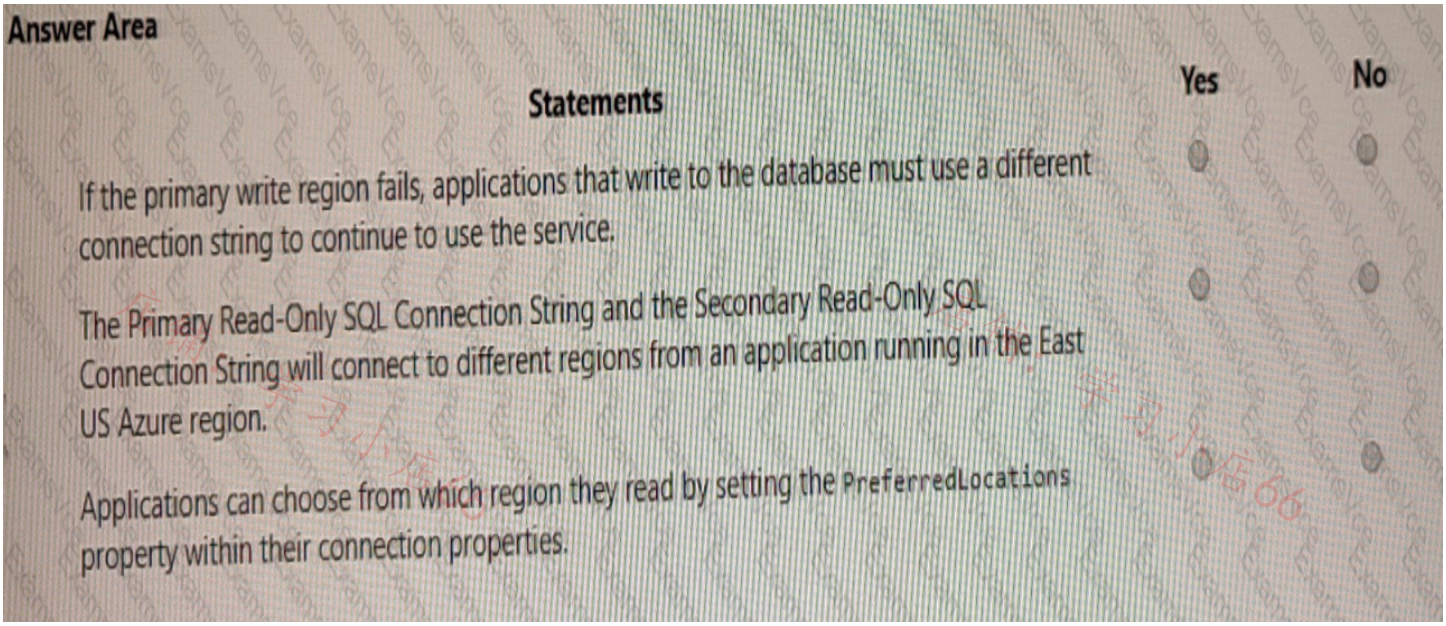
The account has the following connect strings. (Line numbers are included for reference only.)



```
{
  "connectionStrings": [
    {
      "connectionString":
      "AccountEndpoint=https://contosodbaccount.documents.azure.com:443/;
      AccountKey=MwUgRnGti4vErT2rfPPFdTFFyI9KyI9Kbe1RPGv7OQdHo6VZ2i45TcJzrd4J8OzYxrEATzyZh0M1nJaNFA==;",
      "description": "Primary SQL Connection String"
    },
    {
      "connectionString"
      "AccountEndpoint=https://contosodbaccount.documents.azure.com:443/;
      AccountKey=gfThRnGti4vErT2rfPPFdTFFyI43529Kbe1RPGv7OQdHo6VZ2i45TcJzrd4J8OzYxrfatzyZh0M1nJaNFA==;",
      "description": "Secondary SQL Connection String"
    },
    {
      "connectionString":
      "AccountEndpoint=https://contosodbaccount.documents.azure.com:443/;
      AccountKey=WGykBc1PHJoos6MdErT2rfPPFx9yI9Kbe1RPGv7OQlIQwQNxq6QdOXjxgyLLebXBp8uJu7FyJy3Uv1vuK2A==;",
      "description": "Primary Read-Only SQL Connection String"
    },
    {
      "connectionString":
      "AccountEndpoint=https://contosodbaccount.documents.azure.com:443/;
      AccountKey=k2DZI0oY4Jc7QeUJqVGH3csda6EyI9Kbe1RPGv7OQErT2rfPPFtbwTPfKAgl9zVxC0MDNn8xPpQrednVYcQ==;",
      "description": "Secondary Read-Only SQL Connection String"
```

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:

Explanation

If the primary write region fails, applications that write to the database must use a different connection string to continue to use the service. = NOYou do not need to use a different connection string to continue to use the service if the primary write region fails. This is because Azure Cosmos DB supports automatic failover, which means that it will automatically switch the primary write region to another region in case of a regional outage**2**. The application does not need to change the connection string or specify the failover priority**3**. The connection string contains a list of all the regions associated with your account, and Azure Cosmos DB will route the requests to the appropriate region based on the availability and latency**1**.

The primary Read-Only SQL Connection String and the Secondary Read-Only SQL Connection String will connect to different regions from an application running in the East US Azure region = YesThe primary read-only SQL connection string and the secondary read-only SQL connection string will connect to different regions from an application running in the East US Azure region. This is because the primary read-only SQL connection string contains the endpoint for the East US region, which is the same as the primary write region. The secondary read-only SQL connection string contains the endpoint for the West US region, which is the additional read region. Therefore, if an application running in the East US Azure region uses these connection strings, it will connect to different regions depending on which one it chooses.

Applications can choose from which region by setting the PreferredLocations property within their connection properties = Yes

Applications can choose from which region by setting the PreferredLocations property within their connection properties. This property allows you to specify a list of regions that you prefer to read from based on their proximity to your application**2**. Azure Cosmos DB will route the requests to the appropriate region based on the availability and latency**1**. You can also set the ApplicationRegion property to the region where your application is deployed, and Azure Cosmos DB will automatically populate the PreferredLocations property based on the geo-proximity from that location**1**.

Question #:21 - (Exam Topic 2)

Note: This question is part of a series of questions that present the same scenario. Each question in the series contains a unique solution that might meet the stated goals. Some question sett might have more than one correct solution, while others might not have a correct solution.

After you answer a question in this section, you will NOT be able to return to it. As a result, these questions will not appear in the review screen.

You have a container named conlainer1 in an Azure Cosmos DB for NoSQL account.

You need to make the contents of container1 available as reference data for an Azure Stream Analytics job.

Solution: You create an Azure function to copy data to another Azure Cosmos DB for NoSQL container.

Does this meet the goal?

A. Yes

B. No

Answer: B

Question #:22 - (Exam Topic 2)

The following is a sample of a document in orders.

```
{
    "orderId" : "d4a91979b-5ead-43a3-b851-add9a71ac4b6",
    "customerId" : "f6e39103-bdc7-4346-9cfb-45daa4b2becf",
    "orderDate" : "2021-09-29",
  "orderItems" : [
    {
    "itemId" : "6c30412f-3cd7-4cab-813c-05942345720d",
    "name": "blue pen",
    "type" : "pens",
    "count" : 10,
    },
    ...
  ],
  "total" : 12345,
  "status" : "ordered"
}
```

The orders container uses customer as the partition key.

You need to provide a report of the total items ordered per month by item type. The solution must meet the following requirements:

Ensure that the report can run as quickly as possible.

Minimize the consumption of request units (RUs).

What should you do?

A. Configure the report to query orders by using a SQL query.

B. Configure the report to query a new aggregate container. Populate the aggregates by using the change feed.

C. Configure the report to query orders by using a SQL query through a dedicated gateway.

D. Configure the report to query a new aggregate container. Populate the aggregates by using SQL queries that run daily.

Answer: D

Explanation

You can facilitate aggregate data by using Change Feed and Azure Functions, and then use it for reporting.

Reference: https://docs.microsoft.com/en-us/azure/cosmos-db/change-feed

Question #:23 - (Exam Topic 2)

You are creating a database in an Azure Cosmos DB Core (SQL) API account. The database will be used by an application that will provide users with the ability to share online posts. Users will also be able to submit comments on other users' posts.

You need to store the data shown in the following table.

Type	Description
Users	Information about a user who will use the application
Posts	Text of up to 1,000 characters that a user will share with other users
Comments	Text of up to 280 characters that users will submit as a comment on a post
Interests	Information about a user's interests

The application has the following characteristics:

Users can submit an unlimited number of posts.

The average number of posts submitted by a user will be more than 1,000.

Posts can have an unlimited number of comments from different users.

The average number of comments per post will be 100, but many posts will exceed 1,000 comments.

Users will be limited to having a maximum of 20 interests.

For each of the following statements, select Yes if the statement is true. Otherwise, select No.

NOTE: Each correct selection is worth one point.

Statements	Yes	No
If you embed the posts data into the users data instead of creating a separate document for each post, you will increase the write operation costs for new posts	○	○
If you embed the comments data into the posts data instead of creating a separate document for each comment you will increase the write operation costs for new comments	○	○
If you embed the interests data into the users data instead of creating a separate document for each interest, you will increase the read operation costs for displaying the users and their associated interests	○	○

Answer:

Explanation

Statements	Yes	No
If you embed the posts data into the users data instead of creating a separate document for each post, you will increase the write operation costs for new posts	○	○
If you embed the comments data into the posts data instead of creating a separate document for each comment you will increase the write operation costs for new comments	○	○
If you embed the interests data into the users data instead of creating a separate document for each interest, you will increase the read operation costs for displaying the users and their associated interests	○	○

Box 1: Yes

Non-relational data increases write costs, but can decrease read costs.

Box 2: Yes

Non-relational data increases write costs, but can decrease read costs.

Box 3: No

Non-relational data increases write costs, but can decrease read costs.

Question #:24 - (Exam Topic 2)

You have a container named container1 in an Azure Cosmos DB Core (SQL) API account.

The following is a sample of a document in container1.

```
{
"studentId": "631282",
"firstName": "James",
"lastName": "Smith",
"enrollmentYear": 1990,
"isActivelyEnrolled": true,
"address": {
"street": "",
"city": "",
"stateProvince": "",
"postal": "",
}
}
```

The container1 container has the following indexing policy.

```
{
"indexingMode": "consistent",
```

"includePaths": [

{

"path": "/*"

},

{

"path": "/address/city/?"

}

],

"excludePaths": [

{

"path": "/address/*"

},

{

"path": "/firstName/?"

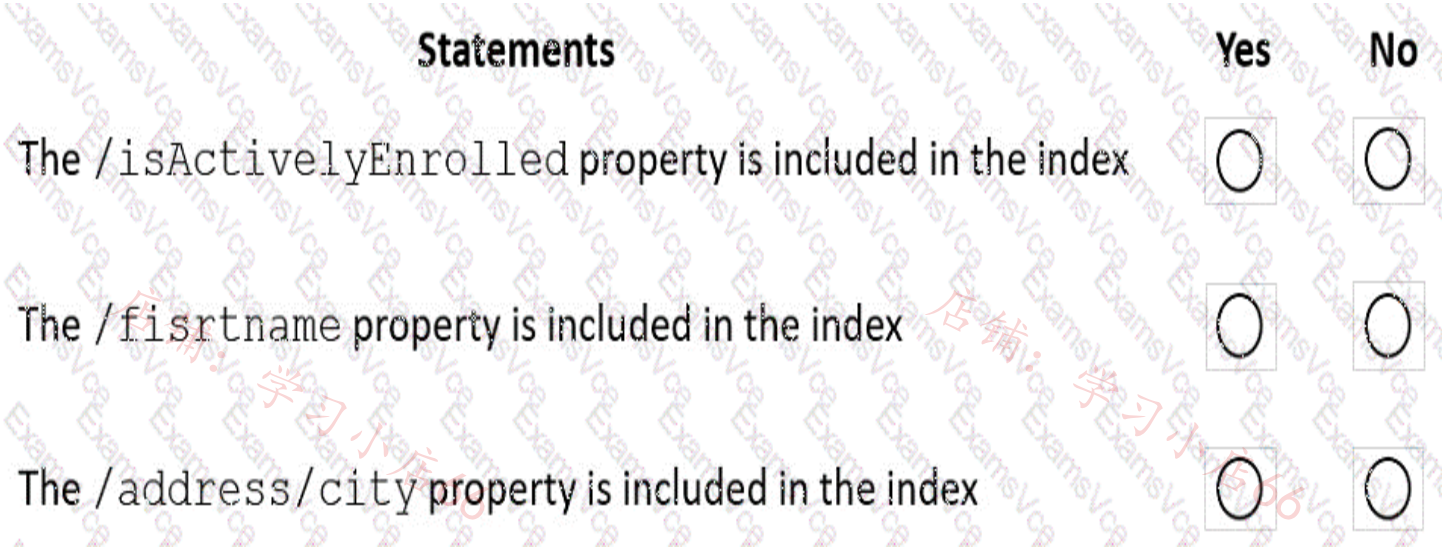
}

]

}

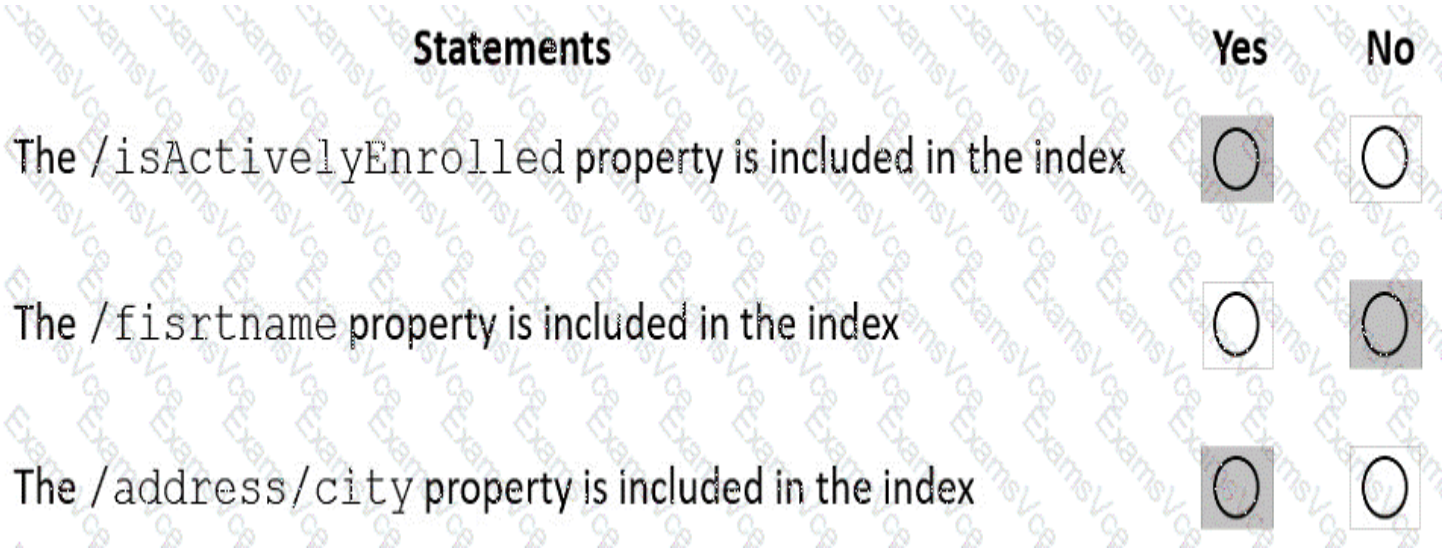
For each of the following statements, select Yes if the statement is true. Otherwise, select No.

NOTE: Each correct selection is worth one point.



Answer:

Explanation



Box 1: Yes

"path": "/*" is in includePaths.

Include the root path to selectively exclude paths that don't need to be indexed. This is the recommended approach as it lets Azure Cosmos DB proactively index any new property that may be added to your model.

Box 2: No

"path": "/firstName/?" is in excludePaths.

Box 3: Yes

"path": "/address/city/?" is in includePaths

Reference: https://docs.microsoft.com/en-us/azure/cosmos-db/index-policy

Question #:25 - (Exam Topic 2)

You have an Azure Cosmos DB Core (SQL) API account that is configured for multi-region writes. The account contains a database that has two containers named container1 and container2.

The following is a sample of a document in container1:

{

"customerId": 1234,

"firstName": "John",

"lastName": "Smith",

"policyYear": 2021

}

The following is a sample of a document in container2:

{

"gpsId": 1234,

"latitude": 38.8951,

"longitude": -77.0364

}

You need to configure conflict resolution to meet the following requirements:

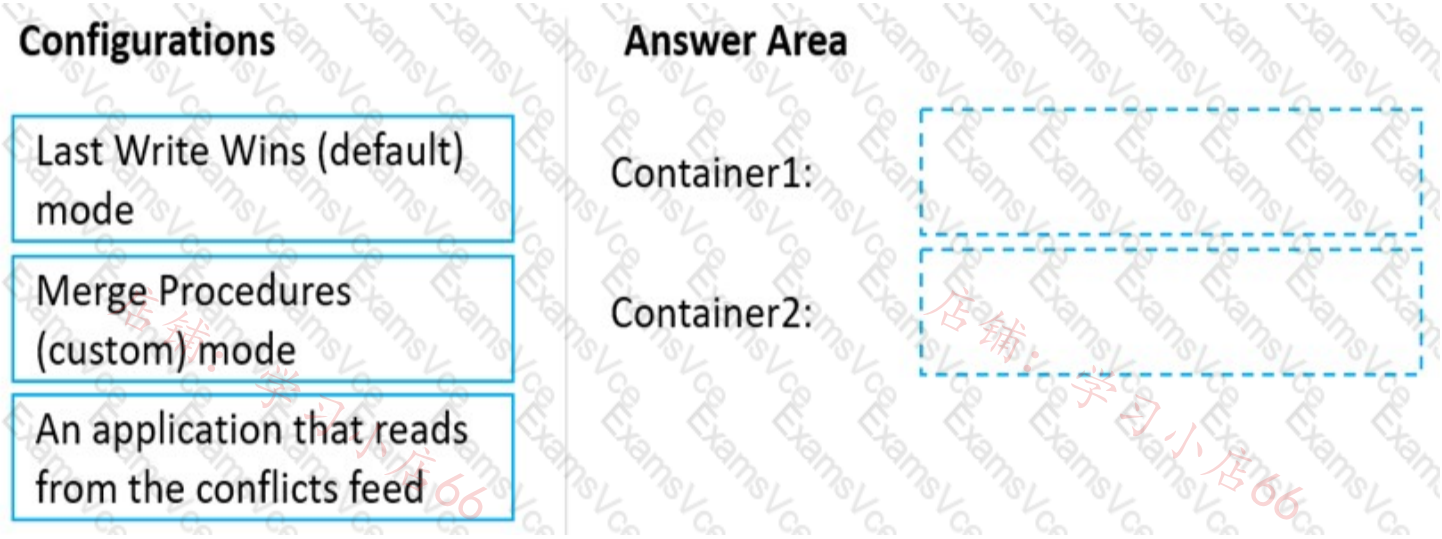
For container1 you must resolve conflicts by using the highest value for policyYear.

For container2 you must resolve conflicts by accepting the distance closest to latitude: 40.730610 and longitude: -73.935242.

Administrative effort must be minimized to implement the solution.

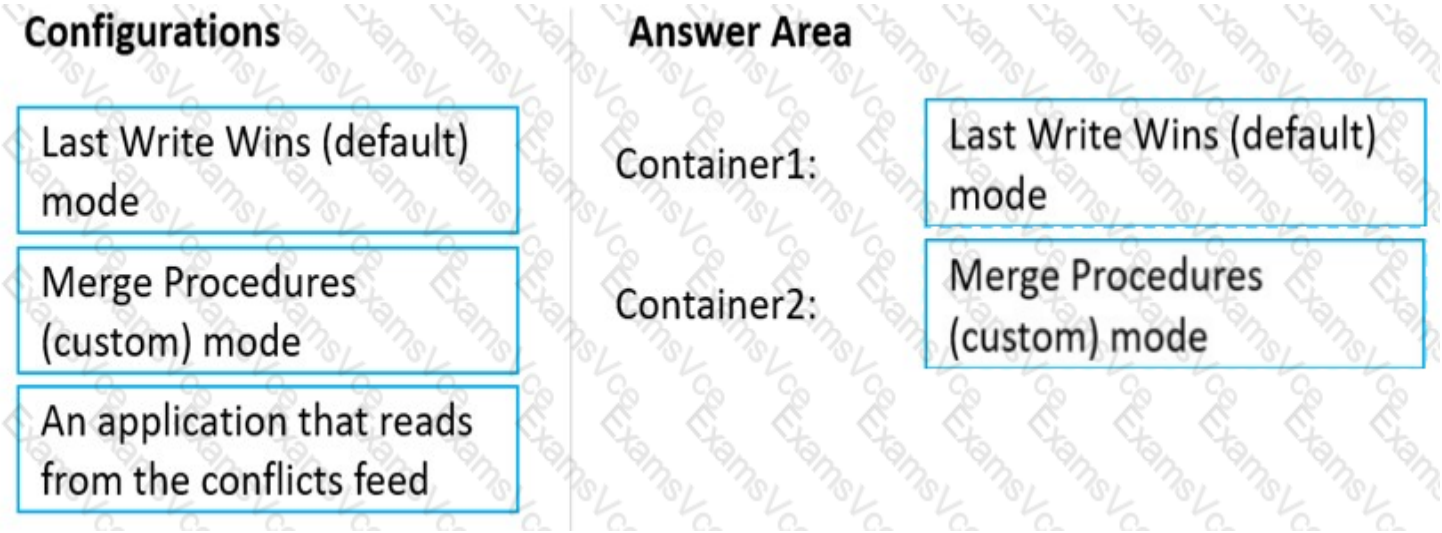
What should you configure for each container? To answer, drag the appropriate configurations to the correct containers. Each configuration may be used once, more than once, or not at all. You may need to drag the split bar between panes or scroll to view content.

NOTE: Each correct selection is worth one point.



Answer:

Explanation



Box 1: Last Write Wins (LWW) (default) mode

Last Write Wins (LWW): This resolution policy, by default, uses a system-defined timestamp property. It's based on the time-synchronization clock protocol.

Box 2: Merge Procedures (custom) mode

Custom: This resolution policy is designed for application-defined semantics for reconciliation of conflicts. When you set this policy on your Azure Cosmos container, you also need to register a merge stored procedure. This procedure is automatically invoked when conflicts are detected under a database transaction on the server. The system provides exactly once guarantee for the execution of a merge procedure as part of the commitment protocol.

Reference:

https://docs.microsoft.com/en-us/azure/cosmos-db/conflict-resolution-policies

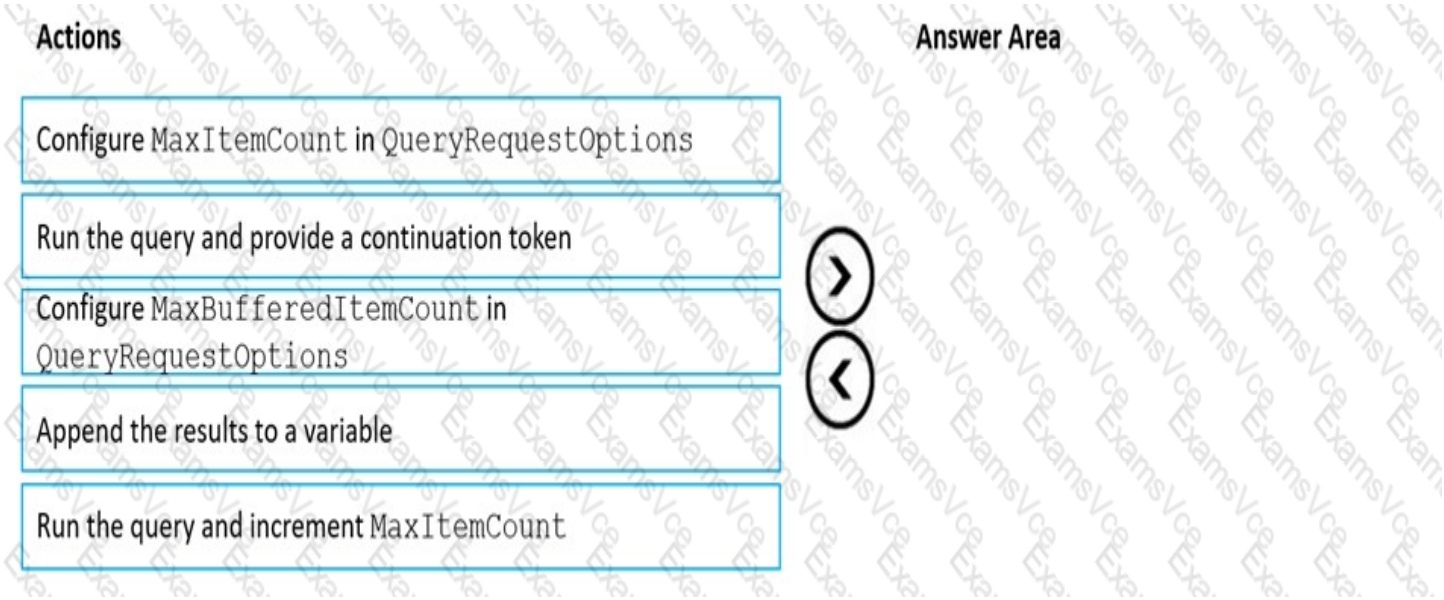
https://docs.microsoft.com/en-us/azure/cosmos-db/sql/how-to-manage-conflicts

Question #:26 - (Exam Topic 2)

You have an app that stores data in an Azure Cosmos DB Core (SQL) API account The app performs queries that return large result sets.

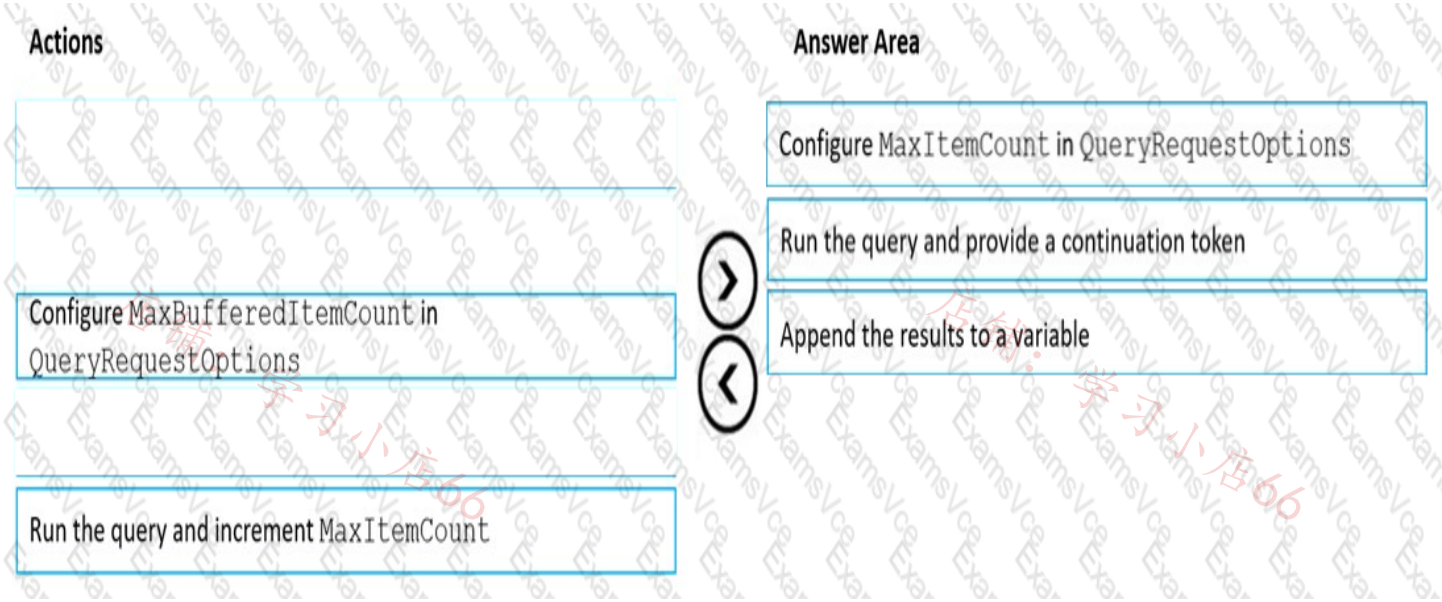
You need to return a complete result set to the app by using pagination. Each page of results must return 80 items.

Which three 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.



Answer:

Explanation



Step 1: Configure the MaxItemCount in QueryRequestOptions

You can specify the maximum number of items returned by a query by setting the MaxItemCount. The MaxItemCount is specified per request and tells the query engine to return that number of items or fewer.

Box 2: Run the query and provide a continuation token

In the .NET SDK and Java SDK you can optionally use continuation tokens as a bookmark for your query's progress. Azure Cosmos DB query executions are stateless at the server side and can be resumed at any time using the continuation token.

If the query returns a continuation token, then there are additional query results.

Step 3: Append the results to a variable

Reference: https://docs.microsoft.com/en-us/azure/cosmos-db/sql/sql-query-pagination

Question #:27 - (Exam Topic 2)

You have an Azure Cosmos DB for NoSQL account that has multiple write regions.

You need to receive an alert when requests that target the database exceed the available request units per second (RU/s).

Which Azure Monitor signal should you use?

A. Region Removed

B. Document Quota

C. Metadata Requests

D. Data Usage

Answer: B

Explanation

Azure Monitor is a service that provides comprehensive monitoring for Azure resources, including Azure Cosmos DB. You can use Azure Monitor to collect, analyze, and alert on metrics and logs from your Azure Cosmos DB account. You can create alerts for Azure Cosmos DB using Azure Monitor based on the metrics, activity log events, or Log Analytics logs on your account**1**.

For your scenario, if you want to receive an alert when requests that target the database exceed the available request units per second (RU/s), you should use the Document Quota metric. This metric measures the percentage of RU/s consumed by your account or container. You can create an alert rule on this metric from the Azure portal by following these steps**2**:

- In the Azure portal, select the Azure Cosmos DB account you want to monitor.
- Under the Monitoring section of the sidebar, select Alerts, and then select New alert rule.
- In the Create alert rule pane, fill out the Scope section by selecting your subscription name and resource type (Azure Cosmos DB accounts).
- In the Condition section, select Add condition and choose Document Quota from the list of signals.
- In the Configure signal logic pane, specify the threshold value and operator for your alert condition. For example, you can choose Greater than or equal to 90 as the threshold value and operator to receive an alert when your RU/s consumption reaches 90% or more of your provisioned throughput.
- In the Alert rule details section, specify a name and description for your alert rule.
- In the Actions section, select Add action group and choose how you want to receive notifications for your alert. For example, you can choose Email/SMS/Push/Voice as an action type and enter your email address or phone number as a receiver.
- Review your alert rule settings and select Create alert rule to save it.

Question #:28 - (Exam Topic 2)

You plan to create an Azure Cosmos DB account that will use the NoSQL API.

You need to create a grouping strategy for items that will be stored in the account. The solution must ensure that write and read operations on the items can be performed within the same transact!

What should you use to group the items?

A. logical partitions

B. physical partitions

C. databases

D. containers

Answer: A

Question #:29 - (Exam Topic 2)

Note: This question is part of a series of questions that present the same scenario. Each question in the series contains a unique solution that might meet the stated goals. Some question sets might have more than one correct solution, while others might not have a correct solution.

After you answer a question in this section, you will NOT be able to return to it. As a result, these questions will not appear in the review screen.

You have a container named container1 in an Azure Cosmos DB Core (SQL) API account.

You need to make the contents of container1 available as reference data for an Azure Stream Analytics job.

Solution: You create an Azure function that uses Azure Cosmos DB Core (SQL) API change feed as a trigger and Azure event hub as the output.

Does this meet the goal?

A. Yes

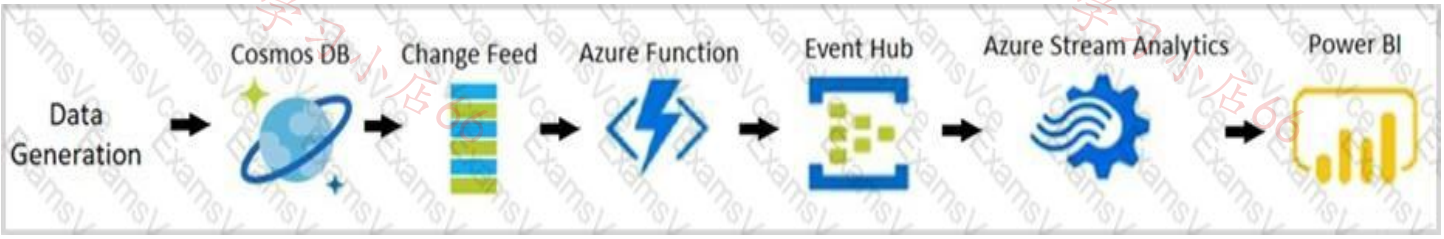
B. No

Answer: B

Explanation

The Azure Cosmos DB change feed is a mechanism to get a continuous and incremental feed of records from an Azure Cosmos container as those records are being created or modified. Change feed support works by listening to container for any changes. It then outputs the sorted list of documents that were changed in the order in which they were modified.

The following diagram represents the data flow and components involved in the solution:



Reference: https://docs.microsoft.com/en-us/azure/cosmos-db/sql/changefeed-ecommerce-solution

Question #:30 - (Exam Topic 2)

You have an Azure Cosmos DB for NoSQL account.

The change feed is enabled on a container named invoice.

You create an Azure function that has a trigger on the change feed.

What is received by the Azure function?

A. all the properties of the updated items

B. only the partition key and the changed properties of the updated items

C. all the properties of the original items and the updated items

D. only the changed properties and the system-defined properties of the updated items

Answer: A

Explanation

According to the Azure Cosmos DB documentation**12**, the change feed is a persistent record of changes to a container in the order they occur. The change feed outputs the sorted list of documents that were changed in the order in which they were modified.

The Azure function that has a trigger on the change feed receives all the properties of the updated items**2**. The change feed does not include the original items or only the changed properties. The change feed also includes some system-defined properties such as _ts (the last modified timestamp) and _lsn (the logical sequence number)**3**.

Therefore, the correct answer is:

A. all the properties of the updated items

Question #:31 - (Exam Topic 2)

You have a container named container1 in an Azure Cosmos DB for NoSQL account named account1 that is set to the session default consistency level. The average size of an item in container1 is 20 KB.

You have an application named App1 that uses the Azure Cosmos DB SDK and performs a point read on the same set of items in container1 every minute.

You need to minimize the consumption of the request units (RUs) associated to the reads by App1. What should you do?

A. In account1, change the default consistency level to bounded staleness.

B. In App1, change the consistency level of read requests to consistent prefix.

C. In account1, provision a dedicated gateway and integrated cache

D. In App1, modify the connection policy settings.

Answer: B

Explanation

The cost of a point read for a 1 KB item is 1 RU. The cost of other operations depends on factors such as item size, indexing policy, consistency level, and query complexity**1**. To minimize the consumption of RUs, you can optimize these factors according to your application needs.

For your scenario, one possible way to minimize the consumption of RUs associated to the reads by App1 is to change the consistency level of read requests to consistent prefix. Consistent prefix is a lower consistency level than session, which is the default consistency level for Azure Cosmos DB. Lower consistency levels consume fewer RUs than higher consistency levels**2**. Consistent prefix guarantees that reads never see out-of-order writes and that monotonic reads are preserved**1**. This may be suitable for your application if you can tolerate some eventual consistency.

Question #:32 - (Exam Topic 2)

You have a container in an Azure Cosmos DB for NoSQL account that stores data about orders. The following is a sample of an order document.

```
{
  "orderId" : "d4a9179b-5ead-43a3-b851-add9a71ac4b6",
  "customerId" : "f6e39103-bdc7-4346-9cfb-45daa4b2becf",
  "orderDate" : "2022-09-29",
  "orderItems" : [...],
  "total" : 12345
}
```

Documents are up to 2 KB.

You plan to receive one million orders daily.

Customers will frequently view then past order history.

You are the evaluating whether to use orderDate as the partition key.

What are two effects of using orderDate as the partition key? Each correct answer presents a complete solution.

NOTE: Each correct selection is worth one point.

A. You will exceed the maximum number of partition key values.

B. Queries will run cross-partition.

C. You will exceed the maximum storage per partition.

D. There will always be a hot partition.

Answer: B D

Question #:33 - (Exam Topic 2)

Note: This question is part of a series of questions that present the same scenario. Each question in the series contains a unique solution that might meet the stated goals. Some question sets might have more than one correct solution, while others might not have a correct solution.

After you answer a question in this section, you will NOT be able to return to it. As a result1 these questions will not appear in the review screen.

You have a database in an Azure Cosmos DB for NoSQL account that is configured for multi-region writes.

You need to use the Azure Cosmos OB SDK to implement the conflict resolution policy for a container. The solution must ensure that any conflicts are sent to the conflicts feed.

Solution: You set ConflictResolutionMode to Laswriterwins and you use the default settings for the policy.

Does this meet the goal?

A. Yes

B. No

Answer: A

Question #:34 - (Exam Topic 2)

You have a container named container1 in an Azure Cosmos DB Core (SQL) API account. The container1 container has 120 GB of data.

The following is a sample of a document in container1.

```
{
   "customerId": "5425",
   "orderId" : "9d7816e6-f401-42ba-ad05-0e03de35c0b8",
   "orderDate" : "2019-05-03",
   "orderDetails" : []
}
```

The orderId property is used as the partition key.

For each of the following statements, select Yes if the statement is true. Otherwise, select No.

NOTE: Each correct selection is worth one point.

Statements	Yes	No

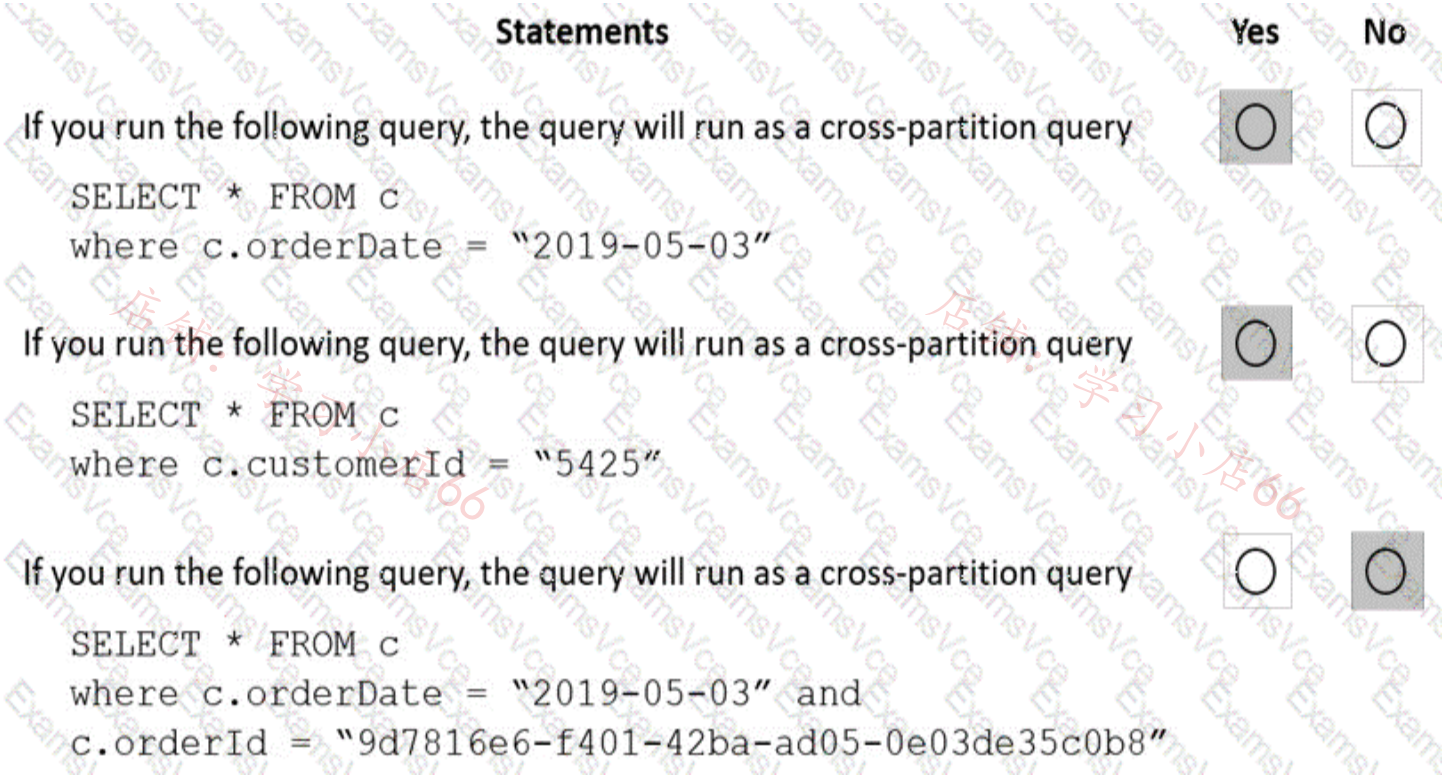
If you run the following query, the query will run as a cross-partition query
`SELECT * FROM c`
`where c.orderDate = "2019-05-03"`
	○	○

If you run the following query, the query will run as a cross-partition query
`SELECT * FROM c`
`where c.customerId = "5425"`
	○	○

If you run the following query, the query will run as a cross-partition query
`SELECT * FROM c`
`where c.orderDate = "2019-05-03" and`
`c.orderId = "9d7816e6-f401-42ba-ad05-0e03de35c0b8"`
	○	○

Answer:

Explanation



Box 1: Yes

Records with different OrderIDs will match.

Box 2: Yes

Records with different OrderIDs will match.

Box 3: No

Only records with one specific OrderId will match

Question #:35 - (Exam Topic 2)

You have an Azure Cosmos DB Core (SQL) API account that uses a custom conflict resolution policy. The account has a registered merge procedure that throws a runtime exception.

The runtime exception prevents conflicts from being resolved.

You need to use an Azure function to resolve the conflicts.

What should you use?

A. a function that pulls items from the conflicts feed and is triggered by a timer trigger

B. a function that receives items pushed from the change feed and is triggered by an Azure Cosmos DB trigger

C. a function that pulls items from the change feed and is triggered by a timer trigger

D. a function that receives items pushed from the conflicts feed and is triggered by an Azure Cosmos DB trigger

Answer: D

Explanation

The Azure Cosmos DB Trigger uses the Azure Cosmos DB Change Feed to listen for inserts and updates across partitions. The change feed publishes inserts and updates, not deletions.

Reference: https://docs.microsoft.com/en-us/azure/azure-functions/functions-bindings-cosmosdb

Question #:36 - (Exam Topic 2)

You provision Azure resources by using the following Azure Resource Manager (ARM) template.

```
{
    "$schema": "https://schema.management.azure.com/schemas/2019-04-01/deploymentTemplate.json#",
    "contentVersion": "1.0.0.0",
    "parameters": {
        "db": {
            "defaultValue": "[resourceId('Microsoft.DocumentDB/databaseAccounts', 'prod1')]",
            "type": "String"
        },
        "sms": {
            "defaultValue": "[resourceId(microsoft.insights/actionGroups','sms')]
            "type": "String"
        }
    },
    "variables": {},
    "resources": [
        {
            "type": "microsoft.insights/actionGroups",
            "apiVersion": "2019-06-01",
            "name": "sms",
            "location": "Global",
            "properties": {
                "groupShortName": "Send message",
                "enabled": true,
                "emailReceivers": [],
                "smsReceivers": [
                    {
                        "name": "Action-SMS",
                        "countryCode": "44",
                        "phoneNumber": "7111111111"
                    }
                ]
            }
        },
        {
            "type": "microsoft.insights/activityLogAlerts",
            "apiVersion": "2020-10-01",
            "name": "Alert1",
            "location": "Global",
            "dependsOn": ["sms"],
            "properties": {
                "scopes": [ "[parameters('db')]" ],
                "condition": {
                    "allOf": [
                        {
                            "field": "category",
                            "equals": "Administrative"
                        },
                        {
```

```
                    "field": "operationName",
                    "equals": "Microsoft.DocumentDB/databaseAccounts/regenerateKey/action"
                }
            ]
        },
        "actions": {
            "actionGroups": [
                {
                    "actionGroupId": "[parameters('sms')]",
                    "webhookProperties": {}
                }
            ]
        },
        "enabled": true
    }
  }
 ]
}
```

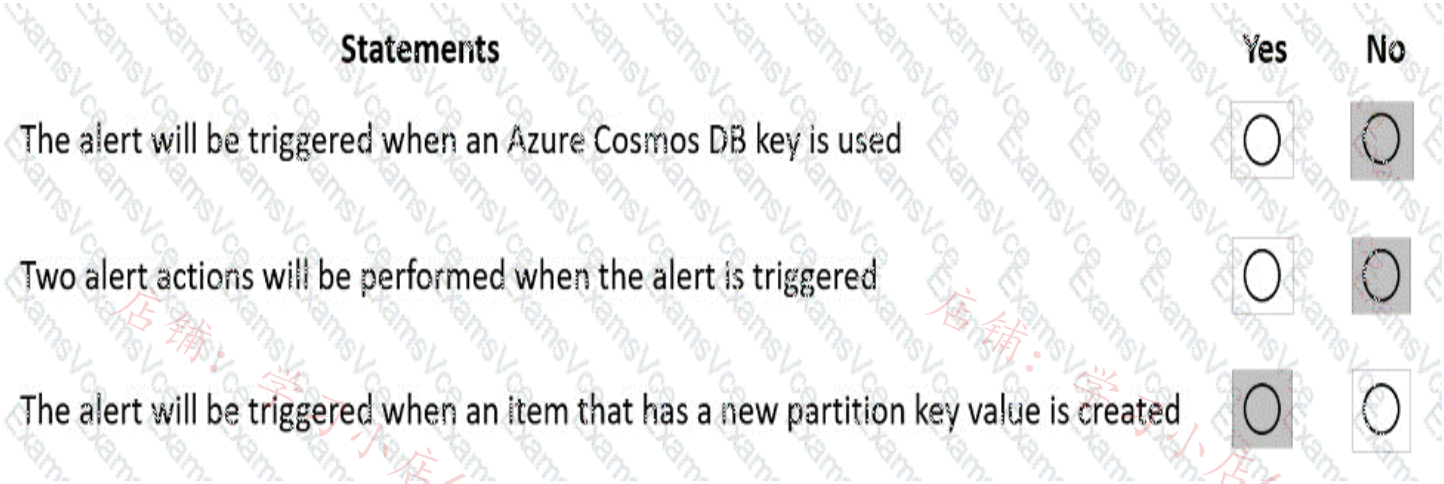
For each of the following statements, select Yes if the statement is true. Otherwise, select No.

NOTE: Each correct selection is worth one point.

Statements	Yes	No
The alert will be triggered when an Azure Cosmos DB key is used	○	○
Two alert actions will be performed when the alert is triggered	○	○
The alert will be triggered when an item that has a new partition key value is created	○	○

Answer:

Explanation



Box 1: No

An alert is triggered when the DB key is regenerated, not when it is used.

Note: The az cosmosdb keys regenerate command regenerates an access key for a Azure Cosmos DB database account.

Box 2: No

Only an SMS action will be taken.

Emailreceivers is empty so no email action is taken.

Box 3: Yes

Yes, an alert is triggered when the DB key is regenerated.

Reference: https://docs.microsoft.com/en-us/cli/azure/cosmosdb/keys

Question #:37 - (Exam Topic 2)

You are designing a data model for an Azure Cosmos DB for NoSQL account.

What are the partition limits for request units per second (RU/s) and storage? To answer, select the appropriate options in the answer area.

NOTE: Each correct selection is worth one point.

Answer:

Explanation

Question #:38 - (Exam Topic 2)

You need to create a data store for a directory of small and medium-sized businesses (SMBs). The data store must meet the following requirements:

* Store companies and the users employed by them. Each company will have less than 1,000 users.

* Some users have data that is greater than 2 KB.

* Associate each user to only one company.

* Provide the ability to browse by company.

* Provide the ability to browse the users by company.

* Whenever a company or user profile is selected, show a details page for the company and all the related users.

* Be optimized for reading data.

Which design should you implement to optimize the data store for reading data?

A. In a directory container, create a document for each company and a document for each user. Use company ID as the partition key.

B. In a company container, create a document for each company. Embed the users into company documents. Use the company ID as the partition key.

C. Create a user container that uses the user ID as the partition key and a company container that container that uses the company ID as the partition key. Add the company ID to each user documents.

D. In a user container, create a document for each user. Embed the company into each user document. Use the user ID as the partition key.

Answer: B

Explanation

Azure Cosmos DB is a multi-model database that supports various data models, such as documents, key-value, graph, and column-family**3**. The core content-model of Cosmos DB's database engine is based on atom-record-sequence (ARS), which allows it to store and query different types of data in a flexible and efficient way**3**.

To optimize the data store for reading data, you should consider the following factors:

- The size and shape of your data
- The frequency and complexity of your queries
- The latency and throughput requirements of your application
- The trade-offs between storage efficiency and query performance

Based on these factors, one possible design that you could implement is B. In a company container, create a document for each company. Embed the users into company documents. Use the company ID as the partition key.

This design has the following advantages:

- It stores companies and users as self-contained documents that can be easily retrieved by company ID**1**.
- It avoids storing redundant data or creating additional containers for users**1**.
- It allows you to browse by company and browse the users by company with simple queries**1**.
- It shows a details page for the company and all the related users by fetching a single document**1**.
- It leverages the benefits of embedding data, such as reducing the number of requests, improving query performance, and simplifying data consistency**2**.

This design also has some limitations, such as:

- It may not be suitable for some users who have data that is greater than 2 KB, as it could exceed the maximum document size limit of 2 MB**2**.
- It may not be optimal for scenarios where users need to be associated with more than one company or queried independently from companies**2**.
- It may not be scalable for companies that have more than 1,000 users, as it could result in hot partitions or throttling**2**.

Depending on your specific use case and requirements, you may need to adjust this design or choose a different one. For example, you could use a hybrid data model that combines embedding and referencing data**2**, or you could use a graph data model that expresses entities and relationships as vertices and edges.

Question #:39 - (Exam Topic 2)

You have a database in an Azure Cosmos DB for NoSQL account that is configured for multi-region writes.

You need to use the Azure Cosmos DB SDK to implement the conflict resolution policy for a container. The solution must ensure that any conflict sent to the conflict feed.

Solution: You set ConfilictResolutionMode to Custom. You Set ResolutionProcedures to a custom stored procedure. You configure the custom stored procedure to use the conflictingItems parameter to resolve conflict.

Does this meet the goal?

A. Yes

B. No

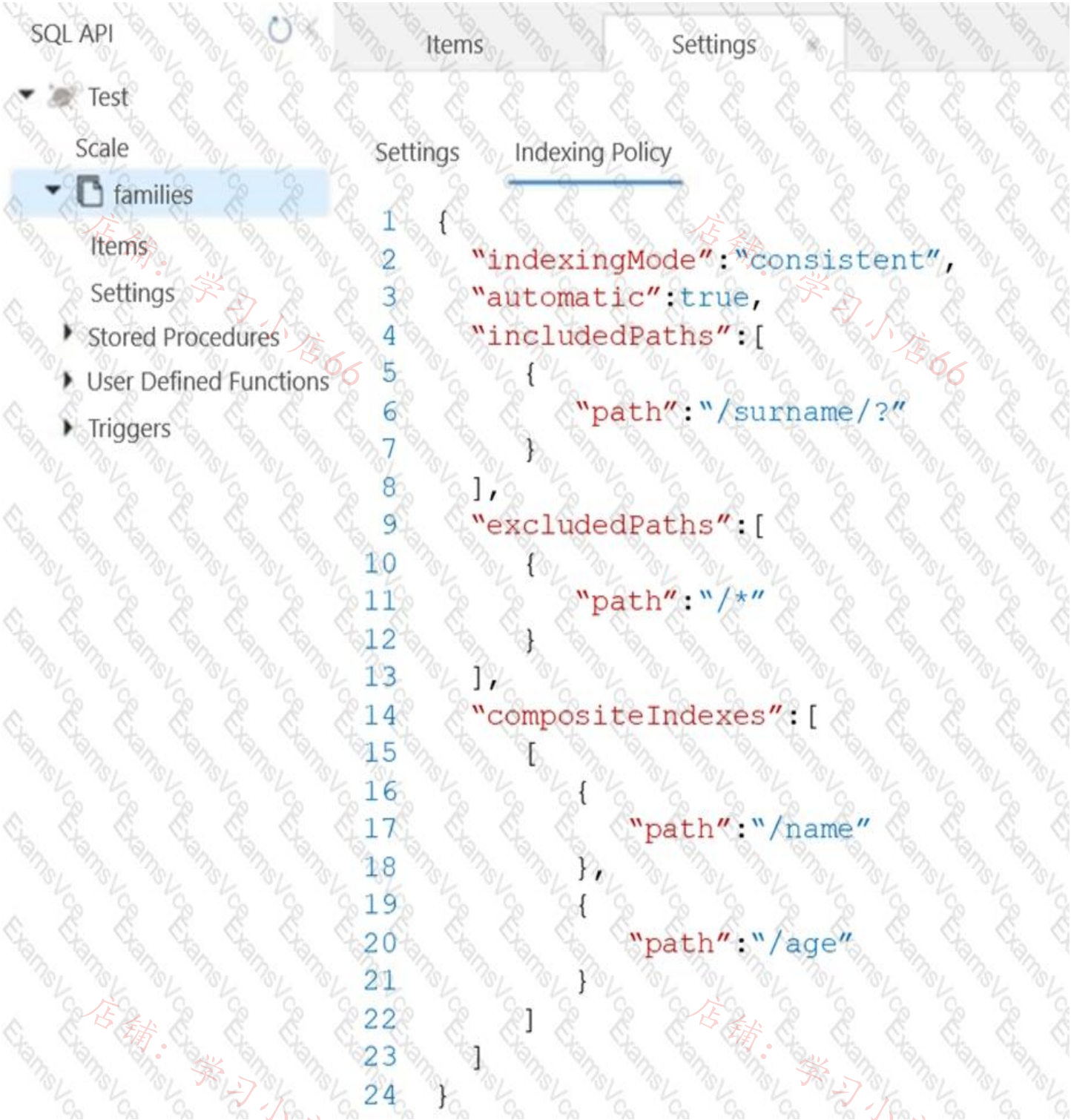
Answer: A

Explanation

Setting ConflictResolutionMode to Custom and configuring a custom stored procedure with the "conflictingItems" parameter will allow you to implement a custom conflict resolution policy. This will ensure that any conflicts are sent to the conflict feed for resolution.

Question #:40 - (Exam Topic 2)

You have the indexing policy shown in the following exhibit.



Use the drop-down menus to select the answer choice that answers each question based on the information presented in the graphic.

NOTE: Each correct selection is worth one point.

When creating a query, which ORDER BY statement will execute successfully?

- ORDER BY c.age ASC, c.name ASC
- ORDER BY c.age DESC, c.name DESC
- ORDER BY c.name ASC, c.age DESC
- ORDER BY c.name DESC, c.age ASC
- ORDER BY c.name DESC, c.age DESC

During the creation of an item, when will the index update?

- Never
- At a scheduled interval
- At the same time as the item creation
- After the item appears in the change feed

Answer:

Explanation

When creating a query, which ORDER BY statement will execute successfully?

- ORDER BY c.age ASC, c.name ASC
- ORDER BY c.age DESC, c.name DESC
- ORDER BY c.name ASC, c.age DESC
- ORDER BY c.name DESC, c.age ASC
- ORDER BY c.name DESC, c.age DESC

During the creation of an item, when will the index update?

- Never
- At a scheduled interval
- At the same time as the item creation
- After the item appears in the change feed

Box 1: ORDER BY c.name DESC, c.age DESC

Queries that have an ORDER BY clause with two or more properties require a composite index.

The following considerations are used when using composite indexes for queries with an ORDER BY clause with two or more properties:

If the composite index paths do not match the sequence of the properties in the ORDER BY clause, then the composite index can't support the query.

The order of composite index paths (ascending or descending) should also match the order in the ORDER BY clause.

The composite index also supports an ORDER BY clause with the opposite order on all paths.

Box 2: At the same time as the item creation

Azure Cosmos DB supports two indexing modes:

Consistent: The index is updated synchronously as you create, update or delete items. This means that the consistency of your read queries will be the consistency configured for the account.

None: Indexing is disabled on the container.

Reference: https://docs.microsoft.com/en-us/azure/cosmos-db/index-policy

Question #:41 - (Exam Topic 2)

You have a container named container1 in an Azure Cosmos DB Core (SQL) API account. Upserts of items in container1 occur every three seconds.

You have an Azure Functions app named function1 that is supposed to run whenever items are inserted or replaced in container1.

You discover that function1 runs, but not on every upsert.

You need to ensure that function1 processes each upsert within one second of the upsert.

Which property should you change in the Function.json file of function1?

A. checkpointInterval

B. leaseCollectionsThroughput

C. maxItemsPerInvocation

D. feedPollDelay

Answer: D

Explanation

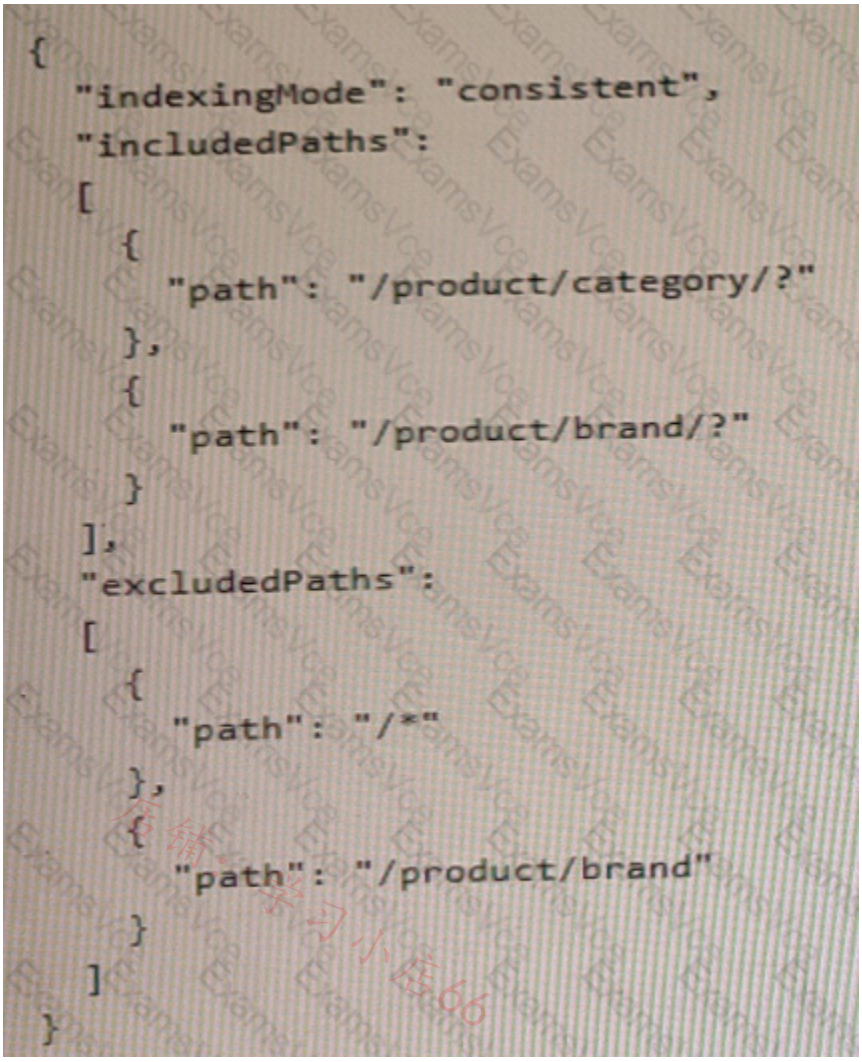
With an upsert operation we can either insert or update an existing record at the same time.

FeedPollDelay: The time (in milliseconds) for the delay between polling a partition for new changes on the feed, after all current changes are drained. Default is 5,000 milliseconds, or 5 seconds.

Reference: https://docs.microsoft.com/en-us/azure/azure-functions/functions-bindings-cosmosdb-v2-trigger

Question #:42 - (Exam Topic 2)

You have an Azure Cosmos DB database named databaset contains a container named container1. The container1 container store product data and has the following indexing policy.



Which path will be indexed?

A. /product/brand

B. /product/category

C. /product/[ ]/category

D. /product/brand/tailspin

Answer: A

Explanation

The indexing policy has an includedPaths array that contains only one path: /product/brand/? . This means that only the properties under /product/brand will be indexed. The ? symbol indicates that only scalar values will be indexed, not arrays or objects**1**.

The excludedPaths array contains a single path: /* . This means that all other properties will be excluded from indexing. The * symbol indicates a wildcard that matches any property name**1**.

Therefore, the paths /product/category , /product/[ ]/category , and /product/brand/tailspin will not be indexed.

Question #:43 - (Exam Topic 2)

You have an Azure Cosmos DB account named account1.

You have several apps that connect to account1 by using the account's secondary key.

You then configure the apps to authenticate by using service principals.

You need to ensure that account1 will only allow apps to connect by using an Azure AD identity.

Which account property should you modify?

A. disableKeyBasedMetadataWriteAccess ,

B. disableLocalAuth

C. userAssignedldentatxe

D. allowedOrxgins

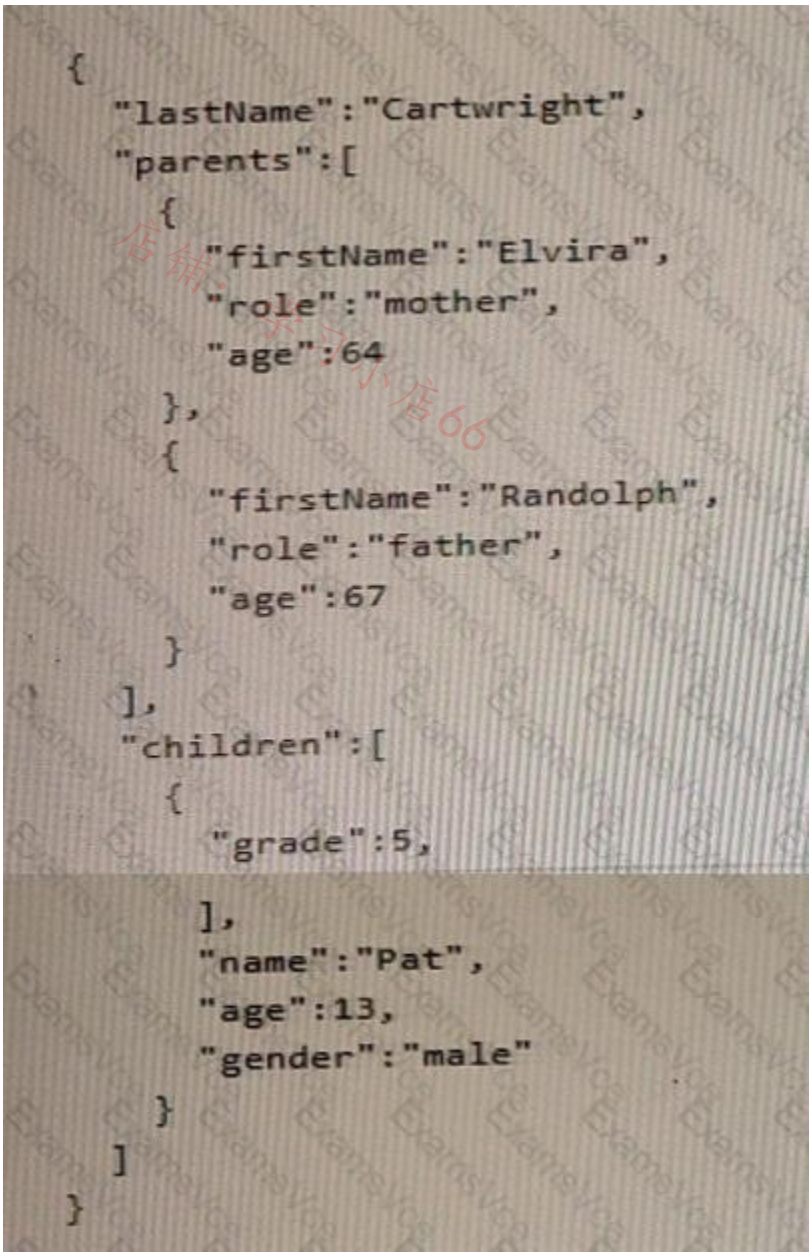
Answer: B

Explanation

The disableLocalAuth property is a boolean flag that indicates whether local authentication methods such as primary/secondary keys are disabled for the Azure Cosmos DB account. Setting this property to true improves security by ensuring that Azure Cosmos DB accounts exclusively require Azure Active Directory identities for authentication**1**.

Question #:44 - (Exam Topic 2)

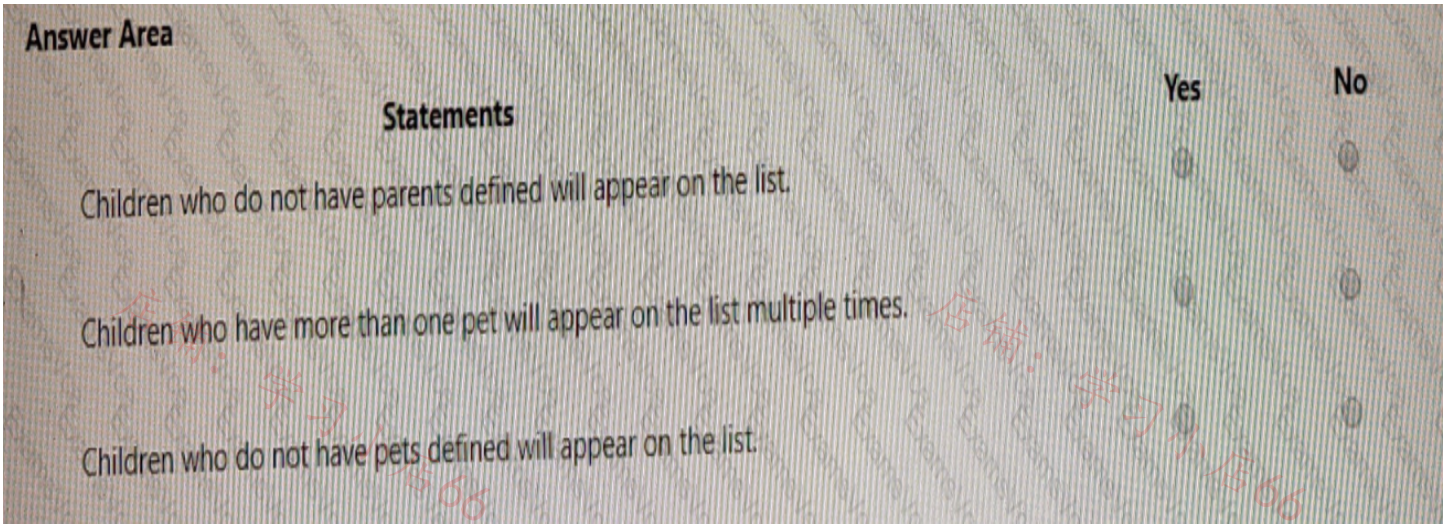
You have a container that stores data about families. The following is a sample document.



```
{
  "lastName":"Cartwright",
  "parents":[
    {
      "firstName":"Elvira",
      "role":"mother",
      "age":64
    },
    {
      "firstName":"Randolph",
      "role":"father",
      "age":67
    }
  ],
  "children":[
    {
      "grade":5,
    ],
    "name":"Pat",
    "age":13,
    "gender":"male"
  }
]
}
```

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:

Explanation

Children who do no have parents defined will appear on the list = NO

Children who do not have parents defined will not appear on the list. This is because the document schema defines the children property as an array of objects that contain the firstName and gender properties of each child, as well as a parents property that references the id values of the parents. If a child does not have parents defined, it means that the parents property is either missing or empty for that child. Therefore, such a child will not be included in the list of children who have parents defined.

Children who have more than one pet will appear on the list multiple times. = Yes

Children who have more than one pet will appear on the list multiple times. This is because the document schema defines the pets property as an array of objects that contain the givenName and type properties of each pet, as well as a children property that references the id values of the children who own the pet. If a child has more than one pet, it means that the child's id value will appear in the children property of multiple pet objects. Therefore, such a child will be included in the list of children who have pets multiple times.

Children who do no have pets defined will appear on the list = No

Children who do not have pets defined will not appear on the list. This is because the document schema defines the pets property as an array of objects that contain the givenName and type properties of each pet, as well as a children property that references the id values of the children who own the pet. If a child does not have pets defined, it means that the child's id value does not appear in the children property of any pet object. Therefore, such a child will not be included in the list of children who have pets defined.

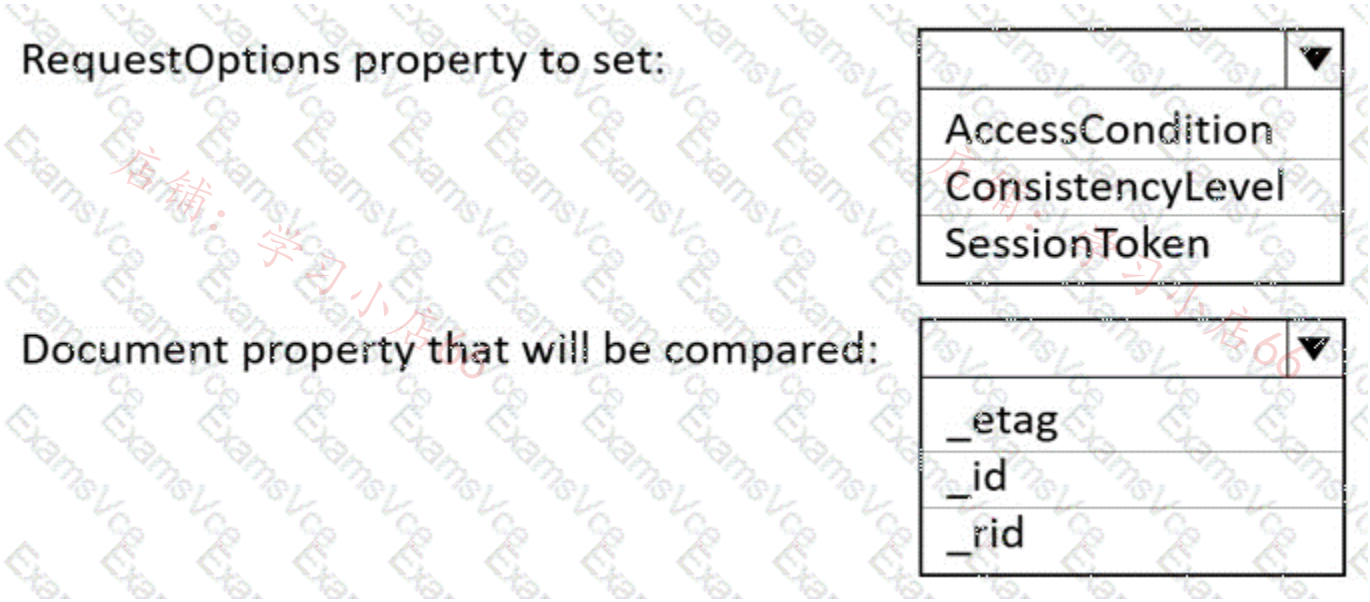
Question #:45 - (Exam Topic 2)

You have a container in an Azure Cosmos DB Core (SQL) API account.

You need to use the Azure Cosmos DB SDK to replace a document by using optimistic concurrency.

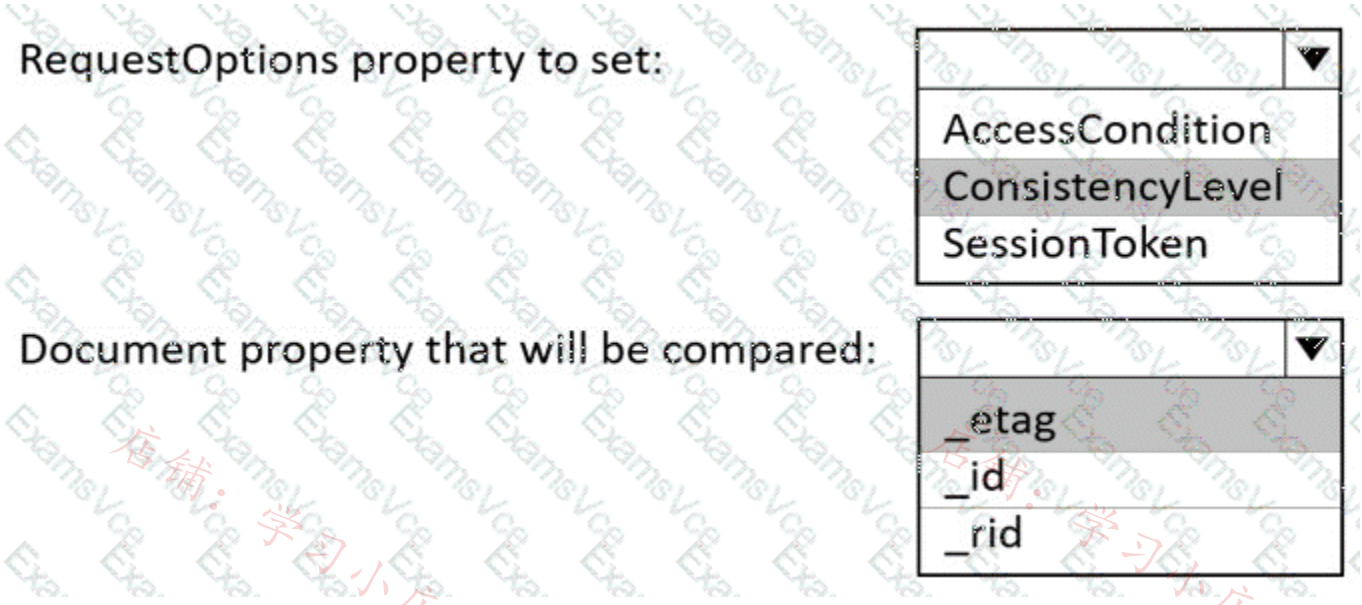
What should you include in the code? To answer, select the appropriate options in the answer area.

NOTE: Each correct selection is worth one point.



Answer:

Explanation



Box 1: ConsistencyLevel

The ItemRequestOptions Class ConsistencyLevel property gets or sets the consistency level required for the request in the Azure Cosmos DB service.

Azure Cosmos DB offers 5 different consistency levels. Strong, Bounded Staleness, Session, Consistent Prefix and Eventual - in order of strongest to weakest consistency.

Box 2: _etag

The ItemRequestOptions class helped us implement optimistic concurrency by specifying that we wanted the SDK to use the If-Match header to allow the server to decide whether a resource should be updated. The If-Match value is the ETag value to be checked against. If the ETag value matches the server ETag value, the resource is updated.

Reference:

https://docs.microsoft.com/en-us/dotnet/api/microsoft.azure.cosmos.itemrequestoptions

https://cosmosdb.github.io/labs/dotnet/labs/10-concurrency-control.html

Question #:46 - (Exam Topic 2)

You have a database named db1 in an Azure Cosmos DB f You have a third-party application that is exposed thro You need to migrate data from the application to a What should you use?

A. Database Migration Assistant

B. Azure Data Factory

C. Azure Migrate

Answer: B

Explanation

you can migrate data from various data sources to Azure Cosmos DB using different tools and methods. The choice of the migration tool depends on factors such as the data source, the Azure Cosmos DB API, the size of data, and the expected migration duration**1**. Some of the common migration tools are:

- Azure Cosmos DB Data Migration tool: This is an open source tool that can import data to Azure Cosmos DB from sources such as JSON files, MongoDB, SQL Server, CSV files, and Azure Cosmos DB collections. This tool supports the SQL API and the Table API of Azure Cosmos DB**2**.
- Azure Data Factory: This is a cloud-based data integration service that can copy data from various sources to Azure Cosmos DB using connectors. This tool supports the SQL API, MongoDB API, Cassandra API, Gremlin API, and Table API of Azure Cosmos DB**3**.
- Azure Cosmos DB live data migrator: This is a command-line tool that can migrate data from one Azure Cosmos DB container to another container within the same or different account. This tool supports live migration with minimal downtime and works with any Azure Cosmos DB API**4**.

For your scenario, if you want to migrate data from a third-party application that is exposed through an OData endpoint to a container in Azure Cosmos DB for NoSQL, you should use Azure Data Factory. Azure Data Factory has an OData connector that can read data from an OData source and write it to an Azure Cosmos DB sink using the SQL API**5**. You can create a copy activity in Azure Data Factory that specifies the OData source and the Azure Cosmos DB sink, and run it on demand or on a schedule.

Question #:47 - (Exam Topic 2)

You have an application named App1 that reads the data in an Azure Cosmos DB Core (SQL) API account. App1 runs the same read queries every minute. The default consistency level for the account is set to eventual.

You discover that every query consumes request units (RUs) instead of using the cache.

You verify the IntegratedCacheiteItemHitRate metric and the IntegratedCacheQueryHitRate metric. Both metrics have values of 0.

You verify that the dedicated gateway cluster is provisioned and used in the connection string.

You need to ensure that App1 uses the Azure Cosmos DB integrated cache.

What should you configure?

A. the indexing policy of the Azure Cosmos DB container

B. the consistency level of the requests from App1

C. the connectivity mode of the App1 CosmosClient

D. the default consistency level of the Azure Cosmos DB account

Answer: C

Explanation

Because the integrated cache is specific to your Azure Cosmos DB account and requires significant CPU and memory, it requires a dedicated gateway node. Connect to Azure Cosmos DB using gateway mode.

Reference: https://docs.microsoft.com/en-us/azure/cosmos-db/integrated-cache-faq

Question #:48 - (Exam Topic 2)

You have an Azure Cosmos DB Core (SQL) API account named storage1 that uses provisioned throughput capacity mode.

The storage1 account contains the databases shown in the following table.

Name	Throughput	Max request units per second (RU/s)	Geo-redundancy	Multi-region writes	Number of regions
db1	Autoscale	5,000	Disabled	Disabled	1
db2	Autoscale	8,000	Enabled	Enabled	3

The databases contain the containers shown in the following table.

Name	Database	Throughput
cn01	db1	Container - autoscale maximum RU/s of 10,000
cn02	db1	Database
cn03	db1	Database
cn04	db1	Database
cn05	db1	Database
cn11	db2	Database
cn12	db2	Database
cn13	db2	Database
cn14	db2	Database
cn15	db2	Database
cn16	db2	Database
cn17	db2	Database
cn18	db2	Database

For each of the following statements, select Yes if the statement is true. Otherwise, select No.

NOTE: Each correct selection is worth one point.

Statements	Yes	No
At a minimum, you will be billed for 4,000 RU/s per hour for db1	○	○
The maximum throughput that can be consumed by cn11 is 400 RU/s	○	○
To db2, you can add a new container that uses database throughput	○	○

Answer:

Explanation

Statements	Yes	No
At a minimum, you will be billed for 4,000 RU/s per hour for db1	○	○
The maximum throughput that can be consumed by cn11 is 400 RU/s	○	○
To db2, you can add a new container that uses database throughput	○	○

Box 1: No

Four containers with 1000 RU/s each.

Box 2: No

Max 8000 RU/s for db2. 8 containers, so 1000 RU/s for each container.

Box 3: Yes

Max 8000 RU/s for db2. 8 containers, so 1000 RU/s for each container. Can very well add an additional container.

Reference:

https://docs.microsoft.com/en-us/azure/cosmos-db/plan-manage-costs

https://azure.microsoft.com/en-us/pricing/details/cosmos-db/

Question #:49 - (Exam Topic 2)

You have a database in an Azure Cosmos DB for NoSQL account that is configured for multi-region writes.

You need to use the Azure Cosmos DB SDK to implement the conflict resolution policy for a container. The solution must ensure that any conflict sent to the conflict feed.

Solution: You set ConfilictResolutionMode to Custom and you use the default settings for the policy.

Does this meet the goal?

A. Yes

B. No

Answer: B

Explanation

Setting ConflictResolutionMode to Custom and using the default settings for the policy will not ensure that conflicts are sent to the conflict feed. You need to define a custom stored procedure using the "conflictingItems" parameter to handle conflicts properly.

Question #:50 - (Exam Topic 2)

You plan to create an Azure Cosmos DB database named db1 that will contain two containers. One of the containers will contain blog posts, and the other will contain users. Each item in the blog post container will include:

• A single blog post

• All the comments associated to the blog post

• The names of the users who created the blog post and added the comments.

You need to design a solution to update usernames m the user container without causing data integrity issues. The solution must minimize administrative and development effort. What should you include in the solution? To answer, select the appropriate options in the answer area.

NOTE: Each correct selection is worth one point.

Answer:

Explanation

Question #:51 - (Exam Topic 2)

You are developing an application that will use an Azure Cosmos DB Core (SQL) API account as a data source.

You need to create a report that displays the top five most ordered fruits as shown in the following table.

Name	Type	Orders
apple	fruit	1,000
orange	fruit	600
banana	fruit, exotic	400
plum	fruit	300
mango	fruit, exotic	200

A collection that contains aggregated data already exists. The following is a sample document:

{

"name": "apple",

"type": ["fruit", "exotic"],

"orders": 10000

}

Which two queries can you use to retrieve data for the report? Each correct answer presents a complete solution.

NOTE: Each correct selection is worth one point.

A)

```
SELECT TOP  i.name, i.types, i.orders
FROM items i
WHERE EXISTS(SELECT VALUE t FROM t IN i.types WHERE t.name = 'fruit')
ORDER BY i.orders,i.types
```

B)

```
SELECT TOP  i.name, i.types, i.orders
FROM items i
WHERE EXISTS(SELECT VALUE t FROM t IN i.types WHERE t.name = 'fruit')
ORDER BY i.orders DESC
```

C)

```
SELECT TOP  i.name, i.types, i.orders
FROM items i
WHERE EXISTS(SELECT VALUE t FROM t IN i.types WHERE t.name = 'fruit')
ORDER BY i.types DESC
```

D)

```
SELECT TOP  i.name, i.types, i.orders
FROM items i
WHERE ARRAY_CONTAINS(i.types, {name: 'fruit'})
ORDER BY i.orders DESC
```

A. Option A

B. Option B

C. Option C

D. Option D

Answer: B D

Explanation

ARRAY_CONTAINS returns a Boolean indicating whether the array contains the specified value. You can check for a partial or full match of an object by using a boolean expression within the command.

Reference: https://docs.microsoft.com/en-us/azure/cosmos-db/sql/sql-query-array-contains

Question #:52 - (Exam Topic 2)

You have a container named container! in an Azure Cosmos DB for NoSQL account.

You need to provide a user named User1 with the ability to insert items into container1 by using role-based access The solution must use the principle of least privilege.

Which roles should you assign to User1?

A. Cosmos DB Built-in Data Contributor only

B. Cosmos DB Operator only

C. DocumentDB Account Contribute only

D. DocumentDB Account Contributor and Cosmos DB Built-in Data Contributor

Answer: A

Explanation

The Cosmos DB Built-in Data Contributor role provides the necessary permissions to insert items into a container in an Azure Cosmos DB for NoSQL account. This role grants the minimum required privileges for the described task, adhering to the principle of least privilege.

Question #:53 - (Exam Topic 2)

You have an Azure Cosmos DB Core (SQL) API account.

You run the following query against a container in the account.

SELECT

IS_NUMBER("1234") AS A,

IS_NUMBER(1234) AS B,

IS_NUMBER({prop: 1234}) AS C

What is the output of the query?

A. [{"A": false, "B": true, "C": false}]

B. [{"A": true, "B": false, "C": true}]

C. [{"A": true, "B": true, "C": false}]

D. [{"A": true, "B": true, "C": true}]

Answer: A

Explanation

IS_NUMBER returns a Boolean value indicating if the type of the specified expression is a number.

"1234" is a string, not a number.

Reference: https://docs.microsoft.com/en-us/azure/cosmos-db/sql/sql-query-is-number

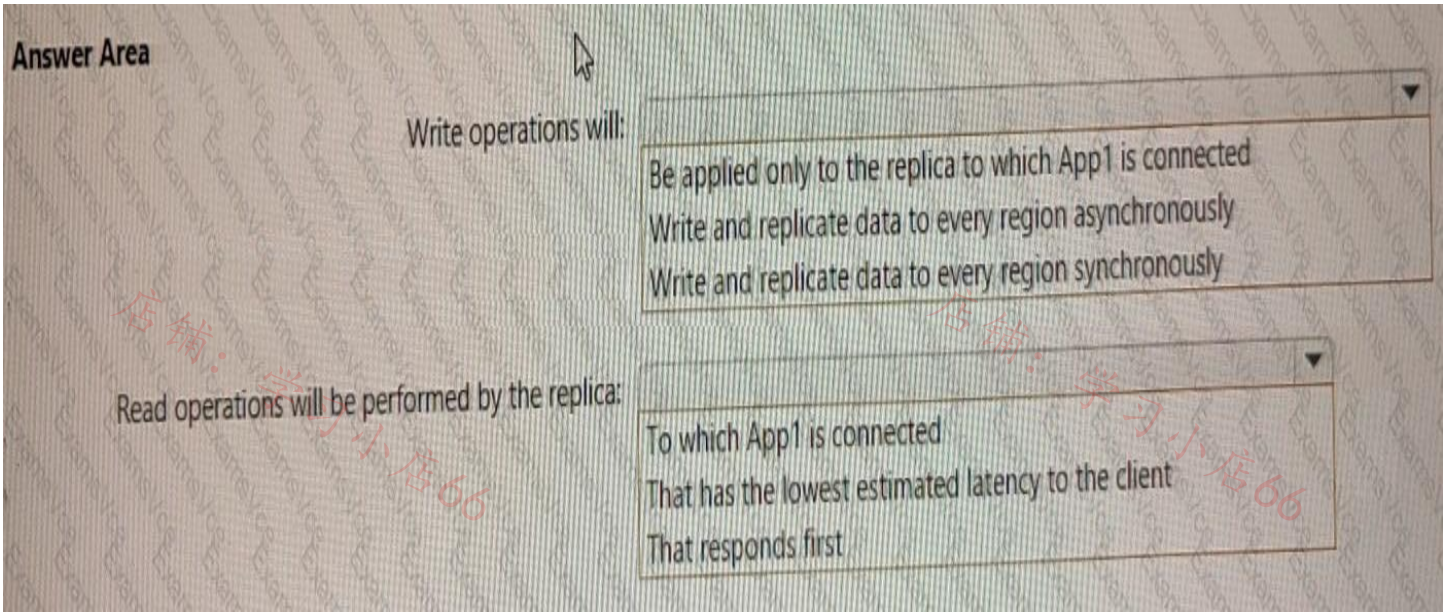
Question #:54 - (Exam Topic 2)

You have a multi-region Azure Cosmos DB account named account1 that has a default consistency level of strong.

You have an app named App1 that is configured to request a consistency level of session.

How will the read and write operations of App1 be handled? To answer, select the appropriate options in the answer area.

NOTE: Each correct selection is worth one point.



Answer:

Explanation

Box 1 = Write and replicate data to every region synchronously

This is because the write concern is mapped to the default consistency level configured on your Azure Cosmos DB account**2**, which is strong in this case. Strong consistency ensures that every write operation is synchronously committed to every region associated with your Azure Cosmos DB account**1**. The request level consistency level of session only applies to the read operations of App1**1**.

Box 2: That has the lowest estimated latency to the client

This is because the read operations of App1 will use the session consistency level that is specified in the request options. Session consistency is a client-centric consistency model that guarantees monotonic reads, monotonic writes, and read-your-own-writes within a session. A session is scoped to a client connection or a stored procedure execution. Session consistency allows clients to read from any region that has the lowest latency to the client.

Question #:55 - (Exam Topic 2)

You have an Azure Cosmos DB for NoSQL account.

You plan 10 create a container named container1. The container1 container will store items that include two properties named nm and age

The most commonly executed queries will query container1 for a specific name. The following is a sample of the query.

```
SELECT *
FROM c
WHERE c.name = 'ben smith'
ORDER BY c.name DESC, c.age ASC
```

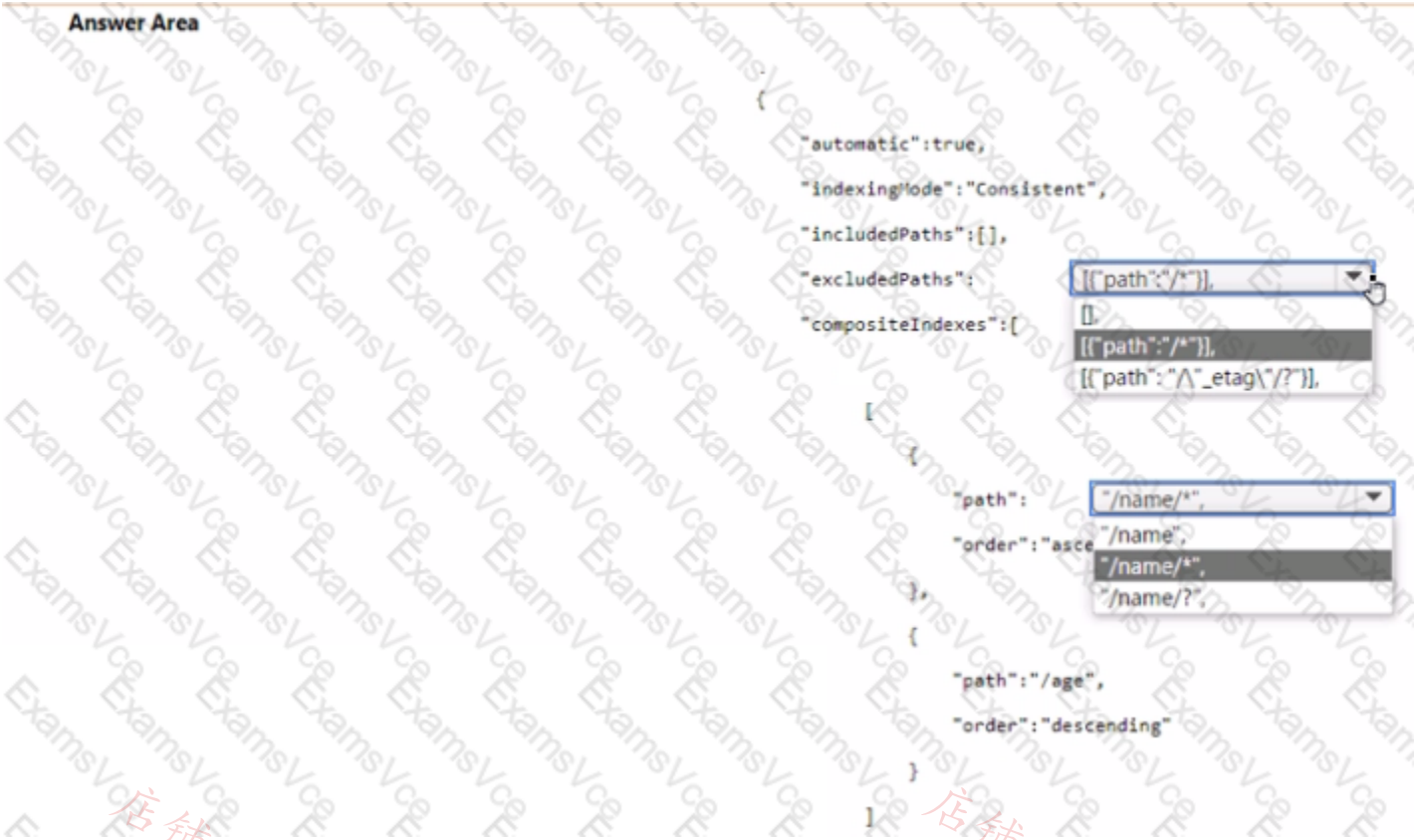
You need to define an opt-in Indexing policy for container1. The solution must meet the following requirements:

• Minimize the number of request units consumed by the queries.

• Ensure that the _etag property is excluded from indexing.

How should you define the indexing poky? To answer, select the appropriate options in the answer area.
NOTE: Each correct selection Is worth one point.

Answer Area



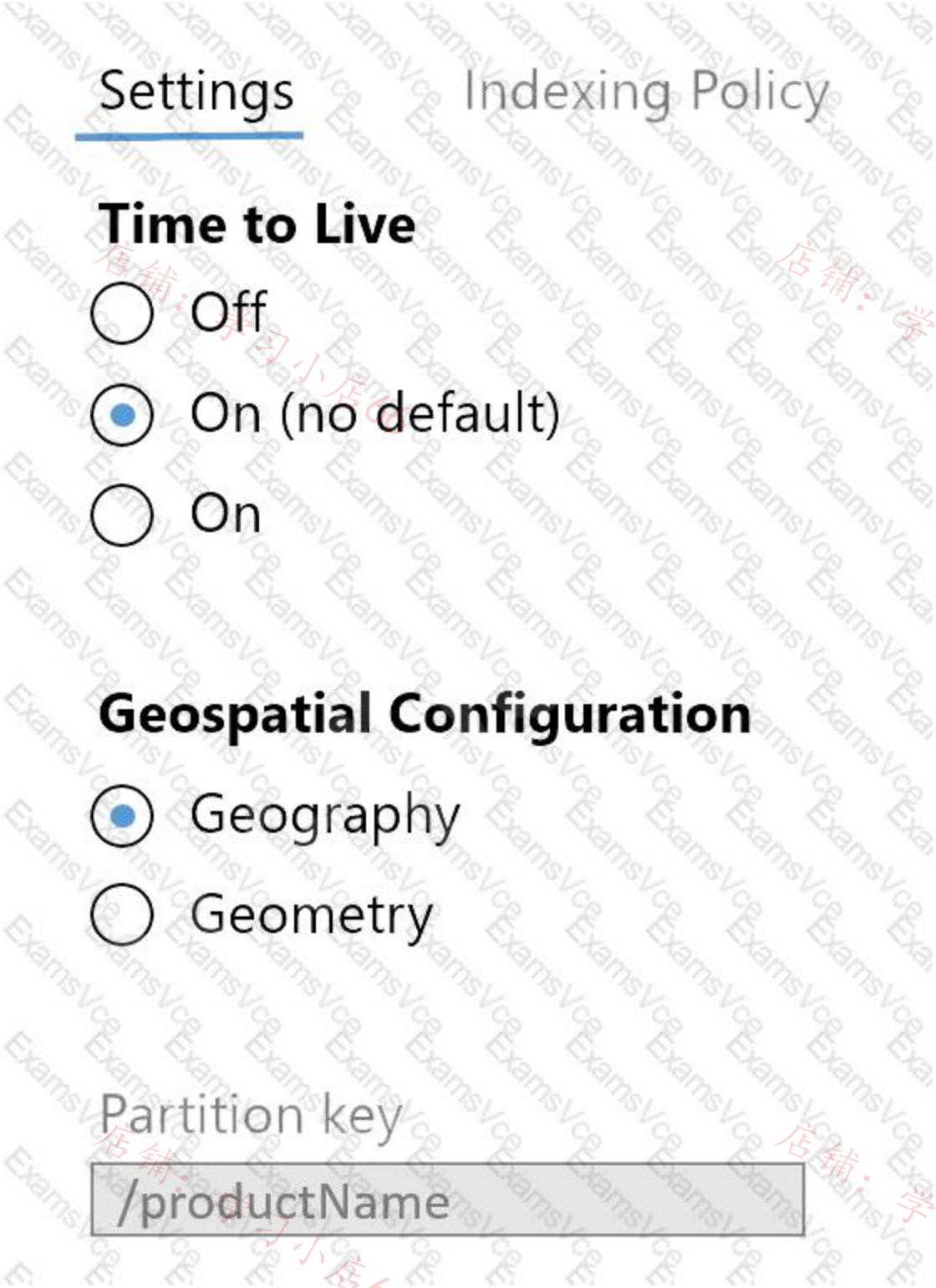
Answer:

Explanation

C:\Users\GlobeTech-Cert\Desktop\hanzala\Untitled.jpg

Question #:56 - (Exam Topic 2)

The settings for a container in an Azure Cosmos DB Core (SQL) API account are configured as shown in the following exhibit.



Which statement describes the configuration of the container?

A. All items will be deleted after one year.

B. Items stored in the collection will be retained always, regardless of the items time to live value.

C. Items stored in the collection will expire only if the item has a time to live value.

D. All items will be deleted after one hour.

Answer: C

Explanation

When DefaultTimeToLive is -1 then your Time to Live setting is On (No default)

Time to Live on a container, if present and the value is set to "-1", it is equal to infinity, and items don't expire by default.

Time to Live on an item:

This Property is applicable only if DefaultTimeToLive is present and it is not set to null for the parent container.

If present, it overrides the DefaultTimeToLive value of the parent container.

Reference: https://docs.microsoft.com/en-us/azure/cosmos-db/sql/time-to-live

Question #:57 - (Exam Topic 2)

You have a database in an Azure Cosmos DB Core (SQL) API account. The database is backed up every two hours.

You need to implement a solution that supports point-in-time restore.

What should you do first?

A. Enable Continuous Backup for the account.

B. Configure the Backup & Restore settings for the account.

C. Create a new account that has a periodic backup policy.

D. Configure the Point In Time Restore settings for the account.

Answer: A

Reference: https://docs.microsoft.com/en-us/azure/cosmos-db/provision-account-continuous-backup

Question #:58 - (Exam Topic 2)

You have an Azure Cosmos DB for NoSQL account.

You need to create an Azure Monitor query that lists recent modifications to the regional failover policy.

Which Azure Monitor table should you query?

A. CDBPartitionKeyStatistics

B. CDBQueryRunTimeStatistics

C. CDBDataPlaneRequests

D. CDBControlPlaneRequests

Answer: D

Question #:59 - (Exam Topic 2)

You have an Azure Cosmos DB Core (SQL) API account named account1.

In account1, you run the following query in a container that contains 100GB of data.

```
SELECT *
FROM c
WHERE LOWER(c.categoryid) = "hockey"
```

You view the following metrics while performing the query.

```
Retrieved Document Count                    :          45,654
Retrieved Document Size                     :     543,765,234 bytes
Output Document Count                       :              12
Output Document Size                        :             451 bytes
Index Utilization                           :            0.00 %
Total Query Execution Time                  :       2,400.34 milliseconds
  Query Preparation Times
    Query Compilation Time                  :            0.09 milliseconds
    Logical Plan Build Time                 :            0.04 milliseconds
    Physical Plan Build Time                :            0.03 milliseconds
    Query Optimization Time                 :            0.01 milliseconds
  Index Lookup Time                         :            0.00 milliseconds
  Document Load Time                        :       3,167.26 milliseconds
  Runtime Execution Times
    Query Engine Times                      :          299.16 milliseconds
    System Function Execution Time          :           79.34 milliseconds
    User-defined Function Execution Time    :            0.00 milliseconds
  Document Write Time                       :            0.01 milliseconds
Client Side Metrics
  Retry Count                               :               0
  Request Charge                            :       3,898.95 RUs
```

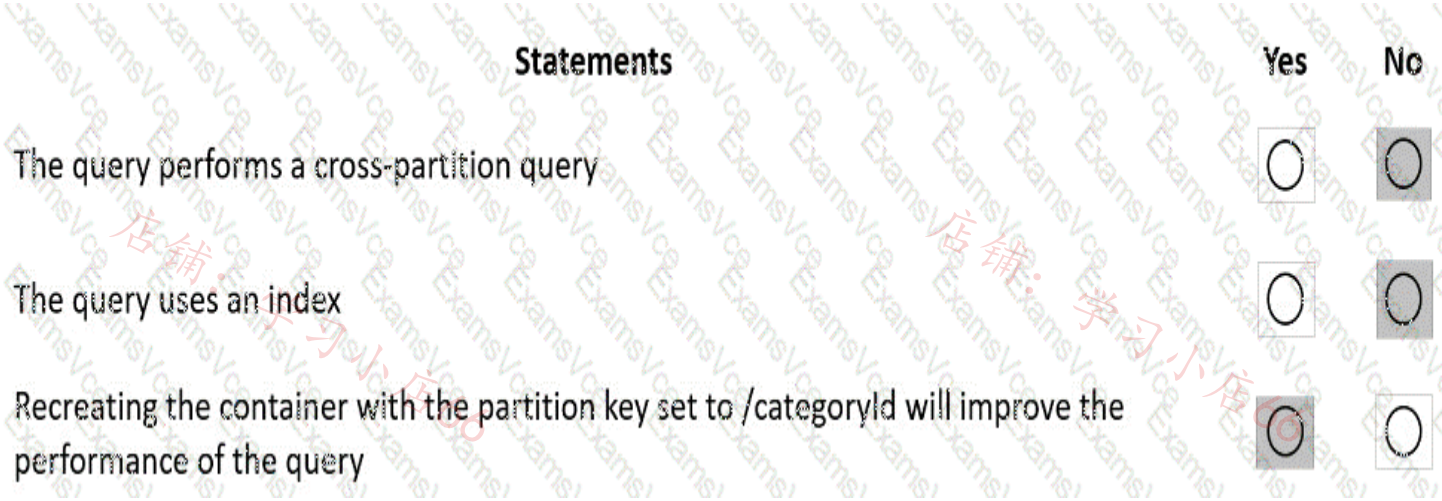
For each of the following statements, select Yes if the statement is true. Otherwise, select No.

NOTE: Each correct selection is worth one point.

Statements	Yes	No
The query performs a cross-partition query	○	○
The query uses an index	○	○
Recreating the container with the partition key set to /categoryId will improve the performance of the query	○	○

Answer:

Explanation



Box 1: No

Each physical partition should have its own index, but since no index is used, the query is not cross-partition.

Box 2: No

Index utilization is 0% and Index Look up time is also zero.

Box 3: Yes

A partition key index will be created, and the query will perform across the partitions.

Reference: https://docs.microsoft.com/en-us/azure/cosmos-db/sql/how-to-query-container

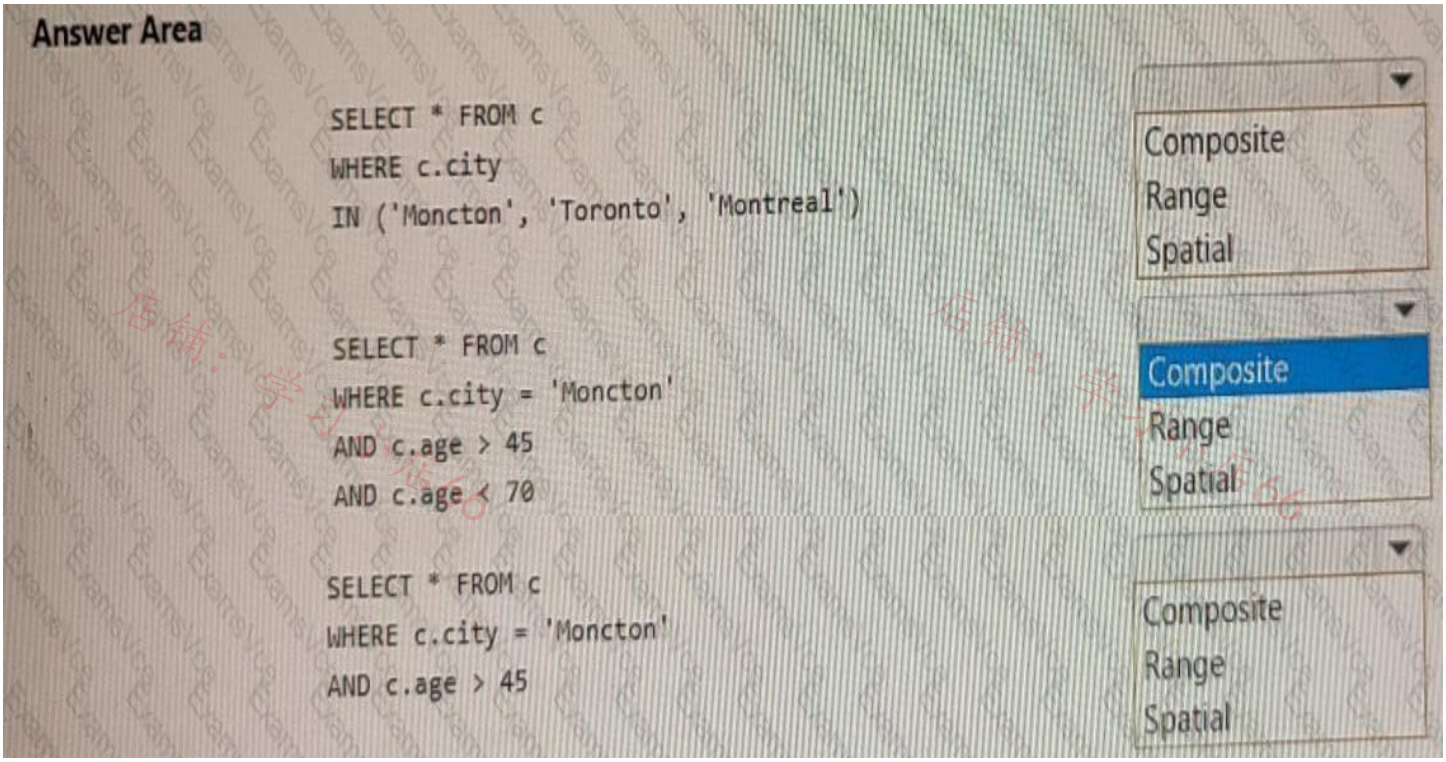
Question #:60 - (Exam Topic 2)

You have an Azure Cosmos DB for NoSQL account that frequently receives the same three queries.

You need to configure indexing to minimize RUs consumed by the queries.

Which type of index should you use for each query? To answer, select the appropriate options in the answer area.

NOTE: Each correct selection is worth one point.



Answer:

Explanation

Box 1 = Range Azure Cosmos DB supports three types of indexes: range, spatial and composite. For the query you provided, which is an equality query on a single property, the best type of index to use is range. Range index is based on an ordered tree-like structure and it is used for equality queries, range queries and checking for the presence of a property**1**. Range index also supports any string or number**2**.

Box 2 = Composite

Azure Cosmos DB supports three types of indexes: range, spatial and composite. For the query you provided, which is an order by query on two properties, the best type of index to use is composite. Composite index is used for optimizing order by queries on multiple properties**1**. Composite index allows you to specify a list of property paths and sort orders that are used for ordering items**2**.

Box 3 = spatial

Azure Cosmos DB supports three types of indexes: range, spatial and composite. For the query you provided, which is a spatial query on a point property, the best type of index to use is spatial. Spatial index is used for querying items based on their location or proximity to a given point**1**. Spatial index supports point, polygon and linestring data types**2**.

Question #:61 - (Exam Topic 2)

You have a container m an Azure Cosmos DB for NoSQL account.

Data update volumes are unpredictable.

You need to process the change teed of the container by using a web app that has multiple instances. The change feed will be processed by using the change feed processor from the Azure Cosmos DB SDK. **The** multiple instances must share the workload.

Which three actions should you perform? Each correct answer presents part of the solution.

NOTE: Each correct selection is worth one point.

A. Configure the same processor name for all the instances.

B. Configure a different processor name for each instance.

C. Configure a different lease container configuration for each instance.

D. Configure the same instance name for all the instances. 13

E. Configure a different instance name for each instance.

F. Configure the same lease container configuration for all the instances.

Answer: A E F

Question #:62 - (Exam Topic 2)

You have an Azure Cosmos DB for NoSQL account named account1 that has a single read-write region and one additional read region. Account1 uses the strong default consistency level.

You have an application that uses the eventual consistency level when submitting requests to account1.

How will writes from the application be handled?

A. Writes will use the strong consistency level.

B. Azure Cosmos DB will reject writes from the application.

C. The write order is not guaranteed during replication.

D. Writes will use the eventual consistency level.

Answer: A

Explanation

This is because the write concern is mapped to the default consistency level configured on your Azure Cosmos DB account, which is strong in this case. Strong consistency ensures that every write operation is synchronously committed to every region associated with your Azure Cosmos DB account. The eventual consistency level that the application uses only applies to the read operations. Eventual consistency offers higher availability and better performance, but it does not guarantee the order or latency of the reads.

Question #:63 - (Exam Topic 2)

You have a container named container1 in an Azure Cosmos DB Core (SQL) API account.

You need to provide a user named User1 with the ability to insert items into container1 by using role-based access control (RBAC). The solution must use the principle of least privilege.

Which roles should you assign to User1?

A. CosmosDB Operator only

B. DocumentDB Account Contributor and Cosmos DB Built-in Data Contributor

C. DocumentDB Account Contributor only

D. Cosmos DB Built-in Data Contributor only

Answer: A

Explanation

Cosmos DB Operator: Can provision Azure Cosmos accounts, databases, and containers. Cannot access any data or use Data Explorer.

Reference: https://docs.microsoft.com/en-us/azure/cosmos-db/role-based-access-control

Question #:64 - (Exam Topic 2)

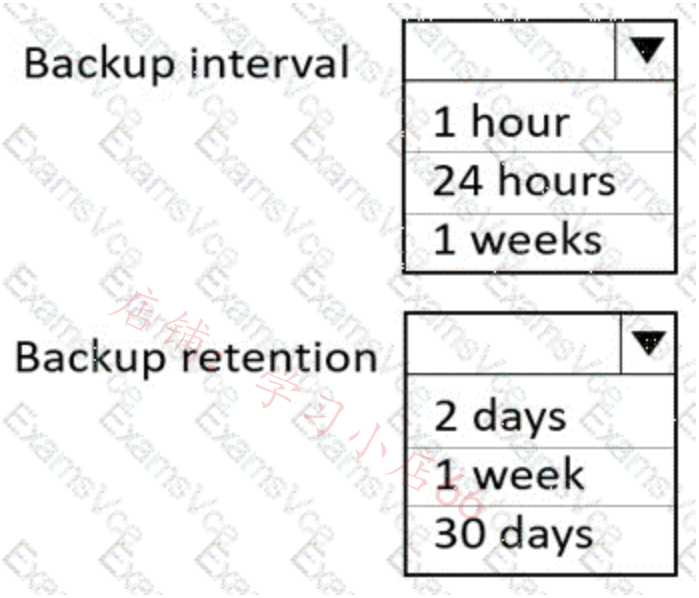
You have a database in an Azure Cosmos DB SQL API Core (SQL) account that is used for development.

The database is modified once per day in a batch process.

You need to ensure that you can restore the database if the last batch process fails. The solution must minimize costs.

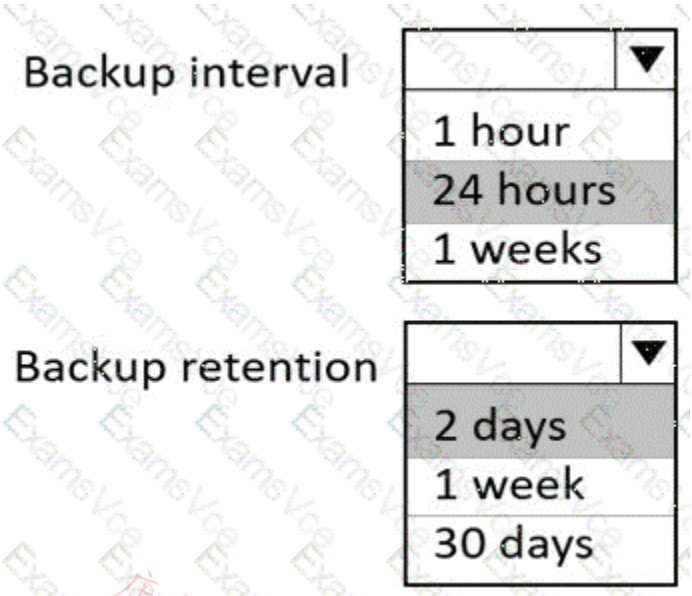
How should you configure the backup settings? To answer, select the appropriate options in the answer area.

NOTE: Each correct selection is worth one point.



Answer:

Explanation



Question #:65 - (Exam Topic 2)

You have an Azure Synapse Analytics workspace named workspace1 that contains a server less SQL pool.

You have an Azure Table Storage account that stores operational data.

You need to replace the Table storage account with Azure Cosmos DB for NoSQL The solution must meet the following requirements:

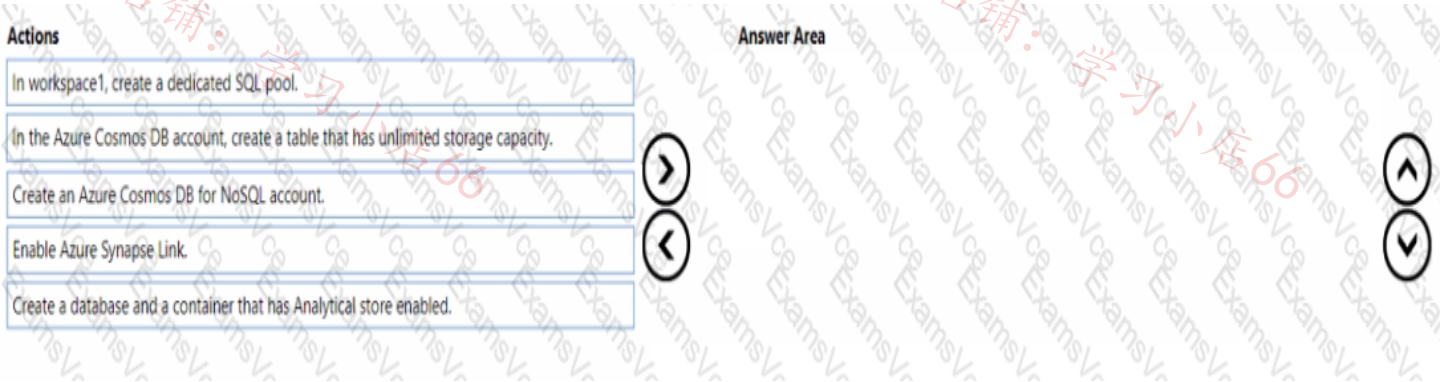
• Support Queries from the server less SQL pool.

• Only pay for analytical compute when running queries.

• Ensure that analytical processes do

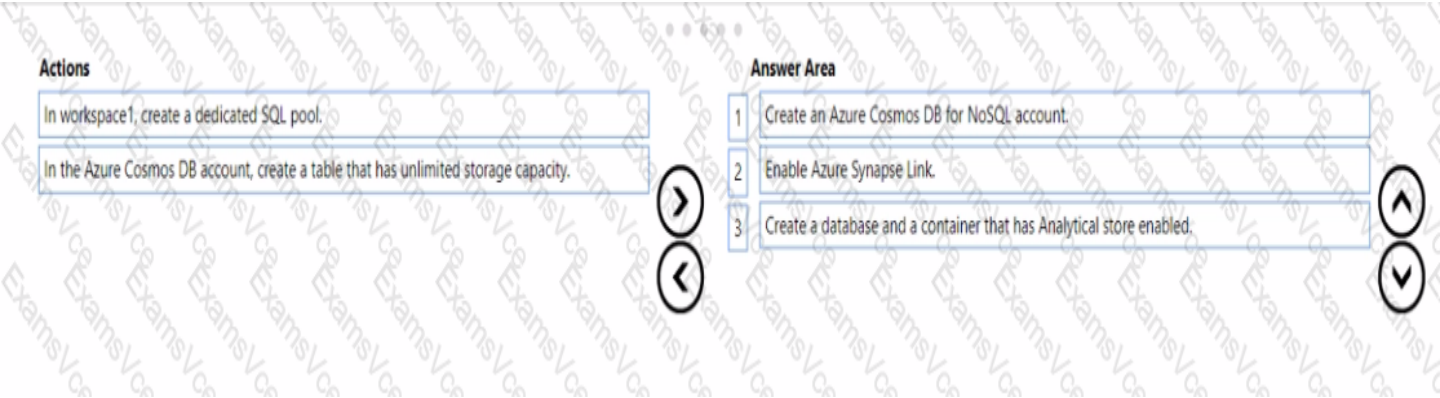
NOTE: affect operational processes.

Which three 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.



Answer:

Explanation



Question #:66 - (Exam Topic 2)

You plan to use a multi-region Azure Cosmos DB for NoSQL account to store data for a new application suite. The suite contains the applications shown in the following table.

Name	Requirement
Reporting	Must be able to track the total order counts within five minutes of orders being placed.
Purchasing	Must guarantee that the latest committed stock quantities are used always.
Fulfillment	Must ensure that orders are read in the order in which they are placed.

Each application should use the weakest consistency level possible.

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

Answer:

Explanation

Question #:67 - (Exam Topic 2)

You have a database in an Azure Cosmos DB Core (SQL) API account.

You need to create an Azure function that will access the database to retrieve records based on a variable named accountnumber. The solution must protect against SQL injection attacks.

How should you define the command statement in the function?

A. cmd = "SELECT * FROM Persons p

WHERE p.accountnumber = 'accountnumber'"

B. cmd = "SELECT * FROM Persons p

WHERE p.accountnumber = LIKE @accountnumber"

C. cmd = "SELECT * FROM Persons p

WHERE p.accountnumber = @accountnumber"

D. cmd = "SELECT * FROM Persons p

WHERE p.accountnumber = '" + accountnumber + "'"

Answer: C

Explanation

Azure Cosmos DB supports queries with parameters expressed by the familiar @ notation. Parameterized SQL provides robust handling and escaping of user input, and prevents accidental exposure of data through SQL injection.

For example, you can write a query that takes lastName and address.state as parameters, and execute it for various values of lastName and address.state based on user input.

SELECT *

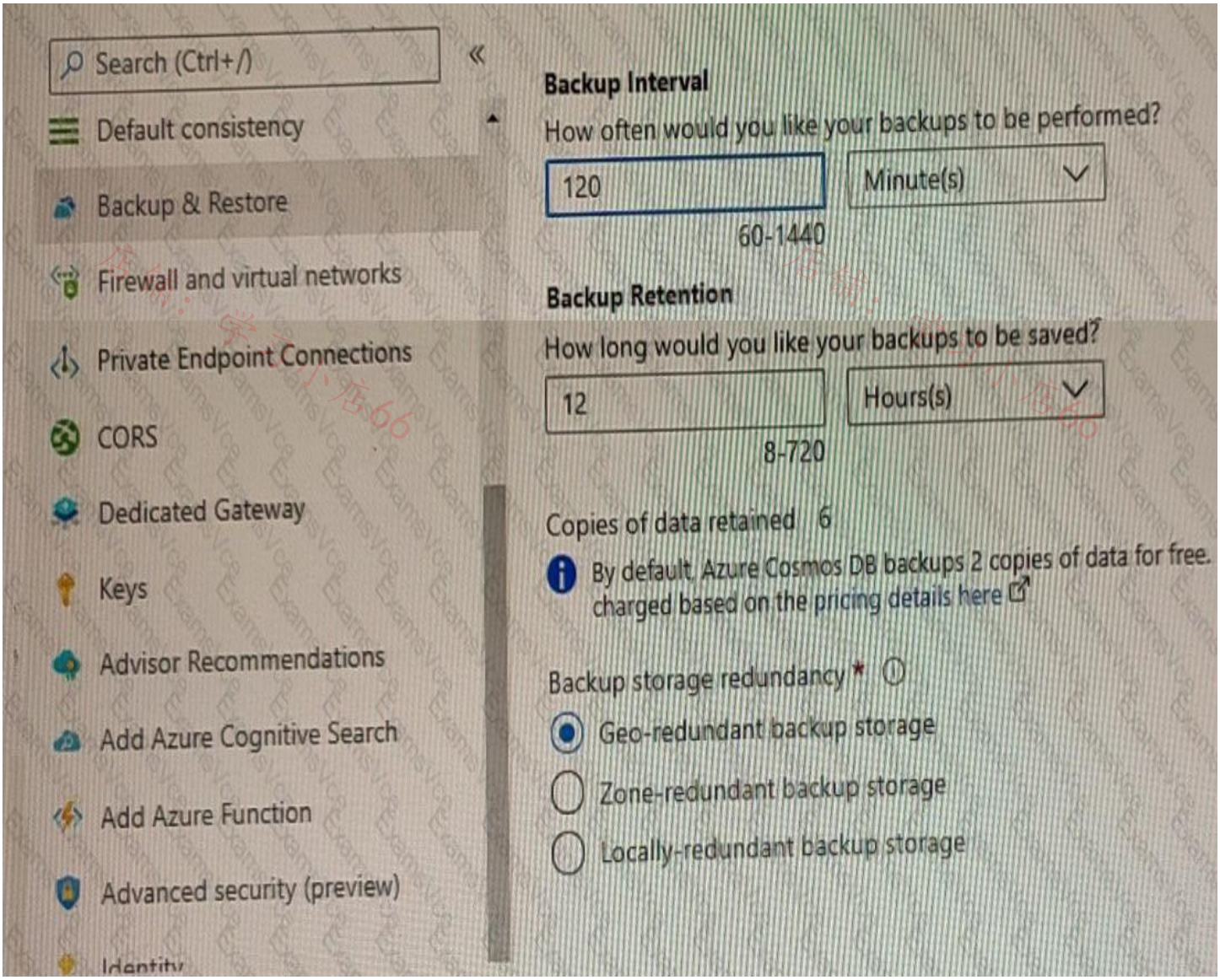
FROM Families f

WHERE f.lastName = @lastName AND f.address.state = @addressState

Reference: https://docs.microsoft.com/en-us/azure/cosmos-db/sql/sql-query-parameterized-queries

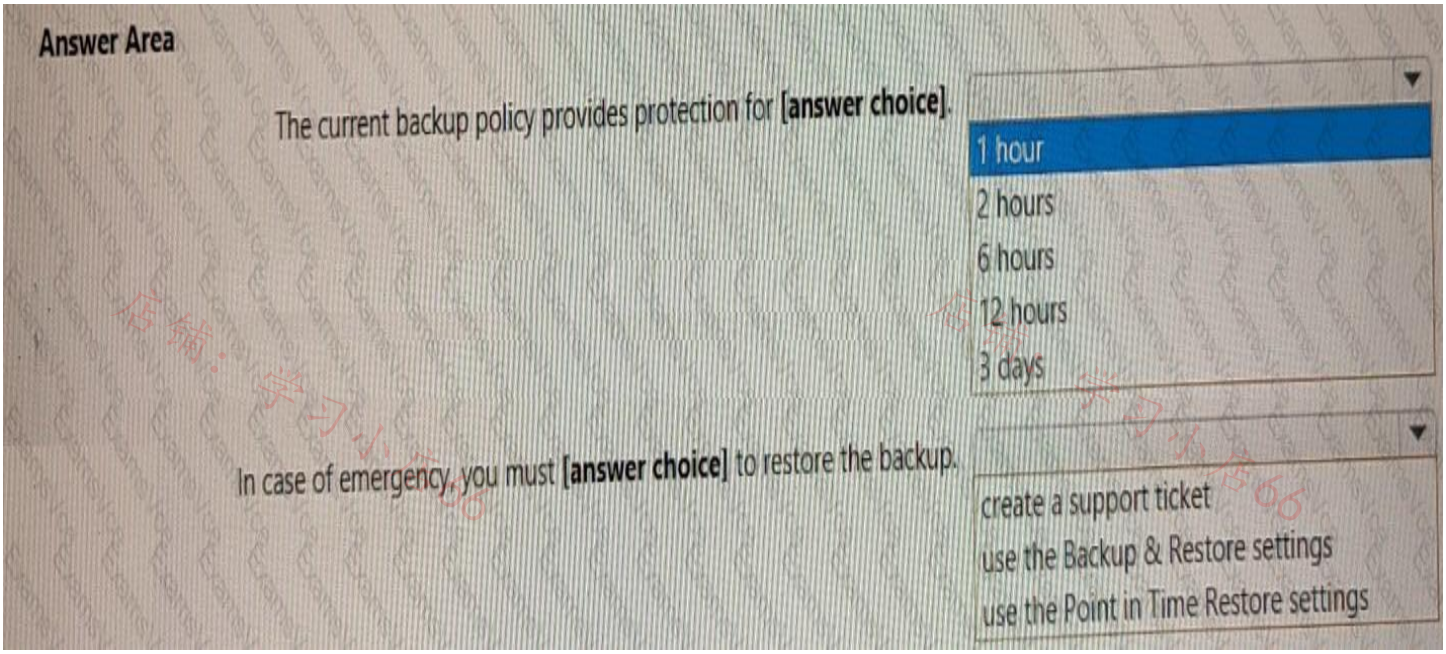
Question #:68 - (Exam Topic 2)

You configure a backup for an Azure Cosmos DB for NoSQL account as shown in the following exhibit.



Use the drop-down menus to select the answer choice that completes each statement based on the information presented in the graphic.

NOTE: Each correct selection is worth one point.



Answer:

Explanation

Box 1 = The current backup policy provides protection for: 2 Hours Azure Cosmos DB automatically takes backups of your data at regular intervals. The backup interval and the retention period can be configured from the Azure portal. You can also choose between two backup modes: periodic backup mode and continuous backup mode. Periodic backup mode is the default mode for all existing accounts and it takes a full backup of your database every 4 hours by default. Continuous backup mode is a new mode that allows you to restore to any point of time within either 7 or 30 days**1**.

For your scenario, based on the exhibit, you have configured a backup for an Azure Cosmos DB for NoSQL account using the periodic backup mode with a backup interval of 1 hour and a retention period of 2 hours. This means that Azure Cosmos DB will take a full backup of your database every hour and keep only the latest two backups. Therefore, the current backup policy provides protection for 2 hours.

Box 2: In case of emergency, you must (answer choice) to restore the backup = create a support ticket

Azure Cosmos DB automatically takes backups of your data at regular intervals. You can configure the backup interval and the retention period from the Azure portal. You can also choose between two backup modes: periodic backup mode and continuous backup mode. Periodic backup mode is the default mode for all existing accounts and it takes a full backup of your database every 4 hours by default. Continuous backup mode is a new mode that allows you to restore to any point of time within either 7 or 30 days**1**.

For your scenario, based on the exhibit, you have configured a backup for an Azure Cosmos DB for NoSQL account using the periodic backup mode with a backup interval of 1 hour and a retention period of 2 hours. This means that Azure Cosmos DB will take a full backup of your database every hour and keep only the latest two backups. In case of emergency, you must create a support ticket to restore the backup. This is the answer to your question.

To restore data from a periodic backup, you need to create a support request with Azure Cosmos DB team and provide the following information:

- The name of your Azure Cosmos DB account
- The name of the database or container that you want to restore
- The date and time (in UTC) that you want to restore from
- The name of the target Azure Cosmos DB account where you want to restore the data
- The name of the target resource group where you want to restore the data

The Azure Cosmos DB team will then initiate the restore process and notify you when it is completed**2**.

Question #:69 - (Exam Topic 2)

You have three containers in an Azure Cosmos DB Core (SQL) API account as shown in the following table.

Name	Database	Time to Live
cn1	db1	On (no default)
cn2	db1	Off
cn3	db1	On (no default)

You have the following Azure functions:

A function named Fn1 that reads the change feed of cn1

A function named Fn2 that reads the change feed of cn2

A function named Fn3 that reads the change feed of cn3

You perform the following actions:

Delete an item named item1 from cn1.

Update an item named item2 in cn2.

For an item named item3 in cn3, update the item time to live to 3,600 seconds.

For each of the following statements, select Yes if the statement is true. Otherwise, select No.

NOTE: Each correct selection is worth one point.

Statements	Yes	No
Fn1 will receive item1 from the change feed	○	○
Fn2 can check the _etag of item2 to see whether the item is an update or an insert	○	○
Fn3 will receive item3 from the change feed	○	○

Answer:

Explanation

Statements	Yes	No
Fn1 will receive item1 from the change feed	○	●
Fn2 can check the _etag of item2 to see whether the item is an update or an insert	○	●
Fn3 will receive item3 from the change feed	●	○

Box 1: No

Azure Cosmos DB's change feed is a great choice as a central data store in event sourcing architectures where all data ingestion is modeled as writes (no updates or deletes).

Note: The change feed does not capture deletes. If you delete an item from your container, it is also removed from the change feed. The most common method of handling this is adding a soft marker on the items that are being deleted. You can add a property called "deleted" and set it to "true" at the time of deletion. This document update will show up in the change feed. You can set a TTL on this item so that it can be automatically deleted later.

Box 2: No

The _etag format is internal and you should not take dependency on it, because it can change anytime.

Box 3: Yes

Change feed support in Azure Cosmos DB works by listening to an Azure Cosmos container for any changes.

Reference:

https://docs.microsoft.com/en-us/azure/cosmos-db/sql/change-feed-design-patterns

https://docs.microsoft.com/en-us/azure/cosmos-db/change-feed

Question #:70 - (Exam Topic 2)

You have a global ecommerce application that stores data in an Azure Cosmos OB for NoSQL account. The account is contoured for multi-region writes.

You need to create a stored procedure for a custom conflict resolution policy for a new container. In the event of a conflict caused by a deletion the deletion must always take priority.

Which parameter should you check m the stored procedure function?

A. conf1ictingItems

B. is Tombstone

C. existingitem

D. incoming1tem

Answer: B

Question #:71 - (Exam Topic 2)

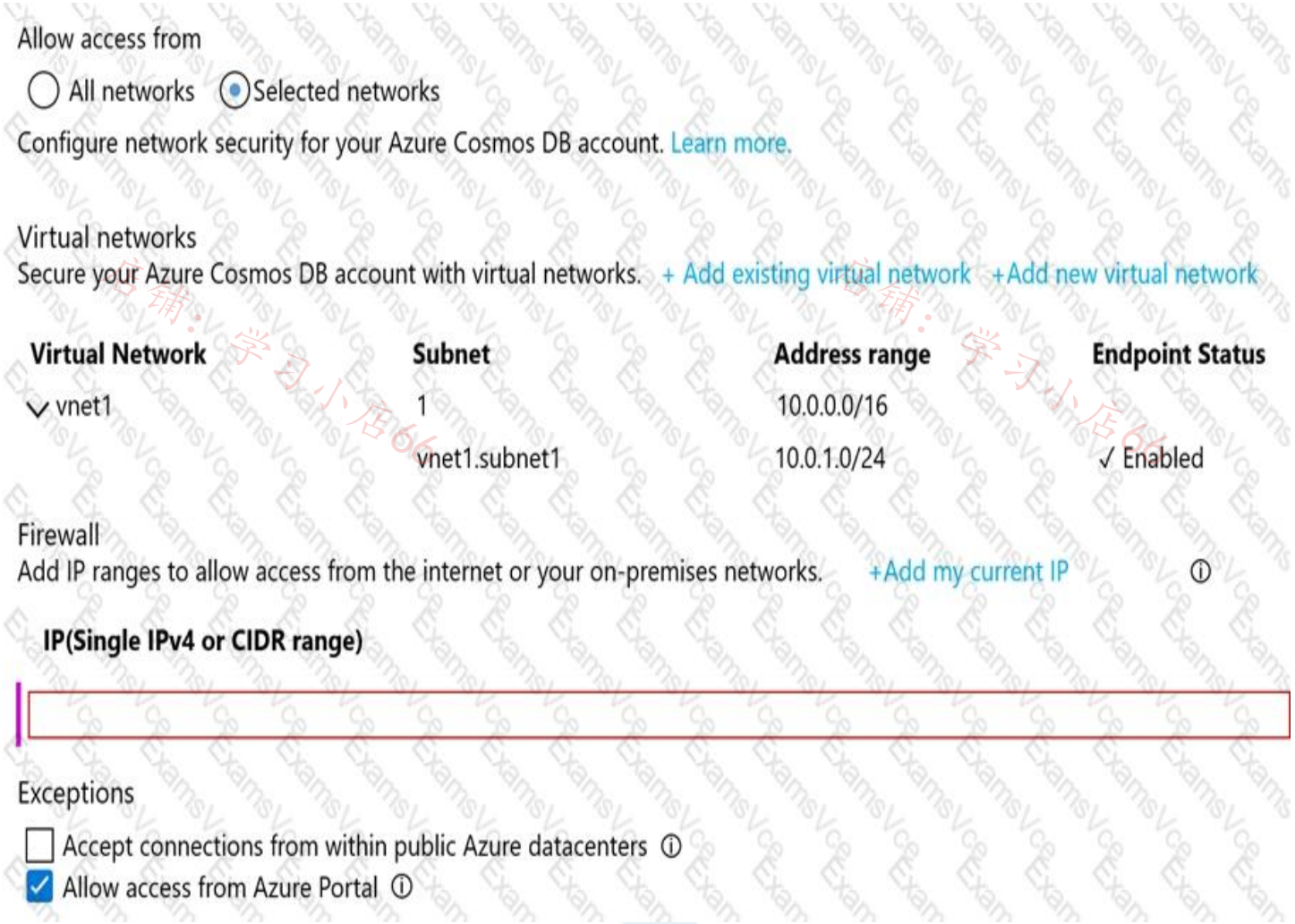
You have an Azure Cosmos DB Core (SQL) API account named account1.

You have the Azure virtual networks and subnets shown in the following table.

Subnet	Network	IP address range	Virtual machine
subnet1	vnet1	10.0.0.0/24	VM1
subnet2	vnet1	10.0.1.0/24	VM2
subnet3	vnet2	10.1.0.0/24	VM3

The vnet1 and vnet2 networks are connected by using a virtual network peer.

The Firewall and virtual network settings for account1 are configured as shown in the exhibit.



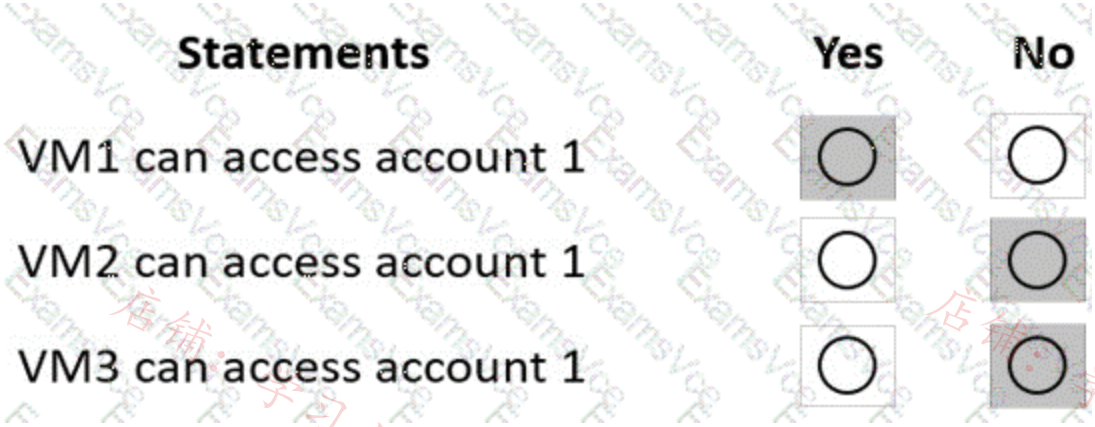
For each of the following statements, select Yes if the statement is true. Otherwise, select No.

NOTE: Each correct selection is worth one point.

Statements	Yes	No
VM1 can access account 1	○	○
VM2 can access account 1	○	○
VM3 can access account 1	○	○

Answer:

Explanation



Box 1: Yes

VM1 is on vnet1.subnet1 which has the Endpoint Status enabled.

Box 2: No

Only virtual network and their subnets added to Azure Cosmos account have access. Their peered VNets cannot access the account until the subnets within peered virtual networks are added to the account.

Box 3: No

Only virtual network and their subnets added to Azure Cosmos account have access.

Reference: https://docs.microsoft.com/en-us/azure/cosmos-db/how-to-configure-vnet-service-endpoint

Question #:72 - (Exam Topic 2)

You need to create a database in an Azure Cosmos DB for NoSQL account. The database will contain three containers named coll1, coll2 and coll3. The coll1 container will have unpredictable read and write volumes. The col!2 and coll3 containers will have predictable read and write volumes. The expected maximum throughput for coll1 and coll2 is 50,000 request units per second (RU/s) each.

How should you provision the collection while minimizing costs?

A. Create a provisioned throughput account. Set the throughput for coll1 to Manual. Set the throughput for coll2 and coll3 to Autoscale.

B. Create a provisioned throughput account. Set the throughput for call1 to Autoscale. Set the throughput for call2 and coll3 to Manual.

C. Create a serverless account.

Answer: B

Explanation

Azure Cosmos DB offers two different capacity modes: provisioned throughput and serverless**1**. Provisioned

throughput mode allows you to configure a certain amount of throughput (expressed in Request Units per second or RU/s) that is provisioned on your databases and containers. You get billed for the amount of throughput you've provisioned, regardless of how many RUs were consumed**1**. Serverless mode allows you to run your database operations without having to configure any previously provisioned capacity. You get billed for the number of RUs that were consumed by your database operations and the storage consumed by your data**1**.

To create a database that minimizes costs, you should consider the following factors:

- The read and write volumes of your containers
- The predictability and variability of your traffic
- The latency and throughput requirements of your application
- The geo-distribution and availability needs of your data

Based on these factors, one possible option that you could choose is B. Create a provisioned throughput account. Set the throughput for coll1 to Autoscale. Set the throughput for coll2 and coll3 to Manual.

This option has the following advantages:

- It allows you to handle unpredictable read and write volumes for coll1 by using Autoscale, which automatically adjusts the provisioned throughput based on the current load**1**.
- It allows you to handle predictable read and write volumes for coll2 and coll3 by using Manual, which lets you specify a fixed amount of provisioned throughput that meets your performance needs**1**.
- It allows you to optimize your costs by paying only for the throughput you need for each container**1**.
- It allows you to enable geo-distribution for your account if you need to replicate your data across multiple regions**1**.

This option also has some limitations, such as:

- It may not be suitable for scenarios where all containers have intermittent or bursty traffic that is hard to forecast or has a low average-to-peak ratio**1**.
- It may not be optimal for scenarios where all containers have low or sporadic traffic that does not justify provisioned capacity**1**.
- It may not support availability zones or multi-master replication for your account**1**.

Depending on your specific use case and requirements, you may need to choose a different option. For example, you could use a serverless account if all containers have low or sporadic traffic that does not require predictable performance or geo-distribution**1**. Alternatively, you could use a provisioned throughput account with Manual for all containers if all containers have stable and consistent traffic that requires predictable performance or geo-distribution**1**.

Question #:73 - (Exam Topic 2)

You have a database named db1in an Azure Cosmos DB for NoSQL account named account 1.

You need to write JSON data to db1 by using Azure Stream Analytics. The solution must minimize costs.

Which should you do before you can use db1 as an output of Stream Analytics?

A. in account, add a private endpoint.

B. In db1, create containers that have a custom indexing policy and analytical store disabled.

C. In account, enable a dedicated gateway.

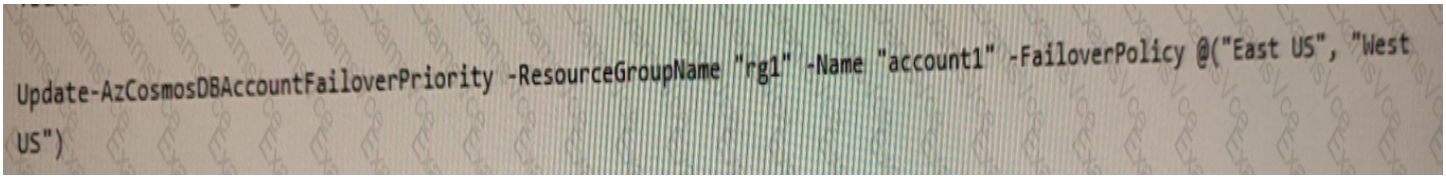
D. In db1, create containers that have an automatic indexing policy and analytical store enabled.

Answer: A

Question #:74 - (Exam Topic 2)

You have an Azure Cosmos DB for NoSQL account1 that is configured for automatic failover. The account1 account has a single read-write region in West US and a and a read region in East US.

You run the following PowerShell command.



What is the effect of running the command?

A. A manual failover will occur.

B. The account will be unavailable to writes during the change.

C. The provisioned throughput for account1 will increase.

D. The account will be configured for multi-region writes.

Answer: D

Explanation

You can use the Set-AzCosmosDBAccountRegion cmdlet to update the regions that an Azure Cosmos DB account uses. You can use this cmdlet to add a region or change the region failover order. The cmdlet requires a resource group name, an Azure Cosmos DB account name, and a list of regions in desired failover order**1**.

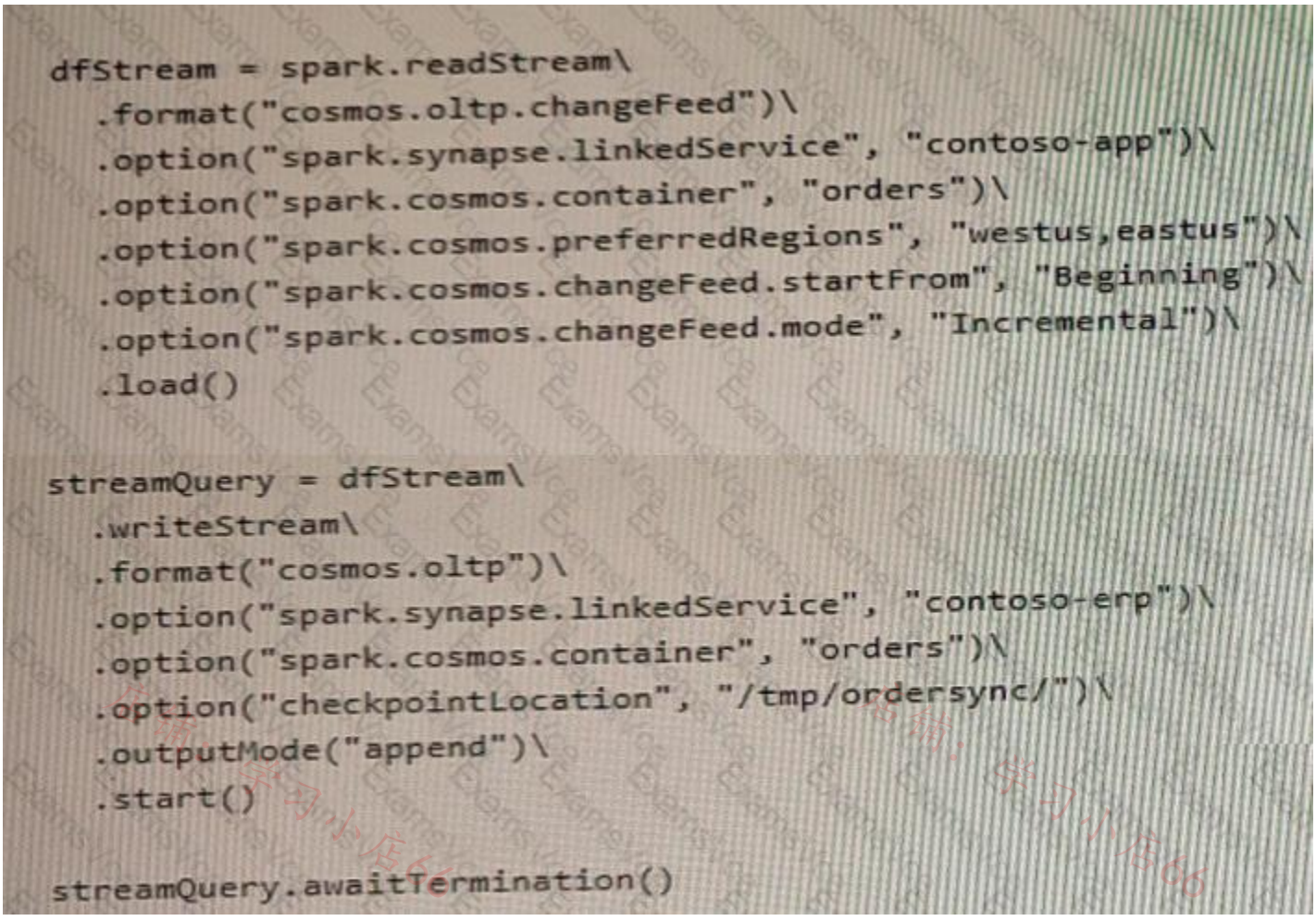
For your scenario, based on the PowerShell command, you are using the Set-AzCosmosDBAccountRegion

cmdlet to update the regions for an Azure Cosmos DB account named account1 that is configured for automatic failover. The command specifies two regions: West US and East US. The effect of running the command is that the account will be configured for multi-region writes.

Multi-region writes is a feature of Azure Cosmos DB that allows you to write data to any region in your account and have it automatically replicated to all other regions. This feature provides high availability and low latency for write operations across multiple regions. To enable multi-region writes, you need to specify at least two regions in your account and set them as write regions**2**. In your command, you are setting both West US and East US as write regions by using the -IsZoneRedundant parameter with a value of $true for both regions.

Question #:75 - (Exam Topic 2)

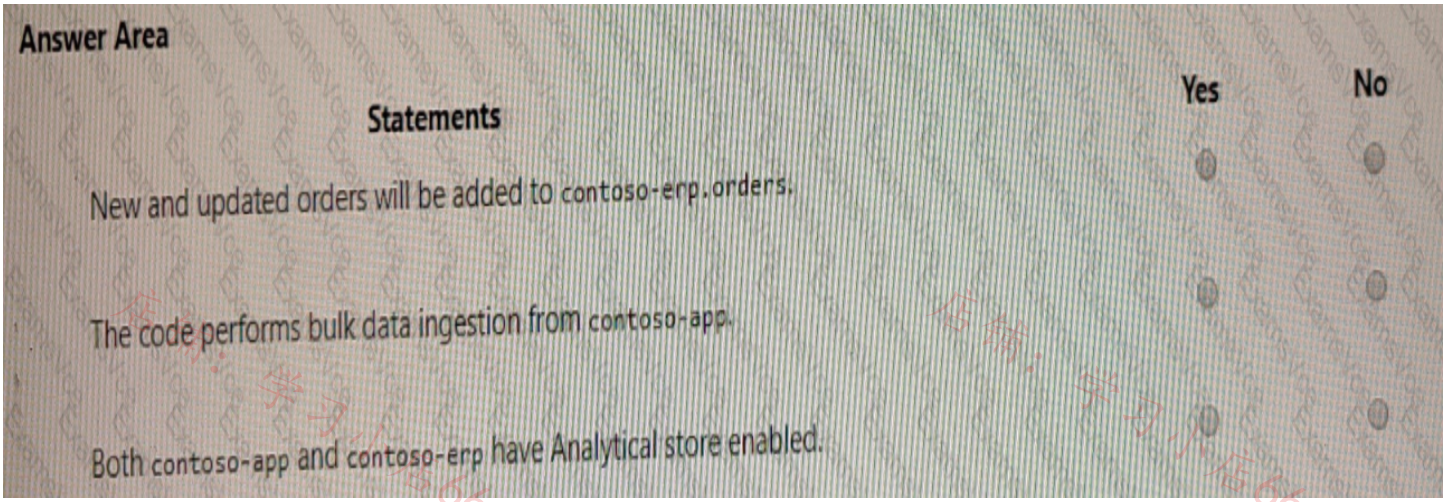
You have an Apache Spark pool in Azure Synapse Analytics that runs the following Python code in a notebook.



```
dfStream = spark.readStream\
  .format("cosmos.oltp.changeFeed")\
  .option("spark.synapse.linkedService", "contoso-app")\
  .option("spark.cosmos.container", "orders")\
  .option("spark.cosmos.preferredRegions", "westus,eastus")\
  .option("spark.cosmos.changeFeed.startFrom", "Beginning")\
  .option("spark.cosmos.changeFeed.mode", "Incremental")\
  .load()

streamQuery = dfStream\
  .writeStream\
  .format("cosmos.oltp")\
  .option("spark.synapse.linkedService", "contoso-erp")\
  .option("spark.cosmos.container", "orders")\
  .option("checkpointLocation", "/tmp/ordersync/")\
  .outputMode("append")\
  .start()

streamQuery.awaitTermination()
```

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:

Explanation

New and updated orders will be added to contoso-erp.orders: Yes

The code performs bulk data ingestion from contoso-app: No

Both contoso-app and contoso-erp have Analytics store enabled: Yes

The code uses the spark.readStream method to read data from a container named orders in a database named contoso-app. The data is then filtered by a condition and written to another container named orders in a database named contoso-erp using the spark.writeStream method. The write mode is set to “append”, which means that new and updated orders will be added to the destination container**1**.

The code does not perform bulk data ingestion from contoso-app, but rather stream processing. Bulk data ingestion is a process of loading large amounts of data into a data store in batches. Stream processing is a process of continuously processing data as it arrives in real-time**2**.

Both contoso-app and contoso-erp have Analytics store enabled, because they are both accessed by Spark pools using the spark.cosmos.oltp method. This method requires that the containers have Analytics store enabled, which is a feature that allows Spark pools to query data stored in Azure Cosmos DB containers using SQL APIs**3**.

Question #:76 - (Exam Topic 2)

You plan to deploy two Azure Cosmos DB Core (SQL) API accounts that will each contain a single database. The accounts will be configured as shown in the following table.

Name	Description
development	• Supports the development of new application features • Used intermittently as needed during development
shipments	• Captures over 100,000 updates per second generated at unpredictable rimes throughout the business day • Used with Azure Synapse Link for analytics

How should you provision the containers within each account to minimize costs? To answer, select the appropriate options in the answer area.

NOTE: Each correct selection is worth one point.

development:

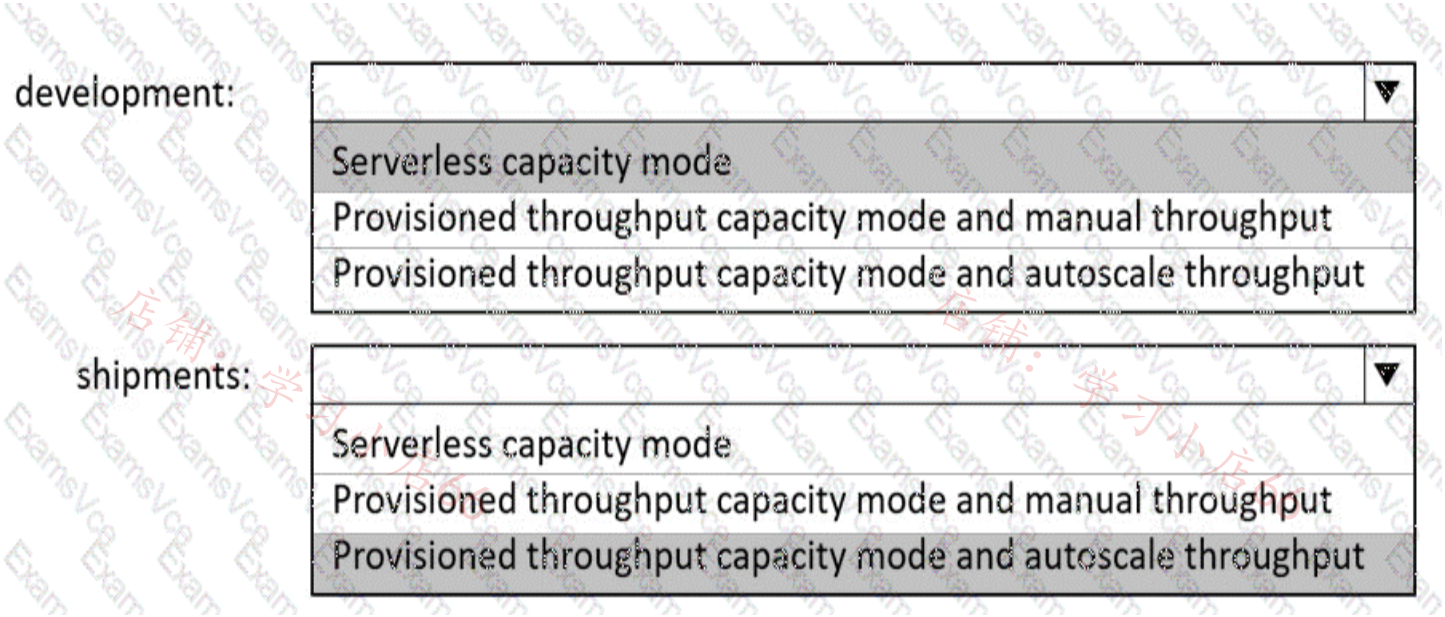
- Serverless capacity mode
- Provisioned throughput capacity mode and manual throughput
- Provisioned throughput capacity mode and autoscale throughput

shipments:

- Serverless capacity mode
- Provisioned throughput capacity mode and manual throughput
- Provisioned throughput capacity mode and autoscale throughput

Answer:

Explanation



Box 1: Serverless capacity mode

Azure Cosmos DB serverless best fits scenarios where you expect intermittent and unpredictable traffic with long idle times. Because provisioning capacity in such situations isn't required and may be cost-prohibitive, Azure Cosmos DB serverless should be considered in the following use-cases:

Getting started with Azure Cosmos DB

Running applications with bursty, intermittent traffic that is hard to forecast, or low (<10%) average-to-peak traffic ratio

Developing, testing, prototyping and running in production new applications where the traffic pattern is unknown

Integrating with serverless compute services like Azure Functions

Box 2: Provisioned throughput capacity mode and autoscale throughput

The use cases of autoscale include:

Variable or unpredictable workloads: When your workloads have variable or unpredictable spikes in usage, autoscale helps by automatically scaling up and down based on usage. Examples include retail websites that have different traffic patterns depending on seasonality; IOT workloads that have spikes at various times during the day; line of business applications that see peak usage a few times a month or year, and more. With autoscale, you no longer need to manually provision for peak or average capacity.

Reference:

https://docs.microsoft.com/en-us/azure/cosmos-db/serverless

https://docs.microsoft.com/en-us/azure/cosmos-db/provision-throughput-autoscale#use-cases-of-autoscale

Question #:77 - (Exam Topic 2)

You have a database in an Azure Cosmos DB for NoSQL account that is configured for multi-region writes.

You need to use the Azure Cosmos DB SDK to implement the conflict resolution policy for a container. The solution must ensure that any conflict sent to the conflict feed.

Solution: You set ConfilictResolutionMode to Custom. You Set ResolutionProcedures to a custom stored procedure. You configure the custom stored procedure to use the isTomstone parameter to resolve conflict.

Does this meet the goal?

A. Yes

B. No

Answer: A

Explanation

The solution is incorrect because there is no "isTom" parameter in the Azure Cosmos DB SDK. The correct parameter is "isTombstone".

Question #:78 - (Exam Topic 2)

You have a database in an Azure Cosmos DB for NoSQL account. The database contains a container named container1. The indexing mode container1 is set to none. You configure Azure Cognitive Search to extract data from container1 and make the data searchable. You discover that the Cognitive Search index is missing all the data from the Azure Cosmos DB index. What should you do to resolve the issue?

A. Modify The index attributes in Cognitive Search to searchable.

B. Modify the index attributes in Cognitive Search to Retrievable.

C. Change the indexing mode of container 1 to consistent-

D. Modify the indexing policy of container 1 to exclude the / * path

Answer: C

Question #:79 - (Exam Topic 2)

Note: This question is part of a series of questions that present the same scenario. Each question in the series contains a unique solution that might meet the stated goals. Some question sets might have more than one correct solution, while others might not have a correct solution.

After you answer a question in this section, you will NOT be able to return to it. As a result, these questions will not appear in the review screen.

You have a container named container1 in an Azure Cosmos DB Core (SQL) API account.

You need to make the contents of container1 available as reference data for an Azure Stream Analytics job.

Solution: You create an Azure Data Factory pipeline that uses Azure Cosmos DB Core (SQL) API as the input and Azure Blob Storage as the output.

Does this meet the goal?

A. Yes

B. No

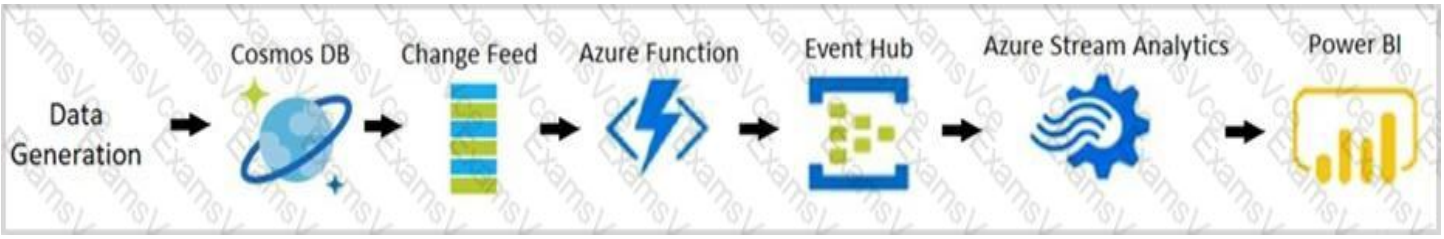
Answer: B

Explanation

Instead create an Azure function that uses Azure Cosmos DB Core (SQL) API change feed as a trigger and Azure event hub as the output.

The Azure Cosmos DB change feed is a mechanism to get a continuous and incremental feed of records from an Azure Cosmos container as those records are being created or modified. Change feed support works by listening to container for any changes. It then outputs the sorted list of documents that were changed in the order in which they were modified.

The following diagram represents the data flow and components involved in the solution:



Reference: https://docs.microsoft.com/en-us/azure/cosmos-db/sql/changefeed-ecommerce-solution

Question #:80 - (Exam Topic 2)

You plan to create an operational system that will store data in an Azure Cosmos OB 'or NoSQL account. You need to configure the account to meet the following requirements:

• Support Spar* queries.

• Support the analysis of data from the last six months.

• Only pay for analytical compute when running queries.

Which three actions should you perform? Each correct answer presents part of the solution. NOTE Each correct selection is worth one point.

A. Create an Azure Synapse linked service.

B. Create a container and set the time to live to six months.

C. Create a container and set the analytical property to six months.

D. Create an Azure Synapse pipeline.

E. Create an Azure Databanks notebook.

F. Enable Azure Synapse Link for the account

Answer: C E F

Question #:81 - (Exam Topic 2)

Note: This question is part of a series of questions that present the same scenario. Each question in the series contains a unique solution that might meet the stated goals. Some question sets might have more than one correct solution, while others might not have a correct solution.

After you answer a question in this section, you will NOT be able to return to it. As a result, these questions will not appear in the review screen.

You have an Azure Cosmos DB Core (SQL) API account named account 1 that uses autoscale throughput.

You need to run an Azure function when the normalized request units per second for a container in account1 exceeds a specific value.

Solution: You configure the function to have an Azure CosmosDB trigger.

Does this meet the goal?

A. Yes

B. No

Answer: B

Explanation

Instead configure an Azure Monitor alert to trigger the function.

You can set up alerts from the Azure Cosmos DB pane or the Azure Monitor service in the Azure portal.

Reference: https://docs.microsoft.com/en-us/azure/cosmos-db/create-alerts

Question #:82 - (Exam Topic 2)

You have an application that queries an Azure Cosmos 06 for NoSQL account.

You discover that the following two queries run frequently,

```
SELECT * FROM c WHERE c.name = @name ORDER BY c.name DESC, c.timestamp DESC

SELECT * FROM c WHERE c.name = @name AND c.timestamp > @timestamp ORDER BY c.name ASC, c.timestamp ASC
```

You need to minimize the request units (RUs) consumed by reads and writes. What should you create?

A. a composite index for (name DESC, time stamp ASC)

B. a composite index for (name ASC, time stamp DESC)

C. a composite index for (name ASC time stamp ASC) and a composite index for *(name*, time stamp disc)

D. a composite index for (name ASC, time stamp ASC)

Answer: D

Question #:83 - (Exam Topic 2)

You have an Azure Cosmos DB Core (SQL) account that has a single write region in West Europe.

database named db

```
az cosmosdb update -n $accountName -g $resourceGroupName \
   --locations regionName='West Europe' failoverPriority=0 isZoneRedundant-False \
   --locations regionName='North Europe' failoverPriority=1 isZoneRedundant-False

az cosmosdb failover-priority-change -n $accountName -g $resourceGroupName \
   --failover-policies 'North Europe=0' 'West Europe=1'
```

For each of the following statements, select Yes if the statement is true. Otherwise, select No.

NOTE: Each correct selection is worth one point.

Statements	Yes	No
After running the script, there will be an instance of Azure Cosmos DB in North Europe that is writable	○	○
After running the script, the Azure Cosmos DB instance in West Europe will be writable	○	○
The cost of the Azure Cosmos DB account is unaffected by running the script	○	○

Answer:

Explanation

Statements	Yes	No
After running the script, there will be an instance of Azure Cosmos DB in North Europe that is writable	○	○
After running the script, the Azure Cosmos DB instance in West Europe will be writable	○	○
The cost of the Azure Cosmos DB account is unaffected by running the script	○	○

Box 1: Yes

The Automatic failover option allows Azure Cosmos DB to failover to the region with the highest failover priority with no user action should a region become unavailable.

Box 2: No

West Europe is used for failover. Only North Europe is writable.

To Configure multi-region set UseMultipleWriteLocations to true.

Box 3: Yes

Provisioned throughput with single write region costs $0.008/hour per 100 RU/s and provisioned throughput with multiple writable regions costs $0.016/per hour per 100 RU/s.

Reference:

https://docs.microsoft.com/en-us/azure/cosmos-db/sql/how-to-multi-master

https://docs.microsoft.com/en-us/azure/cosmos-db/optimize-cost-regions

Question #:84 - (Exam Topic 2)

You have a database named telemetry in an Azure Cosmos DB Core (SQL) API account that stores IoT data. The database contains two containers named readings and devices.

Documents in readings have the following structure.

id

deviceid

timestamp

ownerid

measures (array)

- type

- value

- metricid

Documents in devices have the following structure.

id

deviceid

owner

- ownerid

- emailaddress

- name

brand

model

For each of the following statements, select Yes if the statement is true. Otherwise, select No.

NOTE: Each correct selection is worth one point.

Statements	Yes	No
To return for all devices owned by a specific `emailaddress`, multiple queries must be performed	○	○
To return `deviceid`, `ownerid`, `timestamp`, and `value` for a specific `metricid`, a join must be performed	○	○
To return `deviceid`, `ownerid`, `emailaddress`, and `model`, a join must be performed	○	○

Answer:

Explanation

Statements	Yes	No
To return for all devices owned by a specific `emailaddress`, multiple queries must be performed	○	○
To return `deviceid`, `ownerid`, `timestamp`, and `value` for a specific `metricid`, a join must be performed	○	○
To return `deviceid`, `ownerid`, `emailaddress`, and `model`, a join must be performed	○	○

Box 1: Yes

Need to join readings and devices.

Box 2: No

Only readings is required. All required fields are in readings.

Box 3: No

Only devices is required. All required fields are in devices.

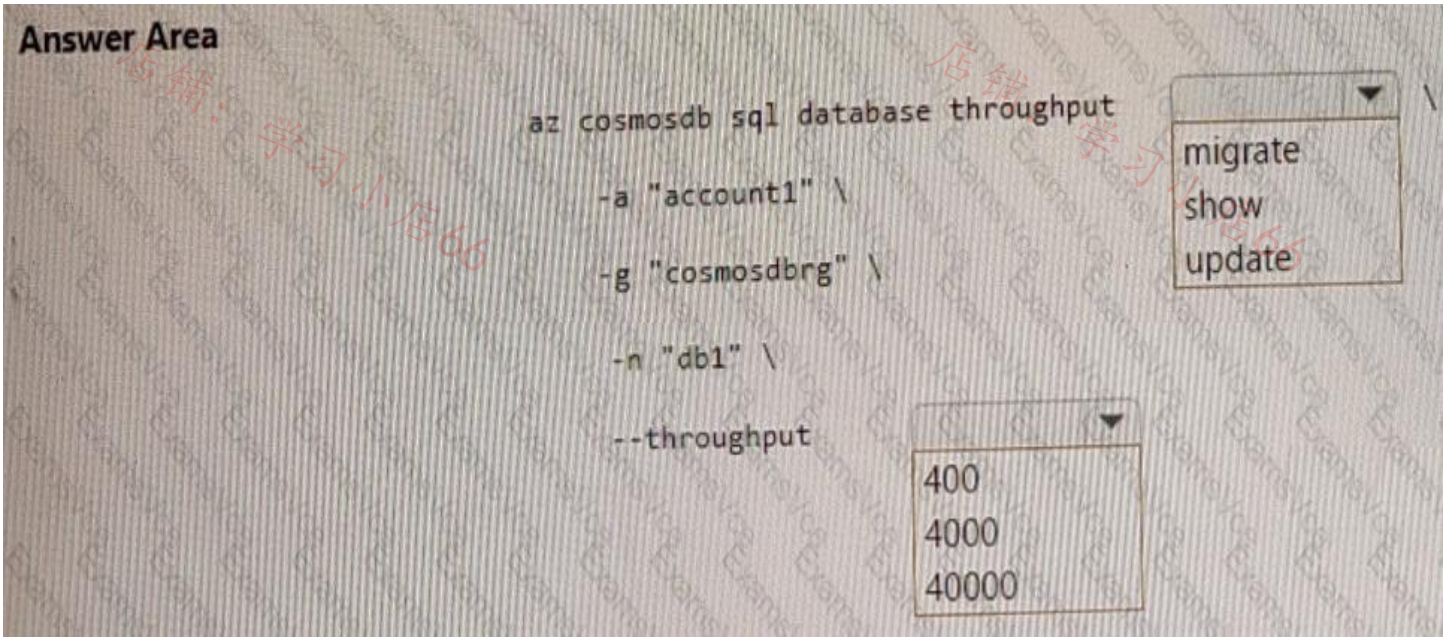
Question #:85 - (Exam Topic 2)

You have a database named db1 in an Azure Cosmos DB for NoSQL account named account1. The db1 database has a manual throughput of 4,000 request units per second (RU/s).

You need to move db1 from manual throughput to autoscale throughput by using the Azure CLI. The solution must provide a minimum of 4,000 RU/s and a maximum of 40,000 RU/s.

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

NOTE: Each correct selection is worth one point.



Answer:

Explanation

Migrate

40000

According to the Azure CLI reference**1**, you need to use the az cosmosdb sql database throughput migrate command to migrate the throughput of the SQL database between autoscale and manually provisioned. You also need to use the --throughput-type parameter to specify the type of throughput to migrate to, and the --max-throughput parameter to specify the maximum throughput resource can scale to (RU/s).

To complete the CLI statements, you should replace the missing values with:

- --throughput-type autoscale
- --max-throughput 40000

The final command should look like this:

az cosmosdb sql database throughput migrate \

--account-name account1 \

--name db1 \

--resource-group rg1 \

--throughput-type autoscale \

--max-throughput 40000

Question #:86 - (Exam Topic 2)

Note: This question is part of a series of questions that present the same scenario. Each question in the series contains a unique solution that might meet the stated goals. Some question sets might have more than one correct solution, while others might not have a correct solution.

After you answer a question in this section, you will NOT be able to return to it. As a result, these questions will not appear in the review screen.

You have an Azure Cosmos DB Core (SQL) API account named account 1 that uses autoscale throughput.

You need to run an Azure function when the normalized request units per second for a container in account1 exceeds a specific value.

Solution: You configure an Azure Monitor alert to trigger the function.

Does this meet the goal?

A. Yes

B. No

Answer: A

Explanation

You can set up alerts from the Azure Cosmos DB pane or the Azure Monitor service in the Azure portal.

Note: Alerts are used to set up recurring tests to monitor the availability and responsiveness of your Azure Cosmos DB resources. Alerts can send you a notification in the form of an email, or execute an Azure Function when one of your metrics reaches the threshold or if a specific event is logged in the activity log.

Reference: https://docs.microsoft.com/en-us/azure/cosmos-db/create-alerts

Question #:87 - (Exam Topic 2)

You are building an application that will store data in an Azure Cosmos DB for NoSQL account. The account uses the session default consistency level. The account is used by five other applications. The account has a single read-write region and 10 additional read regions.

Approximately 20 percent of the items stored in the account are updated hourly.

Several users will access the new application from multiple devices.

You need to ensure that the users see the same item values consistently when they browse from the different devices. The solution must not affect the other applications.

Which two actions should you perform? Each correct answer presents part of the solution.

NOTE: Each correct selection is worth one point.

A. Use implicit session management when performing read requests.

B. Provide a stored session token when performing read requests.

C. Associate a session token to the user account.

D. Set the default consistency level to eventual.

E. Associate a session token to the device.

Answer: B C

Question #:88 - (Exam Topic 2)

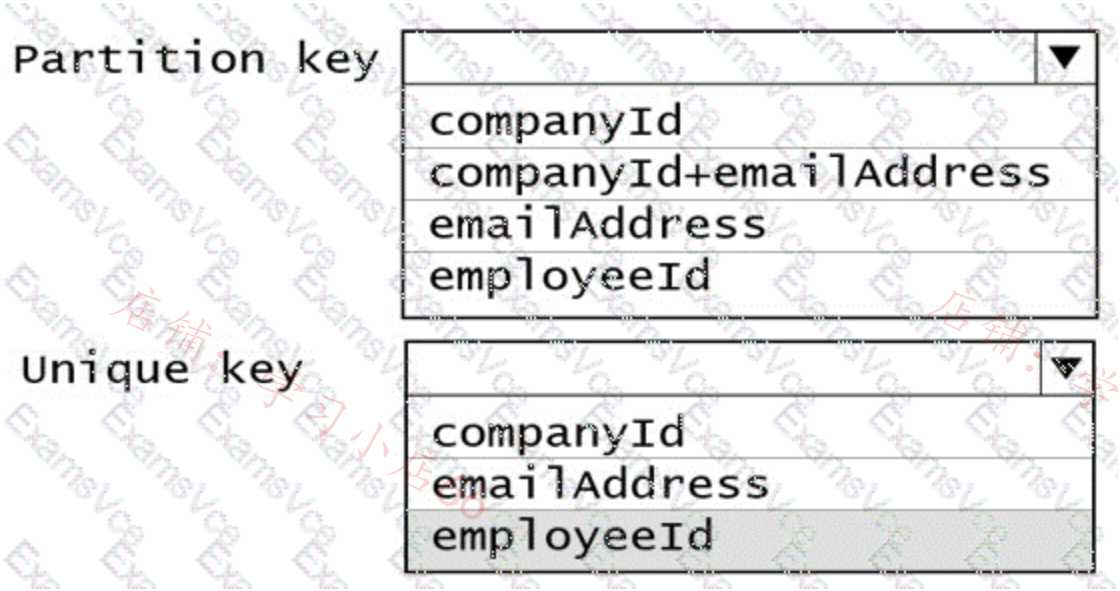
You have a database in an Azure Cosmos DB Core (SQL) API account.

You plan to create a container that will store employee data for 5,000 small businesses. Each business will have up to 25 employees. Each employee item will have an emailAddress value.

You need to ensure that the emailAddress value for each employee within the same company is unique.

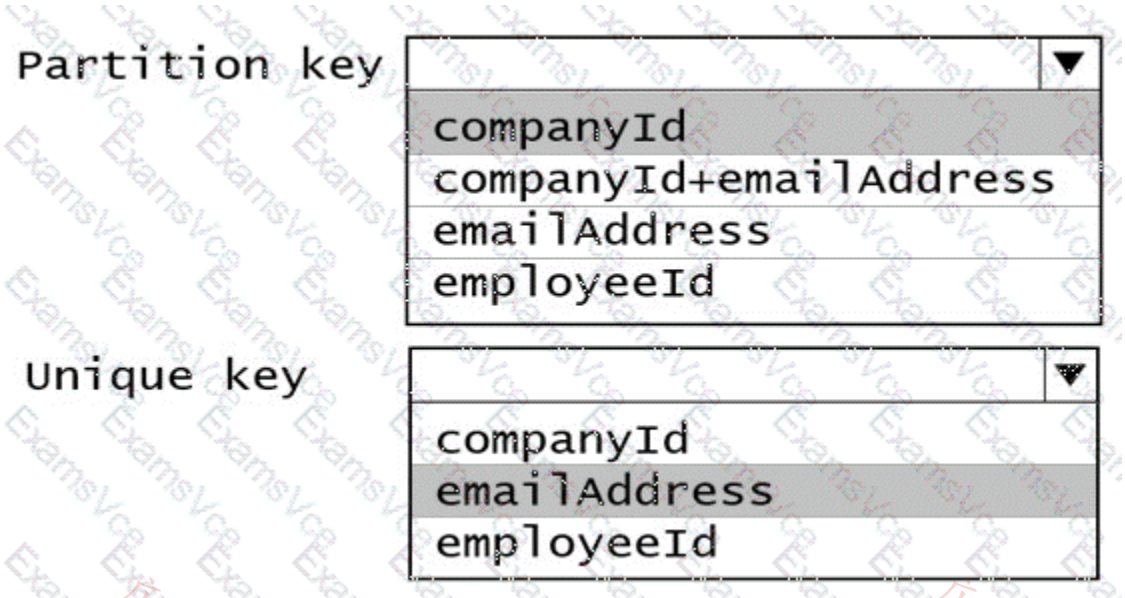
To what should you set the partition key and the unique key? To answer, select the appropriate options in the answer area.

NOTE: Each correct selection is worth one point.



Answer:

Explanation



Box 1: CompanyID

After you create a container with a unique key policy, the creation of a new or an update of an existing item resulting in a duplicate within a logical partition is prevented, as specified by the unique key constraint. The partition key combined with the unique key guarantees the uniqueness of an item within the scope of the container.

For example, consider an Azure Cosmos container with Email address as the unique key constraint and CompanyID as the partition key. When you configure the user's email address with a unique key, each item has a unique email address within a given CompanyID. Two items can't be created with duplicate email addresses and with the same partition key value.

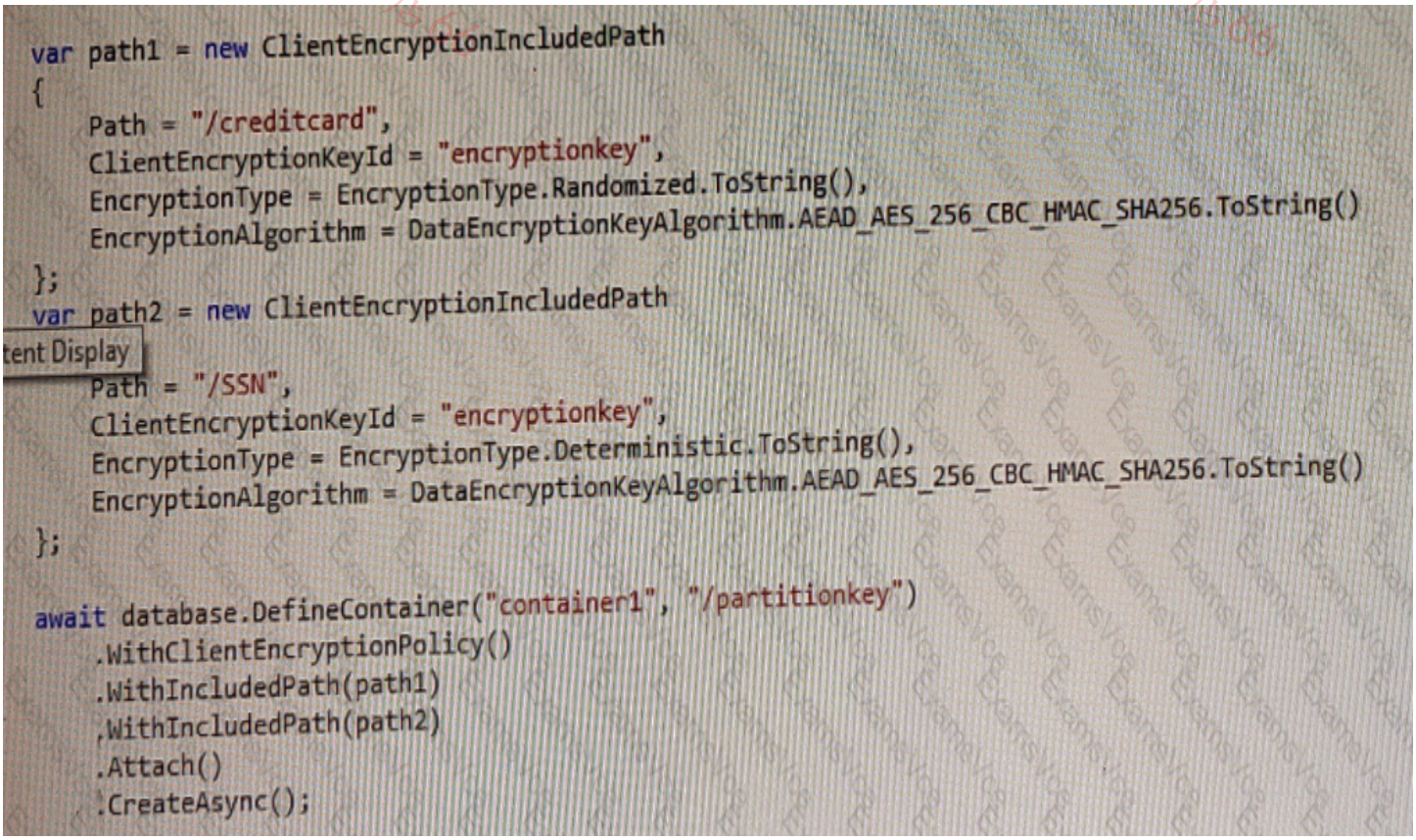
Box 2: emailAddress

Reference: https://docs.microsoft.com/en-us/azure/cosmos-db/unique-keys

Question #:89 - (Exam Topic 2)

You have a container named container1 in an Azure Cosmos DB for NoSQL account named account1.

You configure container1 to use Always Encrypted by using an encryption policy as shown in the C# and the Java exhibits. (Click the C# tab to view the encryption policy in C#. Click the Java tab to see the encryption policy in Java.)

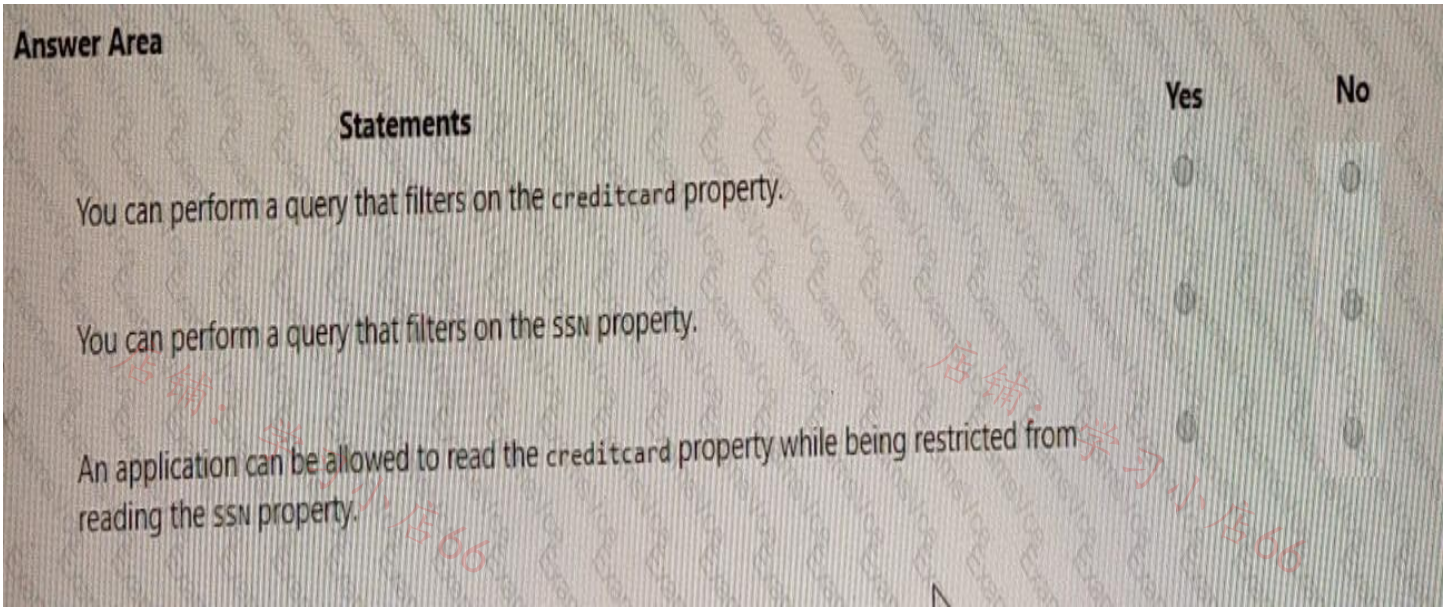


```
var path1 = new ClientEncryptionIncludedPath
{
    Path = "/creditcard",
    ClientEncryptionKeyId = "encryptionkey",
    EncryptionType = EncryptionType.Randomized.ToString(),
    EncryptionAlgorithm = DataEncryptionKeyAlgorithm.AEAD_AES_256_CBC_HMAC_SHA256.ToString()
};
var path2 = new ClientEncryptionIncludedPath
tent Display
    Path = "/SSN",
    ClientEncryptionKeyId = "encryptionkey",
    EncryptionType = EncryptionType.Deterministic.ToString(),
    EncryptionAlgorithm = DataEncryptionKeyAlgorithm.AEAD_AES_256_CBC_HMAC_SHA256.ToString()
};

await database.DefineContainer("container1", "/partitionkey")
    .WithClientEncryptionPolicy()
    .WithIncludedPath(path1)
    .WithIncludedPath(path2)
    .Attach()
    .CreateAsync();
```

```
ClientEncryptionIncludedPath path1 = new ClientEncryptionIncludedPath();
path1.path = "/creditcard";
path1.clientEncryptionKeyId = "encryptionkey";
path1.encryptionType = CosmosEncryptionType.RANDOMIZED;
path1.encryptionAlgorithm = CosmosEncryptionAlgorithm.AEAES_256_CBC_HMAC_SHA_256;

ClientEncryptionIncludedPath path2 = new ClientEncryptionIncludedPath();
path2.path = "/SSN";
path2.clientEncryptionKeyId = "encryptionkey";
path2.encryptionType = CosmosEncryptionType.DETERMINISTIC;
path2.encryptionAlgorithm = CosmosEncryptionAlgorithm.AEAES_256_CBC_HMAC_SHA_256;

tent Display entEncryptionIncludedPath> paths = new ArrayList<>();
paths.add(path1);
paths.add(path2);

CosmosContainerProperties containerProperties =
    new CosmosContainerProperties("container1", "/partitionkey");
containerProperties.setClientEncryptionPolicy(new ClientEncryptionPolicy(paths));
database.createEncryptionContainerAsync(containerProperties);
```

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:

Explanation

According to the Azure Cosmos DB documentation**1**, Always Encrypted is a feature designed to protect

sensitive data, such as credit card numbers or national identification numbers, stored in Azure Cosmos DB. Always Encrypted allows clients to encrypt sensitive data inside client applications and never reveal the encryption keys to the database.

To use Always Encrypted, you need to define an encryption policy for each container that specifies which properties should be encrypted and which data encryption keys (DEK) should be used. The DEKs are stored in Azure Cosmos DB and are wrapped by customer-managed keys (CMK) that are stored in Azure Key Vault.

Based on the encryption policy shown in the exhibits, the creditcard property is encrypted with a DEK named dek1, and the SSN property is encrypted with a DEK named dek2. Both DEKs are wrapped by a CMK named cmk1.

To answer your statements:

- You can perform a query that filters on the creditcard property = No. This is because the creditcard property is encrypted and cannot be used for filtering or sorting operations**1**.
- You can perform a query that filters on the SSN property = No. This is also because the SSN property is encrypted and cannot be used for filtering or sorting operations**1**.
- An application can be allowed to read the creditcard property while being restricted from reading the SSN property = Yes. This is possible by using different CMKs to wrap different DEKs and applying access policies on the CMKs in Azure Key Vault. For example, if you use cmk2 to wrap dek2 instead of cmk1, you can grant an application access to cmk1 but not cmk2, which means it can read the creditcard property but not the SSN property**2**.

Question #:90 - (Exam Topic 2)

Note: This question is part of a series of questions that present the same scenario. Each question in the series contains a unique solution that might meet the stated goals. Some question sets might have more than one correct solution, while others might not have a correct solution.

After you answer a question in this section, you will NOT be able to return to it. As a result, these questions will not appear in the review screen.

You have a container named container1 in an Azure Cosmos DB Core (SQL) API account.

You need to make the contents of container1 available as reference data for an Azure Stream Analytics job.

Solution: You create an Azure Synapse pipeline that uses Azure Cosmos DB Core (SQL) API as the input and Azure Blob Storage as the output.

Does this meet the goal?

A. Yes

B. No

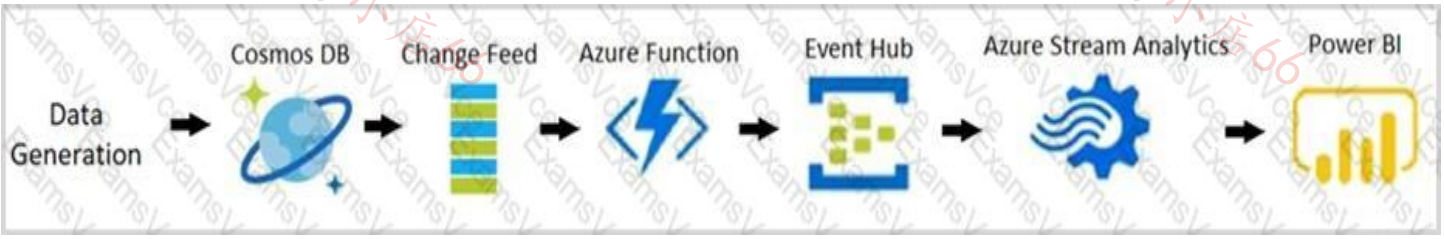
Answer: B

Explanation

Instead create an Azure function that uses Azure Cosmos DB Core (SQL) API change feed as a trigger and Azure event hub as the output.

The Azure Cosmos DB change feed is a mechanism to get a continuous and incremental feed of records from an Azure Cosmos container as those records are being created or modified. Change feed support works by listening to container for any changes. It then outputs the sorted list of documents that were changed in the order in which they were modified.

The following diagram represents the data flow and components involved in the solution:



Reference: https://docs.microsoft.com/en-us/azure/cosmos-db/sql/changefeed-ecommerce-solution

Question #:91 - (Exam Topic 2)

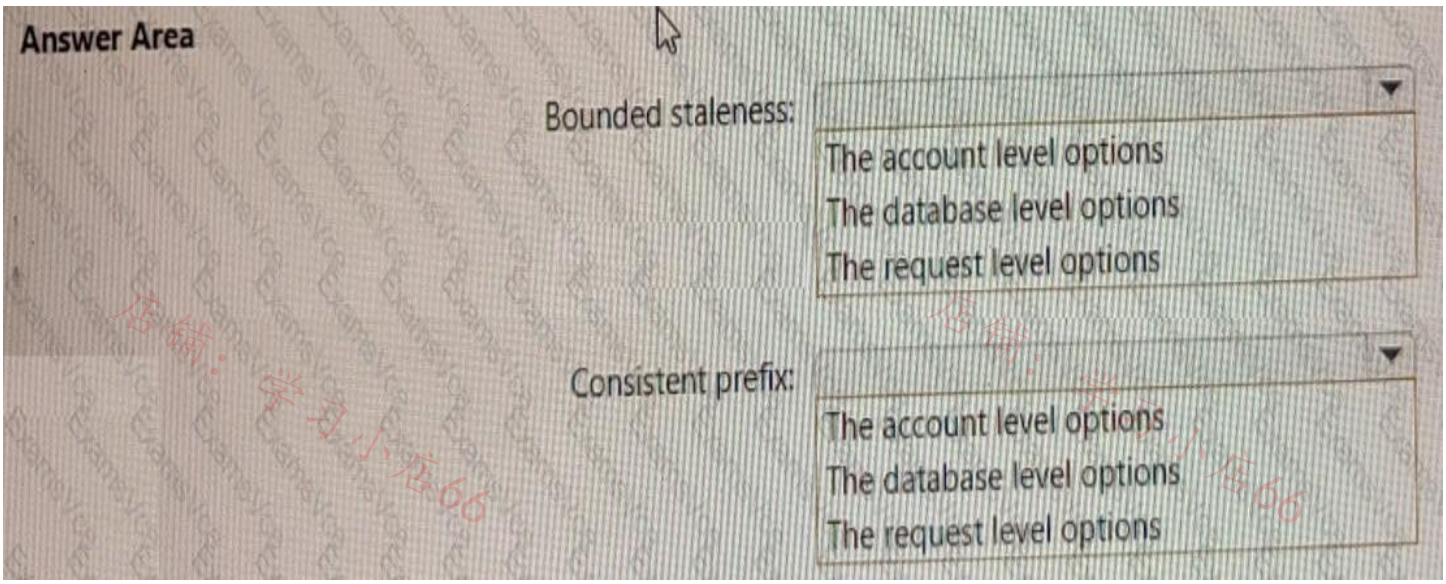
You have an Azure Cosmos DB account named account1 that has a default consistency level of session.

You have an app named App1.

You need to ensure that the read operations of App1 can request either bounded staleness or consistent prefix consistency.

What should you modify for each consistency level? To answer, select the appropriate options in the answer area.

NOTE: Each correct selection is worth one point.



Answer:

Explanation

Box 1 = The request level options

Azure Cosmos DB offers five well-defined consistency levels: strong, bounded staleness, session, consistent prefix and eventual. You can configure the default consistency level on your Azure Cosmos DB account at any time**2**. The default consistency level applies to all databases and containers under that account**1**. You can also override the default consistency level for a specific request by using the request options**2**.

Box 2 = The request level options

To modify the consistency level of a read operation in Azure Cosmos DB, you can use request-level options to override the account's default consistency setting. Therefore, to ensure that the read operations of App1 can request either consistent prefix or session consistency, you need to modify the request-level options for each operation. Reference: - https://docs.microsoft.com/en-us/azure/cosmos-db/consistency-levels

Question #:92 - (Exam Topic 2)

You are implementing an Azure Data Factory data flow that will use an Azure Cosmos DB (SQL API) sink to write a dataset. The data flow will use 2,000 Apache Spark partitions.

You need to ensure that the ingestion from each Spark partition is balanced to optimize throughput.

Which sink setting should you configure?

A. Throughput

B. Write throughput budget

C. Batch size

D. Collection action

Answer: C

Explanation

Batch size: An integer that represents how many objects are being written to Cosmos DB collection in each batch. Usually, starting with the default batch size is sufficient. To further tune this value, note:

Cosmos DB limits single request's size to 2MB. The formula is "Request Size = Single Document Size * Batch Size". If you hit error saying "Request size is too large", reduce the batch size value.

The larger the batch size, the better throughput the service can achieve, while make sure you allocate enough RUs to empower your workload.

Reference: https://docs.microsoft.com/en-us/azure/data-factory/connector-azure-cosmos-db

Question #:93 - (Exam Topic 2)

You are designing an Azure Cosmos DB Core (SQL) API solution to store data from IoT devices. Writes from the devices will be occur every second.

The following is a sample of the data.

```
{
    "id" : "03c1ca5a-db18-4231-908f-09a9bc7a7c3e",
    "deviceManufacturer" : "Contoso, Ltd",
    "deviceId" : "f460df85-799f-4d58-b051-67561b4993c6",
    "timestamp" : "2021-09-19T13:47:45",
    "sensor1Value" : true,
    "sensor2Value" : "75",
    "sensor3Value" : "4554",
    "sensor4Value" : "454",
    "sensor5Value" : "42128"
}
```

You need to select a partition key that meets the following requirements for writes:

Minimizes the partition skew

Avoids capacity limits

Avoids hot partitions

What should you do?

A. Use timestamp as the partition key.

B. Create a new synthetic key that contains deviceId and sensor1Value.

C. Create a new synthetic key that contains deviceId and deviceManufacturer.

D. Create a new synthetic key that contains deviceId and a random number.

Answer: D

Explanation

Use a partition key with a random suffix. Distribute the workload more evenly is to append a random number at the end of the partition key value. When you distribute items in this way, you can perform parallel write operations across partitions.

Reference: https://docs.microsoft.com/en-us/azure/cosmos-db/sql/synthetic-partition-keys

Question #:94 - (Exam Topic 2)

You need to implement a trigger in Azure Cosmos DB Core (SQL) API that will run before an item is inserted into a container.

Which two actions should you perform to ensure that the trigger runs? Each correct answer presents part of the solution.

NOTE: Each correct selection is worth one point.

A. Append pre to the name of the JavaScript function trigger.

B. For each create request, set the access condition in RequestOptions.

C. Register the trigger as a pre-trigger.

D. For each create request, set the consistency level to session in RequestOptions.

E. For each create request, set the trigger name in RequestOptions.

Answer: C

Explanation

C: When triggers are registered, you can specify the operations that it can run with.

F: When executing, pre-triggers are passed in the RequestOptions object by specifying PreTriggerInclude and then passing the name of the trigger in a List object.

Reference: https://docs.microsoft.com/en-us/azure/cosmos-db/sql/how-to-use-stored-procedures-triggers-udfs

Question #:95 - (Exam Topic 2)

You have an Azure Cosmos DB for NoSQL account named account1 that supports an application named App1. App1 uses the consistent prefix consistency level.

You configure account1 to use a dedicated gateway and integrated cache.

You need to ensure that App1 can use the integrated cache.

Which two actions should you perform for APP1? Each correct answer presents part of the solution.

NOTE: Each correct selection is worth one point.

A. Change the connection mode to direct

B. Change the account endpoint to https://account1.sqlx.cosmos.azure.com.

C. Change the consistency level of requests to strong.

D. Change the consistency level of requests to session.

E. Change the account endpoint to https://account1.documents.azure.com

Answer: B D

Explanation

the Azure Cosmos DB integrated cache is an in-memory cache that is built-in to the Azure Cosmos DB dedicated gateway. The dedicated gateway is a front-end compute that stores cached data and routes requests to the backend database. You can choose from a variety of dedicated gateway sizes based on the number of cores and memory needed for your workload**1**. The integrated cache can reduce the RU consumption and latency of read operations by serving them from the cache instead of the backend containers**2**.

For your scenario, to ensure that App1 can use the integrated cache, you should perform these two actions:

- Change the account endpoint to https://account1.sqlx.cosmos.azure.com. This is the dedicated gateway endpoint that you need to use to connect to your Azure Cosmos DB account and leverage the integrated cache. The standard gateway endpoint (https://account1.documents.azure.com) will not use the integrated cache**2**.
- Change the consistency level of requests to session. This is the highest consistency level that is supported by the integrated cache. If you use a higher consistency level (such as strong or bounded staleness), your requests will bypass the integrated cache and go directly to the backend containers